UNIT-III

DYNAMIC PROGRAMMING

Dynamic Programming is a method to solve the given problem by taking sequence of decisions. In order to get the optimal solution of the given problem. We should write the possible decisions by using principal of optimality; we will select the optimal solution of the given problem.

The difference between greedy method and dynamic programming is, In greedy method we take only one decision to find out the solution. Whereas in dynamic programming we take sequence of decisions, which satisfy the condition and finally we get the optimal solution by using principal of optimality.

Principal of optimality:

The principle of optimality states that no matter what the first decision, the remaining decisions must be optimal with respect to the state that results from this first decision. This principle implies that an optimal decision sequence is comprised for some formulations of some problem.

Since the principle of optimality may not hold for some formulations of some problems, it is necessary to verify that it does not hold for the problem being solved.

Dynamic programming cannot be applied when this principle does not hold.

Steps for Dynamic programming :

1 : Characteristics the given problem by using mathematical equation which gives the solution (or)

Sub-solution for given problem.

2: Recursively identify the value of the optimal solution.

3 : By using backtracking calculate the optimal solution.

4 : Finalize the optimal solution from computing information.

Applications of Dynamic Programming

1 : Optimal Binary Search Tree (OBST)

2 : All Pairs Shortest Path Problem

3 : Travelling Sales Person Problem.

4 : 0 /1 Knapsack Problem -

5 : Reliability Design

OPTIMAL BINARY SEARCH TREE (OBST) : The given set of identifiers { a1, a2,...., an} with a1<a2<a3...an. Let p(i) be the probability with which we can search for a_i. Let q(i) be the probability that the identifier x being searched. such that $a_i < x < a_{i+1}$ and $0 \le i \le n$. In other words p(i) is the probability of successful search and q(i) be the probability of unsuccessful search.

Clearly $\sum_{1 \le i \le n} p(i) + \sum_{1 \le i \le n} q(i)$ then obtain a tree with minimum cost. Such a tree with optimum Cost is

called optimal binary search tree.

To solve this problem using dynamic programming method by using following formulas.

1 : $c(i, j) = \min_{i < k \le j} \{ c(i, k-1) + c(k, j) + w(i, j) \}$

2 : $w(i, j) = p(j) + q(j) + w(i, j-1)$

3 : $r(i, j) = k$

Example1 : Using algorithm OBST compute w(i,j), r(i,j) and c(i,j), $0 \le i \le j \le 4$ for the identifier set (a1, a2, a3, a4) = (do, while, for, if) with (p1, p2, p3, p4) = (3, 3, 1, 1) and (q0, q1, q2, q3, q4) = (2, 3, 1, 1, 1) using r (i , j) construct the optimal binary search tree.

Solution :

Successful Probability : (p1, p2, p3, p4) = (3, 3, 1, 1)

UnSuccessful Probability : (q0, q1, q2, q3, q4) = (2, 3, 1, 1, 1)

identifier set : (a1, a2, a3, a4) = (end, goto, print, stop)

Initial Conditions :

w(i,j)= q(i)

c(i,j) = 0

r(i,j) = 0

Formulas :

1. $w(i, j) = p(j) + q(j) + w(i, j-1)$
2. $c(i, j) = \min_{i < k \le j} \{ c(i, k-1) + c(k, j) + w(i, j) \}$
3. $r(i, j) = k$

Step1 : $j – i = 0$

$w (i , j) = q(i)$

$w(0,0) = q(0)= 2$	$c(0,0) = 0$	$r(0,0) = 0$
$w(1,1) = q(1)= 3$	$c(1,1) = 0$	$r(1,1) = 0$
$w(2,2) = q(2)= 1$	$c(2,2) = 0$	$r(2,2) = 0$
$w(3,3) = q(3)= 1$	$c(3,3) = 0$	$r(3,3) = 0$
$w(4,4) = q(4)= 1$	$c(4,4) = 0$	$r(4,4) = 0$

Step 2 : $j – i = 1$, ($i =0, j = 1, k = 1$)

$w(i , j) = p(j) + q(j) + w (i , j - 1)$

$w (0, 1) = p(1) + q(1) + w(0,0)$

$=3 + 3 + 2 = 8$

$c (i, j) = \min \{ c(i , k - 1) + c(k , j)\} + w (i , j)$

$i < k \leq j$

$c(0,1) = \min \{ c(0,0)+c(1,1)\} + w(0,1)$

$= \min \{ 0 +0 \} + 8$

$= 0+8 = 8$

$r(0,1) = 1$

($i =1, j = 2, k = 2$)

$w (1, 2) = p(2) + q(2) + w(1,1)$

$= 3 + 1 + 3 = 7$

$c (i, j) = \min \{ c(i , k - 1) + c(k , j)\} + w (i , j)$
$i < k \leq j$

$c(1,2) = \min \{ c(1,1)+c(2,2)\} + w(1,2)$

$= \min \{ 0 + 0 \} + 7= 7$

$r(1,2) = 2$

($i =2, j = 3, k = 3$)

$w (2, 3) = p (3) + q(3) + w(2,2)$

$= 1 + 1 + 1 = 3$

$c (i , j) = \min \{ c(i , k - 1) + c(k , j)\} + w (i , j)$
$i < k \leq j$

$c(2,3) = \min \{ c(2,2) + c(3,3)\} + w(2,3)$

$= \min \{ 0 + 0 \} + 3 = 3$

$r(2,3) = 3$

(i =3, j = 4, k = 4)

$w(3,4) = p(4)+q(4)+w(3,3)$

$= 1 + 1 + 1 = 3$

$c(i,j) = \min\{c(i,k-1) + c(k,j)\} + w(i,j)$

$i < k \leq j$

$c(3,4) = \min\{c(3,3) + c(4,4)\} + w(3,4)$

$= \min\{0 + 0\} + 3 = 3$

$r(3,4) = 4$

Step 3 : j – i = 2 , (i =0, j= 2, k= 1,2)

$w(0,2) = p(2) + q(2) + w(0,1)$

$= 3 + 1 + 8 = 12$

$c(0,2) = \min\{c(0,0)+c(1,2), c(0,1)+c(2,2)\} + w(0,2)$

$= \min\{0 + 7, 8 + 0\} + 12$

$= \min\{7, 8\} + 12$

$= 7 + 12 = 19$

$r(0,2) = 1$

(i =1, j= 3, k=2,3)

$w(1,3) = p(3) + q(3) + w(1,2)$

$= 1 + 1 + 7 = 9$

$c(1,3) = \min\{c(1,1)+c(2,3), c(1,2) +c(3,3)\} + w(1,3)$

$=\min\{0 + 3, 7 + 0\} + 9$

$= \min\{3, 7\} + 9$

$= 3 + 9 = 12$

$r(1,3) = 2$

(i =2, j= 4, k=3)

$w(2,4) = p(4) + q(4) + w(2,3)$

$= 1 + 1 + 3 = 5$

$c(2,4) = \min\{c(2,2)+c(3,4)\} + w(2,4)$

$=\min\{0 + 3\} + 5 = 8$

$r(2,4) = 3$

Step 4 : j – i = 3 ,   (i =0, j= 3, k=1,2,3)

w(0,3) = p(3) + q(3) + w(0,2)

= 1 + 1 + 12 = 14

c(0,3) = min { c(0,0)+c(1,3), c(0,1)+c(2,3), c(0,2) +c(3,3) } + w(0,3)

= min { 0 + 12, 8 + 3 , 19 + 0 } + 14

= min { 12, 11, 19 } + 14

= 11 + 14 = 25

r(0,3) = 2

(i =1, j= 4, k=2,3,4)

w(1,4) = p(4) + q(4) + w(1,3)

= 1 + 1 + 9 = 11

c(1,4) = min { c(1,1)+c(2,4), c(1,2)+c(3,4), c(1,3)+c(4,4) } + w(1,4)

= min { 0 + 8, 7 + 3, 12 + 0 } + 11

= min { 8, 10, 12 } + 11

= 8 + 11 = 19

r(1,4) = 2

Step 5 : j – i = 4   (i =0, j= 4, k=1,2,3,4)

w(0,4) = p(4) + q(4) + w(0,3)

= 1 + 1 + 14 = 16

c(0,4) = min { c(0,0)+c(1,4), c(0,1)+c(2,4), c(0,2)+c(3,4), c(0,3)+c(4,4)} + w(0,4)

