CS 229r Information Theory in Computer Science March 26, 2019

Lecture 1

Instructor: Madhu Sudan *Scribe: Nari Johnson*

1 Overview

Today we will discuss:

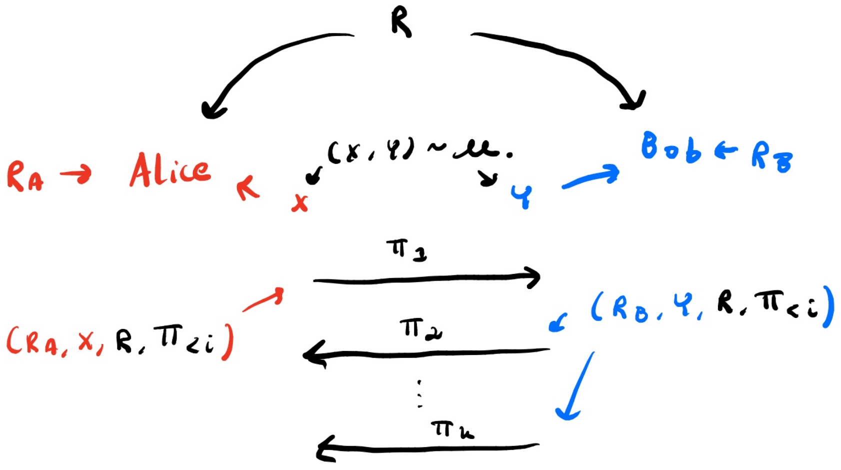
- Compressing Interactive Communication
- Correlated Sampling

2 Review of protocol

We'll begin by discussing single-shot communication, which occurs when some players Alice and Bob only have 1 conversation. In the future, we'll discuss amortized communication and examine what happens when Alice and Bob can have multiple conversations.

2.1 Setup

- Alice has input $X \sim \mu$ unknown to Bob.
- Bob has input $Y \sim \mu$ unknown to Alice.
- Alice has private randomness R_A unknown to Bob.
- Bob has private randomness R_B unknown to Alice.
- Both players observe public randomness R.



2.2 Protocol

The protocol π is the transcript of Alice and Bob's communication. The protocol begins by Alice sending a bit π_1. Bob sends the next bit π_2, and Alice and Bob continue to alternate sending bits until k total bits have been sent.

Each bit $(\pi_i, i \in \{1, k\})$ is a random variable. Because Alice and Bob only have access to their own inputs, their own private randomness, public randomness R, and the transcript so far, we can rewrite each π_i as a function of these variables:

$$\pi_i = \begin{cases} \pi_i(X, R, R_A, \pi_{<i}) & \text{i is odd} \\ \pi_i(Y, R, R_B, \pi_{<i}) & \text{i is even} \end{cases}$$

Because there are k π_i bits exchanged, $CC(\pi) = k$ (the communication complexity of the entire protocol is k).

3 Information Complexity

Definition 1 (Internal Information Complexity)**.** The internal information complexity of protocol π, denoted IC_μ^{int}, is given by

$$IC_\mu^{int}(\pi) = I(X; \pi | Y, R) + I(Y; \pi | X, R)$$

$IC_\mu^{int}(\pi)$ is the information the protocol π conveys to Alice and Bob about each others' inputs. In contrast,

Definition 2 (External Information Complexity)**.** The external information complexity of protocol π, denoted IC_μ^{ext}, is given by

$$IC_\mu^{ext}(\pi) = I(XY; \pi | R)$$

$IC_\mu^{ext}(\pi)$ is the information the protocol π conveys to an outside observer (without prior knowledge of X or Y) about X and Y. Because information is symmetric, this is also equal to the information that X and Y convey about transcript π.

We can rewrite $IC_\mu^{int}(\pi)$ as

$$IC_\mu^{int}(\pi) = \sum_{i=1}^{k} I(\pi_i; X | Y, R, \pi_{<i}) + I(\pi_i; Y | X, R, \pi_{<i})$$

Notice: $\forall i$, one of these two terms is always equal to 0. It's because in each round (for each π_i), we're examining only the randomness of X or Y depending on whether it's Alice or Bob's turn.

