COMS-E6998: Information Theory in Computer Science **Fall 2017**

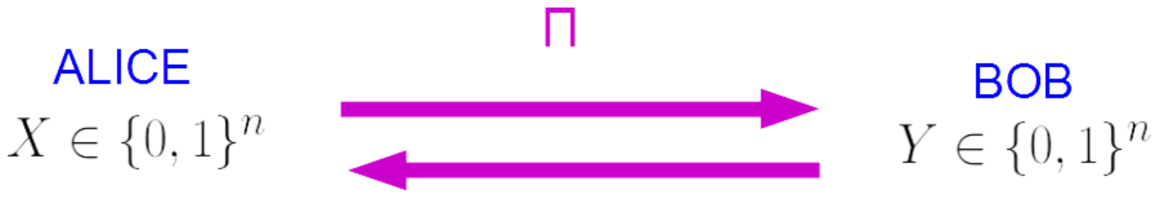
Lecture 5: Deterministic Communication Complexity

Lecturer: Omri Weinstein *Scribes: Natania Wolansky, Andy Xu*

5.1 Motivation

Before, we dealt with just one-way communication – Alice is trying to communicate something to Bob. the goal was to communicate message $\mathcal{X}$ in a lossless way. The question was always, "how much does Alice need to communicate?"

We now consider a reciprocal communication structure, described by a protocol Π, shown below:



Now we would like to compute some function (loosely defined) of the two messages: $f(X,Y) : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. Now, the question to ask is not how to transmit message X, but WHAT to transmit so that either Bob or Alice (or both) can compute $f\{X,Y\}$ with as little information as possible. For now, we assume that our protocol Π needs to be able to work for ANY given $\{X,Y\}$ pair.

Of course, in any situation like this, the naive protocol is simply for Alice to send her entire message to Bob, and to have him compute the function – so we can upper bound any protocol by n bits of communication.

But in many situations, we will need fewer bits than that. Consider three examples below.

5.1.1 Example 1: XOR

Consider the example where our function is $f\{X,Y\} := \oplus_{i=1}^{n}(x_i \oplus y_i)$ – that is, we xor each bit in X with each bit in Y and then xor all these bits together.

This surprisingly requires only 1 bit of communication – Alice can simply compute $\oplus_{i=1}^{n}(x_i)$ and send this to Bob, where he can then do the same, and simply compare these two numbers.

5.1.2 Example 2: Equality

Consider the example where $f(X,Y)$ is the equality function: 1 if $X = Y$, 0 otherwise. This example needs $n+1$ bits, since we need to check if every single bit matches. (This is for the deterministic case, when we cannot be wrong for ANY combination of X and Y. In reality/practice, we would use hash functions, where there is a small probability of collision, or error in computing the equality function. In fact, we can use multiple hashes, and it is proven by some wise mathematicians that with $O(log(\frac{1}{\epsilon}))$ random hashes, we will be correct (non-colliding) with a probability $\geq 1 - \epsilon$).

5.1.3 Example 3: Greater Than

Consider the example where $f(X, Y) = 1$ if $X \geq Y$, 0 otherwise – where X can be considered the binary encoding of a single number. In this case, the deterministic communication complexity is n – since it's possible that we'd have to compare every bit to see the first one that differs (perhaps the numbers are only 1 away from each other). But again, in the randomized case, we'd need only $O(log^2(n))$ bits, because we could do a binary search for the first differing bit.

5.2 Formal Definition of Protocol

Now let's define formally what we mean by a process for Alice and Bob to send information to each other.

