On the Hardness of Information-Theoretic Multiparty Computation*

Yuval Ishai and Eyal Kushilevitz

Computer Science Department, Technion, Haifa 32000, Israel.
{yuvali,eyalk}@cs.technion.ac.il

Abstract. We revisit the following open problem in information-theoretic cryptography: Does the communication complexity of unconditionally secure computation depend on the computational complexity of the function being computed? For instance, can computationally unbounded players compute an *arbitrary* function of their inputs with polynomial communication complexity and a linear threshold of unconditional privacy? Can this be done using a constant number of communication rounds?
We provide an explanation for the difficulty of resolving these questions by showing that they are closely related to the problem of obtaining efficient protocols for (information-theoretic) private information retrieval and hence also to the problem of constructing short locally-decodable error-correcting codes. The latter is currently considered to be among the most intriguing open problems in complexity theory.

Keywords. Information-theoretic cryptography, secure multiparty computation, private information retrieval, locally decodable codes.

1 Introduction

In STOC 1990, Beaver, Micali, and Rogaway [5] posed the following question:

> Is there a constant-round protocol that allows k computationally unbounded players to defeat a computationally unbounded adversary, while using only a polynomial amount of communication in the total length of their inputs?

This question is still wide open today: it is not known whether all functions admit such a protocol, even in the simple case that the adversary can *passively* corrupt only a *single* player, and even without any restriction on the number of rounds.

A partial answer to the above question was given by Beaver, Feigenbaum, Kilian and Rogaway [4]. They showed that such a round- and communication-efficient protocol exists when the number of players is roughly as large as the

* Research supported in part by a grant from the Israel Science Foundation and by the Technion V.P.R. Fund.

total input size. More precisely, every function f of n input bits can be t-securely computed by $k = O(tn/\log n)$ computationally unbounded players using poly(n) communication complexity and a constant round complexity. Note that this result is meaningless when the number of players is fixed (even when $t = 1$), since it requires the number of players to grow with the input size . This should be contrasted with the fact that, ignoring complexity issues, the optimal security threshold is a constant fraction of the number of players, regardless of the input size. Again, the problem of resolving these difficulties was posed as an open question in [4].[1]

As noted above, if there is no limit on the resources used by the players, then any function f can be computed by k players with a linear threshold of information-theoretic security. This can also be done in a constant number of rounds. However, all general-purpose protocols achieving this have a somewhat unexpected common feature: their *communication complexity* depends on the computational complexity of f (either its circuit complexity if there is no restriction on the number of rounds [7, 10, 12], or its formula- or branching program complexity in the constant-round case [2, 19]). It seems quite unlikely that a purely information-theoretic complexity measure would be so closely linked with computational measures. However, so far there has been no significant negative evidence against this link nor a positive evidence to support it.

The main goal of this work is to establish a close connection between the above questions and other well-known open problems. These problems are discussed below.

Private information retrieval (PIR). A *private information retrieval* (PIR) protocol allows a user to retrieve an item i from a database of size N while hiding i from the servers storing the database. The main cost-measure of such protocols is the *communication complexity* of retrieving one out of N bits of data. There are two main settings for PIR. In the information-theoretic setting [11, 1, 6], there are $k \geq 2$ servers holding copies of the database and the default privacy requirement is that each *individual* server learn absolutely no information about i. In the computational setting for PIR [8, 22, 9] there is typically only a single server holding the database, and the privacy requirement is relaxed to *computational* privacy. While the complexity of PIR in the computational setting is pretty well understood (an "essentially optimal" protocol with polylogarithmic communication can be based on a reasonable cryptographic assumption [9]), the situation is very different in the information-theoretic setting. For any constant k, the best upper bound on the communication complexity of k-server PIR is some fixed polynomial in N, i.e., $O(N^{1/c_k})$ where c_k is a constant depending on k. (The current best bound on c_k is $\Omega(k \log k/ \log\log k)$ [6].) On the other hand, the best known general lower bound on the communication complexity of

[1] These questions should not be confused with another major open problem in information-theoretic MPC: does every *polynomial-time* computable function admit a constant-round protocol which is *computationally efficient*? Our results do not have direct relevance to this question. However, our results do have relevance to the variant of this question which allows the protocols to be computationally inefficient.

k-server PIR is logarithmic in N [23]. Hence, there is an exponential gap between known upper and lower bounds. From now on, the term PIR will refer by default to information-theoretic PIR.

Symmetrically Private Information Retrieval (SPIR). The original PIR model is not concerned with protecting the privacy of the data, and allows the user to learn arbitrary additional information (in addition to the selected bit). The stronger SPIR primitive [15] requires, on top of the PIR requirement, that the user learn no additional information about the database other than the selected bit. This may be viewed as an information-theoretic analogue of $\binom{N}{1}$-Oblivious Transfer. We use SPIR as an intermediate primitive for establishing the connection between PIR and multi-party computation. In doing so, we need to establish a tighter reduction from SPIR to PIR than the one shown in [15].

Locally-decodable codes (LDC). Standard error-correcting codes can provide high fault tolerance while only moderately expanding the encoded message. However, their decoding procedure requires to read the entire encoded message even if one is only interested in decoding a single bit of this message. LDC simultaneously provide high fault tolerance and a sublinear-time "local" decoding procedure. To make this possible, the decoding procedure must use randomness for selecting which bits to probe, and some error probability must be tolerated. More formally, a code $C : \{0,1\}^N \to \Sigma^M$ is said to be (k, δ, ϵ)-*locally decodable* if every bit x_i of x can be decoded from $y = C(x)$ with success probability$\geq 1/2+\epsilon$ by reading k (randomly chosen) symbols of y, even if up to a δ-fraction of the symbols in y were adversarially corrupted. The main complexity question related to LDC is the following: Given a constant number of queries k, what is the minimal length $M(N)$ of a (k, δ, ϵ)-LDC? In studying this question, one typically requires δ, ϵ to be bounded by some fixed constants (independently of N). However, the problem appears to be as difficult even if δ, ϵ are sub-constant (say, $\delta, \epsilon = 2^{-\log^c n}$) as long as they are not exponentially small.

