COMS 4995: Introduction to Communication Complexity Sep 19, 2011

Lecture 4 – Communication Complexity of Relations

Instructor: *Xi Chen* Scribe: *Timothy Sun*

1 Introduction

This lecture deals with the communication complexity of relations. Earlier, we considered boolean functions of the form $f : X \times Y \to Z$. A more general notion is that of a *relation* $R \subseteq X \times Y \times Z$. The goal of Alice and Bob on an input (x, y) is to compute a z such that $(x, y, z) \in R$ or declare that no such z exists.

Definition 1. *An input (x, y) is* legal *if there is $z \in Z$ such that $(x, y, z) \in R$ and* illegal *otherwise.*

The protocol tree model for the communication complexity of functions also works for the complexity of relations. In particular, we can define $D(R)$ as the minimum depth over all protocols that compute R. Once again, the following fact about the leaves of the protocol holds:

Observation 2. *Every leaf is associated with a rectangle. If the protocol computes $R \subset X \times Y \times Z$, then the rectangles are monochromatic.*

It is natural to consider relations corresponding to boolean functions, where Z represents the set of possible certificates for $f(x, y) = 1$. For example, we can consider a generalization of $\overline{EQ}$.

Example 3. *The relation $U = \{(x, y, i) \mid x, y \in \{0, 1\}^n, x_i \neq y_i\}$ satisfies*

$$n - 2 \leq D(U) \leq n + \log_2 n.$$

Proof. The left inequality is equivalent to $D(EQ) = n \leq D(U) + 2$. Alice and Bob can run the optimal protocol for U and get back some $i \in [n]$. Alice can send x_i over, and Bob can check if $x_i \neq y_i$. This requires two extra bits, so $n = D(EQ) \leq D(U) + 2$. For the upper bound, Alice can send her entire string to Bob using n bits, and Bob can send back the first i for which $x_i \neq y_i$ with $\log n$ bits. □

Example 4. *Define another relation $U' = \{(x, y, i) \mid x_i = 1, y_i = 0\}$. U' is related to $DISJ$.*

2 Karchmer-Wigderson games

Why are we interested in the communication complexity of relations? Relations are also related to formula sizes for boolean functions by the so-called *Karchmer-Wigderson games.*

Definition 5. *Given a boolean function $f : \{0, 1\}^n \to \{0, 1\}$, define $R_f = \{(x, y, i) \mid f(x) = 1, f(y) = 0, x_i \neq y_i\}$.*

Definition 6. *A* boolean formula *F is a binary tree where the leaves have the value z_i or $\overline{z_i}$ and the internal nodes are either AND or OR gates.*

Definition 7. *Given a boolean formula F, its* size *is the total number of input nodes (that is, the number of leaves). Given a boolean function $f : \{0,1\}^n \to \{0,1\}$, its* size complexity *$L(f)$ is defined to be the minimum size of a boolean formula computing f.*

Theorem 8 (Karchmer and Wigderson '88)**.** *$L(f) = C^P(R_f)$.*

Proof. We show that we can transform any boolean formula to a protocol tree with the same number of leaf nodes and vice versa. We induct on the depth of the formula. If the formula consists of a single literal, Alice and Bob do not have to do anything, so there is a single rectangle covering all of $X \times Y$. Suppose the optimal formula F is a disjunction $g_1 \vee g_2$. That is, the root node is an OR node. In the case of $g_1 \wedge g_2$, switch the roles of Alice and Bob. Alice knows that $f(y) = 0$, so $(g_1 \vee g_2)(y) = 0$ implies that $g_1(y) = g_2(y) = 0$. On the other hand $f(x) = 1$, so at least one of g_1 and g_2 evalutes to 1 on input x, so Alice can send Bob one bit telling him which g_i satisfies $g_i(x) = 1$. g_i has depth one less than F, and $g_i(x) = 1$ and $g_i(y) = 0$.

