COMS 6998-2: Information Theory in Theoretical CS November 21, 2017

Lecture 9 – Streaming

Instructor: *Omri Weinstein* Scribes: *Rishabh Dudeja (rd2714)*

1 Recap of Last Lecture

In the last lecture we introduced the Streaming Model. In the streaming model we receive a stream $\boldsymbol{x} = x_1, x_2 \dots x_n$ consisting of elements x_i from a universe of size m, (that is, $x_i \in [m]$) in an online fashion. The Goal is to design an algorithm $\mathcal{A}$ that approximately computes a given function on the stream: $f(\boldsymbol{x})$ using $o(m)$ (sublinear) space and a single pass.

We will require our algorithms to be (ϵ, δ) approximate which is defined as follows:

Definition 1 ((ϵ, δ) Approximate)**.** *A streaming algorithm $\mathcal{A}$ is said to (ϵ, δ) approximate a function f if,*

$$\Pr[\mathcal{A}(\boldsymbol{x}) \in (1 \pm \epsilon) f(\boldsymbol{x})] \geq 1 - \delta$$

Given a stream $\boldsymbol{x}$ one can define its frequency vector $f(\boldsymbol{x}) = (f_1, f_2, f_3 \dots f_m)$ where f_i is the number of times item i occurs in the stream.

A very important streaming problem is to compute the moments of the frequency vector for the stream. Given a stream $\boldsymbol{x}$ we define its pth moment $F_p(\boldsymbol{x})$ as:

$$F_p(\boldsymbol{x}) := \begin{cases} p = 0 & |\{i : |f_i| > 0\}| \\ p = \infty & \max_{i \in [m]} |f_i| \\ p \neq 0, \infty & \sum_{i=1}^{m} |f_i|^p \end{cases}$$

The following theorem gives a characterization of the streaming complexity of Frequency Moments:

Theorem 2. *1. For $p \in [0, 2]$ there is a randomized streaming algorithm that (ϵ, δ) approximates F_p in space $s = O(poly(\log m, \log n))$.*

2. For $p > 2$, There is a streaming algorithm that (ϵ, δ) approximates F_p in space $s = O(m^{1-2/p})$. Furthermore, any randomized streaming algorithm that does so requires $s \geq \Omega(m^{1-2/p})$

In this lecture we will see:

1. The AMS algorithm [AMS96] for streaming F_2 in $O(\log(m))$ space.
2. A communication complexity based $\Omega(m)$ lower bound on the space complexity of streaming F_∞ which will also imply a space lower bound of $\Omega(m^{1-2/p})$ for streaming F_p, $p > 2$.

2 Upper Bounds

2.1 The AMS Algorithm for F_2

The general approach to designing streaming algorithms consist of 2 steps:

1. Design a succinct (low space), one pass estimator $A(\boldsymbol{x})$ such that $\mathbb{E}[A(\boldsymbol{x})] = f(\boldsymbol{x})$
2. Construct t independent copies of the estimator $A_i(\boldsymbol{x})$ and output

$$\mathcal{A}(\boldsymbol{x}) = \frac{1}{t}\sum_{i=1}^{t} A_i(\boldsymbol{x})$$

   This is done because averaging several independent copies of A will improve the concentration of our estimate around $f(\boldsymbol{x})$ and allow us to achieve (ϵ, δ) approximation.

We first address step 2 using the following lemma which tells us how big t should be to achieve (ϵ, δ) approximation.

Lemma 3. *Suppose for an estimator $A(\boldsymbol{x})$, the following hold:*

1. $\mathbb{E}[A(\boldsymbol{x})] = f(\boldsymbol{x})$
2. $Var(A(\boldsymbol{x})) \leq K\mathbb{E}[A(\boldsymbol{x})]^2$

Then, $t = O\left(\frac{K}{\epsilon^2\delta}\right)$ suffice to ensure that $\mathcal{A}(\boldsymbol{x})$ (ϵ, δ) approximates $f(\boldsymbol{x})$

Proof. By Linearity of Expectation,

$$\mathbb{E}[\mathcal{A}(\boldsymbol{x})] = g(\boldsymbol{x})$$

Since the copies $A_i(\boldsymbol{x})$ were independent,

$$\mathrm{Var}(\mathcal{A}(\boldsymbol{x})) = \frac{\mathrm{Var}(A)}{t} \leq \frac{K\mathbb{E}[A]^2}{t}$$

Finally we use Chebychev's Inequality to compute:

$$\begin{aligned}
\mathbb{P}[\mathcal{A}(\boldsymbol{x}) \notin (1 \pm \epsilon) f(\boldsymbol{x})] &= \Pr[|\mathcal{A}(\boldsymbol{x}) - \mathbb{E}[\mathcal{A}(\boldsymbol{x})]| \geq \epsilon\mathbb{E}[\mathcal{A}(\boldsymbol{x})]] \\
&\leq \frac{\mathrm{Var}(\mathcal{A})}{\epsilon^2\mathbb{E}[A]^2} && \text{[Chebychev's Inequality]} \\
&\leq \frac{K}{\epsilon^2 t}
\end{aligned}$$

