Lecture 9: Communication complexity

Rajat Mittal

IIT Kanpur

We move on to another very interesting area of complexity theory, called *communication complexity*. It is deeply related to query complexity. Communication complexity has been instrumental in proving bounds in various areas of computer science: VLSI, data structures, circuit complexity, streaming algorithms, game theory, proof complexity (to name a few).

Many a times query complexity can be used to solve problems in communication complexity. We will give a basic introduction to the model, see some lower bounding techniques, and discuss the randomized version too. We will finish by looking at the connection between communication complexity and query complexity.

1 The model

In an informal sense, communication complexity of a function is the minimum amount of communication needed by two computationally unbounded parties (sometimes more) to compute the function if they are given different part of the input. The important points to remember are:

– The parties are computationally unbounded and have complete knowledge of the function. Though, they are missing the part of the input given to other party.
– We are only interested in the number of bits communicated between the parties.

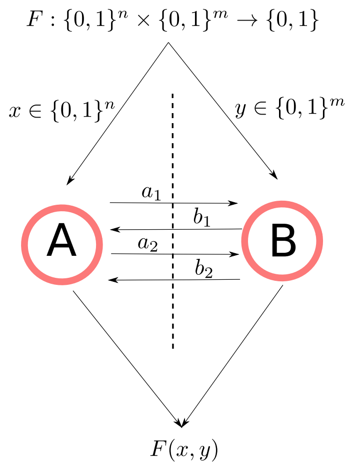

Fig. 1. Communication complexity setup for computing F.

Let us formalize the model. Note that we are restricting ourselves to the most studied setup (there are many other settings). Suppose we are interested in the communication complexity of a function $F : \mathcal{X} \times \mathcal{Y} \to$

$\{0,1\}$; generally $\mathcal{X} \subseteq \{0,1\}^n$ and $\mathcal{Y} \subseteq \{0,1\}^m$. The two parties, Alice and Bob, get $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ respectively. Their goal is to compute $F(x,y)$ by exchanging minimum number of bits between them.

Once again we are interested in the worst case communication (over the inputs) of a communication protocol. In other words, we want to find the best communication protocol/strategy whose worst case communication is the least. Notice that Alice and Bob are cooperating players, and can decide any strategy together before they receive their respective inputs x and y. Look at Figure 1 for a pictorial representation.

We are interested in the asymptotic communication complexity of these functions. So, it does not matter who outputs the answer, because they can share the answer using only one bit. Similarly, we can assume that Alice and Bob take turns while communicating, and transfer only one bit at a time.

Exercise 1. Convince yourself that the above assumption only increases the communication complexity by a constant factor.

Similarly, it is easy to see an upper bound on the communication complexity.

Exercise 2. Show a trivial protocol which succeeds with communication $O(\min(n,m))$.

Let us take a very simple example. What is the communication complexity of $F(x,y) = (\sum_i (x_i \oplus y_i))$ mod 2? The answer turns out to be very simple if we rearrange the terms, $F(x,y) = (\sum_i x_i) \mod 2 \oplus (\sum_i y_i)$ mod 2.

Exercise 3. Can you give a protocol which works by communicating only one bit?

With the rearrangement, it is easy to see that Alice can compute the parity of her bits, and send the answer (one bit) to Bob. Bob can calculate parity of his part and calculate $F(x,y)$. So the communication complexity of this Parity function is 1.

A convenient way to represent the function F is by a $\mathcal{X} \times \mathcal{Y}$ matrix, where $M_F[x,y] = F(x,y)$. This will be called the communication matrix of F.

Representation of a protocol as a tree: Similar to query algorithm, a communication protocol can also be viewed as a tree, look at Figure 2.

In this case we are assuming that Alice and Bob take turns and communicate one bit at a time. The root node corresponds to the first bit sent by Alice, a_1. The two children correspond to this bit being 0 or 1. The two nodes on the second layer correspond to Bob's first message.

