Upper bound on the communication complexity of private information retrieval

Andris Ambainis
Institute of Mathematics and Computer Science
University of Latvia
Raina bulv. 29, Riga, Latvia
e-mail: ambainis@cclu.lv

Abstract

We construct a scheme for private information retrieval with k databases and communication complexity $O(n^{1/(2k-1)})$.

1 Introduction

Much attention has been given to the problem of protecting a database from the user that tries to retrieve the information that he is not allowed to access[2, 8, 12].

In some scenarios, the opposite problem can appear: a user wishes to retrieve some infomation from a database without revealing to the database what information he needs. For example[7], an investor wishes to receive information about certain stock but he does not wishe others (even the database) to know in which particular stock he is interesed.

However, there is only one way to reach complete privacy: the user should ask for the copy of entire database. Otherwise, the database will get some information what the user wishes to know. This is not a good solution because it requires much time and much communiction from the database to the user.

If there are several identical copies of the database, another scenario is possible[7]:

The user asks a query to each database and combines the results of the queries, obtaining the desired information. Each query alone gives no information what user is interested in.

Chor, Coldreich, Kushilevitz, Sudan[7] introduced this model and constructed several schemes for the private retrieval of one bit from a database:

1. A scheme for 2 databases with $O(n^{1/3})$ communication. (n is the size of the database)

2. A scheme for k databases with $O(n^{1/k})$ communication.

3. A scheme for $O(\log n)$ databases with $O(\log^2 n \log\log n)$ communication.

They also considered modifications of these schemes which allow to retrieve blocks of information and give higher degree of privacy (knowing $k-1$ of k queries gives no information about which bit the user wishes to retrieve).

In this paper, we improve their result, constructing a protocol for k databases with $O(n^{1/(2k-1)})$ communication.

2 Related work

Instance hiding. In [1, 5, 6] the instance hiding problem has been studied. It is the problem of obtaining the i^{th} bit from the oracle so that i remains secret. The main difference is that instance hiding deals with databases of exponential size. Private information retrieval[7] considers the case when the size of the database is feasible quantity and the number of databases is small (constant or logarithmic).

Communication complexity. Several researchers have studied related problems in multiparty communication complexity. Pudlak, Rödl, Sgall[10, 11] and Ambainis[3] have considered the problem of computing $x_{(i+j)\bmod n}$ where x is a string of n bits and i, j are integers in the following situation:

Player 1 knows x, i, Player 2 knows x, j. Each of them sends one message to Player 3. Player 3 computes the result, using only the messages received from Players 1 and 2.

Any protocol for the above problem can be easily transformed into protocol for private information retrieval. Thus, we can obtain nontrivial protocols for private information retrieval with $o(n)$ communication.

Another communication complexity problem was studied by Babai, Kimmel and Lokam[4]. It also can be applied to private information retrieval.

However, all these protocols are less efficient than the protocols for private information retrieval designed in [7]. Still, the ideas from [3, 4, 10, 11] (not explicit protocols) can be useful in the study of private information retrieval.

Private information storage. Ostrovsky and Shoup[9] have extended the results of [7] and designed schemes for private information storage. Using their schemes, the user can both read and write to the database without revealing which bit is accessed. They have shown that any protocol for private information retrieval can be transformed to the protocol for private information storage with a slight increase in the number of databases and communication.

3 Result

Consider some protocol for private information retrieval. Does the user uses all bits in messages from the databases? In some protocols, only a few bits are really neccessary. If the user knows in advance which bits are necessary, two protocols can be combined, obtaining the third with more databases and less communication.

Below, we show how to combine a protocol for 2 databases and a protocol for $k-1$ databases, obtaining the protocol for k databases with less communication.

1. The user in k database protocol simulates the user in the protocol for 2 databases. Let x_1 denote the message sent to the 1^{st} database and x_2 the message sent to the 2^{nd} database in the 2 database protocol.

   The user sends x_1 to the 1^{st} database and x_2 to the 2^{nd}, ..., k^{th} database.

