Correlated Pseudorandomness and the Complexity of Private Computations

Donald Beaver *

Abstract.

The race to find the weakest possible assumptions on which to base cryptographic primitives such as oblivious transfer was abruptly halted by Impagliazzo's and Rudich's surprising result: basing oblivious transfer or other related problems on a black-box one-way permutation (as opposed to a one-way trapdoor permutation) is tantamount to showing P≠NP. In contrast, we show how to generate OT – in the sense of random number generation – using any one-way function in a black-box manner. That is, an initial "seed" of k OT's suffices to generate $O(k^c)$ OT's.

In turn, we show that such generation is impossible in an information-theoretic setting, thus placing OT on an equal footing with random number generation, and resolving an artificial asymmetry in the analysis of randomness and partially-correlated randomness.

We also initiate a complexity theory of privately-computable probabilistic functions[1] and show that there is a provably rich hierarchy among them. Previous work has considered deterministic functions of possibly-random inputs, and focused on whether reductions exist, the class of primitives that are complete, and the amount of information leaked *vs.* message complexity. We show that any complete boolean function gives rise to a nontrivial complexity hierarchy of privately-computable functions, measured according to invocations of a complete primitive – and that this hierarchy collapses when restricted to "computational" security.

*Transarc Corp., Gulf Tower, 707 Grant St., Pittsburgh, PA 15219. Email: beaver@transarc.com.

[1] Functions mapping inputs to joint distributions.

STOC'96, Philadelphia PA,USA
© 1996 ACM 0-89791-785-5/96/05..$3.50

1 Introduction

Oblivious Transfer, a broadly used primitive introduced by Rabin [Rab81], is a protocol for sending a bit that arrives with precisely 50-50 probability – without the sender knowing the result. This asymmetry in knowledge makes OT a natural basis for achieving security in a wide variety of interactive protocols, ranging from bit commitment to zero-knowledge proofs to multiparty computations to coin tossing, and most of cryptography (*cf.* [GMW87, Kil88]).

Despite the widespread use of OT as a primitive, implementations of OT rely on relatively strong assumptions, such as the existence of trapdoor one-way permutations and the difficulty of factoring or taking discrete logarithms [Rab81, EGL82, BM89, Boe91].

In 1989, Impagliazzo and Rudich showed a remarkable but negative result: basing OT on weaker assumptions would be a difficult task [IR89]. In particular, if there exists an OT protocol that uses a one-way function as a black-box, then P ≠ NP. This result bears strong contrast to pseudorandom number generation, which similarly started with number-theoretic assumptions [BM84] yet was indeed reduced to any one-way function [ILL89].

Expenses and Strong Complexity Assumptions. Imagine that quantum OT devices are finally in mass production, but each bit costs a penny to send. Cryptographers rejoice that complexity assumptions are no longer needed for security, but the price is heavy. Or imagine that the security of known trapdoor one-way permutations has been cast in doubt, or that computing them is as expensive as quantum OT. Meanwhile, tantalizingly cheap one-way functions beckon! But images of Impagliazzo and Rudich stand in their way.

Our work shows how to move past [IR89] by placing OT on an equal footing with pseudorandom number generation. In particular, a short "seed" of initial OT's can be expanded into a polynomially-long sequence of OT's, based only on the existence of a one-way function (used as a black box). In light of the intricacies and high OT cost of general two-party protocols and methodologies, it is somewhat surprising that this can

be achieved *without requiring more OT's for processing than are produced in the end.*

Non-Interactive Oblivious Transfer. Protocols for "non-interactive" (with preprocessing) OT and "non-interactive" zero-knowledge proofs based on it were presented in [BM89, KMO89]. Although remarkably elegant, these methods suffered from correlations among the reception patterns, which were the same in each ZKP (albeit unknown to the prover). If the prover learned whether the verifier rejected, he could infer a few bits of the reception pattern; hence the verifier (and all other verifiers) would immediately have to quit. Thus "non-interactive" would have to be replaced by "never-interactive" for the prover (*i.e.* the prover could never learn the success or failure of a proof) or "universally-interactive" among verifiers.

The bits produced by our method are independent (to any computationally bounded participant), and thus do not suffer these problems. They meet the intuition of a general, computationally-secure "non-interactive oblivious transfer channel." Note that the preprocessing step permits us to escape the impossibility result of [OVY91].

Complexity of Private Computation. Informally, a n-input function f is t-private if there is a protocol for n players to compute f, such that no coalition of at most t players can infer anything more than the value of f. If multi-input function f can be privately computed with access to a black box for g, then f is said to be *reducible* to g. If *any* function is reducible to g, g is said to be *complete* for multi-party private computations.

Starting with Chor and Kushilevitz's work, a long line of research has characterized privately-computable 2-party functions and boolean n-party functions [CK91, Kus92, Bea89, Kil91]. Tradeoffs between privacy and message complexity have been found [BCKO93]. More recently, Kushilevitz, Micali and Ostrovsky identified the class of complete boolean functions, showing that a boolean g is complete if and only if it cannot be computed n-privately [KMO94]. These constructions typically invoke a primitive g a reasonable number of times,[2] in proportion to the circuit complexity of f.

We give lower bounds on the complexity of such reductions and show that in the information-theoretic setting, there is a provably rich hierarchy of private functions, both in the two-party case and the n-party case. In particular, *any* function can be characterized by the minimal number m of OT's needed to evaluate it privately, and for each $m \geq 0$ there is a nonempty set of functions of invocation complexity m. This characterization holds when invocation complexity is measured according to *any complete* (not necessarily boolean) g.

As implied by our first result, the hierarchy collapses when merely *computational* privacy suffices, if a one-way function exists.