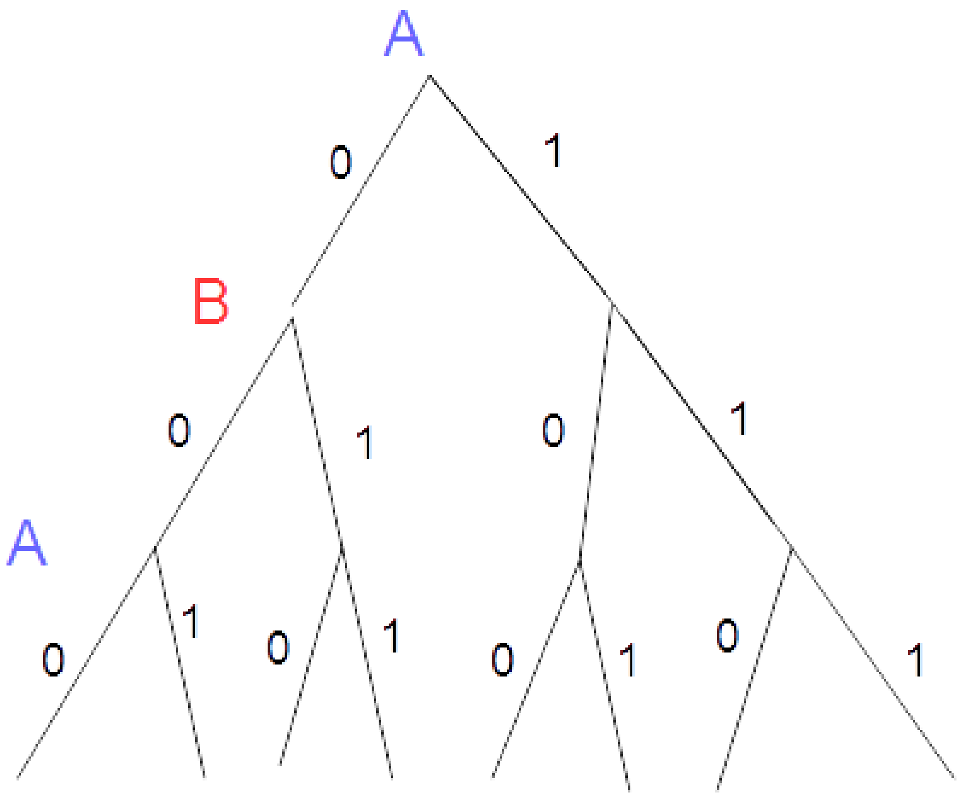
Definition 5.2.1. *We define a Communication Protocol* Π *(basically the formalization of a conversation) as a pair of mappings:*

$$\begin{aligned} f_A :& (\Pi_{<i}, X) \to \{0,1\}, \textit{even } i \\ f_B :& (\Pi_{<i}, Y) \to \{0,1\}, \textit{odd } i \end{aligned}$$

By convention, we assume Alice always speaks first.

What do we mean by $\Pi_{<i}$? Essentially, Alice has access to all the communication in the past (what she told Bob, and what Bob told her) plus her string, X. Bob has access to the previous information exchanged in the conversation, plus Y, but NOT X.

We can visualize a deterministic protocol Π as a binary tree τ_Π:



At each node in the tree, the designated party has access to everything that happened above. If we consider the total information sent between the two as the message m, then a v at the ith level of the tree is associated with some partial transcript $v = m_{\leq i}$. He follows the protocol to know whether to communicate a 0 or 1 based on this v and his own knowledge of either X or Y.

5.3 Combinatorial Rectangles

5.3.1 Communication Matrix

Now let's look at the communication matrix $M_f : 2^n \times 2^n$ – the matrix of all possible combinations of X and Y, and the output of the function $f(X,Y)$ for these specific inputs. The rows are all possible Xs and the columns are all possible Ys. The (i,j)th entry of the matrix will be $f(i,j)$ (what you want Alice and Bob to be able to compute given Alice has i and Bob has j. For example, the Equality matrix, M_{eq}, for $n = 2$, looks like this:

$$\mathbf{P} = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 1 & 0 \\ 11 & 0 & 0 & 0 & 1 \end{array}$$

As we can see, it's the identity matrix. Let's look at what happens on the communication matrix as we go through the protocol Π. In the first step, Alice sends one bit to Bob, either a 0 or 1. Perhaps she sends the first bit of her message. She therefore cuts the communication matrix in half, rowwise. For instance, if she sends a 0 then Bob knows her message is either 00 or 01 – so it lies in the top half of the communication matrix.

Now, Bob sends a message back. Perhaps he also sends the first bit of his string. Let's say Bob has 10, so he sends a 1 – then Alice knows that the answer lies in the right half of the top half of the matrix – So Bob has now further cut the submatrix in half, columnwise. With each new piece of communication, the submatrix Alice and Bob currently live in is split into two, either rowwise or columnwise depending on who is speaking. They can both stop when the submatrix is monochromatic (same value) – meaning there is only one possible value for $f(X, Y)$. So in our case, the top right half quarter of the matrix is monochromatic in 0, so Alice and Bob can stop communicating and output 0. Note, they do NOT need to know EXACTLY which string the other has – they are only trying to calculate the function.

Definition 5.3.1. *A subset $R \subset \{0,1\}^n \times \{0,1\}^n$ is a* combinatorial rectangle *IFF:*

$$\begin{aligned} R &= A \times B \\ &= \{(x, y) | x \in A, y \in B\} \end{aligned}$$

That is, we take some subset A of the rows and some subset B of the columns, and we have a combinatorial rectangle iff every combination of these subsets is in our subset R.

