Lecture 8 - Private Information Retrieval (PIR)

CS 355 - Spring 2019

April 24, 2019

Henry Corrigan-Gibbs

Logistics

* HW2 Due Friday, 4/26 at 5pm
  ↳ Come to OH if you need help!
* Please give feedback on PSETS
* Also, anonymous feedback form online.
  ↳ Anything that would improve course or make us better teachers.
* Grades for HW1 out now
  ↳ regrade policy

Plan

* Recap: Multiparty Computation
* PIR: What it is, why it's amazing
  ↳ Formal defn's
* Constructions
  - Two-server PIR
  - One-Server PIR

A "perfect" cryptosystem

Adopted from Spielman's "perfect algorithmic result"

1) Has a beautiful theory
2) Works in practice
3) Solves a problem that people should? care about

* When you're working on a problem, ask yourself how your work does against this rubric

Today

- One of my favorite "almost perfect" ideas in crypto
- Lots of activity, even in last few years
  - Even today at Stanford....
- A classic crypto result
  Seems impossible, then turns out to be simple/elegant

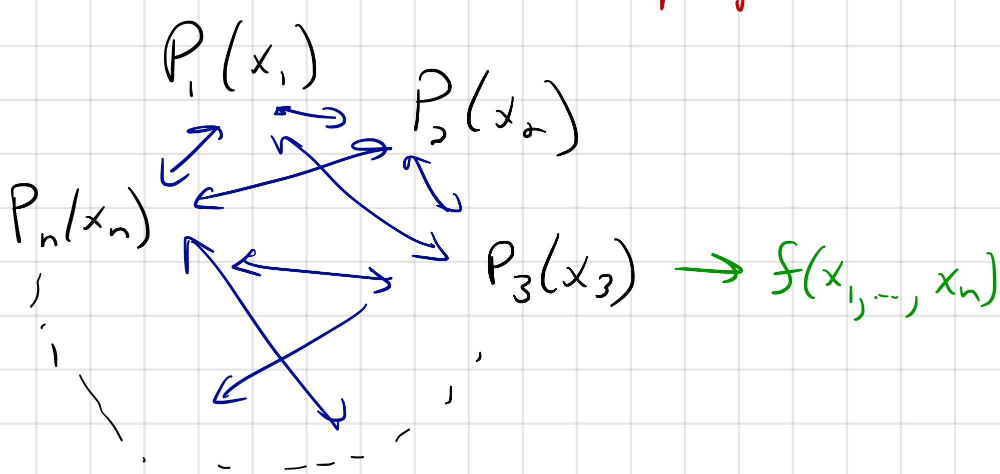
The catch: For reasons we'll see, it's not quite practical yet.....

Recap: MPC

- Each party P_i holds secret input x_i
- Parties want to compute a joint function $f(x_1, \ldots, x_n)$ of their private inputs

... without leaking anything else!

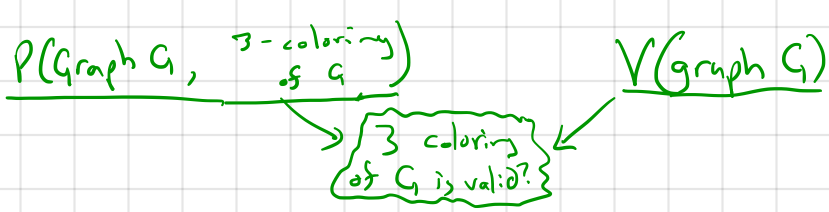
→ "Best possible" result — can compute **any** fn in secure multiparty manner



Why want this?

* Train a spam classifier over millions of peoples email w/o having to share mail
* Compute election results w/o having to publish votes
* Check if your password is in use in a database w/o leaking your password (or site leaking its DB)

⇒ Implies ZK ... e.g.

