Completeness in Two-Party Secure Computation Revisited

Danny Harnik[*] Moni Naor[†] Omer Reingold[‡] Alon Rosen[*]

Preliminary Version

Abstract

A Secure Function Evaluation (SFE) of a two-variable function $f(\cdot,\cdot)$ is a protocol that allows two parties with inputs x and y to evaluate $f(x,y)$ in a manner where neither party learns "more than is necessary". A rich body of work deals with the study of completeness for secure two-party computation. A function f is *complete* for SFE if a protocol for securely evaluating f allows the secure evaluation of all (efficiently computable) functions. The questions investigated are which functions are complete for SFE, which functions have "trivial" SFE and whether there are functions that are neither complete nor trivial.

The previous study of these questions was mainly conducted from an Information Theoretic point of view and provided strong answers in the form of combinatorial properties. However, we show that there are major differences between the information theoretic and computational settings. In particular, we show functions that are considered trivial by the combinatorial criteria but are actually complete in the computational setting.

We initiate the fully computational study of these fundamental questions. Somewhat surprisingly, we manage to provide an almost full characterization of the complete functions in this model as well. More precisely, we present a *computational criterion* (called *computational row non-transitivity*) for a function f to be complete for the asymmetric case. Furthermore, if f "disobeys" the new criterion, then f has a simple SFE (based on no additional assumptions).

[*]Department of Computer Science and Applied Mathematics, Weizmann Institute, Rehovot, 76100 Israel. mail: {harnik, alon}@wisdom.weizmann.ac.il. Research supported in part by a grant from the Israel Science Foundation.

[†]Incumbent of the Judith Kleeman Professorial Chair, Department of Computer Science and Applied Math, Weizmann Institute of Science, Rehovot 76100 Israel. Email: naor@wisdom.weizmann.ac.il. Research supported in part by a grant from the Israel Science Foundation.

[‡]AT&T Labs - Research. Room A201, 180 Park Avenue, Bldg. 103, Florham Park, NJ, 07932, USA. E-mail: omer@research.att.com. Part of this research was performed while visiting the Institute for Advanced Study, Princeton, NJ.

1 Introduction

A Secure Function Evaluation (SFE) of a two-variable function $f(\cdot,\cdot)$ is a protocol between two parties Alice and Bob, where Alice holds an input x and Bob holds an input y. Loosely speaking, at the end of the protocol Alice should learn the value $f(x, y)$ but learn nothing more (other than what can be efficiently deduced from x and $f(x, y)$). Bob, on the other hand, must learn nothing.

There are various definitions and models for SFE, and indeed the above definition describes just one of them. For example, one may consider a symmetric version where both parties learn the value $f(x, y)$. However, in this work we choose to concentrate on this asymmetric version, where only Alice receives the output.[1] Furthermore, at this point we conduct our work in the "honest but curious" model where Alice and Bob follow the protocol honestly, but may later try to extract more information from the transcript of the protocol.[2] We refer the reader to [12] for an extensive overview of the various possible models and definitions of SFE.

1.1 Completeness in Secure Function Evaluation

A major component in the construction of SFE protocols is the Oblivious Transfer protocol (OT), Rabin's brainchild [30]. OT refers to several different versions of SFE protocols, all of which turned out to be equivalent. For instance, consider the 1-2-OT [10], where Bob has two secret bits b_0, b_1 and Alice has a choice bit c. At the end of the protocol Alice learns b_c but learns nothing about b_{1-c}, while Bob learns nothing about Alice's choice. This can be viewed as an SFE protocol for the function $f_{\mathrm{OT}}(c,(b_0, b_1)) = b_c$.

OT plays a key role in secure computation since it was shown to be *complete for SFE* [33, 15, 22], i.e. the SFE of every efficiently computable function f can be efficiently reduced to OT. In other words, an SFE protocol for f can be constructed from any implementation of OT.[3] Indeed, several implementations of OT protocols have been suggested relying on various computational assumptions (see Section 2.4 for more details).

The fact that there exists a complete function for secure evaluation is intriguing in its own right and has led to the natural question of what other functions are complete. Let us denote by $\mathcal{SFE}$-C the set of functions which are complete for SFE. This set in particular contains the function f_{OT}. Let us also denote by Eff-$\mathcal{SFE}$ the set of functions for which there exists (efficient) SFE. The set Eff-$\mathcal{SFE}$ is also non empty as there are functions for which trivial SFE exists (such as functions $f(x, y)$ which only depend on x). There are many fundamental open problems regarding these sets, in particular, the following questions are the foci of our paper.

1. Which functions other than f_{OT} are complete for SFE? Is there a natural way of characterizing the functions in $\mathcal{SFE}$-C? The ability of identifying functions in $\mathcal{SFE}$-C can be particularly useful as a tool for implementing OT: If we are able to design SFE for a function f which is $\mathcal{SFE}$-C we are immediately guaranteed an implementation of OT (we show a specific example in Section 1.5).

2. How do the two sets relate? One possibility which is consistent with our current knowledge is that $\mathcal{SFE}$-C = Eff-$\mathcal{SFE}$ and they both contain every efficiently computable function. This is indeed implied by the existence of OT. If however OT does not exist, we have that

[1]The choice of the asymmetric model can be justified by the fact that a symmetric protocol seems problematic to achieve (as brought forth in [1]). This is since the first party to receive its output may maliciously end the protocol, thus preventing the other party from learning its output.

[2]Secure protocols in the honest but curious model can later be transformed to be secure in a malicious model as well (see Section 2).

[3]The notion of completeness is made formal in Section 2.3.

	y_0	y_1
x_0	a	a
x_1	a	b

	y_0	y_1
x_0	a	a
x_1	b	c

Table 1: Imbedded Or Insecure Minor

$\mathcal{SFE}$-C$\cap$Eff-$\mathcal{SFE} = \emptyset$. In this case, the picture is not as clear: Are there interesting functions that can still have "non-trivial" SFE? Consider in this case the variety of possible assumptions of the sort $f \in$ Eff-$\mathcal{SFE}$ for functions $f \notin \mathcal{SFE}$-C. Are any of these assumptions useful "fall backs" in the unfortunate scenario where OT does not exist?

1.2 Related Work

Indeed the above questions were also the focus of a large body of work [3, 6, 26, 23, 1, 27, 25, 24]. This study was mainly conducted from an Information Theoretic point of view. Specifically, in most of these papers SFE protects against computationally unbounded parties. Matching the definition of SFE, the notion of completeness is usually information theoretic as well. Most importantly, these papers usually only consider SFE of *finite functions*. That is, functions defined over a finite domain. As we will demonstrate below, when finite domain is interpreted as constant size domain, many of the computational concerns are avoided. From information theoretic perspective this line of research obtains very strong results. Loosely, these papers classify functions as either "complete" or "trivial" and provide combinatorial properties for separating the two. Formalizing the various models considered by this line of work (and specifically the various notions of completeness and triviality) is beyond the scope of this paper. Instead, we concentrate here on the combinatorial criteria provided by these works and analyze their applicability to the computational setting:

Imbedded Or: The papers initiating this line of research are those of Chor and Kushilevitz [6, 26] and Kilian [23]. They consider the symmetric version of SFE and prove that a function f is complete iff it contains an *imbedded OR*. An imbedded OR of a function f consists of inputs x_0, x_1 and y_0, y_1 such that $f(x_0, y_0) = f(x_0, y_1) = f(x_1, y_0) \neq f(x_1, y_1)$ (see Table 1.2).[4] When discussing Boolean functions, it turns out that the non-complete functions (functions not containing an Imbedded Or) are trivial.