An alternate definition is: R is a combinatorial rectangle IFF:

$$(x, y), (x', y') \in R \implies (x, y'), (x', y) \in R$$

meaning, if x shows up with some y in the rectangle, then it must show up with EVERY y in the rectangle, and vice versa with y.

5.3.2 Protocol Trees - Combinatorial Rectangles

Our claim is that each node of the tree described above can be equated with a specific combinatorial rectangle in the communication matrix.

Claim 5.3.2. *$\forall v_{\leq i}$ (some node in level i of the protocol tree) let $S_{v_{\leq i}}$ be the set of possible values of X and Y at $v_{\leq i}$ – formally:*

$$S_{v_{\leq i}} := \{(x, y) | \Pi(x, y)_{\leq i} = v_{\leq i}\}$$

The claim is that $S_{v_{\leq i}}$ is a rectangle.

Proof. We will do a proof by induction on i. For the root of the tree, the claim obviously holds true. That takes care of the base case. Now, we assume that the claim holds for level i. Suppose i is even, WLOG, and suppose $S_{v_{\leq i}}$ is a rectangle. Then by the definition of rectangles, $S_{v_{\leq i}} = A_v \times B_v$. Say, WLOG, that Alice at this point chooses to communicate 1 to Bob. Then at level $i + 1$, we have a new set $S_{v_{\leq i+1}}$:

$$S_{v_{\leq i+1}} = \big(A_i \cap \{x | \Pi_{v_{\leq i}}(x) = 1\}\big) \times B_v.$$

which is also a rectangle. Thus we've proven this connection. □

This also holds for the leaves of the tree. So finally, we can say that all S_l (leaves of tau_Π) are f-monochromatic (monochromatic under the function f that we are trying to compute) combinatorial rectangles. So The number of communication steps, and thus the deterministic communication complexity of the function f, is the depth of the tree – the max number of steps to reach the leaves of the tree. formally,

Definition 5.3.3. *For all valid protocols – that is, for all* $\Pi : \Pi(x, y) = f(x, y)$ *for every* (x, y) *in the domain of the function* f*, the deterministic communication complexity is defined as:*

$$D(f) = \min_{\Pi}(||\Pi||)$$

And by this connection, we can go the other way: any deterministic communication complexity protocol Π partitions M_f into (at most $2^{||\Pi||}$ f-monochromatic rectangles.

5.4 Lower Bound on Deterministic Communication Complexity

Suppose we are given a monochromatic partitioning – does this correspond to a protocol? No! Because every intermediate step also needs to be a rectangle. We can't assume that a monochromatic partitioning is also a protocol – that's why we can't naively just find the minimum coloring of the matrix and call it a day. Let the partition number of M_f, $P(M_f)$ be this minimum number of partition rectangles. Then we know that:

$$log(P(M_f)) \leq D(f)$$

but we can't turn that inequality into an equality.

We CAN set a (better) lower bound on CC for a given f by proving what the minimum number of monochromatic rectangles can be in M_f, given that these rectangles come from a protocol. Here we give a couple of techniques to do this.

5.4.1 Fooling Sets

Definition 5.4.1. *A fooling set (FS) of a communication matrix* M_f *is a set* $R = \{(x_i, y_i)\}_{i=1}^{k}$ *(k entries of* M_f*) where:*

$$f(x_i, y_i) = 1 \forall i \in R \tag{5.1}$$

$$f(x_i, y_{/i}) or f(x_{/i}, y_i) = 0 \tag{5.2}$$

That is, in this set, the combination of i and i makes 1, but only that – the combination with any other letter in R makes 0. So no two fooling set elements can live in the same monochromatic rectangle.

So, if we can find the largest fooling set FS for some given f, then we know that

$$D(f) \geq log(|FS(M_f)|)$$

because we'd need at least $|FS(M_f)|$ rectangles (so $log(|FS(M_f)|)$) levels to have this many leaves in the tree), one for each element in the fooling set.

5.4.1.1 Equality

Claim 5.4.2. *Claim: For the equality function,* $D(EQ_n) \geq n$

Proof. Let's use a fooling set to show the minimum deterministic CC of the equality function. Since we know from earlier that the communication matrix is the identity matrix, we can take as a fooling set all the elements that lie in the diagonal, or all (x, x) elements. Clearly, this is a fooling set – if we take any (x, x') in the set, the two strings are not equal so they give 0. The size of this fooling set is 2^n (because there are 2^n possible strings of length n) so we have placed a lower bound $D(f) \geq n$. □

5.4.1.2 Set Disjointedness

This is the "queen of communication problems." It's really hard to do and we wish it weren't. Consider the situation in which there are n elements in some shopping list, and Alice and Bob both buy some subset of these elements $X, Y \subset [n]$. The sets X, Y are disjoint if they are non-overlapping – if Alice and Bob didn't buy any of the same stuff. So, $DJS_n(X, Y) = 1 \implies X \cap Y = 0$ (it's 0 if they bought the same thing).

