SOLVED QUESTION BANK

[Sequence given as per syllabus]

INTRODUCTION

The main idea of dynamic programming is to solve a given problem by characterizing its subproblems using a small set of integer indices. The goal of this characterization is to allow an optimal solution to a subproblem to be defined by the combination of solutions to even smaller subproblems. This technique underlies the Floyd-warshall algorithm. In this chapter, we describe the general framework of dynamic programming and give several applications such as multistage graphs, all pairs, single source shortest paths, longest common subsequence and 0/1 knapsack problems.

DYNAMIC PROGRAMMING BASIC STRATEGY

Q.1. Explain basic principle of dynamic programming.

CT : S-06,08(5M),S-07,12(4M), S-09,11(2M), W-07,12,13(5M), W-11(6M), W-09(4M)

OR Explain the basic principle of dynamic programming algorithm design technique. *CT : S-10(5M)*

Ans. Basic principle of dynamic programming :

- Dynamic programming solves each subproblem once and stores the result in a table so that it can be rapidly retrieved if needed again.
- It is often used in optimization problems (A problem with many possible solutions for which we want to find an optimal (best) solution).
- Dynamic programming is an approach developed to solve sequential or multistage, decision problems; hence the name "Dynamic" programming.
- Dynamic programming can be thought of as being the reverse of recursion. Recursion is a top-down mechanism-where as dynamic programming is a bottom-up mechanism. We solve all possible small problems and then combine them to obtain solution for bigger problems.

Components of Dynamic programming :

- Dynamic programming solution has three components :

(1) Formulate the answer as a recurrence relation.

(2) Show that the number of different instances of the recurrence is bounded by a polynomial.

(3) Specify an order of evolution for the recurrence.

Dynamic programming solution :

- The development of a dynamic programming algorithm can be broken in to a sequence of four steps :

(1) Characterize the structure of an optimal solution.

(2) Rercusively define the value of an optimal solution.

(3) Compute the value of an optimal solution in a bottom-up fashion.

(4) Construct an optimal solution from computed information.

Examples of Dynamic programming :

(1) Knapsack problem.

(2) Shortest path problems.

(3) Matrix chain multiplication.

(4) Longest common subsequence.

(5) Optimal binary search tree.

Q.2. Explain what is principle of optimality?

CT : W-06(6M), S-07,12,13(2M)

Ans. Principle of optimality :

- The dynamic programming works on principle of optimality. The principle states that in an optimal sequence of decision or choices, each subsequence must be optimal.

Example : In matrix chain multiplication problem, not only the value we are intrested in is optimal but all the other entries in the table are also optimal.

- The principle of optimality is related as follows : "the optimal solution to a problem is a combination of optimal solution to some of its subproblem".

- The difficulty one face during the principle of optimality in an algorithm is that it is not usually obvious which subprograms are relevant to the problem under consideration.

Q.3. Design an algorithm to compute the sum of first 'n' Fibonacci numbers using dynamic programming. Justify that the dynamic programming solution is better than the recursive algorithm.

CT : W-07(8M), S-11, W-11(7M), S-09(6M), W-09(3M)

Ans. Algorithm Fibonacci [n]

```
{
  if (n ≤ 1) then
  write (n)
  else
     {
     f1 = 0
     f2 = 1
          for i = 2 to n do
     {
        f3 = f1 + f2
        f1 = f2
        f2 = f3
     }
     write f (n)
     }
}
```

Dynamic programming solution is better than the recursive algorithm because of following reasons :

- If a problem has optimal substructure, then we can recursively define an optimal solution.
- It a problem doesn't have optimal substructure there is no basis for defining a recursive algorithm to find the optimal solutions.
- If the space of subproblems is small enough (i.e. polynomial in the size of the input) dynamic programming can be much more efficient than recursion.

MULTISTAGE GRAPH

Q.4. Define multistage graph. Give an algorithm to find out the shortest distance between source vertex of a multistage graph. Discuss its time complexity. ***CT : W-07,09(7M)***

OR Write the both backward and forward approach for finding shortest path between source and destination in multistage graph, discuss its time complexity. ***CT : S-10(8M)***

OR Write backward algorithm for multistage graph.

CS : S-10(7M), S-13,14(6M)

OR Define multistage graph. ***CT : S-07, W-13(2M)***

Ans. Multistage graphs : It is a method to represent the information above the distance between various vertices of graph stagewise.

Characteristics of multistage graphs :

- In this graph there is one source and one destination. The vertices are divided into stages [1…K].
- They are connected in such a fashion that vertex of stage 'i' will be only connected with vertex of stage 'i + 1'.
- The vertices selected in the path from source to destination are such that each vertex will represent one stage.

Objective : To find shortest path between source and destination.

Backward Algorithm :

- In this method, the path between source and destination is generated by using the information about stage (i – 1) while calculating for stage (i).

```
Algorithm backward - MS (cost, d, n, K ; P)
{
  bcost [1] = 0
  for j = 2 to n do
  {
     Find vertex 'r' such that < r, j > is edge in
     graph and bcost [r] + cost [r, j] is minimum.
     bcost [j] = bcost [r] + cost [r, j]
        d [j] = r
  }
     P [1] = 1 ;
     P [K] = n ;
```

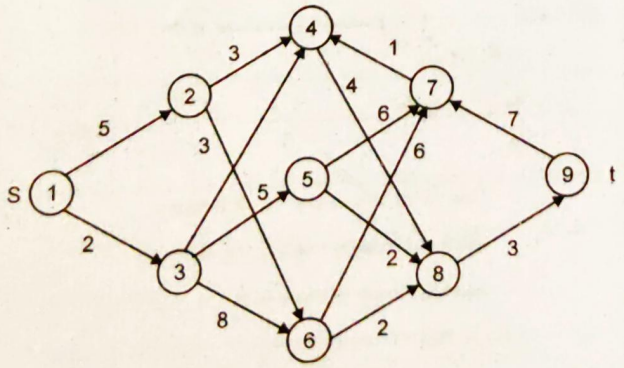
```
    for j = K – 1 to 2 do
    P [j] = d [P [j + 1]]
}
```

Forward Algorithm :

- In this algorithm shortest path is generated from destination to source.
- During calculation the information about next stage will be used and the vertices of next stage having connectivity to current stage will be considered.

```
Algorithm forward - MS (cost, n, d, K, fcost : P)
{
  fcost [b] = 0
  for j = n to 1 do
  {
    Find vertex 'i' such that (j, r) ∈ E and
    fcost [r] + cost [j, r] = min^m
      d [j] = r
    fcost [i] = fcost [r] + cost [j, r]
  }
    P [1] = 1
    P [K] = n
  for j = 2 to K – 1 do
    P [j] = d [P [j – 1]]
}
```

Q.5. Obtain a recurrence relation for finding the shortest path from source vertex to sink vertex. Using given relation find out the shortest distance from vertex 'S' to vertex 't'.



CT : S-07(5M), W-09, S-13(7M), W-13(4M)

CS : W-11(7M)

Ans. bcost [1, 1] = 0

bcost [2, 2] = bcost [1, 1] + cost [1, 2] = 5

bcost [2, 3] = 2

$$\text{bcost}[3,4] = \min_{\substack{l=2\\l=3}} \begin{bmatrix} [\text{bcost}[2,2] + \text{cost}[2,4] \\ \text{bcost}[2,3] + \text{cost}[3,4] \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 \\ 8 \end{bmatrix} = 8$$

$$\text{bcost}[3,5] = \min_{l=3} [\text{bcost}[2,3] + \text{cost}[3,5]]$$

$$= \min [5 + 2] = 7$$

$$\text{bcost}[3,6] = \min_{\substack{l=2\\l=3}} \begin{bmatrix} [\text{bcost}[2,2] + \text{cost}[2,6] \\ \text{bcost}[2,3] + \text{cost}[3,6] \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 \\ 10 \end{bmatrix} = 8$$

$$\text{bcost}[4,7] = \min \begin{bmatrix} 8+1 \\ 7+6 \\ 8+6 \end{bmatrix} = 9$$

$$\text{bcost}[4,8] = \min \begin{bmatrix} 8+8 \\ 7+2 \\ 8+2 \end{bmatrix} = 9$$

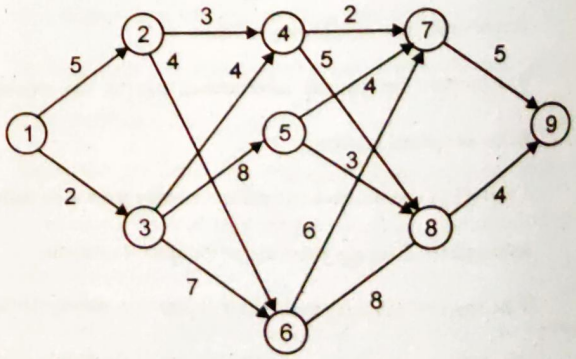
$$\text{bcost}[5,9] = \min \begin{bmatrix} 9+7 \\ 9+3 \end{bmatrix} = 12$$

path $1 \xrightarrow{2} 3 \xrightarrow{5} 5 \xrightarrow{2} 8 \xrightarrow{3} 9$

cost = 12

Q.6. For the following multistage graph, write backward algorithm, to find shortest path from source to destination. Also explain the calculations for shortest path.

CS : W-10(7M)



Ans. **Formula :**

bcost (i, j)

$$= \min_{l \in V_{j-1}} \{(i, j) \in E [bcost(i-1, l) + cost(l, j)]$$

bcost [1, 1] = 0

bcost [2, 2] = 5

bcost [2, 3] = 2

$$bcost[3,4] = \min_{\substack{l=2 \\ l=3}} \begin{bmatrix} bcost[2,2] + cost[2,4] \\ bcost[2,3] + cost[3,4] \end{bmatrix}$$

$$= \min \begin{bmatrix} 5+3 \\ 2+4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

= 6

$$bcost[3,5] = \min_{l=3} [bcost[2,3] + cost[3,5]]$$

= min [2 + 8] = 10

$$bcost[3,6] = \min_{\substack{l=2 \\ l=3}} \begin{bmatrix} bcost[2,2] + cost[2,6] \\ bcost[2,3] + cost[3,6] \end{bmatrix}$$

$$= \min \begin{bmatrix} 5+4 \\ 2+7 \end{bmatrix}$$

$$= \min \begin{bmatrix} 9 \\ 9 \end{bmatrix}$$

= 9

$$bcost[4,7] = \min_{\substack{l=4 \\ l=5 \\ l=6}} \begin{bmatrix} bcost[3,4] + cost[4,7] \\ bcost[3,5] + cost[5,7] \\ bcost[3,6] + cost[6,7] \end{bmatrix}$$

$$= \min \begin{bmatrix} 6+2 \\ 10+4 \\ 9+6 \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 \\ 14 \\ 15 \end{bmatrix} = 8$$

$$bcost[4,8] = \min_{\substack{l=4 \\ l=5 \\ l=6}} \begin{bmatrix} bcost[3,4] + cost[4,8] \\ bcost[3,5] + cost[5,8] \\ bcost[3,6] + cost[6,8] \end{bmatrix}$$

$$= \min \begin{bmatrix} 6+5 \\ 10+3 \\ 9+8 \end{bmatrix}$$

$$= \min \begin{bmatrix} 11 \\ 13 \\ 17 \end{bmatrix}$$

= 11

$$bcost[5,9] = \min \begin{bmatrix} 8+5 \\ 11+4 \end{bmatrix}$$

$$= \begin{bmatrix} 13 \\ 15 \end{bmatrix}$$

= 13

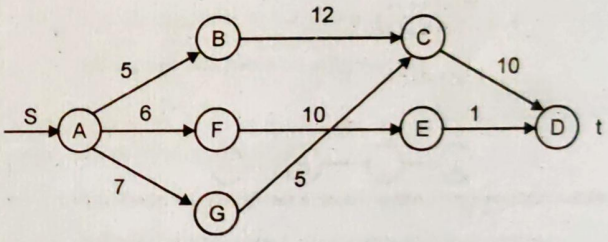
Total cost = 13

Path : 1 → 3 → 4 → 7 → 9 (2, 4, 2, 5)

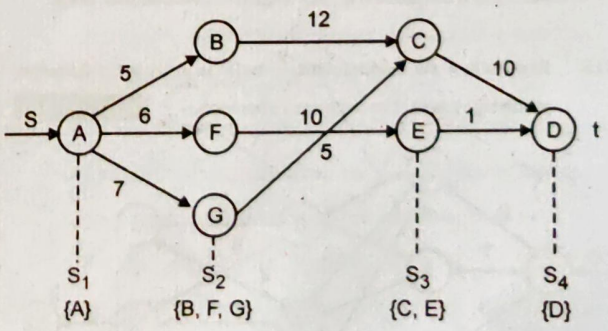
= 13

Q.7. Find a minimum cost path from s to t in the following multistage graph. Is it possible to apply the principle of optimality in finding minimum cost path ? Justify your answer.

CT : W-11(8M)



Ans.



Number of stage = 4

Decision (K – 2) = K – 2 = 4 – 2 = 2

Apply forward algorithm for multistage graph to find minimum cost path from s to t.

Forward Algorithm :

cost (i, j) = min {c (i, *l*) + cost (i + 1, *l*)

Take node B and compare it with nodes of stage S_2.

(1) i = B, j = F, *l* = E

(2) i = B, j = G, *l* = C

Cost (B, F) = min {C (F, E) + cost (C, E) }

= min {10 + 1}

= 11

Cost (C, E) = min {C (E, D) + cost (C, D) }

= min {1 + 0}

= 01

Cost (B, G) = min {C (G, C) + cost (C, C) }

= min {5 + 10}

= 15

Cost (C, C) = min {C (C, D) + cost (D, D) }

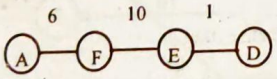
= min {10 + 0}

= 10

$$\text{Cost (A, A)} = \min \begin{Bmatrix} C(A,B) + \text{cost (B, B)} \\ C(A,F) + \text{cost (B, F)} \\ C(A,G) + \text{cost (B, G)} \end{Bmatrix}$$

$$= \min \begin{Bmatrix} 5+22 \\ 6+11 \\ 7+15 \end{Bmatrix}$$

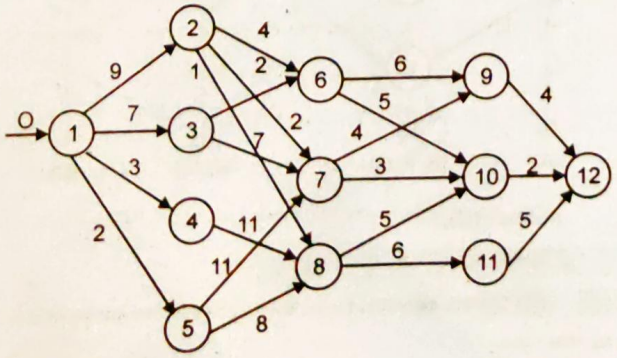
cost (A, A) = 11



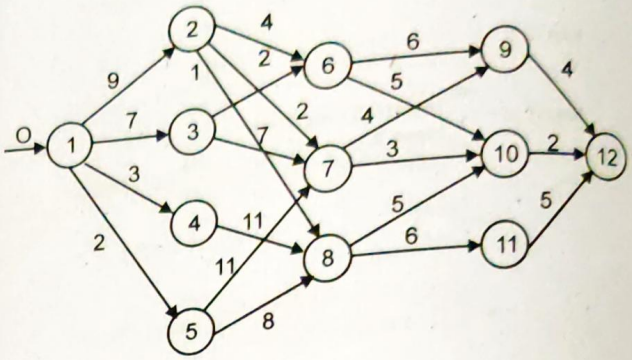
= 17

We can apply the principle of optimality because the path itself can be obtained easily if the decision at each stage is recorded in same array.

