Cryptography Meets Algorithms (15893) Lecture Notes

Lecture 7: Preprocessing PIR Lower Bounds

Scribe: William Seo

February 28, 2024

In this lecture, we will prove a lower bound about the client space and server computation tradeoff for preprocessing PIR schemes. We will borrow techniques for proving time-space tradeoff from the complexity theory literature.

Specifically, we consider a 1-server preprocessing PIR scheme with the following syntax:

- *Preprocessing algorithm.* Suppose there is a (possibly randomized and unbounded) preprocessing function denoted $\mathsf{Prep} : \{0,1\}^n \to \{0,1\}^S$ that takes in an n-bit database $\mathsf{DB} \in \{0,1\}^n$ as input, and outputs an S-bit hint string denoted h.
- *A single query.* The client and the server perform a (possibly randomized) query protocol. The client takes in h and some index $i \in [n]$ as input, and the server takes the database $\mathsf{DB} \in \{0,1\}^n$ as input. To answer the query, the server is allowed to read at most T locations of the database.

In other words, we do not place any restriction on the amount of work performed during preprocessing. The only constraints are that at the end of the preprocessing: 1) the client is allowed to store only the hint h; and 2) the server is allowed to store only the original database DB and no extra information.

We assume perfect correctness, i.e., for any $\mathsf{DB} \in \{0,1\}^n$, any query $i \in [n]$, correctness holds with probability 1. Let $\mathsf{view}_S(\mathsf{DB}, i)$ denote the server's view during the query phase when we run the PIR scheme (i.e., preprocessing followed by the query protocol) over inputs DB and i. For security, we require that for any $\mathsf{DB} \in \{0,1\}^n$, any $i, j \in [n]$, $\mathsf{view}_S(\mathsf{DB}, i) \approx \mathsf{view}_S(\mathsf{DB}, j)$ where $\approx$ means computational indistinguishability.

For such a preprocessing PIR scheme, we will prove a tradeoff between S and T as stated in the following theorem:

Theorem 1 (Time-space tradeoff for preprocessing PIR [CGK20])**.** *Given a 1-server preprocessing PIR, let S be the client space, and let T be the server compuation per query. Then, $(S+1)(T+1) \geq N$.*

Again, note that the lower bound holds even when we allow the preprocessing to be unbounded, even when there is only a single query after the preprocessing, and even when the query phase may have arbitrarily many rounds of interaction.

Piano. Recall that in an earlier lecture, we covered Piano [ZPSZ23], a preprocessing 1-server PIR scheme. Piano enjoys the following performance bounds:

- Client space: $\widetilde{O}(\sqrt{n})$ where $\widetilde{O}(\cdot)$ hides a(n arbitrarily small) superlogarithmic function.

- Communication per query: $O(\sqrt{n})$
- Server computation per query: $O(\sqrt{n})$

We can see that Piano achieves optimal client space and server computation tradeoff (up to polylogarithmic factors) in light of Theorem 1.

1 Warmup: Yao's Box Problem

Before proving Theorem 1, we first prove a time-space tradeoff for a classical problem called the Yao's box problem [Yao90], and we shall see why Yao's box problem is closely related to preprocessing PIR.

We have a server with n boxes, each covering a bit. Henceforth we use $\mathsf{DB} := (\mathsf{DB}[1], \ldots, \mathsf{DB}[n])$ to denote the n bits. Consider the following game:

- *Preprocessing.* During a preprocessing phase, the server and client can perform an unbounded amount of computation. At the end of preprocessing, the client obtains an S-bit hint h; the server does not store any extra information besides DB itself where each bit is covered by some box.
- *Query.* The client wants to find out $\mathsf{DB}[i]$ for some index $i \in [n]$. The client and server now engage in some protocol at the end of which the client outputs an answer β. The query protocol must satisfy two requirements: 1) the server is *not allowed to open box i* during the protocol; and 2) the server can open at most T boxes.

We allow the preprocessing algorithm and query protocol to be possibly randomized. We require that the protocol to have *perfect correctness*, i.e., for any $\mathsf{DB} \in \{0,1\}^n$, any index $i \in [n]$, after performing the preprocessing and the query protocol for index i, the client's output $\beta = \mathsf{DB}[i]$ with probability 1.

