15-859: Information Theory and Applications in TCS **CMU: Spring 2013**

Lecture 18-19: Communication complexity lower bounds

April 2 & 4, 2013

Lecturer: Venkatesan Guruswami *Scribe: Arda Antikacioglu*

1 Lower Bounds for Deterministic CC

In the previous lecture we discussed the fooling set method for proving lower bounds on deterministic communication complexity. In this lecture we'll see two more ways we can prove these lower bounds: the Rectangle Size Method and the Rank Method. Both of these are stronger than the fooling set method. We'll show $\Omega(n)$ lower bounds for the dot product function with both methods while the fooling set method can only give $\Omega(\log(n))$. However, the two new techniques aren't necessarily comparable among themselves.

1.1 Rectangle Size Method

The rectangle size is another tool in our arsenal to prove lower bounds for deterministic communication complexity. Recall that for a given function $f : X \times Y \to \{0,1\}$, M_f is the 0/1 matrix where the $(x,y)^{th}$ corresponds to $f(x,y)$. The basic idea of the next bound is that if the size of a monochromatic rectangle is low, then any covering by of M_f by monochromatic rectangles needs quite a few rectangles which need a good number of bits to even index.

Lemma 1.1. *If every monochromatic rectangle has size $\leq s$, then $D(f) \geq \log 2^{2n}/s = 2n - \log(s)$*

Example 1.2. Let $DP(x,y) = x \cdot y = \sum_i x_i y_i \pmod 2$ be the inner/dot product function. We will lower bound its communication complexity with the rectangle size method. Consider a monochromatic rectangle $R = A \times B$ which means that $\langle a, b\rangle = 0$ for all $a \in A$, $b \in B$. Without loss of generality, we can consider the the spans of A and B as the dot product of linear combinations of elements from A and B will still be 0 by the bilinearity of the dot product. So letting $r = \text{rank}(A)$ and $s = \text{rank}(B)$ we can bound the sizes of A and B as $|A| = 2^r$ and $|B| = 2^s$. This means that $|R| = |A||B| = 2^{r+s}$. Since A and B are orthogonal by definition, by the rank nullity theorem we must have $r + s \leq n$ which shows that the size of a monochromatic rectangle is at most 2^n. The complete our lower bound, we note that at least half the entries in M_{DP} are 0s so that $D(\text{DP}) \geq \log \frac{2^{2n-1}}{2^n} = n - 1$. If you're a bit more careful, we could have shown the $n + 1$ lower bound.

We can also generalize the idea above so that the size of a rectangle can be defined according to any measure. In the example we did above we used the uniform measure, but other ones can be useful as well:

Lemma 1.3. *If there exists a measure μ on $X \times Y$ such that $\mu(R) \leq \delta$ for every monochromatic rectangle R in M_f, then $D(f) \geq \log_2 1/\delta$.*

Exercise 1.4. Prove that $D(\text{DISJ}) \geq \Omega(n)$ using the method above by bounding the size of a 1-monochromatic rectangle by 2^n.

1.2 Rank Method

Lemma 1.5. *Let $f : X \times Y \to \{0,1\}$ be a function and M_f its matrix representation. If $D(f) \leq c$ then $rank_F(M_f) \leq 2^c$ where the rank may be taken over any field. It follows by contrapositive that $D(f) \geq \log_2(rank(M_f))$.*

Proof. Suppose that there's a deterministic protocol which has communication complexity is at most c. Then there exists a partition of M_f into 2^c 1-monochromatic rectangles. Let these rectangles be $R_1, \ldots R_l$ for some $l \leq 2^c$. For each rectangle define a matrix M_i where the x, y entry is 1 if x, y is in R_i. Now we have $M_f = \sum_{i=1}^{l} M_i$. Clearly each of the M_is has rank at most 1 since each M_i only has one type of non-zero row. Since rank is subadditive,

$$\text{rank}(M) \leq \sum_{i=1}^{l} \text{rank}(M_i) = l \leq 2^c$$

□

A more careful analysis gives $D(f) \geq \log_2(2\text{rank}(M_f) - 1)$, but this is at most 1 more than the bound given above. Furthermore, since we only used the fact that rank is subadditive which is true for any field, we can work in any field when using this theorem. There's really only 2 natural choices for the field though. We can use $\mathbb{F}_2$ for convenience, or we can use $\mathbb{R}$ to obtain the best lower bounds since rank is maximized over the reals. Below, we do two examples to illustrate the method.