Hence to ensure $\mathcal{A}$ (ϵ, δ) approximates $f(\boldsymbol{x})$, we need to ensure $\frac{K}{\epsilon^2 t} \leq \delta$. Hence taking $t = \frac{K}{\epsilon^2\delta}$ is sufficient. □

Remark 1. *It is possible to achieve (ϵ, δ) approximation with $t = O(K\log(1/\delta)/\epsilon^2)$ by computing the median (instead of the mean) of independent estimators and analyzing the concentration of the Median using Chernoff's Inequality.*

Next we describe the AMS algorithm (also known as Tug-Of-War sketch) for F_2 which uses $s = O(\log(m))$ space. The algorithm is shown in Figure 1.

A streaming algorithm for estimating F_2 ("Tug of War")

1. Initialize:
2. Choose $h : [m] \mapsto \{\pm 1\}$ independently at random[a] (e.g., (+1,+1,-1,+1,-1,-1,-1,+1,. . . , -1))
3. $Y \longleftarrow 0$.
4. Process:
5. foreach i do
6. $y \longleftarrow y + h(x_i)$
7. Output: Y^2.

[a] We'll actually see that choosing $h()$ randomly from a family of 4-wise independent hash functions suffices, and requires only $O(\lg m)$ bits of space to represent.

Figure 1: A streaming algorithm for F_2

The following lemma analyzes the expected value and the variance of the estimator returned by the algorithm.

Lemma 4. *The estimator returned by AMS algorithm satisfies:*

$$\mathbb{E}[Y] = \|f\|_2^2 = F_2(\boldsymbol{x})$$
$$Var[Y] \leq 2\mathbb{E}[Y]^2$$

Proof. First we compute the mean of the estimator:

$$\begin{aligned}\mathbb{E}[Y] &= \mathbb{E}[Z^2] \\ &= \mathbb{E}\left[\left(\sum_{j=1}^{m} f_j h_j\right)^2\right] \\ &= \sum_{i,j} \mathbb{E}[f_i f_j h_i h_j]\end{aligned}$$

Next we note that since for $i \neq j$, $h_i \perp h_j$, hence, $\mathbb{E}[h_i h_j] = \mathbb{E}[h_i]\mathbb{E}[h_j] = 0$. Hence,

$$\mathbb{E}[Y] = \sum_{i=1}^{m} f_j^2 = \|f\|_2^2$$

Next we compute the variance:

$$\begin{aligned}
\text{Var}[Y] &= \mathbb{E}[Y^2] - \mathbb{E}[Y]^2 \\
&= \mathbb{E}[Z^4] - \|f\|_2^2 \\
&= \mathbb{E}\left[\left(\sum_{j=1}^{m} f_j h_j\right)^4\right] - \|f\|_2^2 \\
&= \sum_{i,j,k,l} \mathbb{E}[f_i f_j f_k f_l h_i h_j h_k h_l] - \|f\|_2^2
\end{aligned}$$

As before we note, because of independence, the only values of (i, j, k, l) with $\mathbb{E}[h_i h_j h_k h_l] \neq 0$ are those which consist of 2 equal pairs or $i = j = k = l$. Hence,

$$\begin{aligned}
\text{Var}(Y) &= 3\sum_{i \neq j} f_i^2 f_j^2 + \sum_i f_i^4 - \|f\|_2^2 \\
&= 3\sum_{i,j} f_i^2 f_j^2 - 3\sum_i f_i^4 + \sum_i f_i^4 - \|f\|^2 \\
&\leq 3\|f\|_2^2 - \|f\|_2^2 \\
&= 2\|f\|_2^2
\end{aligned}$$

□

Hence the AMS estimator, satisfies lemma 3 with $K = 2$. Hence averaging $t = O(\frac{1}{\epsilon^2 \delta})$ independent AMS estimators is sufficient to get an (ϵ, δ) approximation to F_2.

