CS 671: Graph Streaming Algorithms and Lower Bounds **Rutgers: Fall 2020**

Lecture 7

October 20, 2020

Instructor: Sepehr Assadi *Scribe: Sepehr Assadi*

In this lecture, we will primarily focus on the following paper:

- "Sepehr Assadi, Sanjeev Khanna, Yang Li, Tight Bounds for Single-Pass Streaming Complexity of the Set Cover Problem. In STOC 2016, SICOMP 2019"

1 The Minimum Set Cover Problem

In this lecture, we slightly deviate from graph problems and instead consider another family of classical optimization problems, namely, *coverage* problems[1], and in particular the *set cover* problem.

In the set cover problem, we a collection of m sets $S_1, \ldots, S_m$ from a universe $\{1, \ldots, n\}$ and the goal is to find the minimum number of sets whose union cover the entire universe. The set cover problem is a classical optimization problem that has been studied from numerous angles in combinatorics, computer science, operation research, complexity theory, and so on. In particular, set cover is one of Karp's original 21 NP-hard problems and its study from both (approximation) algorithms as well as hardness results has led to the development of fundamental techniques for the entire TCS.

We are going to study this problem in the streaming model. However, let us start by a warm-up on getting an $O(\log n)$ approximation for this problem using the greedy algorithm in the classical setting.

Proposition 1. *Consider the following greedy algorithm for set cover: as long as there is an element left uncovered, pick the set that covers the maximum number of uncovered elements in the set cover. This greedy algorithm achieves an $O(\log n)$ approximation.*

Proof. Suppose $O_1, \ldots, O_k$ is an optimal set cover of the instance. For any iteration i of this algorithm, let n_i denote the number of uncovered elements at the beginning of iteration i (so $n_1 = n$). The important observation is that in any iteration i, there is a set that covers at least n_i/k elements; this is because $O_1, \ldots, O_k$ together cover all of n_i elements and so one of them has to cover at least n_i/k elements (if we have chosen some of O_i's already, this ratio can only become better).

As such, for any iteration i, $n_{i+1} \leq n_i - \frac{n_i}{k} = \left(1 - \frac{1}{k}\right) n_i$. This means that the number of uncovered elements shrink by a factor of at least $(1 - 1/k)$ in every iteration of the greedy algorithm. Hence, after $i^* = k \ln n$ iterations,

$$n_{i^*+1} \leq \left(1 - \frac{1}{k}\right)^{i^*} \cdot n < \exp\left(-\ln n\right) \cdot n < 1,$$

where we used the fact that $1 - x < e^{-x}$ for $x \in (0, 1)$. As such, after picking at most $k \ln n$ sets, greedy covers the entire universe, thus it is a $\ln n$-approximation. □

[1] We should however emphasize that these problems are closely tied to graph optimization problems and have natural graph-theoretic interpretation as well. Moreover, in the streaming setting, these problems are often studied alongside graph problems

Streaming Set Cover Problem. We can model the set cover problem in the streaming model as follows: the universe $\{1,\ldots,n\}$ is known in advance and each set S_i arrives with all its elements together in a streaming fashion; the algorithm can make one pass (or a few in case of multi-pass algorithms) over these sets and at the end, it needs to output an approximate set cover of the universe plus a *certificate of coverage*: for any element e in the universe, the algorithm should specify one set in the output that covers this element.

Notice that in this setting, the input size to the problem is $O(mn)$, while the output is of size $O(n)$. As such, we can potentially hope to achieve a streaming algorithm with space as low as $O(n)$, while at the same time, getting any algorithm with space $o(mn)$ would be non-trivial. Indeed, there are two settings of parameters that have been studied for the set cover problem:

- Streaming algorithms with memory $\widetilde{O}(n)$;
- Streaming algorithms with memory $\widetilde{O}(m+n)$ of even $\widetilde{O}(mn^{\delta})$ for $\delta \in (0,1)$.

We will consider both regimes in this paper and as before, focus solely on single-pass algorithms.

2 Simple Algorithms for Streaming Set Cover

Let us start by designing some simple algorithms for the streaming set cover problem. The first algorithm is a clever, yet quite simple, algorithm with $\widetilde{O}(n)$ space that achieves a $O(\sqrt{n})$-approximation. This result is originally due to Emek and Rosén [5] and was further refined (and extended to multi-pass algorithms) by Chakrabarti and Wirth [3].

