CS 2429 - Foundations of Communication Complexity

Lecture #7: 31 October 2012

Lecturer: Lila Fontes

1 Historical background

Another important technique for proving lower bounds in communication complexity comes from information theory.

In Information Complexity and the Direct Sum Problem for Simultaneous Message Complexity (FOCS 2001), Chakrabarti, Shi, Wirth, and Yao introduced the notion of **information complexity** and used it to prove new lower bounds on the direct sum of the equality function.

Follow-up work by Bar-Yossef, Jayram, Kumar, and Sivakumar (An Information Statistics Approach to Data Stream and Communication Complexity, FOCS 2002) extended the notion of information complexity to develop a general method for proving strong lower bounds in communication complexity.

Recent work by Braverman and Rao (some with collaborators Barak and Chen) has extended the uses of information theory to obtain communication complexity lower bounds, and attempted to understand information complexity as its own concept.

Today we'll look at the paper How to Compress Interactive Communication by Barak, Braverman, Chen, and Rao (STOC 2010).

2 Information Theory

An important quantity for this technique and information theory in general is entropy. Throughout these notes, random variables will be denoted by bold capital letters ($\mathbf{X}$, $\mathbf{Y}$, etc.) and the set of possible values by calligraphic letters (values of $\mathbf{X}$ are drawn from $\mathbb{X}$); lowercase symbols represent individual items in the set ($x \in \mathbb{X}$).

Definition The **entropy** of a random variable $\mathbf{X}$ from some distribution over $\mathbb{X}$ is defined:

$$\mathrm{H}(\mathbf{X}) = -\sum_{\mathbf{x}\in\mathbb{X}} \Pr[\mathbf{x}=\mathbf{X}] \log \Pr[\mathbf{x}=\mathbf{X}] = \mathbb{E}_{\mathbf{x}\in\mathbb{X}} \log \frac{\mathbf{1}}{\Pr[\mathbf{x}=\mathbf{X}]}$$

The conditional entropy is defined likewise, as the expected entropy of random variable $\mathbf{Y}$ conditioned on $\mathbf{X}$'s distribution:

$$\begin{aligned} \mathrm{H}(\mathbf{Y}|\mathbf{X}) &= \sum_{x\in\mathbb{X}} \Pr[x=\mathbf{X}]\, \mathrm{H}(\mathbf{Y}|\mathbf{x}=\mathbf{X}) && (1)\\ &= \sum_{x\in\mathbb{X}, y\in\mathbb{Y}} -\Pr[x=\mathbf{X}, \mathbf{y}=\mathbf{Y}] \log(\Pr[\mathbf{y}=\mathbf{Y}|\mathbf{x}=\mathbf{X}]) && (2) \end{aligned}$$

The mutual information of two random variables is the amount of information that one contains about the other, and is defined by the change in entropy.

Definition The **mutual information** of two random variables $\mathbf{X}$ and $\mathbf{Y}$ is defined:

$$\begin{aligned} \mathrm{I}\left(\mathbf{X};\mathbf{Y}\right) &= \sum_{x\in\mathbb{X}}\sum_{y\in\mathbb{Y}} \Pr[x=\mathbf{X},\mathbf{y}=\mathbf{Y}]\log\frac{\Pr[\mathbf{x}=\mathbf{X},\mathbf{y}=\mathbf{Y}]}{\Pr[\mathbf{x}=\mathbf{X}]\Pr\left[\mathbf{y}=\mathbf{Y}\right]} && (3)\\ &= \mathrm{H}(\mathbf{X})-\mathrm{H}(\mathbf{X}|\mathbf{Y}) && (4)\end{aligned}$$

And conditional mutual information is similar:

$$\mathrm{I}(\mathbf{X};\mathbf{Y}|\mathbf{Z}) = \mathrm{H}(\mathbf{X}|\mathbf{Z}) - \mathrm{H}(\mathbf{X}|\mathbf{Y},\mathbf{Z})$$

We're interested in the information that some protocol π transmits about its inputs $\mathbf{X}$ and $\mathbf{Y}$.

