CS 229r Information Theory in Computer Science Mar 5, 2019

Lecture 11

Instructor: Madhu Sudan *Scribe: Luke Minton*

1 Administrative

Problem Set 3 is out and will be due Friday, March 15. Madhu will have Extra Office Hours this week from 1 - 3 PM on Friday. Start thinking about the final project; you're welcome to choose anything for the projects as long as it uses information theory. Topics for the final project are forthcoming; start looking for partners if you haven't already so you can work on project during Spring Break.

2 Overview

- Communication Complexity
- Low Bounds for IP
  - Distributional Complexity
  - Discrepancy
  - Spectrum
- Disjointness

3 Communication Complexity

Recall the model - we have two players, Alice and Bob, who are communicating but have private inputs. Alice has input x, Bob has input y, each is an element of $\{0,1\}^n$. We want to compute some function f from $\{0,1\}^{2n} \to R$. Usually we want to compute boolean outputs, making our $R = \{0,1\}$, but sometimes it is some index in n so $R = [n]$; other times we only partially specify the function. We will consider cases where both Alice and Bob have access to shared randomness R as well as private randomness R_A, R_B. (the distinction doesn't seem to be quite important right now, but we will find cases where private randomness is helpful). The deterministic communication complexity of f is

$$CC(f)_{\pi s.t. \pi \ computes \ f} = \min\{\#bitsexchanges\}$$

We also have notions of random communication complexity. $CC^{Priv}(f)$ if we have private randomness, $CC^{Pub(}f)$ are the communication complexities corresponding to private and public randomness respectivies.. The obvious inequalities should be that

$$CC^{Pub}(f) \leq CC^{Priv}(f) \leq CC(f)$$

Each type of the right is also one on the types on the left. However we also have inequalities going the other direction; the first is:

$$CC^{Priv}(f) \leq CC^{Pub}(f) + O(\log n)$$

Another is

$$CC(f) \leq 2^{O(CC^{Priv})}$$

The first of these is an exercise in the union bound (take n random strings and use Chernov bounds). The second is much simpler, we just communicate the entire distribution of messages with private randomness. i.e. If I was specifying private randomness, this the kind of messages that would be sent - increases exponentially.

The essential objects in the analysis of polar codes are martingales. Let $x_0, \ldots, x_t$ be a $[0,1]$-bounded Martingale. There are two ways to characterize the polarization:

4 A Brief Aside

4.1 Hamming Distance

Before we examine lower bounds, let's look at some interesting problems. The first of these is the Hamming Distance problem. Hamming $Dis_k(x,y) = 1$ if $\Delta(x,y) \leq k$, $Dis_k(x,y) = 0$ if $\Delta(x,y) > k$, where here $\Delta(x,y) = \#\{i | x_i \neq y_i\}$. This can be done in $\Theta(k \log k)$ bits with shared randomness. This shouldn't be too much of a surprise because EQ is a special case. (set k equals zero). The important thing is that there is no dependence on n. if you can do this well - without randomness, you can do it to constant amount of communication, which is very, very interesting.
Maybe an exercise here to prove this

Moving for a second, there is a similar problem is small set disjointness. Alice gets a subset $S \subset n$, Bob gets $T \subset n$ The question $S \cap T =?$ is a hard question and takes $\Omega(n)$ communication. We will deal with this harder question more at a later time. However, today we consider case where size $|S|, |T| \leq k$ for a small k. We both have very small sets and want to know if they are overlapping or not. We have a really nice protocol that runs in $\Theta(k)$ communication. We will show something a bit weaker today, which is $\Theta(k \log k)$ This is interesting because there is no n in the answer. Even though we are dealing with elements of a very large universe, the size of the universe does not play a role. (The heuristic explanation behind what we are doing here is that we are restricting our elements from the large universe of size n to a much smaller one of size k) We will use randomness to create a really nice hash function. we use $h : [n] \to [k^2]$. We don't care how much randomness we use here, so we use as much as we can! That is, we pick h at random among all possible functions (and this function is publicly agreed on by Alice and Bob). Why k^2? This is the right number to avoid the birthday paradox. If there were k elements that we want to keep collision free, the k^2 hash functions do so with probability. We want to achieve a small error probability, say $1/100$ The feature that we want is that $Pr_h[\exists i \neq j \in W s.t. h(i) = h(j) \leq] 1/100]$. For small set disjointness, just send hash values of the elements, we will then apply to $W = S \cup T$ With high probability these things remain collision free. If Alice sends hashes of all elements. Bob would be able to see if there is something that matches his elements; if collision, likely they came from the intersecting case. If no collision then obviously they aren't. By deciding to hash everything down, we have elements of size k in universe of size k^2. High probability over the choice of h; that's how we can solve the small set disjointness problems. Again, Alice sends the hashes of her elements, $h(i)_{i \in S}$ to Bob. That is $\approx k \log k$ in complexity; there is more clever one in k, but that is a paper.
Exercise: Show this

