- two integer of length is multiplied

<u>Split</u> $m = \underline{n/2}$

- Negative edge weight cycle

  - Dijktra's method may or may not work Negative edge weight.

Matrix Chain Multiplication

Matrix Multiplication

- Iterative algorithm
- Recursive version

- Chain of compitable matrices given

Recursive

Matrix Multiplication

- On multiplying two Matrices of size $p \times q$ and $q \times r$, number of scalar matrix multiplication possible is

$$p \times q \times r$$

Matrix Chain Multiplication

- consider the problem of a chain $\{A_1. A_2 A_3\}$ of three matrices. Suppose that the dimensions of the matrices are 10×100, 100×5, and 5×50, respectively. What is the number of scalar product for different parenthesization. How many ways to we can do the parenthesization.

A_1 A_2 A_3

10×100 100×5 5×50

$(A_1(A_2 A_3))$

$((A_1 A_2) A_3)$

How many different ways we can multiply

$A_1, A_2 A_3$

Matrix Chain Multiplication

- 10×100, 100×5, and 5×50,

$(A_1 (A_2 \cdot A_3))$

$((A_1 \cdot A_2) A_3)$

$\begin{bmatrix} a & b \\ c & d \end{bmatrix}_{2\times 2} \begin{bmatrix} e & f & I \\ g & h & J \end{bmatrix}_{2\times 3}$

2

$\begin{bmatrix} a*e + b*g \\ \\ \end{bmatrix}$

dimension

$n \times m \quad m \times p \quad - \quad (n \times m \times p)$

$2 \times \boxed{2 \times 3}$

$\boxed{12}$

No. of Scalar multiplication

$(n \times m \times p)$

Matrix Chain Multiplication

- Our next example of dynamic programming is an algorithm that solves the problem of matrix-chain multiplication. We are given a sequence (chain) $\{A_1;\ A_2; \ldots.\ ; A_n\}$ of n matrices to be multiplied, and we wish to compute the product $A_1;\ A_2; \ldots.\ ; A_n$:

What is the number of scalar multiplication

- $((A_1A_2).A_3)$ (I) — 10×100, 100×5 5×50
- $(A_1(A_2.A_3))$

$10 \times 100 \times 5 = 5000$ multiplying $A_1 A_2$

the dimension $A_1 \cdot A_2 = \boxed{10 \times 5}$

Speed up

$(A_1 A_2) \cdot \cancel{5 \times 50}\ A_3 \Rightarrow$

10×5 $\quad$ 5×50

cost $10 \times 5 \times 50 = 2500$

total $5000 + 2500 = 7500$

cost of $A_2 A_3$

$100 \times 5 \times 50 = 25000$

Size dimension - 100×50

$A_1 (A_2 A_3)$

10×100 $\quad$ 100×50 =

cost $10 \times 100 \times 50 = 50000$ $\quad$ total = $50,000 + 25000 = 75000$

Solution

- If we multiply according to the parenthesization $((A_1A_2).A_3)$ we perform $10 \cdot 100 \cdot 5 = 5000$ scalar multiplications to compute the 10×5 matrix product A_1A_2, plus another $10 . 5 . 50 = 2500$ scalar multiplications to multiply this matrix by $A3$, for a total of 7500 scalar multiplications.

Solution

Optimal

- If instead we multiply according to the parenthesization $(A_1(A_2.A_3))$ we perform 100 • 5 • 50 = 25,000 scalar multiplications to compute the 100 x 50 matrix product A_2A_3,*plus* another 10 • 100 • 50 = 50,000 scalar multiplications to multiply A_1 by this matrix, for a total of 75,000 scalar multiplications. Thus, computing the product according to the first parenthesization is 10 times faster.

Fully parenthesized

$T(3) = 2$ $T(4) = 5$

- How many different ways we can parenthesized $A_1A_2A_3A_4$

$A_1 A_2 A_3$

$(A_1 (A_2 A_3))$

$((A_1 A_2) \cdot A_3)$

$(A_1 A_2 A_3 A_4)$ — $(A_1 (A_2 A_3 A_4))$

$(A_1 ((A_2 A_3) A_4))$ $(A_1 (A_2 (A_3 A_4)))$

total No. of ways = 5

$((A_1 A_2)(A_3 A_4))$ — only one way

$((A_1 A_2 A_3) A_4)$ — 2 ways we can paranthesize

Counting the number of parenthesizations

$T_3 = 2$

$T_4 = 5$

$(A_1 A_2 A_3 A_4 A_5)$

Sub problem

$(A_1 (A_2 A_3 A_4 A_5))$ — 5 ways

$((A_1 A_2)(A_3 A_4 A_5))$ — 2 ways

$((A_1 A_2 A_3)(A_4 A_5))$ — 2 ways

$((A_1 A_2 A_3 A_4) A_5)$ — 5 ways

14 ways

Minimize this

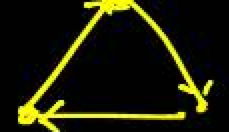
No. of wais n matrices can be paranthesized Represented by (n-1) catalan No.

$T(n-1)$

Catalan No $T(n) = \frac{1}{n+1} {}^{2n}C_n$

10^9

$$\underbrace{(23|45 \times 3652)}_{n}$$



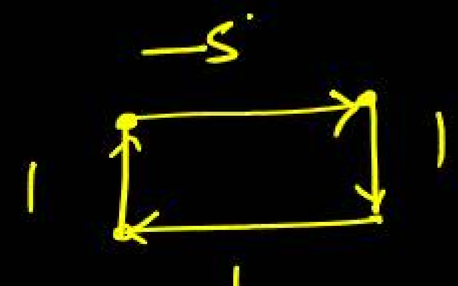
$A = m = n/2$

$$(23 \times 10^2 + 45)(36 \times 10^2 + 52)$$

$$(w \times 10^m + x)(y \times 10^m + z)$$

$T(n) = 4T(n/2) +$

$$(\underbrace{wy}_{n/2} \times 10^{2m} + \underbrace{wz}_{n/2} \times 10^m + \cancel{w}\, \underbrace{xy}_{n/2} \times 10^m + \underbrace{xz}_{n/2})$$



$= -2$ Negative

Counting the number of parenthesizations

$$P(n) = \begin{cases} 1 & if\ n = 1 \\ \sum_{k=1}^{n-1} p(k)P(n-k) & if\ n \geq 2 \end{cases}$$

This is leading to Catalan number

Problem Statement

Given a chain of compitible matrices

$A_1 A_2 A_3 \cdots A_n$

find the way to parenthesized the multiplication that minimize No. of Scalar Multiplication.

