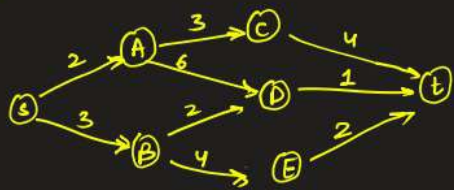
Dynamic Programming Lecture 6

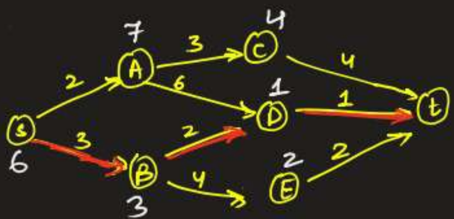
Saturday, 31 August 2024 2:33 PM

Multi - Stage Graphs

↳ directed graph where the nodes are arranged in stages, and edges only connect nodes from one stage to the next stage.



Objective :- Given a multi - stage graph, find the shortest path from a source vertex 's' to the terminal vertex 't'.



$$s(A \to t) = \min \begin{cases} A \to C + s(C \to t) \\ A \to D + s(D \to t) \end{cases} = 7$$

$$s(B \to t) = \min \begin{cases} B \to D + s(D \to t) \\ B \to E + s(E \to t) \end{cases} = 3$$

$$s(s \to t) = \min \begin{cases} s \to A + s(A \to t) \\ s \to B + s(B \to t) \end{cases} = 6$$

{s : (A,2), (B,3), A : (C,3) . . . }

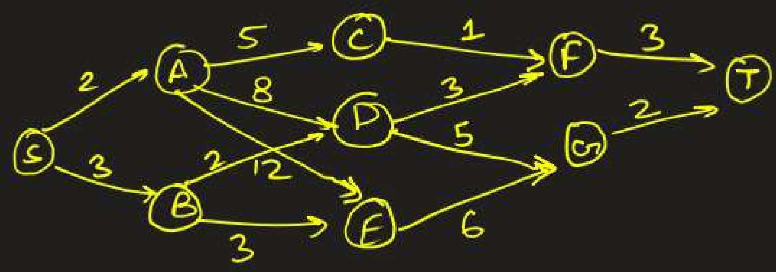
Assuming that vertices $1 \to n$ are in stage ordering & $s = $ '1', $t = $ 'n'

```
function minCostPath (edges, n):
    cost = array of size n initialized
           to ∞
    cost [n] = 0
    for (i : n-1 → 1):
        for (j, w) in edges [i] :
            cost[i] = min( cost[i], w + cost[j])

    return cost[1]
```

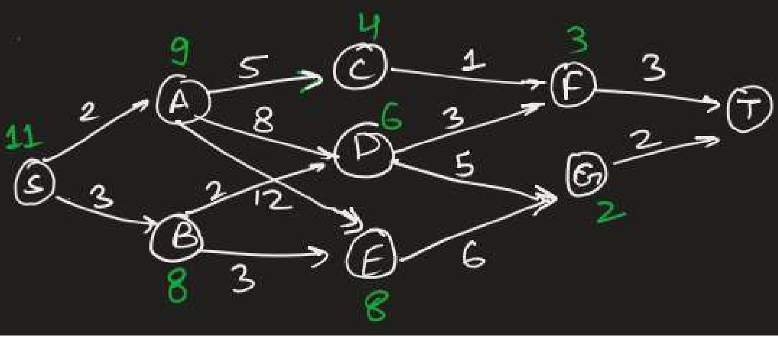
$T = O(E)$

$S = O(V)$

eg.



What is the shortest distance from S to T ? 11



S → A → C → F → T

S → B → D → F → T

Shortest

All pairs Shortest path

↳ Find the shortest paths between all pairs of nodes in a weighted graph.

$\boxed{\text{Floyd-Warshall Algorithm}}$

Dij from every node :- $E \log V \cdot V$

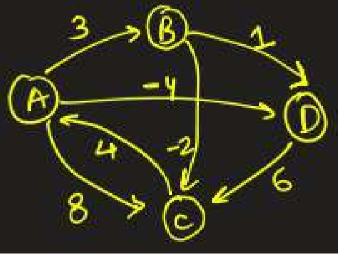
$= VE \log V$

$\sim V^3 \log V$

Bell. Ford every node :- $V \cdot E \cdot V$

$= V^2 \cdot E$

$\sim V^4$



$$D^0 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 3 & 8 & -4 \\ \infty & 0 & -2 & 1 \\ 4 & \infty & 0 & \infty \\ \infty & \infty & 6 & 0 \end{bmatrix} \end{array}$$