Insecure Minor: Beimel, Malkin and Micali [1] discuss the asymmetric model (which is the focus of our work). They defined an *insecure minor* to consist of inputs x_0, x_1 and y_0, y_1 such that $f(x_0, y_0) = f(x_0, y_1)$ but $f(x_1, y_0) \neq f(x_1, y_1)$ (shown in Table 1.2),[5] and showed that every finite function with an insecure minor is complete, while every other finite function is trivial. This provides a very complete answer to the questions we posed above with respect to finite functions. This work was conducted under computational definitions of security, using the compiler of [15] to assure security against malicious parties rather than honest but curious parties (as we do in this work, see Section 2.1).

Other Work: Other works include the generalization of the criteria to the case of multi-parties (in the symmetric case) [27, 25], and probabilistic functionalities [24] (as opposed to deterministic functions).

[4]The name "imbedded OR", comes from the fact that the function projected to these four inputs looks like an OR or an AND gate.

[5]Note that any imbedded OR is also an insecure minor but not vice versa

The only other work that we are aware of that discusses a computational property rather than a combinatorial one is in [5], where a criterion for universally composable secure computation is given.

1.3 Computational Considerations:

While the combinatorial approach gives very elegant and tight answers with respect to finite functions, we stress that this is not satisfactory for general functions. Particularly, functions with no insecure minor can still be very far from trivial. In fact, such functions can even be complete!

The problem can be illustrated nicely by looking at a special case of functions with no insecure minor - the one-to-one functions.[6] By the BMM-criterion a one-one function f is "trivial" for SFE. This is justified by the following simple protocol: Let Bob send y to Alice. Indeed Bob learns nothing and Alice learns $f(x, y)$. Furthermore, Alice's view could be simulated from x and $f(x, y)$ as y is fully determined by these values. A-priory however, the running time of this simulator is exponential in n. This is acceptable if we think of n as small (as [1] did) but is impermissible when considering a large domain size. For general functions with no insecure minor (not necessarily one-to-one) the situation may even be worse. Not only may the simulator's running time be large, but the running time of the SFE protocol given by [1] may in itself be exponential in the input's length.

We show an example, where the BMM-classification *does not hold* when considering a polynomial-time function on large input size under the assumption that one-to-one one-way function exist.[7]

Example 1 *Let g be a one-to-one one-way function ($g : \{0,1\}^n \to \{0,1\}^n$ for all n). The function $f : \{0,1\}^1 \times \{0,1\}^{2n} \to \{0,1\}^{2n+1}$ is defined by:*

$$f(c, (y_0, y_1)) = (c, y_c, g(y_{1-c}))$$

As f is one-to-one it does not have an insecure minor. On the other hand, we claim that f is actually complete. As a proof sketch we present the following construction of OT from an SFE for f. Let Alice hold the choice bit c and Bob hold the secrets b_0, b_1. Alice and Bob run the SFE protocol for f with the bit c as Alice's input and random strings $y_0, y_1 \in \{0,1\}^n$ as Bob's inputs. Now Bob sends to Alice $b_0 \oplus h_g(y_0)$ and $b_1 \oplus h_g(y_1)$, where h_g is a hardcore bit of the function g (recall that g is one-way). Clearly Bob learns nothing from the protocol, since other than participating in the SFE he only sends information and receives none. Alice, on the other hand, learns y_c and hence she also learns $h_g(y_c)$ and subsequently the secret b_c (as required). However, due to the computational hardness of inverting g, Alice can't guess the bit b_{1-c} with more than negligible advantage over a coin toss (even though she learns $g(y_{1-c})$).

1.4 Our Results

Realizing that a combinatorial characterization does not suffice for all functions, one wonders whether at all there exist simple criteria that can capture the above notions in the general computational scenario? We show that the answer to this question is yes.

This paper presents computational criteria for completeness and triviality (being in $\mathcal{SFE}$-C and Eff-$\mathcal{SFE}$ respectively). These criteria are computational in nature and hold for functions of unbounded input length. Also, they are very close to being complementary to each other, thus almost fully categorizing all functions.

[6] In one-to-one functions $f(x_0, y_0) \neq f(x_1, y_0)$ for all x_0, x_1, y_0. So no insecure minor can exist

[7] Under the existence of OT every function is complete for SFE. Nevertheless, the existence of one-to-one one-way functions seems a much weaker assumption and in particular does not imply OT wrt. black-box reduction [21].

We define the following properties on a function f:

- f is called **(Computational) Row Transitive** if for every x_0, x_1 and y, one can efficiently compute $f(x_1, y)$ when given x_0, x_1 and $f(x_0, y)$.
- f is said to be **(Computational) Row Non-Transitive** if for some x_0, x_1 it is hard to compute $f(x_1, y)$ from x_0, x_1 and $f(x_0, y)$ for a random (unknown) y.

Comment: We call this property computational row transitivity since given a value of f taken from the row $f(x_0, \cdot)$ (viewing f as a table), Alice can efficiently deduce the corresponding value in row $f(x_1, \cdot)$. In essence this means that for Alice, learning the value at any row is equivalent.

Superficially, the two properties may seem exactly complementary to each other. However the exact statement (provided in Section 3) leaves a small gap between the two.

Main Theorem *Let $f : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$ be a polynomial time computable function.*

1. *If f is row transitive then it has an efficient SFE protocol ($f \in$ Eff-$\mathcal{SFE}$).*
2. *If f is row non-transitive then it is complete ($f \in \mathcal{SFE}$-C).*

Proof Sketch: In the following discussion we try to present the main ideas for the proofs, while the full proofs are presented in Sections 3 and 4. The proofs are conducted in the honest but curious model but the statements can apply in the malicious model as well (see Section 2.1).

(1) We show an efficient SFE protocol for a computational row transitive f. In general, the protocol simply consists of Bob choosing any input $\hat{x}$ and sending it to Alice along with $f(\hat{x}, y)$. Alice can then compute $f(x, y)$ by the row transitivity property. The view of Alice can be simulated by choosing $\hat{x}$ and computing $f(\hat{x}, y)$ from her output $f(x, y)$ (again by the transitivity property).

