DSC 40B - Homework 08

Due: Wednesday, May 29

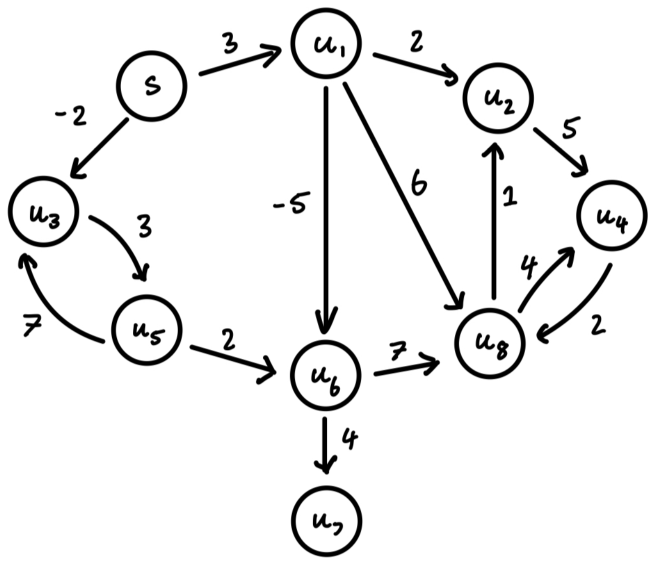
Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer. Homeworks are due via Gradescope at 11:59 p.m.

Problem 1.

Run Bellman-Ford on the following graph using node s as the source. Below each node u, write the shortest path length from s to u. Mark the predecessor of u by highlighting it or making a bold arrow. You can assume that `graph.edges` produces the graph's edges in the following order:

$$(u_3, u_5), (u_1, u_2), (u_5, u_6), (u_4, u_8), (u_6, u_7), (s, u_1), (u_5, u_3),$$

$$(u_1, u_8), (u_6, u_8), (u_8, u_4), (s, u_3), (u_2, u_4), (u_1, u_6), (u_8, u_2)$$

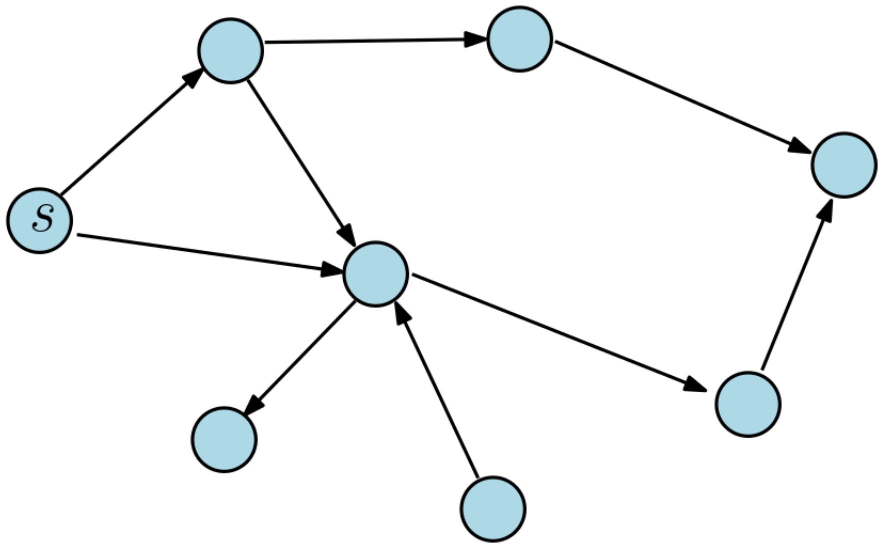


Problem 2.

Recall that the Bellman-Ford algorithm (with early stopping) will terminate early if, after updating every edge, no predecessors have changed. Suppose it is known that the greatest shortest path distance in a graph $G = (V, E)$ has $\Theta(\sqrt{V})$ edges. What is the worst case time complexity of Bellman-Ford when run on this graph? State your answer using Θ notation.

Problem 3.

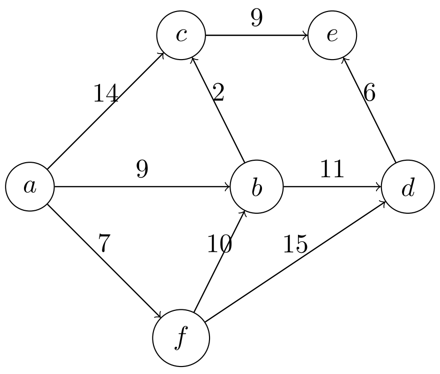
Suppose we are given a directed and weighted graph $G = (V, E)$ that look like the following figure. It is known that there are no negative cycles.



Suppose the Bellman-Ford algorithm **with early stopping** is run on this graph shown above using node s as the source (the edge weights are purposely not shown). In the worst case, how many iterations of the outer for-loop of Bellman-Ford will be performed? Briefly justify your answer.

Problem 4.

Run Dijkstra's Algorithm on the following graph using node a as the source. Below each node u, write the shortest path length from a to u. Mark the predecessor of u by highlighting it or making a bold arrow.



Problem 5.

True or False. Suppose $G = (V, E, \omega)$ is a weighted graph for which all edges are positive, except for those edges of a node s which may or may not be negative. If Dijkstra's algorithm is run on G with s as the source, the correct shortest paths will be found. Assume that the graph does not have any negative loops. Justify your answer.

Problem 6.

Suppose $G = (V, E, \omega)$ is a weighted graph without negative cycles. Suppose we have an edge $(u, v) \in E$ whose weight is $w(u, v) = 5$. Let $s \in V$ be the source node. Let $\delta(s, u)$ and $\delta(s, v)$ denote the shortest path distance from s to u and to v, respectively.

a) (True or False): Now suppose that G is a **directed** graph. Then, it is possible that $\delta(s, u) = 50$ but $\delta(s, v) = 10$.

b) (True or False): Now suppose that G is a **undirected** graph. Then, it is possible that $\delta(s, u) = 50$ but $\delta(s, v) = 10$.

Problem 7.

Consider the following scenario:

Lucas is hungry, and he wants to find a restaurant to eat at on campus. Unfortunately, all the restaurants on campus have a wait time.

Lucas has a weighted undirected graph $G = (V, E, \omega)$ of the campus, where notable locations are nodes, and the edge weights denotes time it takes to traverse that edge. Among the notable locations, some are restaurants. Each restaurant is marked with an associated wait time, indicating how long Lucas would need to wait for food *after arriving at the restaurant.*

Design an algorithm that **for all possible locations** of Lucas, $\forall v \in V$, computes the minimum time Lucas needs to spend before he can get food. Explain your algorithm in words and state its time complexity.

To receive full credit, your solutions must have optimal time complexity.

You may assume the graph is implemented using dictionary of sets.

Hint: Modify the graph or construct a new graph so that edge weights can include the information of wait times.