Problem Statement

- We state the matrix-chain multiplication problem as follows: given a chain {A1;A2;…..;An} of n matrices, where for i = 1;2;…….;n, matrix Ai has dimension $p_{i-1} \times p_i$, fully parenthesize the product $A1A2\ldots\ldots\ldots A$n in a way that ***minimizes the number of scalar multiplications.***

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

	1	2	3	4
4			12	0
3		15	0	
2	15	0		
1	0			

S

No. of Scalar multiplication

$\boxed{A_1}$

$i = j$

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

			0
		0	
15	0		
0			

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

			0
	15	0	
15	0		
0			

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

	1	2	3	4
4			12	0
3	24	15	0	
2	15	0		
1	0			

$(A_1 A_2 A_3)$

3×5   5×3

$A_1 \cdot (A_2 A_3)$

$0 + 15 + 45$

$= \underline{60}$

$(A_1 A_2) A_3$

3×1   1×3

$15 + 0 + 9$

$\underline{24}$

tabulation

Method work

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

	1	2	3	4
4		32	12	0
3	24	15	0	
2	15	0		
1	0			

$A_2 A_3 A_4$

$\underbrace{(A_2 A_3)}_{5\times3} \underbrace{A_4}_{3\times4}$ $\quad$ $\underbrace{A_2}_{5\times1} \underbrace{(A_3 A_4)}_{1\times4}$

$15 + 0 + 60$ $\quad$ $0 + 12 + 20$

min $(\underline{75} , 32)$

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

	1	2	3	4
4	39	32	12	0
3	24	15	0	
2	15	0		
1	0			

Plan -
cost of 4 matrices
2 computation
2 computation
3 computation
7 computation

$(A_1 A_2 A_3 A_4)$

$\underset{3\times5}{A_1}\underset{5\times4}{(A_2 A_3 A_4)}$ $\qquad$ $\underset{3\times1}{(A_1 A_2)}\underset{1\times4}{(A_3 A_4)}$ $\qquad$ $\underset{3\times3}{(A_1 A_2 A_3)}\underset{3\times4}{A_4}$

min $(0+32+60$, $\quad 15+12+12$, $\quad 24+0+36)$

min $(92$, $\quad \underline{39}$, $\quad 60)$

$((A_1 A_2)(A_3 A_4))$

Compute the Cost

- A1, A2 , A3 and A4 have the dimension 3× 5 , 5×1; 1×3; 3×4

39	32	12	0
24	15	0	
15	0		
0			

$B[1 \ldots n] = \min \begin{cases} B[1,1] + B[2..n] + r_1 c_1 c_n \\ B[1,2] + B[3..n] + r_1 c_2 c_n \\ B[1.3] + B[4..n] + r_1 c_3 c_n \\ \vdots \\ B[1..n-1] + B[n,n] + r_1 c_{n-1} c_n \end{cases}$

$r_1 c_1 \qquad r_2 c_n$

$A_1 \; A_2 \; A_3 \; A_4$

$= B[1 \ldots 4]$

$(A_1 \; A_2 \; A_3) \ldots A_n$

$r_1 c_1 \quad r_2 c_2 \quad r_3 c_3 \quad r_n c_n$

$r_1 c_{n-1} \cdot c_n$

The structure of an optimal Parenthesization

$$B[1 \ldots\ldots\ldots..n] = \min \left\{ \begin{array}{c} \overset{i}{B[1,1]} + B[2,n] + r_1 c_1 c_n \\ B[1,2] + B[3,n] + r_1 c_2 c_n \\ B[1,3] + B[4,n] + r_1 c_3 c_n \\ \cdots\cdots \\ B[1,n-1] + B[\mathrm{n},n] + r_1 c_{n-1} c_n \end{array} \right\}$$

Tomorrow

$$B[i \ldots j] = \min \begin{cases} 0 & i = j \\ B[i,k] + B[k+1,j] + r_i c_k c_j & \end{cases}$$

$$i \le k \le j-1$$

A Recursive Solution

$$B[i \dots\dots\dots.. j] = \min \begin{cases} \min\limits_{k=i \ to \ j-1} \{B[i,k] + B[k+1,j] + r_i c_k c_j\} & i < j \\ 0 & i = j \end{cases}$$

$\Uparrow$ $\Uparrow$

code

knapsack

*

Travelling salse person problem

Gate 2016 Set-II

- A1, A2 , A3 aLet A_1, A_2, A_3, and A_4 be four matrices of dimensions 10×5, 5×20, 20×10, and 10×5, respectively. The minimum number of scalar multiplications required to find the product $A_1A_2A_3A_4$ using the basic matrix multiplication method is ______ .

	1	2	3	4
4	1500	1250	1000	0
3	1500	1000	0	
2	1000	0		
1	0			

7 calculation

$(A_1((A_2\ A_3)\ A_4))$

$A_1A_2A_3 = \begin{cases} A_1(A_2A_3) \\ 0+1000+500 \\ (A_1A_2)A_3 \\ 1000+0+2000 \end{cases}$

$10\times5\times5$

$(A_2A_3A_4) = \begin{cases} A_2(A_3A_4) \\ 0+1000+500 = 1500 \\ (A_2A_3)A_4 \\ 1000+0+250 = \boxed{1250} \end{cases}$

$(A_1A_2A_3A_4)^{min}\begin{cases} A(A_2A_3A_4) \\ 0+1250+250 = 1500 \\ (A_1A_2)(A_3A_4) \\ 1000+1000+ \text{—} \\ (A_1A_2A_3)A_4 \\ 1500+0+ \text{—} \end{cases}$

The answer is 1500.

Matrix Parenthesizing: $A_1 ((A_2A_3)A_4)$

Check my solution below, using dynamic programming

A_1	A_2	A_3	A_4
10×5	5×20	20×10	10×5

- $A_{12} = 10 \times 5 \times 20 = 1000$
- $A_{23} = 5 \times 20 \times 10 = 1000$
- $A_{34} = 20 \times 10 \times 5 = 1000$

$$\mathrm{A}_{13} = \min \begin{cases} A_{12} + A_{33} + 5\times 20\times 10 = 2000 \\ A_{11} + A_{23} + 10\times 5\times 10 = 1500 \end{cases}$$

$$\mathrm{A}_{24} = \min \begin{cases} A_{23} + A_{44} + 5\times 20\times 10 = 2000 \\ A_{22} + A_{34} + 10\times 5\times 10 = 1500 \end{cases}$$

