University of Washington Due: June 14
Department of Computer Science and Engineering
CSE 522, Spring 2014

Problem Set 2

Problem #1:

Using any of the heavy hitters algorithms, we showed that we can solve the (γ, ϵ)-*heavy hitters* problem: Output any element that is a $(\gamma, 1)$-heavy hitter but do not output any element that is not a $(\gamma - \varepsilon, 1)$-heavy hitter. In this problem, you will consider the related (γ, ε)-quantile problem: Produce a small space output $\tilde{R}$ that allows one to estimate the relative rank of any element in the data stream, where the relative rank of j is the fraction of elements in the stream that are at most j; i.e., $\sum_{j' \leq j} f_{j'}/||f||_1$.

View the stream as $m = \log_2 M$ different streams of length n, where the values in the b-th stream are described by the first b bits of each element of the original stream. Show that by maintain information sufficient to solve the $(\gamma/m, \varepsilon/m)$-heavy hitters problem for each of these m streams, we can produce $\tilde{R}$ from which we can estimate the relative rank of any element $j \in [M]$ within ε. What is the space complexity of your resulting algorithm using the Misra-Gries or Count-Min sketches?

Hint: Use the fact that any range $[1, j]$ can be thought of as the disjoint union of m intervals in $[M]$, at most one corresponding to each of the m streams.

Problem #2:

Prove that using the coreset construction we gave for the MEB cost function, if S_1 is a $(1+\gamma)$-coreset for set P_1 and S_2 is a $(1+\gamma)$-coreset for P_2 then the $(1+\gamma)$-coreset S for $S_1 \cup S_2$ is actually a $(1+\gamma)$-coreset for $P_1 \cup P_2$. Use this to show that $\inf_{x \in \mathbb{R}^d} \max_{y \in \sigma} ||y - x||_2$ can be approximated by a data stream algorithm using space $O(1/\varepsilon^{(d-1)/2})$.

Problem #3:

In the *k-means* clustering problem, the input consists of a set of points $x_1, \ldots, x_n \in \mathbb{R}^d$ and a positive integer k and the goal is to output some partition $\mathcal{P}$ of $[n]$ into k disjoint subsets $P_1, \ldots, P_k$ as well as some "cluster centers" $z = (z_1, \ldots, z_k) \in (\mathbb{R}^d)^k$ not necessarily in the input set in order to minimize:

$$cost_{\mathcal{P}}(x) = \min_z \sum_{j=1}^{k} \sum_{i \in P_j} ||x_i - z_j||_2^2,$$

the sum of the squared Euclidean distances of the input points to their cluster centers.

Finding the optimal clustering for k-means is NP-hard but efficient approximation algorithms exist to find algorithms that are close to optimal. You will show that wlog such algorithms do not need to consider large dimensions.

(a) Given a cluster P_j show that the optimal value of z_j to choose is the *centroid* $z_j = \frac{1}{|P_j|}\sum_{i \in P_j} x_i$.

(b) Show that for any ε with $0 < \varepsilon < 1/2$ there is a linear map $A : \mathbb{R}^d \to \mathbb{R}^\ell$ for $\ell = O(\varepsilon^{-2} \log n)$ such that for all partitions $\mathcal{P}$.

$$(1 - \varepsilon) cost_{\mathcal{P}}(x) \le cost_{\mathcal{P}}(Ax) \le (1 + \varepsilon) cost_{\mathcal{P}}(x)$$

and that such a map A can be chosen randomly from a suitable distribution with small failure probability. Thus, up to a $1 \pm \varepsilon$ change in the approximation factor we can assume that the input vectors are in $O(\varepsilon^{-2} \log n)$ dimensions.

Hint: Use the fact that given each fixed P_j, the optimum choice of each z_j is a linear function of the input vectors.

Problem #4:

A property testing algorithm is called a *tolerant* testing algorithm with parameters $(\varepsilon', \varepsilon)$ iff it not only accepts inputs that satisfy property P but also for parameter $\varepsilon' < \varepsilon$ accepts those inputs that are ε'-close to having property P. Tolerant testing is natural to consider since input data collection can be noisy.

Show that when we measure distance as the usual fractional Hamming distance between input strings of length n, then any property testing algorithm that always accepts inputs in P and rejects all inputs ε-far from P using at most $q(\varepsilon, n)$ queries, is already an $(\varepsilon', \varepsilon)$-tolerant testing algorithm for P with success at least $2/3$ for $\varepsilon' = 1/(3q(\varepsilon, n))$.

Hint: Use a union bound over the queries made by the algorithm.