Sublinear Time and Space Algorithms 2024A – Final Assignment

Robert Krauthgamer

Due: March 22, 2024 at 11:00

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 stumble upon the solution online — I will not deduct points, but I want to know.

Part I (36 points)

Answer all of the following 3 "short" questions. Provide a justification/proof sketch/counter-example, even for true/false questions. Keep your answer short, about 2-5 sentences; points might be deducted if the answer is too long.

A. Let A be a 0-1 matrix of size $(2^t-1)\times t$ with all possible (distinct) nonzero rows $A_i \in \{0,1\}^t$. Define the family $H = \{h_p : p \in \{0,1\}^t\}$, where each function $h_p : [2^t-1] \to \{0,1\}$ is defined by $h_p(i) := (Ap)_i = \langle A_i, p\rangle$, where all operations are performed in $GF[2]$ (i.e., modulo 2).

   Prove that this family H is pairwise independent, and conclude that one can generate n pairwise-independent random signs $X_1,\ldots,X_n \in \{\pm 1\}$ using $O(\log n)$ truly random bits.

B. We saw in class an ℓ_1 point query of a frequency vector $x \in \mathbb{R}^n$, where on query $i \in [n]$ it outputs $\tilde{x}_i$ such that

$$\Pr\left[\tilde{x}_i \notin x_i \pm \alpha\|x\|_1\right] \le 1/4.$$

   Explain how to refine the algorithm and/or analysis to achieve smaller error

$$\Pr\left[\tilde{x}_i \notin x_i \pm \alpha\|x_{-i}\|_1\right] \le 1/4,$$

   where x_{-i} is the vector x but with coordinate i zeroed.

   Assume for simplicity that $x_i \ge 0$ for all $i \in [n]$ (even though it is not necessary).

C. Consider the frequency-vector model with deletions, where the input is a stream of additive updates to a vector $x \in \mathbb{R}^n$, whose coordinates are integers bounded in absolute value by $\mathrm{poly}(n)$.

Explain how a streaming algorithm with storage requirement $(\varepsilon^{-1}\log n)^{O(1)}$ bits can approximate the number of indices $i \in [n]$ with $x_i = 1$, within additive error $\pm\varepsilon\|x\|_0$.

Hint: We mentioned in class that an ℓ_0-sampler draws a uniformly random index $i \in \mathrm{supp}(x)$, and can also report the corresponding frequency x_i.

Part II (64 points)

Answer 2 of the following 3 questions. As done in class, do not count storage of a streaming algorithm's random coins.

1. Design a sublinear-time algorithm that, given $0 < \varepsilon < 1$ and access to an n-vertex graph G, approximates the number of connected components, within additive error εn. The access to G allows to query, for each vertex, its degree and its neighbors list (e.g., ask for the i-th neighbor). What is the running time of your algorithm?

   Hint: Start with the special case that all connected components have size at most $1/\varepsilon$. Consider sampling a uniformly random vertex v and computing its connected component C_v.

2. Consider a dynamic geometric stream over $[\Delta]^d$ (i.e., a stream of insertions and deletions of points), let $X \subset [\Delta]^d$ be the final set of points, and assume it is well-defined (a point can be deleted only if it was inserted earlier). Assume also that $n := |X| \le \mathrm{poly}(\Delta^d)$. The cost of clustering X to a center point $c \in [\Delta]^d$ is defined as

   $$\mathrm{cost}_c(X) = \sum_{x \in X} \|x - c\|_2.$$

   Design a streaming algorithm that uses storage of $(d\log\Delta)^{O(1)}$ bits, and reports an $O(d\log\Delta)$-approximation to the cost of an optimal center, given by

   $$\mathrm{cost}^*(X) := \min_{c \in [\Delta]^d} \mathrm{cost}_c(X).$$

   Hint: Build a quadtree and at each tree level use an ℓ_1 point query, or its refinement from Question B above. Start with an easier task: given a query point $c \in [\Delta]^d$ at the end of the stream, report an $O(d\log\Delta)$-approximation of $\mathrm{cost}_c(X)$. Next, to estimate the cost of an optimal center, find at each tree level if some quadtree cell contains at least $n/2$ points.

3. In the sliding-window model (not seen in class), the input is an insertion-only stream σ of items from domain $[n]$, and there is also a parameter $w \le \mathrm{poly}(n)$ known in advance. At each time t, the *active window* is the list of w most recent items, defined as $W_t = (\sigma_{t-w+1}, \ldots, \sigma_t)$. The challenge is to avoid storing the entire active window, e.g., deleting each item that leaves the window requires storing the entire window.

   Design an algorithm that uses storage of $\mathrm{polylog}(n)$ bits, and can report, at any desired time t, a uniformly random sample from the distinct elements of the window W_t.

   Remark: A sample is requested only once, at time t not known to the algorithm.

   Hint: Assign to each item a random priority, using a random hash function $h : [n] \to [0,1]$. Start with the special case where every window W_t contains w distinct items, and analyze the expected storage requirement of your algorithm.

THE END. Good luck!