Finally we note that if the AMS algorithm is implemented naively, it requires $O(m)$ space to store the hash function h. However, a crucial observation is that we don't really need h to be a fully independent hash function. It is sufficient for the above arguments to go through that h satisfies:

$$\forall i \neq j \neq k \neq l, (h_i, h_j, h_k, h_l) \text{ are independent}$$

Such a hash function is called a 4-wise independent hash function. It turns out 4-wise independent hash functions can be stored in $O(\log(m))$ space. This is done as follows:

1. First we construct a random polynomial over $\mathbb{F}_{m^2}$ by sampling 4 random coefficients $(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \in_R [m] \times [m] \times [m] \times [m]$. We define the random polynomial as:

$$P(x) = (\alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3) \mod m^2$$

2. We construct our hash function $h : [m] \to \{-1, 1\}$ by computing $h(x)$ as:

$$h(x) := \begin{cases} 1 & \text{Leading Bit of } P(x) \text{ is 1} \\ -1 & \text{Leading Bit of } P(x) \text{ is 0} \end{cases}$$

It can be shown that the result h is 4-wise indepedent. Furthermore storing h just requires storing $\alpha_0, \alpha_1 \ldots \alpha_3$ which requires $4\log(m)$ bits. In conclusion, we have the following theorem for approximating F_2:

Theorem 5. *There exists a streaming algorithm which uses $O_{\epsilon,\delta}(\log(m))$ space to (ϵ, δ) approximate F_2.*

2.2 F_1: Estimating the Length of a Stream (Morris' Algorithm [Mor78])

Suppose we're given a stream of characters, and we'd like to simply estimate the length of the stream n. This can be done trivially using a $(\lg n)$-bit counter, hence requires $s = \Theta(\lg n)$ space naively. If we're willing to settle for (arbitrarily small constant) *approximation*, then turns out we can do much better – using only $s = O(\lg \lg n)$ space.

The algorithm is shown in Figure 2.

Intuitively, the streaming algorithm $\mathcal{A}$ is attempting to store the *logarithm* of n. Indeed, by induction on n, it is straightforward to prove that

$$\mathbb{E}[2^{C_n}] = n + 1,$$

A streaming algorithm for estimating F_1
1. Initialize a counter C to 0.
2. For each incoming character, set $C \longleftarrow C + 1$ w.p $1/2^C$.
3. Output $A := 2^C - 1$.

Figure 2: A streaming algorithm for F_1

The harder step is showing that $Var[A] = O((\mathbb{E}[A])^2)$.

3 Streaming Lower Bounds Via Communication Complexity

3.1 Warm Up: Space Lower Bound for Exact Computation of F_∞

The basic idea behind using Communication Complexity to prove lower bounds for various applications (for example streaming) is as follows: First, we cook up a communication game closely related to the problem of interest. Second, we show that a too-good-to-be-true streaming algorithm for the problem of interest would imply a too-good-to-be-true Communication Protocol for the communication game. Finally, we analyze the Communication Game using the information theoretic machinary we learnt so far.

Given a streaming problem of computing a function g on stream $\boldsymbol{x}$, there is a natural communication game associated with it: We split the stream into two parts: The Past: $\boldsymbol{x}_P$ stream and the future $\boldsymbol{x}_F$ stream. Alice receives $\boldsymbol{x}_P$ and Bob receives $\boldsymbol{x}_F$ and the Communication Problem is to compute the function $g(\boldsymbol{x}) = g(\boldsymbol{x}_P, \boldsymbol{x}_F)$ using a communication protocol. The following theorem makes this reduction precise.

Theorem 6. *If there exists a space s deterministic streaming algorithm for computing a function g on stream $\boldsymbol{x}$, then there exists a deterministic 1-way communication protocol Π for computing $g(\boldsymbol{x}_P, \boldsymbol{x}_F)$ such that $\|\Pi\| \leq s$*

Proof. Given a streaming algorithm $\mathcal{A}$, we construct the communication protocol as follows: Alice runs $\mathcal{A}$ on $\boldsymbol{x}_P$. Since $\mathcal{A}$ was a streaming algorithm, she can send a snapshot of the state of the algorithm at the end of her string to Bob who can use it to resume the execution of the algorithm on his string. The correctness of Π follows from the correctness of $\mathcal{A}$ for computing $g(\boldsymbol{x})$. Furthermore, since $\mathcal{A}$ uses just s bits of memory, $\|\Pi\| \leq s$. □

Remark 2. *The fact the communication protocol induced by the algorithm is 1-way can be leveraged to prove stronger lower bounds. In particular we can use communication problems which are hard for one way communication protocols but not for interactive communication in our lower bounds.*

Remark 3. *The above reduction splits the string into 2 substrings distributed to Alice and Bob. One can imagine lower bounds where the string is divided k substrings and distributed among k parties. Such k party communication lower bounds are often stronger and can help close logarthmic gaps between upper and lower bounds.*

A corollary of the above theorem is that any deterministic streaming algorithm for F_∞ requires $\Omega(m)$ space.

Corollary 7. *Any deterministic streaming algorithm for F_∞ requires $\Omega(m)$ space.*

Proof. We show that a streaming algorithm using space s on the universe $[m]$ can be used to construct a communication protocol for DISJ_m with communication complexity s. Once we have shown this, the space lower bound follows because the deterministic communication complexity of DISJ_m is $\Omega(m)$. We first recall that an instance of DISJ_m is a pair $(\boldsymbol{u}, \boldsymbol{v}) \in \{0,1\}^m \times \{0,1\}^m$ where $\boldsymbol{u}$ and $\boldsymbol{v}$ encode subsets of $[m]$: namely, $u_i = 1$ iff $i \in \boldsymbol{u}$. Furthermore, $\text{DISJ}_m(\boldsymbol{u}, \boldsymbol{v}) = \mathbb{I}(\boldsymbol{u} \cap \boldsymbol{v} = \phi)$.