2. Then, the user computes the length of the reply from the 2^{nd} database in 2 database protocol and the positions of necessary bits in this reply. Further, m denotes the length and $n_1, \ldots, n_i$ denote the positions of necessary bits.

The user simulates the user in the protocols for $k-1$ databases where n_1^{th}, ..., n_i^{th} bits from m bit database are retrieved, sending to the $(i+1)^{\text{st}}$ database the messages which are sent to the i^{th} database in the $(k-1)$ database protocol.

3. The 1^{st} database simulates the 1^{st} database in 2 database protocol and sends the user the same message.

4. The 2^{nd}, ..., k^{th} databases simulate the 2^{nd} database in 2 database protocol. Instead of sending the message to the user, they consider it as a new m-bit database.

   Further, they simulate databases in the $(k-1)$ database protocol for retrieval of n_1^{th}, ..., n_i^{th} bits and send the messages from these protocols to the user.

5. The user simulates the user in the $(k-1)$ database protocol for retrieval of n_1^{th}, ..., n_i^{th} bits. Then, knowing the message from the 1^{st} database and all the necessary bits from the second message, the user simulates the user in the 2 database protocol. The result of this simulation is the bit that the user wishes to retrieve.

If we wish to apply this idea, 2 database protocol should satisfy certain constraints:

1. The most of communication goes from the databases to the user. (The amount of communication from the user to databases increases when two protocols are combined. Hence, if it is already large, the combination is useless.)

2. Only few bits from the received messages are really important to the user.

3. The user knows in advance which bits are necessary, i.e. the positions of these bits do not depend on the databases' contents.

Below, we use the idea of combining two protocols to prove

Theorem 1 *Let $k \geq 2$. There exists a protocol for private information retrieval with k databases and $O(n^{2k-1})$ bits of communication.*

Proof. By induction.

The protocol for 2 databases was constructed by Chor, Goldreich, Kushilevitz and Sudan[7]. The protocol for k databases is obtained as the combination of the protocols for 2 databases and $(k-1)$ databases.

First, we describe the 2 database protocol that we use to obtain a k database protocol from a $(k-1)$ database protocol.

1. Let $l = \lceil \sqrt[2k-1]{n} \rceil$. The database can be considered as $2k-1$ dimensional cube $\{0, \ldots, l-1\}^{2k-1}$. Each position $i \in \{0, \ldots, n-1\}$ in the database coresponds to some position $(i_1, \ldots, i_{2k-1})$ in the cube.

   The user chooses independently $(2k-1)$ random subsets of $\{0, \ldots, l-1\}$: S_1^1, ..., S_{2k-1}^1. Let $S_1^2 = S_1^1 \oplus i_1$, ..., $S_{2k-1}^2 = S_{2k-1}^1 \oplus i_{2k-1}$ where $(i_1, \ldots, i_{2k-1})$ is the position of the required bit in the $(2k-1)$ dimensional cube.

2. The 1^{st} database computes the exclusive-or of the bits in positions $(j_1, \ldots, j_{2k-1})$ such that $j_1 \in S_1^1$, ..., $j_{2k-1} \in S_{2k-1}^1$ and sends it to the user.

   The database also computes the exclusive-or of the bits in positions $(j_1, \ldots, j_{2k-1})$ such that $j_1 \in S_1'$, ..., $j_{2k-1} \in S_{2k-1}'$ for each possible S_1', ..., S_{2k-1}' such that

   (a) $S_j' = S_j^1 \oplus t$ for some $j \in \{1, \ldots, 2k-1\}$ and $t \in \{0, \ldots, l-1\}$;
   (b) $S_i' = S_i^1$ for all $i \neq j$.

   The exclusive-xor for each possible S_1', ..., S_{2k-1}' is sent to the user, too.

3. The 2^{nd} database computes the exclusive-or of the bits in positions $(j_1, \ldots, j_{2k-1})$ such that $j_1 \in S_1^2$, ..., $j_{2k-1} \in S_{2k-1}^2$ and sends it to the user.

