CS 671: Graph Streaming Algorithms and Lower Bounds **Rutgers: Fall 2020**

Problem set 6

Due: 11:59PM, October 20, 2020

Please solve *exactly one* of the problems below.

Problem 1. Recall that in Problem Set 1, we designed a single-pass semi-streaming algorithm that was able to recover an *exact* minimum spanning tree (MST) of a given weighted undirected graph in *insertion-only* streams. In this problem, we instead show that in *dynamic* streams, any single-pass algorithm for the MST problem requires $\Omega(n^2)$ space. Our proof here, instead of relying on the linear sketching characterization of dynamic streaming algorithms and multi-party simultaneous communication complexity, works in the two-party one-way communication model but with edge deletions.

A note about the model: In dynamic stream for weighted graphs, each input in the stream is a tuple $(u, v, \Delta, w(u, v))$ where u, v are two vertices of the graph, $\Delta \in \{-1, 1\}$ denotes whether the edge (u, v) is being inserted or deleted, and $w(u, v)$ specifies the weight of the edge (u, v). This in particular means that the updates cannot change the weight of edges directly; rather they would first delete the current edge (while specifying the weight it had when it was inserted first), and then insert it back again with the new weight.

(*i*) Consider the following *Augmented Index* communication problem: Alice is given a string $x \in \{0, 1\}^N$ and Bob is given an index $i \in [N]$ plus the *prefix* $x_1, \ldots, x_{i-1}$. Prove that similar to the original Index problem, the randomized one-way communication complexity of Augmented Index is also $\Omega(N)$ bits.

Hint: You should be able to easily modify the proof given for the Index problem in Lecture 1 to get this lower bound as well.

Test your intuition: What happens if Bob is additionally given the *suffix* $x_{i+1}, \ldots, x_N$ as well, i.e., he knows all of x except for x_i—does the lower bound still holds?

(*ii*) Design a reduction from Augmented Index problem to the following communication problem: Alice is given a *bipartite* graph $G_A = (L, R, E_A)$ with a unique weight over each edge; Bob is given a *subgraph* $G_B = (L, R, E_B)$ for $E_B \subseteq E_A$ (with the same weight of corresponding edges); the goal is to find the edge with the *minimum weight* in the graph $G_A \setminus G_B = (L, R, E_A \setminus E_B)$. Prove that the latter problem requires $\Omega(n^2)$ bits of communication.

(*iii*) Use the previous part to prove that any *dynamic* streaming algorithm for MST requires $\Omega(n^2)$ space.

Problem 2. In Lecture 6, we saw that the $\Theta(n^2/\alpha^3)$ space is sufficient and necessary for α-approximation of maximum matching in dynamic streams (up to $n^{o(1)}$-terms). Our goal in this problem is to design a better streaming algorithm for estimating the *size* of the maximum matching as opposed to finding the edges. For simplicity, we are going to focus on the following simpler problem: distinguishing between the case when a graph G has a perfect matching (*Yes* case) vs when its maximum matching is of size n/α at most (*No* case).

(*i*) Suppose we sample each vertex of the graph with probability $p := \frac{8}{\alpha}$ and consider the subgraph of G between the sampled vertices denoted by H. Prove that size of the maximum matching of H is with high probability: (1) more than $32 \cdot \frac{n}{\alpha^2}$ in the *Yes* case, and (2) less than $32 \cdot \frac{n}{\alpha^2}$ in the *No* case.

(*ii*) Use the previous part to design a single-pass streaming algorithm for distinguishing between *Yes* and *No* cases of the graph G given in a dynamic stream with high probability in $\widetilde{O}(n^2/\alpha^4)$ space.

(*iii*) **Bonus:** Extend this algorithm for estimating size of the maximum matching to within an α-approximation in any arbitrary graph G. Also, can you improve the space even further?