Advanced Topics in Cryptography

Lecture 11
Private Information Retrieval (PIR)

Benny Pinkas

Related papers

- PIR
  - B. Chor, E. Kushilevitz, O. Goldreich, M. Sudan: Private Information Retrieval. J. ACM 45(6): 965-981 (1998)

Private Information Retrieval (PIR)

- A special case of secure two-party computation
  - One party (aka sender, server) has a large database.
  - The other party (aka receiver, client) wants to learn a specific item in the database, while hiding its query from the database owner.
  - For example, a patent database, or web access.
- The model:
  - Sender has N bits, $b_1,\ldots,b_N$.
  - Receiver has a query $i \in [1,N]$.
  - Receiver learns b_i (and possibly additional information)
  - Sender learns nothing.
  - The communication is sublinear, i.e. $o(N)$.
- (This model is not very realistic, but is convenient since it's the most basic form of PIR)

Results

- Unconditional security
- Unconditional privacy, with a **single** server, requires $\Omega(N)$ communication and is therefore inefficient [CGKS]
  - A transcript $c=(x,i)$ is called "*possible*" if for a database x and user interested in i there is a positive probability for c.
  - Fix i. For every possible value of the database there is a value for c. But since the communication is smaller than N bits, the total number of possible transcripts c is smaller than 2^N.
  - Therefore there are two values of the pair (database,query): (x,i) and (y,i), s.t. c is possible for both.
  - By the privacy requirement, c must be possible for (x,i) for all possible values of i, and similarly for every (y,i) (otherwise database owner learns i).
  - Since $x \neq y$, there is an index j for which $x \neq y$.
  - But c is possible for both (x,j) and (y,j). A contradiction! (since the receiver's output is a function of c alone)

Results

- Unconditional security
  - consider a setting where
    - $k \geq 2$ servers know the database
    - Servers do not collude. No single server learns about i.
    - The client can send different queries to different servers

- Results [CGKS and subsequent work]
  - 2 servers: $O(N^{1/3})$ communication
  - K servers: $O(N^{1/\Omega\{k\}})$ communication, or even a bit better.
  - log N servers: Poly(log(N)) communication.

Two-server PIR

- Best result: $N^{1/3}$ communication. We will show a protocol with $N^{1/2}$ communication.
- There is a simple protocol with O(N) communication:
  - Receiver picks a random vector V_0 of length N.
  - It sets V_1 to be equal to V_0, except for the bit in location i, whose value is reversed.
  - It sends V_0 to $Server_0$, and V_1 to $Server_1$.
  - $Server_0$ sends to R a bit c^0, which is the xor of the bits b_i, for which the corresponding bit in V_0 is 1. Namely $c^0 = \oplus V_{0,i} b_i$.
  - $Server_1$ sends a bit c^1, computed using V_1.
  - The receiver computes $b_i = c^0 \oplus c^1$.
  - Privacy: Each server sees a random vector.
  - Protocol seems suboptimal since communication from receiver to client is much higher than in the other direction.

Two-server PIR with $O(N^{1/2})$ communication

- Suppose $N=m\times m$.
- Database is $\{ b_{i,j} \}_{1\leq i,j \leq m}$
- Receiver is interested in $b_{\alpha,\beta}$
- It picks a random vector V_0 of length m.
- V_1 is equal to V_0 with bit α reversed.
- Sends V_0 to S_0 and V_1 to S_1
- S_0 computes and sends the corresponding xor of every column: $c^0_j = \oplus_{i=1\ldots m} V_{0,i}\, b_{i,j}$ (m results in total)
- S_1 computes and sends similar values c^1_j with V_1
- The receiver ignores all values but c^0_β, c^1_β. Computes $b_{\alpha,\beta} = c^0_\beta \oplus c^1_\beta$ (but can also compute all $b_{\alpha,j}$).
- What else does the receiver learn?

Four-server PIR with $O(N^{1/2})$ communication (same communication as in the two server case)

- Here the receiver can only compute $b_{\alpha,\beta}$ (and some additional xors of inputs)
- Four servers, $S_{0,0}, S_{0,1}, S_{1,0}, S_{1,1}$. Each sends only O(1) bits.

- Database is $\{ b_{i,j} \}_{1 \le i,j \le m}$. Receiver is interested in $b_{\alpha,\beta}$.
- Receiver picks random V^R_0, V^C_o of m bits each. Computes V^R_1, V^C_1 by reversing bit α in V^R_0, and bit β in V^C_0.
- Sends vectors V^R_0, V^C_0 to $S_{0,0}$, vectors V^R_0, V^C_1 to $S_{0,1}$, etc.

- Each $S_{a,b}$ computes the xor of the bits whose coordinates correspond to "1" values in $V^r_a \times V^C_b$, and returns the result.
- The receiver computes the xor of the bits it receives…
- Correctness? Communication? Privacy?

