Protecting Data Privacy in Private Information Retrieval Schemes[*]

Yael Gertner[†] Yuval Ishai [‡] Eyal Kushilevitz[§] Tal Malkin[¶]

Abstract

Private Information Retrieval (PIR) schemes allow a user to retrieve the i-th bit of an n-bit data string x, replicated in $k \geq 2$ databases (in the information-theoretic setting) or in $k \geq 1$ databases (in the computational setting), while keeping the value of i private. The main cost measure for such a scheme is its communication complexity.

In this paper we introduce a model of *Symmetrically-Private Information Retrieval (SPIR)*, where the privacy of the *data*, as well as the privacy of the user, is guaranteed. That is, in every invocation of a SPIR protocol, the user learns only a single physical bit of x and no other information about the data. Previously known PIR schemes severely fail to meet this goal. We show how to transform PIR schemes into SPIR schemes (with information-theoretic privacy), paying a constant factor in communication complexity. To this end, we introduce and utilize a new cryptographic primitive, called *conditional disclosure of secrets*, which we believe may be a useful building block for the design of other cryptographic protocols. In particular, we get a k-database SPIR scheme of complexity $O(n^{1/(2k-1)})$ for every constant $k \geq 2$, and an $O(\log n)$-database SPIR scheme of complexity $O(\log^2 n \cdot \log\log n)$. All our schemes require only a single round of interaction, and are resilient to any dishonest behavior of the user.

These results also yield the first implementation of a distributed version of $\binom{n}{1}$-OT (1-out-of-n oblivious transfer) with information-theoretic security and sublinear communication complexity.

1 Introduction

Private Information Retrieval (PIR) schemes allow a user to retrieve information from a database while maintaining its query private. In this model, the database is viewed as an n-bit string x out of which the user retrieves the i-th bit x_i, while giving the database no information about the index i. The main cost measure for such schemes is their communication complexity. The notion of PIR

[*]A preliminary version of this work appeared in STOC '98.

[†]Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104, USA. e-mail: `ygertner@saul.cis.upenn.edu`. Supported by DARPA grant DABT63-96-C-0018.

[‡]Department of Computer Science, Technion, Haifa 32000, Israel. e-mail: `yuvali@cs.technion.ac.il`

[§]Department of Computer Science, Technion, Haifa 32000, Israel. e-mail: `eyalk@cs.technion.ac.il` URL: http://www.cs.technion.ac.il/~eyalk

[¶]Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. e-mail: `tal@theory.lcs.mit.edu`. Supported by DARPA grant DABT63-96-C-0018.

was introduced in [12], where it was shown that if there is only one copy of the database available then n bits of communication are needed (for information-theoretic user-privacy). However, if there are $k \geq 2$ non-communicating copies of the database, then there are solutions with much better (sublinear) communication complexity.

In this paper, we introduce the stronger model of *Symmetrically Private Information Retrieval (SPIR)*, where privacy of the *data*, as well as of the user, is guaranteed. That is, every invocation of a SPIR scheme, in addition to maintaining the user's privacy, prevents the user (even a dishonest one) from obtaining any information other than a single *physical* bit of the data. Data privacy is a natural and crucial requirement in many settings. For example, consider a commercial database which sells information, such as stock information, to users, charging by the amount of data that the user retrieved. Here, both user-privacy and data-privacy are essential.

The original PIR model was only concerned with user-privacy, without requiring any protection of data-privacy. Indeed, previous PIR schemes allow the user to obtain other physical bits of the data (i.e., x_j for $j \neq i$) or other information such as the exclusive-or of certain subsets of the bits of x. A good example of where this happens is a single invocation of the best 2-database information-theoretic scheme currently known [12], from which a user can systematically retrieve $\Theta(n^{1/3})$ *physical* bits of data (see Section 5, Example 2).

To efficiently realize SPIR schemes, we introduce and utilize a new cryptographic primitive, called *"conditional disclosure of secrets"*, which may also be of independent interest as a building block for designing more general cryptographic protocols. Informally, conditional disclosure of secrets allows a set of players to disclose a secret to an external party Carol, subject to a given condition on their joint inputs. In the setting we consider, Carol knows all the inputs held by the players except for the secret to be conditionally disclosed, so she knows whether the condition holds and whether she will obtain the secret. Each player on the other hand only sees its portion of the input and does not necessarily know whether Carol will obtain the secret. The protocol involves only a unidirectional communication from the players to Carol. A simple example that illustrates the use of "conditional disclosure of secrets" is one in which each player has the input bit b_i, indicating whether it agrees to reveal the secret s to Carol. Carol obtains the secret s subject to the condition that the majority of the players agree to reveal the secret.

This work is concerned with the information-theoretic setting for SPIR. The techniques used in this work can also be applied to *computational* PIR schemes (c.f. [11, 23, 10]), in which the privacy requirement is relaxed to computational privacy (against computationally bounded databases). However, in this computational setting a better solution for realizing SPIR may be constructed using pseudo-random functions [24, 14]. We note that in addition to their theoretical significance and their unconditional security, information theoretic schemes possess other advantages over known computational schemes; they are much more time-efficient, and their communication complexity is typically smaller for moderately sized data strings (even when their asymptotic complexity is higher).

Realizing SPIR involves a modification to the previous multi-database model. This is necessary because information-theoretic SPIR schemes, regardless of their complexity, cannot possibly be achieved in the original PIR setting, in which the databases do not interact with each other at all (see Appendix A.1). We thus use a minimal extension of the original setting: continue to disallow direct interaction between the databases, but grant them access to a *shared* random string, unknown to the user. A similar kind of extension has been studied before in the contexts of private computation [16, 18], non-interactive zero-knowledge [6] and other scenarios. Here, this extension is particularly natural since, even in the basic PIR setting, databases are required to maintain *identical* copies of the same data string. (In the next subsection we discuss an alternative approach of using shared *pseudo-random* strings rather than sharing truly random strings.)

1.1 Our Results

We construct efficient SPIR schemes, with sublinear communication complexity, which may be even further improved if better PIR schemes are designed. More precisely, we present transformations from PIR schemes to SPIR schemes, preserving the user's privacy and guaranteeing data-privacy as well, with a small penalty in the communication complexity. We give two types of reductions.

A General Reduction We show that using *any* PIR scheme it is possible to construct a SPIR scheme with the same number of rounds, a constant factor overhead in communication complexity, and linear (in n) shared randomness (per query). The resultant SPIR scheme requires the use of an additional auxiliary database, which does *not* need to hold the original data (only the shared random string). That is, we achieve:

- $(k+1)$-database SPIR scheme of communication complexity $O(C(n))$, for any k-database PIR scheme of complexity $C(n)$.

However, the additional database requirement may be costly. In particular, it does not allow to obtain an information-theoretic sublinear SPIR solution with only 2 databases. This case is important, since 2 is the minimal number of databases required for such a solution to exist. Indeed, via more specific reductions we manage to avoid the additional database, and in particular obtain a good solution for the 2-database case. Moreover, these specific reductions require significantly less shared randomness.

Specific Reductions We present reductions which exploit specific structural properties of existing PIR schemes to transform them into SPIR schemes which use the same number of databases as the underlying PIR scheme, communication complexity which is at most a small constant factor over the PIR scheme, and shared randomness complexity (per query) which is of the same order of magnitude as the communication complexity. In particular, extending schemes from [12, 1] we obtain:

- k-database SPIR scheme of complexity $O(n^{1/(2k-1)})$ for any constant $k \geq 2$;
- $O(\log n)$-database SPIR scheme of complexity $O(\log^2 n \cdot \log\log n)$.

Our schemes maintain the general paradigm of existing PIR schemes: all databases hold an *identical* copy of x, and all protocols use a *single* queries-answers round.

If one is willing to settle for *computational* privacy of the data (while still maintaining the *information-theoretic* privacy of the user) then we can also consider a slight variation of the model, by replacing the shared random strings with pseudo-random ones. More specifically, the databases may share a short random seed from which longer shared pseudo-random strings can be generated "on the fly", without extra communication [7, 28]. This allows the databases to save storage space and save on the amount of random bits they need to produce. We also remark that by using pseudo-random functions [17] it is possible for the databases, in each execution of the protocol, to directly expand from the seed only the portion of the expanded string that is needed for this particular execution (without actually expanding the whole string).

Our results, as of most cited PIR works, concentrate mainly on the case of 1-privacy. The more general notion of t-privacy requires that the view of any collusion of t databases is independent of the user's retrieval index i. A generalization of our SPIR protocols that satisfies this stronger t-privacy requirement is described later in the paper (Subsection 6.2).

Note that we restrict our attention to retrieval of *single* bits, rather than the retrieval of blocks consisting of multi-bit records. In Subsection 6.1 we address block retrieval, and show that for single-round schemes, concentrating on single-bit records does not compromise generality. We then describe how to generalize our results for multi-round schemes as well, achieving SPIR for multi-bit records.

Finally, an interesting observation is that the SPIR problem may be viewed as a distributed version of a known cryptographic primitive called $\binom{n}{1}$-Oblivious-Transfer (OT) [25, 15, 8, 9]. An $\binom{n}{1}$-OT protocol allows Bob to secretly choose one of n secret bits held by Alice, in a way that at the end of the protocol Bob learns only a single bit of his choice, and Alice learns nothing about Bob's choice. The results of our work give the first 1-round distributed implementations of $\binom{n}{1}$-OT with information-theoretic security and sublinear communication complexity. Since $\binom{n}{1}$-OT is a useful tool for cryptographic protocol design, it is our hope that SPIR might also be found a useful tool for the design of cryptographic protocols.

1.2 Related Work

Private information retrieval (with information-theoretic user privacy) was introduced in [12], where the schemes achieve communication complexity of $O(n^{1/3})$ bits with 2 databases; $O(n^{1/k})$ bits with $k \geq 3$ databases; and $O(\log^2 n \log\log n)$ bits with $k = O(\log n)$ databases. In [1] the k-database upper bound is improved to $O(n^{1/(2k-1)})$ for any constant k (see [19] for improved dependence on k and generalization to t-privacy).

The *computational* counterpart of PIR (i.e., schemes where the user-privacy is only with respect to polynomial-time databases, relying on certain intractability assumptions) was first considered in [11]; they show how to obtain schemes with communication complexity $O(n^c)$ (for any constant $c > 0$) for $k=2$ databases, assuming the existence of one-way functions. The first computational PIR scheme for a *single* database was obtained in [23], achieving communication complexity $O(n^c)$ (for any constant $c > 0$), under the quadratic residuosity assumption. A single-database computational PIR with polylogarithmic communication complexity is presented in [10], under a new intractability assumption called the Φ-hiding assumption. All the above schemes require only a single round of queries and answers. In [4] it is shown that a necessary assumption for any single database PIR with less than n communication complexity, is the existence of one-way functions. In [14] this result is strengthened to show that oblivious transfer is necessary for PIR.

Subsequent to our work, the computational counterpart of SPIR has been addressed in [24, 14], showing an efficient transformation from (single-database, low communication) PIR to SPIR (in [24] a transformation is constructed assuming a 1-out-of-2 oblivious transfer primitive, and in [14] the assumption is removed by constructing this primitive from PIR).

1.3 Organization

In Section 2 we introduce notations and basic definitions. In Section 3 we show a general transformation of PIR schemes into SPIR schemes, including the introduction of "conditional disclosure of secrets" in Subsection 3.2. The following sections present specific schemes, which outperform the ones obtained by applying the general transformation. Section 4 includes SPIR schemes which rely on the user being honest. In Section 5 we present schemes which keep the data private from *any*, possibly dishonest, user (with a minor extra communication cost). Section 6 contains extensions and generalization of our results: Subsection 6.1 generalizes the results for block retrieval of multi-bit records; Subsection 6.2 generalizes the results to schemes with higher levels of user-privacy (that is, privacy against coalitions of databases); and Subsection 6.3 outlines a generalization of SPIR, called private retrieval with costs, where our techniques and results can be used. Finally, Appendix A.1 contains a proof of the impossibility of SPIR in the usual PIR setting (without direct interaction between the databases or shared randomness), and Appendix A.2 gives a lower bound on the amount of shared randomness necessary for our general PIR to SPIR transformation.

2 Preliminaries

2.1 General Notations and Definitions

The following notations and conventions are used throughout the paper. Let $[\ell]$ denote the set $\{1, 2, \ldots, \ell\}$ and $Z_\ell \overset{\text{def}}{=} \{0, 1, \ldots, \ell - 1\}$ denote the additive group of residues modulo ℓ. For any two sets S, S', let $S \oplus S'$ denote the symmetric difference between S and S' (i.e., $S \oplus S' = (S \setminus S') \cup (S' \setminus S)$). For a set $S \subseteq [\ell]$ let χ_S denote the *characteristic vector* of S: an ℓ-bit binary string whose j-th bit is equal to 1 iff $j \in S$. To simplify notation, $S \oplus j$ and χ_j are used instead of $S \oplus \{j\}$ and $\chi_{\{j\}}$, respectively. For any binary string $\sigma \in \{0, 1\}^d$, let weight(σ) denote the number of nonzero entries in σ (in particular $0 \leq \text{weight}(\sigma) \leq d$). For any n-tuple y and index set $B \subseteq [n]$, let $y|_B$ denote the restriction of y to its entries with indices from B. By default, whenever referring to a *random* choice of an element from a finite domain A, the associated distribution is uniform over A, and this random choice is independent of all other random choices. Finally, addition and multiplication operations will sometimes be carried over a finite field or group, as implied by the context.

A Boolean function $h : \{0, 1\}^m \rightarrow \{0, 1\}$ is called *monotone* if for every $A, B \subseteq [m]$ s.t. $A \subseteq B$, if $h(\chi_A) = 1$ then also $h(\chi_B) = 1$. A *Boolean formula* over the variables $y_1, \ldots, y_n$ is a labeled binary tree, whose leaves (representing inputs) are labeled by literals from $\{y_1, \overline{y_1}, \ldots, y_n, \overline{y_n}\}$, and whose internal nodes (representing boolean operators) are labeled by "$\wedge$" or "$\vee$". Such a formula computes a Boolean function $h : \{0, 1\}^n \rightarrow \{0, 1\}$ in the natural way. A formula is said to be *monotone* if all of its leaves are labeled by positive literals (which implies that the function that the formula computes is monotone). Finally, the *size* of a formula is measured by the number of leaves.

2.2 PIR Schemes

Let k denote the number of databases, $\mathcal{DB}_j$ (for $1 \leq j \leq k$) denote the j-th database, x denote an n-bit data string which is held by each of the k databases, $\mathcal{U}$ denote the user, and i denote the position (also called *index*) of a data bit which the user wants to retrieve ($1 \leq i \leq n$).

A <u>*PIR* scheme</u> is a randomized protocol between $\mathcal{U}$ and $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$, where $\mathcal{U}$ has an access to a random input ρ, unknown to the databases, and $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$ have access to a shared random input r, unknown to the user[1]. In each round of the protocol messages are exchanged between the user and the databases: *queries* are sent from the user to each database, and *answers* are sent from each database to the user.[2] The *view* of the user in the protocol, denoted $\mathtt{view}_U(x, i, r, \rho)$, consists of its input i, its random input ρ, and all the answers received from the k databases during the execution of the protocol (with inputs x, i, r, ρ). Similarly, the view of the j-th database, denoted $\mathtt{view}_j(x, i, r, \rho)$, consists of the data string x, the shared random input r, and all the queries sent from the user to $\mathcal{DB}_j$ during the execution of the protocol. At the end of the execution, the user applies some *reconstruction* function Ψ to its view and outputs the corresponding value $\Psi(\mathtt{view}_U(x, i, r, \rho))$.

