All Complete Functionalities are Reversible

Dakshita Khurana[1(✉)], Daniel Kraschewski[2], Hemanta K. Maji[3], Manoj Prabhakaran[4], and Amit Sahai[1]

[1] Department of Computer Science, Center for Encrypted Functionalities, UCLA, Los Angeles, USA
{dakshita,sahai}@cs.ucla.edu
[2] TNG Technology Consulting GmbH, Munich, Germany
daniel.kraschewski@tngtech.com
[3] Department of Computer Science, Purdue University, West Lafayette, USA
hmaji@purdue.edu
[4] Department of Computer Science, University of Illinois, Urbana-Champaign, USA
mmp@uiuc.edu

Abstract. Crépeau and Santha, in 1991, posed the question of reversibility of functionalities, that is, which functionalities when used in one direction, could securely implement the identical functionality in the reverse direction. Wolf and Wullschleger, in 2006, showed that oblivious transfer is reversible. We study the problem of reversibility among 2-party SFE functionalities, which also enable general multi-party computation, in the information-theoretic setting.

We show that any functionality that enables general multi-party computation, when used in both directions, is reversible. In fact, we show that any such functionality can securely realize oblivious transfer when used in an a priori fixed direction. This result enables secure computation using physical setups that parties can only use in a particular direction due to inherent asymmetries in them.

D. Khurana and A. Sahai—Research supported in part from a DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, NSF grants 1228984, 1136174, 1118096, and 1065276, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the ARL under Contract W911NF-15-C-0205. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

D. Khurana, H.K. Maji, M. Prabhakaran and A. Sahai—Work done in part while visiting the Simons Institute for Theoretical Computer Science, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant #CNS-1523467.

D. Kraschewski—Part of the research leading to these results was done while the author was at KIT and Technion. Supported by the European Union's Tenth Framework Programme (FP10/2010-2016) under grant agreement no. 259426 – ERC Cryptography and Complexity.

M. Prabhakaran—Research supported by NSF grant 1228856.

M. Fischlin and J.-S. Coron (Eds.): EUROCRYPT 2016, Part II, LNCS 9666, pp. 213–242, 2016.
DOI: 10.1007/978-3-662-49896-5_8

Keywords: Secure function evaluation · Information-theoretic security · UC-security · Reversibility of functionalities · Fixed-role reduction

1 Introduction

In 1991, Crépeau and Santha [7] posed the following question. Given oblivious transfers in one direction can we implement oblivious transfer in the opposite direction? That is, given oblivious transfers where Alice is the sender and Bob is the receiver, can we securely realize an oblivious transfer where Bob is the sender and Alice is the receiver? Wolf and Wullschleger [22] resolved this question in the affirmative. This result inspired several interesting results in cryptography, like offline generation of correlated private randomness independent of the target functionality being computed in secure computation [4,12] and (comparatively) easily introducing adaptive-security to secure computation protocols [18]. The proof of reversibility for oblivious transfer of [22] appears to be intimately tied to the specifics of the oblivious transfer functionality. Could reversibility, however, be a more general phenomenon?

Some functionalities, like simultaneous exchange, are inherently reversible. But we are most interested in functionalities which provide us general secure [3] multi-party computation [9,23], i.e. the *complete* functionalities. Restricted to the class of complete functionalities, the line of inquiry initiated in 1991 naturally leads to the following fundamental question.

Which Complete Functionalities can be Reversed?

We study this problem in the two-party setting for secure function evaluation (SFE) functionalities. Our work provides a full characterization of SFE functionalities that are reversible as well as sufficient for information-theoretic general secure multi-party computation. In fact, we show that *every* complete SFE functionality is reversible. In other words, we show that if using a functionality in both directions is powerful enough to enable general secure function evaluation, then in fact using the functionality in just one direction is enough.

Aside from its inherent theoretical appeal, the question of reversibility is also motivated by asymmetries that may be present in different systems. For example, if some physical phenomenon between two parties Alice and Bob is being utilized in order to carry out secure computations, it may be that only a powerful entity can play the role of Alice, but a weak device can play the role of Bob. In such an scenario, it would be critical to ensure that the cryptographic advantage offered by the physical phenomenon is sufficient for secure computation even if roles cannot be reversed.

We obtain our characterization of reversibility, in fact, by studying the more general problem of characterizing all 2-party complete functionalities that can be used in fixed roles to enable secure information-theoretic two-party computation, i.e. the characterization of fixed-role completeness.

1.1 Our Contributions

In this work, we study 2-party secure function evaluation (SFE) functionalities in the information-theoretic UC-setting [3]. Our first result shows that any complete 2-party SFE functionality is reversible.

Informal Theorem 1 (Reversibility Characterization). *Any complete 2-party SFE functionality $\mathcal{F}$ is reversible.*

Our construction is also constant rate. That is, n instances of the functionality in one direction is used to implement $\Theta(n)$ instances of the functionality in the reverse direction.

A functionality $\mathcal{F}$ is complete if it can be used (in both directions) to securely realize the oblivious transfer functionality. For the stronger security notion of fixed-role completeness, we show that any complete functionality, when used in fixed-role, is also complete.

Informal Theorem 2 (Fixed-Role Completeness Characterization). *Any complete 2-party SFE functionality $\mathcal{F}$ is also fixed-role complete.*

Similar to the previous result, this result is also constant rate. That is, using n instances of the $\mathcal{F}$ functionality in a fixed direction, we implement $\Theta(n)$ instances of the oblivious transfer functionality.

Additionally, we also show that the commitment functionality can be securely realized in the $\mathcal{F}$-hybrid if and only if $\mathcal{F}$ is complete (see Corollary 1). The proof is sketched in Sect. 1.4. This rules out the possibility of a functionality $\mathcal{F}$ which is of an *intermediate complexity* in the following sense: it enables the computation of the commitment functionality (a non-trivial functionality) but not the (all powerful) oblivious transfer functionality.

1.2 Prior Works

The problem of reversibility was initially posed by Crépeau and Santha [7] and the reversibility of oblivious transfer (and oblivious linear function evaluation) was exhibited by Wolf and Wullschleger [22].

There are several results characterizing completeness of functionalities in different settings. The oblivious transfer functionality was identified by Wiesner and Rabin [20,21]. Brassard et al. [2] showed the equivalence between various flavors of OT. In a seminal work, Kilian showed the active-completeness of OT [13]. Prior to this, the passive-completeness of OT was shown in [10,11]. Crépeau and Kilian showed that noisy channels are active-complete [5].

The first characterization of completeness appeared in the seminal work of Kilian [14]. In the asymmetric SFE setting, Beimel et al. [1] provided a characterization. Kilian, in another seminal work in 2000, vastly generalized these results [15]. Subsequent works extended Kilian's result for active-completeness in two different directions: [6] considered "channel functions;" [17] considered deterministic functions.

Recently, the full characterization of 2-party complete functionalities in the semi-honest [19] and malicious [16] settings were obtained.

1.3 Technical Overview: Reversibility of Functionalities

Let $\mathcal{F}$ be a randomized two-party functionality between parties A and B, and let $\mathcal{F}_{\mathsf{core}}$ denote the redundancy-free core of $\mathcal{F}$ (obtained after removing redundancies from $\mathcal{F}$, as described in Sect. 3.2 of our paper). Kraschewski et al. [16] showed that $\mathcal{F}$ is complete $\iff$ $\mathcal{F}_{\mathsf{core}}$ is not simple.

To develop intuition for 'simple' functions, consider the following example of a 'simple' two-party functionality $\mathcal{F}_{\mathsf{coin}}$. $\mathcal{F}_{\mathsf{coin}}$ ignores the inputs of both parties and just outputs a common uniform independent random bit to both parties. The formal notion of a simple function generalizes this to arbitrary randomized functions, by ensuring that if the parties start with independent inputs, then conditioned on the "common information" present after evaluating $\mathcal{F}_{\mathsf{core}}$, the views of the two players remain independent of each other. Naturally then, a *non-simple function* is one where the views of the two players are *not independent* conditioned on the "common information" present after evaluating $\mathcal{F}_{\mathsf{core}}$ on independent inputs. For the rest of this exposition, we will assume that $\mathcal{F}$ is redundancy-free, and thus $\mathcal{F} = \mathcal{F}_{\mathsf{core}}$.

Kraschewski et al. [16] also showed how to obtain UC commitments from either $A \to B$ or $B \to A$, but not necessarily in both directions, using any non-simple $\mathcal{F}$. W.l.o.g. for our case analysis and the examples below, we assume that $\mathcal{F}$ already gives commitments from $A \to B$.

The main technical challenge in our paper, is to obtain commitments from $B \to A$ using any complete (equivalently, non-simple) $\mathcal{F}$. This is done by partitioning all complete functionalities into three exhaustive cases: 1(a), 1(b) and 2. We will illustrate how we achieve this with the help of representative examples for each case (Figs. 1, 2 and 3). We define the notion of 'extreme views' and 'intersection' below, after which we describe our partition and explain the main ideas that allow us to obtain commitments in each case.

Extreme Views: Consider the example function matrices in Figs. 1, 2 and 3. For simplicity, these examples have no redundancies, and are therefore equivalent to their core. Alice views are rows, and each row is a tuple (x, w): where x is her input and w is the output she received. Bob views are columns and each column is a tuple (y, z), where y is his input and z is his output. $\perp$ denotes no input. Double-lines separate sets of columns that correspond to the same input of Bob. The entry in row (x, w) and column (y, z) denotes $\Pr_{\mathcal{F}}[(w, z) \mid (x, y)]$.

A view of Bob corresponds to a column in the matrix, labelled by the (input, output) for that view. An extreme view of Bob is a column that cannot be written as a convex linear combination of other columns in the matrix. Note that for any non-simple $\mathcal{F}$, both parties will have at least one extreme view.

Warmup: Extreme views guarantee binding. Looking ahead, extreme views will form an important part of our analysis. Consider the following illustrative situation: Suppose Alice and Bob invoke the functionality in Fig. 2 many times on uniformly random inputs (assume they picked their inputs honestly). After this, Bob is supposed to send Alice the indices of all executions where he received

$(1,0)$. Suppose malicious Bob instead decides to send to Alice some indices where his view was $(0,1)$ or $(0,0)$.

Note that corresponding to Bob's view $(1,0)$, Alice always obtains view $(\perp,1)$. On the other hand corresponding to Bob's view $(0,1)$, Alice obtains view $(\perp,0)$ with constant probability. Corresponding to Bob's view $(0,0)$, Alice always obtains view $(\perp,0)$. Since Bob cannot guess what view Alice obtained, if Bob tries to cheat by claiming that his view was $(1,0)$ when actually his view was $(0,1)$ or $(0,0)$, Alice will sometimes end up with a view of $(\perp,0)$ and thus immediately detect Bob's cheating with constant probability. This weakly binds Bob to his views. We use repetition techniques (error-correcting codes) to amplify this weak binding property.

More generally, since extreme views cannot be expressed as a convex linear combination of other views, it impossible for any party to obtain other views and claim that he obtained a specific extreme view without getting caught. In the example situation above, no convex linear combination of other views $(0,1)$ and $(0,0)$ can be claimed to be the extreme view $(1,0)$. The same thing is true for *all extreme views* in any functionality $\mathcal{F}$.

Intersecting Views: A view of Alice, V_A, intersects with a view of Bob, V_B, if the joint view (V_A, V_B) occurs with non-zero probability on invoking $\mathcal{F}$ with uniform distribution over both inputs.

Case Analysis. Given this terminology, we partition the set of all complete functionalities into three sets, corresponding to Cases 1(a), 1(b) and 2. [16] already show how to obtain commitments from any functionality in what we call Case 1(a). The major technical contribution of our paper is to obtain commitments from functionalities that lie in Cases 1(b) and 2.

We will now walk through these cases using example functionalities from Figs. 1, 2 and 3. We will first define Case 1(a), and then describe how we partition the remaining possibilities for complete functionalities into Cases 1(b) and 2. At this level, the fact that they are exhaustive will be trivial to see. For Cases 1(b) and 2, we will then explain the main ideas behind obtaining commitments from $B \rightarrow A$, with Y with the help of examples.

Alice \ Bob	$(\perp,0)$	$(\perp,1)$
$(\perp,0)$	$1/2$	$1/6$
$(\perp,1)$	0	$1/3$

Fig. 1. Case 1(a). Both columns are extreme.