Algorithm 1. An $O(\sqrt{n})$-approximation algorithm for streaming set cover.

(*i*) Let $U \leftarrow \{1,\ldots,n\}$ denote the uncovered elements and $Sol \leftarrow \emptyset$ denote the solution.

(*ii*) For each arriving set S_i in the stream: if S_i covers $\geq \sqrt{n}$ elements from U, then

$$Sol \leftarrow Sol \cup \{S_i\} \qquad \text{and} \qquad U \leftarrow U \setminus S_i;$$

(for each newly covered element, mark S_i as the "coverer" of this element in the certificate).

(*iii*) In parallel, for any element $e \in \{1,\ldots,n\}$, pick one arbitrary set S_e from the stream that covers e.

(*iv*) Output $Sol \cup \bigcup_{e \in U} S_e$.

The space complexity of this algorithm is $O(n)$ simply because for every element, we store (the name of) at most 2 sets during the stream. Its correctness also follows immediately because elements in $\{1,\ldots,n\} \setminus U$ are covered by Sol and the remaining elements will be covered by $\bigcup_{e \in U} S_e$. We now analyze its approximation.

Lemma 2. *Algorithm 1 is an $O(\sqrt{n})$-approximation algorithm.*

Proof. The collection Sol can only contain $\sqrt{n}$ sets as each one of them covers at least $\sqrt{n}$ *new* elements and there are n elements in total. Any set not picked in Sol, including the optimal sets, can only cover $\sqrt{n}$ elements from U. Thus, optimum needs at least $|U|/\sqrt{n}$ sets to cover U, while Algorithm 1 uses at most $|U|$ sets to cover U. As a result, the solution returned by Algorithm 1 is at most $2\sqrt{n}$ times larger than the optimum, proving the lemma. □

As such, Algorithm 1 is an $O(n)$-space streaming algorithm for $O(\sqrt{n})$-approximation of set cover. But is there a better streaming algorithm? Emek and Rosén [5] also proved that if one aims for a better approximation, then the space of the algorithm needs to grow to at least $\Omega(m)$. This lower bound was also further refined (and extended to multi-pass algorithms) by Chakrabarti and Wirth [3].

Theorem 3 (cf. [5, 3]). *Any single-pass streaming algorithm for $\sqrt{n}$-approximation of set cover requires $\Omega(m)$ space (in instances where $m = \text{poly}(n)$ for some large degree > 1).*

We shall not prove this lower bound for now as we will prove a stronger bound later in this lecture.

The results of [5] essentially settle the complexity of streaming set cover as long as we focus on the $o(m)$ space regime; but what is we allow for larger space of $\widetilde{O}(m)$ (or even larger as long as it is $o(mn)$)? Let us first show that one can obtain better than $\sqrt{n}$ approximations in $o(mn)$ space via a straightforward algorithm.

Algorithm 2. An α-approximation algorithm for streaming set cover in $O(mn/\alpha)$ space.

(*i*) Take the union of every α consecutive sets in the stream and store them as sets $T_1, \ldots, T_{m/\alpha}$ (for every element in the union, store the name of one of the original sets that covers it also).

(*ii*) Store all sets $T_1, \ldots, T_{m/\alpha}$ during the stream in $O(mn/\alpha)$ space.

(*iii*) Compute a minimum set cover of $T_1, \ldots, T_{m/\alpha}$ at the end of the stream; for any set T_i in the cover, pick all the α sets merged in T_i in the final set cover.

The correctness of this algorithm is immediate to verify. As for the approximation ratio, the optimum solution of $T_1, \ldots, T_{m/\alpha}$ has size at most as large as the original set system; since for every set of this optimal solution we pick α sets from the original set system, we immediately obtain an α-approximation.

Algorithm 2 suggests that one can do (slightly) better than the $\sqrt{n}$-approximation when the memory can be $\Omega(m)$. But is this the limit? Or can we also get an $O(1)$-approximation (or even, say, $n^{1/3}$) in $\widetilde{O}(m)$ space? This question was first addressed by Har-Peled, Indyk, Mahabadi, and Vakilian [6] who proved that, among other interesting results for multi-pass algorithms, any better-than-3/2-approximation algorithm requires $\Omega(mn)$ space. This however still leaves open the possibility of the type of algorithms above. But then Assadi, Khanna, and Li [1] proved that, perhaps surprisingly, the straightforward α-approximation algorithm in $O(mn/\alpha)$ is in fact already optimal.