=min{ 0 + 19, 8+8, 19+3 , 25+0} + 16

=min{19, 16, 21, 25 }+16

=16+16= 32

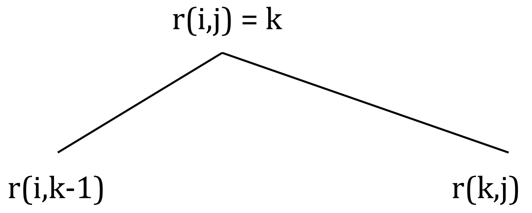
r(0,4) = 2

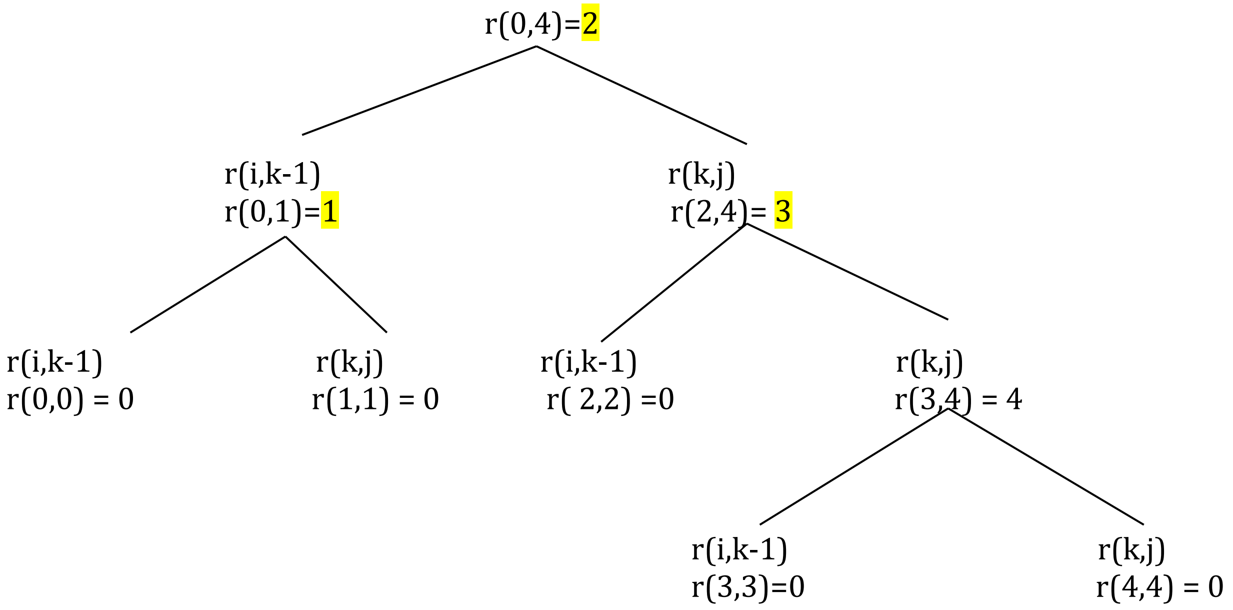
To build Optimal Binary Search Tree (OBST)

for a set(i, j) **j – i = 0**	**w (0, 0) = 2** **c (0, 0) = 0** **r (0, 0) = 0**	**w (1, 1) = 3** **c (1, 1) = 0** **r (1, 1) = 0**	**w (2, 2) = 1** **c (2, 2) = 0** **r (2, 2) = 0**	**w (3, 3) = 1** **c (3, 3) = 0** **r (3, 3) = 0**	**w (4, 4) = 1** **c (4, 4) = 0** **r (4, 4) = 0**
j – i = 1	w (0, 1) = 8 c (0,1) = 8 r (0, 1) = 1	w (1, 2) = 7 c (1, 2) = 7 r (1, 2) = 2	w (2, 3) = 3 c (2, 3) = 3 r (2, 3) = 3	w (3, 4) = 3 c (3, 4) = 3 r (3, 4) = 4	✗
j – i = 2	w (0, 2) = 12 c (0, 2) = 19 r (0, 2) = 1	w (1, 3) = 9 c (1, 3) = 12 r (1, 3) = 2	w (2, 4) = 5 c (2, 4) = 8 r (2, 4) = 3	✗	✗
j – i = 3	w (0, 3) = 14 c (0, 3) = 25 r (0, 3) = 2	w (1, 4) = 11 c (1, 4) = 19 r (1, 4) = 2	✗	✗	✗
j – i = 4	**w (0, 4) = 16** **c (0, 4) = 32** **r (0, 4) = 2**	✗	✗	✗	✗

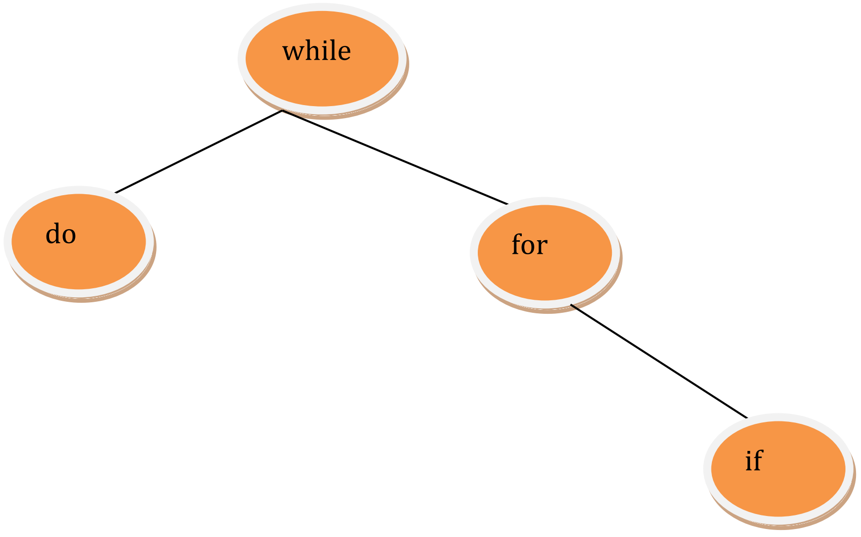
To build OBST, r (0,4) = 2, K =2.

Hence a_2 becomes root node.





(a1, a2, a3, a4) = (do, while, for, if)



Optimal Binary Search Tree with cost = 32

Example 2 : Using the algorithm OBST, compute W(i,j), R(i,j) and C(i,j), 0<=i<j<=4 for the identifier set (a1,a2,a3,a4)=(end, goto, print, stop) with p(1)=1/20, p(2)=1/5, p(3)=1/10, p(4)=1/20; q(0)=1/5, q(1)=1/10, q(2)=1/5, q(3)=1/20 and q(4)=1/20. Using the R(i,j)'s construct the OBST.

Solution :

Successful Probability

P(1) = 1/20 * 20 = 1

P(2) = 1/5 * 20 = 4

P(3) = 1/10 * 20 = 2

P(4) = 1/20 * 20 = 1

(p1,p2,p3,p4) = (1, 4, 2, 1)

UnSuccessful Probability

q(0)=1/5 * 20 = 4

q(1)=1/10 * 20 = 2

q(2)=1/5 * 20 = 4

q(3)=1/20 * 20=1

q(4)=1/20* 20=1

(q0,q1,q2,q3,q4) = (4,2,4,1,1)

(a1,a2,a3,a4)=(end, goto, print, stop)

Initial Conditions :

w(i,j)= q(i)

c(i,j) = 0

r(i,j) = 0

Formulas :

1. w(i , j) = p(j) + q(j) + w (i , j - 1)
2. c (i, j) = min { c(i , k - 1) + c(k , j) + w (i , j) }
   $i < k \leq j$
3. r(i , j) = k

Step1 : j – i = 0

$w(i, j) = q(i)$

$w(0,0) = q(0)= 4$	$c(0,0) = 0$	$r(0,0) = 0$
$w(1,1) = q(1)= 2$	$c(1,1) = 0$	$r(1,1) = 0$
$w(2,2) = q(2)= 4$	$c(2,2) = 0$	$r(2,2) = 0$
$w(3,3) = q(3)= 1$	$c(3,3) = 0$	$r(3,3) = 0$
$w(4,4) = q(4)= 1$	$c(4,4) = 0$	$r(4,4) = 0$

Step 2 : j – i = 1 , (i =0, j = 1, k = 1)

$w(i, j) = p(j) + q(j) + w(i, j - 1)$

$w(0, 1) = p(1) + q(1) + w(0,0)$

$=1 + 2 + 4 = 7$

$$c(i, j) = \min_{i < k \leq j} \{ c(i, k - 1) + c(k, j)\} + w(i, j)$$

$c(0,1) = \min \{ c(0,0)+c(1,1)\} + w(0,1)$

$= \min \{ 0 +0 \} + 7$

$= 0+7 = 7$

$r(0,1) = 1$

(i =1, j = 2, k = 2)

$w(1, 2) = p(2)+q(2)+w(1,1)$

$= 4 + 4 + 2 = 10$

$$c(i, j) = \min_{i < k \leq j} \{ c(i, k - 1) + c(k, j)\} + w(i, j)$$

$c(1,2) = \min \{ c(1,1)+c(2,2)\} + w(1,2)$

$= \min \{ 0 + 0 \} + 10= 10$

$r(1,2) = 2$

(i =2, j = 3, k = 3)

$w(2, 3) = p(3) + q(3) + w(2,2)$

$= 2 + 1 + 4 = 7$

$$c(i, j) = \min_{i < k \leq j} \{ c(i, k - 1) + c(k, j)\} + w(i, j)$$

$c(2,3) = \min \{ c(2,2) + c(3,3)\} + w(2,3)$

$= \min \{ 0 + 0 \} + 7 = 7$

$r(2,3) = 3$

(i =3, j = 4, k = 4)

w (3, 4) = p(4)+q(4)+w(3,3)

= 1 + 1 + 1 = 3

c (i, j) = min { c(i , k - 1) + c(k , j)} + w (i , j)

i < k ≤ j

c(3,4) = min { c(3,3) + c(4,4) } + w(3,4)

= min { 0 + 0} + 3 = 3

r(3,4) = 4

Step 3 : j – i = 2 , (i =0, j= 2, k= 1,2)

w(0,2) = p(2) + q(2) + w (0,1)

= 4 + 4 + 7 = 15

c(0,2) = min { c(0,0)+c(1,2), c(0,1)+c(2,2)} + w(0,2)