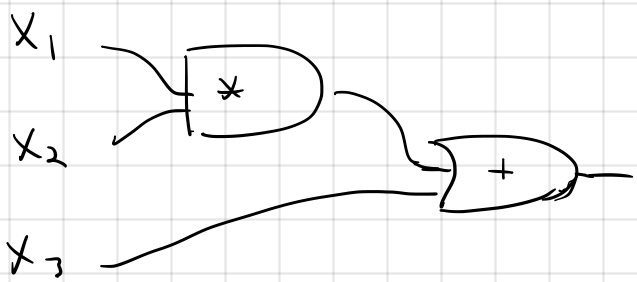


Many real-world complications

Recap: MPC

Idea: View $f(x_1, \ldots, x_n)$ as $\underbrace{\text{an arithmetic ckt}}_{\text{gates are } + \text{ and } * \text{ mod } p\text{, wires are values in } \mathbb{F}_p}$

Think: ints mod p



Note: Reexpressing computation $f(\cdot)$ as an arith ckt is without loss of generality.

↳ Any poly time computation has a poly-sized arith ckt

↳ If f has Boolean ckt of size S, it has an arithmetic ckt of size $O(S)$.

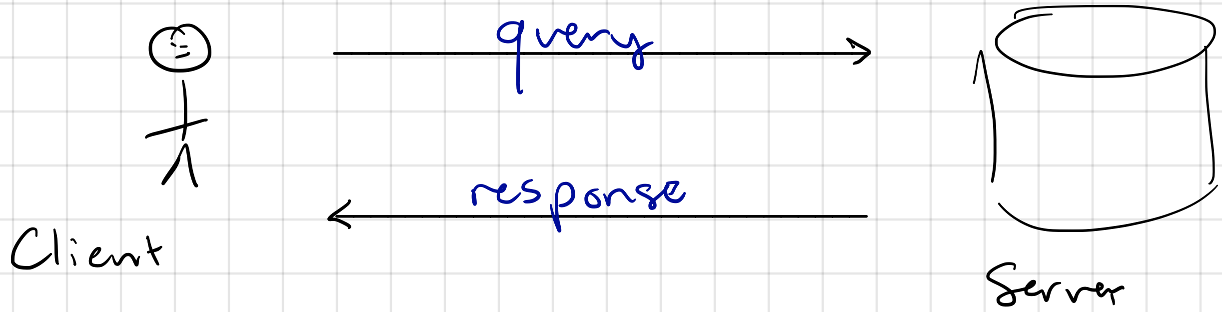
MPC Protocol (Ben-Or, Goldwasser, Wigderson '88)

* Parties start holding shares of input wires
* Parties jointly compute shares of internal wires
* Finally parties hold shares of output wire

↳ Publish shares to recover output $f(x_1, \ldots, x_n)$.

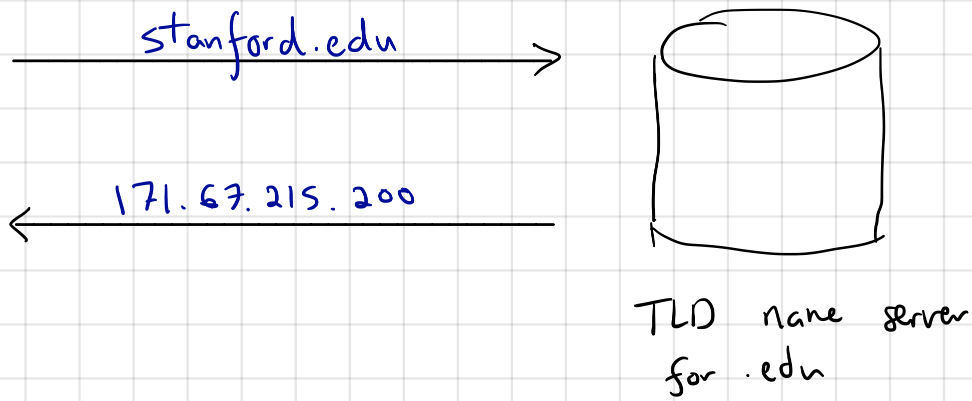
Private Information Retrieval

Every day on the Internet...