(2) As for the other direction, consider the following toy example: Suppose that we are given x_0, x_1 such that computing $f(x_1, y)$ from $f(x_0, y)$ is hard. Then using an SFE protocol for f we construct a version of OT we call Naive-OT [8], where the sender Bob has a secret bit b and the receiver Alice has a bit c that chooses whether she should learn Bob's secret or not (Bob learns nothing). Let Bob choose y and Alice choose either x_0 or x_1. The two parties run the SFE protocol for f with their chosen inputs. Now Bob sends $b \oplus h(f(x_1, y))$, where h is the Goldreich-Levin hardcore bit ([13]). If Alice chose x_1, then she learned $f(x_1, y)$ in the SFE and hence can learn Bob's secret bit, however if she chose x_0 she only learned $f(x_0, y)$ and cannot predict the hard core bit with better than negligible probability. Bob learns essentially nothing in the process.

The proof uses the Goldreich-Levin hardcore bit even though we are not dealing with a one-way function. This somewhat non-standard use calls for careful treatment:

- In order to prove the hardness of predicting the hardcore bit one must be able to efficiently sample pairs of values $f(x_0, y), f(x_1, y)$ for different y.
- Predicting the hardcore bit can be shown to be hard as long as the hardness of computing $f(x, y_1)$ is "strong hardness" (this is equivalent to the situation in one-way functions where strong one-way functions are needed).

While the first issue of sampling poses no problem, the need for strong hardness is problematic since requiring such hardness will leave a great gap between the criteria for $\mathcal{SFE}$-C and Eff-$\mathcal{SFE}$. To overcome this obstacle, what we actually do is take a hardcore bit that is only "weakly hard"

[8] In the honest but curious model, this is equivalent to the so called Rabin-OT, that in turn was shown to be equivalent to 1-2-OT [7]

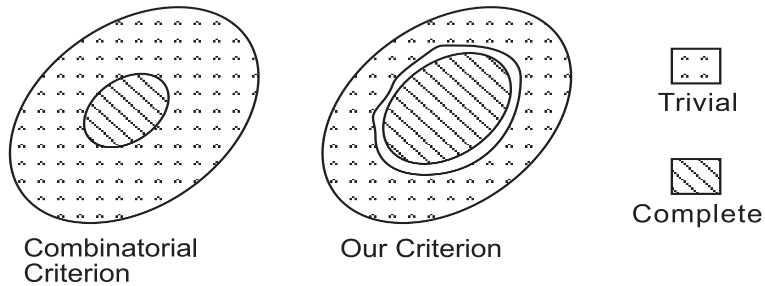

Figure 1: The area inside the outer circle depicts all efficiently computable functions. The left picture shows the tight characterization of functions according to the insecure minor criterion, while the right picture shows the characterization according to the "row transitivity" criteria.

to predict, and thus create an OT with weakened security (called Weak-OT [9] in this paper). We then show how one can use this weakened version of OT to construct a fully secure OT. This is shown by an amplification argument using Yao's XOR lemma.

1.5 The Meaning of our Result

The main theorem can be viewed as essentially categorizing all "nice" functions as either complete or trivial. Such results have been proved before, but were never shown for general (non-Boolean) functions, where computational considerations come into account. This new view allows the inclusion of more functions in the family $\mathcal{SFE}$-C, that were considered as "trivial" by the combinatorial criterion of [1]. The price paid is that the characterization by the row transitivity criteria is not as tight as the one given by the combinatorial criterion, but rather "almost" tight. This is depicted in Figure 1. Moreover, we show that this gap in our criteria cannot be closed altogether. This is shown in Appendix 5 by presenting an example of a function tailored not to be in either category. Hence, this picture reflects the actual state of affairs.

From a complexity point of view, this result should be considered as follows: Consider the case that OT protocols exist, then all efficiently computable functions are both complete and have SFE protocols ($\mathcal{SFE}$-C = Eff-$\mathcal{SFE}$). In such a world (called 'Cryptomania' by Impagliazzo in [19]) our result has no implication complexity wise, although it is still interesting as a tool for construction OT (see Example 2 below). On the other hand, consider a world where OT does not exist but One-Way Functions do exist (this world is called 'Minicrypt' in [19]). In such a world one can ask whether there are stronger assumptions than the existence of one-way functions that are meaningful. For example, is there an assumption that allows the secure evaluation of a family of non-trivial functions? Our results imply that the answer to the latter question is no. In other words, our main theorem essentially claims that as far as SFE protocols go, there are no additional worlds between Minicrypt and Cryptomania.

The result also has interesting aspects even in the case that OT does exist. This is due to the constructive nature of the proof in the sense that it actually describes how to construct an OT from an SFE for $f \in \mathcal{SFE}$-C. Consider the following example:

Example 2 *Consider the function $f_p(g, y) = g^y \mod p$, where p is a prime and q is a prime dividing $p-1$.*[10] *Let g be chosen from a multiplicative subgroup $Q \subseteq \mathbb{Z}_p^*$ of order q, and $y \in [q]$.*

[9] The notion of Weak-OT in this paper (defined in Section 3) is different than various versions of OT with the same name defined in other papers (e.g. [9, 24])

[10] For convenience choose p to be of the form $p = 2q + 1$ for prime q.

Let the security parameter n be $|p|$.

Under the Computational Diffie Helman (CDH) assumption, one cannot efficiently compute g_1^y given g_0, g_1 and g_0^y. This means that f is computational row non-transitive, and therefore complete (on the other hand, note that this function does not contain an insecure minor).

The following is a simple SFE protocol for f where Alice knows g, Bob knows y and at the end of the protocol Alice learns $g^y \mod p$. Let Alice choose a random $r \in [q]$ and send g^r to Bob. Bob then calculates $z = g^{ry} = (g^r)^y$ and sends it to Alice. Alice now takes the rth root from z and gets $z^{r^{-1}} = (g^{ry})^{r^{-1}} = g^y$. It is simple to show that the view of either side can be simulated and that this is indeed an SFE for f.

Now running the reduction from the proof of our main theorem actually produces the following protocol for Naive-OT: Let Bob be the sender holding a secret bit b and Alice be the receiver holding a choice bit c. Alice chooses random generators g_0, g_1 and $r \in \mathbb{Z}_p \setminus \{0\}$ and sends g_0, g_1, g_c^r to Bob. Bob calculates $z = g_c^{ry}$ and g_1^y and sends $z, h(g_1^y) \oplus b$ to Alice ($h(\cdot)$ again stands for the GL hardcore bit). Finally, Alice calculates $z^{r^{-1}} = g_c^y$. If $c = 1$ Alice can calculate $h(g_1^y)$ and subsequently learn the secret bit b. However, if $c = 0$ then Alice learns practically nothing.

This turns out to be equivalent to the well known OT protocol of Bellare and Micali [2]. It remains open whether one can use this framework to give new constructions for OT, using SFE protocols for other complete functions.

1.6 Paper Organization

Section 5 discusses alternative definitions that allow us to narrow the gap between the two criteria. Section 6 shows the amplification of Weak-OT to strong OT. In Section 2 we present the relevant formal definitions, including the notions of reductions and completeness. In Sections 3 and 4 we present our criteria and prove the main theorems. Section 7 mentions some further issues and questions.