Note that unlike PIR, Yao's box problem does not have any privacy requirement. In particular, it is perfectly ok for the query index i to be leaked to the server.

In the above formulation, S denotes the client space at the end of preprocessing, and T can be viewed as a lower bound on the server's running time during the query phase. We care about characterizing the tradeoff between S and T, that is, the client-space and server-time tradeoff.

Yao [Yao90] proved the following theorem:

Theorem 2 (Yao's box problem)**.** *For any protocol that solves Yao's box problem, it must be that $S \cdot (T+1) \geq n$, where S is the client space and T is the maximum number of boxes opened during the query.*

1.1 Upper Bound

Before proving Theorem 2, let us first see a simple upper bound that can match the $S \cdot (T+1) = n$ tradeoff. For simplicity, assume n is a perfect square.

1. Divide the n boxes into $\sqrt{n}$ segments each of size $\sqrt{n}$.
2. Preprocessing: the client stores the parity of each segment, denoted $p_1, \ldots, p_{\sqrt{n}}$ respectively.
3. Query for index $i \in [n]$: suppose i belongs to the j-th segment. The server opens every other box except i in the j-th segment, and responds with the parity (denoted p^*) of all opened bits. The client reconstructs the answer as $p^* \oplus p_j$.

In this construction, $S = \sqrt{n}$ and $T = \sqrt{n} - 1$.

You might have observed that this upper bound for Yao's box problem is reminiscient of the Piano PIR scheme [ZPSZ23].

1.2 Lower Bound

We now prove Theorem 2. The intuition of the proof is if we can obtain a too-good-to-be-true tradeoff between S and T, then we can construct an encoding of $\mathsf{DB} \in \{0,1\}^n$ whose length is less than n, which violates Shannon's fundamental theorem of information theory. Specifically, we can obtain some compression in the encoding by leveraging the following fact: every time we open T boxes not including i, we learn not just the values under the T opened boxes but also an additional bit, namely, the i-th bit — this effectively gains us one bit of advantage.

The formal proof works as below.

Proof of Theorem 2. Suppose we have some protocol for solving Yao's box problem with parameters S and T. We fix all the coins in the protocol denoted coins, and we will give coins as input to both the encoder and decoder. As long as DB is randomly sampled independently of coins, we can use Shannon's theorem to argue that the encoding length must be at least n.

Encoding. The encoding algorithm is given coins and DB as input, and constructs an encoding of DB as follows.

1. Seed the protocol (for solving Yao's box problem) with coins, and perform preprocessing such that the client obtains an S-bit hint h.
2. Initially, let $\mathsf{known} = \{\}$. Repeat the following: in each time step i, the client
   - finds the smallest index $q_i \notin \mathsf{known}$, and runs the query protocol for index q_i;
   - let $\mathsf{known} \leftarrow \mathsf{known} \cup \{q_i\} \cup \{\text{all boxes opened during this query}\}$
   - if $\mathsf{known} = [n]$, break.

Define the *newly* opened boxes in time step i as follows: all boxes opened in time step i that are not in the known set yet. Output an encoding containing the following terms:

1. The client's hint h;
2. Let V_i be the *newly* opened values (in the order they are opened) for time step i. Include $V_1, V_2, \ldots, V_k$ in the encoding, where k is the total number of time steps.

Decoding. The decoding algorithm is given coins and an encoding C as input, and outputs a decoded string as follows.

1. Seed the protocol (for solving Yao's box problem) with coins. Treat the first S bits of C as the client's hint, and in the steps below, consume the rest of C bit by bit in a streaming manner.
2. Initially, let $\mathsf{known} = \emptyset$. Every time step i,
   - Let q_i be the smallest index not in known;

- The client performs a query for q_i with the server. Whenever the server needs to open some box: if the box was already opened in some earlier time step, use the same opened value as previously learned; else if the box has not been opened, treat the next bit in C as the opened value.
- For all newly opened boxes, record the opened values. Additionally, reconstruct the value at queried index q_i.
- Let $\mathsf{known} \leftarrow \mathsf{known} \cup \{q_i\} \cup \{\text{all boxes opened during this query}\}$; if $\mathsf{known} = [n]$, break.

