Computational Complexity: A Modern Approach

Draft of a book: Dated January 2007
Comments welcome!

Sanjeev Arora and Boaz Barak
Princeton University
complexitybook@gmail.com

This is an Internet draft. Some chapters are more finished than others. References and attributions are very preliminary and we apologize in advance for any omissions (but hope you will nevertheless point them out to us).

Please send us bugs, typos, missing references or general comments to complexitybook@gmail.com — **Thank You!!**

Chapter 13

Communication Complexity

Communication complexity concerns the following scenario. There are two players with unlimited computational power, each of whom holds an n bit input, say x and y. Neither knows the other's input, and they wish to collaboratively compute $f(x, y)$ where the function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is known to both. Furthermore, they had foreseen this situation (e.g., one of the parties could be a spacecraft and the other could be the base station on earth), so they had already —before they knew their inputs x, y— agreed upon a protocol for communication.[1] The *cost* of this protocol is the *number of bits communicated* by the players for the *worst-case* choice of inputs x, y.

Researchers have studied many modifications of the above basic scenario, including randomized protocols, nondeterministic protocols, and average-case protocols. Furthermore, lower bounds on communication complexity have uses in a variety of areas, including lower bounds for parallel and VLSI computation, circuit lower bounds, polyhedral theory, data structure lower bounds, and more.

In this chapter we only give a very rudimentary introduction to this area. In Section 13.1 we provide the basic definition of two-party deterministic communication complexity. In Section 13.2 we survey some of the techniques used to prove *lower bounds* for the communication complexity of various functions, using the equality function (i.e., $f(x, y) = 1$ iff $x = y$) as a running example. In Section 13.3 we define *multiparty* communication complexity and show a lower bound for the generalized inner product function. Section 13.4 contains a brief survey of other models studied, including probabilistic and non-deterministic communication complexity. The chapter notes mention some of the many applications of communication complexity.

13.1 Definition of two-party communication complexity.

Now we formalize the informal description of communication complexity given above:

[1] Do not confuse this situation with *information theory*, where an algorithm is given messages that have to be transmitted over a noisy channel, and the goal is to transmit them robustly while minimizing the amount of communication. In communication complexity the channel is not noisy and the players determine what messages to send.

Definition 13.1 (Two party communication complexity)
Let $f : \{0,1\}^{2n} \to \{0,1\}$ be a function. A *t-round two party protocol* Π *for computing* f is a sequence of t functions $P_1, \ldots, P_t : \{0,1\}^* \to \{0,1\}^*$. An execution of Π on inputs x, y involves the following: Player 1 computes $p_1 = P_1(x)$ and sends p_1 to Player 2, Player 2 computes $p_2 = P_2(y, p_1)$ and sends p_2 to Player 1, and so on. Generally, at the i^{th} round, if i is odd then Player 1 computes $p_i = P_i(x, p_1, \ldots, p_{i-1})$ and sends p_i to Player 2, and similarly if i is even then Player 2 computes $p_i = P_i(y, p_1, \ldots, p_{i-1})$ and sends p_i to Player 1.

The protocol Π is valid if for every pair of inputs x, y, the last message sent (i.e., the message p_t) is equal to the value $f(x, y)$. The *communication complexity* of Π is the maximum number of bits communicated (i.e., maximum of $|p_1| + \ldots + |p_t|$) over all inputs $x, y \in \{0,1\}^n$. The *communication complexity* of f, denoted by $C(f)$ is the minimum communication complexity over all valid protocols Π for f.

For every function, $C(f) \leq n + 1$ since the trivial protocol is for first player to communicate his entire input, whereupon the second player computes $f(x, y)$ and communicates that single bit to the first. Can they manage with less communication?

Example 13.2 (Parity)
Suppose the function $f(x, y)$ is the *parity* of all the bits in x, y. Then $C(f) = 2$. Clearly, $C(f) \geq 2$ since the function depends nontrivially on each input, so each player must transmit at least one bit. The fact that $C(f) \leq 2$ is demonstrated by the following protocol: Player 1 sends the parity a of the bits in x and Player 2 sends a XOR'd with the parity of the bits in y.

Example 13.3 (Halting Problem)
Consider the function $\mathrm{H} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ defined as follows. If $x = 1^n$ and $y = \mathrm{code}(M)$ for some Turing Machine M such that M halts on x then $H(x, y) = 1$ otherwise $H(x, y) = 0$. The communication complexity of this is at most 2; first player sends a bit indicating whether or not his input is 1^n. The second player then determines the answer and sends it to the first player. This example emphasizes that the players have unbounded computational power, including ability to solve the Halting Problem.

Sometimes students ask whether a player can communicate by not saying anything? (After all, they have three options in each round: send a 0, or 1, or not send anything.) We can regard such protocols as having one additional bit of communication, and analyze them analogously.

13.2 Lower bound methods

Now we discuss methods for proving lower bounds on communication complexity. As a running example in this chapter, we will use the equality function:

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

It turns out that almost no improvement is possible over the trivial $n + 1$ bit communication protocol for this function:

Theorem 13.4 (Equality has linear communication complexity)
$C(EQ) \geq n$

We will prove Theorem 13.4 by several methods below.