Asymmetry and Correlated Pseudorandomness. On a more abstract level, OT can be regarded as a pair of correlated random sources, where the knowledge of the results is asymmetric. Our analysis expands the art of pseudorandom number generation to encompass not merely correlated pseudorandom sources (highly useful in accomplishing tasks such as Byzantine Agreement [Ben83, BS93]) but, more surprisingly, *asymmetrically* correlated pseudorandomness (highly useful in achieving security [Rab81, Kil88, Yao86]), such as OT.

2 Background

We consider three variants of Rabin's pioneering notion of Oblivious Transfer:

1. OT: Rabin's original protocol, in which Alice sends bit b, Bob receives either $(0,0)$ ("failed") or $(1,b)$ ("received b") uniformly at random,but Alice does not know which occurred;
2. $\frac{1}{2}$OT: One-out-of-two oblivious transfer [EGL82], in which Alice has input bits b_0 and b_1, Bob receives (c, b_c) for a random c outside his control, but Alice does not learn c;
3. $\binom{2}{1}$OT: Chosen one-out-of-two oblivious transfer, in which Alice has input bits b_0 and b_1, Bob chooses $c \in \{0,1\}$ and obtains b_c, but Alice does not learn c.

2.1 Security

Computational. In the computational setting, we consider the usual standard of static 1-adversaries. Static 1-adversaries decide in advance to corrupt either Alice or Bob, and may depart from their instructions. The notion of security is simpler to capture than in a more robust model where an adversary might corrupt one or both adaptively.

We follow standard methodology, requiring a simulator to show that Alice gains no information and exerts no undue influence on the output in the case that an adversary corrupts Alice. A second simulator for Bob is also required, to show that Bob is similarly limited. Details are standard and omitted for space (*cf.* [Kil88, GMR89]).

[2] [KMO94] furthermore show that complete boolean g's need be invoked at only a fixed number of different input-tuples (although repeatedly), assuming $x_1, \ldots, x_n$ can be permuted at will.

Information Theoretic. We address a more general scenario for privately-computable functions than previously considered (see Remarks below). In particular, we consider probabilistic functions, namely functions mapping inputs to distributions.

Let $\Pi = (P_1, \ldots, P_n)$ be a set of programs for n parties in a synchronous network having private channels between every pair. Each party is provided an *input* x_i from domain X_i and a *random input* r_i selected from some distribution R_i. At the end of a protocol execution, each party writes an *output* $y_i \in Y_i$. An execution of $\Pi(x_1, \ldots, x_n)$ maps inputs to a distribution on outputs, namely:

$$\Pi : X_1 \times \cdots \times X_n \rightarrow \text{dist}(Y_1 \times \cdots \times Y_n)$$

Let F be a **probabilistic function**, namely a function mapping inputs to distributions on outputs:

$$F : X_1 \times \cdots \times X_n \rightarrow \text{dist}(Y_1 \times \cdots \times Y_n)$$

Nothing prevents F from placing all weight on a single value, thereby behaving like a function with range $Y_1 \times \cdots \times Y_n$. We employ a vector notation $\vec{z}$ for $(z_1, \ldots, z_n)$.

If D_0 and D_1 are distributions having combined support S, respectively, we define the **distance** between them as:

$$\| D_0 - D_1 \| = \frac{1}{2} \sum_{y \in S} |\Pr[D_0 = y] - \Pr[D_1 = y]|.$$

We say that Π computes F **ϵ-correctly** if

$$(\forall x, y) \; \| \Pi(x, y) - F(x, y) \| \leq \epsilon.$$

We address privacy issues in the style of Chor and Kushilevitz [CK91]. Let $S(\vec{x}, \vec{r})$ tabulate the communication among the parties; let S_T be the restriction to those messages sent or received by parties in coalition $T \subseteq \{1, \ldots, n\}$ (called "bad" or "dishonest" players). $S_T(\vec{x}_T, \vec{r}_T, \vec{x}_{\overline{T}})$ can be regarded as a distribution on coalition-seen messages with respect to the random inputs $\vec{r}_{\overline{T}}$ of good players.

Protocol Π is **strongly t-private** if for any coalition T with $|T| \leq t$, for any $\vec{x}_T$, $\vec{x}_{\overline{T}}$, $\vec{z}_{\overline{T}}$ such that $F(\vec{x}_T, \vec{x}_{\overline{T}}) = F(\vec{x}_T, \vec{z}_{\overline{T}})$, we have

$$S_T(\vec{x}_T, \vec{r}_T, \vec{x}_{\overline{T}}) = S_T(\vec{x}_T, \vec{r}_T, \vec{z}_{\overline{T}}).$$

Protocol Π is **weakly (δ, t)-private** if for any coalition T with $|T| \leq t$, for any $\vec{x}_T$, $\vec{x}_{\overline{T}}$, $\vec{z}_{\overline{T}}$ such that $F(\vec{x}_T, \vec{x}_{\overline{T}}) = F(\vec{x}_T, \vec{z}_{\overline{T}})$, we have

$$\| S_T(\vec{x}_T, \vec{r}_T, \vec{x}_{\overline{T}}) - S_T(\vec{x}_T, \vec{r}_T, \vec{z}_{\overline{T}}) \| \leq \delta.$$

Remarks. Although earlier treatment has considered probabilistic protocols Π, the usual goal has been to compute a single-valued function of the inputs [CK91, Kus92, Bea89, Kil91, BCKO93, KMO94]. In other words, the following requirement is typically made:

$$(\forall x, y) \; \Pr[\Pi(x, y) = f(x, y)] \geq 1 - \epsilon,$$

Such treatment does not distinguish those random portions of a player's input that affect the output distribution but which may (in presumably rare cases) lead to complete failure of the protocol.