Example 1.6. For $f = \text{EQ}$, we have $M_f = I_{2^n}$ which has rank 2^n over any field, so $D(\text{EQ}) \geq n$

Example 1.7. We will obtain a $\Omega(n)$ lower bound for the dot product function. While it would be nice to work over $\mathbb{F}_2$, this actually gives us a terrible lower bound of since M_{DP} is a gram matrix. This means that the rank is n, and the lower bound we get from this method would be $\Omega(\log(n))$. Instead we'll work over reals. We claim that the rank over $\mathbb{R}$ is at most $2^n - 1$. We'll do this by switching from the $0/1$ matrix M_{DP} to the $1/-1$ matrix $M_{\widetilde{\text{DP}}}$ where the x, y entry is $(-1)^{x \cdot y}$. This new matrix can be written as

$$M_{\widetilde{DP}} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n}$$

Since $\text{rank}(A \otimes B) = \text{rank}(A)\text{rank}(B)$ and the rank of the 2×2 matrix given above is 2, we easily get $\text{rank}(M_{\widetilde{DP}}) = 2^n$ by induction. However $M_{\widetilde{DP}}$ is a simple linear transformation of M_{DP}. In particular we can write $M_{\text{DP}} = J - 2M_{\widetilde{\text{DP}}}$ where J is the all 1s matrix which has rank 1. This means that M_{DP} has rank at least $2^n - 1$ over $\mathbb{R}$, so $D(\text{DP}) \geq \Omega(n)$ as claimed.

Exercise 1.8. As an exercise, we can prove that $\text{rank}(M_{\text{DISJ}}) \geq 2^n$

Since, this is a good lower bound people have wondered if the rank of M_f can be used to upper bound communication complexity too? The best people have done is show that $D(f) \leq O(\text{rank}(M_f))$ which still leaves an exponential gap between the lower bound we showed above and the best upper bound. Log-rank conjecture by Lovàsz and Saks states that this is way off and that for any f we actually have $D(f) \leq O(\text{rank}(M_f))^c$ for some $c > 1$. The best construction gives a function for which $c = \log_3(6) \approx 1.631$ is needed. That is, very little progress has been made in either direction and people won't be shocked if the conjecture is false.

2 Randomized Communication Complexity

2.1 Private Coin Model

This is the first of our 3 different models for randomized communication complexity. Alice has one part of the input x and has the other part y. Alice and Bob both have access to their own sources of randomness r_A and r_B respectively which the other one can't peek into. We'll assume that r_A and r_B are sampled according to the distributions π_A and π_B respectively. Protocol Π computes a function $f : X \times Y \to \{0, 1\}$ with error $\epsilon > 0$ if for all x and y, the probability over r_A and r_B of having $\Pi(x, y) = f(x, y)$ is at least $1 - \epsilon$. In this model, we assume that the inputs are arbitrary (that is, they're no drawn from a distribution).

We can think of a randomized protocol in terms of the decision trees we used for deterministic communication complexity by replacing Alice's decision function at each step $a_v(x)$ with $a_v(x, r_A)$ (same with Bob). When we include randomness, the path we take in the protocol is a random variable. Then cost of Π on x, y is the maximum number of steps you can take using this protocol regardless of which random strings are used. The cost $CC(\Pi)$ is the maximum over all x, y, r_A, r_B of the number of steps we in this randomized decision tree.

For our final definition, we let $R_\epsilon(f)$ be the minimum cost $C(\Pi)$ of a protocol Π that computes $f(x, y)$ with error at most ϵ. If we drop the ϵ from the notation, we will be talking about error at most $1/3$.

Given all these definitions, we can prove our first result that shows that randomized decision protocols can do much better than deterministic ones. We showed above that $D(\text{EQ}) \geq n$, which means that the trivial protocol of Alice sending Bob her input and Bob sending back whether they are equal is best possible. It turns out that we can do exponentially better in a randomized setting.

Lemma 2.1. $R_{1/3}(EQ) \leq O(\log(n))$

Proof. We first encode the Alice's string as $A(z) = \sum_i a_i z^i$ and Bob's string as $B(z) = \sum_i b_i z^i$. After this encoding, the protocol works as follows:

- Alice picks a prime p such that $n^2 \leq p \leq 2n^2$.
- Alice uniformly randomly samples θ from $0, \ldots, p - 1$.
- Alice sends $p, \theta, A(\theta) \mod p$
- Bob accepts if and only if $B(\theta) = A(\theta) \mod p$

If $a = b$ Bob always accepts correctly. If $a \neq b$, then $\mathbf{Pr}[A(\theta) = B(\theta)] \leq n/p \leq 1/n$, so we're correct with high probability. Alice sends three strings of length at most p, which can be written in $\log_2 p \leq O(\log(n))$ bits, the cost of the protocol is at most $O(\log(n))$ bits in total. This completes the proof. □

We picked polynomials because we like them but functions could work too. We couldn't have used a very large hash family since Alice needs to send Bob which hash function she used along with the output of that function, but a polynomial sized family with a polynomial sized output would work just fine and give the same bound.