13.2.1 The fooling set method

The first proof of Theorem 13.4 uses an idea called *fooling sets*. For any communication protocol for any function, suppose x, x' are any two different n-bit strings such that the communication pattern (i.e., sequence of bits transmitted) is the same on the input pairs (x, x) and (x', x'). Then we claim that the players' final answer must be the same on all four input-pairs $(x, x), (x, x'), (x', x), (x', x')$. This is shown by an easy induction. If player 1 communicates a bit in the first round, then by hypothesis 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 x and x' since he receives the same bit from player 1. And so on. We conclude that at the end, the players' answer on (x, x) must agree with their answer on (x, x').

To show $C(EQ) \geq n$ it suffices to note that if a protocol exists whose complexity is at most $n-1$, then there are only 2^{n-1} possible communication patterns. But there are 2^n choices for input pairs of the form (x, x) and so by the pigeonhole principle, there exist two distinct pairs (x, x) and (x', x') on which the communication pattern is the same. But then the protocol must be incorrect, since $\mathrm{EQ}(x, x') = 0 \neq \mathrm{EQ}(x, x)$. This completes the proof. This argument can be easily generalized as follows (Exercise 13.1):

Lemma 13.5 *Say that a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ has a* size M fooling set *if there is an M-sized subset $S \subseteq \{0,1\}^n \times \{0,1\}^n$ and a value $b \in \{0,1\}$ such that **(1)** for every $\langle x, y\rangle \in S$, $f(x, y) = b$ and **(2)** for every distinct $\langle x, y\rangle, \langle x', y'\rangle \in S$, either $f(x, y') \neq b$ or $f(x', y) \neq b$.*

If f has a size-M fooling set then $C(f) \geq \log M$.

Example 13.6 (Disjointness)
Let x, y be interpreted as characteristic vectors of subsets of $\{1, 2, \ldots, n\}$. Let $\mathrm{DISJ}(x, y) = 1$ if these two subsets are disjoint, otherwise $\mathrm{DISJ}(x, y) = 0$. As a corollary of Lemma 13.5 we obtain that $C(\mathrm{DISJ}) \geq n$ since the following 2^n pairs constitute a fooling set:

$$S = \left\{(A, \overline{A}) : A \subseteq \{1, 2, \ldots, n\}\right\}.$$

13.2.2 The tiling method

The tiling method for lower bounds takes a more global view of the function f. Consider the matrix of f, denoted $M(f)$, which is a $2^n \times 2^n$ matrix whose (x, y)'th entry is the value $f(x, y)$ (see Figure 13.1.) We visualize the communication protocol in terms of this matrix. A *combinatorial rectangle* (or just rectangle for short) in the matrix M is a submatrix of M that corresponds to entries in $A \times B$ where $A \subseteq \{0,1\}^n$, $B \subseteq \{0,1\}^n$, we say that $A \times B$ is *monochromatic* if for all x in A and y in B, $M_{x,y}$ is the same. If the protocol begins with the first player sending a bit, then $M(f)$ partitions into two rectangles of the type $A_0 \times \{0,1\}^n$, $A_1 \times \{0,1\}^n$, where A_b is the subset of the input for which the first player communicates the bit b. Notice, $A_0 \cup A_1 = \{0,1\}^n$. If the next bit is sent by the second player, then each of the two rectangles above is further partitioned into two smaller rectangles depending upon what this bit was. Finally, if the total number of bits

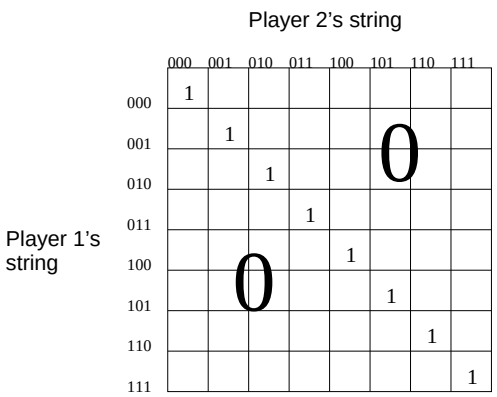

Figure 13.1: Matrix M(f) for the equality function when the inputs to the players have 3 bits. The numbers in the matrix are values of f.

communicated is k then the matrix gets partitioned into 2^k rectangles. Note that each rectangle in the partition corresponds to a subset of input pairs for which the communication pattern thus far has been identical. (See Figure 13.2 for an example.) When the protocol stops, the value of f is determined by the sequence of bits sent by the two players, and thus must be the same for all pairs x, y in that rectangle. Thus the set of all communication patterns must lead to a partition of the matrix into *monochromatic* rectangles.

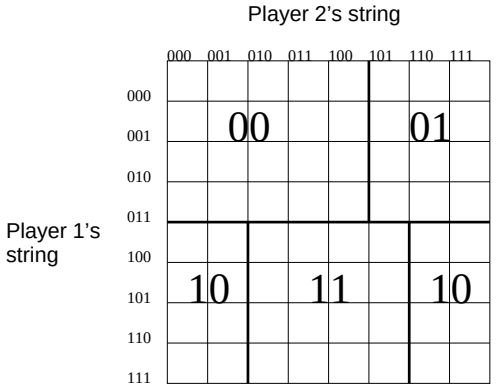

Figure 13.2: Two-way communication matrix after two steps. The large number labels are the concatenation of the bit sent by the first player with the bit sent by the second player.

