

## Lecture 7: Preprocessing PIR Lower Bounds

Scribe: William Seo

February 12, 2024

### 1 Introduction

This is our final lecture on private information retrieval. We will prove a lower bound about the client space and server computation trade-off for preprocessing PIR schemes. For the proof, we will use techniques from complexity theory literature.

#### 1.1 PIANO

PIANO[ZPSZ23] is a preprocessing 1-server PIR scheme covered in lecture 5. It uses the client-specific preprocessing model, meaning that each client has a subscription phase with the server, during which it will perform preprocessing. The scheme has the following specs:

- Client Space:  $\tilde{O}(\sqrt{n})$
- Communication BW per Query:  $O(\sqrt{n})$
- Server/Client Compute per Query:  $\tilde{O}(\sqrt{n})$

Consider the server computation and client space trade-off. For PIANO, this trade-off [Client Space:  $\tilde{O}(\sqrt{n})$ , Compute per Query:  $\tilde{O}(\sqrt{n})$ ] is nearly optimal up to poly-log factors. This is because of the following lower bound.

### 2 Space/Computation Lower Bound

**Claim 1.** *Given a 1-Server Preprocessing PIR, define variables as follows:*

- $S$  : Client Space
- $T$  : Server Compute per Query

*Then,*

$$(S + 1)(T + 1) \geq N$$

This LB holds even for a scheme that supports 1 query, regardless of server space, BW, and amount of preprocessing work. Before proceeding with the proof, we first have to discuss another slightly different problem called Yao's Box Problem [Yao90].

#### 2.1 Yao's Box Problem

We have a server with  $N$  boxes, each covering a bit. Note that this is not a PIR scheme, because it does not provide any privacy guarantees.

### 2.1.1 Preprocessing Phase

Client and server can open all bits, and run arbitrary and unbounded computations.

**Constraint:** At the end of the preprocessing, client can only store  $S$  bits of information.

### 2.1.2 Query

Client wants to know the  $i$ -th bit. We allow the client and server to have unbounded communication and work.

**Constraint:** Server can open at most  $T$  boxes and it cannot open box  $i$ .

### 2.1.3 Upper Bound

1. Divide the  $N$  boxes into  $\sqrt{N}$  segments.
2. Preprocessing Phase: Have the client store the parity of each segment.
3. Query Phase ( $i$ ): Server opens every box in  $i$ 's segment except  $i$  and sends the parity back to the client. As the client knows the parity of each segment, it can easily reconstruct the value of bit  $i$ .

Here,  $S = \sqrt{N}$  and  $T = \sqrt{N} - 1$

### 2.1.4 Lower Bound

**Theorem 2.**  $S(T + 1) \geq N$

*Proof.* We will be using an encoding type argument. Consider the following experiment.

1. Run preprocessing
2. Define an empty set called  $known = \{\}$ . In each step  $i$ , the client finds the smallest  $q_i \notin known$ , queries  $q_i$ .  
With this step,  $known \leftarrow known \cup \{q_i\} \cup$  all boxes opened during query
3. If  $known \neq [N]$ , break. Otherwise, repeat step 2 again.

Let the “client’s hint” denote whatever information that the client stores after the preprocessing phase. The hint is at most  $S$  bits long.

Define the encoding  $enc$  for this process as follows:

$$enc = \text{client's hint} + \text{all “newly opened” boxes in all queries}$$

We write “newly opened” because we do not want to include values that have already been recorded in the encoding.

By Shannon’s theorem, we will show that  $length(enc) \geq N$ .

Let  $t_1, t_2, \dots, t_k$  be the number of newly opened boxes at each step  $i \in [k]$ .

Note here that we have the power of **plus one**; if I open  $t$  boxes, I end up learning  $t + 1$  new bits. This is because I also learn the value of the query without opening its box.

Thus, our  $known$  set increments with the following pattern.

- We newly open  $t_1$  boxes:  $known$  increments by  $t_1 + 1$
- We newly open  $t_2$  boxes:  $known$  increments by  $t_2 + 1$
- and so on...

