A Full Characterization of Completeness for Two-party Randomized Function Evaluation

Daniel Kraschewski[1], Hemanta K. Maji[2], Manoj Prabhakaran[3], and Amit Sahai[4]

[1] Technion, Haifa, Israel.
[2] Los Angeles, USA.
[3] Univ. of Illinois, Urbana-Champaign, USA.
[4] Univ. of California, Los Angeles, USA.

Abstract. We settle a long standing open problem which has pursued a full characterization of completeness of (potentially randomized) finite functions for 2-party computation that is secure against active adversaries. Since the first such complete function was discovered [Kilian, FOCS 1988], the question of which finite 2-party functions are complete has been studied extensively, leading to characterization in many special cases. In this work, we completely settle this problem.

We provide a polynomial time algorithm to test whether a 2-party finite secure function evaluation (SFE) functionality (possibly randomized) is complete or not. The main tools in our solution include:

- A formal linear algebraic notion of *redundancy* in a general 2-party randomized function.
- A notion of *statistically testable games*. A kind of interactive proof in the information-theoretic setting where *both* parties are computationally unbounded but differ in their knowledge of a secret.
- An extension of the (weak) *converse of Shannon's channel coding theorem*, where an adversary can adaptively choose the channel based on its view.

We show that any function f, if complete, can implement any (randomized) circuit C using only $O(|C| + \kappa)$ calls to f, where κ is the statistical security parameter. In particular, for any two-party functionality g, this establishes a universal notion of its quantitative "cryptographic complexity" independent of the setup and has close connections to circuit complexity.

1 Introduction

Understanding the *complexity* of functions is central to theoretical computer science. While the most studied notion of complexity in this literature is that of computational complexity, there have also been other important aspects explored, most notably, *communication complexity* [35]. Another aspect of complexity of a (distributed) function is its *cryptographic complexity*, which seeks to understand the cryptographic utility of a function, stemming from how it hides and reveals information. While it is only recently that the term has been

explicitly used, cryptographic complexity theory has been vigorously pursued at least since Kilian introduced the notion of *completeness* of cryptographic primitives [23].

Completeness (of functions with finite domains) has been the first and most important question of cryptographic complexity: what properties of a function let all other cryptographic tasks (in the context of secure computation) be *reduced* to it. This question has been asked and answered several times [23, 11, 24, 25, 12, 28, 31] each time for a different class of functions, or restricted to different kinds of reductions (see Fig. 1 for a summary of the state of the art). These works produced several exciting ideas and advances, and brought together concepts from different fields. For instance, [25] used the Nash equilibrium in a zero-sum game defined using the function to obtain a secure protocol; earlier [11] identified the binary symmetric channel (noisy channel) as a complete function, paving the way to a fruitful and successful connection with information-theory literature.

However, these works left open what is arguably the hardest part of the characterization: completeness of *randomized* functions with finite domain under reductions that are secure against an *active* adversary (see Fig. 1). Indeed, even with a (usually simplifying) restriction that only one of the two parties receives an output from the function, it was not known which *randomized* functions are complete. In this work, we finally provide a full characterization of completeness of general[5] 2-party functions with finite domains. This work brings to close this rich line of investigation, but also introduces several new ideas and notions, and poses new questions regarding cryptographic complexity.

Prior to our work, the only completeness results known for randomized functions against active adversaries were for the very restricted case of channels [12], *i.e.* randomized functions that *only take input from one party*, and deliver the output to the other. Thus, in particular, before our work, no completeness characterization results against active adversaries were known for any randomized function classes that take input from both parties.

Also, along the way to our main construction, we generalize a result in another line of work, on black-box protocol constructions [20, 19, 22, 9]. We give a black-box transformation from a passive-secure OT protocol *in a hybrid setting* (wherein the protocol has access to an ideal functionality) to a UC-secure OT protocol in the same hybrid setting, with access to the commitment functionality.[6] Our transformation relativizes with respect to any ideal functionality, as long as that functionality is "redundancy free" (see later). Though our focus is on information-theoretic security, we note that by considering ideal functionali-

[5] By a general function, we mean one without any restrictions on which parties have inputs and which parties have outputs. Earlier work on characterizing randomized functions considered only "symmetric" (both parties get same output) and "asymmetric" (only one party gets any output) functions. Beyond this, only specific examples were known, like correlated random variables considered by Beaver [1].

[6] It is interesting to note that, unlike in many other settings, a black-box transformation in the plain model does not imply a transformation in a hybrid model. That is, there is no analogue of universal composition for *black-box protocol compilation*.

ties that are *not* information-theoretically complete, our transformation implies black-box equivalence of related *computational assumptions*.

	Passive Completeness	Active Completeness
Deterministic	Symmetric: [24]-1991 Asymmetric: [3]-1999 General: [28]-2011	Symmetric: [24]-1991 Asymmetric: [25]-2000 General: [28]-2011
Randomized	Symmetric: [25]-2000 Asymmetric: [25]-2000 General: [31]-2012	Channels: [12]-2004 Symmetric/Asymmetric/General: **Open** *Settled in this paper*

Fig. 1. Summary of Completeness Characterization Results.