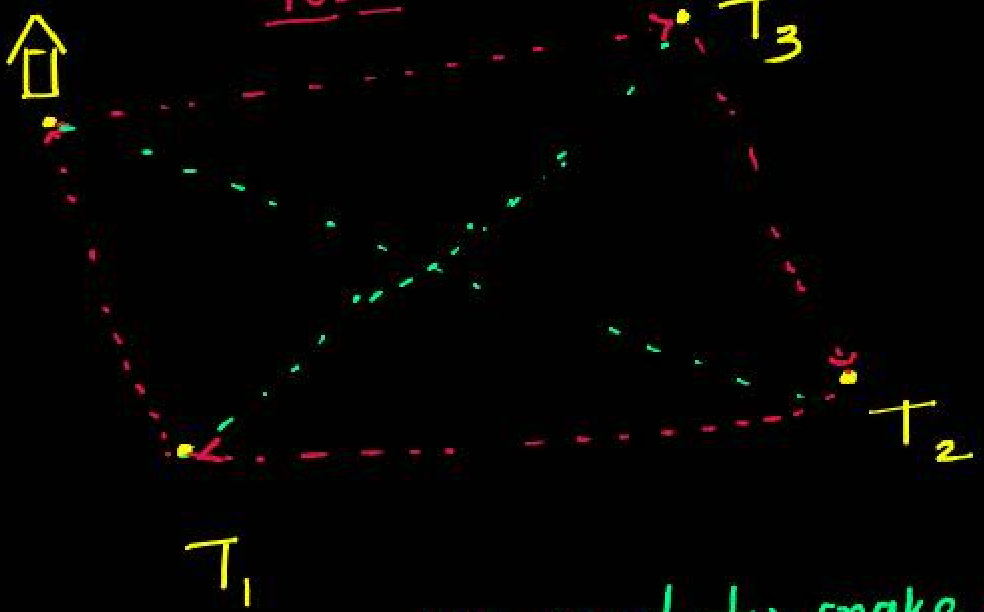
$$\mathrm{A}_{14} = \min \begin{cases} A11 + A24 + 10 \ \times 5 \times 5 = 1500 \\ A12 + A34 + 10 \times 20 \times 5 \geq 2000 \\ A13 + A44 + 10 \times 20 \times 5 = 2000 \end{cases}$$

Graph Source. vertex

Travelling SalesPerson Problem

given 3 task assigned

tour



we need to make a tour So that cost (distance) covered in minimum.

Permutation Paradigm

Catalan NO

knapsack - $\{ob_1, ob_2, ob_3\}$

How many combination I need check for brute force.

Subset

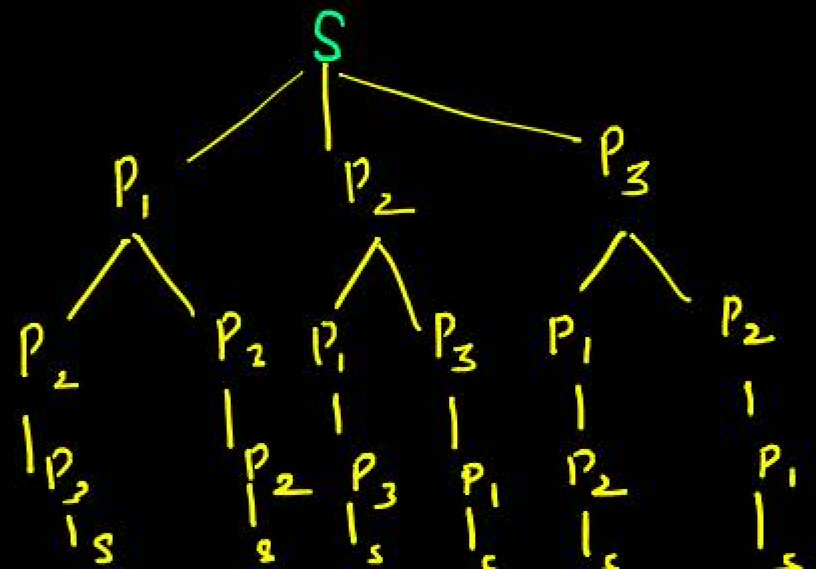
2^n

P_1, P_2, P_3 -

How many different ways we can visit 3 place.

$n!$ } permutation paradigm

S .



Permutation Paradigm

- We have seen how to apply dynamic programming to a subset selection problem (0/1 knapsack). Now we turn our attention to a permutation problem. Note that permutation problems usually are much harder to solve than subset problems as there are n! different permutations of n objects whereas there are only 2^n different subsets of n objects ($n! > 2^n$).

Problem Statement

- Tour of Graph:

Tour: A tour of G (directed graph) is a cycle that includes every vertex in V. once.

The cost of tour is sum of the distance to between pair of vertices.

Travelling sales person problem is to find a tour with minimum cost.

Problem Statement

- A tour of G is a directed simple cycle that includes every vertex in V. The cost of a tour is the sum of the cost of the edges on the tour. The traveling salesperson problem is to find a tour of minimum cost.

Number of Tour possible

Hamiltonian cycle

A cycle that includes every vertex of the graph and first & Last vertex is Same. called Hamiltonian cycle.

Problem Statement

- Let G = (V,E) be a directed graph with edge costs cij. The variable cij is defined such that cij>0 for all i and j and $c_{ij} = \infty$ if $\langle i, j \rangle \notin E$. Let |V| = n and assume n>1.

Application of TSP

- The traveling salesperson problem finds application in a variety of situations. Suppose we have to route a postal van to pick up mail from mail boxes located at n different sites. An n+1 vertex graph can be used to represent the situation.

Problem Statement

- In the following discussion we shall, without loss of generality, regard a tour to be a simple path that starts and ends at vertex 1.

Problem Statement

- In the following discussion we shall, without loss of generality, regard a tour to be a simple path that starts and ends at vertex 1.
- Every tour consists of an edge $\langle 1, k \rangle$ for some $k \in V - \{1\}$ and a path from vertex k to vertex 1.
- The path from vertex k to vertex 1 goes through each vertex in V-{1,k} exactly once.

Principal of Optimality

Principal of Optimality

- It is easy to see that if the tour is optimal, then the path from k to 1 must be a shortest k to 1 path going through all vertices in V-{1,k}. Hence, the principle of optimality holds.

Principal of Optimality

$|V| = n$

let $g(i, S)$ be the length of shortest path at vertex i and going ~~to~~ thrugh all vertices in S and terminating in i.