2 Formal Setting

Some general notations for this paper:

- PPTM stands for Probabilistic Polynomial Time Turing Machine.
- By a **Distributions Ensemble** we mean a series $\{D_s\}_{s \in S}$ where S is an infinite set of strings and D_s is a distribution.
- A **Samplable Distribution** is a PPTM D accepting a unary number and outputting a binary string of polynomially related length i.e $D(1^n) \in \{0,1\}^{l(n)}$ where $l(n)$ is a polynomial. This is actually a distribution ensemble $\{D_n\}_{n \in \mathbb{N}}$ that can be efficiently sampled in the world of uniform computation.

2.1 The Honest but Curious Model

In all of the protocols discussed in the paper, we allow ourselves to assume that Alice and Bob are "Honest but Curious". This means that they follow the protocol exactly as specified (they are honest), but after the protocol is executed, the parties may try to extract more information than actually intended by inspecting the transcript of the protocol (they are curious). The justification for this assumption stems the compiler of Goldreich, Micali and Wigderson [15], that showed how any protocol that is secure in the honest but curious model, can be transformed into a protocol secure in the malicious model (assuming one-way functions exist). This means that assuming one-way functions do exist, all of the results we present carry over to the real world were parties may be malicious.

Note that the existence of SFE for any complete function implies the existence of OT in the honest but curious model, and this in turn implies the existence of one-way functions (as shown in [20]). So for the completeness theorem we get the existence of one-way functions for free.

2.2 Secure Function Evaluation

Discussing the various definitions of SFE is beyond the scope of this paper. The reader is referred to [12] for a good overview of possible definitions. In this work we focus on a specific version of SFE, namely the computational asymmetric version, where only one party gets the output of the deterministic function f (as opposed to probabilistic functionality). The definitions we present here are along the lines of [12, 4].

Unlike the discussion of finite functions, we allow the functions to receive very long inputs. This is done in the standard fashion by relating the complexity and security parameters to the function's input length. In order to accommodate this, we make the following assumptions:

- f is computable in time polynomial in the input length (combined length of the inputs).
- The security parameter for the SFE protocol is the input length (usually denoted n).
- For sake of simplicity and without loss of generality, assume that for input length n the output is always of length $m(n)$ (where $m(\cdot)$ is a polynomial). This assumption is justified by a padding argument.[11]
- Assume that all parties know n (the length of the input). Also, when giving partial inputs to various PPTM, one should also give 1^n as an extra input. However we omit this extra parameter for ease of notation.

Let Π be a protocol between Alice and Bob. Denote Alice's output by $\Pi_A(x, y)$ and Bob's output by $\Pi_B(x, y)$. In our case we simply denote $\Pi(x, y) = \Pi_A(x, y)$ (since $\Pi_B(x, y)$ is always empty). Denote Alice's view of the protocol by $\text{VIEW}_A^{\Pi}(x, y)$ (this includes Alice's local input, local randomness, her output and all of the messages received from Bob). Similarly Bob's view is $\text{VIEW}_B^{\Pi}(x, y)$.

Definition 2.1 *Let S be an unbounded set of strings of integers. And let $\{X_s\}_{s\in S}$ and $\{Y_s\}_{s\in S}$ be distribution ensembles. We say that $\{X_s\}$ and $\{Y_s\}$ are* **computationally indistinguishable** *(Denoted $\{X_s\} \approx_c \{Y_s\}$) if for every PPTM M running in polynomial-time in its first input, every polynomial $q(\cdot)$, all sufficiently large n, all $s \in S \cap \{0,1\}^n$ and all auxiliary information $w \in \{0,1\}^*$ we have:*

$$|\Pr[M(1^n, w, X_s)] - \Pr[M(1^n, w, Y_s)]| < \frac{1}{q(n)}$$

The probability is taken over the distributions X_s, X_y and the randomness of M

The formal definition of an SFE protocol requires that each party's view of the protocol can be efficiently simulated, even when only seeing the party's local input and output. Hence practically nothing is gained by seeing the view of the protocol.

[11] (Any general function can be turned into such a function - If the output is too short, we simply pad it with zeros to the appropriate length).

Definition 2.2 (SFE Protocol) *Let $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be a poly-time function. A polynomial time protocol Π is a* **Secure Function Evaluation (SFE)** *for f if the following holds:*

1. *Correctness: For every $x, y \in \{0,1\}^*$: $\Pi(x,y) = f(x,y)$.*
2. *Security:*
   - *Bob's Privacy: There exists a PPTM S_A such that:*
   $$\{S_A(x, f(x,y))\}_{x,y\in\{0,1\}^*} \approx_c \{\text{VIEW}_A^{\Pi}(x,y)\}_{x,y\in\{0,1\}^*}$$
   - *Alice's Privacy: There exists a PPTM S_B such that:*
   $$\{S_B(y)\}_{x,y\in\{0,1\}^*} \approx_c \{\text{VIEW}_B^{\Pi}(x,y)\}_{x,y\in\{0,1\}^*}$$

Denote the set of functions that have an SFE protocol by Eff-$\mathcal{SFE}$. These functions are sometimes referred to as "trivial". While the phrase "trivial" might be a bit misleading (since efficient SFE protocols may be very complex at times), we note that all of our results showing $f \in$ Eff-$\mathcal{SFE}$ do in fact introduce very trivial algorithms.

2.3 Reductions and Completeness in SFE

The definitions of reductions and completeness in the context of SFE are simply extensions of these notions in general (polynomial time) computation. We say that g securely reduces to f if one can construct an SFE protocol for g using the execution of an SFE protocol for f. A function f is SFE-complete if given an SFE protocol for f, it is possible to securely evaluate any function. Formally:

Definition 2.3 (Secure Reduction) *We say that a polynomial time function $g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$* **securely reduces** *to a polynomial time function $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ if there exists a polynomial time protocol $T_g^{\Pi_f}$ (polynomial in the length of the input to g) with the ability to invoke a protocol Π_f such that if Π_f is an SFE protocol for f, then the resulting protocol $T_g^{\Pi_f}$ is an SFE for g.*

By "the ability to invoke Π_f" we mean a protocol in which the parties can flip coins, send and receive messages, and also jointly run a protocol Π_f. The parties choose local inputs for f and run the protocol Π_f in which one side acts as Alice (and receives the output $f(x)$) while the other plays Bob.[12]

Definition 2.4 (SFE-Complete) *We say that a function f is* **SFE-Complete** *if every efficiently computable function g securely reduces to f. Denote:*

$$\mathcal{SFE}\text{-}C = \{f \mid f \text{ is complete for SFE }\}$$

These definitions have the following essential composability properties (as shown by Canetti [4]):

- If h reduces to g and g reduces to f then h reduces to f.
- If f reduces to g and f is complete then g is complete.

This paper aims at finding a simple criterion as to what functions are in $\mathcal{SFE}$-C. The common method of proving that a function is SFE-complete is showing a secure reduction from a previously known complete function, namely, to Oblivious Transfer.

[12] Since the whole process is polynomial time, the length of the inputs to Π_f is also bounded by a polynomial in the length of the input to g. An invocation of the protocol Π_f may be considered as a constant time operation.

