Communication Complexity and Applications

Lecturer: Toniann Pitassi

1 Introduction

In this course we will define the basic two-party model of communication, as introduced in the seminal 1979 paper by Yao. We will discuss different measures of complexity for the basic model, focusing mostly on deterministic, randomized and nondeterministic complexities of protocols. We will introduce the very important set disjointness function and show that it has near maximal communication complexity. We will also introduce extensions of the basic model to more than two players– the number-in-hand model, as well as the number-on-forehead model. We will show lower bounds for set disjointness in these models as well, as well as give a general discussion of lower bound methods, and the state-of-the-art in terms of lower bounds.

We will then switch to discuss the many fundamental applications of communication complexity for a variety of problems: data structures, streaming, privacy, machine learning, game theory, extended formulations, proof complexity, distributed computing, and circuit lower bounds. We will cover some recent applications in depth, with a particular focus on lower bounds for extended formulations of linear programs.

2 The model

Let X, Y, Z be arbitrary finite sets and let $f : X \times Y \to Z$ be an arbitrary function. There are two players, Alice and Bob, who wish to compute the function $f(x, y)$. The main obstacle is that Alice *only* knows x and Bob *only* knows y. Thus, to compute the value $f(x, y)$, they will need to communicate with each other. We are assuming that they both follow a fixed protocol agreed upon beforehand. The protocol consists of the players sending bits to each other until the value of f can be determined.

We are only interested in the amount of communication between Alice and Bob, and we wish to ignore the question of the internal computations of each player. Thus, we assume that Alice and Bob have *unlimited* computational power. The cost of a protocol Π is the *worst* case cost of Π over all inputs (x, y). The complexity of f is the minimum cost of a protocol that computes f.

Formally how do we specify a protocol? In each step one of the players sends one bit of information to the other player. The bit depends on the input of the player who sends it, and all the previous bits communicated so far.

In every step, a protocol specifies:

1. Which player sends the next bit;
2. Value of this bit (as a function of that players' input, and history so far).

Without loss of generality, we can assume that the players always alternate, and that the last bit sent is the value $\Pi(x, y)$ output by the protocol. This only increases the length of the protocol by a constant factor.

We say that a protocol Π computes a function f if for all x, y, $f(x, y) = \Pi(x, y)$. Usually we set $X = Y = \{0,1\}^n$, and $Z = \{0,1\}$. In this case, the cost of a protocol, $c(n)$, on inputs x, y, of length n is the worst case cost of Π over all inputs of length n, and the communication complexity of f is the minimum cost $c(n)$ over all protocols that compute f.

Another view of a protocol which may be more convenient is the following:

Definition 1. *A protocol Π over $X \times Y$ with range Z is a binary tree where each internal node v is labelled either by a function $a_v : X \to \{0,1\}$ or by a function $b_v : Y \to \{0,1\}$*[1]*, and each leaf is labelled with an element of Z. The value of the protocol Π on input (x, y) is the label of the leaf reached by starting from the root, and traversing the tree. The cost of the protocol Π on input (x, y) is the length of the path on input (x, y). The cost of the protocol Π is the height of the tree.*

A simple general protocol : Let any function $f(x, y)$, with $|x| = |y| = n$. Alice sends x. Bob sends $f(x, y)$. The total communication is $n+1$ bits. Therefore, the (deterministic) communication complexity of any boolean function is at most $n + 1$. However, for many functions, we can develop much more efficient protocols, i.e., protocols with poly-logarithmic communication bits for specific functions.

Next, we give some examples of functions that we will study in the up-coming lectures.

Example 1 (Parity). *The parity function of (x, y) has value 1 if x, y have the same parity. A simple protocol is the following: Alice sends the parity of x (1 if the number of 1's in x is odd, and 0 otherwise). Then Bob replies 1 if and only if the parity of y is equal to the parity of x.*

Parity is easy for deterministic communication.

Example 2 (Equality). *Equality function: $EQ(x, y) = 1$ iff $x = y$.*

We will see shortly that Equality requires maximal communication complexity (linear in n) but has constant-cost randomized protocols.

Example 3 (Set disjointness). *$DISJ(x, y) = 1$ iff there exists i such that $x_i = y_i = 1$. That is, we think of x, y as characteristic vectors of subsets of $[n]$ and we want to output 1 if the sets intersect, and 0 of they are disjoint. A related problem is the set disjointness problem where x, y are subsets of size $k << n$.*

We will see in upcoming lectures that set disjointness requires linear randomized communication complexity (and thus also linear deterministic communication complexity). However it has low cost nondeterministic complexity. Set disjointness is the most important function in communication complexity as it is complete for nondeterministic communication complexity, and therefore shares the same important stature that the satisfiability problem (SAT) has in ordinary Turing machine time complexity. However unlike SAT where is is a major open problem to prove (or disprove) that SAT is not solvable in polynomial time, a major result in communication complexity establishes that set disjointness is hard for deterministic and randomized complexity. We will cover two proofs of this result later in the course. Nearly all of the applications of communication complexity that we will discuss are obtained by reductions to the disjointness problem.