C_{1k} Represent edge weight of $(1, k)$

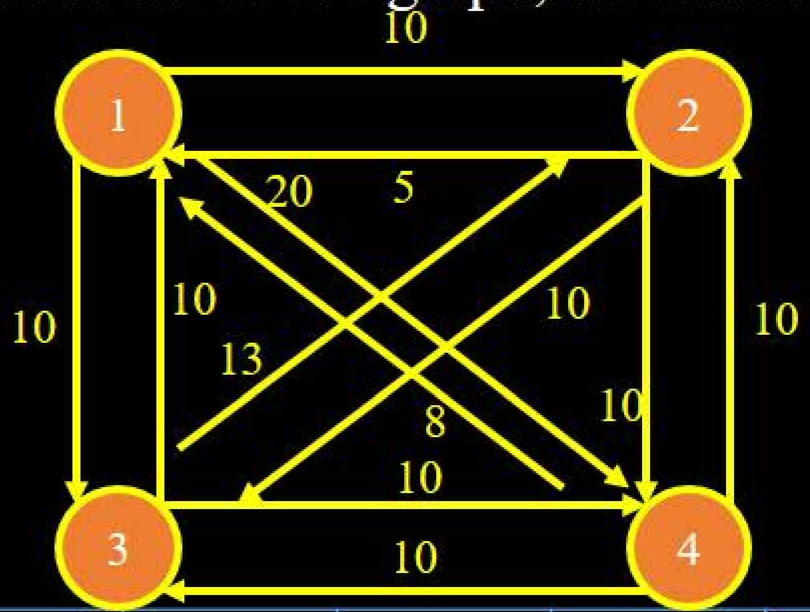
$$g(1, V - \{1\}) = \min_{2 \le k \le n} \left\{ C_{1k} + g(k, V - \{1, k\}) \right.$$

Principal of Optimality

- Let g(i,S) be the length of a shortest path starting at vertex i, going through all vertices in S, and terminating at vertex 1. The function g(1,V-{1}) is the length of an optimal salesperson tour. From the principle of optimality it follows that
  - $g(1, V - \{1\}) = \min_{2 \le k \le n} \{c_{1k} + g(k, V - \{1, k\})\}$
- Generalizing above equation we obtain (for $i \notin S$)
  - $g(i, S) = \min_{j \in S} \{c_{ij} + g(j, S - \{j\})\}$

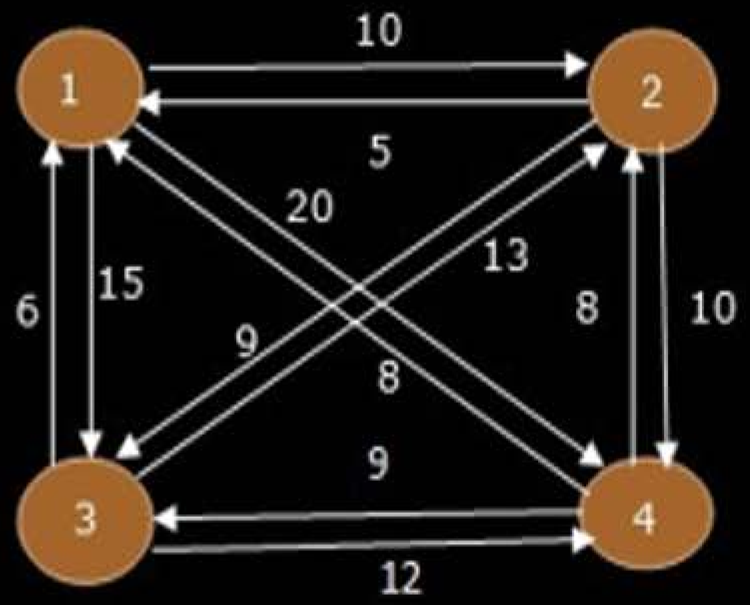
Example

from the above graph, the following table is prepared.



10

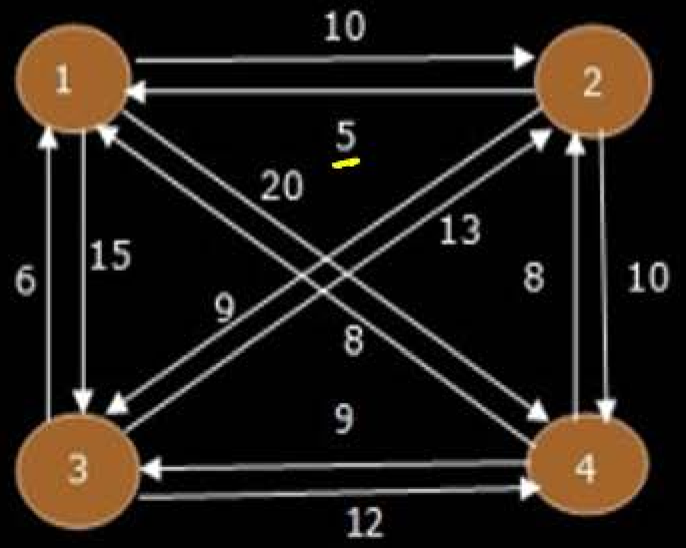
	1	2	3	4
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0



Example

from the above graph, the following table is prepared.

principle of optimality



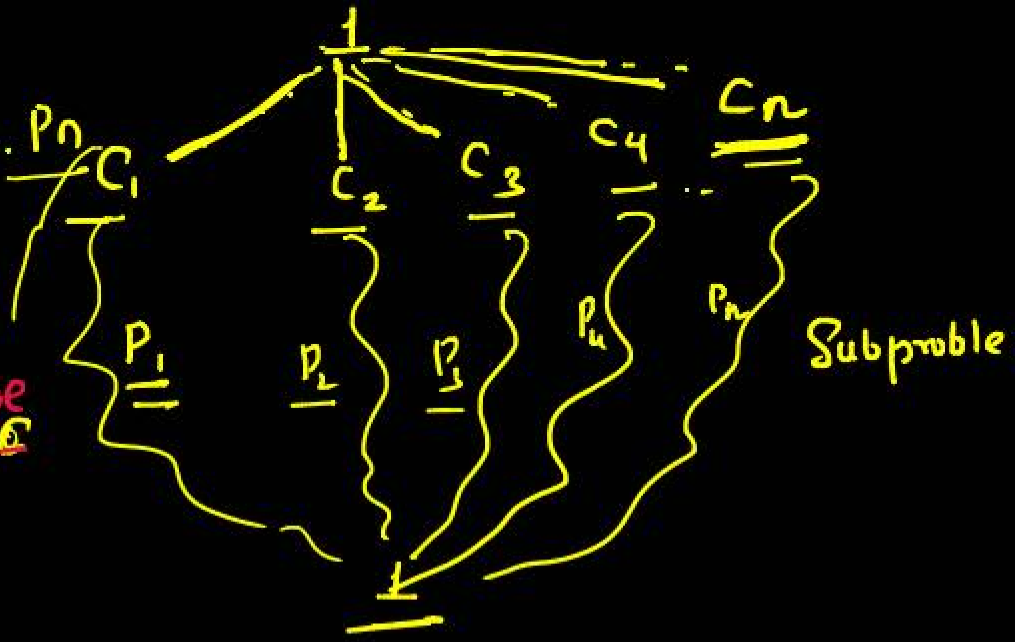
$$\min \begin{cases} w_{1c_1} + P_1 \\ w_{1c_2} + P_2 \\ \vdots \\ w_{1c_n} + P_n \end{cases}$$

tour: 1. visit all other vertices ~~exac~~ exactly once and come back to 1.

