Computationally Private Information Retrieval

(extended abstract)

Benny Chor[*] Niv Gilboa[†]

Abstract

Private information retrieval (PIR) schemes enable a user to access k replicated copies of a database ($k \geq 2$), and *privately* retrieve one of the n bits of data stored in the databases. This means that the queries give each individual database no partial information (in the information theoretic sense) on the identity of the item retrieved by the user. Today, the best two database scheme ($k = 2$) has communication complexity $O(n^{1/3})$, while for any constant number, k, the best k database scheme has communication complexity $O(n^{1/(2k-1)})$. The motivation for the present work is the question whether this complexity can be reduced if one is willing to achieve *computational privacy*, rather than *information theoretic privacy*. (This means that privacy is guaranteed only with respect to databases that are restricted to polynomial time computations.)

We answer this question affirmatively, and show that the computational approach leads to substantial savings. For every $\varepsilon > 0$, we present a *two database* computational PIR scheme whose communication complexity is $O(n^{\varepsilon})$. This improved efficiency is achieved by a combination of a novel balancing technique, together with careful application of pseudo random generators. Our schemes preserve some desired properties of previous solutions. In particular, all our schemes use only one round of communication, they are fairly simple, they are memoryless, and the database contents is stored in its plain form, without any encoding.

1 Introduction

Publicly accessible databases are an indispensable resource for retrieving up-to-date information. However, accessing such databases also poses a significant risk to the privacy of the user, since a curious database operator can follow the user's queries and infer what the user is after. Recently, it has been shown that if the data is replicated at k ($k \geq 2$) databases, and these databases do not communicate, then it is possible to protect the user's privacy. The solutions to this private information retrieval (PIR) problem enable the user to retrieve a desired data item, while giving each individual database no partial information on the query. The quality of a solution is measured primarily

[*]Computer Science Dept., Technion, Haifa, Israel. Email: benny@cs.technion.ac.il. Supported by Technion V.P.R Fund - E. and M. Mendelson Research Fund.

[†]Computer Science Dept., Technion, Haifa, Israel. Email: gilboa@cs.technion.ac.il

by its communication overhead.

To make the private information retrieval problem more concrete, we view the database as a string x of length n. Identical copies of this string are stored in $k \geq 2$ sites. The user has some index i, and is interested in obtaining the value of the bit x_i. In the first work that introduced this model, Chor, Goldreich, Kushilevitz and Sudan [3] present[1] various schemes that solve the retrieval problem with significantly lower communication complexity than the obvious n-bit solution (i.e., asking for a copy of x). In particular they obtain the following:

- A two-database scheme with communication complexity $O(n^{1/3})$.
- A scheme for a constant number, k, of databases with communication complexity $O(n^{1/k})$.

This last result was subsequently improved in an elegant work by Ambainis [1]. The communication complexity of his k database protocol is $O(n^{1/(2k-1)})$ bits.

The PIR schemes mentioned above achieve information theoretic privacy: The communication between the user and each database is identically distributed, regardless of which data item the user seeks. It is natural to ask whether cryptographic techniques can be used in the context of PIR. This implies that information theoretic privacy will be replaced by computational privacy (privacy with respect to polynomial time computations), and will require the use of an appropriate intractability assumption. However, if one is willing to make these "compromises", there is much to be gained.

We make the mildest acceptable cryptographic assumption – the existence of pseudo random generators [2, 10]. (This is equivalent to the existence of one way functions [6, 5].) Under this assumption, we develop a family of computational PIR schemes. All these schemes are two database schemes with one round of communication (one query and one response per database).. For every $\varepsilon > 0$, we have a scheme in this family with (worst case) communication complexity $O(n^{\varepsilon})$. All the schemes are fairly simple, and do not require any coding in storing the database contents. They are memoryless – neither users nor databases have to remember any of the communication's history.[2]

The first step in our construction is the design of a new balancing scheme. Balancing was introduced in [3] and proved to be a powerful technique in the PIR context. Starting with a given one round scheme, balancing modifies it in a way that changes the ratio of communication from the user to the communication from the database. Our balancing scheme by itself increases, rather than decreases, the total communication complexity. But its structure enables us to incorporate pseudo random generators in order to cut down the communication from the user: we replace random strings of the original scheme by shorter random seeds, that are subsequently expanded at the database end using pseudo random generators. These expanded strings are used by the database as queries of the original scheme. We rigorously prove that this construction maintains *computational privacy.*

Starting with a scheme with $O(n^{1/\ell})$ communication complexity, the new scheme has essentially $O(n^{1/(\ell+1)})$ communication complexity. This enables us to apply this process recur-

[1]Details on related models and techniques can also be found in [3].

[2]Different computational PIR schemes, based on techniques from oblivious RAM [4, 8] were found by Itkis [7] and by Ostrovsky and Shoup [9].

sively, till a desired threshold of $O(n^\varepsilon)$ communication complexity is achieved. The unraveling of this recursive procedure is done at the database end, and requires no additional communication rounds.

