Sublinear Algorithms	March 29, 2012
Lecture 21	
Lecturer: Sofya Raskhodnikova	*Scribe(s): Megan Heysham*

1 Introduction

In this lecture, we construct (s,t)-Ruzsá-Szemerèdi graphs that, in combination with the results from the previous lecture, demonstrate a lower bound on query complexity of testing monotonicity of functions on posets [FLN+02].

2 Construction of (s,t)-Ruzsá-Szemerèdi graphs

A (s,t)-Ruzsá-Szemerèdi graph is a bipartite graph that has at least t incuded edge-disjoint matchings of size s.

We can think of each node on the top (and separately, each node on the bottom) as a string in $[a]^m$. Then we have a^m nodes in the top half of the graph and a^m nodes in the bottom. Thus our graph has $n = 2a^m$ nodes.

We will choose subets $T \subseteq [m]$ of size $m/3$. (We will determine later how to choose our subsets.) We will construct a matching M_T for each set T, and our edge set will be the union of these matchings.

2.1 Constructing M_T from T

First, we will partition the nodes $x \in [a]^m$ into levels according to $\sum_{i \in T} x_i$. For example, the bottom level will contain all strings that have $x_i = 1$ for each $i \in T$, while the rest of the string can take any value.

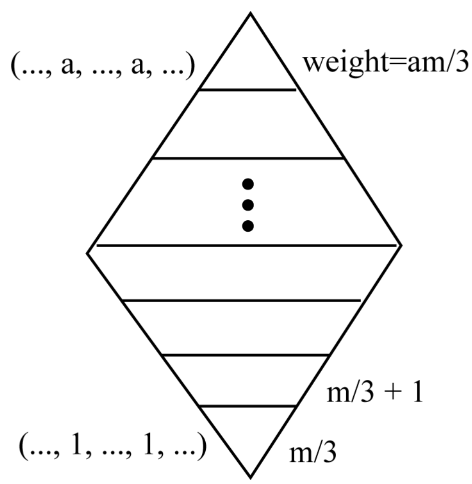

Figure 1: Nodes partitioned into levels

Then we will divide these levels into strips of width $m/3$ and color the nodes according to which strip they are in, following the pattern $(red, white, blue, white, red...)$.

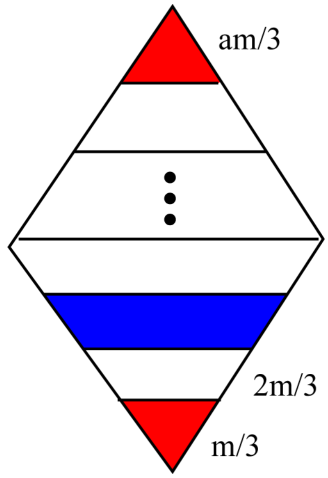

Figure 2: Levels divided into strips and colored

We will now use this coloring to construct M_T. For each red node r in the bottom of our graph, if, for all $i \in T$, $r_1 \leq a-2$, then we will add an edge from r to a blue node b in the top of our graph if $r_i = b_i$ for all $i \in T$ and $b_i = r_i + 2$ for all $i \notin T$.

Claim 1. *For any choice of T, the edges in M_T form a matching.*

Proof. For every node r in the bottom of our graph, there is at most one node b in the top of our graph such that the requirements that $r_i = b_i$ for all $i \in T$ and $b_i = r_i + 2$ for all $i \notin T$ hold. □

Claim 2. *Matchings formed from different choices of T are edge-disjoint.*

Proof. For each edge in M_T, the endpoints of that edge differ on all indices except those in T, for which they agree. Therefore, if $T \neq T'$, M_T and $M_{T'}$ are edge-disjoint. □

Claim 3. $|M_T| = n/8 - o(n)$

Recall $n = 2a^m$.
Intuition: Almost all red nodes on bottom and almost all blue nodes on top are matched.

Proof. $n/2$ nodes are in the bottom of the graph. Then $n/8 - o(n)$ of those nodes are red. Of the red nodes, there are $o(n)$ that have $r_i > a-2$ for some $i \in T$. These are the only ones that are not matched. Therefore, $|M_T| = n/8 - o(n)$. □

2.2 Choosing sets T

We say that a matching is *induced* if the induced subgraph of the nodes in the matching is equal to the matching itself.

We now consider the conditions that we need to place on our sets T to ensure that our matchings M_T are induced matchings. Consider matchings M_T and M_{T_1} such that for some edge $(r, b) \in M_T$, both r and b appear as end points of edges in M_{T_1}. (Since each edge in our graph comes from some matching, this is the only way that M_{T_1} would not be an induced matching.)

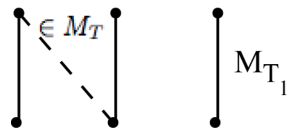

Figure 3: Example of noninduced matching

We know that in T_1's coloring, r is colored red and b is colored blue.

So $|\sum_{i\in T_1} b_i - \sum_{i\in T_1} r_i| \geq m/3$ (*)

Since $(r,b) \in M_T$, $|\sum_{i\in T_1} b_i - \sum_{i\in T_1} r_i| = |\sum_{i\in T_1}(b_i - r_i)| = |\sum_{i\in T_1}(2\cdot 1_T)| = 2\cdot|T\cap T_1|$

Now, by (*), we know that $2\cdot|T\cap T_1| \geq m/3$. So we know that if we have a noninduced matching, then $|T\cap T_1 \geq m/6$, and if we want all matchings to be induced, we must avoid this. Therefore, we require all sets T to agree pairwise on $\leq m/7$ positions.

This means that we will choose our Ts from a binary error correcting code with block length m, weight $m/3$, and minimum distance $8m/21$. We can get error correcting codes with these parameters that contain $2^{\Omega(m)}$ codewords.

Our lower bound on query complexity for testing monotonicity of functions on posets is thus $n^{\Omega(1/\log\log n)}$.

References

[FLN+02] Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In John H. Reif, editor, *STOC*, pages 474–483. ACM, 2002.