# Algorithm

## Spanning tree

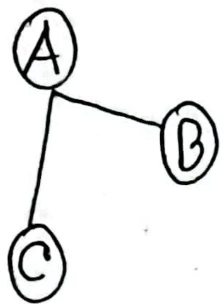
A tree is a graph with no-simple circuit.

A spanning tree of a connected graph is a

some { * Maximum set of edges that contains no cycle
  * Minimum set of " " connects all vertices

A S.T is a subset of graph G, which has all the vertices converted with minimum possible edges.

A complete undirected graph can have max. $n^{n-2}$ no of S.T
[n = ~~nuber~~ number of nodes]



### Graph of S.T :-

-> A connected Graph G can be more than one.

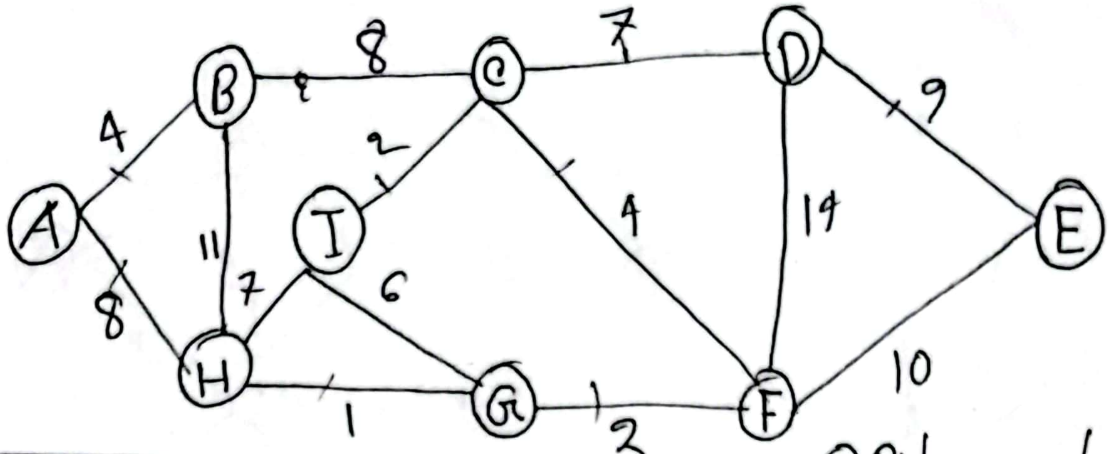
-> All possible S.T have ~~more~~ some number of edges & verticies.

-> Doesn't have any cycle.

-> Removing one edge from S.T will make the graph disconnect. (S.T is milimatly connected).

-> Adding one edge will create a loop, S.T is maximally a cycle.