	1	2	3	4
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0

weight of the edge

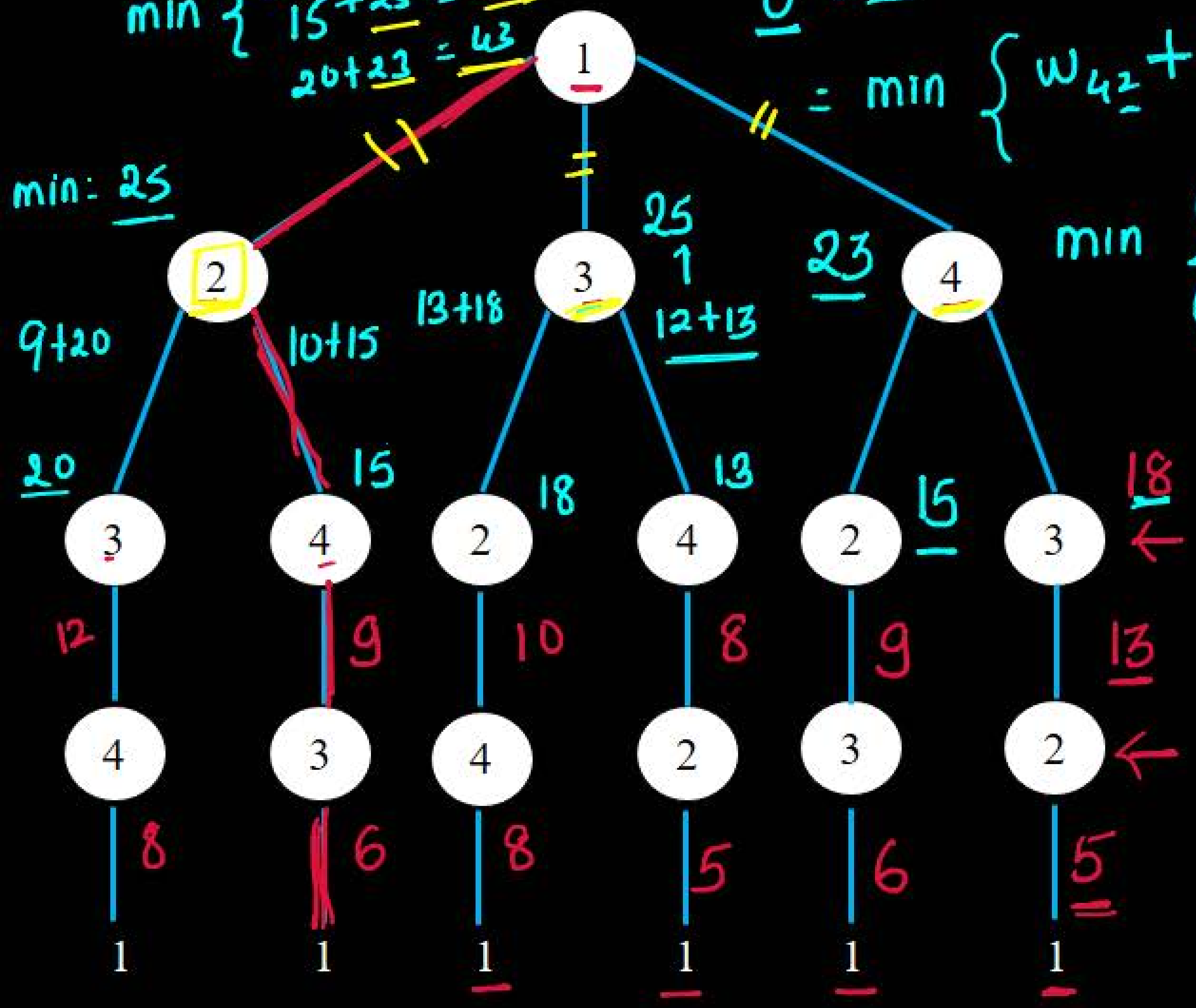
$P_1, P_2, P_3 \ldots P_n$



from the above graph, the following table is prepared.

$g(1, \{2,3,4\})$

$= \min \begin{cases} 10+25 = 35 \checkmark \\ 15+25 = 40 \\ 20+23 = 43 \end{cases}$



$g(4, \{2,3\})$

$= \min \{ w_{42} + g(2, \{3\})$

$\min \begin{cases} 9+18 \\ 8+15 \end{cases}$

	1	2	3	4
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0

Cost of reaching 3 to 1

tour - (1 - 2 - 4 - 3 - 1)

Minimum cost

$g(4, \emptyset)$ $g(3, \emptyset)$ $g(2, \emptyset)$

- Thus $g(2, \phi) = c_{21} = 5, g(3, \phi) = c_{31} = 6$, *and* $g(4, \phi) = c_{41} = 8$.
- Using first equation, we obtain
- $g(2, \{3\}) = c_{23} + g(3, \phi) = 15,$
- $g(2, \{4\}) = 18$
- $g(3, \{2\}) = 18,$
- $g(3, \{4\}) = 20$
- $g(4, \{2\}) = 13,$
- $g(4, \{3\}) = 15$

