CS369G: Algorithmic Techniques for Big Data Spring 2015-2016

Problem Set 3

Prof. Moses Charikar *Due: June 1, 2016,* ***1:29pm***

Policy: You are permitted to discuss and collaborate on the homework but you must write up your solutions on your own. Furthermore, you need to cite your collaborators and/or any sources that you consulted. All homework submissions are subject to the Stanford Honor Code.

Submission: homework is due before class on Tuesday May 31, 2016. Email your solution to `cs369g.stanford@gmail.com` using the subject, `[PS3_2016]SUNetID` (e.g `[PS3_2016]jdoe`). Late submissions will not be accepted.

Length of submissions: include as much of the calculations that show that you understand everything that is going through the answer. As a rule of thumb after you have solved the problem, try to identify what are the main steps taken and critical points of a proof and include them. Unnecessary long answers to questions will be penalized. The points next to each question are indicative of the hardness/length of the proof.

1 k-Connectivity[10 points]

In the algorithm for k-connectivity that we saw in class, why does the algorithm need to maintain k independent ℓ_0 sketches? Identify what condition is violated if the algorithm maintains a single ℓ_0 sketch instead and performs all operations on this single sketch.

2 Lower bounds for existence of triangles[15 points]

Consider a graph stream describing an unweighted, undirected n-vertex graph G. Prove that $\Omega(n^2)$ space is required to determine, in one pass, whether or not G contains a triangle, even with randomization allowed.

3 Lower bounds for exact computation of F_2 [10 points]

Prove that computing F_2 exactly, in one pass with randomization allowed, requires $\Omega(\min\{m, n\})$ space. Construct an appropriate hard stream of length m, with universe size n, where $m = \Theta(n)$ and show that $\Omega(n)$ space is required on this stream.Then extend the result to multiple passes, with randomization allowed. The lower bound for p passes should be $\Omega(\min\{m, n\}/p)$.

4 Spanners for Weighted Graphs [25 points]

Recall that the distance estimation problem asks us to process a streamed graph G so that, given any $x, y \in V(G)$, we can return an t-approximation of $d_G(x, y)$, i.e., an estimate $\hat{d}(x, y)$ with the property:

$$d_G(x, y) \leq \hat{d}(x, y) \leq t \cdot d_G(x, y).$$

Here t is a fixed integer known beforehand. In class, we solved this using space $\tilde{O}(n^{1+2/t})$, by computing a subgraph H of G that happened to be a t-spanner.

(a) **[10 points]** Now suppose that the input graph is edge-weighted, with weights being integers in $[W]$. Each token in the input stream is of the form (u, v, w_{uv}), specifying an edge (u, v), and its weight $w_{uv} \in [W]$. Distances in G are defined using weighted shortest paths, i.e.,

$$d_{G,w}(x, y) := \min \left\{ \sum_{e \in \pi} w_e : \pi \text{ is a path from x to y} \right\}.$$

Give an algorithm that processes G using space $\tilde{O}(n^{1+2/t} \log W)$ so that, given $x, y \in V(G)$, we can then return a $(2t)$- approximation of $d_{G,w}(x, y)$. Give careful proofs of the quality and space guarantees of your algorithm.

(b) **[15 points]** In class, we saw that even the unweighted case, space $\Omega(n^{1+2/t})$ is necessary to preserve all distances up to a factor of t. What if we only care about only the max distance of a connected graph? The diameter of a graph $G = (V, E)$ is defined as $\mathrm{diam}(G) = \max\{d_G(x, y) : x, y \in V\}$, i.e., the largest vertex-to-vertex distance in the graph. A real number $\hat{d}$ satisfying

$$\mathrm{diam}(G) \leq \hat{d} \leq \alpha \cdot \mathrm{diam}(G)$$

is called an α-approximation to the diameter. Suppose that $1 \leq \alpha < 1.5$. Prove that, in the vanilla graph streaming model, a 1-pass randomized algorithm that α- approximates the diameter of a connected graph must use $\Omega(n)$ space. How does the result generalize to p passes?

5 Bipartite Graphs [40 points]

A graph G is called bipartite if $V(G)$ can be partitioned into two sets S, S^c such that all edges lie between vertices of those two sets $|E_G(S, S^c)| = |E(G)|$. Equivalently there exists a valid two coloring of the vertices, where a coloring is valid if there is no monochromatic edge. In the vanilla graph streaming model (only edge insertions)

(a) **[10 points]** Give a deterministic algorithm that uses $O(n \log n)$ space that decides whether a graph is bipartite. Give a proof of correctness.

(b) **[15 points]** Show that any randomized one-pass streaming algorithm that decides whether a graph is bipartite requires $\Omega(n)$ space.

(c) **[15 points]** Given a undirected graph G, we define its *bipartite double cover* by constructing a vertex set $\tilde{V}$ that contains two copies v_1, v_2 of every vertex $v \in V(G)$, and an edge set $\tilde{E}$ that contains the edges $\{u_1, v_2\}$ and $\{v_1, u_2\}$ for all edges $\{u, v\} \in E(G)$. Prove that the graph G is bipartite is equivalent to

$$\#\text{Connected Components}(\tilde{G}) = 2 \cdot \#\text{Connected Components}(G)$$

where $\tilde{G} = (\tilde{V}, \tilde{E})$. Show how to use this fact to design a streaming algorithm to test whether a graph is bipartite. Give the space requirements of your algorithm.