The communication protocol will be as follows: Given an instance of of DISJ_m, Alice takes her input and creates a stream $\boldsymbol{x}_P$ consisting of the 1-indices of her input $\boldsymbol{u}$, that is $x_P = \{i : u_i = 1\}$. Likewise Bob creates a stream $\boldsymbol{x}_F$ out of $\boldsymbol{v}$. Alice runs the streaming algorithm $\mathcal{A}$ on her string and then sends the snapshot of the state to Bob who uses the snapshot to run $\mathcal{A}$ on his stream. The protocol outputs 1 iff $F_\infty(\boldsymbol{x}_P, \boldsymbol{x}_F) = 1$.

The correctness of this protocol follows because $\text{DISJ}_m(\boldsymbol{u}, \boldsymbol{v}) = 1$ iff $F_\infty(\boldsymbol{x}_P, \boldsymbol{x}_F) = 1$. This is because if there was any element common in Alice and Bob's input the frequency of that element would be exactly 2. Finally to finish the proof we note that $\|\Pi\| = s$. □

Remark 4. *The trivial algorithm for exactly computing F_∞ which maintains a counter for each element of the universe has space complexity $O(m \log n)$. Hence the lower bound for exactly computing F_∞ is tight upto logarithmic dependence on stream length.*

Next we will prove a space lower bound for approximately computing F_∞. However, first we will do a quick review of our information theory toolbox.

3.2 Review of Hellinger Distance

We recall the definition and basic properties of Hellinger Distance.

The hellinger distance h between two distributions μ and ν is defined as:

$$h(\mu, \nu) := \frac{1}{2}\|\sqrt{\mu} - \sqrt{\nu}\|_2$$

Squaring both sides of this equation we obtain:

$$h^2(\mu, \nu) = 1 - \langle \sqrt{\mu}, \sqrt{\nu} \rangle$$

The hellinger distance can be sandwiched between the statistical distance:

$$h^2(\mu, \nu) \leq \Delta(\mu, \nu) \leq \sqrt{2} h(\mu, \nu)$$

Finally we state a lemma we haven't seen before but will be useful later:

Lemma 8 (Hellinger Distance v.s. Information)**.** *Let Z be a random variable distributed uniformly on the set $\{z_1, z_2\}$. Let f be a possibly randomized function of Z. Let $f_{z_i}, i = 1, 2$, denote the distribution of f when $Z = z_i$, then, $I(f(Z); Z) \geq h^2(f_{z_1}, f_{z_2})$*

The proof of this lemma will be an exercise in HW4. Intuitively this lemma says that if the randomized function f provides little information about Z, then the distribution of f when $Z = z_1$ and $Z = z_2$ must be close.

3.3 Space Lower Bound for Approximate Computation of F_∞

The main theorem we will prove in this section is:

Theorem 9. *Any randomized c-approximate streaming algorithm for computing F_∞ which succeeds with probability $1/2 + \epsilon$ requires $\Omega(m\epsilon^2/c^2)$ space.*

Before we prove this result, we explore some consequences of the result for streaming other ℓ_p, $p \in (2, \infty)$ norms. The basic fact that we will use is the following relation between ℓ_p and ℓ_∞ norms:

Lemma 10. *For any vector $\boldsymbol{x} \in \mathbb{R}^m$, and any $p \geq 2$, we have:*

$$\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_p \leq m^{1/p} \|\boldsymbol{x}\|_\infty$$

Proof. The proof follows from the following sequence of inequalities:

$$\begin{aligned}
\|\boldsymbol{x}\|_\infty^p &= \max_{i \in [m]} |x_i|^p \\
&\leq \sum_{i=1}^{p} |x_i|^p \\
&= \|\boldsymbol{x}\|_p^p \\
&\leq m \max_{i \in [m]} |x_i|^p \\
&= m \|\boldsymbol{x}\|_\infty^p
\end{aligned}$$

Taking the p root gives us the required claim. □

A corollary of Theorem 3 is:

Corollary 11. *Any randomized streaming algorithm which computes a constant fraction approximation to F_p , $p \in [2, \infty)$ with success probability $2/3$ requires $\Omega(m^{1-2/p})$ space.*