3 Generating Oblivious Transfer

We approach our solution through a series of refinements. In refinement I, we imagine that a trusted third party, T, were available. Alice constructs two lists of B random bits, $\{r_{i0}\}_{i=1..B}$ and $\{r_{i1}\}_{i=1..B}$, and send them to T. When Alice wishes to execute the i^{th} OT on some bit b_i, she asks T to flip a coin c and send (c, r_{ic}) to Bob. Alice herself flips a coin d and sends $(d, b_i \oplus r_{id})$ to Bob. Bob receives values which he calls (e, f) from T and (g, h) from Alice. If $e = g$ then Bob concludes, "received $f \oplus h$;" else Bob concludes, "failed." In fact, we might as well let Bob choose c, as long as Alice does not learn it.

Of course, if a third party were available, such efforts would be unnecessary: we would simply ask it to perform the desired OT's directly. For refinement II, we invoke a rich variety of results permitting us to replace a third party by an interactive protocol that simulates a circuit evaluation without revealing anything but the results [Yao86, Kil88].

Expressing T's program as a circuit and applying known compilation techniques present no difficulty. But the number of oblivious transfers used by such compilations exceeds the number we wish to generate, as discussed below.

Oblivious Circuit Evaluation. To illustrate the problem, consider the "Yao-gate" approach, a generalization by Goldreich, Micali, and Wigderson of of Yao's methods for two-party circuit evaluation, and further refined elsewhere [Yao86, GMW87, GHY87, GV87, BG89, GL90]. A circuit is composed of a number of wires and gates. The value of a wire is not represented directly as a string or a number, but as the *knowledge* of a secret key. Specifically, Alice chooses and associates two secret keys κ_{i0} and κ_{i1} with wire w_i. If Bob learns κ_{i0}, this means that w_i carries a 0; If Bob learns κ_{i1}, this means that w_i carries a 1. Clearly, the protocol must

ensure that Bob learns precisely one key for each wire. Moreover, the wire keys should have no connection to the wire values themselves.

To evaluate a gate $g(w_i, w_j)$, Bob must learn the appropriate wire key $\kappa_{k,g(w_i,w_j)}$ for the output wire w_k. Fig. 1 illustrates the technique, which we sketch briefly. Alice chooses eight random strings $\alpha_0, \beta_0, \ldots, \alpha_3, \beta_3$. She encrypts or exclusive-or's the strings as illustrated, hiding each output wire $\kappa_{k,g(w_i,w_j)}$ using an α and a β. She enters the results in a table, permutes the *rows* randomly, and gives it to Bob.

If Bob knows κ_{i1} and κ_{j0}, for example, he can decrypt α_2, α_3, β_0, and β_2. Using $\alpha_2 \oplus \beta_2$, he can read $\kappa_{k,g(1,0)}$ from the table. The other entries remain hidden, and the answer has occurred in a random row; thus Bob does not learn anything about the actual wire values themselves.[3,4,5]

In summary, Alice compiles a circuit into a set of such tables and gives it to Bob. The primary difficulty is for Bob to obtain an initial set of keys. Alice might simply tell Bob the appropriate keys for the wires corresponding to her input to the circuit, but this would reveal which keys Bob wants to learn, namely which ones correspond to his input bits. Instead, they engage in one-out-of-two string oblivious transfer: for each wire corresponding to Bob's input, Alice transfers precisely one k-bit key of Bob's choosing, but Alice does not discover which keys Bob chose.

It is this step that defeats our purposes. Alice must transfer at least k bits for each of Bob's input bits. If we are to replace the third party T described above in refinement II, then Bob will need to specify B choices (one y for each transfer), requiring Alice to transfer at least kB bits. Using kB transfers to generate B transfers is hardly a newsworthy accomplishment!

3.1 Adapting OCE to Generate OT

In refinement III, Bob no longer specifies a choice c for each bit. Instead, he sends a k-bit random seed, s, to T. T expands s using a pseudorandom number generator, G.

Refinement IV is simply a compilation of this trusted-party protocol into a two-party protocol. In particular, Alice expresses G as a circuit and performs the compilation described above, with a simple modification. Whenever G would output a bit, c_i, we add an extra gate through which c_i selects either $r_{i,0}$ or $r_{i,1}$. (This simpler gate contains only two rows: $\langle E(\alpha_0, \kappa_{i0}), \alpha_0 \oplus (0, r_{i,0})\rangle$, and $\langle E(\alpha_1, \kappa_{i1}), \alpha_1 \oplus (1, r_{i,1})\rangle$. Bob will learn κ_{i,c_i}, discover α_{c_i}, and deduce (c_i, r_{i,c_i}).)

Alice's inputs to this protocol consist of the two sequences $\{r_{i0}\}_{i=1..B}$ and $\{r_{i1}\}_{i=1..B}$. Bob's inputs consist of the bits of s. Alice learns nothing, while Bob discovers the sequence $\{(c_i, r_{i,c_i})\}_{i=1..B}$, where $G(s) = c_1 c_2 \ldots c_B$.

Theorem 1 *If a one-way function exists, then for any constant $c > 0$, there is a protocol to precompute k^c one-out-of-two oblivious transfers using only k one-out-of-two oblivious transfers. This protocol is secure against static 1-adversaries.*

Proof. Let f be a one-way function; then a cryptographically strong pseudorandom number generator G_f exists that maps K-bit seeds to $B = K^c$ bit sequences [ILL89]. Let C_G be the circuit that takes inputs $\{r_{i0}, r_{i1}\}_{i=1..B}$ from Alice and and s from Bob, computes $G_f(s) = b_1 b_2 \ldots b_B$, and produces $(c_1, r_{1,c_1}), (c_2, r_{2,c_2}), \ldots, (c_B, r_{B,c_B})$ as its output.

Applying the results sketched above (*cf.* [Yao86, Kil88]) to C_G, we obtain a secure implementation of a trusted third party, at a cost of $O(K)$ string $\binom{2}{1}$OT's.