A party (user or database) in a PIR scheme is called *honest* if it follows the protocol's specification. When the user $\mathcal{U}$ interacts with (possibly dishonest) databases $\mathcal{DB}_1^*, \ldots, \mathcal{DB}_k^*$, we denote the view of the j-th database by $\mathtt{view}_j^*(x, i, r, \rho)$. Similarly, when the k databases $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$ interact with a (possibly dishonest) user $\mathcal{U}^*$ we denote the view of the user by $\mathtt{view}_U^*(x, i, r, \rho)$.

A (1-private, information-theoretic) PIR scheme is a protocol as above, which satisfies the following two requirements:

(1) *correctness:* When both the user and the k databases are honest, the user always reconstructs the data bit x_i. That is, for every x, i, r, ρ as above,

$$\Psi(\mathtt{view}_U(x, i, r, \rho)) = x_i.$$

(2) *user-privacy:* The view of any *single* database is independent of the retrieval index i. Formally, for any (possibly dishonest) databases $\mathcal{DB}_1^*, \ldots, \mathcal{DB}_k^*$ interacting with the (honest) user $\mathcal{U}$, for any shared random input r, any data string x, any two retrieval indices $1 \leq i, i' \leq n$, any database index $1 \leq j \leq k$, and any view $view_j$ of $\mathcal{DB}_j^*$,

$$\Pr_{\rho}[\mathtt{view}_j^*(x, i, r, \rho) = view_j] \quad = \quad \Pr_{\rho}[\mathtt{view}_j^*(x, i', r, \rho) = view_j] \ .$$

It should be noted that the definition of PIR schemes in the literature does not allow for a shared randomness between the databases. However, in the context of PIR the definitions are equivalent. It is only in the SPIR context where the shared-randomness becomes crucial.

2.3 SPIR Schemes

A <u>*SPIR* scheme</u> is a PIR scheme such that in any invocation of the scheme, the user cannot learn any information which doesn't follow from a single physical bit of data. Formally, a SPIR scheme should satisfy, in addition to the correctness and the user-privacy requirements, the following third requirement:

(3) *data-privacy:* For any (possibly dishonest) user $\mathcal{U}^*$ interacting with the honest databases $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$, and for any random input ρ held by $\mathcal{U}^*$, and any i', there exists an index i, such that for every data strings x, y satisfying $x_i = y_i$, and every view *view* of $\mathcal{U}^*$,

$$\Pr_{r}[\mathtt{view}_U^*(x, i', r, \rho) = view] \quad = \quad \Pr_{r}[\mathtt{view}_U^*(y, i', r, \rho) = view] \ .$$

[1] It is assumed, without loss of generality, that all databases are otherwise deterministic.

[2] As is the case in most of the PIR literature, we will mostly be interested in single-round schemes. The following definitions may take a slightly simpler form when the schemes are restricted to a single round.

Let us argue that the above definition yields the "intuitive notion" of data privacy. The intuitive notion that we want to capture is that the user cannot learn any information about the data which does not follow from a single physical bit. One may be tempted to require that for any user $\mathcal{U}^*$ there exists a single index i, such that the view of $\mathcal{U}^*$ is independent of the data string x given x_i. However, this (stronger) variant of the definition cannot be satisfied. To see that, consider a SPIR scheme $\mathcal{S}$ satisfying this latter requirement, and consider a user $\mathcal{U}^*$ which starts by randomly choosing an index i, and then proceeds to run according to $\mathcal{S}$ with retrieval index i. Clearly, there is no single index i such that the view of such user depends on x_i alone. What our definition requires is that, for every random string ρ held by the user, the user must (explicitly or implicitly) fix an index i such that its view depends only on x_i.[3] Finally, note that an equivalent formulation of the data-privacy requirement is the following one: For any *deterministic* user $\mathcal{U}^*$, there exists an index i, such that the user's view is independent of the data string x given x_i.

An *honest-user SPIR* scheme is a PIR scheme that satisfies the data-privacy requirement with respect to $\mathcal{U}$, the *honest (but curious)* user, which follows the scheme's specification but may try to deduce extra information from the communication.

Notice that the above formulation of the model is only concerned with answering a single retrieval query made by a single user. Multiple queries (possibly originating from different users) may be handled by independent repetitions of the single-query scheme, where in each invocation the databases use an independent source of shared randomness (or a "fresh" portion of a single shared random string).

By default, the terms "PIR scheme" and "SPIR scheme" refer to *1-round*, *1-query*, *information theoretically private* schemes.

2.4 Complexity

The main complexity measure for PIR and SPIR schemes is their *communication complexity*. The communication complexity of a k-database scheme will be denoted $(\alpha_k(n), \beta_k(n))$, where $\alpha_k(n)$ is the total number of query bits sent from the user to all k databases and $\beta_k(n)$ is the total number of answer bits sent from all k databases to the user, when the data string is of size n. We sometimes use a single parameter to measure the communication complexity of a given scheme, which is the total communication complexity $\alpha_k(n) + \beta_k(n)$.

The *shared randomness complexity* of a SPIR scheme is defined as the entropy of the shared random input r (which equals to the length of the string r in the case it is uniformly distributed over all strings of some fixed length).

Finally, while the definitions in Subsections 2.2 and 2.3 do not address the aspect of *computational* efficiency, all protocols constructed in this work will also be computationally efficient (that is, polynomial in n).

[3] Also note that if the user has some a-priori information regarding the data string x (e.g., that $x_j = x_i$) then the retrieval of x_i, together with its a-priori information, may give it information about other bits of x; this is obviously unavoidable.

3 A General Reduction from SPIR to PIR

In this section we present a construction of a SPIR scheme by using *any* PIR scheme as a black-box. This construction introduces an overhead of a single auxiliary database, a constant factor in communication complexity, and a linear amount of shared randomness over the corresponding PIR scheme. The auxiliary database *need not* hold a copy of the data string x; it only needs to have access to the shared random string r.

More specifically, we present two general reductions. The first is with respect to an honest user and costs only an *additive* logarithmic factor in communication complexity (Subsection 3.1). The second strengthens the first to deal with *any* user, possibly dishonest (Subsection 3.3). The latter is constructed by utilizing a new cryptographic primitive, called "conditional disclosure of secrets" (introduced in Subsection 3.2), which will also be used in later sections. We note that both reductions (Theorems 1 and 3) are stated and proved for a single round PIR, but can be generalized to apply to PIR schemes with any number of rounds.

3.1 A General Reduction with Respect to Honest Users

Theorem 1. Let $\mathcal{P}$ be any 1-round k-database PIR scheme with communication complexity $(\alpha_k(n), \beta_k(n))$. Then, there exists a 1-round $(k+1)$-database honest-user SPIR scheme $\mathcal{S}_{\mathcal{P}}$ with communication complexity $(\alpha_k(n) + (k+1)\lceil \log_2 n \rceil, \beta_k(n) + 1)$, and shared randomness complexity n.

Proof. To simplify notation, assume that the index i is taken from the set $Z_n = \{0, 1, \ldots, n-1\}$ (rather than from $[n]$). The scheme $\mathcal{S}_{\mathcal{P}}$ involves k databases $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$, corresponding to databases of the original scheme $\mathcal{P}$, and an auxiliary database $\mathcal{DB}_0$. All databases share a random string $r \in \{0,1\}^n$. The scheme $\mathcal{S}_{\mathcal{P}}$ proceeds as follows:

Queries: First the user picks queries $q_1, \ldots, q_k$ as specified by the PIR scheme $\mathcal{P}$, and independently picks a random shift amount $\Delta \in Z_n$. Then the user sends to each $\mathcal{DB}_j$, for $1 \le j \le k$, the same shift amount $\Delta_j = \Delta$, along with the query q_j. Finally, the user sends the shifted index $i' \stackrel{\text{def}}{=} (i - \Delta) \bmod n$ to $\mathcal{DB}_0$.

Answers: Each database $\mathcal{DB}_j$, for $1 \le j \le k$, locally computes a "virtual data string" $x' \stackrel{\text{def}}{=} x \oplus (r \gg \Delta)$, where $\oplus$ denotes bitwise exclusive-or, and $r \gg \Delta$ denotes a cyclic shift of the random string r by Δ places to the right. Then, $\mathcal{DB}_j$ answers the query q_j as it would do in the original PIR scheme $\mathcal{P}$ with respect to the computed string x'. Finally, the auxiliary database $\mathcal{DB}_0$ replies with the single bit $r_{i'}$.

Reconstruction: The user reconstructs x_i by first reconstructing from the answers of $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$ a bit $b_{\mathcal{P}}$ according to PIR scheme $\mathcal{P}$, and then computing the exclusive-or of this bit with the bit $r_{i'}$ received from $\mathcal{DB}_0$.

By the correctness of $\mathcal{P}$, we have $b_{\mathcal{P}} = x'_i$. Therefore, the reconstruction step of $\mathcal{S}_{\mathcal{P}}$ yields $b_{\mathcal{P}} \oplus r_{i'} = x'_i \oplus r_{i'} = (x_i \oplus r_{i'}) \oplus r_{i'} = x_i$, which proves the correctness of $\mathcal{S}_{\mathcal{P}}$. The user's privacy follows from the privacy of $\mathcal{P}$, and from the fact that each of the additional queries Δ and i' is uniformly distributed in Z_n, independently of the $\mathcal{P}$-queries $q_1, \ldots, q_k$. Finally, to show that the scheme $\mathcal{S}_{\mathcal{P}}$ meets the data-privacy requirement with respect to the honest user, we will use the following, more general, claim.

Claim 1. Let $q = \langle i', (q_1, \Delta_1), (q_2, \Delta_2), \ldots, (q_k, \Delta_k)\rangle$ be any $(k+1)$-tuple of queries (possibly, but not necessarily, picked by an honest user). Moreover, suppose that $\Delta_1 = \Delta_2 = \cdots = \Delta_k \stackrel{\text{def}}{=} \Delta$. Then, the joint answers of $\mathcal{DB}_0, \ldots, \mathcal{DB}_k$ to their corresponding queries in q are independent of x given $x_{i'+\Delta}$ (where the probability space is over the choice of r, and where the sum $i' + \Delta$ is taken modulo n).

Proof. Let $x' \stackrel{\text{def}}{=} x \oplus (r \gg \Delta)$. Note that x' is the virtual data string computed by each database $\mathcal{DB}_j$, $1 \le j \le k$, in the process of answering to its own query from q, i.e., q_j. Now, consider the joint distribution of $(x', r_{i'})$. This distribution is uniform over the set

$$\{(y, b) \,:\, y \in \{0,1\}^n, b \in \{0,1\}, y_{i'+\Delta} \oplus b = x_{i'+\Delta}\},$$

thus depending only on $x_{i'+\Delta}$. Since x' *determines* the answers of $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$ given the query-tuple q, and since $r_{i'}$ is the answer of $\mathcal{DB}_0$, it follows that the joint distribution of all answers given such query-tuple q depends on $x_{i'+\Delta}$ alone. $\square$

Claim 1 implies that the distribution of the view of an honest user, given that it holds input i and random input ρ, depends only on a single data bit, because an honest user sets $\Delta_1 = \Delta_2 = \cdots = \Delta_k = \Delta$. This shows the data-privacy of $\mathcal{S}_{\mathcal{P}}$ with respect to an honest user, and concludes the proof of Theorem 1. $\square$

Note that in the above scheme $\mathcal{S}_{\mathcal{P}}$, a dishonest user can either send invalid $\mathcal{P}$-queries, or send different shifts Δ_j to different databases. However, by Claim 1, only the latter dishonest behavior could potentially give the user more information on the data. In other words, if the user sends the same shifts to all databases, then data-privacy will always be maintained, regardless of the validity of the other queries. Thus, to extend this scheme for a dishonest user, it would suffice to have the databases (each of which sees only a single Δ_j) send their answers disguised so that the user learns the answers only if the condition $\Delta_1 = \ldots = \Delta_k$ is satisfied. To this end, we use the primitive of "conditional disclosure of secrets", introduced in the next subsection.

A natural question regarding the above transformation is whether its shared randomness complexity may be reduced. A partial answer to this question is given in appendix A.2, where it is shown that for *our* transformation to be general (i.e. applicable to any underlying PIR scheme), the shared n-bit string used there must be *uniformly* distributed over $\{0,1\}^n$, namely linear shared randomness is required regardless of the communication complexity of the underlying PIR scheme.

Finally, we note that Claim 1 implies that if $\mathcal{P}$ is the trivial 1-database PIR scheme in which the entire data string is being sent to the user, then the 2-database SPIR scheme $S_{\mathcal{P}}$ constructed above is resilient also against a dishonest user. We thus have:

Corollary 1. There exists a 1-round, 2-database SPIR scheme $\mathcal{S}_2^*$ with communication complexity $(2\lceil\log_2 n\rceil, n+1)$, and shared randomness complexity n.

While this scheme $\mathcal{S}_2^*$ is inefficient on its own, as it requires linear communication complexity, it will be used as a subprotocol (with small data strings) in our later constructions.

3.2 Conditional Disclosure of Secrets

In this subsection we describe and implement a new cryptographic primitive, called *conditional disclosure of secrets* (or *CDS* for short). This primitive is then used in the next subsection to obtain a general reduction from SPIR to PIR withstanding *any* user behavior.

Informally, the conditional disclosure setting involves k players, each holding some input, and an external party Carol, who knows all inputs held by the players. In addition, there is a secret s which is known to at least one of the players but not to Carol. The goal is for the players to disclose the secret to Carol, subject to a given condition on their joint input (namely if the condition holds, Carol learns the secret, and if it doesn't she obtains no information about the secret). The model allows all the players to have access to a shared random string (hidden from Carol), and the only communication allowed is a single unidirectional message sent from each player to Carol. A simple example illustrating the use of CDS is one in which each player has an input bit $b_i \in \{0,1\}$, and the condition for disclosing the secret to Carol is that the majority of the players' bits are set to 1.

A formal definition is given below. For convenience, we start by defining a version where the secret s to be disclosed is known to *all* players (we call this version conditional disclosure of a common secret).

Let $h : \{0,1\}^n \rightarrow \{0,1\}$ be a fixed boolean function (the *condition*); let $B_1, \ldots, B_k$ be a partition of $[n]$ into k sets (each $B_j \subseteq [n]$ is called the *j-th player input portion*); and let *SD* be some *secret domain* (e.g., all binary strings of a particular length). A *conditional disclosure of a common secret* for the condition h, input partition $B_1, \ldots, B_k$, and secret domain *SD*, consists of a set of k players $P_1, \ldots, P_k$ (modeled as *functions*) and (an *external party*) Carol, as follows. Let r denote a *shared random input* of the players, drawn from some distribution R. For any fixed $y = y_1 \ldots y_n \in \{0,1\}^n$ (the *input*), $s \in SD$ (the *secret*), and $1 \leq j \leq k$, we define a random variable $m_j = P_j(y|_{B_j}, s, r)$ (the *j-th player message*), where the randomness is over the choice of r. Then the following two conditions must hold:

1. **correctness:** For every $y \in \{0,1\}^n$, if $h(y) = 1$, then $\forall s, r$, $\text{Carol}(y, m_1, \ldots, m_k) = s$. That is, if the condition holds, then Carol is always able to reconstruct the secret s from her input and the messages she received.

2. **secrecy:** For every $y \in \{0,1\}^n$, if $h(y) = 0$, then for any $s_0, s_1 \in SD$ the k-tuples of random variables $\left\langle m_j^{s_0} = P_j(y|_{B_j}, s_0, r)\right\rangle_{j=1}^{k}$ and $\left\langle m_j^{s_1} = P_j(y|_{B_j}, s_1, r)\right\rangle_{j=1}^{k}$ are identically distributed (where the probability is over the choice of r). That is, if the condition does not hold, Carol obtains no information about the secret s (the messages received by Carol are identically distributed for any two possible secrets s_0 and s_1).