Four-server PIR with $O(N^{1/3})$ communication

- We showed a four-server PIR where the receiver sends $O(N^{1/2})$ bits and each server sends $O(1)$ bits.
- We can use this protocol as a subroutine:
  - Given a database of size N, divide it to $N^{1/3}$ smaller databases of size $N^{2/3}$ each.
  - Apply the previous protocol to all of them in parallel. The receiver constructs sets V^R, V^C for the database which stores the bit it is interested in, and uses these sets for all databases.
  - The receiver sends $O((N^{2/3})^{1/2})=O(N^{1/3})$ bits.
  - Each sender returns $N^{1/3} \cdot O(1) = O(N^{1/3})$ bits.
  - The receiver learns one value from every database.
- (why didn't this approach work with the two server protocol?)

Computational PIR [Kushilevitz Ostrovsky]

- Security is not unconditional, but rather depends on a computational assumption about the hardness of some problem
- Enables to run PIR with a single server (unlike the infeasibility result for unconditional PIR)

Computational PIR

- We will show computational PIR based on the existence of additively homomorphic encryption
- Additively homomorphic encryption
  - Semantically secure public key encryption
  1. Given E(x) it is possible to compute, without knowledge of the secret key, E(c·x), for every c.
  2. Given E(x) and E(y), it is possible to compute E(x+y)

- We actually need a weaker property
- Can be implemented based on the hardness of ElGamal encryption, Quadratic Residuosity, etc.
- We might talk more about additively homomorphic encryption in future lectures.

Computational PIR: basic scheme

- Suppose $N = s \times t$.
- Database is $\{ b_{i,j} \}_{1 \le i \le s,\ 1 \le j \le t}$
- Receiver is interested in $b_{\alpha,\beta}$
- Receiver computes a vector V of size t: $(E(e_1),\ldots,E(e_t))$, where $e_j=0$ if $j \neq \beta$, and $e_\beta=1$.
- Receiver sends V to sender.
- Sender computes, for every row $1 \le i \le s$,
  $c_i = \sum_{j=1}^{t} E(e_j \cdot b_{i,j}) = E(\sum_{j=1}^{t} e_j \cdot b_{i,j}) = b_{i,\beta}$ (O(N) exponen.)

- Sender sends $c_1,\ldots,c_s$ to receiver. Receiver learns c_α.
- Setting $s=t=N^{1/2}$ results in $O(N^{1/2})$ communication.
- Is this secure? Can we do better?

Computational PIR: reducing the communication via recursion

- In the final step the sender sends s values, while the receiver is interested in only one of them.
  - They can run a PIR in which the receiver learns this value!
- Set $t=N^{1/3}$. Run the previous protocol without the final step.
  - $O(t)=O(N^{1/3})$ communication for this step.
  - At the end of the protocol the sender has $N_1=N^{2/3}$ values (each of length k, which is the length of the encryption).
  - The parties run the previous protocol k times (for each bit of the answers),setting $s=t=(N_1)^{1/2}=N^{1/3}$.
  - Communication: $R \Rightarrow S$: $kN^{1/3}+k^2N^{1/3} = O(N^{1/3})$
  - $S \Rightarrow R$: $k^2N^{1/3} = O(N^{1/3})$

Computational PIR: continuing the recursion

- Start from $t = N^{1/4}$.
- There are $N^{3/4}$ answers, each of length k.
- Run the previous protocol on these answers, once for every bit of the answer (a total of k times).
  - The communication overhead is $O(k^3 N^{1/3})$ bits.

- In the general case
  - The recursion has L steps
  - Start from $t=N^{1/(L+1)}$
  - The total communication is $O(N^{1/(L+1)} \cdot k^L)$
  - Setting $L=O((\log N / \log k)^{1/2})$ results in $N^{1/(L+1)} = k^L$, and total communication $2^{O(\sqrt{(\log N \log k)})}$

- There is another PIR protocol with polylogN comm.

Sender privacy

- PIR does not prevent receiver from learning more than a single element of the database.

- PIR
  - Sender learns nothing about the query (i.e., about i).
  - Receiver might learn more than the item it is interested in (b_i).
  - Communication is sublinear in N.

- 1-out-of-N Oblivious transfer
  - Sender learns nothing about the query (i.e., about i).
  - Receiver learns nothing but the result of its query (b_i).
  - Communication can be linear in N.

- Is it possible to get the best in both worlds?

Symmetric PIR (SPIR)

- SPIR is PIR with sender privacy:
  - Sender learns nothing about the query (i.e., about i).
  - Receiver learns nothing but the result of its query.
  - Communication is sublinear in N.

- OT + PIR = SPIR
  - Recall 1-out-of-N OT:
    - 2logN keys are used to encrypt N items.
    - Receiver uses logN invocations of OT to learn logN keys.
    - All N encrypted items are sent to the receiver, who can decrypt on of them.
    - The last step can be replaced by PIR.