Module 8

- SHORTEST PATH ALGORITHM

Terms

1] Path : $P(U \rightarrow V)$

- Sequence of vertices where each adjacent pair is connected by edge.
- In weighted graph each edge is associated with some weight.
- can be directed or undirected.

2] Cycle :

- Starts and end vertex are same.
- It is special path

3] Shortest path : $SP(U,V)$ or $\delta(U,V)$

- Path with minimum total weight among all possible paths.

4] Negative weight :

- Value assign to edge that is less than 0.
- Used to represent costs, penalties, etc.

5] Negative Weight Cycle :

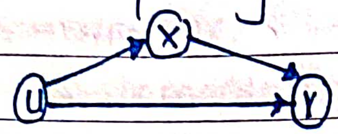
- Cycle with sum of the weights of its constituent edges is negative.

Notations :

1. Weight $= \omega(U,V)$
2. Path $= P(U \rightarrow V)$
3. Weight of P $= W(P(U \rightarrow V))$
4. SP $= SP(U,V)$
5. Weight of SP $= \delta(U,V)$

Properties :

1] Triangle Inequality -



$$d(u,v) \leq d(u,x) + d(x,v)$$

2] Upper Bound -

$$v.d \geq d(s,v)$$

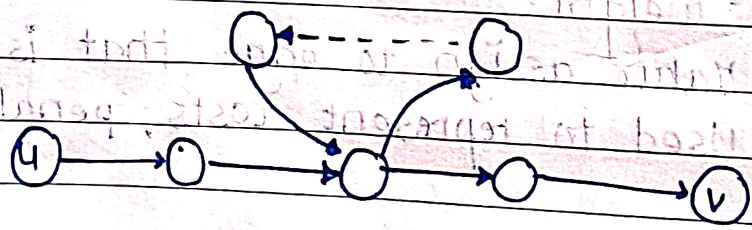
3] No path -

If there is no path from s to v

then $v.d = \delta(s,v) = \infty$

4] Complexities involved :

1) Edge can be negative
2) Negative weight cycle.



Shortest Path variants :

- SSSP [Single Source Shortest Path]
- SDSP [Single destination —"—]
- SPSP [Single Pair —"—]
- APSP [All pair —"—]

- Shortest Path betⁿ U and V is min distance or weight from U and V vertex.
- defined as,

$$\delta(U,V) = \begin{cases} \min\{W(P) : U \sim V\} \\ \infty \quad \text{otherwise} \end{cases}$$

For graph $G = (V, E)$
S is starting Vertex

→ $V.\pi$ - Predecessor vertex
→ $V.d = \infty$ (Initially ∞)
↳ shortest path estimated from s to V

→ Initialization
$V.d = \infty$
$V.\pi$ = NIL or NULL
$s.d = 0$

→ Relaxing the edge :
Testing whether we can improve the SP from s to V, found so far by going through U.
→ if so
update $V.d$ & $V.\pi$

Relaxation : Trying to improve SP.

Relaxation :
Trying for improvement of SP from s to V.

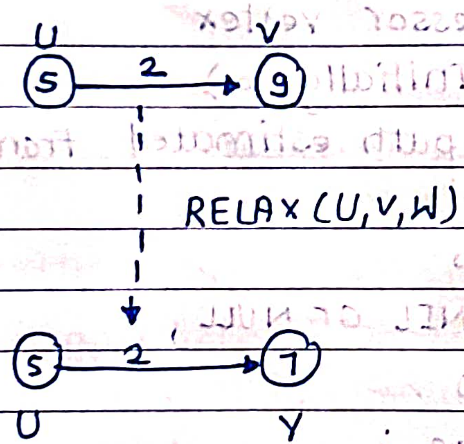
Conditions

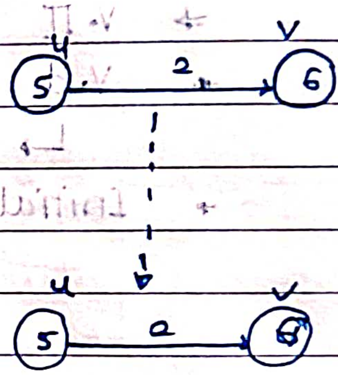
Relax (U, V, W) — U = Vertex 1
V = Vertex 2
W = weight

i) → if ($V.d > U.d + W(U,V)$)
{
$V.d = U.d + W(U,V)$
$V.\pi = U$ }

* v.d → distance estimated from s to v
* u.d → —"— s to u
* v.π → start vertex.

description:
- Relaxation is commonly used to find SP
- → It start with initial estimates for a vertices
- → during process, these estimates are updated iteratively to find better estimates.
- → Goal is to get optimal value.

i]


ii]


if (9 > 5 + 2)
then
v.d = 5 + 2
v.π = u

∴ v.d = 7
v.π = u

if (6 > 5+2)
then
v.d = 7
v.π = u

No modification as condition not satisfied.

Bellman Ford Algorithm : [SSPS]

- Computes shortest path from single source vertex to all of the vertices in a weighted digraph.
- doesn't work for graph with Negative ~~edge~~ weight cycle.