Finally, our tools for analysis are novel in this line of work. In particular, we introduce the notion of **statistically testable games**, which is a kind of interactive proof in the information-theoretic setting where *both* parties can be computationally unbounded, but differ in their knowledge of some secret. We discuss these in more detail in Section 1.3 and in subsequent sections.

We also formulate and prove a new converse of Shannon's Channel Coding theorem to obtain a hiding property from a "channel." This is perhaps an unusual (but in hindsight, natural) use of a converse of the channel coding theorem, which was originally used to establish the optimality of the channel coding theorem.

1.1 Our Results

We provide the first *algorithmic* characterization of all finite 2-party (potentially randomized) functions that are complete for secure function evaluation against active adversaries: Namely, our results provide the first explicit algorithm (see Fig. 2 for an abridged version; the full figure is provided in the full version of the paper [27]) that can analyze any given (randomized) function f, and output whether or not f is complete against active adversaries.

Input: A 2-party randomized SFE f, given as a matrix $\mathfrak{P}^f$ of conditional probabilities $\mathfrak{p}^f[w, z|x, y]$.
Output: Whether f is UC-complete or not.

1. Compute a core $\widehat{f}$ of f.
2. Check if $\widehat{f}$ is simple or not (using combinatorial characterization in [31].
3. If $\widehat{f}$ is simple, then f is not complete.
4. Else (i.e., $\widehat{f}$ is not simple), f is complete.

Fig. 2. This algorithm tests whether a function f is UC-complete or not. More detailed version is provided in [27].

The algorithm has two steps: finding what we call the "*core*" of a given function f and then checking if it is "*simple*" or not. A function f is complete if and only if its core is not simple.

We now provide a high-level intuitive explanation of our algorithmic characterization works, by considering some easy and well-known examples. This will help in understanding our exact characterization, which is somewhat more involved since it covers general randomized functions.

The *core* of f is computed by removing "redundant" parts of the function f. To develop some intuition for this, consider the one-sided OR function which takes two bits from Alice and Bob and outputs the logical OR of these two bits to only Bob. This function is *not* complete against active adversaries, and in fact is trivial: the reason is that a corrupt Bob can always choose his input to be "0" – and by doing so, it can always learn Alice's input, without Alice detecting this. (Thus, even a trivial protocol in which Alice sends her bit to Bob is indeed secure against active adversaries, since if Bob is corrupt, he could have learned Alice's input even in the ideal world.) Because of this, we say that Bob's input "1" is redundant from the adversary's point of view: the adversary is always better off using the input "0".

When extended to the setting of randomized functions, redundancy becomes more subtle. For instance, an input can become redundant because instead of using that input, an adversary could use a *distribution* over other inputs, without being detected. Another form of redundancy that appears for randomized functions is that of redundant outputs (for the same input). As an example, suppose in the above example, 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 the symbol β with probability $1/4$. Here, we observe that the two outcomes α and β give Bob the same information about Alice's input, and could be merged into a single outcome. More generally, if two possible outputs that the adversary can obtain for the same input have identical conditional distributions for the other party's input-output pair, then the distinction between these two output values is redundant.

We provide a novel formal definition of redundancy that fully captures both these forms of redundancy: (1) it identifies inputs that are useless for the adversary; and (2) it identifies 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. While the above intuition is useful, it is not exactly the motivation behind our formal definition. The formal definition balances the following two requirements on redundancy:

- Adding or removing redundancy does not change a function's complexity (as far as security against active corruption alone is concerned): in particular, f is complete if and only if its core is complete.
- A redundancy free function removes the possibility for a party to freely deviate from its interaction with a functionality without the rest of the system (the environment and the other party) detecting any difference.

The formal definition (based on Equation 1) is linear algebraic, inspired by simulatability considerations, and seemingly more general; but as will be discussed in Section 1.3 and later, this definition coincides with exactly the above two forms of redundancies. An explicit algorithm for removing redundancy and finding the "core" is given in the full version of the paper [27].

The second phase of our algorithm determines whether the core of f is *simple*, a notion defined earlier by [31] generalizing Kilian's condition for passive completeness [25]. Informally, a function g is simple if it preserves the independence of views. To develop intuition for this, consider a common randomness function that ignores the inputs of the two parties and simply outputs a uniform independent random bit to both parties. This function is intuitively useless because, at least in the passive-security setting, this function can be trivially realized by one party sampling this bit, and sending it to the other party. 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 function evaluation, the views of the two players remain independent of each other (see the full version [27] for details). A natural explicit algorithm for determining whether a function is simple was already given by [31], which we use here.

