CMPS 2200 Introduction to Algorithms – Fall 15

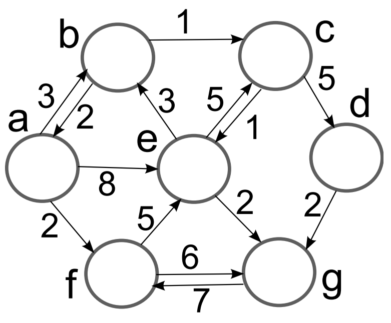
11/4/15

8. Homework

Due **11/12/15** at the beginning of the lab

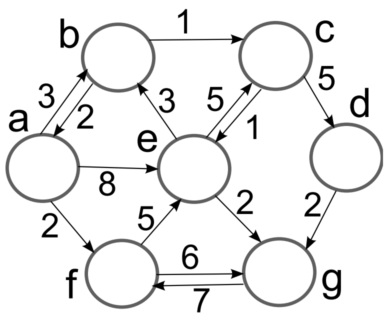
1. **Dijkstra (6 points)**
   Run Dijkstra's algorithm on the graph below, with source vertex a.

   (a) Show all the different stages of the algorithm, including d-values for each vertex, the set S, the priority queue, the vertex extracted from the priority queue, and the tree edges stored in the predecessor array. Also draw the shortest path tree edges into the graph.



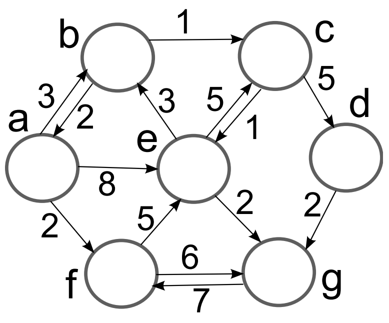
Q:

π: a b c d e f g



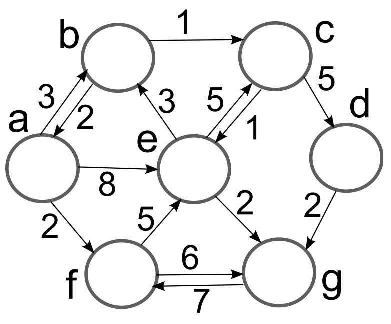
Q:

π: a b c d e f g



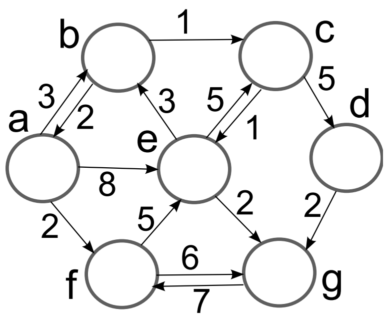
Q:

π: a b c d e f g



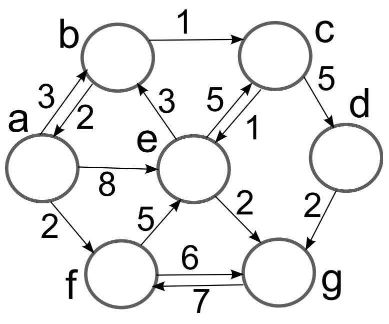
Q:

π: a b c d e f g



Q:

π: a b c d e f g



Q:

π: a b c d e f g

   (b) List the shortest paths from a to all other vertices.

Flip over to back page ⟹

2. **Dijkstra and negative edge weights (4 points)**
   Give an example of a directed connected graph with real edge weights (that may be negative) for which Dijkstra's algorithm produces incorrect answers. Justify your answer.

3. **Dijkstra variant (4 points)**
   Consider making the following change to Dijkstra's algorithm:

   ```
   while |Q|>1
   ```

   This means that the while loop runs until the queue consists of one element only. Argue why this change does not impact the correctness of Dijkstra's algorithm, i.e., the outputs of the original algorithm and the modified algorithm are the same.

4. **Bellman-Ford (4 points)**
   Let $G = (V, E)$ be a weighted, directed graph that possibly has negative weights but that has no negative weight cycles. And let $s \in V$ be a source vertex.

   Bellman-Ford's algorithm uses that shortest paths have at most $|V| - 1$ edges. But we want to make the runtime adaptive to the number of edges the shortest paths in G actually have. For any $v \in V$ define l_v to be the minimum number of edges in a shortest path from s to v. And define $k = \max_{v \in V} l_v$ to be the largest number of edges in a shortest path.

   Suggest a simple change to the Bellman-Ford algorithm that allows it to terminate in $k + 1$ passes. Give your answer in pseudo code. **Do not assume that k is known in advance.**

5. **Floyd-Warshall (6 points)**
   (a) (3 points) During the Floyd-Warshall all-pairs shortest paths algorithm, the shortest paths can be stored in a predecessor matrix. This is similar to storing a predecessor array for Dijkstras algorithm, just that there is such an array for every vertex. Modify Floyd-Warshalls algorithm to include the computation of the predecessor matrix. Give pseudo code as your solution. How much additional storage does this require?

   (b) (3 points) Describe how to use Floyd Warshalls algorithm to detect whether a weighted graph contains a negative weight cycle.