Theorem 4 ([1]). *For any $\alpha = o\left(\frac{\sqrt{n}}{\log n}\right)$ and $m = \text{poly}(n)$, any single-pass α-approximation streaming algorithm for set cover requires $\Omega(mn/\alpha)$ space.*

We will prove this theorem in the rest of this lecture.

3 A Hard Distribution for Set Cover

The proof of Theorem 4 is via a communication complexity lower bound for set cover. As before, the first step in proving this lower bound is identifying a hard distribution of inputs which we do in this section[2].

Distribution 1. A hard distribution $\mathcal{D}_{\mathsf{SC}}$ for one-way two-player set cover communication problem.

Notation. Let $s := \frac{n}{10\alpha}$ and $t := 10\alpha \log m$.

(*i*) The input to Alice is the following: Sample m sets $S_1, \ldots, S_m$ independently and uniformly random among s-subsets of $[n]$, and give them to Alice.

(*ii*) The input to Bob is the following: Sample $i^* \in [m]$ uniformly at random and let $T := [n] \setminus E$ where E is chosen uniformly at random among all sets such that $|E| = t$ and $|E \setminus S_{i^*}| = 1$; give T to Bob.

[2]For simplicity of the notation, the number of sets in these instances will be $m + 1$ as opposed to m.

We establish some key properties of this distribution with respect to the set cover problem that helps explain why we expect it to be hard for set cover. The proof of both claims are application of Chernoff bound (for negatively correlated random variables) and are omitted in this version.

Claim 5. *For any instance $(S_1, \ldots, S_m, T)$ sampled from $\mathcal{D}_{\mathsf{SC}}$, with probability $1-o(1)$, the optimum solution has size 3 (and otherwise the instance is not feasible, i.e., there is no way to cover the entire universe).*

Claim 6. *For any instance $(S_1, \ldots, S_m, T)$ sampled from $\mathcal{D}_{\mathsf{SC}}$, with probability $1-o(1)$, no collection of 3α sets can cover more than half of E.*

We will reduce the set cover problem over $\mathcal{D}_{\mathsf{SC}}$ to the following problem that we call the *Trap* problem.

Definition 7 (Trap Communication Problem). Alice is given a set $A \subseteq [n]$ of size $s < n/2$. Bob is given a set B of size $t = o(s)$ such that $B \setminus A = \{e^*\}$ (e^* is called the *target* element). The goal for Bob is to output a set $C \subseteq B$ of size $t/2$ such that $e^* \in C$, i.e., "trap" the target element among half of his input B.

The plan for the rest of the proof is as follows.

(*i*) We show that the message sent by Alice should allow the players to solve the Trap problem for the pair (S_{i^*}, E) in the distribution $\mathcal{D}_{\mathsf{SC}}$; this is roughly because Bob should cover the element in $|E \setminus S_{i^*}|$ using set other S_{i^*} and by Claim 5 and Claim 6, for this to happen, he should be able to trap $e^* = E \setminus S_{i^*}$ in a set of size at most half of E (so that the trap set can be covered by 3α sets other than S_{i^*}).

(*ii*) We will then prove that solving the Trap problem requires $\Omega(s)$ communication which would be $\Omega(n/\alpha)$ in our case.

(*iii*) The above steps only allow us to prove a lower bound of $\Omega(n/\alpha)$ as opposed to $\Omega(mn/\alpha)$ that we need. However, notice that even though Alice and Bob only need to solve Trap for (S_{i^*}, E), Alice is unaware of which set is S_{i^*}; so effectively Alice's message should be able to solve Trap not only given S_{i^*} but for almost all of the sets $S_1, \ldots, S_m$. We thus would like to be able to argue that solving set cover is "equivalent" to solving m instances of the Trap problem, and thus "should be" m times harder as well: this will then give us the $\Omega(mn/\alpha)$ lower bound.