Alice \ Bob	$(0,0)$	$(0,1)$	$(1,0)$
$(\perp,0)$	$1/4$	$1/12$	0
$(\perp,1)$	0	$2/3$	1

Fig. 2. Case 1(b). (0,0) and (1,0) are extreme. $\mathsf{col}(0,1) \equiv 1/3 \times \mathsf{col}(0,0) + 2/3 \times \mathsf{col}(1,0)$

Alice \ Bob	$(0,0)$	$(0,1)$	$(0,2)$	$(1,0)$	$(1,1)$
$(\perp,0)$	$1/5$	0	0	$1/20$	0
$(\perp,1)$	0	$3/5$	0	$9/20$	$9/20$
$(\perp,2)$	0	0	$1/5$	0	$1/20$

Fig. 3. Case 2. (0,0), (0,1) and (0,2) are extreme. $\mathsf{col}(1,0) \equiv 1/4 \times \mathsf{col}(0,0) + 3/4 \times \mathsf{col}(1,0)$. $\mathsf{col}(1,1) \equiv 1/4 \times \mathsf{col}(0,2) + 3/4 \times \mathsf{col}(1,0)$.

– **Case 1(a):** Kraschewski et al. [16] obtained commitments from $P_1 \to P_2$ using any functionality between parties P_1 and P_2 which has the following property: There exist at least 2 extreme views $(\mathcal{V}^1_{P_1}, \mathcal{V}^2_{P_1})$ of P_1 which intersect with the same view V_{P_2} of P_2, i.e. both joint views $(\mathcal{V}^1_{P_1}, V_{P_2})$ and $(\mathcal{V}^2_{P_1}, V_{P_2})$ occur with non-zero probability. They also show that any complete functionality must satisfy this property in at least one direction, either $P_1 \to P_2$ or $P_2 \to P_1$.

Recall that we require commitments from $B \to A$. We define Case 1(a) as the set of all $\mathcal{F}$ which satisfy the above property in the $B \to A$ direction. That is, Case 1(a) consists of all $\mathcal{F}$ for which there exist at least 2 extreme views $(\mathcal{V}^1_B, \mathcal{V}^2_B)$ of Bob that intersect with the same view V_A of Alice, i.e. both joint views $(\mathcal{V}^1_B, V_A)$ and $(\mathcal{V}^2_B, V_A)$ occur with non-zero probability.

Observe that in the example in Fig. 1, both Bob views $(\perp, 0)$ and $(\perp, 1)$ are extreme, and they intersect with common Alice view $(\perp, 0)$. Figure 1 satisfies the above property from $B \to A$ and lies in Case 1(a). Thus, [16] give $B \to A$ commitments for this case.

At a very intuitive level, Bob is committed to the views he obtained. He reveals these views in the decommitment phase. The common intersecting view of Alice occurs sometimes, and in these instances, she does not know what view Bob obtained. This property is amplified to obtain hiding. As illustrated above, Bob cannot equivocate extreme views, and [16] used this property of the extreme views to obtain binding as illustrated above.

Remaining Cases are Exhaustive. Let $\mathcal{V}_B$ denote the set of all extreme views of Bob. Let $\widehat{Y}_B := \{y : \exists z, \text{ such that } (y,z) \in \mathcal{V}_B\}$, that is $\widehat{Y}_B$ denotes the set of Bob inputs, which have at least one corresponding view in $\mathcal{V}_B$. Let $\widehat{V}_B$ denote the set of all views of Bob that have some $y \in \widehat{Y}_B$ as input, i.e., $\widehat{V}_B = \{(y,z) : y \in \widehat{Y}_B, (y,z) \text{ occurs with non-zero probability}\}$. Note: $\widehat{V}_B$ contains all extreme Bob views, and may also contain some non-extreme Bob views.

- Case 1, i.e. Case 1(a) ∪ Case 1(b), consists of all complete functionalities for which two views in $\widehat{V}_B$ intersect with a common Alice view.
- Case 2 consists of all complete functionalities for which no two views in $\widehat{V}_B$ intersect with a common Alice view.

It is easy to see that Cases 1 and 2 are an exhaustive partition of all complete $\mathcal{F}$. Next,

- Case 1(a) consists of all functionalities $\mathcal{F}$ in Case 1, where there are at least two extreme views in $\widehat{V}_B$ that intersect with a common Alice view.

- Case 1(b) consists of all functionalities in Case 1 that are not in Case 1(a). In particular, the fact that $\mathcal{F}$ is in Case 1(b) requires that no two extreme views in $\widehat{V}_B$ intersect with a common Alice view. This means that either an extreme and non-extreme view of Bob in $\widehat{V}_B$ intersect with a common Alice view, or two non-extreme views of Bob in $\widehat{V}_B$ intersect with a common Alice view. Note that if two non-extreme views intersect, then an extreme and non-extreme view also intersect (by the definition of extreme views).

– **Case 1(b):** Recall that this case consists of complete functionalities for which an extreme and a non-extreme view of Bob in $\widehat{V}_B$ intersect with a common Alice view, for $\widehat{V}_B$ defined above. An illustrative example for this case is in Fig. 2 above. The views $(0,0)$ and $(1,0)$ of Bob are extreme, $\widehat{Y}_B = \{0,1\}$, $\widehat{V}_B = \{(0,0),(0,1),(1,0)\}$. Moreover, views $(0,0)$ and $(0,1)$ in $\widehat{V}_B$ intersect with a common Alice view. Also, views $(1,0)$ and $(0,1)$ in $\widehat{V}_B$ intersect with a common Alice view. But no two extreme Bob views intersect with a common Alice view.

To obtain $B \to A$ commitments, Alice and Bob invoke $\mathcal{F}$, with Alice using a uniform distribution over her inputs and Bob using a uniform distribution over inputs in $\widehat{Y}_B$. Assume for simplicity that Alice and Bob can be forced to use the correct distribution over their inputs. (This can be ensured using cut-and-choose techniques and extreme views of Bob.)

Binding. We split Bob's views into two categories: extreme and non-extreme. The main idea behind building commitments will be to ensure that he cannot obtain views in one category and later claim that they belong in another category. To understand this, consider the following example scenario w.r.t. the functionality in Fig. 2: Bob obtains view $(0,0)$, which is an extreme view, and claims later that he obtained $(0,1)$, which is a non-extreme view. We would like to prevent this situation. We would also like to prevent Bob from obtaining view $(0,1)$, which is a non-extreme view, and later claiming that he obtained $(0,0)$, which is an extreme view. In both these situations, we would like Alice to catch such a cheating Bob with high probability. Ensuring that she catches such a cheating Bob will (weakly) bind Bob to the category of views he obtained. Here is how we ensure this.

- Suppose Bob obtains $(0,1)$ and later claims it was $(0,0)$. By a similar argument as the warmup, Alice will catch him with constant probability: Note that Alice obtains view $(\perp,1)$ with constant probability corresponding to Bob's view $(0,1)$, but she never obtains view $(\perp,1)$ corresponding to Bob's view $(0,0)$. Since Bob doesn't know what view Alice obtained, if he actually obtained the view $(0,1)$ and tried to claim that he obtained $(0,0)$, Alice will sometimes end up with view $(\perp,1)$ and detect Bob's cheating with constant probability. This can be amplified to full-fledged binding using error correction.
- Suppose Bob obtains $(0,0)$ and claims that it was $(0,1)$. In this case, the previous argument no longer works since $(0,1)$ is not an extreme view. However, because both parties used uniform inputs, Bob will obtain some 'correct' distribution over his outputs. Also by the previous item, Bob

cannot have obtained $(0,1)$ and claim that it is $(0,0)$. Thus, if he obtains $(0,0)$ and claims that he obtained $(0,1)$, then $(0,1)$ will appear too often in his claimed views and Alice will detect this. In general, to equivocate extreme views to non-extreme views, Bob will have to "steal" probability mass from the extreme views and add more mass to the non-extreme views – which Alice will detect.

Hiding. For a uniform distribution over her inputs, with constant probability Alice obtains a common view that intersects both an extreme and a non-extreme view of Bob. Thus she cannot tell which category Bob's view was in, at the end of the commit stage. This gives a weak form of hiding which can then be amplified. For example in the functionality in Fig. 2, Alice's view $(\perp,0)$ intersects with the extreme view $(0,0)$ and non-extreme view $(0,1)$ of Bob. Only one such intersection suffices to obtain hiding. For a complete analysis of this case, please refer to Sect. 5.

– **Case 2:** Recall that this case consists of complete functionalities for which no two views of Bob in $\widehat{V}_B$ intersect with a common Alice view, for $\widehat{V}_B$ defined above. Nevertheless, note that at least 2 views of Bob must intersect with a common Alice view, because otherwise $\mathcal{F}$ is trivial. Moreover, if two views outside $\widehat{V}_B$ intersect with a common Alice view, then both views must be non-extreme (by the definition of $\widehat{V}_B$). This means that at least one extreme and non-extreme view pair intersect with a common Alice view, which means that in this case necessarily, one Bob view inside $\widehat{V}_B$ and one outside $\widehat{V}_B$ intersect with a common Alice view.

In the illustrative example in Fig. 3, since the first three columns can be convex-linearly combined to obtain the fourth and fifth columns, only the first three views $(0,0),(0,1),(0,2)$ of Bob are extreme. Moreover, all extreme views of Bob correspond to input 0, thus $\widehat{Y}_B = \{0\}, \widehat{V}_B = \{(0,0),(0,1),(0,2)\}$ and views in $\widehat{V}_B$ do not intersect with any common Alice view. Note also that Bob's input 1 is not redundant, because the distribution over Alice's views induced by Bob's input 1 is different from the distribution induced by Bob's input 0.

To obtain $B \rightarrow A$ commitments in this case, Alice and Bob invoke $\mathcal{F}$ with Alice using a uniform distribution over her inputs and Bob using a uniform distribution over all his inputs.

Binding. We partition Bob's views into two categories: views inside $\widehat{V}_B$ and views outside $\widehat{V}_B$, then argue that he cannot equivocate between these categories. Again, here we only argue that a cheating Bob will be caught with constant probability – this can be amplified using error-correcting codes to obtain full-fledged binding.

In this case, it is not straightforward to argue that Bob can be forced to use a uniform (or some requisite) distribution over his inputs – in fact arguing this forms the crux of our binding argument. Consider the example in Fig. 3. Here are two representative strategies of a malicious Bob:

- Bob actually obtains view $(1, 0)$, and later claims that it was $(0, 1)$. However, unbeknownst to Bob, Alice may obtain view $(\perp, 0)$ and therefore detects Bob's cheating with constant probability. More generally, if Bob uses input 1 and claims that it is a 0, Alice will catch him with constant probability.
- Bob actually uses input 0 all the time, and later claims that in some invocations he used input 1. Here, we note that the distributions over Alice's views corresponding to Bob's inputs 0 and 1 in the example functionality are different. If this were not the case, then Bob's input 1 would be redundant. This means that Alice, by simply checking her output distribution, will catch Bob whenever he launches such an attack.

  We generalize this argument (refer to Lemma 3) to show that in any redundancy-free core of a complete functionality, in Case 2, there exists at least one Bob input outside of $\widehat{Y}_B$ (this input is 1 in the representative example) which cannot be mimicked using any input in $\widehat{Y}_B$ (this input is 0 in this example).

  Hiding. We show that there exists a common Alice view which intersects at least one Bob view in $\widehat{Y}_B$ (which is 0 in the representative example in Fig. 3) and one Bob view corresponding to the un-mimickable input outside $\widehat{Y}_B$ (which is a 1 in the example). In the example functionality, Alice's view $(\perp, 0)$ intersects with the views $(0, 0)$ in $\widehat{V}_B$ and $(1, 0)$ corresponding to input 1 outside $\widehat{Y}_B$. When using a uniform distribution over her inputs (this can be easily ensured), with constant probability Alice obtains this intersecting view. This gives a weak form of hiding which can then be amplified. A complete analysis of this case is in Sect. 6.

