

Weekly Challenge 12: K-edge Shortest Paths in Directed Graphs with Negative Weights

CS/CE 412/471 Algorithms: Design and Analysis

Spring 2025

Total points: 30

Objective

In this WC, you will work in groups and:

- understand the basic algorithm/component/building block for various shortest path algorithms.
- demonstrate step-by-step working of the building block with examples and hand-drawn illustrations.
- implement the algorithm and verify correctness of your hand-crafted solutions.

The Problem

Ashraf Courier Service is responsible for delivering urgent messages between cities. The king has decreed that messages should reach their destination as quickly as possible, but couriers can take at most two roads to complete their journey.

Some cities are directly connected by a road, while others require passing through an intermediate city. The time taken to travel along each road varies, and couriers must always choose the fastest possible route.

Your task is to help the Ashraf Courier Service determine the shortest possible delivery time between a source city and all other cities while ensuring that no courier ever takes more than two roads in a single trip.

Task 1 (10 points)

For the graph shown in Figure 1, dry run Algorithm 1 shown in Figure 2 which finds at most 2-edge shortest distances from a source vertex a to all destination vertices k through intermediate vertices j . The source vertex would be S . You just need to submit **the final graph obtained after completing execution of the algorithm** as Figure 4. Please make sure the positioning of the vertices remains the same, i.e., do not morph the graph, just modify weights or add new edges.

Task 2 (10 points)

Implement Algorithm shown in Figure 2 and **visualize the resulting graph** using any graph visualization library. Include this image as Figure 5. You may use any programming language of your choice. Additionally, **mention in the caption** of Figure 5 what the output graph will illustrate if we run Algorithm 1 on this resulting graph one more time.

Task 3 (10 points)

Run Algorithm 1 for $p-1$ times, where p is the total number of vertices in the graph (i.e., $p = 10$), and **visualize the resulting graph** using any graph visualization library. Include this image as Figure 6. Additionally, **mention in the caption** of Figure 6 what Algorithm 2 finds out.

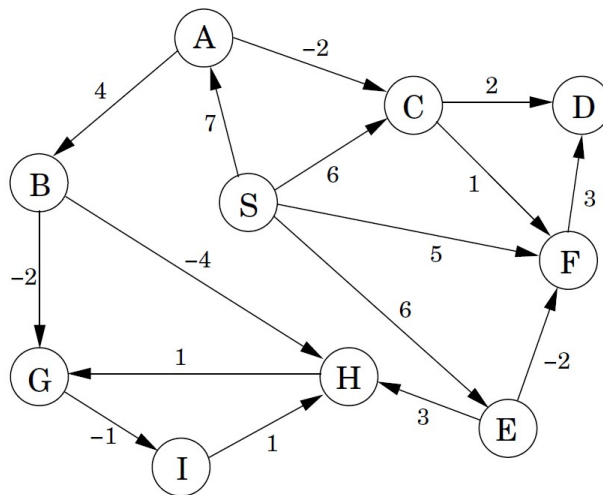


Figure 1: The sample input graph

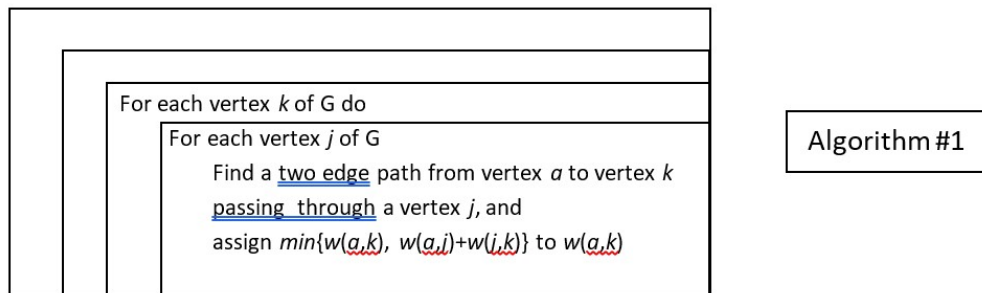


Figure 2: Algorithm 1 finds at-most 2-edge distances from a source vertex \mathbf{a} to all destinations \mathbf{k} through all intermediate nodes \mathbf{j} . $w(\mathbf{a}, \mathbf{k})$ refers to the weight of the edge $\mathbf{a} \rightarrow \mathbf{k}$.