Proof. Consider a streaming algorithm $\mathcal{A}$ which c-approximates F_p with probability 2/3. That is, $F_p(\boldsymbol{x}) \leq \mathcal{A}(\boldsymbol{x}) \leq cF_p(\boldsymbol{x})$ with probability 2/3. By Lemma 4, $F_\infty(\boldsymbol{x}) \leq \mathcal{A}(\boldsymbol{x}) \leq cm^{1/p}F_\infty(\boldsymbol{x})$. This means that $\mathcal{A}(\boldsymbol{x})$ is a $O(m^{1/p})$ approximation to $F_\infty(\boldsymbol{x})$ with probability 2/3. Hence by Theorem 3, $\mathcal{A}$ must use $\Omega(m^{1-2/p})$ space. □

We will now present the proof of Theorem 3. A roadmap of the proof is as follows:

Step 1: Show that a too good to be true streaming algorithm would imply a too good to be true communication protocol for a suitable communication game. The game we will consider is called the Gap-$\mathrm{L}_\infty^{m,c}$ Problem. This reduction is proved in Lemma 12.

Step 2: Analyze the Randomized Communication Complexity of Gap-$\mathrm{L}_\infty^{m,c}$ using tools we have learned so far.

Step 1: Reduction to Gap-$\mathbf{L}_\infty^{m,c}$

Communication Problem 1 (Gap-$\mathrm{L}_\infty^{m,c}$)**.** *Given two vectors $\boldsymbol{u}, \boldsymbol{v} \in [c]^m$, with the promise that either $\|\boldsymbol{u} - \boldsymbol{v}\|_\infty \geq c$ or $\|\boldsymbol{u} - \boldsymbol{v}\|_\infty \leq 1$, compute:*

$$\textit{Gap-}L_\infty^{m,c}(\boldsymbol{u}, \boldsymbol{v}) := \begin{cases} 1 & \|\boldsymbol{u} - \boldsymbol{v}\|_\infty \geq c \\ 0 & \|\boldsymbol{u} - \boldsymbol{v}\|_\infty \leq 1 \end{cases}$$

The following lemma establishes the reduction between streaming and the communication problem we defined:

Lemma 12. *Let $\mathcal{A}$ be an streaming algorithm (that can handle both increments as well as decrements to the frequency vector) that $c/2-$approximates F_∞ with probability atleast $1/2 + \epsilon$, The $\mathcal{A}$ must use atleast $R^{1/2-\epsilon}(\textit{Gap-}L_\infty^{m,c})$ Space.*

Proof. Let $\mathcal{A}$ be any such streaming algorithm which uses space s. We will use it to construct a communication protocol Π for Gap-$\mathrm{L}_\infty^{m,c}$ which succeeds with probability $1/2+\epsilon$ and $\|\Pi\| = s$. This would imply that $s \geq R^{1/2-\epsilon}(\text{Gap-}\mathrm{L}_\infty^{m,c})$ as required. The protocol is defined as follows: Given an instance $\boldsymbol{x}, \boldsymbol{y}$ of Gap-$\mathrm{L}_\infty^{m,c}$), Alice interprets her input $\boldsymbol{x}$ as a sequence of positive updates of the frequency vector for the set of items $[m]$ and runs $\mathcal{A}$ on it. She then sends a snapshot of the state of the algorithm to Bob who interprets his input $\boldsymbol{y}$ as a sequence of decrements to the frequency vector and continues to run $\mathcal{A}$ on them. By the approximation guarantee for $\mathcal{A}$, with probability atleast $1/2+\epsilon$, $\|\boldsymbol{x}-\boldsymbol{y}\|_\infty \leq \mathcal{A}(\boldsymbol{x}-\boldsymbol{y}) \leq c\|\boldsymbol{x}-\boldsymbol{y}\|_\infty/2$. If $\mathcal{A}(\boldsymbol{x} - \boldsymbol{y}) > c/2$, then the protocol outputs 1 else it outputs 0. The correctness of Π follows because if $\|\boldsymbol{x} - \boldsymbol{y}\|_\infty \geq c$, then w.p. atleast $1/2 + \epsilon$, $\mathcal{A}(\boldsymbol{x} - \boldsymbol{y}) \geq \|\boldsymbol{x} - \boldsymbol{y}\|_\infty \geq c$. Hence with probability $1/2 + \epsilon$, $\Pi(\boldsymbol{x}, \boldsymbol{y}) = 1$. On the other hand if $\|\boldsymbol{x} - \boldsymbol{y}\|_\infty \leq 1$, with probability atleast $1/2 + \epsilon$, $\mathcal{A}(\boldsymbol{x} - \boldsymbol{y}) \leq c/2$, and hence with probability atleast $1/2 + \epsilon$, $\Pi(\boldsymbol{x}, \boldsymbol{y}) = 0$. Finally we note that clearly, $\|\Pi\| = s$ which finishes the proof. □

Step 2: Analyzing $R^{1/2-\epsilon}$(Gap-$\mathbf{L}_\infty^{m,c}$) In light of the previous lemma it suffices to show that:

$$R^{1/2-\epsilon}(\text{Gap-}\mathrm{L}_\infty^{m,c}) \geq \Omega(m\epsilon^2/c^2)$$

.