[1] If a node is labelled by a_v intuitively means that Alice is sending a bit at this point, similarly for b_v and Bob.

Example 4 (Inner product)**.** *The inner product function is defined as* $IP(x, y) = \sum_{i=1}^{n} x_i y_i$ (mod 2)*.*

t

Example 5 (Clique versus Independent Set)**.** *Let* G *be a fixed undirected graph with* n *vertices. Alice is given a subset* $S \subset [n]$ *such that the vertices in* S *form a clique in* G*, and Bob is given a subset* $T \subset [n]$ *such that the vertices in* T *form an independent set in* G*. They want to output 1 if* S *and* T *intersect, and 0 otherwise.*

We will see in upcoming lectures that inner product is hard for nondeterministic, randomized and deterministic communication complexity.

3 Randomized vs. Deterministic CC

Definition 2. *For a function* $f : X \times Y \to Z$*, the (deterministic) communication complexity of* f*, denoted by* $\mathsf{P}^{\mathsf{cc}}(f)$*, is the minimum cost of* Π*, over all protocols* Π *that compute* f*.* $\mathsf{P}^{\mathsf{cc}}(f)$ *is a function of* n*, the length of* x *and* y*.*

In the probabilistic case, players can toss random bits. There are two models depending on whether the coin tosses are public or private. In the public random string model the players share a common random string, while in the private model each player has his/her own private random string. The following definitions apply to the model with private strings. For example, $\mathsf{BPP}_{\varepsilon}^{\mathsf{cc}}(f)$ denotes the randomized communication complexity of computing f with two-sided error ϵ, in the private coin model. In the public coin model, we denote the randomized BPP communication complexity of f by $\mathsf{BPP}_{\varepsilon}^{\mathsf{cc}pub}(f)$.

Definition 3. *Let* Π *be a randomized protocol.*

Zero-sided error: Π *computes a function* f *with zero-sided error if for every* (x, y)*,*

$$\Pr[\Pi(x, y) \;=\; f(x, y)] = 1.$$

The cost of a zero-sided error protocol Π *for* f *is the minimum over all inputs* (x, y)*, of the expected communication complexity of* Π *on* (x, y)*.*

One-sided error: Π *computes a function* f *(with one sided error* ε*) if for every* (x, y) *such that* $f(x, y) = 0$*,*

$$\Pr[\Pi(x, y) \;=\; 0] \;=\; 1,$$

and for every (x, y) *such that* $f(x, y) = 1$*,*

$$\Pr[\Pi(x, y) \;=\; 1] \;\geq\; 1 - \varepsilon, i$$

Two-sided error: Π *computes a function* f *(with error* ε*) if*

$$\forall\ x \in X,\ y \in Y, \quad \Pr\left[\Pi(x, y) = f(x, y)\right] \geq 1 - \varepsilon,$$

Definition 4. *Let $f : X \times Y \to \{0,1\}$ be a function. We consider the following complexity measures for f:*

- $\mathsf{ZPP}^{\mathsf{cc}}(f)$ *is the minimum average case cost of a randomized protocol that computes f with zero error.*
- *For $0 < \varepsilon < 1/2$, $\mathsf{BPP}^{\mathsf{cc}}_{\varepsilon}(f)$ is the minimum worst case cost of a randomized protocol that computes f with error ε. We define $\mathsf{BPP}^{\mathsf{cc}}(f) = \mathsf{BPP}^{\mathsf{cc}}_{1/3}(f)$.*
- *For $0 < \varepsilon < 1/2$, $\mathsf{RP}^{\mathsf{cc}}_{\varepsilon}(f)$ is the minimum worst case cost of a randomized protocol that computes f with one-sided error ε. We define $\mathsf{RP}^{\mathsf{cc}}(f) = \mathsf{RP}^{\mathsf{cc}}_{1/3}(f)$.*

Lemma 1. *(Markov)* $\mathsf{RP}^{\mathsf{cc}}_{\epsilon}(f) \leq \frac{1}{\epsilon}\mathsf{ZPP}^{\mathsf{cc}}(f)$.

Lemma 2. $\mathsf{BPP}^{\mathsf{cc}}(f) = \Omega(\log \mathsf{P}^{\mathsf{cc}}(f))$.

Now, let's give an example for the above two protocols for the equality function.