Here comes the important part. We can partition the inputs of Alice into two parts, $S_1 \subseteq \{0,1\}^n$ and $\bar{S_1}$, where S_1 consists of all x where Alice is supposed to communicate 0. In this case, after the first message being 0, anyone just watching the communication knows that Alice and Bob's inputs are in the set $S_1 \times \{0,1\}^m$. If the message was 1, their inputs are in the set $\bar{S_1} \times \{0,1\}^m$.

Exercise 4. Interpret the second row of nodes in the similar manner in Figure 2.

Next comes another very important point, what is the structure of the remaining inputs (for someone watching the communication) at any stage of the communication protocol? These sets are what we call a *combinatorial rectangle*. A combinatorial rectangle for $S \in \mathcal{X}, T \in \mathcal{Y}$ is the set of all inputs x, y where $x \in S$ and $y \in T$.

Exercise 5. Why are we calling it a rectangle?

A good way to understand a combinatorial rectangle is, if (x,y) and (u,v) are in the combinatorial rectangle then so do (x,v) and (u,y). Moving forward, we will just call a combinatorial rectangle to be a rectangle.

We can infer from the protocol tree (and induction) that the set of remaining possible inputs at any stage will form a rectangle of the communication matrix. If the protocol has to succeed, the final combinatorial rectangle (for any sequence of messages) should have the same value, called a *monochromatic rectangle*.

In other words, a communication protocol partitions the communication matrix into monochromatic rectangles.

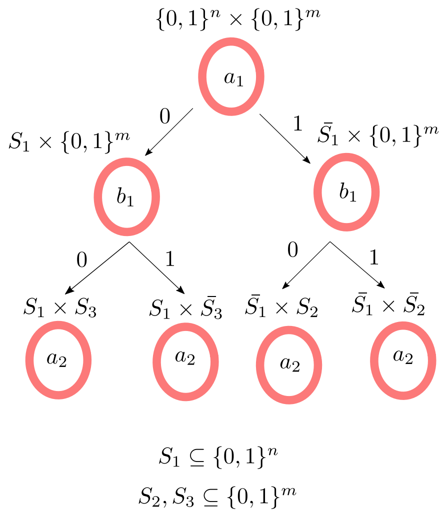
$\{0,1\}^n \times \{0,1\}^m$
a_1
0
1
$S_1 \times \{0,1\}^m$
$\bar{S}_1 \times \{0,1\}^m$
b_1
b_1
0
1
0
1
$S_1 \times S_3$
$S_1 \times \bar{S}_3$
$\bar{S}_1 \times S_2$
$\bar{S}_1 \times \bar{S}_2$
a_2
a_2
a_2
a_2
$S_1 \subseteq \{0,1\}^n$
$S_2, S_3 \subseteq \{0,1\}^m$

Fig. 2. Tree representation of a communication protocol.

Exercise 6. How many monochromatic rectangles do you get at the end of a communication protocol with c bits of communication?

You can see that we get 2^c many monochromatic rectangles for a protocol of c bits of communication. Flipping it, we get a lower bound technique for communication complexity of F. If any partition of the communication matrix of F needs at least m monochromatic rectangles, we get that the communication complexity of F is at least $\log m$. Define $\chi(F)$ to be the logarithm of the minimum number of monochromatic rectangles needed to partition M_F. We get that $C(F) \geq \chi(F)$, where $C(F)$ is the communication complexity of F.

Exercise 7. Is this lower bound tight?

Unfortunately, every partition of monochromatic rectangles need not correspond to a communication protocol (see [1]). This means that $C(F)$ need not be $\Theta(\chi(F))$. Luckily, we can show that $C(F) \leq \chi(F)^2$ (the idea of the proof is taken from [1]).

Theorem 1. *For any communication function F,*

$$\Omega(\chi(F)) = C(F) = O(\chi(F)^2).$$

One way to get to the correct rectangle is, Alice sends all rectangles containing x, Bob's input will lie in a unique rectangle out of those sent by Alice (why?). This requires a lot of communication. Instead we will try to send less information (say the name of just one rectangle) and try to cut down the number of alive rectangles by half.

