Sublinear Time and Space Algorithms 2022B – Problem Set 4

Robert Krauthgamer

Due: July 11, 2022

General instructions: Please keep your answers short and easy to read. You can use results, calculations or notation seen in class without repeating them, unless asked explicitly to redo them.

1. Consider the frequency-vector model, where the stream contains additive updates to a vector $x \in \mathbb{R}^n$ whose coordinates are integers bounded by $\text{poly}(n)$.

   Design a streaming algorithm that, given a query $i \in [n]$ (at the end of the stream), reports a $(1+\epsilon)$-approximation to $\|x_{[n]\setminus\{i\}}\|_2$ (the ℓ_2-norm of x when coordinate i is zeroed).

   Hint: Estimate the ℓ_2-norm of a virtual stream formed by subsampling the coordinates of x.

   Remark: As done in class, do not count storage of the algorithm's random coins.

2. In the Steiner Forest problem on a graph $G = (V, E)$, given k vertex-pairs $(u_1, v_1), \ldots, (u_k, v_k)$, the goal is to find a subset of the edges $E' \subset E$ of minimum size, such that in the subgraph $G' = (V, E')$, each u_i is connected to its respective v_i. Notice that an optimal G' can have between 1 and k connected components.

   Design a streaming algorithm for the following restricted setting: The graph G is known in advance and fixed to be a complete binary tree $T = (V, E)$ with n leaves, hence $|V| = O(n)$, and the input stream contains k vertex-pairs $(u_1, v_1), \ldots, (u_k, v_k)$. The algorithm should $(1+\epsilon)$-approximate the optimal size $|E'|$ (no need to report E' itself), with storage requirement of $(\epsilon^{-1} \log n)^{O(1)}$ bits.

   Hint: Since T is a tree, there is a unique path connecting u_i with v_i.

   Remark: As done in class, do not count storage of the algorithm's random coins.

3. An unweighted graph G is called *k-connected* if every cut $(S, \bar{S})$ contains at least k edges.

   Design a streaming algorithm that determines whether a dynamic graph G on vertex set $V = [n]$ (i.e., a stream of edge insertions and deletions) is 2-connected, using storage $\tilde{O}(n)$.

   Hint: First verify that G is connected by constructing a spanning tree T. Then classify all possible cuts $(S, \bar{S})$ into those that contain two or more edges of the tree T and the rest, and finally use additional (independent) samples to verify whatever is still needed.

4. We saw in class an algorithm (due to Feige) that, given a *connected* graph G and $\varepsilon \in (0, 1)$, estimates the graph's average degree d within factor $2 + \varepsilon$ in time $O((\frac{1}{\varepsilon})^{O(1)}\sqrt{n})$.

   Can you extend this algorithm to r-uniform hypergraphs (see definitions below)? Explain your modifications and the running time you obtain, or why that algorithm does not extend to hypergraphs.

Definitions: A *hypergraph* $G = (V, E)$ is a generalization of graphs where every hyperedge $e \in E$ is a subset (of arbitrary size) of the vertex set V. It is called *r-uniform* if every hyperedge $e \in E$ has cardinality $|e| = r$. Similarly to ordinary graphs, the *degree* of a vertex is the number of hyperedges containing it.

Guidelines: Focus on small r, say $r = 4$. Explain the differences and do not repeat proofs that are the same. There is no need to optimize dependence on ε.