1.4 Technical Overview: Commitment Reducible Only to Complete SFE Functionalities

We have already shown what if f is a 2-party SFE which is malicious-complete then $\mathcal{F}_{\mathsf{com}}$ fixed-role reduces to it. So, it suffices to show that if $\mathcal{F}$ has a simple core, then $\mathcal{F}_{\mathsf{com}}$ does not reduce to $\mathcal{F}$. Suppose a protocol Π securely realizes $\mathcal{F}_{\mathsf{com}}$ in the $\mathcal{F}$-hybrid, where $\mathcal{F}$ has a simple core. Note that, given a public transcript, since $\mathcal{F}$ has a simple core, a party can always sample joint-views consistent with it. Therefore, either each transcript can be equivocated or it is not hiding. Hence, we have the following result:

Corollary 1. *For every 2-party SFE $\mathcal{F}$, we have: $\mathcal{F}_{\mathsf{com}} \sqsubseteq_{\text{UC}} \mathcal{F}$ iff $\mathcal{F}_{\mathsf{OT}} \sqsubseteq_{\text{UC}} \mathcal{F}$.*

2 Preliminaries

In this section, we recall some primitives useful in stating unified completeness results for 2-party SFE in various security notions.

2.1 Secure Function Evaluation

A Functionality. Consider a two-party finite randomized functionality $\mathcal{F}$ between Alice and Bob, where Alice has input $x \in \mathcal{X}$ and Bob has input $y \in \mathcal{Y}$. They invoke the functionality with their respective inputs and obtain outputs $w \in \mathcal{W}$ and $z \in \mathcal{Z}$. We recall that such a functionality can be denoted by a matrix. The rows of this matrix are indexed by Alice views $(x, w) \in \mathcal{X} \times \mathcal{W}$ and columns are indexed by Bob views $(y, z) \in \mathcal{Y} \times \mathcal{Z}$. The entry in the cell in row (x, w) and column (y, z) equals $\Pr[w, z|x, y]$.

This matrix can also be viewed as a collection of stochastic sub-matrices, where each sub-matrix corresponds to some input $x \in \mathcal{X}$ of Alice and $y \in \mathcal{Y}$ of Bob. Each cell in this sub-matrix, with row indexed by Alice output w and column indexed by Bob output z equals $\Pr[w, z|x, y]$.

Graph of an SFE Functionality. Given a 2-party SFE $\mathcal{F}(f_A, f_B)$ we define a bipartite graph $G(\mathcal{F})$ as follows.

Definition 1. *Graph of a 2-party SFE. Given a SFE functionality $\mathcal{F}(f_A, f_B)$, its corresponding graph $G(\mathcal{F})$ is a weighted bipartite graph constructed as follows. Its partite sets are $X \times Z_A$ and $Y \times Z_B$. For every $(x, a) \in X \times Z_A$ and $(y, b) \in Y \times Z_B$, the edge joining these two vertices is assigned weight*

$$\mathsf{wt}\left((x, a), (y, b)\right) := \frac{\Pr_{r \xleftarrow{\$} R}[f_A(x, y, r) = a \ \wedge \ f_B(x, y, r) = b]}{|X \times Y|}$$

The choice of the normalizing constant $1/|X \times Y|$ is arbitrary. For this particular choice of constant, we can view the weight of an edge as representing the joint-distribution probability of input-output pairs seen by the two parties when $(x, y, r) \xleftarrow{\$} X \times Y \times R$.

The *kernel* of a 2-party function f is a function which outputs to the two parties only the "common information" that f makes available to them. To formalize this, we define a weighted bipartite graph $G(f)$ with partite sets $X \times W$ and $Y \times Z$, and for every $(x, w) \in X \times W$ and $(y, z) \in Y \times Z$, the edge joining these two vertices is assigned weight $\frac{p_f[w,z|x,y]}{|X \times Y|}$. The kernel of $\mathcal{F}$ is a randomized function which takes inputs $x \in X$ and $y \in Y$ from the parties, samples $(w, z) \xleftarrow{\$} f(x, y)$, and outputs to both parties the connected component of $G(\mathcal{F})$ which contains the edge $(x, w), (y, z)$.

2-Party Secure Function Evaluation. A two-party randomized function (also called a secure function evaluation (SFE) functionality) is specified by a single randomized function denoted as $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{W} \times \mathcal{Z}$. Despite the notation, the range of f is, more accurately, the space of probability distributions over $\mathcal{W} \times \mathcal{Z}$. The functionality takes an input $x \in \mathcal{X}$ from Alice and an input $y \in \mathcal{Y}$ from Bob, and samples $(w, z) \in \mathcal{W} \times \mathcal{Z}$ according to the distribution $f(x, y)$; then it delivers w to Alice and z to Bob. Throughout, we shall denote the probability of outputs being (w, z) when Alice and Bob use inputs x and y respectively is

represented by $\beta^{\mathcal{F}}[w, z|x, y]$. We use the following variables for the sizes of the sets $\mathcal{W}, \mathcal{X}, \mathcal{Y}, \mathcal{Z}$: $|\mathcal{X}| = m, |\mathcal{Y}| = n, |\mathcal{W}| = q, |\mathcal{Z}| = r$.

As is conventional in this field, in this paper, we shall restrict to function evaluations where m, n, q and r are constants, that is, as the security parameter increases the domains do not expand. (But the efficiency and security of our reductions are only polynomially dependent on m, n, q, r, so one could let them grow polynomially with the security parameter. We have made no attempt to optimize this dependency.) W.l.o.g., we shall assume that $\mathcal{X} = [m]$ (that is, the set of first m positive integers), $\mathcal{Y} = [n], \mathcal{W} = [q]$ and $\mathcal{Z} = [r]$.

We consider standard security notions in the information-theoretic setting: UC-security, standalone security and passive-security against computationally unbounded adversaries (and with computationally unbounded simulators). Using UC-security allows to compose our sub-protocols securely [3]. Error in security (simulation error) is always required to be negligible in the security parameter of the protocol, and the communication complexity of all protocols are required to be polynomial in the same parameter. However, we note that a protocol may invoke a sub-protocol with a security parameter other than its own (in particular, with a constant independent of its own security parameter).

Complete Functionalities. A two-party randomized function evaluation $\mathcal{F}$ is standalone-complete (respectively, UC-complete) against information theoretic adversaries if any functionality $\mathcal{G}$ can be standalone securely (respectively, UC securely) computed in the $\mathcal{F}$ hybrid. We shall also consider passive-complete functions where we consider security against passive (semi-honest) adversaries.

Redundancy-free core of a functionality. The core of a functionality is computed by removing redundant parts of the functionality f. A redundancy may be of two forms. It could consist of inputs which are useless for the adversary, that is, using another input gives the adversary strictly more *information* about the view of the (other) honest party, while the honest party cannot distinguish the cases in which the adversary used the less informative or the more informative input. In this case, the less informative input is called redundant and is removed to obtain the core of the functionality.

Another kind of redundancy is an output redundancy, where two or more outputs can be compressed into a single output if they convey identical information to the adversary about the honest party's view. As an example, consider a functionality in which when Bob's input is 0, if Alice's input is 0 then he receives 0, but if her input is 1, he receives the output symbol α with probability 3/4 and β with probability 1/4. Here, the two outcomes α and β give Bob the same information about Alice's input, and could be merged into a single output. We recall the formal linear algebraic definition of redundancies from Kraschewski et al. [16] in Sect. 3.2.

Simple core of functionalities. The core of a functionality f is *simple* if for parties starting with independent inputs, the views of the parties remain independent of each other conditioned on the common information after the function evaluation.

Recall that Kraschewski et al. [16] showed that a finite randomized functionality is complete if and only if the redundancy-free core of $\mathcal{F}$ is not simple.

Extreme views and mimicking inputs. Consider the matrix $\beta^{\mathcal{F}}$ obtained after removing the above-mentioned redundancies from the matrix $\mathcal{F}$. The entry in the cell in row (x, w) and column (y, z) is denoted by $\beta^{\mathcal{F}}_{x,w,y,z}$ and equals $\Pr[w, z|x, y]$.

Then a view (y, z) of Bob is an *extreme* view if the column indexed by (y, z) in $\beta^{\mathcal{F}}$ cannot be written as a convex linear combination of other columns in $\beta^{\mathcal{F}}$. Note that there necessarily exist at least two extreme views for each party in any non-trivial functionality. We say that a view (y, z) of Bob intersects with a view (x, w) of Alice if the entry $\beta^{\mathcal{F}}_{x,w,y,z} \neq 0$.

Let $Y_0 \subset \mathcal{Y}$ be a set of Bob inputs. We say that an input $y^* \in Y \backslash Y_0$ of Bob, is *mimicked* by Y_0, if there exists a probability distribution η over Y_0 such that Alice's view when Bob is choosing inputs from this distribution is indistinguishable from her view when Bob uses y^*.

2.2 Leftover Hash Lemma

The *min-entropy* of a discrete random variable X is defined to be $H_\infty(X) = -\log \max_{x \in \mathsf{Supp}(X)} \mathfrak{p}^f[X = x]$. For a joint distribution (A, B), the *average min-entropy* of A w.r.t. B is defined as $\widetilde{H}_\infty(A|B) = -\log\left(\mathbb{E}_{b \sim B}\left[2^{-H_\infty(A|B=b)}\right]\right)$.

Imported Lemma 1 (Generalized Leftover Hash Lemma(LHL) [8]). *Let* $\{H_x : \{0,1\}^n \to \{0,1\}^\ell\}\}_{x \in X}$ *be a family of universal hash functions. Then, for any joint distribution* (W, I):$\mathsf{SD}\left((H_X(W), X, I), (\mathcal{U}_\ell, X, I)\right) \leq \frac{1}{2}\sqrt{2^{-\widetilde{H}_\infty(W|I)} 2^\ell}$.

3 Technical Tools

This section is mainly based on concepts introduced in [16].

3.1 Notation and Definitions

Consider the matrix $\beta^{\mathcal{F}}$ of the redundancy-free core of $\mathcal{F}$, whose columns are indexed by Bob views $(y, z) \in \mathcal{Y} \times \mathcal{Z}$ and rows are indexed by Alice views $(x, w) \in \mathcal{X} \times \mathcal{W}$. The entry in the cell in row (x, w) and column (y, z) is denoted by $\beta^{\mathcal{F}}_{x,w,y,z}$ and equals $\Pr[w, z|x, y]$.

We will also consider the compressed matrix $\beta^{\mathcal{F}}_B$ whose rows are indexed by Bob inputs y and rows are indexed by Alice views $(x, w) \in \mathcal{X} \times \mathcal{W}$. The entry in the cell in row (x, w) and column y is denoted by $\beta^{\mathcal{F}}_{x,w,y}$ and equals $\Pr[w|x, y]$.

The maps ϕ_A and ϕ_B. These maps define equivalence classes of views. Roughly, two rows (or columns) in $\beta^{\mathcal{F}}$ lie in the same equivalence class if they are scalar multiples of each other. Formally, for each $(x, w) \in \mathcal{X} \times \mathcal{W}$, let the vector $\beta^{\mathcal{F}}|(x, w) \in \mathbb{R}^{nr}$ be the row indexed by (x, w) in the matrix $\beta^{\mathcal{F}}$. Let $\phi_A : [m] \times [q] \to [\ell]$ (for a sufficiently large $\ell \leq mq$) be such that $\phi_A(x, w) = \phi_A(x', w')$ iff $\beta^{\mathcal{F}}|_{(x,w)} = c \cdot \beta^{\mathcal{F}}|_{(x',w')}$ for some positive scalar c. ϕ_B is defined similarly for column vectors indexed by Bob views (y, z).

3.2 Characterizing Irredundancy

Redundancy in a function allows at least one party to deviate in its behavior in the ideal world and not be detected (with significant probability) by an environment. In our protocols, which are designed to detect deviation, it is important to use a function in a form in which redundancy has been removed. We use definitions of irredundancy from [16], and give a brief overview here for completeness. There also exists an efficient algorithm to remove redundancies following [16].

Irredundancy of a 2-Party Secure Function Evaluation Function. Recall that a 2-party SFE function f with input domains, $X \times Y$ and output domain $W \times Z$ is defined by probabilities $\mathfrak{p}^f[w, z|x, y]$. Output redundancies identify if the output can be compressed to remove aspects of the output that are useless for the adversary's goal of gaining information about the honest party's inputs. For input redundancy, we define left and right redundancy of f as follows. Below, $|X| = m, |Y| = n, |W| = q, |Z| = r$. To define left-redundancy, consider representing f by the matrices $\{P^x\}_{x \in X}$ where each P^x is an $nr \times q$ matrix with $P^x_{(y,z),w} = \mathfrak{p}^f[w, y, z|x]$. Here, $\mathfrak{p}^f[w, y, z|x] \triangleq \frac{1}{n}\mathfrak{p}^f[w, z|x, y]$ (where we pick y independent of x, with uniform probability $\mathfrak{p}^f[y|x] = \frac{1}{n}$).