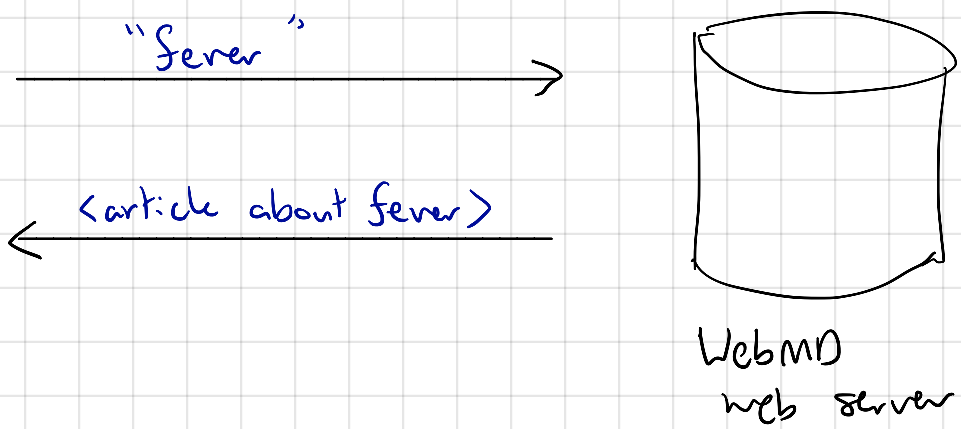


Examples:

1 DNS



2 WebMD



3 Many more: Querying stock price, looking at courses in course catalog, ...

Notice: The client's query can be sensitive!

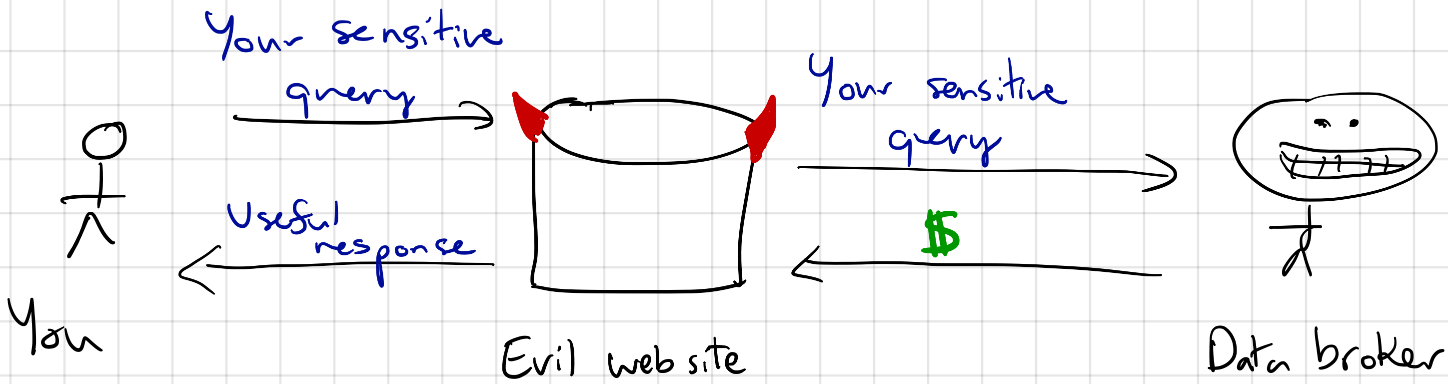
It can leak:

- What website you're visiting
- Your medical condition
- What stocks you're thinking about buying

⋮

Today, the client just sends this sensitive data directly to the server!

In systems today...



Question: "Can you query a database without the database learning your query?"

Trivial answer: "Just download the entire DB."

The DB server doesn't learn your query...

Still, this scheme is somewhat unsatisfying

Let's ask a better question...