To show this we follow the same steps as we did to show the lower bound for DISJ. Given a too-good-to-be-true protocol for Gap-$L_\infty^{m,c}$, we will construct a too-good-to-be-true protocol for the scalar version of Gap-L_∞, which we call the DIST Problem:

Communication Problem 2 (DIST$^c(x,y)$)**.** *Given two inputs $(x,y) \in [c] \times [c]$ with the promise that either $|x-y| \geq c$ or $|x-y| \leq 1$, compute the DIST$^c(x,y)$ function:*

$$DIST^c(x,y) := \begin{cases} 1 & |x-y| \geq c \\ 0 & |x-y| \leq 1 \end{cases}$$

We note that:

$$\text{Gap-L}_\infty^{m,c}(\boldsymbol{x},\boldsymbol{y}) = \bigvee_{i=1}^{m} \text{DIST}^c(x_i,y_i)$$

.

Given a protocol Π for Gap-$L_\infty^{m,c}$, which succeeds on all inputs with probability $1/2+\epsilon$ and $\|\Pi\| = \delta m$ we will design a protocol π for DIST^c which is correct with probability $1/2+\epsilon$ for all instances of DIST^c but has low information cost $\text{IC}_\mu(\pi)$ for a well designed hard distribution μ. The protocol π is shown below:

	Protocol π for DIST$^c(x,y)$
0	Initialize $\boldsymbol{X},\boldsymbol{Y} := (0,0\ldots0)$
1	Using Public randomness sample $I \in_R$ [m] and set $X_I = x, Y_I = y$
2	$\forall i \neq I$, Draw (X_i,Y_i) independently from distribution μ as follows:
	First, Sample $R_i := (S_i,T_i) \in_R [0:c-1] \times \{A,B\}$ using public randomness.
	If $T_i = A$, Alice draws $X_i \in_R \{S_i, S_i+1\}$ and Bob sets $Y_i = S_i+1$.
	If $T_i = B$, then Alice sets $X_i = S_i$ and Bob draws $Y_i \in_R \{S_i, S_i+1\}$
3	Alice and Bob execute $\Pi(\boldsymbol{X},\boldsymbol{Y})$ and output its result.

Now before analyzing the protocol we introduce some notation: We will use π_{ij} to denote the distribution over transcripts of the protocol π when it is run with $x = i, y = j$. We define the vector $R := (R_1, R_2 \ldots R_m)$.

The first step is to analyze the error probability of π

Lemma 13 (π is a low error protocol)**.** *For any instance x,y of the DIST^c problem, $\mathbb{P}[\pi(x,y) = DIST^c(x,y)] \geq 1/2+\epsilon$. Furthermore, $h(\pi_{00},\pi_{0c}) \geq \sqrt{2}\epsilon$ and $h(\pi_{c0},\pi_{cc}) \geq \sqrt{2}\epsilon$*

Proof. Let I denote the random location at which the protocol π plants x and y. Since the distribution that was used to sample the remaining entries of $\boldsymbol{x}$ and $\boldsymbol{y}$ in the protocol π is supported on the NO instances of the DIST problem, we have with probability $1/2+\epsilon$ (because of the guarantee of Π):

$$\pi(x,y) = \Pi(\boldsymbol{X},\boldsymbol{Y}) = \text{Gap-L}_\infty^{m,c}(\boldsymbol{X},\boldsymbol{Y}) = \bigvee_{i=1}^{m} \text{DIST}^c(X_i,Y_i) = \text{DIST}(x,y)$$

Next we note that $x=0, y=0$ is a NO instance of DIST while $x=0, y=c$ is a YES instance of DIST. Hence by definition of statistical distance:

$$\Delta(\pi_{00},\pi_{0c}) \geq \mathbb{P}[\pi(0,0)=0] - \mathbb{P}[\pi(0,c)=0] \geq 2\epsilon$$

Finally relating the hellinger distance to statistical distance gives us:

$$h(\pi_{00}, \pi_{0c}) \geq \Delta(\pi_{00}, \pi_{0c})/\sqrt{2} \geq \sqrt{2}\epsilon$$

The claim about $h(\pi_{c0}, \pi_{cc})$ follows analogously. □

Next we show that under the hard distribution μ, π has low information cost.