= min { 0 + 10 , 7 + 0 } + 15

= min { 10, 7 } + 15

= 7 + 15 = 22

r(0,2) = 2

(i =1, j= 3, k=2,3)

w(1,3) = p(3) + q(3) + w(1,2)

= 2 + 1 + 10 = 13

c(1,3) = min{c(1,1)+c(2,3), c(1,2) +c(3,3) } + w(1,3)

=min { 0 + 7, 10 + 0 } + 14

= min { 7, 10 } + 13

= 7+13 = 20

r(1,3) = 2

(i =2, j= 4, k=3)

w(2,4) = p(4) + q(4) + w(2, 3)

= 1 + 1 + 7 = 9

c(2,4) = min { c(2,2)+c(3,4)} + w(2,4)

=min{ 0 + 3 }+9 = 12

r(2,4) = 3

Step 4 : j – i = 3 , (i =0, j= 3, k=1,2,3)

w(0,3) = p(3) + q(3) + w(0,2)

= 2 + 1 + 15 = 18

c(0,3) = min { c(0,0)+c(1,3), c(0,1)+c(2,3), c(0,2) +c(3,3) } + w(0,3)

= min { 0 + 20, 7 + 7 , 22 + 0 } + 18

= min { 17,14,22} + 18

= 14 + 18 = 32

r(0,3) = 2

(i =1, j= 4, k=2,3,4)

w(1,4) = p(4) + q(4) + w(1,3)

= 1 + 1 + 13 = 15

c(1,4) = min { c(1,1)+c(2,4), c(1,2)+c(3,4), c(1,3)+c(4,4) } + w(1,4)

= min { 0 + 12, 10 + 3, 20 + 0 } + 15

= min { 12, 13, 20 } + 16

= 12 + 15 = 27

r(1,4) = 2

Step 5 : j – i = 4 (i =0, j= 4, k=1,2,3,4)

w(0,4) = p(4) + q(4) + w(0,3)

= 1 + 1 + 18 = 20

c(0,4) = min { c(0,0)+c(1,4), c(0,1)+c(2,4), c(0,2)+c(3,4), c(0,3)+c(4,4)} + w(0,4)

=min{ 0 + 27, 7+12, 22+3 , 32+0} + 21

=min{27, 19, 25, 32 }+20

=19+20= 39

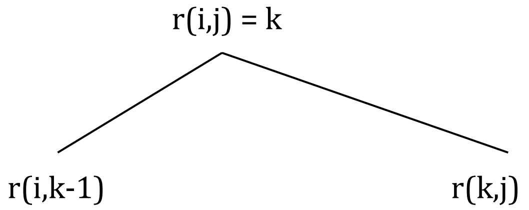
r(0,4) = 2

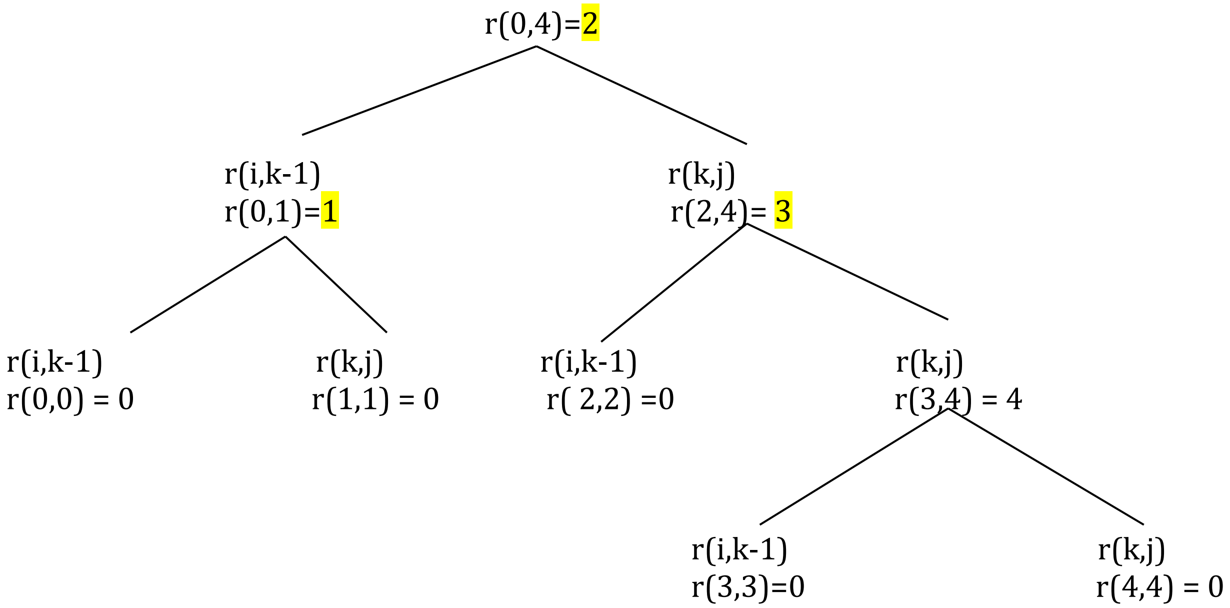
To build Optimal Binary Search Tree (OBST)

for a set(i, j) **j – i = 0**	**w (0, 0) = 4** **c (0, 0) = 0** **r (0, 0) = 0**	**w (1, 1) = 2** **c (1, 1) = 0** **r (1, 1) = 0**	**w (2, 2) = 4** **c (2, 2) = 0** **r (2, 2) = 0**	**w (3, 3) = 1** **c (3, 3) = 0** **r (3, 3) = 0**	**w (4, 4) = 1** **c (4, 4) = 0** **r (4, 4) = 0**
j – i = 1	w (0, 1) = 7 c (0,1) = 7 r (0, 1) = 1	w (1, 2)= 10 c (1, 2) = 10 r (1, 2) = 2	w (2, 3) = 7 c (2, 3) = 7 r (2, 3) = 3	w (3, 4) = 3 c (3, 4) = 3 r (3, 4) = 4	
j – i = 2	w (0, 2) = 15 c (0, 2) = 22 r (0, 2) = 2	w (1, 3) = 13 c (1, 3) = 20 r (1, 3) = 2	w (2, 4) = 9 c (2, 4) = 12 r (2, 4) = 3		
j – i = 3	w (0, 3) = 18 c (0, 3) = 21 r (0, 3) = 2	w (1, 4) = 15 c (1, 4) = 27 r (1, 4) = 2			
j – i = 4	**w (0, 4) = 20** **c (0, 4) = 39** **r (0, 4) = 2**				

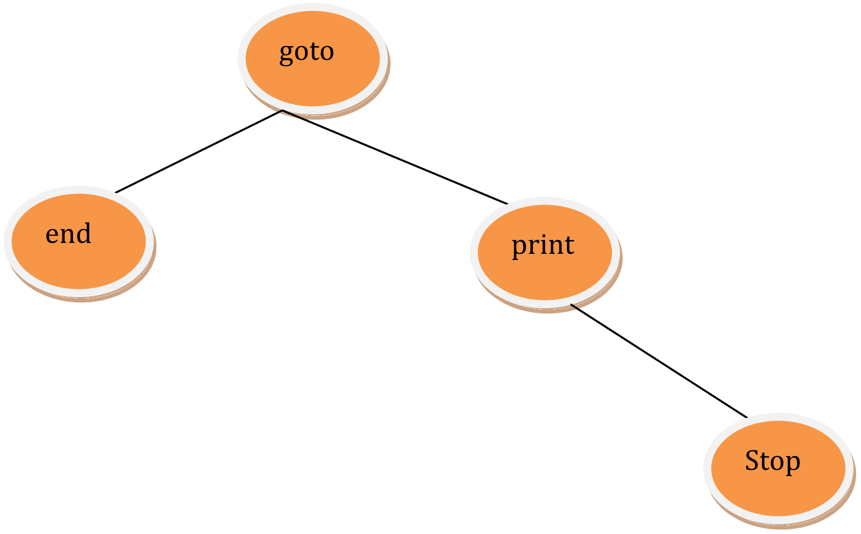
To build OBST, r (0,4) = 2, K =2.

Hence a_2 becomes root node.





(a1, a2, a3, a4)=(end, goto, print, stop)



Optimal Binary Search Tree with cost = 39

ALL PAIRS SHORTEST PATHS PROBLEM: Let G = (V, E) be a directed graph consisting of n vertices and each edge is associated with a weight. The problem of finding the shortest path between all pairs of vertices in a graph is called all pairs shortest path problem.

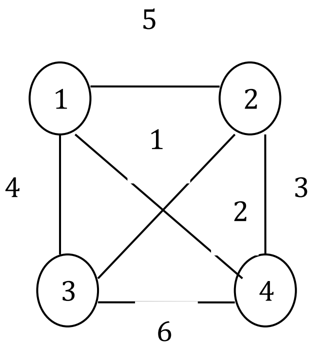
This problem can be solved by using Dynamic Programming technique. The all pairs shortest path problem is to determine a matrix A such that A(i , j) is the length of a shortest path from vertex i to vertex j. Assume that this path contains no cycles. If k is an intermediate vertex on this path, then the sub paths from i to k and from k to j are the shortest paths from i to k and k to j respectively. Otherwise the path I to j is not shortest path. If k is intermediate vertex with higest index, then the path i to k is the shortest path going through no vertex with index greater than k – 1. Similarly the path k to j is the shortest path going through no vertex with index greater than k – 1.

The shortest path can be computed using following recursive method.