Beyond the basic feasibility result, we also show that secure evaluation of any finite function g to a complete finite function f can be carried out, asymptotically, at "constant rate." That is, n copies of g can be evaluated with access to $O(n+\kappa)$ copies of f, and in fact, only $O(n+\kappa)$ communication, overall. Here κ is a statistical security parameter; that is, the error in security (simulation error) is negligible in κ. In fact, the total amount of communication in the protocol (including the interaction with copies of f) is also bounded by $O(n+\kappa)$. This leads to our main theorem:

Theorem 1. *A finite 2-party function is UC-complete (or equivalently, standalone-complete) against active adversaries if and only if its core is not simple. Further, if f is such a function, n copies of any finite 2-party function can be securely evaluated by a protocol in f-hybrid with communication complexity $O(n+\kappa)$, where κ is the security parameter.*

Connections to Circuit Complexity. An interesting measure of complexity of a function g (modeled as a 2-party function) is its "OT complexity" – the number of (1 out of 2, bit) OT instances needed for securely evaluating it.[7] As sketched below, the OT complexity of a function is closely related to its circuit complexity and may provide an approach to proving explicit circuit lowerbounds. Our results show that instead of OT complexity, one could consider f-complexity, for any f whose core is not simple. *This establishes "cryptographic complexity" as a fundamental complexity measure of (2-party) functions, independent of which complete finite 2-party function is used to securely realize it, just the same way*

[7] One may also define OT complexity to be the total amount of communication (possibly amortized) needed for securely evaluating g, in the OT-hybrid model.

circuit complexity is independent of which specific set of universal finite gates are used to implement it.

Circuit complexity and OT complexity are closely related to each other as follows. By a simple protocol due to [16–18], we know that the OT complexity of a function g (defined with respect to passive security) is $O(C(g))$, where $C(g)$ stands for the circuit complexity of g. This means that a super-linear lowerbound for OT complexity of g gives a super-linear lowerbound on $C(g)$. Of course, this only shows that it is a hard problem to lowerbound OT complexity. But interestingly, this connection does open up a new direction of approaching circuit complexity lowerbounds: the fact that most functions have exponential circuit complexity is an easy consequence of a counting argument due to Shannon; but *for OT complexity, even such an existential lowerbound is not known.* Resolving this could be an easier problem than finding explicit circuit lowerbounds, yet could lead to new insights to proving explicit OT complexity and circuit complexity lowerbounds.

The same argument applies for OT complexity defined with respect to active adversaries as well, due to the result of [22]. Note that it would be easier to lowerbound OT complexity when it is defined this way, than when defined with respect to passive adversaries. The relevance of our result is that instead of OT, one can consider any 2-party function f whose core is not simple. As we show that OT can be reduced to any such function at a constant rate, a super-linear lowerbound on (amortized) f-complexity will indeed translate to a super-linear lowerbound on circuit complexity. We discuss this more in the full version of our paper [27] and leave it as an important direction to study. Recently Beimel et al. [2] have shown that the OT-complexity of random functions is significantly lower than their (AND) circuit complexity, but still exponential in the input length, in the worst case.

1.2 Related Work

We briefly summarize the results on completeness from prior work (also refer to Fig. 1). The function oblivious transfer (OT) was identified independently by Wiesner and Rabin [32, 34]. Brassard et al. [5] showed that various flavors of OT can be reduced to each other with respect to security against active adversaries. In a seminal work, Kilian identified OT as the first active-complete function [23]. Prior to this Goldreich and Vainish, and independently Micali and Haber, showed that OT is passive-complete [18, 17]. Crépeau and Kilian then showed that the noisy channel is also active-complete [11]. The first characterization of completeness appeared in [24] where it was shown that among deterministic "symmetric" functions (in which both parties get the same output) a function f is active-complete if and only if there is an "OR minor" in the matrix representing f. Beimel, Malkin and Micali showed that among "asymmetric" functions (in which only one party gets the output), a function is passive-complete if and only if it is not "trivial" [3]. ([3] also concerned itself with the computational setting and asked cryptographic complexity questions regarding computational assumptions.) Kilian vastly generalized this by giving several

completeness characterizations: active-complete deterministic asymmetric functions, passive-complete symmetric functions and passive-complete asymmetric functions [25]. Kilian's result for active-completeness was extended in two different directions by subsequent work: Crépeau, Morozov and Wolf [12] considered "channel functions" which are randomized asymmetric functions (only one party has output), but with the additional restriction that only one party has input; Kraschewski and Müller-Quade [28] considered functions in which both parties can have inputs and outputs, but restricted to deterministic functions.

Kilian's result for passive-completeness was extended to all functions in a recent work [31], which also presented a unification of all the prior characterizations and posed the question of completing the characterization. The full characterization we obtain matches the unified conjecture from [31].

A related, but different line of work investigated secure computability and completeness for *multi-party* computation (with more than 2 parties) (e.g., [8, 4, 33, 29, 26, 14, 13]). We restrict ourselves to 2-party functions in this work. Another direction of research considers whether a short protocol for f (instead of a black-box implementing f) is complete or not [30].

1.3 Technical Overview

An important ingredient of our result is a combinatorial/linear-algebraic characterization of "redundancy" in a general 2-party function. The importance of redundancy is two fold:

- Any function f is "equivalent" (or *weakly isomorphic*, as defined in [31]) to a "core" function $\widehat{f}$ which is redundancy free, so that f is complete against active adversaries if and only if $\widehat{f}$ is. Thus it is enough to characterize completeness for redundancy free functions.
- Our various protocols rely on being given access to a redundancy free function. Redundancy makes it possible for an adversary to deviate from a prescribed interaction with a function without any chance of being detected. Thus the statistical checks used to enforce that the adversary does not deviate from its behavior crucially rely on the protocol using only redundancy free functions.

