CS 473: Algorithms, Fall 2016
HW 3 (due Tuesday, September 20th at 8pm)

This homework contains three problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this home work, each student can work in a group with upto three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

1. You are given a directed graph $G = (V, E)$ where each edge e has a length/cost c_e and you want to find shortest path distances from a given node s to all the nodes in V. Suppose there are only k edges $f_1 = (u_1, v_1), f_2 = (u_2, v_2), \ldots, f_k = (u_k, v_k)$ that have negative length, and the rest have non-negative lengths. Here think k is small, say a constant number. The Bellman-Ford algorithm for shortest paths with negative length edges takes $O(nm)$ time where $n = |V|$ and $m = |E|$. Show that you can take advantage of the fact that there are only k negative length edges to find shortest path distances from s in $O(kn \log n + km)$ time — effectively this is the running time for running Dijkstra's algorithm k times. Your algorithm should output the following: *either* that the graph has a negative length cycle reachable from s, *or* the shortest path distances from s to all the nodes $v \in V$.

   Hint: First solve the case when there is only a single negative length edge. You will get half the credit if you only solve this case. Then solve the case of two negative length edges and see if you can generalize that approach. Note that this is only one approach, there may be others.

2. In the selection problem we are given an array A of n numbers (not necessarily sorted) and an integer k, and the goal is to output the rank k element of A. Consider a randomized algorithm where we pick a number x uniformly at random from A and use it as a pivot as in quick sort to partition A into numbers less than equal to x and numbers greater than x. The algorithm recurses on one of these arrays depending on k and the size of the two arrays. It can be shown that this algorithm runs in $O(n)$ expected time and has the advantage of being quite simple when compared to the median of median (deterministic) algorithm.

   (A) Write down a description of randomized quick selection in pseudocode. Show that the expected depth of the recursion of randomized quick selection is $O(\log n)$. (You can also prove that the expected running time is $O(n)$ but you should prove this for yourself since it will help you in the next part. However, you don't have to submit it as part of the home work.)

   Hint: Write a recurrence for the depth of the recursion.

   (B) Let $A_1, A_2, \ldots, A_h$ be h *sorted* arrays where A_i has n_i elements. Let $n = \sum_{i=1}^{h} n_i$. Assume that the arrays have distinct elements. Describe a randomized algorithm that given integer k finds the k'th smallest element in the combined set of arrays in $O(h \log^2 n)$ expected time.

   Hint: Adapt the randomized quick selection algorithm and the analysis from the first part.

3. Let $G = (V, E)$ be an undirected graph. Recall that $S \subseteq V$ is a dominating set if the following property holds: for every $u \in V$ we have $u \in S$ or some neighbor of u is in S. In the domatic partition problem we are given a graph G and the goal is to find a maximum number of mutually disjoint dominating sets in G. Let δ be the degree of a minimum degree node in G. It is easy to see that the domatic number is at most $(\delta + 1)$ since each dominating set has to contain u or some neighbor of u where u is a node with degree δ. In this problem we will see that the domatic number of a graph on n nodes and minimum degree δ is at least as large as $\lceil \frac{\delta+1}{c \ln n} \rceil$ for some sufficient large universal constant c. Note that this guarantees only 1 dominating set if $\delta < c \ln n$ (the entire vertex set can be chosen as the dominating set). Let $k = \lceil \frac{\delta+1}{c \ln n} \rceil$. Consider the following randomized algorithm. For each node u independently assign a color $g(u)$ that is chosen uniformly at random from the colors $\{1, 2, \ldots, k\}$.

   (A) For a fixed node v and a fixed color i show that with probability at least $1 - \frac{1}{n^2}$ there is a node with color i that is either v or a neighbor of v. Choose c sufficiently large to ensure this.

   (B) Using the above show that for a fixed color i the set of nodes that are colored i form a dominating set for G with probability at least $1 - \frac{1}{n}$.

   (C) Using the above two parts argue that the domatic number of G is at least k.

   Hint: The simple union bound is useful for this problem.

The remaining problems are for self study. Do *NOT* submit for grading.

- Show that quick select takes $O(n)$ time in expectation. See if you can prove that the number of comparisons is at most $4n$ for an array of size n.
- Solve the first problem on shortest paths via dynamic programming in the file below. `https://courses.engr.illinois.edu/cs374/fa2015/labs/lab16-shortest-path-dynamic-prog.pdf`.
- Solve the first problem on shortest walks in the file below. `https://courses.engr.illinois.edu/cs374/fa2015/homework/hw8.pdf`.
- Read Jeff's notes on randomized quick sort but in particular the nuts and bolts problem. Solve some of the basic problems on probability and randomized algorithms at the back of the notes. `http://jeffe.cs.illinois.edu/teaching/algorithms/notes/09-nutsbolts.pdf`.