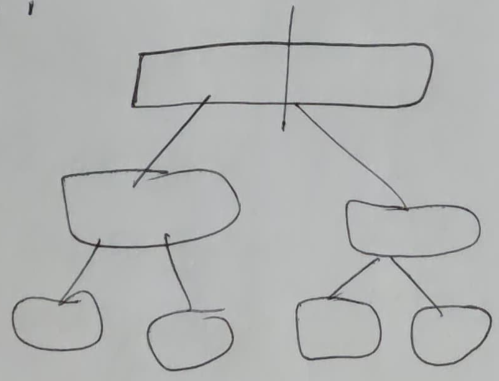
MODULE-4
DYNAMIC PROGRAMMING

// Dynamic programming :- It divides the problem into series of overlapping sub-problems.

~~To~~ 2 features 1) Optimal substructure
2) Overlapping subproblems

Minimum cost.

* Dynamic programming is an algorithm design method that can be used when the solution to a problem can be viewed as the result of a sequence of decisions.



In D & C, there is no repeat.
But in DP, the sub-problems repeat.
store and retrieve data.

Eg :- Fibonacci series $f(n) = f(n-1) + f(n-2)$

MULTISTAGE GRAPH

Definition → It is defined as

* Directed graph $G = (V, E)$ in which vertices are partitioned into 'k' stages, where $k >= 2$.
* Each stage contains disjoint sets V_i where $1 <= i <= k$, i.e., any stage the number of vertices should not exceed

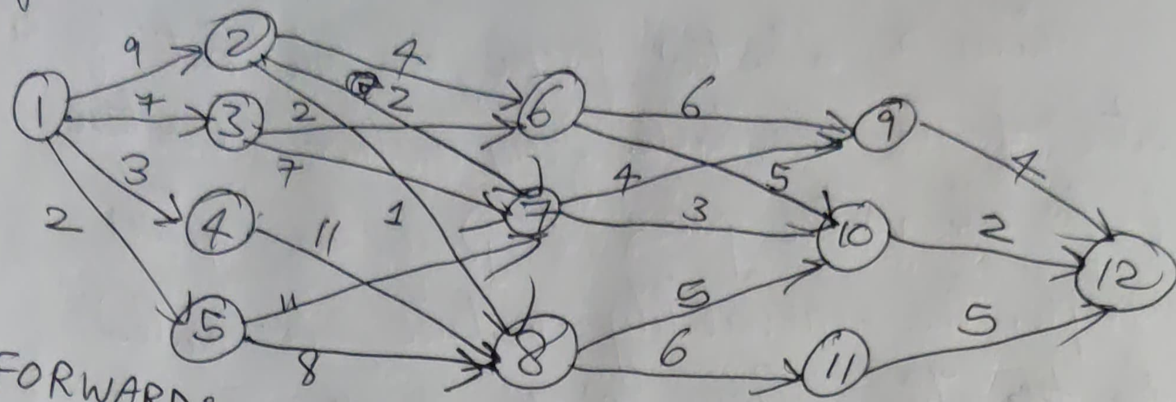
SCP

Forward Approach ←

Backward Approach →

} Directed graph

Eg:-



FORWARD:- 1 ← 12

BACKWARD:- 1 → 12

Solution:- A is a 2-D array of weight

n=12, s=1 (source)
d=12 (destination)