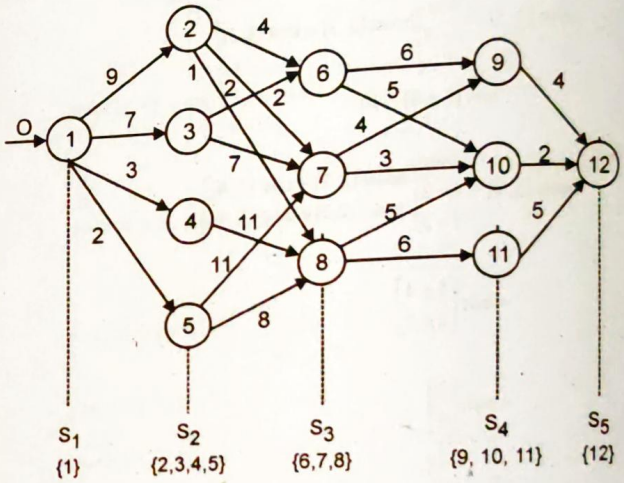
Q.8. Explain how the backtracking is useful in solving the following multistage graph. Use backward approach. *CS : S-12(8M)*



OR Implement shortest path algorithm using backward algorithm on following multistage graph. Also write an algorithm for the same. Explain the calculation steps. *CT : S-14(9M)*



Ans.



Stage 1 :

bcost (1, 1) = 0

Stage 2 :

$$\text{bcost (2, 2)} = \min_{l=1} [\text{bcost (l, 1)} + \text{cost (l, 2)}]$$

= 9

bcost (2, 3) = 7

bcost (2, 4) = 3

bcost (2, 5) = 2

Stage 3 :

$$\text{bcost (3, 6)} = \min_{l=2,3} \begin{bmatrix} \text{bcost (2, 2)} + \text{cost (2, 6)} \\ \text{bcost (2, 3)} + \text{cost (3, 6)} \end{bmatrix}$$

= 9

$$bcost(3,7) = \min_{l=2,3,5} \begin{bmatrix} bcost(2,2) + cost(2,7) \\ bcost(2,3) + cost(3,7) \\ bcost(2,5) + cost(5,7) \end{bmatrix}$$

$$= 11$$

$$bcost(3,8) = \min_{l=2,4,5} \begin{bmatrix} bcost(2,2) + cost(2,8) \\ bcost(2,4) + cost(4,8) \\ bcost(2,5) + cost(5,8) \end{bmatrix}$$

$$= 10$$

Stage 4 :

$$bcost(4,9) = \min_{l=6,7} \begin{bmatrix} bcost(3,6) + cost(6,9) \\ bcost(3,7) + cost(7,9) \end{bmatrix}$$

$$= 15$$

$$bcost(4,10) = \min_{l=6,7,8} \begin{bmatrix} bcost(3,6) + cost(6,10) \\ bcost(3,7) + cost(7,10) \\ bcost(3,8) + cost(8,10) \end{bmatrix}$$

$$= 14$$

$$bcost(4,11) = \min_{l=8} [bcost(3,8) + cost(8,11)]$$

$$= 16$$

Stage 5 :

b cos t (5, 12)

$$= \min_{l=9,10,11} \begin{bmatrix} bcost(4,9) + cost(9,12) \\ bcost(4,10) + cost(10,12) \\ bcost(4,11) + cost(11,12) \end{bmatrix}$$

$$= \min \begin{bmatrix} 15+4 \\ 14+2 \\ 16+5 \end{bmatrix}$$

bcost (5, 12) = 16

Shortest path from source to destination is of length 16

Path : $1 \overset{9}{} 2 \overset{2}{} 7 \overset{3}{} 10 \overset{2}{} 12$

= 16

ALL PAIRS SHORTEST PATH

Q.9. Explain all pairs shortest path method.

Ans. All pairs shortest path :

- The basic objective is to find shortest path from a vertex to all possible destination, assuming all the vertices as source vertex.
- To simplify calculation in each step one vertex will be added as intermediate and initially only direct paths are considered.
- All pair shortest path method computes the shortest path between each pair of vertices in a weighted directed graph. The problem is efficiently solved by Floyd's Warshall algorithm.
- The all-pairs shortest path problem can be considered as the mother of all routing problems.
- Using standard single source algorithms, we can expect to get a naive implementation of $O(n^3)$.
- To solve all pair shortest path algorithm on an input adjacency matrix, we need to compute not only the shortest path weight but also a predecessor matrix $\pi = (\pi_{ij})$ where π_{ij} is NIL. If either i = j or there is no path from i to j and an otherwise π_{ij} is some predecessor of j on a shortest path from i.

Q.10. Discuss Floyd-Warshall algorithm for all pairs shortest path.

CT : S-06,08(8M), W-10(3M)

OR Write an algorithm for all pairs shortest path.

CS : W-11(2M), CT : S-14(3M)

OR Write Floyd's algorithm. *CS : S-11(3M)*

OR Give an algorithm to find shortest path between all pairs of a given graph using dynamic programming.

CT : W-08, S-09(6M)

Ans. Floyd Warshall algorithm :

- The Floyd Warshall algorithm takes the dynamic programming approach. This essentially means that independent subproblems are solved and the results are stored for later use.
- The algorithm allows "negative edges" but no negative cycles, as per usual with such shortest path problems.
- The basic concept is simple. If for path (u, v) and its length estimate D [u] [v] if we can take a detour via W and shorten the path, then it should be taken. This translates into the following equation :

  D [u], [v] = min (D [u] [v], D [u] [w] + D [w] [v])
- The easiest way to calculate this is bottom up. Always start with basic assumptions "n" times for each vertex pair and the results will coverge.

Algorithm APSP (cost, n : A)

```
{
    for i = 1 to n do
        A [i, j] = cost [i, j]
    for K = 1 to n do
        for i = 1 to n do
            for j = 1 to n do
            A [i, j] = min [A [i, j], A [i, k] + A [k, j]]
}
```

Explanation of algorithm :

Step 1 : Initially set the weight given on the edges in row which is equal to number of vertices present in the graph.

Step 2 : Initialize matrix $D^{(0)}$.

Step 3 : We apply for loop

i.e. for k = 1 to n do

for i = 1 to n do

for j = 1 to n do

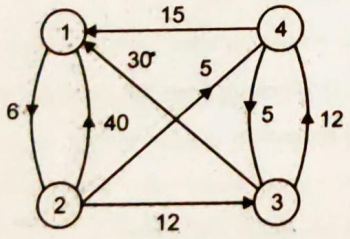
A (i, j) = min [A [i, j], A [i, k] + A [k, j]]

A [i, j] is the output

Cost [n × n] is input.

Considering one vertex as intermediate in each step we get the updated entries.

Q.11. Using all pair shortest path algorithm find the cost and the shortest path between all pair of nodes for the following graph when its length are given : *CT : W-06(8M)*



Ans. Four vertices | Five matrices

$$A_0 = \begin{matrix} \\ 1 \\ 2 \\ 3 \\ 4 \end{matrix} \begin{matrix} 1 & 2 & 3 & 4 \\ \end{matrix} \begin{bmatrix} 0 & 6 & \infty & \infty \\ 40 & 0 & 12 & 5 \\ \infty & \infty & 0 & 12 \\ 15 & \infty & 5 & 0 \end{bmatrix}$$

A_1 = use vertex 1 as intermediate.

$$A_1 = \begin{bmatrix} 0 & 6 & \infty & \infty \\ 40 & 0 & 10 & 5 \\ 30 & 33 & 0 & 12 \\ 15 & 21 & 5 & 0 \end{bmatrix}$$

A_2 = use vertex 1 and 2 as intermediate.

$$A_2 = \begin{bmatrix} 0 & 6 & 18 & 11 \\ 40 & 0 & 10 & 5 \\ 30 & 33 & 0 & 12 \\ 15 & 21 & 5 & 0 \end{bmatrix}$$

A_3 = use vertex 1, 2 and 3 as intermediate.

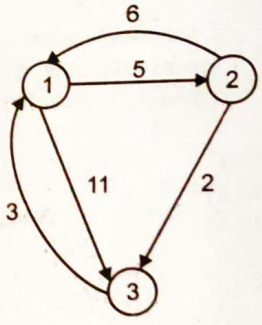
$$A_3 = \begin{bmatrix} 0 & 6 & 18 & 11 \\ 40 & 0 & 10 & 5 \\ 27 & 33 & 0 & 12 \\ 15 & 21 & 5 & 0 \end{bmatrix}$$

A_4 = use vertex 1 and 2, 3 and 4 as intermediate.

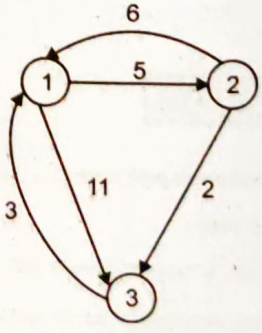
$$A_4 = \begin{bmatrix} 0 & 6 & 16 & 11 \\ 20 & 0 & 10 & 5 \\ 21 & 33 & 0 & 12 \\ 15 & 21 & 5 & 0 \end{bmatrix}$$

Final matrix.

Q.12. Find shortest path between all pairs of a given graph using dynamic programming. *CT : W-07(6M)*



Ans. Complexity of all pair shortest path algorithm is $O(n^3)$



Direct path matrix (3 × 3)

$$A_0 = \begin{bmatrix} 0 & 5 & 11 \\ 6 & 0 & 2 \\ 3 & \infty & 0 \end{bmatrix}$$

A_1 = use vertex 1 as intermediate

$$A_1 = \begin{bmatrix} 0 & 5 & 11 \\ 6 & 0 & 2 \\ 3 & 7 & 0 \end{bmatrix}$$

A_2 = use vertex V_1 and V_2 as intermediate.

$$A_2 = \begin{bmatrix} 0 & 5 & 6 \\ 6 & 0 & 2 \\ 3 & 7 & 0 \end{bmatrix}$$

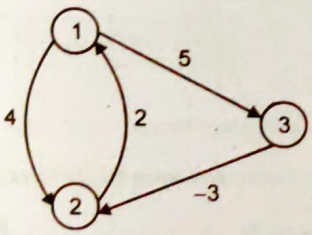
A_3 = use vertex V_1, V_2 and V_3 as intermediate.

$$A_3 = \begin{bmatrix} 0 & 5 & 6 \\ 5 & 0 & 2 \\ 3 & 7 & 0 \end{bmatrix}$$

The final matrix will allow any vertex as source vertex and remaining vertices as destination.

Q.13. Generate distance matrix for the following graph.

CT : W-10(4M)



Ans. $$A_0 = \begin{bmatrix} 0 & 4 & 5 \\ 2 & 0 & \infty \\ \infty & -3 & 0 \end{bmatrix}$$

Use V_1 as source vertex

$$A_1 = \begin{bmatrix} 0 & 4 & 5 \\ 2 & 0 & 7 \\ \infty & -3 & 0 \end{bmatrix}$$

Use V_1 and V_2 as intermediate

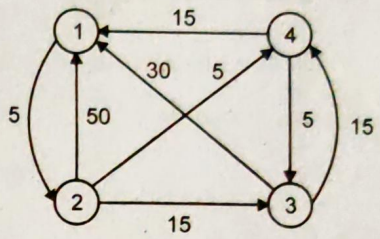
$$A_2 = \begin{bmatrix} 0 & 4 & 5 \\ 2 & 0 & 7 \\ -1 & -3 & 0 \end{bmatrix}$$

Use V_1, V_2 and V_3 as intermediate

$$A_3 = \begin{bmatrix} 0 & 4 & 5 \\ 2 & 0 & 7 \\ -1 & -3 & 0 \end{bmatrix}$$

Q.14. How Floyds algorithm works for the following graph.

CS : S-11, W-11(3M)



Ans. $$A_0 = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & \infty & 0 & 15 \\ 15 & \infty & 5 & 0 \end{bmatrix}$$

Use V_1 as intermediate.

$$A_1 = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

Use vertex V_1 and V_2 as intermediate.

$$A_2 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 50 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

Use V_1, V_2 and V_3 as intermediate.

$$A_3 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 45 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

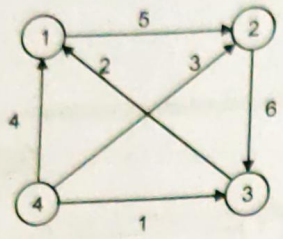
Use V_1, V_2, V_3, V_4 as intermediate.

$$A_4 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 20 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

VBD

Q.15. Find all pair shortest path using Floyd Warshall algorithm for given graph.

CT : W-12(9M)



Ans.

$$A_0 = \begin{bmatrix} 0 & 5 & \infty & 4 \\ 0 & 0 & 6 & \infty \\ 2 & \infty & 0 & \infty \\ \infty & 3 & 1 & 0 \end{bmatrix}$$

$$A_1 = \begin{bmatrix} 0 & 5 & \infty & 4 \\ 0 & 0 & 6 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 3 & 1 & 0 \end{bmatrix}$$

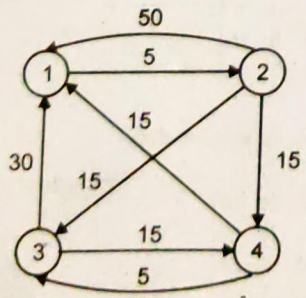
$$A_2 = \begin{bmatrix} 0 & 5 & 0 & 4 \\ 0 & 0 & 6 & 0 \\ 1 & 0 & 0 & 0 \\ 3 & 3 & 1 & 0 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 5 & 0 & 4 \\ 8 & 0 & 6 & 0 \\ 1 & 0 & 0 & 0 \\ 3 & 3 & 1 & 0 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} 0 & 5 & 0 & 4 \\ 8 & 0 & 6 & 0 \\ 1 & 0 & 0 & 0 \\ 3 & 3 & 1 & 0 \end{bmatrix}$$

Q.16. Generate all pair shortest path in the following graph given :

CS : S-12(6M), CT : S-14(3M)



Ans.

$$A_0 = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 15 \\ 30 & \infty & 0 & 15 \\ 15 & \infty & 5 & 0 \end{bmatrix}$$

Use vertex 1 as intermediate.

$$A_1 = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 15 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

Use vertex 1, 2 as intermediate.

$$A_2 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 50 & 0 & 15 & 15 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

Use vertex 1, 2, 3 as intermediate.

