Please refer to Home-Work template uploaded on the course web-site [link]

**Problem 3** Give an efficient algorithm which takes as input a directed graph \( G = (V; E) \), and determines whether or not there is a vertex \( s \in V \) from which all other vertices are reachable.

Find source node (or meta node). Why source node?? How (Anything to do with post value)??

Check if other nodes are reachable from the source node. 'Reachable from' immediately suggests that we apply DFS, where explore(v) finds all vertices reachable from vertex v.

**Problem 4** You are given a directed graph in which each node \( u \in V \) has an associated price \( p_u \) which is a positive integer. Define the array cost as follows:

for each \( u \in V \),

\[
\text{cost}[u] = \text{price of the cheapest node reachable from } u \text{ (including } u \text{ itself)}
\]

For instance, in the graph below (with prices shown for each vertex), the cost values of the nodes A; B; C; D; E; F are 2; 1; 4; 1; 4; 5, respectively. Your goal is to design an algorithm that fills in the entire cost array (i.e., for all vertices)

![Graph with prices](image)

Figure 1:
part a. Give a linear-time algorithm that works for directed acyclic graphs. (Hint: Handle the vertices in a particular order.)

As you saw in Lecture 1, try to first understand the problem using small examples. Here’s a small example.

Example Graph 1 → 2 → 3, 2 → 4.

Linearize the DAG, 1 - 2 - 3 - 4.

Start with 4, cost of 4 will remain same?

Next go to 3, cost of 3 will again be same (No edge from node 3 to node 4).

Next node is 2, updated cost of 2 will be minimum of (cost(2), updated cost(3), updated cost(4)).? (Node 1 is not involved?)

Next node is 1, cost of 1 will be minimum of (cost(1), updated cost of (2)). (No need to care about cost of 3 or 4 now. Why?)

Linearize the DAG, go through the example and figure out why??

Start from the sink node and keep updating the costs.

Complexity is linear, how??

part b. Extend this to a linear-time algorithm that works for all directed graphs. (Hint: Recall the two-tiered structure of directed graphs.)

Why does the problem separately address the DAG case and the general undirected graph case? What will the cost values look like in a given SCC of a digraph? Notice in SCC all nodes are reachable from each other!

Linearize the DAG of meta-nodes, similar to part (a) now?