CS 810: Complexity Theory 4/20/2007

Lecture 30: Communication Complexity

Instructor: Dieter van Melkebeek Scribe: Piramanayagam Arumuga Nainar

In our study of complexity theory in non-standard settings, last time we discussed query complexity - the number of bit positions in the input that must be looked up while evaluating a query. Today we discuss communication complexity which is motivated by distributed computing. In this setting, several parties, each with access to a private input, collaborate with each other to realize some activity. The objective function to be minimized is the number of bits exchanged between the parties during the activity. Our focus is only on the communication aspect and we do not care about the computational complexity.

First, we introduce the formal model in deterministic, randomized and quantum settings. Next, we give the lower bounds for some example tasks and discuss techniques that can be used to establish such bounds. Finally, we demonstrate an application of communication complexity by using it to derive bounds on circuit complexity.

1 Model

In the formal model, we make the usual assumption that we are solving decision problems. We also make the simplifying assumption that there are only two communicating entities: Alice and Bob. Given a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ and two strings $x, y \in \{0,1\}^n$, such that Alice has private access to x and Bob to y, Alice and Bob must evaluate $f(x, y)$ by exchanging as few bits as possible.

A solution to the above problem is called a *communication protocol*. It is specific to f and can be visualized as a binary-tree. The interior nodes are labeled A (for Alice) or B (for Bob) that denote the party initiating the communication at that step. The edges are labeled 0 or 1 denoting the value of the exchange. At each node, the corresponding party can decide which bit to send based on its input. The leaves of the tree are labeled 0 or 1 and denote the result on the given inputs. This result should then be conveyed to the other party. Thus, the complexity of the protocol is one more than the depth of the tree. The communication complexity of f is that of the protocol with smallest complexity.

Deterministic protocols, $\mathbf{D(f)}$: In a deterministic protocol, the decisions made at each node of the tree model are deterministic and the tree itself is fixed. Most interesting functions depend on all of their input and hence all n bits of one party's inputs must be sent to the other. Thus, they have a communication complexity of $n + 1$.

Randomized protocols: In the randomized setting, we allow Alice and Bob to toss coins during the computation. Also, we want the computation to be correct on a significant fraction ($\frac{2}{3}$) of the random strings. The cost of the protocol is the worst case communication complexity over all random strings. There are two variants in this setting:

1. **Public coin**: In query complexity, we discussed how a randomized decision procedure can be viewed as a probability distribution over deterministic decision procedures. Similarly, a

randomized communication protocol can be viewed as a probability distribution over deterministic protocols. Then, Alice and Bob use a public random string to sample a protocol from this distribution and use it for the rest of the communication. It can be shown that this is equivalent to Alice and Bob using a common random string for their computation. The complexity of f under public coin randomness is denoted by $R_{pub}(f)$.

2. **Private coin**: Here, Alice and Bob use coin tosses locally and the outcomes are not known to each other. Communicating local random bits also contributes to communication complexity. The complexity of f under private coin randomness is denoted by $R(f)$.

Quantum protocols: In this setting, Alice and Bob are quantum machines and exchange qubits between them. Like in the randomized case, there are two measures of complexity. The two machines may have a prior set of entangled states, in which case the complexity is denoted by $Q_{ent}(f)$. If they do not share entangled states, then the complexity is denoted by $Q(f)$.

We will prove that there is an exponential gap between D and R. We will also establish a relation between R and R_{pub}. All known bounds for fully specified functions establish only a polynomial gap between randomized and quantum settings. For functions that are not fully specified, exponential gaps are known to exist.

1.1 Examples

The following are some interesting problems in this setting:

1. Equality: Whether the two input strings given to Alice and Bob are equal? $EQ(x, y) = \chi[x = y]$
2. Disjointness: Interpreting the input strings as the characteristic sequence of subsets of a set of size n, are the two subsets disjoint? $DISJ(x, y) = \chi[x \cap y = \emptyset] = \neg \vee_{i=1}^{n} (x_i y_i)$
3. Inner product: $IP(x, y) = \Sigma_{I=1}^{n} x_i y_i$.

Table 1 shows the complexities for these problems under different models. We prove some of these results in the next subsection.