Katz and Trevisan [20] have shown an intimate connection between this question and information-theoretic PIR. In particular, a k-server PIR protocol in which the user sends $\alpha(N)$ bits to each server and receives $\beta(N)$ bits in return can be used to construct a k-query LDC of length $O(k2^{\alpha(N)})$ over $\Sigma = \{0,1\}^{\beta(N)}$. Accordingly, the best upper bound on the length of a k-query LDC is exponential in N and the best general lower bound is polynomial in N [20]. The question of obtaining stronger lower bounds for LDC has recently received a significant amount of attention [20, 17, 13, 27, 21], and progress on this question appears to be very difficult.

1.1 Our Results

We prove that the problem of obtaining communication-efficient constant-round protocols for arbitrary functions is closely related to the problem of obtaining communication-efficient PIR protocols. Relying on known connections between PIR and locally decodable codes [20], we obtain a similar connection between the communication complexity of unconditionally secure multiparty computation and the length of locally decodable codes. In particular, strong negative results for the former problem would imply strong negative results for the latter, which so far seem elusive.

The above high-level statements hide some subtleties. By default, we will view the number of players as constant, and measure the complexity of protocols in terms of their input size. Hence, by referring to the existence of communication-efficient protocols with a linear security threshold we mean the following: there exists a constant $c < 1/2$ such that for all k there exists a polynomial $p(\cdot)$ (possibly depending on k) such that for all functions $f : \{0,1\}^n \to \{0,1\}$ there exists a k-player $\lfloor ck \rfloor$-private protocol that computes f with $p(n)$ communication.[2]

Also, the term "security" refers here to security against honest-but-curious, computationally unbounded players (or equivalently a passive, unbounded external adversary).

With the above terminology in hand, we can now informally state our main results (which are actually special cases of more general theorems). The first of these results connect between the existence of very efficient PIR protocols and the existence of communication-efficient multiparty computation (MPC):

- (from PIR to MPC) If there exists a 1-round, polylog communication, PIR with a constant number of servers then there exist communication-efficient, statistically private, constant-round, multiparty protocols with a linear privacy threshold.
  Moreover, if the PIR protocol that we start with is so-called *linear* then this transformation yields perfect multiparty protocols.
- (from MPC to PIR) If there exist communication-efficient multiparty protocols with a linear privacy threshold then there exists polylog communication PIR with a constant number of servers. Moreover, this transformation maintains the number of rounds.

Using the above results, combined with the connections between PIR and locally decodable codes mentioned above, we get the following additional corollaries:

- (from LDC to MPC) If there exist constant-query LDCs of quasi-polynomial length and alphabet of quasi-polynomial size then there exist communication-efficient, statistically private, constant-round, multiparty protocols with a linear security threshold.
- (from MPC to LDC) If there exist communication-efficient multiparty protocols with a linear privacy threshold then there exists a constant-query LDC of quasi-polynomial length, quasi-polynomial size alphabet and parameters ϵ, δ which are $1/\text{quasipoly}(N)$. (It should be noted that all currently known LDC with these parameters are of exponential size (i.e., $2^{N^{\Omega(1)}}$); therefore, codes with quasi-polynomial parameters, as those mentioned here, are considered non-trivial.)

[2] Here and in the following, the n input bits of f may be arbitrarily partitioned between the k players.

To conclude, strong (upper or lower) bounds on the communication complexity of MPC should be roughly as difficult as strong bounds on LDCs, up to some loss in the achieved parameters.

1.2 Related Work

There is a vast literature on secure computation in the information-theoretic setting and on private information retrieval. However, most related to the current work are [4] and [24].

As noted above, [4] obtain communication-efficient protocols for arbitrary functions, whose security threshold decreases almost linearly with the input size. Their protocol was related to constructions of locally-random reductions [3], which in turn are related to PIR. However, the protocol of [4] made a heavy use of special "easiness" properties of the underlying locally-random reductions, and thus did not provide an indication that a more general relation exists.

Naor and Nissim [24] study the question of turning a communication-efficient two-party protocol into a secure one without incurring a significant communication overhead. In doing so, they make use of an idealized $\binom{N}{1}$-OT, which in turn (using reductions from [25, 14]) can be based on single-server PIR with polylogarithmic communication. However, in the two-party setting considered in [24] our main result becomes trivial, as the secure computation of an arbitrary function reduces to a single table lookup.

Organization: In Section 2 we provide some necessary definitions and notation. Section 3 deals with transforming PIR protocols into SPIR protocols and Section 4 with transforming the latter into MPC protocols. Section 5 describes a construction of PIR protocols from multiparty protocols. Finally, in Section 6 we discuss the relation between LDC and PIR.

2 Preliminaries

In this section we sketch the definitions of the main primitives considered in this work. Since these are very basic and well known primitives, the purpose of this section is mainly to set up the notation and terminology used in this paper. For more detailed definitions the reader is referred to the relevant literature.

2.1 MPC

A secure multiparty computation (MPC) protocol allows a set of k players $\mathcal{P}_1, \ldots, \mathcal{P}_k$ to compute some function f of their local inputs while hiding the inputs from each other. By default, we consider functions $f : (\{0,1\}^n)^k \to \{0,1\}$. When computing such a function, each player $\mathcal{P}_i$ holds an n-bit input $a_i \in \{0,1\}^n$, and all players output $f(a_1, \ldots, a_k)$. Our results easily extend to more general types of functionalities (e.g., allowing non-boolean outputs and different outputs to different players).

In this work we consider MPC in the *pure information-theoretic setting*, where both the legitimate players running the protocol and the adversary attacking it have unlimited computational resources. We restrict our attention to security against a *passive* adversary (or honest-but-curious players), also referred to as *privacy*. In this setting, a k-party protocol is said to t-*privately* compute f (where $1 \leq t \leq k$) if the following requirements are met:

- **Correctness.** The players always output the correct output $f(a_1, \ldots, a_k)$.
- t**-privacy.** The view of any set B of at most t players depends only on the inputs of the players in B and the output of the function. That is, on any two input vectors $\boldsymbol{a}, \boldsymbol{a}'$ such that $\boldsymbol{a}_B, \boldsymbol{a}'_B$ and $f(\boldsymbol{a}) = f(\boldsymbol{a}')$, the view of players in B is identically distributed.