$i \to j$ with 0 vertices in between

$B \to C$

$\underline{B \to A} \to C$

$$D^A = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 3 & 8 & -4 \\ B & \infty & 0 & -2 & 1 \\ C & 4 & 7 & 0 & 0 \\ D & \infty & \infty & 6 & 0 \end{array}$$

$$D[i][j] = \min \begin{cases} D[i][j] \\ D[i][k] + D[k][j] \end{cases}$$

$D[A][B] = \min(D[A][B],$
$D(A)(A) + (A \to B))$

$$D^B = \begin{bmatrix} 0 & 3 & 1 & -4 \\ \infty & 0 & -2 & 1 \\ 4 & 7 & 0 & 0 \\ \infty & \infty & 6 & 0 \end{bmatrix}$$

$C \to A$	$C \to B \to A$	$D \to A$	$D \to B \to A$
4	$7 + \infty$	∞	∞
$C \to D$	$C \to B \to D$	$D \to C$	$D \to B \to C$
0	$7 + 1$	6	∞

$$D^C = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 3 & 1 & -4 \\ B & 2 & 0 & -2 & -2 \\ C & 4 & 7 & 0 & 0 \\ D & 10 & 13 & 6 & 0 \end{array}$$

$$\boxed{D^D = \begin{bmatrix} 0 & 3 & 1 & -4 \\ 2 & 0 & -2 & -2 \\ 4 & 7 & 0 & 0 \\ 10 & 13 & 6 & 0 \end{bmatrix}}$$ All pair shortest path

At any point if the diagonal elements $D[i][i] < 0$ then there is a negative weight cycle.

```
function allPairsShortestPath (V, M):
    D = M
    for (k : 1 → V):
        for (i : 1 → V):
            for (j : 1 → V):
                D[i][j] = min(D[i][j], D[i][k] + D[k][j])

    return D
```

$T = O(V^3)$

$S = O(V^2)$

[GATE CS 2016 Set 2]

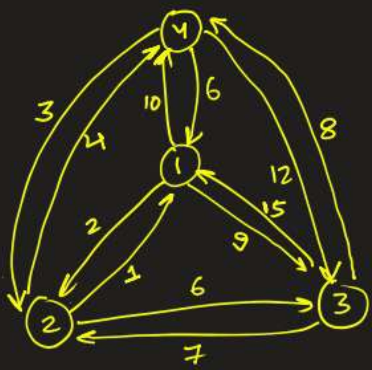
The Floyd-Warshall algorithm for all-pair shortest paths computation is based on

A. Greedy Paradigm
B. Divide and Conquer Paradigm
C. Dynamic Programming Paradigm
D. None of these

Travelling Salesman Problem

Given:- A weighted graph

Objective:- start from a given city, find the shortest path to visit every city and come back to the source.



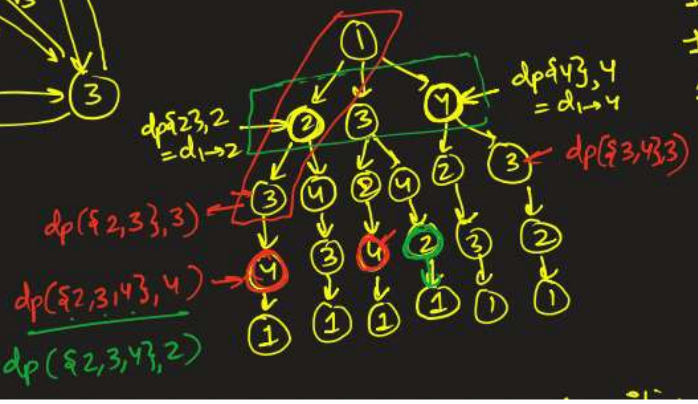
S = 1

Brute:- All possible orderings

$1 \to 2 \to 3 \to 4 \to 1$
$1 \to 2 \to 4 \to 3 \to 1$
$1 \to 3 \to 4 \to 2 \to 1$
$1 \to 3 \to 2 \to 4 \to 1$
$1 \to 4 \to 2 \to 3 \to 1$
$1 \to 4 \to 3 \to 2 \to 1$