Exercise 3. *Show that $IC_\mu^{int}(\pi) \leq IC_\mu^{ext}(\pi)$. Hint: Use the above sum expansion of $IC_\mu^{int}(\pi)$ and the property that only one term is nonzero for each π_i.*

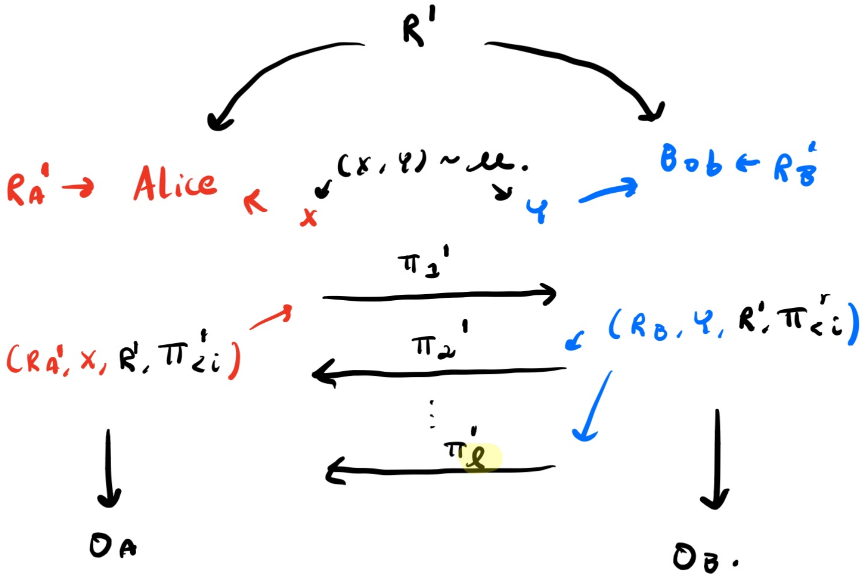
Exercise 4. *Show that $I(X; \pi, R | Y) + I(Y; \pi, R | X) = I(X; \pi | Y, R) + I(Y; \pi | X, R)$.*

3.1 Protocol Simulation

Generally, if a protocol reveals very little information, or if the entropy of a protocol is small, then the protocol itself can be compressed. What does it mean to "compress" a communication? Consider the following modifications to the above interaction:

- We will now consider 1-way communication. Alice will send messages to Bob, but Bob will not send messages to Alice.
- Alice will now send a compressed message to Bob. Intuitively, we will show that Bob can reconstruct Alice's original message (before compression!) from the compressed message.
- The output of our protocol will be this reconstructed transcript.
- Both players have access to the entire transcript, which is synonymous to observing the entire interaction.

More generally, when Alice and Bob can communicate (i.e. when we no longer have the constraint of 1-way communication), we can define our protocol as follows:



Notice:

- Our compressed protocol π' only requires $l < k$ bits.
- O_A, Alice's output transcript, and O_B, Bob's output transcript, are random variables. They are both functions of (X, Y, R, π').

Definition 5 (Protocol Simulation). π' simulates π if $O_A = O_B = (R, \pi)$ and the distributions are exactly preserved.

4 BBCR

BBCR is a compression theorem published by Barak, Braverman, Chen, and Rao in 2007.

Theorem 6 (BBCR). *$\forall \pi$ with $CC(\pi) = k$, $IC(\pi) = I$, then $\exists\, \pi'$ simulating π s.t. $CC(\pi') = O(\sqrt{I \cdot k}\log(k))$.*

BBCR says that by using simulation, we can achieve a communication complexity that's roughly the geometric mean of the information and communication cost. BBCR is the best known compression algorithm for this problem.

Theorem 7 (Braverman)**.** *Using the same assumptions as BBCR,*
$\forall \pi$ *with* $CC(\pi) = k$, $IC(\pi) = I$, *then* $\exists\ \pi'$ *simulating* π *s.t.* $CC(\pi') = 2^{O(I)}$.

Therefore you can use Braverman's to find an expression for the communication complexity of π' depending only on the information complexity of π.

Theorem 8 (Ganor, Kol, Raz)**.** *The above are tight.*

One motivation of BBCR is understanding what happens when we engage in multiple conversations simultaneously. In this scenario,

- We have many inputs $(X_1, ..., X_n)$ and $(Y_1, ..., X_n) \sim \mu$.
- We carry on n conversations, each with input (X_i, Y_i), in parallel.