While redundancy of special classes of 2-party functions have appeared in the literature previously, it turns out that for general 2-party functions, the nature of redundancy is significantly more intricate. Recall that we discussed redundancy informally by considering an adversary that tries to learn about the other party's input-output pair: any input it can avoid, and distinction between outputs (for the same input) that provide it with identical information are both redundant. However, the role of redundancy in showing completeness is somewhat different: redundancy in a function makes it hard (if not impossible) to use it in a protocol, as it allows an active adversary to *deviate* from behavior prescribed by a protocol, with no chance of being caught. Possible deviation includes replacing its prescribed input to the function by a probabilistically chosen input, *and*

probabilistically altering the output it receives from the function before using it in the protocol, *at the same time*. The goal of this deviation is to minimize detectability by the other party (and the environment). Our formal definition of redundancy uses this point of view. We define *irredundancy* quantitatively (Definition 1) as a lowerbound on the ratio of the detection advantage to the extent of deviation ("irredundancy = detection/deviation").

The first step in our characterization is to bridge the gap between these two formulations of redundancy. While the definition of irredundancy is what allows us to use a redundancy-free function in our protocols, to find the core of a function, we rely on the formulation in terms of redundancy of individual inputs – we shall reduce redundancy one input or output at a time, until we obtain a redundancy free function. Clearly when redundancy is present, irredundancy would be 0 (i.e., can deviate without being detected); but we show that conversely, when irredundancy is 0, then one of the two forms of redundancy must be present. We stress that *a priori*, it is not at all obvious that irredundancy cannot be 0 even if there is no redundancy (i.e., detection/deviation could approach 0 by a sequence of deviations that are smaller and smaller, achieving even smaller detectability). We provide a non-trivial linear algebraic analysis of irredundancy and show that this is not the case (Lemma 1).

Simple Function. Following [31], we define a simple function. First, we present a combinatorial characterization (given in Lemma 1 in [31]) of a simple function, which constitutes the algorithm for determining if a function is simple or not.

A 2-party randomized function f is described by a joint distribution over Alice-Bob output space $W \times Z$ for every Alice-Bob input pair in $X \times Y$. We consider the $|Y||Z| \times |X||W|$ matrix $\mathfrak{P}^f$, with rows indexed by $(y, z) \in Y \times Z$ and columns indexed by $(x, w) \in X \times W$, such that $\mathfrak{P}^f_{(y,z),(x,w)} = \mathfrak{p}^f[w, z|x, y]$. The function f is simple if $\mathfrak{P}^f$ can be partitioned into a set of rank-1 minors such that no row or column of the matrix pass through two of these minors. Being of rank 1, each minor has all its rows (equivalently, columns) parallel to each other. (In [31], this is described in terms of a bipartite-graph in which each connected component is a complete bipartite graph, with weights on the edges being proportional to the product of the weights on the two end points of the vertex.)

To better understand what being simple means, we briefly explain how it is defined. The *kernel* of a function f is a symmetric function that provides both the parties with only the "common information" that f provides them with. A simple function is one which is "isomorphic" to its kernel: i.e., given just the output from the kernel, the rest of the information from f can be locally sampled by the two parties, independent of each other.

As stated in [31], the *passive*-complete functions are exactly those which are not simple. Our construction shows that *restricted to the class of redundancy free functions*, the same characterization holds for complete functions for active-security as well.

1.4 The construction

Our main construction shows that any redundancy free function f which is not simple is also UC-complete. This construction separates into two parts:

- A protocol to UC-securely reduce the commitment functionality $\mathcal{F}_{\text{COM}}$ to f.
- A protocol in the $\mathcal{F}_{\text{COM}}$-hybrid model that UC-securely reduces OT to f, starting from a passive-secure reduction of OT to f (since f is passive-complete, such a protocol exists). That is, we compile (in a black-box manner) a passive-secure OT protocol using f, to a UC-secure OT protocol using f (and $\mathcal{F}_{\text{COM}}$).

In building the commitment functionality we rely on a careful analysis of functions that are redundancy free and not simple, to show that there will exist two or more *extreme views* for one party (which cannot be equivocated) that are *confusable* by the second party (provided it uses inputs from an "unrevealing distribution" — something that can be verified by the first party). We interpret the function invocations as a channel through which the first party transmits a message using the set of its extreme views as the alphabet. This message is encoded using an error correcting code of rate $1 - o(1)$ and $o(1)$ distance; the distance would be sufficient to prevent equivocation during opening. To argue hiding, we rely on a well-known result from information theory, namely the (weak) *converse of Shannon's Channel Coding Theorem*. We extend this theorem to the case of adaptively chosen channel characteristics, corresponding to the fact that the receiver can adaptively choose its input to the function and that determines the channel characteristics. Due to confusability, the capacity of this channel will be less than 1 (measured with the logarithm of the input alphabet size as the base). Since the rate of the code is higher than the capacity of the channel, this gives us some amount of hiding (which is then refined using an extractor).

The second part, which gives a compiler, is similar in spirit to prior protocols that established that a passive-secure OT protocol (in the plain model) can be converted to an active-secure OT protocol *in a black-box manner* [20, 19, 9]. In particular, its high-level structure resembles that of the protocol in [9]. However, the key difference in our protocol compared to these earlier protocols (which were all in the computational setting), is that the passive-secure OT protocol that we are given is not in the plain model, but is in the f-hybrid model. The technical difficulty in our case is in ensuring 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. This is precisely where the statistical testability of redundancy free functions (see below) is invoked.