$(n-1)! \times \underset{\text{distance}}{n}$

$= O(n!)$



$dp(S, i)$: minimum cost to visit all cities in S starting from 1 and ending at i

$$ans = \min_{k \neq 1} \{ dp(\{2,3 \dots n\}, k) + d_{k \to 1} \}$$

$$dp(S, i) = \begin{cases} d_{1 \to i} , & S = \{i\} \\ \min\limits_{\substack{k \neq i, \\ k \in S}} dp(S \setminus \{i\}, k) + d_{k \to i} \end{cases}$$

$$dp(\{3,4\}, 3)$$
$$dp(\{4\}, 4) + d_{4 \to 3}$$

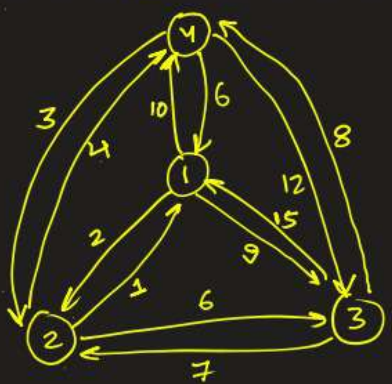
$$dp(\{2,3,4\}, 3)$$
$$dp(\{2,4\}, 4) + d_{4 \to 3}$$
$$dp(\{2,4\}, 2) + d_{2 \to 3}$$

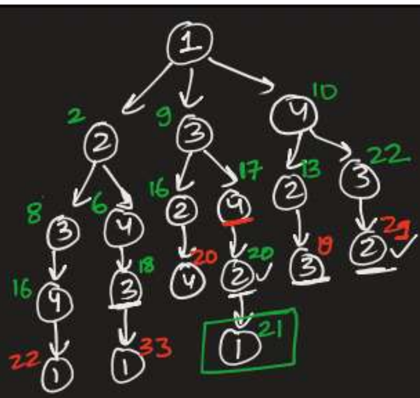
$$dp(\{2,3,\cancel{4},5\}, 4) = \min \begin{cases} dp(\{2,3,5\}, 2) + d_{2 \to 4} \\ dp(\{2,3,5\}, 3) + d_{3 \to 4} \\ dp(\{2,3,5\}, 5) + d_{5 \to 4} \end{cases}$$

```
function minPathTravellingSalesman (G, n):

    for (i : 2 → n):
        dp({i}, i) = d_{1→i}              } set of size 1

    for (s : 2 → n-1):                    ← set size from 2 → n-1
        for all S ⊆ {2,3,...,n} , |S| = s :
                for all k ∈ S :
        n2          dp(S, k) = ∞
                    for all m ∈ S , m ≠ k :
                        dp(S,k) = min( dp(S,k), dp(S \ {k}, m) + d_{m→k} )

    minPath = ∞
    for (k : 2 → n):                                          } calculate
        minPath = min( minPath, dp({2,3...n}, k) + d_{k→1} )  } ans

    return minPath
```

Subsets = $O(2^n)$

$$\boxed{\begin{array}{l} T = O(n^2 \cdot 2^n) \\ S = O(n \cdot 2^n) \end{array}}$$





$dp(\{2\},2) = d_{1\to2} = 2$

$dp(\{3\},3) = d_{1\to3} = 9$

$dp(\{4\},4) = d_{1\to4} = 10$

$dp(\{2,3\},2) = dp(\{3\},3) + d_{3\to2}$

$dp(\{2,3\},3) = dp(\{2\},2) + d_{2\to3}$

$dp(\{2,4\},2)$

$dp(\{2,4\},4)$

$dp(\{3,4\},3)$

$dp(\{3,4\},4)$

$$dp(\{2,3,4\},2) = \min\begin{cases} \overset{22+7=29}{dp(\{3,4\},3)+d_{3\to2}} \\ \underset{17+3=20}{dp(\{3,4\},4)+d_{4\to2}} \end{cases} = 20$$

$$dp(\{2,3,4\},3) = \min\begin{cases} \overset{13+6=19}{dp(\{2,4\},2)+d_{2\to3}} \\ \underset{6+12=18}{dp(\{2,4\},4)+d_{4\to3}} \end{cases} = 18$$

$$dp(\{2,3,4\},4) = \min\begin{cases} \overset{16+4=20}{dp(\{2,3\},2)+d_{2\to4}} \\ \underset{8+8=16}{dp(\{2,3\},3)+d_{3\to4}} \end{cases} = 16$$

Shortest Path: $1\to3\to4\to2\to1$, distance = 21