Question: "Can you query a database without the database learning your query with communication sublinear in DB size?"

Answer: Unconditionally.....

No.  [CGKS'95]

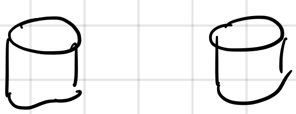
We won't prove this, but it's not too hard to show... See paper for 1-para proof.

What do we do when we get stuck in life?

Option I:

Change the model!
(e.g. ROM)

What if we have two non-colluding copies of DB?



"two-server PIR"

[Can think of $k > 2$ non-colluding servers "k-server PIR"]

Option II

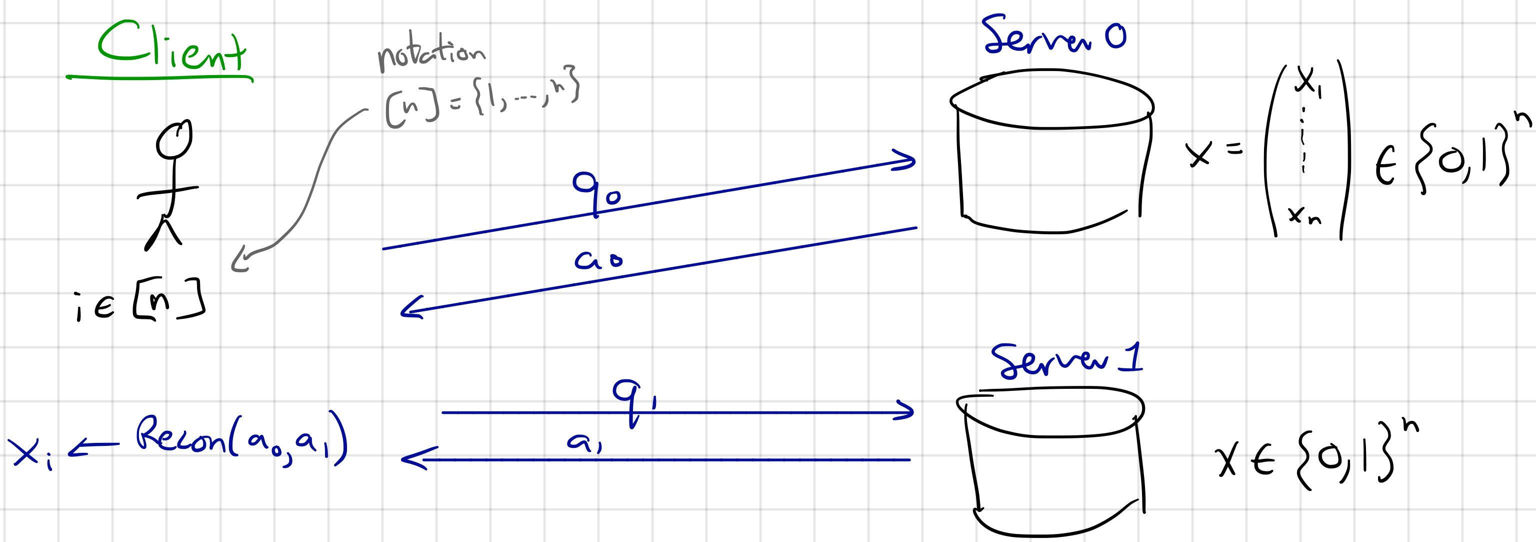
Make assumptions!

Under pretty basic assumptions (DDH, ...), we can build non-trivial single-server PIR

[Kushilevitz & Ostrovsky FOCS'97]

Lets first consider two-server PIR...