3. At the end of the protocol, $\mathsf{known} = [n]$, i.e., the values for n indices have been discovered. Output this reconstructed string.

□

Correctness of the decoding is easy to verify: the decoding algorithm opens the same sequence of boxes as the encoding algorithm, and for every newly opened box, decoding observes the correct value from C.

Let $t_1, \ldots, t_k$ be the number of *newly* opened boxes in each of the k time steps. Thus, the number of elements added to known in time step i is $t_i + 1$. We also know that $t_i \leq T$ for all $i \in [k]$, and regardless of the choice of DB. Henceforth, let $t = \frac{1}{k}\sum_{i\in[k]} t_i$. Since the the encoding/decoding algorithm stops as soon as $\mathsf{known} = [n]$, it means that $\sum_{i\in[k]}(t_i + 1) = (t+1)\cdot k = n$. Thus, $k = n/(t+1)$.

The length of the encoding is $S + \sum_{i\in[k]} t_i = S + t \cdot k$. Note that t and k are random variables that depend on the choice of DB. By Shannon's theorem, we know that

$$\mathop{\mathbf{E}}_{\mathsf{DB} \xleftarrow{\$} \{0,1\}^n}[S + t \cdot k] = \mathop{\mathbf{E}}_{\mathsf{DB} \xleftarrow{\$} \{0,1\}^n}\left[S + t \cdot \frac{n}{t+1}\right] \geq n$$

Observe that for any DB, $\frac{t}{t+1} \leq \frac{T}{T+1}$. Thus, we have that

$$S + n \cdot \frac{T}{T+1} \geq n$$

which directly implies that $S(T+1) \geq n$.

2 Time-Space Tradeoff for Preprocessing PIR

We will next prove a time-space tradeoff for preprocessing PIR. Intuitively, the privacy requirement of PIR implies that during a query for some index $i \in [n]$, the probability that the server actually visits position i is rather small. This allows us to rely on the space-time tradeoff for Yao's box problem to prove a PIR lower bound.

Theorem 3 (Time-space tradeoff for preprocessing PIR [CGK20])**.** *Suppose we have a 1-server prepossessing PIR with perfect correctness and* $\mathsf{negl}(n)$ *privacy loss where* $\mathsf{negl}(\cdot)$ *denotes a negligible function. Let S be the client space and let T be the server computation per query. Then,*

$$S \cdot T \geq \Omega(n)$$

This lower bound holds even for computationally private schemes. It holds even for a single query, regardless of bandwidth, number of rounds, and preprocessing cost. However, we require that the server store only the original database during the query phase and no extra information.

To prove Theorem 3, we actually need a version of Yao's box problem that allows probalistic correctness, as stated below.

Yao's box problem with probabilistic correctness. We say that some protocol Π solves Yao's box problem with probablistic correctness $\delta(n)$, iff for any $\mathsf{DB} \in \{0,1\}^n$, it holds that

$$\Pr\left[i \xleftarrow{\$} [n], \text{run } \Pi \text{ with } \mathsf{DB} \text{ and } i : \text{output is correct}\right] \geq \delta(n)$$

We can extend the proof to Yao's box problem to allow probabilistic correctness, as stated in the following theorem:

Theorem 4 (Yao's box problem with probabilistic correctness)**.** *Given a possibly randomized protocol that solves Yao's box problem with correctness probability 0.8 or higher, let S be the client space and let T be the server time during the query. It must be that $S \cdot T \geq \Omega(n)$.*

Below, we will first assume that Theorem 4 is true, and prove a lower bound for preprocessing PIR in Section 2.1. Next, in Section 2.2, we show how to modify the earlier proof for Yao's box problem to get the probalistic version (i.e., Theorem 4) that we need.

2.1 Preprocessing PIR $\Longrightarrow$ Yao

Given a preprocessing PIR scheme, we will construct a solution to Yao's box problem (with probablistic correctness) as follows:

- Client's hint: PIR's hint.
- Query for $i \in [n]$: Run PIR for query i. If server is about to look at $\mathsf{DB}[i]$, then stop and simply output "error"; else, output the PIR's reconstructed answer.