Also, in contrast with the above mentioned compilers, we do not use a two-step compilation to first obtain security against active corruption of the receiver and then that of the sender. Instead, we directly obtain a somewhat "noisy" OT protocol that is secure against active corruption of either player, and use techniques from [22, 21] to obtain the final protocol. In particular, we show how the result in [22] can be extended so that it works in a noisy OT-hybrid rather

than a regular OT-hybrid. (A similar extension was used in [21], to allow using a noisy channel hybrid instead of a regular OT-hybrid.) These tools help us achieve a constant rate in implementing OTs from instances of f.

Statistically Testable Games. We introduce a formal notion of statistically testable game, which is an information-theoretic analogue of interactive proofs where both players can be computationally unbounded. Note that interactive proofs are not interesting in this information-theoretic setting (or if P=PSPACE). In a statistically testable game, the statements being proven (tested) are statements regarding the private observations of the prover in a system, which provides partial observations to the verifier as well. The non-triviality of such a proof system stems not from the computational limitations of the verifier, but from the fact that the verifier cannot observe the entire system. While such proofs have been implicitly considered in several special cases in many prior works (e.g. [11, 12, 22, 21]), the class of games we consider is much more general than those implicitly considered in these earlier instances, and the soundness of the tests we consider is not at all obvious.

The game we consider is of 2-party function evaluation, in which the prover and the verifier interact with a (stateless) trusted third party which carries out a randomized function evaluation for them. The prover first declares a sequence of n inputs it will feed the function (the verifier chooses its inputs privately and independently). After n invocations of the function, the prover declares to the verifier the sequence of the n outputs it received from the invocations. A statistical test is a sound and complete proof system which convinces the verifier that the input and output sequences declared by the prover has a $o(1)$ fraction Hamming distance from the actual sequences in its interaction with the trusted party. Note that the verifier can use its local observations (its input-output sequences) to carry out the verification.

A major technical ingredient of our compiler is the following theorem:

> *Evaluation of a 2-party function f is statistically testable if and only if f is redundancy free.*

Clearly, if a function is not redundancy free, it admits no sound statistical test. But *a priori*, it may seem possible that even if no single input has redundancy, the prover can map the entire sequence of inputs and outputs to a different sequence, with only a small statistical difference in the verifier's view, such that this difference vanishes with the length of the sequence. We show that this is not the case: if the function is redundancy-free, then there is a lowerbound on the ratio of the "detection advantage" to "extent of deviation" that does not vanish with the number of invocations.

This naturally motivates our approach of compiling a passive-secure protocol in f-hybrid, where f is redundancy free, into one that is secure against active adversaries. We should be able to enforce honest behavior by "auditing" randomly chosen executions from a large number of executions, and the auditing would use the statistical tests. However, this idea does not work directly: the statistical test models a test by an *environment*: it lets the adversary arbitrarily interact

with f and report back a purported output, but the purported input it sent to f was fixed by the environment before the adversary obtained the output from f. On the other hand, in a protocol, the honest party does not get to see the input to be sent to the functionality ahead of time. It is to solve this issue that we rely on the commitment functionality: the input each party should be sending to f is fixed *a priori* using commitments (and coin-tossing-in-the-well). When a session is chosen for auditing, the adversary could have sent a different input to f than it was supposed to, and it can lie about the output it received from f as well, but it cannot choose the purported input it sent to f after interacting with f.

2 Preliminaries

Matrix Definitions. In the following we shall refer to the following matrix norms: $\|A\|_\infty = \max_i \sum_j |a_{ij}|$ (maximum absolute row sum norm), and $\|A\|_{\text{sum}} = \sum_{i,j} |a_{ij}|$ (absolute sum norm). We shall also use the function $\max(A) = \max_{i,j} a_{ij}$ (maximum value among all entries); note that here we do not consider the absolute value of the entries in A. For a probability distribution $\mathfrak{p}^X$ over a space X (denoted as vectors), we define $\min(\mathfrak{p}^X) = \min_{x \in X} \mathfrak{p}^X[x]$, the minimum probability it assigns to an element in X. The norm $\|\cdot\|_\infty$ when applied to a column vector simply equals the largest absolute value entry in the vector. We say that a matrix P is a *probability matrix* if its entries are all in the range $[0, 1]$ and $\|P\|_{\text{sum}} = 1$. We say that a matrix is a *stochastic matrix* (or row-stochastic matrix) if all its entries are in the range $[0, 1]$ and every row sums up to 1. For convenience, we define the notation $\langle M \rangle_I$ for a square matrix M to be the diagonal matrix derived from M by replacing all non-diagonal entries by 0.

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 $f : X \times Y \to W \times Z$. Despite the notation, the range of f is, more accurately, the space of probability distributions over $W \times Z$. The functionality takes an input $x \in X$ from Alice and an input $y \in Y$ from Bob and samples $(w, z) \in W \times Z$ 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 by $\mathfrak{p}^f[w, z|x, y]$. We use the following variables for the sizes of the sets W, X, Y, Z:

$$|X| = m \qquad |Y| = n \qquad |W| = q \qquad |Z| = r.$$

In this paper we shall restrict to function evaluations where m, n, q and r are constants, i.e. 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 $X = [m]$ (i.e., the set of first m positive integers), $Y = [n]$, $W = [q]$ and $Z = [r]$.