No. of g values

$g(1, \{2,3,4\})$

$g(2, \{3,4\})$

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

$g(4, \phi)$

Subset values

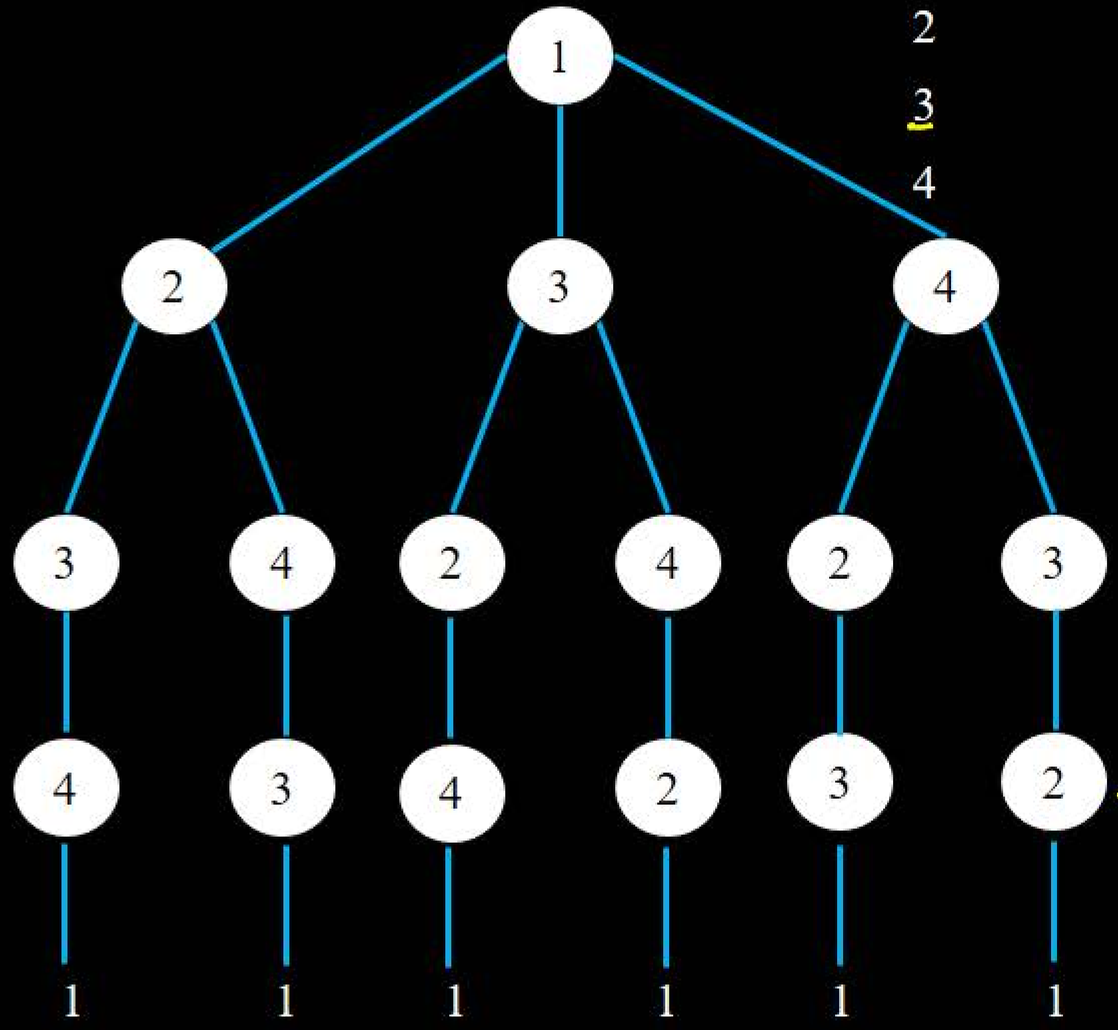
2^n subsets

$(n-1)$ choices

total - $n2^n$

Exponential

from the above graph, the following table is prepared.



	1	**2**	**3**	**4**
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0

$g(3, \{4\}) = C_{34} + g(4, \emptyset)$

$12 + 8 = 20$

$g(4, \{3\}) = C_{43} + g(3, \emptyset)$

$9 + 6 = 15$

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

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

$g(2, \{3\})$

$g(3, \{2\})$

- Next, we compute g(i,S) with |S| = 2, i≠1, 1∉S and i∉S.

(n-1)

$$g(2,\{3,4\}) = \min\{c_{23} + g(3,\{4\}), c_{24} + g(4,\{3\})\} = \boxed{25}$$

- $g(3,\{2,4\}) = \min\{c_{32} + g(2,4), c_{34} + g(4,\{2\})\} = 25$

$$g(4,\{2,3\}) = \min\{c_{42} + g(2,\{3\}), c_{43} + g(3,\{2\})\} = 23$$

25

10

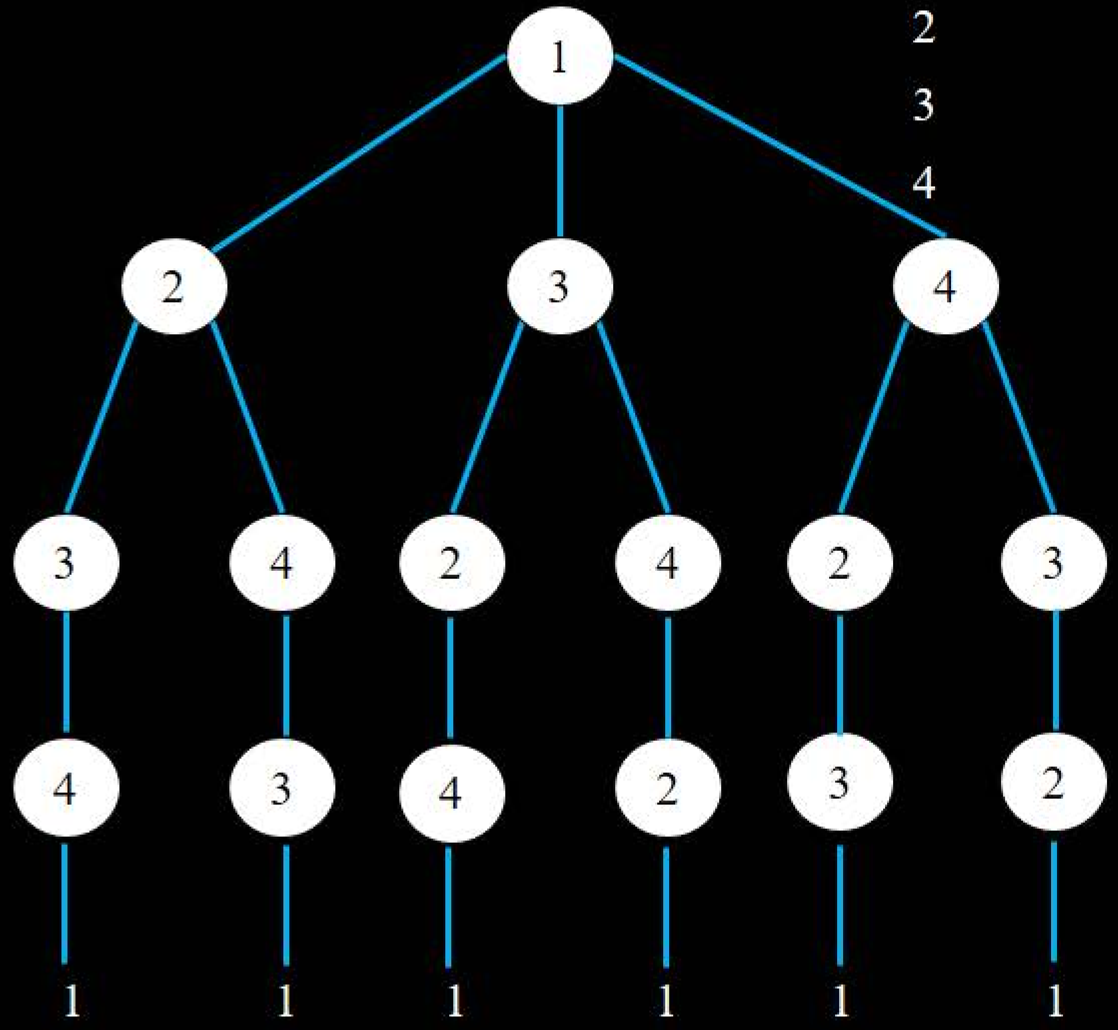
$$g(1,\{2,3,4\}) : \min \begin{cases} C_{12} + g(2,\{3,4\}) = 35 \\ C_{13} + g(3,\{2,4\}) \quad 40 \\ C_{14} + g(4,\{2,3\}) = 43 \end{cases}$$