Table 1: Complexities of $EQ, DISJ, IP$ under $D, R, R_{pub}, Q, Q_{ent}$

f	$D(f)$	$R(f)$	$R_{pub}(f)$	$Q(f)$	$Q_{ent}(f)$
EQ	$\theta(n)$	$\theta(\log n)$	$\theta(1)$	$\theta(\log n)$	$\theta(1)$
$DISJ$	$\theta(n)$	$\theta(n)$	$\theta(n)$	$\theta(\sqrt{n})$	$\theta(\sqrt{n})$
IP	$\theta(n)$	$\theta(n)$	$\theta(n)$	$\theta(n)$	$\theta(n)$

1.2 Protocols

There are several classes of protocols that can be used to solve communication problems. We will discuss some of them and use them to derive a few bounds listed in Table 1.

Trivial: The trivial protocol is for Alice to send all of its input x to Bob and Bob will compute and share the value of $f(x, y)$. The complexity will be $\theta(n)$ and is the best we can hope for all the examples functions under the deterministic model.

Fingerprinting: Alice can create a short fingerprint of x and send it to y. The finger print should be in such a way that Bob can guess the correct answer with a reasonable accuracy. For example, consider the following $\theta(\log n)$ protocol for EQ under R. Alice computes $x \mod p$ for a randomly chosen prime number p of $2 \log n$ bits and sends the modulus as well as p to Bob. Bob concludes the result by computing $y \mod p$ and comparing it to $x \mod p$. If $x \neq y$, $x \equiv y \mod p$ for at most n choices of primes p since $x < 2^n$. Using $2 \log n$ bits, we have approximately $\theta(n^2)$ choices for p and $x \neq y \mod p$ for most of them.

Hadamard code: Using a public random string r of length n, Alice sends the r^{th} bit of the Hadamard encoding of x to Bob. If $x \neq y$, the bit that Bob computes using y will be different with probability $\frac{1}{2}$. We can repeat this protocol many times to get the desired accuracy. This gives a $\theta(1)$ R_{pub} protocol for EQ.

Quantum query: If f can be mapped to another function g whose *query complexity* is $p(n)$ such that each bit of the mapping can be computed using minimal communication, say $q(n)$, then the communication complexity of f is $p(n)q(n)$. For example, in last lecture, we mentioned that the quantum query complexity of the OR function is $\theta(\sqrt{n})$. The disjointness function $DISJ$ can be mapped to the OR function, as we saw earlier. Thus, we have a $\theta(\sqrt{n} \log n)$ protocol for $DISJ$ as follows: Alice runs the decision tree corresponding to OR and every time the i^{th} bit of the input is queried, Alice sends the configuration $|i, b, \xi\rangle$ to Bob after encoding x_i in ξ. Bob extracts x_i, computes $z_i = x_i y_i$ and sends back the configuration $|i, b \oplus z_i, \xi\rangle$. Alice returns this configuration as the answer to the query. Finally, Alice sends the negation of the decision tree's output to Bob. The length of ξ for this exchange turns out to be constant and hence the index i, which takes $\log n$ bits, is the dominating term. Since the query complexity of OR is $\sqrt{n}$, this protocol communicates $\theta(\sqrt{n} \log n)$ bits. Note: This protocol does not have the tight bound specified in Table 1.

2 Lower Bounds

In this part of the lecture, we establish the lower bounds for the problems discussed earlier. In order to do that, we need to introduce some new terminology.

Definition 1. *The* characteristic matrix M_f *of* f *is a* $2^n \times 2^n$ *matrix such that* $M_f(i, j) = f(b_i, b_j)$ *where* b_i, b_j *are the binary representations of* i *and* j *respectively.*

Definition 2. *A* combinatorial rectangle *in a matrix* M *is a* $p \times q$ *matrix induced by a subset of* p *rows and* q *columns of* M*.*

Definition 3. *A combinatorial rectangle of the characteristic matrix* M_f *is* f-monochromatic *if all the entries in the rectangle are equal.*

In the binary tree visualization of a communication protocol, there is a combinatorial rectangle A, B attached with each node of the tree that represent the subset of inputs for which the protocols reach that node. A node corresponding to Alice splits A into two disjoint sets, one for each of its successors. A node corresponding to Bob splits B into two disjoint sets. It is easy to see that the combinatorial rectangles associated with the leaves of the binary tree are *f-monochromatic*. Since f is fully specified, each cell of M_f is present in the combinatorial rectangle of at least one of the 2^{depth} leaves of the tree. Thus, each protocol for f involving an exchange of b bits induces a tiling of M_f into at most 2^b f-monochromatic combinatorial rectangles.