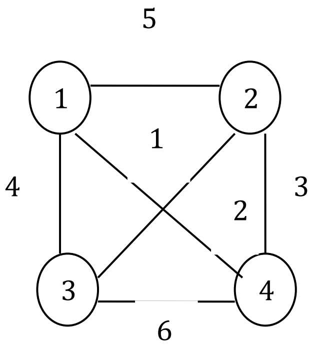
$A^{k} (i, j) = w (i, j), \quad \text{if } k = 0$

$= \min \{ A^{k-1} (i, j), A^{k-1} (i, k) + A^{k-1} (k, j) \}, \text{ if } k \geq 1.$

Ex : find the shortest path between all pairs of node in the following graph.



Sol :



Weight matrix for this undirected graph is as follows.

$$\begin{array}{c}1\\2\\3\\4\end{array}\left(\begin{array}{cccc}\mathbf{0}&\mathbf{5}&\mathbf{4}&\mathbf{1}\\\mathbf{5}&\mathbf{0}&\mathbf{2}&\mathbf{3}\\\mathbf{4}&\mathbf{2}&\mathbf{0}&\mathbf{6}\\\mathbf{1}&\mathbf{3}&\mathbf{6}&\mathbf{0}\end{array}\right) = \mathbf{A^0}$$

The formula for finding the all pairs shortest path problem as follows,

$$A^k(i, j) = \min\{A^{k-1}(i, j), A^{k-1}(i, k) + A^{k-1}(k, j)\}$$

Where if $k \geq 1$.

A^k, number of iterations to be taken .

These iterations depends on the number of weights of vertices given, in our problem, we have 5 vertices. we will take 4 iterations and we will always start from A^0 is the given matrix. So we have to find A^1, A^2, A^3 A^4 for the shortest path. According to the formula

$K = 1$

$A^{1}(1, 1) = \min\{A^{1-1}(1, 1), A^{1-1}(1, 1) + A^{1-1}(1, 1)\}$

$= \min\{0, 0 + 0\} = 0$

$A^{1}(1, 2) = \min\{A^{1-1}(1, 2), A^{1-1}(1, 1) + A^{1-1}(1, 2)\}$

$= \min\{5, 0 + 5\} = 5$

$A^{1}(1, 3) = \min\{A^{1-1}(1, 3), A^{1-1}(1, 1) + A^{1-1}(1, 3)\}$

$= \min\{4, 0 + 4\} = 4$

$A^{1}(1, 4) = \min\{A^{1-1}(1, 4), A^{1-1}(1, 1) + A^{1-1}(1, 4)\}$

$= \min\{1, 0 + 1\} = 1$

$A^{1}(2, 1) = \min\{A^{1-1}(2, 1), A^{1-1}(2, 1) + A^{1-1}(1, 1)\}$

$= \min\{5, 5 + 0\} = 5$

$A^{1}(2, 2) = \min\{A^{1-1}(2, 2), A^{1-1}(2, 1) + A^{1-1}(1, 2)\}$

$= \min\{0,5 + 5\} = 0$

$A^{1}(2, 3) = \min\{A^{1-1}(2, 3), A^{1-1}(2, 1) + A^{1-1}(1, 3)\}$

$= \min\{2, 5 + 4\} = 2$

$A^{1}(2, 4) = \min\{A^{1-1}(2, 4), A^{1-1}(2, 1) + A^{1-1}(1, 4)\}$

$= \min\{3, 5 + 1\} = 3$

$A^{1}(3, 1) = \min\{A^{1-1}(3, 1), A^{1-1}(3, 1) + A^{1-1}(1, 1)\}$

$= \min\{4, 4 + 0\} = 4$

$A^{1}(3, 2) = \min\{A^{1-1}(3, 2), A^{1-1}(3, 1) + A^{1-1}(1, 2)\}$

$= \min\{2,4 + 5\} = 2$

$A^{1}(3, 3) = \min\{A^{1-1}(3, 3), A^{1-1}(3, 1) + A^{1-1}(1, 3)\}$

$= \min\{0, 4 + 4\} = 0$

$A^{1}(3, 4) = \min\{A^{1-1}(3, 4), A^{1-1}(3, 1) + A^{1-1}(1, 4)\}$

$= \min\{6, 4 + 1\} = 5$

$A^{1}(4, 1) = \min\{A^{1-1}(4, 1), A^{1-1}(4, 1) + A^{1-1}(1, 1)\}$

$= \min\{1, 1 + 0\} = 1$

$A^{1}(4, 2) = \min\{A^{1-1}(4, 2), A^{1-1}(4, 1) + A^{1-1}(1, 2)\}$

$= \min\{3,1 + 5\} = 3$

$A^{1}(4, 3) = \min\{A^{1-1}(4, 3), A^{1-1}(4, 1) + A^{1-1}(1, 3)\}$

$= \min\{6, 1 + 4\} = 5$

$A^{1}(4, 4) = \min\{A^{1-1}(4, 4), A^{1-1}(4, 1) + A^{1-1}(1, 4)\}$

$= \min\{0, 1 + 1\} = 0$

$$\mathbf{A^1 =} \begin{pmatrix} \mathbf{0} & \mathbf{5} & \mathbf{4} & \mathbf{1} \\ \mathbf{5} & \mathbf{0} & \mathbf{2} & \mathbf{3} \\ \mathbf{4} & \mathbf{2} & \mathbf{0} & \mathbf{5} \\ \mathbf{1} & \mathbf{3} & \mathbf{5} & \mathbf{0} \end{pmatrix}$$

K = 2

$A^2 (1, 1) = \min \{ A^{2-1} (1, 1), A^{2-1} (1, 2) + A^{2-1} (2, 1) \}$

$= \min \{ 0, 5 + 5 \} = 0$

$A^2 (1, 2) = \min \{ A^{2-1} (1, 2), A^{2-1} (1, 2) + A^{2-1} (2, 2) \}$

$= \min \{ 5, 5 + 0 \} = 5$

$A^2 (1, 3) = \min \{ A^{2-1} (1, 3), A^{2-1} (1, 2) + A^{2-1} (2, 3) \}$

$= \min \{ 4, 5 + 2 \} = 4$

$A^2 (1, 4) = \min \{ A^{2-1} (1, 4), A^{2-1} (1, 2) + A\backslash^{2-1} (2, 4) \}$

$= \min \{ 1, 5 + 3 \} = 1$

$A^2 (2, 1) = \min \{ A^{2-1} (2, 1), A^{2-1} (2, 2) + A^{2-1} (2, 1) \}$

$= \min \{ 5, 0 + 5 \} = 5$

$A^2 (2, 2) = \min \{ A^{2-1} (2, 2), A^{2-1} (2, 2) + A^{2-1} (2, 2) \}$

$= \min \{ 0,0 + 0 \} = 0$

$A^2 (2, 3) = \min \{ A^{2-1} (2, 3), A^{2-1} (2, 2) + A2^{-1} (2, 3) \}$

$= \min \{ 2, 0 + 2 \} = 2$

$A^2 (2, 4) = \min \{ A^{2-1} (2, 4), A^{2-1} (2, 2) + A^{2-1} (2, 4) \}$

$= \min \{ 3, 0 + 3\} = 3$

$A^2 (3, 1) = \min \{ A^{2-1} (3, 1), A^{2-1} (3, 2) + A^{2-1} (2, 1) \}$

$= \min \{ 4, 2 + 5 \} = 4$

$A^2 (3, 2) = \min \{ A^{2-1} (3, 2), A^{2-1} (3, 2) + A^{2-1} (2, 2) \}$

$= \min \{ 2,2 + 0 \} = 2$

$A^2 (3, 3) = \min \{ A^{2-1} (3, 3), A^{2-1} (3, 2) + A2^{-1} (2, 3) \}$

$= \min \{ 0, 2 + 2 \} = 0$

$A^2 (3, 4) = \min \{ A^{2-1} (3, 4), A^{2-1} (3, 2) + A^{2-1} (2, 4) \}$

$= \min \{ 5, 2 + 3 \} = 5$

$A^2 (4, 1) = \min \{ A^{2-1} (4, 1), A^{2-1} (4, 2) + A^{2-1} (2, 1) \}$

$= \min \{ 1, 3 + 5 \} = 1$

$A^2 (4, 2) = \min \{ A^{2-1} (4, 2), A^{2-1} (4, 2) + A^{2-1} (2, 2) \}$

$= \min \{ 3,3 + 0 \} = 3$

$A^2 (4, 3) = \min \{ A^{2-1} (4, 3), A^{2-1} (4, 2) + A2^{-1} (2, 3) \}$

$= \min \{ 5, 3 + 2 \} = 5$

$A^2 (4, 4) = \min \{ A^{2-1} (4, 4), A^{2-1} (4, 2) + A^{2-1} (2, 4) \}$

$= \min \{ 0, 3 + 3 \} = 0$

$$\mathbf{A^2 =} \begin{pmatrix} \mathbf{0} & \mathbf{5} & \mathbf{4} & \mathbf{1} \\ \mathbf{5} & \mathbf{0} & \mathbf{2} & \mathbf{3} \\ \mathbf{4} & \mathbf{2} & \mathbf{0} & \mathbf{5} \\ \mathbf{1} & \mathbf{3} & \mathbf{5} & \mathbf{0} \end{pmatrix}$$