Thus, we have that  $\sum_{i=1}^k (t_i + 1) \geq N$ . Let  $t = \frac{\sum_{i=1}^k t_i}{k}$ . Note that  $t$  is the average number of boxes opened on each iteration so it must be upper bounded by  $T$ , the maximum number of boxes that can be opened in an iteration. Since we repeat the steps until  $known = [N]$ , we have that

$$\sum_{i=1}^k (t_i + 1) \geq N \implies (t + 1)k \geq N \implies k \geq \frac{N}{t + 1} \quad (1)$$

The encoding size is  $S + \sum_{i=1}^k t_k = S + tk$ . By Shannon's we have that  $S + tk \geq N$ . By (1), we have that

$$\begin{aligned} S + tk &\geq N \\ \implies S + t \frac{N}{t + 1} &\geq N && \text{(By (1))} \\ \implies S(t + 1) + tN &\geq N(t + 1) \\ \implies S(t + 1) &\geq N \\ \implies S(T + 1) &\geq N && \text{(As } t \leq T) \end{aligned}$$

□

## 2.2 PIR Lower Bound

Using our knowledge of Yao's Box Problem, let's now try to prove the PIR lower bound.

**Theorem 3.** *Suppose we have a 1 server preprocessing PIR with perfect correctness and  $\text{negl}(n)$  privacy loss with client space  $S$  and server computation  $T$  per query. Then,*

$$(S + 1)(T + 1) \geq N \quad [CGK20]$$

This lower bound holds even for computationally private schemes. It holds even for a single query, regardless of bandwidth, server space, even when preprocessing can be unbounded.

**Restriction:** The server stores the original database and nothing else. That is, it does not store any encoding of the database.

For the proof, we will show that a solution to the PIR problem can be used to construct an algorithm to solve a probabilistic version of Yao's Box Problem.

Suppose we have a working PIR scheme. Now, we will construct a solution to probabilistic Yao's Box Problem as follows:

- Client's Hint: PIR's hint
- Query for  $i \in [N]$ : Run PIR for query  $i$ . If server looks at  $DB[i]$ , then output "error".

We want to show the following:  $\text{PIRExpt}: i \xleftarrow{\$} [N]$ , PIR preprocessing, PIR query on  $i$ .

$$p = \mathbb{P}[\text{PIRExpt opens } i] \leq \frac{T}{N} + \text{negl}(N)$$

That is, we want to show that if we run a PIR query with the a random index  $i$ , the probability we open that index is small.

For a fixed  $i$ , define the following probability  $p_i = \mathbb{P}[\text{PIR on } i \text{ looks at } i]$ . Then,

$$p = \frac{1}{N} \sum_i p_i$$

Assume for the sake of contradiction that  $p > \frac{T}{N} + \mu$ , where  $\mu$  is non-negligible.

Let  $p_{ji} = \mathbb{P}[\text{PIR on } j \text{ opens } i]$ . Since our scheme is private, the different indices should be computationally indistinguishable. Thus, this probability should be equally distributed. As a result,

$$p_{ji} = \mathbb{P}[\text{PIR on } j \text{ opens } i] \geq p_i - \text{negl}(N)$$

$$\begin{aligned} E[\text{server work for PIR on } j] &\geq \sum_{i=1}^N p_{ji} \\ &\geq \sum_{i=1}^N (p_i - \text{negl}(N)) \\ &= Np - \text{negl}(N) && \text{(By def. of } p) \\ &> T + \mu N - \text{negl}(N) && \text{(By our AFTSOC assumption)} \end{aligned}$$

Thus, we have a contradiction because the expected number of locations the server needs to look at is strictly greater than  $T$ . Thus, we have shown that

$$\mathbb{P}[\text{PIRExpt opens } i] \leq \frac{T}{N} + \text{negl}(N)$$

Shifting our focus back to Yao's Box problem with probabilistic correctness on random index, the probabilistic correctness is

$$\mathbb{P}[i \xleftarrow{\$} \text{correct for } i] \geq 1 - \frac{T}{N} - \text{negl}(N)$$

Randomness comes from two parts: the preprocessing part (the client's hint), and the query part. With this in mind, we will be using an augmented version of the encoding type argument in 2.1.4.