Definition 13.7
A monochromatic tiling of $M(f)$ is a partition of $M(f)$ into disjoint monochromatic rectangles. We denote by $\chi(f)$ the minimum number of rectangles in any monochromatic tiling of $M(f)$.

We have the following connection to communication complexity.

Theorem 13.8 (Tiling and communication complexity [**AhoUlYa83**])
$\log_2 \chi(f) \leq C(f) \leq 16(\log_2 \chi(f))^2$.

PROOF: The first inequality follows from our above discussion, namely, if f has communication complexity k then it has a monochromatic tiling with at most 2^k rectangles. The second inequality is left as Exercise 13.4. ■

The following observation shows that for every function f whose communication complexity can be lower bounded using the fooling set method, the communication complexity can also be lower bounded by the tiling method. Hence the latter method subsumes the former.

Lemma 13.9 *If f has a fooling set with m pairs, then $\chi(f) \geq m$.*

PROOF: If (x_1, y_1) and (x_2, y_2) are two of the pairs in the fooling set, then they cannot be in a monochromatic rectangle since not all of $(x_1, y_1), (x_2, y_2)$, $(x_1, y_2), (x_2, y_1)$ have the same f value. ■

13.2.3 The rank method

Now we introduce an algebraic method to lower bound $\chi(f)$ (and hence the communication complexity of f). Recall the notion of *rank* of a square matrix: the size of the largest subset of rows that are linearly independent. The following lemma (left as Exercise 13.5) gives an equivalent characterization of the rank:

Lemma 13.10 *The* rank *of an $n \times n$ matrix M over a field $\mathbb{F}$, denoted by rank(M), is the minimum value of ℓ such that M can be expressed as*

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

where each B_i is an $n \times n$ matrix of rank 1.

Note that $0, 1$ are elements of every field, so we can compute the rank of a binary matrix over any field we like. The choice of field can be crucial; see Exercise 13.8.

Observing that every monochromatic rectangle can be viewed (by filling out entries outside the rectangle with 0's) as a matrix of rank at most 1 , we obtain the following theorem:

Theorem 13.11
For every function f, $\chi(f) \geq$ rank$(M(f))$.

Example 13.12
The matrix for the equality function is simply the identity matrix, and hence $\text{rank}(M(Eq)) = 2^n$. Thus, $C(EQ) \geq \log \chi(EQ) \geq n$, yielding another proof of Theorem 13.4.

13.2.4 The discrepancy method

For this method it is convenient to transform f into a ± 1-valued function by using the map $b \mapsto (-1)^b$ (i.e., $0 \mapsto +1, 1 \mapsto -1$. Thus $M(f)$ will also be a ± 1 matrix. We defined the *discrepancy* of a rectangle $A \times B$ in a $2^n \times 2^n$ matrix M to be

$$\frac{1}{2^{2n}} \left| \sum_{x \in A, y \in B} M_{x,y} \right| .$$

The *discrepancy* of the matrix $M(f)$, denoted by $\text{Disc}(f)$, is the maximum discrepancy among all rectangles. The following easy lemma relates it to $\chi(f)$.

Lemma 13.13 $\chi(f) \geq \frac{1}{Disc(f)}$.

PROOF: If $\chi(f) \leq K$ then there exists a monochromatic rectangle having at least $2^{2n}/K$ entries. Such a rectangle will have discrepancy at least $1/K$. ■

Lemma 13.13 can be very loose. For the equality function, the discrepancy is at least $1 - 2^{-n}$ (namely, the discrepancy of the entire matrix), which would only give a lower bound of 2 for $\chi(f)$. However, $\chi(f)$ is at least 2^n, as already noted.
Now we describe a method to upper bound the discrepancy using *eigenvalues*.

Lemma 13.14 (Eigenvalue bound) *For any real matrix M, the discrepancy of a rectangle $A \times B$ is at most $\lambda_{max}(M)\sqrt{|A|\,|B|}/2^{2n}$, where $\lambda_{max}(M)$ is the magnitude of the largest eigenvalue of M.*

PROOF: Let $\mathbf{1}_S \in \mathbb{R}^{2^n}$ denote the characteristic vectors of a subset $S \subseteq \{0,1\}^n$ (i.e., the x^{th} coordinate of $\mathbf{1}_S$ is equal to 1 if $x \in S$ and to 0 otherwise). Note $\|\mathbf{1}_S\|_2 = \sqrt{\sum_{x \in S} 1^2} = \sqrt{|S|}$. Note also that for every $A, B \subseteq \{0,1\}^n$, $\sum_{x \in A, y \in B} M_{x,y} = \mathbf{1}_A^\dagger M \mathbf{1}_B$.

The discrepancy of the rectangle $A \times B$ is

$$\frac{1}{2^{2n}} \mathbf{1}_A^\dagger M \mathbf{1}_B \leq \frac{1}{2^{2n}} \lambda_{max}(M) \left| \mathbf{1}_A^\dagger \mathbf{1}_B \right| \leq \frac{1}{2^{2n}} \lambda_{max}(M) \sqrt{|A|\,|B|}\,,$$

where the last inequality uses Cauchy-Schwartz. ■

Example 13.15
The *mod 2 inner product* function defined as $f(x,y) = x \odot y = \sum_i x_i y_i (\bmod 2)$ has been encountered a few times in this book. To bound its discrepancy, let N be the $pm1$ matrix corresponding to f (i.e., $M_{x,y} = (-1)^{x \odot y}$). It is easily checked that every two distinct rows (columns) of N are orthogonal, every row has ℓ_2 norm $2^{n/2}$, and that $N^T = N$. Thus we conclude that $N^2 = 2^n I$ where I is the unit matrix. Hence every eigenvalue is either $+2^{n/2}$ or $-2^{n/2}$, and thus Lemma 13.14 implies that the discrepancy of a rectangle $A \times B$ is at most $2^{-3n/2}\sqrt{|A|\,|B|}$ and the overall discrepancy is at most $2^{-n/2}$ (since $|A|, |B| \leq 2^n$).

