CS 101 Fall 2021 - Quiz 8-A

12/06/2021 - 25 Minutes **Name:** **ID number:**

Problem 1 True or False (5×1 pts)

The following questions are True or False questions, you should judge whether each statement is true or false.

Note: You should write down your answers in the box below.

Problem 2.1	Problem 2.2	Problem 2.3	Problem 2.4	Problem 2.5
F	F	T	T	F

(1) A DAG has its unique topological sorting if it has exactly one source.

(2) A topological sorting of graph G is $(v_1, v_2, \cdots, v_i, \cdots, v_j, \cdots, v_{|V|})$, then there exists a path from v_i to v_j in G.

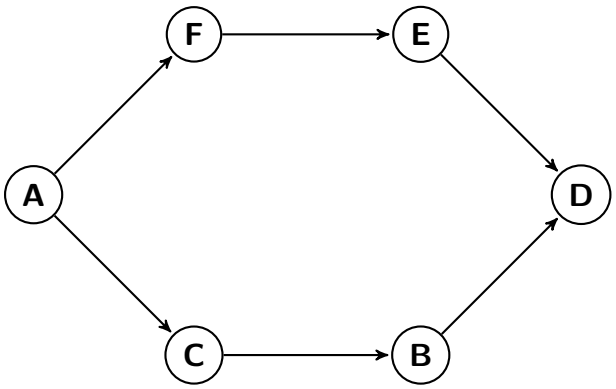
(3) If we add a constraint that each vertex can only appear at most once in the shortest path, Dijkstra's algorithm still works for positive-weighted graphs.

(4) The run time of Dijkstra's algorithm with binary heap implementation is $O((|V| + |E|) \log |V|)$.

(5) Bellman-Ford algorithm works for any graph without negative cycle while Dijkstra's algorithm only works for any acyclic graph.

Problem 2 Topological Sort (2 + 2 pts)

Given the DAG below:



(1) Run topological sort on the given DAG and write down the topological sorting you obtain.
Note: When pushing several vertices into a queue at the same time, push them alphabetically. You are NOT required to show your steps.

ACFBED

(2) How many different topological sortings does the DAG have? Write them down.

6 : ACBFED, ACFBED, ACFEBD, AFCBED, AFCEBD, AFECBD

CS 101 Fall 2021 - Quiz 8-A

12/06/2021 - 25 Minutes **Name:** **ID number:**

Problem 3 Does Shortest Path Change? (3 pts)

Given a shortest path $P = (s, v_1, v_2, \cdots, t)$ from s to t in graph $G = (V, E)$. Now Ge Ziwang changes the weight of each edge in G to twice the orginal weight i.e. $w(e') = 2 \times w(e)$. By doing this, Ge Ziwang obtains a new graph $G' = (V, E')$. Is the original shortest path P still guaranteed to be a shortest path from s to t in G'?

If yes, briefly explain why; If not, give a counterexample.

Yes. Changing to the costs of all edges as twice as before just makes the total costs of all paths as twice as before. This will not affect the original relative relationship between the costs of all paths. (Or it is equivalent to construct a new graph G'' where for each edge (u, v) of weight w in the original graph G, create a new vertex e and add two new edges (u, e) and (e, v) both of weight w. Finding the shortest path P' in G' is equivalent to find P'' in G'', which can be converted to P in G by combining u, e, v in P'' to u, v in P.)

Problem 4 Dijkstra's Algorithm Tiebreak (5 pts)

Given a directed weighted graph G, we want to find the shortest path from s to t, however, among all shortest paths, we want to choose the one that has the fewest number of edges. How would you modify Dijkstra's algorithm in this setting? Briefly write down your main idea. Assume weights are all positive and different from each other. If multiple paths have the same number of edges and the same total weight, choose any of them.

Hint: Consider how to modify the relax/update step of Dijkstra's algorithm.

Solution 1: Maintain an array $num[v]$ to keep track of the number of edges on the current shortest path to v. Initially $num[s] \leftarrow 0$ for the source vertex s and $num[v] \leftarrow \infty$ for all rest $v \in V$. When we relax an edge (u, v) of weight w, if $dis[u] + w < dis[v]$, we update $num[v] \leftarrow num[u] + 1$ besides the original implementation; If $dis[v] = dis[u] + w$ but $num[v] > num[u] + 1$, update the predecessor of v to u to record the shortest path and $num[v] \leftarrow num[u] + 1$.

```
1: function ModifiedRelax(u, v, w)
2:     if dis[v] > dis[u] + w then
3:         dis[v] ← dis[u] + w
4:         num[v] ← num[u] + 1
5:         pre[v] ← u
6:     else if dis[v] = dis[u] + w and num[v] > num[u] + 1 then
7:         num[v] ← num[u] + 1
8:         pre[v] ← u
9:     end if
10: end function
```

Solution 2: Add a very very small ϵ to the weight of each vertex to obtain a new graph G'. Run Dijkstra's algorithm on G' from s to t to obtain the shortest path. Notice this ϵ must be less than $\delta/|E|$, where δ is the minimum difference of weights of the edges.