FORWARD

c[12] = 0, p[12] = 12

c[11] = min{a[11,12] + c[12]}
= 5

c[10] = 2

c[9] = 4

c[8] = min{5+2, 6+5} = 7

c[7] = min{3+2, 4+4} = 5

c[6] = min{6+4, 5+2} = 7

c[5] = min{8+6+5, 11+3+2}
= 16

c[4] = min{11+6+5} = 22

c[3] = min{2+6+4, 7+3+2}
= 12

c[2] = min{4+6+4, 2+3+2,
2+6+5}
= 7

c[1] = min{~~9+4+6+4~~ 12,
9+7, 12+7
3+22, 2+16} = 16

BACKWARD

c[1] = 0, p[1] = 0

c[2] = ~~min {9, 7, 3, 2}~~ 9

c[3] = 7

c[4] = 3

c[5] = 2

c[6] = min{9+4, 7+2} =

c[7] = min{9+2, 7+7, 2+11} =

c[8] = min{2+8, 3+11} = 10

c[9] = min{⑥+6, ⑦+4} = 15

c[10] = min{⑥+5, ⑦+3, ⑧+5} = 1
14 14 15

c[11] = min{⑧+6} = 16

c[12] = min{⑨+4, ⑩+2, ⑪+5}
~~= min~~

$\boxed{c[12] = 16}$

TRANSITIVE CLOSURE

Given a directed graph, we find out if a vertex j is reachable from another vertex i for all vertex pairs (i, j) in the given graph. The reachability matrix is called the transitive closure of the group.

WARSHALL'S ALGORITHM

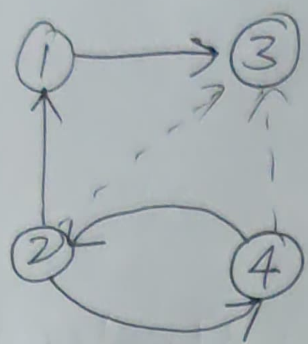
$R^{(0)} = A$

for $k=1$ to n do
  for $i=1$ to n do
    for $j=1$ to n
      $R^{(k)}[i,j] \leftarrow R^{(k-1)}[i,j]$ or $(R^{(k-1)}[i,k]$ and $R^{(k-1)}[k,j])$

  return $R^{(n)}$

~~R~~

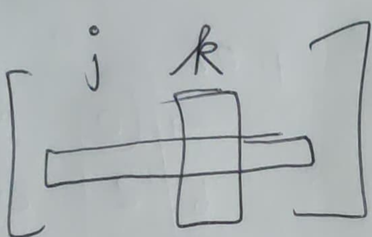
Eg:-



$$\begin{bmatrix} 0 & \cancel{0}\,1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ \cancel{0}\,1 & 0 & \cancel{0}\,1 & 0 \end{bmatrix}$$

If 1 is present, leave it as it is.
If 0, change to 1 in $R^{(k)}$ if and only if the element in its row i and column k and the element in column j and row k are both 1s in $R^{(k-1)}$

$R^{(k-1)}$



$\rightarrow$

$$R^{(k)} = \begin{array}{c|cc} & j & k \\ \hline k & 1 & \\ i & 1 & 1 \end{array}$$

If both are 1, change to 1

$$R^{(0)} = \begin{array}{c|cccc} & a & b & c & d \\ \hline a & 0 & \cancel{0}\,1 & 0 & 0 \\ b & 0 & 0 & 0 & 1 \\ c & 0 & 0 & 0 & 0 \\ d & \cancel{0}\,1 & \cancel{0}\,1 & \cancel{0}\,1 & 0 \end{array}$$

$$R^{(1)} = \cancel{0} \begin{bmatrix} 0 & 1 & 0 & \cancel{0}\,1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & \cancel{0}\,1 & 1 & \cancel{0}\,1 \end{bmatrix}$$

$$R^{(2)} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \cancel{0} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$R^{(3)} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

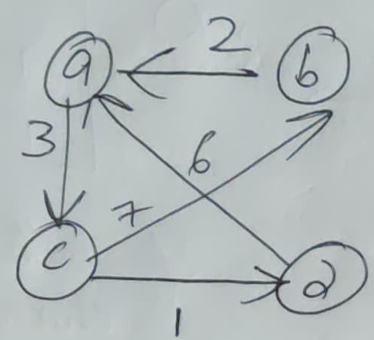
$$R^{(4)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ \cancel{0}\,1 & 1 & 1 & 1 \end{bmatrix}$$

$O(n^3)$

FLOYD'S ALGORITHM

```
/ Floyd (W[1---n, 1--n])
for (k=1 to n)
{
    for (i=1 to n)
    {
        for (j=1 to n)
        {
            D[i,j] <- min {D[i,j], D[i,k] ~~and~~ + D[k,j]}
        }
    }
}
return D
```

All pair shortest path ~~→ it is used to find the~~ .