The first two steps of the plan above should sound familiar by now considering we already covered multiple streaming lower bounds. The last step however is somewhat new; we need to be able to prove a statement of the following type, often referred to as a **direct sum** problem:

> *if solving a problem P requires c unit of "resources", the solving m* independent *copies of P should require $\geq m \cdot c$ resources.*[3]

An extremely powerful yet simple tool for answering direct sum questions in the context of communication problems is *information complexity*. Roughly speaking, in this paradigm, instead of focusing on the *length* of the messages communicated between Alice and Bob, we focus on the *amount of information* revealed about their inputs (to themselves or to external observers) when solving the problem. In the rest of this lecture, we focus on building the background on properly defining these notions (what do even mean by "information") and how they can be used to address these direct sum questions. We then will come back in the next lecture and use these to implement our three step plan for the lower bound of the set cover problem.

[3]Notice that while such a statement may sound "obviously" true, it is in fact not even true in most scenarios. For instance, consider multiplying a $n \times n$ fixed matrix A with a vector x; this obviously requires $\Theta(n^2)$ time. But multiplying A with n vectors $x_1, \ldots, x_n$ can be done via fast matrix multiplication in $O(n^{2.37\cdots}) \ll n^3$ time which is much less resources than solving n independent problems individually (we shall emphasize that some care is needed to turn this question into a proper direct sum question to ensure that we are indeed solving n independent problems and not correlated ones).

4 A Quick Intro to Information Theory

Entropy

Suppose you have a random variable X from a domain $\mathcal{X}$. How many bits to do you need to "encode" a sample X and send it over to someone else? Well, in the *worst-case*, definitely $\log|\mathcal{X}|$ bits is needed, just by pigeonhole principle. But what about *average-case* (over the randomness of X)? At this point, the answer depends on the distribution of X. For instance, if X has a uniform distribution, intuitively you cannot do much better than (almost) $\log|\mathcal{X}|$ bits even on average, while if almost all of the mass of X is on a single value x_0, you can do much better by encoding x_0 with a shorter message than the rest.

A rough idea is to do the following: start from the largest probability p of any element under X; we can have at most $1/p$ different $x \in \mathcal{X}$ with $\Pr(X = x) = p(x) = p$ as otherwise the sum of their probabilities will be more than 1; thus, we can encode each of these x's with $\log(1/p)$ bits and then move to the next value of p and continue like this. This way, for any element x, the *expected* length of encoding used for the element x is $p(x) \cdot \log(1/p(x))$ (ignoring all ceiling/floor issues, etc.).[4]

Based on the above discussion, we define the *entropy* of a random variable X with PDF $p(x)$ for $x \in \mathcal{X}$ as:

$$\mathbb{H}(X) := \mathop{\mathbb{E}}_{x\sim X}\left[\log\frac{1}{p(x))}\right] = \sum_{x\in\mathcal{X}} p(x)\cdot\log\frac{1}{p(x)}; \tag{1}$$

(we use the convention that $0 \cdot \log(1/0) = 0$ in the definition above).

By the above intuition, we can think of $\mathbb{H}(X)$ as a "measure" of the average length of best encoding of X.[5]

Examples. What are the entropy of each of the random variables below over the domain $\Omega = \{1, \ldots, n\}$? (compare these with the encoding length of variables you "expect" to achieve by your "own" coding scheme.)

- X: uniform over $\mathcal{X}$;
- Y: deterministically equal to 1 always;
- Z: $\Pr(Z = 1) = \frac{1}{2}$ and $\Pr(Z = i) = \frac{1}{2(n-1)}$ for any other $i \neq 1 \in \Omega$.

Let us now establish several useful properties of entropy. For that, we first need to recall *Jensen's inequality*.

Proposition 8 (**Jensen's inequality**). *Let f a* convex *function and X be a random variable over Ω. Then,*

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)].$$

Moreover, the equality holds iff X is deterministic or f is linear.

Considering $\log(1/x)$ function is a convex function of x, we can use Jensen's inequality to establish several simple properties of entropy.

Proposition 9 (**entropy and support size**). *For a random variable X with domain $\mathcal{X}$, we have,*

$$0 \leq \mathbb{H}(X) \leq \log|\mathcal{X}|.$$

The LHS is tight iff X is deterministic and the RHS is tight iff X is uniform.

[4] This is a very handwavy argument and is only meant to help the reader put the notion of entropy in some context; that being said, this is also not too far from what happens with Huffman coding.

[5] And again, while this is not precise, this is also not too far from the truth as Huffman coding achieves average length for a random variable X which is between $\mathbb{H}(X)$ and $\mathbb{H}(X) + 1$.