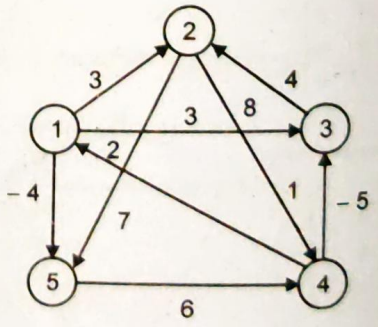
$$A_3 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 45 & 0 & 15 & 15 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

Use vertex 1, 2, 3 and 4 as intermediate.

$$A_4 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 20 & 0 & 15 & 15 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

Q.17. Find all pair shortest path using Floyds Warshall algorithm for given directed graph.

CT : W-13(6M)



Ans.

$$A_0 = \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

VBD

$$A_1 = \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{bmatrix}$$

$$A_5 = \begin{bmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{bmatrix}$$

SINGLE SOURCE SHORTEST PATH

Q.18. Explain single source shortest path method.

Ans. Given a graph G = (V, E) we want to find a shortest path from a given source vertex $S \in V$ to every vertex $v \in V$.

There are two methods of single source shortest path method.

(1) Dijkstra's Algorithm : Dijkstra's algorithm solves the single source shortest path problem when all edges have nonnegative edges.

(2) Bellman Ford Algorithm : Bellman ford algorithm solves the single source shortest path problem in the general case in which edges of a given digraph can have negative weight as long as G contains no negative cycles.

Q.19. Give Dijkstra's algorithm for shortest path and explain it. Also discuss its time complexity. **CT : S-11(5M)**

Ans. Dijkstra's Algorithm :

Dijkstra (G, W, S)

Step 1 : Initialize - single source (G, S)

Step 2 : S ← { }

Step 3 : Initialize priority Queue Q i.e. Q ← V [G]

Step 4 : While priority Queue Q is not empty

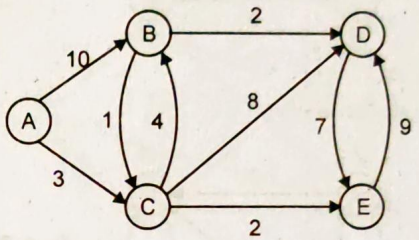
Step 5 : do u ← EXTRACT-MIN(Q)

Step 6 : S ← S U {u}

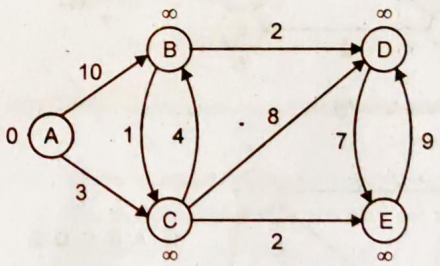
Step 7 : for each vertex V ∈ Adj[u]

Step 7 : do RELAX (u, v, w)

Example :

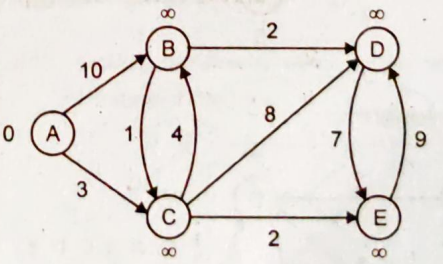


Initialize :



Q : A B C D E
0 ∞ ∞ ∞ ∞

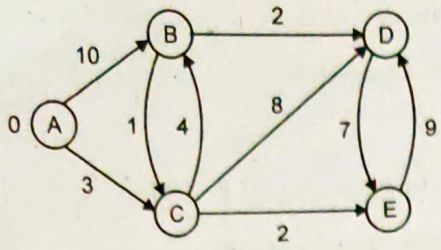
"A" ← EXTRACT-Min (Q)



Q : A B C D E
0 ∞ ∞ ∞ ∞

S : {A}

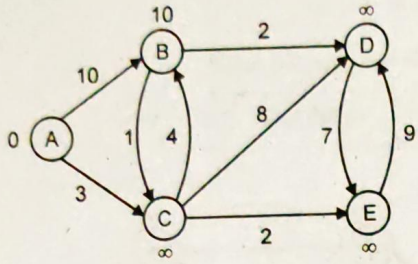
Relax all edges leaving A :



Q : A B C D E
0 ∞ ∞ ∞ ∞
10 3

S : {A}

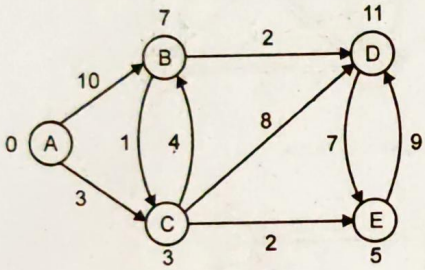
"C" ← Extract-Min (Q)



Q : A B C D E
0 ∞ ∞ ∞ ∞
10 3

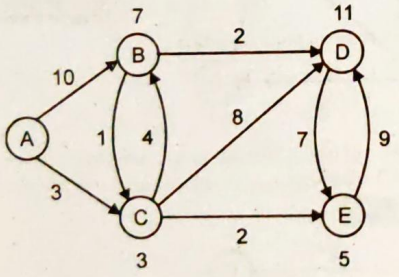
S : {A, C}

Relax all edges leaving C



Q : A B C D E
0 ∞ ∞ ∞ ∞
10 3 - -
7

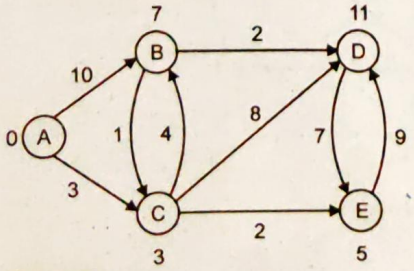
E ← EXTRACT-Min (Q) :



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

S : {A, C, E}

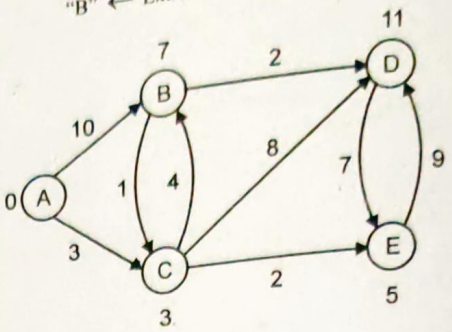
Relax all edges leaving E :



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

S : {A, C, E}

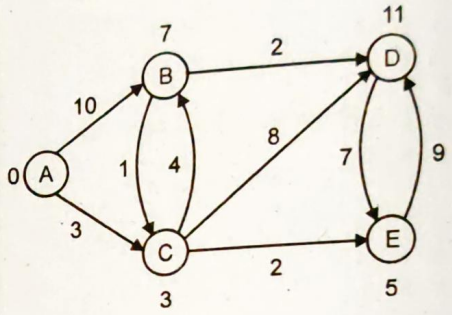
"B" ← Extract- Min (Q)



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

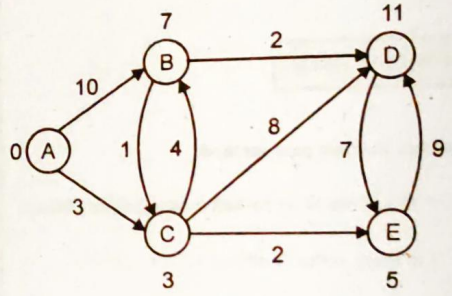
S : {A, C, E, B}

Relax all edges leaving B :



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

"D" ← EXTRACT-Min (Q) :



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

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

Time complexity : The time complexity of Dijkstra algorithm is $O(|V|^2 + |E|) = O(V^2)$.

Q.20. Write Bellman Ford algorithm. *CS : S-10(3M), W-12(5M)*

OR Explain with example the Bellman ford algorithm.

Ans. Bellman Ford algorithm :

Bellman Ford (G, W, S)

Step 1 : Initialize - single source (G, S)

Step 2 : For each vertex i = 1 to V [G] – 1 do

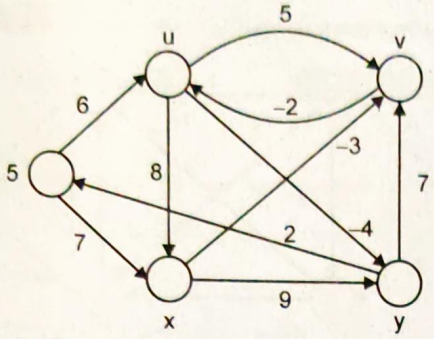
Step 3 : For each edge (u, v) ∈ E (G) do

Step 4 : Relax (u, v, w)

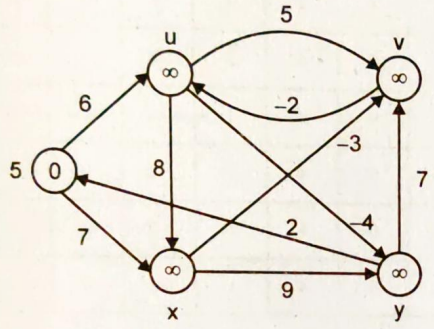
Step 5 : For each edge (u, v) in E (G)

Step 6 : Do if d [u] + w (u, v) < d [v]
Step 7 : Then return false
Step 84 : Return true

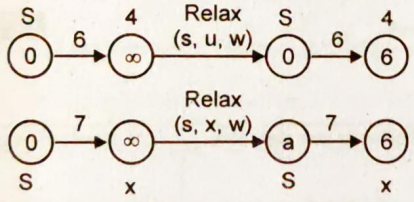
Example : Step - by - step execution of Bellman ford algorithm.

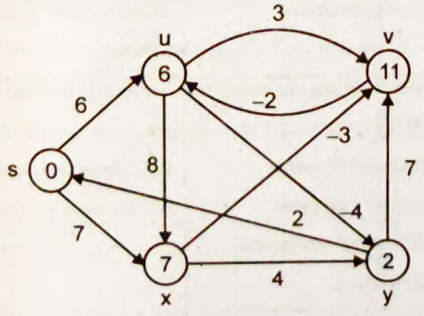


Step 1 : Given initial graph G = (V, E) let all nodes have infinite cost except source node S, which has O cost.

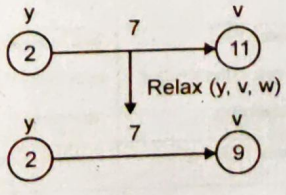


For each vertex apply RELAX (u, v, w)

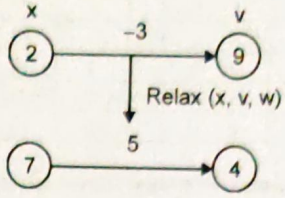




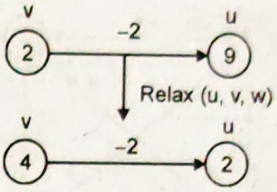
Step 3 : Apply Relax (y, v, w)



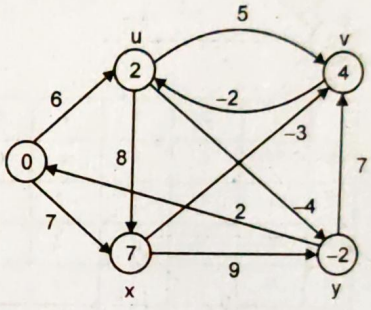
Step 4 : Again apply Relax (x, v, w)



Step 5 : Apply Relax (v, u, w)



Step 6 : Apply Relax (u, y, w)



Shortest path from vertex s to y is

$$d_{sy} = d_{sx} + d_{xv} + d_{vu} + d_{uy}$$

$$= 7 + (-3) + (-2) + (-4) = -2$$

Q.21. What are the main characteristics of Bellman Ford algorithm?

CS : W-10(2M)

Ans. Characteristics of Bellman Ford algorithm :

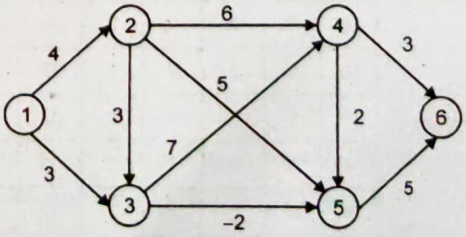
(1) This algorithm find shortest path from one source to all possible destination for a given graph.

(2) In the graph negative edges are permitted.

(3) This algorithm provide the information about parent vertex used to reach destination.

Q.22. Generate the distance matrix for the following graph using Bellman ford algorithm.

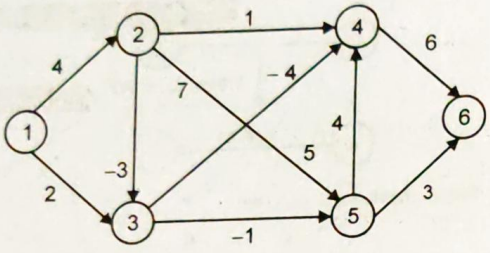
CS : S-10(3M)



Ans. Distance matrix :

↓ length		1	2	3	4	5	6
	1	0	4	3	∞	∞	∞
	2	0	4	3	10	1	∞
	3	0	4	3	10	1	6
	4	0	4	3	10	1	6
	5	0	4	3	10	1	6

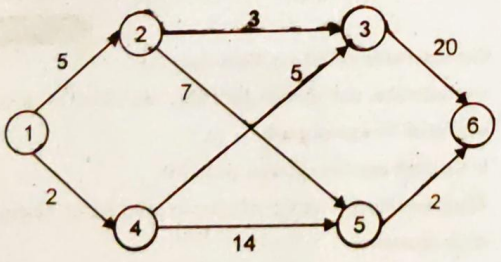
Q.23. Write algorithm and show three execution steps for the following : **CS : W-10(2M)**



Ans. Distance matrix :

↑ length		1	2	3	4	5	6
	1	0	4	2	∞	∞	∞
	2	0	4	1	−2	1	∞
	3	0	4	1	−2	0	4
	4	0	4	1	−2	0	4
	5	0	4	1	−2	0	4

Q.24. Find the shortest path from node 1 to node 6 in the given graph using Bellman Ford algorithm. What is its computing time? **CT : W-11(3M)**

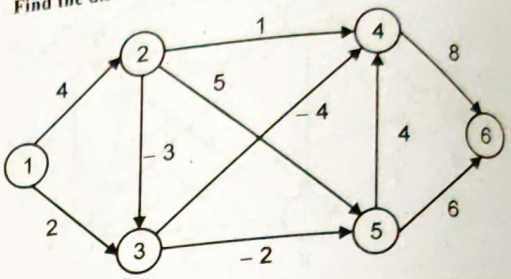


Ans.