2.4 Oblivious Transfer

A central component in the construction of SFE protocols is the Oblivious Transfer (OT) protocol. OT refers to several equivalent versions of two-party protocols. For example, an important formulation is the 1-2-OT (due to [10]), where Bob has two secret bits b_0, b_1 and Alice has a choice bit c. At the end of the protocol Alice learns b_c but learns nothing about b_{1-c}, while Bob learns nothing about Alice's choice. In principal, we can view a 1-2-OT protocol through the framework of SFE. Namely, an SFE protocol for the function $f(c,(b_0,b_1)) = b_c$.

Another important version is known as Noisy-OT or Rabin-OT (due to Rabin [30]), and was shown to be equivalent to 1-2-OT in [7]. This version goes as follows: Bob holds a secret bit b. After the protocol, Alice receives the bit b with probability $\frac{1}{2}$ (with probability $\frac{1}{2}$ she learns nothing), and Bob doesn't know if Alice received the bit or not.

In this paper we use a slightly different version that we call Naive-OT, that is clearly equivalent to the Rabin-OT in the honest but curious model. In this version Alice simply chooses whether to receive Bob's secret bit or not, while Bob learns nothing of this choice.

Definition 2.5 *A* **Naive-OT** *protocol is an SFE protocol for the function:*

$$f(c,b) = \begin{cases} b & c = 1 \\ \perp & c = 0 \end{cases}$$

As mentioned before, the importance of OT stems from the fact that it is complete for SFE. This was shown in an array of works, originally attributed to [31, 33] with a line of further works, e.g. [14, 17, 22]. In turn, OT can be constructed from trapdoor permutations and public key encryption with particular "niceness" properties [10, 12, 11] and various specific intractability assumptions (e.g. the Diffie Helman assumptions [2, 29]). In contrast, OT cannot be reduced in a black-box manner to weaker primitives such as one-way functions, general public key encryption or even trapdoor one-to-one functions [21, 19, 11].

3 Completeness

Our main result presents criteria for functions to being complete or trivial. While the two criteria are not complementary of each other, they are in a sense a "strong negation" of each other. For simplicity of exposition, the criteria we present in Sections 3 and 4 are not as tight as possible, and give results for the most natural definition of SFE. In Appendix 5 we present alternative definitions that bring us closer to a tight characterization.

Definition 3.1 (Computational Row Non-Transitivity:) *We say that a function $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ is* **(Computational) Row Non-Transitive** *if there exist polynomial-time samplable distributions $D_x(\cdot), D_y(\cdot)$ and there exists a polynomial $p(\cdot)$ such that for every PPTM M and for all but finitely many n:*

$$\Pr[M(x_0, x_1, f(x_0, y)) = f(x_1, y)] < 1 - \frac{1}{p(n)}$$

Where the probability is taken over $x_0, x_1 \in D_x(1^n), y \in D_y(1^n)$ and the randomness of M.

Theorem 3.1 *If a function f is computational row non-transitive then it is complete for SFE.*

To prove the theorem we show a construction of an OT protocol using an SFE protocol for a non-transitive f. Rather than directly showing this, we first construct a weakened implementation of Naive-OT that is later shown to imply OT. This is called Weak-OT and is essentially a relaxation of the security restrictions on the receiver.

Definition 3.2 *A* **Weak implementation of Naive-OT (Weak-OT in short)** *consists of two parties, the sender and the receiver. The sender holds a secret bit b and the receiver holds a choice bit c. Both parties have a security parameter n. At the end of the protocol, the following holds:*

1. *Correctness: If $c = 1$ then the receiver outputs bit b.*

2. *Security:*

   - *Receiver's view: If $c = 0$ then there exists a polynomial $p'(\cdot)$ such that for every PPTM A, for all but finitely many n:*

   $$\Pr[A(\text{VIEW}_{receiver}^{weak-OT}(c, b)) = b] < 1 - \frac{1}{p'(n)}$$

   - *Sender's view: for every PPTM B, for any polynomial $q(\cdot)$, for all but finitely many n:*

   $$\Pr[B(\text{VIEW}_{sender}^{weak-OT}(c, b)) = c] < \frac{1}{2} + \frac{1}{q(n)}$$

The definition of Weak-OT is justified by the following claim:

Lemma 3.2 *The existence of Weak-OT implies the existence of OT.*

The Lemma is proved in Appendix 6. We now turn to prove Theorem 3.1:

Proof. (of Theorem 3.1) We show a construction of a Weak-OT protocol using an SFE protocol for a row non-transitive f.

In the Weak-OT the sender holds a secret bit b and the receiver holds a choice bit c. The two sides can engage in an SFE protocol Π_f for the non-transitive function f. Let the receiver play Alice (holds x) and the sender play Bob (holds y).

Weak-OT$^{\Pi_f}(c, b)$:

1. The Receiver chooses random x_0, x_1 according to the distribution $D_x(1^n)$ and sends x_1 to the Sender (receiver).

2. The receiver and sender jointly run the protocol Π_f.

   The receiver uses $x = x_c$

   The sender chooses a random y according to $D_y(1^n)$.

   At the end of the protocol the receiver learns $\Pi_f(x, y)$.

3. The sender:

   Computes $z = f(x_1, y)$ and chooses a string r uniformly at random in $\{0, 1\}^{|z|}$.

   Sends to the receiver r and $\langle z, r\rangle \oplus b$, where $\langle z, r\rangle$ denotes the inner product of the strings z and r mod 2 (this is actually the Goldreich-Levin predicate).

4. If $c = 1$ then the receiver retrieves the bit b by calculating $\langle \Pi_f(x, y), r\rangle = \langle z, r\rangle$.

We show that the above protocol constitutes of a Weak-OT protocol:

Correctness: If $c = 1$, then the receiver learns $\Pi_f(x, y) = f(x_1, y) = z$, and can therefore learn $\langle z, r\rangle$ and can retrieve the bit b.

Security:

- Sender's view: The sender's view consists only of his view in the protocol Π_f. By the properties of Π_f, anything the sender can learn with this view, he can deduce without interaction at all (up to a negligible difference). Hence learning the receiver's bit with a non-negligible advantage implies the ability to guess the bit c without interaction at all, which is of course impossible.

- Receiver's view: If $c = 0$, then the receiver learns $\Pi_f(x, y) = f(x_0, y)$. We show that the receiver cannot use this information to predict the bit $\langle f(x_1, y), r\rangle$ (and equivalently the bit b), with probability better than $1-\frac{1}{p(n)}$ for some polynomial $p(\cdot)$. This statement is formalized and proved in the following lemma.

Lemma 3.3 *Suppose f is row non-transitive. Then there exists a polynomial $p(\cdot)$ such that for any PPTM A and for all but finitely many n's:*

$$\Pr[A(x_0, x_1, f(x_0, y), r) = \langle f(x_1, y), r\rangle] < 1 - \frac{1}{p(n)}$$

Where the probability is taken over $x_0, x_1 \in D_x(1^n), y \in D_y(1^n), r \in \{0,1\}^{|f(x_1,y)|}$ and the randomness of A.

Note that the above Lemma is equivalent the weak version of Goldreich-Levin's hardcore bit.