Proof. The LHS is obviously true because $\mathbb{H}(X)$ is expectation of non-negative terms and the only case they are all zero is when X is deterministic. For the RHS, we apply Jensen's inequality as follows:

$$\mathbb{H}(X) = \mathop{\mathbb{E}}_{x\sim X}\left[\log\left(1/p(x)\right)\right] \le \log\left(\mathop{\mathbb{E}}_{x\sim X}\left[1/p(x)\right]\right) = \log|\mathcal{X}|.$$

Jensen's inequality will be tight here iff all $p(x)$ values are the same (making the random variable $1/p(x)$ deterministic), thus implying the second part. □

Proposition 10 (**sub-additivity of entropy**). *For random variables X, Y, we have,*

$$\mathbb{H}(X,Y) \le \mathbb{H}(X) + \mathbb{H}(Y);$$

moreover, the equality holds iff $X \perp Y$. Here $\mathbb{H}(X,Y)$ is simply entropy of the joint *random variable (X,Y).*

Proof. We can compute $\mathbb{H}(X,Y) - (\mathbb{H}(X) + \mathbb{H}(Y))$ as follows:

$$\begin{aligned}
\mathbb{H}(X,Y) - \mathbb{H}(X) - \mathbb{H}(Y) &= \sum_{x,y} p(x,y)\log\frac{1}{p(x,y)} - \sum_{x,y} p(x,y)\log\frac{1}{p(x)} - \sum_{x,y} p(x,y)\log\frac{1}{p(y)} \\
&\qquad\qquad (p(x) = \textstyle\sum_y p(x,y) \text{ and } p(y) = \sum_{x,y} p(x,y)) \\
&= \sum_{x,y} p(x,y)\log\frac{p(x)p(y)}{p(x,y)} = \mathop{\mathbb{E}}_{(x,y)\sim(X,Y)}\left[\log\frac{p(x)p(y)}{p(x,y)}\right] \\
&\le \log\left(\mathop{\mathbb{E}}_{(x,y)\sim(X,Y)}\left[\frac{p(x)p(y)}{p(x,y)}\right]\right) \qquad \text{(by Jensen's inequality)} \\
&= \log\left(\sum_{x,y} p(x)p(y)\right) = \log 1 = 0.
\end{aligned}$$

Moreover, Jensen's inequality above is tight whenever $p(x)p(y) = p(x,y)$ for all x, y, meaning $X \perp Y$. □

Conditional Entropy

Recall the motivating example for the entropy and now suppose that you have joint random variables (X,Y): Given a sample $(x,y) \sim (X,Y)$, how many bits do you need to encode x alone and send it to someone who already *knows* y? Again, we are interested in this question on *average* over choices of both X and Y. You can again see that this question depends on the distribution of X, Y and the correlation between the two; for instance for X, Y with fixed marginals, say, both uniform, the answer would be very different when $X = Y$ so highly correlated vs. when $X \perp Y$.

This discussion brings us to the notion of *conditional entropy*:

$$\mathbb{H}(X \mid Y) := \mathop{\mathbb{E}}_{y\sim Y}\left[\mathbb{H}(X \mid Y = y)\right] = \sum_{y\in\mathcal{Y}} p(y) \cdot \sum_{x\in\mathcal{X}} p(x \mid y) \cdot \log\frac{1}{p(x \mid y)}; \tag{2}$$

(notice that here $\mathbb{H}(X \mid Y = y)$ is just the entropy of random variable X' with distribution $X \mid Y = y$).

Examples. What are the conditional entropy of each of the random variables? (compare these with the encoding length you "expect" to achieve by your "own" coding scheme.)

- $\mathbb{H}(A \mid B)$: when A, B are independent and uniform over $\{0,1\}^n$;
- $\mathbb{H}(C \mid D)$: when C is uniform over $\{0,1\}^n$ and D is XOR of bits of D.
- $\mathbb{H}(E \mid F)$: when E is uniform over $\{0,1\}^n$ and F is the indicator random variable for $E = 0^n$.

Let us now present on some useful properties of conditional entropy.

