PRINCETON UNIVERSITY COS 522: COMPUTATIONAL COMPLEXITY

Lecture 18: Communication complexity

Lecturer: *Sanjeev Arora* Scribe: *Tony Wirth*

Although our discussion of communication is brief, it is a necessary one, as the concept will be useful in future lectures on circuit complexity.

We consider a function f that maps $2n$ bits, partitioned into two equal length halves, into a single bit. That is,

$$f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$$

In the two party (or player) model, each party has an n-bit string; say player 1 has x and player 2 has y. Neither party knows anything about the other's bit string. Both want to know $f(x,y)$, and to do this they communicate. (The parties are not adversaries; they help and trust each other.) The complexity of the function f is the number of bits that need to be exchanged for the worst case inputs. Each party has unlimited computational power.

Formally, a t-round communication protocol for f is a sequence of function pairs $(S_1, C_1), (S_2, C_2), \ldots, (S_t, C_t), (f_1, f_2)$. The input of S_i is the communication pattern of the first $i-1$ rounds and the output is from $\{1,2\}$, indicating which player will communicate in the ith round. The input of C_i is the input string of this selected player as well as the communication pattern of the first $i-1$ rounds. The output of C_i is the bit that this player will communicate in the ith round. Finally, f_1, f_2 are 0/1-valued functions that the players apply at the end of the protocol to their inputs as well as the communication pattern in the t rounds in order to compute the output. These two outputs must be $f(x,y)$.

As our example for this lecture, consider the equality function:

$$\mathrm{EQ}(x,y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}$$

We claim the (deterministic) complexity of EQ is in fact n. For contradiction's sake, suppose a protocol exists whose complexity is at most $n-1$. Then there are only 2^{n-1} communication patterns possible between the players. Consider the set of all 2^n pairs (x,x). Using the pigeonhole principle we conclude there exist two pairs (x,x) and (x',x') on which the communication pattern is the same. Of course, thus far we have nothing to object to, since the answers $\mathrm{EQ}(x,x)$ and $\mathrm{EQ}(x',x')$ on both pairs are 1. However, now imagine giving one player x and the other player x' as inputs. A moment's thought shows that the communication pattern will be the same as the one on (x,x) and (x',x'). (Formally, this can be shown by induction. If player 1 communicates a bit in the first round, then clearly this bit is the same whether his input is x or x'. If player 2 communicates in the 2nd round, then his bit must also be the same on both inputs since he receives the same bit from player 1. And so on.) Hence the player's answer on (x,x) must agree with their answer on (x,x'). But then the protocol must be incorrect, since $\mathrm{EQ}(x,x') = 0 \neq \mathrm{EQ}(x,x)$.

The lowerbound argument above is called a *crossing sequence* argument.

Now we give another way to perform the analysis. Consider the matrix of f, denoted $M(f)$, which is a $2^n \times 2^n$ matrix whose (x,y)'th entry is $f(x,y)$. See Figure 1. We visualize

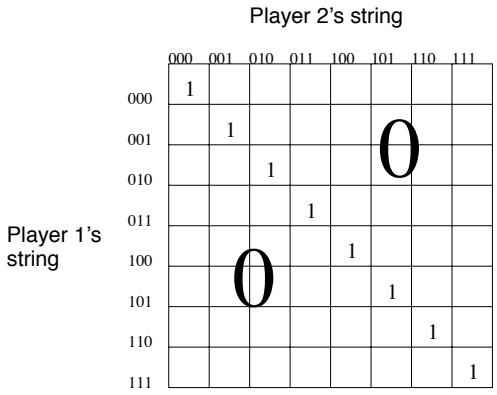

Figure 1: Two-way communication matrix, M(f), for the equality function with 3-bit inputs. The numbers inside the matrix are the values of f on the inputs.

the communication protocol in terms of this matrix. After each bit of communication, we divide the matrix into two parts, each of which is a rectangle. The rectangles represent the possible combinations of input strings that lead to a particular communication sequence. (Note that our definition of *rectangle* here is that $A \times B$ is a rectangle in $X \times Y$ whenever $A \subseteq X$ and $B \subseteq Y$.) For example, if the communication protocol has the first player sending one bit and then the second player sending one bit, then the communication matrix might look like Figure 2. After k steps, the matrix has been partitioned into 2^k rectangles. If the

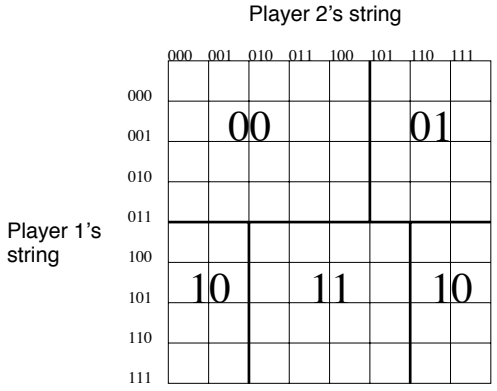

Figure 2: Two-way communication matrix after two steps. The large number labels are the concatenation of the bit sent by the first party with the bit sent by the second party.