13.2.5 A technique for upper bounding the discrepancy

We describe an upper bound technique for the discrepancy that will later be useful also in the multiparty setting (Section 13.3). As in Section 13.2.4, we assume that f is a ± 1-valued function. We define the following quantity:

Definition 13.16
$\mathcal{E}(f) = \mathsf{E}_{a_1,a_2,b_1,b_2} \left[\prod_{i=1,2} \prod_{j=1,2} f(a_i, b_j) \right]$.

Note that $\mathcal{E}(f)$ can be computed, like the rank, in time polynomial in the size of the matrix $M(f)$. By contrast, the definition of discrepancy involves a maximization over all possible subsets A, B, and a naive algorithm for computing it would take time exponential in the size of $M(f)$. The following Lemma relates these two quantities.

Lemma 13.17

$$Disc(f) \leq \mathcal{E}(f)^{1/4}.$$

PROOF: The proof follows in two steps.

CLAIM 1: *For every function* $h:\{0,1\}^n \times \{0,1\}^n \to \{1,-1\}$, $\mathcal{E}(h) \geq (\mathsf{E}_{a,b}[f(a,b)])^4$.

We will use the Cauchy-Schwartz inequality, specifically, the version according to which $\mathsf{E}[z^2] \geq (\mathsf{E}[z])^2$ for every random variable z.

$$\mathcal{E}(h) = \mathop{\mathsf{E}}_{a_1,a_2}\left[\mathop{\mathsf{E}}_{b_1,b_2}\left[\prod_{i=1,2}\prod_{j=1,2} h(a_i,b_j)\right]\right] \tag{1}$$

$$= \mathop{\mathsf{E}}_{a_1,a_2}\left[\left(\mathop{\mathsf{E}}_{b}[h(a_1,b)h(a_2,b)]\right)^2\right] \tag{2}$$

$$\geq \left(\mathop{\mathsf{E}}_{a_1,a_2}\left[\mathop{\mathsf{E}}_{b}[h(a_1,b)h(a_2,b)]\right]\right)^2 \qquad \text{(Cauchy Schwartz)} \tag{3}$$

$$\geq \left(\mathop{\mathsf{E}}_{a,b}[h(a,b)]\right)^4. \qquad \text{(repeating previous two steps)} \tag{4}$$

CLAIM 2: *For every function* f *there is a function* h *such that* $\mathcal{E}(f) = \mathcal{E}(h)$ *and* $\mathsf{E}_{a,b}[h(a,b)] \geq$ *Disc*(f).

First, we note that for every two functions $g_1, g_2 : \{0,1\}^n \to \{-1,1\}$, if we define $h = f \circ g_1 \circ g_2$ as

$$h(a,b) = f(a,b)g_1(a)g_2(b)$$

then $\mathcal{E}(f) = \mathcal{E}(h)$. The reason is that for all a_1, a_2, b_1, b_2,

$$\prod_{i=1,2}\prod_{j=1,1} h(a_i,b_j) = g_1(a_1)^2 g_1(a_2)^2 g_2(b_1)^2 g_2(b_2)^2 \prod_{i=1,2}\prod_{j=1,2} f(a_i,b_j)$$

and the square of any value of g_1, g_2 is 1.

Now we prove Claim 2 using the probabilistic method. Fix $A, B \subseteq \{0,1\}^n$ and define two random functions $g_1, g_2 : \{0,1\}^n \to \{-1,1\}$ as below. First, for each $a \notin A$ pick a random value r_a in $\{-1,1\}$ and for each $b \notin B$ pick a random value s_b in $\{-1,1\}$. All random choices are independent of one another. Let

$$g_1(a) = \begin{cases} 1 & \text{if } a \in A \\ r_a & \text{else} \end{cases}$$

$$g_2(b) = \begin{cases} 1 & \text{if } b \in B \\ s_b & \text{else} \end{cases}$$

Let $h = f \circ g_1 \circ g_2$, and therefore $\mathcal{E}(h) = \mathcal{E}(f)$. Furthermore

$$\mathop{\mathsf{E}}_{g_1,g_2}\left[\mathop{\mathsf{E}}_{a,b}[h(a,b)]\right] = \mathop{\mathsf{E}}_{a,b}\left[\mathop{\mathsf{E}}_{g_1,g_2}[f(a,b)g_1(a)g_2(b)]\right] \tag{5}$$

$$= \frac{1}{2^{2n}} \sum_{a\in A, b\in B} f(a,b) \tag{6}$$

$$= \mathrm{Disc}(f) \tag{7}$$

where the second line follows from the fact that $\mathsf{E}_{g_1}[g_1(a)] = \mathsf{E}_{g_2}[g_2(b)] = 0$ for $a \notin A$ and $b \notin B$.