Proposition 11 (**chain rule of entropy**). *For a random variables X, Y, we have,*

$$\mathbb{H}(X, Y) = \mathbb{H}(X) + \mathbb{H}(Y \mid X).$$

Proof. We can compute $\mathbb{H}(X, Y) - (\mathbb{H}(X) + \mathbb{H}(Y \mid X))$ as follows:

$$\begin{aligned}
\mathbb{H}(X, Y) - \mathbb{H}(X) - \mathbb{H}(Y \mid X) &= \sum_{x,y} p(x, y) \log \frac{1}{p(x, y)} - \sum_{x,y} p(x, y) \log \frac{1}{p(x)} - \sum_{x} p(x) \sum_{y} p(y \mid x) \log \frac{1}{p(y \mid x)} \\
&= \sum_{x,y} p(x, y) \log \frac{1}{p(x, y)} - \sum_{x,y} p(x, y) \log \frac{1}{p(x)} - \sum_{x,y} p(x, y) \log \frac{1}{p(y \mid x)} \\
&\qquad\qquad (p(x) \cdot p(y \mid x) = p(x, y)) \\
&= \sum_{x,y} p(x, y) \log \frac{p(x) \cdot p(y \mid x)}{p(x, y)} = \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x, y)} \\
&= \sum_{x,y} p(x, y) \log 1 = 0.
\end{aligned}$$

□

We shall note that chain rule is one of the most important properties of entropy as it gives us *additivity*.

Proposition 12 (**conditioning cannot increase entropy**). *For a random variables X, Y, we have,*

$$\mathbb{H}(X \mid Y) \leq \mathbb{H}(X);$$

moreover, the equality holds iff $X \perp Y$.

Proof. By sub-additivity $\mathbb{H}(X, Y) \leq \mathbb{H}(X) + \mathbb{H}(Y)$ while by chain rule $\mathbb{H}(X, Y) = \mathbb{H}(Y) + \mathbb{H}(X \mid Y)$. Plugging in these two implies the result. The second part follows from the second part of Proposition 10. □

Let us emphasize that this proposition is only true when conditioning on a random variable and not an event (i.e., realization of a random variable). Can you give an example when the latter does not hold?

Mutual Information

Finally, we can talk about "information" between two random variables X, Y. In the context of the motivating examples before, we would like to quantify how much the knowledge of Y helped us in encoding X, i.e., the gap between the average encoding of X with or without the extra knowledge of Y (and vice versa).

Formally, the *mutual information* between two variables X, Y is defined as:

$$\mathbb{I}(X\,;\,Y) := \mathbb{H}(X, Y) - (\mathbb{H}(X \mid Y) + \mathbb{H}(Y \mid X)). \tag{3}$$

By applying chain rule of entropy, we can see right away that

$$\mathbb{I}(X\,;\,Y) := \mathbb{H}(X) - \mathbb{H}(X \mid Y) = \mathbb{H}(Y) - \mathbb{H}(Y \mid X). \tag{4}$$

which might sound more familiar. Finally, *conditional mutual information* is defined analogously:

$$\begin{aligned}
\mathbb{I}(X\,;\,Y \mid Z) &:= \mathbb{H}(X, Y \mid Z) - (\mathbb{H}(X \mid Z, Y) + \mathbb{H}(Y \mid Z, X)) &(5)\\
&= \mathbb{H}(X \mid Z) - \mathbb{H}(X \mid Z, Y) = \mathbb{H}(Y \mid Z) - \mathbb{H}(Y \mid Z, X). &(6)
\end{aligned}$$

Examples. What are the mutual information between the following random variables?

- $\mathbb{I}(A\,;\,B)$: when A, B are independent and uniform over $\{0,1\}^n$;
- $\mathbb{I}(C\,;\,D)$: when C is uniform over $\{0,1\}^n$ and D is XOR of bits of D.
- $\mathbb{I}(E\,;\,F)$: when E is uniform over $\{0,1\}^n$ and F is the indicator random variable for $E = 0^n$.

Some of the important properties of mutual information is as follows.

Proposition 13 (**mutual information is non-negative**). *For any random variables X, Y,*

$$0 \le \mathbb{I}(X\,;\,Y) \le \min\{\mathbb{H}(X), \mathbb{H}(Y)\}\,;$$

moreover, the LHS inequality is tight iff $X \perp Y$.

Proof. The LHS is because $\mathbb{I}(X\,;\,Y) = \mathbb{H}(X) - \mathbb{H}(X \mid Y)$ and conditioning cannot increase entropy. The RHS is by non-negativity of entropy. The second part of the result also follows from Proposition 12. □

Proposition 14 (**chain rule of mutual information**). *For any random variables X, Y, Z,*

$$\mathbb{I}(X, Y\,;\,Z) = \mathbb{I}(X\,;\,Z) + \mathbb{I}(Y\,;\,Z \mid X).$$

(note that (X, Y) is one argument of the mutual information term as a joint variable and Z the other, which are separated by ';').