The above perfect correctness requirement can be naturally relaxed to ϵ-correctness, allowing the output to be incorrect with probability ϵ. Similarly, the perfect privacy requirement can be relaxed to (t, ϵ)-privacy, requiring that for any set B of at most t players the distributions of its view on any two inputs vectors $\boldsymbol{a}, \boldsymbol{a}'$ as above are in statistical distance of at most ϵ. Moreover, it is convenient to assume that the above ϵ-privacy requirement hold given *every* choice of the random inputs of players in B.[3]

While the case of perfect MPC is the more interesting one and is the one usually considered in the literature, some of our transformations will only yield non-perfect protocols. In all such cases, ϵ can be made negligible in n.

2.2 PIR

Private Information Retrieval (PIR) schemes are protocols for $k+1$ parties: *servers* $\mathcal{S}_1, \ldots, \mathcal{S}_k$, which are given an N-bit string $x \in \{0,1\}^N$ as input (sometimes referred to as a *database*), and a *user* $\mathcal{U}$, which is given as input an index $i \in [N]$. A PIR protocol allows communication between the user and the servers; we assume, without loss of generality, that the servers do not communicate with each other directly.[4] The goal of the protocol is for the user to learn the value x_i while, at the same time, keeping i private. This is captured by the following requirement.

User-privacy: Denote by $V_j[x, i]$ the random variable containing the *view* of server $\mathcal{S}_j$ in the protocol when the database is x and the user wishes to retrieve x_i. *User-privacy* requires that, for any server $\mathcal{S}_j$, the view V_j is independent of i (i.e., for all x, i, i' the views $V_j[x, i]$ and $V_j[x, i']$ are identically distributed). We will also consider a relaxed variant, termed ϵ-*PIR*, in which we only require that

[3] This assumption is without loss of generality, since there is at most an $\sqrt{\epsilon}$-fraction of the random input choices given which the distance is larger than $\sqrt{\epsilon}$. For sufficiently small ϵ, these bad choices can be eliminated without significantly altering the protocol's behavior.

[4] Since we are interested in the honest-but-curious setting, and since there is no privacy requirement with respect to the user, communication between the servers can always be done with the help of the user.

the statistical distance between $V_j[x,i]$ and $V_j[x,i']$ be bounded by ϵ. The latter requirement will be referred to as ϵ-user-privacy.

The complexity of PIR schemes is measured mainly by their *communication complexity*. We denote by $\alpha(N)$ the total number of bits sent in the protocol from the user to the servers, by $\beta(N)$ the total number of bits sent from the servers to the user, and by $m(N)$ the total communication (in either direction).

2.3 SPIR

Symmetrically Private Information Retrieval (SPIR) schemes are PIR schemes that satisfy an additional *data-privacy* requirement, guaranteeing that the only information obtained by the user in the protocol is the intended output x_i:

Data-privacy: Denote by $V_U[x,i]$ the random variable which is the *view* of the user in the protocol where the servers hold database x the and user's input is i. We require that, for all i and for all strings x, x' such that $x_i = x'_i$, the views $V_U[x,i]$ and $V_U[x',i]$ are identically distributed. We will also consider a relaxed variant, termed *ϵ-SPIR*, in which we require that the statistical distance between these two views be bounded by ϵ and, as in the case of PIR, also allow ϵ-user-privacy.

It should be noted that in the literature (see [15]) information-theoretic SPIR is discussed in a setting where all servers share a common random string (CRS) which is unknown to the user. This assumption is necessary if no direct communication between the servers is allowed. In contrast, the use of SPIR in this paper cannot allow the servers to share a CRS. We therefore allow servers in a SPIR protocol to directly communicate with each other.

Note that SPIR in this setting can also be viewed as a special case of MPC: the MPC consists of $k+1$ players, the user and the k servers, whose inputs are restricted to so that all servers hold an identical input x, and whose privacy constraints are those obtained by setting $t=1$ in the formal definitions of MPC. A similar view can be taken with respect to PIR, except that here the privacy constraint for the user is removed.

3 From PIR to SPIR

In this section we show how to transform a (perfect or non-perfect) PIR scheme with communication complexity $m(N)$ into an ϵ-SPIR scheme with communication complexity $\text{poly}(m(N))$ (in the model where no CRS is available). This transformation maintains the number of rounds[5] but has a small penalty of increasing the number of servers from k to $k+1$.

A good starting point for presenting our transformation is to recall the transformation of [15], obtaining SPIR with perfect data-privacy in the case where a

[5] In the context of PIR, a round is an exchange of messages from the user to the servers and back. In the context of SPIR we also allow, in parallel, a communication between the servers.

CRS is available. Its main disadvantage from our point of view is that the CRS in use is very long, and so modifying it to the setting with no CRS does not seem obvious. We will show, however, that such a modification can still be done. We therefore start with the solution from [15]; it assumes a CRS denoted r of length N that is available to $k+1$ servers $\mathcal{S}_1, \ldots, \mathcal{S}_k, \mathcal{S}_{k+1}$.

1. The user $\mathcal{U}$ picks a random shift $\Delta \in [N]$ and sends it to the servers $\mathcal{S}_1, \ldots, \mathcal{S}_k$.
   The user also sends the shifted index $i+\Delta$ to $\mathcal{S}_{k+1}$ (here and below, whenever an index is larger than N it should be understood that N is subtracted from it).
2. $\mathcal{U}, \mathcal{S}_1, \ldots, \mathcal{S}_k$ execute the assumed PIR scheme where $\mathcal{U}$ uses i as its input and the servers use $y = x \oplus (r \ll \Delta)$ as their input. This scheme allows $\mathcal{U}$ to compute $y_i = x_i \oplus r_{i+\Delta}$ (but may potentially leak additional information about y).
   $\mathcal{U}$ also receives from $\mathcal{S}_{k+1}$ the bit $r_{i+\Delta}$. It xors this bit with y_i to obtain x_i.