Which of the Alice's rectangle is important (should be sent to Bob?). We need the following terminology. A rectangle intersects the row x if there exist a y such that (x, y) is in the rectangle. Two rectangles intersect row-wise if there is a row where they intersect. Notice that does not mean that there is common entry between the two rectangles. Similarly we can define column-wise intersection of two rectangles.

If Alice says that her input is in rectangle U, all rectangles which do not intersect U row-wise can be discarded.

Exercise 8. Alice's input might be present in many rectangles, which one should she send to Bob?

If Alice knows a rectangle (containing x) which intersects with less than half the rectangles rowwise, we can cut down the rectangles by half. Similarly, if Bob knows a rectangle (containing y) which only intersects with less than half the rectangles, we are done. The following claim allows us to cut down the list of rectangles by half at any point in the algorithm.

Claim. Let R be a list of disjoint rectangles. There cannot be a rectangle in R which intersects more than half the rectangles in R row-wise as well as more than half the rectangles in R column-wise.

Proof. You can try to prove it on your own first.

Exercise 9. Prove the claim.

Suppose, for the sake of contradiction, there is a rectangle S which falsifies the premises of the claim. Since the size of row-wise intersecting rectangles and column-wise intersecting rows is more than half. There exist a rectangle T which intersects S row-wise as well as column-wise.

Exercise 10. Show that S and T intersect at a point (x, y).

But this gives us a contradiction, since R is a disjoint partition. □

With this claim, we are ready to prove Theorem 1.

Proof of Theorem 1. The lower bound has already been proved. For the upper bound, we need to show a protocol which works with communication of $O(\chi(F)^2)$ bits.

Let $r = 2^{\chi(F)}$ be the number of monochromatic rectangles in the best disjoint partition of M_F. The strategy for creating a protocol of size $\chi(F)^2$ is akin to binary search. We will transfer the name of a rectangle in each round (taking $\log r = \chi(F)$ bits of communication) and reduce the number of *alive* rectangles by half (total communication being $\chi(F)^2$).

Note 1. A rectangle is alive if the given input (x, y) could be in that rectangle for someone watching the communication. By definition, the rectangle containing (x, y) (unique one) will always be alive. At any point in the protocol R will represent the list of alive rectangles.

You can already think of the protocol. Alice and Bob maintain the list of alive rectangles R at any point. At Alice's turn, she tries to find a rectangle which intersects x in row, and intersects less than half the rectangles in R. If she finds such a rectangle, she sends the name of the rectangle to Bob ($\chi(F)$ bits); otherwise, she sends unsuccessful to Bob. Now Bob looks at the modified R (modified if Alice was successful), and finds a rectangle which contains y and intersects less than half the rectangles in R column-wise.

This keeps going on for $2\chi(F)$ rounds (half the rounds Alice transmits, half the rounds Bob transmits).

Exercise 11. Show that the communication cost is $O(\chi(F))^2$.

We need to show that R will contain only the correct rectangle at the end of the protocol. We claim that in every two rounds (one for Alice and one for Bob), both cannot be unsuccesful. If they were, then the correct rectangle (containing (x, y)) in R intersects both row-wise and column-wise more than half the rectangles in R. This contradicts the above claim. □

We have seen the importance of monochromatic rectangles in communication complexity. Most of the lower bounding techniques rely on this disjoint partition in terms of monochromatic rectangles.

2 Lower bounding techniques

How can this fact about monochromatic rectangles and communication protocols allow us to lower bound the communication complexity of an explicit function?

Let us start with an example. One of the simple function is equality, $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. $E(x,y)$ is one iff $x = y$. Clearly n is an upper bound on the communication complexity, can you prove the lower bound of n too?

We will show that any partition of monochromatic rectangles of M_E will require at least 2^n rectangles, proving the lower bound. Notice the 2^n diagonal entries of M_E, these are the only non-zero entries in the matrix.

Exercise 12. Can two diagonal entries be part of the same monochromatic rectangle?