A similar version can be defined when the secret s is known to at least one of the players (not necessarily to all of them). In this case we let $m_j = P_j(y|_{B_j}, r)$ for players P_j who do not hold s (their message is constructed only based on their portion of the input and the shared randomness). We call this (more general) version *conditional disclosure of a secret*.

The *communication complexity* of a conditional disclosure protocol is the maximal total size of all messages sent by the players (over the choices of r), and its *shared randomness complexity* is the entropy of R.

We note that the model of conditional disclosure is similar to the non-interactive model of private computation from [16], which is described in Subsection 4.2. Known results in that (in a sense more general) model are sufficient to yield *some* solutions to the conditional disclosure problem. For instance, results of [16, 18] imply conditional disclosure protocols with communication which is quadratic in the size of a branching program or a formula describing the condition h (see Remark 2 for discussion). However, the solutions obtained via these general results are usually not efficient enough for our purposes. Instead, we show below how to achieve much more efficient solutions, which use communication at most linear in the size of h.

3.2.1 Reduction to Generalized Secret Sharing

In the following we show how to implement conditional disclosure of secrets under an arbitrary condition by reducing it to generalized secret sharing [5, 20] relative to a corresponding access structure.

Generalized secret sharing. The problem of *generalized secret sharing* is an extension of the usual notion of t-out-of-m secret sharing [26]. Informally, a generalized secret sharing protocol is a randomized protocol for sharing a secret into m shares such that the secret can be reconstructed from any qualified set of shares, whereas any combination of an unqualified set of shares should give no information about the secret. Formally, a generalized secret sharing scheme with secret domain SD is defined by a triple $(\mathcal{D}, R, \mathcal{C})$, where $\mathcal{D}$ (the *dealing function*) maps a secret $s \in SD$ and a random input r into an m-tuple of shares $\langle s_1, \ldots, s_m \rangle$, R is the distribution from which the random input r is chosen, and $\mathcal{C}$ (the *reconstruction function*) maps a set $A \subseteq [m]$ and an $|A|$-tuple of shares into a reconstructed secret $s \in SD$. The collection of qualified sets is specified by a monotone Boolean function $h_M : \{0,1\}^m \rightarrow \{0,1\}$, called an *access structure*, where a set $A \subseteq [m]$ of shares is said to be qualified if $h_M(\chi_A) = 1$ and otherwise is said to be unqualified. The scheme $\mathcal{S} = (\mathcal{D}, R, \mathcal{C})$ is said to be a generalized secret sharing scheme realizing the access structure h_M if it satisfies the following two requirements: (1) **correctness:** for any *qualified* set $A \subseteq [m]$, every secret $s \in SD$, and every random input r, the reconstruction succeeds; that is, $\mathcal{C}(A, \mathcal{D}(s,r)|_A) = s$; and (2) **secrecy:** for any *unqualified* set $A \subseteq [m]$ and secrets $s_1, s_2 \in SD$, the random variables $\mathcal{D}(s_1, r)|_A$ and $\mathcal{D}(s_2, r)|_A$ are identically distributed (where the probability is over the choice of r, distributed according to R). Finally, the *share complexity* of $\mathcal{S}$ is the maximum total size of all shares in an m-tuple $\mathcal{D}(s,r)$, and its *randomness complexity* is the entropy of R.

Lemma 1. Let $h_M : \{0,1\}^m \rightarrow \{0,1\}$ be a monotone Boolean function. Let $h : \{0,1\}^n \rightarrow \{0,1\}$ be a Boolean function defined by $h(y_1, \ldots, y_n) = h_M(g_1, \ldots, g_m)$, where each g_i depends on a single variable y_j; that is, $g_i \in \{y_1, \overline{y_1}, \ldots, y_n, \overline{y_n}\}$ for $1 \le i \le m$ (such h will be referred to as a *projection* of h_M). Let $\mathcal{S}$ be a generalized secret sharing scheme with secret domain SD realizing the access structure h_M, with share complexity β and randomness complexity γ. Then, for any partition $B_1, \ldots, B_k$ among k players of the inputs to h, there exists a protocol $\mathcal{P}$ for disclosing a common secret $s \in SD$ subject to the condition h, with communication complexity β and shared randomness complexity γ.

Proof. Recall that the CDS protocol $\mathcal{P}$ involves players $P_1, \ldots, P_k$ each holding a portion of the input $y = y_1, \ldots, y_n$ (player P_j holds B_j) and the secret $s \in SD$. The players wish to reveal their secret to Carol subject to the condition $h(y) = 1$. We show how to construct $\mathcal{P}$ using the generalized secret sharing scheme $\mathcal{S} = (\mathcal{D}, R, \mathcal{C})$ realizing the access structure h_M, where $h_M(g_1, \ldots, g_m) = h(y_1, \ldots, y_n)$

The protocol $\mathcal{P}$ uses a shared random string r distributed according to R, and proceeds as follows. First, each player P_j evaluates $\mathcal{D}(s, r)$, generating an m-tuple of shares $\langle s_1, \ldots, s_m \rangle$ (note that all players generate the same shares, since they use the same secret and the same random input when evaluating $\mathcal{D}$). Next, for each $i \in [m]$, player P_j includes the share s_i in the message sent to Carol if and only if the following two conditions hold: (1) g_i is "owned" by P_j (i.e., g_i is either y_l or $\overline{y_l}$ for some $l \in B_j$); and (2) g_i evaluates to 1. That is, the message sent to Carol by the player P_j consists of the restriction of the shares $\langle s_1, \ldots, s_m \rangle$ to those which satisfy the above two conditions.

Observe that since each input variable y_l is held by some player, Carol receives exactly those shares s_i for which $g_i = 1$. By this observation, if $h_M(g_1, \ldots, g_m) = 1$ then Carol has exactly the s_i for which $g_i = 1$, which according to the definition of generalized secret sharing is a qualified set of shares, and can thus reconstruct the secret s (using the reconstruction function $\mathcal{C}$). On the other hand, if $h_M(g_1, \ldots, g_m) = 0$ then Carol receives an unqualified set of shares, and hence gains no information about s. To complete the proof, recall that $h_M(g_1, \ldots, g_m) = h(y_1, \ldots, y_n)$; thus, Carol can reconstruct s whenever the condition $h(y)$ holds, and otherwise obtains no information on s.

Finally, the shared randomness complexity of $\mathcal{P}$ is the same as the randomness complexity of $\mathcal{S}$, and the communication complexity of $\mathcal{P}$ is no larger than the share complexity of $\mathcal{S}$ (since each share is sent by at most one player). $\square$

We now use Lemma 1 to obtain an upper bound on the complexity of conditional disclosure of secrets, depending on the size of a formula computing the condition. The proof of the following theorem will use a known result about the complexity of generalized secret sharing.

Fact 1. [5] Suppose that $h_M : \{0,1\}^m \to \{0,1\}$ can be computed by a monotone Boolean formula of size S. Then, there exists a generalized secret sharing scheme realizing h_M with $SD = \{0,1\}$, whose communication complexity and shared randomness complexity are bounded by S.

Theorem 2. Suppose that $h : \{0,1\}^n \to \{0,1\}$ can be computed by a Boolean formula of size S, and let $SD = \{0,1\}$. Then, for every partition $B_1, \ldots, B_k$ of the inputs to h,

1. there exists a protocol $\mathcal{P}$ for disclosing a common secret bit $s \in SD$ (known to all players) subject to the condition h, with communication complexity and shared randomness complexity bounded by S.

2. there exists a protocol $\mathcal{P}'$ for disclosing a secret bit $s \in SD$ (known to at least one player) subject to the condition h, with communication complexity and shared randomness complexity bounded by $S + 1$.

Proof. A protocol $\mathcal{P}$ for conditional disclosure of a common secret bit s known to all players is constructed as follows. Let H be a Boolean formula over the variables $y_1, \ldots, y_n$ computing h, whose size is S. Replacing each negative literal $\overline{y}_j$ with a positive literal w_j, we obtain a *monotone* Boolean formula H_M of size S computing a monotone function $h_M(y_1, \ldots, y_n, w_1, \ldots, w_n)$. Note that h is a projection of h_M, since $h(y_1, \ldots, y_n) = h_M(y_1, \ldots, y_n, \overline{y}_1, \ldots, \overline{y}_n)$. Using Fact 1, it follows from Lemma 1 that the players can disclose the bit s subject to the condition h using at most S communication bits and at most S shared random bits, which completes the proof of the first part of the theorem.

For the second part, a protocol $\mathcal{P}'$ for conditional disclosure of a secret bit s known to at least one player, proceeds as follows. The players first conditionally disclose a *shared* random bit r_0, known to *all* of them, subject to the condition h. This is done using the protocol $\mathcal{P}$ described above. Finally, a single player holding s sends the bit $s \oplus r_0$ to Carol. Clearly, if Carol can reconstruct r_0 then she can also reconstruct s, and if she obtains no information on r_0 then she can obtain no information on s, and the theorem follows. $\square$

Remark 1. Using best known general upper bounds on the complexity of generalized secret sharing [21], the result of Theorem 2 can be strengthened to apply to any function h with a *span program* over GF(2) of size S (see [21] for a definition of the span program model).

3.2.2 Direct Constructions for Special Cases

In the sequel, the conditional disclosure primitive will be used in our reductions for dealing with dishonest behavior of the user. These applications of conditional disclosure require only a simple condition (e.g., testing equality between inputs). Therefore, in the following we give direct constructions of conditional disclosure protocols realizing these specific conditions. These direct constructions are more efficient than the ones obtained by a straightforward application of Theorem 2. We stress though that the more general results described above are still useful in other cryptographic scenarios, such as the one described in Subsection 6.3.

The next lemma shows an efficient implementation of conditional disclosure of secrets, where the condition tests whether the sum of k field elements equals 0. Later it will mostly be used with $k = 2$, to implement conditional disclosure of secrets where the condition tests for *equality* between two strings.

Lemma 2. Let F be a finite field (all arithmetic operations below are in this field). Suppose that each of k players P_j holds an input $y_j \in F$, and that a secret $s \in F$ is known to at least one player. Then, there exists a protocol for disclosing the secret s subject to the condition "$\sum_{j=1}^{k} y_j = 0$" in which each player sends a single field element, and whose shared random string consists of k random field elements.

Proof. Assume without loss of generality that player P_k holds the secret s, and let $r_0, r_1, ..., r_{k-1}$ be independent random elements of F, shared by the parties. The protocol can then proceed as follows:

- Each player P_j, $1 \le j \le k-1$, sends to Carol the single field element $m_j \overset{\text{def}}{=} y_j r_0 + r_j$;
- The player P_k sends to Carol $m_k \overset{\text{def}}{=} s + y_k r_0 - \sum_{j=1}^{k-1} r_j$.

First, note that if all inputs y_j add up to 0, then s can be reconstructed as the sum of all messages m_j:

$$\sum_{j=1}^{k} m_j = \sum_{j=1}^{k-1} (y_j r_0 + r_j) + s + y_k r_0 - \sum_{j=1}^{k-1} r_j = s + r_0 \sum_{j=1}^{k} y_j = s.$$

We now show that if $\sum y_j \neq 0$, the k-tuple of messages $(m_1, \ldots, m_k)$ is uniformly distributed over F^k independently of s. For any sequence of messages $m_1, \ldots, m_k \in F^k$, we define its *support* as the set of all choices $r_0, r_1, \ldots, r_{k-1}$ which make the players send this sequence of messages to Carol (when the inputs are $y_1, \ldots, y_k$ and the secret is s). By the construction of the protocol, the support consists of exactly all $r_0, r_1, \ldots, r_{k-1}$ satisfying the system of equations

$$\begin{array}{rcccl}
y_1 r_0 & +r_1 & & & = m_1 \\
y_2 r_0 & & +r_2 & & = m_2 \\
\vdots & & & & \\
y_{k-1} r_0 & & & +r_{k-1} & = m_{k-1} \\
y_k r_0 & -r_1 & \ldots & -r_{k-1} & = m_k - s
\end{array}$$

This is a system of k linear equations in the k variables $r_0, r_1, \ldots, r_{k-1}$. When $\sum y_j \neq 0$ the k equations are linearly independent, since adding the first $k-1$ equations to the last one yield a triangular system of equations. Therefore, any sequence of messages $m_1, \ldots, m_k \in F^k$ has a support which is a singleton, and in particular all sequences have the *same size* support. This implies that the uniform distribution of the field elements $r_0, r_1, \ldots, r_{k-1}$ induces a uniform distribution of the messages $m_1, \ldots, m_k$ over F^k, for any input tuple $y_1, \ldots, y_k$ with nonzero sum and any secret $s \in F$. $\square$

Note that the above lemma outperforms the general construction of Theorem 2. Using the general construction, the communication and randomness required for disclosing a *single bit* secret is larger than the total size of k field elements (which is a lower bound on the size of a formula evaluating the condition), whereas in the specific construction of Lemma 2 communication and randomness of this size are sufficient for the disclosure of a *longer* secret, namely a field element. The following lemma shows that it is possible to further reduce the communication to be dominated by the secret size, even when the secret is smaller than the inputs.

Lemma 3. Suppose that each of k players holds an input string[4] $y_j \in \{0,1\}^\ell$, and a secret string $s \in \{0,1\}^m$ is known to at least one player. Then, there exists a protocol for disclosing the secret s subject to the condition "$\bigoplus_{j=1}^k y_j = 0^\ell$" in which each player sends a string of length m, and whose shared randomness complexity is $k \cdot \max(\ell, m)$.

Proof. For a finite field $F = \mathrm{GF}(2^w)$, we use a standard representation of field elements by w-bit strings, such that each element of F is represented by the coefficient vector of the polynomial associated with it. (Recall that an element of $\mathrm{GF}(2^w)$ may be identified with a polynomial over $\mathrm{GF}(2)$ of degree $\leq w-1$, modulo some irreducible degree-w polynomial). Such a representation defines an *isomorphism* between the groups $\langle F, + \rangle$ and $\langle \{0,1\}^w, \oplus \rangle$.

We now consider two possible cases. If $\ell \leq m$, then the protocol from the proof of Lemma 2 can be used *as is*, letting $F = \mathrm{GF}(2^m)$, and associating the secret s with the corresponding field element and each input string $y_j \in \{0,1\}^\ell$ with the field element corresponding to its m-bit padding $y_j 0^{m-\ell}$.

In the second case ($\ell > m$), we use the same protocol with $F = \mathrm{GF}(2^\ell)$, except that each field element sent in the original protocol is *projected* to the m leftmost bits of its representation; that is, if m_j is the field element originally sent by P_j and is represented by the string $\sigma_1\sigma_2\cdots\sigma_\ell$, then the message sent from P_j to Carol in the new protocol would be the m-bit prefix $\sigma_1\sigma_2\cdots\sigma_m$. A key observation is that, under the above representation, the projection operator commutes with the field addition. Hence, the sum of all ℓ-bit projections sent in the new protocol is equal to the projection of $\sum_{j=1}^k m_j$. It follows from the above observation and from the analysis in the proof of Lemma 2 that if the condition "$\bigoplus_{j=1}^k y_j = 0^\ell$" holds, then s can be reconstructed as the exclusive-or of all messages. On the other hand, if the condition does not hold, then the original k messages are uniformly and independently distributed over F, from which it follows that the projected m-bit messages are independently and uniformly distributed over $\{0,1\}^m$. This proves the correctness and secrecy of this protocol.

Finally, since in both cases each player sends a message string of length m, the specified communication bound is met, and since in both cases the protocol of Lemma 2 is invoked with $F = \mathrm{GF}(2^{\max(\ell,m)})$, the specified shared randomness bound is met as well. $\square$

[4]The lemma is formulated for binary strings, but can be generalized to strings over any finite field.

