CS-621 Theory Gems November 27, 2012

Problem Set 4

Lecturer: Aleksander Mądry *Due: December 21, 2012*

Problem 1. Imagine that we want to use a bit array B of only m bits to implement a membership oracle for a set $S = \{s_1, \ldots, s_n\}$ of size n, in the case when the elements of S are coming from some (very) large universe $\mathcal{U}$, i.e., $n << |\mathcal{U}|$.

To this end, we start with all the bits of the array B set to zero and, for each $s_i \in S$ and each $j = 1, \ldots, k$, we set $B[h_j(s_i)]$ to one. Here, $h_1, \ldots, h_k$ are uniformly random (and independent) hash functions with $h_j : \mathcal{U} \to [m]$, for each j. (That is, each h_j hashes elements of the universe $\mathcal{U}$ into the set of numbers between 1 and m.)

Now, to answer a membership query for a given element $s \in \mathcal{U}$, we output "Yes" if $B[h_j(s)]$ is set to 1 for *all* $j = 1, \ldots, k$; and "No" otherwise.

(a) We cannot expect this membership oracle B to give always correct answers. (Why?) Compute, for a given query $s \in \mathcal{U}$ and fixed set S, what is the probability p_1 of having a "false negative", i.e., of B answering "No" when actually $s \in S$? What is the probability p_2 of having a "false positive", i.e., of B answering "Yes" when actually $s \notin S$. (Probability here is taken with respect to the randomness of the hash functions $h_1, \ldots, h_k$.)

(b) For a given ratio $\rho := \frac{m}{n}$ of the number of bits m to the size n of the represented set S, what is the value of k (as a function of ρ) that minimizes the sum $p_1 + p_2$?

Note: This approach is a very popular method for storing sparse sets, i.e., sets whose size is much smaller than the size of the universe.

Problem 2. We want to show that the Count-Min algorithm can be used to solve k-sparse ℓ_1-approximation problem. More precisely, we want to design a streaming algorithm that, for any $k \geq 1$, $\varepsilon > 0$ and $\delta > 0$, has $O(\frac{k}{\varepsilon} \log n \log \frac{m}{\delta})$ space complexity and, with probability at least $1 - \delta$, computes a k-sparse vector $\tilde{x}$ such that

$$|x - \tilde{x}|_1 \leq (1 + O(\varepsilon))\mathsf{Err}_1^k, \tag{1}$$

where x is the vector of true element frequencies and $\mathsf{Err}_1^k = \min_{\bar{x}, \|\bar{x}\|_0 \leq k} |\bar{x} - x|$ is the error of the best k-sparse ℓ_1-approximation to x.

(a) Consider a variation of the Count-Min algorithm in which each of the $d = \Theta(\log \frac{m}{\delta})$ arrays A_l has length $r = \frac{4k}{\varepsilon}$ (instead of $\frac{2}{\varepsilon}$). Show that, as long as, the frequency vector x is non-negative (i.e., the stream is well-formed)[1], with probability at least $1 - \delta$, the estimate vector $\hat{x}$ returned by this algorithm is such that

$$|x_j - \hat{x}_j| \leq \frac{\varepsilon}{k}\mathsf{Err}_1^k,$$

for each element $j \in [m]$.

Note: If you get stuck on this problem, email the lecturer to get a hint.

(b) For a given vector x', let x'_U, for some $U \subseteq [m]$, be the $|U|$-sparse vector resulting from zeroing out all the coordinates of x' except the ones in the set U. Show that if we run the variation of the Count-Min algorithm from (a) and take $\tilde{x} := \hat{x}_{\hat{U}}$, where $\hat{U}$ is the set of k largest coordinates of $\hat{x}$, then such $\tilde{x}$ satisfies condition (1).

Hint: Note that $|x - \tilde{x}|_1 = |x|_1 - |x_{\hat{U}}|_1 + |x_{\hat{U}} - \hat{x}_{\hat{U}}|_1$ and that the best k-sparse approximation to x corresponds to taking $\bar{x} := x_{\bar{U}}$ for some (unknown to us) $\bar{U} \subseteq [m]$ of size k.

[1] Note that this assumption is required for the version of the Count-Min algorithm we analyzed in class to work.

Problem 3. Consider a scenario in which the data stream consists of m (distinct) edges of a graph over n vertices (think about edges being elements of $[n] \times [n]$).

(a) Prove that any deterministic streaming algorithm that can determine whether the graph is bipartite or not has to have $\Omega(n)$ space complexity.

(b) Design a deterministic streaming algorithm that solves this task using $O(n \log n)$ space.

Note: The number of edges m can be $\Omega(n^2)$, so a trivial algorithm that just stores all the edges will not have $O(n \log n)$ space complexity.

Problem 4. Let us again consider the scenario when the data stream encodes an n-vertex graph with m edges. Let T be the number of triangles of this graph, i.e., T is the number of triples $\{u, v, w\}$ such that all the three edges (u, v), (v, w), and (u, w) are present in the graph.

(a) Prove that any deterministic algorithm that computes T has to have $\Omega(n^2)$ space complexity.

(b) (Extra credit) Design a randomized algorithm that, for any $\varepsilon > 0$ and $P > 0$, has $O(\frac{1}{\varepsilon^2 P})$ space complexity and approximates T up to an *additive* εmn error.

Note: If you get stuck on this problem, email the lecturer to get a hint.