K = 3

$A^3 (1, 1) = \min \{ A^{3-1} (1, 1), A^{3-1} (1, 3) + A^{3-1} (3, 1) \}$

$= \min \{ 0, 4 + 4 \} = 0$

$A^3 (1, 2) = \min \{ A^{3-1} (1, 2), A^{3-1} (1, 3) + A^{3-1} (3, 2) \}$

$= \min \{ 5, 5 + 0 \} = 5$

$A^3 (1, 3) = \min \{ A^{3-1} (1, 3), A^{3-1} (1, 3) + A^{3-1} (3, 3) \}$

$= \min \{ 4, 5 + 2 \} = 4$

$A^3 (1, 4) = \min \{ A^{3-1} (1, 4), A^{3-1} (1, 3) + A\backslash^{3-1} (3, 4) \}$

$= \min \{ 1, 4 + 5 \} = 1$

$A^3 (2, 1) = \min \{ A^{3-1} (2, 1), A^{3-1} (2, 3) + A^{3-1} (3, 1) \}$

$= \min \{ 5, 4 + 4 \} = 5$

$A^3 (2, 2) = \min \{ A^{3-1} (2, 2), A^{3-1} (2, 3) + A^{3-1} (3, 2) \}$

$= \min \{ 0,2 + 2 \} = 0$

$A^3 (2, 3) = \min \{ A^{3-1} (2, 3), A^{3-1} (2, 3) + A^{3-1} (3, 3) \}$

$= \min \{ 2, 2 + 0 \} = 2$

$A^3 (2, 4) = \min \{ A^{3-1} (2, 4), A^{3-1} (2, 3) + A^{3-1} (3, 4) \}$

$= \min \{ 3, 2 + 5\} = 3$

$A^{3} (3, 1) = \min \{ A^{3-1} (3, 1), A^{3-1} (3, 3) + A^{3-1} (3, 1) \}$

$= \min \{ 4, 0 + 4 \} = 4$

$A^{3} (3, 2) = \min \{ A^{3-1} (3, 2), A^{3-1} (3, 3) + A^{3-1} (3, 2) \}$

$= \min \{ 2,0 + 2 \} = 2$

$A^{3} (3, 3) = \min \{ A^{3-1} (3, 3), A^{3-1} (3, 3) + A^{3-1} (3, 3) \}$

$= \min \{ 0, 0 + 0 \} = 0$

$A^{3} (3, 4) = \min \{ A^{3-1} (3, 4), A^{3-1} (3, 3) + A^{3-1} (3, 4) \}$

$= \min \{ 5, 0 + 5\} = 5$

$A^{3} (4, 1) = \min \{ A^{3-1} (4, 1), A^{3-1} (4, 3) + A^{3-1} (3, 1) \}$

$= \min \{ 1, 5 + 4 \} = 1$

$A^{3} (4, 2) = \min \{ A^{3-1} (4, 2), A^{3-1} (4, 3) + A^{3-1} (3, 2) \}$

$= \min \{ 3,5 + 2 \} = 3$

$A^{3} (4, 3) = \min \{ A^{3-1} (4, 3), A^{3-1} (4, 3) + A^{3-1} (3, 3) \}$

$= \min \{ 5, 5 + 0 \} = 5$

$A^{3} (4, 4) = \min \{ A^{3-1} (4, 4), A^{3-1} (4, 3) + A^{3-1} (3, 4) \}$

$= \min \{ 0, 5 + 5\} = 0$

$$\mathbf{A^{3}} = \begin{pmatrix} \mathbf{0} & \mathbf{5} & \mathbf{4} & \mathbf{1} \\ \mathbf{5} & \mathbf{0} & \mathbf{2} & \mathbf{3} \\ \mathbf{4} & \mathbf{2} & \mathbf{0} & \mathbf{5} \\ \mathbf{1} & \mathbf{3} & \mathbf{5} & \mathbf{0} \end{pmatrix}$$

K = 4

$A^{4} (1, 1) = \min \{ A^{4-1} (1, 1), A^{4-1} (1, 4) + A^{4-1} (4, 1) \}$

$= \min \{ 0, 1 + 1 \} = 0$

$A^{4} (1, 2) = \min \{ A^{4-1} (1, 2), A^{4-1} (1, 4) + A^{4-1} (4, 2) \}$

$= \min \{ 5, 1 + 3 \} = 4$

$A^{4} (1, 3) = \min \{ A^{4-1} (1, 3), A^{4-1} (1, 4) + A^{4-1} (4, 3) \}$

$= \min \{ 4, 1 + 5 \} = 4$

$A^{4} (1, 4) = \min \{ A^{4-1} (1, 4), A^{4-1} (1, 4) + A\backslash^{4-1} (4, 4) \}$

$= \min \{ 1, 1 + 0 \} = 1$

$A^4 (2, 1) = \min \{ A^{4-1} (2, 1), A^{4-1} (2, 4) + A^{4-1} (4, 1) \}$

$= \min \{ 5, 3 + 1 \} = 4$

$A^4 (2, 2) = \min \{ A^{4-1} (2, 2), A^{4-1} (2, 4) + A^{4-1} (4, 2) \}$

$= \min \{ 0, 3 + 3 \} = 0$

$A^4 (2, 3) = \min \{ A^{4-1} (2, 3), A^{4-1} (2, 4) + A^{4-1} (4, 3) \}$

$= \min \{ 2, 3 + 5 \} = 2$

$A^4 (2, 4) = \min \{ A^{4-1} (2, 4), A^{4-1} (2, 4) + A\backslash^{4-1} (4, 4) \}$

$= \min \{ 3, 3 + 0 \} = 3$

$A^4 (3, 1) = \min \{ A^{4-1} (3, 1), A^{4-1} (3, 4) + A^{4-1} (4, 1) \}$

$= \min \{ 4, 5 + 1 \} = 4$

$A^4 (3, 2) = \min \{ A^{4-1} (3, 2), A^{4-1} (3, 4) + A^{4-1} (4, 2) \}$

$= \min \{ 2, 5 + 3 \} = 2$

$A^4 (3, 3) = \min \{ A^{4-1} (3, 3), A^{4-1} (3, 4) + A^{4-1} (4, 3) \}$

$= \min \{ 0, 5 + 5 \} = 0$

$A^4 (3, 4) = \min \{ A^{4-1} (3, 4), A^{4-1} (3, 4) + A\backslash^{4-1} (4, 4) \}$

$= \min \{ 5, 5 + 0 \} = 5$

$A^4 (4, 1) = \min \{ A^{4-1} (4, 1), A^{4-1} (4, 4) + A^{4-1} (4, 1) \}$

$= \min \{ 1, 0 + 1 \} = 1$

$A^4 (4, 2) = \min \{ A^{4-1} (4, 2), A^{4-1} (4, 4) + A^{4-1} (4, 2) \}$

$= \min \{ 3, 0 + 3 \} = 3$

$A^4 (4, 3) = \min \{ A^{4-1} (4, 3), A^{4-1} (4, 4) + A^{4-1} (4, 3) \}$

$= \min \{ 5, 0 + 5 \} = 5$

$A^4 (4, 4) = \min \{ A^{4-1} (4, 4), A^{4-1} (4, 4) + A\backslash^{4-1} (4, 4) \}$

$= \min \{ 0, 0 + 0 \} = 0$

$$\mathbf{A^4 =} \begin{pmatrix} \mathbf{0} & \mathbf{4} & \mathbf{4} & \mathbf{1} \\ \mathbf{4} & \mathbf{0} & \mathbf{2} & \mathbf{3} \\ \mathbf{4} & \mathbf{2} & \mathbf{0} & \mathbf{5} \\ \mathbf{1} & \mathbf{3} & \mathbf{5} & \mathbf{0} \end{pmatrix}$$

Travelling Salesperson Problems : If there are n cities and cost of traveling from one city to other city is given. A salesman has to start from any one of the city and has to visit all the cities exactly once and has to return to the starting place with shortest distance or minimum cost.

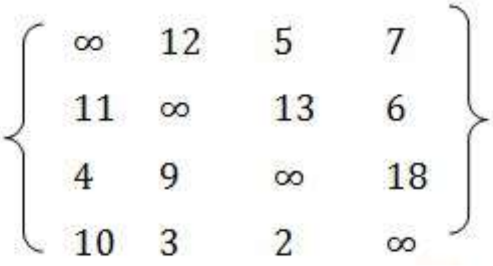
Travelling Sales person problem can be computed following recursive method.

1 : $g(i, \Phi) = C_{i,1}$

2 : $g(i, S) = \min \{ C_{ij} + g(j, S - \{ j \} \}$

Here **g(i, S)** means i is starting node and the nodes in S are to be traversed. **min** is considered as the intermediate node **g(j , S – { j })** means j is already traversed. So next we have to traverse S – { j } with j as starting point.