Lemma 14.

$$IC_\mu(\pi) \leq \delta$$

Proof.

$$\mathrm{IC}_\mu(\pi) = I(\Pi; X_I | Y_I, I, R) + I(\Pi; Y_I | X_I, I, R)$$

Next we note that conditioned on the public randomness, $X_I \perp Y_I | I, R$. By Problem 2ii in HW3,

$$IC_\mu(\pi) = I(\Pi; X_I, Y_I | R, I)$$

Next we utilize the chain rule of Mutual Information as follows:

$$\begin{aligned}
\mathrm{IC}_\mu(\pi) &= I(\Pi; X_I, Y_I | R, I) \\
&= \frac{1}{m} \sum_{i=1}^{m} I(\Pi; X_i, Y_i | R) \\
&\leq \frac{1}{m} \sum_{i=1}^{m} I(\Pi; X_i, Y_i | R_i, X_{<i}, Y_{<i}) && [\text{Since } (X_i, Y_i) \perp (X_{<i}, Y_{<i}))] \\
&= \frac{I(\Pi; X, Y)}{m} && [\text{Chain Rule}] \\
&= \delta
\end{aligned}$$

Finally we observe that in the process of proving this lemma, we showed that:

$$\frac{1}{m} \sum_{i=1}^{m} I(\Pi; X_i, Y_i | R) \leq \delta$$

In particular this means there exists $i \in [m]$ such that Π doesn't reveal to much information about coordinate i:

$$I(\Pi; X_i, Y_i | R) \leq \delta$$

Hence while designing π instead of planting the giving inputs (x, y) at a publically drawn random coordinate, we can plant the inputs at this fixed coordinate i. We will assume this is what π does in the remainder of the proof. □

Next we show that since π is a Low Information protocol, the distribution of the transcript on $(x = 0, y = 0)$ is close to the distribution of the transcript on $x = c, y = c$.

Lemma 15 ($\pi_{00} \approx \pi_{cc}$)**.** *The protocol π satisfies:*

$$h(\pi_{00}, \pi_{cc}) \leq 2c\sqrt{\delta}$$

Proof. We begin with the result of lemma 14:

$$\begin{aligned}
\delta &\geq I(\Pi; X_i, Y_i | R) \\
&\geq I(\Pi; X_i, Y_i | R_i) && [(X_i, Y_i) \perp R_{-i}] \\
&= \frac{1}{c}\sum_{s=0}^{c-1}\left(\frac{1}{2}I(\Pi; X_i, Y_i | S_i = s, T_i = A) + \frac{1}{2}I(\Pi; X_i, Y_i | S_i = s, T_i = B)\right) \\
&= \frac{1}{c}\sum_{s=0}^{c-1}\left(\frac{1}{2}I(\Pi; X_i, s+1 | S_i = s, T_i = A) + \frac{1}{2}I(\Pi; s, Y_i | S_i = s, T_i = B)\right)
\end{aligned}$$

Here, in the last step we used the fact that when Alice is tapped $T_i = A$, Bob's Coordinate is set deterministically to $Y_i = s + 1$ and likewise for the second term. Next we note that conditioned on $S_i = s, T_i = A$, $X_i \in_R \{s, s+1\}$. Hence by Lemma 8,

$$I(\Pi; X_i | S_i = s, T_i = A) \geq h^2(\pi_{s,s}, \pi_{s,s+1})$$

Likewise, we have,

$$I(\Pi; Y_i | S_i = s, T_i = B) \geq h^2(\pi_{s,s+1}, \pi_{s+1,s+1})$$

Substituting these inequalities, we get,

$$\frac{1}{2c}\sum_{s=0}^{c-1}\left(h^2(\pi_{s,s}, \pi_{s,s+1}) + h^2(\pi_{s,s+1}, \pi_{s+1,s+1})\right) \leq \delta$$

Next we use the following algebraic inequality which is a consquence of Cauchy Schwartz Inequality: For any real numbers $q_i, i \in [k]$,

$$\sum_{i=1}^{k} q_i \leq \sqrt{k}\left(\sum_{i=1}^{k} q_i^2\right)^{1/2}$$

Applying this to the previous display gives us:

$$\frac{1}{2c}\sum_{i=1}^{c} h(\pi_{s,s}, \pi_{s,s+1}) + h(\pi_{s,s+1}, \pi_{s+1,s+1}) \leq \sqrt{\delta}$$

Applying Triangle Inequality and Telescoping the sum gives:

$$h(\pi_{0,0}, \pi_{c,c}) \leq 2c\sqrt{\delta}$$

Which is what we had to show. □

The final step is to combine the lower bounds on $h(\pi_{0,0}, \pi_{c,0}), h(\pi_{c,0}, \pi_{c,c})$ from lemma 13 and the upper bound on $h(\pi_{0,0}, \pi_{s,s})$ from lemma 15 to deduce a lower bound on δ. The consequences of these two lemmas are summarized in Figure 3.