By doing this, we can always guarantee that one of the children of the root satisfies the property that $f(x) = 1$ and $f(y) = 0$, so by induction the cost of this protocol is exactly the depth of the boolean formula. That is, if we were to replace OR nodes with Alice nodes, and AND nodes with Bob nodes, we get a protocol tree for the relation. Similarly, the reverse operation produces a boolean formula from a protocol tree. □

Definition 9. *The $PARITY$ function of $z_1, \ldots, z_n$ is defined to be $\sum_{i=1}^{n} z_i \pmod 2$.*

Lemma 10 (Khrapchenko's bound)**.** *Let $C \subseteq X \times Y$, where $C = \{(x,y) \mid f(x) = 1, f(y) = 0, d(x,y) = 1\}$ where d is the* Hamming distance*. Then, $C^D(f) \geq \frac{|C|^2}{|X||Y|}$.*

Proof. Let $C^D(f) = t$ with rectangles $R_1, R_2, \ldots, R_t$. Let $m_i = |C \cap R_i|$. Since we have a partition,

1. $\sum_{i=1}^{t} |R_i| = |X||Y|$ and
2. $|C| = \sum_{i=1}^{t} m_i$.

We need to show that $m_i^2 \leq |R_i|$. Suppose $(x,y), (x,y') \in C$. Then consider a rectangle $R_i = X' \times Y'$. The number of elements of R_i in C is at most $\min(|X'|, |Y'|)$, so $m_i^2 \leq \min(|X'|, |Y'|)^2 \leq |X'||Y'| = |R_i|$. By Cauchy-Schwarz, we obtain

$$|C|^2 = (\sum_{i=1}^{t} m_i)^2 \leq t(\sum_{i=1}^{t} m_i^2) \leq t(\sum_{i=1}^{t} |R_i|) = t|X||Y|.$$

Rearranging yields the desired inequality. □

Example 11. *For $PARITY$, we have $|X| = |Y| = 2^{n-1}$ because the last bit is uniquely determined, and $|C| = n2^{n-1}$, since each $x \in X$ has n places where it can differ by one bit. Thus, $C^D(f) \geq n^2$.*

Theorem 12. *$L(PARITY) = n^2$ when n is a power of 2.*

Proof. From Khrapchenko's bound, $L(f) = C^D(f) \geq n^2$. Now, we exhibit a deterministic protocol with $2 \log n$ bits to get equality. The corresponding relation $R_{PARITY} = \{(x, y, i) \mid PARITY(x) = 1, PARITY(y) = 1, x_i \neq y_i\}$. Alice and Bob compute the parity of the first half of their input and send the result to each other. If they are equal, then they recurse on the second half. If they differ, they recurse on the first half. While there may be places where the bits differ in the half that was thrown away, we can guarantee that there exists a place they differ in the half that was kept. This binary search technique yields a protocol with cost $2 \log n$. □

3 The FORK relation

Definition 13. *The FORK relation is defined as follows: Alice and Bob each have an n-bit boolean string, and for $i \in [n]$, b_i is defined to be $\overline{EQ}(x_i, y_i)$. That is, b_i is 0 if $x_i = y_i$ and 1 otherwise. Let $b_0 = 0$ and $b_{n+1} = 1$. The goal is to find $i \in \{0, 1, \ldots, n+1\}$ such that in the string $b_0 b_1 \ldots b_n b_{n+1} = 0 b_1 \ldots b_n 1$, $b_i = 0$ and $b_{i+1} = 1$.*

Theorem 14. $D(FORK) \leq 2 \log_2 n$.

Proof. Alice and Bob can compute $b_{n/2}$ using two bits. If $b_{n/2} = 1$, then they recurse on the first half. If $b_{n/2} = 0$, they recurse on the second half. This yields a protocol with $2 \log_2 n$ bits of communication. □

$FORK_{n,m}$ is defined to be $FORK$ over $x_1 \ldots x_n, y_1 \ldots y_n \in [m]$, where $b_i = \overline{EQ}(x_i, y_i)$. By the same argument, $D(FORK_{n,m}) = O(\log n \log m)$. We will show in the next lecture that $D(FORK_{n,m}) = \Omega(\log m \log n)$, so this binary search protocol is asymptotically optimal.