Intuitively, user-privacy follows from the fact that the view of each of $\mathcal{S}_1, \ldots, \mathcal{S}_k$ is exactly as in the PIR protocol and the view of $\mathcal{S}_{k+1}$ consists of a random index (independent of i). Data-privacy follows from the fact that y is uniformly distributed in $\{0,1\}^N$ and that the only bit of r which is available for $\mathcal{U}$ is $r_{i+\Delta}$. This intuition is formally proved in [15]. The communication complexity of the above SPIR protocol is dominated by the communication complexity of the PIR. The round complexity also remains unchanged (note that Step 1 can be executed in parallel to the first message of Step 2).

Next, we wish to modify the transformation to work in the setting where no CRS is available. A natural approach is to let the server $\mathcal{S}_{k+1}$ choose the string $r \in_R \{0,1\}^N$, distribute it among all other servers (but not the user) and then run the protocol above. While this modification still respects both user-privacy and data-privacy, the communication complexity grows by $k \cdot N$ (since the length of r is N) and hence it makes the whole protocol useless for our purposes.

To overcome this, we will show the existence of a "small" set of strings $\mathcal{R} \subset \{0,1\}^N$ that "fools" the protocol; namely, the user's views obtained in the modified protocol in which $\mathcal{S}_{k+1}$ picks $r \in_R \mathcal{R}$ are statistically close to those obtained in the protocol above. The overhead of this transformation will only be $k \cdot \log|\mathcal{R}|$ (rather than $k \cdot N$), which will be small enough. However, the transformation will no longer obtain *perfect* data-privacy. The theorem that we prove is as follows:

Theorem 1. *Fix $k \geq 2$ and $\epsilon > 0$. Assume that there exists a k-server PIR protocol $\mathcal{P}$ with communication complexity $m(N)$ and round complexity $d(N)$ that satisfies ϵ_1-user-privacy, for some $\epsilon_1 \geq 0$. Then, there exists a $(k+1)$-server SPIR protocol $\mathcal{P}'$ with communication complexity $O(m(N) + \log(1/\epsilon))$ and round complexity $d(N)$ that satisfies ϵ_1-user-privacy and ϵ-data-privacy.*

The rest of this section is organized as follows. We first formalize a technical lemma about the existence of a set $\mathcal{R}$ as needed. Then, based on this lemma, we present and analyze the modified transformation from PIR to SPIR. Finally, we prove the lemma (this is a fairly standard proof, in complexity theory contexts, that uses a probabilistic argument and is given here for the sake of completeness).

Let $\mathcal{R} \subseteq \{0,1\}^N$ and let $C : \{0,1\}^N \to [M]$ be a function. Denote by $C(\mathcal{R})$ the random variable obtained by applying C to a random element of $\mathcal{R}$ and by $C(U)$ the random variable obtained by applying C to a uniformly random N-bit string. We say that $\mathcal{R}$ ϵ-fools the function C if the statistical distance between $C(\mathcal{R})$ and $C(U)$ is bounded by ϵ. Let $\mathcal{C}$ be a family of functions. We say that $\mathcal{R}$ ϵ-fools $\mathcal{C}$ if it ϵ-fools every function $C \in \mathcal{C}$.

Lemma 1. *Let $\mathcal{C}$ be a family of functions from $\{0,1\}^N$ to $[M]$ and let $\epsilon > 0$. Then, there exists a set $\mathcal{R}_{\mathcal{C}} \subset \{0,1\}^N$ of size* $\mathrm{poly}(1/\epsilon, M, \log|\mathcal{C}|)$ *that ϵ-fools $\mathcal{C}$.*

It should be noted that we will apply the above claim with $\mathcal{C}$ which is significantly smaller than the set of all M^{2^N} functions. Also note that $\mathcal{R}_{\mathcal{C}}$ may depend on $\mathcal{C}$; obviously, there can be no single $\mathcal{R}$ that is good for all families $\mathcal{C}$, even if $\mathcal{C}$ can only contain a single function.

We defer the proof of the lemma and now describe the modified transformation. We are given a PIR protocol $\mathcal{P}$, and assume for now that $\mathcal{P}$ is a perfect, one-round protocol (which is the case for all known PIR protocols; the multi-round case will be discussed in Remark 1 below and the non-perfect case in Remark 2 below). The protocol starts by server $\mathcal{S}_{k+1}$ picking $r \in_R \mathcal{R}$, from a carefully chosen $\mathcal{R}$ (specified below) and sending its index ($\log|\mathcal{R}|$ bits) to all other servers. The SPIR protocol then proceeds as the SPIR protocol described above.

User-privacy is easy to argue, independently of the choice of $\mathcal{R}$; indeed, user-privacy in the original transformation holds for *any* choice of r, in particular for all $r \in \mathcal{R}$. To argue the data-privacy, we first have to define the set $\mathcal{R}$. For this, we define a family of functions $\mathcal{C}$ that our set $\mathcal{R}$ will be able to fool. Fix some database $x \in \{0,1\}^N$ and a sequence of queries $q = (q_1, \ldots, q_k, q_{k+1})$ that may be sent in our protocol from the user to the $k+1$ servers. Let $C_{x,q}(r)$ be the function that returns the sequence of all answers that the user gets from the servers, as a function of r, when the database is x and its queries were q. Let $\mathcal{C}$ be the family of all functions $C_{x,q}(r)$, parameterized by the choice of x and q. Note that the length of the queries is bounded by $\alpha(N)$ and the length of the answers is bounded by $\beta(N)$ (it is therefore convenient to set $M \stackrel{\text{def}}{=} 2^{\beta(N)}$). Also note that the size of $\mathcal{C}$ is $2^N \cdot 2^{\alpha(N)}$. For this family $\mathcal{C}$, we pick $\mathcal{R} = \mathcal{R}_{\mathcal{C}}$ as promised by Lemma 1. This choice of $\mathcal{R}$ guarantees that the view seen by the user (which is determined by $C_{x,q}(r)$) is ϵ-close if r is truly random or if $r \in_R \mathcal{R}$. Hence, by the perfect data-privacy of the original transformation, we get ϵ-privacy of the modified transformation.