Thus in particular there *exist* g_1, g_2 such that $|\mathsf{E}_{a,b}[h(a,b)]| \geq \mathrm{Disc}(f)$. ■

We will see an example for a lower bound using this technique in Section 13.3.

13.2.6 Comparison of the lower bound methods

The tiling argument is the strongest lower bound technique, since bounds on rank, discrepancy and fooling sets imply a bound on $\chi(f)$, and hence can never prove better lower bounds than the tiling argument. Also, as Theorem 13.10, $\log \chi(f)$ fully characterizes the communication complexity of f up to a constant factor. The rank and fooling set methods are incomparable, meaning that each can be stronger than the other for some function. However, if we ignore constant factors, the rank method is always at least as strong as the fooling set method (see Exercise 13.6). Also, we can separate the power of these lower bound arguments. For instance, we know functions for which a polynomial gap exists between $\log \chi(f)$ and $\log \text{rank}(M(f))$. However, the following conjecture (we only state one form of it) says that rank is in fact optimal up to a polynomial factor.

Conjecture 13.18 (log rank conjecture)
There is a constant $c > 1$ such that $C(f) = O(\log(\text{rank}(M(f)))^c)$ for all f and all input sizes n, where rank is taken over the reals.

Of course, the difficult part of the above conjecture is to show that low rank implies a low-complexity protocol for f. Though we are still far from proving this, Nisan and Wigderson have shown that at least low rank implies low value of $1/\text{Disc}(f)$.

Theorem 13.19 ([**NisanWi95**])
$1/Disc(f) = O(rank(f)^{3/2})$.

13.3 Multiparty communication complexity

There is more than one way to generalize communication complexity to a multiplayer setting. The most interesting model turns out to be the "number on the forehead" model: each player has a string on his head which everybody else can see but he cannot. That is, there are k players and k strings $x_1, \ldots, x_k$, and Player i gets all the strings *except* for x_i. The players are interested in computing a value $f(x_1, x_2, \ldots, x_k)$ where $f : (\{0,1\}^n)^k \to \{0,1\}$ is some fixed function. As in the 2-player case, the k players have an agreed-upon protocol for communication (which was decided before they were given their strings), and all their communication is posted on a "public blackboard" that all of them can see (the protocol also determines the order in which the players write on the blackboard). The last message sent should contain (or at least easily determine) the value $f(x_1, \ldots, x_k)$ of the function on the inputs. By analogy with the 2-player case, we denote by $C_k(f)$ the number of bits that must be exchanged by the best protocol. Note that it is at most $n + 1$, since it suffices for any $j \neq i$ to write x_i on the blackboard, at which point the ith player knows all k strings and can determine and publish $f(x_1, \ldots, x_k)$.

Example 13.20
Consider computing the function

$$f(x_1, x_2, x_3) = \bigoplus_{i=1}^{n} \text{maj}(x_{1i}, x_{2i}, x_{3i})$$

in the 3-party model where x_1, x_2, x_3 are n bit strings. The communication complexity of this function is 3: each player counts the number of i's such that she can determine the majority of x_{1i}, x_{2i}, x_{3i} by examining the bits available to her. She writes the parity of this number on the blackboard, and the final answer is the parity of the players' bits. This protocol is correct because the majority for each row is known by either 1 or 3 players, and both are odd numbers.

Example 13.21 (Generalized Inner Product)
The *generalized inner product function* $GIP_{k,n}$ maps nk bits to 1 bit as follows

$$f(x_1, \ldots, x_k) = \bigoplus_{i=1}^{n} \bigwedge_{j=1}^{k} x_{ji}. \tag{8}$$

Notice, for $k = 2$ this reduces to the mod 2 inner product of Example 13.15.

For the 2-player model we introduced the notion of a monochromatic rectangle in order to prove lower bounds. Specifically, a communication protocol can be viewed as a way of partitioning the matrix $M(f)$: if the protocol exchanges c bits, then the matrix is partitioned into 2^c rectangles, all of which must be monochromatic if the protocol is valid.

The corresponding notion in the k-party case is a cylinder intersection. A *cylinder in dimension* i is a subset S of the inputs such that if $(x_1, \ldots, x_k) \in S$ then $(x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_k) \in S$ for all x_i' also. A *cylinder intersection* is $\cap_{i=1}^{k} T_i$ where T_i is a cylinder in dimension i. Since player i's communication does not depend upon x_i, it can be viewed as partitioning the set of inputs according to cylinders in dimension i. Thus we conclude that at the end of the protocol, the cube $\{0,1\}^{nk}$ is partitioned using cylinder intersections, and if the protocol communicates c bits, then the partition consists of at most 2^c monochromatic cylinder intersections. Thus we have proved:

Lemma 13.22 *If every partition of $M(f)$ into monochromatic cylinder intersections requires at least R cylinder intersections, then the k-party communication complexity is at least $\lceil \log_2 R \rceil$, where $M(f)$ is the k-dimensional table whose $(x_1, \ldots, x_k)^{th}$ entry is $f(x_1, \ldots, x_k)$.*

Discrepancy-based lower bound

In this section, we will assume as in our earlier discussion of discrepancy that the range of the function f is $\{-1, 1\}$. We define the *k-party discrepancy* of f by analogy to the 2-party case

$$\mathrm{Disc}(f) = \frac{1}{2^{nk}} \max_T \left| \sum_{(a_1, a_2, \ldots, a_k) \in T} f(a_1, a_2, \ldots, a_k) \right|,$$

where T ranges over all cylinder intersections.

