Sublinear Time and Space Algorithms 2018B – Final Assignment

Robert Krauthgamer

Due: July 19, 2018 at 13:00

General instructions: Please keep your answers short and easy to read. You can use class material (results, notation, or references) without repeating it, unless asked explicitly to do so.

Specifically for this assignment: (1) Work on the problems completely by yourself, do not discuss it with other people or search for the solution online or at other sources, this is not the intention. (2) If you use any sources other than the class material (e.g., books, online lecture notes, wikipedia, or prior knowledge), point it out, even if you happened to find the solution online — I will not deduct points, but I want to know.

1. 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.

2. A matrix $A \in \mathbb{R}^{n\times n}$ is called *positive semidefinite* (in short PSD) if there exist vectors $v_1, \ldots, v_n \in \mathbb{R}^n$ such that the matrix entries can be written as $A_{ij} = \langle v_i, v_j\rangle$.

   We would like to estimate the absolute-sum norm of such A, defined as $|||A|||_1 := \sum_{ij} |A_{ij}|$ (this is the ℓ_1 norm when A is considered as a "flat" vector). The idea is to first read *all* diagonal entries of A, and then use these diagonal values to *sample* (read) more entries.

   (a) Let A be PSD, and assume for now $\sum_i A_{ii} = 1$. Define random variables

   $$Z_{ij} = \begin{cases} \frac{|A_{ij}|}{p_{ij}} & \text{with probability } p_{ij} := \sqrt{A_{ii}A_{jj}}, \\ 0 & \text{otherwise;} \end{cases}$$

   and let $Z := \sum_{i,j} Z_{ij}$. Prove that $\mathbb{E}[Z] = |||A|||_1$ and $\text{Var}(Z) \le |||A|||_1 \le |||A|||_1^2$, and also that $\sum_{i,j} p_{ij} \le n$.

   Hint: Verify (for yourself) that the PSD condition implies $A_{ii} \ge 0$ and $|A_{ij}| \le \sqrt{A_{ii}A_{jj}}$.

   (b) Design a fast algorithm that, given as input a PSD matrix A and $\varepsilon \in (0, 1)$, outputs with high probability a $(1 \pm \varepsilon)$-approximation to $|||A|||_1$. The running time should be much faster than reading all n^2 entries of A, for example $O(n/\varepsilon^2)$.

3. Let $L \in \mathbb{R}^{t \times n}$ be a randomized sketching matrix for ℓ_2-norm, namely, it satisfies

$$\forall y \in \mathbb{R}^n, \qquad \Pr_L \Big[\|Ly\|_2 \in (1 \pm \varepsilon)\|y\|_2 \Big] \geq 1 - \tfrac{1}{n}.$$

   Suppose we know this matrix and its evaluation on input $x \in \mathbb{R}^n$ (i.e., we have Lx and of course L itself). Show how to use this information to answer ℓ_2 point queries, namely, to estimate any desired coordinate $x_i = \langle x, e_i \rangle$ within additive error $O(\varepsilon)\|x\|_2$. Write explicitly the success probability of your estimate.

   Hint: For intuition, start with $\varepsilon = 0$ and $\|x\|_2 = 1$, and consider the estimator $1 - \frac{1}{2}\|Lx - Le_i\|_2^2$.

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.

THE END. Good luck!