

Querying Databases Privately

Eyal Kushilevitz – Technion

<http://www.cs.technion.ac.il/~eyalk>

Private Information Retrieval (PIR)

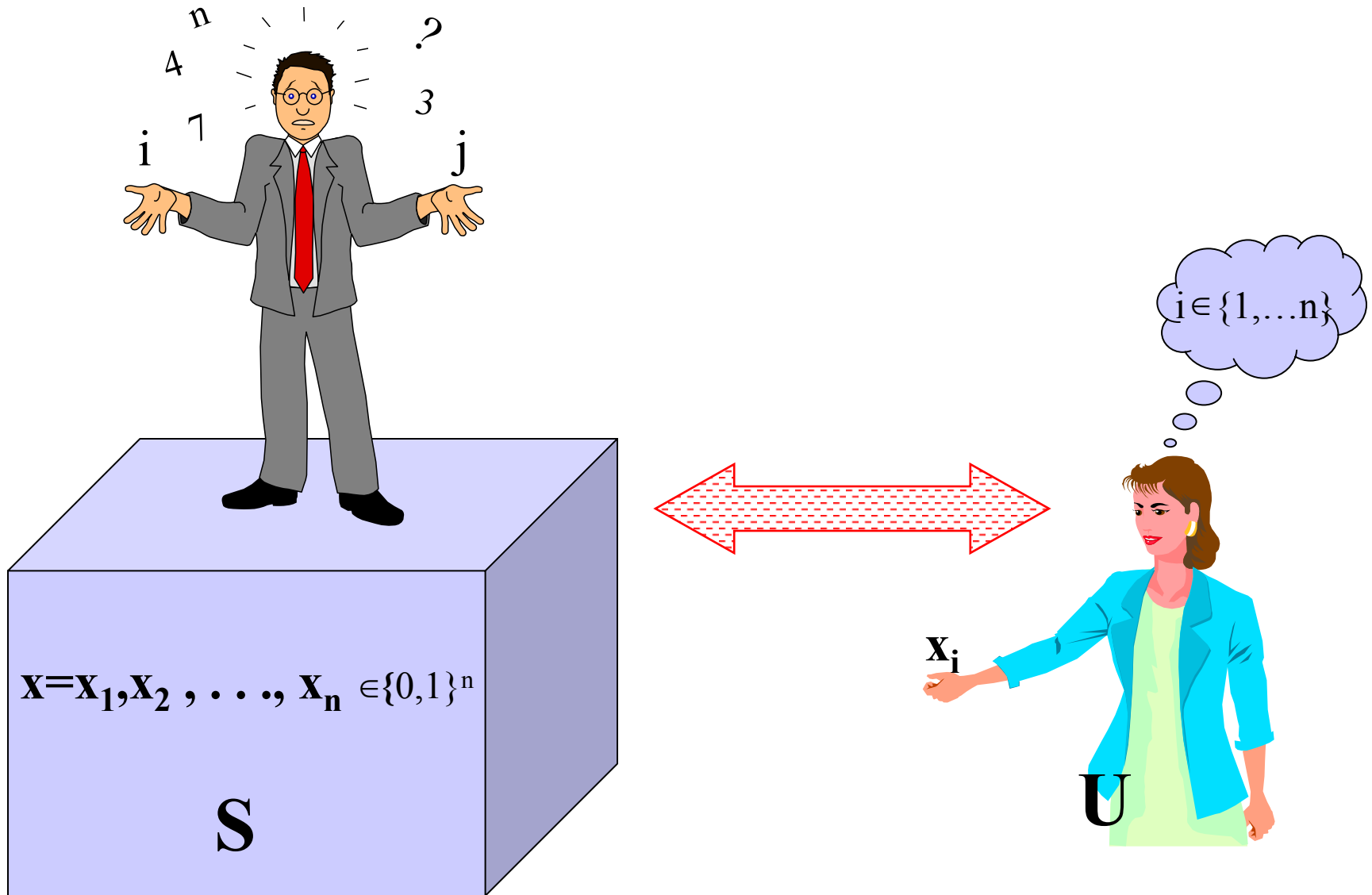
- **Goal:** allow user to query database while hiding the identity of the data-items she is after.
- **Note:** hides identity of data-items; not the mere existence of interaction with the user.
- **Motivation:** patent databases; stock quotes; web access; many more....
- **Paradox(?):** imagine buying in a store without the seller knowing what you buy.

(Encrypting requests is useful against third parties; not against owner of data.)

Modeling

- **Data:** n -bit string x
 n should be thought of as **very large**
- **User:** wishes
 - to retrieve x_i and
 - to keep i private
- **Remark:** this is the most basic version; serves as a building block to more involved types of retrieval.