Definition 2. *For an SFE function $f : X \times Y \to W \times Z$, represented by matrices $\{P^x\}_{x \in X}$, with $P^x_{(y,z),w} = \Pr[w, y, z|x]$, we say that an input $\hat{x} \in X$ is* left-redundant *if there is a set $\{(\alpha_x, M_x)|x \in X\}$, where $0 \leq \alpha_x \leq 1$ with $\sum_x \alpha_x = 1$, and each M_x is a $q \times q$ stochastic matrix such that if $\alpha_{\hat{x}} = 1$ then $M_{\hat{x}} \neq I$, and $P^{\hat{x}} = \sum_{x \in X} \alpha_x P^x M_x$. We say $\hat{x}$ is* strictly left-redundant *if it is left-redundant as above, but $\alpha_{\hat{x}} = 0$. We say $\hat{x}$ is* self left-redundant *if it is left-redundant as above, but $\alpha_{\hat{x}} = 1$ (and hence $M_{\hat{x}} \neq I$). We say that f is* left-redundancy free *if there is no $x \in X$ that is left-redundant.*

Right-redundancy notions for inputs $\hat{y} \in Y$ are defined analogously. f is said to be *redundancy-free* if it is left-redundancy free and right-redundancy free.

3.3 Statistically Testable Function Evaluation

Statistical tests [16] help ensure that a cut-and-choose technique can be used to verify an adversary's claims about what inputs it sent to a 2-party function and what outputs it received, when the verifier has access to only the other end of the function. It is important to note that such statistical tests can only be applied when an adversary declares (or commits to) his claimed inputs beforehand and is not allowed to adaptively choose his input claims adaptively based on function output. Kraschewski et al. [16] show that *evaluation of a 2-party function is statistically testable iff the function is redundancy free.* We repeat the statistical test game and the proof of the above statement in the full version of the paper.

3.4 Weak Converse of the Channel Coding Theorem, Generalization

A converse of the channel coding theorem states that message transmission is not possible over a noisy channel at a rate above its capacity, except with a non-vanishing rate of errors. We use a generalization of the (weak) converse of channel coding theorem due to [16] where the receiver can adaptively choose the channel based on its current view. Then if in at least a μ fraction of the transmissions, the receiver chooses channels which are noisy (i.e., has capacity less than that of a noiseless channel over the same input alphabet), it is possible to lower bound its probability of error in predicting the input codeword as a function of μ, an upper bound on the noisy channel capacities, and the rate of the code. We import the following lemma from [16].

Imported Lemma 2. *Let $\mathcal{F} = \{\mathcal{F}_1, \ldots, \mathcal{F}_K\}$ be a set of K channels which take as input alphabets from a set Λ, with $|\Lambda| = 2^\lambda$. Let $\mathcal{G} \subseteq [K]$ be such that for all $i \in \mathcal{G}$, the capacity of the channel $\mathcal{F}_i$ is at most $\lambda - c$, for a constant $c > 0$.*

Let $\mathcal{C} \subseteq \Lambda^N$ be a rate $R \in [0, 1]$ code. Consider the following experiment: a random codeword $c_1 \ldots c_N \equiv \boldsymbol{c} \xleftarrow{\$} \mathcal{C}$ is drawn and each symbol $c_1 \ldots c_N$ is transmitted sequentially; the channel used for transmitting each symbol is chosen (possibly adaptively) from the set $\mathcal{F}$ by the receiver.

Conditioned on the receiver choosing a channel in $\mathcal{G}$ for μ or more transmissions, the probability of error of the receiver in predicting $\boldsymbol{c}$ is

$$P_e \geq 1 - \frac{1}{NR\lambda} - \frac{1 - c\mu/\lambda}{R}.$$

4 Summary and Exhaustive Case Analysis

4.1 Summary

Given a 2-party SFE $\mathcal{F}$, we represent by $\mathcal{F}_{A \to B}$ the functionality which takes its first input from Alice and its second input from Bob. Similarly, we define the functionality $\mathcal{F}_{B \to A}$. We say $\mathcal{F}$ reduces to $\mathcal{G}$, represented by $\mathcal{F} \sqsubseteq_{\text{UC}} \mathcal{G}$, if there exists a information-theoretic UC-secure protocol for $\mathcal{F}$ in the $\mathcal{G}$-hybrid. The functionality $\mathcal{F}^{\otimes n}$ represents n independent copies of the functionality $\mathcal{F}$.

We observe that Kraschewski et al. [16] obtain oblivious transfer using any finite randomized functionality $\mathcal{F}$ with a non-simple core, in a fixed direction, if there exist commitments in both directions. Furthermore, they already show that for any finite randomized functionality $\mathcal{F}$ with a non-simple core, commitments can be obtained from either Alice to Bob or from Bob to Alice.

Our main technical contribution will be to show that, in fact, for any finite randomized functionality $\mathcal{F}$ with a non-simple core, commitments can be obtained *both* from Alice to Bob *and* from Bob to Alice, by using $\mathcal{F}$ in a fixed direction.

Analogous to the above statement, we also have a statement where $\mathcal{F}_{A \to B}$ is replaced by $\mathcal{F}_{B \to A}$. Next, once we get $\mathcal{F}_{\mathsf{OT}}$ at constant rate, we can implement $\mathcal{F}_{B \to A}$ at constant rate using [12]. This gives our main result.

Theorem 1 (Reversible Characterization). *For every 2-party SFE $\mathcal{F}$: if $\mathcal{F}_{\mathsf{OT}} \sqsubseteq_{\mathrm{UC}} \mathcal{F}$ in the malicious setting (possibly using $\mathcal{F}$ in both directions), then there exists $c > 0$ such that $\mathcal{F}_{A\to B}^{\otimes\sigma} \sqsubseteq_{\mathrm{UC}} \mathcal{F}_{B\to A}^{\otimes\kappa}$ in the malicious setting and $\sigma \geq c\kappa$.*

Again, once we have commitments in both directions, by using the SFE functionality in only one direction, we can use the compiler of [16] to directly obtain the following theorem.

Theorem 2 (Fixed-Role Completeness Characterization). *For every 2-party SFE $\mathcal{F}$: $\mathcal{F}_{\mathsf{OT}} \sqsubseteq_{\mathrm{UC}} \mathcal{F}$ in the malicious setting (possibly using $\mathcal{F}$ in both directions) if and only if there exists $c > 0$ such that $\mathcal{F}_{\mathsf{OT}}^{\otimes\sigma} \sqsubseteq_{\mathrm{UC}} \mathcal{F}_{A\to B}^{\otimes\kappa}$ in the malicious setting and $\sigma \geq c\kappa$.*

4.2 Exhaustive Case Analysis

First, we will classify any functionality $\mathcal{F}$ with a non-simple redundancy-free core, into a set of exhaustive cases. In each case, we demonstrate that it is possible to obtain commitments using $\mathcal{F}$, from Bob to Alice. Let $\mathcal{V}_B$ denote the set of extreme Bob views, and $\widehat{Y}$ be the set of inputs of Bob that admit at least one extreme view, that is, $\widehat{Y} := \{y\colon \exists z, \text{ such that } (y,z) \in \mathcal{V}_B\}$. Let $\widehat{V}_B$ denote the set of all Bob views corresponding to inputs in $\widehat{Y}$, that is $\widehat{V}_B = \{(y,z)\colon y \in \widehat{Y}\}$. Our cases are listed in Table 1.

Table 1. Exhaustive summary of cases

1	There exists an Alice view with which ≥ 2 Bob views in $\widehat{V}_B$ intersect.
(a)	There exists an Alice view with which ≥ 2 *extreme* Bob views in $\widehat{V}_B$ intersect. In this case, it is possible to obtain commitments from Bob to Alice [16].
(b)	There exists an Alice view with which one extreme and ≥ 1 non-extreme Bob view in $\widehat{V}_B$ intersect.
2	No two Bob views in $\widehat{V}_B$ intersect with the same Alice view.

Claim. In a non-simple functionality $\mathcal{F}$, if no two extreme Bob views intersect with the same Alice view, then there exists an Alice view which intersects with one extreme and one non-extreme Bob view.

Proof. In a non-simple functionality $\mathcal{F}$, if no two extreme Bob views intersect with the same Alice view, then we have the following possibilities:

1. There is an Alice view intersecting an extreme and non-extreme Bob view,
2. Or, there is an Alice view which intersects 2 non-extreme Bob views,
3. Or, no Alice view intersects any two Bob views.

We show that $2 \implies 1$, and 3 contradicts the fact that $\mathcal{F}$ is non-simple.

Let the number of extreme views of Bob be γ. Denote the extreme views of Bob by (y_i^*, z_i^*), for $i \in [\gamma]$. Suppose Alice view $V_A = (x, z)$ intersects with two non-extreme Bob views $V_B^1 = (y_1, z_1)$ and $V_B^2 = (y_2, z_2)$. Then, the columns $\beta^{\mathcal{F}}_{|(y_1,z_1)}$ and $\beta^{\mathcal{F}}_{|(y_2,z_2)}$ of $\beta^{\mathcal{F}}$ have non-zero entries in the row corresponding to (x, z). Since both views (V_B^1, V_B^2) are non-extreme, the columns $\beta^{\mathcal{F}}_{|(y_1,z_1)}$ and $\beta^{\mathcal{F}}_{|(y_2,z_2)}$ of $\beta^{\mathcal{F}}$ can be expressed as a linear combination of extreme columns (y_i^*, z_i^*), for $i \in [\gamma]$. This means that there necessarily exists at least one extreme view $(y^*, z^*) \in \{(y_1^*, z_1^*), (y_2^*, z_2^*), \ldots (y_\gamma^*, z_\gamma^*)\}$ such that the column $\beta^{\mathcal{F}}_{|(y^*,z^*)}$ of $\beta^{\mathcal{F}}$ has a non-zero entry in the row corresponding to (x, z). This proves $2 \implies 1$.

Suppose that in a non-simple functionality $\mathcal{F}$, no view of Alice intersects with any two views of Bob. That is, every view of Alice intersects with at most one view of Bob. In this case, the common information/kernel obtained after function evaluation is the view of Bob. It is straightforward to see that both parties can independently sample their views, conditioned on any view of Bob. This completes the proof of this claim.

In the following sections, we construct commitments $\mathcal{F}_{\mathsf{com},B\to A}$, for any functionality $\mathcal{F}$ depending on which of the two cases it falls in.

We observe that in case there exists an Alice view with which at least two *extreme* Bob views in $\widehat{V}_B$ intersect, the protocol of [16] can be used to obtain commitments from Bob to Alice. We re-state their result in the following lemma. In the following lemma, we will recall appropriate notions of confusability from [16]. Any functionality $\mathcal{F}$ in which at least two *extreme* Bob views in $\widehat{V}_B$ intersect with a common Alice view, will be said to have a confusable $\mathfrak{b}^{\mathcal{F}}$.

Imported Lemma 3. *Denote the set of extreme views of Bob by $\mathfrak{b}^{\mathcal{F}}$. For each Alice view (x, w) denote by $\mathfrak{b}^{\mathcal{F}}|_{(x,w)}$ all the extreme views of Bob which intersect with the specific Alice view (x, w). That is, $\mathfrak{b}^{\mathcal{F}}|_{(x,w)}$ is the set of extreme views (y, z) of Bob such that the row in $\beta^{\mathcal{F}}$ indexed by (y, z) has a positive entry in the column indexed by (x, w). $\mathfrak{b}^{\mathcal{F}}$ is said to be confusable if there exists $(x, w) \in \mathcal{X} \times \mathcal{W}$ and two elements $(y_1, z_1), (y_2, z_2) \in \mathfrak{b}^{\mathcal{F}}|_{(x,w)}$ such that $\phi_B(y_1, z_1) \neq \phi_B(y_2, z_2)$. $\mathfrak{a}^{\mathcal{F}}$ is defined similarly for extreme views of Alice. Then,*

1. *If the redundancy-free core of $\mathcal{F}$ is simple, either $\mathfrak{a}^{\mathcal{F}}$ or $\mathfrak{b}^{\mathcal{F}}$ is confusable.*
2. *If $\mathfrak{a}^{\mathcal{F}}$ is confusable, it is possible to obtain commitments from Alice to Bob. If $\mathfrak{b}^{\mathcal{F}}$ is confusable, it is possible to obtain commitments from Bob to Alice.*

5 Case 1(b): Commitments

5.1 Construction

Let $\mathcal{V}_B$ denote the set of all extreme views of Bob and let $\widehat{Y}$ denote the set of all inputs of Bob that contain at least one extreme view, that is $\widehat{Y} := \{y\colon \exists z, \text{ such that } (y, z) \in \mathcal{V}_B\}$. Further, let $\widehat{V}_B$ denote the set of all Bob views corresponding to inputs in $\widehat{Y}$, that is $\widehat{V}_B = \{(y, z)\colon y \in \widehat{Y}\}$.