Definition Given a distribution μ on inputs $\mathbf{X}$ and $\mathbf{Y}$, and protocol π, let $\pi(\mathbf{X},\mathbf{Y})$ denote the transcript of the protocol and the randomness used. The **information cost** of π with respect to μ is defined as:

$$\mathrm{ICost}_{\pi,\mu} = I(X,Y;\pi(X,Y)) = \mathrm{IC}^{\mathrm{ext}}_{\mu}(\pi)$$

The information cost captures the information that *an eavesdropper* would learn from the transcript $\pi(\mathbf{X},\mathbf{Y})$ about the inputs $\mathbf{X}$ and $\mathbf{Y}$.

Definition Given a distribution μ on inputs X and Y, and protocol π, let $\pi(X,Y)$ be the transcript of the protocol and the randomness used. The **information content** of π is defined as:

$$\mathrm{IC}_{\mu}(\pi) = \mathrm{I}(X;\pi(X,Y)|Y) + \mathrm{I}(Y;\pi(X,Y)|X) = \mathrm{IC}^{\mathrm{int}}_{\mu}(\pi)$$

The **information content** is the information that the *parties themselves* learn from the transcript $\pi(\mathbf{X},\mathbf{Y})$.

For both internal and external information cost/content, we can define $\mathrm{IC}^{\mathrm{int}}_{\mu,\epsilon}(f)$ as the maximum over all π (such that $\forall x\forall y\ \Pr\left[\pi(x,y)\neq f(x,y)\right]\leq\epsilon$) of $\mathrm{IC}^{\mathrm{int}}_{\mu}(pi)$. The value $\mathrm{IC}^{\mathrm{ext}}_{\mu,\epsilon}(f)$ is defined similarly.

Lemma 1 *Given a distribution μ on inputs $\mathbf{X}$ and $\mathbf{Y}$ and a randomized protocol π,*

$$\mathrm{IC}^{int}_{\mu}(\pi)\leq\mathrm{IC}^{ext}_{\mu}(\pi)\leq\mathrm{CC}(\pi)$$

where $\mathrm{CC}(\pi)$ is the communication complexity of protocol π. Also, for all functions f on $\mathbb{X}\times\mathbb{Y}$, for all ϵ:

$$\mathrm{IC}^{int}_{\mu,\epsilon}(f)\leq\mathrm{IC}^{ext}_{\mu,\epsilon}(f)\leq R^{pub}_{\mu,\epsilon}(f)$$

Proof The first inequality holds because an eavesdropper will always learn more than the participants themselves. Each participant already knows one of the inputs, so for example Alice can only learn from the bits that Bob sends, and if her x is correlated with Bob's y, she may not learn much from these.

The second inequality holds because an eavesdropper can learn at most one bit per bit of protocol transcript.

3 Protocol Compression

Given some communication protocol π, can it be shortened? Our particular goal is to try to shorten π so that its total communication cost is about the same as its information content – no extraneous bits are sent. We also want to keep the error rate about the same.

We might try to compress each round of the protocol by some encoding. But this will still require at least one bit sent per round of π, which may be much bigger than the information content of π.

Compression is hard.

This paper says: forget about keeping the error small when we compress. How low can we get the communication complexity? Hopefully near the information content.

Theorem 2 *There is a universal constant c such that for every distribution μ, every protocol π, and every $\epsilon > 0$, there exist functions π_x and π_y and a protocol τ such that:*

1. $|\pi_x(\mathbf{X}, \tau(\mathbf{X}, \mathbf{Y})) - \pi(\mathbf{X}, \mathbf{Y})| < \epsilon$*, that is, function π_x closely approximates π,*
2. $\Pr[\pi_x(\mathbf{X}, \tau(\mathbf{X}, \mathbf{Y})) \neq \pi_{\mathbf{y}}(\mathbf{Y}, \tau(\mathbf{X}, \mathbf{Y}))] < \epsilon$*, that is, functions π_x and π_y agree with high probability, and*
3. $\mathrm{CC}(\tau) \leq c\sqrt{\mathrm{CC}(\pi)\,\mathrm{IC}_\mu\,(\pi)}\frac{\log(\mathrm{CC}(pi)/\epsilon)}{\epsilon}$*, that is, protocol τ compresses protocol π.*

If the players want to run π, they will instead run τ and use the functions π_x and π_y to "reconstruct" the effects of running π. Condition 1 says that π_x will sample some leaf of the protocol tree for π which is "close" (in statistical distance) to one sampled by π. Condition 2 enforces that both players sample the same leaf with high probability.