We can create a fooling set FS as all possible subsets of $[n]$ and their compliments – $FS = \{(A, \overline{A})\}_{A \in N}$. We know that these yield 1 by definition – nothing can overlap it's compliment. But consider comparing set A to some other set A'. Clearly, either A overlaps with A' or A' overlaps with $\overline{A}$. So either $f(A, A') = 0$ or $f(A', \overline{A}) = 0$. So each set needs its own monochromatic rectangle. thus, $|FS| = 2^n$ (the number of possible subsets of set of size n) and $D(DJS_n) \geq n$. We will return to this later to try to get a better lower bound.

5.5 Rank

5.5.1 Basics

The next method for finding lower bounds on DCC is using M_f directly, through its rank. But to use rank, we have to delve into some linear algebra. Don't be scared, Computer Scientists, you can do it!

Definition 5.5.1. *We will use three definitions for the rank under $\mathbb{F}$ of a matrix A:*

1. $rank_{\mathbb{F}}(A) :=$ *max number of linearly independent rows/columns*
2. $rank_{\mathbb{F}}(A) :=$ *min number of rows that span the rest of the rows in the matrix*
3. $rank_{\mathbb{F}}(A) := r$ *s.t.* $A = \sum_{i=1}^{r} A_i, rank_{\mathbb{F}}(A_i) = 1$

The first definition is self-explanatory. The second definition, also, self explanatory. The third is basically saying that if you can split the matrix A into the sum of matrices with rank 1, then the rank of A is the number of these matrices.

We will also use two properties of rank and one fact, all of which we will not prove because this is not a mathematics class. Just trust me about this – they are all correct.

a Subadditivity: $rank(A + B) \leq rank(A) + rank(B)$

b Subproductability?: $rank(A \times B) \leq min\{rank(A), rank(B)\}$

c The rank of Boolean matrices is maximized over $\mathbb{R}$ (as opposed to over, for example, GF2)

What does this ”under $\mathbb{F}$” thing mean? Basically, if we are working in mod 2, for instance, the rank of a matrix may change – although 0 is not a multiple of 1 in $\mathbb{R}$, it is in GF2 – it's 2×1. That's why it makes intuitive sense that the rank would be maximized over the real numbers (when all the entries in the matrix are 0 or 1 – there's no way that adding linear combinations of rows would ”spill over” and equal 0 or 1, creating a linear dependence where in the real numbers there would be none.

Lemma 5.5.2. *$\forall f, \forall \mathbb{F}, log(rank_{\mathbb{F}}(M_f)) \leq D(f) \leq rank_{\mathbb{F}}(M_f)$*

Well, you saw this coming – why would we bring up rank in this section if it had nothing to do with finding minimum DCC? The bottom limit kind of makes sense. But the top limit is somewhat surprising! Let's prove the lower limit first:

Proof. First we prove the lower Limit:
We prove this using the third definition of rank – the one with the sum of matrices. Suppose that $\exists$ some protocol Π where $||\Pi|| = C$, where we suppose this is the minimal communication complexity Π, and thus the minimum C for some function f. This induces a partition of M_f into $\leq 2^C$ monochromatic rectangles. Each of these rectangles has rank 1, because they are all the same number (so we can, for example, take the first row and show that this row spans the rest of the rows). We know that we can get our original M_f back by adding together all these rectangles (to see this, expand each rectangle to be the size of M_f, with all entries 0 that are not in the original rectangle. Add them. Voila!) So:

$$M_f = \sum_{i=1}^{2^C} M_i, rank_{\mathbb{R}}(M_i) = 1 \tag{5.3}$$

$$rank(M_f) = rank\left(\sum_{i=1}^{2^C} M_i\right) \tag{5.4}$$

$$\leq \sum_{i=1}^{2^C} rank(M_i) \tag{5.5}$$

$$= 2^C \tag{5.6}$$

where inequality (5) comes from the subadditive property of rank. Rearranging the following equation gives the inequality we are looking for:

$$rank(M_f) \leq 2^C$$
$$log(rank(M_f)) \leq C$$

So we can say that the minimum communication complexity, C, is at least as big as the log of the rank of the communication matrix M_f. Done! ☐

5.5.2 Example – Inner Product

Let the function f be inner product: $P_n(X, Y) := \langle x, y\rangle \mod 2 = \sum_{i=1}^{n} x_i y_i$ (where the product in the second definition is done in GF2. So we obviously need to do our math in $\mathbb{F} = GF2$, not $\mathbb{R}$.