To upper bound the discrepancy we introduce the k-party analogue of $\mathcal{E}(f)$. Let a *cube* be a set D in $\{0,1\}^{nk}$ of 2^k points of the form $\{a_{1,1}, a_{2,1}\} \times \{a_{1,2}, a_{2,2}\} \times \cdots \times \{a_{1,k}, a_{2,k}\}$, where each $a_{i,j} \in \{0,1\}^n$.

$$\mathcal{E}(f) = E_D \left[\prod_{\overline{a} \in D} f(\overline{a}) \right].$$

Notice that the definition of $\mathcal{E}(f)$ for the 2-party case is recovered when $k = 2$. The next lemma is also an easy generalization.

Lemma 13.23

$$\mathit{Disc}(f) \leq (\mathcal{E}(f))^{1/2^k}.$$

The proof is analogous to Lemma 13.17 and is left as Exercise 13.11. The only difference is that instead of defining 2 random functions we need to define k random functions $g_1, g_2, \ldots, g_k : \{0,1\}^{nk} \to \{-1,1\}$, where g_i depends on every one of the k coordinates except the ith.

Now we can prove a lower bound for the Generalized Inner Product (GIP) function. Note that since we changed the range to $\{-1,1\}$, this function is now defined as

$$GIP_{k,n}(x_1, x_2, \ldots, x_k) = (-1)^{\sum_{i \leq n} \prod_{j \leq k} x_{ji} (mod 2)}. \tag{9}$$

Theorem 13.24 (Lower bound for generalized inner product)
The function $GIP_{k,n}$ has k-party communication complexity $\Omega(n/4^k)$.

PROOF: We use induction on k. For $k \geq 1$ let $\beta_k = 1 - 2^{-k}$. Note that for every k, $\beta_{k+1} = \frac{1+\beta_k}{2}$. We claim that

$$\mathcal{E}(GIP_{k,n}) \leq (\beta_k)^n.$$

The base case $k = 1$ is trivial— in this case $GIP_{1,n}(x) = \prod(-1)^{x_i}$, and $\mathcal{E}(GIP_{1,n}) = \mathsf{E}_{a,b}[GIP_{1,n}(a)GIP_{1,n}(b)] = \mathsf{E}_a[GIP_{1,n}(a)]\,\mathsf{E}_b[GIP_{1,n}(b)] = 0$ because a, b are independent and GIP has an equal number of $+1$ and -1 outputs. Assuming truth for $k-1$ we prove for k. A random cube D in $\{0,1\}^{nk}$ is picked by picking $a_{11}, a_{21} \in \{0,1\}^n$ and then picking a random cube D' in $\{0,1\}^{(k-1)n}$.

$$\mathcal{E}(GIP_{k,n}) = \mathop{\mathsf{E}}_{a_{11},a_{21}} \left[\mathop{\mathsf{E}}_{D'} \left[\prod_{\overline{a} \in \{a_{11},a_{21}\} \times D'} GIP_{k,n}(\overline{a}) \right] \right] \tag{10}$$

Suppose the strings a_{11} and a_{21} agree on t coordinates. Examining the expression for $GIP_{k,n}$ in (9) we see that these coordinates contribute nothing once we multiply all the terms in the cube, since their contributions get squared and thus become 1. Thus the inductive hypothesis implies that the contribution is at most $(\beta_{k-1})^{n-t}$. Since the probability that two randomly chosen n-bit strings a_{11} and a_{21} agree on t coordinates is $\binom{n}{t}2^{-n}$, we conclude that

$$\mathcal{E}(GIP_{k,n}) \leq \sum_{t=0}^{n} \frac{\binom{n}{t}}{2^n} (\beta_{k-1})^{n-t} = \text{ (by binomial expansion)} \tag{11}$$

$$(\frac{1}{2} + \frac{\beta_{k-1}}{2})^n = \tag{12}$$

$$(\beta_k)^n\,. \tag{13}$$

This completes the proof since $(\beta_k)^n = (1 - 2^{-k})^n \sim e^{-n/2^k}$. Hence $Disc(f) = O(e^{-(n/2^k)2^{-k}}) = 2^{-\Omega(n/4^k)}$. ■

At the moment, we do not know of any explicit function f for which $C_k(f) \geq n2^{-o(k)}$ and in particular have no non-trivial lower bound for computing explicit functions $f : (\{0,1\}^n)^k \to \{0,1\}$ for $k \geq \log n$.

13.4 Overview of other communication models

We outline some of the alternative settings in which communication complexity has been studied.

Randomized protocols: One can consider *randomized* protocols for jointly computing the value of a function. In such protocols, all players have access to a shared random string r, which they use in determining their actions. We define $R(f)$ to be the *expected* number of bits communicated by the players. It turns out that randomization can sometimes make a significant

difference. For example, the equality function has a randomized communication protocol with $O(\log n)$ complexity (see Exercise 13.12). Nevertheless, there are techniques to prove lower bounds for such protocols as well.

