Unit III – Dynamic Programming

- Dynamic Programming
  - Computing a Binomial Coefficient
  - Warshall’s algorithm
  - **Floyd’s algorithm**
  - **Optimal Binary Search Trees**
  - Knapsack Problem and Memory functions

Knapsack problem and memory Functions

- Knapsack problem – given n items with weight and values. Have to find the subset of items that find the knapsack capacity W with highest value.
- Dynamic Programming – Divide the subsets into 2 categories
  - Don't include i^{th} element → V[i-1, j] , $j-w_i<0$
  - Includes ith element → max{V[i-1, j], v_i + V[i-1,j-w1]}, $j-w_i>=0$

Example – **capacity – W = 5**

Item	1	2	3	4
Weight	2	1	3	2
value	12	10	20	15

- **To find : V[n,W] – maximal value of subset of the n given items that fit the knapsack of capaity 5 →v[4,5]**

Knapsack problem and memory Functions

Item	1	2	3	4
W	2	1	3	2
V	12	10	20	15

V[n,W] → V [4, 5]

Initial Conditions:

1. v[0,j] = 0
2. V[i,0] = 0

Formula:

1. V[i-1, j] , $j-w_i<0$
2. max{V[i-1, j], v_i + V[i-1,j-w1]}, $j-w_i>=0$

Capacity j

i	0	1	2	3	4	5
0	0	0	0	0	0	0
1						
2						
3						
4						

Capacity j

i	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0					
3	0					
4	0					

Item	1	2	3	4
W	2	1	3	2
V	12	10	20	15

W1=2, v1=12

W2=1, v2=10

W3=3, v3=20

W4=2, v4=15

Formula:

1. V[i-1, j] , $j\text{-}w_i<0$

2. max{V[i-1, j], v_i + V[i-1,j-w_i]} , $j\text{-}w_i>=0$

i=1, j=2,wi=w1=2,vi=v1=12

Max{v[0,2], v1+ v[0, 0]}

Max{0, 12+0}

Max{0,12}

12

$j\text{-}w_i$	Formula (i=1)
1-2 = -1	V[0,1] = 0
2-2 = 0	max(V[0,2] , v1+v[0,0])= Max(0,12+0) = max(0,12) =12
3-2 = 1	V[0,3] , v1+v[0,1] = 0,12+0
4-2 = 2	V[0,4] , v1+v[0,2] = 0,12+0
5-2 = 3	V[0,5] , v1+v[0,3] = 0,12+0

Capacity j

i	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0	10	12			
3	0					
4	0					

Item	1	2	3	4
W	2	1	3	2
V	12	10	20	15

W1=2, v1=12

W2=1, v2=10

W3=3, v3=20

W4=2, v4=15

Formula:

1. V[i-1, j] , $j-w_i<0$

2. max{V[i-1, j], v_i + V[i-1,j-w_i]}, $j-w_i>=0$

$j-w_i$	Formula (i=2)
1-1 = 0	Max(V[1,1], 10+v[1,0]) Max(0, 10+0)= max(0,10) =10
2-1 = 1	Max(v[1,2], 10+ v[1,1]) Max(12, 10+0) = 12
3-1 = 2	
4- 1= 3	
5-1 = 4	

Capacity j

i	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0	10	12	22	22	22
3	0	10	12	22	30	32
4	0	10	15	25	30	37

Item	1	2	3	4
W	2	1	3	2
V	12	10	20	15

W1=2, v1=12

W2=1, v2=10

W3=3, v3=20

W4=2, v4=15

Formula:

1. V[i-1, j] , $j-w_i<0$

2. max{V[i-1, j], v_i + V[i-1,$j-w_i$]}, $j-w_i>=0$

C[i ,j] = C[4,5]

1. here i=4, j=5

2. j- wi = j-w1 =

Capacity j

i	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0	10	12	22	22	22
3	0	10	12	22	30	32
4	0	10	15	25	30	**37**

Item	1	2	3	4
W	2	1	3	2
V	12	10	20	15

W1=2, v1=12

W2=1, v2=10

W3=3, v3=20

W4=2, v4=15

Optimal Subset (tracking back the table for finding the subset)

Maximum value V(4,5) = 37

Optimal Subset = {4,2,1}= 15+10+12 = 37

Knapsack Capacity W= 5 → 5-2 =3-1=2-2 = 0