In particular, the result of Lemma 3 can be applied with $k = 2$ for conditionally disclosing a secret s subject to a condition which tests *equality* of strings held by two players. This protocol clearly outperforms any protocol obtainable via the general result of Theorem 2; indeed, since testing equality between ℓ-bit strings requires a formula of size $\Theta(\ell)$, the best protocol obtainable via Theorem 2 would require $\Theta(\ell)$ communication bits for conditionally disclosing a *single* bit subject to equality between two ℓ-bit strings (compared to only 2 communication bits required using Lemma 3). The improved efficiency obtained via Lemma 3 will be used in the next subsection.

3.3 A General Reduction with Respect to Dishonest Users

Using the conditional disclosure of secrets primitive described above, the following theorem gives a general reduction from any PIR scheme to a SPIR scheme for the case of *any* user (possibly dishonest).

Theorem 3. Let $\mathcal{P}$ be any 1-round k-database PIR scheme with communication complexity $(\alpha_k(n), \beta_k(n))$. Then, there exists a 1-round, $(k+1)$-database SPIR scheme $\mathcal{S}^*_{\mathcal{P}}$ with communication complexity at most $(\alpha_k(n) + (k+1)\lceil \log_2 n \rceil \,, 2\beta_k(n)+1)$, and shared randomness complexity $O(n + \beta_k(n))$.

Proof. Let $\mathcal{S}_{\mathcal{P}}$ be the protocol from the general (honest-user) reduction of Theorem 1. By Claim 1, $\mathcal{S}_{\mathcal{P}}$ satisfies data-privacy as long as the user sends to every database $\mathcal{DB}_j$ the same shift amount Δ_j. Thus we make $\mathcal{S}^*_{\mathcal{P}}$ be the following modification of $\mathcal{S}_{\mathcal{P}}$, effectively forcing the user to send the same shifts.

The user's queries are the same as in $\mathcal{S}_{\mathcal{P}}$, and so are the answers of $\mathcal{DB}_0$ (the auxiliary database) and $\mathcal{DB}_1$. In addition, for each $2 \le j \le k$, we let $\mathcal{DB}_j$ and $\mathcal{DB}_1$ disclose the original $\mathcal{S}_{\mathcal{P}}$-answer of $\mathcal{DB}_j$ subject to the condition $\Delta_j = \Delta_1$ (where Δ_j is the $\lceil \log_2 n \rceil$-bit shift sent to $\mathcal{DB}_j$). This conditional disclosure is implemented using Lemma 3.

The user-privacy in the original $\mathcal{S}_{\mathcal{P}}$ is clearly maintained. The scheme $\mathcal{S}^*_{\mathcal{P}}$ meets the data-privacy requirement, since the use of conditional disclosure guarantees that the (possibly dishonest) user will obtain information only on answers of databases $\mathcal{DB}_j$ such that $\Delta_j = \Delta_1$, which by Claim 1 implies that the user learns at most a single physical bit of data. Hence, $\mathcal{S}^*_{\mathcal{P}}$ is indeed a SPIR scheme.

We now analyze the complexity of this scheme. For each $0 \le j \le k$ we let $\beta^j_k(n)$ denote the length of the answer sent by $\mathcal{DB}_j$ in the scheme $\mathcal{S}_{\mathcal{P}}$. By Theorem 1, we know that $\beta^0_k = 1$ and that $\sum_{j=0}^{k} \beta^j_k(n) = \beta_k(n) + 1$. Using Lemma 3, the communication complexity required to implement the conditional disclosure subprotocol involving the databases $\mathcal{DB}_1$ and $\mathcal{DB}_j$ in the scheme $\mathcal{S}^*_{\mathcal{P}}$ is $2\beta^j_k(n)$. The total communication sent from all databases to the user is therefore $\beta^0_k(n) + \beta^1_k(n) + \sum_{j=2}^{k}(2\beta^j_k(n)) \le 1 + 2\sum_{j=1}^{k} \beta^j_k(n) = 1 + 2\beta_k(n)$. The total communication sent from the user is the same as in $\mathcal{S}_{\mathcal{P}}$, namely $\alpha_k(n) + (k+1)\lceil \log_2 n \rceil$. The shared randomness complexity is the same as in $\mathcal{S}_{\mathcal{P}}$ plus the randomness required by Lemma 3, which sums up to $n + 2\sum_{j=2}^{k} \max(2\lceil \log_2 n \rceil, \beta^j_k(n)) = O(n + \beta_k(n))$. □

In subsequent sections we present SPIR schemes which rely on specific structural properties of some underlying PIR schemes, and exploit them to outperform the above general transformations. In particular, they use sublinear shared randomness, and do not require an auxiliary database.

4 Specific SPIR Schemes with Respect to Honest Users

In this section we construct *honest-user* SPIR schemes which perform as well as their PIR counterparts, up to a multiplicative constant, both in terms of communication and randomness. Our constructions utilize two primitives: private simultaneous messages protocols (described below), and conditional disclosure of secrets (introduced in Subsection 3.2 above). Since our schemes rely on specific PIR schemes from the literature, we first review some details of those PIR schemes which are important for our constructions.

4.1 Some Known PIR Schemes

We start by describing a PIR scheme from [12], referred to as the *basic cube scheme*. This scheme is the basis for the 2-database scheme $\mathcal{B}_2$ from [12], also described below, which in turn serves as the basis for the recursive k-database scheme $\mathcal{B}_k$ from [1]. The schemes $\mathcal{B}_k$ and the polynomial interpolation scheme of [12, 3] are described later on, in the proofs of Theorems 6 and 7 respectively.

Basic d-dimensional Cube Scheme: This is a PIR scheme for $k = 2^d$ databases. Assume without loss of generality that the database size is $n = \ell^d$, where ℓ is an integer. The index set $[n]$ can then be identified with the d-dimensional cube $[\ell]^d$, where each index $i \in [n]$ can be naturally identified with a d-tuple $(i_1, \ldots, i_d)$. A d-dimensional *subcube* is a subset $S_1 \times \cdots \times S_d$ of the d-dimensional cube, where each S_m is a subset of $[\ell]$. Such a subcube is denoted by the d-tuple $C = (S_1, \ldots, S_d)$. The $k(= 2^d)$ databases are assigned all of the binary strings of length d, $\mathcal{DB}_\sigma \forall \sigma \in \{0,1\}^d$. The scheme proceeds as follows.

Queries: The user picks a random subcube $C = (S_1^0, \ldots, S_d^0)$, where $S_1^0, \ldots, S_d^0$ are independent random subsets of $[\ell]$. Let $S_m^1 = S_m^0 \oplus i_m$ $(1 \le m \le d)$, where $i = (i_1, \ldots, i_d)$ is the index that the user wishes to retrieve. For each $\sigma = \sigma_1\sigma_2 \cdots \sigma_d \in \{0,1\}^d$, the user sends to database $\mathcal{DB}_\sigma$ the subcube $C_\sigma = (S_1^{\sigma_1}, \ldots, S_d^{\sigma_d})$, where each set $S_m^{\sigma_m}$ is represented by its characteristic ℓ-bit string.

Answers: Each database $\mathcal{DB}_\sigma$, $\sigma \in \{0,1\}^d$, computes the exclusive-or of the data bits residing in the subcube C_σ, and sends the resultant bit b_σ to the user.[5]

Reconstruction: The user computes x_i as the exclusive-or of the k bits b_σ's it has received.

The scheme's correctness follows from the fact that every bit in x except x_i appears in an even number of subcubes C_σ, $\sigma \in \{0,1\}^d$, while x_i appears in exactly one such subcube (see [12] for details). The communication complexity of this 2^d-database scheme is $O(n^{1/d})$, much worse than the following scheme $\mathcal{B}_2$ and its generalization $\mathcal{B}_k$, which achieves communication $O(n^{1/(2k-1)})$ for a constant number of databases k.

The scheme $\mathcal{B}_2$: This scheme may be regarded as a 2-database implementation of the basic 8-database (3-dimensional) cube scheme described above. Let $\ell = n^{1/3}$, and let $i = (i_1, i_2, i_3)$ be the index of the data bit being retrieved. Each of the two databases $\mathcal{DB}_{000}$ and $\mathcal{DB}_{111}$ emulates the 4 databases $\mathcal{DB}_\sigma$, $\sigma \in \{0,1\}^3$, such that the Hamming distance of σ from its own index is at most 1. This is done in the following way. The user sends to $\mathcal{DB}_{000}$ the subcube $C_{000} = (S_1^0, S_2^0, S_3^0)$ and to $\mathcal{DB}_{111}$ the subcube $C_{111} = (S_1^1, S_2^1, S_3^1)$ as in the basic cube scheme. We would like the answers of

[5]The exclusive-or of an empty set of bits is defined to be 0.

each of the two databases to include the 4 answer bits of the 4 databases it emulates. To this end, $\mathcal{DB}_{000}$ replies with its own answer bit b_{000} along with 3 ℓ-bit long strings, each of which contains the answer bit of one of the other databases it emulates. For instance, the i'_1-th bit of the string emulating $\mathcal{DB}_{100}$ is obtained by computing the exclusive-or of all data bits residing in the subcube $(S_1^0 \oplus i'_1, S_2^0, S_3^0)$, implying that the i_1-th bit in this string is equal to b_{100}. Symmetrically, $\mathcal{DB}_{111}$ sends the single bit b_{111} along with 3 ℓ-bit long strings, each of which corresponds to the subcubes obtained from C_{111} by "masking" the set S_m^1 with all ℓ possible values of i_m. Altogether, the user receives 8 answer strings $a_\sigma, \sigma \in \{0,1\}^3$, six of which contain ℓ bits each, and the other two (namely, a_{000} and a_{111}) contain single bits. In each of the ℓ-bit long strings, the required answer bit b_σ can be found in either the i_1 bit of the string (for $\sigma = 100, 011$), the i_2 bit (for $\sigma = 010, 101$), or the i_3 bit (for $\sigma = 001, 110$). Since the user can locate all 8 bits b_σ, $\sigma \in \{0,1\}^3$, in the answer strings, it can reconstruct x_i by computing their exclusive-or.

4.2 The Private Simultaneous Messages (PSM) Model

In a typical PIR scheme, the honest user can extract from the databases' answers more information than just the reconstructed value x_i. Towards solving this problem, we use the following idea. Consider any 1-round PIR scheme. In an execution of such scheme, the user first produces k queries $q_1, \ldots, q_k$, depending on the index i. It then sends each query to the corresponding database and in response receives k answer strings $a_1, \ldots, a_k$. Finally, the user applies a *reconstruction function* Ψ to obtain the desired bit x_i. Our idea is to have the user compute the output of Ψ without actually getting the answers $a_1, \ldots, a_k$, from which it can obtain more information, but rather get some other messages $m_1, \ldots, m_k$ that keep the privacy of the string x.

Precisely this idea is captured by the model of non-interactive private computation introduced in [16] and further studied in [18], called the *Private Simultaneous Messages (PSM)* model. In this model there are k players, each player P_j holding a private input string y_j, and an external referee called Carol. All players have access to a shared random input, which is unknown to Carol. The goal of a *PSM protocol* is to let Carol evaluate a function $f(y_1, \ldots, y_k)$ without learning any additional information about the inputs $y_1, \ldots, y_k$. The scenario of the PSM protocol is similar to a conditional disclosure protocol (see Subsection 3.2), except that in PSM there is no input to Carol, and there is no other input to the players except $y_1, \ldots, y_k$. More formally, in a PSM protocol each player P_j sends a single message to Carol, based on its private input y_j and the shared random input, and Carol applies some reconstruction function to the k messages she received. A PSM protocol computing a k-argument function f must satisfy the following requirements: (1) *correctness*: for any input tuple $y = (y_1, \ldots, y_k)$ and any shared random input, the value reconstructed by Carol is $f(y)$; and (2) *privacy*: given any two input tuples $y = (y_1, \ldots, y_k), y' = (y'_1, \ldots, y'_k)$ such that $f(y) = f(y')$, the messages viewed by Carol are identically distributed.

The *communication complexity* and the *shared randomness complexity* of a PSM protocol are defined as in the conditional disclosure of secrets model. We denote the communication complexity of a k-player PSM protocol by $c_k(m)$, where m is the total number of input bits held by the k players, and its shared randomness complexity by $d_k(m)$.

In [16, 18] several upper bounds on PSM complexity are obtained. In particular, it is shown that any Boolean function with a branching program of size $S(m)$ (with any partition of the m input bits among k players) can be computed by a PSM protocol whose communication complexity and shared randomness complexity are $O(k \cdot S(m)^2)$ [18]. In general, this quadratic overhead will turn out to be too expensive for our purposes. However, some functions do admit simple PSM protocols with *linear* complexity as we see in the following lemma.

Lemma 4. Let $(G, +, 0), (H, \tilde{+}, \tilde{0})$ be finite Abelian groups, and $f : G^k \to H$ be a linear function (that is, $f((y_1{+}z_1), \ldots, (y_k{+}z_k)) = f(y_1, \ldots, y_k)\tilde{+}f(z_1, \ldots, z_k)$ for all $(y_1, \ldots, y_k), (z_1, \ldots, z_k) \in G^k$). Then, there exists a PSM protocol computing f whose communication complexity and shared randomness complexity are no larger than m, where m is the total number of input bits to f.

Proof. The PSM protocol for f proceeds as follows. Each player P_j masks its input y_j with r_j, setting $w_j \overset{\text{def}}{=} y_j + r_j$, where $(r_1, \ldots, r_k) \in G^k$ is a random shared tuple satisfying $f(r_1, \ldots, r_k) = 0$. Then, P_j sends the masked input w_j to Carol. Carol can now compute $f(w_1, \ldots, w_k) = f((y_1 + r_1), \ldots, (y_k + r_k)) = f(y_1, \ldots, y_k)\tilde{+}f(r_1, \ldots, r_k) = f(y_1, \ldots, y_k)\tilde{+}\tilde{0} = f(y_1, \ldots, y_k)$, which is the desired output value. The privacy of this protocol follows by observing that for any input tuple $y = (y_1, \ldots, y_k)$ and message tuple $w = (w_1, \ldots, w_k)$ such that $f(y) = f(w)$, there exists a unique random input r (namely, $r = w - y$) such that $f(r) = 0$ and the messages induced by the inputs y and the random input r are w. Therefore, every message tuple w such that $f(y) = f(w)$ has the same size support (a singleton), implying identical distribution of all such messages. Finally, the communication and shared randomness complexity are clearly as specified. $\square$

This lemma is used in the sequel, when the groups G, H are the binary strings of a fixed length, and the operation is $\oplus$ (exclusive-or).

Remark 2. (CDS from PSM) Note that the conditional disclosure of secrets (CDS) primitive described in Subsection 3.2 and used in Theorem 2 may be implemented (less efficiently) using PSM computation. Specifically, disclosing a bit s subject to a condition $g(y)$ may be reduced to the PSM computation of the function $f(y, s) = g(y) \wedge s$. Indeed, by the correctness of the PSM protocol for f, if $g(y) = 1$ then Carol can reconstruct $s = g(y) \wedge s$. On the other hand, if $g(y) = 0$ then, by the privacy of the PSM protocol, Carol's view is identically distributed under the inputs $(y, 0)$ and $(y, 1)$, implying that Carol learns nothing about s. However, the general upper bound on the complexity of conditional disclosure of secrets, established by Theorem 2, is *linear* in the size of a formula (or a span program) computing the condition, whereas best known results on PSM complexity yield a bound which is quadratic in such representation size. This is because every function with formula size $S(m)$ is also computable by a branching program of size $S(m) + 1$ (see [27, Chapter 14]). This, as mentioned above, gives a PSM complexity of $O(S(m)^2)$.