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 [7]. 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 f is *standalone-complete* (respectively, *UC-complete*) against information theoretic adversaries if any functionality g can be standalone securely (respectively, UC securely) computed in f hybrid. We shall also consider passive-complete functions where we consider security against passive (semi-honest) adversaries.

3 Main Tools

In this section we introduce the three main tools used in our construction.

3.1 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 protocol, which are designed to detect deviation, it is important to use a function in a form in which redundancy has been removed. We define irredundancy in an explicit linear algebraic fashion, and introduce a parameter to measure the extent of irredundancy.

Irredundancy of a System of Stochastic Matrices. Let P_i, $i = 1, \ldots, m$ be a collection of $s \times q$ probability matrices (i.e., entries in the range $[0, 1]$, with $\|P_i\|_{\text{sum}} = 1$). Consider tuples of the form $(j, \{M_i, \alpha_i\}_{i=1}^m)$, where $j \in [m]$, M_i are $q \times q$ stochastic matrices, and $\alpha_i \in [0, 1]$ are such that $\sum_i \alpha_i = 1$. Then we define the irredundancy of this system as

$$\mathfrak{D}(P_1, \ldots, P_m) = \inf_{(j, \{\alpha_i, M_i\}_{i=1}^m)} \frac{\|(\sum_{i=1}^m \alpha_i P_i M_i) - P_j\|_\infty}{1 - \alpha_j \|P_j \cdot \langle M_j \rangle_I\|_{\text{sum}}} \tag{1}$$

where the infimum is over tuples of the above form. (Recall that $\langle M_j \rangle_I$ refers to the diagonal matrix with the diagonal entries of M_j.)

Intuitively, consider the rows of P_i to be probability distributions over a q-ary alphabet produced as the outcome of a process with the row index corresponding to a hidden part of the outcome, and the column index being an observable outcome. Then, irredundancy measures how well a P_j can (or rather, cannot) be approximated by a convex combination of all the matrices P_i, possibly with the

observable outcome transformed using a stochastic matrix (corresponding to a probabilistic mapping of the observable outcomes); the denominator normalizes the approximability by how much overall *deviation* (probability of changing the process or changing the outcome) is involved. This excludes the trivial possibility of perfectly matching P_j by employing zero deviation (i.e., taking $\alpha_j = 1$ and $M_j = I$).

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 $\mathfrak{p}^f[w, z|x, y]$. 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 1. *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. A function f is said to be *redundancy-free* if it is left-redundancy free and right-redundancy free. The main result about irredundancy is the following quantitative lemma:

Lemma 1. *Suppose a 2-party function $f : X \times Y \to W \times Z$ is left redundancy free. Let $\mathfrak{p}^Y$ be a probability distribution over Y. Let the probability matrices $\{P^x\}_{x \in X}$, be defined by $P^x_{(y,z),w} = \mathfrak{p}^f[w, z|x, y]\mathfrak{p}^Y[y]$. Then there is a constant $\epsilon_f > 0$ (depending only on f) such that $\mathfrak{D}(P^1, \ldots, P^m) \geq \epsilon_f \min(\mathfrak{p}^Y)$.*

The analogous statement holds for right redundancy.

3.2 Statistically Testable Function Evaluation

In this section we consider the notion of a statistically testable function evaluation game. (The notion is more general and could be extended to reactive systems, or multi-player settings; for simplicity we define it only for the relevant setting of 2-party functions.) We informally defined a statistical test in Section 1.3. As mentioned there, we shall show that *evaluation of a 2-party function is statistically testable if and only if the function is redundancy free.*

For simplicity, we define a particular test and show that it is sound and complete for redundancy free functions (without formally defining statistical tests in general). (It is easy to see that functions with redundancy cannot have a sound and complete test. Since this is not relevant to our proof, we omit the details.)

Let f be redundancy free. Consider the following statistical test, formulated as a game between an honest challenger (verifier) and an adversary (prover) in the f-hybrid.

Left-Statistical-Test$(f, \mathfrak{p}^Y; N)$:

1. The adversary picks $\tilde{\mathbf{x}} = (\tilde{x}_1, \ldots, \tilde{x}_N) \in X^N$, and for each $i \in [N]$ the challenger (secretly) picks uniform i.i.d $y_i \in Y$, according to the distribution $\mathfrak{p}^Y$.
2. For each $i \in [N]$, the parties invoke f with inputs x_i and y_i respectively; the adversary receives w_i and the challenger receives z_i, where $(w_i, z_i) \xleftarrow{\$} f(x_i, y_i)$.
3. The adversary then outputs $\tilde{\mathbf{w}} = (\tilde{w}_1, \ldots, \tilde{w}_N) \in W^N$.