To handle malicious behavior, we implement bit commitment and zero-knowledge proofs based on $f()$ [GMW86, BC86, GMR89, Kil88]. We require that Alice prove in zero knowledge that the cloaked circuit she sent is constructed correctly. For simulation reasons, we also require Bob to commit s before Alice sends the circuit to him. Note that these methods require no invocations of oblivious transfer.

Each string $\binom{2}{1}$OT can be implemented using $O(K)$ $\frac{1}{2}$OT's [BC86, BCR86, BCR86, Cr87]. In total, $O(K^2)$ $\frac{1}{2}$OT's are needed. Setting $K = k^{1/2-\epsilon}$, we obtain the desired result. □

3.2 Storage; Incremental Evaluation

Using the results of [Bea95], the storage requirements are surprisingly small: once the protocol is complete, Alice and Bob each need store only two bits for each future oblivious transfer. In other words, they need not store the full circuit description.

Moreover, the output stream need not be computed to its full length (B). Alice can add extra layers to the simulated circuit later on, without requiring a set of oblivious transfers. In other words, Alice could transmit bits 1024 at a time, by sending the appropriate portion of the cloaked circuit that covers the next window of outputs of G.

[3] He may have known something already, particularly if one wire came from his own input, but he *learns* nothing he couldn't have otherwise guessed *a priori*.

[4] Note that the rightmost columns of the output gates may encode desired output bits directly, rather than a last layer of wire keys $\kappa_{\cdot,\cdot}$, so that Bob can read the output directly.

[5] Note that Bob must be able to deduce when a decryption is successful. There are a multitude of ways to accomplish this; one simple way is to pad the cleartexts with k 0's.

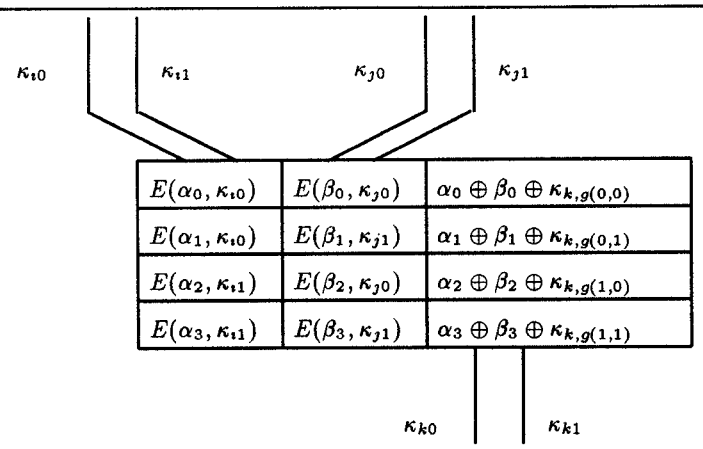

Figure 1: Yao-style gate (before permuting the rows). The κ's are encryption/decryption keys.

Finally, as in the case of pseudorandom number generation, we can imagine using the last group of $O(k)$ transfers to generate an extension to the sequence. This is useful when the polynomial bound on the number of transfers required by a protocol is not explicitly known to the participants.

4 Information Theoretic Impossibility

It is known that OT cannot be generated from scratch when Alice and Bob are computationally unbounded [Kil88]. The preceding discussion raises the possibility that an initial set of transfers might be extended, however.

Indeed, a simplistic entropy argument raises no contradiction. Bob's conditional entropy about Alice's bit b is easily seen to be $H(b|B) = 1/2$. The same conditional entropy is achieved when Alice simply flips a bit b, and then with probability $1/2$ sends b directly to Bob. In a sense, Alice is always sending "half a bit" to Bob. (Note that Alice's knowledge is symmetric with Bob's in the latter case; thus a more refined measure is necessary.)

In fact, generating OT is impossible in an information-theoretic setting, even when both Alice and Bob are honest (but curious). We prove this through a series of lemmas concerning reductions among privately-computable functions. For the moment, we focus on functions of two inputs and on 1-privacy.

We write $F \leq m \times G$ if there exists a strongly private protocol to compute F, making at most m invocations of G as a fundamental (black-box) primitive.

If $f : D \to R$, let $f^{(m)}$ be the function mapping $D^m \to R^m$ defined by:

$$f^{(m)}(x_1, x_2, \ldots, x_m) = (f(x_1), f(x_2), \ldots, f(x_m)).$$

Similarly, if $F = (f_A, f_B)$ is a pair of functions, let $F^{(m)}$ denote $(f_A^{(m)}, f_B^{(m)})$. (Nothing constrains D from itself being a cartesian product $D_1 \times D_2$, or $D_1 \times \cdots \times D_n$, of subdomains which describe each player's input domain.)

Note that there is a difference between $f^{(m)}$, which is a simultaneous evaluation at m inputs, and $m \times f$, which denotes m evaluations at different times.

The "$F \leq m \times G$" notation lends itself to some natural and convenient abuse. For example, "$m \times F \leq l \times G$" should be read to say that any protocol using m invocations of F can be implemented with equal security using l invocations of G instead.[6] Some simple observations support intuitive conclusions:

Lemma 2

(1) $F \leq m \times G$ *and* $G \leq l \times H$ *imply* $F \leq ml \times H$. *We abusively write this as:*

$$F \leq m \times G \leq m \times l \times H \leq ml \times H.$$

(2) $F \leq m \times G$ *implies* $l \times F \leq lm \times G$.

(3) $F^{(m)} \leq m \times F$.

Proof. Part (1) follows by invoking the protocol behind the reduction $G \leq l \times H$, m times sequentially. Part (2) is an application of (1). Part (3) is straightforward: let Alice hold $x = (x_1, x_2, \ldots, x_m)$ and let Bob

[6] This is analogous to the overloading of O-notation in cases like $n = O(n^2) = O(n^3)$.

hold $y = (y_1, y_2, \ldots, y_m)$; invoke $(a_i, b_i) \leftarrow F(x_i, y_i)$ for $i = 1..m$; Alice's output is $a = (a_1, a_2, \ldots, a_m)$, and Bob's is $b = (b_1, b_2, \ldots, b_m)$. □

Note that the converse of (3) is not generally true, reflecting the difference between a batch computation ($F^{(m)}$) and m individual, adaptive computations ($m \times F$). In fact, the converse of (3) fails for Rabin's version of OT, while it holds in both directions for $\binom{2}{1}$OT and $\frac{1}{2}$OT (see [Bea95]).

