Homework 12: Due December 11 (11:59 p.m.)

Instructions

- **Answer each question on a separate page.**
- **Honors questions are optional. They will not count towards your grade in the course. However you are encouraged to submit your solutions to these problems to receive feedback on your attempts. Our estimation of the difficulty level of these problems is expressed through an indicative number of stars** $('*' =$ **easiest**$)$ **to** $('* * * * *' =$ **hardest**$)$**.**
- **You must enter the names of your collaborators or other sources as a response to Question** 0**. Do NOT leave this blank; if you worked on the homework entirely on your own, please write "None" here. Even though collaborations in groups of up to** 3 **people are encouraged, you are required to write your own solution.**

Question 0: List all your collaborators and sources: ($-\infty$ points if left blank)

Question 1: (10 points)

Fill in Table 1 to trace the execution of Dijkstra's algorithm on the example graph G (Figure 1) with A as the source vertex. The first row (Step 0) of Table 1 represents the initial values. You should show the updated distances to all the vertices (i.e., the key of the vertex in the priority queue) after each step in the algorithm.

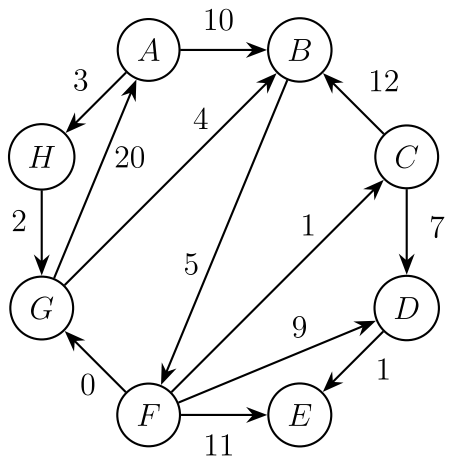

Figure 1: Directed and weighted graph G.

Step	Vertex Removed from Q	$A.d$	$B.d$	$C.d$	$D.d$	$E.d$	$F.d$	$G.d$	$H.d$
0	-	0	∞	∞	∞	∞	∞	∞	∞
1	A								
2									
3									
4									
5									
6									
7									
8									

Table 1: Execution trace of Dijkstra on G.

Question 2: (10 points)

Given a weighted DAG $G = (V, E)$ and a vertex $s \in V$, design an algorithm to find the lengths of the shortest paths from s to all other vertices in G in $O(|V| + |E|)$ time. Justify the correctness and running time of your algorithm.

Question 3: (2+6+2+10=20 points)

Sam, who lives in a vertex s of a weighted directed graph $G = (V, E)$ with non-negative edge weights, is invited to a birthday party located at $h \in V$. Sam wants to get from s to h as soon as possible, but he is told to buy some beer on the way over. He can get beer at any supermarket, and the supermarkets form a subset of the vertices $B \subseteq V$. Hence, starting at s, Sam must go to some vertex $b \in B$, and then head from b to h. Your goal is to design an algorithm to minimize the cost of Sam's trip.

1. Compute the shortest distance from s to all supermarkets $b \in B$.
2. Compute the shortest distance from every supermarket $b \in B$ to h. Note that we are now asking for the shortest path *from* every vertex *to* a particular target vertex.
3. Combine Parts 1 and 2 to solve the full problem.
4. A correct solution for Part 3 runs Dijkstra's algorithm twice (once in Part 1 and again in Part 2). We now ask for an alternate solution to the problem that uses Dijkstra only once. The key idea is to use reductions. Define a new graph G' on $2|V|$ vertices and at most $2|E| + |V|$ edges (and appropriate weights on these edges), so that the original problem can be solved using a *single* Dijkstra call on G'. Prove the correctness of your reduction.

Question 4: (5 points)

A negative weight cycle is a cycle where the sum of the weights of the edges in the cycle is negative. In many algorithms, these negative weight cycles can prove to be problematic, so we want a way to detect them ahead of time. How can we use the output of the Floyd-Warshall algorithm to detect the presence of a negative-weight cycle in a directed graph? (*Hint:* Say u is part of a negative cycle. What is the distance from u to u computed by the algorithm?)