Protocol 3 (π) *For our analysis, it will be helpful to phrase the protocol π_r (where r are some fixed random bits) as follows. Each node v of the protocol tree for π_r is either owned by Alice or Bob.*

1. *The players begin at the root node.*
2. *If the players are at some node v a leaf, the computation ends with the protocol output according to that leaf. Otherwise, the player who owns v sends one bit (after possibly using r to sample a random number and make a probabilistic choice) to the other player to indicate which child they should move to.*
3. *Repeat step 2 until the computation ends.*

3.1 Constructing τ, π_x, and π_y

Both players know the protocol tree for π, and the outcome of τ will be that both players *agree* on some path from root to leaf in π's protocol tree.

The general idea of the protocol τ is that each player will (independently) simulate a transcript of π using his "best guess" of what the other player will do. The players then use lemma 8 to find the earliest place where their guessed transcripts differ (with high probability and communication cost polylog in CC), and fix it. They repeatedly perform this repair until they agree on the entire transcript. The communication cost of this protocol scales with the expected number of inconsistencies on the path to a correct leaf, which is bounded by $\sqrt{\mathrm{CC} \cdot \mathrm{IC}}$.

Protocol 4 (τ) *Fix error parameters β and γ. Protocol τ will simulate π by having Alice and Bob avoid communication by guessing each others' samples, then correcting these mistakes.*

1. *Both players sample r from the publicly available randomness; r will be the random bits used in π.*

2. *For each non-leaf node v in the protocol tree for π_r, both players use public randomness to uniformly pick $k_v \in [0,1]$.*

3. **Probabilities at nodes.** *Each player builds his own "guess" of the protocol transcript from each node in the tree, as follows. Suppose Alice owns node v. If*

$$k_v < \Pr[\Pr_v^A(1) \mid x = \mathbf{X} \textit{ and } \pi_{\mathbf{r}}(\mathbf{X},\mathbf{Y}) \textit{ reaches node } \mathbf{v}]$$

then Alice sets the child $c^A(v)$ of v to be 0, and otherwise 1. Bob does likewise, using $\widehat{\Pr}_v^B(1)$ to set the child $c^B(v)$ of v.

4. **Sampling transcripts.** *Each player's "best guess" of the transcript of $\pi_r(\mathbf{X},\mathbf{Y})$ constructed in this way will be the unique path from the root to a leaf.*

5. **Amending transcripts.** *Players use Lemma 8 to repeatedly find (in communication $O(\log(n/\beta)$ bits) the earliest place where their paths differ, and fix it. They correct $\sqrt{\mathrm{CC}(\pi)\cdot \mathrm{IC}_\mu(\pi)}/\gamma$ errors this way.*

3.1.1 Probabilities at nodes

Let v be some node in the protocol tree of π. WLOG at node v, it is player x's turn to speak (player x owns v). The x player Alice therefore knows the correct probabilities $\Pr_v^A(0)$ and $\Pr_v^A(1)$ of the bit he is about to send (which child node the protocol should continue at). The y player Bob does not know these probabilities, but can estimate them ($\widehat{\Pr}_v^B(0)$ and $\widehat{\Pr}_v^B(1)$) based on $\mathbf{Y}$ (according to the probability of seeing an 0 or a 1 conditioned on $\mathbf{Y}$ and reaching node v).

3.1.2 Sampling transcripts

For every node v in the protocol tree for π, both players have these probabilities ($\Pr_v^A(0)$, $\Pr_v^A(1)$, $\widehat{\Pr}_v^B(0)$ and $\widehat{\Pr}_v^B(1)$). They used shared randomness to pick a *shared* random number k_v at each node v. WLOG Alice owns node v.

- If $k_v < \Pr_v^A(1)$ then Alice sets the child $c^A(v)$ of v to be 0, and 1 otherwise.