Steps :

1) Initialization

$v.\pi = NIL$

$v.d = \infty$

$s.d = 0$

2) Relax the edges repeatedly

For (# vertices - 1)

- For all edges Relax — (Relaxation)

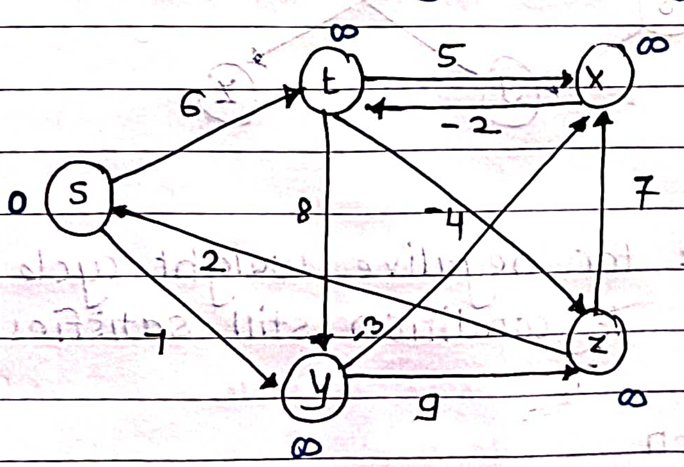
3) Check for negative weight cycle.

if you found

$v.d > u.d + w(u,v)$

then contains negative weight cycle.

Example



1] $s.d = 0$

$\left.\begin{array}{l} t.d \\ x.d \\ z.d \\ y.d \end{array}\right\} = \infty$

Using. $v.d > u.d + w(u,v)$

then
- $v.d = u.d + w(u,v)$
- $v.\pi = u$

2]

Iteration 1 =

$t.d = 6$ $t.\pi = s$

$y.d = 7$ $y.\pi = s$

Iteration 2 =

$x.d = 11$ $x.\pi = t$

$z.d = 16$ $z.\pi = y$

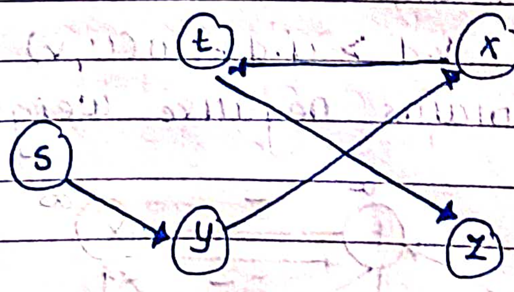
- $x.d = 4$ $x.\pi = y$
- $z.d = 2$ $z.\pi = t$

Iteration 3 =

$t.d = 2$ $t.\pi = x$

Iteration 4 =

$z.d = -2$ $z.\pi = -2$



3] check for negative weight cycle.

If condition still satisfies $u.d + w(u,v) < v.d$

then

Not correct result.

- complexity $= O(V \times E)$

$=$ Vertices X Edges.

for complete graph edges $= \frac{V(V-1)}{2}$

$\therefore$ complexity $= V * \frac{V(V-1)}{2}$

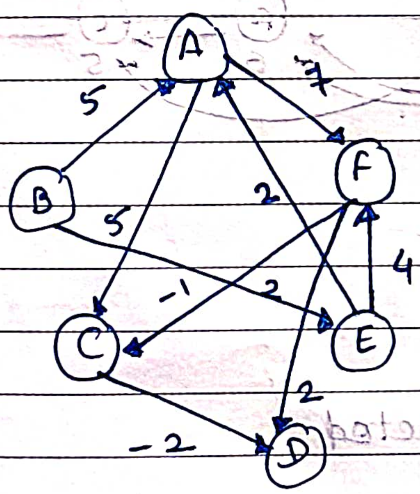
$\approx O(V^3)$

* Order in which edges are processed affects how quickly the algorithm works.

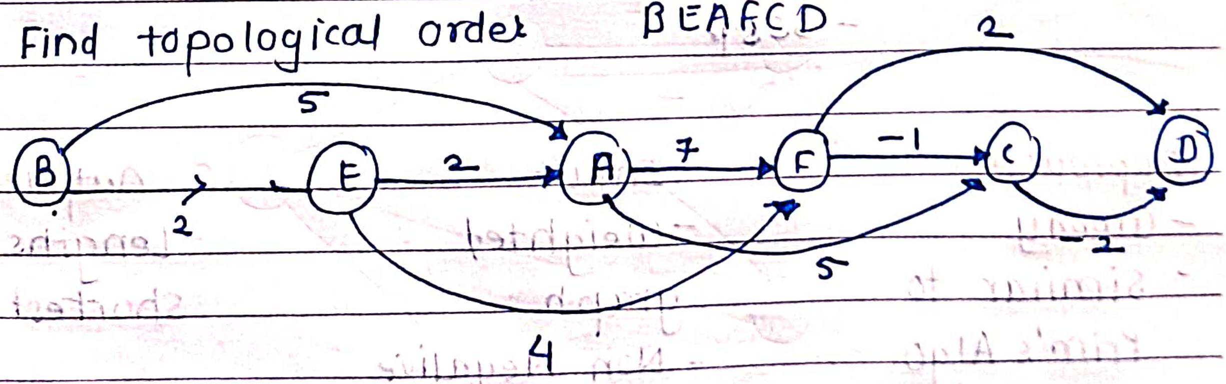
* SSSP in DAG
  └ directed Acyclic path

Complexity = $\underline{V+E}$?