The remainder of this paper is organized as follows: Section 2 contains definitions and notations. Section 3 describes the new balancing scheme. Section 4 presents our computational PIR schemes and contains their analysis and correctness proof.

2 Definitions and Notations

Definition 1: Let $\mathcal{P}$ be a one round scheme for information retrieval, in which the parties are a user, $\mathcal{U}$, and two databases, $\mathcal{DB}_1, \mathcal{DB}_2$. Each database holds a copy of an n bit binary string x to which we refer as the content of the database. The user wishes to retrieve a bit x_i $(1 \leq i \leq n)$. $\mathcal{U}$ sends one query $Q_1(i)$ to $\mathcal{DB}_1$, and one query $Q_2(i)$ to $\mathcal{DB}_2$ (these queries depend on i and on a source of randomness r that is local to $\mathcal{U}$). Each database responds with one answer (which is a function of the query and of x). These two answers enable $\mathcal{U}$ to recover x_i. The scheme is called *information theoretically private* if for every $1 \leq i_1 \leq i_2 \leq n$, $Q_1(i_1)$ and $Q_1(i_2)$ are identically distributed, and also $Q_2(i_1)$ and $Q_2(i_2)$ are identically distributed. The scheme is called *computationally private* if for every $1 \leq i_1 \leq i_2 \leq n$, $Q_1(i_1)$ and $Q_1(i_2)$ are indistinguishable in $poly(n)$ time, and also $Q_2(i_1)$ and $Q_2(i_2)$ are indistinguishable in $poly(n)$ time.

Definition 2: A one round, two database information retrieval scheme $\mathcal{P}$ is *symmetric* if the following conditions hold:

1. For every $1 \leq i \leq n$, the queries $Q_1(i)$ and $Q_2(i)$ are identically distributed.
2. Let $A_1(q,x)$ and $A_2(q,x)$ denote the responses of $\mathcal{DB}_1$ and $\mathcal{DB}_2$ to the query q. Then for every q, $A_1(q,x) = A_2(q,x)$.

Definition 3 : An information retrieval scheme $\mathcal{P}$ is *polynomial* if the user and the databases execute their respective parts in $\mathcal{P}$ in time that is polynomial in the length of x.

Definition 4: We say that a private information retrieval scheme $\mathcal{P}$ is a *computationally private, one round, polynomial and symmetric scheme* (abbreviated as *COPS* scheme) if the following conditions hold: P is a one round, polynomial and symmetric scheme for two databases. For every desired data item i, the queries $Q_1(i,x)$ and $Q_2(i,x)$ sent by the user to each database in $\mathcal{P}$ are taken from a pseudo random distribution (individually, not jointly). Finally, we require that each database replies to every query of the right length.

Notice that being a COPS scheme implies that P is computationally private, but not necessarily information theoretically private.

In [3] a scheme for solving $PIR(n)$ with communication complexity $O(n^{1/3})$ was presented. This scheme is the most efficient scheme known to date (in terms of communication complexity) for two databases. We will refer to it as $\mathcal{B}_2$ from here on. We end this section with the following notations: For every $n \in \mathcal{N}$ we denote the set $\{1, \ldots, n\}$ by $[n]$. Given a set S and an element a we denote:

$$S \bigoplus a = \begin{cases} S \cup \{a\} & \text{if } a \notin S \\ S \setminus \{a\} & \text{if } a \in S \end{cases}$$

3 A New Balancing Scheme

The importance of balancing schemes in the private information retrieval context was established in [3]. A balancing scheme receives as input a scheme $\mathcal{O}$ that solves $PIR(n)$, in which $\mathcal{U}$ sends $\alpha(n)$ bits to each database, and $\mathcal{DB}_1, \mathcal{DB}_2$ reply with $\beta(n)$ bits each. The authors showed two balancing schemes of two different types:

1. A *database dominated scheme-* in this type of scheme there is an increase in the number of bits sent by each database to the user, and a decrease in the number of bits sent by the user.

2. A *user dominated scheme-* in this type of scheme there is an increase in the number of bits sent by the user to each database, and a decrease in the number of bits sent by the databases.

3.1 A Generic User Dominated Balancing Scheme

In this subsection we present a novel user dominated balancing scheme for two databases which can take as input *any* one round $PIR(n)$ scheme, $\mathcal{O}$.

Our scheme has essentially the same communication complexity of the user-dominated scheme that [3] provided for two databases. The difference between the two ideas is in the larger class of PIR schemes that can be balanced using the protocol we present below. In particular, the scheme from [3] can not be applied to $\mathcal{B}_2$ or any other scheme in which the user's computation on the messages it received from the databases is dependent on the bit being retrieved.