In this section, we demonstrate how to obtain commitments from any functionality $\mathcal{F}$ for which the following is true: $\widehat{V}_B$ "is confusable", that is, there exists an Alice view (x, w) and two distinct Bob views $(\widehat{Y}_1, \hat{z}_1)$ and $(\widehat{Y}_2, \hat{z}_2) \in \widehat{V}_b$ (where possibly $\widehat{Y}_1 = \widehat{Y}_2$) such that $\beta^{\mathcal{F}}_{x,\widehat{Y}_1,w,\hat{z}_1} \neq 0$ and $\beta^{\mathcal{F}}_{x,\widehat{Y}_2,w,\hat{z}_2} \neq 0$. The protocol is described in Fig. 4.

5.2 Proof of Security

Receiver Security (Statistical Binding/Extractability). In the UC setting, it suffices to consider a dummy sender $\mathcal{S}$ and malicious environment $\mathcal{Z}_{\mathcal{S}}$, such that the dummy sender forwards all messages from $\mathcal{Z}_{\mathcal{S}}$ to the honest receiver/simulator, and vice-versa. Without loss of generality, the malicious simulation strategy $\mathsf{Sim}_{\mathcal{S}}$ can be viewed to interact directly with $\mathcal{Z}_{\mathcal{S}}$. $\mathsf{Sim}_{\mathcal{S}}$ is described in Fig. 5.

Lemma 1. *There exists a constant c such that the simulation error for the malicious sender is at most $2^{-c\kappa}$.*

Proof. The simulator performs Steps $1(a), (b)$ and (c) as per the honest receiver strategy, and also emulates the functionality $\mathcal{F}$ honestly for the sender. It remains to show that the unique bit b' extracted by the simulator equals the bit b committed by the sender Bob. The crux of this proof relies on the fact that the protocol requires the sender to use one extreme view and on the minimum distance of the code used.

Bob cannot claim non-extreme views to be extreme. In the opening made by Bob, consider the positions where Bob claimed his view to be extreme, that is, $(y_i, z_i) = (y^*, z^*) \in \mathcal{V}_B$, such that the equivalence class of this view $\phi_B(y^*, z^*) = \Phi$. Consider the fraction of these positions where the actual view of Bob (y', z') such that $\phi_B(y', z') \neq \Phi$. In these positions, the expected view of Alice is given by a linear combination of the columns $\beta^{\mathcal{F}}|_{(y',z')}$ (with coordinates scaled appropriately). If this linear combination is not close to the vector $\beta^{\mathcal{F}}|_{(y^*,z^*)}$ (scaled appropriately) then with all but negligible probability, the opening will not be accepted by the receiver. On the other hand, if the linear combination is close to $\beta^{\mathcal{F}}|_{(y^*,z^*)}$, since $\beta^{\mathcal{F}}|_{(y^*,z^*)}$ is outside the linear span of other $\beta^{\mathcal{F}}|_{(y',z')}$ with $\phi_B(y'z') \neq \phi_B(y^*, z^*)$, only at a small number (sub-linear fraction, say $\kappa^{2/3}$) of places can Bob open to (y^*, z^*) but have had an actual view (y', z'). This is because, an extreme view can't be expressed as a linear combination of other views of Bob, without being detected by Alice with constant probability.

Bob uses close to uniform distribution over inputs in $\widehat{Y}_B$. Consider an input $y^* \in \widehat{Y}_B$ and let (y^*, z^*) denote its corresponding extreme view. Alice will not accept the extreme view (y^*, z^*) in the opening of Bob (except with probability $2^{-c\kappa^{2/3}}$) unless Bob actually obtained the particular view in all but $\kappa^{2/3}$ of these

Inputs: Sender $\mathcal{S}$ has input bit $\mathsf{bit} \in \{0,1\}$ and receiver $\mathcal{R}$ has no input.
Hybrid: $\mathcal{F}$ for non-simple function $\mathcal{F}$, and $\widehat{Y}$ as defined above is confusable. $\mathcal{F}$ provides commitments (Com) from Alice to Bob.
The protocol is presented in terms of a $(\kappa, \kappa - \kappa^{15/16}, \Omega(\kappa^{15/16}))$-linear code $\mathcal{C}$ over the binary alphabet. (An explicit code is not necessary: the receiver can pick random $\Omega(\kappa^{15/16})$ "parity checks" to construct the code and announce it to the sender.) The protocol is parameterized by κ.

1. Commit Phase:
   (a) $\mathcal{R}$ (Alice) picks inputs $(X_1, X_2, \ldots, X_{2\kappa^2})$ uniformly from $\mathcal{X}^{2\kappa^2}$. She commits to each of them using fresh randomness and sends $\mathsf{Com}(X_1), \mathsf{Com}(X_2), \ldots \mathsf{Com}(X_{2\kappa^2})$ to $\mathcal{S}$.
   (b) $\mathcal{S}$ (Bob) picks inputs $(Y_1, Y_2, \ldots, Y_{2\kappa^2})$ from a uniform distribution over $\widehat{Y}^{2\kappa^2}$. $\mathcal{R}$ and $\mathcal{S}$ invoke $\mathcal{F}$, 2κ times, with inputs $(X_1, X_2, \ldots, X_{2\kappa^2})$ and $(Y_1, Y_2, \ldots, Y_{2\kappa^2})$ respectively.
   (c) Cut-and-Choose: $\mathcal{R}$ picks $r_1 \xleftarrow{\$} \{0,1\}^*$ and sends $\mathsf{Com}(r_1)$ to $\mathcal{S}$. $\mathcal{S}$ sends $r_2 \xleftarrow{\$} \{0,1\}^*$ to $\mathcal{R}$. $\mathcal{R}$ uses randomness $(r_1 \oplus r_2)$ to pick a subset $I \leftarrow \binom{2\kappa^2}{[\kappa^2]}$ of the κ^2 indices. $\mathcal{R}$ decommits to r_1. Furthermore, for all $i \in I$, $\mathcal{R}$ decommits to input X_i and also opens her view (X_i, W_i).
   $\mathcal{S}$ aborts if the decommitments are not correct, or the inputs of $\mathcal{R}$ are not close to a uniform distribution, or if (X_i, W_i) for $i \in I$ satisfy the consistency checks in the Left-Statistical-Tests.
   Else, $\mathcal{S}$ and $\mathcal{R}$ set $S = [2\kappa^2] \setminus I$ and reorder the indices in S to $[\kappa^2]$.
   (d) $\mathcal{S}$ does the following for all $i \in [\kappa]$.
      – Construct the j^{th} characteristic vector $\mathbf{u}_j$ such that for all $i \in [\kappa]$, $u_{j,i} = 0$ if and only if $(Y_{j\kappa+i}, Z_{j\kappa+i}) \in \mathcal{V}$, else $u_{j,i} = 1$.
      – Pick κ random codewords $\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_\kappa \in \mathcal{C}^\kappa$. Pick $h \xleftarrow{\$} \mathcal{H}$, a universal hash function mapping $\{0,1\}^{\kappa^2} \to \{0,1\}$, and for $j \in [\kappa]$, compute $y = h(\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_\kappa) \oplus \mathsf{bit}, \mathsf{offset}_j = (\mathbf{c}_j \oplus \mathbf{u}_j)$. Send $(h, y, \mathsf{offset}_1, \mathsf{offset}_2, \ldots \mathsf{offset}_\kappa)$ to $\mathcal{R}$.
2. Reveal Phase:
   (a) $\mathcal{S}$ sets $b' = \mathsf{bit}, \mathbf{u}'_j = \mathbf{u}_j$ for $j \in [\kappa]$ and sends $b', \mathbf{u}'_1, \mathbf{u}'_2, \ldots \mathbf{u}'_\kappa$ to $\mathcal{R}$ as his opening. $\mathcal{S}$ also sends (Y_i, Z_i) for all $i \in [\kappa^2]$, to $\mathcal{R}$.
   (b) $\mathcal{R}$ accepts if all the following conditions hold:
      – For $j \in [\kappa]$, $\mathbf{c}_j = \mathbf{u}'_j \oplus \mathsf{offset}_j$, is a valid codeword.
      – $b' = h(\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_\kappa) \oplus y$.
      – For all $i \in [\kappa^2]$, (Y_i, Z_i) satisfy input-output frequency tests.

Fig. 4. $\mathcal{F}_{\mathsf{com}}$ in Case 1(b).

The simulator $\mathsf{Sim}_{\mathcal{S}}$ does the following.

1. Commit Phase:
   (a) $\mathsf{Sim}_{\mathcal{S}}$ picks inputs $(X_1, X_2, \ldots, X_{2\kappa^2})$ uniformly from $\mathcal{X}^{2\kappa^2}$. $\mathsf{Sim}_{\mathcal{S}}$ then commits to each of them using fresh randomness and sends $\mathsf{Com}(X_1), \mathsf{Com}(X_2), \ldots \mathsf{Com}(X_{2\kappa^2})$ to $\mathcal{S}$. Note that $\mathsf{Sim}_{\mathcal{S}}$ has the capability to equivocate these commitments.
   (b) $\mathsf{Sim}_{\mathcal{S}}$ obtains inputs $(Y_1, Y_2, \ldots Y_{2\kappa^2})$ from $\mathcal{S}$ and emulates the functionality $\mathcal{F}$ honestly for $\mathcal{S}$ with inputs $(X_1, X_2, \ldots X_{2\kappa^2})$ and $(Y_1, Y_2, \ldots Y_{2\kappa^2})$.
   (c) Cut-and-Choose: $\mathsf{Sim}_{\mathcal{S}}$ picks $r_1 \xleftarrow{\$} \{0,1\}^*$ and sends $\mathsf{com}_1 = \mathsf{Com}(r_1)$ to $\mathcal{S}$. $\mathcal{S}$ sends $r_2 \xleftarrow{\$} \{0,1\}^*$ to $\mathsf{Sim}_{\mathcal{S}}$. $\mathsf{Sim}_{\mathcal{S}}$ uses $(r'_1 \oplus r_2)$ to pick subset $I \xleftarrow{\$} \binom{2\kappa^2}{[\kappa^2]}$ of the κ^2 indices. $\mathsf{Sim}_{\mathcal{S}}$ decommits com_1 to r_1 and, for all $i \in I$, $\mathsf{Sim}_{\mathcal{S}}$ decommits to input X_i and also opens the view (X_i, W_i). Set $S = [2\kappa^2] \setminus I$ and reorder the indices in S to $[\kappa^2]$.
   (d) $\mathsf{Sim}_{\mathcal{S}}$ obtains $(h, y, \mathsf{offset}_j)$ for $j \in [\kappa]$ from $\mathcal{S}$. It constructs characteristic vectors $\mathbf{u}_j$ such that for all $i \in \mathcal{S}$, $u_i = 0$ if and only if $(Y_i, Z_i) \in \mathcal{V}$, else $u_i = 1$. It then computes $\tilde{c}_j = \mathbf{u}_j \oplus \mathsf{offset}_j$, sets c'_j to be the nearest codeword[a] to $\tilde{c}_j$, and sets bit $b' = y \oplus h(c'_1, c'_2, \ldots c'_\kappa)$.
2. Reveal Phase:
   (a) Obtain $b', u'_1, u'_2, \ldots u'_\kappa, (Y_i, Z_i)$ for all $i \in [\kappa^2]$ from $\mathcal{S}$ as his opening.
   (b) Allow the ideal functionality to output the extracted bit b' if all the following conditions hold (and otherwise reject):
      – $(u'_j \oplus \mathsf{offset}_j)$ is a valid codeword for $j \in [\kappa]$.
      – (Y_i, Z_i) for all $i \in [\kappa^2]$ satisfy input-output frequency tests.

