CSE 521: Design and Analysis of Algorithms **Winter 2017**

Problem Set 2

Deadline: Jan 30th (at 12:00PM) in Canvas

The goal of this problem set is to learn the idea of *minhash*. Minhash is a hash function which is commonly used in practice to estimate the *Jaccard similarity* of two sets.

1) Suppose we have a universe U of elements. For$A, B \subseteq U$, the Jaccard distance of A, B is defined as

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

This definition is used practice to calculate a notion of similarity of documents, webpages, etc. For example, suppose U is the set of English words, and any set A represents a document considered as a bag of words. Note that for any two $A, B \subseteq U$, $0 \leq J(A, B) \leq 1$. If $J(A, B)$ is close to 1, then we can say $A \approx B$.

Let $h : U \to [0, 1]$ where for each $i \in U$, $h(i)$ is chosen uniformly and independently at random. For a set $S \subseteq U$, let $h_S := \min_{i \in S} h(i)$. Show that

$$\mathbb{P}[h_A = h_B] = J(A, B).$$

Now, if we have sets $A_1, A_2, \ldots, A_n$, we can use the above idea to figure out which pair of sets are "close" in time essentially $O(n|U|)$. We can also obtain a $1 \pm \epsilon$ approximation of $J(A, B)$ with high probability by using $O(\log(n)/\epsilon^2)$ independently chosen hash functions. Note that the naive algorithm would take $O(n^2|U|)$ to calculate all pairwise similarities.

2) Let $X_1, \ldots, X_n$ be independent random variables uniformly distributed in $[0, 1]$ and let $Y = \min\{X_1, \ldots, X_n\}$. Show that $\mathbb{E}[Y] = \frac{1}{n+1}$ and $\mathrm{Var}(Y) \leq \frac{1}{(n+1)^2}$.

3) Consider the following algorithm for estimating F_0, the number of unique elements in a sequence $x_1, \ldots, x_m$ in the set $\{0, 1, \ldots, n-1\}$. Let $h : \{0, 1, \ldots, n-1\} \to [0, 1]$ s.t., $h(i)$ is chosen uniformly and independently at random in $[0, 1]$ for each i. We start with $Y = 1$. After reading each element x_i in the sequence we let $Y = \min\{Y, h(x_i)\}$.

   a) Show that by the end of the stream $\frac{1}{\mathbb{E}[Y]} - 1$ is equal to F_0.

   b) Use the above idea to design a streaming algorithm to estimate the number of distinct elements in the sequence with multiplicative error $1 \pm \epsilon$. For the analysis you can assume that you have access to k independent hash functions as described above. Show that $k \leq O(1/\epsilon^2)$ many such hash functions is enough to estimate the number of distinct elements within $1 + \epsilon$ factor with probability at least 9/10.

4) We are given a directed graph $G = (V, E)$ with n vertices and m edges. For each vertex $v \in V$, let N_v denote the set of all nodes reachable from v (including v itself) and let $n_v = |N_v|$. Our goal is to find n_v for each vertex $v \in V$. The best exact algorithm for this problem runs in time $O(\min\{m \cdot n, n^{2.373}\})$. Here, we will explore a randomized approximation algorithms that runs in time $O(m \log n)$. The algorithm works as follows:

   Step 1) For each vertex $v \in V$ choose a uniformly and independently random number $s_v \in [0, 1]$. Call this number the *minhash* of v.

   Step 2) For each vertex $v \in V$, let $h_v := \min_{w \in N_v} s_w$. The h_v values for all vertices $v \in V$ can be found in time $O(m \log n)$!

Step 3) Let $\tilde{n}_v = \frac{1}{h_v}$.

Repeat all of the above steps k times and for each v output as your estimate for n_v, the median of $\tilde{n}_v$ values from the k runs.

a) Describe an implementation of step 2 that runs in $O(m \log n)$ time.

b) Implement the above algorithm for $k = 20$ to approximate n_v for each vertex v in the vertex set of a directed graph. Implement also an exact algorithm for computing these n numbers. Run your algorithm on oregon2_010331.txt file in the course website. The first line of the input indicates the number of nodes and directed edges and each following line indicates the source and destination of a directed edge in order.

   Use the following format for your output: In each line write 3 numbers separated by space corresponding to a node v, (i) the id of v, (ii) n_v (iii) $\tilde{n}_v$ returned by your implementation of the randomized algorithm. Sort all lines increasingly according to the node ids.

   Please write down your output in the Canvas textbox for Problem 3.

c) Give the best bound you can on the error of the estimate of n_v's output by the approximate algorithm as a function of k. You can use Chebycheff bounds or Chernoff bounds.