Theorem 1: Let $\mathcal{O}$ be a scheme that solves $PIR(n)$, in which the user sends each database $\alpha(n)$ bits and receives in return $\beta(n)$ bits from each one. For every $m \in [n]$ there exists a scheme $\mathcal{N}$ that solves $PIR(n)$, in which the user sends each database $m(\alpha(n/m)+1)$ bits and each database replies with $2\beta(n/m)$ bits.

Proof: We may assume that the parameter m is a divisor of n and that the scheme $\mathcal{O}$, is symmetric. Both requirements can be dispensed with at low communication cost (at most, a multiplicative constant of 2), and therefore we assume w.l.o.g that they are satisfied.

The string x is viewed as a matrix of m rows by $\lceil n/m \rceil$ columns. Let the bit that $\mathcal{U}$ wishes to retrieve be the i-th bit in the j-th row.

The scheme $\mathcal{N}$

1. $\mathcal{U}$ chooses uniformly a subset $S \subseteq [m]$. $\mathcal{U}$ sends S to $\mathcal{DB}_1$ and $S \bigoplus j$ to $\mathcal{DB}_2$.
2. $\mathcal{U}$ chooses $m+1$ messages $v_1, \ldots, v_{j-1}, v_j^1, v_j^2, v_{j+1}, \ldots, v_m$ in the following manner: for each of the m rows $\mathcal{U}$ chooses an address of a bit in the range $1, \ldots, \lceil n/m \rceil$. The address chosen for the j-th row is i, and the address for any other row is 1. Assume that the address chosen for the q-th row is i'. The user executes the scheme $\mathcal{O}$ in order to retrieve the i'-th bit out of $\lceil n/m \rceil$ bits. The scheme is executed only up to the point at which $\mathcal{U}$ decides on the two messages- $w_1(i'), w_2(i')$ that are to be sent to the databases. If $q = j$ define $v_j^1 \triangleq w_1(i)$ and $v_j^2 \triangleq w_2(i)$. Otherwise choose $b \in \{1,2\}$ at random and define $v_q \triangleq w_b(i')$. $\mathcal{U}$ intends to send the databases one message for each of the m rows of the database contents. $\mathcal{U}$ sends the messages $v_1, \ldots, v_{j-1}, v_j^1, v_{j+1}, \ldots, v_m$ to $\mathcal{DB}_1$, and the messages $v_1, \ldots, v_{j-1}, v_j^2, v_{j+1}, \ldots, v_m$ to $\mathcal{DB}_2$.
3. Each database executes the scheme $\mathcal{O}$ in parallel m times. In each execution the content of the database is considered to be of length $\lceil n/m \rceil$. Let $v_1, \ldots, v_m$ denote the m messages a database received (v_j equals v_j^1 for $\mathcal{DB}_1$, and equals v_j^2 for $\mathcal{DB}_2$). In the q-

th execution of $\mathcal{O}$ each database considers v_q as the query, and the q-th row of the matrix as the database contents. The database $\mathcal{DB}_b$ ($b \in \{1,2\}$), produces an output a_q^b which is the answer it would have returned to $\mathcal{U}$ as part of the scheme $\mathcal{O}$. Since $\mathcal{O}$ is symmetric, for any $q \neq j$, $a_q^1 = a_q^2$. Each output is of size $\beta(n/m)$ by assumption. Each database now computes two sums(bitwise XOR): $\mathcal{DB}_1$ computes $A_1 \triangleq \sum_{q \in S} a_q^1$, and $B_1 \triangleq \sum_{q \in \overline{S}} a_q^1$. $\mathcal{DB}_2$ computes $A_2 \triangleq \sum_{q \in S \bigoplus j} a_q^2$, and $B_2 \triangleq \sum_{q \in \overline{S \bigoplus j}} a_q^2$. The 4 elements A_1, B_1, A_2, B_2 are all sent to $\mathcal{U}$.

4. $\mathcal{U}$ obtains a_j^1 and a_j^2 using the four elements it received. If $j \in S$ then $A_1 - A_2 = a_j^1$ and $B_2 - B_1 = a_j^2$. Otherwise, $j \notin S$, we have $A_2 - A_1 = a_j^2$, and $B_1 - B_2 = a_j^1$. In the PIR scheme $\mathcal{O}$ the messages a_j^1 and a_j^2 are sent by $\mathcal{DB}_1$ and $\mathcal{DB}_2$ respectively to $\mathcal{U}$ when they receive the inputs v_j^1 and v_j^2. Our choice of v_j^1 and v_j^2 was such that $\mathcal{U}$ can now obtain the value of the i-th bit in the j-th row, using the same computation on a_j^1, a_j^2 as in the scheme $\mathcal{O}$.

