CS 153 Communication Complexity	**Timur Kuzhagaliyev**

Problem Set #2 Solutions

Out: May 22 *Due:* **June 1**

1. (a) We can show that the inequality in the question holds by considering the structure of M_1 and M_2. The breakdown of matrix of M into M_1 and M_2looks like

$$\begin{pmatrix} \times & X_1 \times T & \times \\ S \times Y_1 & S \times T & S \times Y_2 \\ \times & X_2 \times T & \times \end{pmatrix}$$

where the highlighted row is matrix M_1, the highlighted column is the matrix M_2 and X_1, S, X_2 with Y_1, T, Y_2 are partitions of X and Y respectively. Note that due to the structure of given matrices, the standard rank of M_1 is bound above by $|S|$ and the standard rank of M_2 is bound above by $|T|$, and the same bounds apply to triangle-rank.

Note also that $S \times T$ is a zero matrix - this means that regardless of how rows or columns from $S \times T$ are permuted within M_1 or M_2, they will not change the triangle-rank. As such, there is no reason to permute rows indexed by S in M_2 and no reason to permute columns indexed by T in M_1. This creates a convenient "mask" that lets us permute rows and columns of M and build desired square submatrices inside M_1 and M_2 without undoing any progress.

Now we can put the two points from above together to define a procedure that proves the desired inequality. We start off with M. Next we permute rows and columns of M to build the largest square submatrix with ones in diagonals on zeros below the diagonal in both M_1 and M_2 (showing triangle-rank of these matrices). As discussed previously, we can do that without any clashes.

Denote such square submatrices S_1 in M_1 and S_2 in M_2. Finally, we permute columns of resultant M to put S_1 into the top left corner of block $S \times Y_2$, and then permute rows of resultant M to put S_2 into the bottom right corner of block $X_1 \times T$. As a result of this operation, we have transformed M in such a way that it now contains a square matrix that is a tensor product of S_1 and S_2 (with the exception of the top right corner which contains arbitrary values and doesn't affect triangle-rank). Hence we have shown that if matrices M_1 and M_2 have triangle-rank of m and n respectively, we can permute rows and columns of M to build a square submatrix of side length $m + n$ with ones on the diagonal and zeroes below the diagonal, showing that the triangle-rank of matrix M must be at least $m + n$.

(b) We can prove the desired bound using induction on triangle-rank. We begin by outling a protocol that uses $N^0(M_f) + 1$ bits per round of communication. Note that the smallest non-disjoint 0-monochromatic cover of M_f contains $2^{N^0(M_f)}$ rectangles. Note that if we enumerate these rectangles from 1 to $2^{N^0(M_f)}$, it will take exactly $N^0(M_f)$ bits to communicate the index of the rectangle.

Now our protocol proceeds as follows. Alice and Bob maintain sets A and B respectively. A indicates the set of *columns* in the search space (for Alice). B indicates the set of *rows* in the search space (for Bob). Initially, both A and B contain the entire matrix M_f.

In the beginning, Alice holds some row x and Bob holds some column y in M_f (as the inputs). Alice begins iterating over all rectangles $R_A = S \times T$ in the 0-cover we fixed earlier such that:

i. R_A intersects her row x,
ii. Restricting the matrix to columns indexed by S cuts the triangle-rank of the matrix by at least a factor of 2, and
iii. At least one of the columns of R_A overlaps with set A.

All rectangles that do not satisfy above conditions are ignored. For every R_A, Alice transmits the index of R_A to Bob. Bob replies with either 0 or 1 to indicate whether his column y also intersects R_A. If Bob returns 1, then x and y both intersect the same 0-rectangle, and the output $f(x, y)$ must be 0, so we're done. Otherwise, Alice removes columns covered by R_A from her set A and proceeds to the next loop iteration. The amount of bits exchanged per iteration is $N^0(M_f)+1$.

Eventually, Alice and Bob either agree on some rectangle that Alice proposed or Alice runs out of rectangles that satisfy the three conditions we specified. If Alice runs out of 0-rectangle candidates, she signals this to Bob using 1 bit. Assume that the actual output of $f(x, y)$ is 0. In this case, there must exist a 0-monochromatic rectangle $S \times T$ that intersects both row x and column y. Denote the triangle-rank of submatrix $S \times Y$ by ΔM_1 and triangle-rank of submatrix $X \times T$ by ΔM_2. Since Alice cannot find the desired 0-rectangle, it must be the case that $\Delta M_1 > \frac{1}{2}\Delta M_f$. Combining this with the inequality from part (a), we get $\Delta M_2 < \frac{1}{2}\Delta M_f$. This means that Bob can run the same loop using his x and B to find the desired 0-rectangle and conclude that the output of $f(x, y)$ is indeed 0, exchanging the same amount of bits per iteration.