Proof. By the definition of mutual information and chain rule of entropy,

$$\begin{aligned}\mathbb{I}(X, Y\,;\,Z) &= \mathbb{H}(X, Y) - \mathbb{H}(X, Y \mid Z) = \mathbb{H}(X) + \mathbb{H}(Y \mid X) - \mathbb{H}(X \mid Z) - \mathbb{H}(Y \mid Z, X)\\ &= \mathbb{H}(X) - \mathbb{H}(X \mid Z) + \mathbb{H}(Y \mid X) - \mathbb{H}(Y \mid X, Z) = \mathbb{I}(X\,;\,Z) + \mathbb{I}(Y\,;\,Z \mid X).\end{aligned}$$

□

Unlike entropy, mutual information does not behave that straightforwardly under conditioning (which is to be expected from our intuition). For instance (you should prove both statements below):

- if X, Y are independent and uniform over $\{0,1\}$ and $Z = X \oplus Y$, then,

$$\mathbb{I}(X\,;\,Y) = 0 \quad \text{but} \quad \mathbb{I}(X\,;\,Y \mid Z) = 1;$$

- on the other hand, if $X = Y = Z$ and X is uniform over $\{0,1\}$, then,

$$\mathbb{I}(X\,;\,Y) = 1 \quad \text{but} \quad \mathbb{I}(X\,;\,Y \mid Z) = 0,$$

Still, there are many cases that we can say something interesting about the effect of conditioning on a mutual information term.

Proposition 15 (**conditioning on an independent random variable cannot decrease information**). *For any random variables X, Y, Z, if $X \perp Z$, then,*

$$\mathbb{I}(X\,;\,Y) \le \mathbb{I}(X\,;\,Y \mid Z).$$

Proof. By the definition of mutual information and the fact that conditioning cannot increase entropy,

$$\begin{aligned}\mathbb{I}(X\,;\,Y \mid Z) &= \mathbb{H}(X \mid Z) - \mathbb{H}(X \mid Y, Z)\\ &= \mathbb{H}(X) - \mathbb{H}(X \mid Y, Z) && \text{(by moreover part of Proposition 12 since } X \perp Z)\\ &\ge \mathbb{H}(X) - \mathbb{H}(X \mid Y) && \text{(as conditioning on } Z \text{ cannot increase the entropy)}\\ &= \mathbb{I}(X\,;\,Y).\end{aligned}$$

□

Remark. This section was only a glance into the amazing area of information theory and in no ways can do justice to this field. Interested reader is referred to the excellent textbook of Cover and Thomas [4] as well as numerous courses on information theory tools in TCS for further background.

5 Information Theory Methods in Communication Complexity[6]

Before we even get to the notion of information complexity, it is worth to examine the power of information theory tools in addressing communication complexity questions, by studying our favorite problem, the Index problem, through this lens. Recall that in the Index problem, Alice is given a string $x \in \{0,1\}^n$ and Bob is given an index $i \in [n]$; the goal is for Alice to send a single message m to Bob and Bob uses m and i to output x_i. We prove the following lower bound for the Index problem (compare this with our previous lower bound from Lecture 1).

Theorem 16. *For any $\delta \in (0,1)$, any one-way δ-error protocol for Index over the uniform distribution of $x \in \{0,1\}^n$ and $i \in [n]$ requires*

$$(1 - H_2(\delta)) \cdot n,$$

communication where

$$H_2(\delta) := \delta \cdot \log \frac{1}{\delta} + (1-\delta) \cdot \log \frac{1}{(1-\delta)},$$

is the binary entropy function (i.e., entropy of a Bernoulli random variable with mean δ).[7]

To prove this theorem, we need just one more tool from information theory: the Fano's inequality.

Proposition 17 (Fano's inequality). *Let A, B be two random variables and suppose that there is a function $g : supp(B) \to supp(A)$ such that $\Pr(A \neq g(B)) = \delta$. Then, $H(A \mid B) \leq H_2(\delta)$.*

Proving this proposition is left as an exercise in Problem set 7. Using this, we prove the theorem.