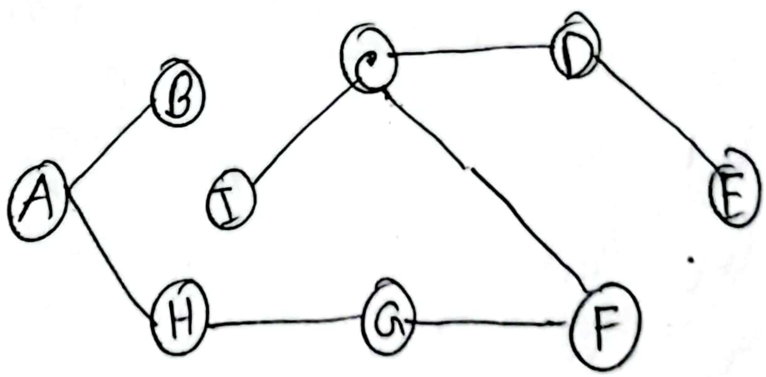
15/10/23

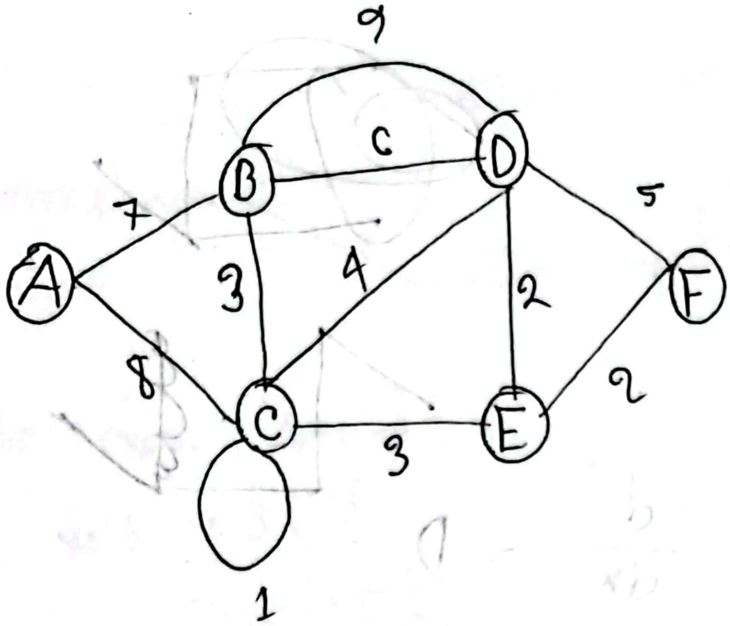


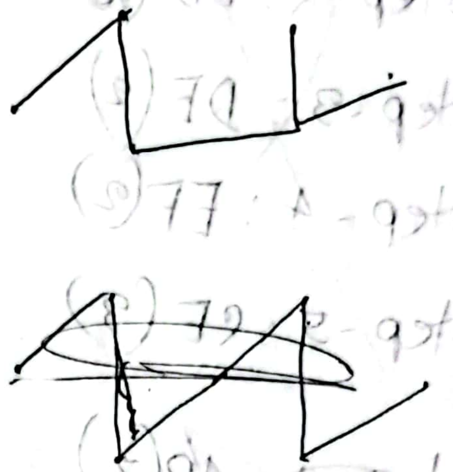

1. Pick a node a starting node.
3. Don't make loops
2. Select small cost edges from starting node & keep selection.

~~Step-1: AB (4)~~
~~Step-2: BC (8)~~
~~Step-3: CI (2)~~
~~Step-4: IG (6)~~
~~Step-5: GH (1)~~

Step-1: AB (4)
Step-2: AH (8)
Step-3: HG (1)
Step-4: GF (2)
Step-5: CI (2)
Step-6: CF (4)
Step-7: CD (7)
Step-8: DE (9)





Step:1 : EF (2)
Step:2 : DE (2)
Step-3 : BC (3)
Step-4 : CD (4)
Step-5 : AB (7)

~~Minimu~~ ~~cost: 18~~

Step : 1 : EF (2)
Step : 2 : DE (2)
Step-3 : CE (3)
Step-4 : BC (3)
Step-5 : AB (7)

cost : 17

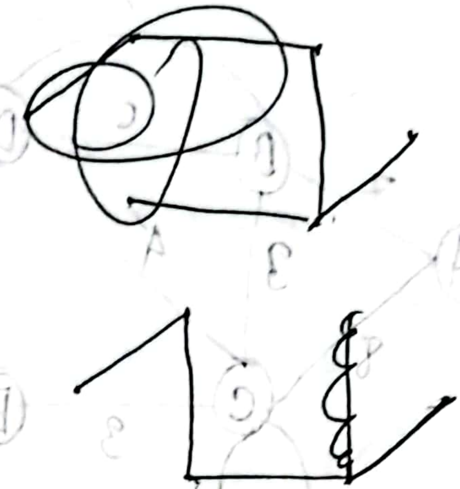
~~Step-1 : AB (7)~~

~~Step-2 : BD (6)~~

~~Step-3 : DE (2)~~

~~Step-4 : EF (2)~~

~~Step-5 : CE (3)~~



Step-1 : AB (7)

Step-2 : BC (3)

Step-3 : CE (3)

Step-4 : EF (2)

Step-5 : DF (5)