Then, how do we measure Hamming Distance? If we have universe of size n and our hash function h; we have k^2 buckets $h(i) = 0$,$h(i) = 1 ... h(i) = k^2$; so we have things in up to k^2 buckets; Alice calculates $u_j = \oplus x_i, i | h(i) = j$, that is the parity of the number of elements hashing to each bucket. This is what she sends to Bob. Bob performs the analogous calculations on his y_i bits, say $v_j = \oplus y_i, i | h(i) = j$ - if $\Delta(x,y) \leq k$, the x_i, y_i differ in $\leq k$ indices which means that $\Delta(u_i, v_i) \leq k$ as well (since each x_i, y_i is hashed to no more than one bucket). Conversely, if $\Delta(u_i, v_i) > k$, then $\Delta(x,y) > k$ with high probability as well, giving us a $\Theta(k^2)$ protocol (this could be improved with binary search).
Exercise: Show this

4.2 Squared Distance

All of the problems we have considered, and most of the problems we usually consider, use Boolean functions. Therefore, let's describe at least one case where we have real numbers as inputs:
Alice gets $x \in \mathbb{R}^n$,$||x||_2 = 1$ Bob gets $y \in \mathbb{R}^n$,$||y||_2 = 1$. Note: these norm constraints aren't used directly in our proof, but the constraint allows us to have a well-defined sense of the relative error bound.. Let $f(x, y) = \sum x_i - y_i \pm \varepsilon$. Alice Discretizes all her bits (so we aren't sending an infinite number of bits) and sends $(\sum_i x_i) \pm \varepsilon$ it to him. This should take around $\log(1/\varepsilon)$ bits of communication (as we get ever more precise, we require more bits). This is quite easy, so let's consider a harder problem. We want to compute $\sum_i (x_i - y_i)^2 \pm \varepsilon$ (this is just the squared Euclidean distance between two points). This is less trivial - clearly we do not want to send the entire x; that's too much communication. Let's pick n random "bits" $R_1 ... R_n$, representing the R_i not as elements of $\{0, 1\}$ but rather as elements of $\{-1, 1\}$ (we'll see this useful technique more below), and Alice will send to Bob $(\sum (x_i)^2) \pm \varepsilon$ (we're sending discretization, so $\pm\varepsilon$, this isn't the big cost here). Alice will send $\sum_{R_i} x_i$. Now, if we consider $\sum_i (x_i - y_i)^2$ and expand this sum into $\sum_i x_i^2 - 2x_i y_i + y_i^2$, we note that Alice is sending the left-hand term and Bob will already have the right-hand term. Our claim is going to be the following: $E_R((\sum_{R_i} x_i \sum_{R_j} y_j)) = \sum (x_i y_i)$ which will allow Bob to finish calculating the sum. $E_R((\sum_{R_i} x_i \sum_{R_j} y_j)) = E_R(\sum_i R_i^2 x_i y_i) + E_R(\sum_{i \neq j} R_i R_j x_i y_j)$ The same terms just evaluate to $\sum_i x_i y_i$ since R_i^2 is always 1; conversely, the cross terms $E_R(\sum_{i \neq j} R_i R_j x_i y_j)$ sum to 0 by symmetry. Therefore, our protocol should give us a bound of $1/\varepsilon$ for our communication complexity.
Exercise: Use the disjointness lower bound to prove the $1/\varepsilon$ lower bound for communication complexity here. It is actually possible to show a stronger bound of $1/\varepsilon^2$, but this requires Gap Hamming, which is not something that will be covered in the course.

5 Inner Product (IP) Lower Bound

We consider the inner product function: $IP(x, y), x, y \in \{0, 1\}^n$. We define:

$$IP(x, y) = \bigoplus_i x_i \oplus y (mod\ 2)$$

Here is a place where we believe randomness doesn't really help us. Why is this? We have seen in the past that the rank method sometimes doesn't help us. For example, for EQ, the identity matrix gets us what we need; but that matrix has a large rank which doesn't do much to bound the communication complexity. How do we prove that $\Omega(n)$ is a lower bound for communication? We will need a more "robust" metric in order to help us do so. We will want to do this with public randomness. . Proving the lower bound on this is not trivial, so here we have a few ideas at play that help us out.