Example 1 : Construct an optimal travelling sales person tour using Dynamic Programming.



$$\left\{ \begin{array}{cccc} \infty & 12 & 5 & 7 \\ 11 & \infty & 13 & 6 \\ 4 & 9 & \infty & 18 \\ 10 & 3 & 2 & \infty \end{array} \right\}$$

Solution : The formula for solving this problem is,

1 : $g(i, \Phi) = C_{i\,1}$

2 : $g(i, S) = \min \{ C_{i\,j} + g(j, S - \{ j \} \}$

Step1 : Consider set of 0 elements, such that

$S = \Phi$

$g(i, \Phi) = C_{i\,1}$

$g(1, \Phi) = C_{1\,1} = \infty$

$g(2, \Phi) = C_{2\,1} = 11$

$g(3, \Phi) = C_{3\,1} = 4$

$g(4, \Phi) = C_{4\,1} = 10$

Step2 : Consider set of 1 elements, such that

S = 1, {2} , { 3 } , { 4 }

$g(i, S) = \min \{ C_{ij} + g(j, S - \{ j \}) $

$$\begin{aligned} g(2, \{3\}) &= \min \{ c_{23} + g(3, \{3\} - \{3\}) \} \\ &= \min \{ c_{23} + g(3, \Phi) \} \\ &= \min \{ 13 + 4 \} = 17. \end{aligned}$$

$$\begin{aligned} g(2, \{4\}) &= \min \{ c_{24} + g(4, \{4\} - \{4\}) \} \\ &= \min \{ c_{24} + g(4, \Phi) \} \\ &= \min \{ 6 + 10 \} = 16. \end{aligned}$$

$$\begin{aligned} g(3, \{4\}) &= \min \{ c_{34} + g(4, \{4\} - \{4\}) \} \\ &= \min \{ c_{34} + g(4, \Phi) \} \\ &= \min \{ 18 + 10 \} = 28. \end{aligned}$$

$$\begin{aligned} g(3, \{2\}) &= \min \{ c_{32} + g(2, \{2\} - \{2\}) \} \\ &= \min \{ c_{32} + g(2, \Phi) \} \\ &= \min \{ 9 + 11 \} \\ &= 20. \end{aligned}$$

$$\begin{aligned} g(4, \{2\}) &= \min \{ c_{42} + g(2, \{2\} - \{2\}) \} \\ &= \min \{ c_{42} + g(2, \Phi) \} \\ &= \min \{ 3 + 11 \} = 14. \end{aligned}$$

$$\begin{aligned} g(4, \{3\}) &= \min \{ c_{43} + g(3, \{3\} - \{3\}) \} \\ &= \min \{ c_{43} + g(3, \Phi) \} \\ &= \min \{ 2 + 4 \} = 6. \end{aligned}$$

Step 3 : Consider set of 2 elements, such that

S =2, { 2,3} { 2,4 }, { 3,4}

$g(i, S) = \min \{ C_{ij} + g(j, S - \{ j \}) \}$

$$\begin{aligned} g(2, \{ 3,4 \}) &= \min \{ c_{23} + g(3, \{3,4\} - \{3\}) , c_{24} + g(4, \{3,4\} - \{4\}) \} \\ &= \min \{ c_{23} + g(3, \{4\}), c_{24} + g(4, \{3\}) \} \\ &= \min \{ 13 + 28, 6 + 6 \} \\ &= \min \{ 41, 12 \} \\ &= 12. \end{aligned}$$

$$\begin{aligned} g(3, \{ 2,4 \}) &= \min \{ c_{32} + g(2, \{2,4\} - \{2\}) , c_{34} + g(4, \{2,4\} - \{4\}) \} \\ &= \min \{ 9 + g(2, \{4\}), 18 + g(4, \{2\}) \} \\ &= \min \{ 9 + 16, 18 + 14 \} \\ &= \min \{ 25, 32 \} \\ &= 25. \end{aligned}$$

g(4, { 2,3 }) = min { c_{42} + g(2, {2,3 } – {2}) , c_{43} + g(3, {2,3 } – {3}) }
= min { 3 + g(2, {3}), 2 + g(3, {2}) }
= min { 3 + 17, 2 + 20 }
= min { 20, 22 }
= 20.

Step 4: Consider set of 4 elements, such that

S = 3, { 2,3,4 }

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

= min { c_{12} + g(2, {3,4}), 5 + g(3, {2,4}), 7 + g(4, {2,3}) }

= min { 12 + 12, 5 + 25, 7 + 20 }

= min { 24, 30, 27 }

= 24

The optimal Solution is : c_{12} + g (2, { 3,4 }),
c_{12} + c_{24} + g(4, {3}),
c_{12} + c_{24} + c_{43} + g(3, Φ),
c_{12} + c_{24} + c_{43} + $C_{3\,1}$

The optimal Cost : 1-2-4-3-1 = 12 + 6 + 2 + 4 = 24

Example 2 : Construct an optimal travelling sales person tour using Dynamic Programming.

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

Solution : The formula for solving this problem is,

1 : $g(i, \Phi) = C_{i\,1}$

2 : $g(i, S) = \min\{ C_{i\,j} + g(j, S - \{j\})\}$

Step1 : Consider set of 0 elements, such that

$S = \Phi$

$g(i, \Phi) = C_{i\,1}$

$g(1, \Phi) = C_{1\,1} = 0$

$g(2, \Phi) = C_{2\,1} = 5$

$g(3, \Phi) = C_{3\,1} = 6$

$g(4, \Phi) = C_{4\,1} = 8$

Step2 : Consider set of 1 elements, such that

S = 1, {2} , { 3 } , { 4 }

$g(i, S) = \min\{ C_{i\,j} + g(j, S - \{j\})\}$

$g(2,\{3\}) = \min\{ c_{23} + g(3, \{3\} - \{3\})\}$
$= \min\{ c_{23} + g(3, \Phi)\}$
$= \min\{ 9 + 6 \} = 15$

$g(2,\{4\}) = \min\{ c_{24} + g(4, \{4\} - \{4\})\}$
$= \min\{ c_{24} + g(4, \Phi)\}$
$= \min\{ 10 + 8 \} = 18$

$g(3,\{2\}) = \min\{ c_{32} + g(2, \{2\} - \{2\})\}$
$= \min\{ c_{32} + g(2, \Phi)\}$
$= \min\{ 13 + 5 \} = 18$

$g(3,\{4\}) = \min\{ c_{34} + g(4, \{4\} - \{4\})\}$
$= \min\{ c_{34} + g(4, \Phi)\}$
$= \min\{ 12 + 8 \} = 20$

$g(4,\{2\}) = \min\{ c_{42} + g(2, \{2\} - \{2\})\}$
$= \min\{ c_{42} + g(2, \Phi)\}$
$= \min\{ 8 + 5 \} = 13$

$g(4,\{3\}) = \min\{ c_{43} + g(3, \{3\} - \{3\})\}$
$= \min\{ c_{43} + g(3, \Phi)\}$
$= \min\{ 9 + 6 \} = 15$

Step 3 : Consider set of 2 elements, such that

S =2, { 2,3} { 2,4 }, { 3,4}

$g(i, S) = \min \{ C_{i\,j} + g(j, S - \{ j \}) \}$

$g(2,\{3,4\}) = \min \{ c_{23} + g(3,\{3,4\} - \{ 3 \}), c_{24} + g(4,\{3,4\}- \{ 4 \}) \}$
$= \min \{ c_{23} + g (3, \{ 4 \}), c_{24} + g (4 , \{ 3 \}) \}$
$= \min \{ 9 + 20, 10 + 15 \}$
$= \min \{ 29 , 25 \}$
$= 25$

$g(3,\{ 2,4\}) = \min \{ c_{32} + g(2,\{2,4\} - \{ 2 \}), c_{34} + g(4,\{2,4\}- \{ 4 \}) \}$
$= \min \{ c_{32} + g(2, \{ 4 \}), c_{34} + g(4, \{ 2 \}) \}$
$= \min \{ 13 + 18, 12 + 13 \}$
$= \min \{ 31,25 \}$
$= 25$

$g(4,\{2,3\}) = \min \{ c_{42} + g(2,\{2,3\} - \{ 2 \}), c_{43} + g(3,\{2,3\}- \{ 3 \}) \}$
$=\min \{ c_{42} + g(2, \{ 3 \}), c_{43} + g(3, \{ 2 \}) \}$
$= \min \{ 8 + 15 , 9 + 18 \}$
$= \min \{ 23,27 \}$
$= 23$

Step 4: Consider set of 3 elements, such that

S = 3, { 2,3,4 }

$g(i, S) = \min \{ C_{i\,j} + g(j, S - \{ j \}) \}$

$g(1, \{ 2,3,4 \}) = \min \{ c_{12} + g (2, \{ 2,3,4 \} - \{ 2 \}), c_{13} + g (3, \{ 2,3,4 \} - \{ 3 \}), c_{14} + g (4, \{ 2,3,4 \} - \{ 4 \}) \}$
$= \min \{ c_{12} + g (2, \{ 3,4 \}), c_{13} + g (3, \{ 2,4 \}), c_{14} + g (4, \{ 2,3 \})\}$
$= \min \{ 10 + 25, 15 + 25, 20 + 23 \}$
$= \min \{ 35, 40, 43 \}$
$= 35$

The optimal Solution is : $c_{12} + g (2, \{ 3,4 \}),$
$c_{12} + c_{24} + g (4 , \{ 3 \}),$
$c_{12} + c_{24} + c_{43} + g(3, \Phi),$
$c_{12} + c_{24} + c_{43} + c_{3\,1}$

The optimal Cost: 1-2-4-3-1 =10 + 10 + 9 + 6 = 35

0 /1 Knapsack Problem: If we are given n objects and a knapsack or a bag in which the object i that has weight wi is to be placed. The knapsack has a capacity W. Then the profit that can be earned is pixi . The objective is to obtain filling of knapsack with maximum profit earned. Maximized pixi. Subject to constraint wixi<=W Where 1<=i<=n and n is total no. of objects and xi =0 or 1.

0 /1 Knapsack Problem can be computed following recursive method.