↑ length		1	2	3	4	5	6
	1	0	5	∞	2	∞	∞
	2	0	5	7	2	12	∞
	3	0	5	7	2	12	14
	4	0	5	7	2	12	14
	5	0	5	7	2	12	14

- Complexity i.e computing time for algorithm is $O(V^2)$.

Q.25. Implement Bellman Ford algorithm on the following graph. Find the distance matrix. **CS : W-13(6M)**



Ans.

	1	2	3	4	5	6
1	0	4	2	∞	∞	∞
2	0	4	1	−2	0	∞
3	0	4	1	−2	0	6
4	0	4	1	−2	0	3
5	0	4	1	−2	0	3

Q.26. Compare minimum cost and shortest path algorithm with example. **CT : W-10(6M)**

Ans.

Sr. No.	Minimum cost algorithm	Shortest path algorithm
(1)	Graph is undirected.	In shortest path, graph is directed.
(2)	Methods to find minimum cost are (i) Kruskal algorithm (ii) Prim's algorithm (iii) Optimal randomized algorithm	Methods to find shortest path are (i) Bellman ford algorithm (ii) Dijkstra's algorithm.
(3)	Technique of relaxation is not applied.	Technique of relaxation is applied.
(4)	**For example :** Find minimum cost using Kruskal or Prim's algorithm.	**For example :** Find shortest path using Dijkstra's algorithm.
(5)	Running time $\theta(E \log v)$.	Running time $O(\|V\| \|E\|)$.

OPTIMAL BINARY SEARCH TREE

Q.27. What is optimal binary search tree? How it is different from normal binary tree? How optimal binary search tree is useful for information storage? **CT : S-06(6M), S-08(4M)**

Ans. OBST :

- Optimal Binary search tree is a binary search tree with an optimal cost. OBST is used to find the optimal cost of the tree and construct the tree having minimum cost.
- To construct the optimal binary search tree three matrices are generated.

(i) W = Probability matrix.

(ii) E = Evolutionary matrix.

(iii) R = Root matrix.

Cost : $[\Sigma P_i \times \text{depth} = \Sigma q_i \times \text{depth}]$

Formula :

(1) $$W[i, j] = \begin{cases} q[i-1] & \text{if } j = i-1 \\ W[i, j-1] + p_j + q_j & \text{otherwise} \end{cases}$$

(2) $$e[i, j] = \begin{cases} q[i-1] & \text{if } j = i-1 \\ e[i, r-1] + e[r+1, j] + w[i, j] & \text{otherwise} \end{cases}$$

- OBST is different from normal binary tree in following ways :

(i) Binary search tree is drawn by considering a first value of given information as root but in OBST we have to find the root value from given set of information.

(ii) BST is a static approach whereas OBST is a dynamic approach.

Q.28. Explain the difference between BST and OBST. **CS : W-11(2M)**

Ans.

Sr. No.	BST	OBST
(1)	BST is drawn by considering a first value of given information as root.	OBST find the root value from given set of information.
(2)	BST is static approach.	OBST is a dynamic approach.
(3)	Tree may be unbalanced.	Tree is balanced.
(4)	Searching cannot be performed in time.	Searching is performed in time.

Q.29. Write an algorithm for optimal binary search tree. Discuss its time complexity. **CT : S-08(4M)**

OR Give a dynamic programming formulation the optimal binary search tree organization. Give algorithm also.

CT : W-06(6M), W-08, S-10(8M)

OR Explain how dynamic programming can be applied to OBST.

CT : S-13(4M)

Ans.

```
Algorithm OBST (p, q, n : w, e, root)
{
  for i = 1 to n do
  {
    w [i, i – 1] = q [i – 1]
    e [i, i – 1] = q [i – 1]
  }
  for l = 1 to n do
    for i = 1 to n – l + 1 do
    {
      j = i + l – 1
      e [i, j] = 99
      w [i, j] = w [i, j – 1] + p [j] + q [j]
        for r = i to j
        {
          t = e [i, r – j] + e [r + 1, j] + w [i, j]
            if (t < e [i, j])
              {
                e [i, j] = t
                root [i, j] = r
              }
        }
    }
}
```

Q.30. For the following set of information design the matrix required to construct optimal BST. Draw OBST and find cost of the tree.

CS : W-11(7M)

i	0	1	2	3	4	5
pi	-	0.12	0.10	0.07	0.08	0.20
qi	0.07	0.11	0.05	0.05	0.05	0.10

Ans. $\sum pi + qi = 0.07\ 0.23\ 0.15\ 0.12\ 0.13\ 0.30$

$$W = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0.07 & 0.30 & 0.45 & 0.57 & 0.70 & 1.00 \\ & 0.11 & 0.26 & 0.38 & 0.51 & 0.81 \\ & & 0.05 & 0.17 & 0.30 & 0.60 \\ & & & 0.05 & 0.18 & 0.48 \\ & & & & 0.05 & 0.35 \\ & & & & & 0.10 \end{bmatrix} \end{array}$$

$$e = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0.07 & 0.47 & 0.94 & 1.31 & 1.79 & \boxed{2.5} \\ & 0.11 & 0.42 & 0.76 & 1.21 & 1.79 \\ & & 0.05 & 0.27 & 0.62 & 1.21 \\ & & & 0.05 & 0.28 & 0.56 \\ & & & & 0.05 & 0.50 \\ & & & & & 0.10 \end{bmatrix} \end{array}$$

Cost of tree = 2.5

Q.31. Construct the OBST on following keys. Draw the tree and again find the cost of tree. **CS : W-12(8M)**

i	0	1	2	3	4
pi	-	0.14	0.16	0.10	0.15
qi	0.05	0.16	0.10	0.05	0.15

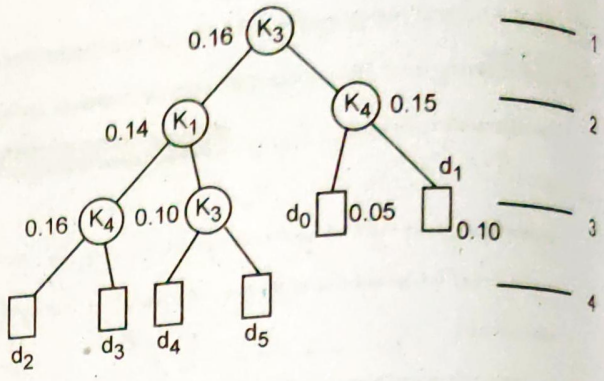
Ans. $\sum pi + qi = 0.05\ 0.24\ 0.26\ 0.15\ 0.30$

$$W = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0.05 & 0.29 & 0.55 & 0.70 & 1.00 \\ & 0.10 & 0.36 & 0.51 & 0.81 \\ & & 0.10 & 0.25 & 0.55 \\ & & & 0.05 & 0.35 \\ & & & & 0.15 \end{bmatrix} \end{array}$$

$$e = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0.05 & 0.44 & 1.09 & 1.54 & \boxed{2.54} \\ & 0.10 & 0.56 & 1.01 & 2.92 \\ & & 0.10 & 0.40 & 1.10 \\ & & & 0.05 & 0.55 \\ & & & & 0.15 \end{bmatrix} \end{array}$$

Cost of tree = 2.54

Tree :



Q.32. Draw OBST for the following. Also generate the matrices required for tree construction. **CS : W-11(6M)**

i	0	1	2	3	4	5
pi	-	0.08	0.05	0.12	0.20	0.10
qi	0.05	0.10	0.05	0.11	0.10	0.04

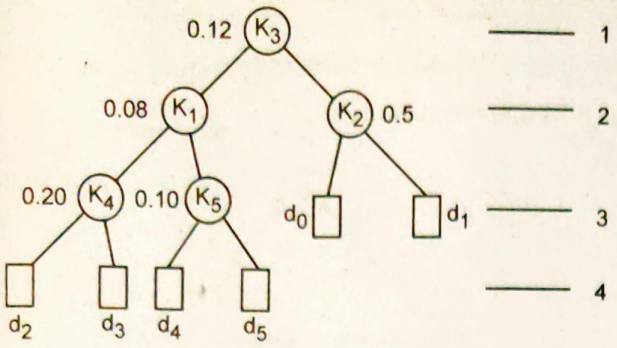
Ans. $\sum pi + qi = 0.05\ 0.18\ 0.10\ 0.23\ 0.30\ 0.14$

$$W = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0.05 & 0.23 & 0.33 & 0.56 & 0.86 & 1.00 \\ & 0.10 & 0.20 & 0.43 & 0.73 & 0.82 \\ & & 0.05 & 0.28 & 0.58 & 0.72 \\ & & & 0.11 & 0.31 & 0.45 \\ & & & & 0.10 & 0.24 \\ & & & & & 0.04 \end{bmatrix} \end{array}$$

$$e = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0.05 & 0.38 & 0.73 & 1.35 & 2.33 & \boxed{2.67} \\ & 0.10 & 0.35 & 0.74 & 1.42 & 1.94 \\ & & 0.05 & 0.44 & 1.12 & 1.54 \\ & & & 0.11 & 0.52 & 0.94 \\ & & & & 0.10 & 0.38 \\ & & & & & 0.04 \end{bmatrix} \end{array}$$

Cost of tree = 2.67

$$r = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 3 & 3 \\ & 2 & 2 & 3 & 3 \\ & & 3 & 2 & 2 \\ & & & 4 & 1 \\ & & & & 5 \end{bmatrix} \end{array}$$



Q.33. Determine the cost and structure of an OBST for a set of n = 6 keys with probabilities.

i	0	1	2	3	4	5	6
pi	-	0.04	0.06	0.08	0.02	0.10	0.26
qi	0.06	0.06	0.06	0.06	0.05	0.05	0.10

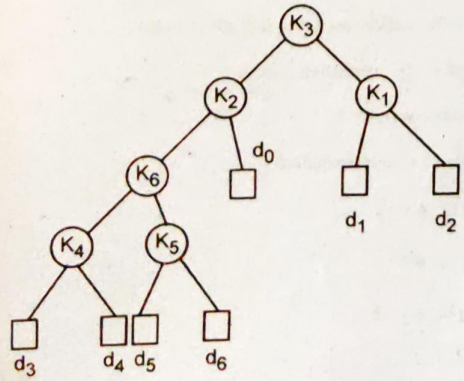
CT : S-12(13M)

Ans.

$$W = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 0.06 & 0.16 & 0.28 & 0.42 & 0.49 & 0.64 & 1.00 \\ & 0.06 & 0.18 & 0.32 & 0.39 & 0.54 & 0.90 \\ & & 0.06 & 0.20 & 0.27 & 0.42 & 0.78 \\ & & & 0.06 & 0.12 & 0.27 & 0.63 \\ & & & & 0.05 & 0.20 & 0.56 \\ & & & & & 0.05 & 0.46 \\ & & & & & & 0.10 \end{bmatrix}$$

$$e = \begin{bmatrix} 0.06 & 0.16 & 0.44 & 0.78 & 1.04 & 1.47 & 2.46 \\ & 0.06 & 0.18 & 0.50 & 0.69 & 1.11 & 1.84 \\ & & 0.06 & 0.20 & 0.39 & 0.81 & 1.79 \\ & & & 0.06 & 0.12 & 0.39 & 1.02 \\ & & & & 0.05 & 0.20 & 0.76 \\ & & & & & 0.05 & 0.46 \\ & & & & & & 0.10 \end{bmatrix}$$

Cost of OBST = 2.46



Q.34. For the following probability value implement OBST. Reconstruct the tree and verify the results. *CS : S-13(7M)*

i	0	1	2	3	4
pi	-	0.10	0.14	0.09	0.12
qi	0.14	0.12	0.13	0.10	0.06

Ans. $\sum pi + qi =$ 0.14 0.22 0.27 0.19 0.16

$$W = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 0.14 & 0.36 & 0.63 & 0.82 & 1.00 \\ & 0.12 & 0.39 & 0.58 & 0.76 \\ & & 0.13 & 0.32 & 0.5 \\ & & & 0.10 & 0.28 \\ & & & & 0.06 \end{bmatrix}$$

$$e = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 0.14 & 0.62 & 1.38 & 1.99 & 2.69 \\ & 0.12 & 0.64 & 1.25 & 1.84 \\ & & 0.13 & 0.55 & 1.07 \\ & & & 0.10 & 0.44 \\ & & & & 0.06 \end{bmatrix}$$

Cost of the tree = 2.69

For successful search

$(0.09 \times 1) + 2\,(0.14 \times 0.12)$

$+ 3\,(0.10) = 0.89$

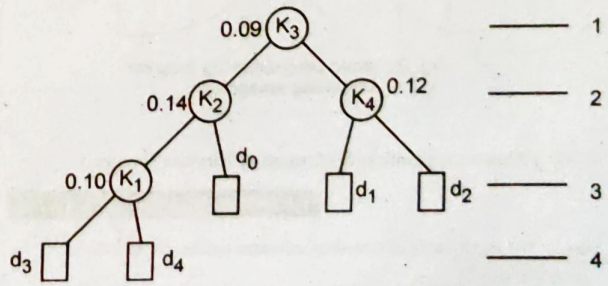
For unsuccessful search

$3\,(0.14 + 0.12 + 0.13) + 4\,(0.10 + 0.06)$

$= 1.80$

Successful + unsuccessful search

= 2.69 i.e. cost of tree



TRAVELLING SALESMAN PROBLEM

Q.35. Explain the method of implementing travelling salesman problem. Give the recurrence relation. *CT : S-09(5M)*

OR What is travelling salesman problem?

CS : S-11(3M), S-14, W-13(2M)

OR Explain how dynamic programming technique can be applied to solve travelling salesman problem. *CT : W-08(6M), S-11(7M)*

Ans. Travelling salesman problem :

- Travelling salesman problem is implemented on complete graph, using this method the basic objective is to find shortest path between source vertex visting all the vertices in the graph and communicating at source vertex.
- If there are n vertices in graph then there will be n + 1 vertices in their path, first and last vertex repeated.
- In greedy method travelling salesman problem is solved by exhaustive search method and found that the problem is untraceable as n! permutation for the path are considered.

Dynamic programming approach to the travelling salesman problem :

- Assume that the tour starts at node no. 1 and terminates on the same node.
- Every possible tour can be viewed as consisting of edge (1, k) for path from k to 1 goes exactly through each node in V = {1, k} exactly once.
- It is clear that if we are going to have an optimal tour then the path [k, 1] should also be optimal. Thus the principle of optimality holds.

