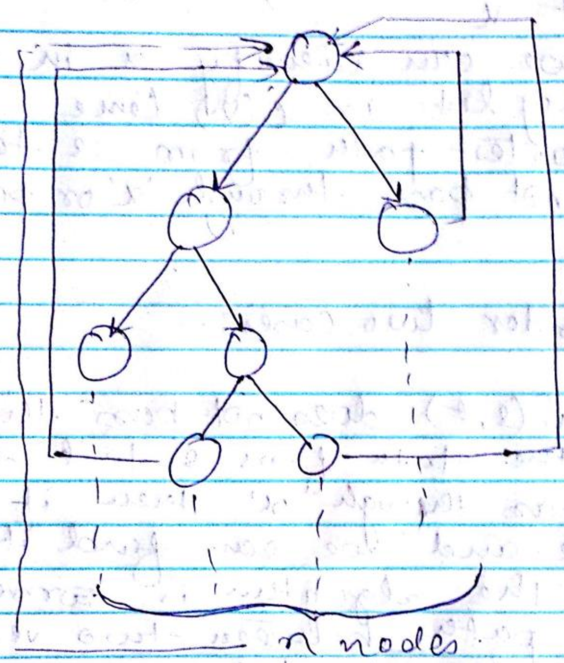
ASSIGNMENT 8

Chitrarth Singh
A20381080

HOMEWORK ASSIGNMENT 8

1.

(a)



Solution:- Dijkstra runs in $O(E \log V)$ time.
In a looped tree, each vertex has at most two outgoing edges so the number of edges in the graph is at most 2V.
Therefore we get $E = O(V)$ and the algorithm will run in $O(V \log V)$ time

ASSIGNMENT 8

(b) Describe and Analyze faster algorithm

Solution :- There's exactly one vertex in the graph with in-degree ~~> 1~~ > 1. We call it x
Now we can identify x in the adjacency list in $O(V)$ time
The shortest path from s to t will either not pass through 'x' or pass through 'x'

Lets consider two cases:-

1. When $\delta(s,t)$ does not pass through
If the path from s to t does not pass through 'x' then it is unique. and we can find it in $O(V)$ time. The algorithm is correct as a unique path between two vertices must be the shortest one.

2. $\delta(s,t) = \delta(s,x) + \delta(x,t)$. → This gives us the shortest path from s to t because $\delta(s,t)$ must go through 'x' and subpaths of shortest paths are shortest paths. We already ~~know~~ know how to find $\delta(x,t)$ in $O(V)$ so we need to find only ~~$\delta(s,x)$~~ $\delta(s,x)$. We can compute $\delta(s,x)$ for node s from the values of its children in the tree in DP fashion, bottom up starting from x

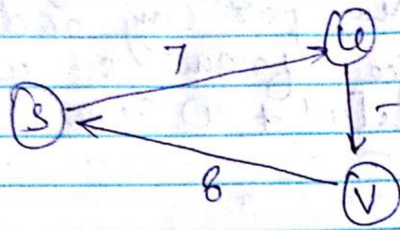
ASSIGNMENT 8

So Total running time of the algorithm is
$O(V+E) = O(V)$

ASSIGNMENT 8

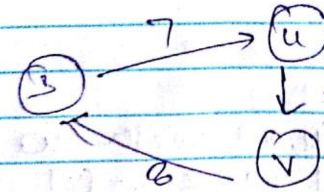
2.

(a).



The path from (s) to (v) is 8 ~~as~~ ~~both~~ when (u) is added s → u is 7.
The path s → u → v will ~~be~~ have a lesser cost i.e. 7 − 6 = 1. Since Dijkstra has already traversed through v it will fail to update the lesser cost. Hence ~~the~~ Dijkstra Algorithm fails or give incorrect result when ~~ever~~ negative weight is present.

Modified version of Dijkstra to deal with Negative weight edge.



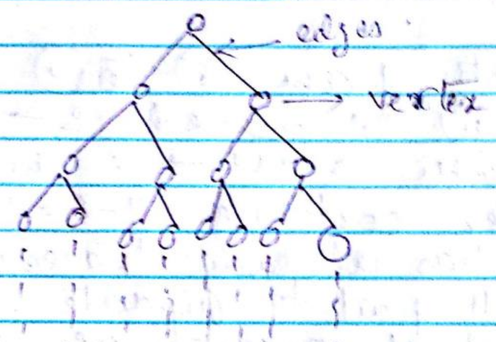
(u, v) is a negative weight edge.

We first remove the negative edge and run Dijkstra from s.
Check if $d_s[u] + w(u,v) < d_s[v]$.
If false then do nothing.
If True then run Dijkstra from v,

ASSIGNMENT 8

with negative weight edge still removed. Then for any node t, its shortest distance from s will be $\min(d_s[t],\ d_s[u] + w(u,v) + d_v[t])$.

(b)



Infinite family of graph

Construction :-

- For each vertex v in V
- If v is the source then $w[V] = 0$
- Else $w[V] = \infty$
- $\pi[V]$ = null

RELAXATION

- For $i = 1$ to $v - 1$
- For each edge (u, v) with w in edges
- If $w[V] + w < w[V]$
- $w[v] = w[u] + w$.
- $\pi[V] = u$.

NEGATIVE EDGE

- For each edge (u, v) with weight w in edges
- If $d_s[u] + w(u,v) < d_s[v]$
- Run Dijkstra from v.
- $d_s[t] = \min(d_s[t],\ d_s[u] + w(u,v) + d_v[t])$

ASSIGNMENT 8

3. FLOYD-WARSHALL ALGORITHM

Solutions: modified version of the algorithm that correctly returns shortest path distance with negative cycles.

Input: A digraph G with $V(G) = \{1 ... n\}$ & weights $c: E(G) \rightarrow R$

Output:- An $n \times n$ matrix M such that $M[i, j]$ contains the length of a shortest path from vertex i to vertex j.

```
1. M[i, j] = ∞ ∀ i ≠ j.
2. M[i, j] = 0 ∀ i.
3. M[i, j] = c((i, j)) ∀ (i, j) ∈ E(G).
4. for i = 1 to n do
5.     for j = 1 to n do
6.         for K = 1 to n do
7.             if M[j, K] > M[j, i] + M[i, K] then
8.                 M[j, K] = M[j, i] + M[i, K]
9. for i = 1 to n do
       if M[i, i] < 0 then return (Graph contains a negative cycle) - ∞
```

The above algorithm runs correctly & finds the length of the shortest directed path for non negative pairs of vertices.
If there's a negative cycle the algorithm outputs $-\infty$ as the length of the shortest path from u to V.

ASSIGNMENT 8

REFERENCES

https://courses.csail.mit.edu/6.006/oldquizzes/solutions/q2-f2010-sol.pdf

http://www.bowdoin.edu/~ltoma/teaching/cs231/fall09/Homeworks/old/H9-sol.pdf

https://dl2.pushbulletusercontent.com/6ljsJbULTENqPT1Kdu2dhw2RqUwMaxFW/0601e865fcfd191fe49a1467ad13ab35a39d.pdf

http://people.math.gatech.edu/~randall/Algs06/HW3_solns.pdf