1. **Initially** $s^0 = \{(0,0)\}$ (P,W)
2. **Merging Operation**

   $\mathbf{S^{i+1} = S^i + S^i_1}$
3. **Purging rule (OR) dominance rule**

   If $\mathbf{S^{i+1}}$ contains (Pj , Wj) and (Pk, Wk) these two pairs such that Pj<=Pk and Wj>=Wk, then (Pj ,Wj) can be eliminated . This purging rule is also called as dominance rule. In purging rule basically the dominated tuples gets purged. In short remove the pair with less profit and more weight.

Example 1 : Consider the following 0 / 1 Knapsack problem using dynamic programming m = 6, n = 3, (w1, w2, w3) = (2, 3, 4), (p1, p2, p3) = (1, 2, 5).

Solution : we have to build the sequence of decisions S^0, S^1, S^2, S^3

Initially S^0 = { (0,0) }

S^0_1 = Select next (p1,w1) pair and add it with S^0

= (1,2) + { (0,0) }

= { (1+0), (2 + 0) } = { (1,2) }

$S^{i+1} = S^i \cup S^i_1$

$S^1 = S^0 + S^0_1$

= { (0,0) } ∪ { (1,2) }

= { (0,0), (1,2) }

To apply Purging Rule : There will be no deleted

S^1_1 = **Select next (P2,W2) pair and add it with S^1**

= (2,3) + { (0,0), (1,2) }

= { (2 + 0, 3 + 0), (2 + 1, 3 + 2) }

= { (2,3) , (3,5) }

$S^2 = S^1 \cup S^1_1$

= { (0,0), (1,2) } ∪ { (2,3) , (3,5) }

= { (0,0) , (1,2), (2,3),(3,5) }

To apply Purging Rule : There will be no deleted

S^2_1 = **Select next (P3,W3) pair and add it with S^2**

= (5,4) + { (0,0) , (1,2), (2,3),(3,5) }

= { (5+0,4+0),(5+1,4+2),(5+2,4+3),(5+3,4+5)}

= { (5,4),(6,6), (7,7), (8,9) }

$S^3 = S^2 \cup S^2_1$

= { (0,0) , (1,2), (2,3),(3,5) } ∪ { (5,4),(6,6), (7,7), (8,9) }

= { (0,0),(1,2),(2,3),(3,5),(5,4),(6,6),(7,7),(8,9) }

To apply Purging Rule

(3,5) and (5,4)
(pj,wj) and (pk,wk)

Pj<=pk and wj>=wk
3<=5 and 5>=4 -- True, (pj,wj) pair can be deleted (3,5)

S^3 = { (0,0),(1,2),(2,3),(5,4),(6,6),(7,7),(8,9) }

Here Capacity of the Knapsack is 6, Now we have to remove the pairs, in which wi>m, i.e wi>6

S^3 = { (0,0),(1,2),(2,3),(5,4),(6,6) }

Here M = 6, we will find the tuple, denoting the weight ‘ 6 ‘

i,e (6,6) belongs to S^3

(6,6) does not belongs to S^2

Therefore , We must set $x_3 = 1$

The pair (6,6) came from the pair (6-p3,6-w3)

(6-5, 6-4) = (1,2)

Here (1,2) belongs to S^2

(1,2) belongs to S^1

Therefore , We must set $x_2 = 0$

(1,2) does not belongs to S^0

Therefore , We must set $x_1 = 1$

Hence an Optimal Solution is (x_1 , x_2, x_3) = (1, 0 , 1)

Maximum Profit = p1x1+p2x2+p3x3

= 1*1+2.0+5.1

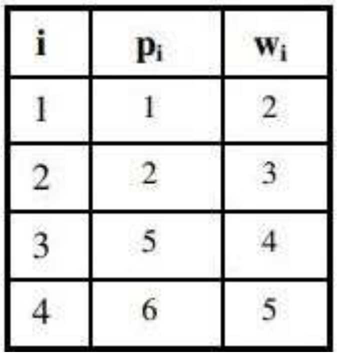
= 1 + 0 + 5 = 6

Maximum Weight = w1x1+w2x2+w3x3

= 2*1+3.0+4.1

= 2 + 0 + 4 = 6

Example 2 : Consider the following 0 / 1 Knapsack problem using dynamic programming m = 8, n = 4.



i	p_i	w_i
1	1	2
2	2	3
3	5	4
4	6	5

Solution : we have to build the sequence of decisions S^0, S^1, S^2, S^3, S^4

Initially S^0 = { (0,0) }

S^0_1 = Select next (p1,w1) pair and add it with S^0

= (1,2) + { (0,0) }

= { (1+0), (2 + 0) } = { (1,2) }

$S^{i+1}= S^i \cup S^i_1$

$S^1 = S^0 \cup S^0_1$

= { (0,0) } ∪ { (1,2) }

= { (0,0), (1,2) }

To apply Purging Rule : There will be no deleted

S^1_1 = Select next (P2,W2) pair and add it with S^1

= (2,3) + { (0,0), (1,2) }

= { (2 + 0, 3 + 0), (2 + 1, 3 + 2) }

= { (2,3) , (3,5) }

$S^2 = S^1 \cup S^1_1$

= { (0,0), (1,2) } ∪{ (2,3) , (3,5) }

= { (0,0) , (1,2), (2,3),(3,5) }

To apply Purging Rule : There will be no deleted

S^2_1 = Select next (P3,W3) pair and add it with S^2

= (5,4) + { (0,0) , (1,2), (2,3),(3,5) }

= { (5+0,4+0),(5+1,4+2),(5+2,4+3),(5+3,4+5)}

= { (5,4),(6,6), (7,7), (8,9) }

$S^3 = S^2$ ⍰ S^2_1

= { (0,0) , (1,2), (2,3),(3,5) } ⍰ { (5,4),(6,6), (7,7), (8,9) }

= { (0,0),(1,2),(2,3),(3,5),(5,4),(6,6),(7,7),(8,9) }

To apply Purging Rule

(3,5) and (5,4)
(pj,wj) and (pk,wk)

Pj<=pk and wj>=wk
3<=5 and 5>=4 -- True, (pj,wj) pair can be deleted (3,5)

S^3 = { (0,0),(1,2),(2,3),(5,4),(6,6),(7,7),(8,9) }

S^3_1 = Select next (P4,W4) pair and add it with S^3

= (6,5) + { (0,0),(1,2),(2,3),(5,4),(6,6),(7,7),(8,9) }

= { (6+0,5+0), (6+1,5+2),(6+2,5+3),(6+5,5+4),(6+6,5+6), (6+7 , 5+7),(6+8 , 5+9) }

= { (6,5),(7,7) , (8,8), (11,9), (12,11), (13,12), (14,14) }

$S^4 = S^3 \cup S^3_1$

= { (0,0),(1,2),(2,3),(5,4),(6,6),(7,7),(8,9) } ∪ { (6,5),(7,7) , (8,8), (11,9), (12,11), (13,12), (14,14) }

= { (0,0),(1,2),(2,3),(5,4), (6,5) ,(6,6),(7,7), (8,8), (8,9), (11,9), (12,11), (13,12), (14,14) }

To apply Purging Rule : There will be no deleted

Here Capacity of the Knapsack is 8, Now we have to remove the pairs, in which wi>m, i.e wi>8

Therefore S^4 = { (0,0),(1,2),(2,3),(5,4), (6,5) ,(6,6),(7,7), (8,8) }

Here M = 8, we will find the tuple, denoting the weight ‘ 8 ‘

i,e (8,8) belongs to S^4

(8,8) does not belongs to S^3

Therefore , We must set $x_4 = 1$

The pair (8,8) came from the pair (8-p4,8-w4)

(8-6, 8-5) = (2,3)

Here (2,3) belongs to S^3

(2,3) belongs to S^2

Therefore , We must set x_3 = 0

(2,3) belongs to S^2

(2,3) does not belongs to S^1

Therefore , We must set x_2 = 1

The pair (2,3) came from the pair (2-p2,3-w2)

(2-2, 3-3) = (0,0)

Here (0,0) belongs to S^1

(0,0) belongs to S^0

Therefore , We must set x_1 = 0

Hence an Optimal Solution is (x_1 , x_2, x_3, x_4) = (0, 1 , 0, 1)

Maximum Profit = p1x1+p2x2+p3x3+p4x4

= 1.0+2.1+5.0 + 6.1

= 0 + 2 + 0 + 6 = 8

Maximum Weight = w1x1+w2x2+w3x3+w4x4

= 2.0+3.1+4.0+5.1

= 0 + 3 + 0 + 5 = 8

Reliability Design:

The dynamic programming approach to solve a problem with multiplicative constraints. Let us consider the example of a computer return in which a set of nodes are connected with each other. Let r_i be the reliability of a node i.e. the probability at which the node forwards the packets correctly in r_i . Then the reliability of the path connecting from one node s to another node d is $\prod_{i=1}^{k} r_i$ where k is the number of intermediate node. Similarly, we can also consider a system with n devices connected in series, where the reliability of device i is r_i . The reliability of the system is $\prod_{i=1}^{k} r_i$.