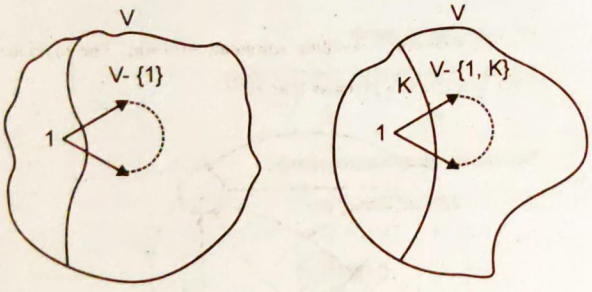


Fig. Dynamic programming step for travelling salesman

Q.36. What is the significance of traveling salesman problem?

CT : W-10(2M), CS : W-10,S-12(2M)

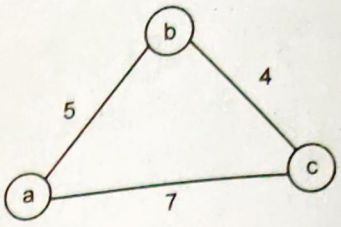
Ans. The significance of travelling salesman problem is as follows :

(1) To find the shortest path from a source vertex to all the vertices and travelling back to source i.e. generating a cycle from source vertex. The path should be such that all the vertices are covered in minimum distance. This concept is applicable on directed and complete graph.

(2) The graph is directed.

(3) Edge weight satisfy the triangle inequality.

$w(a, b) + w(b, c) \geq w(a, c)$



$w(a, b) = 5,$

$w(b, c) = 4,$

$w(a, c) = 7$

$w(a, b) + w(b, c)$

$5 + 4$

$9 > w(a, c)$

$9 > 7.$

Q.37. Find a tour of minimum length starting from vertex 1 going through each vertex only once and coming back to vertex 1. The length matrix and the graph is given as follows :

CT : W-06(6M)

	1	2	3	4
1	0	5	2	3
2	2	0	4	1
3	6	3	0	5
4	2	1	6	0

Ans. Formula :

$$g(i, s) = \min_{j \in s} [Cij + g(j, (s - \{j\}))]$$

where i : Source vertex

j : Vertex used as intermediate

s : The vertices set which will not included i

and s – {j} represent the vertices.

Source vertex = 1

Step 1 : No intermediate

$g(2, \phi) = 2$

$g(3, \phi) = 6$

$g(4, \phi) = 2$

Step 2 : One intermediate

$g(2, \{3\}) = C_{ij} + g(j, \varphi)$

$= C_{23} + g(3, \varphi) = 4 + 6 = 10$

$g(2, \{4\}) = C_{24} + g(4, \varphi)$

$= 1 + 2 = 3$

$g(3, \{2\}) = C_{32} + g(2, \varphi)$

$= 3 + 2 = 5$

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

$= 5 + 2 = 7$

$g(4, \{2\}) = C_{42} + g(2, \varphi)$

$= 1 + 2 = 3$

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

$= 6 + 6 = 12$

Step 3 : Two Intermediate

$$g(2, \{3, 4\}) = \min \begin{bmatrix} C_{23} + g(3, \{4\}) \\ C_{24} + g(4, \{3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 4 + 7 \\ 1 + 12 \end{bmatrix}$$

$$= \min \begin{bmatrix} 11 \\ 13 \end{bmatrix}$$

$= 11$

$$g(3, \{2, 4\}) = \min \begin{bmatrix} C_{32} + g(2, \{4\}) \\ C_{34} + g(4, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 3 + 3 \\ 5 + 6 \end{bmatrix}$$

$$= \min \begin{bmatrix} 6 \\ 8 \end{bmatrix}$$

$= 6$

$$g(4, \{2, 3\}) = \min \begin{bmatrix} C_{42} + g(2, \{3\}) \\ C_{43} + g(3, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 1 + 10 \\ 6 + 5 \end{bmatrix}$$

$$= \min \begin{bmatrix} 11 \\ 11 \end{bmatrix}$$

$= 11$

Step 4 : Three Intermediate

$$g(1, \{2, 3, 4\}) = \min \begin{bmatrix} C_{12} + g(2, \{3, 4\}) \\ C_{13} + g(3, \{2, 4\}) \\ C_{14} + g(4, \{2, 3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 5 + 11 \\ 2 + 6 \\ 3 + 11 \end{bmatrix}$$

$$= \min \begin{bmatrix} 16 \\ 8 \\ 14 \end{bmatrix}$$

$g(1, \{2, 3, 4\}) = 8$

Path : $1 - 3 - 4 - 2 - 1$

$2 + 5 + 1 = 8$

Q.38. Implement TSP on following matrix.

$$\begin{bmatrix} 0 & 10 & 15 & 20 \\ 5 & 0 & 9 & 10 \\ 6 & 13 & 0 & 12 \\ 8 & 8 & 9 & 0 \end{bmatrix}$$

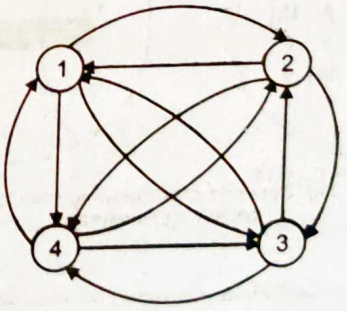
CT : W-10(6M), CS : S-14(4M)

OR Implement the TSP on the following matrix assuming vertex 2 as source vertex

$$\begin{bmatrix} 0 & 10 & 15 & 20 \\ 5 & 0 & 9 & 10 \\ 6 & 13 & 0 & 12 \\ 8 & 8 & 9 & 0 \end{bmatrix}$$

CS : S-12(5M)

OR Solve the following travelling salesman problem. The distance matrix and graph is given as follows :



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

CT : S-13(7M)

Ans. Step 1 :

$g(2, \phi) = 5$

$g(3, \phi) = 6$

$g(4, \phi) = 8$

Step 2 :

$g(2, \{3\}) = C_{23} + g(3, \phi) = 15$

$g(2, \{4\}) = C_{24} + g(4, \phi) = 18$

$g(3, \{2\}) = C_{32} + g(2, \phi) = 18$

$g(3, \{4\}) = C_{34} + g(4, \phi) = 20$

$g(4, \{2\}) = C_{42} + g(2, \phi) = 13$

$g(4, \{3\}) = C_{43} + g(3, \phi) = 15$

Step 3 :

$$g(2, \{3, 4\}) = \min\begin{bmatrix} 9+20 \\ 10+15 \end{bmatrix} = 25$$

$$g(3, \{2, 4\}) = \min\begin{bmatrix} 13+18 \\ 12+13 \end{bmatrix} = 25$$

$$g(4, \{2, 3\}) = \min\begin{bmatrix} 8+15 \\ 9+18 \end{bmatrix} = 23$$

$$g(1, \{2, 3, 4\}) = \min\begin{bmatrix} C_{12} + g(2, \{3, 4\}) \\ C_{13} + g(3, \{2, 4\}) \\ C_{14} + g(4, \{2, 3\}) \end{bmatrix}$$

$$= \min\begin{bmatrix} 10+25 \\ 15+25 \\ 20+23 \end{bmatrix} = 35$$

Path : 1 — 2 — 4 — 3 — 1

Q.39. Implement TSP for the following graph.

$$\begin{bmatrix} 0 & 8 & 16 & 15 \\ 14 & 0 & 9 & 12 \\ 7 & 10 & 0 & 6 \\ 11 & 13 & 10 & 0 \end{bmatrix}$$

CS : S-11(4M),W-13(5M)

Ans. Step 1 :

$g(2, \phi) = C_{21} = 14$

$g(3, \phi) = C_{31} = 7$

$g(4, \phi) = C_{41} = 11$

Step 2 :

$g(2, \{3\}) = C_{23} + g(3, \phi)$

$= 9 + 7 = 16$

$g(2, \{4\}) = C_{24} + g(4, \phi)$

$= 12 + 11 = 23$

$g(3, \{2\}) = C_{32} + g(2, \phi)$

$= 10 + 14 = 24$

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

$= 6 + 11 = 17$

$g(4, \{2\}) = C_{42} + g(2, \phi)$

$= 13 + 14 = 27$

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

$= 10 + 7 = 17$

Step 3 :

$$g(2, \{3, 4\}) = \min\begin{bmatrix} C_{23} + g(3, \{4\}) \\ C_{24} + g(4, \{3\}) \end{bmatrix}$$

$$= \min\begin{bmatrix} 9+17 \\ 12+17 \end{bmatrix}$$

$$= \min\begin{bmatrix} 26 \\ 29 \end{bmatrix} = 26$$

$$g(3, \{2, 4\}) = \min\begin{bmatrix} C_{32} + g(2, \{4\}) \\ C_{34} + g(4, \{2\}) \end{bmatrix}$$

$$= \min\begin{bmatrix} 10+23 \\ 6+27 \end{bmatrix}$$

$$= \min\begin{bmatrix} 33 \\ 33 \end{bmatrix} = 33$$

$$g(4, \{2, 3\}) = \min\begin{bmatrix} C_{42} + g(2, \{3\}) \\ C_{43} + g(3, \{2\}) \end{bmatrix}$$

$$= \min\begin{bmatrix} 13+16 \\ 10+24 \end{bmatrix}$$

$$= \min\begin{bmatrix} 29 \\ 34 \end{bmatrix} = 29$$

Step 4 : Three Intermediate :

$$g(1, \{2, 3, 4\}) = \min\begin{bmatrix} C_{12} + g(2, \{3, 4\}) \\ C_{13} + g(3, \{2, 4\}) \\ C_{14} + g(4, \{2, 3\}) \end{bmatrix}$$

$$= \min\begin{bmatrix} 8+26 \\ 16+33 \\ 15+29 \end{bmatrix}$$

$$= \min\begin{bmatrix} 34 \\ 49 \\ 44 \end{bmatrix}$$

$g(1, \{2, 3, 4\}) = 34$

Cost = 34

Q.40. Implement TSP on following matrix and find out shortest path.

$$\begin{bmatrix} 0 & 12 & 10 & 7 \\ 15 & 0 & 9 & 14 \\ 7 & 8 & 0 & 16 \\ 5 & 11 & 10 & 0 \end{bmatrix}$$

CS : W-12(5M)

Ans. **Step 1 :**

$g(2, \phi) = 15$

$g(3, \phi) = 7$

$g(4, \phi) = 5$

Step 2 :

$g(2, \{3\}) = C_{23} + g(3, \phi)$

$= 9 + 7 = 16$

$g(2, \{4\}) = C_{24} + g(4, \phi)$

$= 14 + 5 = 19$

$g(3, \{2\}) = C_{32} + g(2, \phi)$

$= 8 + 15 = 23$

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

$= 16 + 5 = 21$

$g(4, \{2\}) = C_{42} + g(2, \phi)$

$= 11 + 15 = 26$

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

$= 10 + 7 = 17$

Step 3 :

$$g(2, \{3, 4\}) = \min \begin{bmatrix} C_{23} + g(3, \{4\}) \\ C_{24} + g(4, \{3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 9 + 21 \\ 14 + 17 \end{bmatrix} = \begin{bmatrix} 30 \\ 31 \end{bmatrix} = 30$$

$$g(3, \{2, 4\}) = \min \begin{bmatrix} C_{32} + g(2, \{4\}) \\ C_{34} + g(4, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 + 19 \\ 16 + 26 \end{bmatrix} = \begin{bmatrix} 27 \\ 42 \end{bmatrix} = 27$$

$$g(4, \{2, 3\}) = \min \begin{bmatrix} C_{42} + g(2, \{3\}) \\ C_{43} + g(3, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 11 + 16 \\ 10 + 23 \end{bmatrix} = \begin{bmatrix} 27 \\ 33 \end{bmatrix} = 27$$

Step 4 :

$$g(1, \{2, 3, 4\}) = \min \begin{bmatrix} C_{12} + g(2, \{3, 4\}) \\ C_{13} + g(3, \{2, 4\}) \\ C_{14} + g(4, \{2, 3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 12 + 30 \\ 10 + 27 \\ 7 + 27 \end{bmatrix} = \begin{bmatrix} 42 \\ 37 \\ 34 \end{bmatrix}$$

$g(1, \{2, 3, 4\}) = 34$

Cost = 34

Q.41. Implement TSP on following matrix, assuming vertex 2 as source vertex.

$$\begin{bmatrix} 0 & 12 & 14 & 9 \\ 9 & 0 & 8 & 12 \\ 10 & 14 & 0 & 16 \\ 8 & 7 & 18 & 0 \end{bmatrix}$$

CS : W-10(4M)

Ans. Given vertex 2 as source vertex.

Step 1 :

$g(1, \phi) = 12$

$g(3, \phi) = 14$

$g(4, \phi) = 7$

Step 2 :

$g(1, \{3\}) = C_{13} + g(3, \phi)$

$= 14 + 14 = 28$

$g(1, \{4\}) = C_{14} + g(4, \phi)$

$= 9 + 7 = 16$

$g(3, \{1\}) = C_{31} + g(1, \phi)$

$= 10 + 12 = 22$

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

$= 16 + 7 = 23$

$g(4, \{1\}) = C_{41} + g(1, \phi)$

$= 8 + 12 = 20$

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

$= 18 + 14 = 32$

Step 3 :

$$g(1, \{3, 4\}) = \min \begin{bmatrix} C_{13} + g(3, \{4\}) \\ C_{14} + g(4, \{3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 14 + 23 \\ 9 + 32 \end{bmatrix} = \begin{bmatrix} 37 \\ 39 \end{bmatrix} = 37$$

$$g(3, \{1, 4\}) = \min \begin{bmatrix} C_{31} + g(1, \{4\}) \\ C_{34} + g(4, \{1\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 10 + 16 \\ 16 + 20 \end{bmatrix} = \begin{bmatrix} 26 \\ 36 \end{bmatrix} = 26$$

Q.40. Implement TSP on following matrix and find out shortest path.

$$\begin{bmatrix} 0 & 12 & 10 & 7 \\ 15 & 0 & 9 & 14 \\ 7 & 8 & 0 & 16 \\ 5 & 11 & 10 & 0 \end{bmatrix}$$

CS : W-12(5M)

Ans. **Step 1 :**

$g(2, \phi) = 15$

$g(3, \phi) = 7$

$g(4, \phi) = 5$

Step 2 :

$g(2, \{3\}) = C_{23} + g(3, \phi)$

$= 9 + 7 = 16$

$g(2, \{4\}) = C_{24} + g(4, \phi)$

$= 14 + 5 = 19$

$g(3, \{2\}) = C_{32} + g(2, \phi)$

$= 8 + 15 = 23$

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

$= 16 + 5 = 21$

$g(4, \{2\}) = C_{42} + g(2, \phi)$

$= 11 + 15 = 26$

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

$= 10 + 7 = 17$

Step 3 :

$$g(2, \{3, 4\}) = \min \begin{bmatrix} C_{23} + g(3, \{4\}) \\ C_{24} + g(4, \{3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 9 + 21 \\ 14 + 17 \end{bmatrix} = \begin{bmatrix} 30 \\ 31 \end{bmatrix} = 30$$

$$g(3, \{2, 4\}) = \min \begin{bmatrix} C_{32} + g(2, \{4\}) \\ C_{34} + g(4, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 + 19 \\ 16 + 26 \end{bmatrix} = \begin{bmatrix} 27 \\ 42 \end{bmatrix} = 27$$

$$g(4, \{2, 3\}) = \min \begin{bmatrix} C_{42} + g(2, \{3\}) \\ C_{43} + g(3, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 11 + 16 \\ 10 + 23 \end{bmatrix} = \begin{bmatrix} 27 \\ 33 \end{bmatrix} = 27$$

Step 4 :

$$g(1, \{2, 3, 4\}) = \min \begin{bmatrix} C_{12} + g(2, \{3, 4\}) \\ C_{13} + g(3, \{2, 4\}) \\ C_{14} + g(4, \{2, 3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 12 + 30 \\ 10 + 27 \\ 7 + 27 \end{bmatrix} = \begin{bmatrix} 42 \\ 37 \\ 34 \end{bmatrix}$$

$g(1, \{2, 3, 4\}) = 34$

Cost = 34

Q.41. Implement TSP on following matrix, assuming vertex 2 as source vertex.

$$\begin{bmatrix} 0 & 12 & 14 & 9 \\ 9 & 0 & 8 & 12 \\ 10 & 14 & 0 & 16 \\ 8 & 7 & 18 & 0 \end{bmatrix}$$

CS : W-10(4M)

Ans. Given vertex 2 as source vertex.