The claim is that $rank_{GF2}(M_P) \leq n$. Let's show this. Naively, we might say, "Oh! Let's decompose M_f into matrices $M_i(x,y)$ where the matrix entry is 1 only when x and y are both 1 in the ith bit. We can see that this matrix is the inner product matrix projected on the ith coordinate, and to achieve the total inner product, we just need to add these. That is, $M_P(X,Y) = \langle x,y \rangle_2 = \sum_{i=1}^{n} x_i y_i$. Great! The rank, in any of these projected inner product matrices, is 1, so using the same process as above, we can say that the total matrix is is a sum of n of these matrices (one for each bit in the input X or Y. But then we've shown that $D(f) \geq log(n)$. Can we do better – get a tighter lower bound?

Yes, we can! let's look at the squared matrix, M_P^2 The rank of M_P is at least as big as the rank of M_P^2 because of subproductness thingy that we defined as the second characteristic of rank. Let's look at a simple matrix to see what this squared matrix would look like:

$$\mathbf{M_P} = \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

In the squared matrix, we do simple matrix multiplication. We can represent this algebraically as:

$$M_P^2(X,Y) = \sum_{x \in (0,1)^n} \langle x,z \rangle \cdot \langle z,y \rangle$$

In words, the entry for (x,y) is the number of strings z (in the real numbers, now) where both $\langle x,z \rangle$ and $\langle z,y \rangle$ is 1. On the diagonal, where $x = y$, this is exactly half the strings (the number of strings where an odd number of 1 bits match up). So this accounts for 2^{n-1} strings. Where $x \neq y$, not on a diagonal, its a fourth of the strings, so 2^{n-2}.

So our matrix M_P^2 is a matrix with all 2^{n-1} on the diagonals and 2^{n-2} off the diagonals. We can see this as the linear combination of the 1s matrix and the identity matrix: specifically,2^{n-2} 1 matrices and 2^n identity matrices. Each of these has rank 1 in $\mathbb{R}$, so total we have $rank(M_f) \geq rank(M_f^2) = 2^{n-2} + 2^n$. Thus,

$$D(f) \geq n - 1$$

which is a tighter bound than we had before.

5.5.3 Upper Bound

What about the other side of the inequality – the one claiming $D(f) \leq rank_{\mathbb{R}}(M_f)$? This is difficult to prove, so it takes a little leap of faith for us. Intuitively, we can see using the span definition of rank that if M_f is spanned by r, then the entire matrix is a linear combination of these r spanning rows. So instead of telling Bob the whole matrix, she can just communicate the coefficients of these linear combinations, with r coefficients describing X. For a Boolean matrix the coefficients should only be 0 or 1 – so the coefficients shouldn't be too large – each can be described with one bit. So Alice could potentially just communicate r bits to Bob to describe exactly what the rows of M_f look like.

Theorem 5.5.3. ***Log-Rank Conjecture***

$$\forall f, \exists c \ st. \ D(f) = O((\log rk(M_f))^c)$$

Notice that the Log-Rank Conjecture suggests that the constant c is universal for all function f.

Not until 2013 did there are advancements on proving the Log-Rank Conjecture unconditionally.

Theorem 5.5.4 (Lovett13)**.**

$$\mathsf{D}(f) \leq O(\sqrt{rk(f)} \cdot \log rk(f))$$

5.6 Nondeterminism

5.6.1 Covers

We learned that an f-monochromatic partition of the communication matrix might not be a protocol-induced partition, and their relationship is $C^D(f) \leq C^P(f)$. A natural question to ask is, how good are the lower bound techniques (the Fooling Set technique and the Rank lower bound technique) for communication complexity that use lower bounds on the number of monochromatic rectangles in partition of the space $X \times Y$, **ignoring the additional restriction that these partitions have to be induced by a protocol.** Therefore, we shall consider relaxing the need the partition the space $X \times Y$ into f-monochromatic rectangles by allowing covering of this space (i.e. by allowing intersections between rectangles).

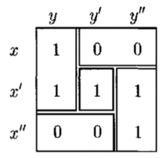


Figure 2.1: A partition that does not correspond to any protocol

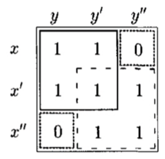


Figure 2.2: An overlapping cover

Figure 5.1: Partition number and cover number. Adapted from *Communication Complexity* by Eyal Kushilevitz

5.6.2 Communication Complexity Measures

To better study properties of partitions and covers of communication matrices, we shall review and introduce some basic measures on communication complexity.