$\frac{d}{dx} = D$

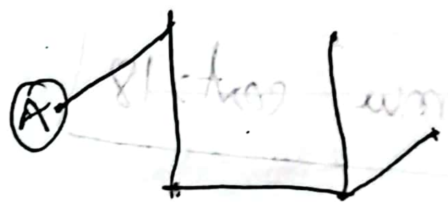
Step-1 : AB (7)

Step-2 : BC (3)

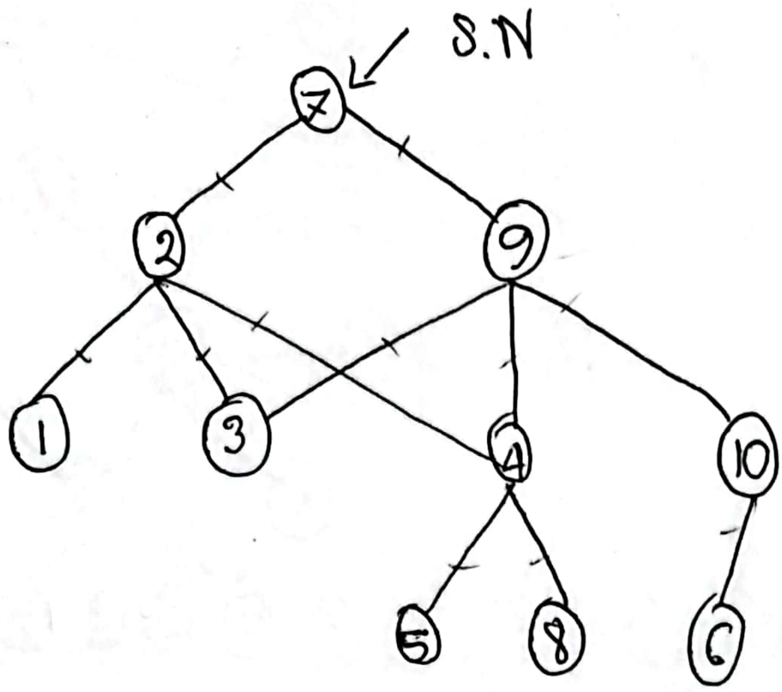
Step-3 : CE (3)

Step-4 : EF (2)

Step-5 : DE (2)




cost : 17




BFS: 7, 2, 9, 1, 3, 4, 10, 5, 8, 6

Queue: 7|2|9|1|3|4|10|5|8|6

Visit: 7, 2, 9, 1, 3, 4, 10, 5, 8, 6

DFS: 7, 9, 10, 6, -

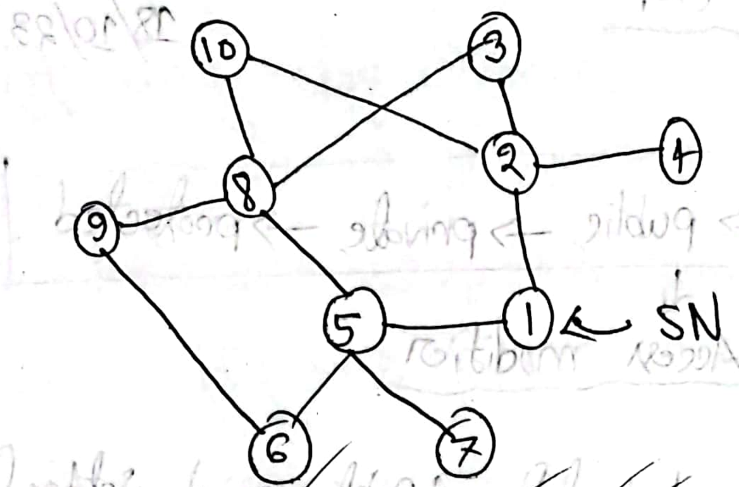
Stack

| |
|---|
| ~~6~~ 10 |
| 4 |
| 3 |
| 9 |
| 2 |
| 7 |

~~Visit~~:

Visit: 7, 2, 9, 3, 4, 10, 6, 5

222-115-019




BFS: 1, 5, 2, 6, 7, 8, 3, 4, 9, 10, 9

Queue:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ~~1~~ | ~~5~~ | ~~2~~ | ~~6~~ | ~~7~~ | ~~8~~ | ~~3~~ | ~~4~~ | ~~9~~ | ~~10~~ | ~~9~~ |