Definition 4. *T(f) is the minimum number of f-monochromatic rectangles induced in M_f by any communication protocol for f.*

Fact: $D(f) \geq \log T(f)$

2.1 Fooling set method

Claim 1. *If $S \subseteq \{0,1\}^n \times \{0,1\}^n$ such that:*

1. $(\exists b)(\forall (x,y) \in S) f(x,y) = b$
2. $\forall (x_1, y_1) \neq (x_2, y_2) \in S : f(x_1, y_2) \neq b \vee f(x_2, y_1) \neq b$

then, $|S| \leq T(f)$.

Proof. Let S' be the combinatorial rectangle induced by $A = \{x | (x,y) \in S\}$ and $B = \{y | (x,y) \in S\}$. The two conditions on S ensure that every f-monochromatic combinatorial rectangle in the characteristic matrix of S' under f is of size 1. Thus, no tiling of the characteristic matrix of S', and consequently M_f, can have fewer than $|S'|$ rectangles. Moreover, $|S'| = |A \times B| \leq |S|$. □

Examples: The set $S = \{(x,x) | x \in \{0,1\}^n\}$ satisfies the conditions in claim 1 for EQ. Thus, $D(EQ) \geq \log |S| = \log 2^n = n$. This is also easily seen by the fact that M_{EQ} is the identity matrix and requires 2^n tiles to cover each of the 2^n non-zero entries in it.

As another example, $S = \{(x, \bar{x}) | x \in \{0,1\}^n\}$ satisfies condition 2 in claim 1 for $DISJ$ because for any two pairs $(x, \bar{x})$ and $(y, \bar{y})$ if $x \neq y$ then y has at least one element in common with $\bar{x}$. Thus, $D(DISJ) \geq \log |S| = n$.

2.2 Bounding the size of maximum monochromatic combinatorial rectangle

To illustrate this method, consider the inner product function IP. We will show shortly that the size of any 0-chromatic combinatorial rectangle in M_{IP} is at most 2^n. At least half of the 2^{2n} entries in M_{IP} are zeroes. Thus, $T(f) \geq \frac{2^{2n-1}}{2^n} = 2^{n-1}$. And, $D(IP) \geq \lceil \log T(f) \rceil = n$.

To prove the upper bound on the size of a 0-chromatic combinatorial rectangle, consider a combinatorial rectangle $A \times B$ where $A, B \subseteq \{0,1\}^n$. Then, $f(A \times B)$ is equal to 0 and so is $f(A' \times B')$ where $A' = span(A)$ and $B' = span(B)$. Here, the *span* of a set A of vectors is the set of all linear combination of A over the finite field of length 2. We can show that for IP, $dim(A') + dim(B') \leq n$ and hence $|A \times B| \leq |A' \times B'| \leq 2^n$.

2.3 Using rank of M_f

Claim 2. $Rank(M_f) \leq T(f)$

Proof. For a tile t of any tiling of M_f, consider the matrix C_t obtained by setting all entries outside of t in M_f to zero. C_t will be the all zero matrix for all zero-tiles. Thus, if T_1 is the set of all 1-tiles of the tiling,

$$\begin{aligned} M_f &= \sum_{t \in T_1} C_t \\ Rank(M_f) &\leq \sum_{t \in T_1} Rank(C_t) \\ Rank(M_f) &\leq |T_1| \leq T(f) \end{aligned}$$

The second equation is obtained using the fact that the *Rank* function is subadditive. The last line is derived using the fact that the rank of C_t for any t in T_1 has rank 1. $\square$

As a corollary to the above claim, $D(f) \geq \log Rank(M_f)$. As examples for this technique, since the matrix M_{EQ} is the identity matrix, $Rank(M_{EQ}) = 2^n$. Thus, $D(EQ) \geq n$.

Exercise 1. *Derive $D(IP)$ using this technique.*

Conjecture 1 (*Log rank Conjecture*)**.** *There exists a constant c such that the complexity of deterministic communication protocols for f is bounded by $(\log Rank(M_f))^c$ where M_f is the characteristic matrix of f.*

3 Randomized Lower Bounds:

For R_{pub}, we can view the communication protocol as a two player game and use the Min-Max strategy similar to the one we used for randomized query complexity. In this section, we discuss other techniques and results for R and R_{pub}.