It is easy to see that if the diagonal entries are part of a single rectangle, say (x,x) and (y,y), then (x,y) is also a part of the same rectangle. Though the value of the function is different on (x,y) and (x,x), so the rectangle could not have been monochromatic.

This means each (x,x) goes in a separate monochromatic rectangle, and hence any partition will have at least 2^n monochromatic rectangles. So any communication protocol needs at least n bits of communication. This proves that the communication complexity of equality is $\Theta(n)$.

The idea used in the previous lower bound can be generalized, and is known as *fooling set.* A set of inputs $S \subseteq \{0,1\}^n \times \{0,1\}^m$ is called a fooling set iff any two elements of S cannot be in the same monochromatic rectangle (this is with respect to a function F).

It is easy to see that,

$$C(F) \geq \chi(F) \geq \log |S|.$$

We will take another example to make this lower bounding technique clear. The disjointness function is a standard example in the world of communication complexity. Alice and Bob are both given a subset of $[n]$, they need to answer 1 if their sets are disjoint. Notice that the input can be viewed as a string of length n, the indicator of the subset.

Exercise 13. How much communication do you think they need for this problem?

We will again show that they need to transmit $\Theta(n)$ bits in the worst case. This is done by making a fooling set of size 2^n,

$$S = \{(X, \bar{X}) : \ X \subseteq [n]\}.$$

Again, we see that no two elements can be part of the same monochromatic rectangle. Suppose $(X, \bar{X})$ and $(Y, \bar(Y))$ are in same rectangle, that means $(X, \bar{Y})$ and $(Y, \bar{X})$ are also in the same rectangle and has value 1.

Exercise 14. Show that for any $X, Y \subseteq [n]$, either $(X, \bar{Y})$ or $(Y, \bar{X})$ intersect.

The above exercise gives us a contradiction.

2.1 rank as a lower bounding technique

One of the most important measure in communication complexity is the rank of the communication matrix M_F. It seems to be closely related to $C(F)$ and $\chi(F)$. We first see that the logarithm of rank is a lower bound on $\chi(F)$ and hence $C(F)$.

To prove the above assertion, we just need to show that rank is of M_F is a lower bound on the least number of monochromatic rectangles required to partition M_F. Remember that rank is the minimum r such that

$$M_F = \sum_{i=1}^{r} x_i y_i^T.$$

Here x_i, y_i's are vectors so that the dimensions match.

From the above characterization, it is enough to find a similar decomposition of M_F in terms of monochromatic rectangles.

Exercise 15. Show that each monochromatic rectangle can be written as a one dimensional matrix. What are the vectors?

For any partition $R_1, R_2, \cdots, R_s$ of M_F in terms of monochromatic rectangles with output 1, let $R_i = x_i y_i^T$ (where x_i is the indicator of the rows of R_i and y_i the columns). Then,

$$M_F = \sum_{i=1}^{s} x_i y_i^T.$$

This shows that the rank of M_F (rank(M_F)) is less than s, and hence $\log \operatorname{rank}(M_F) \leq \chi(F) \leq C(F)$.

Revisiting Equality function, notice that M_F is the identity matrix with 2^n columns and ranks. So the communication complexity of Equality is more than $\log(2^n) = n$.

Let us consider another example, Greater-than function. It is defined on $[n] \times [n]$ and is 1 iff $x \geq y$. We notice that the matrix for this function is triangular and hence has full rank. Again showing that the communication complexity is $\log n$.

We take one more example [3], called the inner-product function, $IP : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. The function is defined by $IP(x, y) = \sum_i x_i y_i \mod 2$. The communication matrix for this function is neither diagonal nor triangular.

Denote the IP matrix with n variables to be IP_n. We first replace every 0 entry in the IP matrix with 1 and every 1 entry with -1. In other words, the new matrix is $J - 2IP_n := P_n$.

Exercise 16. Prove that P_n and IP_n ranks can differ by at most 1.