Privacy: It is enough to show that $\mathcal{DB}_1$ can gain no information about the bit being retrieved, since the same arguments will prove the privacy in regard to $\mathcal{DB}_2$. $\mathcal{DB}_1$ receives as input $S, v_1, \ldots, v_{j-1}, v_j^1, v_{j+1}, \ldots, v_m$. S is distributed uniformly regardless of the bit being retrieved. Each of the v_q's is drawn out of the same distribution whichever bit is being retrieved, since $\mathcal{O}$ is information theoretically private. Therefore the distribution of the messages that $\mathcal{DB}_1$ receives is the same for every bit $\mathcal{U}$ wishes to retrieve, and consequently the scheme $\mathcal{N}$ maintains information-theoretic privacy.

Communication complexity: The user sends each database one subset of $[m]$ (m bits), and m messages of size $\alpha(n/m)$. Thus exactly $m(\alpha(n/m)+1)$ bits are sent to each database. In the other direction, $\mathcal{DB}_1$ and $\mathcal{DB}_2$ each send $2\beta(n/m)$ bits.

4 A computationally private scheme

Proposition 2: Suppose that $\mathcal{PIR}(n)$ can be solved by a COPS scheme, $\mathcal{O}$, in which the user sends $\alpha(n)$ bits to each database and receives $\beta(n)$ bits in return. Furthermore, suppose that there is a pseudo random generator G which expands random seeds of length $s(n)$ to strings of length n, such that the expanded strings are pseudo random with respect to $poly(n)$-distinguishers. Then for every m, $m \leq n^{3/4}$, $\mathcal{PIR}(n)$ can be solved by a COPS scheme, $\mathcal{N}$, in which the user sends $2\alpha(n/m)+m\cdot(1+s(n))$ bits to each database, and receives $2\beta(n/m)$ bits in return.

Proof: Starting with the original scheme, $\mathcal{O}$, the high level idea is to apply the balancing scheme of Theorem 1. However, the user in that scheme sends $m \cdot (1 + \alpha(n/m))$ bits to each database, instead of the desired $2\alpha(n/m) + m \cdot (1 + s(n))$ bits. To overcome this gap, the user will send shorter messages that will be expanded by each database and then "interpreted" as queries for the original scheme (after balancing). The data $x \in \{0,1\}^n$ is viewed as an $m \times \lceil n/m \rceil$ binary matrix. Suppose the desired item is the i-th item of the j-th row, then $\mathcal{DB}_1$ should, after interpretation, have queries $v_1, \ldots, v_{j-1}, v_j^1, v_{j+1}, \ldots, v_m$. After interpretation, $\mathcal{DB}_2$ should have queries $v_1, \ldots, v_{j-1}, v_j^2, v_{j+1}, \ldots, v_m$. The crucial observation is that the v_k can in fact be taken from a much smaller query space than the space of all strings of length $\alpha(n/m)$, as long as two conditions hold: For $k \neq j$, the k-th interpreted messages for both databases are the same. while for $k = j$, the queries v_j^1 and v_j^2 are queries in the original scheme $\mathcal{O}$ (for the i-th item in a database with $\lceil n/m \rceil$ bits). Of course, these conditions alone do not guarantee privacy, which has to be argued separately.

To reduce $m \cdot \alpha(n/m)$ communication down to $m \cdot s(n)$, the user generates $m + 1$ random independent *seeds* $s_1,\ldots,s_{j-1},s_j^1,s_j^2,s_{j+1},\ldots,s_m$ of length $s(n)$ each. The seeds $s_1,\ldots,s_{j-1},s_j^1$, $s_{j+1},\ldots,s_m$ are sent to $\mathcal{DB}_1$, and $s_1,\ldots,s_{j-1}$, $s_j^2,s_{j+1},\ldots,s_m$ are sent to $\mathcal{DB}_2$. Each database applies the generator G to the m seeds it received , and expands each of them to a string of the "right" length, $\alpha(n/m)$ (the seeds could in fact be expanded to n bit strings, but this will not be useful in our context). For every $k \neq j$, $G(s_k)$ can potentially be used as the k-th query by both databases. However, for $k = j$ the pair $G(s_j^1)$ and $G(s_j^1)$ is typically *not* a proper pair of queries for retrieving the i-th item, and can not be used as the j-th pair of queries. To overcome this last difficulty, suppose v_j^1 and v_j^2 form a proper pair (chosen according to the pseudo random distribution induced by the original scheme, where both strings are of length $\alpha(n/m)$). In addition to the m seeds, each database will also receive two strings of length $\alpha(n/m)$: One is $v_j^1 + G(s_j^1)$, the other is $v_j^2 + G(s_j^2)$ (addition is over Z_2, namely bitwise XOR, or over any other fixed finite field). Each database in effect receives v_j^1 and v_j^2, "masked" by $G(s_j^1)$ and $G(s_j^2)$ respectively. $\mathcal{DB}_1$, holding s_j^1, can remove the mask and recover v_j^1. Similarly, $\mathcal{DB}_2$, holding s_j^2, can remove the mask and recover v_j^2. Not having the other seed, the database cannot unmask the other v_j. We now describe in detail the mechanism which guarantees that this unmasking procedure creates m queries of the desired form ($v_1,\ldots,v_{j-1},v_j^1,v_{j+1},\ldots,v_m$ for $\mathcal{DB}_1$ and $v_1,\ldots,v_{j-1},v_j^2,v_{j+1},\ldots,v_m$ for $\mathcal{DB}_2$).