Non-deterministic protocols: One can also define *non-deterministic communication complexity* analogously to the definition of the class **NP**. In a non-deterministic protocol, the players are both provided an additional third input z ("nondeterministic guess") of some length m that may depend on x, y. Apart from this guess, the protocol is deterministic. We require that $f(x, y) = 1$ iff there exists a string z that makes the players output 1, and the cost of the protocol is m plus the number of bits communicated. Once again, this can make a significant difference. For example both the *inequality* and *intersection* functions (i.e., the negations of the functions EQ and the function DISJ of Example 13.6) are easily shown to have logarithmic non-deterministic communication complexity. Analogously to the definition of **coNP**, one can define the co-non-deterministic communication complexity of f to be the non-deterministic communication complexity of the function $g(x, y) = 1 - f(x, y)$. Interestingly, it can be shown that if f has non-deterministic communication complexity k and co-non-deterministic communication complexity ℓ, then $C(f) \leq 10k\ell$, hence implying that in the communication complexity world the intersection of the classes corresponding to **NP** and **coNP** is equal to the class corresponding to **P**. In contrast, we believe that $\mathbf{P} \neq \mathbf{NP} \cap \mathbf{coNP}$.

Average case protocols: Just as we can study average-case complexity in the Turing machine model, we can study communication complexity when the inputs are chosen from a distribution $\mathcal{D}$. This is defined as

$$C_{\mathcal{D}}(f) = \min_{\mathcal{P}\text{protocol for } f} \mathsf{E}_{(x,y)\in_R\mathcal{D}}[\text{Number of bits exchanged by } \mathcal{P} \text{ on } x, y.]$$

Computing a non Boolean function: Here the function's output is not just $\{0, 1\}$ but an m-bit number for some m. We discuss one example in the exercises.

Asymmetric communication: In this model the "cost" of communication is asymmetric: there is some B such that it costs the first player B times as much to transmit a bit than it does the second player. The goal is to minimize the total cost.

Computing a relation: One can consider protocols that aim to hit a relation rather than computing a function. That is, we have a relation $R \subseteq \{0, 1\}^n \times \{0, 1\}^n \times \{1, 2, \ldots, m\}$ and given $x, y \in \{0, 1\}^n$ the players seek to agree on any $b \in \{1, 2, \ldots, m\}$ such that $(x, y, b) \in R$. See Exercise 13.13.

These and many other settings are discussed in [**kushNis**].

WHAT HAVE WE LEARNED?

- The *communication complexity* of a two input function f is the number of bits that a player holding x and a player holding y need to exchange to compute $f(x, y)$.
- Methods to lower bound the communication complexity of specific functions include the fooling set, tiling, rank, and discrepancy methods. Using these methods we have several examples of explicit functions on two n-bit inputs whose communication complexity is at least n.
- The *multiparty communication complexity* of a k-input function f is the number of bits that k parties need to exchange to compute f where the i^{th} player has all the inputs *except* the i^{th} input. The best known lower bound of the k-party communication complexity of an explicit function is of the form $n/2^{-\Omega(k)}$.
- Other models of communication complexity studies include probabilistic, nondeterministic, and average-case communication complexity, and the communication complexity of computing relations.

Chapter notes and history

This chapter barely scratched the surface of this self-contained mini-world within complexity theory; an excellent and detailed treatment can be found in the book by Kushilevitz and Nisan [**kushNis**] (though it does not contain some of the newer results).

Communication complexity was first defined by Yao [**yao79**]. Other early papers that founded the field were Papadimitriou and Sipser [**papadimitriouS84**], Mehlhorn and Schmidt [**mehlhornS82**] (who introduced the rank lower bound) and Aho, Ullman and Yannakakis [**ahoUY83**].

We briefly discussed parallel computation in Chapter 6. Yao [**yao79**] invented communication complexity as a way to lower bound the running time of parallel computers for certain tasks. The idea is that the input is distributed among many processors, and if we partition these processors into two halves, we may lower bound the computation time by considering the amount of communication that must necessarily happen between the two halves. A similar idea is used to prove time/space lower bounds for VLSI circuits. 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.

Communication complexity is also useful in time-space lower bounds for Turing machines (see Exercise 13.3), and circuit lower bounds (see Chapter 14).

Data structures such as heaps, sorted arrays, lists etc. are basic objects in algorithm design. Often, algorithm designers wish to determine if the data structure they have designed is the best possible. Communication complexity lower bounds can be used to establish such results. See [**kushNis**].

Yannakakis [**yannakakis91**] has shown how to use communication complexity lower bounds to prove lower bounds on the size of polytopes representing **NP**-complete problems. Solving the open problem mentioned in Exercise 13.10 would prove a lower bound for the polytope representing vertex cover.

Theorem 13.24 is due to Babai, Nisan and Szegedy, though our proof follows Raz's simplification [**Raz00**] of Chung's proof [**Chung90**].

Lovasz and Saks [**LovaszSa93**] have observed that the log rank conjecture is related to a conjecture in discrete mathematics concerning chromatic number and rank of the adjacency matrix.

The original log rank conjecture was that $C(f) = O(\log \text{rank}(M(f)))$ but this was disproved by Raz and Spieker [**RazSpi95**]. A comparison of rank and fooling set arguments appears in the paper by Dietzfelbinger, Hromkovic and Schnitger [**DietzfelbingerHrSc96**].