The structure of P_n is pretty nice.

$$P_n = \begin{pmatrix} P_{n-1} & P_{n-1} \\ P_{n-1} & -P_{n-1} \end{pmatrix}$$

This means that $P_n = L \otimes P_{n-1}$ for a matrix L with rank 2 (what is L?). From the assignment question, $\operatorname{rank}(P_n) = 2\operatorname{rank}(P_{n-1})$. By induction, IP_n matrix has full rank and the communication complexity of IP function is n.

log rank conjecture: We noticed that log-rank is a lower bound on communication complexity, is it tight?

Exercise 17. Show that $C(F) = \Theta(\log \operatorname{rank}(M_f))$ is NOT true.

Computer scientists have conjectured (from a long time) that still log rank is tight in a certain sense, $C(F) = \Theta(\log^c \operatorname{rank}(M_f))$ for some constant c. This is called the *log rank conjecture* and is one of the main open questions in complexity theory.

What is the best possible separation? We will show that the communication complexity and log rank can be asymptotically different using the separation between sensitivity and degree [2]. Their strategy was to show the existence of a function f (Kushilevitz function) such that $s(f) = \Theta(n)$ (full sensitivity obtained at all 0 input), $deg(f) = \Omega(n^{0.63\cdots})$ and the number of monomials are $2^{O(n^{0.63\cdots})}$. The following argument creates a gap between communication complexity and log rank using this f (the argument taken from [1]).

The separation is obtained by *lifting* the Kushilevitz function (say f) to the communication world,

$$F(x, y) := f(x_1 \cdot y_1, x_2 \cdot y_2, \cdots, x_n \cdot y_n).$$

We relate F and f in two ways.

- Since f is fully sensitive at all 0 input, F has communication complexity $\Omega(n)$. This can be shown by noticing that F agrees with the partial function *Unique disjointness* (the subsets are disjoint or intersect at exactly one place), whose communication complexity was already shown to be $\Omega(n)$.

– Another observation is that the polynomial representation of f gives a rank decomposition of M_F. Suppose $f(x) = \sum_S c_S \chi_S(x)$, then

$$M_F(x, y) = F(x, y) = \sum_S c_s \chi_S(x)\chi_S(y) = \sum_S c_S M_S.$$

Observe that M_S are rank one matrices (why?). So, the number of monomials is an upper bound on rank.

From the properties of f it is clear that the $C(F) = \Omega(n)$ but $\log \mathrm{rank}(F) = O(n^{0.63\cdots})$, showing a super-linear separation between them.

3 Extra reading: Randomized communication complexity

Similar to the randomized model of query complexity, we can ask what is the randomized communication complexity of a communication function $F : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$.

Exercise 18. Try to define this randomized model formally.

The idea would be that Alice and Bob can use their input, communication and random bits to decide on their answer. This will only be helpful if we relax the output condition, for every input (x, y), they should succeed with *high probability*. Again, using repetition, we can increase the success probability. So, for the rest of this note we will assume that they have to answer correctly with probability greater than 2/3 for *every input pair*.

In other words, a randomized protocol P is correct for F if for all inputs the probability of success if more than 2/3. The randomized communication complexity is defined as,

$$C_R(F) = \min_{P \text{ correct on } F} \max_{(x,y)} \textit{bits transmitted by } P \textit{ on } (x, y).$$

Note 2. This is an exact analogue of how we defined the randomized query complexity. We are replacing query algorithms with communication protocol.

The first question ought to be, does randomization help? Are there functions for which randomized communication complexity is much better (asymptotically) than the deterministic query complexity. We take the example of Equality function EQ. We know that $C(EQ)$ is full, let us see a simple protocol to show that $C_R(EQ) = O(1)$.

Exercise 19. Can you think of a protocol?

What should Alice and Bob do? How can they use randomness to check equality between x and y by transmitting only constant number of bits?

First intuitive strategy could be: pick a random small subset of $[n]$ of constant size c and check whether x and y are equal on those co-ordinates. Notice that this will take $c \log n$ bits of communication, and not constant (why). Even if we are okay with this much communication, the discussed strategy will fail.

Exercise 20. Show that if x, y are very close to each other, this strategy will not succeed with high enough probability.

