Assignment 1

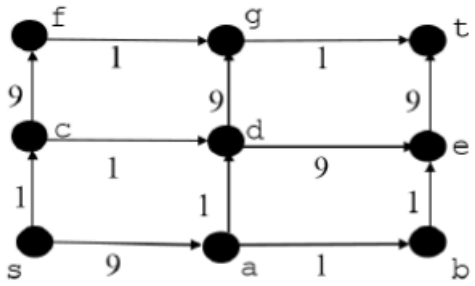
Harsh Kabra - EE18BTECH11019

Github repository

https://github.com/harsh006/C−DS/tree/main

1 Problem

In a directed acyclic graph with source vertex s, the quality-score of a directed path is defined to be the product of the weights of the edges on the path. Further, for a vertex v other than s, the quality-score of v is defined to be the maximum among the quality-scores of all the paths from s to v. The quality-score of s is assumed to be 1.



Then the sum of quality-scores of all vertices in the graph shown above is?

2 Solution

2.1 *About the problem*

- So, we are given a Directed Acyclic Graph (DAG), as shown in the above figure so we are supposed to find maximum of quality-scores for each node from a given source S.
- Where each quality-score is found by multiplying weight of edge from one node to other node.
- As the above problem, we have a single source and have to find Longest Path
- Synonymous to longest path, here instead of addition of weights,we have multiplication and all positive values
- We, will hence use **Bellman-Ford algorithm** to solve this problem.

2.2 *About Bellman-Ford algorithm*

- The algorithm calculates longest paths in a bottom-up manner.That is,
- It first calculates the shortest distances which have at-most one edge from source to our current node in the path.
- Then, it calculates the shortest paths with at-most 2 edges, and so on.
- After the i-th iteration, the shortest paths with at most i edges in path are calculated. As,evident this to similar to BFS style of traversal.
- There can be maximum V–1 edges in any simple path, that is why the outer loop runs v – 1 times. Where, V = number of nodes.

The exact steps are -

- Initialize distances from the source to all vertices as negative infinity(lowest possible value) and distance to the source itself as

---

$source \leftarrow 0$
$cost \leftarrow$[-INF]*N
$cost[source] \leftarrow 1$

- This step calculates longest distances.

$it \leftarrow 1$
while $it \neq V - 1$ **do**
  for
    einAdj[it] **do**
    $u \leftarrow destnode$
    $w \leftarrow weight$
    if $cost[v] * w$ ¿ $cost[u]$ **then**
      $cost[u] = cost[v] * w$
    end if
  end for
  $it \leftarrow it + 1$
end while

– Looping through V-1 time

  - Inside the above loop, again Looping through each edge at each iteration and checking if the cost is higher any time.
- Analyzing time complexity:

$$T(n) = O(n * (n-1)) \implies O(n^2) \qquad (2.2.1)$$

*2.3 **Solution***

The following are the values we get after applying the above algorithm:

- dist (s, s) = 1
- dist (s, a) = 9
- dist (s, b) = 9
- dist (s, c) = 1
- dist (s, d) = 9
- dist (s, e) = 81
- dist (s, f) = 9
- dist (s, g) = 81
- dist (s, h) = 729

Therefore, finally sum of all the values = **929**
Code can be found in:

https://github.com/harsh006/C−DS/tree/main/code