Private Information Retrieval

Krishnaram Kenthapadi

November 16, 2003

Introduction

Consider a huge database and a user who wants to query. We want the user to be able to query the database while hiding the identity of the data-items she is after. Our aim is to hide the identity of data-items, *not* the existence of interaction with the user. Applications could include patent databases, stock quotes, media databases, etc.

In the basic model, Bob holds an n-bit string x (the database) and Alice wants to retrieve x_i while keeping i private (n is very large). We are interested in the communication complexity of the protocol.

Using techniques from communication complexity, we can prove a lower bound of $1 + \log n$ bits for any such protocol, without using the requirement of privacy. In fact, the simple protocol in which Alice sends i and Bob sends back x_i requires $1 + \log n$ bits, but is not private.

The trivial solution is to let Bob send the entire string x to Alice, requiring n bits. (Note that we are not concerned about the privacy of the database.) [CGKS95], who first formulated the Private Information Retrieval (PIR) problem, proved that we cannot do better: In any $1 -$ *server* PIR with *information theoretic* privacy[1], the communication is at least n bits.

This suggests two directions:

- Replicate the same database in $k \geq 2$ servers: We provide unconditional privacy against t (colluding) servers.
- Assume that Bob has bounded computational power: We provide computational privacy, based on cryptographic hardness assumptions.

Related approaches

- Alice asks Bob for additional random queries: This still reveals a lot of information about i.
- Use secure multiparty computation techniques to compute the function $x_i = f(i, x)$ *privately*: This is very inefficient as the communication needed is polynomial in n.
- Anonymity: hides the identity of Alice, *not* the fact that x_i is retrieved.

[1] For every possible content of the database, x and any two indices i and j, Bob should not be able to distinguish between the case that Alice holds i and the case that Alice holds j, i.e., the communication between Alice and Bob should be identically distributed, irrespective of the index i.

- Oblivious Transfer: One-out-of-n oblivious transfer is similar to single server non-trivial PIR, except that (a) the latter also requires the communication complexity to be less than n and (b) the former requires Alice not to learn any information about the rest of the database.

An Information Theoretic PIR scheme

Suppose two (non-colluding) servers D_0 and D_1 hold the same database x. Alice picks a random subset Q^0 of $[n]$ (i.e., $\{1, 2, \ldots, n\}$) by picking each element with probability $\frac{1}{2}$. Let $Q^1 = Q^0 \oplus i$ be obtained by complementing the presence of query bit, i in Q^0 (i.e., include i if it is absent in Q^0 and vice versa). Clearly Q^1 is also a random subset. D_0 gets Q^0 (as a bit string) from Alice and sends back the XOR of the bits with indices in Q^0 ($\bigoplus_{j \in Q^0} x_j$). D_1 acts similarly. Alice obtains x_i by XOR-ing the bits received while neither database receives any information about i, as each gets a uniformly distributed subset of $[n]$. Even though this scheme does not save any communication compared to the trivial solution, it serves as a building block for more efficient schemes.

We can obtain a $O(\sqrt{n})$ protocol with $k = 2^2 = 4$ databases, say, D_{00}, D_{01}, D_{10} and D_{11}. Let $n = l^2$ and associate bits of x with a $l*l$ array A of bits. Let (i_1, i_2) be the desired entry in the array. Alice chooses uniformly and independently random subsets $Q_1^0, Q_2^0 \subseteq [l]$. Define $Q_1^1 = Q_1^0 \oplus i_1$ and $Q_2^1 = Q_2^0 \oplus i_2$. Alice sends $(Q_{\sigma_1}^1, Q_{\sigma_2}^2)$ to the database $D_{\sigma_1 \sigma_2}$ which then sends back the XOR of all bits in the rectangle defined by $(Q_{\sigma_1}^1, Q_{\sigma_2}^2)$ ($\bigoplus_{j_1 \in Q_{\sigma_1}^1,\ j_2 \in Q_{\sigma_2}^2} x_{j_1 j_2}$). Alice obtains $x_{i_1 i_2}$ by XOR-ing the four bits received. As each database gets a pair of uniformly distributed subsets of $[l]$, the scheme is private. The communication used is $O(\sqrt{n})$. These two schemes can be viewed as the special cases of a protocol with $k = 2^d$ databases and communication $O(2^d.(d.\sqrt[d]{n} + 1))$.