[a] If the nearest codeword is not unique, then $\mathsf{Sim}_{\mathcal{S}}$ commits to an arbitrary bit.

Fig. 5. Sender simulation strategy in Case 1(b).

indices. In order to obtain the view (y^*, z^*) in $1/\hat{\mathcal{Y}}_B \times \beta^{\mathcal{F}}_{z^*|y^*}$ fraction of indices, Bob should have used the input y^* to the functionality with probability at least $1/|\hat{\mathcal{Y}}_B|$.

Bob cannot equivocate outputs. Since Bob uses all inputs in $\widehat{Y}_B$ with nearly the correct probability (except on $O(\kappa^{2/3})$ indices, then in the real and simulated worlds, he also obtains views in $\widehat{V}_B$ with nearly the expected probability. Furthermore, he cannot obtain views not in $\mathcal{V}_B$ and pretend that they were in $\mathcal{V}_B$ except for $O(\kappa^{7/8})$ indices. Therefore, he cannot obtain views in $\mathcal{V}_B$ and pretend that they were not in $\mathcal{V}_B$ except for $O(\kappa^{7/8})$ indices, otherwise he will fail the frequency tests on the outputs.

To summarize,

The simulator $\mathsf{Sim}_{\mathcal{R}}$ does the following.

1. Commit Phase:
   (a) $\mathsf{Sim}_{\mathcal{R}}$ obtains commitments $c_1, c_2, \ldots c_{2\kappa^2}$ from $\mathcal{R}$.
   (b) $\mathsf{Sim}_{\mathcal{R}}$ obtains inputs $(X_1, X_2, \ldots X_{2\kappa^2})$ from $\mathcal{R}$ and emulates the functionality $\mathcal{F}$ honestly for $\mathcal{R}$ with inputs $(X_1, X_2, \ldots X_{2\kappa^2})$ and $(Y_1, Y_2, \ldots Y_{2\kappa^2})$.
   (c) Cut-and-Choose: $\mathsf{Sim}_{\mathcal{R}}$ obtains com_1 from $\mathcal{R}$. $\mathsf{Sim}_{\mathcal{R}}$ sends $r_2 \xleftarrow{\$} \{0,1\}^*$ to $\mathcal{R}$. $\mathcal{R}$ decommits to r_1 and sends subset $I \xleftarrow{\$} \binom{2\kappa^2}{[\kappa^2]}$ of the κ^2 indices. For all $i \in I$, $\mathsf{Sim}_{\mathcal{R}}$ obtains decommitments X_i and also the openings (X_i, W_i). $\mathsf{Sim}_{\mathcal{R}}$ aborts if the decommitments are not correct, or the inputs of $\mathcal{R}$ are not from a uniform distribution, or if (X_i, W_i) for $i \in I$ do not satisfy the consistency checks in Left-Statistical-Tests.
   (d) $\mathsf{Sim}_{\mathcal{R}}$ follows honest strategy to commit to a uniformly random bit $\mathsf{bit}' \xleftarrow{\$} \{0,1\}$.

Fig. 6. Receiver simulation strategy in Case 1(b).

- For any input $y^* \in \hat{\mathcal{Y}}_B$, if Alice accepts the decommitment, Bob should have actually used the input to the functionality $\mathcal{F}$ in exactly $1/|\hat{\mathcal{Y}}_B|$ fraction of the places, except cheating in at most $\kappa^{2/3}$ indices.
- For any (extreme) view $(y^*, z^*) \in \widehat{V}_B$, Bob cannot have claimed to obtain (y^*, z^*) at specific indices unless he obtained the view in (y^*, z^*) at all but $O(\kappa^{7/8})$ of these indices.
- For any non-extreme view $(y^*, z^*) \in \widehat{V}_B$, Bob cannot have claimed to obtain (y^*, z^*) at specific indices unless he actually obtained *some* non-extreme view at all but $O(\kappa^{7/8})$ of these indices.

By using a code such that the minimum distance of the code $(\Omega(\kappa^{15/16}))$ is much larger than the number of positions where the sender can cheat as above $(O(\kappa^{7/8}))$, we guarantee that the sender is bound to his committed bit.

Specifically, the simulator computes the nearest codeword to the codeword extracted from the sender, and uses this to extract his committed bit. The sender cannot equivocate this codeword without cheating in $\Omega(\kappa^{15/16})$ views, and if he does so, his decommitment is not accepted except with probability at least $(1 - 2^{-c\kappa})$. This completes the proof of this lemma. x;w N2=3 0. If not, it aborts the protocol.

Sender Security (Statistical Hiding/Equivocability). It suffices to consider a dummy receiver $\mathcal{R}$ and malicious environment $\mathcal{Z}_{\mathcal{R}}$, such that the dummy receiver forwards all messages from $\mathcal{Z}_{\mathcal{R}}$ to the honest sender/simulator, and vice-versa. Without loss of generality, the malicious simulation strategy $\mathsf{Sim}_{\mathcal{R}}$ can be viewed to interact directly with $\mathcal{Z}_{\mathcal{R}}$. $\mathsf{Sim}_{\mathcal{R}}$ is described in Fig. 6.

Lemma 2. *There exists a constant c such that the simulation error for the malicious receiver is at most $2^{-c\kappa}$.*

Proof. Consider the use of the function f as a "channel", which accepts $x_{i,j}$ from Alice, $c_{i,j}$ from Bob, samples $(y_{i,j}, w_{i,j}, z_{i,j})$ and outputs $z_{i,j}$ to Bob, and $a_{i,j} \oplus c_{i,j}$ to Alice where $a_{i,j} = \phi_B(y_{i,j}, z_{i,j})$.

The cut-and-choose verification in Step 1(c) ensures that Alice uses (close to) a uniform distribution over her inputs. This is done by invoking Left-Statistical-Tests on committed inputs $X_1, x_2 \ldots X_{2\kappa^2}$ of Alice, and her claimed outputs $W_1, W_2, \ldots W_{2\kappa^2}$.

This test ensures that she obtains the view (x, w) that intersects with an extreme and a non-extreme view in $\widehat{V}_B$ in at least $\beta^{\mathcal{F}}_{|x,z}\kappa^2 - O(\kappa)$ invocations. At all these invocations, given her view, Alice has confusion about whether the corresponding view of Bob was extreme or non-extreme. Therefore, the views obtained by Alice act as a channel transmitting information about the corresponding views of Bob. It is that the capacity of this channel is a constant, that is less than 1.

Then we appeal to an extension of the weak converse of Shannon's Channel Coding Theorem (Imported Lemma 2) to argue that since the code has rate $1-o(1)$, Alice errs in decoding each codeword with at least a constant probability. We need this extension of the (weak) converse of the channel coding theorem to handle that the facts that:

1. The receiver can adaptively choose the channel characteristic, by picking $y_{i,j}$ adaptively, and
2. Some of the channel characteristics that can be chosen include a noiseless channel, but the number of times such a characteristic can be used cannot be large (except with negligible probability). The reason this restriction can be enforced is because Alice's view intersects with views of Bob corresponding to characteristic index 0 and 1.

Then, applying the Leftover Hash Lemma, we get that for a universal hash function h, if Bob sends κ codewords over such a channel, the output of the hash function is at least $1 - 2^{-c\kappa}$ close to uniform. Thus, the simulation error is at most $2^{-c\kappa}$.

6 Case 2: Commitments

As before, let $\mathcal{V}_B$ denote the set of all extreme views of Bob and let $\widehat{Y}$ denote the set of all inputs of Bob that contain at least one extreme view, that is $\widehat{Y} := \{y\colon \exists z, \text{ such that } (y, z) \in \mathcal{V}_B\}$. Further, let $\widehat{V}_B$ denote the set of all Bob views corresponding to inputs in $\widehat{Y}$, that is $\widehat{V}_B = \{(y, z)\colon y \in \widehat{Y}\}$.

In this section, we demonstrate how to construct commitments from any function $\mathcal{F}$ for which the following is true: $\widehat{V}_B$ has no confusion, that is no two Bob views in $\widehat{V}_B$ intersect with the same Alice view. In other words, all views corresponding to all inputs $y \in \hat{Y}$ are extreme and also disjoint.

First, we make the following basic observation about disjoint extreme views. Let $\mathcal{V}_B$ denote the set of extreme views of Bob. If there is no Alice view V_A which intersects two or more Bob views in $\mathcal{V}_B$, then each Bob view in $\mathcal{V}_B$ is in one-to-one correspondence with the equivalence class ϕ of Alice views. In particular, each Bob view (y, z) in $\mathcal{V}_B$ reveals $\phi(V_A)$ for any view V_A which the Bob view (y, z) intersects. Then, we note that for all inputs $\hat{y}$ in $\hat{Y}$, each output view $(\hat{y}, \hat{z})$ completely reveals the equivalence class ϕ of Alice views. The following lemma is imported from [16].

Imported Lemma 4 [16]**.** *Suppose $\hat{Y} \subseteq Y$ is a set of inputs, where each view $(\hat{y}, z)$ for each input $\hat{y} \in \hat{Y}$ is completely revealing about the equivalence class ϕ of Alice views. If some input $y^* \in Y \backslash \hat{Y}$ can be* fully*-mimicked by $\hat{Y}$ then y^* is a strictly redundant input.*

Note that if $y \notin Y_0$ can be mimicked by Y_0, it does not necessarily mean that y^* is redundant, because for redundancy there must exist a probabilistic mapping from $Y_0 \times Z$ to $y^* \times Z$. However, if Y_0 are all completely revealing about the equivalence class ϕ of Alice views, it can be shown that y^* is indeed redundant. For completeness, we repeat the formal proof from [16] in the full version.

Lemma 3. *Suppose $\hat{Y} \subseteq Y$ is a set of inputs, where each view $(\hat{y}, z)$ for each input $\hat{y} \in \hat{Y}$ is completely revealing about an equivalence class of Alice views. Let $Y' = Y \backslash \hat{Y}$. If every input in Y' can be mimicked using a probability distribution over other inputs that assigns constant non-zero weight to $\hat{Y}$, then every input in Y' is strictly redundant.*

Proof. Our proof follows along the lines of Gaussian elimination, removing one variable dependency at a time. As is the case with Gaussian elimination, the invariant we maintain is that the i^{th} variable does not influence anything beyond the i^{th} constraint. Our proof uses an inductive argument where the above invariant is iteratively maintained in each iteration.

Consider inputs $y^* \in Y'$ that can be mimicked using non-zero constant weight in $\hat{Y}$. We prove that if all inputs $y^* \in Y'$ can be mimicked using non-zero constant weight in $\hat{Y}$, then they can in fact be *fully* mimicked only by $\hat{Y}$. Once we prove this, we can invoke Imported Lemma 4 to prove that all such inputs y^* must be strictly redundant. We first set up some notation for the proof.

Notation. Let $Y' = \{y_1^*, y_2^*, \ldots y_\ell^*\}$ and $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \ldots \hat{y}_{|\mathcal{Y}|-\ell}\}$, where $\ell < |\mathcal{Y}|$. Let M be an $\ell \times (\ell + 1)$ matrix whose entries are set such that for all $i \in [\ell]$, $y_i^* = \sum_{j \in [\ell]} (M_{i,j}) y_j^* + \sum_{j \in [|\mathcal{Y}|-\ell]} \mathfrak{p}_{i,j} \hat{y}_j$. Then $M_{i,(\ell+1)} = \sum_{j \in [|\mathcal{Y}|-\ell]} \mathfrak{p}_{i,j}$.

That is, for $(i, j) \in [\ell] \times [\ell]$, the row M_i denotes the probability distribution over inputs y_j^* used to mimic the input y_i^*. The entry $M_{i,\ell+1}$ denotes the total weight of inputs in $\hat{Y}$ assigned by the probability distribution, for mimicking the input y_i^*.

Transformation. Assume, contrary to the statement of the lemma, that every entry $M_{i,\ell+1}$ *for all* $i \in [1, \ell]$ is a non-zero constant, denote the i^{th} such entry

by c_i. We give a series of transformations on M, such that the resulting matrix M' has non-zero entries only in the $(\ell+1)^{th}$ column. This suffices to prove that all inputs can be fully mimicked using some distribution over inputs *only* in $\hat{Y}$, therefore proving the lemma.

We inductively set $M_{i,j} = 0$ for all $(i,j) \in [1,k] \times [1,k]$.

Base Case. In the base case, if $M_{1,1} = 0$, we are done.