4.3 SPIR Schemes Based on PSM and CDS Protocols

In this subsection we use PSM and CDS protocols to construct honest-user SPIR schemes. First, in lemma 5 we apply PSM solutions to a PIR scheme with a particular type of reconstruction function in order to get an honest-user SPIR scheme. We then discuss the implications of this lemma and provide an example in which it is used. This example and lemma are also helpful in our later constructions, in particular ones which involve PIR schemes with a more general reconstruction function.

Lemma 5. Suppose $\mathcal{P}$ is a 1-round k-database PIR scheme with communication complexity $(\alpha_k(n), \beta_k(n))$, such that: (1) the reconstruction function Ψ depends only on the answers of the databases, and (2) the function Ψ can be computed by a PSM protocol whose communication complexity is $c_k(m)$ and whose shared randomness complexity is $d_k(n)$. Then, there exists a 1-round k-database honest-user SPIR scheme $\mathcal{S}$ whose communication complexity is $(\alpha_k(n), c_k(\beta_k(n)))$ and whose shared randomness complexity is $d_k(\beta_k(n))$.

Proof. A scheme $\mathcal{S}$ of the specified complexity can be obtained from $\mathcal{P}$ as follows. The user chooses queries $q_1, \ldots, q_k$ as it does in the PIR scheme $\mathcal{P}$ and sends each query q_j to the corresponding database $\mathcal{DB}_j$. Each database $\mathcal{DB}_j$ computes its answer a_j as it would do in $\mathcal{P}$, but instead of sending the answer to the user, the databases (using their shared randomness) simulate the PSM computation of $\Psi(a_1, \ldots, a_k)$. That is, each database $\mathcal{DB}_j$ sends to the user the message that player P_j would send to Carol in the PSM protocol for Ψ. The correctness and privacy of $\mathcal{S}$ follow from the correctness and privacy of $\mathcal{P}$ and of the PSM protocol for Ψ, and the complexity is clearly as stated. $\square$

We stress that Lemma 5 only yields *honest-user* SPIR schemes; indeed, a dishonest user can potentially generate "invalid" queries, such that applying the reconstruction function to their answers gives forbidden information which does not follow from any physical data bit. (Here the idea of hiding the input to the reconstruction function will not help, since the dishonest user may get information from the output of the reconstruction function). A direct application of Lemma 5 is given in the following example.

Example 1. PSM-based honest-user SPIR scheme for the d-dimensional cube scheme.. Consider the basic d-dimensional cube scheme from Subsection 4.1, in which the reconstruction function consists of computing the exclusive-or of the k answer bits sent from the databases. This scheme does not maintain data-privacy, since the user learns the exclusive-or of $k = 2^d$ different subsets of data bits. In this case, the extra information can be eliminated by applying Lemmas 4 and 5. Specifically, instead of sending the original answer b_σ, each database $\mathcal{DB}_\sigma$ will send a *masked* answer $b_\sigma \oplus r_\sigma$, where $r = r_{0\cdots00} r_{0\cdots01} \cdots r_{1\cdots10}$ is a $(k-1)$-bit *shared* random string, and $r_{1\cdots11}$ is computed as the exclusive-or of the bits of r. Under the modified scheme, an honest user's view is uniformly distributed among all k-tuples whose exclusive-or is $\bigoplus_{\sigma\in\{0,1\}^d} b_\sigma$, which by the scheme's correctness is equal to the physical bit x_i.

Other PIR schemes with linear reconstruction function, to which Lemma 5 is applicable with no communication overhead, include the polynomial-interpolation schemes for $O(\log n)$ databases of [12, 3], for which (dishonest-user) SPIR counterparts will be given in Subsection 5.2.

Remark 3. (On the generality of Lemma 5) Note that Lemma 5 requires that in the underlying PIR scheme $\mathcal{P}$, the reconstruction function depends only on the answers computed by the databases. While this is the case with the basic cube scheme (see Example 1 above), this is *not* the case with the scheme $\mathcal{B}_2$, for instance, where reconstruction heavily depends on the index i held by the user. In order to satisfy this requirement, any PIR scheme $\mathcal{P}$, whose reconstruction function Ψ may also depend on the the index i and the queries q_j, may be augmented into a PIR scheme $\mathcal{P}'$, whose reconstruction Ψ' depends only on the answers, as follows. First, the user secret-shares the index i between two databases independently of its original queries (e.g., by sending a $\lceil \log_2 n \rceil$-bit random string to one database and the exclusive-or of this random string with the binary representation of i to the other database). Such a sharing of i does not violate the user's privacy and introduces only a minor overhead on the query complexity. Then, each database $\mathcal{DB}_j$ appends to its original answer a_j the query q_j it received (including the share of i). The original reconstruction function Ψ induces a reconstruction function Ψ' for the augmented scheme $\mathcal{P}'$, which depends on the databases' answers alone. Hence, Lemma 5 can be applied to the augmented scheme. However, the complexity of this solution can be prohibitive.

In the remainder of this section we derive an honest-user SPIR scheme from the 2-database PIR scheme $\mathcal{B}_2$.[6] In this case, it is possible to use the PSM methodology of Lemma 5 and Remark 3 to efficiently meet this goal. However, towards constructions in the next sections, we introduce an alternative, conceptually simpler, methodology of using conditional disclosure of secrets *on top* of PSM. A similar methodology may also be useful in different contexts, as will be demonstrated in Subsection 6.3.

Theorem 4. There exists a 2-database honest-user SPIR scheme, $\mathcal{B}'_2$, with communication complexity and shared randomness complexity $O(n^{1/3})$.

Proof. Recall the PIR scheme $\mathcal{B}_2$ (see Section 4.1) and, in particular, its reconstruction function which may be viewed as a two-stage procedure: (1) the user selects a *single* bit from each of 8 answer strings, depending only on the index $i = (i_1, i_2, i_3)$; and (2) the user takes the exclusive-or of the 8 bits it has selected to obtain x_i. Thus, if we let the honest user learn only the exclusive-or of the 8 bits corresponding to i, the data-privacy requirement will be met. This can be achieved by using the conditional disclosure of secrets primitive on top of a PSM protocol computing the exclusive-or of 8 bits. The scheme $\mathcal{B}'_2$, an honest-user SPIR version of $\mathcal{B}_2$, proceeds as follows:

QUERIES: The user sends the subcubes C_{000} to $\mathcal{DB}_{000}$ and C_{111} to $\mathcal{DB}_{111}$, as in the scheme $\mathcal{B}_2$. In addition, the user independently shares the three characteristic vectors χ_{i_m}, $m = 1, 2, 3$, among the two databases. This is done by picking random ℓ-bit strings i_m^0, i_m^1 such that $i_m^0 \oplus i_m^1 = \chi_{i_m}$ and sending the three strings i_m^0 to $\mathcal{DB}_{000}$ and the three strings i_m^1 to $\mathcal{DB}_{111}$. [7]

ANSWERS: Each of the two databases computes 3 answer strings of length $n^{1/3}$ and 1 one bit answer as in the $\mathcal{B}_2$ scheme. Denote by a_σ the answer string emulating $\mathcal{DB}_\sigma$, $\sigma \in \{0,1\}^3$. The databases treat each bit of a string a_σ as an input to a PSM protocol computing the exclusive-or of 8 bits, and using their shared randomness they compute (but do not send) the PSM message sent for each such bit. Under the simple PSM protocol for XOR (see Lemma 4 or Example 1), each such message is by itself a single bit. Let w_σ denote the string obtained by replacing each bit from a_σ by its corresponding PSM message bit. In this case, w_σ is obtained by masking every bit of a_σ with the same random bit r_σ, where the bits $\{r_\sigma\}$ are 8 random bits whose exclusive-or is 0. Finally, for every $\sigma \in \{0,1\}^3$ and $1 \le j \le |w_\sigma|$, the databases use their shared randomness to disclose to the user the j-th bit of w_σ, $(w_\sigma)_j$, subject to an appropriate condition. For $\sigma = 100, 011$ the condition is $(i_1^0)_j \oplus (i_1^1)_j = 1$, for $\sigma = 010, 101$ it is $(i_2^0)_j \oplus (i_2^1)_j = 1$, and for $\sigma = 001, 110$ it is $(i_3^0)_j \oplus (i_3^1)_j = 1$. The single bits w_{000}, w_{111} can be sent in a plain form.

[6]While it is possible to extend our construction to apply to $\mathcal{B}_k$, the k-database generalization from [1], we postpone this generalization to the next section, which deals with the case of a dishonest user.

[7]When the user is honest, this extra sharing of χ_{i_m} is redundant since the characteristic vectors of the sets S_m^0, S_m^1 sent by the user may be viewed as these shares; however, this presentation more closely resembles the solution for a dishonest user, described in the the next section.

Reconstruction: The user reconstructs the eight PSM message bits corresponding to the index i (using the reconstruction function of the conditional disclosure protocol), and computes their exclusive-or to obtain x_i.

The correctness of the above scheme and the user's privacy follow from the correctness and user's privacy of the PIR scheme $\mathcal{B}_2$ and the correctness of the CDS and the PSM schemes used, and are easy to verify. We turn to show that the scheme meets the data-privacy requirement with respect to an honest user. We first introduce some notation. By $A(x, i, r, \rho)$ we denote the 8-tuple of $\mathcal{B}_2$-answers a_σ, computed by the databases in the execution of $\mathcal{B}'_2$ (or $\mathcal{B}_2$) induced by (x, i, r, ρ), where x is the data string, i is the user's input query, r is the shared randomness of the databases, and ρ is the random input of the user. Similarly, by $W(x, i, r, \rho)$ we denote the 8-tuple of PSM strings w_σ, computed by the databases in the corresponding execution of $\mathcal{B}'_2$. Finally, given an 8-tuple $w = (w_\sigma)_{\sigma\in\{0,1\}^3}$ and an index i, we let $w|_i$ denote the restriction of w to the 8 bits corresponding to the index i.

Since the user is honest and by the correctness of $\mathcal{B}_2$, the exclusive-or of the eight bits in $A(x, i, r, \rho)|_i$ is equal to x_i. Thus, by the privacy of the PSM protocol for XOR, it follows that for any x, x', i such that $x_i = x'_i$, any ρ and $z \in \{0,1\}^8$,

$$\Pr_r[W(x, i, r, \rho)|_i = z] \;=\; \Pr_r[W(x', i, r, \rho)|_i = z]. \tag{1}$$

By the secrecy of the conditional disclosure protocol and the independence of its shared randomness from the PSM randomness, it follows that for any x, x', i, ρ, v, and $z \in \{0,1\}^8$ we have:

$$\Pr_r[\mathtt{view}_U(x, i, r, \rho) = v \mid W(x, i, r, \rho)|_i = z] \;=\; \Pr_r[\mathtt{view}_U(x', i, r, \rho) = v \mid W(x', i, r, \rho)|_i = z]. \tag{2}$$

Finally, combining equations (1) and (2) we get that for any x, x', i, ρ, v such that $x_i = x'_i$:

$$\begin{aligned}\Pr_r[\mathtt{view}_U(x,i,r,\rho) = v] &= \sum_{z\in\{0,1\}^8} \Pr_r[\mathtt{view}_U(x,i,r,\rho) = v \mid W(x,i,r,\rho)|_i = z] \cdot \Pr_r[W(x,i,r,\rho)|_i = z] \\ &= \sum_{z\in\{0,1\}^8} \Pr_r[\mathtt{view}_U(x',i,r,\rho) = v \mid W(x',i,r,\rho)|_i = z] \cdot \Pr_r[W(x',i,r,\rho)|_i = z] \\ &= \Pr_r[\mathtt{view}_U(x',i,r,\rho) = v],\end{aligned}$$

concluding the proof of the data-privacy property. (We note that while the above proof explicitly refers to all relevant random variables, in subsequent proofs of a similar nature such detailed analysis will be replaced by higher level arguments.)

It remains to show that the scheme meets the specified complexity bounds. Since the condition for disclosing each of the $O(n^{1/3})$ bits of the strings w_j is of the form "$y_1 \oplus y_2 = 1$" (or equivalently $\overline{y_1} \oplus y_2 = 0$), where y_1, y_2 are single bits, it follows from Lemma 3 (or Theorem 2) that all such masked answer bits can be conditionally disclosed with total communication and shared randomness cost of $O(n^{1/3})$ bits. Altogether, the communication complexity of the scheme and its shared randomness complexity are $O(n^{1/3})$, as required. $\square$

5 Specific SPIR Schemes with Respect to Dishonest Users

In the previous section we were concerned with an honest but curious user. In this section we construct SPIR schemes which guarantee data-privacy with respect to *dishonest* users as well. The following example demonstrates the extra information that a dishonest user may obtain in ordinary PIR schemes and in the honest-user SPIR scheme constructed above.

Example 2. Consider the scheme $\mathcal{B}_2$. Suppose that a user sends the subcube $C_{000} = (\{i_1\}, \{i_2\}, \{i_3\})$ as a (legitimate) query to the first database. Then, the answers of this database alone, which include the bits $x_{(i_1,i_2,i_3)}$, $x_{(i_1,i_2,i_3)} \oplus x_{(j,i_2,i_3)}$, $x_{(i_1,i_2,i_3)} \oplus x_{(i_1,j,i_3)}$, and $x_{(i_1,i_2,i_3)} \oplus x_{(i_1,i_2,j)}$ for all $j \in [n^{1/3}]$, reveal about $3n^{1/3}$ *physical* bits of data. Note that by randomly setting this query an honest user can also learn that many physical data bits, but this occurs with only an exponentially small probability. Moreover, even in the scheme $\mathcal{B}'_2$ (which perfectly maintains data-privacy for an honest user), a dishonest user may similarly obtain $\Theta(n^{1/3})$ physical data bits. To do this, the user sends to the first database the same cube C_{000} as above, and sends to the second database the empty cube $C_{111} = (\emptyset, \emptyset, \emptyset)$. Instead of sharing the characteristic vectors χ_{i_m}, the user will now share three all-ones vectors, which would automatically satisfy all disclosure conditions and allow the user to learn the entirety of the eight strings w_σ. Then, about $3n^{1/3}$ physical bits can be reconstructed from the combined answers of the two databases. For instance, for every $j \in [n^{1/3}]$ the user may reconstruct the bit $x_{(j,i_2,i_3)}$ by computing $w_{000} \oplus (w_{100})_j \oplus (w_{010})_{i_2} \oplus (w_{001})_{i_3} \oplus (w_{011})_1 \oplus (w_{101})_1 \oplus (w_{011})_1 \oplus w_{111}$.

Observe that in the honest-user SPIR scheme $\mathcal{B}'_2$, a dishonest user can cheat in two ways. One way is to improperly share the characteristic vector of its index (e.g., share the all-ones vector instead). The other way is to send invalid $\mathcal{B}_2$-queries. This may give the user extra information even when the index is properly shared, because invalid $\mathcal{B}_2$-queries can make the output of the reconstruction function depend on more than one bit of data. In order to become resilient to dishonest users, any honest-user SPIR scheme can (in principle) be modified to filter every original answer bit using the conditional disclosure primitive, such that the condition tests for the validity of the user's queries. However, the complexity of disclosing each answer bit subject to a full validity test will be prohibitive. In the next subsections we use alternative means to transform the best known PIR schemes into SPIR schemes. All these transformations involve at most a constant multiplicative communication overhead.

5.1 Cube Schemes

In this subsection we construct, for any constant $k \geq 2$, a k-database SPIR scheme whose communication complexity is $O(n^{1/(2k-1)})$ (as of the best known k-database PIR scheme). We first address the 2-database case, from which we then generalize to a k-database scheme.

Theorem 5. There exists a 2-database SPIR scheme, $\mathcal{B}''_2$, with communication complexity and shared randomness complexity $O(n^{1/3})$.