Proof of Theorem 16. Let X, M, I denote the random variable for x, m, i, respectively. Let $g(m,i)$ denote the deterministic function used by Bob to output the answer given m and i.

Since the protocol errs with probability at most δ, we know that

$$\mathop{\mathbb{E}}_{(m,i)} \Pr_x (x_i \neq g(m,i) \mid M = m, I = i) \leq \delta.$$

By taking A to be the random variable X_I and $B = (M, I)$ in Fano's inequality, we obtain that

$$\mathbb{H}(X_I \mid M, I) \leq \delta.$$

On the other hand, we know that $\mathbb{H}(X_I \mid I) = 1$, as the distribution of X_I even conditioned on I is uniform over $\{0,1\}$ and thus we can apply Proposition 9. Putting these two together implies that

$$\mathbb{I}(X_I \,;\, M \mid I) = 1 - H_2(\delta),$$

i.e., M needs to reveal $1 - H_2(\delta)$ about X_I conditioned on I; this perfectly matches our intuition that Alice should be able to reveal x_i to Bob via the message m.

[6] The title of this section is borrowed from the title of the pioneering work of [2] which is one of the first papers that initiated the extremely fruitful approach of using information theory methods to address communication complexity questions.

[7] Note that for any constant $\delta < 1/2$, $H_2(\delta) < 1$ is a constant and thus the lower bound is $\Omega(n)$.

Let us now show that considering Alice is oblivious to the identity of the index i, for revealing this much information about x_i, Alice actually needs to reveal n times more information across all indices. Formally,

$$\begin{aligned}
\mathbb{I}(X_I\,;\,M\mid I) &= \mathop{\mathbb{E}}_{i\sim I}[\mathbb{I}(X_I\,;\,M\mid I=i)] && \text{(by definition of conditional mutual information)}\\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(X_i\,;\,M\mid I=i) && \text{(as } I \text{ is uniform and since } X_I = X_i \text{ conditioned on } I=i)\\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(X_i\,;\,M)\\
&\quad\text{(as the distribution of } X \text{ which uniquely determines both } X_i \text{ and } M \text{ is independent of the event } I=i)\\
&\leq \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(X_i\,;\,M\mid X^{<i})\\
&\quad\text{(where } X^{<i}=(X_1,\ldots,X_{i-1}) \text{ and by Proposition 15 since } X_i \perp X^{<i} \text{ as } X \text{ is uniform over } \{0,1\}^n)\\
&= \frac{1}{n}\cdot\mathbb{I}(X\,;\,M).\\
&\quad\text{(by a repeated application of the chain rule of mutual information in Proposition 14)}
\end{aligned}$$

Again, this perfectly matches our intuition; the information revealed about a random coordinate is $1/n$ times smaller than the information revealed about the entire string.

Finally, we would like to show that to reveal almost n bit of information about X, the message M has to be of length almost n as well—this is simply because one bit of message cannot carry more than one bit of information. Formally, by applying Proposition 13 and Proposition 9, we have,

$$\mathbb{I}(X\,;\,M)\leq\mathbb{H}(M)\leq\log|\mathit{supp}(M)|.$$

Plugging in this into the bound obtain above, we obtain that

$$\log|\mathit{supp}(M)|\geq(1-H_2(\delta))\cdot n,$$

which means that we need to have at least $2^{(1-H_2(\delta))\cdot n}$ many different messages and thus at least one message needs to be of size $(1-H_2(\delta))\cdot n$, proving the theorem. □

References

[1] S. Assadi, S. Khanna, and Y. Li. Tight bounds for single-pass streaming complexity of the set cover problem. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 698–711, 2016. 3

[2] Z. Bar-Yossef, T. S. Jayram, R. Kumar, and D. Sivakumar. Information theory methods in communication complexity. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity, Montréal, Québec, Canada, May 21-24, 2002*, pages 93–102, 2002. 9

[3] A. Chakrabarti and A. Wirth. Incidence geometries and the pass complexity of semi-streaming set cover. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1365–1373, 2016. 2, 3

[4] T. M. Cover and J. A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006. 9

[5] Y. Emek and A. Rosén. Semi-streaming set cover - (extended abstract). In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 453–464, 2014. 2, 3

[6] S. Har-Peled, P. Indyk, S. Mahabadi, and A. Vakilian. Towards tight bounds for the streaming set cover problem. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 371–383, 2016. 3