A Brief Introduction to Communication Complexity

Ronald de Wolf, rdewolf@cwi.nl

May 16, 2012

1 Introduction

The area of communication complexity deals with the following type of distributed problem. There are two separated parties, called Alice and Bob. Alice receives some input $x \in X$, Bob receives some $y \in Y$, and together they want to compute some function $f(x, y)$. As the value $f(x, y)$ will generally depend on both x and y, neither Alice nor Bob will have sufficient information to do the computation by themselves, so they will have to communicate in order to achieve their goal. In this model, individual *computation* is free, but *communication* is expensive and has to be minimized. How many bits do they need to communicate between them in order to solve this? Clearly, Alice can just send her complete input to Bob, but sometimes more efficient schemes are possible.

This model was introduced by Yao [Yao79] and has been studied extensively, both for its applications (like lower bounds on Boolean circuits and on all sorts of data structures) and for its own sake. Kushilevitz and Nisan [KN97] is the standard book about this area.

2 Deterministic protocols

First we sketch the setting for deterministic communication complexity. Alice and Bob want to compute $f : \mathcal{D} \to \{0, 1\}$. We will usually consider $\mathcal{D} = X \times Y$ with $X = Y = \{0, 1\}^n$, so both inputs are n-bit strings. Alice receives input $x \in X$, Bob receives input $y \in Y$, and their goal is to compute $f(x, y)$ on these inputs. Some often studied functions are:

- *Equality:* $\mathrm{EQ}(x, y) = 1$ iff $x = y$
- *Inner product:* $\mathrm{IP}(x, y) = x \cdot y \pmod 2 = \sum_i x_i y_i \pmod 2$
  (for $x, y \in \{0, 1\}^n$, x_i is the ith bit of x and $x \wedge y \in \{0, 1\}^n$ is the bit-wise AND of x and y)
- *Disjointness:* $\mathrm{DISJ}(x, y) = \mathrm{NOR}(x \wedge y)$. This function is 1 iff there is no i where $x_i = y_i = 1$
  (viewing x and y as characteristic vectors of sets, the function is 1 iff the two sets are disjoint)

A communication *protocol* is a distributed algorithm where first Alice does some individual computation, and then sends a message (of one or more bits) to Bob, then Bob does some computation and sends a message to Alice, etc. Each message is called a *round*. After one or more rounds the protocol terminates and both parties should know the outcome. The *cost* of a protocol is the total number of bits communicated on the worst-case input. A *deterministic* protocol for f always has to output the right value $f(x, y)$ for all $(x, y) \in \mathcal{D}$. Since we want both parties to know the final answer, we will assume that the last bit communicated is the output. We use $D(f)$ to denote the minimal cost among all deterministic protocols

for f. Note that if $X = \{0,1\}^n$ then always $D(f) \leq n+1$: the trivial protocol where Alice sends her complete input to Bob gives Bob full information (he already has y), so he can compute $f(x,y)$ and send it back.

Many functions can be computed with very little communication. For instance, suppose f is the parity function on $2n$ bits. Then Alice can just compute the parity of her n-bit x, send it over in 1 bit, Bob adds it to the parity of his y, and sends back the 1-bit result. Similarly, if f is the majority function, then Alice can just count the number of 1s in x and send over the result in $\log(n+1)$ bits.

2.1 Rectangles

A *rectangle* is a set $R \subseteq X \times Y$ that is of the form $R = A \times B$ with $A \subseteq X$ and $B \subseteq Y$. For example, if $n = 2$ and $A = \{00, 01\}$, $B = \{01, 10\}$ then $R = A \times B = \{(00,01), (00,10), (01,01), (01,10)\}$ is a rectangle. The following result is a fundamental property of deterministic protocols.

Lemma 1 *If a deterministic protocol has communication c, then there exist 2^c rectangles $R_1, \ldots, R_{2^c}$ that partition $X \times Y$, such that the protocol gives the same output a_i for each $(x,y) \in R_i$.*

We omit the easy proof of this lemma, which is by induction on c. For example, suppose there is only one k-bit message m going from Alice to Bob and then Bob returns the 1-bit output, then the 2^{k+1} rectangles would be of the form $R_{m,a} = A_m \times Y_{m,a}$, with $m \in \{0,1\}^k$ and $a \in \{0,1\}$, where A_m is the set of x's for which Alice sends k-bit message m, and $Y_{m,a}$ is the set of y's for which Bob returns output a when receiving message m. Note that if our protocol computes f correctly, then the rectangles are "monochrome": the protocol returns the same answer $f(x,y)$ for all $(x,y) \in R_i$.