The adversary wins this game (breaks the soundness) if the following conditions hold:

1. Consistency: Let $\mu_{\tilde{w},\tilde{x},y,z}$ be the number of indices $i \in [N]$ such that $\tilde{w}_i = \tilde{w}, \tilde{x}_i = \tilde{x}$, $y_i = y$ and $z_i = z$. Also, let $\mu_{\tilde{x},y}$ be the number of indices $i \in [N]$ such that $\tilde{x}_i = \tilde{x}$ and $y_i = y$. The consistency condition requires that $\forall (w, x, y, z) \in W \times X \times Y \times Z$,

$$\mu_{\tilde{w},\tilde{x},y,z} = \mu_{\tilde{x},y} \times \mathfrak{p}^f[\tilde{w}, z|\tilde{x}, y] \pm N^{2/3}.$$

2. Separation: Let vectors $\mathbf{A}, \tilde{\mathbf{A}} \in (W \times X)^N$ be defined by $A_i := (w_i, x_i)$ and $\tilde{A}_i = (\tilde{w}_i, \tilde{x}_i)$. The separation condition requires that the Hamming distance between the vectors $\mathbf{A}$ and $\tilde{\mathbf{A}}$ is $\Delta(\mathbf{A}, \tilde{\mathbf{A}}) \geq N^{7/8}$.

The *Right-Statistical-Test*$(f, \mathfrak{p}^X; N)$ is defined analogously. The experiment *Statistical-Test*$(f, \mathfrak{p}^X, \mathfrak{p}^Y; N)$ consists of the left and right statistical tests, and the adversary wins if it wins in either experiment.

Before proceeding, we note that the above statistical test is indeed "complete": if the prover plays "honestly" and uses $\tilde{\mathbf{x}} = \mathbf{x}$ and $\tilde{\mathbf{w}} = \mathbf{w}$, then the consistency condition will be satisfied with all but negligible probability (for any choice of $\mathbf{x}$).

Lemma 2. *If f is redundancy free, and $\mathfrak{p}^X$ and $\mathfrak{p}^Y$ are constant distribution which have full support over X and Y respectively, then the probability that any adversary wins in Statistical-Test$(f, \mathfrak{p}^Y, \mathfrak{p}^X; N)$ is* $\mathrm{negl}(N)$.[8]

[8] The distributions $\mathfrak{p}^X$ and $\mathfrak{p}^Y$ are constant while N is a growing parameter.

3.3 A Converse of The Channel Coding Theorem

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 (see, for e.g., [10]). We give a generalization of the (weak) converse of channel coding theorem where the receiver can adaptively choose the channel based on its current view. We show that 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), then we can 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.

Lemma 3 (Weak Converse of Channel Coding Theorem, Generalization). *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 $c_1 \ldots c_N \equiv \mathbf{c} \xleftarrow{\$} \mathcal{C}$ 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 $\mathbf{c}$ is

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

4 Main Construction

The main ingredient for the proof of Theorem 1 is the following result (details are provided in the full version [27]):

Theorem 2. *If f is a redundancy free 2-party function and f is passive-complete, then there is a constant rate UC-secure protocol for $\mathcal{F}_{\text{OT}}$ in the f-hybrid model.*

Since f is passive-complete we know that OT does reduce to f against passive adversaries. We shall take such a passive-secure OT protocol in the f-hybrid, and convert it into a UC-secure protocol. For this we need two ingredients: first a UC-secure commitment protocol in the f-hybrid model, and secondly a compiler to turn the passive secure OT protocol in the f-hybrid model to a UC-secure protocol in the commitment-hybrid model. In building the UC-secure commitment protocol, we rely on the irredundancy of f as well as the combinatorial characterization that passive-complete functions are exactly those that are not simple (see Section 1.3).

4.1 A UC Secure Commitment Protocol

In this section we present the outline of a UC-secure commitment protocol in the f-hybrid model, for any 2-party randomized function f that is redundancy free (Definition 1) and is not simple (see Section 1.3).

The high-level structure of the protocol is as follows. The definition of the underlined terms cannot be accommodated due to lack of space. Interested readers should refer to [27].

1. Commitment phase:
   (a) The sender plays the role of (say) Alice in f, and the receiver plays the role of Bob in f. The sender invokes f several times, with random inputs $x \in X$; and the receiver will be required to pick its inputs from an *unrevealing distribution* $\mathfrak{p}^Y$.
   (b) The sender checks if the frequencies of all the input-output pairs (x, w) it sees are consistent with the receiver using $\mathfrak{p}^Y$.
   (c) The sender announces a subset of indices for which in the corresponding invocations, it obtained an *extreme* input-output pair.
   (d) The sender picks a random codeword from an appropriate code, and masks this codeword with the sequence of input-output pairs from the previous step, and sends it to the receiver.
   (e) The sender also sends the bit to be committed masked by a bit extracted from the codeword in the previous step.
2. Reveal phase: The sender sends its view from the commitment phase. The receiver checks that this is consistent with its view and the protocol (in particular, the purported codeword indeed belongs to the code, and for each possible value (x, w) of the sender's input-output pair to f, the frequency of input-output pairs (y, z) on its side are consistent with the function). If so, it accepts the purported committed bit.