Finally the communication complexity consists of the communication complexity of the original PIR (which is $m(N) = \alpha(N) + \beta(N)$), the communication between the user and $\mathcal{S}_{k+1}$ (which is $\log N + 1$ bits) and the cost of sending r from $\mathcal{S}_{k+1}$ to all other servers (which is $k \cdot \log|\mathcal{R}_{\mathcal{C}}| = O(\log 1/\epsilon + \log M + \log\log|\mathcal{C}|) =$

$O(\log 1/\epsilon + \beta(N) + \alpha(N) + \log N) = O(m(N) + \log 1/\epsilon))$. This implies that the communication overhead of the transformation is fairly small.

Proof of Lemma 1: We prove the lemma by picking at random a set $R \subset \{0,1\}^N$ of w strings (each is chosen uniformly and they are all independent). To prove the lemma, it suffices to show that for all $C \in \mathcal{C}$ no (statistical) distinguisher can distinguish between the random variables $C(\mathcal{R})$ and $C(U)$ with more than an ϵ-advantage, where a (statistical) distinguisher is just a subset $T \subset [M]$ of all possible outputs. For this, we first fix some C and T and bound from above the probability that, for a random $\mathcal{R}$, the distinguisher can tell apart $C(\mathcal{R})$ from $C(U)$. Namely, for some "small" δ we wish to prove that

$$\Pr_{\mathcal{R}}\left[|\Pr(C(U) \in T) - \Pr(C(\mathcal{R}) \in T)| > \epsilon\right] \leq \delta.$$

Let $p \stackrel{\text{def}}{=} \Pr(C(U) \in T)$. Therefore, we need to prove that when sampling w times a binomial distribution that gives 1 with probability p, the probability that the average will deviate from p by more than ϵ is bounded by δ. This kind of bounds is given by Chernoff bounds. Specifically, it can be shown that if $w = \text{poly}(1/\epsilon, \log(1/\delta))$ then this probability is indeed bounded by δ. Now, if we set $\delta < 1/(|\mathcal{C}| \cdot 2^{|M|})$ it follows by a union bound argument that there exists a choice of $\mathcal{R}$ such that for all 2^M distinguishers, and for each of the $|\mathcal{C}|$ functions $C \in \mathcal{C}$, we have $|\Pr(C(U) \in T) - \Pr(C(\mathcal{R}) \in T)| \leq \epsilon$, as needed. The size of this $\mathcal{R}$ is $w = \text{poly}(1/\epsilon, M, \log|\mathcal{C}|)$, as needed. ■

Remark 1. We dealt above with the case that the PIR scheme $\mathcal{P}$ is a one-round scheme. We outline here how a similar construction can be applied in the case where $\mathcal{P}$ is a multi-round PIR. Essentially, we apply the same transformation as above; we just need to re-define the set of functions $\mathcal{C}$ and as a result the set $\mathcal{R}_{\mathcal{C}}$ that fools these functions. The set $\mathcal{C}$ is defined by the collection of all functions $C_{x,q}$ as before, except that this time q includes all the communication sent by the user in all rounds and $C_{x,q}(r)$ returns all the answers sent by the servers over all rounds. $\mathcal{R}_{\mathcal{C}}$ is now defined by applying the lemma to this $\mathcal{C}$ and with $\epsilon' = \epsilon/2^{\alpha(N)}$. We claim that the resulting SPIR protocol, $\mathcal{P}'$, is indeed ϵ-private. Suppose to the contrary that there is a distinguisher T that participates in $\mathcal{P}'$ and can tell, with advantage more than ϵ, whether r is chosen from U or from $\mathcal{R}$. We argue that this allows us to construct a distinguisher T' that can tell $C(R)$ from $C(U)$, for some C, with advantage better than ϵ', contradicting the choice of $\mathcal{R}$. The distinguisher T' works by guessing q, i.e. guessing all the messages sent by the user over all rounds of the protocol (a total of $\alpha(N)$ bits), randomly picking the user's random input, and asking to see the value of $C_{x,q}(r)$. (In case where the servers in $\mathcal{P}$ are randomized, the latter should also depend on their uniformly chosen random inputs.) If the answers are consistent with the queries guessed by T', it applies T to guess whether r comes from U or from $\mathcal{R}$; otherwise, it just guesses at random. The advantage of T' in this guess is $1/2^{\alpha(N)}$ (the probability of guessing q correctly) times the advantage of T. Note that even though $\epsilon' \ll \epsilon$, since the communication grows by $\log |\mathcal{R}|$ the effect of using ϵ' rather than the original ϵ is just an additive factor of $\alpha(N)$.

Remark 2. The same transformation, as described above, can be applied to an ϵ_1-PIR. The user-privacy of the SPIR that we obtain remains as in the PIR (i.e., ϵ_1) and the data-privacy has a parameter ϵ.

Remark 3. It is important to note that our transformation is inherently non-perfect. However, we point out that there is an important special case in which an alternative *perfect* transformation can be presented; this is the case of *linear* PIR (or LPIR, for short). LPIR is a variant of PIR discussed in the literature; it is a one-round protocol where the servers' answers are viewed as vectors in a space F^{β}, and the user computes its output x_i by taking a linear combination of the k answers, whose coefficients may depend on i and on the user's random input. All information-theoretic PIR schemes from the literature are linear in this sense. The perfect transformation is now obtained as a combination of two facts: The first is that any k-server LPIR protocol with $m(N)$ communication can be transformed into a linear $2k$-server protocol with query length $m(N)$ in which the user outputs the sum of the $2k$ answers [16] (see [6] for details). The second is the existence of a simple MPC protocol to privately compute the sum of k elements in F with $O(k)$ communication and two rounds. Our transformation for this case will therefore work as follows: given the LPIR protocol, we construct the protocol with short answers but instead of the servers sending their answers to $\mathcal{U}$ they will invoke the private protocol for computing x_j in a way that only $\mathcal{U}$ will learn the result. In fact, it is possible to avoid doubling the number of servers by replacing the second step above with a private multiparty protocol for the following function. The input of each server is its answer to the user's query in the LPIR protocol. The user's input is a vector u representing the linear combination of the servers' answers which is needed to reconstruct x_i (note that u should remain private, as it may depend on i). The function should return the value of x_i, which is a degree-2 polynomial in the inputs. Note that, due to the easiness of the above function, it can be efficiently computed using standard MPC protocols (e.g., [7]).