In the case when $f(x, y) = 1$, Bob will still attempt to execute the same loop, but he will eventually run out of candidate 0-rectangles (since Alice will just reject everything). If this happens, Bob communicates this to Alice with one bit and the parties will conclude that the output is 1.

We can now use the protocol outline to prove the desired bound on communication complexity using induction on triangle-rank.

- **Inductive hypothesis**: Assume that the formula

$$D(M_f) \leq (\log_2(\text{triangle-rank}(M_f) + 1) + 1)(N^0(M_f) + 1)$$

  holds for all $M_f = A' \times B'$ such that triangle-rank$(A' \times B') < t$.
- **Base case**: triangle-rank$(M_f) = 0$. In this case, we have an all 0 matrix, so there is a 0-cover of size 1, hence $N^0(M_f) = \log_2(1) = 0$. Using our protocol, Alice transmits $N^0(M_f)$ bits, or, in other words, doesn't transmit anything since the index is trivial. Bob replies with 0 to acknowledge that he's in the same rectangle. In total, we have 1 bit of communication, and $D(M_f) = 1$. Indeed, the formula holds for our base case:

$$\begin{aligned} D(M_f) &\leq (log_2(\text{triangle-rank}(M_f) + 1) + 1)(N^0(M_f) + 1) \\ &= (log_2(0 + 1) + 1)(0 + 1) \\ &= (0 + 1)(1) \\ &= 1 \end{aligned}$$

- **Inductive step**: Need to show that formula holds for $M_f = A \times B$ s.t. triangle-rank$(A \times B) = t$. As per the protocol we outlined above, Alice starts searching for a suitable 0-rectangle from the cover first.

  Case 1: Alice finds some $S \times T$ rectangle R_A and sends its index over to Bob. If Bob replies with 1, parties have found a common 0-rectangle and the protocol can terminate with output 0 after having exchanged only $N^0(M_f) + 1$ bits. Clearly, the formula is satisfied.

On the other hand, if Bob rejects R_A, Alice can remove the columns that touch R_A from her set A. We can show that this action decreases the triangle-rank of her search space by a factor of 2: For Alice to pick R_A, restricting M_f to the rows indexed by S must've cut the triangle-rank but
Case 2: Alice doesn't find a suitable rectangle, so she has to tell Bob that he needs to begin the search. There are two possibilities here: If $f(x, y) = 0$,

2. We can prove that the log-rank conjecture holds for our notion of rank by reducing the problem of calculating the output $f(x, y)$ to an instance of CIS_G. I will assume that matrix multiplication is done using the standard dot product of rows and columns (as opposed to $\langle \cdot, \cdot \rangle \mod 2$).

Denote the Boolean-rank of an $N \times N$ matrix M as r. Then there must exist an $N \times r$ Boolean matrix U and an $r \times N$ Boolean matrix V such that $M = UV$. Note that if we use standard matrix multiplication, this is only possible when $\sum_{q=0}^{r} U_{iq} * V_{qj} \in \{0, 1\}$ for all $i, j \in \{1, \ldots, N\}$. This, in turn, implies that every row x in U and every row y in V share at most one index p such that $x_p * y_p = 1$ (for all remaining indices p', it must be the case that $x_{p'} * y_{p'} = 0$). We can exploit this fact in our reduction to CIS_G.

Given a function $f : X \times Y \to \{0, 1\}$ and the corresponding M_f with Booleank-rank$(M_f) = r$, we can construct a graph G as follows. G will have r nodes, denoted a_i for $i \in \{1, \ldots, r\}$. The node a_i corresponds to ith column of U and ith row of V. We then add edges as follows. For every row x in U, we treat x as a bitmap of nodes from G, where $x_i = 1$ means that the bitmap includes node a_i. We connect all nodes a_i that appear in the bitmap x together, so that they form a clique. Graph G is now ready to be used in CIS_G.

The protocol works as follows. Before communiction begins, both Alice and Bob know G, U and V. On input x^* that corresponds to row x in M_f, Alice looks at row x in U and treats it as a bitmap of nodes from G that she has to choose as her clique, C. By defintion of G, her choice of C is guaranteed to be a clique.

On input y^* that corresponds to column y in M_f, Bob looks at column y of V and treats it as a bitmap of nodes from G that he has to choose as his independent set, I. We can show by contradiction that Bob's choice of I must be an independent set. Assume that I is not an independent set. This means that there is an edge between some nodes in I. By construciton of G, this in turn implies that matrix U has some row y^+ that shares more than one index i with x such that $y_i^+ * x_i = 1$, which implies that not all matrices in relationship $M = UV$ are Boolean matrices, which is a contradiction. Hence I must be an indepdenent set.