Private Information Retrieval (PIR)



Some “solutions”

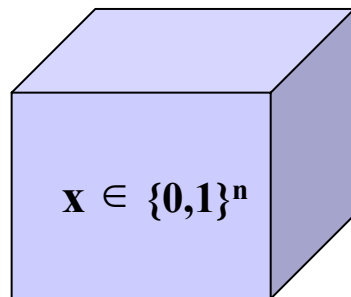
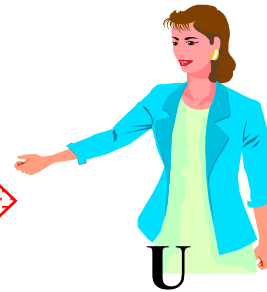
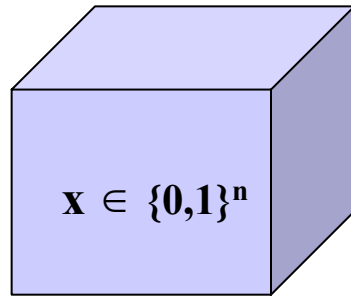
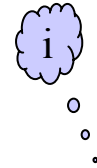
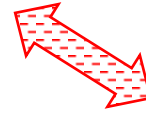
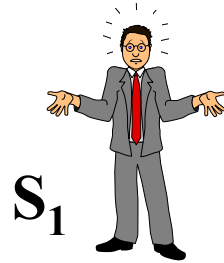
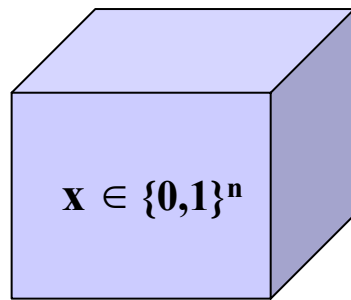
1. User asks for a copy of x (buying the whole store...)
Drawback: communication complexity is large.
2. User asks for additional random indices.
Drawback: gives partial information (e.g., indices in which the user is not interested).
3. Employ general crypto protocols to compute x_i (and more complicated functions $f(x, i)$) privately.
Drawback: highly inefficient (polynomial in n).
4. Anonymity (e.g., via Anonymizers).
Note: addresses a different concern: hides identity of user; not the fact that x_i is retrieved (+ questions of trust).

Fact: PIR in this setting requires $\Omega(n)$ communication bits

PIR: Settings and Assumptions

- **Replication** [CGKS,...]
 - k servers S_1, \dots, S_k each hold a **copy** of x .
 - Each set of t servers gets **no info** about i .
 - $t=1$ by default
- **Computational PIR** [CG,KO,CMS,...]
 - Computational (cryptographic) Assumptions
 - **Computational Privacy**
 - Replication not needed

k -Server PIR



Correctness: User obtains x_i

Privacy: No *single* server gets information about i

Known Comm. Upper Bounds

Multiple servers, information-theoretic PIR:

- k servers, comm. $n^{1/\Omega(k)}$ [CGKS95, Amb96, ..., BIKR02]
- $\log n$ servers, comm. $\text{polylog}(n)$ [BF90, CGKS95]

Single server, computational PIR:

Comm. $\text{Poly}(\log(n), \text{security-parameter})$

Under appropriate computational assumptions [KO97, CMS99]

Example 1

2-server information theoretic PIR that gets $n^{1/2}$ -bit block.

- Set $n=m^2$; View database as $B_1 B_2 \dots B_m$, each of length m .
- U wishes to retrieve B_i
- $U \rightarrow S_1$ A_1 a random subset of $[m]$
- $U \rightarrow S_2$ $A_2 = A_1 \oplus i$
- $S_b \rightarrow U$ $a_b = \bigoplus_{j \in A_b} B_j$ (for $b=1,2$)
- U computes $a_1 \oplus a_2 = B_i$
- **Privacy:** Each of A_1, A_2 independent of i .
- (total) communication complexity $4m = O(n^{1/2})$.

Example 2

1-server computational PIR that gets $n^{1/2}$ -bit block.

- As before: set $n=m^2$; U wishes to retrieve B_i
- **Homomorphic encryption:** $E(b_1) \otimes E(b_2) = E(b_1 \oplus b_2)$,
 $E(\cdot)$ of length sec. parameter
Implementation: quadratic residuosity, DDH,...
- **U** \rightarrow **S**: a_1, \dots, a_m where $a_i = E(1)$ and for $r \neq i$ $a_r = E(0)$.
- **S**: computes $E^*(B_j) = b_{j1}, \dots, b_{jm}$ where
if $B_{jr} = 0$ then $b_{jr} = E(0)$ and if $B_{jr} = 1$ then $b_{jr} = a_r$
(Note: for $j \neq i$ $E^*(B_j) = E(0), \dots, E(0)$)
- **S** \rightarrow **U**: $\otimes_{j \in [m]} E^*(B_j)$ which is exactly $E^*(B_i)$
- **U**: decrypts B_i

Communication complexity: $O(m * \text{sec. parameter})$

PIR -- Related Work

- Extensions:
 - Symmetric PIR [GIKM,NP]
 - t -privacy [CGKS,IK,BI]
- More settings [OS,GGM,DIO,BIM,...]
- PIR as a building-block [NN,FIMNSW,...]

Current (& Future) Research

Focus so far: communication complexity

Obstacle: time complexity

- All existing protocols require **high computation by the servers** (*linear computation per query*).
- **Theorem**: expected computation of the servers is $\Omega(n)$

Major research goal:

Improving time complexity via

preprocessing / amortization / off-line computation

(single-user amortization solved [IKOS]; multi-user case is open)