2.2 Lower bounds: the rank method

Let M_f be the $|X| \times |Y|$ matrix whose entries are defined by $M_f(x,y) = f(x,y)$. This is called the *communication matrix* of f. It can be viewed as a 2-dimensional truth table. We use $\text{rank}(f)$ to denote the rank of this matrix over the field of real numbers. For example, the communication matrix for the equality function is the $2^n \times 2^n$ identity matrix, which has 1s on its diagonal and 0s elsewhere. Hence $\text{rank}(\text{EQ}) = 2^n$.

Suppose we have some c-bit deterministic protocol that computes f. By the previous section, this partitions the input space $X \times Y$ into rectangles $R_1, \ldots, R_{2^c}$. Each input (x,y) that has $f(x,y) = 1$, occurs in exactly one rectangle R_i for which $a_i = 1$, hence we have

$$M_f = \sum_{i:a_i=1} R_i,$$

where we view R_i as a $|X| \times |Y|$ matrix with 1s on its elements and 0s elsewhere. Note that R_i is a matrix of rank 1. Hence, using $\text{rank}(A+B) \leq \text{rank}(A) + \text{rank}(B)$, we get

$$\text{rank}(M_f) = \text{rank}\left(\sum_{i:a_i=1} R_i\right) \leq \sum_{i:a_i=1} \text{rank}(R_i) = \sum_{i:a_i=1} 1 \leq 2^c.$$

But that means that a lower bound on the rank of M_f implies a lower bound on the communication! We have proved:

Theorem 1 $D(f) \geq \log \text{rank}(f)$.

For example, for equality we have rank(EQ) $= 2^n$ and hence $D(\text{EQ}) \geq n$. This means that the trivial protocol for equality is optimal! (up to 1 bit, but the argument can be refined to get an $n+1$ lower bound.) Similarly, we can show that the matrices for disjointness and for inner product have full rank, and hence deterministic protocols need to communicate n bits to compute those functions as well. Also, if you pick a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ at random (by choosing all values $f(x,y)$ at random), then with high probability this will have rank$(f) = 2^n$ and hence requires maximal communication.

3 Randomized protocols

In a *randomized* protocol, Alice and Bob may flip coins and the protocol has to output the right value $f(x,y)$ with probability $\geq 2/3$ for all $(x,y) \in \mathcal{D}$. We can either allow Alice and Bob to toss coins individually (private coin) or jointly (public coin). We will opt for private coin flips here. This makes not much difference: a public coin can save at most $O(\log n)$ bits of communication [New91], compared to a protocol with a private coin. We use $R(f)$ to denote the minimal cost among all randomized protocols (with private coin) that compute function f.

3.1 An efficient protocol for the Equality problem

In one of the previous lectures we already described an efficient randomized protocol for testing if two polynomials are equal or not: just choose a random point from a sufficiently large field, and see if the two polynomials are equal on that point. Alice and Bob view their inputs $x = x_0 \dots x_{n-1}$ and $y = y_0 \dots y_{n-1}$ as the coefficients of single-variate polynomials of degree at most $n-1$:

$$A(z) = \sum_{i=0}^{n-1} x_i z^i \text{ and } B(z) = \sum_{i=0}^{n-1} y_i z^i.$$

Consider some field $\mathbb{F}$ of size $q \geq 3n$. If $x = y$ then $A(z) = B(z)$ for all $z \in \mathbb{F}$, but if $x \neq y$ then $A(z) \neq B(z)$ for at least 2/3 of the $z \in \mathbb{F}$ (by Schwartz-Zippel). Hence if Alice sends Bob a randomly chosen $z \in \mathbb{F}$ and the value $A(z)$, then with probability at least 2/3, Bob can give the right answer by computing $B(z)$ and comparing it with $A(z)$. The communication is only $O(\log n)$ bits, so we have proved that $R(\text{EQ}) = O(\log n)$. Accordingly, randomized protocols for equality are *exponentially* more efficient than deterministic protocols in this communication context.

