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

Lecture 17: Introduction to Communication Complexity

March 28, 2013

Lecturer: Venkatesan Guruswami *Scribe: Amit Datta*

1 Introduction

The question that we ask when dealing with communication complexity [referenced to hereafter as CC] is "How many bits do two or more parties need to exchange to compute a function on their inputs?". This is a rich topic, so much that an entire book was written on it way back in 1997.

1.1 CC vs IT

The starting point for Information Theory [IT] is that certain communication needs to take place and the particular question that we ask is "**How** to carry out the communication?", for example which codes to use.

However, for CC, the starting point is that some problem needs to be solved [e.g. compute (A+B) when A, B are with different parties]. The specific question that we are interested in here is "**What** needs to be communicated?" So, the basic difference between IT and CC is that one is about the 'how' while the other is about the 'what'.

Recently many connections have emerged between CC and IT. "Most lower bound techniques for CC actually lower bound the information exchanged (not just the number of bits exchanged but, which could be greater)"

2 Classic CC

The problem of two-party CC was first introduced by Yao in 1979. If Alice and Bob are two parties, Alice having an input x and Bob having an input y, at the end of the 2-party communication protocol, each should end up with the value $f(x, y)$, where f is some function that is known to both Alice and Bob. In particular, we want to deal with functions of the form:

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

A trivial protocol to achieve this is the following:

- Alice sends x to Bob [n bits]
- Bob computes $f(x, y)$ and returns the result to Alice [1 bit]

The trivial protocol requires an exchange of $n + 1$ bits.

CC is useful in

1. circuit complexity
2. streaming computation
3. data structure lower bounds

Definition 2.1 (Communication Protocol)**.** *A communication is a sequence of instructions such that, given a party's input and the sequence of bits exchanged so far, it can be determined which bit is to sent next / what to output.*

The communication protocol can be viewed as a binary tree, each node having two edges, labeled with 0 and 1 each. Each node v is labeled with a_v(or b_v), which is a function that computes what Alice (or Bob) needs to send to Bob (or Alice) depending on x(or y) and the transcript so far.

Protocol Π is said to correctly compute a function f iff $\forall(x, y)$, following Π leads to a leaf labeled with $f(x, y)$. For a protocol Π that correctly computes f, define CC(Π, f) = depth of the protocol tree. This also represents the maximum number of bits exchanged between the parties for any inputs (x, y).

Definition 2.2. *The deterministic CC of f, denoted by $D(f)$ is given by:*

$$D(f) = \min_{\Pi \ s.t \ \Pi \ \textit{correctly computes} \ f} CC(\Pi, f)$$

We already saw that $D(f) \leq n + 1$. Let us see a few examples now.

- When inputs can be only of the form 1^n or 0^n, $D(f) \leq 2$
- When f is parity bit of both x, y, i.e. $f(x, y) = \oplus_i x_i \oplus \oplus_j y_j$, where x_i is the i^{th} bit of x. In this case, $D(f) \leq 2$, since Alice can compute her parity bit and send it to Bob, who can compute the joint parity and send it back.
- $f = EQ$, i.e. $f(x, y) = 1$ when $x = y$; 0 otherwise. In this case, $D(EQ) = n + 1$. We will see a concrete proof of this later in the text.

The communication protocol can be seen as a matrix, with the rows have all possible values of x, and the columns having all possible values of y, each entry of the matrix corresponding to elements (x, y) having $f(x, y)$.

Consider a protocol tree, we define a set R_l for a leaf l as follows:

$$R_l = \{(x, y) | \Pi \text{ leads to l on input } (x, y)\}$$

Claim 2.3. $\forall$ *leaves l, R_l is a rectangle on the communication matrix.*

In fact, the above claim holds for all nodes in the tree.

Proof. We prove this by induction.

Base Case

This trivially holds for the root of the tree, since R_{root} includes all the elements in the matrix, which is a rectangle.

Inductive Hypothesis

The claim holds for some intermediate node w, with children u, v, i.e. R_w is a rectangle.

Inductive Step

W.l.o.g, let us assume that w node represents a turn for Alice [A similar proof will hold if its Bob's turn]. In order to reach u, some additional constraints [that this bit should be a 0] need to be enforced on the input x of Alice. So, some of the rows of the rectangle R_w will be removed, but none of the columns of R_w would be affected. After the removal of these rows, the R_u obtained would still be a rectangle. Similarly, R_v can be showed to be rectangle. □

Definition 2.4. *A rectangle R is said to be f-monochromatic if* $\exists b \in \{0,1\} : \forall (x,y) \in R, f(x,y) = b$

Note 1 For any protocol Π, the leaves give a partition of $X \times Y$ in to rectangles.

Note 2 If Π correctly computes f, then each R_l corresponding to a leaf l must be f-monochromatic.

So, if $D(f) = c$ then $\exists$ a partition of $X \times Y$ into 2^c f-monochromatic rectangles. The contrapositive of this is presented as a theorem:

Theorem 2.5. *If any partition of $X \times Y$ into f-monochromatic rectangles needs $\geq t$ rectangles, then $D(f) \geq \lceil \log_2 t \rceil$*

Let us observe a few more examples:

EQ: Observe that the communication matrix for this function would be an identity matrix. No two 1s would be in one monochromatic rectangle. So, the number of rectangles enclosing the 1s would be 2^n. For the remaining 0s, the number of rectangles would be ≥ 1. So, $D(EQ) \geq \lceil \log_2(2^n + 1) \rceil = n + 1$

We take a slight detour here to introduce *Fooling Sets*. A 0-fooling set is a set of pairs $(x, y) \in f^{-1}(0)$ such that no two of them are together in a monochromatic rectangle. $FS_0(f) =$ max size of a 0-fooling set. Similarly, $FS_1(f) =$ max size of a 1-fooling set.

Theorem 2.6.

$$D(f) \geq \lceil \log_2(FS_0(f) + FS_1(f)) \rceil$$

So, for $D(EQ)$, $FS_1(f) \geq 2^n$, $FS_0(f) \geq 1$.

Exercise 2.7. *Prove that $D(GT) = n + 1$, where $GT(x, y) = 1$ if $x > y$; 0 otherwise.*

DISJ: Disjointness is like the "3-SAT of CC". DISJ$(x, y) = 1$ if $x \cap y = \phi; 0$ otherwise. Think of x as the set $\{i | x_i = 1\}$. DISJ$(x, y) = (x_1 \wedge y_1) \vee (x_2 \wedge y_2) \vee \cdots \vee (x_n \wedge y_n)$

Theorem 2.8.

$$D(DISJ) = n + 1$$

Proof. Lets try to prove that $FS_1(\text{DISJ})$ is large. Consider two pairs of inputs $(A, \overline{A})$ and $(B, \overline{B})$ such that $A \neq B$. Since one is a complement of the other, we have DISJ$(A, \overline{A}) = 1$ and DISJ$(B, \overline{B}) = 1$, since they are definitely disjoint.

Since $A \neq B$, $\exists a \in A \backslash B$. So, $a \in A \cap \overline{B}$. Hence, $(A, \overline{B})$ are not disjoint and DISJ$(A, \overline{B}) = 0$. Similarly DISJ$(B, \overline{A}) = 0$. Hence, for no two inputs of this type will they be with the same monochromatic rectangle.

So, $FS_1(\text{DISJ}) \geq 2^n$ and $FS_0(\text{DISJ}) \geq 1$. This implies $D(\text{DISJ}) = n + 1$. □