Note that no communication has yet occurred. Alice and Bob now solve problem CIS_G using inputs C and I, using output of $CIS_G(C, I)$ as the result of running $f(x^*, y^*)$. Proof of correctness is trivial: if CIS_G outputs 1, then C and I share some common node a_i. This implies that $x_i * y_i = 1$, $\langle U_x, V^y \rangle = 1$. The last expression is equivalent to caculating the value of the element in M_f on which row x and column y intersect each other, i.e. the output of $f(x, y)$. Similarly, if CIS_G outputs 0, we have $\langle U_x, V^y \rangle = 0$ and $f(x, y) = 0$.

We have shown in class that communication complexity of CIS_G is bounded above by $O(\log^2 n)$, where n is the size of the graph. In our case, size of the graph is $r =$ Boolean-rank(M_f), so we can say that

$$D(M_f) \leq O(\log^2 \text{Boolean-rank}(M_f))$$

This proves that log-rank conjecture holds for the proposed notion of rank.

3. (a) First, we can observe that the definition of discrepancy implies that we can invert Boolean outputs of the function without changing the discrepancy, since we're using the absolute difference between probabilities.

We can start by looking at the full $X_1 \times \ldots \times X_k$ cylinder first. With this choice of S, the calculation for

$$\left| \Pr_\mu[f(x_1, \ldots, x_k) = 0 \wedge (x_1, \ldots x_k) \in S] - \Pr_\mu[f(x_1, \ldots, x_k) = 1 \wedge (x_1, \ldots x_k) \in S] \right|$$

reduces into just

$$\left| \Pr_\mu[f(x_1, \ldots, x_k) = 0] - \Pr_\mu[f(x_1, \ldots, x_k) = 1] \right|.$$

It's clear that the maximum possible value is 1, when the function either outputs always 1 or always 0. When we don't have this trivial case, the two terms begin to cancel each other out. That is, if we WLOG assume $\Pr_\mu[f(\ldots) = 1]$ is the largest of the two, we know that

$$\left| \Pr_\mu[f(\ldots) = 1] \right| > \left| \Pr_\mu[f(\ldots) = 0] - \Pr_\mu[f(\ldots) = 1] \right|.$$

Due to this fact, we can aim to maximise the value by somehow getting rid of the smaller component. We can rewrite the definition of discrepancy as follows:

$$\text{disc}_\mu(f) = \max_S \left| \Pr_\mu \left(f(x^k) = 1 \right) \Pr_\mu \left(x^k \in S \mid f(x^k) = 1 \right) - \Pr_\mu \left(f(x^k) = 0 \right) \Pr_\mu \left(x^k \in S \mid f(x^k) = 0 \right) \right|$$

From now on, we will WLOG focus on 1-monochromatic cylinder intersections and assume given f has nondeterministic complexity t. Identical argument applies to the co-nondeterminstic version. If we WLOG consider only 1-monochromatic cylinder intersections, it's clear the second term becomes zero and we're left off with just $\Pr_\mu \left(f(x^k) = 1 \right) \Pr_\mu \left(x^k \in S \mid f(x^k) = 1 \right)$.

We're given that f has nondeterministic complexity t, which implies that there exists a non-disjoint cover of f with 1-monochromatic cylinder intersections, with the size of this cover being 2^t. Note that because this cover can be non-disjoint, total sum of probabilities of an element belonging to these cylinders can be greater than 1. Since these cylinder intersections cover *all* 1s and there are 2^t of them, they must include at least one cylinder intersection S such that $\Pr_\mu(x^k \in S) \geq 2^{-t}$. Otherwise, we would have $\Pr_\mu(x^k \in S) < 2^{-t}$ for all of the 2^t 1-cylinder intersections in the cover, which is a contradiction since the sum of probabilities will be less than 1. As a result, we can conclude that there exists a 1-monochromatic cylinder intersection S such that $\Pr_\mu \left(x^k \in S \mid f(x^k) = 1 \right) \geq 2^{-t}$.

The function f and given distribution μ are fixed, so we can conclude that

$$\Omega(\Pr_\mu \left(f(x^k) = 1 \right) \Pr_\mu \left(x^k \in S \mid f(x^k) = 1 \right))$$

simplifies to just $\Omega(\Pr_\mu \left(x^k \in S \mid f(x^k) = 1 \right))$, and, finally, using the result from above, to $\Omega(2^{-t})$ when we use the largest 1-monochromatic cylinder intersection. This shows that the discrepancy is bounded below by $\Omega(2^{-t})$ given that nondeterministic or co-nondeterministic communication complexity of f is t.

(b) We can view the input for the problem DISJ_k as a $k \times n$ Boolean matrix, where each player i holds the string corresponding to ith row. In this formulation, the co-DISJ_k problem reduces to finding a column of all 1's (which implies a global intersection).

An all powerful prover can point out the index of the column that contains all 1's, which would require $\log n$ bits to represent. Then, the first player can check everyone else's values in that column and output 1 if they are indeed all 1's. The second player can check first player's value and output 1 in the same case, requring $O(\log n)$ bits in total to solve co-DISJ_k.