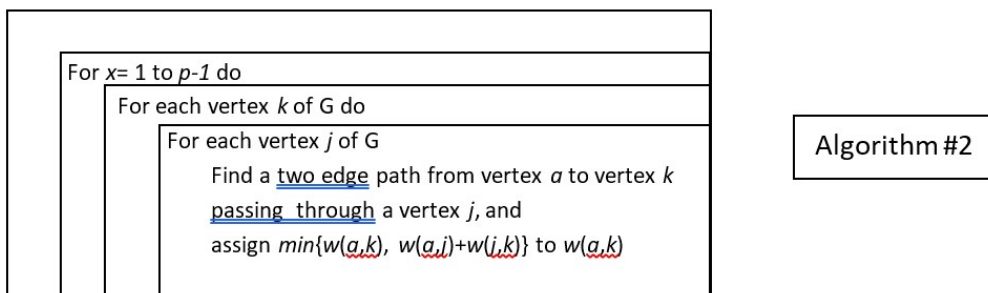


Figure 3: Algorithm 2 is designed by repeating Algorithm 1 (Figure 2) $p-1$ times.

Solution - Task 1 (Figure 4)

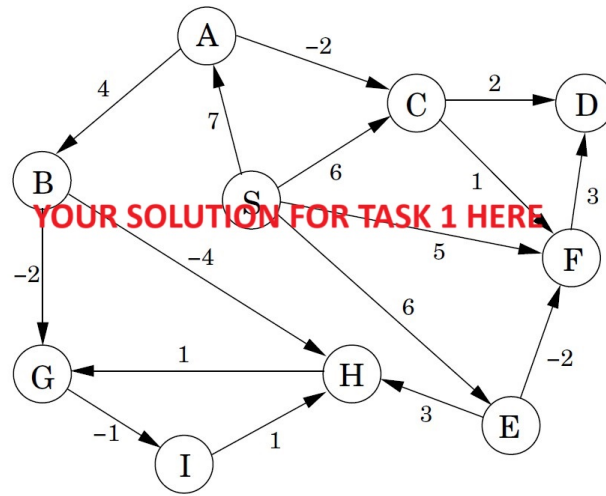


Figure 4: Modified graph showing the at-most 2-edge shortest distances from source vertex S to all other vertices for the graph in Figure 1.

Solution - Task 2 (Figure 5)

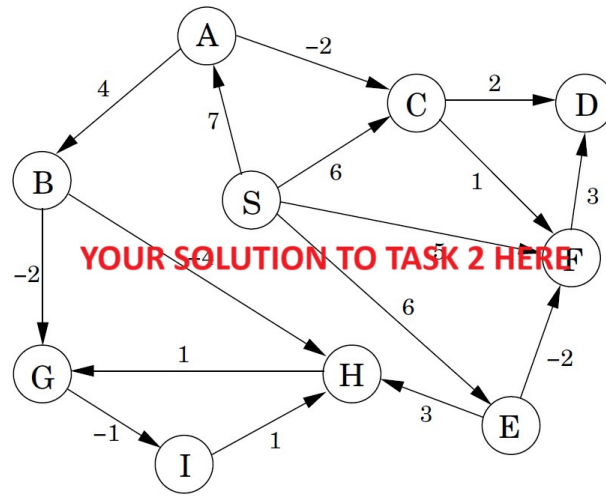


Figure 5: Code-generated graph showing the at-most 2-edge shortest distances from source vertex S to all other vertices for the graph in Figure 1.

Solution - Task 3 (Figure 6)

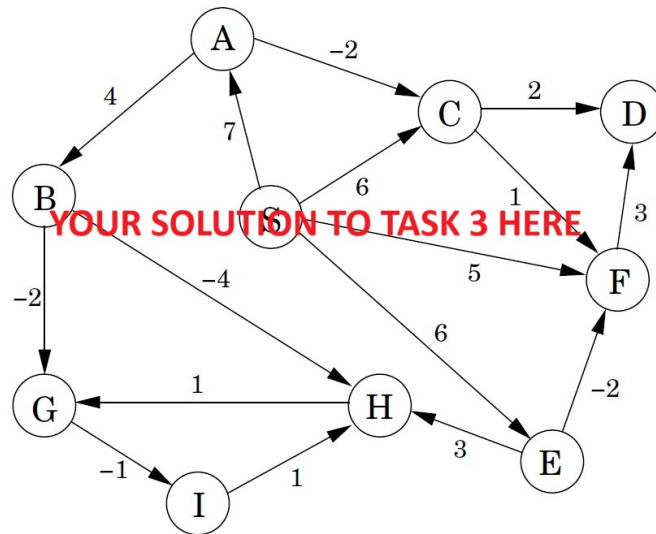


Figure 6: Code-generated graph showing shortest distances from source vertex S to all other vertices for the graph in Figure 1.

Submission

1. Submit one PDF file including your solutions.
2. Submit one source file including your code.
3. Clearly write your group number, member names, and IDs.