The Scheme $\mathcal{N}$

1. $\mathcal{U}$ chooses uniformly a subset $S \subseteq [m]$. The user also chooses uniformly and independently $m + 1$ random independent seeds $s_1,\ldots,s_{j-1},s_j^1,s_j^2,s_{j+1},\ldots,s_m$ of length $s(n)$ each. $\mathcal{U}$ applies the pseudo random generator G and expands both s_j^1 and s_j^2 to strings of length $\alpha(n/m)$, denoted by $G(s_j^1)$ and $G(s_j^2)$. The user now begins to execute the original scheme $\mathcal{O}$ for the retrieval of the i-th bit out of $\lceil n/m \rceil$ bits. The execution is carried on up to the point at which $\mathcal{U}$ decides on the two queries v_j^1, v_j^2 it wishes to send to $\mathcal{DB}_1$ and $\mathcal{DB}_2$ respectively. Both queries are of length $\alpha(n/m)$. Finally, the user computes two "masked messages"

$$M_1 = \begin{cases} v_j^1 + G(s_j^1) & \text{if } j \in S \\ v_j^2 + G(s_j^2) & \text{if } j \notin S \end{cases}$$

$$M_2 = \begin{cases} v_j^2 + G(s_j^2) & \text{if } j \in S \\ v_j^1 + G(s_j^1) & \text{if } j \notin S \end{cases}$$

2. The user $\mathcal{U}$ sends the following:

- The set S to $\mathcal{DB}_1$.
- The set $S \bigoplus j$ to $\mathcal{DB}_2$.
- The masked messages M_1, M_2 to *both* databases.
- The seeds $s_1,\ldots,s_{j-1},s_j^1,s_{j+1},\ldots,s_m$ to $\mathcal{DB}_1$.
- The seeds $s_1,\ldots,s_{j-1},s_j^2,s_{j+1},\ldots,s_m$ to $\mathcal{DB}_2$.

(Every database receives $m(1 + s(n)) + 2\alpha(n/m)$ bits.)

3. Each database expands the seeds $s_1, \ldots, s_m$ it received to $\alpha(n/m)$ long strings, by applying the pseudo-random generator G, and obtains $G(s_1), \ldots, G(s_m)$ (s_j equals s_j^1 for $\mathcal{DB}_1$, and equals s_j^2 for $\mathcal{DB}_2$). Each database now executes the original scheme in parallel m times. In the k-th execution ($k = 1, \ldots, m$) the $\lceil n/m \rceil$ bits in the k-th row of x are regarded as the database contents by both databases. Denote by T the set received by the database (T equals S for $\mathcal{DB}_1$ and $S \bigoplus j$ for $\mathcal{DB}_2$). Each database sets his "interpreted query" v_k as follows:

$$v_k = \begin{cases} M_1 - G(s_k) & \text{if } k \in T \\ M_2 - G(s_k) & \text{if } k \notin T \end{cases}$$

Consider some row k that is different from j. Since $k \neq j$, either $k \in T$ for both databases, or $k \notin T$ for both databases. We get

$$v_k = \begin{cases} v_j^1 + G(s_j^1) - G(s_k) & k \in S, j \in S \\ v_j^2 + G(s_j^2) - G(s_k) & k \notin S, j \in S \\ v_j^2 + G(s_j^2) - G(s_k) & k \in S, j \notin S \\ v_j^1 + G(s_j^1) - G(s_k) & k \notin S, j \notin S \end{cases}$$

We see that both databases produce the same "interpreted query" v_k for this row k ($k \neq j$).

For the j-th row ($k = j$), the two databases produce different "interpreted queries". The query v_j produced by $\mathcal{DB}_1$ equals

$$v_j = \begin{cases} M_1 - G(s_j^1) & \text{if } j \in S \\ M_2 - G(s_j^1) & \text{if } j \notin S \end{cases}$$

Which in both cases equals v_j^1 ,the query in the original scheme. A similar argument shows that the j-th interpreted query of $\mathcal{DB}_2$ equals v_j^2.

To summarize, the m interpreted queries of $\mathcal{DB}_1$ equal

$$v_1, v_2, \ldots, v_{j-1}, v_j^1, v_{j+1}, \ldots, v_m \ .$$

The m interpreted queries of $\mathcal{DB}_2$ equal

$$v_1, v_2, \ldots, v_{j-1}, v_j^2, v_{j+1}, \ldots, v_m \ .$$

We are therefore in the required situation for applying the balanced scheme.