In general, the complexity of computing $C(f)$ and $C_k(f)$ is not understood, and this may have some connection to why it is difficult in practice for us to prove lower bounds on these quantities. It is also intriguing that the lower bounds that we do prove involve quantities such as rank and fooling sets that are computable in polynomial time given $M(f)$. (This is an instance of the more widespread phenomenon of *natural proofs* encountered in Chapter 23.) In this regard, it is interesting to note that the *Discrepancy* parameter is **NP**-hard to compute, but can be approximated within a constant multiplicative factor in the 2-player setting by a polynomial-time algorithm [**AlonNa06**]. In contrast, computing the discrepancy in the 3-player setting seems very hard (though no hardness results seem to appear anywhere); this may perhaps explain why lower bounds are so difficult in the multiplayer setting.

One relatively recent area not mentioned in this chapter is *quantum* communication complexity, where the parties may exchange quantum states with one another, see [**Brassard04**]. Interestingly, some techniques developed in this setting [**Sherstov07**] were used to obtain new $\Omega(n^{1/(2k)}/2^{2^k})$ lower bounds on the k-party communication complexity of the *disjointness* function [**LeeSh07**, **ChattopadhyayAd08**], thus obtaining a strong separation of non-deterministic and deterministic k-party communication complexity.

Exercises

13.1. Prove Lemma 13.5.

13.2. Prove that a single tape TM takes at least $O(n^2)$ to decide the language of *palindromes* $\mathsf{PAL} = \{x_n \cdots x_1 x_1 \cdots x_n : x_1, \ldots, x_n \in \{0,1\}^n, n \in \mathbb{N}\}$ of Example 1.1.

> **Hint:**Suppose this could be decided by a TM that travels at most k times from the $(n/3)^{th}$ position of the tape to the $(2n/3)^{th}$ position. Show that this implies an $O(k)$-bit communication protocol for deciding equality of $n/3$-bit long strings.

13.3. If $S(n) \leq n$, show that a space $S(n)$ TM takes at least $\Omega(n^2/S(n))$ steps to decide the language $\{x\#x : x \in \{0,1\}^*\}$.

13.4. Prove the second inequality of Theorem 13.8. That is, prove that for every $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, $C(f) = O(\log^2 \chi(f))$.

> **Hint:** Arbitrarily number the rectangles in the monochromatic tiling and let $N = \chi(f)$. Define graphs G_R, G_C on $\{1, \ldots, N\}$ where $\{i,j\}$ is an edge in G_R (resp., G_C) iff rectangles i, j share a row (resp., column). Let $deg_R(\cdot)$ and $deg_C(\cdot)$ denote degrees in these graphs. At each step, the row player tries to look for a rectangle i containing his input with $deg_L(i) \leq 3|G_R|/4$ and sends such an index i if it exists. Both players then remove from G_L, G_C all vertices that are not neighbors of i. Similarly, the column player tries to find a column j containing his input such that $deg_C(j) \leq 3|G_C|/4$. We claim if either such an i, j can be found, it represents progress— can you see why? Furthermore, can you show they will always find such i, j? It may be helpful to note that in a N-vertex graph with minimum degree at least $N/2 + 1$, each two vertices have a common neighbor.

13.5. Prove Lemma 13.10.

> **Hint:** First, show that for every two matrices A, B, $\text{rank}(A + B) \leq \text{rank}(A) + \text{rank}(B)$, implying that if $A = \sum_{i=1}^{\ell} \alpha_i B_i$ for rank-1 matrices $B_1, \ldots, B_\ell$ then

rank$(A) \leq \ell$. Then, use the fact that if A has rank at most ℓ then it has ℓ rows such that all other rows are linear combination of these rows to express A as a sum of ℓ rank-1 matrices $B_1, \ldots, B_\ell$ (the rows of the matrix B_i will be scalar multiples of some row of A).

13.6. Show that if a function f has a fooling set of size S then the rank argument can be used to give a lower bound of at least $\frac{1}{2}\lceil \log S \rceil$.

13.7. Show that if M is 0/1 real matrix, and M' is the ± 1 matrix obtained by applying the transformation $a \mapsto (-1)^a$ to the entries of M, then $\text{rank}(M) - 1 \leq \text{rank}(M') \leq \text{rank}(M) + 1$.

Hint: Use the fact that $M' = J - 2M$ where J is the all 1's matrix.

13.8. Consider x, y as vectors over $GF(2)^n$ and let $f(x, y)$ be their inner product mod 2. Prove using the rank method that the communication complexity is n.

Hint: Transform the problem to ± 1 first and compute rank over the reals. Could you prove this by taking rank in GF(2)?

13.9. Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be such that all rows of $M(f)$ are distinct. Show that $C(f) \geq \log n$.

Hint: Lower bound the rank.

13.10. 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 $O(\log^2 n)$ upper bound on the communication complexity.

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

13.11. Prove Lemma 13.23.

13.12. Prove that the randomized communication complexity of the equality function (i.e., $R(EQ)$) is at most $O(\log n)$. (Note that a randomized communication protocol is allowed to output the wrong answer with probability at most 1/3.)

Hint: Use the *fingerprinting* technique encountered in Section 7.2.3.

13.13. (Karchmer-Wigderson games [**karchmerW90**]) Consider the following problem about computing a relation. 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. (The maximum fan-in of each gate is 2.)

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

13.15. Show that the following computational problem is in **EXP**: given the matrix $M(f)$ of a Boolean function, and a number K, decide if $C(f) \leq K$.

(Open since Yao [**yao79**]) Can you show this problem is complete for some complexity class?