4 From SPIR to MPC

In this section we show how to construct, based on a one-round k-server SPIR, constant-round, 1-private multiparty protocols for $k' = k^2 + 2$ players that can compute *any* function f. If the communication complexity of the SPIR is $m(N)$ then the communication complexity of the multiparty protocols will be $\text{poly}(m(N))$. If the SPIR protocol is only ϵ-private then the MPC protocol is $O(\epsilon)$-private (where as usual, k is viewed as a constant).

Let $\mathcal{P}$ be the given SPIR protocol. Denote the k^2+2 players of the multiparty protocol by $\mathcal{S}_{i,j}, i, j \in [k]$ and $\mathcal{P}_1, \mathcal{P}_2$. Also assume, without loss of generality, that in the given function f only $\mathcal{P}_1, \mathcal{P}_2$ have inputs [6]. We therefore denote the input of these two players by a_1, a_2 and the desired output by $f(a_1, a_2)$. Intuitively, the protocol views the function as a table F of size $N \times N$ where $N \stackrel{\text{def}}{=} 2^n$. The goal is for, say, $\mathcal{P}_1$ to retrieve the (a_1, a_2) index of this table, which is just the desired $f(a_1, a_2)$. The MPC protocol proceeds as follows:

1. Player $\mathcal{P}_1$ applies the SPIR protocol with index $a_1 \in [N]$ to generate queries $q_1, \ldots, q_k$. It sends each query q_i to all players $\mathcal{S}_{i,j}, j \in [k]$ (i.e., each query is sent to k players; intuitively, this is done to create the replication needed in the next step of the protocol).[7]
   Each player $\mathcal{S}_{i,j}$, upon receiving the query q_i, computes (but does *not* send) the answer in the SPIR protocol to the query q_i if the database was F_{a_2}, the a_2-th column of the table F; since the actual value of a_2 is not known to $\mathcal{S}_{i,j}$ it does so for all possible values $a_2 \in [N]$ hence obtaining a vector A_i consisting of all N answers to q_i (each is a $\beta(N)$-bit string). In particular, note that each A_i is therefore replicated among k players.
2. Player $\mathcal{P}_2$ applies the SPIR protocol with index $a_2 \in [N]$ to generate queries $q'_1, \ldots, q'_k$. It sends each query q'_j to all players $\mathcal{S}_{i,j}, i \in [k]$.
   Each player $\mathcal{S}_{i,j}$, upon receiving the query q'_j, computes the answer it would give in the SPIR protocol, when its database is A_i (as computed in the previous step).[8] It sends this answer, $b_{i,j}$ to $\mathcal{P}_2$.
3. Upon receiving the answers $b_{i,j}$, the player $\mathcal{P}_2$ does the following: It uses, for each i, the k answers $b_{i,j}, j \in [k]$ to obtain the a_2-th block of A_i (for this it applies the reconstruction procedure as in the SPIR protocol). By definition of A_i, this block contains the answer given in the SPIR protocol to the query q_i on database F_{a_2}. Denote this answer by b_i.
   $\mathcal{P}_2$ sends the reconstructed information $b_1, \ldots, b_k$ (total of $\beta(N)$ bits) to $\mathcal{P}_1$ who can now also apply the reconstruction procedure of the SPIR protocol to construct the a_1-th entry of F_{a_2}; $\mathcal{P}_1$ sends this value to all other players. This, by definition, is exactly $f(a_1, a_2)$, as needed.

The communication complexity of the above protocol is bounded by the communication complexity of applying the SPIR protocol $k+1$ times. Once, initiated by $\mathcal{P}_1$, on databases of length $N = 2^n$ but repeated k times (hence its

[6] In the general case where all players have inputs we simply add a preliminary step where each player $\mathcal{S}_{i,j}$ shares its input between $\mathcal{P}_1, \mathcal{P}_2$. Then, we proceed as in the case where only these two players have an input, where the input for each of $\mathcal{P}_1, \mathcal{P}_2$ consists of its original input together with the shares received from other players

[7] If the SPIR protocol requires also communication among the servers then this is done in parallel to the described step.

[8] SPIR (as well as PIR) is defined above to allow the retrieval of a single bit. However, both primitives have a standard extension that deals with the retrieval of "blocks" [11]: the user sends one set of queries and the servers answer them by considering the blocks in a bitwise manner. If the blocks are of length ℓ then the query complexity of this solution, $\alpha(N)$, remains unchanged and the answer complexity, $\beta(N)$ grows by a factor of ℓ. Since A_i consists of blocks of $\ell = \beta(N)$ bits then this extension is needed here.

cost is $O(m(N))$) and the others, initiated by $\mathcal{P}_2$, for k retrievals of $\beta(N)$-bit blocks (hence its cost is $O(m(N)\cdot\beta(N))$).[9] The total communication complexity is therefore poly$(m(N))$, as needed.