3.2 Lower bounds: the discrepancy method

Randomized protocols flip coins, but we can fix these coins to obtain a deterministic protocol. Suppose randomized protocol A uses c bits of communication and has success probability $2/3$ on all inputs. Let $A(x,y,r_A,r_B) = 1$ if the protocol gives the correct output $f(x,y)$ on input x,y using specific coin flips r_A for Alice and r_B for Bob, and $A(x,y,r_A,r_B) = 0$ otherwise. For each input x,y we have

$$\mathbb{E}_{r_A,r_B}[A(x,y,r_A,r_B)] \geq 2/3,$$

where the expectation is taken over uniformly chosen strings r_A and r_B. Now let $\mu : X \times Y \to [0,1]$ be a probability distribution on the set of inputs for f. Then also

$$\mathbb{E}_{\mu,r_A,r_B}[A(x,y,r_A,r_B)] \geq 2/3,$$

where the expectation is taken over r_A, r_B, and x, y distributed according to μ. By the averaging principle, there exists a way to fix r_A and r_B such that the success probability (under μ) of the resulting *deterministic* protocol is at least $2/3$. Accordingly, if we want to lower bound the randomized communication complexity of a function, it suffices to find some "hard" input distribution μ, and to show that all *deterministic* protocols that have error at most $1/3$ under that distribution, need a lot of communication.

The reason the step to deterministic protocols is helpful, is that deterministic protocols partition the input space into rectangles as we've seen before. Suppose we can show that all "large" rectangles in the communication matrix have roughly as many 0s as 1s in them (weighed according to μ). Then the protocol will make a large error on all large rectangles. Conversely, if we know the protocol does not make a large error, most of its rectangles must have been "small". But that can only be if the protocol partitions the input set into many rectangles. Since the number of rectangles induced by the protocol is 2^c, the communication c must have been large. This idea leads to the following lower bound method. The *discrepancy* of rectangle $R = A \times B$ under μ is the difference between the weight of the 0s and the 1s in that rectangle:

$$\delta_\mu(R) = \left|\mu(R \cap f^{-1}(1)) - \mu(R \cap f^{-1}(0))\right|$$

The discrepancy of f under μ is the maximum over *all* possible rectangles:

$$\delta_\mu(f) = \max_R \delta_\mu(R).$$

If f has small discrepancy, that means that all "large" rectangles are roughly balanced. Suppose a deterministic protocol partitions the input space into rectangles $R_1, \ldots, R_{2^c}$. Suppose it has success probability $2/3$. The best thing that the protocol can do if it has to give one output a_i for all inputs in the rectangle R_i, is to set a_i to the bit-value with highest weight in that rectangle. This contributes $\mu(R_i \cap f^{-1}(a_i))$ to the success probability, and $\mu(R_i \cap f^{-1}(1-a_i)$ to the failure probability. Hence the overall success probability is $\sum_i \mu(R_i \cap f^{-1}(a_i))$, and the overall error probability is $\sum_i \mu(R_i \cap f^{-1}(1-a_i))$. Since the difference between these two has to be at least $2/3 - 1/3 = 1/3$, we have

$$\begin{aligned} 1/3 &\leq \sum_{i=1}^{2^c} \mu(R_i \cap f^{-1}(a_i)) - \sum_{i=1}^{2^c} \mu(R_i \cap f^{-1}(1-a_i)) \\ &\leq \sum_{i=1}^{2^c} |\mu(R_i \cap f^{-1}(a_i)) - \mu(R_i \cap f^{-1}(1-a_i))| = \sum_{i=1}^{2^c} \delta_\mu(R_i) \leq 2^c \delta_\mu(f). \end{aligned}$$

This is a lower bound on the communication: $c \geq \log(1/3\delta_\mu(f))$. Accordingly, a distribution μ where $\delta_\mu(f)$ is small gives a lower bound on the communication of deterministic protocols for f under μ, and then the same lower bound applies to randomized protocols. We have proved