1. $C^P(f)$: Protocol-induced partition number. The minimum number of of leaves among all protocol induced trees.
2. $C^D(f)$: Partition number. The minimum size of a partition of the communication matrix into f-monochromatic rectangles, **without the constraint that the partition is derived from a communication protocol**.
3. $C(f)$: Cover size. The smallest number of f-monochromatic rectangles needed to cover the communication matrix.

4. $C^0(f)$: 0-cover number. The smallest number of **rectangles** needed to cover the 0-inputs of f.
5. $C^1(f)$: 1-cover number. The smallest number of **rectangles** needed to cover the 1-inputs of f.
6. $\chi_1(f)$: 1-Partition number. The minimum partition size of 1-inputs of f. This is what Yannakakis called as **"Unambiguous Communication Complexty"**, in which only 1 certificate is allowed for each input (x, y) for f.
7. $\chi_0(f)$: 0-Partition number.

We also provide some fundamentals class notations involved in communication complexity problems.

1. $D(f)$: Deterministic communication complexity. Among all possible communication protocols for Alice and Bob to compute f, the minimum number of bits needed to communicate between them.
2. $NP(f) = \log(C^1(f))$: Nondeterministic Communication Complexity (discussed later in the next subsection).
3. co-$NP(f) = \log(C^0(f))$: the co-problem of nondeterminism.
4. $P_1(f) = \log(C_1^U(f))$: Unambiguous communication complexity.

Lemma 5.6.1. *For any binary function $f : X \times Y \to \{0,1\}$, the below inequality holds*

$$C^0(f) + C^1(f) = C(f) \leq C^D(f) \leq C^P(f) \leq 2^{D(f)}$$

Using two lower bound techniques introduced earlier in the lecture, we can obtain the following lower bounds. For proofs of these lower bounds, please refer to the textbook by Anup Rao.

For any binary function $f : X \times Y \to \{0,1\}$

1. $C^D(f) \geq 2 \cdot rk(f) - 1$
2. $D(f) \leq C^z(f) + 1$ (for $z \in \{0,1\}$)
3. $\log_2 C^P(f) \leq D(f) \leq 3 \log_2 C^P(f)$
4. $N(f) = log_2 C(f) = log_2 C^1(f) + C^0(f)$

5.6.3 Nondeterministic communication complexity

To understand the natural interpretation of $C^1(f)$, now we shall introduce the concept of a proof system.

NCC Proof System: We shall go back to the communication model between Alice and Bob. Assume that there is an all powerful prover, who sees x and y and can talk to Alice and Bob. The prover tries to convince Alice and Bob that $f(x, y) = 1$ by giving them a certificate. If $f(x, y) \neq 1$, then Alice and Bob must be able to detect that the proof is wrong. However, if indeed $f(x, y) = 1$, then the prover must be able to convince them that indeed $f(x, y) = 1$. Likewise, we can also consider the co-problem using the prove system as well; namely the proof system for $C^0(f)$. The idea is similar and should apply from $C^1(f)$ proof system without the loss of generality.

Example of a Proof System: For example, consider non-equivalence function f that takes in two n-bit strings x and y. $x = y \leftrightarrow f(x, y) = 0$, whereas $x \neq y \leftrightarrow f(x, y) = 1$. Alice has x and Bob has y and the try to cooperatively compute $f(x,y)$. A prover who sees both x and y and tries to convince Alice and Bob that $f(x,y)=1$ will simply send the first index i at which string x differs from y to Alice and Bob, as the certificate. They can then exchange i-th bit of x and y to verify the correctness of the proof. If $x \neq y$, then there is (at least) a proof, but if $x = y$, such proof does not exist.

Now given the definition of a proof system, we claim that the most efficient proof system, denoted by $NP(f)$, must have the following complexity.

Theorem 5.6.2.

$$NP(f) = \log(C^1(f))$$

We shall give a brief explanation for the above theorem.

On one hand, consider any cover of the 1-inputs on a communication matrix. Each cover gives a proof system where a valid certificate, provided by the prover, is the "name" of a cover rectangle $S \times T$ in which input (x,y) resides. The number of bits required for the prover to send the certificate to Alice and Bob is $logC^1$. After that, Alice and Bob can convince themselfs by checking whether $x \in S$ and $y \in T$ and exchange their findings.

