Fall 2014, CS 598: Algorithms for Big Data

Homework 2
Due: 09/23/2014

Instructions and Policy: Each student should write up their own solutions independently. You need to indicate the names of the people you discussed a problem with; ideally you should discuss with no more than two other people.

Solve as many problems as you can.

You need to typeset your solutions in Latex. Please write clearly and concisely - clarity and brevity will be rewarded. Refer to known facts as necessary. Your job is to convince me that you know the solution, as quickly as possible.

Problem 1 We have seen an algorithm to estimate the number of distinct elements in a stream. Equivalently it estimates the number of non-zero frequencies. Adapt the idea to estimate the number of frequencies that are odd.

Problem 2 The distinct element algorithm we saw in class (there is another one in the lecture notes) did not give a (linear) sketch. Adapt the ideas to develop a linear sketch that gives similar guarantees as the algorithm we saw but has the advantage of working for strict turnstile streams.

Problem 3 We discussed the AMS algorithm for estimating the frequency moments F_k that uses $\tilde{O}(n^{1-1/k})$ space. Explain how it can be generalized to the cash register model; i.e., each token is of the form (i_t, Δ) where Δ is a positive integer. Give careful pseudocode. Prove the correctness of your algorithm. In your proof, you may rely on the correctness proof we gave in class for the vanilla case: so dont reproduce those calculations, just explain what is new.

Problem 4 In class we saw the AMS sketch for F_2 estimation. To obtain a (ϵ, δ)-approximate estimate the sketch used $k = O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ pair-wise independent hash functions. Equivalently the algorithm can be viewed as computing $\mathbf{z} = M\mathbf{x}$ where M is a $k \times n$ projection matrix. To compute the final estimate we grouped the k entries $z_1, z_2, \ldots, z_k$ into $d = \log \frac{1}{\delta}$ groups, averaged the squares in each group to reduce variance, and took the median of the averages to obtain the final estimate.

Suppose we maintained a CountMin sketch of width w (assume w is even) and depth d using d pair-wise hash functions from $[n] \rightarrow [w]$. Now let the estimate for row ℓ be $\sum_{i=1}^{w/2} (C[\ell, 2i] - C[\ell, 2i-1])^2$. Finally we take the median of the row estimates. Calculate the mean and variance of each row estimate and show that if $w = O(1/\epsilon^2)$ and $d = O(\log \frac{1}{\delta})$ we obtain a (ϵ, δ)-approximation for F_2 as before.

The advantage of the new scheme is in the update time is to update $d = O(\log \frac{1}{\delta})$ counters instead of $k = wd = O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ counters.

Problem 5 In the Misra-Greis algorithm when the data structure is full (has k distinct items) and a new item that is not in the data structure arrives, all counters of current items are decremented; otherwise the counter for the item is incremented (and added if the data structure is not full). Consider the following variant. algorithm. The only difference from the Misra-Greis algorithm is when the data structure is full and the item is not already in the data structure. We replace the item in the data structure with the smallest counter value (break ties arbitrarily) by the new item and the counter is incremented by 1. Show that this algorithm has a performance similar to that of Misra-Gries.

Problem 6 Recall that we defined the α-heavy hitters in a stream as $S_\alpha = \{i \mid x_i \geq \alpha \|\mathbf{x}\|_1\}$. Our goal was to output a set S such that $S_\alpha \subseteq S \subseteq S_{\alpha-\epsilon}$ for some given $\epsilon < \alpha$. In class we used the CountMin sketch to find such a set but the algorithm was slow; it evaluated point queries for all $i \in [n]$ and picked the ones that had $\tilde{x}_i \geq \alpha \|\mathbf{x}\|_1$. Figure out a different sketch that would allow you find S much faster. In particular you should output S with polynomial dependence on $\frac{1}{\epsilon}, \log m, \log n$. You can assume that you are in the strict turnstile model. *Hint:* Maintain additional sketches as in the scheme we saw for answering range-queries so that you can do binary search for the heavy hitters from $[n]$.