Proof. Assume that $\ell = n^{1/3}$ is an integer. The scheme $\mathcal{B}''_2$ proceeds as follows:

QUERIES: The user sends to $\mathcal{DB}_{000}$ the subcube $C_{000} = (S_1^0, S_2^0, S_3^0)$ and to $\mathcal{DB}_{111}$ the subcube $C_{111} = (S_1^1, S_2^1, S_3^1)$, as in the scheme $\mathcal{B}_2$. In addition, the user independently shares *dense* representations of the index components i_m, $m = 1, 2, 3$ (as opposed to the unary representation in the scheme $\mathcal{B}'_2$). This is done by viewing each index component i_m as an element of Z_ℓ, picking random $\lceil \log_2 \ell \rceil$-bit elements $i_m^0, i_m^1 \in Z_\ell$ such that $i_m^0 + i_m^1 \equiv i_m \pmod{\ell}$, and sending the three strings i_m^0 to $\mathcal{DB}_{000}$ and the three strings i_m^1 to $\mathcal{DB}_{111}$.

Answers: The answers in $\mathcal{B}_2''$ are constructed on top of some intermediate computations from the scheme $\mathcal{B}_2'$. Recall that b_σ denotes the answer from database $\mathcal{DB}_\sigma$ in the basic 3-dimensional cube scheme, a_σ denotes the answer string corresponding to $\mathcal{DB}_\sigma$ in the original scheme $\mathcal{B}_2$, and w_σ denotes the strings constructed by taking the exclusive-or of each bit in the string a_σ with the same random bit r_σ (these correspond to messages in a PSM protocol for computing XOR). Let t_1, t_2, t_3 be shared random strings of length ℓ each, and u_1, u_2, u_3 be shared random bits (these will be used as "masks" to guarantee that the user gets no information on x if the subcubes it sent are not consistent with the index whose binary representation was shared). The databases reply with the following messages:

1. $\mathcal{DB}_{000}$ sends to the user the three bits $v_m^0 \stackrel{\text{def}}{=} \langle \chi_{S_m^0}, t_m \rangle \oplus u_m$, $m = 1, 2, 3$, where $\langle \cdot, \cdot \rangle$ denotes inner product over GF(2). Similarly, $\mathcal{DB}_{111}$ sends the bits $v_m^1 \stackrel{\text{def}}{=} \langle \chi_{S_m^1}, t_m \rangle \oplus u_m$.

2. $\mathcal{DB}_{000}$ sends to the user the bit w_{000}. Similarly, $\mathcal{DB}_{111}$ sends the bit w_{111}.

3. $\mathcal{DB}_{000}, \mathcal{DB}_{111}$ use the SPIR scheme $\mathcal{S}_2^*$ of Corollary 1 to provide the user with a single bit from each of the six ℓ-bit strings $w_{100}, w_{010}, w_{001}$ (known to $\mathcal{DB}_{000}$), and $w_{011} \oplus t_1, w_{101} \oplus t_2, w_{110} \oplus t_3$ (known to $\mathcal{DB}_{111}$)[8], in the positions corresponding to the shared index. This is done by using the user's queries i_m^0, i_m^1 as the queries for the scheme $\mathcal{S}_2^*$, where $m = 1$ for retrieval from w_{100} and $w_{011} \oplus t_1$, $m = 2$ for retrieval from w_{010} and $w_{101} \oplus t_2$, and $m = 3$ for retrieval from w_{001} and $w_{110} \oplus t_3$. Since the index retrieved in the scheme $\mathcal{S}_2^*$ is the sum of the queries to both databases, this means that the user obtains the bits in position i_1 from the first pair of strings, i_2 from the second pair, and i_3 from the third.

Reconstruction: An honest user reconstruct x_i as follows. For $m = 1, 2, 3$ the user reconstructs the bit $(t_m)_{i_m}$ by computing $v_m^0 \oplus v_m^1$. Then, using these 3 bits and the bits obtained from the $\mathcal{S}_2^*$ invocations, it computes

$$\begin{aligned} & (t_1)_{i_1} \oplus (t_2)_{i_2} \oplus (t_3)_{i_3} \oplus w_{000} \oplus w_{111} \oplus (w_{011})_{i_1} \oplus (w_{101})_{i_2} \oplus (w_{110})_{i_3} \\ & \qquad \oplus (w_{100} \oplus t_1)_{i_1} \oplus (w_{010} \oplus t_2)_{i_2} \oplus (w_{001} \oplus t_3)_{i_3} \\ = & \bigoplus_{\sigma \in \{0,1\}^3} b_\sigma \\ = & \; x_i \end{aligned}$$

The correctness and the user's privacy in this scheme are easy to verify. We now show the scheme's data-privacy, relative to *any* user.

Lemma 6. Denote by S_m^b, i_m^b, $b = 0, 1$, $m = 1, 2, 3$, queries sent by a possibly dishonest user, and let $i_m^* \stackrel{\text{def}}{=} i_m^0 + i_m^1 \pmod{\ell}$. If these queries satisfy $S_m^0 \oplus S_m^1 = \{i_m^*\}$ for $m = 1, 2, 3$ then the answers reveal the bit $x_{(i_1^*, i_2^*, i_3^*)}$ and no other information about the data. Otherwise, the answers reveal no information about the data.

Proof. First, observe that using the random bits u_m guarantees that for $m = 1, 2, 3$ the answers v_m^0, v_m^1 are two uniformly distributed bits satisfying $v_m^0 \oplus v_m^1 = \langle \chi_{S_m^0 \oplus S_m^1}, t_m \rangle$. Thus if the user is

[8]Recall that in $\mathcal{S}_2^*$ only one of the two databases needs to know the data, and the other one only needs access to the shared random string.

honest then $S_m^0 \oplus S_m^1 = \{i_m^*\}$ and so the user can obtain $(t_m)_{i_m^*}$, but if $S_m^0 \oplus S_m^1 \neq \{i_m^*\}$ then the messages (v_m^0, v_m^1), $m = 1,2,3$, jointly give no information about $(t_m)_{i_m^*}$. (Note that in the latter case a user may learn the exclusive-or of the bit $(t_m)_{i_m^*}$ with other bits in t_m, but this still gives no information on $(t_m)_{i_m^*}$.)

Next, observe that the data privacy of the SPIR scheme $\mathcal{S}_2^*$ guarantees that the user learns a single physical bit from each of the six ℓ-bit strings to which the scheme was applied. Moreover, the position of this bit corresponds to a shared index component i_m^*. By the properties of the underlying PSM protocol, the only information revealed by these bits is their exclusive-or which is

$$\Big(\bigoplus_{\sigma \in \{0,1\}^3} b_\sigma\Big) \oplus (t_1)_{i_1^*} \oplus (t_2)_{i_2^*} \oplus (t_3)_{i_3^*}. \tag{3}$$

Altogether, the only information on x the user can obtain is what follows from $\langle \chi_{S_m^0 \oplus S_m^1}, t_m \rangle$ and the outcome of expression (3) above. Now, if $S_m^0 \oplus S_m^1 = \{i_m^*\}$ for $m = 1,2,3$ then $\bigoplus_{\sigma \in \{0,1\}^3} b_\sigma = x_{(i_1^*, i_2^*, i_3^*)}$, implying that $x_{i_1^*, i_2^*, i_3^*}$ is the only information on x learned by the user. On the other hand, if $S_m^0 \oplus S_m^1 \neq \{i_m^*\}$ for some m, then there exists some m for which the user gets no information about $(t_m)_{i_m^*}$, and thus it learns no information about the data. □

Finally, using Corollary 1 the $\mathcal{S}_2^*$ invocations can be implemented with a total of $O(\ell)$ communication complexity and shared randomness complexity. Thus, the scheme meets the specified complexity bounds. □

We note that the SPIR scheme $\mathcal{B}_2''$ constructed above is in fact as communication efficient as the PIR scheme $\mathcal{B}_2$ up to an *additive* logarithmic overhead.

Next, we give a k-database generalization of Theorem 5.

Theorem 6. For every *constant* $k \geq 2$ there exists a k-database SPIR scheme, $\mathcal{B}_k''$, with communication complexity and shared randomness complexity $O(n^{1/(2k-1)})$.

Proof. We start by giving a short description of the PIR scheme $\mathcal{B}_k$ from [1]. Let $d = 2k-1$ and $\ell = n^{1/d}$. In the scheme $\mathcal{B}_k$, the k databases (denoted $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$) jointly emulate the 2^d databases of the d-dimensional cube scheme. The scheme proceeds as follows. The user sends to $\mathcal{DB}_1$ the subcube C_{0^d} as in the basic cube scheme, and sends to each of $\mathcal{DB}_2, \ldots, \mathcal{DB}_k$ the subcube C_{1^d}. In its answers, $\mathcal{DB}_1$ emulates all databases $\mathcal{DB}_\sigma$ of the original scheme such that $\sigma \in \{0,1\}^d$ is at Hamming distance at most 1 from 0^d, similarly to the way such an emulation is done in the scheme $\mathcal{B}_2$. Simultaneously, the remaining databases $\mathcal{DB}_2, \ldots, \mathcal{DB}_k$ jointly emulate the remaining databases of the original scheme, namely all $\mathcal{DB}_\sigma$ such that σ contains at least two 1's. This is done using a constant number $(2^d - d - 1)$ of recursive invocations of the scheme $\mathcal{B}_{k-1}$ between the user and $\mathcal{DB}_2, \ldots, \mathcal{DB}_k$. In each such invocation the user retrieves a single bit b_σ from a virtual data string, whose entries correspond to the different subcubes possibly sent to $\mathcal{DB}_\sigma$ in the basic cube scheme (i.e., each bit of the virtual data strings is the exclusive-or of data bits residing in such a potential subcube). By taking the exclusive-or of the $d+1$ bits selected from the answers of $\mathcal{DB}_1$ together with the $2^d - d - 1$ bits retrieved by the recursive invocations of $\mathcal{B}_{k-1}$, the user reconstructs x_i.

We now show how to adapt the proof of Theorem 5 to this k-database generalization. Intuitively, we combine the recursive construction outlined above with the techniques used for constructing the scheme $\mathcal{B}_2''$ (of Theorem 5). Note that in $\mathcal{B}_2''$ each of the two databases had a role as a "main database" having some information to send to the user, as well as an "auxiliary database" to help the other database disclose its own information without revealing any extra information. Similarly in $\mathcal{B}''_k$ we will have $\mathcal{DB}_1$ be the "main database" in emulating the databases $\mathcal{DB}_\sigma$ of Hamming distance at most 1 from 0^d in the original cube scheme, and $\mathcal{DB}_2$ be the "auxiliary database" for this purpose. In addition, $\mathcal{DB}_2, \ldots, \mathcal{DB}_k$ will recursively emulate the other databases of the original cube scheme, as in the scheme $\mathcal{B}_k$ described above. We start by describing the induction assumption we will be using, followed by a description of the scheme.

Suppose we have a $(k-1)$-database SPIR scheme $\mathcal{B}''_{k-1}$ of communication complexity and shared randomness complexity $O(n^{1/(2k-3)})$. In this case we make an additional assumption on $\mathcal{B}''_{k-1}$: we assume that the user is required to *commit* to the index being retrieved. This assumption is made precise in the following way. We say that a 1-round PIR scheme $\mathcal{P}$ satisfies the *strong data-privacy requirement* with parameter d', if the following conditions hold:

1. On a data string x of length $n' = \ell^{d'}$, the user sends special queries Q^0_m, Q^1_m, $1 \le m \le d'$ (each of which is an element of Z_ℓ); and

2. If a user (possibly a dishonest user) sends queries in which $Q^0_m + Q^1_m \equiv i^*_m \pmod{\ell}$ for each $1 \le m \le d'$, then the answers reveal at most the bit $x_{(i^*_1, \ldots, i^*_{d'})}$.

Notice that strong data-privacy implies the usual data-privacy. Also note that the scheme $\mathcal{B}''_2$ satisfies this stronger requirement with $d' = 3$, as follows from Lemma 6. Our additional assumption on $\mathcal{B}''_{k-1}$ (which will be carried on to $\mathcal{B}''_k$) is that it satisfies the strong data-privacy requirement with $d' = 2(k-1) - 1 = 2k - 3$. The scheme $\mathcal{B}''_k$ proceeds as follows:

QUERIES: The user sends to $\mathcal{DB}_1$ the subcube $C_{0^d} = (S^0_1, \ldots, S^0_d)$ and to each of $\mathcal{DB}_2, \ldots, \mathcal{DB}_k$ the subcube $C_{1^d} = (S^1_1, \ldots, S^1_d)$. In addition, the user independently shares dense representations of the index components i_m, $m = 1, 2, \ldots, d$, between $\mathcal{DB}_1$ and $\mathcal{DB}_2$, using additive shares over Z_ℓ as in the scheme $\mathcal{B}''_2$. Finally, the user sends the queries necessary for the recursive invocations of $\mathcal{B}''_{k-1}$ described in item 4 below.

ANSWERS: As before, let w_σ denote the strings corresponding to the PSM message strings for emulating database $\mathcal{DB}_\sigma$ in the d-dimensional cube scheme. For σ such that weight$(\sigma) \ge 2$ these strings are described below, whereas for σ of weight 0 or 1 these can be constructed from the query C_{0^d} exactly as before. In particular, we consider w_{e_m} where e_m denotes the m-th unit vector of length d (note that the databases whose index is in Hamming distance at most 1 from 0^d are $\mathcal{DB}_{0^d}$ and $\mathcal{DB}_{e_m}$ $1 \le m \le d$, and they can be emulated by $\mathcal{DB}_{0^d}$ as before). Let $t_1, t_2, \ldots, t_d$ be shared random strings of length ℓ, and $u_1, u_2, \ldots, u_d$ be shared random bits. The databases reply with the following messages:

1. $\mathcal{DB}_1$ sends to the user the d bits $v^0_m \stackrel{\text{def}}{=} \langle \chi_{S^0_m}, t_m \rangle \oplus u_m$, $1 \le m \le d$. Similarly, $\mathcal{DB}_2$ sends the bits $v^1_m \stackrel{\text{def}}{=} \langle \chi_{S^1_m}, t_m \rangle \oplus u_m$.

2. $\mathcal{DB}_1$ sends the bit $w_{0^d} \oplus s$, where s is a shared random bit (to be conditionally disclosed in item 5 below).

3. $\mathcal{DB}_1$ computes all ℓ-bit long PSM message strings w_{e_m}, $1 \le m \le d$, emulating databases $\mathcal{DB}_{e_m}$ in the d-dimensional cube scheme. Then $\mathcal{DB}_1$ and $\mathcal{DB}_2$ use the SPIR scheme $\mathcal{S}^*_2$ to provide the user with the bit in position i_m of each string $w_{e_m} \oplus t_m$. Like in the scheme $\mathcal{B}''_2$, this is done by using the shares of i_m as the queries in $\mathcal{S}^*_2$.

4. For each $\sigma \in \{0,1\}^d$ such that $\text{weight}(\sigma) \geq 2$, the user and the databases $\mathcal{DB}_2, \mathcal{DB}_3, \ldots, \mathcal{DB}_k$ recursively invoke $\mathcal{B}''_{k-1}$ on the virtual data string w_σ defined in the following. Let $d' = d-2$ and $n' = \ell^{d'}$. Let m^σ_z, $1 \leq z \leq \text{weight}(\sigma)$, denote the position of the z-th zero in σ. With every σ such that $\text{weight}(\sigma) \geq 2$ and tuple $i' = (i'_1, \ldots, i'_{d'}) \in [\ell]^{d'}$ we associate a subcube $C^\sigma_{i'}$ (of the cube $[\ell]^d$), which is obtained from C_{1^d} by replacing each set S^1_z, $1 \leq z \leq \text{weight}(\sigma)$, with the set $S^1_z \oplus i_{m^\sigma_z}$. Each w_σ is defined to be the n'-bit string, whose i'-th bit is equal to the exclusive-or of data bits residing in the subcube $C^\sigma_{i'}$ together with the PSM random bit r_σ. In a recursive invocation of $\mathcal{B}''_{k-1}$ on the virtual data string w_σ, the user retrieves the bit whose index is represented by the d'-tuple $i'_\sigma = (i_{m^\sigma_1}, i_{m^\sigma_2}, \ldots, i_{m^\sigma_p}, 1, \ldots, 1)$, where $p = \text{weight}(\sigma)$.