How can we compress the protocols π of these n parallel conversations? We will formalize this more next time, but we can use BBCR to prove a lower bound of the communication complexity:

Theorem 9. $CC(\pi^{xn}) \geq CC(\pi)\sqrt{n}$

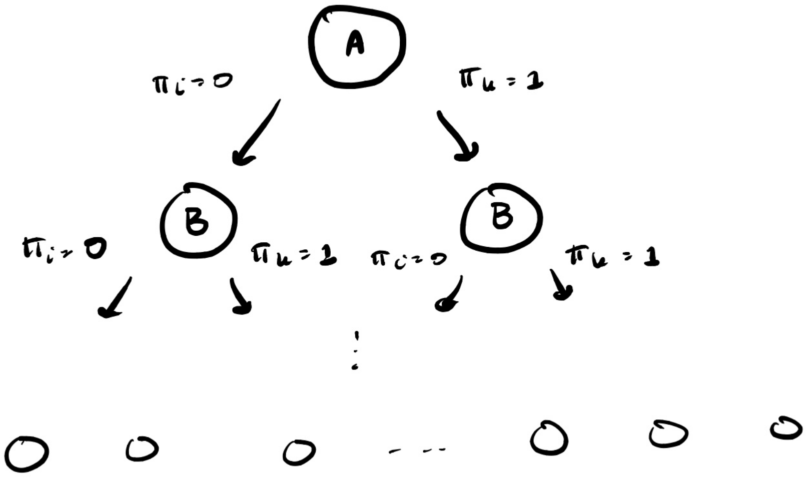
(The above notation π^{xn} is expressed in terms of the direct product, which we will define in a future lecture.)

Theorem 10. $CC(\pi^{xn}) = nI(c\pi)(1 + O(1))$

Today we will prove BBCR as stated in Theorem 6. In order to do this, we'll introduce the notion of a protocol tree.

5 Protocols, Priors, Information Cost

Assume that π has no common randomness. We'll define a protocol tree as follows:



Here, at each node, the path taken (whether or not we traverse right down the tree) represents the bit of transcript that the player whose turn it is sends. At each node of the tree, we need to keep track of who "owns" that node - whether it's Alice or Bob's turn to communicate. Because there are k rounds of interaction, the tree is of depth k. At the final level of the tree, note that each individual leaf represents a distinct "path" taken through the tree. Therefore each leaf represents a distinct transcript of Alice and Bob's communication.

At any level of the tree, what determines if we should go right or left? The public randomness R, the private randomness R_A or R_B, and the input of the player who owns that node. Therefore we're interested in $(\pi_i | \pi_{<i}, X)$. For each node U at level i in the tree, define:

$(P_U^A = \pi_i | \pi_{<i}, X)$, where P_U^A represents the distribution on the direction that Alice believes the path will continue in, and
$(P_U^B = \pi_i | \pi_{<i}, Y)$, where P_U^B represents the distribution on the direction that Bob believes the path will continue in.

Notice that at each step of communication, it's not the public or private randomness, but instead private inputs (X, Y) that bind Alice and Bob to go one way or another. Therefore these two distributions P_U^A and P_U^B will not be identical. The divergence between P_U^A and P_U^B is the information cost.

First, let's consider the setting where the information cost is very low: when $I = 0$. This means that at any level of the tree, the players will learn nothing about the other player's input based on the path that was taken. In order for this to occur, the distributions P_U^A and P_U^B must be identical at every node.

Claim 11. $I = 0$ *only if* $\forall U, P_U^A = P_U^B$.

This means that at every node, the distribution of going left or going right, regardless of whether it's conditioned on Alice or Bob's prior knowledge, is identical. Now we can simulate the entire path throughout the protocol tree using 0 communication. We can do so by using shared randomness R. The players can divide R into real numbers between 0 and 1, and choose to go right or left according to their shared distributions. Therefore our π' to simulate π will have common randomness.

Our π' when $I = 0$ is the "easy" case. Now we will examine the communication complexity when I is very small. What is our distribution on the leaves?

6 Correlated Sampling

We will define a third distribution on each node of the tree, P_U. P_U will model what we will call "the right distribution" on the probability of going right or left at that node.