- If $k_v < \widehat{\Pr}_v^B(1)$ then Bob sets the child $c^B(v)$ of v to be 0, and 1 otherwise.

This sampling method guarantees that if the players' guesses are close to the correct distribution, then the probability that they sample the same bit is high. (If the players guesses are close, that means that this bit in the transcript of π contains very little information content!)

Let $v_0, \ldots, v_{\mathrm{CC}(\pi)}$ be the correct path in the tree. The node $v_{\mathrm{CC}(\pi)}$ has the same distribution as a leaf in π. The players' goal is to find $v_{\mathrm{CC}(\pi)}$ with very little communication.

Alice has some sequence of nodes $v_0^A, \dots, v_{\mathrm{CC}(\pi)}^A$, where $v_{i+1} = c^A(v_i)$, which is the most likely transcript according to her guesses and the sampled numbers k_v. Similarly, Bob has $v_0^B, \dots, v_{\mathrm{CC}(\pi)}^B$.

Note: If these two sequences agree up to the i^{th} node, then they must be correct up to the i^{th} node. Thus the protocol τ will have Alice and Bob attempt to rectify their entire sequences. (Once they agree on all the nodes in the sequence, they have a correct transcript!)

3.1.3 Amending Transcripts

The players now must communicate to find the first i where $v_i^A \neq v_i^B$. Then the owner of node v_{i-1} sends the correct bit, and the other player corrects the remainder of their transcript guess $v_i, v_{i+1}, \dots v_{\mathrm{CC}(\pi)}$.

They keep fixing their transcripts in this way. By Lemma 8, this can be done with communication $O(\log \mathrm{CC}(\pi)/\epsilon)$.

The total communication cost of τ is bounded by (the number of mistakes that Alice and Bob correct) times (the communication cost of finding each mistake):

$$\mathrm{CC}(\tau_{\beta,\gamma}) = O\Big(\sqrt{\mathrm{CC}(\pi) \cdot \mathrm{IC}_\mu(\pi)}\frac{\mathrm{CC}(\pi)/\beta}{\gamma}\Big)$$

3.2 Proof of Theorem 2

Our analysis of the protocol τ defined above will yield Theorem 2.

Lemma 5 *Let* $\mathbf{R}$ *be the random variable for the public randomness used in* π. *Let* π_r *be the (deterministic) protocol* π *with random bits fixed to be* r.

$$\mathrm{IC}_\mu(\pi) = \mathbb{E}_{\mathbf{R}}[\mathrm{IC}_\mu(\pi_{\mathbf{R}})]$$

Proof Let μ be some distribution on $\mathbf{X}$, $\mathbf{Y}$. Let $\mathbf{R}$ be the random variable for the randomness used in protocol π.

$$\begin{aligned}
\mathrm{IC}_\mu(\pi) &= \mathrm{I}(\mathbf{X};\pi(\mathbf{X},\mathbf{Y})|\mathbf{Y}) + \mathrm{I}(\mathbf{Y};\pi(\mathbf{X},\mathbf{Y})|\mathbf{X}) \\
&= \mathrm{I}(\mathbf{X};\mathbf{R},\pi_{\mathbf{R}}(\mathbf{X},\mathbf{Y})|\mathbf{Y}) + \mathrm{I}(\mathbf{Y};\mathbf{R},\pi_{\mathbf{R}}(\mathbf{X},\mathbf{Y})|\mathbf{X}) \\
&= \mathrm{I}(\mathbf{X};\mathbf{R}|\mathbf{Y}) + \mathrm{I}(\mathbf{Y};\mathbf{R}|\mathbf{X}) + \mathrm{I}(\mathbf{X};\pi_{\mathbf{R}}(\mathbf{X},\mathbf{Y})|\mathbf{Y},\mathbf{R}) + \mathrm{I}(\mathbf{Y};\pi_{\mathbf{R}}(\mathbf{X},\mathbf{Y})|\mathbf{X},\mathbf{R}) \\
&= \mathrm{I}(\mathbf{X};\pi_{\mathbf{R}}(\mathbf{X},\mathbf{Y})|\mathbf{Y},\mathbf{R}) + \mathrm{I}(\mathbf{Y};\pi_{\mathbf{R}}(\mathbf{X},\mathbf{Y})|\mathbf{X},\mathbf{R}) \\
&= \mathbb{E}_{\mathbf{R}}[\mathrm{IC}_\mu(\pi_{\mathbf{R}})]
\end{aligned}$$

The key insight is that the randomness $\mathbf{R}$ is independent of the distribution μ.