Proof. Suppose for the sake of contradiction that there exists a PPTM A that manages to predict $\langle f(x_1, y), r\rangle$ given $f(x_0, y)$ and r with overwhelming probability. More precisely: there exists a PPTM A such that for any polynomial $q(\cdot)$, for infinitely many n's:

$$\Pr[A(x_0, x_1, f(x_0, y), r) = \langle f(x_1, y), r\rangle] > 1 - \frac{1}{q(n)} \tag{1}$$

We derive a contradiction to the non-transitivity of f by presenting a procedure B (with black box access to A) that correctly computes $f(x_1, y)$ given $f(x_0, y)$ with very high probability.

We concentrate our efforts on inputs y on which the algorithm A succeeds with probability greater than $\frac{9}{10}$. Define:

$$E_n = \{y|\Pr[A(x_0, x_1, f(x_0, y), r) = \langle f(x_1, y), r\rangle] > \frac{9}{10}\}$$

Since A has a very high success probability, the set E_n must contain almost all of the inputs. We use the following easy claim (brought here without a proof):

Claim 3.4 *For any polynomial $q(\cdot)$, for all but finitely many n's:*

$$\Pr_{y\in D_y(1^n)}[E_n] > 1 - \frac{1}{q(n)}$$

The claim is specifically true for $q(n) = 2p(n)$, where $p(\cdot)$ is the polynomial given by the non-transitivity of f. So $\Pr_{y \in D_y(1^n)}[E_n] > 1 - \frac{1}{2p(n)}$ for all but finitely many n.

Next we introduce the procedure B that uses A to calculate $f(x_1, y)$ given $f(x_0, y)$ for any $y \in E_n$ and with very high probability (over the procedure's random bits). Consider a specific $y \in E_n$. The calculation of $f(x_1, y)$ is done bit by bit. In order to calculate the ith bit (denoted $f(x_1, y)_i$), choose a random $r \in \{0,1\}^{|f(x_1,y)|}$ and compute $A(x_0, x_1, f(x_0, y), r)$ and $A(x_0, x_1, f(x_0, y), r \oplus e^i)$ (where e^i is the binary vector with one in the ith place and zero in all others). If A succeeds on both inputs then:

$$\begin{aligned} A(x_0, x_1, f(x_0, y), r) \oplus A(x_0, x_1, f(x_0, y), r \oplus e^i) &= \langle f(x_1, y), r\rangle \oplus \langle f(x_1, y), r \oplus e^i\rangle \\ &= f(x_1, y)_i \end{aligned}$$

The probability that A succeeds on both of the above inputs is at least $1 - 2 \cdot (1 - \frac{9}{10}) = \frac{8}{10}$. Repeating this n times and taking a majority gives the bit $f(x_1, y)_i$ with exponentially small error probability $2^{-\Omega(n)}$ (by a Chernoff bound). This procedure is carried out for every bit of $f(x_1, y)$ separately, ultimately outputting the full string $f(x_1, y)$ with probability at least $1 - n \cdot 2^{-\Omega(n)} = 1 - 2^{-\Omega(n)}$ for infinitely many n. Combining this with Claim 3.4 and looking at all inputs $y \in D_y(1^n)$, the described efficient procedure B computes $f(x_1, y)$ from $f(x_0, y)$ with probability at least: $\Pr[E_n] \cdot (1 - 2^{-\Omega(n)}) > (1 - \frac{1}{2p(n)})(1 - 2^{-\Omega(n)}) > 1 - \frac{1}{p(n)}$, contradicting the non-transitivity of f.

□

To conclude the proof suppose that there exists a PPTM A that predicts the bit b from the receiver's view, with probability better than $1 - \frac{1}{q(n)}$ for any polynomial $q(\cdot)$ and for infinitely many n's. Notice that the receiver's view only consists of the view of Π_f and the bit $\langle f(x_1, y), r\rangle \oplus b$. Suppose we replace the view of Π_f by $S_A(f(x_0, y))$, where S_A is the simulator guaranteed by the definition of SFE (Appendix 2.2). The algorithm A acts on the new simulated view as it does on the real view, up to a negligible error (this is due to the definition of SFE). Now define A' to be the efficient procedure that takes input $f(x_0, y)$, simulates the view using S_A and runs A on this simulated view concatenated with the zero bit. By the above argument, A' predicts $\langle f(x_1, r)\rangle$ given only $f(x_0, y)$ with probability better than $1 - \frac{1}{q(n)}$ for any polynomial $q(\cdot)$, and this contradicts Lemma 3.3.

■

The BMM Criterion Implies Non-Transitivity An important test case for our completeness criterion is whether it encapsulates previous results, namely, the insecure minor criterion of [1]. Recall that an insecure minor consists of inputs x_0, x_1 and y_0, y_1 such that $f(x_0, y_0) = f(x_0, y_1)$ but $f(x_1, y_0) \neq f(x_1, y_1)$.[13]

Choosing D_x to be uniform on $\{x_0, x_1\}$ and D_y to be uniform on $\{y_0, y_1\}$ we get that for any PPTM M (or any function at all in this case):

$$\Pr[M(x_0, x_1, f(x_0, y)) = f(x_1, y)] \leq \frac{1}{2}$$

Hence the function f is also computational row non-transitive.

[13] We note that this definition is for functions of constant input length. The generalization to general functions requires to have an insecure minor for every input length n. Also require that finding the values x_0, x_1 and y_0, y_1 can be done in polynomial time.

4 Triviality

While the computational row non-transitive condition requests the "weak hardness" of computing $f(x_1, y)$ from $f(x_0, y)$, the condition for triviality requests that this computation is easy.

Definition 4.1 Computational Row Transitivity *We say that a function* $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ *is* **(computational) row transitive** *if there exists a PPTM* M *such that for all inputs* $x_0, x_1 \in \{0,1\}^n, y \in \{0,1\}^{n'}$ *:*

$$M(x_0, x_1, f(x_0, y)) = f(x_1, y)$$

Claim 4.1 *Let* f *be an efficiently computable function and suppose it is computational row transitive then there exists an SFE protocol for* f*.*[14]

Proof. Suppose that f is row transitive, then the following is an SFE protocol for f:

$\Pi_f(x, y)$

1. Bob chooses an arbitrary $\hat{x} \in \{0,1\}^n$ and sends $\hat{x}$ and $f(\hat{x}, y)$ to Alice.
2. Alice calculates and outputs $M(\hat{x}, x, f(\hat{x}, y))$.

The above preocedure is easily shown to be an SFE protocol for f. The correctness follows since Alice learns $M(\hat{x}, x, f(\hat{x}, y)) = f(x, y)$ as required. As for security: Bob does not learn anything from the protocol simply because he gets no message from Alice. On the other hand, Alice's view can be simulated given the output $f(x, y)$, simply by choosing a random $\hat{x}$ and calculating $M(x, \hat{x}, f(x, y)) = f(\hat{x}, y)$. ■

5 Narrowing the Gap

Our ultimate goal is to categorize all functions as either complete or trivial. However, this does not seem to reflect the actual world. Following is an example of a function that seems neither complete nor trivial, even when considering very relaxed notions of secure computation (e.g. allowing SFE where the correctness is only on most of the inputs).

