Problem 12.2 Shortest Path Problem

Algorithm:

1. Add large constant to each edge weight such that all weights become positive.
2. Run Dijkstra's algorithm for shortest path

From a theoretical perspective, a directed graph represents states and transitions where there is a cost associated with each transition (weight).

In Dijkstra's algorithm, once a vertex is marked closed (out of the open set) → implies the algorithm has found the shortest path to it.

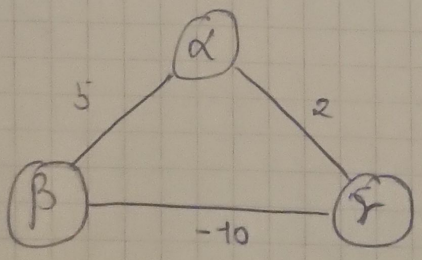
↳ Once we introduce negative weights, may no longer be true.

In every relaxation step of Dijkstra, the algorithm assumes that the cost to the closed nodes is in fact minimal.

↳ Regardless of how large a value we add to it, the minimality will never change.

↳ HENCE, it is necessary to have non-negative weights for applying Dijkstra

E.g. Suppose we have the following graph



→ each vertex is developed only once
↳ not re-visited

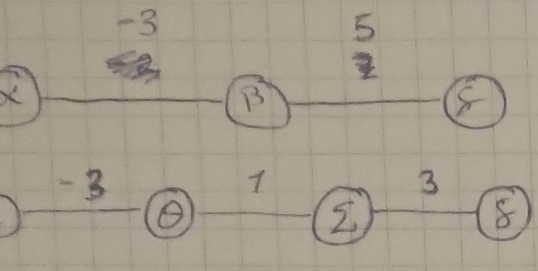
→ starting at α, will first develop γ and fail to find the path α, β, γ

→ The algorithm is centered around the fact that once a node is developed, it will not be modified because the minimality will not be affected by adding a positive value.

↳ If negative weights, minimality will change.

→ Now if we talk about the first step (adding a large constant)

This is best explained through an example.

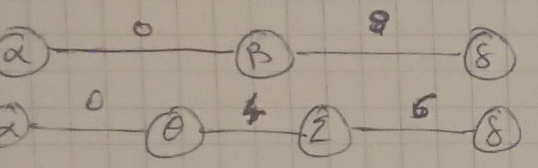


path $(\alpha, \delta) = 2$

path $(\alpha, \delta) = 1$

} Choose the 2nd one because min

Now lets add a constant large enough so we have non-negative weights.



path $(\alpha, \delta) = 8$

path $(\alpha, \delta) = 10$

} we end up with different result

→ This happens because the second path has more nodes to reach from α to δ

↳ recall we add a large constant to all EDGES

↳ since number of edges in the first path is less, that is chosen as the shortest path.

↳ We get a different outcome which is incorrect as can be seen from the example.