15

25

20 + 23

from the above graph, the following table is prepared.



	1	2	3	4
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0

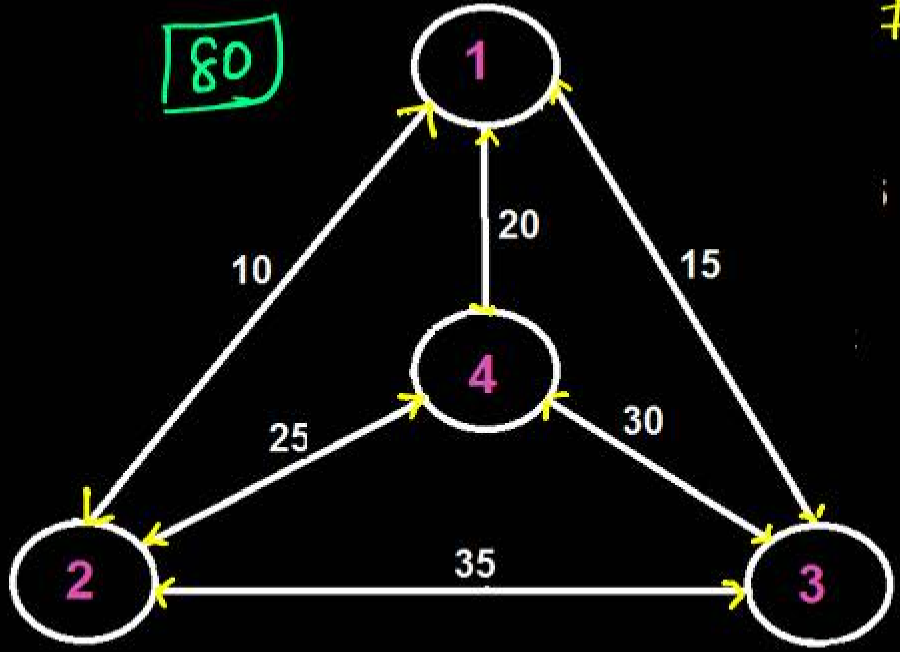
$g(2, \{3,4\})$ $9 + 20$

$$= \min \begin{cases} \underline{C_{23}} + g(3, \{4\}) \\ \underline{C_{24}} + g(4, \{3\}) \end{cases}$$

$10 + 15 = 25$

$$\min \begin{cases} 29 \\ 25 \end{cases} = 25$$

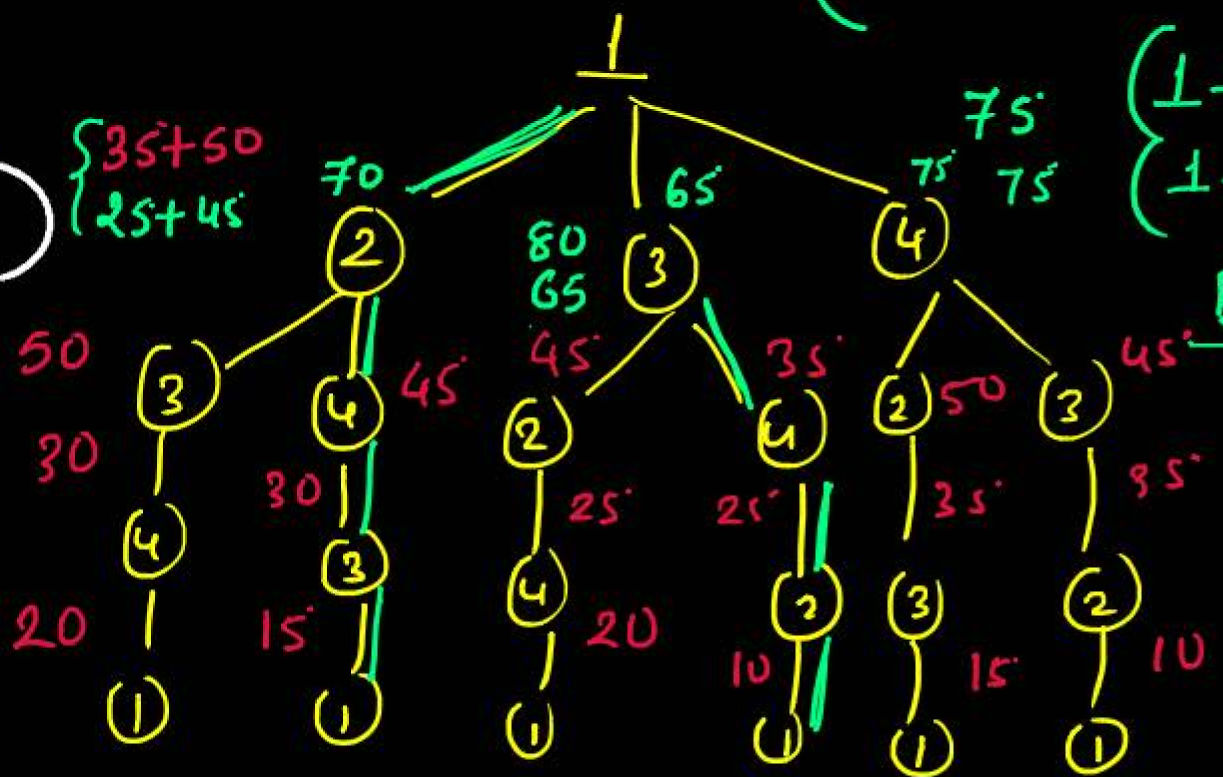
$\boxed{80}$



#1 Solve TSP. 1 is source vertex

Assume bi-directional

$$\min \begin{cases} 70 + 10 = 80 \\ 65 + 15 = 80 \\ 75 + 20 - 95 \end{cases}$$



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

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

Bidirectional

- Finally, from (4.4) we obtain
- $g(1,\{2,3,4\}) = \min\{c_{12} + g(2,\{3,4\}), c_{13} + g(3,\{2,4\}), c_{14} + g(4,\{2,3\})\}$
- $= \min\{35,40,43\} = 35$