Else we can rewrite the first row equations as:

$$y_1^* = \sum_{j\in[\ell]} (M_{i,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}_{i,j}\hat{y}_j \tag{1}$$

$$= M_{1,1}y_1^* + \sum_{j\in[2,\ell]} (M_{i,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}_{i,j}\hat{y}_j \tag{2}$$

$$y_1^* - M_{1,1}y_1^* = \sum_{j\in[2,\ell]} (M_{1,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}_{1,j}\hat{y}_j \tag{3}$$

$$y_1^*(1 - M_{1,1}) = \sum_{j\in[2,\ell]} (M_{1,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}_{1,j}\hat{y}_j \tag{4}$$

If $M_{1,1} \neq 0$, we rewrite this as:

$$y_1^* = \sum_{j\in[2,\ell]} \frac{M_{1,j}}{(1-M_{1,1})} y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \frac{\mathfrak{p}_{1,j}}{(1-M_{1,1})}\hat{y}_j \tag{5}$$

At the end of this manipulation, we have an equivalent system of equations represented by matrix M', such that $M'_{1,1} = 0$ and for all $j \in [\ell]$, $M'_{1,j} = \frac{M_{1,j}}{(1-M_{1,1})}$. In shorthand, we denote this by $M_{1,1} \to 0, M_{1,j} \to \frac{M_{1,j}}{(1-M_{1,1})}$ for $j \in [2,\ell]$.

Inductive Hypothesis. Assume that after the k^{th} transformation, all entries $M_{i,j} = 0$ for $(i,j) \in [1,k] \times [1,k]$. This gives us, that for $i' \in [1,k]$, the probability distribution over other inputs for mimicking inputs $y_{i'}^*$ are of the form:

$$y_{i'}^* = \sum_{j\in[k+1,\ell]} (M_{k+1,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}_{k+1,j}\hat{y}_j \tag{6}$$

Induction Step. This consists of the following two transformations:

1. The probability distribution over other inputs for mimicking the input y_{k+1}^* can be written as:

$$y_{k+1}^* = \sum_{j\in[k]} (M_{k+1,j}) y_j^* + \sum_{j\in[k+1,\ell]} (M_{k+1,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}_{k+1,j}\hat{y}_j \tag{7}$$

   Then, it is possible to substitute the first k terms in this equation using Eq. 6 to obtain another equation of the form:

$$y_{k+1}^* = \sum_{j\in[k+1,\ell]} (M'_{k+1,j}) y_j^* + \sum_{j\in[|\mathcal{Y}|-\ell]} \mathfrak{p}'_{k+1,j}\hat{y}_j, \tag{8}$$

for suitably modified values $(M'_{k+1,j})$ and $\mathfrak{p}'_{k+1,j}$.
At the end of this set of transformations, for all $j \in [k], M_{k+1,j} \to 0$ and for $j \in [k+1, \ell], M_{k+1,j} \to M'_{k+1,j}$.

2. Now, we can write the $(k+1)^{th}$ row Eq. 8 as:

$$y^*_{k+1} = (M'_{k+1,k+1})y^*_{k+1} + \sum_{j \in [k+2,\ell]} (M'_{k+1,j})y^*_j + \sum_{j \in [|\mathcal{Y}|-\ell]} \mathfrak{p}'_{k+1,j}\hat{y}_j \quad (9)$$

If $M_{k+1,k+1} \neq 0$, this can be rewritten as:

$$y^*_{k+1} = \sum_{j \in [k+2,\ell]} \frac{M'_{k+1,j}}{(1 - M'_{k+1,k+1})} y^*_j + \sum_{j \in [|\mathcal{Y}|-\ell]} \frac{\mathfrak{p}_{k+1,j}}{(1 - M'_{k+1,k+1})} \hat{y}_j \quad (10)$$

At the end of this transformation, the matrix entry $M'_{k+1,k+1} \to 0$.

3. Substituting Eq. 10 into the first k rows, we get that for $i' \in [1, \kappa+1]$, the probability distribution over other inputs for mimicking inputs $y^*_{i'}$ are of the form:

$$y^*_{i'} = \sum_{j \in [k+1,\ell]} (M''_{k+1,j})y^*_j + \sum_{j \in [|\mathcal{Y}|-\ell]} \mathfrak{p}''_{k+1,j}\hat{y}_j \quad (11)$$

At the end of these transformations, we obtain an matrix $\bar{M}$ representing an equivalent system of equations, such that for all $(i,j) \in [\ell] \times [\ell]$, $\bar{M}_{i,j} = 0$ and $\bar{M}_{i,\ell+1} \neq 0$. This completes the proof of this lemma.

Now, suppose that for all inputs $y^* \in Y \backslash \hat{Y}$, Bob can mimic y^* using non-zero weight in $\hat{Y}$. Then, since Lemma 3 proves that all inputs $y^* \in Y \backslash \hat{Y}$ can be written as a convex linear combination of inputs entirely in $\hat{Y}$. This contradicts Imported Lemma 4. Since the functionalities we study only have a constant-sized domain, it is always easy to find such an input y^*.

6.1 Construction

The protocol is described in Fig. 7. Without loss of generality, we can assume that there exists a commitment protocol from Alice to Bob. We construct a commitment protocol with Bob as sender, and Alice as receiver.

6.2 Proof of Security

Receiver Security (Statistical Binding/Extractability). In the UC setting, it suffices to consider a dummy sender $\mathcal{S}$ and malicious environment $\mathcal{Z}_{\mathcal{S}}$, such that the dummy sender forwards all messages from $\mathcal{Z}_{\mathcal{S}}$ to the honest receiver/simulator, and vice-versa. Without loss of generality, the malicious simulation strategy $\mathsf{Sim}_{\mathcal{S}}$ can be viewed to interact directly with $\mathcal{Z}_{\mathcal{S}}$. $\mathsf{Sim}_{\mathcal{S}}$ is described in Fig. 8.

Lemma 4. *There exists a constant c such that the simulation error for the malicious sender is at most $2^{-c\kappa}$.*

Inputs: Sender $\mathcal{S}$ has input bit $\mathsf{bit} \in \{0,1\}$ and receiver $\mathcal{R}$ has no input.
Hybrid: $\mathcal{F}$ for non-simple function $\mathcal{F}$ where $\hat{Y}$ is not confusable, and y^* as defined above, exists. $\mathcal{F}$ provides commitments (Com) from Alice to Bob.
The protocol is presented in terms of a $(\kappa', \kappa' - \kappa'^{8/9}, \omega(\kappa^{15/16}))$-linear code $\mathcal{C}$ over the binary alphabet. (An explicit code is not necessary: the receiver can pick random $\omega(\kappa^{7/8})$ "parity checks" to construct the code and announce it to the sender.) The protocol is parameterized by κ.

1. Commit Phase:
   (a) For all $i \in [\kappa]$, $\mathcal{R}$ (Alice) picks inputs $(X_1, X_2, \ldots, X_{2\kappa^2})$ from a uniform distribution over $\mathcal{X}^{2\kappa^2}$. Alice also sends $\mathsf{Com}(X_1), \mathsf{Com}(X_2), \ldots \mathsf{Com}(X_{2\kappa^2})$ to Bob.
   (b) $\mathcal{S}$ (Bob) picks inputs $(Y_1, Y_2, \ldots, Y_{2\kappa^2})$ from a uniform distribution over $(\hat{Y} \cup y^*)^{2\kappa^2}$. $\mathcal{R}$ and $\mathcal{S}$ invoke $\mathcal{F}$, 2κ times, with inputs $(X_1, X_2, \ldots, X_{2\kappa^2})$ and $(Y_1, Y_2, \ldots, Y_{2\kappa^2})$ respectively.
   (c) Cut-and-Choose: $\mathcal{R}$ picks $r_1 \xleftarrow{\$} \{0,1\}^*$ and sends $\mathsf{Com}(r)$ to $\mathcal{S}$. $\mathcal{S}$ sends $r_2 \xleftarrow{\$} \{0,1\}^*$ to $\mathcal{R}$. $\mathcal{R}$ uses $r_1 \oplus r_2$ to pick a subset $I \xleftarrow{\$} \binom{2\kappa^2}{[\kappa^2]}$ of the κ^2 indices. $\mathcal{R}$ decommits to r_1. Furthermore, for all $i \in I$, $\mathcal{R}$ decommits to input X_i and also opens her view (X_i, W_i).
   $\mathcal{S}$ aborts if the decommitments are not correct, or the inputs of $\mathcal{R}$ are not close to a uniform distribution, or if (X_i, W_i) for $i \in I$ satisfy the consistency checks in the Left-Statistical-Tests.
   (d) $\mathcal{S}$ does the following for all $i \in [\kappa]$.
      – Construct the j^{th} characteristic vector $\mathbf{u}_j$ such that for all $i \in [\kappa]$, $u_{j,i} = 0$ if and only if $Y_{j\kappa+i} \in \hat{Y}$, else $u_{j,i} = 1$.
      – Pick κ random codewords $\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_\kappa \in \mathcal{C}^\kappa$. Pick $h \xleftarrow{\$} \mathcal{H}$, a universal hash function mapping $\{0,1\}^{\kappa^2} \to \{0,1\}$, and for $j \in [\kappa]$, compute $y = h(\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_\kappa) \oplus \mathsf{bit}, \mathsf{offset}_j = (\mathbf{c}_j \oplus \mathbf{u}_j)$. Send $(h, y, \mathsf{offset}_1, \mathsf{offset}_2, \ldots \mathsf{offset}_\kappa)$ to $\mathcal{R}$.
2. Reveal Phase:
   (a) $\mathcal{S}$ sets $b' = b, \mathbf{u}' = \mathbf{u}$ and sends b', u' to $\mathcal{R}$ as his opening. $\mathcal{S}$ also sends (Y_i, Z_i) for all $i \in [\kappa^2]$, to $\mathcal{R}$.
   (b) $\mathcal{R}$ accepts if all the following conditions hold:
      – For $j \in [\kappa]$, $\mathbf{c}_j = \mathbf{u}'_j \oplus \mathsf{offset}_j$, is a valid codeword.
      – $b' = h(\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_\kappa) \oplus y$.
      – Input-output frequency tests on (Y_i, Z_i) pass for all $i \in [\kappa^2]$.

Fig. 7. $\mathcal{F}_{\mathsf{com}}$ in Case 2.

Proof. The simulator performs Steps $1(a), (b)$ and (c) as per the honest receiver strategy, and also emulates the functionality $\mathcal{F}$ honestly for the sender. It remains to show that the unique bit b' extracted by the simulator equals the bit b committed by the sender Bob. The crux of this proof relies on the fact that the

The simulator $\mathsf{Sim}_{\mathcal{S}}$ does the following.

1. Commit Phase:
   (a) $\mathsf{Sim}_{\mathcal{S}}$ picks inputs $(X_1, X_2, \ldots, X_{2\kappa^2})$ uniformly from $\mathcal{X}^{2\kappa^2}$. $\mathsf{Sim}_{\mathcal{S}}$ then commits to each of them using fresh randomness and sends $\mathsf{Com}(X_1), \mathsf{Com}(X_2), \ldots \mathsf{Com}(X_{2\kappa^2})$ to $\mathcal{S}$. Note that $\mathsf{Sim}_{\mathcal{S}}$ has the capability to equivocate these commitments.
   (b) $\mathsf{Sim}_{\mathcal{S}}$ obtains inputs $(Y_1, Y_2, \ldots Y_{2\kappa^2})$ from $\mathcal{S}$ and emulates the functionality $\mathcal{F}$ honestly for $\mathcal{S}$ with inputs $(X_1, X_2, \ldots X_{2\kappa^2})$ and $(Y_1, Y_2, \ldots Y_{2\kappa^2})$.
   (c) Cut-and-Choose: $\mathsf{Sim}_{\mathcal{S}}$ picks $r_1 \xleftarrow{\$} \{0,1\}^*$ and sends $\mathsf{com}_1 = \mathsf{Com}(r_1)$ to $\mathcal{S}$. $\mathcal{S}$ sends $r_2 \xleftarrow{\$} \{0,1\}^*$ to $\mathsf{Sim}_{\mathcal{S}}$. $\mathsf{Sim}_{\mathcal{S}}$ uses $(r'_1 \oplus r_2)$ to pick subset $I \xleftarrow{\$} \binom{2\kappa^2}{[\kappa^2]}$ of the κ^2 indices. $\mathsf{Sim}_{\mathcal{S}}$ decommits com_1 to r_1 and, for all $i \in I$, $\mathsf{Sim}_{\mathcal{S}}$ decommits to input X_i and also opens the view (X_i, W_i). Set $S = [2\kappa^2] \setminus I$ and reorder the indices in S to $[\kappa^2]$.
   (d) $\mathsf{Sim}_{\mathcal{S}}$ obtains $(h, y, \mathsf{offset}_j)$ for $j \in [\kappa]$ from $\mathcal{S}$. It constructs characteristic vectors $\mathbf{u}_j$ such that for all $i \in \mathcal{S}$, $u_{j,i} = 0$ if and only if $Y_{j\kappa+i} \in \hat{Y}$, else $u_i = 1$. It then computes $\tilde{c}_j = \mathbf{u}_j \oplus \mathsf{offset}_j$, sets c'_j to be the nearest codeword[a] to $\tilde{c}_j$, and sets bit $b' = y \oplus h(c'_1, c'_2, \ldots c'_\kappa)$.
2. Reveal Phase:
   (a) Obtain $b', u'_1, u'_2, \ldots u'_\kappa, (Y_i, Z_i)$ for all $i \in [\kappa^2]$ from $\mathcal{S}$ as his opening.
   (b) Allow the ideal functionality to output the extracted bit b' if all the following conditions hold (and otherwise reject):
      – $(u'_j \oplus \mathsf{offset}_j)$ is a valid codeword for $j \in [\kappa]$.
      – (Y_i, Z_i) for all $i \in [\kappa^2]$ satisfy the input-output frequency tests in the Right-Statistical-Tests.