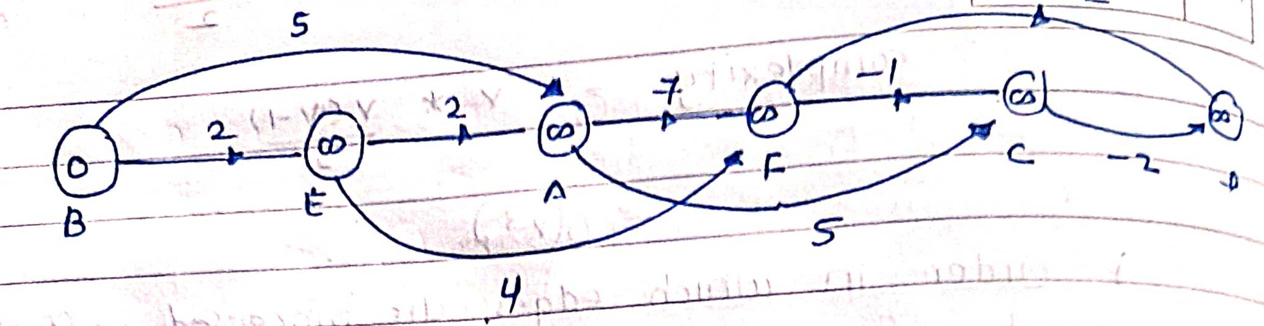
Example



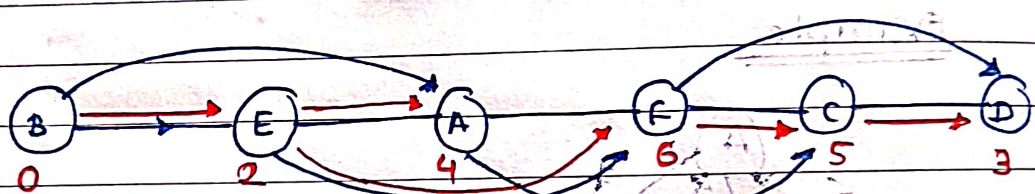
1) Find topological order BEAFCD



- Relax (U, V, W)
- Allows negative weight edges
- Not contains any cycle.

1) 

- $E.d = 2$ — $E.\pi = B$
  $A.d = 5$ — $A.\pi = B$
- $A.d = 4$ — $A.\pi = E$
  $F.d = 6$ — $F.\pi = E$
- $C.d = 5$ — $C.\pi = F$
- $D.d = 3$ — $D.\pi = C$

∴ 

→ Dijkstra's Algorithm:

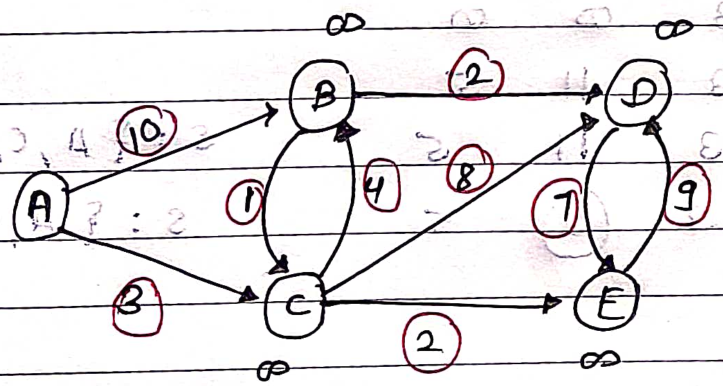
- No Negative edges
- Works on both directed and undirected.
- SSSP

Approach:
- Greedy
- Similar to Prim's Algo

Input
- Weighted graph
- Non Negative edges

Output
- Lengths of shortest path.

1]



A	B	C	D	E	
0	∞	∞	∞	∞	{A}

A	B	C	D	E	
0	∞	∞	∞	∞	
	10	3	∞	∞	{A}

A	B	C	D	E
0	∞	∞	∞	∞
0	10	3	∞	∞

S : {A, C}

A	B	C	D	E
0	∞	∞	∞	∞
0	10	3	∞	∞
0	7	3	11	5

S : {A, C}

A	B	C	D	E
0	∞	∞	∞	∞
0	10	3	∞	∞
0	7	3	11	5

S : {A, C, E}

A	B	C	D	E
0	∞	∞	∞	∞
0	10	3	∞	∞
0	7	3	11	5
0	7	3	11	5
0	7	3	9	5

S : {A, C, E, B}

S : {A, C, E, B, D}