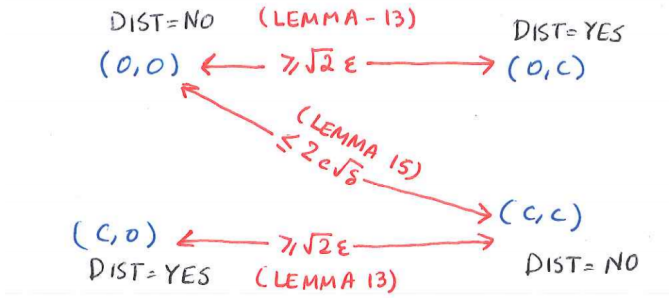

Figure 3: Consequences of Lemma 13 and 15

In the case of DISJ_n we used the cut and paste lemma to accomplish this. But this does not work here. To see why, applying the cut-paste lemma, $h(\pi_{0,0}, \pi_{c,c}) = h(\pi_{0,c}, \pi_{c,0}) \leq 2c\sqrt{\delta}$. But both $(0, c)$ and $(c, 0)$ are YES instances and so we expect them to be close and the cut and paste lemma doesn't give us closeness of a YES and a NO instance. To get around this, we use Pythagorean Inequality for hellinger distance called the Z-lemma

Lemma 16 (Z-lemma)**.** *Consider a randomized communication protocol $\tau : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$. Then for any $x, x' \in \mathcal{X}$ and y, y' in $\mathcal{Y}$, we have:*

$$h^2(\tau_{x,y}, \tau_{x',y'}) \geq h^2(\tau_{x,y}, \tau_{x,y'})) + h^2(\tau_{x',y}, \tau_{x',y'})$$

Proof. This will be an Exercise in HW4. □

The failure of Cut-and-Paste and the use of Z-lemma is shown in the following figure.

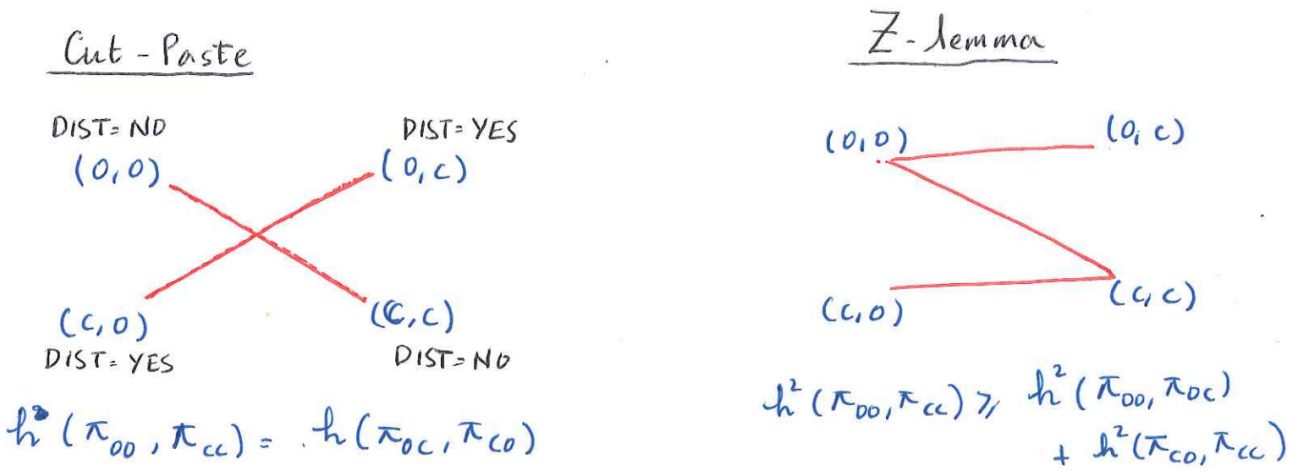

Figure 4: Failure of Cut-And-Paste (left) and Implication of Z-Lemma (right)

Applying the Z lemma, we get along with lemma 13 and 15 we get:

$$\begin{aligned} 4c^2\delta &\geq h^2(\pi_{0,0}, \pi_{c,c}) \\ &\geq h^2(\pi_{0,0}, \pi_{0,c}) + h^2(\pi_{c,0}, \pi_{c,c}) \\ &\geq 2\epsilon^2 + 2\epsilon^2 \\ &= 4\epsilon^2 \end{aligned}$$

Recalling that $\delta = \|\Pi\|/m$, gives us $\|\Pi\| \geq \Omega(m\epsilon^2/c^2)$. In conclusion, we showed:

1. For any randomized protocol Π which succeeds with probability $1/2 + \epsilon$, $\|\Pi\| \geq \Omega(m\epsilon^2/c^2)$. This means that $R^{1/2-\epsilon}(\text{Gap-L}_\infty^{m,c}) \geq \Omega(m\epsilon^2/c^2)$

2. By the reduction proved in Lemma 12, this means that any ϵ-approximate streaming algorithm for F_∞ must use $\Omega(m\epsilon^2/c^2)$. This concludes the proof of Theorem 9.

References

[AMS96] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 20–29. ACM, 1996.

[Mor78] Robert Morris. Counting large numbers of events in small registers. *Communications of the ACM*, 21(10):840–842, 1978.