[a] If the nearest codeword is not unique, then $\mathsf{Sim}_{\mathcal{S}}$ commits to an arbitrary bit.

Fig. 8. Sender simulation strategy in Case 2.

protocol requires the sender to use all his extreme views, and some non-extreme views; and on the minimum distance of the code used.

Bob cannot claim non-extreme views to be extreme. Equivalently, Bob cannot claim an input outside $\hat{Y}_B$ to be an input inside $\hat{Y}_B$. In the opening made by Bob, consider the positions where Bob claimed his view to be $(y_i, z_i) = (y^*, z^*) \in \mathcal{V}_B$, such that the equivalence class of this view $\phi_B(y^*, z^*) = \Phi$. Consider the fraction of these positions where the actual view of Bob (x', w') such that $\phi_B(y', z') \neq \Phi$.

In these positions, the expected view of Alice is given by a linear combination of the columns $\beta^{\mathcal{F}}|_{(y',z')}$ (with coordinates scaled appropriately). If this linear combination is not close to the vector $\beta^{\mathcal{F}}|_{(y^*,z^*)}$ (scaled appropriately) then with all but negligible probability, the opening will not be accepted by the receiver. On the other hand, if the linear combination is close to $\beta^{\mathcal{F}}|_{(y^*,z^*)}$, since $\beta^{\mathcal{F}}|_{(y^*,z^*)}$

is outside the linear span of other $\beta^{\mathcal{F}}|_{(y',z')}$ with $\phi_B(y'z') \neq \phi_B(y^*, z^*)$, only at a small number (sub-linear fraction, say $\kappa^{2/3}$) of places can Bob open to (y^*, z^*) but have had an actual view (y', z'). Thus, extreme views cannot be claimed to be obtained as a result of using inputs which exclusively yield non-extreme views.

Bob cannot claim an input inside $\widehat{Y}_B$ to be outside $\widehat{Y}_B$. By Lemma 3, we also know that y^* cannot be mimicked with any non-zero weight in $\hat{Y}$, without getting caught by the receiver in the Right-Statistical-Tests. Thus, it is not possible to use inputs in $\hat{Y}$ and equivocate them to y^*. This gives that the sender cannot equivocate at more that $O(\kappa^{2/3})$ indices.

Bob cannot equivocate. By using a code such that the minimum distance of the code ($\Omega(\kappa^{3/4})$) is much larger than the number of positions where the sender can cheat in one of the two situations above ($O(\kappa^{2/3})$, we guarantee that the sender is bound to his committed bit.

Specifically, the simulator computes the nearest codeword to the codeword extracted from the sender, and uses this to extract his committed bit. The sender cannot equivocate this codeword without cheating in $\Omega(\kappa^{3/4})$ views, and if he does so, his decommitment is not accepted except with probability at least $(1 - 2^{-c\kappa})$. This completes the proof of this lemma.

Sender Security (Statistical Hiding/Equivocability). It suffices to consider a dummy receiver $\mathcal{R}$ and malicious environment $\mathcal{Z}_{\mathcal{R}}$, such that the dummy receiver forwards all messages from $\mathcal{Z}_{\mathcal{R}}$ to the honest sender/simulator, and vice-versa. Without loss of generality, the malicious simulation strategy $\mathsf{Sim}_{\mathcal{R}}$ can be viewed to interact directly with $\mathcal{Z}_{\mathcal{R}}$. $\mathsf{Sim}_{\mathcal{R}}$ is described in Fig. 9.

Lemma 5. *There exists a constant c such that the simulation error for the malicious receiver is at most $2^{-c\kappa}$.*

Proof. Consider the use of the function f as a "channel", which accepts $x_{i,j}$ from Alice, $c_{i,j}$ from Bob, samples $(y_{i,j}, w_{i,j}, z_{i,j})$ and outputs $z_{i,j}$ to Bob, and $a_{i,j} \oplus c_{i,j}$ to Alice where $a_{i,j} = \phi_B(y_{i,j}, z_{i,j})$.

The cut-and-choose verification in Step 1(c) ensures that Alice uses (close to) a uniform distribution over her inputs. Then, she obtains the view (x, w) that intersects with an extreme and a non-extreme view in $\widehat{V}_B$ in at least a constant fraction of the invocations. At all these invocations, given her view, Alice has confusion about whether the corresponding view of Bob was extreme of non-extreme. Formally, we can show that the capacity of the above channel is a constant, that is less than 1.

Then we appeal to an extension of the weak converse of Shannon's Channel Coding Theorem (Imported Lemma 2) to argue that since the code has rate 1, Alice errs in decoding each codeword with at least a constant probability. We need this extension of the (weak) converse of the channel coding theorem to handle that the facts that:

The simulator $\mathsf{Sim}_{\mathcal{R}}$ does the following.

1. Commit Phase:
   (a) $\mathsf{Sim}_{\mathcal{R}}$ obtains commitments $c_1, c_2, \ldots c_{2\kappa^2}$ from $\mathcal{R}$.
   (b) $\mathsf{Sim}_{\mathcal{R}}$ obtains inputs $(X_1, X_2, \ldots X_{2\kappa^2})$ from $\mathcal{R}$ and emulates the functionality $\mathcal{F}$ honestly for $\mathcal{R}$ with inputs $(X_1, X_2, \ldots X_{2\kappa^2})$ and $(Y_1, Y_2, \ldots Y_{2\kappa^2})$.
   (c) Cut-and-Choose: $\mathsf{Sim}_{\mathcal{R}}$ obtains com_1 from $\mathcal{R}$. $\mathsf{Sim}_{\mathcal{R}}$ sends $r_2 \xleftarrow{\$} \{0,1\}^*$ to $\mathcal{R}$. $\mathcal{R}$ decommits to r_1 and sends subset $I \xleftarrow{\$} \binom{2\kappa^2}{[\kappa^2]}$ of the κ^2 indices. For all $i \in I$, $\mathsf{Sim}_{\mathcal{R}}$ obtains decommitments X_i and also the openings (X_i, W_i). $\mathsf{Sim}_{\mathcal{R}}$ aborts if the decommitments are not correct, or the inputs of $\mathcal{R}$ are not from a uniform distribution, or if (X_i, W_i) for $i \in I$ do not satisfy the consistency checks in Left-Statistical-Tests.
   (d) $\mathsf{Sim}_{\mathcal{R}}$ follows honest sender strategy to commit to a uniformly random bit $\mathsf{bit}' \xleftarrow{\$} \{0,1\}^*$.

Fig. 9. Receiver simulation strategy in Case 2.

1. The receiver can adaptively choose the channel characteristic, by picking $y_{i,j}$ adaptively, and
2. Some of the channel characteristics that can be chosen include a noiseless channel, but the number of times such a characteristic can be used cannot be large (except with negligible probability). The reason this restriction can be enforced is because Alice's view intersects with views of Bob corresponding to characteristic index 0 and 1.

Then, applying the Leftover Hash Lemma, we get that for a universal hash function h, if Bob sends κ codewords over such a channel, the output of the hash function is at least $1 - 2^{-c\kappa}$ close to uniform. Thus, the simulation error is at most $2^{-c\kappa}$.

References

1. Beimel, A., Malkin, T., Micali, S.: The all-or-nothing nature of two-party secure computation. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 80–97. Springer, Heidelberg (1999)
2. Brassard, G., Crépeau, C., Robert, J.M.: Information theoretic reductions among disclosure problems. In: 27th Annual Symposium on Foundations of Computer Science, Toronto, Ontario, Canada, 27–29 October 1986, pp. 168–173. IEEE Computer Society Press (1986)
3. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**(1), 143–202 (2000)

4. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th Annual ACM Symposium on Theory of Computing, Montréal, Québec, Canada, 19–21 May 2002, pp. 494–503. ACM Press (2002)
5. Crépeau, C., Kilian, J.: Achieving oblivious transfer using weakened security assumptions (extended abstract). In: 29th Annual Symposium on Foundations of Computer Science, White Plains, New York, 24–26 October 1988, pp. 42–52. IEEE Computer Society Press (1988)
6. Crépeau, C., Morozov, K., Wolf, S.: Efficient unconditional oblivious transfer from almost any noisy channel. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 47–59. Springer, Heidelberg (2005)
7. Crépeau, C., Sántha, M.: On the reversibility of oblivious transfer. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 106–113. Springer, Heidelberg (1991)
8. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008). http://dx.org/10.1137/060651380
9. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th Annual ACM Symposium on Theory of Computing, New York City, New York, USA, 25–27 May 1987, pp. 218–229 (1987)
10. Goldreich, O., Vainish, R.: How to solve any protocol problem - an efficiency improvement. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 73–86. Springer, Heidelberg (1988)
11. Haber, S., Micali, S.: Unpublished manuscript (1986)
12. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer - efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (2008)
13. Kilian, J.: Founding cryptography on oblivious transfer. In: 20th Annual ACM Symposium on Theory of Computing, Chicago, Illinois, USA, 2–4 May 1988, pp. 20–31. ACM Press (1988)
14. Kilian, J.: A general completeness theorem for two-party games. In: 23rd Annual ACM Symposium on Theory of Computing, New Orleans, Louisiana, USA, 6–8 May 1991, pp. 553–560. ACM Press (1991)
15. Kilian, J.: More general completeness theorems for secure two-party computation. In: 32nd Annual ACM Symposium on Theory of Computing, Portland, Oregon, USA, 21–23 May 2000, pp. 316–324. ACM Press (2000)
16. Kraschewski, D., Maji, H.K., Prabhakaran, M., Sahai, A.: A full characterization of completeness for two-party randomized function evaluation. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 659–676. Springer, Heidelberg (2014)
17. Kraschewski, D., Müller-Quade, J.: Completeness theorems with constructive proofs for finite deterministic 2-party functions. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 364–381. Springer, Heidelberg (2011)
18. Lindell, Y.: Adaptively secure two-party computation with erasures. Cryptology ePrint Archive, Report 2009/031 (2009). http://eprint.iacr.org/2009/031
19. Maji, H.K., Prabhakaran, M., Rosulek, M.: A unified characterization of completeness and triviality for secure function evaluation. In: Galbraith, S.D., Nandi, M. (eds.) INDOCRYPT 2012. LNCS, vol. 7668, pp. 40–59. Springer, Heidelberg (2012)
20. Rabin, M.: How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory (1981)

21. Wiesner, S.: Conjugate coding. SIGACT News **15**, 78–88. http://doi.acm.org/10.1145/1008908.1008920
22. Wolf, S., Wullschleger, J.: Oblivious transfer is symmetric. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 222–232. Springer, Heidelberg (2006)
23. Yao, A.C.C.: Protocols for secure computations (extended abstract). In: 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, 3–5 November 1982, pp. 160–164. IEEE Computer Society Press (1982)