Step 1 :

$g(1, \phi) = 12$

$g(3, \phi) = 14$

$g(4, \phi) = 7$

Step 2 :

$g(1, \{3\}) = C_{13} + g(3, \phi)$

$= 14 + 14 = 28$

$g(1, \{4\}) = C_{14} + g(4, \phi)$

$= 9 + 7 = 16$

$g(3, \{1\}) = C_{31} + g(1, \phi)$

$= 10 + 12 = 22$

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

$= 16 + 7 = 23$

$g(4, \{1\}) = C_{41} + g(1, \phi)$

$= 8 + 12 = 20$

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

$= 18 + 14 = 32$

Step 3 :

$$g(1, \{3, 4\}) = \min \begin{bmatrix} C_{13} + g(3, \{4\}) \\ C_{14} + g(4, \{3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 14 + 23 \\ 9 + 32 \end{bmatrix} = \begin{bmatrix} 37 \\ 39 \end{bmatrix} = 37$$

$$g(3, \{1, 4\}) = \min \begin{bmatrix} C_{31} + g(1, \{4\}) \\ C_{34} + g(4, \{1\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 10 + 16 \\ 16 + 20 \end{bmatrix} = \begin{bmatrix} 26 \\ 36 \end{bmatrix} = 26$$

$$g(4, \{1, 3\}) = \min \begin{bmatrix} C_{41} + g(1, \{3\}) \\ C_{43} + g(3, \{1\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 + 28 \\ 18 + 22 \end{bmatrix} = \begin{bmatrix} 36 \\ 40 \end{bmatrix} = 36$$

Step 4 :

$$g(2, \{1, 3, 4\}) = \min \begin{bmatrix} C_{21} + g(1, \{3, 4\}) \\ C_{23} + g(3, \{1, 4\}) \\ C_{24} + g(4, \{1, 3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 9 + 37 \\ 8 + 26 \\ 12 + 36 \end{bmatrix} = \begin{bmatrix} 46 \\ 34 \\ 48 \end{bmatrix}$$

$g(2, \{1, 3, 4\}) = 34$

Cost = 34

Q.42. Implement TSP using dynamic programming for following matrix :

$$\begin{bmatrix} 0 & 12 & 10 & 8 \\ 15 & 0 & 9 & 14 \\ 6 & 8 & 0 & 15 \\ 5 & 11 & 10 & 0 \end{bmatrix}$$

CS : S-14(7M)

Ans. **Step 1 :**

$g(2, \phi) = 15$

$g(3, \phi) = 6$

$g(4, \phi) = 5$

Step 2 :

$g(2, \{3\}) = C_{23} + g(3, \phi)$

$= 9 + 6 = 15$

$g(2, \{4\}) = C_{24} + g(4, \phi)$

$= 14 + 5 = 19$

$g(3, \{2\}) = C_{32} + g(2, \phi)$

$= 8 + 15 = 23$

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

$= 15 + 5 = 20$

$g(4, \{2\}) = C_{42} + g(2, \phi)$

$= 11 + 15 = 26$

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

$= 10 + 6 = 16$

Step 3 :

$$g(2, \{3, 4\}) = \min \begin{bmatrix} C_{23} + g(3, \{4\}) \\ C_{24} + g(4, \{3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 29 \\ 30 \end{bmatrix} = 30$$

$$g(3, \{2, 4\}) = \min \begin{bmatrix} C_{32} + g(2, \{4\}) \\ C_{34} + g(3, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 8 + 19 \\ 15 + 26 \end{bmatrix}$$

$$= \begin{bmatrix} 27 \\ 41 \end{bmatrix} = 27$$

$$g(4, \{2, 3\}) = \min \begin{bmatrix} C_{42} + g(2, \{3\}) \\ C_{43} + g(3, \{2\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 26 \\ 33 \end{bmatrix} = 26$$

Step 4 :

$$g(2, \{1, 3, 4\}) = \min \begin{bmatrix} C_{12} + g(2, \{3, 4\}) \\ C_{13} + g(3, \{2, 4\}) \\ C_{14} + g(4, \{2, 3\}) \end{bmatrix}$$

$$= \min \begin{bmatrix} 42 \\ 37 \\ 34 \end{bmatrix}$$

$g(2, \{1, 3, 4\}) = 34$

LONGEST COMMON SUBSEQUENCE

Q.43. What is longest common subsequence? Give its methodology. Write a recurrence equation for LCS. *CT : S-14(2M)*

Ans. **Longest common subsequence :**

- We can define longest common subsequence (LCS) problem as follows : "Given two sequence $x = (x_1, \ldots, x_m)$ and $y = (y_1, \ldots, y_n)$, find the longest subsequence $z = (z_1, \ldots, z_k)$ that is common to x and y.
- A subsequence is a subset of element from the sequence with strictly increasing order.

For example : $x = < A, B, C, B, D, A, B >$

$y = < B, D, C, A, B, A >$

Then common subsequences are :

- A
- B
- C
- D
- < A, A >
- < B, B >
- < B, C, A >
- < B, C, B, A >
- < B, D, A, B >

Methodology :

- To find longest common subsequence there are three possible cases.
- Let C [i, j] be the matrix representing the LCS and variable i, j represents the pointer for sequence a and b.

Case 1 : C [i, j] = 0 if i = 0 or j = 0

Case 2 : let a [1....i] and b [1....j] be the part of sequence 'a' and 'b' respectively.

if a [i] ≠ b [j] then selected sequence will be max [C [i – 1, j], C [i, j – 1]]

Case 3 : If a [i] = b [j]

Resultant sequence will be C [i – 1, j – 1] + 1

Final recurrence :

$$C[i,j] = \begin{cases} 0 \text{ if } i=0 \text{ or } j=0 \\ \max[C[i-1,j], C[i,j-1]] \text{ if } i,j>0 \text{ and } a(i) \neq b[j] \\ C[i-1,j-1]+1 \text{ if } i,j>0 \text{ and } a[i]=b[j] \end{cases}$$

Q.44. Write algorithm to generate LCS_matrix and Print_LCS.

CS : W-12, S-14(4M)

OR Write an algorithm to find longest common sequence with its time complexity and find the optimal solution for the given sequence

x : < A B C B D A B >

y : < B D C A B A >. **CT : S-10(13M)**

OR Write an algorithm to generate longest common subsequence.

CS : W-10, S-12(2M)

OR Write an algorithm to find out longest common subsequence given two strings of characters. Also write an algorithm to print the longest common subsequence. **CS : S-13(6M)**

OR Write an algorithm to compute c and b tables of an LCS.

CT : W-13(2M)

OR Write an algorithm to generate longest common subsequence (LCS) to print the LCS given two character strings of length 'm' and 'n'. **CS : W-13 (6M)**

Ans. Algorithm LCS :

```
LCS – LENGTH (X, Y)
  m ← length [X]
  n ← length [Y]
  for i ← 1 to m
      do C [i, 0] ← 0
  for j ← 0 to m
      do C [0, j] ← 0
  for i ← 1 to m
    do for j ← 1 to n
        do if Xi = Yj
            then C [i, j] ← C [i – 1, j – 1] + 1
                 b [i, j] ← "↖"
  else if C [i – 1, j] ≥ C [i, j – 1]
    then C [i, j] ← C [i – 1, j]
         b [i, j] ← "↑"
    else C [i, j] ← C [i, j – 1]
         b [i, j] ← "←"
  return c and b
```

Example : x : < A B C B D A B >

y : < B D C A B A >

The entry 4 in C [7, 6] is the length of the LCS < B, C, B, A > for X and Y.

	y_i	(B)	D	(C)	A	(B)	(A)
x_i	0	0	0	0	0	0	0
A	0	0↑	0↑	0↑	↖1	←1	↖1
(B)	0	↖1	←1	←1	↑1	↖2	←2
(C)	0	↑1	↑1	↖2	←2	↑2	↑2
(B)	0	↖1	↑1	↑2	↑2	↖3	←3
D	0	↑1	↖2	↑2	↑2	↑3	↑3
(A)	0	↑1	↑2	↑2	↖3	↑3	↖4
B	0	↖1	↑2	↑2	↑3	↖4	↑4

VBD

Complexity : The LCS < BC BA > takes time O (m + n) since atleast one of i and j is decremented in each stage of recursion.

- Algorithm print_LCS (b, X, i, j)

  If i = 0 or j = 0

  then return

  if b [i, j] = " ↖

  then PRINT_LCS (b, X, i

  print X i

  else if b [i, j] = " ↑ "

  then print _LCS (b, X, i – 1, j)

  else PRINT_LCS (b, X, i, j – 1)

Q.45. Apply LCS algorithm on the following strings and generate LCS with the help of LCS matrix.

CS : W-10(2M), W-12(5M), CT : S-13(6M), S-14(5M)

String A = exponential

String B = polynomial

Ans.

	y_i	p	o	l	y	n	o	m	i	a	l
	0	0	0	0	0	0	0	0	0	0	0
x_i	0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0
e	0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0
x	0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0	↑ 0
p	0	↖ 1	← 1	← 1	← 1	← 1	← 1	← 1	← 1	← 1	← 1
o	0	↑ 1	↖ 2	← 2	← 2	← 2	↖ 2	← 2	← 2	← 2	← 2
n	0	↑ 1	↑ 2	↑ 2	↑ 2	↖ 3	← 3	← 3	← 3	← 3	← 3
e	0	↑ 1	↑ 2	↑ 2	↑ 2	↑ 3	↑ 3	↑ 3	↑ 3	↑ 3	↑ 3
n	0	↑ 1	↑ 2	↑ 2	↑ 2	↖ 3	↑ 3	↑ 3	↑ 3	↑ 3	↑ 3
t	0	↑ 1	↑ 2	↑ 2	↑ 2	↑ 3	↑ 3	↑ 3	↑ 3	↑ 3	↑ 3
i	0	↑ 1	↑ 2	↑ 2	↑ 2	↑ 3	↑ 3	↑ 3	↖ 4	← 4	← 4
a	0	↑ 1	↑ 2	↑ 2	↑ 2	↑ 3	↑ 3	↑ 3	↑ 4	↖ 5	← 5
l	0	↑ 1	↑ 2	↑ 2	↑ 2	↑ 3	↑ 3	↑ 3	↑ 4	↑ 5	↖ 6

Length of LCS = 6

LCS = Ponial

Q.46. Implement LCS for the following sequences.

X = a a b a a b a b a a

Y = b a b a a b a b

CS : S-11(6M), W-11(7M)

Ans.

	y	b	a	b	a	a	b	a	b
x	0	0	0	0	0	0	0	0	0
a	0	↑ 0	↖ 1	← 1	↖ 1	↖ 1	← 1	↖ 1	← 1
a	0	↑ 0	↖ 1	↑ 1	↖ 2	↖ 2	← 2	↖ 2	← 2
b	0	↖ 1	↑ 1	↖ 2	↑ 2	↑ 2	↖ 3	← 3	↖ 3
a	0	↑ 1	↖ 2	↑ 2	↖ 3	↖ 3	↑ 3	↖ 4	← 4
a	0	↑ 1	↖ 2	↑ 2	↖ 3	↖ 4	← 4	↖ 4	↑ 4
b	0	↖ 1	↑ 2	↖ 3	↑ 3	↑ 4	↖ 5	← 5	↖ 5
a	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 4	↑ 5	↖ 6	← 6
b	0	↖ 1	↑ 2	↖ 3	↑ 4	↑ 4	↖ 5	↑ 6	↖ 7
a	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 5	↑ 5	↖ 6	↑ 7
a	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 5	↑ 5	↖ 6	↑ 7

Length = 7,

LCS = abaabab

Q.47. Determine LCS of X = < a, b, b, b, a, a, b, a > and

Y = < a, b, a, a, b, a, a, b >.

CT : W-12(8M)

Ans.

Complex

and j

	y_j	a	b	a	a	b	a	a	b
x_i	0	0	0	0	0	0	0	0	0
a	0	↖ 1	← 1	↖ 1	↖ 1	← 1	↖ 1	↖ 1	← 1
b	0	↑ 1	↖ 2	← 2	← 2	↖ 2	← 2	← 2	↖ 2
b	0	↑ 1	↖ 2	↑ 2	↑ 2	↖ 3	← 3	← 3	↖ 3
b	0	↑ 1	↖ 2	↑ 2	↑ 2	↖ 3	↑ 3	↑ 3	↖ 4
a	0	↖ 1	↑ 2	↖ 3	↖ 3	↑ 3	↖ 4	↖ 4	↑ 4
a	0	↖ 1	↑ 2	↖ 3	← 4	↖ 4	↖ 4	↖ 5	← 5
b	0	↑ 1	↖ 2	↑ 3	↑ 4	↖ 5	← 5	↑ 5	↖ 6
a	0	↑ 1	↑ 2	↖ 3	↖ 4	↑ 5	↖ 6	↖ 6	⑥

Length = 6
LCS = a, b a a b a

Q.48. Apply LCS algorithm on following string and generate LCS with help of LCS matrix

String A < 1 0 0 1 0 1 0 1 >

String B < 0 1 0 1 1 0 1 1 0 >

Ans.