Private Information Retrieval (Chor, Goldreich, Kushilevitz, Sudan) FOCS '95



To keep things simple, we'll focus on "one-round" PIR scheme

↳ One message from client to server ("query")
↳ One message from server to client ("response")

Also, we'll think about the DB as holding bits

↳ Can handle longer msgs

Syntax: Three eff algs

$$(q_0, q_1) \leftarrow \text{Query}(i)$$ ... for $i \in [n]$, where n = length of database

$$a \leftarrow \text{Answer}(x, q)$$

$$x_i \leftarrow \text{Reconstruct}(a_0, a_1)$$

Properties

1. Correctness: Client gets the bit it wants

$\forall\, i \in [n] \quad \forall x \in \{0,1\}^n$

$$\Pr\left[x_i = \text{Reconstruct}(a_0, a_1) : \begin{array}{l} (q_0, q_1) \leftarrow \text{Query}(i) \\ a_0 \leftarrow \text{Answer}(x, q_0) \\ a_1 \leftarrow \text{Answer}(x, q_1) \end{array} \right] = 1.$$

2. Security: No single server learns the bit the client wants

$\exists$ eff Sim s.t. $\forall \beta \in \{0,1\}$

$$\{ \text{Sim}(\beta) \} \approx_c \{ q_\beta \mid (q_0, q_1) \leftarrow \text{Query}(i) \}$$

Simulation! So useful!

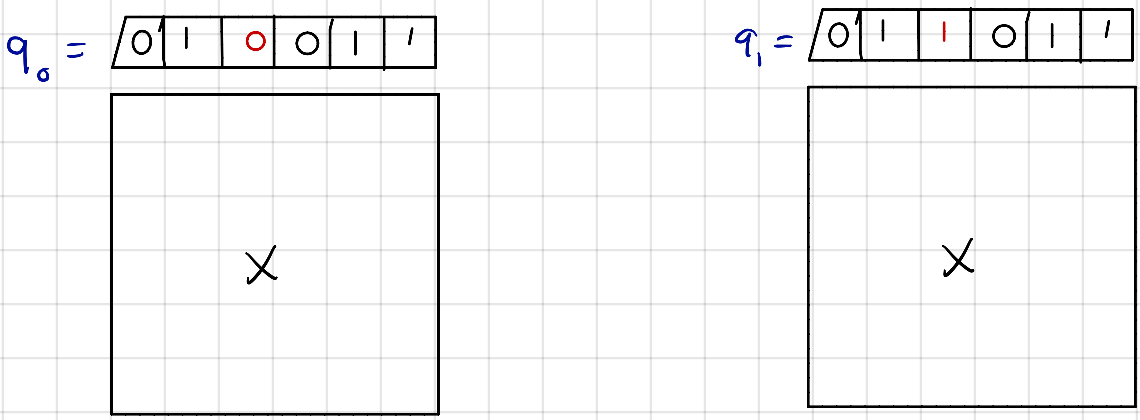
Can also be $\equiv$ for info. theoretic or "perfect" security.

N.B. If both servers collude and share their q's all bets are off!

An $O(\sqrt{n})$-communication PIR scheme.

↑ Already very non-trivial!

View database as a matrix $X \in \mathbb{Z}_2^{\sqrt{n} \times \sqrt{n}}$



Client wants to read X_{ij} $\quad i,j \in [\sqrt{n}]$

Query$(i,j) \to (q_0, q_1)$

Sample two random vectors $q_0, q_1 \in \mathbb{Z}_2^{\sqrt{n}}$
s.t.

$$q_0 + q_1 = (0\,0\,0\,0\,0\,1\,0\,0\,0\,0) \in \mathbb{Z}_2^{\sqrt{n}}$$

↑ jth position

return (q_0, q_1)

Answer$(X, q) \to a$

Output $X \cdot q \in \mathbb{Z}_2^{\sqrt{n}}$.

Reconstruct$(a_0, a_1) \to x_{ij}$

Compute $col_j \leftarrow a_0 + a_1 \in \mathbb{Z}_2^{\sqrt{n}}$
output ith element as x_{ij}.

Why it works.