Visit: 1, 5, 2, 6, 7, 8, 3, 4, 9, 10, ~~10~~

DFS: 1, 2, 10, 8, 9, 6, 4, 3, 7, 5

Stack:

| |
|---|
| ~~7~~ |
| ~~8~~ |
| ~~9~~ |
| ~~8~~ |
| ~~10~~ |
| ~~4~~ |
| ~~3~~ |
| ~~2~~ |
| ~~5~~ |
| ~~1~~ |

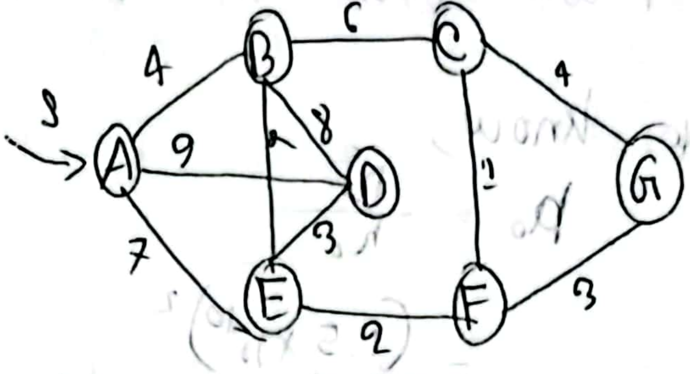
Visit: 1, 5, 2, 3, 4, 10, 8, 9, 6, 7

30/10/23

## Single Source Path shortest Path Algorithm.

1. Dijkstra
2. Bellman Ford

$G = \{V, E\}$

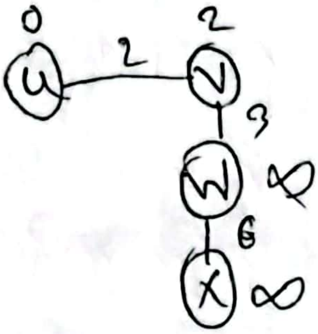



$D(u)$

$c(u,v)$

$d(u) \rightarrow$ from source to u cost (min)

$c(u,v) \rightarrow$ edge between u to v (cost)