5. The databases conditionally disclose the shared bit s subject to a conjunction of the following conditions:

   (a) For every $3 \leq j \leq k$, the subcube sent to $\mathcal{DB}_j$ is equal to the subcube sent to $\mathcal{DB}_2$.

   (b) For every $\sigma \in \{0,1\}^d$ such that $\text{weight}(\sigma) \geq 2$, the index i' shared by the user in the invocation of $\mathcal{B}''_{k-1}$ on w_σ (in accordance with the strong data-privacy assumption made on $\mathcal{B}''_{k-1}$) is equal to i'_σ. This can be verified by comparing each component of i' with the corresponding component of i as shared by the user.

   (For efficiently disclosing s under the conjunction of all these conditions, the databases may write s as the exclusive-or of several independent random bits, and disclose each of these bits subject to a single condition of equality between two strings).

RECONSTRUCTION: The user reconstructs x_i by recursively reconstructing the bits retrieved via $\mathcal{B}''_{k-1}$, and taking their exclusive-or with all other bits disclosed to the user.

We start by analyzing the communication and shared randomness complexity. By Lemma 3 and Corollary 1, the conditional disclosure of the bit s and the SPIR retrievals from the strings $w_{e_m} \oplus t_m$ can be implemented with $O(\ell)$ communication and shared randomness complexity, for a constant k. Thus, by induction (using $\mathcal{B}''_2$ as basis) the communication complexity is $c_k(n) = O(\ell) + (2^d - d - 1) \cdot c_{k-1}(\ell^{d-2}) = O(\ell) = O(n^{1/(2k-1)})$, and similarly the shared randomness complexity is also $O(n^{(1/(2k-1)})$.

The correctness and the user's privacy can be easily verified. It remains to show that the strong data-privacy requirement also holds for $\mathcal{B}''_k$. We argue that if the user commits to an index $i = (i_1, \ldots, i_d)$ (by sharing its components between $\mathcal{DB}_1$ and $\mathcal{DB}_2$), then it can learn at most the bit x_i. As in the $\mathcal{B}''_2$ scheme, an *honest* user learns x_i alone. In order to learn some information involving other bits, a dishonest user must deviate from the scheme's specification either by sending to $\mathcal{DB}_1, \ldots, \mathcal{DB}_k$ subcubes which don't meet the requirements imposed by i, or by trying to retrieve from the recursive invocations of $\mathcal{B}''_{k-1}$ different bits than those corresponding to i. The specified disclosure conditions, the data privacy of $\mathcal{S}^*_2$, and the strong data-privacy assumption made on $\mathcal{B}''_{k-1}$ guarantee that in both of these cases, the user will learn no information at all. □

5.2 A Polynomial Interpolation Based Scheme

In this section we prove that the polynomial interpolation based PIR scheme for $k = \lceil \log_2 n + 1 \rceil$ databases from [12] (see also [3]) can be transformed into a SPIR scheme with the same number of databases and a constant factor of communication and randomness overhead.

Theorem 7. There exists a $\lceil \log_2 n + 1\rceil$-database SPIR scheme, with communication complexity and shared randomness complexity $O(\log^2 n \cdot \log\log n)$.

Proof. We start by describing the underlying PIR scheme, which is based on the method of low-degree polynomial interpolation (see [3, 12] for more details). Assume without loss of generality that $n = 2^s$, where s is a positive integer, and let $k = s+1$ be the number of databases. Let GF(q) be a finite field with at least $k+1$ elements, and α_j, $1 \le j \le k$, be distinct, nonzero elements of GF(q). With every index $i \in [n]$ we associate an s-tuple $\vec{i} = (i_1, i_2, \ldots, i_s) \in \{0,1\}^s$, corresponding to the binary representation of i. For each data string $x \in \{0,1\}^n$, let $p_x(y_1, \ldots, y_s)$ denote a multivariate degree-s polynomial such that $p_x(\vec{i}) = x_i$ for every $i \in [n]$ (such p_x may be taken to be the multilinear extension of the function $f(\vec{i}) \overset{\text{def}}{=} x_i$). The user picks a random s-tuple $\vec{c} = (c_1, \ldots, c_s) \in \text{GF}(q)^s$, and sends to each database $\mathcal{DB}_j$, $1 \le j \le k$, the query $\vec{u}^j = \alpha_j \cdot \vec{c} + \vec{i}$. Each database $\mathcal{DB}_j$ replies with a single field element $a_j \overset{\text{def}}{=} p_x(\vec{u}^j)$. The user reconstructs x_i by interpolation: if p' is the unique degree-s univariate polynomial (over GF(q)) such that $p'(\alpha_j) = a_j$ for every $1 \le j \le k$, then $x_i = p'(0)$. The communication complexity of this scheme is $O(\log^2 n \log\log n)$.

As noted in Subsection 4.3, the linearity of the reconstruction function (interpolation) allows to obtain a PSM-based honest-user SPIR scheme with the same communication complexity. To prevent a dishonest user from obtaining any illegitimate information on x, we require the user to prove that its queries are consistent with some $\vec{i} \in \{0,1\}^s$ and $\vec{c} \in \text{GF}(q)^s$. Such a proof will consist of sharing each entry of $\vec{c}$ and $\vec{i}$, and its validation will consist of verifying that $\vec{i} \in \{0,1\}^s$ and that $\vec{u}^j = \alpha_j \cdot \vec{c} + \vec{i}$ for each $1 \le j \le k$.

We begin with the following observation, which also yields a slight improvement to the original PIR scheme described above. Note that the user reconstructs x_i by computing some *fixed* linear combination over GF(q) of the k field elements replied by the databases. Thus, as a first step, we can let each database multiply its original answer by the corresponding coefficient, so that reconstruction will consist of computing the *sum* of all answers over GF(q). Then, if q is chosen to be a power of 2 ($q = 2^{\lceil \log_2(k+1)\rceil}$ suffices), it is enough for the databases to reply only with the "least significant bit" of each answer, and for the user to reconstruct x_i by taking the exclusive-or of the k answer bits. From now on we refer to this modified scheme. The corresponding SPIR scheme we construct is formally described as follows:

QUERIES: The user sends to each database $\mathcal{DB}_j$ a query $\vec{u}^j$ as in the original scheme. In addition, the user picks random tuples $\vec{i}^0, \vec{i}^1, \vec{c}^0, \vec{c}^1 \in \text{GF}(q)^s$ such that $\vec{i}^0 + \vec{i}^1 = \vec{i}$ and $\vec{c}^0 + \vec{c}^1 = \vec{c}$, and sends $\vec{i}^0, \vec{c}^0$ to $\mathcal{DB}_1$ and $\vec{i}^1, \vec{c}^1$ to each of $\mathcal{DB}_2, \ldots, \mathcal{DB}_k$.

ANSWERS: Let $r_1, r_2, \ldots, r_k$ be independent random bits (included in the databases' shared randomness), and let r denote their exclusive-or. Each database $\mathcal{DB}_j$ replies with $a'_j \overset{\text{def}}{=} a_j \oplus r_j$, where a_j is its answer according to the modified scheme. In addition, the databases use their shared randomness to disclose the bit r, subject to a conjunction of the following conditions: (1) for every $3 \le j \le k$, the shares of $\vec{i}$ and $\vec{c}$ sent to $\mathcal{DB}_j$ are identical to those sent to $\mathcal{DB}_2$; (2) for every $1 \le m \le s$, either $i^0_m + i^1_m = 0$ or $i^0_m + i^1_m = 1$ (where i^b_m denotes the m-th entry of the b-th share of $\vec{i}$); and finally (3) for every $1 \le j \le k$ and $1 \le m \le s$, $\alpha_j(c^0_m + c^1_m) + (i^0_m + i^1_m) = u^j_m$. Note that the above condition may be expressed by a Boolean formula over $O(ks) = O(\log\log n)$ atomic conditions, each testing equality between two elements of GF(q) known to two different databases. For instance, if $j > 1$ then verifying the condition $\alpha_j(c^0_m + c^1_m) + (i^0_m + i^1_m) = u^j_m$ is equivalent to comparing $\alpha_j c^0_m + i^0_m$, which is known to $\mathcal{DB}_1$, and $u^j_m - \alpha_j c^1_m - i^1_m$, which is known to $\mathcal{DB}_j$. Using

Theorem 2, the conditional disclosure of r can be implemented with communication complexity and shared randomness complexity of $O(\log^2 n \cdot \log\log n)$.

RECONSTRUCTION: The user reconstructs r, and computes x_i as the exclusive-or of $a'_1, \ldots, a'_k$ and r.

The correctness and the user's privacy of the original scheme are clearly maintained. To see the data-privacy of this scheme, consider two possible cases. If the user's queries are valid, then the tuple $(a'_1, a'_2, \ldots, a'_k, r)$ is uniformly distributed among all $(k+1)$-tuples over GF(2) which add up to x_i, implying that the answer distribution depends only on x_i. Otherwise, the user obtains no information on r, and consequently $a'_1, \ldots, a'_k$ (which are uniformly and independently distributed over GF(2)) are independent of the conditional disclosure messages. It follows that in the latter case the user obtains no information on x.

Excluding the conditional disclosure of r, the communication complexity of the scheme is dominated by the query complexity, which is $O(\log^2 n \cdot \log\log n)$. Together with the complexity of disclosing r, which is discussed above, the entire scheme requires $O(\log^2 n \cdot \log\log n)$ communication and shared randomness bits. $\square$

6 Conclusion and Extensions

We have presented a methodology which allows to implement communication efficient SPIR schemes, requiring only one round of interaction and withstanding any dishonest behavior of the user. This methodology may be useful for dealing with other variants of the basic PIR question, as we demonstrate in this section, as well as in other cryptographic scenarios. In the following we show how to extend our results in two directions: dealing with retrieval of *blocks* instead of single-bit records; and dealing with t-privacy, namely privacy against *coalitions* of up to t colluding databases. We also present an application which using our methodology for SPIR, and in particular the conditional disclosure of secrets primitive, can be implemented quite efficiently. This application, termed *private retrieval with costs*, allows a user to privately retrieve (in a single round) any collection of data items, provided that their total cost does not exceed what it had previously paid for.

6.1 Block Retrieval SPIR schemes

So far, we have restricted our attention to retrieval of *single* bits rather than *multi-bit* records, also referred to as *blocks*. In this subsection we show how results from the previous sections can be extended to yield block-retrieval SPIR schemes.

We start by observing that for PIR schemes generality is not lost when only single bit retrieval is considered: any PIR scheme for single bit retrieval may simply be invoked ℓ times in parallel to retrieve a block of ℓ bits. However this argument does not carry on to SPIR schemes, because a cheating user may invoke the scheme on ℓ bits which do not belong to the same record, thus obtaining information about more than one physical block. Therefore, we describe a modification of the above procedure which works for single round SPIR schemes.

Given a single round SPIR scheme where the user can retrieve a single bit out of the n-bit data string, one can construct a (single round) SPIR scheme to retrieve an ℓ-bit record from a data string of n such records as follows: the user sends queries as in the original bit-retrieval scheme, and the databases reply ℓ times to the user's queries, once for each bit of the record. Each such reply allows the user to learn a single bit of the selected record, and since the user generates queries only once it is guaranteed that the ℓ bits that it learns indeed form a single record of the database.

The above transformation from single-bit to multi-bit retrieval is not applicable for multi-round SPIR schemes, since the same set of queries cannot be used multiple times for different record bits (queries for each bit must depend on replies received in previous rounds). On the other hand, for multi-round schemes, our general PIR to SPIR transformation of Section 3 may be extended to work for multi-bit block retrieval, by letting each entry of the shared random string r consist of ℓ bits instead of a single bit. The protocols and their proofs can be modified in a straightforward way to support this extension. In addition, note that all our specific SPIR schemes (Sections 4,5) are single round, and thus may be used for block retrieval by the above transformation. This is also true for our general SPIR scheme (Section 3), when used with an underlying single round PIR scheme (which is the case for most PIR schemes known in the literature).

6.2 t-private SPIR schemes

In the general reduction described in Section 3, even if the original PIR scheme $\mathcal{P}$ is t-private for some $t > 1$, the resultant SPIR scheme $\mathcal{S}_{\mathcal{P}}$ will still only be 1-private. This is because if $\mathcal{DB}_0$ colludes with any other database $\mathcal{DB}_j$, the joint view of these two colluding databases includes both the shift Δ and the shifted index $i' = (i - \Delta) \bmod n$, from which the user's index i can easily be recovered. Generalizing the construction of $\mathcal{S}_{\mathcal{P}}$, a *$t$-private* SPIR scheme $\mathcal{S}^t_{\mathcal{P}}$ can be obtained from any t-private PIR scheme $\mathcal{P}$ as follows. Instead of directly asking $\mathcal{DB}_0$ for the $(i-\Delta)$-th bit of the shared random string r, the user can retrieve this bit by recursively invoking the $(t-1)$-private SPIR scheme $\mathcal{S}^{t-1}_{\mathcal{P}}$ with a "fresh" set of databases. As a basis $\mathcal{S}^0_{\mathcal{P}}$ for this recursion, we may take the trivial 1-database scheme in which the user explicitly asks for the desired index. In particular, the $(k+1)$-database 1-private scheme described in Section 3 may be viewed as the second level of the recursion. In general, for any t-private k-database PIR scheme $\mathcal{P}$, applying this recursion yields a t-private $(kt+1)$-database SPIR scheme $\mathcal{S}_{\mathcal{P}}$ whose communication complexity is roughly t times that of our original (1-private) scheme.

In the following generalization of Theorem 3 we show that the number of databases in the t-private SPIR scheme can be reduced to $k+t$, at the expense of increasing communication by a factor of $\binom{k+t-1}{t-1}$.

Theorem 8. Let $\mathcal{P}$ be any 1-round, k-database, t-private PIR scheme with communication complexity $(\alpha_k(n), \beta_k(n))$. Then, there exists a 1-round, $(k+t)$-database, t-private SPIR scheme $\mathcal{S}_{\mathcal{P}}$ with communication complexity $(O(m(\alpha_k(n) + \lceil \log_2 n \rceil), O(m\beta_k(n)))$ and shared randomness complexity $O(mn)$, where $m = \binom{k+t-1}{t-1}$.

Proof. A t-private SPIR scheme $\mathcal{S}_{\mathcal{P}}$ using $K = k+t$ databases $\mathcal{DB}_1, \ldots, \mathcal{DB}_K$ is described in the following. The construction uses a collection $\mathcal{F} = \{S_1, \ldots, S_m, S_{m+1}\} \subseteq 2^{[K]}$ of database sets such that:

- S_{m+1} is a singleton;
- each other set S_h, $1 \le h \le m$, is of size k;
- for any set $T \subseteq [K]$ of size t, there exists a set $S \in \mathcal{F}$ such that $T \cap S = \emptyset$.

Such $\mathcal{F}$ exists with $m = \binom{k+t-1}{t-1}$. E.g., let $S_{m+1} = \{K\}$, and for any subset $T \subseteq [K]$ of size t such that $K \in T$, let $S_T = [K] \setminus T$. [9]

An honest-user SPIR scheme can now proceed as follows (where all actions are performed using one round of communication):

- The user $\mathcal{U}$ picks m random shift amounts $\Delta_1, \Delta_2, \ldots, \Delta_m \in Z_n$;
  The databases hold m shared random strings $r_1, \ldots, r_m$, of length n each, and let $r_0 = x$ denote the data string.