Since $\mathcal{O}$ is a COPS scheme, and all v_k's are of length $\alpha(n/m)$, it is guaranteed that each database can produce an answer to each query. For $k \neq j$, the k-th query to $\mathcal{DB}_1$ and to $\mathcal{DB}_2$ are the same. As $\mathcal{O}$ is a symmetric scheme, the answers to these queries must also be the same. Denote these answers by $a_1, a_2, \ldots, a_{j-1}, a_{j+1}, \ldots, a_m$. For $k = j$, denote the answer to query v_j^1 by a_j^1, and the answer to query v_j^2 by a_j^2. By the balancing scheme, each database computes two sums: One is the sum of all answers with indices in the set T, the other with indices in the complement of T. The two answers sent by $\mathcal{DB}_1$ are $A_1 \stackrel{\triangle}{=} \sum_{k \in S} a_k$, and $B_1 \stackrel{\triangle}{=} \sum_{k \in \overline{S}} a_k$, while $\mathcal{DB}_2$ sends $A_2 \stackrel{\triangle}{=} \sum_{k \in S \bigoplus j} a_k$, and $B_2 \stackrel{\triangle}{=} \sum_{k \in \overline{S \bigoplus j}} a_k$. (Each database sends $2\beta(n/m)$ bits.) Depending on whether $j \in S$ or $j \in \overline{S}$, we have

$$A_1 = \begin{cases} a_j^1 + \sum_{k \in S \setminus \{j\}} a_k & \text{if } j \in S \\ \sum_{k \in S} a_k & \text{if } j \notin S \end{cases}$$

$$B_1 = \begin{cases} \sum_{k \in \overline{S}} a_k & \text{if } j \in S \\ a_j^1 + \sum_{k \in \overline{S} \setminus \{j\}} a_k & \text{if } j \notin S \end{cases}$$

$$A_2 = \begin{cases} \sum_{k \in S \setminus \{j\}} a_k & \text{if } j \in S \\ a_j^2 + \sum_{k \in S} a_k & \text{if } j \notin S \end{cases}$$

$$B_2 = \begin{cases} a_j^2 + \sum_{k \in \overline{S}} a_k & \text{if } j \in S \\ \sum_{k \in \overline{S} \setminus \{j\}} a_k & \text{if } j \notin S \end{cases}$$

4. $\mathcal{U}$ can now compute the desired replies from the scheme $\mathcal{O}$, a_j^1 and a_j^2: If $j \in S$ then $a_j^1 = A_1 - A_2$ and $a_j^2 = B_2 - B_1$. If $j \notin S$, then $a_j^1 = B_1 - B_2$ while $a_j^2 = A_2 - A_1$. From these two replies, $\mathcal{U}$ can recover the desired data item.

This proves the correctness of the scheme $\mathcal{N}$. Its claimed communication complexity was verified in the construction. It remains to show that $\mathcal{N}$ is a COPS scheme, which implies in particular that it is computationally private. By construction, $\mathcal{N}$ is a one round, two database scheme. The computational requirements (from the user and from the databases) are polynomial in n, plus the requirements of $\mathcal{O}$. Since $\mathcal{O}$ is a polynomial scheme, so is $\mathcal{N}$. By construction, and since $\mathcal{O}$ is a symmetric scheme, the same applies to $\mathcal{N}$. What remains to show is that for any desired data item, the queries sent by $\mathcal{U}$ to each database are pseudo random.

Suppose the desired item is x_q, where $1 \leq q \leq n$. We denote the two queries to the databases, when the desired item is x_q by $u(q)$ and $w(q)$ respectively. We will now prove that $u(q)$ is pseudo random (this implies the pseudo randomness of $w(q)$ since the scheme is symmetric and both queries are identically distributed).

In the scheme $\mathcal{N}$ the n bits of x are organized in a matrix. The q-th bit $(1 \leq q \leq n)$ in x is the (j,i)-th bit in the matrix $(1 \leq j \leq m, 1 \leq i \leq \lceil n/m \rceil)$. The query $u(q)$ is the concatenation of $S, M_1, M_2, s_1, \ldots, s_{j-1}, s_j^1, s_{j+1}, \ldots, s_m$. In order to be specific with respect to M_1 and M_2, we will assume that $j \in S$ (the argument for $j \notin S$ is identical). In this case, $M_1 = v_j^1 + G(s_j^1)$ and $M_2 = v_j^2 + G(s_j^2)$. v_j^1 is the $\mathcal{DB}_1$ query for the i-th bit (out of $\lceil n/m \rceil$ bits) in the scheme $\mathcal{O}$, and v_j^2 is the $\mathcal{DB}_2$ query for the i-th bit in the scheme $\mathcal{O}$.