Clearly, if the PIR scheme has client space S and server time T, then the resulting protocol (for solving Yao's box problem) also enjoys client space S and server time as most T, where server time is measured in terms of the number of locations visited by the server. Moreover, we claim that the protocol satisfies the following probabilistic correctness:

Claim 5. *Suppose the PIR scheme has perfect correctness and negligible privacy loss. Then, the above protocol solves Yao's box problem with correctness probability $1 - \frac{T}{n} - \mathsf{negl}(n)$ for a random query.*

Proof. Henceforth, fix an arbitrary DB. Let $\mathsf{PIRExpt}^j$ denote the random experiment that runs the PIR scheme over database DB and query j. let

$$p_{j,i} := \Pr\left[\text{ server visits } i \text{ in } \mathsf{PIRExpt}^j\right]$$

It suffices to prove the following:

$$\Pr\left[i \xleftarrow{\$} [n], \text{ server visits } i \text{ in } \mathsf{PIRExpt}^i\right] = \frac{1}{n} \cdot \sum_{i \in [n]} p_{i,i} \leq \frac{T}{n} + \mathsf{negl}(n) \tag{1}$$

We now prove the above inequality. The privacy requirement of PIR implies that for any $j, i \in [n]$, $|p_{j,i} - p_{i,i}| \leq \mathsf{negl}'(n)$. Since the server can perform at most T amount of work, we have that for any $j \in [n]$, $\sum_{i \in [n]} p_{j,i} \leq T$. Combining the above facts, we have that

$$T \geq \sum_{i \in [n]} p_{j,i} \geq \sum_{i \in [n]} (p_{i,i} - \mathsf{negl}'(n))$$

which gives us Equation (1). □

Combining the above reduction and Theorem 4, we immediately get the claimed space-time tradeoff for 1-server preprocessing PIR.

2.2 Proof of Theorem 4

We now prove Theorem 4. We present a variant of the proof[1] described by De, Trevisan, and Tulsiani [DTT10]. The proof requires somewhat more technical calculations in comparison with the earlier proof in Section 1; however, the high-level intuition is still the same as before.

Fact 1 (Extension of Shannon's theorem for codes with probabilistic correctness.)**.** *Suppose there is a randomized encoding procedure* $\mathsf{Enc} : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ *and decoding procedure* $\mathsf{Dec} : \{0,1\}^m \times \{0,1\}^r \to \{0,1\}^n$ *such that*

$$\Pr\left[r \xleftarrow{\$} \{0,1\}^r, \mathsf{DB} \xleftarrow{\$} \{0,1\}^n : \mathsf{Dec}\left(\mathsf{Enc}(\mathsf{DB}, r), r\right) = \mathsf{DB}\right] \geq \delta$$

Then, $m \geq n - \log(\frac{1}{\delta})$.

Proof. There must exist some r such that $\Pr\left[\mathsf{DB} \xleftarrow{\$} \{0,1\}^n : \mathsf{Dec}\left(\mathsf{Enc}(\mathsf{DB}, r), r\right) = \mathsf{DB}\right] \geq \delta$. In other words, there is an r such that at least δ fraction of the databases can decrypt correctly. This means that $\mathsf{Enc}(\mathsf{DB}, r)$ must attain at least $\delta \cdot 2^n$ values as DB varies over $\{0,1\}^n$. As the total number of values $\mathsf{Enc}(\cdot, r)$ can attain is at most 2^m, we have that $2^m \geq \delta \cdot 2^n$ which gives us the desired inequality. □

Fact 2. *Let* $\epsilon \in (0, 1/2)$ *be some constant. Given a set of* M *items, pick a subset of size at most* $(1/2 - \epsilon)M$*. The subset can be encoded using* $M \cdot (1 - 2.86\epsilon^2)$ *bits.*

Proof. The number of ways to choose a subset of size at most $(1/2 - \epsilon)M$ is

$$\sum_{i=0}^{(1/2-\epsilon)M} \binom{M}{i} \leq 2^{\mathbb{H}(1/2-\epsilon)M}$$

where $\mathbb{H}$ is the binary entropy function, and the inequality is due to Theorem 3.1 of Galvin [Gal14] — you can also try proving it as exercise yourself! We can get that $\log_2(2^{\mathbb{H}(1/2-\epsilon)M}) = \mathbb{H}(1/2-\epsilon) \cdot M \leq M \cdot (1 - 2.86\epsilon^2)$ by Taylor expansion.

