University of California, Los Angeles
CS 289A Communication Complexity

Instructor: Alexander Sherstov
Scribe: Rebecca Hicks
Date: January 9, 2012

LECTURE

1

Deterministic Communication

This lecture is a short and general introduction to communication complexity theory. It focuses on basic terms and definitions to be used and built upon throughout the course.

We start with several examples of two-party communication protocols to illustrate the problems to be solved in this course and communication theory in general. Protocols are then defined in two ways: first formally, and then in a way meant to clarify the concept pictorially. Finally, the rectangle property is defined and illustrated through a proof.

1.1 Protocol Examples

In the two-party setting of deterministic communication [1], information from two parts in the system is necessary to complete the computation of a function $f : X \times Y \to Z$. We refer to the two parties as Alice and Bob (see Figure 1.1). Alice and Bob each have a portion of the input: Alice has $x \in X$, and Bob has $y \in Y$. Alice has no knowledge of y, and Bob has no knowledge of x. When determining the cost of computing f in this model, we ignore the computational costs incurred by Alice and Bob individually; we only care about the total number of bits exchanged between them.

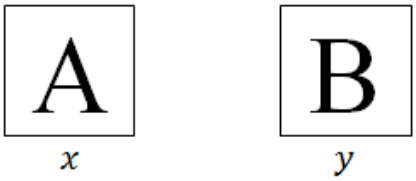

FIGURE 1.1: The two-party model of deterministic communication. Alice (labeled "A" in the figure) has part X of the input needed to compute f, and Bob (labeled "B") has part Y of the input.

EXAMPLE 1.1. Determining the equality of two parties' strings.

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

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

Solution: Alice sends x to Bob.
Cost: $n+1$ bits where the n is Alice's string of n bits, and "+1" is Bob's response.

EXAMPLE 1.2. Determining the parity of two parties' strings.

$$\begin{aligned} PARITY_n(x,y) &= \bigoplus_{i=1}^{n}(x_i \oplus y_i) \\ &= \begin{cases} 1 & \text{if } \sum_{i=1}^{n}(x_i + y_i) \equiv 1 \pmod 2 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Possible Solution: Alice sends x to Bob. This is expensive!
Solution: Alice sends $\bigoplus_{i=1}^{n} x_i$
Cost: 2 bits
Why is this correct? Because $\bigoplus_{i=1}^{n}(x_i \oplus y_i) = (\bigoplus_{i=1}^{n} x_i) \bigoplus (\bigoplus_{i=1}^{n} y_i)$.

EXAMPLE 1.3. Determining the average of two parties' sets.

$$AVG_n(S,T) \text{ where } S,T \subseteq \{1,2,3,\ldots,n\}$$

Solution: Alice sends $\sum_{a\in S} a, |S|$, Bob responds with $\sum_{b\in T} b, |T|$.
Cost: $O(\log n)$.

EXAMPLE 1.4. Determining the median of two parties' sets.

$$MED_n(S,T) \text{ where } S,T \subseteq \{1,2,3,\ldots,n\} \text{ are multisets}$$

This example will illustrate *interaction*, which plays a crucial role in communication theory.
Cost: $O(\log^2 n)$.
Idea: Use binary search to determine the median.

1. Start with a set {1, 2, ..., n}
2. Pick a midpoint, $\frac{n}{2}$
3. Alice and Bob tell each other how many numbers in their collections are larger and smaller than the selected midpoint
4. Recursion on appropriate half or range

1.2 Defining Protocols

DEFINITION 1.5. A communication *protocol* is formally defined by three functions:

1. $NEXT : \{0,1\}^* \to \{A, B, \perp\}$
   - The $NEXT$ function makes it clear which party has the next turn
   - The $NEXT$ function does not depend on Alice's input or Bob's input, just the transmission history

2. $ALICE : \{0,1\}^* \times X \to \{0,1\}$
   - By X, we mean Alice's part in the function $f : X \times Y \to \{0,1\}$
3. $BOB : \{0,1\}^* \times Y \to \{0,1\}$
   - By Y, we mean Bob's part in the function $f : X \times Y \to \{0,1\}$

This formal definition of protocols is very abstract. An easier definition to work with uses trees:

DEFINITION 1.6. A *protocol* $\mathcal{P}$ over domain $X \times Y$ with range Z is a binary tree where each internal node v is labeled either by a function $a_v : X \to \{0,1\}$ or by a function $b_v : Y \to \{0,1\}$, and each leaf is labeled with an element $z \in Z$.
The *value of the protocol* $\mathcal{P}$ on input (x, y) is the label of the leaf reached by starting from the root, and walking on the tree. At each internal node v labeled by a_v walking left if $a_v(x) = 0$ and right if $a_v(x) = 1$, and at each internal node labeled by b_v walking left if $b_v(y) = 0$ and right if $b_v(y) = 1$. The *cost of the protocol* $\mathcal{P}$ *on input* (x, y) is the length of the path taken on input (x, y). The *cost of the protocol* $\mathcal{P}$ is the height of the tree. [1]

See Figures 1.2 and 1.3 for two example protocol trees.