Question 5: (5 points)

Consider a graph $G = (V, E)$ on n vertices *without* any negative weight cycles. We want to run the following (simplified) pseudocode of the Floyd-Warshall algorithm on G. Here, $W[i, j]$ is the weight of the edge $i \to j$ (or ∞ if the edge is absent in G). At the end of the execution of this algorithm, $D[i, j]$ contains the length of the shortest path from i to j along edges in G. First, convince yourself that this is indeed the case (no need to submit this part).

```
FLOYD-WARSHALL($W, n$)
1    for $i = 1$ to $n$
2        for $j = 1$ to $n$
3            $D[i, j] = W[i, j]$
4    for $k = 1$ to $n$
5        for $i = 1$ to $n$
6            for $j = 1$ to $n$
7                $cost = D[i, k] + D[k, j]$
8                if $cost < D[i, j]$
9                    $D[i, j] = cost$
```

Now, consider the modification where, instead of at line 4 above, the iterator for k is nested within the loop for the variable j, i.e.:

```
FLOYD-WARSHALL-MODIFIED($W, n$)
1    for $i = 1$ to $n$
2        for $j = 1$ to $n$
3            $D[i, j] = W[i, j]$
4    for $i = 1$ to $n$
5        for $j = 1$ to $n$
6            for $k = 1$ to $n$
7                $cost = D[i, k] + D[k, j]$
8                if $cost < D[i, j]$
9                    $D[i, j] = cost$
```

What does the value stored at $D[1, 2]$ at the end of the execution of this modified algorithm represent? Justify your answer.

Extra Practice Problems: (Problems are roughly at exam level)

These problems will *not* be graded, but are highly recommended.

Question 1

Suppose we have negative edges in G. If we run Dijkstra on G, what goes wrong? Think of what the shortest path will be. What can happen if there is a cycle in G where the total weight of edges in the cycle is negative. Can we even define shortest paths in this case?

Question 2

You are helping a group of ethnographers analyze some oral history data they have collected by interviewing members of a village to learn about the lives of people who have lived there over the past two hundred years.

From these interviews, they have learned about a set of n people (all of them now deceased), whom we denote $P_1, P_2, \ldots, P_n$. They have also collected facts about when these people lived relative to one another. Each fact has one of the following two forms:

- For some i and j, person P_i died before person P_j was born; or
- for some i and j, the life spans of P_i and P_j overlapped at least partially.

Naturally, they are not sure that all these facts are correct; memories are not so good, and a lot of this was passed down by word of mouth. So what they would like you to determine is whether the data they have collected is at least internally consistent, in the sense that there could have existed a set of people for which all the facts they have learned simultaneously hold.

Give an efficient algorithm to do this: Given the set of n people and a set of F facts, it should either produce proposed dates of birth and death for each of the n people so that all the facts hold true, or it should report (correctly) that no such dates can exist — that is, the facts collected by the ethnographers are not internally consistent.

(Hint: Try to represent the information as a graph where births and deaths are vertices. What feature in your graph would mean that the information cannot be consistent?)

Honors

Question 1

(*) Consider the problem defined Question 3. However, now, Sam must also pick up a pizza for their friend. You have another set of nodes $P \subseteq V$, which denote the set of pizza shops. Sam needs to pick up a pizza in addition to the beer. Sam can do this in any order, i.e., go from s to $p \in P$ and then to $b \in B$ before reaching h or go from s to $b \in B$ to $p \in P$ to h. We want to solve this problem by the technique of reduction where a new graph G' is created and a single Dijkstra call on G' is made to solve the problem of going from s to h, having picked up a cake and beer, traversing the shortest distance possible.

Question 2

(*) How would your answer to previous problem change if Sam had to first pick up the beer before picking up the pizza, to make sure the pizza is hot and fresh?