Suppose, towards a contradiction, that $u(q)$ is *not* pseudo random. Define the ensemble $h(q)$ to be identical to $u(q)$, except replacing $M_2 = v_j^2 + G(s_j^2)$ by $v_j^2 + r_2$, where r_2 is a random independent string of length $\alpha(n/m)$. Since s_j^2 is random and independent of all other entries in $u(q)$, and G is a pseudo random generator, the two ensembles $u(q)$ and $h(q)$ are indistinguishable in $poly(n)$ time. Therefore $h(q)$ is not pseudo random. Consider now the ensemble $g(q)$ which is identical to $h(q)$, except replacing $v_j^2 + r_2$ by r_2. XORing a string of an arbitrary source with a random independent string results in a random independent string. Therefore $h(q)$ and $g(q)$ are distributed identically, and so the latter is not pseudo random either. It is not possible to replace $G(s_j^1)$ (in $g(q)$) by another random string r_1 and apply the previous argument with respect to $g(q)$, because s_j^1, the seed to the generator G, is an explicit part of $g(q)$. Instead, we use the fact that $g(q)$ is not pseudo random in order to distinguish, in polynomial time, between v_j^1 and a random string of the same length. Given n, i, and a string r_1 of length $\alpha(n/m)$, we construct the string d which is the concatenation of $S, r_1 + G(s_j^1), r_2, s_1, \ldots, s_{j-1}, s_j^1, s_{j+1}, \ldots, s_m$, where $S, r_2, s_1, \ldots, s_{j-1}, s_j^1, s_{j+1}, \ldots, s_m$ are independent random strings of the appropriate lengths, and all are independent of r_1. In case r_1 is a random string, d is uniformly distributed. On the other hand, if r_1 is taken from the v_j^1 distribution, d is distributed according to $g(q)$, and is therefore polynomial time distinguishable from a uniformly distributed string. Thus v_j^1 is not pseudo random – contradicting the fact that $\mathcal{O}$ is a COPS scheme. This completes the proof of Proposition 2. □

In the proof of proposition 2 we showed how to construct a COPS scheme out of a different COPS scheme, and thereby obtain a reduction in communication complexity. In the next theorem we take the same idea one step further, and repeat recursively the construction of proposition 2.

The pseudo-random generator G is central to our discussion. We first deal with the general case, in which G expands strings of length $s(n)$ to strings of length n. We assume that the databases are limited to polynomial time computations, and therefore they are unable to distinguish between the pseudo-random strings and random strings of length n. The user sends seeds which contain $s(n)$ bits, and the databases expand them to strings of length n. Since, typically, not all n bits are needed in the scheme of proposition 2, the databases simply choose the prefix of the expanded string, which has the required length.

The most common assumption about pseudo-random generators is that $s(n) = n^{\delta}$, where $\delta > 0$ is any constant. The pseudo-random distribution is assumed to be indistinguishable from the uniform distribution on n bit strings, if the distinguishing algorithm is polynomial in n. We state an explicit upper bound on the communication complexity of COPS schemes which use this type of generator.

Theorem 3: Let G be a pseudo-random generator, which expands strings of length $s(n)$ to strings of length n. For any $k \geq 0$, there exists a COPS scheme with communication complexity $O(3^k n^{\frac{1}{k+3}} s(n)^{\frac{k}{k+3}})$.

Proof: We inductively construct a family of information retrieval schemes $\mathcal{P}_0, \mathcal{P}_1, \mathcal{P}_2, \ldots$ such that for any k, $0 \leq k$, $\mathcal{P}_k$ is a COPS scheme.

As the scheme $\mathcal{P}_0$ we choose the most efficient known PIR scheme for two databases, $\mathcal{B}_2$, which has communication complexity $O(n^{1/3})$. Recall that $\mathcal{B}_2$ is a one round, information theoretically private scheme. Since queries in $\mathcal{B}_2$, are just random strings of length $O(n^{1/3})$, ,$\mathcal{B}_2$ is a COPS scheme.

For every $k \geq 1$, $\mathcal{P}_k$ is derived from $\mathcal{P}_{k-1}$ by using the construction of proposition 2. $\mathcal{P}_{k-1}$ is viewed as the scheme $\mathcal{O}$ in the proof of the proposition, and $\mathcal{P}_k$ is $\mathcal{N}$. The construction ensures that if $\mathcal{P}_{k-1}$ is a COPS scheme, then so is $\mathcal{P}_k$. Together with the statement that $\mathcal{P}_0$ is COPS, we have that for any $k \geq 0$, $\mathcal{P}_k$ is a COPS scheme.

Communication Complexity: We use the following notations for the scheme $\mathcal{P}_k$:

- m_k denotes the parameter of the scheme (the number of rows in the matrix). The notation in proposition 2 was simply m, because only a single scheme was constructed.
- $\alpha_k(n)$ denotes the user's communication complexity, where n is the length of the database contents.
- $\beta_k(n)$ denotes a database's communication complexity.

