MCS 401: Computer Algorithms I (Fall 2020)

Homework 3

Due at 2:00pm CDT, Wednesday, Oct 28

1. (2 points) Consider the following variant of the Interval Scheduling problem. There are n jobs and each job has a start time s_i and an end time f_i. There is a single machine that can run at most one job at any given time. The jobs are now *daily* jobs. Once accepted, it must run continuously every day between its start and end times. (Note that a job can start before midnight and end after midnight.)

   a) Design an algorithm that accepts as many jobs as possible. Your algorithm should run in time $O(n^2)$ and output an optimal schedule (a set of intervals).

   b) Prove the correctness of your algorithm and analyze its running time.

   Example: Suppose there are 4 jobs, specified by (start-time, end-time) pairs: $(9\text{pm}, 3\text{am})$, $(6\text{pm}, 6\text{am})$, $(3\text{am}, 1\text{pm})$, and $(2\text{pm}, 7\text{pm})$. The optimal solution would be to pick 3 jobs $(9\text{pm}, 3\text{am})$, $(3\text{am}, 1\text{pm})$, and $(2\text{pm}, 7\text{pm})$, which can be scheduled without overlapping.

   (Hint: first enumerate an interval $I_j = 1, \ldots, n$. How could we compute a schedule O_j with maximum size among all valid schedules that contain I_j?)

2. (2 points) Consider the following scheduling problem. There are n jobs and a single machine. Each job has a length ℓ_i and a *weight* w_i. The weight w_i represents the importance of job i.

   a) Let f_i be the finishing time of job i. Design a greedy algorithm to minimize the weighted sum of the completion times $\sum_{i=1}^{n} w_i f_i$. Your algorithm should run in time $O(n \log n)$ and output an ordering of the jobs.

   b) Prove the correctness of your algorithm and analyze its running time.

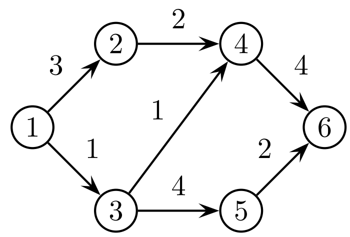
   Example: Suppose there are two jobs: $t_1 = 1$, $w_1 = 2$, $t_2 = 3$, and $w_2 = 1$. Doing job 1 first would give $f_1 = 1$, $f_2 = 4$, and a weighted sum of $2 \cdot 1 + 1 \cdot 4 = 6$, which is optimal. Doing job 2 first would yield $f_1 = 4$, $f_2 = 3$, and a larger weighted sum of $2 \cdot 4 + 1 \cdot 3 = 11$.

   (Hint: how does the weighted sum change if we swap two adjacent jobs?)

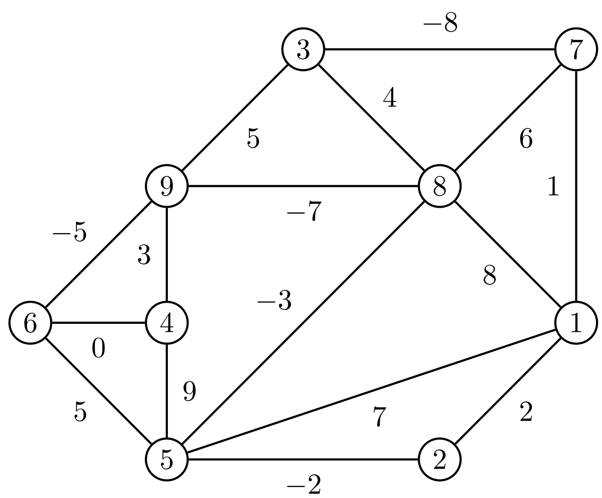
3. Simulate Dijkstra's algorithm on the following (directed) graph with starting node $s = 1$.

   a) What is the output of Dijkstra's algorithm (i.e., the length of the shortest path from s to every other node according to Dijkstra's algorithm)?

   b) Suppose the weight of edge $(2, 4)$ is now changed to -2. What is the output of Dijkstra's algorithm after this change? Does Dijkstra's algorithm work on this modified graph?



4. Compute the Minimum Spanning Tree (MST) of the following (undirected) graph using Kruskal's algorithm and Prim's algorithm. (The description of these two algorithms allows non-positive weights.)



a) Draw the MST. Give the orderings in which the edges are added in Kruskal's algorithm. Give the orderings in which the edges are added in Prim's algorithm starting at node 1.

b) In general, do Kruskal's algorithm and Prim's algorithm work when there are negative weights? (You need to answer only "Yes" or "No".)

5. (2 points) Fix an undirected (weighted) graph $G = (V, E, c)$ where every edge e has cost $c_e \geq 0$. Let $T \subseteq E$ be a spanning tree of G with the guarantee that for every edge $e \in T$, e belongs to *some* MST of G.

   Prove or disprove the following statements:

   a) If the edge costs are all distinct, then T is an MST.

   b) T is an MST.