Definition 12. $P_U = (\pi_i | \pi_{<i}, X, Y)$.

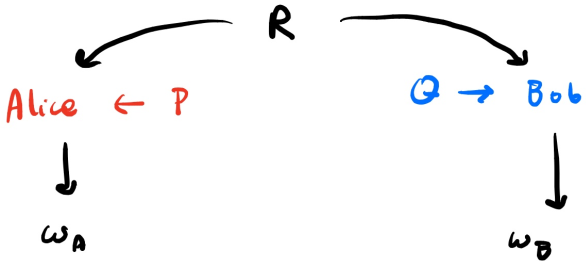
Suppose some player has knowledge of both inputs (X, Y) and wants to know the probability of going right at some node U. Then the player can simulate either Alice or Bob using P_U.

Consider the situation where P_U^A and P_U^B are very close to each other $\forall U$. In this case, we can't just use common randomness R as we did when $I = 0$.

Goal: Sample the root to leaf path of the protocol tree according to the $\{P_U\}_U$, or according to the distribution P_U at each node U.

One solution to this problem comes from correlation sampling. In this problem,

- Alice gets as input a distribution P.
- Bob gets as input a distribution Q.
- Alice and Bob have common randomness R.
- Alice and Bob engage in 0 communication.
- Alice produces output $\omega_A \sim P$.
- Bob produces output $\omega_B \sim Q$.



Example 13. *P, Q can be distributions on the leaves of the tree. Then whp $\omega_A = \omega_B$.*

Objective: Maximize $Pr[\omega_A = \omega_B]$. Or: Minimize $Pr[\omega_A \neq \omega_B]$.

Suppose P and Q have disjoint support. Then $Pr[\omega_A \neq \omega_B] = 1$. On the other hand, as distributions P and Q become closer, then this probability increases. Consider the case when $P = Q$. Then $Pr[\omega_A \neq \omega_B] = 0$. We want something that interpolates nicely between these two extremes.

Exercise 14. *Use the $I = 0$ case to solve this problem when $P = Q$ with common randomness.*

Exercise 15. *Show that $\forall P, Q$,*

$$\min Pr[\omega_A \neq \omega_B] \geq \delta(P, Q)$$

where total variation distance $\delta(P, Q)$ is defined as

$$\delta(P, Q) = \frac{1}{2} \sum_{\omega \in \Omega} |P(\omega) - Q(\omega)|$$

.

Lemma 16 (Broder, Kleinberg, Tardos, Holenstein)**.** *$\exists$ a protocol which achieves*

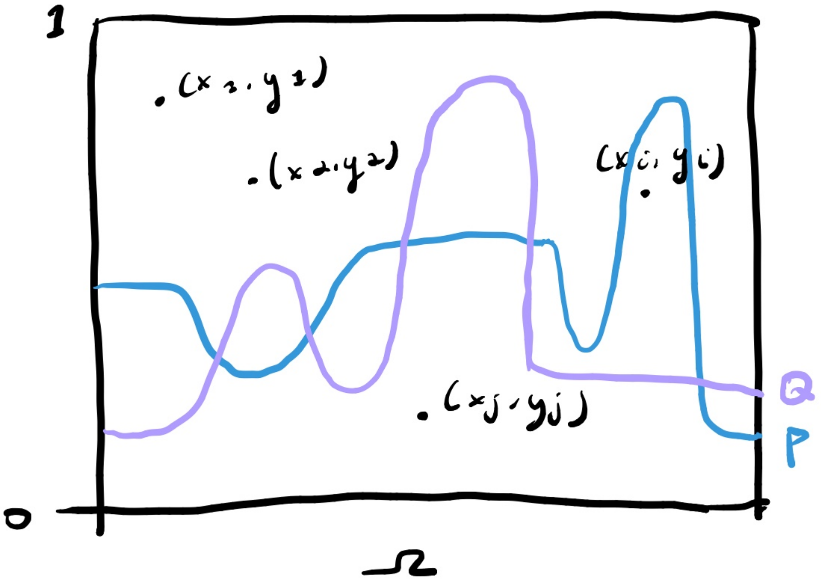
$$Pr[\omega_A \neq \omega_B] \leq \frac{2\delta(P, Q)}{1 + \delta(P, Q)} \leq 2\delta(P, Q)$$

Notice: When $\delta(P, Q) = 1$, we can upper-bound the probability exactly using $\delta(P, Q)$. When $\delta(P, Q)$ is small, then our bound is much closer to $2\delta(P, Q)$.