Consider the following experiment.

1. Run preprocessing, and choose a "reasonably good hint." Initially, let the encoding be just the hint. We will add to this encoding as we go forward.
2. Define an empty set called  $known = \{\}$
3. In each step  $i$ , find the smallest  $q_i \notin known$ .
  - If  $\exists$  online coins such that query  $q_i$  will give the correct answer, choose the lexicographically smallest coin. Execute query.

$$known \leftarrow \cup \{q_i\} \cup \{\text{all newly opened}\}$$

Add "newly opened" to encoding.

- Else add  $q_i$ -th bit to the encoding.

4. Repeat until  $known = [N]$ .

A hint is bad for  $i \in [N]$  if  $\mathbb{P}[\text{query } i \text{ correct} | \text{hint}] = 0$ . That is, there does not exist an online coin such that the query is correct.

**Claim 4.**  $\exists$  hint that's bad for at most  $T + 1$  location.

*Proof.* If all hints are bad for more than  $T + 1$  locations, then

$$\mathbb{P}[i \in [N], \text{correct on } i] < 1 - \frac{T + 1}{N}$$

We have a contradiction, because it disagrees with our probabilistic correctness result above.  $\square$

Finally, we can reason about the encoding length.

Suppose the worst case where the hint is bad for exactly  $T + 1$  locations. Let  $b_{bad}$  be the encoding of the “newly opened” boxes in the bad queries, and  $b_{good}$  be the encoding of the “newly opened” boxes in the good queries.

- $\text{len}(b_{bad}) = T + 1$ , as each bad iteration adds one bit, and we have  $T + 1$  iterations.
- To find  $\text{len}(b_{good})$ , we repeat the argument from 2.1.4. Let  $t_1, t_2, \dots, t_k$  be the number of newly opened boxes at each good step  $i \in [k]$ .

By the “plus one” argument, we have that  $\sum_{i=1}^k (t_i + 1) \geq N - (T + 1)$ . We subtract  $T + 1$  here because those indices have been handled by the bad iterations.

Let  $t = \frac{\sum_{i=1}^k t_i}{k}$ . Then as before, we have  $k \geq \frac{N}{t+1}$

$$\begin{aligned} \sum_{i=1}^k (t_i + 1) &\geq N - (T + 1) \\ \implies tk + k &\geq N - T - 1 \\ \implies k &\geq \frac{N - T - 1}{t + 1} \end{aligned}$$

Thus,  $\text{len}(b_{good}) = tk \geq t \frac{N - T - 1}{t + 1}$

$$\begin{aligned} \text{len}(enc) &= S + \text{len}(b_{good}) + \text{len}(b_{bad}) \geq N \\ \implies S + t \frac{N - T - 1}{t + 1} + T + 1 &\geq N \\ \implies S(t + 1) + (Nt - Tt - t) + Tt + T + t + 1 &\geq Nt + N \\ \implies S(t + 1) + T + 1 &\geq N \\ \implies S(T + 1) + T + 1 &\geq N && (\text{As } t \text{ upper bounded by } T) \\ \implies (S + 1)(T + 1) &\geq N \end{aligned}$$

$\square$

## References

- [CGK20] Henry Corrigan-Gibbs and Dmitry Kogan. Private information retrieval with sublinear online time. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 44–75, Cham, 2020. Springer International Publishing.
- [Yao90] A. C.-C. Yao. Coherent functions and program checkers. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, STOC '90*, page 84–94, New York, NY, USA, 1990. Association for Computing Machinery.
- [ZPSZ23] Mingxun Zhou, Andrew Park, Elaine Shi, and Wenting Zheng. Piano: Extremely simple, single-server pir with sublinear server computation. Cryptology ePrint Archive, Paper 2023/452, 2023. <https://eprint.iacr.org/2023/452>.