We proceed with a series of lemmas that show:

Theorem 3 *Let Alice and Bob be computationally unbounded. For any $m \geq 0$, there exists no secure protocol by which Alice and Bob can generate $m+1$ independent one-out-of-two oblivious transfers, when permitted to invoke up to m one-out-of-two oblivious transfers.*

We first consider a protocol for asymmetric AND, a pair of functions that reveal nothing to Alice but provide $x \wedge y$ to Bob:

Definition 1 *Asymmetric AND (AND_B) is a pair of functions (f_A, f_B) each mapping $\{0,1\}^2 \rightarrow \{0,1\}$, defined by $f_A(x,y) = 0$, $f_B(x,y) = x \wedge y$.*

Lemma 4 *$AND_B \leq 1 \times \binom{2}{1}OT$.*

Proof. Let Alice's input be x and Bob's, y. Alice sets $b_0 = 0$ and $b_1 = x$. Bob sets $c = y$. One invocation of $\binom{2}{1}\mathrm{OT}((b_0, b_1), c)$ provides Bob with $b_c = x \wedge y$. □

Lemma 5 *$\binom{2}{1}OT \leq 2 \times AND_B$.*

Proof. Let Alice's input be (b_0, b_1) and Bob's, c. Invoke AND_B twice to evaluate $(\bar{c} \wedge b_0)$ and $(c \wedge b_1)$ with Bob receiving the answer. Note that if $c = 0$ then neither term leaks any information about b_1, while if $c = 1$ then neither term leaks any information about b_0. Bob then calculates $(\bar{c} \wedge b_0) \vee (c \wedge b_1)$. □

For any integer $m \geq 0$, let $G_m = (g_{mA}, g_{mB})$ be a pair of functions, each mapping $\{0,1\}^m \times \{0,1\} \rightarrow \{0,1\}^m$, and defined as follows:

$$g_{mA}(x,y) = 0^m$$

$$g_{mB}(x,y) = \begin{cases} 0^m & \text{if } y = 0 \\ x & \text{if } y = 1 \end{cases}$$

Lemma 6 *For any integer $m \geq 0$, $G_m \leq 1 \times AND_B{}^{(m)}$.*

Proof. If Bob's input y is 0, Bob sets $Y = 0^m$; else he sets $Y = 1^m$. Alice sets X to be her input, x. Invoking $\mathrm{AND}_B{}^{(m)}(X, Y)$ provides Alice and Bob with the desired results. □

The crux of the impossibility result is the following lemma, which shows that m invocations of AND_B are insufficient to solve G_{m+1}.

Lemma 7 *For all integers $m \geq 0$, $G_{m+1} \not\leq m \times AND_B$.*

Proof. Assume by way of contradiction that Π is a protocol that privately computes G_{m+1} with the assistance of up to m invocations of AND_B. We first convert Π into a canonical form by omitting useless invocations of AND_B and modifying others in a particular way.

Consider a given invocation of AND_B. Let $\beta_0 = \Pr[\text{Bob applies } 1 | y = 0]$ and $\beta_1 = \Pr[\text{Bob applies } 1 | y = 1]$. Because the rules of the protocol are known to both Alice and Bob, they both know β_0, β_1. One of three cases must hold. First, if $\beta_0 = \beta_1 = 0$, the AND_B can be omitted, because Bob's output is always 0. Second, if $\beta_0 > 0$, then Alice's input to the AND_B must be independent of x, as must any future message depending on it, thus the AND_B can be omitted. Otherwise, Alice will leak information about her input to Bob even when $y = 0$.

Third, if $\beta_0 = 0$ but $\beta_1 > 0$, observe that $y = 1$ implies Bob is entitled to learn everything that Alice knows (within the protocol). Thus we can require Bob to apply y with probability 1 in such cases, effectively using a new $\beta_1' = 1$. (Technically speaking, when Bob does this, he should thereafter "pretend" with probability $1 - \beta_1$ that he actually applied a 0. Otherwise his "behavior" may be different than in Π.)

Thus, without loss of generality, we may assume that all invocations of AND_B are such that Bob applies y. Moreover, we can assume that precisely m invocations are made, by running extras at the end and calling on Alice to apply 0's and Bob to apply y.

Alice must learn nothing; thus she must be able to generate the open messages in the conversation as well as her inputs to the AND_B invocations. The transcript she generates consists of an open message portion, τ, and an "applied" portion, σ, of length m, which describes the inputs she chooses to apply to the AND_B invocations.

As an alternate way to carry out Π, we can simply have Alice generate (τ, σ). Bob receives τ. If he holds $y = 1$, he is granted σ as well. This procedure clearly gives the same distributions on conversations as Π. But σ has length m; thus, for any τ, Bob can rule out at least $2^{m+1} - 2^m = 2^m$ inputs x, *even if* $y = 0$. Thus Π is insecure. □

Lemma 8 *For all integers $m \geq 0$, $AND_B{}^{(m+1)} \not\leq m \times AND_B$.*

Proof. Assume by way of contradiction that for some integer $m \geq 0$, $\mathrm{AND}_B{}^{(m+1)} \leq m \times \mathrm{AND}_B$. By lemma 6, $G_{m+1} \leq 1 \times \mathrm{AND}_B{}^{(m+1)}$, hence by lemma 2.1, $G_{m+1} \leq m \times \mathrm{AND}_B$. But this contradicts lemma 7. □

We can now prove that it is impossible to generate additional OT's:

Proof (Theorem 3). Because $\binom{2}{1}\text{OT} \leq 1 \times \frac{1}{2}\text{OT} \leq 1 \times \binom{2}{1}\text{OT}$ and such invocations can be precomputed, it suffices to show $(m+1) \times \binom{2}{1}\text{OT} \not\leq m \times \binom{2}{1}\text{OT}$.