We now prove by induction that by a careful choice of the parameters m_k, we can ensure that for any $k \geq 0$: $\alpha_k(n) = O(3^k n^{\frac{1}{k+3}} s(n)^{\frac{k}{k+3}})$, and $\beta_k(n) \leq \alpha_k(n)$. For $k = 0$ this claim is true because in $\mathcal{P}_0$ all the parties- $\mathcal{U}$, $\mathcal{DB}_1$ and $\mathcal{DB}_2$ send $O(n^{1/3})$ bits.

Suppose that the hypothesis of the induction is true for $k-1$. Proposition 2 states that $\alpha_k(n) = (s(n)+1)m_k + 2\alpha_{k-1}(n/m_k)$, and $\beta_k(n) = 2\beta_{k-1}(n/m_k)$. Together with the induction hypothesis $\beta_{k-1}(n) \leq \alpha_{k-1}(n)$, this implies $\beta_k(n) \leq \alpha_k(n)$. In order to prove the statement for $\alpha_k(n)$, an optimal value for m_k would be one which minimizes the expression $(s(n)+1)m_k + 2\alpha_{k-1}(n/m_k) + 2\beta_{k-1}(n/m_k)$. For the sake of brevity we choose a value which is slightly less than optimal, but is still good enough to get the desired results. We set m_k to satisfy the equation: $(s(n)+1)m_k = (n/m_k)^{\frac{1}{k+2}}(s(n)+1)^{\frac{k-1}{k+2}}$. In essence this constraint minimizes an expression similar to the one required, in which several factors are disregarded.

It follows that $m_k = n^{\frac{1}{k+3}}(s(n)+1)^{\frac{-3}{k+3}}$. Assuming that the hypothesis of the induction is true for $k-1$, we get

$$\begin{aligned}\alpha_k(n) &= O\Big((s(n)+1)m_k + \\ &\qquad 2(n/m_k)^{\frac{1}{k+2}}(s(n)+1)^{\frac{k-1}{k+2}}3^{k-1}\Big) .\end{aligned}$$

By substituting for m_k, we have:

$$\begin{aligned}\alpha_k(n) &= O\left(n^{\frac{1}{k+3}}(s(n)+1)^{\frac{k}{k+3}} + \right. \\ &\qquad \left. 2n^{\frac{1}{k+3}}(s(n)+1)^{\frac{k}{k+3}}3^{k-1}\right) .\end{aligned}$$

Hence,

$$\alpha_k(n) = O\left(3^k n^{\frac{1}{k+3}} s(n)^{\frac{k}{k+3}}\right) . \qquad \square$$

Corollary 4: If one way functions exist, then for any constant $\varepsilon > 0$, there exists a COPS scheme for two databases with communication complexity $O(n^\varepsilon)$.

Proof: The existence of one way functions implies the existence of pseudo random generators with $s(n) = n^\delta$ for any constant $\delta > 0$ [2, 5, 6, 10]. We wish to show a scheme $\mathcal{P}_t$ with communication complexity $O(n^\varepsilon)$. Set t to be the first integer larger than $2/\varepsilon - 3$, and set $\delta = 1/t$. ε is a constant, and therefore so are t and δ. Substituting for the different variables we obtain:

$$\begin{aligned}\alpha_t(n) &= O\left(n^{\frac{1}{t+3}} s(n)^{\frac{t}{t+3}}\right) \\ &= O\left(n^{\frac{2}{t+3}}\right) \\ &= O(n^\varepsilon) . \qquad \square\end{aligned}$$

Acknowledgments: We are grateful to Oded Goldreich, who has the good property of reading early drafts, for helpful discussions and suggestions.

References

[1] A. Ambainis, *"An upper bound for Private Information Retrieval"*, manuscript, 1996.

[2] M. Blum and S.. Micali, *"How to Generate Cryptographically Strong Sequences of Pseudo Random Bits"*, SIAM Jour. on Computing, Vol. 13 (1984), pp. 850–864.

[3] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan, *"Private Information Retrieval"* , Proc. of 36th FOCS (1995), pp. 41–50.

[4] O. Goldreich, *"Towards a Theory of Software Protection and Simulation by Oblivious RAMs"*, Proc. of 19th STOC (1987), pp. 182–194.

[5] J.Håstad *"Pseudo-Random generators with Uniform Assumptions"*, Proc. of 22nd STOC (1990), pp. 395–404.

[6] R. Impagliazzo, L.A. Levin, M. Luby, *"PseudoRandom Generation from One-Way Functions"*, Proc. of 29th FOCS (1989), pp. 12–24.

[7] G. Itkis, private communication, 1996.

[8] R. Ostrovsky, *"Software Protection and Simulation on Oblivious RAMs"*, M.I.T. Ph.D thesis in Computer Science, June 1992. Preliminary version in *STOC*, 1990.

[9] R. Ostrovsky, V. Shoup, *"Private Information Storage"*, these proceedings.

[10] A. Yao, *"Theory and Applications of Trapdoor Functions"*, Proc. of 23rd FOCS (1982), pp. 80–91.