Note: Broder used a protocol of this form to compare files. Correlated sampling has many applications to search engine hash functions. His solution to this problem is called the min-hash protocol.

6.1 Holenstein's Protocol

How can we assess the similarity between two distributions P and Q?



The x-axis of the above chart represents all of the numbers in alphabet Ω, and the y-axis represents all of the real numbers between 0 and 1. The above lines represent the density functions of distributions P and Q. Bob wants to produce a sample according to the purple distribution (Q), and Alice wants to produce a sample according to the blue distribution (P). We also want the samples to equal each other whp. Here, a "sample" represents an x-coordinate, or a letter drawn from Ω.

We can use the following random protocol:

- Using common randomness R, begin sampling randomly generated sequence of points $\{(x_1, y_1), (x_2, y_2), ...\}$ in the rectangular region.
- Alice will stop when her first point (x_i, y_i) is under the blue curve on the chart. Alice will then output $\omega_A = x_i$. Therefore we maintain that Alice is outputting ω_A according to the distribution P.
- After Alice's turn, similarly Bob will continue to sample random points until his first point (x_j, y_j) is under the purple curve on the chart, and output $\omega_B = x_j$. Similarly here, we maintain that Bob outputs ω_B according to Q.
- Alice and Bob can continue alternating turns and sampling until they receive points under their respective distribution curves, outputting their x values.

Exercise 17. *Show that Holenstein's Protocol achieves*

$$Pr[\omega_A \neq \omega_B] \leq \frac{2\delta(P,Q)}{1+\delta(P,Q)}$$

6.2 Back to our goal...

Recall that our goal is to sample the root-to-leaf path of the protocol tree according to $\{P_U\}_U$. If the distributions P_U^A and P_U^B are very close to each other $\forall U$, then there is a relatively high probability that they will output the same leaf. If we can apply the correlated sampling solution to the distributions P^A and P^B only on the leaf nodes, then we can simulate the path.

A simple example: Suppose that our protocol only involves Alice communicating, and Bob listening. Define

$$P_U^B \sim Bern(\frac{1}{2})$$

$$P_U^A \sim Bern(\frac{1}{2} - \delta)$$

where P_U^B is Bob's prior probability of going right at node U, P_U^A is Alice's prior probability of going right at node U, for small δ and for all nodes U.

Can we upper-bound the variation distance between leaves? To construct the distribution over the leaves, consider that we begin at the root node and traverse down the tree. At any level of the tree, if Alice and Bob disagree, then our distributions will produce different leaves. At any node, the probability that Alice and Bob will disagree is $O(\delta)$. There are k levels, so there are k chances for Alice and Bob to disagree. Therefore

$$TVD(leaf^A, leaf^B) = O(k\delta)$$

where TVD is the total variation distance, $leaf^A$ is Alice's distribution over the leaves, and $leaf^B$ is Bob's distribution over the leaves. Therefore the probability that Alice and Bob don't get the same leaf is $O(k\delta)$.

Assumption: $k\delta$ is tiny.

Now how do we relate this problem to information cost? We can define V_i as

$$V_i = I(\pi_i; X|Y, \pi_{<i}) + I(\pi_i; Y|X, \pi_{<i})$$

In our case, the divergence between these two distributions is δ^2, so we can simplify:

$$V_i = \delta^2$$

This implies that $I = k\delta^2$, as $I = \sum_{i=1}^{k} V_i$. Therefore the communication cost is k, and the information cost is $k\delta^2$.

We can rewrite $O(k\delta) = O(\sqrt{k}\sqrt{k\delta^2})$. This is equal to $O(\sqrt{I}\sqrt{k})$. While this looks similar to the communication complexity bound that we're trying to prove for BBCR, we aren't quite there yet. We've just shown an upper bound for the error in our zero-communication protocol as defined in the Correlated Sampling section (below Definition 12), but this is different from the BBCR protocol, which we want to have no errors. In the next lecture we will use this bound to construct the correct BBCR protocol.