Let N be the number of g(i,S)'s that have to be computed before first equation can be used to compute g(1,V-{1}).

$$g(1, \{2,3,4\}) = \min\{c_{12} + g(2, \{3,4\}), c_{13} + g(3, \{2,4\}), c_{14} + g(4, \{2,3\})\}$$

- $g(2, \{3,4\})$
- $g(3, \{2,4\})$
- $g(4, \{2,3\})$
- $g(2, \{3\})$
- $g(2, \{4\})$
- $g(3, \{2\})$

$(n-1)$

- $g(3, \{4\})$
- $g(4, \{2\})$
- $g(4, \{3\})$
- $g(2, \phi)$
- $g(3, \phi)$
- $g(4, \phi)$

$nc_0 - \phi$

$nc_1 : \{1\}\ \{2\}\ \{3\}$

$nc_2 - \{1,2\}\ \{2,3\}\ \{1,3\}$

$nc_3 : \{1,2,3\}$

$nc_0 + nc_1 + nc_2 + nc_n = 2^n$

$2^n \cdot (n-1)$ $\quad n2^n$

- Matrix chain
- Travelling Sales person problem
- All pair shortest path

Dijktra's shortest path

Single Source shortest path

All Pair Shortest Path : Shortest path between every pair of vertices.

Single Source shortest path for every vertices then

$(V+E) \log V$

No. of vertices

$\uparrow$

$V \cdot (V+E) \log V$

$V^2 \log V + VE \log V$

$\simeq O(V^3 \log V)$ (from dense graph)

$E = \frac{(V(V-1))}{2}$

Maximum value

$V \cdot E = V^3$

$E = \frac{V(V-1)}{2}$

$E = V-1$

Minimum

- Given a weighted digraph $G = (V,E)$ with weight function $w : E \rightarrow$ R, (R is the set of real numbers), determine the length of the shortest path (i.e., distance) between all pairs of vertices in G. Here we assume that there are no cycles with zero or negative cost.

First Approach

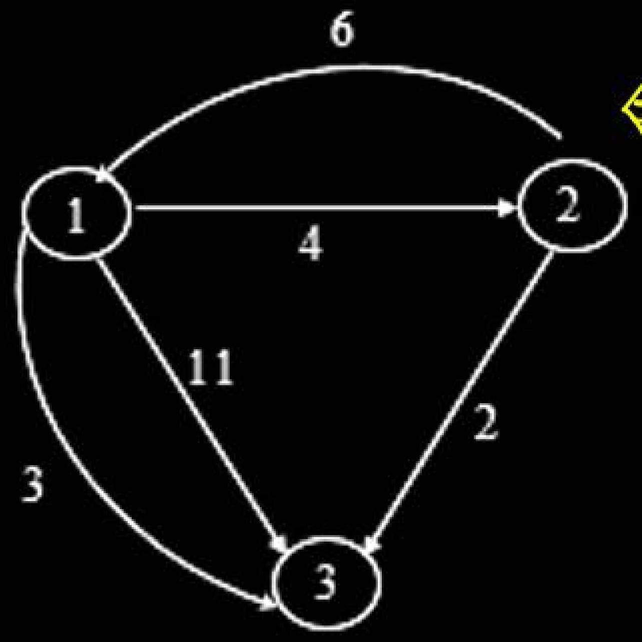
Running dijktra's algorithm for each &

Every vertex.

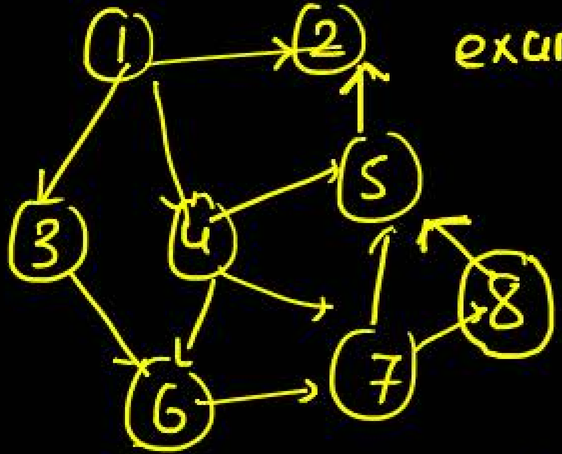
First Approach

- If there are no negative costs edges apply Dijkstra's algorithm to each vertex (as the source) of the digraph.
- Recall that Dijkstra's algorithm runs in $O((|V|+|E|) \log V)$.
- This gives a $O(|V|(|V| + |E|) \log V)$
- $= O(|V|^2 \log V + |V||E| \log V)$ time algorithm,
- If the digraph is dense i.e. complete graph($E = \frac{V(V-1)}{2}$) , this is an $O(|V|^3 \log V)$algorithm.

All Pair Shortest Path

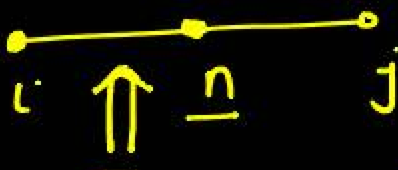


(a) Example diagraph



example graph

H.W



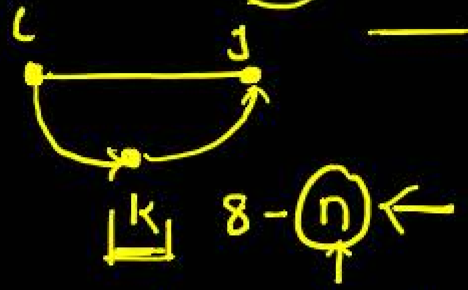
(i, n) (n, j)?

If I take ~~pair~~ pair of vertice i, j

General

i → — j shortest path

maximum index (1, 2, 3, 4 ... 8) of ~~it~~ intermediate vertex can occure



Many intermediate vertex can occur or it may Not occure.