The delicate part of this construction is to show that there will indeed be a set of extreme input-output pairs and an unrevealing distribution as required above. We point out that *we cannot use our results on statistical testability of the function evaluation game from Section 3.2 directly* to argue that binding would hold for all input-output pairs. This is because the game there requires the adversary to declare the input part of its purported view *before* invoking the function. Indeed, once we have a commitment functionality at our disposal, we can exploit the binding nature of this game; but to construct our commitment protocol this is not helpful.

Due to lack of space, we provide rest of our construction in the full version of the paper [27].

Acknowledgments

We thank Vinod Prabhakaran for helpful discussions on the converse of the Channel Coding Theorem.

References

1. Donald Beaver. Precomputing oblivious transfer. In Don Coppersmith, editor, *CRYPTO*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 1995.
2. Amos Beimel, Yuval Ishai, Ranjit Kumaresan, and Eyal Kushilevitz. On the cryptographic complexity of the worst functions. http://www.cs.umd.edu/ ranjit/BIKK.pdf. Retrieved Oct 16, 2013, 2013.
3. Amos Beimel, Tal Malkin, and Silvio Micali. The all-or-nothing nature of two-party secure computation. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 1999.
4. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *STOC*, pages 1–10. ACM, 1988.
5. Gilles Brassard, Claude Crépeau, and Jean-Marc Robert. All-or-nothing disclosure of secrets. In Andrew M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 234–238. Springer, 1986.
6. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Electronic Colloquium on Computational Complexity (ECCC) TR01-016, 2001. Previous version "A unified framework for analyzing security of protocols" available at the ECCC archive TR01-016. Extended abstract in FOCS 2001.
7. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2005. Revised version of [6].
8. David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In Janos Simon, editor, *STOC*, pages 11–19. ACM, 1988.
9. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 387–402. Springer, 2009.
10. Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.
11. Claude Crépeau and Joe Kilian. Achieving oblivious transfer using weakened security assumptions (extended abstract). In *FOCS*, pages 42–52. IEEE, 1988.
12. Claude Crépeau, Kirill Morozov, and Stefan Wolf. Efficient unconditional oblivious transfer from almost any noisy channel. In Carlo Blundo and Stelvio Cimato, editors, *SCN*, volume 3352 of *Lecture Notes in Computer Science*, pages 47–59. Springer, 2004.
13. Matthias Fitzi, Juan A. Garay, Ueli M. Maurer, and Rafail Ostrovsky. Minimal complete primitives for secure multi-party computation. *J. Cryptology*, 18(1):37–61, 2005.
14. Matthias Fitzi and Ueli M. Maurer. From partial consistency to global broadcast. In F. Frances Yao and Eugene M. Luks, editors, *STOC*, pages 494–503. ACM, 2000.
15. Oded Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.
16. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play ANY mental game. In Alfred V. Aho, editor, *STOC*, pages 218–229. ACM, 1987. See [15, Chap. 7] for more details.

17. Oded Goldreich and Ronen Vainish. How to solve any protocol problem - an efficiency improvement. In Carl Pomerance, editor, *CRYPTO*, volume 293 of *Lecture Notes in Computer Science*, pages 73–86. Springer, 1987.
18. Stuart Haber and Silvio Micali. Unpublished manuscript, 1986.
19. Iftach Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 412–426. Springer, 2008.
20. Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In *STOC*, pages 99–108. ACM, 2006.
21. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, Amit Sahai, and Jürg Wullschleger. Constant-rate oblivious transfer from noisy channels. In Phillip Rogaway, editor, *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 667–684. Springer, 2011.
22. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008.
23. Joe Kilian. Founding cryptography on oblivious transfer. In Janos Simon, editor, *STOC*, pages 20–31. ACM, 1988.
24. Joe Kilian. A general completeness theorem for two-party games. In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *STOC*, pages 553–560. ACM, 1991.
25. Joe Kilian. More general completeness theorems for secure two-party computation. In F. Frances Yao and Eugene M. Luks, editors, *STOC*, pages 316–324. ACM, 2000.
26. Joe Kilian, Eyal Kushilevitz, Silvio Micali, and Rafail Ostrovsky. Reducibility and completeness in private computations. *SIAM J. Comput.*, 29(4):1189–1208, 2000.
27. Daniel Kraschewski, Hemanta K. Maji, Manoj Prabhakaran, and Amit Sahai. A full characterization of completeness for two-party randomized function evaluation. *IACR Cryptology ePrint Archive*, 2014:50, 2014.
28. Daniel Kraschewski and Jörn Müller-Quade. Completeness theorems with constructive proofs for finite deterministic 2-party functions. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 364–381. Springer, 2011.
29. Eyal Kushilevitz, Silvio Micali, and Rafail Ostrovsky. Reducibility and completeness in multi-party private computations. In *FOCS*, pages 478–489. IEEE Computer Society, 1994.
30. Yehuda Lindell, Eran Omri, and Hila Zarosim. Completeness for symmetric two-party functionalities - revisited. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 116–133. Springer, 2012.
31. Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A unified characterization of completeness in secure function evaluation. To appear at INDOCRYPT, 2012.
32. M. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981.
33. Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In David S. Johnson, editor, *STOC*, pages 73–85. ACM, 1989.
34. Stephen Wiesner. Conjugate coding. *SIGACT News*, 15:78–88, January 1983.
35. Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *STOC*, pages 209–213. ACM, 1979.