CS 180, Fall 2015 Homework 3

The following homework is due on Wednesday, Oct 21st at the beginning of lecture. When submitting your homework, please include your name at the top of each page. If you submit multiple pages, please staple them together. Late submissions and electronic submissions by Email are not accepted.

1. We discussed a shortest path algorithm in class with running time $O(n^3)$. The algorithm works as follows:

   Input: Graph $G = (V, E)$ with weights on edges, and source node s

   Let S be the set of explored nodes

   For each $u \in S$, we store a distance $d(u)$. Initially $S = \{s\}$ and $d(s) = 0$.

   While $S \neq V$

   Select a node $v \notin S$ with at least one edge from S for which $d'(v) = \min_{e=(u,v):u \in S} d(u) + w_e$ is as small as possible. (w_e is the weight of edge e)

   Add v to S and define $d(v) = d'(v)$

   EndWhile

   If we implement the first line inside While loop in the above algorithm naively, it takes $O(n^2)$ because we need to march over all edges of nodes v, $v \notin S$. Since the while loop repeats n times (once for every vertex) and each iteration takes $O(n^2)$ time, the total running time of this algorithm is $O(n^3)$. (Hint: consider only the edges of the node added last to S.)

   (a) Can you modify the first step in the while loop, so that finding a node $v \notin S$ with the minimum distance to S takes $O(n)$ instead of $O(n^2)$? This will reduce the total running time of the algorithm to $O(n^2)$ instead of $O(n^3)$.

   (b) For sparse graphs where the number of edges m is much smaller that n^2, we can further reduce the running time. Can you change the above algorithm so that it runs in $O(m \log n)$? Note that for sparse graph $O(m \log n)$ is much faster than $O(n^2)$. (hint: Keep the nodes of $V - S$ in a Heap).

2. In class we saw an example of a scheduling problem that can be reduced to finding the longest path in a DAG (*Directed Acyclic Graph*). Consider the following variation on the problem. You are given a set of time intervals. Each interval I has a start time S_I, an end time E_I and a price P_I. We are looking to find a set of non-conflicting intervals with the highest total price. Give a reduction of this problem to the problem of finding the longest path in a DAG?

3. Given an undirected graph with integer edge weights and a source vertex s, describe an algorithm that uses BFS to compute the shortest path from the source vertex to all other vertices. what is the running time of your algorithm?

4. The diameter of a tree is the number of edges in the longest path in the tree.

   (a) Given a tree, show how you can calculate the diameter using BFS twice.

   (b) Write a recursive algorithm to calculate the diameter of the tree using the following idea: prune all the leafs of the tree and then you get a smaller tree. Find the diameter of the smaller tree, and then recover the diameter of the original tree. Give a recursive and iterative algorithm based on the recursion and analyze the time complexity of your iterative algorithm. The complexity of your algorithm should be $O(n)$.