# CS 598 (CRN 62819) Topics in Algorithms, Spring 2015

**Homework # 4** **Due: April 15, 2015**

---

Homework is due in class (on paper) on the due date.

1. **Piazza.** Make a Piazza comment related to Chapter 7 if you have not done so yet.

2. **Heavy hitters.** Recall that in class we gave a streaming algorithm for counting frequent elements that used $O(k \log n + k \log m)$ space, and for each element $s \in \{1, \ldots, m\}$ produced an approximate count $\hat{f}_s \in [f_s - n/(k+1), f_s]$ where $f_s$ is the true count of the number of occurences (frequency) of symbol $s$.

   Here is a different, randomized approach that uses a bit more space. What we will do is choose $t = O(k \log k)$ random locations in the stream, i.e., $t$ independent uniform random numbers $i_1, \ldots, i_t \in \{1, \ldots, n\}$. For each one, we record what we see there in the stream, and then count all subsequent copies of that element. E.g., if $i_1 = 37$ and we see a 3 there, we will count all 3's that we see from location 37 onward. This will be our estimate of the number of 3's in the stream. If multiple locations $i_j$ have 3's in them, then our estimate $\hat{f}_3$ will just be the largest of the counts, i.e., the count from the smallest such $i_j$. If some element $s$ is not seen in any of the locations $i_j$ then our estimate $\hat{f}_s$ is zero.

   (a) It is clear that our estimates $\hat{f}_s$ satisfy $\hat{f}_s \leq f_s$ where $f_s$ is the true frequency of symbol $s$. Argue that with high probability, for all $s$ we will have $\hat{f}_s \geq f_s - n/k$, so long as $t \geq ck \log k$ for sufficiently large constant $c$.

   (b) The above algorithm requires the ability to pick a random location in the stream. If we know $n$ ahead of time, this is easy. What if we don't know $n$ ahead of time? How can we implement the algorithm in that case? For this problem you can think of $t = 1$ (since you will just be replicating this for each counter).

3. **Sampling.** Recall that in the $CUR$ decomposition of $A$, we need to pick rows and columns of $A$ from a length-squared distribution. Specifically, we make $r$ independent draws of row indices from a distribution where index $i$ has probability proportional to the squared length of row $i$, and we make $k$ independent draws of column indices from a distribution where index $j$ has probability proportional to the squared length of columns $j$.

   Suppose we are in the streaming model and the entries of $A$ are arriving in an arbitrary order. Specifically, we are presented with a series of triples $(i, j, A_{ij})$ in some arbitrary order until all non-zero entries of $A$ have been presented.

   (a) Give a streaming algorithm that will select a single row index $i$ of $A$ from the correct distribution. The space used by your algorithm should only be *logarithmic* in $n$ and $m$. (Your space may be linear in the number of bits $b$ it takes to write down the largest single entry of $A$.)

(b) Give a streaming algorithm that will select a single column index $j$ of $A$ from the correct distribution. (This will look very much like your algorithm for part (a)).

(c) Say how to put these together (using a factor $(r + s)$ more space) to select $r$ rows and $s$ columns.