We turn to the 1-privacy of the protocol. Informally, we make the following observations: (1) player $\mathcal{P}_1$ has the same view as the user has in the first invocation of the SPIR protocol and hence from the ϵ-data-privacy of the SPIR follows the ϵ-privacy of the protocol for computing f, with respect to player $\mathcal{P}_1$. (2) for each $i \in [k]$, player $\mathcal{P}_2$ has the same view as the user has in a SPIR protocol for constructing the block b_i from A_i. Also note that $b_1,\ldots,b_k$ may give information on $f(a_1,a_2)$ (and may even determine it completely); however, this information is legal since this is the output of the protocol (and nothing more). By the ϵ-data-privacy of the SPIR it follows that the protocol for computing f is $(k\cdot\epsilon)$-private with respect to player $\mathcal{P}_2$ (which, again, is $O(\epsilon)$ as k is viewed as a constant). (3) each player $\mathcal{S}_{i,j}$ receives one query in each of two (independent) SPIR invocations; By the ϵ-user-privacy of the SPIR protocol the view of such player in the multiparty protocol satisfies ϵ-privacy.
To conclude, we have established the following:

Theorem 2. *Let $k \geq 2$ be a constant. Assume that there exist a k-server one-round SPIR protocol which satisfies ϵ-privacy, for some $\epsilon \geq 0$, and has communication complexity $m(N)$. Then, for every function $f : (\{0,1\}^n)^{k'} \to \{0,1\}$, for $k' = k^2+2$, there exists a multiparty $(1, O(\epsilon))$-private protocol with communication complexity* poly$(m(2^n))$ *and round complexity* $O(1)$.

Remark 4. Similar results can also be proved for t-private MPC with $t > 1$ by applying the player simulation technique of Hirt and Maurer [18]. More specifically, k-party 1-private protocols can be composed with each other to obtain k'-party $\lfloor\frac{k'-1}{k}\rfloor$-private protocols, for any $k' > k$. However, this approach can be efficiently applied in our setting only for a constant number of players k'. It follows that the existence of communication-efficient 1-private protocols for a constant number of players implies the existence of communication-efficient protocols with a linear privacy threshold, in the sense defined in Section 1.1. It is interesting to note that in all other contexts we are aware of, the case of t-privacy can be handled directly without going through intermediate protocols for non-threshold structures as in [18]. We are not aware of a more direct way to obtain t-private protocols in our case, and leave open the question of obtaining protocols with a linear privacy threshold whose communication complexity is polynomial in both the number of players and the input length.

5 From MPC to PIR

In this section we show that if *every* k-argument function f admits a 1-private, k-party MPC protocol with communication complexity $c(n)$, then there exists a

[9] In fact, a more careful examination of block retrieval shows that only the answer complexity grows to $O(\beta^2(N))$ while the query complexity remains at $2\cdot\alpha(N)$. Similarly, the analysis of the other invocations of the SPIR can also be optimized to take into account repeated messages etc.

k-server PIR protocol with communication complexity $c(\log N) + O(\log N)$. [10] This transformation is *perfect* in the sense that if the MPC protocols are perfect then so is the PIR. The PIR protocol works as follows:

1. $\mathcal{U}$ picks at random a, b subject to $a + b = i \bmod N$ (in other words a, b form an additive secret-sharing of i). It also picks random bits r_1, r_2. It sends a, r_1 to server $\mathcal{S}_1$ and b, r_2 to server $\mathcal{S}_2$.
2. The k servers execute the guaranteed MPC protocol for the function

$$f_x((a, r_1), (b, r_2)) \stackrel{\text{def}}{=} x_{a+b} \oplus r_1 \oplus r_2.$$

The output is sent to $\mathcal{U}$ who then masks it with $r_1 \oplus r_2$ to recover x_i.

Clearly, the communication complexity is as promised. To argue the the user-privacy, observe that the input to the MPC protocol provides 1-privacy (since it is a 1-private secret sharing of i), the output of the MPC also maintains the privacy since it is masked by random bits (and each server knows at most one of the two masking bits), and the last part in the view of each server is its view in the MPC protocol, which also maintains 1-privacy. It follows:

Theorem 3. *Let $k \geq 3$ be a constant. Assume that there exists a k-player $(1, \epsilon)$-private multiparty protocol for every function $f : (\{0,1\}^n)^k \to \{0,1\}$ with communication complexity $c(n)$ and round complexity $d(n)$. Then, there exists ϵ-PIR with communication complexity $c(\log N) + O(\log N)$ and round complexity $d(\log N) + 1$.*

We note that any family of multiparty protocols with a linear privacy threshold can be easily turned into a 1-private protocol with a constant number of players by using a standard player partitioning argument.

6 Locally Decodable Codes vs. PIR

Locally decodable codes (LDCs) were introduced in [20] where their close connection with PIR was pointed out. In this section we rely on this connection; most of the material in this section can be derived from explicit and implicit statements in [20].

Recall the relevant parameters for a LDC. We are given a string $x \in \{0,1\}^N$ and encode it into a codeword y of length $M(N)$ over an alphabet Σ. The code is a (k, δ, ϵ)-LDC if after suffering an adversarial corruption of δ fraction of the symbols in the codeword y, it is still possible to reconstruct each bit x_i

[10] Note that if the complexity of every function f can be bounded by some polynomial $c_f(n)$, then there must be a uniform polynomial bound $c(n)$ that is good for *all* functions f. Otherwise, for every n let $f_n : (\{0,1\}^n)^k \to \{0,1\}$ be the "worst" function on n-bit inputs; the family of functions $f = \{f_n\}_n$ has superpolynomial complexity.

with probability at least $0.5+\epsilon$ by reading only k symbols of the (corrupted) codeword.[11]

The first transformation (that follows from implicit statements in [20]) shows that given a (k, δ, ϵ)-LDC of length $M(N)$ over alphabet Σ it is possible to construct 1-round k-server PIR with perfect privacy; its query complexity is $\alpha(N) = O(\log M(N))$, its answer complexity is $\beta(N) = \log|\Sigma|$ and its probability of success (i.e., the probability for correct reconstruction) is $0.5 + \epsilon^2\delta/(2q)$. This probability of success can be amplified to $1 - 2^{-\sigma}$ by repeating the protocol $O(\sigma)$ times.

In the opposite direction (again, using implicit statements in [20]) there is a transformation that takes a 1-round, k-server PIR protocol with success probability $0.5+\epsilon$ and, for all $\delta > 0$, constructs $(k, \delta, \epsilon/2 - k\delta)$-LDC of length $M(N) = O(k \cdot 2^{\alpha(N)}/\epsilon)$ and alphabet $\Sigma = \{0,1\}^{\beta(N)}$. This already implies that a "standard" one-round PIR with polylog(N) communication yields LDC with constant ϵ, δ and length and alphabet size which are both quasi-polynomial in N.