On the other hand, consider any proof system using at most b bits. Since for every (x,y) that satisfies $f(x,y)=1$, there must exist a corresponding proof, and there are at most 2^b such "cover" rectangles. Because rectangles can overlap, we do not require exactly one single proof for each (x,y). (We will talk about the stricter case later, which is unambiguous communication complexity as defined by Yannakakis as he introduced the CIS problem.) We define the complexity for the above-mentioned system as $NP(f)$.

From the two-fold observations discussed above, we can not only treat nondeterminism as a proof system but also define it as a two-party protocol in which Alice and Bob are allow to take nondeterministic steps. For instance, Alice can guess a certificate and ask Bob to verify the certificate.

5.6.4 Open Problems

5.7 [Yannakakis 88] Clique vs. Independent Set Problem

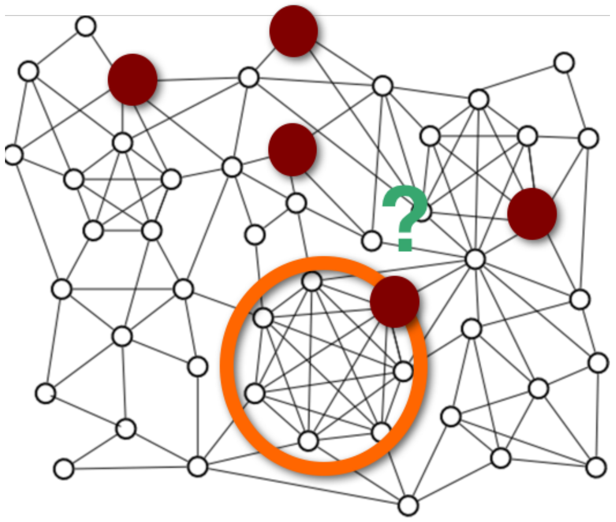
Finally, we look at one application of this Deterministic CC for a cute problem, published by one of our very own, Yannakakis, in 1988.

5.7.1 Problem Statement of CIS_G

- Suppose there is a publicly known graph $G = (V, E)$.
- Alice is given a clique $C \subset V$.
- Bob is given an independent set $I \subset V$
- If Alice and Bob need to determine $C \cap I = \varnothing$

CIS_G considers the communication complexity of solving the Clique vs. Independent-Set Problem.

Lemma 5.7.1. $|C \cap I| \leq 1$. *There can be at most one intersection between Alice's clique C and Bob's independent set I.*



Lemma 5.7.1 can be easily proved by contradiction: Assume $|C \cap I| > 1$. WLOG, consider any distinct $V_1, V_2 \in C \cap I$. Since $V_1, V_2 \in C$, we have $(V_1, V_2) \in E$; on the other hand, $V_1, V_2 \in I$, by the definition of independent set it must be $(V_1, V_2) \notin E$, which leads to a contradiction. Therefore, $|C \cap I| \leq 1$.

5.7.2 CC to Verify Intersection with a prover (NP)

Consider the nondeterministic communication complexity to verify $V' \in C \cap I$.

WLOG, we can assume Alice is given a verification oracle. Alice and Bob shall take turns to communicate to one and another to verify intersection.

- Alice, who has clique C, is given by the oracle s.t. $V' \in C$. Alice then sends V' to Bob in $O(\lg(n))$-bits.
- Bob, who has independent set I, can determine $V' \in I$ immediately upon receiving V' from Alice.

Lemma 5.7.2. *The nondeterministic communication complexity of verifying intersection is given by*

$$NP(CIS_G) = O(\log n)$$

We call that from the previous section, we introduce a stricter class of nondeterministic complexity class - $P_1 \subset NP$. P_1 requires that for each input (x, y) such that $f(x, y) = 1$, there must exists exactly one certificate/proof. As we proved earlier, in CIS, there can be at most 1 vertex intersection. This implies that the 1-input rectangles in the cover **cannot** overlap. Alternatively, we can this simply view P_1 as partition of 1-input into rectangles. We can now rewrite Lemma 5.7.2.

Lemma 5.7.3.

$$P_1(CIS_G) = O(\log n)$$

5.7.3 CC to Verify Non-intersection ($D(CIS_G)$)

Now we shall consider the co-nondeterministic complexity of the above problem - verifying non-intersection. We want to bound the communication complexity of verifying non-intersection, namely deciding whether $C \cap I = \varnothing$. This is the core problem of Clique vs. Independent Set, and we shall denote it as CIS_G.

Yannakakis gave the following upper bound for any G.

Theorem 5.7.4. *co-$NP(CIS_G) \leq O(log^2(n))$*

To prove this bound, we describe a communication protocol as follows.