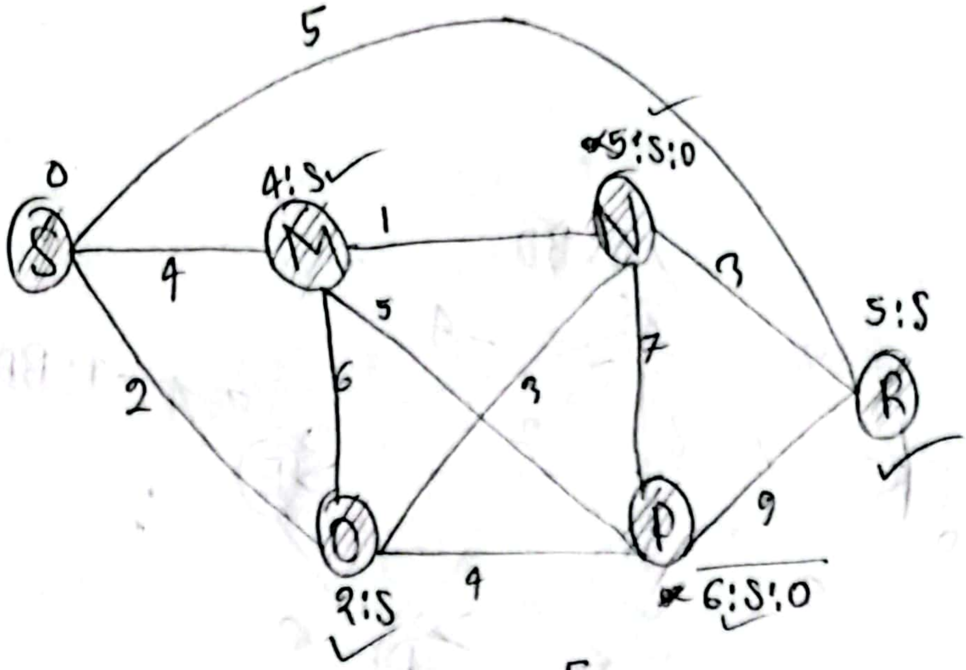


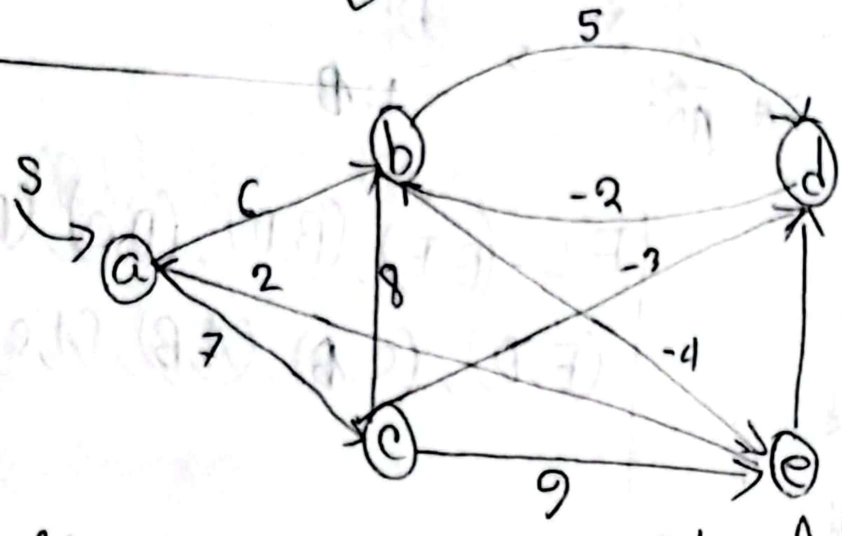

$d(v) + c(v,w) < d(w)$

$2 + 3 < \infty$

$d(w) = 5$

Drawbreak:




Bellman
Ford
(n-1) [Iteration]

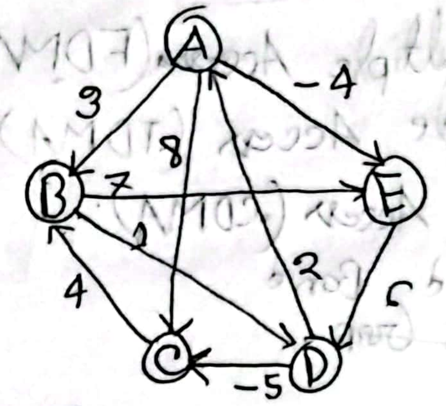
Step-1:
Step-2: changes
Step-3: changes
Step-4: No change

Edge list:

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

# Algo

## All pair shortest Path (Floyed Warshall)



$$M^0 = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix} \end{array} ; \quad P^0 = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} - & A & A & - & A \\ - & - & - & B & B \\ - & C & - & - & - \\ D & - & D & - & - \\ - & - & - & E & - \end{bmatrix} \end{array}$$

$$M^A = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \underline{5} & -5 & 0 & \underline{-2} \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix} \end{array} , \quad P^A = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} - & A & A & - & A \\ - & - & - & B & B \\ - & C & - & - & - \\ D & A & D & - & A \\ - & - & - & E & - \end{bmatrix} \end{array}$$

$$M^B = \begin{array}{cc} & \begin{matrix} A & B & C & D & E \end{matrix} \\ \begin{matrix} A \\ B \\ C \\ D \\ E \end{matrix} & \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & \infty & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & & & \end{bmatrix} \end{array}$$

Algo

Solution Tree/ State space Tree

## Backtrack:

- It is a technique based on algorithum to solve the problem.
- It follows recursive method.
- It uses Brute force approch to the problem & find all possible solution.
- Examples > N-Queen, sum of subset, graph coloring.

Example:-

Section: A B C

Room: 401 402 403

| | | |
|---|---|---|
| A | B | C |