   Further, the 2^{nd} database computes the exclusive-or of the bits in positions $(j_1, \ldots, j_{2k-1})$ such that $j_1 \in S_1'$, ..., $j_{2k-1} \in S_{2k-1}'$ for each possible S_1', ..., S_{2k-1}' such that

   (a) For each $i \in \{1, \ldots, 2k-1\}$ S_i' is equal to S_i^2 or $S_i^2 \oplus t_i$ for some $t_i \in \{0, \ldots, l-1\}$;

(b) There exist at least two $i \in \{1, \dots, 2k-1\}$ such that $S_i' = S_i^2$.

The exclusive-xor for each possible S_1', …, S_{2k-1}' is sent to the user, too.

4. For each possible S_1', …, S_{2k-1}' such that S_i' is either S_i^1 or S_i^2, the user finds the exclusive-or of bits in positions $(j_1, \dots, j_{2k-1})$ satisfying $j_1 \in S_1'$, …, $j_{2k-1} \in S_{2k-1}'$:

   (a) If $S_i' = S_i^2$ for at most one i, then the exclusive-or is one of the bits sent by the 1st database.

   (b) If $S_i' = S_i^2$ for at least two i, then the exclusive-or is one of the bits sent by the 2nd database.

   The user computes the exclusive-or of all these values. It is the necessary bit from the database.

   ($S_j^2 = S_j^1 \oplus i_j$. Hence, i_j belongs to exactly one of S_j^1 and S_j^2 and $i_1 \in S_1'$, …, $i_{2k-1} \in S_{2k-1}'$ for exactly one choice of $S_1', \dots, S_{2k-1}'$.

   For each other position $(i_1', \dots, i_{2k-1}')$ we have $i_1' \in S_1'$, …, $i_{2k-1}' \in S_{2k-1}'$ for even number (possibly zero) of the combinations $S_1', \dots, S_{2k-1}'$.

   Hence, the exclusive-or that the user computes contains the bit in position $(i_1, \dots, i_{2k-1})$ exactly once and any other bit even number of times. It follows that this exclusive-or is equal to the bit in position $(i_1, \dots, i_{2k-1})$, i.e. the bit that the user wishes to retrieve.)

The amount of transmitted bits.

1. Communication from the user to the databases.

   To transmit a set S_j^i, the user needs $l = \sqrt[2k-1]{n}$ bits. (For each $o \in \{0, \dots, l-1\}$, the user must say whether $o \in S_j^i$.) The user transmits $2k-1$ sets (S_1^1, …, S_{2k-1}^1) to the 1st database and $2k-1$ sets to the 2nd database.

   So, the total amount of communication in this direction is $2(2k-1)\sqrt[2k-1]{n} = O(\sqrt[2k-1]{n})$.

2. Communication from the 1st database to the user.

The 1st database computes the exclusive-or of the bits for several combinations of $S'_1, \ldots, S'_{2k-1}$ and sends it to the user. The amount of bits transmitted by the 1st database is equal to the number of the combinations of $S'_1, \ldots, S'_{2k-1}$, i.e. $(2k-1)l+1$.

k is a constant and $l = \lceil \sqrt[2k-1]{n} \rceil$. Hence, the amount of communication in this direction is $O(\sqrt[2k-1]{n})$, too.

3. Communication from the 2nd database to the user.

   Similarly to the previous case, the amount of bits transmitted by the 2nd database is equal to the number of the combinations of $S'_1, \ldots, S'_{2k-1}$ for which the exclusive-or was computed.

   For the 2nd database, the amount of such combinations is at most $(2^{2k-1} - 2k)l^{2k-3} = O(n^{(2k-3)/(2k-1)})$ because:

   (a) Those i for which $S'_i \neq S^2_i$ form a subset of $\{1, \ldots, 2k-1\}$ with at most $2k-3$ elements. (For at least two $i \in \{1, \ldots, 2k-1\}$, $S^2_i = S'_i$.)
   The amount of such subsets is $2^{2k-1} - 2k$.

   (b) If we have chosen for which i $S^2_i \neq S'_i$, it remains to choose t_i. For each i there are l possible values of t_i.
   There are at most $2k-3$ values of i for which t_i must be chosen. Hence, there are at most l^{2k-3} possible combinations of t_i.