Define $d(v)$ to be the degree of a node $v \in V$ of G, $\Gamma(v)$ to be set of v's neighbors. Also define $n = |V|$. Initialize $S = V$

- Alice, who has the clique C, picks a vertex $v \in C \cap S$ with a *small* degree, such that $d(v) \leq \frac{n}{2}$. Alice sends this vertex to Bob.
- Bob, who has the independent set I, picks a vertex $u \in I \cap S$ with a *high* degree, such that $d(v) > \frac{n}{2}$. Bob sends this vertex to Alice.

If both players cannot find such node, then $C \cap I = \varnothing$, because a node in G must have its degree either greater or less than $\frac{n}{2}$. The protocol terminates.

If any player finds a node that the other player, who receives such node, finds that this node is also in its own set, then we find the intersection. The protocol terminates.

Otherwise, we can discuss in these two cases:

- Alice finds a satisfying node $v \in C$ with low degree, Bob receives v and finds that $v \notin I$. We can then restrict $S = \Gamma(v)$. Since $d(v) \leq \frac{1}{2}$, we reduce $|S|$ by at least half.
- Bob finds a satisfying node $u \in I$ with high degree, Alice receives u and finds that $u \notin C$. We can then restrict $S = (S - \Gamma(v) - v)$. We again reduce $|S|$ by at least half because $d(v) > \frac{1}{2}$

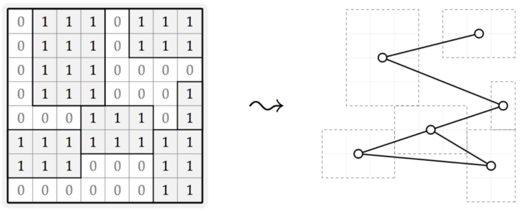
The protocol thus proceeds for $O(\log(n))$ rounds and each round involves $O(\log(n))$ bits of communication (as argued in the previous section). Therefore the communication complexity of the protocol is $O(log^2(n))$ $\square$.

Alternatively, we can likewise prove the deterministic communication complexity of CIS_G problem.

Theorem 5.7.5. $D(CIS_G) \leq O(log^2(n))$

5.7.4 Reduction from the Communication Matrix to GIS Problem

[Yannakakis 88] discusses a reduction technique from any communication problem to a GIS problem.



The general reduction technique is described as follows. Suppose A is any boolean communication matrix that admits a partition of its 1-entries by rectangles $R_1, ..., R_n$. We would like to convert this communication matrix into a CIS graph, which we shall define as CIS_A. The vertices of graph CIS_A correspond to rectangles $R_1, ..., R_n$. Let R_i and R_j be connected by an edge if R_i and R_j intersect the same row (intersection means sharing a same edge). Notice that due to the non-overlapping nature of the partition rectangles, such R_i and R_j can never intersect in columns. Therefore, a CIS_A problem in this case can be formed by a pair (x, y) where x is the index of a row and y is the index of a column. Alice is given the clique described by its vertex set $C_x = \{(R_i, \text{ intersects in row } x\}$, and Bob is given a independent set $I_y = \{R_j, \text{ intersect in column} y\}$.

In the original communication problem A, Alice and Bob need to decide whether $f(x, y) = 1$, whereas in the CIS_A problem, Alice and Bob need to decide whether the intersection of Alice's clique C_x and Bob's independent set I_y is non-empty.

Theorem 5.7.6 (Yannakakis 88). *Any deterministic communication problem f can be embedded, with no extra communication cost, as a CIS problem on a graph $G = CIS_f$ with $P_1(f)$ vertices.*

One conseqence of the reduction scheme is that, we can now upper bound the deterministic communication complexity of any function f, by converting the deterministic communication complexity problem to its equivalent CIS problem. Since $P_1(f) = \log(\chi_1(f)) = \log |V(CIS_G)|$

Theorem 5.7.7.

$$D(f) = (P_1(f))^2$$

5.7.5 Lower bounds on GIS

There have been recent advancements to lower bound the conondeterministic communication complexity of the CIS problem. This is very important because the reduction we introduced earlier have would mean that proving the lower bound for co-$NP(CIS_G)$ is equivalent to prove the lower bound for $D(CIS_G)$.

It's first noteworthy to mention that there are instances of CIS problems that require superlogarithmic communication complexity.

Theorem 5.7.8 (GPW15). *$\exists CIS_G$, s.t. co-$NP(CIS_G) \geq \tilde{\Omega}(\log^2(\chi_1(f)))$*

Also in 2015, Goos proved an $\omega(\log n)$ lower bound on the conondeterministic communication complexity of CIS.

Theorem 5.7.9 (Goo15). *co-$NP(CIS_G) \geq \omega(\log(n))$*