Homework #4

(Due: Dec 4)

Task 1. [**40 Points**] **Shush! Or you will be hashed into a really tiny hash table!**

Please refresh your knowledge on *hash tables* (Section 11.2 of the textbook[1]) before you proceed. Suppose we hash m keys into a hash table of size n. We use hashing with chaining, and assume that each key is equally likely to be hashed into any of the n slots independent of the other keys. Let $L(n, m)$ denote the length of the longest chain in the table after inserting all m keys, and let $l(n, m)$ be the length of the shortest chain. Prove that each of the following bounds hold with high probability in n, where $\alpha > 0$ and $\epsilon \in (0, 1)$ are constants[2].

(*a*) [**8 Points**] $L\left(n, \alpha n^{1-\epsilon}\right) = \mathcal{O}(1)$.

(*b*) [**8 Points**] $L(n, \alpha n) = \mathcal{O}\left(\frac{\log n}{\log \log n}\right)$.

(*c*) [**8 Points**] $L(n, \alpha n \log n) = \mathcal{O}(\log n)$.

(*d*) [**8 Points**] $l(n, \alpha n \log n) \geq 1$.

(*e*) [**8 Points**] $L\left(n, \alpha n^2\right) = \mathcal{O}(n)$.

Task 2. [**40 Points**] **"Well, I guess I've been chosen!"[3] (Jerry Seinfeld in "Seinfeld")**

Consider the randomized median finding algorithm given in Figure 1 which works by choosing a small number of input elements uniformly at random, and using them to possibly reduce the number of candidates for median. Show that parts (*a*) – (*e*) hold w.h.p. in n.

(*a*) [**8 Points**] $n^{\frac{3}{4}} - o(\sqrt{n}) \leq |S| \leq n^{\frac{3}{4}} + o(\sqrt{n})$ (in Step 1).

(*b*) [**8 Points**] $r_x < \frac{n}{2} < r_y$ (in Step 5).

(*c*) [**8 Points**] $|Q| = \mathcal{O}\left(n^{\frac{3}{4}}\right)$ (in Step 6).

[**Hint:** Let z be the median element of A, and let $L = \{\ q \in Q \mid q < z\ \}$ and $H = \{\ q \in Q \mid q > z\ \}$. Show that w.h.p. in n neither $|L| > 2|S|$ nor $|H| > 2|S|$ holds.]

(*d*) [**8 Points**] RandMedian finds the median element of A.

(*e*) [**8 Points**] RandMedian runs in $\mathcal{O}(n)$ time.

[1] Chapter 11 (Hash Tables), Introduction to Algorithms (3rd Edition) by Cormen et al.

[2] you are free to choose a suitable positive value of α for each subtask, but task 1(a) must hold for all $\epsilon \in (0, 1)$

[3] when the doctor finally calls him in after a long wait in a crowded waiting room

RandMedian(A, n)

(Input is a set A of n elements from a totally ordered universe, where n is an odd positive integer. Output is the median element of A, i.e., the $\frac{1}{2}(n+1)$-th smallest element of A.)

1. choose each element of A with probability $n^{-\frac{1}{4}}$ independent of other elements, and collect them in a set S
2. sort the elements in S using an optimal sorting algorithm
3. find $x, y \in S$ such that $rank_{(S)}(x) = \frac{1}{2}|S| - \sqrt{n}$ and $rank_{(S)}(y) = \frac{1}{2}|S| + \sqrt{n}$
4. compute $r_x = rank_{(A)}(x)$ and $r_y = rank_{(A)}(y)$
5. ***if*** $r_x < \frac{n}{2} < r_y$ ***then***
6.   find $Q = \{ z \in A \mid x < z < y \}$
7.   sort the elements in Q using an optimal sorting algorithm.
8.   find $z \in Q$ such that $rank_{(Q)}(z) = \frac{1}{2}(n+1) - r_x$.
9.   return z as the median element of A
10. return Fail

Figure 1: A Monte Carlo algorithm for computing the median of a set.