We observe that a transformation similar to the one used to handle multi-round PIR protocols in Section 3 can be used to show that any multi-round PIR with query complexity $\alpha(N)$, answer complexity $\beta(N)$ and success probability $0.5+\epsilon$ can be transformed into a one-round PIR with similar communication complexity and success probability of $0.5 + \epsilon/2^{\alpha(N)}$. Combining this observation with the transformation from one-round PIR to LDC, we get that if there exists a multi-round k-server PIR protocol with polylog(N) communication then there exist LDC with length and alphabet size which are both quasi-polynomial in N and δ, ϵ which are both 1/quasi-poly(N).

Remark 5. The above transformation from multi-round PIR to 1-round PIR applies also in the case where the servers in the multi-round PIR are randomized. However, the servers in the resulting 1-round PIR will also be randomized, in which case the transformation from PIR to LDC does not directly apply. It is possible to get around this difficulty by letting the user pick the servers' randomness and send it as part of its queries. Using Lemma 1, the amount of servers' randomness can be guaranteed to be of the same order of magnitude as the communication. Hence, this derandomization does not significantly increase the communication complexity of the original protocol.

7 Conclusions and Open Problems

Our results show close connections among several open problems in information-theoretic cryptography. Some of the techniques used in proving these connections may be of independent interest. In particular, the technique used in transforming PIR to SPIR can be used to reduce the amount of randomness used by more general information-theoretic protocols. Moreover, our transformation from PIR

[11] This is a non-adaptive version of the definition. An adaptive version can also be considered.

to MPC can be applied to get an information-theoretic analogue of the communication preserving secure protocol compiler from [24].

An interesting problem is to find an explicit construction of a set $\mathcal{R}$, whose existence is proved in Lemma 1, assuming that the functions it tries to fool are efficient. This requires an extension of the Nisan-Wigderson type pseudo-random generators [26] to ones that fool non-Boolean circuits. Good explicit generators of this type seem necessary for randomness reduction in *computationally-efficient* information-theoretic protocols.

Acknowledgements. We thanks Amos Beimel and Dieter van Melkebeek for helpful related discussions.

References

1. A. Ambainis. Upper bound on the communication complexity of private information retrieval. In *Proc. of 24th ICALP*, pages 401–407, 1997.
2. J. Bar-Ilan and D. Beaver. Non-cryptographic fault-tolerant computing in a constant number of rounds. In *Proc. of 8th PODC*, pages 201–209, 1989.
3. D. Beaver and J. Feigenbaum. Hiding instances in multioracle queries. In *Proc. of 6th STACS*, pages 37–48, 1990.
4. D. Beaver, J. Feigenbaum, J. Kilian, and P. Rogaway. Security with low communication overhead. In *Proc. of CRYPTO '90*, pages 62–76, 1990.
5. D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols (extended abstract). In *Proc. of 22nd STOC*, pages 503–513, 1990.
6. A. Beimel, Y. Ishai, E. Kushilevitz, and J.-F. Raymond. Breaking the $O(n^{1/(2k-1)})$ Barrier for Information-Theoretic Private Information Retrieval. In *Proc. of 43rd FOCS*, pages 261-270, 2002.
7. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. of 20th STOC*, pages 1–10, 1988.
8. B. Chor and N. Gilboa. Computationally private information retrieval. In *Proc. of the 29th STOC*, pages 304–313, 1997.
9. C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *Proc. of EUROCRYPT '99*, pages 402–414, 1999.
10. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proc. of 20th STOC*, pages 11–19, 1988.
11. B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proc. of the 36th FOCS*, pages 41–51, 1995. Journal version: *J. of the ACM*, 45, pages 965–981, 1998.
12. R. Cramer, I. Damgård, and U. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *Proc. of EUROCRYPT 2000*, pages 316–334, 2000.
13. A. Deshpande, R. Jain, T Kavita, V. Lokam, and J. Radhakrishnan. Better lower bounds for locally decodable codes. In *Proc. of 16th CCC*, pages 184–193, 2002.
14. G. Di-Crescenzo, T. Malkin, and R. Ostrovsky. Single-database private information retrieval implies oblivious transfer. In *Proc. of EUROCRYPT 2000*, pages 122–138, 2000.

15. Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. In *Proc. of 30th STOC*, pages 151–160, 1998. Journal version: *J. of Computer and System Sciences*, 60(3), pages 592–629, 2000.
16. O. Goldreich. Personal communication, 2000 (cited in [6]).
17. O. Goldreich, H. Karloff, L. Schulman, and L. Trevisan. Lower bounds for linear locally decodable codes and PIR. In *Proc. of 16th CCC*, pp. 175 – 183, 2002.
18. M. Hirt and U. Maurer. Player Simulation and General Adversary Structures in Perfect Multiparty Computation. In *Journal of cryptology*, 13(1), pages 31–60, 2000.
19. Y. Ishai and E. Kushilevitz. Randomizing Polynomials: A New Representation with Applications to Round-Efficient Secure Computation. In *Proc. of 41st FOCS*, pages 294–304, 2000.
20. J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proc. of 32nd STOC*, pages 80–86, 2000.
21. I. Kerenidis and R. de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In *Proc. of 35th STOC*, pages 106–115, 2003.
22. E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proc. of 38th FOCS*, pages 364–373, 1997.
23. E. Mann. Private access to distributed information. Master's thesis, Technion – Israel Institute of Technology, Haifa, 1998.
24. M. Naor and K. Nissim. Communication Preserving Protocols for Secure Function Evaluation. In *Proc. of 33rd STOC*, pages 590–599, 2001.
25. M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. *Proc. 31st STOC*, pages 245–254, 1999.
26. N. Nisan and A. Wigderson. Hardness vs Randomness. *J. Comput. Syst. Sci.* 49(2), pages 149-167, 1994.
27. K. Obata. Optimal Lower Bounds for 2-Query Locally Decodable Linear Codes. In *Proc. of 6th RANDOM*, pages 39–50, 2002.