Quiz 1 Sample Questions
IE411: Networks and Graphs
Dr. Ralphs

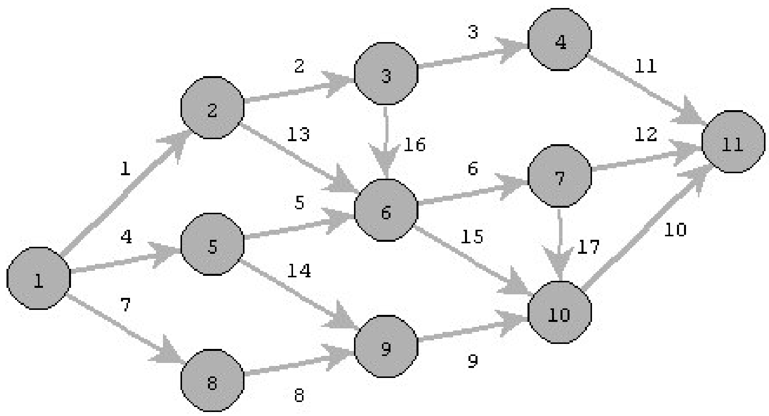

Figure 1: Network for Problem 1

1. Consider the network in Figure 1.
   (a) Write down the order in which the nodes are examined in a depth-first (LIFO) search of this graph if the adjacency list of each node lists adjacent nodes in ascending order by index.
   (b) Write down the order in which the nodes are examined in a breadth-first (FIFO) search of this graph if the adjacency list of each node lists the nodes in ascending order by index.
   (c) Execute the topological sort algorithm on this network. Either produce a topological sort or display a cycle in the graph.

2. Let a network $G = (N, A)$ be given with arc costs $c \in \mathbb{Z}^{|A|}$ and no negative cycles, along with labels $d(i)$ for every $i \in N$ satisfying the optimality conditions for the single-source shortest path problem with source node $s \in N$.

(a) Explain why every arc in a shortest path from source s must have reduced arc length zero with respect to the given labels.

(b) Using the result from part 2a, give an $O(m)$ algorithm for computing a shortest paths tree from s given the arc labels as input. Note that this means you cannot simply run a shortest path algorithm on the instance. You must use the fact that you know that optimal labels. Justify the correctness of your algorithm.

3. Let a network $G = (N, A)$ be given with arc costs $c \in \mathbb{Z}_+^{|A|}$ and source node s.

   (a) What is the *best-case* running time of Dijkstra's Algorithm implemented with heaps? What general property of a given network ensures this running time will be achieved?

   (b) Under what conditions will the *worst-case* running time be achieved for Dijkstra's Algorithm implemented with heaps? Give an example of a graph for which the worst case running time is achieved.

   (c) What are the best- and worst-case running times for Dijkstra's Algorithm with the naive $O(n^2)$ implementation. Under what conditions will these be achieved?

4. Let a directed acyclic graph $G = (N, A)$ be given along with a topological sort of the nodes in the graph.

   (a) Suppose you are given one additional arc to insert into the graph that may or may not be consistent with the given topological sort. You are asked to produce a new topological sort consistent with the original set of arcs and the new one. Write pseudocode for an efficient algorithm to either produce a new topological sort starting from the old one or determine that the new arc forms a cycle in the graph. Your algorithm should take fewer steps in general than an algorithm for computing a topological sort from scratch (though it may have the same worst-case running time in terms of "big-O notation").

   (b) What does the number of steps required for your algorithm depend on?