Known Communication Upper Bounds

Information theoretic PIR with k servers:

- 2 servers - $O(\sqrt[3]{n})$ communication [CGKS95]
- k servers - $O(n^{1/\Omega(k)})$ communication [CGKS95, Amb97, BIKR02]
- $\log n$ servers - $Polylog(n)$ communication [BF90, CGKS95]

Single server, computational PIR:

- $Polylog(n)$ communication, using cryptographic hardness assumptions [CG97, KO97, CMS99]

Extensions

- PIR of Blocks: This is a more realistic model in which the data is partitioned into blocks (or records) rather than single bits. Information theoretic PIR of blocks is discussed in [CGKS95, CGN97].
- Private Information Storage: [OS97] give protocols for private reading and writing in both information-theoretic and computational privacy models.

- Symmetrical PIR [Aso01]: Alice should not be able to learn more than one record of the database. Symmetrical (Database) privacy is important for practical applications (eg: billing the user). This is very similar to one-out-of-n oblivious transfer, with limit on the communication used.

Discussion and Open Problems

- Access control and anonymity have somewhat contradictory objectives. How do we achieve both simultaneously ?

- With more than one server, can we do better in computational PIR ? Can we reduce the exponent (of $\log n$ in the polylog term) for a single server computational PIR protocol by using more servers and the techniques from information-theoretic PIR ?

- **Sublinear Computation**: All existing protocols require high computation by the servers (*linear time per query*). [BIM00] proposed the model of *PIR with preprocessing*, in which each server is allowed to store polynomially many additional bits (by preprocessing x), which helps it to answer each query with less computation. Apart from giving different schemes, they show that the expected computation of all the servers is at least n if no extra bits are stored and is $\Omega(n/e)$ with e extra bits. As the size of the database in practical applications is huge, it would be desirable to **improve time complexity via preprocessing / amortization / off-line computation**.

- **Weaker notion of privacy**: In information-theoretic as well as computational privacy models, we require that the database should *not learn any information* about the query. Can we relax this stringent requirement so as to improve the communication/time complexity ? What is the right model of privacy for practical database problems ?

- **PIR for a more general query**: Can we do better than the trivial approach (i.e., retrieve each bit involved in the query privately) for the following:
  - **Range queries**: Eg: Find the number of ones in $\{x_i, x_{i+1}, \ldots, x_j\}$.
  - **Aggregation queries**: Eg: Find the sum of the entries in a rectangle in a two dimensional database.
  - How to perform other **SQL queries** in a PIR fashion ?

- **PIR by keywords**: Typically the users access a database with keywords, which are then internally translated by the database to physical addresses, using a search structure such as binary tree or hash table. [CGN97] provide a scheme to privately access data by keywords, by combining *any* search structure with *any* underlying PIR scheme. The user wants to privately learn if its string is one of the keywords present in a data structure (in the database) and if so, access the data it represents. How do we extend this model to more practical settings ?

- How can we **query google** in a privacy preserving manner ?

References

[Amb97] A. Ambainis. Upper bound on communication complexity of private information retrieval. In *Proc. of 24th ICALP*, 1997.

[Aso01] D. Asonov. Private information retrieval - an overview and current trends. In *Proc. of ECDPvA Workshop, Informatik*, 2001. `http://www.dbis.informatik.hu-berlin.de/research/2pir/publications/informatik01.pd`

[BF90] D. Beaver and J. Feigenbaum. Hiding instances in multioracle queries. In *Proc. of 7th STACS, LNCS Vol. 415, Springer Verlag*, 1990.

[BIKR02] A. Beimel, Y. Ishai, E. Kushilevitz, and J. Raymond. Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval. In *Proc. of 43rd FOCS*, 2002.

[BIM00] A. Beimel, Y. Ishai, and T. Malkin. Reducing the servers' computation in private information retrieval: PIR with preprocessing. In *Advances in Cryptology: CRYPTO*, 2000.

[CG97] B. Chor and N. Gilboa. Computationally private information retrieval. In *Proc. of 29th STOC*, 1997.

[CGKS95] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. *Journal of the ACM 45(6), 965-982*, 1995.

[CGN97] B. Chor, N. Gilboa, and M. Naor. Private information retrieval by keywords. Technical report, TR CS0917, Department of Computer Science, Technion, 1997. `http://citeseer.nj.nec.com/chor97private.html`.

[CMS99] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *Advances in Cryptology: EUROCRYPT*, 1999.

[KO97] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proc. of 38th FOCS*, 1997.

[OS97] R. Ostrovsky and V. Shoup. Private information storage. In *Proc. of 29th STOC*, 1997.