- For $1 \leq j \leq m$, $\mathcal{U}$ sends Δ_j to each database in S_j, and invokes the PIR scheme $\mathcal{P}$ with database set S_j to privately retrieve the bit b_j in position $i_j \overset{\text{def}}{=} i - \sum_{h=1}^{j-1} \Delta_j \pmod n$ of $r_{j-1} \oplus (r_j \gg \Delta_j)$. (Notice that in particular, $i_1 = i$);

- $\mathcal{U}$ explicitly asks the single database in S_{m+1} for the bit b_{m+1} in position $i_{m+1} \overset{\text{def}}{=} i - \sum_{h=1}^{m} \Delta_h \pmod n$ of r_m;

- $\mathcal{U}$ reconstructs x_i by taking the exclusive-or of the $m+1$ bits $b_1, \ldots, b_m, b_{m+1}$.

We now show that the scheme is correct, and that it satisfies both privacy requirements. It follows by induction that for $h = 1, 2, \ldots, m$, $b_1 \oplus b_2 \oplus \cdots \oplus b_h = x_i \oplus (r_h)_{i_{h+1}}$, and so $(b_1 \oplus b_2 \oplus \cdots \oplus b_m) \oplus b_{m+1} = (x_i \oplus (r_m)_{i_{m+1}}) \oplus b_{m+1} = x_i$. This proves the correctness of the scheme.

To prove the user's privacy, consider the view of a collusion T of t databases. Since $\mathcal{P}$ is t-private, invocations of $\mathcal{P}$ involving members of T do not disclose any information about i. The only potential source of information about i are those messages from the set $\{\Delta_1, \Delta_2, \ldots, \Delta_m, i_{m+1}\}$ that are viewed by members of T. However, the definition of $\mathcal{F}$ guarantees that the collusion T will only view a proper subset of these messages, which contains no information on i.

To prove the data-privacy (against an honest user), it suffices to show that given *any* shift amounts $\Delta_1, \ldots, \Delta_m$ and position i_{m+1} picked by the user, the random variable

$$\left(x \oplus (r_1 \gg \Delta_1), r_1 \oplus (r_2 \gg \Delta_2), r_2 \oplus (r_3 \gg \Delta_3), \ldots, r_{m-1} \oplus (r_m \gg \Delta_m), (r_m)_{i_{m+1}}\right),$$

where the strings $r_1, \ldots, r_m$ are uniformly and independently distributed over $\{0,1\}^n$, depends only on the single data bit x_i, where $i = i_m + \sum \Delta_h$. This can be proved by iterating the argument used in the proof of Theorem 1. Letting $r_0 = x$, it can be shown by backward induction on h that for $h = m-1, m-2, \ldots, 0$, the joint distribution $(r_h \oplus (r_{h+1} \gg \Delta_{h+1}), r_{h+1} \oplus (r_{h+2} \gg \Delta_{h+2}), \ldots, r_{m-1} \oplus (r_m \gg \Delta_m), (r_m)_{i_{m+1}})$ is independent of r_h given $(r_h)_{i_h}$, where $i_h = i_{m+1} + \Delta_m + \Delta_{m-1} + \ldots + \Delta_{h+1} \pmod n$. In particular, for $i = 0$ we obtain the desired result.

Finally, the same conditional disclosure mechanism used in the proof of Theorem 3 can be used here as well to guarantee data-privacy against *any* (possibly dishonest) user. Specifically, in any invocation of $\mathcal{P}$ involving database set S_h, each answer should be disclosed subject to the condition that all corresponding shift amounts sent by the user are equal. The above analysis shows that this suffices to guarantee data-privacy.

Aside from the conditional disclosure protocol, the communication in the resultant scheme $\mathcal{S}_{\mathcal{P}}$ involves m invocations of the scheme $\mathcal{P}$, m extra $\log n$-bit query strings, and one extra answer bit.

[9]It is not hard to observe that the described $\mathcal{F}$ is of minimal cardinality, and that it cannot exist at all for K smaller than $k+t$. However, by *increasing* the number of databases K, the cardinality of $\mathcal{F}$ can be decreased. For instance, m can be made as low as t when $K = tk+1$, corresponding to the recursive scheme described above.

The conditional disclosure protocol induces a constant multiplicative communication and shared randomness overhead. This gives the communication and randomness bounds stated in the theorem. □

6.3 Private Retrieval with Costs

In this subsection we briefly sketch how the conditional disclosure of secrets methodology can be used together with an underlying SPIR scheme to implement private retrieval with costs.

Let $i_1, \ldots, i_m$ denote the indices of the data records which the user wishes to retrieve,[10] c denote a public vector of ℓ-bit integral costs (an n-tuple whose i-th entry c_i contains a binary representation of the cost of the i-th data record $(0 \le c_i \le 2^\ell - 1)$), and p denote a public cost threshold (i.e., the amount of money paid by the user). A scheme for private retrieval with costs allows the user to retrieve the data records indexed by $i_1, \ldots, i_m$ privately (namely without giving the database any information about $i_1, \ldots, i_m$), provided that $\sum_{h=1}^{m} c_{i_h} \le p$ (i.e. the total cost of the records does not exceed the amount pre-paid by the user); on the other hand, it should not allow the user to obtain any information which does not follow from such valid set of records.

The following is a high-level description of a generic implementation of such a scheme, using an underlying (1-round) SPIR scheme $\mathcal{S}$. Without loss of generality (but possibly with a small complexity overhead), we may assume that the reconstruction function applied by the user in $\mathcal{S}$ depends on the answers alone, and not on the index i or its random input ρ. (See Remark 3; also, notice that this is already the case with the schemes $\mathcal{B}''_k$ constructed in Section 5.) The scheme can then proceed as follows.

Queries: The user chooses independently, for each desired retrieval index i_h of x $(1 \le h \le m)$, a k-tuple of queries according to the scheme $\mathcal{S}$. It sends to each of the k databases the m corresponding messages (all in parallel).

Answers: Each database locally computes two answers to each of the user's queries: one by considering x as the data string, and the other by considering the cost vector c as the data string (more precisely, c is considered as ℓ n-bit vectors and the ℓ answers can be used to construct the ℓ-bit entry c_{i_h}). Then, the databases conditionally disclose their x-answers subject to an appropriate condition on the c-answers. That is, the condition on the c-answers should assert that the sum of the costs reconstructed from these answers (each of which can be obtained by applying the reconstruction function of $\mathcal{S}$) is no larger than the public threshold p.

The complexity of realizing conditional disclosure as above can be kept low in the following ways. First, it is better to use an underlying scheme $\mathcal{S}$ whose reconstruction function is computationally easy (this is the case with the schemes constructed in this paper). Second, it is possible to facilitate the realization of disclosures under "complicated" conditions by requiring the user to send a *witness* to the validity of its queries, which will serve as an additional input to the condition. In this setting, the general upper bounds given in Theorem 2 can be extended to apply to *nondeterministic* formulas or span programs, yielding efficient conditional disclosure protocols whenever the condition can be computed by an efficient circuit. Indeed, letting the witness supplied by the user consist of all intermediate gate values, it is possible to verify that the circuit evaluates to 1 using a Boolean

[10] m will be disclosed to the database as an upper bound on the number of data records that the user wishes to retrieve. If the user wants to retrieve less than m records, the rest of the indices will point to a dummy record of cost 0.

formula whose size is linear in the circuit size. Since addition of m ℓ-bit integers can be computed by a circuit of size $O(\ell m)$, the amount of communication required for disclosing each answer bit[11] is $O(\ell m)$ plus m times the size of circuitry required for reconstructing the selected costs from the c-answers.

Acknowledgments

We would like to thank an anonymous referee for very helpful comments and suggestions. We are also grateful to Shafi Goldwasser for many enlightening discussions and to Madhu Sudan for comments on an earlier version of this paper.

References

[1] A. Ambainis. Upper bound on the communication complexity of private information retrieval. In *Proc. of 24th ICALP, LNCS 1256, Springer Verlag*, pages 401–407, 1997.

[2] D. Beaver. Perfect privacy for two party protocols. Technical Report TR-11-89, Harvard University, 1989.

[3] D. Beaver and J. Feigenbaum. Hiding instances in multioracle queries. In *Proc. of 7th STACS, LNCS 415, Springer Verlag*, pages 37–48, 1990.

[4] A. Beimel, Y. Ishai, E. Kushilevitz, and T. Malkin. One-way functions are essential for single database private information retrieval. In *Proc. of 31st STOC*, pages 89–98, 1999.

[5] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In *Proc. of CRYPTO '88*, pages 27–35, 1990.

[6] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications. In *Proc. of 20th STOC*, pages 103–112, 1988.

[7] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Jour. on Computing*, 13:850–864, 1984. Early version appears in *Proc. of 23rd FOCS*, 1982.

[8] G. Brassard, C. Crépeau, and J.M. Robert. Information theoretic reductions among disclosure problems. In *Proc. of 18th STOC*, pages 168–173, 1986.

[9] G. Brassard, C. Crépeau, and M. Santha. Oblivious transfers and intersecting codes. *IEEE Transaction on Information Theory, special issue on coding and complexity*, 42(6):1769–1780, 1996.

[10] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *Proc. of EUROCRYPT '99, LNCS 1592, Springer Verlag*, 1999.

[11] In each of the schemes constructed in Section 5, there exists a *single* answer bit which, when eliminated from the user's view, makes the user learn no information about the data.

[11] B. Chor and N. Gilboa. Computationally private information retrieval. In *Proc. of 29th STOC*, pages 304–313, 1997.

[12] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proc. of 36th FOCS*, pages 41–50, 1995.

[13] B. Chor and E. Kushilevitz. A zero-one law for Boolean privacy. *SIAM Jour. on Disc. Math.*, 4(1):36–47, February 1991. Early version appears in *Proc. of 21th STOC*, 1989, pp. 62-72.

[14] G. Di Crescenzo, T. Malkin, and R. Ostrovsky. Single-database private information retrieval implies oblivious transfer. Manuscript, November 1998.

[15] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Comm. of ACM*, 28:637–647, 1985.

[16] U. Feige, J. Kilian, and M. Naor. A minimal model for secure computation (extended abstract). In *Proc. of 26th STOC*, pages 554–563, 1994.

[17] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *JACM*, 33:792–807, 1986.

[18] Y. Ishai and E. Kushilevitz. Private simultaneous messages protocols with applications. In *Proc. of 5th ISTCS*, pages 174–183, 1997.

[19] Y. Ishai and E. Kushilevitz. Improved upper bounds on information-theoretic private information retrieval. In *Proc. of 31st STOC*, pages 79–88, 1999.

[20] M. Ito, A. Saito, and T. Nishizeki. Secret sharing schemes realizing general access structures. In *Proc. IEEE Global Telecommunication Conf., Globecom 87*, pages 99–102, 1987.

[21] M. Karchmer and A. Wigderson. On span programs. In *Proc. of 8th IEEE Structure in Complexity Theory*, pages 102–111, 1993.

[22] E. Kushilevitz. Privacy and communication complexity. *SIAM Jour. on Disc. Math.*, 5(2):273–284, May 1992. Early version in *Proc. of 30th FOCS*, 1989, pp. 416–421.

[23] E. Kushilevitz and R. Ostrovsky. Single-database computationally private information retrieval. In *Proc. of 38th FOCS*, pages 364–373, 1997.

[24] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proc. of 31st STOC*, pages 245–254, 1999.

[25] M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard University, 1981.

[26] A. Shamir. How to share a secret. *Commun. ACM*, 22(6):612–613, June 1979.

[27] I. Wegener. *The complexity of Boolean functions*. Teubner, 1987.

[28] A. C. Yao. Theory and applications of trapdoor functions. In *Proc. of 23rd FOCS*, pages 80–91, 1982.

A Necessity of Shared Randomness

A.1 Shared Randomness is Necessary for SPIR

In this section we show that the addition of a shared randomness resource to the basic PIR setting is in a sense minimal.

Suppose we allow the databases to use *private* randomness in answering the user's queries, but we still do not allow them to interact without the mediation of the user (and in particular we do not allow them to share a random string unknown to the user). We argue that in this setting, (information-theoretic) SPIR cannot be implemented at all, regardless of its complexity, even when the user is honest.

Claim 2. There exists no (multi-round) k-database SPIR scheme without direct interaction between different databases, even if the databases are allowed to hold *private* and *independent* random inputs, and the user is honest.

Proof. Since the user's view includes *all* of the communication, the strong privacy requirement implies that any single database $\mathcal{DB}_j$ cannot respond to the user's queries in a way that depends on the data string x. Formally, at any round the distribution of $\mathcal{DB}_j$'s answer given the previous communication cannot depend on x. For otherwise, this answer distribution must either not follow from a single bit x_i, thus violating the data-privacy requirement, or alternatively reveal to $\mathcal{DB}_j$ the index i on which it depends, thus violating the user's privacy. The independence of private random inputs held by different databases implies that *given previous communication* the answers of different databases must be independently distributed. Combining the observations made above we have that the *joint* distribution of all k answers given previous communication is independent of x. Fixing an index i, it follows by induction on the number of rounds that for any $w > 0$ the accumulated communication in the first w rounds is distributed independently of x. This implies that the user's output cannot depend on the value of x_i, contradicting the correctness requirement. □

As a special case of Claim 2 we may conclude the following:

Corollary 2. There exists no single-database (information-theoretic) SPIR scheme.

We note that Corollary 2 can also be derived from known results about two-party computation [13, 22, 2].

A.2 Shared Randomness in General Reduction from SPIR to PIR

We have shown above that the resource of shared randomness is necessary in order for SPIR to be achievable. In Section 3 we have presented general transformations from PIR to SPIR using linear shared randomness, and in Sections 4 and 5 specific transformations using about the same shared randomness as the communication complexity.

A natural question concerning the general transformations is whether their shared randomness complexity can be reduced, possibly as a function of their communication complexity. We now argue that if we want the general reduction to apply to *any* PIR scheme, then its shared randomness complexity (in the information-theoretic honest user case) is in a sense minimal; that is, the uniform distribution on $\{0,1\}^n$ from which the shared random string is chosen cannot be replaced by a distribution on $\{0,1\}^n$ whose entropy is less than n. It is straightforward to observe that this is the case with the trivial 1-database PIR scheme in which the database sends the entire data string to the user; the following claim indicates that this is also the case for PIR schemes with arbitrarily small communication complexity.

Claim 3. Any PIR scheme of which one answer bit gives the Boolean "OR" of all data bits requires the shared random string r in the scheme of Theorem 1 to be *uniformly* distributed over $\{0,1\}^n$.

Proof. Let R denote the distribution on $\{0,1\}^n$ from which r is picked, and suppose that R is not uniform; for $n \geq 2$, it easily follows that there exist $y, y' \in \{0,1\}^n$ and an index $i \in [n]$ such that $y_i = y'_i$, and $Pr[R = y] \neq Pr[R = y']$. Let $\mathcal{S}_{\mathcal{P}}^R$ denote the scheme $\mathcal{S}_{\mathcal{P}}$ constructed in the proof of Theorem 1 with the shared random string r distributed according to R, and consider an invocation of $\mathcal{S}_{\mathcal{P}}^R$ in which the user's retrieval index is i and the specified shift is $\Delta = 0$. Now, observe that in this invocation the user can distinguish between the data strings y and y', as

$$\begin{aligned} Pr[\bigvee_{j\in[n]} (y \oplus R)_j = 0] &= Pr[R = y] \\ &\neq Pr[R = y'] = Pr[\bigvee_{j\in[n]} (y' \oplus R) = 0]. \end{aligned}$$

By the correctness of $\mathcal{S}_{\mathcal{P}}^R$, the user must also learn the i-th data bit, implying that it obtains more than a single physical bit of data. $\square$