Assume by way of contradiction that for some integer $L \geq 0$,

$$(L+1) \times \binom{2}{1}\text{OT} \leq L \times \binom{2}{1}\text{OT}.$$

Repeated invocation of the protocol supporting this reduction gives $(2L+1) \times \binom{2}{1}\text{OT} \leq L \times \binom{2}{1}\text{OT}$. Thus:

$$\begin{array}{lll} \text{AND}_B{}^{(2L+1)} & \leq & (2L+1) \times \text{AND}_B \quad \text{(lemma 2.3)} \\ & \leq & (2L+1) \times \binom{2}{1}\text{OT} \quad \text{(lemma 4)} \\ & \leq & L \times \binom{2}{1}\text{OT} \quad \text{(assumption)} \\ & \leq & 2L \times \text{AND}_B \quad \text{(lemma 5)} \end{array}$$

But this violates lemma 8. □

Theorem 3 also implies the following result for 2-party privacy:

Corollary 9 *Let $\binom{2}{1}OT \leq l \times g$ for some fixed $l \geq 1$. Then for all $m \geq 0$, $(m+1) \times g \not\leq m \times g$.*

4.1 Statistical Privacy

For clarity of presentation, we have focused on strong privacy. Let $\preceq$ denote a $k^{-\omega(1)}$ weakly-private reduction with $k^{-\omega(1)}$ error, where k is an additional, "security" parameter; in other words, a *statistically* private reduction. The preceding arguments apply *mutatis mutandis* with respect to $\preceq$, for $m = k^{O(1)}$.[7] In particular, if the given protocols are weakly $(k^{-\omega(1)}, 1)$-private with $k^{-\omega(1)}$ error, the cumulative error of composing $k^{O(1)}$ protocols is still $k^{-\omega(1)}$. In the proof of lemma 7, replace the $\beta_i = 0$ comparisons by $\beta_i < \delta(k)$ for some $\delta(k) = k^{-\omega(1)}$ exceeding the error rate bound. At worst, this skews Π's results negligibly, by $k^{-\omega(1)}$.[8]

While theorem 3 applies equally well to $\frac{1}{2}$OT, Rabin's OT must be handled with error in mind. A *strongly* private reduction $\binom{2}{1}\text{OT} \leq m \times \text{OT}$ has never been given; the only known protocols leak exponentially-small information or permit exponentially-small error [BC86, BCR86, BCR86, Cr87].

Corollary 10 *(Rabin Impossibility) For all integers $m \geq 0$, $(m+1) \times \text{OT} \not\preceq m \times \text{OT}$.*

[7] Naturally, one can also substitute "$m+1$" and "m" by non-negative integer-valued $M(k)$ and $m(k)$, where $M(k) = k^{O(1)}$, $m(k) = k^{O(1)}$, and $M(k) > m(k)$ a.e..

[8] As a technical observation, it should be noted that we are composing the same subprotocol repeatedly, as opposed to different subprotocols; thus convergence thresholds are uniform.

Proof. Weakly-private reductions found in [BC86, BCR86, Cr87] show that OT variants can be e^{-k}-privately reduced to OT at invocation cost ck, where $c > 0$ is a fixed integer. OT can be reduced to $1 \times \binom{2}{1}\text{OT}$ (or to $1 \times \frac{1}{2}\text{OT}$) with strong privacy. Suppose that it were the case that, for some integer $m \geq 0$, $(m+1) \times \text{OT} \preceq m \times \text{OT}$. Then, repeating $O(k^2)$ times, we obtain $(k+1)(ck) \times \text{OT} \preceq k \times \text{OT}$. Thus $(k+1) \times \binom{2}{1}\text{OT} \preceq k \times \binom{2}{1}\text{OT}$, violating the statistical version of theorem 3. □

5 Complexity of Private Computation

We now turn to private multiparty computations. Let BF denote the set of boolean-output, n-input[9] functions (for $n \geq 2$).

We define a complexity measure based on the number of $\binom{2}{1}$OT's needed to compute a function f t-privately. Let $\text{CPX}_{\text{ot},t}(f)$ be the minimum over the maximal number of $\binom{2}{1}$OT's used by any strongly t-private n-party protocol for f, or ∞ if no such protocol exists.

Using this measure, we examine classes of privately-computable functions sharing the same complexity. For $m \in \mathbf{N}$, let $\text{PF}_{\text{ot},t}(m)$ be the class of functions $f \in \text{BF}$ such that $\text{CPX}_{\text{ot},t}(f) \leq m$.

Theorem 11 *(OT Hierarchy) For all $n \geq 2$ and all $m \geq 0$, $\text{PF}_{\text{ot},n}(m) \subset \text{PF}_{\text{ot},n}(m+1)$, where $\subset$ denotes proper containment.*

Proof. Clearly, $\binom{2}{1}\text{OT}^{(m+1)} \in \text{PF}_{\text{ot},n}(m+1)$. Using theorem 3 and the fact that $(m+1) \times \binom{2}{1}\text{OT} \leq \binom{2}{1}\text{OT}^{(m+1)}$ [Bea95], we conclude $\binom{2}{1}\text{OT}^{(m+1)} \notin \text{PF}_{\text{ot},n}(m)$. □

For the more general case, now consider what happens when $\binom{2}{1}$OT is replaced by an arbitrary, non-n-private function f as the "unit" of complexity. Define the measure $\text{CPX}_{f,t}()$ and the classes $\text{PF}_{f,t}(m)$ with respect to f rather than to $\binom{2}{1}$OT. A rich hierarchy results again:

Theorem 12 *(General Hierarchy) Let $f \in \text{BF}$ be non-n-private. For all $n \geq 2$ and all $m \geq 0$, $\text{PF}_{f,n}(m) \subset \text{PF}_{f,n}(m+1)$.*

Proof. Say $f \in \text{BF}$ and f is not n-private. Assume by way of contradiction that for some integer $L \geq 0$, $\text{PF}_{f,n}(L) = \text{PF}_{f,n}(L+1)$. Then $(L+1) \times f \leq L \times f$.

