

Problem type 1:

Answer the problem below:

(See variants below)

P.S. You may **not** use lab solutions as a black box.

a. **BYA**

For what types of graphs does the Bellman-Ford algorithm have roughly the same runtime as the Floyd-Warshall Algorithm.

b. **BYE**

You are given a graph with all *positive* edge weights and want to find the shortest distance between all pairs of nodes. Give an algorithm (as fast as possible) that would give you this information.

c. **BYB**

The Floyd-Warshall algorithm is described by the recurrence:

$$dist(i, j, k) = \min \begin{cases} dist(i, j, k-1) \\ dist(i, k, k-1) + dist(k, j, k-1) \end{cases}$$

In a english sentence, state what the value $dist(i, j, k)$ represents.

d. **BYH**

The Bellman-Ford algorithm is described by the recurrence:

$$d(v, k) = \min \begin{cases} \min_{u \in V} (d(u, k-1) + \ell(u, v)) \\ d(v, k-1) \end{cases}$$

In a english sentence, state what the value $d(v, k)$ represents.

e. **BYG**

Prof. Nani had a great idea to solve the longest path problem (given a graph G , find the longest path between any two vertices). He would simply make all the edges negative and run the Floyd-Warshall algorithm. Why does this not work?

f. **BYC**

Prof. Nani had a great idea to solve the shortest path problem using Djikstra's algorithm. He would simply modify the graph by finding the smallest (most-negative) edge and adding the magnitude of that edge to all the edges weights in the graph. This ensure that all the edge weights are positive which is all that's required for Djikstra's algorithm right? Why does this not work?

g. BYF

Suppose you are given a directed, weighted graph with positive and negative edge weights and you want to see if there is a negative edge cycle present in the graph. Describe an efficient algorithm that returns true if there is a negative cycle.

h. BYD

You are given a directed graph with unweighted edges. Some of the edges are marked as risky and **you can't take more than one** risky edge. Provide an algorithm (as fast as possible) for the shortest path between s and t .