		0	1	2	3	4	5	6	7	8	9
		B_j	0	1	0	1	1	0	1	1	0
0	Ai	0	0	0	0	0	0	0	0	0	0
1	1	0	↑ 0	↖ ①	← 1	↖ 1	↖ 1	← 1	↖ 1	↖ 1	← 1
2	0	0	↖ 1	↑ 1	↖ ②	← 2	← 2	↖ 2	← 2	← 2	← 2
3	0	0	↖ 1	↑ 1	↖ 2	↑ 2	↑ 2	↖ ③	← 3	← 3	↖ 3
4	1	0	↑ 1	↖ 2	↑ 2	↖ 3	↖ 3	↑ 3	↖ ④	↖ 4	← 4
5	0	0	↖ 1	↑ 2	↖ 3	↑ 3	↑ 3	↖ 4	↑ 4	↑ 4	↖ 5
6	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 4	↑ 4	↖ 5	↖ ⑤	↑ 5
7	0	0	↖ 1	↑ 2	↖ 3	↑ 4	↑ 4	↖ 5	↑ 5	↑ 5	↖ ⑥
8	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 5	↑ 5	↖ 6	↖ 6	↑ 6

Length LCS = 6

LCS = (100110), (0, 1, 0, 1, 0, 1) and (001101).

Q.49. Determine LCS of (1, 0, 0, 1, 0, 1, 0, 1) and (0, 1, 1, 0, 1, 0, 0, 1, 0).

Ans.

CT : W-13(5M)

	y_i	0	(1)	1	(0)	(1)	0	(0)	(1)	(0)
x_i	0	0	0	0	0	0	0	0	0	0
(1)	0	0↑	↖1	↖1	←1	↖1	←1	←1	↖1	←1
(0)	0	↖1	↑1	↑1	↖2	←2	↖2	↖2	←2	↖2
0	0	↖1	←1	↑1	↖2	↑2	↖3	↖3	←3	↖3
(1)	0	↑1	↖2	↖2	↑2	↖3	↑3	↑3	↖4	←4
(0)	0	↖1	↑2	↑2	↖3	↑3	↑3	↖4	↑4	↑5
(1)	0	↑1	↖2	↖3	↑3	↖4	↑4	↑4	↖5	↖5
(0)	0	↖1	↑2	↑3	↖4	↖5	↑5	↖5	↑5	↖6
1	0	↑1	↖2	↖3	↑4	↖5	↑5	↑5	↖6	↖6

Length LCS = 6

LCS = (101010).

Q.50. Implement LCS algorithm on following strings to construct LCS matrix :

String A = 1 0 1 1 0 1 1 0 1

String B = 0 1 0 1 1 0 1

CS : S-14(5M)

Ans.

	y_i	(0)	(1)	(0)	(1)	(1)	(0)	(1)
x_i	0	0	0	0	0	0	0	0
1	0	↑0	↖1	←1	↖1	↖1	←1	↖1
0	0	↖1	↑1	↖2	←2	←2	↖2	←2
1	0	↑1	↖2	↑2	↖3	↖3	←3	↖3
1	0	↑1	↖2	↑2	↖3	↖4	←4	↖4
(0)	0	↖1	↑2	↖3	↑3	↑4	↖5	←5
(1)	0	↑1	↖2	↑3	↖4	↖4	↑5	↖6
(1)	0	↑1	↖2	↑3	↖4	↖5	↑5	↖6
(0)	0	↖1	↑2	↖3	↑4	↑5	↖6	↑6
(1)	0	↑1	↖2	↑3	↖4	↖5	↑6	↖7

Length of LCS = 7

∴ LCS = 0 1 0 1 1 0 1

VBD

0/1 KNAPSACK PROBLEM

Q.51. Explain the 0/1 knapsack problem.

Ans. **0/1 knapsack problem :**

- In this problem the list of item are indivisible ; that means either we take the item or discard it.
- The 0/1 knapsack problem is stated as follows : Given a set of n items and a knapsack having capacity w, such that each item has some weight W_i and value V_i then the problem is to pack the knapsack in such a manner so that a maximum total value is achieved. The problem is said to be 0/1 knapsack problem because the item from the list is either rejected or accepted.
- **Formula :**

$$k_p [k, \omega]$$

$$= \begin{cases} k_p [k-1, \omega] & \text{if } \omega_k > \omega_1 \\ \max [k_p [k-1, \omega], k_p [k-1, \omega - \omega_k] + v_k, \text{otherwise} \end{cases}$$

Case I : When $\omega_k > w$

In this case item 'k' cannot become the part of the solution, as the weight is greater than 'w' which is unacceptable.

Case II : When $\omega_k \leq w$

In this case the item k can becomes the part of the solution and the item 'k' is chosen such that it has the maximum value.

Q.52. Using principle of dynamic programming design an algorithm for output knapsack problem. CT : W-09(4M)

Ans. Dynamic 0/1 knapsack (v, w, n, W)

```
for w = 0 to W
    do C [0, w] = 0
        for i = 1 to n
            do C [i, 0] = 0
                for w = 1 to W
                    do if w_i ≤ w
                        then if v_i + C [i − 1, w − w_i]
                            then C [i, w]
                                = v_i + C [i − 1, w − w_i]
                            else C [i, w]
                                = C [i − 1, w]
                        else C [i, w]
                            = C [i − 1, w − W].
```

Q.53. Compare dynamic programming and greedy method using knapsack problem. CT : S-06(7M)

OR Explain the difference between greedy and dynamic strategies. CT : S-14(10M)

Ans. **Difference between greedy and dynamic strategies :**

Greedy :

- Greedy algorithms are simple and straight forward. They are short sighted in their approach in the sense that they take decision on the basis of information in hand without worrying about the effect these decision may have in future.
- They are easy to implement, easy to invent and most of the time quite efficient, some problems may have no efficient solution, but a greedy algorithm may provide a solution that is close to optimal.
- Greedy algorithms are used to solve optimization problem.
- In greedy method we maximum the value for the subject given.
- A greedy algorithm works if a problem exhibits the following two properties :

(1) Greedy choice property.

(2) Optimal substructure.

Dynamic programming :

- Dynamic programming solves each subproblem once and stores the result in the table so that it can be rapidly retrived if needed again. It is often used in optimization problem.
- Dynamic programming is a problem solving technique that, like divide and conquer solves problems by dividing them into subproblems.
- Dynamic programming is used when the sub-problems are not independent.
- In dynamic programming we take the highest profit of the weight given.
- In both fractional and 0/1 knapsack problem exists. But 0/1 knapsack follows dynamic programming approach.

- Dynamic programming consists of two parts :

(1) Optimal substructure.

(2) Overlapping subproblem.

Q.54. What is matrix chain multiplication? Give its formula.

Ans. Matrix chain multiplication :

- In dynamic programming approach to solve chained matrix multiplication we construct a table m_{ij} $1 \le i \le j \le n$ where m_{ij} gives the optimal solution that is required of scalar multiplications.

Formula : For entries with difference of dimension = 1

(1) $m_{ii} = d_{i-1} \times d_i \times d_{i+1}$

(2) Entries with dimension difference > = 2

$$m_{i,\, i+s} = \min_{i \le k < i+s} [\, m_{ik} + m_{k+1,\, i+s} + d_{i-1} \times d_k \times d_{i+s} \,]$$

Q.55. Using principle of dynamic programming write an algorithm for chain matrix multiplication.

$A = 5 \times 13$

$B = 13 \times 7$

$C = 7 \times 89$

$D = 89 \times 3$ ***CS : S-10(9M), S-11,W-13(7M)***

OR Write an algorithm for optimal parenthesization of matrix chain product. ***CT : W-10(7M)***

OR Write a recursive algorithm for chain matrix multiplication. ***CS : S-14(4M)***

Ans. Algorithm matrix chain-order (p)

```
n ← length [p] – 1
    for i ← 1 to n
        do m [i, j] ← 0
    for l ← 2 to n
        do for i ← 1 to n – l + 1
            do j ← i + l – 1
                m [i, j] ← ∞
                for k ← i to j – 1
                    do q ← m [i, k] + m [k + 1, j]
                                + p_{i-1} p_k p_j
                    if q < m [i, j]
                        then m [i, j] ← q
                            s [i, j] ← k
return m and s.
```

Complexity : $O(n^3)$

Algorithm matrix-chain-multiply (A, S, i, j)

if $j > i$

then X ← Matrix chain multiply (A, S, i, S [i, j])

Y ← Matrix chain multiply (A, S, S [i, j] + 1, j)

return matrix - multiply (X, Y)

else return A_i

$A = 5 \times 13$, $B = 13 \times 7$,

$C = 7 \times 89$, $D = 89 \times 3$

$d_0 = 5$, $d_1 = 13$, $d_2 = 7$, $d_3 = 89$, $d_4 = 3$

$$m = \begin{bmatrix} 0 & m_{12} & m_{13} & m_{14} \\ & 0 & m_{23} & m_{24} \\ & & 0 & m_{34} \\ & & & 0 \end{bmatrix}$$

$m_{ij} = d_{i-1} \times di \times d_{i+1}$

$m_{12} = d_0 \times d_1 \times d_2$

$= 455$

$m_{23} = d_1 \times d_2 \times d_3$

$= 8099$

$m_{34} = d_2 \times d_3 \times d_4$

$= 1869$

for dimension ≥ 2

$m_{ij} = m_{i\, i+s}$

$$= \min_{i \le k < i+s} [\, m_{ik} + m_{k+1,\, i+s} + d_{i-1} \times d_k \times d_{i+s} \,]$$

$$m_{13} = \min_{i <= k < 3} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 8099 + 5 \times 13 \times 89 \\ 455 + 0 + 5 \times 7 \times 89 \end{bmatrix}$$

$$= \min \begin{bmatrix} 13884 \\ 3570 \end{bmatrix}$$

$= 3570$

$$m_{24} = \min_{k = 2,3} \begin{bmatrix} m_{22} + m_{34} + d_1 \times d_2 \times d_4 \\ m_{23} + m_{44} + d_1 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 1869 + 13 \times 7 \times 3 \\ 8099 + 0 + 13 \times 89 \times 3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 2142 \\ 11{,}570 \end{bmatrix}$$

$= 2142$

$$m_{14} = \min_{k=1,2,3} \begin{bmatrix} m_{11} + m_{24} + d_0 \times d_1 \times d_4 \\ m_{12} + m_{34} + d_0 \times d_2 \times d_4 \\ m_{13} + m_{44} + d_0 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 2142 + 5 \times 13 \times 3 \\ 455 + 1869 + 5 \times 7 \times 3 \\ 3570 + 0 + 5 \times 89 \times 3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 2337 \\ 2429 \\ 4905 \end{bmatrix}$$

$m_{14} = 2337$

$$m = \begin{bmatrix} 0 & 455 & 8570 & 2337 \\ & 0 & 8099 & 2142 \\ & & 0 & 1869 \\ & & & 0 \end{bmatrix}$$

The best sequence is A (BC) D.

Q.56. Find the optimal solution of matrices

A = 5 × 10

B = 10 × 13

C = 13 × 4

and give its complexity. *CT : W-10(6M)*

Ans. d = (5, 10, 13, 4)

$d_0 = 5$, $d_1 = 10$,

$d_2 = 13$, $d_3 = 4$

$$m = \begin{bmatrix} 0 & m_{12} & m_{13} \\ & 0 & m_{23} \\ & & 0 \end{bmatrix}$$

$m_{12} = d_0 \times d_1 \times d_2$

$= 5 \times 10 \times 13$

$= 650$

$m_{23} = d_1 \times d_2 \times d_3$

$= 10 \times 13 \times 4$

$= 520$

$$m_{13} = \min_{k=1,2} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 720 \\ 910 \end{bmatrix}$$

$m_{13} = 720$

$$m = \begin{bmatrix} 0 & 650 & 720 \\ & 0 & 520 \\ & & 0 \end{bmatrix}$$

Complexity = $\theta(n^3)$.

Q.57. Let $M_1 \times M_2 \times \ldots\ldots \times M_n$ be the chain of matrix product. Design multiplication of the above matrices. Determine time complexity of algorithm. *CT : S-13(7M)*

Ans. In dynamic programming approach to solve chained matrix multiplication we construct a table m_{ij} $1 \le i \le j \le n$ where m_{ij} gives the optimal solution that is required number of scalar multiplication for the part $m_i, m_{i+1}, \ldots\ldots m_j$ of the required product.

Suppose d [o....n] shows the dimensions of the matrices such that the matrix m_i, $1 \le i \le n$ is of dimension $d_{i-1} \times d_i$ we create the table m_{ij} diagonal by diagonals constains the elements m_{ij} such that $j - i = S$.

Step 1 : The diagonal S = 0 therefore contains the elements m_{ii}, $1 \le i \le n$, corresponding to the "product" m_i. There is no multiplication done so $m_{ii} = 0$, for every p. When S = 0, i = 1, 2 .....n.

Step 2 : The diagonal S = 1 contains the element $m_{i, i+1}$ corresponding to the products of the form $m_i\, m_{i+1}$. Here we have no choice but to compute the product directly which we can do by using $d_{i-1}\ d_i\ d_{i+1}$ scalar multiplication as we saw at beginning when S = 1

Then $m_{i, i+1} = d_{i-1}\ d_i\ d_{i+1}$

i = 1, 2, ...n – 1

Step 3 : When S > 1, the diagonal S contains the elements $m_{i\, m+s}$ corresponding to products of the form $m_i, m_{i+1}, \ldots\ldots m_{i+s}$.

Step 4 : To find the optimal, we choose the cut that minimizes the required number of scalar multiplication.

Time complexity of algorithm is $\theta(n^3)$.

Space complexity of algorithm is $\theta(n^2)$.

Q.58. Construct a table for making change with minimum number of coins for an amount less than or equal to 8 units with denominations $d_1 = 1$, $d_2 = 4$, $d_3 = 6$ units. Give algorithm and discuss the time complexity. **CT : S-07,12(7M)**

Ans. For making changes with minimum number of coins we have to apply matrix chain multiplication algorithm as units given are 1, 4, and 6.

i.e. $d_1 = 1$, $d_2 = 4$, $d_3 = 6$

$d = (1, 4, 6)$

Consider $d_0 = 1$

$$m = \begin{bmatrix} 0 & m_{12} & m_{13} \\ - & 0 & m_{23} \\ - & - & 0 \end{bmatrix}$$

$$m_{12} = d_0 \times d_1 \times d_2$$

$$= 1 \times 1 \times 4 = 4$$

$$m_{23} = d_1 \times d_2 \times d_3$$

$$= 1 \times 4 \times 6 = 24$$

$$m_{13} = \min_{k=1,2} \begin{Bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{Bmatrix}$$

$$= \min \begin{Bmatrix} 0 + 24 + 6 \\ 4 + 0 + 24 \end{Bmatrix}$$

$$= \min \begin{Bmatrix} 30 \\ 28 \end{Bmatrix}$$

$$m_{13} = 28$$

$$m = \begin{bmatrix} 0 & 4 & 28 \\ & 0 & 24 \\ & & 0 \end{bmatrix}$$

Q.59. Using chain matrix multiplication method, find out number of operation required to multiply following matrices and find the best sequences. **CS : W-11(9M)**

$A = 12 \times 5$, $B = 5 \times 45$,

$C = 45 \times 11$, $D = 11 \times 10$.