Following [Kil91, KMO94], an embedded OR is, roughly speaking, a pair of input variables and for each,

[9] On a finite domain.

a pair of values, such that the output on three of the four possible input settings is a "1" (or actually, just the same value), while the output on the fourth setting is a "0" (or actually, just something different from the first value). Intuitively, the four input settings form a rectangular 4-point subset of the domain, on which the outputs essentially describe an OR.

More formally, a two-argument function $g(x_1, x_2)$ *contains an embedded OR* if there exist input values x_1, x_2, y_1, y_2 and two output values $z_1 \neq z_2$, such that $g(x_1, y_1) = g(x_1, y_2) = g(x_2, y_1) = z_1$, while $g(x_2, y_2) = z_2$. In the more general case, one must find a pair of input variables among all n variables, and a fixed setting for the remaining variables, so that the two-variable case obtains. In other words, an n-argument function $g(x_1, \ldots, x_n)$ *contains an embedded OR* if there exist indices i and j (with $1 \leq i < j \leq n$), and fixed input values a_k (for all $k \notin \{i, j\}$), such that the two-argument function $h(x, y)$ defined as $g(a_1, \ldots, a_{i-1}, x, a_{i+1}, \ldots, a_{j-1}, y, a_{j+1}, \ldots, a_n)$ contains an embedded OR.

Now, f is not private, so f contains an embedded OR (see [Kil91] for $n = 2$ case, [KMO94] for $n > 2$ case), thus OR $\leq 1 \times f$. (To apply an n-party protocol for OR to the 2-party case, assign each party $\lceil \frac{n}{2} \rceil$ or $\lfloor \frac{n}{2} \rfloor$ of the original roles, taking care to give the role for variable x_i to one player and x_j to the other.)

According to lemma 5, then, $\binom{2}{1}\text{OT} \leq 2 \times \hat{f}$. By [Kil88] for the 2-party case and various sources [GHY87, GV87, BG89, GL90] for the n-party case, there exists a c_f such that $f \leq c_f \times \binom{2}{1}\text{OT}$. Thus, in fact, $(2Lc_f + 2) \times f \leq L \times f$. Without loss of generality, take $c_f \geq 1$.

Taking these observations together, we have

$$\begin{aligned}(Lc_f + 1) \times \binom{2}{1}\text{OT} &\leq (2Lc_f + 2) \times f \\ &\leq L \times f \leq Lc_f \times \binom{2}{1}\text{OT},\end{aligned}$$

contradicting theorem 11. □

We define the n-party **private function hierarchy** $\text{PFH}_{g,n}$ as the collection $\{\text{PF}_{g,n}(m)\}_{m \in \mathbf{N}}$.

Kushilevitz, Micali and Ostrovsky have characterized the privacy of boolean functions according to whether they are n-private or not [KMO94]. We now characterize privacy according to whether a function supports a rich complexity hierarchy or not.

Theorem 13 *(Complexity Characterization of Boolean Functions) Let $g \in \text{BF}$. Then g is n-private* **iff** *the private function hierarchy based on g collapses to two levels.*

Proof. Say g is n-private. Then every n-private $f \in \text{BF}$ lies in $\text{PF}_{g,n}(0)$, while every non-n-private $f \in BF$ lies in $\text{PF}_{g,n}(\infty)$. Conversely, say g is not n-private. Then theorem 12 shows that $\text{PFH}_{g,n}$ contains infinitely-many different complexity classes. □

Remarks. As in the two-party case, the arguments for strong n-privacy extend to statistical n-privacy as well.

6 Conclusions

In a computationally-bounded setting, we have shown that it is possible to expand an initial sequence of oblivious transfers (of any flavor) to create a polynomially-long sequence, based only on the existence of a one-way function, as opposed to a one-way trapdoor permutation. When the participants are computationally unbounded, on the other hand, the generation of oblivious transfer is not only insecure but, in fact, impossible.

The impossibility of generating oblivious transfer in the unbounded scenario leads to a non-trivial measure of the complexity of evaluating an n-private boolean function, in terms of the number of invocations of a given, complete primitive. Any complete, n-private boolean function gives rise to a rich complexity hierarchy of n-private boolean functions that can be reduced to it.

Because OT is universally complete for private computation [Kil88, GHY87, GV87, BG89, GL90], the various hierarchies can be related by a "constant factor," roughly speaking. It is an interesting open question to determine whether $\binom{2}{1}\text{OT}$ is an indivisible or fundamental unit, *ie.* whether the $\binom{2}{1}\text{OT}$ hierarchy is a refinement of all other hierarchies obtained from n-private boolean functions. It is also of interest to characterize specific computations, such as the Millionaire's Problem, according to the number of oblivious transfers they require.

References

[BCKO93] R. Bar-Yehuda, B. Chor, E. Kushilevitz, A. Orlitsky. "Privacy, Additional Information, and Communication." *IEEE Trans. Info. Theory* **39**:6, November 1993, 1930–1943.

[Bea95] D. Beaver. "Precomputing Oblivious Transfer." *Advances in Cryptology – Crypto '95 Proceedings,* Springer–Verlag LNCS 963, 1995, 97–109.