protocol stops, then the value of $f(x, y)$ is determined, and must be a constant. Thus the communication (so far) must have led to a *monochromatic* rectangle—that is a rectangle with all ones or all zeros. (Once again, denoting the rectangle by $A \times B$, the rectangle is *monochromatic* if for all x in A and y in B, $f(x, y)$ is the same.) The adversary's strategy is to give inputs to the players so that at each stage of the protocol, they end up in the rectangle with the largest number of pairs of the form (z, z), for some z in $\{0, 1\}^n$. Since the number of 1s in the (equality) matrix is 2^n, if the number of steps is less than n, then under this strategy the rectangle is not monochromatic.

REMARK 1 There was a question in class about the model of communication. Can a player communicate by not saying anything? (After all, they have three options: send a 0, or 1, or not say anything in that round.) If this silence were allowed, then we would essentially regard the communication as having a ternary, not binary, alphabet. But the same lowerbound arguments would apply.

DEFINITION 1 *A monochromatic tiling of $M(f)$ is a sequence of disjoint monochromatic rectangles whose union is $M(f)$. We denote by $\chi(f)$ the minimum number of rectangles in any monochromatic tiling of $M(f)$.*

The following theorem is immediate from our discussion above.

THEOREM 1
If f has communication complexity C then it has a monochromatic tiling with at most 2^C rectangles.

It follows that the communication complexity of f is at least $\lceil \log \chi(f) \rceil$.

Now we introduce a way to lowerbound $\chi(f)$ (and hence communication complexity). Recall the high-school notion of *rank* of a square matrix: it is the size of the largest subset of rows/colums that are independent. The following is another definition.

DEFINITION 2 *The* rank *of an $n \times n$ matrix M is the minimum value of l such that M can be expressed as*

$$M = \sum_{i=1}^{l} \alpha_i B_i,$$

where $\alpha_i \in \mathbf{R}$ and each B_i is an $n \times n$ matrix of rank 1.

The following theorem is trivial, since each monochromatic rectangle can be viewed as a rank 1 matrix (by filling out entries outside the rectangle with 0's).

THEOREM 2
For every function f,

$$\chi(f) \geq \mathit{rank}(M(f)).$$

Thus in this lecture we have covered three main ways of obtaining a lower bound for communication complexity.

1. Applying a crossing sequence argument.
2. Lowerbound $\chi(f)$.
3. Lowerbound $\mathrm{rank}(M(f))$.

Method 1 is the strongest method, followed by Method 2 and the weakest is Method 3. For instance, having a minimal tiling provides an adversary strategy for a crossing sequence argument. See Exercise 1. Also, we can separate the power of these lowerbound arguments. For instance, we know functions for which there is a significant gap between $\log \chi(f)$ and $\log \mathrm{rank}(M(f))$.

In the late 1970s and early 1980s, communication complexity was used as a model for parallel computation; in particular, for modelling space/time tradeoffs. Yao pointed out that communication complexity could provide lower bounds for the resources used in a VLSI circuit. For instance, in a VLSI chip that is an $m \times m$ grid, if the communication complexity for a function is greater than c, then the time required to compute it is at least c/m.

We will not explicitly study randomized communication, except to note that randomization can significantly reduce the need for communication. For instance with with public random bits, we can use fingerprinting with random primes (explored in Lecture 13), allows us to compute the equality function by exchanging $O(\log n)$ bits: the players just pick a random prime p of $O(\log n)$ bits and exchange $x \pmod p$ and $y \pmod p$.

Exercises

§1 Suppose we know that $\text{rank}(M(f))$ is C. Give a crossing sequence argument that proves the communication complexity is at least $\lceil \log C \rceil$.

§2 Consider x, y as vectors over $GF(2)^n$ and let $f(x, y)$ be their inner product mod 2. Prove that the communication complexity is n. (Hint: Use the rank lowerbound.)

§3 For any graph G with n vertices, consider the following communication problem: Player 1 receives a clique C in G, and Player 2 receives an independent set I. They have to communicate in order to determine $|C \cap I|$. (Note that this number is either 0 or 1.) Prove an $O(\log^2 n)$ upperbound on the communication complexity.

Can you improve your upperbound or prove a lower bound better than $\Omega(\log n)$?

§4 Associate the following communication problem with any function $f : \{0,1\}^n \to \{0,1\}$. Player 1 gets any input x such that $f(x) = 0$ and player 2 gets any input y such that $f(y) = 1$. They have to communicate in order to determine a bit position i such that $x_i \neq y_i$.

Show that the communication complexity of this problem is *exactly* the minimum depth of any circuit that computes f.

§5 Use the previous question to show that computing the parity of n bits requires depth at least $2 \log n$.