Theorem 2 *For every input distribution μ we have $R(f) \geq \log(1/3\delta_\mu(f))$.*

3.3 Discrepancy of the inner product function

Now consider the inner product function, defined by $\mathrm{IP}(x, y) = x \cdot y \pmod 2$. We will show that its discrepancy under the uniform distribution is very small. We analyze the $2^n \times 2^n$ matrix M whose (x, y) entry is $(-1)^{x \cdot y}$. This is just the communication matrix for IP, with 0s replaced by 1s, and 1s replaced by -1s. Lindsey's lemma shows that large rectangles in M are quite balanced:

Lemma 2 (Lindsey) *For every rectangle $R = A \times B$, the absolute value of the sum of M-entries in that rectangle is at most $\sqrt{|A| \cdot |B| \cdot 2^n}$.*

Proof. It is easy to see that M is symmetric and $M^2 = 2^n I$. This implies, for every vector v,

$$\| Mv \|^2 = v^T M^T M v = 2^n v^T v = 2^n \| v \|^2.$$

Let $v_A \in \{0,1\}^{2^n}$ and $v_B \in \{0,1\}^{2^n}$ be the characteristic (column) vectors of the sets A and B. The sum of M-entries in R is $\sum_{a \in A, b \in B} M_{ab} = v_A^T M v_B$. We can bound this using Cauchy-Schwarz:

$$|v_A^T M v_B| \leq \| v_A \| \cdot \| M v_B \| = \| v_A \| \cdot \sqrt{2^n} \| v_B \| = \sqrt{|A| \cdot |B| \cdot 2^n}.$$

□

Let $\mu(x,y) = 1/2^{2n}$ be the uniform input distribution. Note that the discrepancy of the rectangle R under μ is exactly the difference of $+1$'s and -1's in R, divided by 2^{2n}. By Lindsey's lemma, this is $\delta_\mu(R) \leq \sqrt{|A| \cdot |B|}/2^{3n/2}$. Because $|A|$ and $|B|$ are each at most 2^n, it follows that the discrepancy of the inner product function under the uniform distribution is $\delta_\mu(\text{IP}) \leq 2^{-n/2}$. Hence Theorem 2 implies a lower bound of $n/2 - \log(3)$ bits on the randomized communication complexity of IP.

Also for the disjointness function one can prove a linear randomized lower bound, but that requires a new technique (beyond the scope of this introduction), because its discrepancy cannot be made very small: $\delta_\mu(\text{DISJ}) \geq 1/(2n+1)$ for every input distribution μ.

4 Application: lower bound on the area-time tradeoff in chips

Communication complexity has many applications in other computational settings, in particular for proving lower bounds. These applications all have the same flavor: show that, as part of its task, your model is actually computing some communication problem $f(x,y)$. Lower bounds on the communication complexity of f then imply lower bounds for the model at hand. We give a simple example here.

Suppose we have a *chip* that computes a function $f : \{0,1\}^m \to \{0,1\}$. Abstractly, the chip can be viewed as a planar rectangle with m input ports and one output port. Its width a and height b are measured in units Δ, which is the minimal width of a wire. The area is $A = ab$. The chip works in cycles. In each cycle, ports can do local computation, and can send a bit (across some wire) to another port. The time T is the number of cycles that the chip uses for its computation. Now make some imaginary "cut" in the chip, call the m_A input ports on the left "Alice" (and call their m_A bits x), and call the m_B input ports on the right "Bob" (with m_B-bit input y). We can make the cut so that only $O(\sqrt{A})$ wires go between left and right. Note that the chip solves the communication complexity problem $f(x,y)$ using only $O(T\sqrt{A})$ many bits of communication: in each cycle it only sends $O(\sqrt{A})$ bits between left and right. Hence the communication complexity $D(f)$ (for our specific split into A- and B-variables) gives a lower bound on $T\sqrt{A}$.

References

[KN97] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[New91] I. Newman. Private vs. common random bits in communication complexity. *Information Processing Letters*, 39(2):67–71, 1991.

[Yao79] A. C-C. Yao. Some complexity questions related to distributive computing. In *Proceedings of 11th ACM STOC*, pages 209–213, 1979.