The main idea is, for a random string z, if $x = y$ then $x^T z = y^T z \mod 2$ and if $x \neq y$ then $x^T z = y^T z \mod 2$ with probability 1/2.

The formal protocol is, Alice and Bob agree on a randomly uniform string z of length n using randomness. After receiving the strings, Alice sends the inner product modulo 2 between x, z,

$$x^T z \mod 2 = \sum_{i=1}^{n} x_i z_i \mod 2$$

to Bob. Bob also calculates $y^T z \mod 2$, if both values agree then they output 1.

If $x = y$ then $x^T z = y^T z \mod 2$ and they will be released.

Exercise 21. Show that if $x \neq y$ then success probability of Alice and Bob is $1/2$.

To increase the success probability, they can use repetition. They choose multiple random strings $z_1, z_2, \cdots, z_t$ randomly (uniform and independent) and Alice sends the corresponding inner products $\{x^T z_i \mod 2\}$ to Bob. Bob matches the answer and says x, y are equal iff all inner products match.

Exercise 22. What is the probability that they succeed now?

You can show that if they share t random strings, their failure probability is $1/2^t$. It is clear that randomness was crucial in this protocol.

Though, there is a small technicality here. Alice and Bob can use randomness, but how did they agree on *same* random string?

Actually, there are two possible models of randomized communication complexity. In one (where the above mentioned protocol was possible), they have access to same source of randomness. This is called the *shared randomness* model. In the second one, they have different sources of randomness not available to other party, called *private randomness* model.

In general, the communication complexity of a function could be different in these two models. We showed that the randomized communication complexity of EQ is constant in the shared randomness model. What about the private randomness model?

Fortunately, there is a general theorem called *Newman's theorem*, it gives us a private randomness protocol using a shared randomness protocol with only an additional term of $\log n$ in the communication cost.

Theorem 2 (Newman's theorem). *Given a shared randomness protocol with cost C for $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, it can be converted into a private randomness protocol with cost $C + O(\log n)$ and a small increase in error.*

Note 3. The small increase in error can be handled by repetition. In other words, the communication complexity of shared randomness and private randomness only differ by at most a $\log n$ term.

Proof. We will use $U(S)$ to denote the uniform distribution on a set S.

If the shared randomness protocol uses only small amount of shared randomness ($\log n$), then we are already done.

Exercise 23. What will be the private randomness protocol?

Now, our task is to convert the shared randomness protocol with lots of shared random bits into a shared randomness protocol with around $\log n/\delta$ bits of shared randomness and a δ increase in error.

Since the original shared randomness protocol works, $\Pr_{r\sim U(S)}[r \textit{ makes an error on } (x,y)] \leq 1/3$. Notice that without loss of generality, we can assume that r is distributed uniformly in some $S \subseteq \{0,1\}^l$ (why?). Fix an (x,y), let $Er(x,y,r)$ be the indicator random variable for r giving error on (x,y), we know $\mathrm{E}_{r\sim U(S)}[Er(x,y,r)] \leq 1/3$.

We will *simulate* the shared randomness by sampling uniformly from a small set $R = \{r_1, r_2, \cdots, r_t\}$. Next we will show that if we pick R uniformly at ranodm, there is a non-zero probability that the behaviour of R is very close to the behaviour of the original distribution.

If we pick t random r_i's uniformly and independently (let $r = (r_1, r_2, \cdots, r_t)$ be distributed uniformly on S^t),

$$(\textit{Error on } r \sim U(S^t)) = 1/t \sum_i Er(x,y,r_i).$$

The main idea is, using Chernoff inequality, *Error on* $r \sim U(S^t)$ is very close to $1/3$ with high probability. To be precise,

$$\Pr_{r\sim U(S^t)}[(\textit{Error on } r) \geq 1/3 + \delta] \leq 3e^{-\delta^2 t}.$$

If we choose $t = O(\frac{n}{\delta^2})$, then this error is less than $2^{O(-n)}$. Applying union bound (notice that number of pairs is $2^{O(-n)}$),

$$\Pr_{r \sim U(S^t)}[((Error\ on\ r) \geq 1/3 + \delta)\ \ for\ any\ (x,y)] \leq 1.$$

This shows that there exist particular $r_1, r_2, \cdots, r_t$ such that the for all (x,y) the error is less than $(1/3)+\delta$. These r_i's can be ascertained before, without the knowledge of x, y (by computationally unbounded Alice and Bob).