Claim 3. $R(f) = \Omega(\log D(f))$

Proof. In a randomized protocol, each node in the binary tree visualization of a communication protocol decides the bit to send based on its input as well as coin flips. Now, we associate a probability distribution, instead of just a subset, of inputs with each node of the tree. Now, for a given input, for each leaf ℓ, Alice can compute $\Pr_r$[on input x, Alice ends up on leaf ℓ]. Since this probability is computed by performing $R(f)$ operations on the initial probability of 1 for the root node, it can be expressed using $O(R(f))$ bits. Alice will send $2^{R(f)}O(R(f))$ bits. Since the two random coin flips of Bob are independent of those of Alice, Bob can compute the probability of ending up in a leaf ℓ for input (x, y) by computing a value similar to the one computed by Alice and multiplying them together. Based on the label of each leaf, Bob can compute the probability of acceptance. This gives a $2^{R(f)}O(R(f))$ deterministic protocol. Thus, $R(f) \geq \log D(f)$. $\square$

This shows that the randomized protocol for EQ using fingerprinting is optimal.

Claim 4. $R(f) = O(R_{pub}(f) + \log n)$

To show this result, we can amplify the success probability of the public coin randomized protocol and then use an averaging argument to prove the existence of polynomially many public random sequences with sufficiently high success probability on all inputs. So, Alice and Bob select one sequence using $\log n$ random bits and then follow the public coin protocol.

4 Applications in other areas

An important reason for studying communication complexity is that problems in other domains can be reduced to communication problems. Then, lower bounds for communication problems establish lower bounds for the corresponding problem. Examples include Area-Time lower bounds for VLSI chip design and query complexity on data structures. This lecture, we reduce the bounded fan-in circuit depth of a function to a communication problem.

Theorem 1. *Let $f : \{0,1\}^n \to \{0,1\}$. Alice and Bob have an input $x \in f^{-1}(0)$ and $y \in f^{-1}(1)$ respectively. The minimum communication complexity of finding an index i such that $x_i \neq y_i$ is equal to the minimum depth of a bounded fan-in circuit computing f.*

Note: The communication problem in this theorem is not a decision problem.

Proof. $\leq$: Consider a fanin 2 circuit C of depth d that computes f. WLOG, we can assume that all but the bottom-most level has AND and OR gates. Each gate g in the circuit computes a function f_g of the inputs. Suppose the top most gate is an AND gate. Since the output of the gate is 0 on input x at least one of the gate's inputs will be zero. Alice will choose the gate that produced the 0 input and convey it to Bob (using a one bit). On the other hand, if the gate is an OR gate, Bob will choose an input to the OR gate that is 1 and conveys the gate that produced that output to Alice. This process continues for d iterations, while maintaining the invariant that x evaluates to 0 and y evaluates to 1 on the gate under consideration. After d iterations, the process terminates at an input gate, which has different values on x and y. Note that we just have to convey the d bit indices to choose the gates at each depth. The circuit can be hardwired into both the parties and the result need not be communicated because both of them know the output at the end of the protocol.

$\geq$: For this direction, we prove the following stronger result.

Claim 5. *For any A, B such that $A \subseteq f^{-1}(0)$ and $B \subseteq f^{-1}(1)$, if there exists a protocol that works for A, B and is of communication complexity d, then there exists a circuit of depth $\leq d$ that outputs 0 on A and 1 on B.*

Proof by induction. For the base case, when $d = 0$, Alice and Bob know the index i in the input that are different. The corresponding depth 0 circuit is either x_i or $\bar{x}_i$ depending on the i^{th} bit in x and y. For the induction step, suppose there exists a communication protocol of complexity d that works for A, B. Suppose Alice initiates the proceedings in this protocol. Alice sends a 0 if its input x is in a subset A_0 of A and a 1 if its input is in a subset A_1 of A (A_0, A_1 are a partition of A). Now, there exist communication protocols of depth $\leq d - 1$ for $\{A_0, B\}$ and $\{A_1, B\}$. From the induction hypothesis, there exist circuits C_0 and C_1 such that:

1. C_0 evaluates to 0 on an input from A_0 and 1 on an input from B.

2. C_1 evaluates to 0 on an input from A_1 and 1 on an input from B.

We must construct a circuit from C_0 and C_1 that produces 0 on an input from A and 1 on an input from B. Combining the output of C_0 and C_1 using an AND gate does the job.

On the other hand, if Bob starts the communication, the dual argument holds and we combine the two circuits from the induction hypothesis using an OR gate. □

Corollary 1. *Monotone NC^1 circuits cannot decide if a graph has a perfect matching.*