University of Washington
Department of Computer Science and Engineering
CSEP 521, Winter 2021

February 9, 2021

Homework 6, Due Thursday, February 18, 2021

Problem 1 (10 points):

Argue that it is impossible to compute the median of a stream of n values with a sub-linear memory algorithm. Specifically, show that you need to remember $\Omega(n)$ values (which means that you need to remember cn values, for some $c > 0$). For this problem, you can assume you know the value of n, and need to report the median when n values have been processed.

Problem 2 (10 points):

For the count min problem, the bound presented in class for the error in the number of occurrences of x, f_x with one hash table was:

$$\text{Prob}\left[HT[h(x)] > f_x + \frac{2n}{b}\right] \leq \frac{1}{2}.$$

The expected number of items that are hashed to the same bucket as x is $\frac{n}{b}$, so this is saying the the probability that an additional $\frac{n}{b}$ elements are hashed into the cell is at most $\frac{1}{2}$. The purpose of this problem is to derive stronger bounds in two cases where the distribution is given.

a) Suppose that the data set consists of n distinct values. (x is one of these values.) Give a bound on Prob $\left[HT[h(x)] > f_x + \frac{2n}{b}\right]$.

   Hint: The hashing of items to a cell (the one containing x) can be viewed as a Bernoulli process with success probability $p = \frac{1}{b}$. This means that the number of elements hashed to a cell is the sum of n independent random variables, $S_n = X_1 + X_2 + \cdots + X_n$, where $X_i = 1$ with probability p and $X_i = 0$ with probability $1 - p$. The expected value of S_n is pn. *Hoeffding's Inequality* gives a bound on probability that S_n exceeds the mean by more than ϵn for $\epsilon > 0$. The bound you want to apply is Prob$[S_n \geq (p + \epsilon)n)] \leq e^{-2\epsilon^2 n}$.

b) Suppose that the data set has only four distinct values, all of which occur $\frac{n}{4}$ times. (x is one of these values.) Give a bound on Prob $\left[HT[h(x)] > f_x + \frac{2n}{b}\right]$.

Programming Problem 3 (15 points):

Implement the Count-Min Sketch, and evaluate its performance on the provided data set, citylist.txt.

You should parameterise your implementation by b, the number of buckets, and j, the number of hash tables/hash functions.

The dataset is United States automobile accident data reported between 2016 and 2020. The dataset has over four million records with lots of information. The full dataset is available on Kaggle.com

(https://www.kaggle.com/sobhanmoosavi/us-accidents). For the purpose of this assignment, you will only use the city/state of each accident, which have been extracted and put in a text file containing one city per line. (Cities and states are combined into a single string, such as Seattle_WA to disambiguate cities such as Springfield which are present in multiple states.) You should consider this as a data stream, with cities being presented one at a time.

The purpose of the assignment is to identify the approximate counts for the most frequently occurring cities. Focus on being able to identify the cities that contribute at least 0.5% of the total reported accidents. You should compare the Count-Min estimates with the actual occurrences (which you should compute separately).

You will need to use separate hash functions for each table - you are welcome/encouraged to develop or identify your own hash functions, but to potentially save you time, the instructor has made his hash table code available. The code uses Carter-Wegman universal hash functions, which hash x to $ax + b \bmod p$ where p is a prime, and a and b are integers in the range $[1..p-1]$ which are randomly chosen when the hash function is created. Strings are converted to integers by treating the characters as coefficients of a polynomial, which is then evaluated at a fixed value. This arithmetic is again done mod a different prime p'.

Programming Problem 4 (15 points):

Analyze the errors in your counts, and compare with the predicted error from lecture and notes. The main result presented in class for the error in the number of occurrences of x, f_x with one hash table was

$$\text{Prob}\left[HT[h(x)] > f_x + \frac{2n}{b}\right] \leq \frac{1}{2}$$

and the error with j hash tables was

$$\text{Prob}\left[\min_{i=1}^{j} HT[i][h_i(x)] > f_x + \frac{2n}{b}\right] = \prod_{i=1}^{j} \text{Prob}\left[HT[i][h_i(x)] > f_x + \frac{2n}{b}\right] \leq \left(\frac{1}{2}\right)^j.$$

Do these bounds align with results from your data?

For the Heavy Hitters problem, consider the case where we want to find the $\frac{n}{k}$ Heavy Hitters with $k = 200$. For an ϵ approximation, we are going to choose $\epsilon = \frac{1}{800}$, so you want to define your parameters so that you identify everything that occurs at least $\frac{n}{200}$ times, and you don't want to identify anything that occurs fewer than $\frac{3n}{800}$ times. How does your error rate compare with what the theory predicted?

How does varying the number of hash functions influence the results?