The protocol with small amount of shared randomness is, Alice and Bob pick an index i between 1 and t, and run the original protocol on r_i.

Exercise 24. How much shared randomess do we need now?

Since Alice and Bob just have to agree on a shared index, they can do it using $\log t$ amount of shared randomness. In case of private randomness, Alice can send these $\log t = O(\log n)$ bits of randomness to Bob and conduct the original protocol with cost C (taking $C + O(\log n)$ bits in total). By our analysis, the protocol will succeed with error $(1/3) + \delta$ on every (x, y). ☐

3.1 Extra reading: Lifting (relation with query complexity)

To create separations between different measures in query complexity, many a times we used the composition trick. That means, we first found a constant separation for a function f with constant arity. If the measures *behaved well under composition*, the composed function f gave the asymptotic separation.

To give an example, Kushilevitz function [2] has sensitivity 3 and degree 2. Since $\mathrm{s}(f \circ g) \geq \mathrm{s}(f) \cdot \mathrm{s}(g)$ and $\deg(f \circ g) = deg(f)deg(g)$, the function f^d gives an asymptotic separation between sensitivity and degree.

We would like to have a similar separation trick in communication complexity. Unfortunately, directly composing two communication function F and G does not seem reasonable (you will need four parties). Though, there is a reasonable way to define composition between a query complexity function f and a communication complexity function G, $F = f \circ G$.

$$F(x,y) = f(G(x_1,y_1), G(x_2,y_2), \cdots, g(x_n,y_n)),$$

where $x = (x_1, x_2, \cdots, x_n)$ and similarly y.

To start with, you can think of G as AND or XOR function. The hope is that the query complexity measures on f show results about communication complexity measures of F. This is called *lifting*. The function G is called a *Gadget*.

Do you remember the separation between log-rank and communication complexity. We lifted the separation between sensitivity and degree to the separation between communication complexity and log-rank. It has been an active area of research in recent years. To show that the query complexity of f can be lifted to communication complexity of F using a constant bit gadget is a big open question.

There is a very interesting way to see a communication complexity problem F as a query complexity problem f. We can think of a string which captures all possible functions on x as well as y as the input to this query complexity problem f (the queries parallel the bits transmitted). The output would be $F(x, y)$. It can be formally shown that the query complexity of f is same as the communication complexity of the original function F. For details, see these excellent course notes by Shalev Ben-David, `https://cs.uwaterloo.ca/~s4bendav/CS860/CS860S20week8.pdf`.

4 Assignment

Exercise 25. Show: we can assume that Alice and Bob take turns in the communication protocol and transfer only 1 bit at a time.

Exercise 26. Construct a partition of monochromatic rectangles of a 3×3 matrix which cannot be obtained by a communication protocol.

Exercise 27. Let subset be a function from $\{0,1\}^n \times \{0,1\}^n$ such that it is only 1 iff the set associated with y is a subset of the set associated with x. Show by rank argument that this function has full communication complexity. This argument can be extended to any partial order.

Exercise 28. Suppose X, Y are two matrices, show that $\text{rank}(X \otimes Y) = \text{rank}(X)\text{rank}(Y)$.

Exercise 29. Suppose Alice and Bob succeed with probability 2/3 on every input (two sided error). How many do they need to repeat to get success probability more than 3/4?

References

1. Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers.* Springer-Verlag, 2012.
2. Noam Nisan and Avi Wigderson. On rank vs. communication complexity. volume 15, pages 831 – 836, 12 1994.
3. A. Rao and A. Yehudayoff. *Communication Complexity and applications.* Cambridge University Press, 2020.