Example 6 (Equality Revisited)**.** *Recall that $EQ(x, y) = 1$ iff $x = y$. Let's analyse the randomized communication complexity in the public and private coin protocol for the function EQ:*

Public Coin *Let $x \in X$, $y \in Y$, $X = Y = \{0,1\}^n$ be the input strings, and let $r \in \{0,1\}^n$ be the public coin tosses. The protocol is the following: Alice computes the bit $a = (\sum_{i=1}^n x_i r_i) \pmod 2$ and sends it to Bob. Then Bob computes $b = (\sum_{i=1}^n y_i r_i) \pmod 2$. The value of the protocol is*

$$P(x, y, r) = 1 \quad \textit{iff} \quad \sum_{i=1}^{n} x_i r_i = \sum_{i=1}^{n} y_i r_i \pmod 2.$$

Note that the communication is only two bits! Now let's analyse this protocol. If $x = y$, then for all r, the protocol is correct, i.e., $P(x, y, r) = 1$. If $x \neq y$, then with probability $1/2$ (over the public coin tosses) $P(x, y, r) = 1$, i.e., our protocol is wrong. If we repeat the above random experiment c times independently, then the probability that our protocol is wrong on all of the executions is $1/2^c$. This protocol has $O(1)$ probabilistic communication complexity when the error ϵ is a consstant.

Private Coin *In this setting, encode the inputs $x = x_0, \ldots, x_{n-1}$ and $y = y_0, \ldots y_{n-1}$ as the coefficients of single variate polynomials of degree at most $n - 1$:*

$$A(z) = \sum_{i=0}^{n-1} x_i z^i,$$

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

Consider some field F of size $q \geq 3n$. If $x = y$ then $A(z) = B(z)$ for all $z \in F$, but if $x \neq y$, then $A(z) \neq B(z)$ for at least 2/3 of the $z \in F$ (by Schwartz-Zippel). Thus, our protocol is as follows.

1. *Alice samples a randomly chosen $z \in F$ and the value $A(z)$, and sends Bob z and $A(z)$.*
2. *Bob sends 1 if and only if $B(z) = A(z)$.*

Thus Bob computes the right answer with probabilisty at least 2/3. Note that the communication is only $O(logn)$ bits.

3.1 Newman's Theorem

The above example gives two different randomized protocols for equality; with public coins the protocol has constant cost, and with private coins the protocol has cost $O(\log n)$. Furthermore it is known that the equality function requires $\Omega(\log n)$ bits in the private coin model, and thus there can be an additive $O(\log n)$ savings in the public coin model. The following theorem due to Newman states that this gap is as large as possible, since any public coin protocol can be transformed into a private coin protocol with a small penality in the error and a small additive penalty in the communication complexity.

Theorem 3. *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a function. For every $\delta > 0$ and every $\epsilon > 0$,*

$$\mathsf{BPP}^{\mathsf{cc}}_{\epsilon+\delta}(f) \leq \mathsf{BPP}^{\mathsf{cc}pub}_{\epsilon}(f) + O(\log n + \log \delta^{-1})$$

Proof. We will prove that any public coin protocol Π with error ϵ can be transformed into another public coin protocol, Π' such that: (i) the communication complexity of Π' is the same as that of Π; (ii) Π' uses only $O(\log n + \log \delta^{-1})$ random bits; and (iii) the error of Π' is at most $\epsilon + \delta$. The theorem follows since Π' can be easily converted to a private coin protocol with the desired parameters: first Alice will privately flip that many random coins, and then send them to Bob, and then they proceed to follow the protocol Π'.

Let $Z(x, y, r)$ be a random variable that is equal to 1 if $\Pi(x, y, r)$ outputs an incorrect answer, and 0 otherwise. Because Π has error ϵ, $E_r[Z(x, y, r)] \leq \epsilon$ for every x, y. Let $r_1, \ldots, r_t$ be random strings, where we will soon set $t = O(n/\delta^2)$, and define $\Pi_{r_1,\ldots,r_t}(x, y)$ as follows: Alice and Bob choose $i \leq t$ at random, and then run $\Pi(x, y, r_i)$. We will prove that there exist strings $r_1, \ldots, r_t$ such that $E_i[Z(x, y, r_i)] \leq \epsilon + \delta$ for all (x, y). For this choice of strings, the protocol $\Pi_{r_1,\ldots,r_t}$ will be our desired protocol, Π'.