For example if there are 5 devices connected in series and the reliability of each device is 0.99 then the reliability of the system is 0.99×0.99×0.99×0.99×0.99=0.951. Hence, it is desirable to connect multiple copies of the same devices in parallel through the use of switching circuits. The switching circuits determine the devices in any group functions properly. Then they make use of one such device at each stage. Let m_i be the number of copies of device Di in stage i. Then the probability that all m_i have malfunction i.e. (1-ri) m i .Hence, the reliability of stage i becomes 1-(1-ri) m i . Thus, if r_i =0.99 and m_i=2, the reliability of stage i is 0.9999. However, in practical situations it becomes less because the switching circuits are not fully reliable. Let us assume that the reliability of stage i in $\varphi_i(m_i)$, i≤n. Thus the reliability of the system is $\prod_{i=1}^{k} \phi_i(m_i)$.

The reliability design problem is to use multiple copies of the devices at each stage to increase reliability. However, this is to be done under a cost constraint. Let c_i be the cost of each unit of device D_i and let c be the cost constraint. Then the objective is to maximize the reliability under the condition that the total cost of the system $m_i c_i$ will be less than c.

We can assume that each $c_i>0$ and so each m_i must be in the range $1 \le m_i \le u_i$,

Now we have to find out the Upper bound (ui)

$$ui = (c + ci-\sum Cj) / Ci$$

The dynamic programming approach finds the optimal solution $m_1, m_2 \ldots m_n$. An optimal sequence of decision i.e. a decision for each m_i can result an optimal solution.

Example: Design a Three stage system with device tape D1, D2, D3. The costs are 30, 15 and 20 respectively. The cost of the system is to be no more than 105. The reliability of each device type is 0.9, 0.8 and 0.5 respectively.

Solution : Given data

$$c = 105$$

$$c1=30$$

$$c2=15$$

$$c3=20$$

$$r1=0.9$$

$$r2=0.8$$

$$r3=0.5$$

Now we have to find out the Upper bound (ui)

$$ui = (c + ci-\sum Cj) / Ci$$

$u1 = (c + c1-(c1 + c2 + c3)) / c1$

$= 105 + 30 – (30+ 15 + 20) / 30$

$= 105 + 30 -65 / 30$

$= 135-65 / 30$

$= 70 / 30 = 2$

$u2 = (c + c2-(c1 + c2 + c3)) / c2$

$= (105 + 15 – (30+ 15 + 20)) / 15$

$= (105 + 15 – (65) / 15$

$= 120-65 / 15$

$= 55 / 15 = 3$

$U3 = (c + c3-(c1 + c2 + c3)) / c3$

$= (105 + 20 – (30+ 15 + 20)) / 20$

$= (105 + 20 – (65) / 20$

$= 125-65 / 20$

$= 60 / 20 = 3$

We can require

2 Copies of D1

3 Copies of D2

3 Copies of D3

To improve the reliability of the three stage system

Let us start with S^0 = (r, c)

R= reliability

C = cost

Initially S^0 = (1, 0)

Device 1 (D1) : How many copies has to purchase

Now 1<=mi<ui, i.e, 1<=m1<=u1

1<=m1<=2

Therefore m1 = 1 or 2

S^1_1 calculated as i = 1, j= 1 and m1 = 1

Reliability is calculated as Ø1(m1) = 1 – $(1\text{-}r1)^{m1}$

= 1 – $(1\text{-}0.9)^1$

= $1\text{-}(0.1)^1$

= 1-0.1 = 0.9

S^1_1 = { (0.9 *1 , 30+0) }

= { (0.9,30) }

S^1_2 calculated as i = 1, j= 2 and m1 = 2

Reliability is calculated as Ø1(m1) = 1 – $(1\text{-}r1)^{m1}$

= 1 – $(1\text{-}0.9)^2$

= $1\text{-}(0.1)^2$

= 1-0.01 = 0.99

S^1_2 = { (0.99*1, 30*2+0) }

= { (0.99, 60) }

S^1 can be obtained by merging S^1_1, S^1_2

S^1 = { (0.9,30), (0.99, 60) }

Device 2 (D2) : How many copies has to purchase

Now 1<=mi<ui, i.e, 1<=m2<=u2

1<=m2<=3

Therefore m2 = 1 , 2 or 3

S^2_1 calculated as i = 2, j= 1 and m2 = 1

Reliability is calculated as Ø2(m2) = 1 – $(1\text{-}r2)^{m2}$

= 1 – $(1 – 0.8)^1$

= $1\text{-} (0.2)^1$

= 1-0.2 = 0.8

S^2_1 = { (0.8*0.9,30+15), (0.8*0.99,60+15) }

= { (0.72,45), (0.792,75) }

S^2_2 calculated as i = 2, j= 2 and m2 = 2

Reliability is calculated as $\emptyset 2(m2) = 1 – (1\text{-}r2)^{m2}$

$= 1 – (1 – 0.8)^2$

$= 1\text{-} (0.2)^2$

$= 1\text{-}0.04 = 0.96$

S^2_2 = { (0.96*0.9, 30+2*15), (0.96*0.99,60+2*15) }

= { (0.864,60) , (0.9504,90) }

S^2_3 calculated as i = 2, j= 3 and m2 = 3

Reliability is calculated as $\emptyset 2(m2) = 1 – (1\text{-}r2)^{m2}$

$= 1 – (1 – 0.8)^3$

$= 1\text{-} (0.2)^3$

$= 1\text{-}0.008 = 0.992$

S^2_3 = { (0.992*0.9,30+3*15) , (0.992*0.99, 60+3*15) }

= { (0.8928, 75), (0.98208, 105) }

S^2 can be obtained by merging S^2_1, S^2_2, S^2_3

S^2 = { (0.72,45), (0.792,75), (0.864,60) , (0.9504,90), (0.8928, 75), (0.98208, 105) }

To apply Purging Rule

(0.792,75), (0.864,60)

To remove (0.792,75) because cost are always increasing order

Therefore S^2 = { (0.72,45), (0.864,60) , (0.9504,90), (0.8928, 75), (0.98208, 105) }

To remove (0.9504,90), because cost are always increasing order

Therefore S^2 = { (0.72,45), (0.864,60) , (0.8928, 75), (0.98208, 105) }

Device 3 (D3) : How many copies has to purchase

Now 1<=mi<ui, i.e, 1<=m3<=u3

1<=m3<=3

Therefore m3 = 1 , 2 or 3

$S^3{}_1$ calculated as i = 3, j= 1 and m3 = 1

Reliability is calculated as $\emptyset 2(m3) = 1 – (1\text{-}r3)^{m3}$

$= 1 – (1 – 0.5)^1$

$= 1\text{-} (0.5)^1$

$= 1\text{-}0.5 = 0.5$

$S^3{}_1$ = { (0.5*0.72, 45 + 20), (0.5*0.864+ 60+20), (0.5*0.8928, 75+20), (0.5*0.98208, 105+20) }

= { (0.36,65) , (0.432,80), (0.4464,95),(0.49104,125) }

To remove the Pair (0.49104,125), it exceeds 105

$S^3{}_1$ ={ (0.36,65) , (0.432,80), (0.4464,95) }

$S^3{}_2$ calculated as i = 3, j= 2 and m3 = 2

Reliability is calculated as $\emptyset 3(m3) = 1 – (1\text{-}r3)^{m3}$

$= 1 – (1 – 0.5)^2$

$= 1\text{-} (0.5)^2$

$= 1\text{-}0.25 = 0.75$

$S^3{}_2$ = { (0.75*0.72, 45 + 2*20), (0.75*0.864, 60+2*20), (0.75*0.8928, 75+2*20),

(0.75*0.98208, 105+2*20) }

$S^3{}_2$ = { (0.54,85), (0.648,100), (0.6696, 115) , (0.73656,145) }

To remove the Pairs (0.6696, 115) , (0.73656,145) , because it exceeds 105

$S^3{}_2$ = { (0.54,85), (0.648,100) }

$S^3{}_3$ calculated as i = 3, j= 3 and m3 = 3

Reliability is calculated as $\emptyset 3(m3) = 1 – (1\text{-}r3)^{m3}$

$= 1 – (1 – 0.5)^3$

$= 1\text{-} (0.5)^3$

$= 1\text{-}0.125 = 0.875$

$S^3{}_3$={ (0.875*0.72, 45+3*20),(0.875*0.864, 60+3*20),(0.875*0.8928, 75+3*20),

(0.875*0.98208, 105+3*20) }

$S^3{}_3$ = { (0.63,105) , (0.75,120) , (0.7812,135) , (0.8575,165) }

To remove the Pairs (0.75,120) , (0.7812,135) , (0.8575,165), because it exceeds 105

$S^3{}_3$ = { (0.63,105) }

S^3 can be obtained by merging S^3_1, S^3_2, S^3_3

S^2 = { (0.36,65) , (0.432,80), (0.4464,95), (0.54,85), (0.648,100), (0.63,105) }

To apply Purging Rule

(0.4464,95), (0.54,85)

To remove (0.4464,95), because cost are always increasing order

Therefore S^3 = { (0.36,65) , (0.432,80), (0.54,85), (0.648,100), (0.63,105) }

Now remove the pair (0.63,105)

Therefore S^3 = { (0.36,65) , (0.432,80), (0.54,85), (0.648,100) }

The best design is 0.648 is the reliability with cost 100

So, (0.648,100) is present in S^3_2

Here i =3, j = 2 , m3 =2

(0.648,100) can be obtained from (0.864,60) which is present in S^2_2

Here i =2, j = 2 , m2 =2

(0.864,60) can be obtained from (0.9,30) which is present in S^1_1

Here i =1, j = 1 , m1 =1

Therefore

m1 =1

m2= 2

m3 =2

We require 1 copy of device D1

2 copy of device D2

2 copy of device D3