4/8/19

Homework 4, Part 2.

1. Given a weighted, directed graph $G = (V, E)$ with no negative-weight cycles

* all vertices $v \in V$ of the minimum numbers of edges in shortest path from the source s to v

Soln: If the greatest number of edges from source m on any shortest path then from the path relaxation property, we get that after m iterations of BELLMAN-FORD algorithm, every vertex V will achieve its shortest path weight d. (i.e., for every V, ∃ a wright d in v.d, such that, the vertex achieves shortest path weight) after m iterations, there is no change in d values (upper bound property).

Implying from the above statement no d values will change for $m+1^{th}$ iteration. Since, we were not given m in advance, the algorithm cannot iterate exactly m times. But we can stop it when no substantial changes occur in the weights 'd'. The program will now stop at m+1 iterations.

Scanned by CamScanner

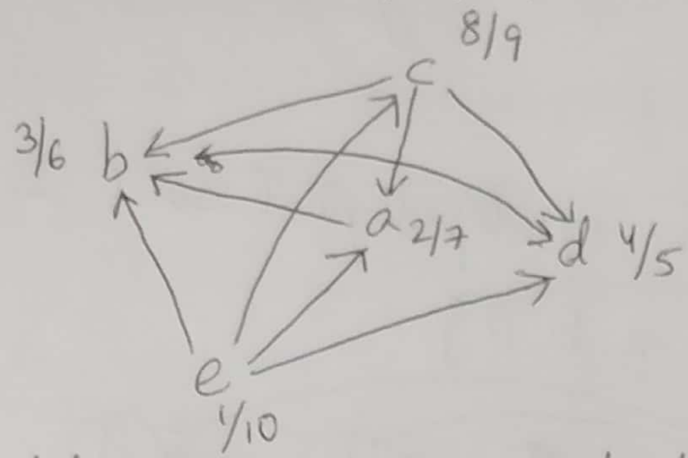
BELLMAN-FORD (G, w, s)

```
{
  Initialize Single Source(G, s)
  diff = True
  while diff == True
      diff = false
    for each edge (u, v) ∈ G.E
        Relax(u, v, w)
}

Relax (u, v, w)
{
  if v.d > u.d + w(u, v)
      v.d = u.d + w(u, v)
      v.π = u
      diff = True
}
```

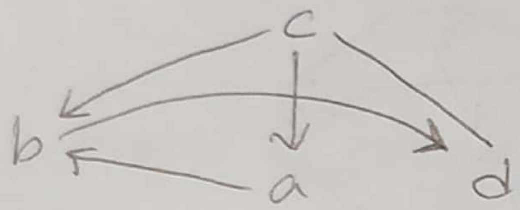
Basing on the d value, which changes on another relaxation step, the test for negative weight cycle has been removed. Because this algorithm doesn't break out of loop until and all d-values stop changing.

2. (a) Topological sort of graph using DFS.



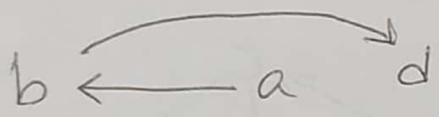
Step 1: Select source 'e' as it has all outgoing edges.

Step 2: After excluding 'e', we get the graph as.



So, next select the source 'c'

Step 3: After excluding 'c', we get the graph as



So, next select the source 'a'

Step 4: After excluding 'a', we get the graph as,

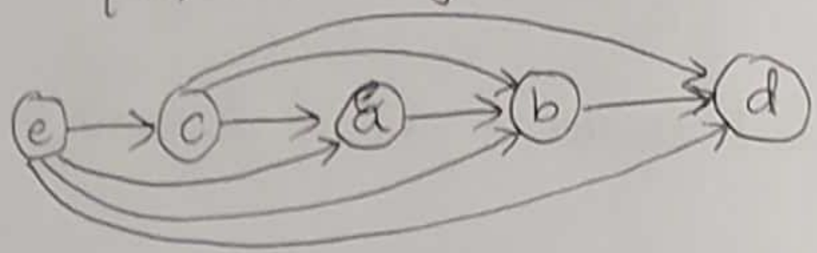
So, next select source 'b'

~~Step 5~~:

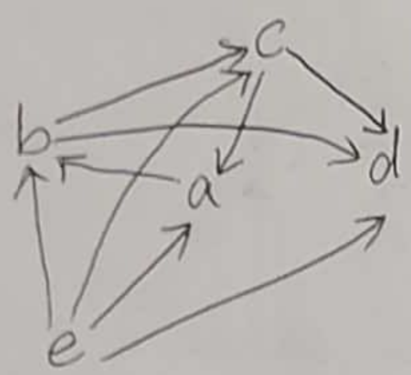
Step 5: After excluding 'b', only source left is 'd'.

So, the order of topological sort is,

{e, c, a, b, d}



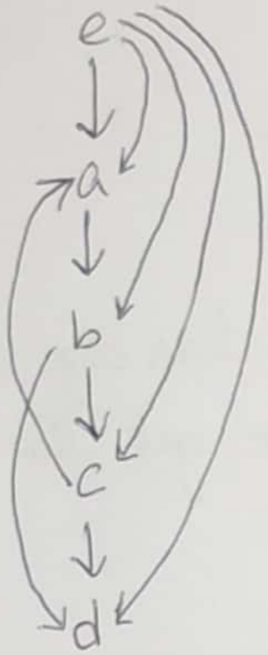
(b) Reverse the edge (b, c)



Step 1: Select source 'e'. Exclude it from graph.

Since there is no other edge, it is not a DAG.

Run DFS on graph, we get,



① Tree-edges.

(e, a), (a, b), (b, c), (c, d)

② Forward edges

(e, b), (e, c), (e, d), (b, d)

③ Back edges

(c, a)

④ Cross edges

No cross edge.

③ Given inequalities are:

$-1 \leq x - z \leq 0$, $0 \leq y - z \leq 1$,

$-3 \leq x - y \leq -2$

Solving the inequalities, we get

$x = -2$, $y = 0$, $z = -1$.

Thus the system is feasible.

Therefore using the difference constraints, a constraint graph is drawn.

$-1 \leq x - z \leq 0 \Rightarrow \quad x - z \leq 0$

$z - x \leq 1$

$$0 \le y - z \le 1 \Rightarrow \begin{aligned} y - z &\le 1 \\ z - y &\le 0 \end{aligned}$$

$$-3 \le x - y \le -2 \Rightarrow \begin{aligned} x - y &\le -2 \\ y - x &\le 3 \end{aligned}$$

∴ Consider a source v_0, which has edge t all vertices x, y and z. Thus, we get the graph as follows.