Unfortunately, there's limits to how much of an improvement we can get from randomization. The gap between the deterministic and private coin complexity of a function cannot be more than exponential. The proof of the following lemma is in the book, but the main idea is that we can simulate a private coin protocol by a deterministic one if Alice and Bob exchange the probabilities that they end up down a certain branch in the randomized decision tree. The analysis needs to be done with some care however since we have to encode real numbers by rationals, so we need to keep track of truncation errors.

Lemma 2.2. $R_{1/3}(f) \geq \Omega(\log(D(f)))$

Corollary 2.3. $R_{1/3}(EQ) = \Theta(\log n)$

2.2 Public Coin Model

Private coin models might capture the essence of communication, but public coin models are much nicer to analyze. Public coin protocols are a much nicer model. In this model Alice and Bob have a shared source of randomness r instead of having their own sources of randomness. We can think of the random bits as coming from an infinite tape of randomness that's sampled from some distribution as in the private coin model. However, a much more fruitful way of thinking about public coin models is as a distribution over deterministic protocols $\Pi(r)$ indexed by r. Once we set r, the randomized protocol becomes a deterministic one. So we can think about such a protocol as setting the randomness as the start and not lose any generality since

$$\mathbf{Pr}[\Pi(x, y) = f(x, y)] = \mathop{\mathbf{Pr}}_{\Pi^{(r)}}[\Pi^{(r)}(x, y) = f(x, y)]$$

We define the error for a protocol Π to be $\max_{x,y} \mathbf{Pr}_{\Pi^{(r)}}[\Pi^{(r)}(x, y) = f(x, y)]$ as in the private coin model. Once again, the inputs are assumed to be arbitrary and the only source of randomness is the public pool of coin tosses. We denote by $R^{pub}_{\epsilon}(f)$ the lowest cost public coin protocol that computes f with error at most ϵ. If we drop the ϵ, the error is assumed to be $1/3$. The first thing to note about public coin protocols is that they are at least as good as private coin protocols.

Theorem 2.4. $R^{pub}_{\epsilon}(f) \leq R_{\epsilon}(f)$

Proof. We can take any private coin protocol and turn it into a public coin protocol by letting the randomness in the public coin protocol come from the product distribution on r_A and r_B with the same error bound. □

However, in the best case, they can be much better than private coin protocols.

Example 2.5. Consider the equality function. In the private coin protocol we needed to send the hash function itself too, so we could use at most a polynomial sized family of hash functions. In the public coin model, we can pretend like the hash function is fixed in advance so there's no need for Alice to send it to Bob during protocol. This lets us use an exponential family of hash functions. In particular, we can give the following 2-bit protocol for equality. Let Alice and Bob sample an n-bit string r from the shared pool of randomness and exchange dot products. If $x = y$, then the protocol is always correct since $\langle x, r\rangle = \langle y, r\rangle$ in this case. However, if $x \neq y$ then $x - y \neq 0$ and $\langle x - y, r\rangle = 0$ with probability exactly $1/2$. So if Alice and Bob sample two random n-bit strings r and r' and run the protocol twice, the protocol has error exactly $1/4$ which shows that $R^{pub}(\text{EQ}) \leq 2$.

This example makes public coins look too wondrous and unrealistic. In reality, they are not too far from private coin protocols. For a fixed ϵ, a public coin is at most an additive logarithmic factor better than a private coin protocol.

Theorem 2.6 (Newman's Lemma)**.** $R_{\epsilon+\delta}(f) \leq R^{pub}_{\epsilon}(f) + O\left(\log \frac{n}{\delta}\right)$

Proof. We need to convert a public coin protocol into a private coin protocol. The obvious idea would be for Alice to sample the maximum number of bits required to carry out the protocol from her source of randomness r_A and send this string to Bob. This way Alice and Bob would have access to the same string of randomness and could carry out the public coin protocol. We will do exactly this, but first we need to reduce the amount of randomness used by the public coin protocol since this cost will be added to the cost of communication in the private coin protocol.

Suppose there existed t strings $r_1, \ldots, r_t$ where $t = \text{poly}(\frac{n}{\delta})$ such that the public coin protocol error was at most $\epsilon + \delta$ when we uniformly sampled a random string from among these choices. Then these strings could be fixed in advance, Alice would only need to index into this sequence and it would cost her only $O(\log \frac{n}{d})$ bits of communication. So we will prove that such a sequence of random choices exists.