□

Lemma 6. *Consider a* **deterministic** *protocol* Π *for solving Yao's box problem with client space* S *and server time* T*, and suppose* $T \in [2, o(n)]$*. Let* $\mathsf{DB} \in \{0,1\}^n$ *be some fixed string such that* Π *gives a correct answer on at least* $\frac{3}{4}$ *fraction of the indices. Then,* $\mathsf{Enc} : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ *and decoding procedure* $\mathsf{Dec} : \{0,1\}^m \times \{0,1\}^r \to \{0,1\}^n$ *such that*

$$\Pr\left[r \xleftarrow{\$} \{0,1\}^r : \mathsf{Dec}\left(\mathsf{Enc}(\mathsf{DB}, r), r\right) = \mathsf{DB}\right] \geq 0.04$$

Further, the encoding length $m \leq n - \Theta(\frac{n}{T}) + S$.

Proof. Suppose we sample each index in $[n]$ with probability $1/10T$ into some random set R. For some $i \in R$, if running the protocol Π on i does not end up visiting any position in R, then we say that i is *good*. Let $G \subseteq R$ be the subset of R that are good. We partition $G = G_0 \cup G_1$ into two disjoint subsets, where G_0 includes every index $i \in G$ such that running Π on i gives a correct answer, and G_1 includes every index $i \in G$ such that running Π on i gives an incorrect answer.

[1] The proof we present below fixes a bug in their proof. Specifically, the way they take Chernoff bound in the proof of Lemma 8.4 is incorrect since the underlying random variables are positively correlated. Thanks to Ashrujit Ghoshal for helping me fix the bug.

Fact 3. $\mathbf{E}[|R|] = n/10T$. *Further, by Chernoff bound,* $\Pr[|R| \le 0.09n/T] \ge \exp(-\Omega(n/T))$, *and* $\Pr[|R| \ge 0.11n/T] \ge \exp(-\Omega(n/T))$.

Claim 7. $\Pr[|G| \ge 0.026n/T] \ge 0.74$ *for sufficiently large* n.

Proof. For a fixed $i \in [n]$, the probability that it lands in R and is bad is

$$\frac{1}{10T} \cdot \left(1 - \left(1 - \frac{1}{10T}\right)^T\right) \le \frac{1}{10T} \cdot \left(1 - \frac{1}{(2e)^{0.1}}\right) \le 0.016/T$$

Therefore, the expected number of indices that land in R and are bad is at most $0.016n/T$. By Markov Inequality, with probability at most $1/4$, the number of indices that land in R and are bad exceeds $0.064n/T$.

Due to Fact 3, and taking a union bound, with probability at least $1 - \exp(-\Omega(n/T)) - 1/4$, $|G| \ge 0.09n/T - 0.064n/T = 0.026n/T$. □

Claim 8. *For sufficiently large* n, *with probability at least* 0.3, $|G_0| - |G_1| \ge 0.01 \cdot n/T$.

Proof. For a fixed $i \in [n]$, the probability that i lands in R and is good is

$$\frac{1}{10T} \cdot \left(1 - \frac{1}{10T}\right)^T \in [0.09/T, 0.091/T]$$

Therefore, we have that

$$\mathbf{E}[|G_0| - |G_1|] \ge 0.09/T \cdot \frac{3n}{4} - 0.091/T \cdot \frac{n}{4} \ge 0.043n/T$$

Therefore, it must be that with probability $p \ge 0.3$, $|G_0| - |G_1| \ge 0.01 \cdot n/T$, since otherwise,

$$\mathbf{E}[|G_0| - |G_1|] \le (p - \exp(-\Omega(n/T))) \cdot 0.11n/T + (1-p) \cdot 0.01n/T + \exp(-\Omega(n/T)) \cdot n < 0.043n/T$$

which contradicts the above. □

Combining Claim 7 and Claim 8 and taking union bound, we know that with probability at least $1 - (1 - 0.74) - (1 - 0.3) = 0.04$ over the choice of R, the following two good events hold:

1. $|G_0| - |G_1| \ge 0.01n/T$, and
2. $|G| \ge 0.026n/T$.