[Bea89] D. Beaver. "Perfect Privacy for Two-Party Protocols." *Proceedings of the DIMACS Workshop on Distributed Computing and Cryptography,* Princeton, NJ, October 1989.

[BG89] D. Beaver, S. Goldwasser. "Multiparty Computation with Faulty Majority." *Proceedings of the* 30^{th} *FOCS,* IEEE, 1989, 1989, 468–473.

[BS93] D. Beaver, N. So. "Global, Unpredictable Bit Generation Without Broadcast." *Advances in Cryptology - Eurocrypt '93 Proceedings,* Springer–Verlag LNCS 765, 1994, 424–434.

[BM89] M. Bellare, S. Micali. "Non-Interactive Oblivious Transfer and Applications." *Advances in Cryptology - Crypto '89 Proceedings,* Springer–Verlag LNCS 435, 1990, 547–557.

[Ben83] M. Ben-Or. "Another Advantage of Free Choice." *Proc. of* 2^{nd} *PODC,* ACM, 1983.

[BM84] M. Blum, S. Micali. "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits." *SIAM J. on Computing* **13**, 1984, 850–864.

[BC86] G. Brassard, C. Crépeau. "Zero-Knowledge Simulation of Boolean Circuits." *Advances in Cryptology - Crypto '86 Proceedings,* Springer–Verlag LNCS 263, 1987, 223-233.

[BCR86] G. Brassard, C. Crépeau, J. Robert. "All or Nothing Disclosure of Secrets." *Advances in Cryptology - Crypto '86 Proceedings,* Springer–Verlag LNCS 263, 1987, 234–238.

[BCR86] G. Brassard, C. Crépeau, J. Robert. "Information Theoretic Reductions Among Disclosure Problems." *Proceedings of the* 27^{th} *FOCS,* IEEE, 1986, 1986, 168–173.

[CK91] B. Chor, E. Kushilevitz. "A Zero-One Law for Boolean Privacy." *SIAM J. Disc. Math.* **4**:1, February 1991, 36–47.

[Cr87] C. Crépeau. "Equivalence Between Two Flavours of Oblivious Transfers." *Advances in Cryptology - Crypto '87 Proceedings,* Springer–Verlag LNCS 293, 1988, 350–354.

[Boe91] B. den Boer. "Oblivious Transfer Protecting Secrecy." *Advances in Cryptology - Eurocrypt '91 Proceedings,* Springer–Verlag LNCS 547, 1991, 31–45.

[EGL82] S. Even, O. Goldreich, A. Lempel. "A Randomized Protocol for Signing Contracts." *Proceedings of Crypto 1982,* Springer–Verlag, 1983, 205–210.

[GHY87] Z. Galil, S. Haber, M. Yung. "Cryptographic Computation: Secure Fault-Tolerant Protocols and the Public-Key Model." *Advances in Cryptology - Crypto '87 Proceedings,* Springer–Verlag LNCS 293, 1988, 1988, 135–155.

[GMW86] O. Goldreich, S. Micali, A. Wigderson. "Proofs that Yield Nothing but Their Validity and a Methodology of Cryptographic Protocol Design." *Proceedings of the* 27^{th} *FOCS,* IEEE, 1986, 1986, 174–187.

[GMW87] O. Goldreich, S. Micali, A. Wigderson. "How to Play Any Mental Game, or A Completeness Theorem for Protocols with Honest Majority." *Proceedings of the* 19^{th} *STOC,* ACM, 1987, 218–229.

[GV87] O. Goldreich, R. Vainish. "How to Solve any Protocol Problem – An Efficiency Improvement." *Proceedings of Crypto 1987,* Springer–Verlag, 1988, 73–86.

[GL90] S. Goldwasser, L. Levin. "Fair Computation of General Functions in Presence of Immoral Majority." *Proceedings of Crypto 1990.*

[GMR89] S. Goldwasser, S. Micali, C. Rackoff. "The Knowledge Complexity of Interactive Proof Systems." *SIAM J. on Computing* **18**:1, 1989, 186–208.

[ILL89] R. Impagliazzo, L. Levin, M. Luby. "Pseudo-Random Generation from One-Way Functions." *Proceedings of the* 21^{st} *STOC,* ACM, 1989, 1989, 12–24.

[IR89] R. Impagliazzo, S. Rudich. "Limits on The Provable Consequences of One-Way Permutations." *Proceedings of the* 21^{st} *STOC,* ACM, 1989, 1989, 44–62.

[Kil88] J. Kilian. "Founding Cryptography on Oblivious Transfer." *Proceedings of the* 20^{th} *STOC,* ACM, 1988, 1988, 20–29.

[Kil91] J. Kilian. "A General Completeness Theorem for Two-Party Games." *Proceedings of the* 23^{rd} *STOC,* ACM, 1991, 553–560.

[KMO89] J. Kilian, S. Micali, R. Ostrovsky. "Minimum Resource Zero-Knowledge Proofs." *Proceedings of the* 30^{th} *FOCS,* IEEE, 1989, 1989, 474–479.

[Kus92] E. Kushilevitz. "Privacy and Communication Complexity." *SIAM J. Disc. Math.* **5**:2, May 1992, 273–284.

[KMO94] E. Kushilevitz, S. Micali, R. Ostrovsky. "Reducibility and Completeness in Multi-Party Private Computations." *Proceedings of the* 35^{th} *FOCS,* IEEE, 1994, 478–489.

[OVY91] R. Ostrovsky, R. Venkatesan, M. Yung. "Fair Games Against an All-Powerful Adversary." SEQUENCES '91, 1991.

[Rab81] M.O. Rabin. "How to Exchange Secrets by Oblivious Transfer." TR-81, Harvard, 1981.

[Yao86] A. Yao. "How to Generate and Exchange Secrets." *Proceedings of the* 27^{th} *FOCS,* IEEE, 1986, 1986, 162–167.