Example 3 *Let the function* $g : \mathbb{N} \to \mathbb{N}$ *be defined recursively as follows.* $g(0) = 1$ *and for all* $k \in \mathbb{N}$*,* $g(k) = 2^{g(k-1)}$ *(i.e.* $g(k) = 2^{2^{\cdot^{\cdot^{2}}}}$ *with* k *exponentiations). The function* f *is defined as follows:*

$$f(x, y) = \begin{cases} y & g(k) < |y| \leq g(k+1) \text{ for even } k \\ OT(x, y) & g(k) < |y| \leq g(k+1) \text{ for odd } k \end{cases}$$

While f cannot be trivial, since it implements OT on infinitely many input lengths, it also cannot be complete (if OT does not exist) since given an SFE for f, it cannot be used for constructing an OT with security parameter $n = g(k)$ where k is odd. The reason being that the next meaningful calculation after length n is of length 2^n, which is cannot be executed in polynomial time.

That being said, we reiterate that the criteria we presented thus far were aimed to be rather simple and to correspond to the strictest definition of SFE. This leaves more functions not categorized than we can actually achieve. In this section we discuss how to narrow the gap left between

[14] Note that Claim 4.1 is unconditional. That is, the validity of the SFE protocol does not rely on any assumption (such as the existence of OT).

the two categories. These changes may impose relaxations on the definition of SFE. While some of the relaxations we present are very natural, others are more problematic. We sketch the different possibilities below:

- The requirement for "all but finitely many n" in the completeness condition is much stronger than what is needed. The real need is that for any security parameter n we can efficiently choose an input length $l(n)$ for which calculating $f(x_1, y)$ from $f(x_0, y)$ is "weakly hard" (probability here should also include the randomness in choosing $l(n)$).

- It is very natural to relax the correctness requirement of the SFE protocol. In the basic definition we require that $\Pi_f(x, y) = f(x, y)$ for all x, y and with probability 1. Allowing a small error here also allows a relaxation of the triviality condition. The most natural definition would be that $\Pi_f(x, y) = f(x, y)$ with negligible error probability for all x, y. However, in order to accommodate a similar slackness in the row transitivity, we relax the correctness further and ask that the probability is taken over the choices of x, y as well (thus allowing a negligible fraction of the inputs for which there is no SFE).

- A far less natural relaxation in the triviality condition is drawn from the negation of the completeness condition. Rather than having a PPTM M that has success probability at least $1 - \frac{1}{q(n)}$ for *every* polynomial $q(\cdot)$, we ask that for every polynomial $q(\cdot)$ there is a PPTM M_q with success probability at least $1 - \frac{1}{q(n)}$. While this requirement means that we can have error probability smaller than any polynomial we wish, this polynomial has to be set in advance. This relaxation induces a similar relaxation in the definition of SFE. Hence, for a given polynomials $p(\cdot)$ and $q(\cdot)$, a (p, q)-SFE protocol is one that allows errors of up to $\frac{1}{p(n)}$ in the correctness and $\frac{1}{q(n)}$ in the simulation of the receiver's view. Such a relaxation will achieve very tight results, but for a far less natural definition of SFE.

6 Weak-OT implies OT

This section show that the weak implementation of OT (Weak-OT) protocol implies a strong OT. We note that this amplification result defers from previous such results like [8, 9] that were all information theoretic. Our amplification works against computational adversaries and requires more complex proofs using tools such as Yao's XOR-lemma.

Recall that the definition of a "strong" Naive-OT protocol requires that the views of the sender and receiver may be simulated (up to a negligible deviance), when seeing just their local input and output. This is equivalent to saying that the sender (or receiver) cannot guess the other side's input bit with non-negligible advantage over flipping a coin (when seeing their local view of the protocol). A Weak-OT is a relaxation of the latter definition. The sender is restricted in the same manner, but the receiver is allowed a higher success probability. We require that the receiver's success probability in guessing the sender's bit b is bounded by $1 - \frac{1}{p(n)}$ for a specific polynomial $p(\cdot)$. We show the following:

Lemma 3.2 *Weak-OT exists if and only if OT exists.*

Proof. Any OT protocol is also a Weak-OT protocol since it withstands harder security requirements. To prove that Weak-OT implies the existence of a strong Naive-OT, we present a construction of an OT protocol based on a protocol for Weak-OT. Let $p(\cdot)$ be the promised polynomial in the Weak-OT's definition of security for the receiver's view. Define $t(n) = p(n)^2$.

OT$^{Weak-OT}(c, b)$:

1. The Sender randomly chooses $t(n)$ random bits $b_1, ..., b_{t(n)-1} \in_R \{0, 1\}$ and sets $b_{t(n)}$ such that $b = \bigoplus_{i=1}^{t(n)} b_i$.
2. The sender and receiver run the protocol Weak-OT(c, b_i) for every i.
3. If $c = 0$, then the receiver computes $b = \bigoplus_{i=1}^{t(n)} b_i$.

Note: Stage 2 can be run in parallel since we are in the honest but curious model.

The correctness of the above protocol follows immediately.

The Sender's view: Suppose the sender can guess the bit c with advantage of $\frac{1}{q(n)}$ for a polynomial $q(\cdot)$. Then by a hybrid argument, there is an algorithm that can guess c on a single weak-OT run with advantage of $\frac{1}{t(n)q(n)}$ thus contradicting the weak-OT definition. So the sender can only guess the bit c with negligible advantage.

The Receiver's view: To show that the receiver's advantage is only negligible we apply the so called Yao's XOR-Lemma. Originally due to Yao (in presentations of [32]), with formal proofs presented in [28, 18, 16]. The XOR-Lemma states the following:

Suppose that $P : \{0, 1\}^m \to \{0, 1\}$ is a predicate that is "weakly hard to compute". Let $x^{(t)} = (x_1, ..., x_t)$ be a t-tuple of independent inputs to P. Denote $P^{(t)}(x^{(t)}) = \bigoplus_{i=1}^{t} P(x_i)$. Then the lemma asserts that P^t is "strongly hard to compute". Formally:

Theorem 6.1 (The XOR-Lemma) *Suppose that any PPTM fails to calculate $P(x)$ with probability better than $1 - \frac{1}{p(n)}$ for all but finitely many n and for a given polynomial $p(\cdot)$. Take $t(n) = p(n)^2$, and suppose that there exists an efficient procedure to sample couples $(x, P(x))$. Then any PPTM fails to calculate $P^{(t(n))}(x^{(t(n))})$ with probability better than $\frac{1}{2} + \frac{1}{q(n)}$ for any polynomial $q(n)$ for all but finitely many n's (the probabilities are taken over a given distribution on the inputs $x \in \{0, 1\}^m$ and the randomness of the PPTMs).*

In our application we take $x_i = \text{VIEW}_{receiver}^{Weak-OT}(c = 0, b_i)$ and $P(x_i) = b_i$. By the definition of Weak-OT, P is indeed weakly hard. The conclusion is that if $c = 0$, then the sender cannot predict the bit $b = \bigoplus_{i=1}^{p(n)^2} b_i$ from the view of OT$^{Weak-OT}$ with a non-negligible advantage, as required in a strong OT protocol.