Below, we construct the encoding and decoding algorithm. We sample some random coins that are shared by the encoding and decoding algorithm, and these random coins are used to sample the set R.

Encoding. If the choice of R is such that both of the above good events hold, then output the following encoding (otherwise output $\perp$):

1. The client's hint.
2. Values for the indices in $[n] \backslash R$.
3. Values for the indices in $R \backslash G$.
4. The set G_1.

The client hint is at most S bits. The second part requires $n - |R|$ bits, the third part requires $|R| - |G|$ bits, and the fourth part requires $|G| - c|G|$ bits for some constant $c \in (0, 1)$ by Fact 2 and due to the aforementioned two good events. Therefore, the total encoding size is at most $S + n - |R| + |R| - |G| + |G| - c|G| = S + n - c|G| \le S + n - \Theta(n/T)$.

Decoding. Given a codeword, if it is not $\perp$, then parse it in the format mentioned above. Set the client's hint accordingly. All values for $[n]\backslash R$ is directly provided by the codeword. The decoder can decode the values for R as follows. For each index $i \in R$, run the query protocol on index i. Whenever the protocol wants to query a value outside R, we can look up the corresponding location in the codeword and provide the value. Now, one of the following cases will happen:

1. The query protocol wants to access a position inside R. In this case, $i \in R\backslash G$, and the value at i is already provided in the codeword.

2. The query protocol never accesses a position inside R. In this case, $i \in G$, and the query protocol will successfully output some value. The decoder checks if $i \in G_1$. If $i \notin G_1$, directly output the outcome of the query protocol; else, flip the outcome and output it.

As long as the encoding algorithm does not output $\perp$, it is easy to check that the decoded outcome must be correct. Therefore, the probability of correct decoding is at least 0.04. □

Completing the proof of Theorem 4. Suppose we have a (possibly randomized) protocol Π for solving Yao's box problem, and for any DB, it achieves probabilistic correctness 0.8 for a random query index. Then, it must be that we can find some good coins, such that for at least 0.1 fraction of the databases, Π^{coin} gives a correct answer on at least 3/4 of the indices. Since otherwise, Π can achieve correctness probability at most $0.1 \cdot 1 + 0.9 \cdot 3/4 < 0.775$ for a random DB and random index $i \in [n]$ which violates the assumption that Π has 0.8 probabilistic correctness.

Now, fixing coins ot be the good choice as mentioned above, we can use the above encoding/decoding procedure to encode the database. We say that a DB is good if Π^{coin} is correct on at least 3/4 of the indices for DB. Henceforth, without loss of generality, we may assume that $T \in [2, o(n)]$. By Lemma 6, as long as DB is good, using the above encoding/decoding procedure, decoding is correct with probability at least 0.04. Further, we know that 0.1 fraction of the DBs are good. Therefore, when we randomly choose DB, the probability of correct decoding is at least $0.1 \cdot 0.04 = \Theta(1)$. By Fact 1, the length of the encoding is at least $n - \Theta(1)$. Therefore, we have that

$$S + n - \Theta(n/T) \geq n - \Theta(1)$$

which gives us $S \cdot T \geq \Omega(n)$.

References

[CGK20] Henry Corrigan-Gibbs and Dmitry Kogan. Private information retrieval with sublinear online time. In *Advances in Cryptology – EUROCRYPT 2020*, pages 44–75, Cham, 2020. Springer International Publishing.

[DTT10] Anindya De, Luca Trevisan, and Madhur Tulsiani. Time space tradeoffs for attacks against one-way functions and prgs. In *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 649–665. Springer, 2010.

[Gal14] David Galvin. Three tutorial lectures on entropy and counting, 2014.

[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.