3.2.1 Expected Number of Mistakes

Lemma 6 *For any two distributions* A *and* B, $D(A||B) \geq |A-B|^2$.

Proof: homework.

Lemma 7 *The expected number of mistakes in simulating* π *is bounded by* $\sqrt{\mathrm{CC}(\pi) \cdot \mathrm{IC}_\mu(\pi_r)}$.

Proof There is a mistake at level i only when $\mathrm{Pr}_v^A(1) \leq k < \widehat{\mathrm{Pr}}_v^B(1)$ (WLOG suppose Alice owns node v, the node they have both reached at depth i).

Using Lemma 6 and Cauchy-Schwartz yields Lemma 7. See the paper for full details.

3.2.2 Finding the First Bit where the Transcripts Differ

Lemma 8 *There is a randomized public coin protocol τ with communication complexity $O(\log(k/\epsilon))$ such that on input two k-bit strings x and y, it outputs the first index $i \in [k]$ such that $x_i \neq y_i$ with probability $\geq 1 - \epsilon$ if such i exists.*

Proof The two parties use hashing and binary search to locate i.

Defining the tree. Fixing some constant C, define a labelled binary tree of depth $C \log(k/\epsilon)$ as follows. Each node will be labelled with the span of indices which parties *think* contains the earliest index $i \in [k]$ such that $x_i \neq y_i$. The root is labelled with the interval $[1, k]$. For j from 0 to $\log k - 1$, every node at depth j labelled $[a, b]$ has two children, each labelled with half the interval: $[a, b - k/2^{j-1}]$ and $[a + k/2^{j-1}, b]$. Every node at depth $\geq \log k$ has exactly one child, with the same label as its parent.

This binary tree will serve as the protocol tree for τ, with one notable modification: since τ is randomized and may make errors, the players are allowed to walk *up* the tree, effectively "taking back" a previous round.

Protocol. Starting at the root of the tree, the protocol τ works as follows:

1. The players are at some node v labelled $[a, b]$ in the tree.
2. The players check whether $i \in [a, b]$, that is, whether their previous moves were correct. They use public randomness to pick hash functions and use these to compare the prefixes of x and y of length a and b, respectively.

   If the tests indicate that $i \in [1, a]$ or $i \notin [1, b]$, then the players move to the parent of v.

   Otherwise, the players again use hash functions to decide which child of v to switch to. If these tests are inconsistent, the players stay at v.
3. Repeat step two $C \log(k/\epsilon)$ times.
4. If the final vertex is labelled $[a, a]$, output a. Otherwise conclude that $x = y$.

Analysis. Step two takes constant communication and is repeated $C \log(k/\epsilon)$ times, giving the desired communication complexity for protocol τ.

If $x = y$, then the protocol is always correct.

Suppose $x \neq y$ and i is the first index where they differ. As long as the protocol finishes at some node labelled $[i, i]$ it will be correct; there are many of these nodes, in a long chain, at the bottom of the tree. Consider directing the edges of the tree to be pointed towards this chain of nodes.

Notice that at each step, the players move from v to a node w which is *closer* to this chain with probability at least $2/3$.[1]

The protocol succeeds as long as the number of correct steps on the tree is at least $\log k$ plus the number of incorrect steps. Let A be the number of correct, and B the number of incorrect steps ($A + B = C \log(k/\epsilon)$). We want:

$$A \geq B + \log k$$

[1]Or some other probability $> 1/2$, depending on how the hash functions are selected.

The expected value of A is $\frac{2}{3}C\log(k/\epsilon)$. The protocol errs only when A differs from this expectation by $\frac{1}{6}C\log(k/\epsilon) - \frac{1}{2}\log k$. Using Chernoff bounds, we can set C to be large enough that the probability of erring is at most ϵ.