Correctness:

$$a_0 + a_1 = Xq_0 + Xq_1$$

$$= X(q_0 + q_1)$$

$$= \begin{pmatrix} & & \\ & X & x_j \\ & & \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

jth position

$$= \begin{pmatrix} \\ x_j \\ \\ \end{pmatrix} \in \mathbb{Z}_2^{\sqrt{n}}$$

Then ith component of $a_0 + a_1$ gives X_{ij}.

Security

$\mathrm{Sim}(\beta)$: $\quad q_\beta \xleftarrow{R} \mathbb{Z}_2^{\sqrt{n}}$

output q_β.

Each query is distributed uniformly at random.

What do we know about 2-server PIR?

* Without privacy, total communication $\geq \log n$ bits

* Best known lower bound (impossibility result) says for PIR
  communication $\geq 5 \log n$ bits
  (Wehner & de Wolf '05)

* When I took CS355, best protocol had
  communication $\leq O(n^{1/3})$ (GKKS '98)
  (not too complicated)

* Relatively recent big result
  (Dvir & Gopi '15)
  communication $\leq n^{O(\sqrt{\log\log n / \log n})}$
  (not simple)

Today, we'll see a scheme that achieves
communication $\leq O(n^{1/2})$

$\Rightarrow$ A good open Q: Are there better PIR schemes? $\Leftarrow$

PIR scheme w/ comm complexity $\leq O(\log^2 n)$?

* With computational assumptions (PRGs), have very good
  recent schemes
  communication $\leq O(\lambda \log n)$

  λ: Security Parameter

These results are essentially best possible... but still good open Qs here.

Single Server PIR: $O(\sqrt{n})$ communication

Say you have an additively hom semantically secure enc scheme

$$E(k, m_0) + E(k, m_1) = E(k, m_0 + m_1)$$

↳ Can build from DDH, Quadratic Residuosity, lattices, ...

IDEA: Client sends encrypted query vector, server computes mat-vec product "under encryption"

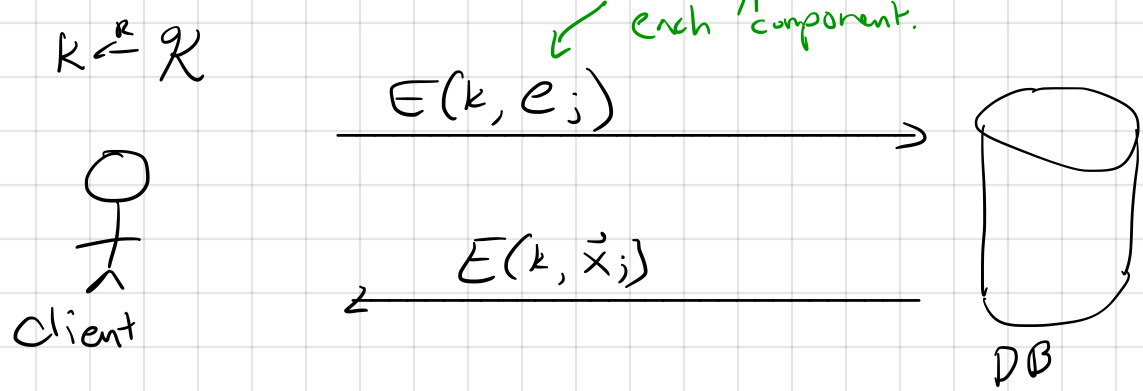
Again write DB as matrix $X \in \mathbb{Z}_2^{\sqrt{n} \times \sqrt{n}}$

Client wants bit $(i, j) \in [\sqrt{n}]^2$.

Notation $e_j = (0000 \cdots 010 \cdots 000)$

jth position

Encryption of each component.



