CMPT 307 — Data Structures and Algorithms

Exercises on Sorting, Selection, and Graph Traversal. Due: Friday, October 11th

Reminder: the work you submit must be your own. Any collaboration and consulting outside resourses must be explicitely mentioned on your submission.

1. You are given an array of integers, where different integers may have different number of digits, but the total number of digits over all the integers in the array is n. Show how to sort the array in $O(n)$ time.

2. Show that the second smallest of n elements can be found with $n + \lceil \log n \rceil - 2$ comparisons in the worst case.

3. Suppose that you have a "black box" worst-case linear time median subrouting. Give a simple, linear time algorithm that solves the k-Smallest problem for an arbitrary k.

4. You are interested in analyzing some hard-to-obtain data from two separate databases. Each database contains n numerical values, so there are $2n$ values total, and you may assume that no two values are the same. You would like to determine the median of this set of $2n$ values, which we will define here to be the n-th smallest value.

   However, the only way you can access these values is through queries to the databases. In a single query, you can specify a value k to one of the two databases, and the chosen database will return k-th smallest value that it contains. Since queries are expensive, you would like to compute the median using as few queries as possible.

   Give an algorithm that finds the median using at most $O(\log n)$ queries.

5. For n distinct elements $x_1, x_2, \ldots, x_n$ with positive weights $w_1, w_2, \ldots, w_n$ such that $\sum_{i=1}^{n} w_i = 1$, the *weighted (lower) median* is the element x_k satisfying

   $$\sum_{x_i < x_k} w_i < \frac{1}{2}$$

   and

   $$\sum_{x_i > x_k} w_i \leq \frac{1}{2}.$$

   (a) Show how to compute the median of n elements in $O(n \log n)$ worst-case time using sorting.

   (b) Show how to compute the weighted median in $\Theta(n)$ worst-case time using a linear time median algorithm such as Selection.

6. The square of a directed graph $G = (V, E)$ is the graph $G^2 = (V, E^2)$ such that $(u, w) \in E^2$ if and only if for some $v \in V$, both $(u, v) \in E$ and $(v, w) \in E$. That is, G^2 contains an edge between u and w whenever G contains a path with exactly two edges between u and w. Describe efficient algorithms for for computing G^2 from G for both the adjacency lists and adjacency matrix representations of G. Analyze the running time of your algorithms.

7. In social networks analysis it is sometimes useful to find not only the distance between two vertices in a graph or a shortest path between them, but the number of such paths. It turns out this problem can be solved efficiently. More precisely, given an undirected graph (no lengths) $G = (V, E)$ with $|V| = n$ and $|E| = m$, and two vertices $v, w \in V$, suggest an algorithm that outputs the number of shortest $v - w$-paths in G. (The algorithm should not list all the paths, just the number will do.) The running time of your algorithm should be $O(m + n)$.

8. Give an algorithm to detect whether a given undirected graph contains a cycle. If the graph contains a cycle, then your algorithm should output one (not all). The running time of your algorithm should be $O(m + n)$ for a graph with n vertices and m edges.