Ans. $d_0 = 12$, $d_1 = 5$, $d_2 = 45$, $d_3 = 11$, $d_4 = 10$

$$m_{12} = d_0 \times d_1 \times d_2$$

$$= 2700$$

$$m_{23} = d_1 \times d_2 \times d_3$$

$$= 5 \times 45 \times 11$$

$$= 2475$$

$$m_{34} = d_2 \times d_3 \times d_4$$

$$= 45 \times 11 \times 10$$

$$= 4950$$

$$m_{13} = \min_{k=1,2} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 2475 + 12 \times 5 \times 11 \\ 2700 + 0 + 12 \times 45 \times 11 \end{bmatrix}$$

$$= \min \begin{bmatrix} 3135 \\ 8640 \end{bmatrix}$$

$$= 3135$$

$$m_{24} = \min_{k=2,3} \begin{bmatrix} m_{22} + m_{34} + d_1 \times d_2 \times d_4 \\ m_{23} + m_{44} + d_1 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 4950 + 5 \times 45 \times 10 \\ 2475 + 0 + 5 \times 11 \times 10 \end{bmatrix}$$

$$= \min \begin{bmatrix} 7200 \\ 3025 \end{bmatrix}$$

$$= 3025$$

$$m_{14} = \min_{k=1,2,3} \begin{bmatrix} m_{11} + m_{24} + d_0 \times d_1 \times d_4 \\ m_{12} + m_{34} + d_0 \times d_2 \times d_4 \\ m_{13} + m_{44} + d_0 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 3025 + 12 \times 5 \times 10 \\ 2700 + 4950 + 12 \times 45 \times 10 \\ 3135 + 0 + 12 \times 11 \times 10 \end{bmatrix}$$

$$= \min \begin{bmatrix} 3625 \\ 13050 \\ 4455 \end{bmatrix}$$

$$m_{14} = 3625$$

Best sequence = A (BC) D.

Q.60. Using chained matrix multiplication method, find out number of operations required to multiply following matrices, also find the best sequence :

VBD

A = 5 × 10,

B = 10 × 3,

C = 3 × 12,

D = 12 × 5

CT : S-14(9M)

Ans. d = [5, 10, 3, 12, 5]

$d_0 = 5,\ d_1 = 10,\ d_2 = 3,\ d_3 = 12,\ d_4 = 5$

$$m = \begin{bmatrix} 0 & m_{12} & m_{13} & m_{14} \\ & 0 & m_{23} & m_{24} \\ & & 0 & m_{34} \\ & & & 0 \end{bmatrix}$$

for S = 1

$m_{12} = d_0 \times d_1 \times d_2$

$= 5 \times 10 \times 3$

$= 150$

$m_{23} = d_1 \times d_2 \times d_3$

$= 10 \times 3 \times 12$

$= 360$

$m_{34} = d_2 \times d_3 \times d_4$

$= 3 \times 12 \times 5$

$= 180$

for S = 2

$$m_{i,\, i+S} = \min_{i \le k \le i+S} \times (m_{ik} + m_{k+1,\, i+S} + d_{i-1}\, d_k\, d_{i+S})$$

$$m_{13} = \min_{k=1,2} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 960 \\ 330 \end{bmatrix} = 330$$

$$m_{24} = \min_{k=2,3} \begin{bmatrix} m_{22} + m_{34} + d_1 \times d_2 \times d_4 \\ m_{23} + m_{44} + d_1 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 330 \\ 90 \end{bmatrix} = 330$$

$$m_{14} = \min_{k=1,2,3} \begin{bmatrix} m_{11} + m_{24} + d_0 \times d_1 \times d_4 \\ m_{12} + m_{34} + d_0 \times d_2 \times d_4 \\ m_{13} + m_{44} + d_0 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 580 \\ 405 \end{bmatrix} = 405$$

$$m = \begin{bmatrix} 0 & 150 & 330 & 405 \\ & 0 & 360 & 330 \\ & & 0 & 180 \\ & & & \end{bmatrix}$$

Best possible sequence = A (BC) D.

Q.61. For the following set of matrices, implement chain matrix multiplication and find out minimum number of multiplications required with best sequence :

A = 13 × 5,

B = 5 × 89,

C = 89 × 3,

D = 3 × 34.

CS: S-14(4M)

Ans. **Step 1 :** d = [13, 5, 89, 3, 34]
0 1 2 3 4

Step 2 : n = 4. Generate matrix n of size n × n.

$$m = \begin{bmatrix} - & m_{12} & m_{13} & m_{14} \\ - & - & m_{23} & m_{24} \\ - & - & - & m_{34} \\ - & - & - & - \end{bmatrix}$$

Step 3 : Calculation for entries with difference of dimension = 1

$m_{12} = d_0 \times d_1 \times d_2$

$= 13 \times 5 \times 89$

$= 5785$

$m_{23} = d_1 \times d_2 \times d_3$

$= 5 \times 89 \times 3$

$= 1335$

$m_{34} = d_2 \times d_3 \times d_4$

$= 89 \times 3 \times 34$

$= 9078$

Step 4 : Entries with dimension difference > = 2

$$m_{13} = \min_{\substack{1 <= k < 3 \\ k=1,2}} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}$$

$$= \min \begin{bmatrix} 1530 \\ 9256 \end{bmatrix}$$

$= 1530$

$$m_{24} = \min_{k=2,3} \begin{bmatrix} m_{22} + m_{34} + d_1 \times d_2 \times d_4 \\ m_{23} + m_{44} + d_1 \times d_3 \times d_4 \end{bmatrix}$$

VBD

$$= \min \begin{bmatrix} 24208 \\ 1845 \end{bmatrix} = 1845$$

$$m_{14} = \min_{k=1,2,3} \begin{bmatrix} m_{11} + m_{29} + d_0 \times d_1 \times d_4 \\ m_{12} + m_{34} + d_0 \times d_2 \times d_4 \\ m_{13} + m_{49} + d_0 \times d_3 \times d_4 \end{bmatrix}.$$

$$= \min \begin{bmatrix} 4055 \\ 5420 \\ 2856 \end{bmatrix} = 2856$$

$$m = \begin{bmatrix} - & 5785 & 1530 & 2856 \\ - & - & 1335 & 1845 \\ - & - & - & 9078 \\ - & - & - & - \end{bmatrix}$$

Sequence A (BC) D = 2856.

Q.62. Implement chain matrix multiplication and find the best sequence to multiply following matrices. *CS : S-12(7M)*

$A_1 = 15 \times 18$, $A_2 = 18 \times 38$,

$A_3 = 38 \times 7$, $A_4 = 7 \times 20$.

Ans. $d_0 = 15,\ d_1 = 18,\ d_2 = 38,\ d_3 = 7,\ d_4 = 20$

$$m_{12} = d_0 \times d_1 \times d_2$$

$$= 15 \times 18 \times 38$$

$$= 10260$$

$$m_{23} = d_1 \times d_2 \times d_3$$

$$= 18 \times 38 \times 7$$

$$= 4788$$

$$m_{34} = d_2 \times d_3 \times d_4$$

$$= 38 \times 7 \times 20$$

$$= 5320$$

$$m_{13} = \min_{k=1,2} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}.$$

$$= \begin{bmatrix} 0 + 4788 + 15 \times 18 \times 7 \\ 10260 + 0 + 15 \times 38 \times 7 \end{bmatrix}$$

$$= \begin{bmatrix} 6678 \\ 14,250 \end{bmatrix} = 6678$$

$$m_{24} = \min_{k=2,3} \begin{bmatrix} m_{22} + m_{34} + d_1 \times d_2 \times d_4 \\ m_{23} + m_{44} + d_1 \times d_3 \times d_4 \end{bmatrix}.$$

$$= \begin{bmatrix} 0 + 5320 + 18 \times 38 \times 20 \\ 4788 + 0 + 18 \times 7 \times 20 \end{bmatrix}$$

$$= \begin{bmatrix} 19000 \\ 7,308 \end{bmatrix} = 7308$$

$$m_{14} = \min_{k=1,2,3} \begin{bmatrix} m_{11} + m_{24} + d_0 \times d_1 \times d_4 \\ m_{12} + m_{34} + d_0 \times d_2 \times d_4 \\ m_{13} + m_{44} + d_0 \times d_3 \times d_4 \end{bmatrix}.$$

$$= \begin{bmatrix} 0 + 7308 + 15 \times 18 \times 20 \\ 10260 + 5320 + 15 \times 38 \times 20 \\ 6678 + 0 + 15 \times 7 \times 20 \end{bmatrix}$$

$$= \begin{bmatrix} 12708 \\ 26980 \\ 8778 \end{bmatrix} = 8778$$

$m_{14} = 8778$

Best sequence = A (BC) D.

Q.63. For the following five matrices find the order of parenthesization for the optimal chain multiplication.

$A_1 = 15 \times 5$, $A_2 = 5 \times 10$, $A_3 = 10 \times 20$,

$A_4 = 20 \times 25$, $A_5 = 25 \times 30$. *CT : W-13(6M)*

Ans. $d_0 = 15,\ d_1 = 5,\ d_2 = 10,\ d_3 = 20,\ d_4 = 25,$

$d_5 = 30$

$$m_{12} = d_0 \times d_1 \times d_2$$

$$= 750$$

$$m_{23} = d_1 \times d_2 \times d_3$$

$$= 1000$$

$$m_{34} = d_2 \times d_3 \times d_4$$

$$= 5000$$

$$m_{45} = d_3 \times d_4 \times d_5$$

$$= 15,000$$

$$m_{13} = \min_{k=1,2} \begin{bmatrix} m_{11} + m_{23} + d_0 \times d_1 \times d_3 \\ m_{12} + m_{33} + d_0 \times d_2 \times d_3 \end{bmatrix}.$$

$$= \begin{bmatrix} 0 + 1000 + 15 \times 5 \times 20 \\ 750 + 0 + 15 \times 10 \times 20 \end{bmatrix}$$

$$= \begin{bmatrix} 2500 \\ 3750 \end{bmatrix} = 2500$$

$$m_{24} = \min_{k=2,3} \begin{bmatrix} m_{22} + m_{34} + d_1 \times d_2 \times d_4 \\ m_{23} + m_{44} + d_1 \times d_3 \times d_4 \end{bmatrix}.$$

$$= \begin{bmatrix} 0 + 5000 + 5 \times 10 \times 25 \\ 1000 + 0 + 5 \times 20 \times 25 \end{bmatrix}$$

$$= \begin{bmatrix} 6250 \\ 3500 \end{bmatrix} = 3500$$

$$m_{35} = \min_{k=3,4} \begin{bmatrix} m_{23} + m_{44} + d_2 \times d_3 \times d_5 \\ m_{24} + m_{45} + d_2 \times d_4 \times d_5 \end{bmatrix}$$

$$= \begin{bmatrix} 1000 + 0 + 10 \times 20 \times 30 \\ 3500 + 15000 + 10 \times 25 \times 30 \end{bmatrix}$$

$$= \begin{bmatrix} 7000 \\ 26000 \end{bmatrix} = 7000$$

$$m_{14} = \min_{k=1,2,3} \begin{bmatrix} m_{11} + m_{24} + d_0 \times d_1 \times d_4 \\ m_{12} + m_{34} + d_0 \times d_2 \times d_4 \\ m_{13} + m_{44} + d_0 \times d_3 \times d_4 \end{bmatrix}$$

$$= \min \begin{bmatrix} 0 + 3500 + 15 \times 5 \times 25 \\ 750 + 5000 + 15 \times 10 \times 25 \\ 1000 + 0 + 15 \times 20 \times 25 \end{bmatrix}$$

$$= \begin{bmatrix} 5375 \\ 9500 \\ 8500 \end{bmatrix}$$

$$m_{14} = 5375$$

$$m_{25} = \min_{k=2,3,4} \begin{bmatrix} m_{22} + m_{35} + d_1 \times d_2 \times d_5 \\ m_{23} + m_{45} + d_1 \times d_3 \times d_5 \\ m_{24} + m_{55} + d_1 \times d_4 \times d_5 \end{bmatrix}$$

$$= \begin{bmatrix} 0 + 7000 + 5 \times 10 \times 30 \\ 1000 + 5000 + 5 \times 20 \times 30 \\ 3500 + 0 + 5 \times 25 \times 30 \end{bmatrix}$$

$$= \begin{bmatrix} 8500 \\ 9000 \\ 7250 \end{bmatrix}$$

$$= 7250$$

$$m_{15} = \min_{k=1,2,3,4} \begin{bmatrix} m_{11} + m_{25} + d_0 \times d_1 \times d_5 \\ m_{12} + m_{35} + d_0 \times d_2 \times d_5 \\ m_{13} + m_{45} + d_0 \times d_3 \times d_5 \\ m_{14} + m_{55} + d_0 \times d_4 \times d_5 \end{bmatrix}$$

$$= \begin{bmatrix} 0 + 7250 + 15 \times 5 \times 30 \\ 750 + 7000 + 15 \times 10 \times 30 \\ 1000 + 5000 + 15 \times 20 \times 30 \\ 5375 + 0 + 15 \times 25 \times 30 \end{bmatrix}$$

$$= \min \begin{bmatrix} 9500 \\ 12,250 \\ 15000 \\ 16,625 \end{bmatrix}$$

$$= 9500$$

$$m_{15} = 9500$$

Best sequence = $A_1 (A_2 A_3) A_4 A_5$.

POINTS TO REMEMBER :

(1) *Dynamic programming is the most powerful designing technique for optimization problem where the solutions are based on multistage optimizing decision.*

(2) *Principle of optimality states that in an optimal sequence of decision on choices each subsequence must be optimal.*

(3) *Multistage graph is a method to represent the information about the distance between various vertices of the graph stagewise.*

(4) *All pairs shortest path method computes the shortest path between each pair of vertices in a weighted directed graph. The problem is efficiently solved by Floyd warshall algorithm.*

(5) *There are two methods of single source shortest path :*

(i) Dijkstra's algorithm.

(ii) Bellman ford algorithm.

(6) *OBST is a binary tree with an optimal cost.*

(7) *Longest common subsequence is a subsequence of a given sequence and where the objective is to find a maximum length common subsequence.*

(8) *In 0/1 knapsack problem the list of items are indivisible, that means we either take the item or discard it.*

(9) *In dynamic programming approach to solve chained matrix multiplication we construct a table m_{ij} $1 \le i \le j \le n$ where m_{ij} gives the optimal solution that is required for scalar multiplications.*