- **Distributional Lower Bounds** In principle our lower bounds could be worst case lower bounds, (selecting a pathological x, y). A distributional lower bound specifies a distribution; i.e. we specify a distribution and find that nothing works well. We specify distribution μ over x, y and claim that no deterministic protocol π achieves small error on $(x, y) \sim mu$.

- **Error on a distribution** How do we define error on a distribution? It is $\delta_m(f, g) = Pr_{x,y \sim \mu}[f(x, y) \neq g(x, y))$. Our claim is equivalent to $\delta_\mu(f, pi) > \varepsilon$. Why does this come up? Let's try to illuminate the link between distributional lower bounds over deterministic protocols and lower bounds over randomized protocols. Suppose our protocol π does very well on every x, y and gets error less than $1/2$, say a $1/3$. If we have something that achieves that on every x, y, then we can repeat protocol $\log(1/\varepsilon)$ times and reduce the error. so let's call our protocol a k bit communication protocol. There exists a $O(k \log(1/\varepsilon))$ bit randomized protocol that gets error less than ε on every x, y. So that's simple; so now we have protocols that make small error (but are still making error). But these are randomized protocol! and we don't want to work with these. So we consider distribution μ, so if it's true for every x, y, $E_R[\pi_{f(x,) \neq pi'(x,y,R)}] < \varepsilon$, where R is the randomization we are using in our protocol and

$pi'(x, y, R)$ is what our randomized protocol is once the randomization has been specified.This is true for every x, y, but we can put this over any expectation over (x, y) - we just sum up over the relevant probabilities. Therefore, we have $E_{x,y}E_R[\pi_{f(x,)\neq pi'(x,y,R)}] < \varepsilon$; but we can exchange the $E_{x,y}$ and the E_R. And the resulting gives us in the inner term $E_{x,y}[\delta(f(\pi_r))]$ π_r is a protocol with the randomness hardwired to $R = r$. This means there's an analogous deterministic protocol which achieves the ε (or else the randomized protocol wouldn't be able to achieve the bound when in expectation over R). This makes us realize an important connection: A randomized protocol is just a distribution over deterministic protocols. Therefore there exists an r such that the protocol pi_r satisfies $\delta_{x,y\sim\mu}(\pi_r) < \varepsilon$ This allows us to go to deterministic protocols; however, our protocol unfortunately is wrong an ε fraction of the time. In some sense, picking the distribution may make the problem harder than dealing with pure worst case bounds; but, the fact remains that so far this has been helpful in many cases to find lower bounds, and it is what we will do here.

- **Distributional Complexity** We will define $D_{\mu,\varepsilon}(f)$ to be the distributional communication complexity of computing f with error ε over inputs from μ. That is, over all protocols pi_ithat always send no more than k bits, and permitting an error ε, $D_{\mu,\varepsilon}(f) = \min_{\pi_i} \#bits\ exchanged\ by\ \pi_i$. We want to prove that this lower bound is linear in n, $D_{\mu,\varepsilon}(IP) = \Omega(n)$. But what distribution do we want to deal with? But here, the thing that makes the problem so easy is the fact that we can just use mu as the uniform distribution, $1/4^n$ in each entry corresponding to the x, y (there are 2^n possible strings each for Alice and Bob so 2^{2n} possible combinations for the inputs). First time we looked at deterministic protocols, we saw how communication of bits separates our matrix into pieces; the final pieces that we see should have all values either 0 or 1. Note: WLOG the final bit communicated by π is the value of the function. (this adds no more than one round of communication, so we won't care about that much difference today). If we consider our protocol, what happens to the matrix? This matrix is going to get partitioned into smaller pieces. After a while, we are left with a bunch of small rectangles; the protocol is constant within each rectangle. Lets label these rectangles $R_1..R_K$ (remember when we say rectangles, we do not mean necessarily, geometrically contiguous, just that they satisfy the product definition). Some fraction of rectangles has correct values; some fraction has errors. So let's say we measure everything in absolute numbers: $p1...p_K$ are correct fractions in rectangles $\varepsilon_1...\varepsilon_k$ are error fraction in rectangles. We easily arrive at $\sum_i \varepsilon_i \leq \varepsilon$. $\sum p_i \geq 1 - \varepsilon$ This means $\sum pi - ei \geq 1 - 2\varepsilon$. For the K rectangles, there must be at least one rectangle which at least achieves the average, so there is an R_i s.t. $pi - ei \geq \frac{1-2\varepsilon}{K}$ or $p_i - \varepsilon \geq \frac{1-2\varepsilon}{2^k}$ since there are 2^k rectangles (remember, this lowercase k is the number of bits communicated).