In order to apply the XOR-Lemma, one must provide an efficient procedure for producing random samples of pairs x and $P(x)$. While this is not necessarily achievable for a general predicate P, it is possible in our case where $x = \text{VIEW}_{receiver}^{Weak-OT}(c = 0, b)$. The receiver can simply choose a random b and simulate a protocol of $Weak - OT(c, b)$, outputting the pair $(\text{VIEW}_{receiver}^{Weak-OT}(0, b), b)$.

■

7 Further Issues

The Symmetric SFE Model: In this model, both sides receive the output $f(x, y)$ at the end of the protocol. Actually, most of the works considering the questions of completeness and triviality were conducted in this setting (e.g. [6, 26, 23, 27, 25] and part of [24]). While the study of Boolean functions in this model, yielded clean and tight results [6], this is not the case when considering non-Boolean functions. In fact Kushilevitz [26] demonstrates a function that is neither complete nor trivial in the information theoretic model, so no tight characterization can be expected.

On the other hand, these works did show that the complete functions are exactly those containing an imbedded OR. However this result does not carry over when discussing computational security, as the following example shows:

Example 4 *Let $g : \{0,1\}^n \leftarrow \{0,1\}^n$ be a one-one one-way function, and let $x_0, x_1, y_0, y_1 \in \{0,1\}^n$ and $c \in \{0,1\}$. Define the function $f : \{0,1\}^{2n+1} \times \{0,1\}^{2n} \to \{0,1\}^{2n}$ as follows:*

$$f((c, x_0, x_1), (y_0, y_1)) = (x_0 \oplus y_c, x_1 \oplus g(y_{1-c}))$$

The function f does not contain an imbedded Or (every row is a one-one function), but can be shown to be complete in the symmetric model. This leaves room to try and find a computational characterization in this model as well.

Probabilistic Functionalities: Kilian [24] gives combinatorial criteria also for secure evaluation of probabilistic functionalities (rather than deterministic functions). Can computational results be found in this model as well. This seems an interesting direction as this setting seems natural for a computational discussion.

Acknowlegments: We are grateful to Tal Malkin for discussions regarding her work on this subject. We thank Ronen Shaltiel for useful advice regarding hardness amplification.

References

[1] A. Beimel, T. Malkin, and S. Micali. The all-or-nothing nature of two-party secure computation. In *Advances in Cryptology - CRYPTO '99, Lecture Notes in Computer Science*, volume 1666, pages 80–97. Springer, 1999.

[2] M. Bellare and S. Micali. Non-interactive oblivious transfer and applications. In *Advances in Cryptology - CRYPTO '89, Lecture Notes in Computer Science*, volume 435, pages 547–557. Springer, 1989.

[3] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *20th ACM Symposium on the Theory of Computing*, pages 1–10, 1988.

[4] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[5] R. Canetti, E. Kushilevitz, and Y. Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *To apear in Eurocrypt '03*, 2003.

[6] B. Chor and E. Kushilevitz. A zero-one law for boolean privacy. *SIAM Journal on Disc. Math.*, 4(1):36–47, 1991.

[7] C. Crepeau. Equivalence between two flavours of oblivious transfers. In *Advances in Cryptology - CRYPTO '87, Lecture Notes in Computer Science*, volume 293, pages 350–354. Springer-Verlag, 1987.

[8] C. Crepeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions. In *29th IEEE Symposium on Foundations of Computer Science*, pages 42–52, 1988.

[9] I. Damgard, J. Kilian, and L. Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In *Advances in Cryptology - Eurocrypt '99, Lecture Notes in Computer Science*, volume 1592, pages 56–73, 1999.

[10] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, 1985.

[11] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *41st IEEE Symposium on Foundations of Computer Science*, pages 325–335, 2000.

[12] O. Goldreich. Foundations of cryptography - volume 2. Working Draft, available at www.wisdom.weizmann.ac.il/oded/foc-vol2.html, 2002.

[13] O. Goldreich and L.A. Levin. A hard-core predicate for all one-way functions. In *21st ACM Symposium on the Theory of Computing*, pages 25–32, 1989.

[14] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game - a completeness theorem for protocols with honest majority. In *19th ACM Symposium on the Theory of Computing*, pages 218–229, 1987.

[15] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity, or all languages in np have zero-knowledge proof systems. *Journal of the ACM*, 38:691–729, 1 1991.

[16] O. Goldreich, N. Nisan, and A. Wigderson. On yao's xor-lemma. In *Electronic Colloquium on Computational Complexity (ECCC) (50)*, volume 2, 1995.

[17] O. Goldreich and R. Vainish. How to solve any protocol problem - an efficiency improvement. In *Advances in Cryptology - CRYPTO '87, Lecture Notes in Computer Science*, volume 293, pages 73–86. Springer-Verlag, 1987.

[18] R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *36th IEEE Symposium on Foundations of Computer Science*, pages 538–545, 1995.

[19] R. Impagliazzo. A personal view of average-case complexity. In *10th Annual Structure in Complexity Theory Conference*, pages 134–147. IEEE Computer Society Press, 1995.

[20] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *30th IEEE Symposium on Foundations of Computer Science*, pages 230–235, 1989.

[21] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *21st ACM Symposium on the Theory of Computing*, pages 44–61, 1989.

[22] J. Kilian. Founding cryptography on oblivious transfer. In *20th ACM Symposium on the Theory of Computing*, pages 20–31, 1988.

[23] J. Kilian. A general completeness theorem for two-party games. In *23rd ACM Symposium on the Theory of Computing*, pages 553–560, 1991.

[24] J. Kilian. More general completeness theorems for secure two-party computation. In *32nd ACM Symposium on the Theory of Computing*, pages 316–324, 2000.

[25] J. Kilian, E. Kushilevitz, S. Micali, and R. Ostrovsky. Reducibility and completeness in private computations. *SIAM Journal of Computing*, 28(4):1189–1208, 2000.

[26] E. Kushilevitz. Privacy and communication complexity. *SIAM Journal on Disc. Math.*, 5(2):273–284, 1992.

[27] E. Kushilevitz, S. Micali, and R. Ostrovsky. Reducibility and completeness in multi-party private computations. In *35th IEEE Symposium on Foundations of Computer Science*, pages 478–489, 1994.

[28] L. A. Levin. One-way functions and pseudorandom generators. *Combinatorica*, 7:357–363, 1987.

[29] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In *SIAM Symposium on Discrete Algorithms (SODA 2001)*, pages 448–457, 2001.

[30] M. O. Rabin. How to exchange secrets by oblivious transfer. TR-81, Harvard, 1981.

[31] A. C. Yao. Protocols for secure computations. In *23rd IEEE Symposium on Foundations of Computer Science*, pages 160–164, 1982.

[32] A. C. Yao. Theory and application of trapdoor functions. In *23rd IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982.

[33] A. C. Yao. How to generate and exchange secrets. In *27th IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.