Just requires summing ctxts!

$$\underbrace{X \cdot E(k, e_j)}$$

$$= E(k, X \cdot e_j)$$
$$= E(k, \vec{x}_j)$$

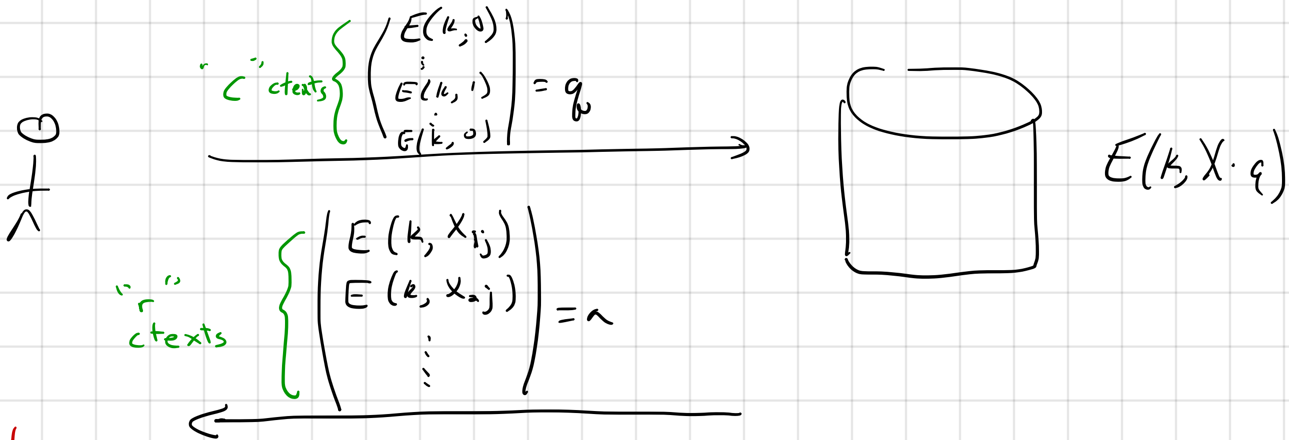
↳ Decrypt to recover $\vec{x}_j$ (jth column of X)

↳ Get bit X_{ij}

Single-Server PIR: Reducing communication (Kushilevitz & Ostrovsky '97)

Lets look more closely at our PIR scheme

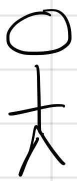
$$X = \begin{array}{cc} & c \\ r & \begin{pmatrix} \quad & \quad & \quad \\ \quad & \quad & \quad \\ \quad & \quad & \quad \end{pmatrix} \end{array}$$



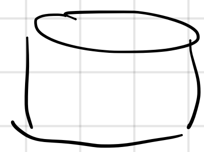
Client discards all but one of responses!

Idea: View answer a as a database. Apply single-server PIR recursively to fetch i-th element of answer!

$$X = \begin{array}{cc} n^{1/3} & \\ \left(\begin{array}{ccc} & & \\ & & \\ & & \end{array} \right) & n^{2/3} \end{array}$$



$$\xrightarrow{E(k, q_0),\ E(k, q_1)}$$



$$\xleftarrow{E(k, a^1)}$$

$q^0 \in \mathbb{Z}_2^{n^{1/3}}$

$q^1 \in \mathbb{Z}_2^{n^{1/3}}$

$$\begin{pmatrix} X \end{pmatrix} \begin{pmatrix} q^0 \end{pmatrix} = \begin{pmatrix} a^0 \end{pmatrix} \Big\} n^{2/3}$$

$$n^{1/3} \Big\{ \underbrace{\begin{pmatrix} a^0 \end{pmatrix}}_{n^{1/3}} \begin{pmatrix} q^1 \end{pmatrix} = \begin{pmatrix} a^1 \end{pmatrix}$$

Total communication is $O(n^{1/3})$!

QR takes $2^{\tilde{\Omega}(n^{1/3})}$ time

Under reasonable assumptions can continue recursion to get complexity $2^{O(\sqrt{\log n \log\log n})}$

Under slightly crazier assumptions (Φ-hiding), can get polylog(n) communication... (See Ostrovsky & Skeith survey).

Extensions

* PIR by keyword
* PIR writing