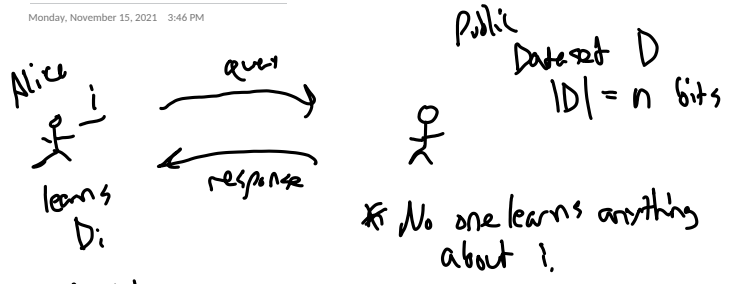
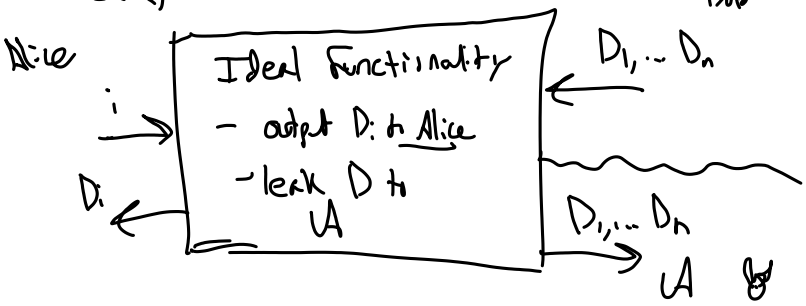
Private Information Retrieval

Monday, November 15, 2021 3:46 PM



* No one learns anything about i.

* Like OT, except no need to hide D_j for $j \neq i$



- Strawman: Send all of D to Alice. Alice selects out D_i. Comm. $O(n)$. Can do better?
- OT is overkill, still $O(n)$
- Generic MPC...
- Rely on expensive preprocessing (e.g. Alice generates garbled circuits ahead of time)

Two PIR approaches
- $O(\sqrt{n})$ cost for Alice
- One based on secret sharing, one on homomorphic encryption.

Basic idea:
- Store D as a matrix $\sqrt{n}$

$$D = \begin{bmatrix} D_{0,0} & D_{0,1} & \cdots & D_{0,\sqrt{n}} \\ D_{1,0} & & & \\ \vdots & & & \end{bmatrix}$$

$\sqrt{n}$ rows, $n^{.75}$ columns, $n^{.25}$ Q

$\cdots \; D_{\sqrt{n},0} \; \cdots \; D_{\sqrt{n},\sqrt{n}}$

- Encode query (i,j) as 1-hot vectors: $\vec{e}_i = 1$

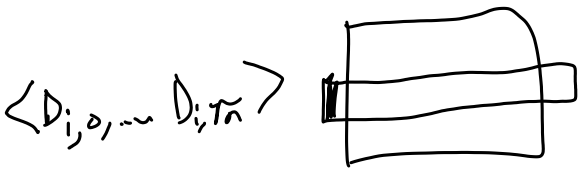
$q_{row} = \vec{e}_i = [0, 0, \ldots 1 \cdots 0]$

$q_{col} \quad \vec{e}_j = [0 0 0, \ldots 1, \ldots 0]$

$\vec{e}_j = 1$

- Server computes $\vec{e}_i \cdot D = \langle D_{i,0}, \ldots D_{i,\sqrt{n}} \rangle$ and sends to client

$\sqrt{n} \times \sqrt{n}$



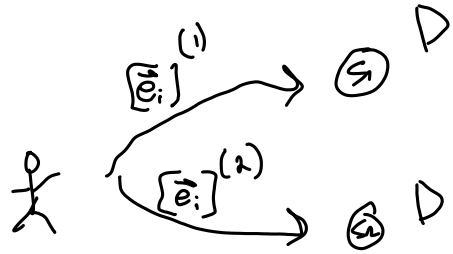
Server cost: $O(n)$

Total Comm: $O(\sqrt{n})$ Client Comp: $O(\sqrt{n})$

IT-PIR

↑ Information theoretic

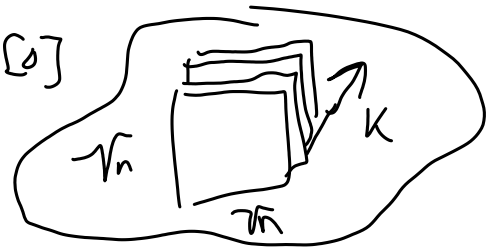
- 2 servers, each have D.



- Clients sends secret shared qkeys

$[\vec{e}_i] = [0], [0], \ldots [1], \ldots [0]$

$[\vec{e}_i]^{(1)} \oplus [\vec{e}_i]^{(2)} = \vec{e}_i$



$D_1 \cdot [\vec{e}_i]$

$D_2 \cdot [\vec{e}_i]$

- Servers compute:

$D \cdot [\vec{e}_i] = [D \cdot \vec{e}_i]$

i

1

- Client receives and reconstructs

$$[D \cdot \vec{e}_i] \Rightarrow D \cdot \vec{e}_i \text{ and selects column.}$$

$$\langle D_{i,0} \dots D_{i,n} \rangle$$

↑ selects jth element

$$(D \cdot \vec{e}_i) \cdot \vec{e}_j^T = D_{i,j}$$

- Using homomorphic enc. e.g. Elgamal

C-PIR

↑ computational

- Encrypt each element query is (q_i, q_i)

$Enc(\vec{e}_i)$

$B = g^b$ is a fixed public key (Alice has b)

$$Enc(\vec{e}_i) = \left\{ \begin{array}{l} A_i = g^{a_i} \text{ for } a_i \xleftarrow{\$} \mathbb{Z}_p \\ (A_i, B^{a_i} \cdot g) \text{ if } i = q_i \\ (A_i, B^{a_i}) \text{ otherwise} \end{array} \right\} i \in [1 \dots n]$$

$$q = [(A_1, C_1), (A_2, C_2) \dots (A_n, C_n)]$$

- Server computes:

$[res_1, \ldots res_m]$

where each element is one column

$$res_j = D_{1,j} \cdot (A_1, C_1) + D_{2,j} \cdot (A_2, C_2) + \ldots$$

$$= \left(\prod_i A_i^{D_{i,j}}, \prod_i C_i^{D_{i,j}} \right)$$

$$\prod_i (g^{a_i})^{D_{i,j}} \qquad \prod_i \left(B^{a_i} \cdot g^{\vec{e}_i} \right)^{D_{i,j}}$$

$$\left(g^{\sum a_i \cdot D_{i,j}} \right), \qquad g^{b \sum a_i \cdot D_{i,j}} \cdot g^{\sum \vec{e}_i \cdot D_{i,j}}$$

$$\left(g^{b} \right)^{\sum a_i \cdot D_{i,j}} \cdot g^{D_{e_i,j}}$$

A^* $\qquad C^*$

Alice decrypts $\qquad (A^*)^{b})^{-1} \cdot C^* = g^{D_{e_i,j}}$