We use the probabilistic method to show the existence of $r_1, \ldots, r_t$. Choose $r_1, \ldots, r_t$ at random, and consider a particular input pair (x, y). The probability that $E_i[Z(x, y, r_i)] > \epsilon + \delta$ is exactly the probability that $1/t \sum_{i=1}^{t} Z(x, y, r_i) > \epsilon + \delta$. By Chernoff inequality, since $E_r[Z(x, y, r)] \leq \epsilon$,

$$Pr_{r_1,\ldots,r_t}[1/t \sum_{i=1}^{t} Z(x, y, r_i) - \epsilon) > \delta] \leq 2e^{-2\delta^2 t}.$$

By choosing $t = O(n/\delta^2)$, this is smaller than 2^{-2n}. Thus for a random choice of $r_1, \ldots, r_t$, the probabilty that there exists a bad (x, y) such that $E_i[Z(x, y, r_i)] > \epsilon + \delta$ is smaller than $2^{2n}2^{-2n} = 1$ (by the union bound). Thus there exists $r_1, \ldots, r_t$ such that for every (x, y) the error of $\Pi_{r_1,\ldots,r_t}$ on (x, y) is at most $\epsilon + \delta$. It is easy to check that the number of random bits used by the protocol $\Pi_{r_1,\ldots,r_t}$ is $\log t = O(\log n + \log \delta^{-1})$, and that the communication complexity of $\Pi_{r_1,\ldots,r_t}$ is the same as that of Π. $\square$

4 Nondeterministic/co-Nondeterministic CC

In the non-deterministic model of communication complexity, players share *nondeterministic* bits z. Now a protocol is a function of x, y, z, and we say that a protocol P computes a function f if for all x, y:

$$\begin{aligned} f(x,y) = 1 &\implies \exists z\ P(x,y,z) = 1 \\ f(x,y) = 0 &\implies \forall z\ P(x,y,z) = 0. \end{aligned}$$

The communication complexity in this model is defined as the maximum length of z plus the number of bits exchanged over all x, y. Similarly, exchanging the position of the existential and for all quantifier we can define co-nondeterministic CC. The basic example is set disjointness (see Example 3). If the players share $\log n$ nondeterministic bits, they guess i and check if $x_i = y_i = 1$.

Define $\mathsf{NP}^{\mathsf{cc}}(f)$ to be the nondeterministic communication complexity of f, and similarly, let $\mathsf{coNP}^{\mathsf{cc}}(f)$ denote the co-nondeterministic communication complexity of f.

5 Communication Complexity Classes

To summarize, we define the following measures of communication complexity for a two-player function f.

Definition 5. *Let f be a two-player function.*

1. $\mathsf{P}^{\mathsf{cc}}(f)$ *is the deterministic communication complexity of f.*
2. $\mathsf{RP}^{\mathsf{cc}}(f)$ *is the randomized one-sided communication complexity of f with error* $1/3$
3. $\mathsf{BPP}^{\mathsf{cc}}(f)$ *is the randomized two-sided communication complexity of f with error* $1/3$.
4. $\mathsf{NP}^{\mathsf{cc}}(f)$ *is the nondeterministic communication complexity for f*
5. $\mathsf{PP}^{\mathsf{cc}}(f)$ *is the randomized communication complexity of f with unbounded error. That is, on all instances, the protocol is correct with probability greater than* $1/2$.
6. $\mathsf{ZPP}^{\mathsf{cc}}(f)$ *is the randomized zero-error communication complexity of f.*

We have the following easy relationships:

$$\mathsf{P}^{\mathsf{cc}}(f) \geq \mathsf{RP}^{\mathsf{cc}}(f) \geq \mathsf{BPP}^{\mathsf{cc}}(f).$$

We also have

$$\mathsf{P}^{\mathsf{cc}}(f) \geq \mathsf{RP}^{\mathsf{cc}}(f) \geq \mathsf{NP}^{\mathsf{cc}}(f).$$

Remark: Unlike the analagous classical complexity classes where the above relationships are not know to be proper, in the communication complexity world, all of the above inequalities are provably proper. (That is, for each inequality $C_1 \geq C_1$, there exists a function f such that $C_1(f)$ is exponentially larger than $C_2(f)$.)

6 Applications

Communication complexity arguments have found numerous application to a large number of different areas. Applications include the following.

1. *Bisection width of networks*
2. *VLSI*
3. *Data structures – cell probe model and dynamic data structures*
4. *Boolean circuit complexity, branching program complexity*
5. *Quantum complexity*
6. *Extension Complexity of linear programming and semi-definite programming.*
7. *Turing machine time-space trade-offs*
8. *Streaming algorithms*
9. *Game theory (truthfulness vs. accuracy, mechanism design)*
10. *Differential privacy*
11. *Property Testing*
12. *Differential Privacy*
13. *Learning Theory*
14. *Proof complexity*
15. *Graph Theory (Alon-Seymour Conjecture)*