So, the user transmits $O(\sqrt[2k-1]{n})$ bits, the 1st database $O(\sqrt[2k-1]{n})$ bits and the 2nd database $O(n^{(2k-3)/(2k-1)})$ bits.

From the 2nd database's answer the user needs constant amount $(2^{2k-1} - 2k)$ of bits. The positions of these bits in the message from the 2nd database do not depend on the contents of database.

Hence, we can combine the described protocol with a $(k-1)$ database protocol, using the method described at the beginning of this section.

Communication in the k database protocol.

1. Communication from the user to the databases.

   The user sends to the databases:

   (a) The information from the 2 database protocol: $\sqrt[2k-1]{n}$ bits to each database.

(b) The information for simulations of $(k-1)$ database protocol: $O(\sqrt[2k-3]{m})$ bits where m is the length of the message from the 2^{nd} database in the 2 database protocol. We proved that $m = O(n^{(2k-3)/(2k-1)})$. Hence, $O(\sqrt[2k-1]{n})$ bits are transmitted for this purpose.

2. Communication from the 1^{st} database to the user. It is the same as in the 2 database protocol, i.e. $O(\sqrt[2k-1]{n})$ bits.

3. Communication from the $2^{\text{nd}}, \ldots, k^{\text{th}}$ databases to the user.

   In each simulation of $(k-1)$ database protocol, these databases communicate

$$O(\sqrt[2k-3]{m}) = O(\sqrt[2k-3]{n^{(2k-3)/(2k-1)}}) = O(\sqrt[2k-1]{n})$$

   bits. The amount of simulations performed by the databases is equal to the amount of bits needed by the user from 2^{nd} database's message, i.e. constant. Hence, the communication by these databases is $O(\sqrt[2k-1]{n})$, too.

We have constructed the protocol with k databases and $O(\sqrt[2k-1]{n})$ communication from a protocol with $(k-1)$ databases and $O(\sqrt[2k-3]{n})$ communication. $\square$

Using the construction of Ostrovsky and Shoup[9] and the protocol described above, we can obtain a scheme in which both reading and writing are private. This scheme has $(k+1)$ databases $(k>2)$ and $O(n^{1/(2k-1)} \log n)$ communication.

References

[1] M. Abadi, J. Feigenbaum and J. Kilian, *On hiding information from an oracle*, Journal of Computer and System Sciences, 39(1989), pp. 21-50

[2] N. Adam and J. Wortmann, *Security control methods for statistical databases: a comparative study*, ACM Somputing Surveys, 21(1989), pp. 515-555

[3] A. Ambainis, *Upper bounds on multiparty communication complexity of shifts*, Proceedings of STACS'96, Lecture Notes in Computer Science, vol. 1047(1996), pp. 631-642

[4] L. Babai, P. Kimmel, S.V. Lokam, *Simultaneous messages versus communication,*, Proceedings of STACS'95, Lecture Notes in Computer Science, vol. 900(1995), pp. 361-372

[5] R. Beaver, J. Feigenbaum, *Hiding instances in multioracle queries*, Proceedings of STACS'90, Lecture Notes in Computer Science, vol. 415(1990), pp. 37-48

[6] R. Beaver, J. Feigenbaum, J. Kilian, P. Rogaway, *Security with low communication overhead*, Crypto'90

[7] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan, *Private information retrieval*, Proceedings of 36th Symposium on Foundations of Computer Science, pp. 41-50

[8] D. Denning, *Cryptography and Data Security*, Addison-Wesley, 1982

[9] R. Ostrovsky, V. Shoup, *Private information storage*, manuscript

[10] P. Pudlak, V. Rödl, *Modified ranks of tensors and the size of circuits*, Proceedings of 25-th ACM STOC, 1993, pp. 523-531.

[11] P. Pudlak, V. Rödl, J. Sgall, *Boolean circuits, ranks of tensors and communication complexity*, to appear in SIAM J. Computing.

[12] J. D. Ullman, *Principles of Database Systems.* 1982.