- **Modified Matrix** When we consider distributional complexity, it's helpful to talk about a slightly different matrix than the one we usually use (similar to what we used above for the Squared Distance problem). Confer last lecture, $f \to M_f$ s.t. that $M_{xy} = 0$ if $f(x, y) = 0$ and $M_{xy} = 1$ if $f(x, y) = 1$. Instead, we make our new matrix $M_{xy} = \mu(x, y)$ if $f(x, y) = 0$ and $M_{xy} = -\mu(x, y)$ if $f(x, y) = 1$ according to the distribution we have picked.

- **Discrepancy** There must exist a large monochromatic rectangle in M_f aka there exist $S, T \in \{0, 1\}^n$ such that $\mid \sum_{x\in S,y\in T} M_{x,y} \mid$ The maximum value that this takes over all rectangles is known as the Discrepancy. In the 0 error case, within each rectangle we have a constant value, so the absolute value of the sum of the $M_{x,y}$ is just the appropriate probability mass. Therefore must be at least one with 2^-k value (one must be at least as great as average). When we introduce ε error, we can analogously find S and T such that $\mid \sum_{x\in S,y\in T} M_{xy}\mid \geq \frac{1-2\varepsilon}{2^k}$ (the entire sum is now $1 - 2\varepsilon$ rather than 1, but there still must be at least one rectangle that achieves the average).

- **Lower Bound** The claim is that $D_{\mu,\varepsilon}(t) \leq k \implies Disc_\mu(f) \geq (1 - 2\varepsilon)/2^k$. We want some methods to upper bound the discrepancy (which then gives lower bound on k). This intuitively makes sense, for example EQ is a terrible function - large block of zeros are all constant, which is why we don't get a good bound. So we're going to try to make this a bit more linearly algebraic. From our Discrepancy definition, we recall that it is $\max_{S,T} \mid \sum_{x\in S,y\in T} M_{x,y} \mid$ however we can re-interpret summing over the

relevant indices in S, T as applying indicator linear operators I_S, I_P (that is, coordinate i is 1 iff $i \in S$) This gives us $\max_{U,V} U^{\perp} M V$ with $U = I_S V = I_P$. We have coordinate restrictions here (U,V can only have 0 or 1 at each index), and these are strange in linear algebra. However, norm restrictions are not strange! So we will expand the set, noting that $\max ||U||_2, ||V||_2 \leq 2^{n/2}$ (when all coordinates are included). so we can scale these vectors down and have this is $2^n \max_{||U||_2, ||V||_2 = 1} U^{\perp} M V = 2^n \lambda_{max}(M)$. We need this eigenvalue to be small, or else this doesn't really help us. We can show inductively that $M_{IP_N, \mu_N} = (M_{IP_1, P_1})^{\otimes n}$

Exercise: Show this The matrix for M_{IP_1, P_1} is

$$\begin{bmatrix} 1/4 & 1/4 \\ 1/4 & -1/4 \end{bmatrix}$$

since the IP of two bits is 1 (which is mapped to $-\mu(x, y)$) iff both bits are 1. Simple linear algebra shows us that $\lambda_{max}(M_1) = \pm 1/\sqrt{8}$. which gives us that $\lambda_{max}(M_N) = (1/8)^n/2 = \frac{1}{2^n} \frac{1}{2^{n/2}}$. Returning to our earlier formula, we have $Disc_{\mu}(f) \leq (1 - 2\varepsilon)/2^k \implies D_{\mu, \varepsilon} \geq k$ and our discrepancy is bound above by $2^n \frac{1}{2^n} \frac{1}{2^{n/2}} = 2^{-n/2}$ If we let $\varepsilon = 1/4$ and $k = n/2 - 1$ we have that $Disc_{\mu} f \leq (1 - 2\varepsilon)/2^k = \frac{1}{2^{n/2}}$ which implies that $D_{\mu, \varepsilon} \geq k = n/2 - 1$ Therefore, since a deterministic algorithm is $\Omega(n)$ for any μ, then $CC^{Pub}(IP) = \Omega(n)$ by the connection we drew earlier.