$$D(0) = \begin{array}{c|cccc} & a & b & c & d \\ \hline a & 0 & \infty & \cancel{a}3 & \infty \\ b & 2 & 0 & \cancel{\infty}5 & \infty \\ c & \infty & 7 & 0 & 1 \\ d & 6 & \infty & \cancel{\infty}9 & 0 \end{array}$$

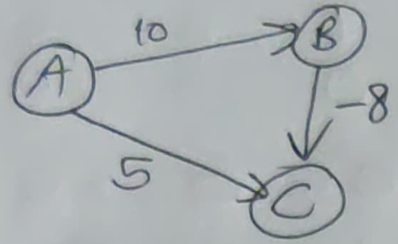
$$D(1) = \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & 5 & \infty \\ \cancel{\infty}9 & 7 & 0 & 1 \\ 6 & \infty & 9 & 0 \end{bmatrix}$$

$$D(2) = \begin{bmatrix} 0 & \cancel{\infty}\,10 & 3 & \cancel{\infty}4 \\ 2 & 0 & 5 & \cancel{\infty}6 \\ 9 & 7 & 0 & 1 \\ 6 & \cancel{\infty}16 & 9 & 0 \end{bmatrix}$$

$$D(3) = \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 9 & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix}$$

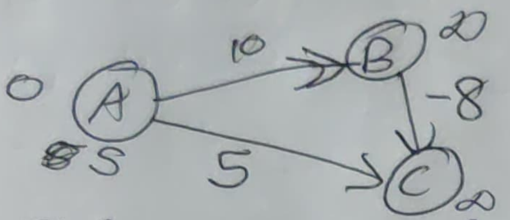
$$D(4) = \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 7 & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix}$$

BELLMAN FORD ALGORITHM



A	B	C
0	∞	∞
	10	5

DIJKSTRA'S X



→ R_1

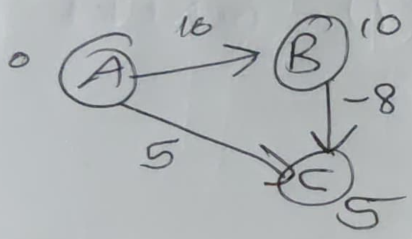
0 + 10 = 10

0 + 5 = 5

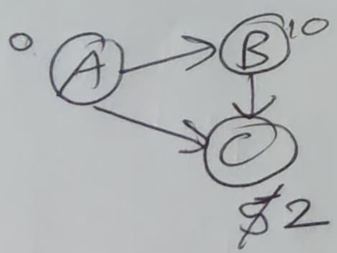
Relax every edge

V-1 number of times

~~3-2~~ 3-1 = 2 times relax



→ R_2

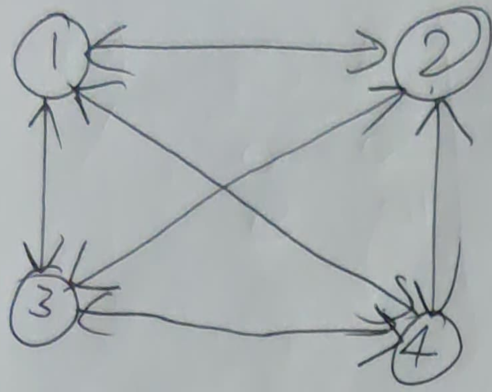


m = 8

n = 4

	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	1	1	1	1
2	0	0	1	2	2	3	3	3	3
3	0	0	1	2	5	5	6	7	7
4	0	0	1	2	5	6	6	7	~~7~~8

TRAVELLING SALESMAN PROBLEM



$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 10 & 15 & 20 \\ 2 & 5 & 0 & 25 & 10 \\ 3 & 15 & 30 & 0 & 5 \\ 4 & 15 & 10 & 20 & 0 \end{array}$$

Start from 1,
travel all and come back to 1.
Minimum cost

$$1 \xrightarrow{10} 2 \xrightarrow{10} 4 \xrightarrow{20} 3 \xrightarrow{15} 1$$

= 55 (NOT MINIMUM) = GREEDY