The key to proving this is using Hoeffding/Chernoff bounds. For a fixed x, y, the probability that the sampled error deviates by more than δ from the real error on t samples of r is at most $2^{\Theta(t)\delta^2}$. If we take $t = 10n/\delta^2$, the probability of this event becomes at most 2^{-3n}. There are only 2^{2n} pairs of inputs (x, y), so by a union bound the probability that a sequence of random strings $r_1, \ldots, r_t$ has mean error at least $\epsilon + \delta$ for some pair of inputs x, y is at most 2^{-n}. By the probabilistic method there must now exists some sequence of random strings that has error at most $\epsilon + \delta$ on all inputs, which proves the claim. □

2.3 Distributional Communication Complexity

Finally we have a third model, but this one is interesting because it allows us to prove lower bounds on the public coin model. In contrast to the previous two models, the protocol itself is deterministic and the randomness comes from the inputs. So far we had only considered the worst case over all inputs. The formal definition is that we have a deterministic protocol Π that computes a function f and a probability distribution μ over the input $X \times Y$. The error of the protocol is defined to be

$err^{\mu}(\Pi, f) = \mathbf{Pr}_{x,y\sim\mu}[\Pi(x,y) \neq f(x,y)]$. However, while the error is defined using the distribution over the inputs, the cost of communication is not. The cost of protocol is still the maximum cost over all possible inputs x and y. Similar to the models above, we define $D^{\mu}_{\epsilon}(f)$ to be the minimum cost of a protocol Π that achieves error at most ϵ over a distribution μ. Distributional complexity can lead to some interesting consequences.

Example 2.7. Let μ be the uniform distribution. Then the probability that x and y sampled according to μ are equal is 2^{-n}. Therefore, to achieve constant error with EQ in this model we don't even need to see the inputs. We can simply output 0 for every output.

Example 2.8. Again with the uniform distribution, DISJ can be computed with constant error with 0 bits of communication. The support for two n-bit numbers are disjoint none of their bits are 1s at the same time. Each pair of bits are not 1 at the same time with probability 3/4, so the probability that the supports of two random strings are disjoint is at most $(3/4)^n$. Once again, we don't even have to look at the input.

These two examples show that in order to get interesting results out of this model we need to pick the input distribution so that $f(x,y)$ evaluates to 0 or 1 with about equal probabilities.

Example 2.9. Consider DISJ again, but this time suppose we sample the bits of x and y from a distribution which picks 1 with probability $1/2$. We picked $1/\sqrt{n}$ because by the birthday paradox DISJ(x,y) will be 0 and 1 with about equal probability. In this case we can use protocol where Alice sends Bob the indices of the indices of the first $3\sqrt{n}$ 1s in x. Bob can then decide the problem himself. The expected number of 1s in x is $\sqrt{n}$, so by Markov's inequality (we can use Chernoff for even better bounds, but it's unnecessary) the probability that Alice won't be able to send all of her input is at most 1/3. In the cases where this doesn't happen, she sends a full description of her input and Bob decides the problem correctly. It takes $\log_2 n$ bits to encode each index so we have $D^{\mu}(f) \leq 3\sqrt{n}\log_2(n)$ for this distribution.

Finally, we prove a theorem that shows that regardless of which distribution we use, the distributional complexity of a function is a lower bound on the public coin communication complexity of a function. Of course, the important part of using this theorem is picking the distribution in a way that leads to a high enough distributional complexity.

Lemma 2.10. *$R^{pub}_{\epsilon}(f) \geq D^{\mu}_{\epsilon}(f)$ for any distribution μ over $X \times Y$*

Proof. Given a public coin protocol as a distribution over deterministic protocols $\Pi^{(r)}$, we know that for any x, y

$$\mathbf{Pr}_{r}[\Pi^{(r)}(x,y) \neq f(x,y)] \leq \epsilon$$

which implies that for any μ.

$$\mathbf{Pr}_{(x,y)\sim\mu,r}[\pi^{(r)}(x,y) \neq f(x,y)] \leq \epsilon$$

but by the probabilistic method, there exists some r such that

$$\mathbf{Pr}_{(x,y)\sim\mu}[\Pi^{(r)}(x,y) \neq f(x,y)] \leq \epsilon$$

which completes the proof. □

In fact $R^{pub}_{\epsilon}(f) = \sup_{\mu} D^{\mu}_{\epsilon}(f)$ which follows from the von Neumann/Yao min-max principle.