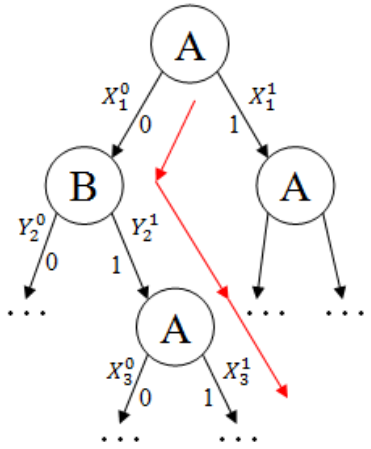

FIGURE 1.2: An example of a tree representation of a communication protocol. The superscripts of the X and Y arrow labels refer to the selection made at the node from which the arrow originates. The subscripts refer to a distinct numbering of the nodes. The example $NEXT$ function $\{0, 1, 1, \ldots\}$ is marked in red.

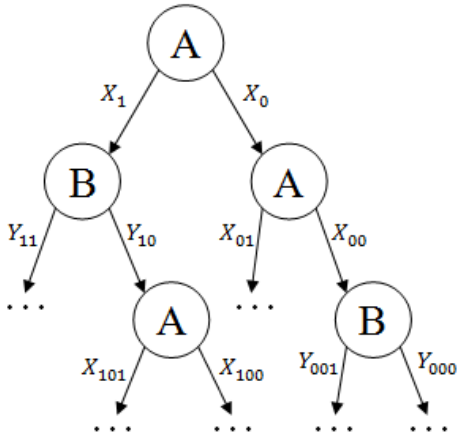

FIGURE 1.3: Another example of a tree representation of a communication protocol. In this version, the subscript of the X and Y arrow labels refers to the path taken so far, defined by the choices made at each node.

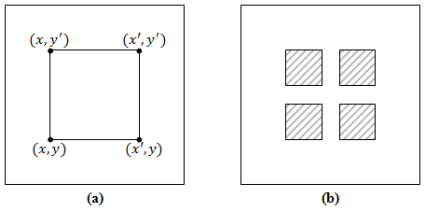

FIGURE 1.4: Rectangle property illustrated. (a) If the set contains (x, y) and (x', y'), the set must contain (x', y) and (x, y'). (b) A set with gaps may have the rectangle property.

1.3 Rectangles

According to [1], "The success in proving good lower bounds on the communication complexity of various functions comes from the *combinatorial* view we take on protocols." This view of protocols allows us to partition the space of possible input pairs $X \times Y$ into sets such that the same communication is sent for all pairs in the same set during the execution of a given protocol. The parts into which the input space is split are called rectangles.

DEFINITION 1.7. Let $f : X \times Y \to \{0, 1\}$ be the given communication problem. A *rectangle* (a.k.a. a *combinatorial rectangle*) is a set $S \subseteq X \times Y$ of the form $S = A \times B$, $A \subseteq X$, $B \subseteq Y$.

The following rectangle property is crucial in the study of communication.

THEOREM 1.8. *A set $S \subseteq X \times Y$ is a rectangle if and only if S has the following property (called the* rectangle property*):*

$$(x, y) \in S, (x', y') \in S \Rightarrow (x, y') \in S, (x', y) \in S.$$

That is, if two points (x, y) and (x', y') are in S, then (x, y') and (x', y) are also in S. See Figure 1.4.

Proof. The forward implication is trivial: a rectangle has the rectangle property by definition. We now consider the reverse implication; see Figure 1.5. Define the projection of S onto X by

$$A = \{a \in X : \text{ such that } (a, b) \in S \text{ for some } b \in Y\}$$

and the projection of S onto Y by

$$B = \{b \in Y : \text{ such that } (a, b) \in S \text{ for some } a \in X\}$$

We will show $S = A \times B$. Indeed, take any $a \in A$ and $b \in B$. Then by definition, $(a, b') \in S$ for some $b' \in Y$ and $(a', b) \in S$ for some $a' \in X$. By the rectangle property of S, we have $(a, b) \in S$ and $(a', b') \in S$. In summary, $(a, b) \in S$ whenever $a \in A$ and $b \in B$. Thus, $S \supseteq A \times B$. Since $S \subseteq A \times B$ for trivial reasons, we have $S = A \times B$ as claimed. □

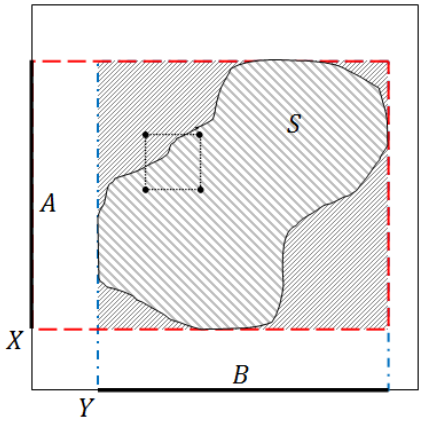

FIGURE 1.5: Rectangle property proof. The bold lines represent A (the projection of S onto X) and B (the projection of S onto Y). The irregular shape is $S \subseteq A \times B$ that has the rectangle property.

References

[1] E. Kushilevitz and N. Nisan. *Communication complexity*. Cambridge University Press, 2nd edition, 2006.