LECTURE # 20

3 - Jan - 2023

Wednesday

→ Dynamic Programming –

Use solution of the subproblems to set the solution of the Larger problem.

Subset Sum Problem

given a total and a set of non-negative integers, is there a subset that adds upto the total.

Total = 11 {2, 3, 7, 8, 10}

A →	0	1	2	3	4	5	6	7	8	9	10	11
2	(T)	F	T	[F]	F	F	F	F	F	F	F	F
3	T	F	T	(T)	F	T	F	[F]	F	F	F	F
7	T	F	T	(T)	F	T	F	T	[F]	T	T	F
8	T	F	T	T	F	T	F	T	T	T	[T]	(T)
10	T	F	T	T	F	T	F	T	T	T	T	(T)

Selection → A[i, j] = ~~A[i-1, j] OR~~ (j - i)

A[i-1, j] OR A[i-1, A]

(3, 8) → Backtracking (only when last one is True).

- Because we took them from above of previous Row.

→ When you are moving upward then don't consider the corresponding element.

Example # 2

From junaid notes Total = 14

Algorithm

```
if ( j < A[i])
{
    Mat[i,j] = Mat[i-1,j]
}
else
{
    Mat[i,j] = Mat[i-1,j] || Mat[i-1, j - A[i]]
}
```

→ **Coin changing Problem**

Total = 11 Coins = 1, 5, 6, 8

Optimal = minimum no. of coins eg # 1 coin of (5) instead of 5 coins of (1).

- Towards minimum no. of coins.

Coin ↓ / rupees →	0	1	2	3	4	5	6	7	8	9	10	11
(11)	0	1	2	3	4	[5]	6	7	8	9	10	11
5	(0)	1	2	3	4	(1)	2	3	4	5	2	3
6	0	1	2	3	4	(1)	1	2	3	4	(2)	(2)
8	0	1	2	3	4	1	1	2	1	2	(2)	(2)

[5] − 5 = 0 + 1 = 1 (Total no. of coins to pay RS)

0 → same row

picking minimum

→ When move upward, we don't add

{5, 6}

Exit point → when hit first Column

Algorithm

```
if ( j ≥ A[i])
{
    Mat[i,j] = min { Mat[i-1,j],
                     Mat[i,j-A[i]] + 1
}
else
{
    Mat[i,j] = Mat[i-1,j]
```

Example # 3 (From Home)

Total = 15 {1, 2, 5, 10}

→ Cutting Rod to maximize profit

Length = 5

1	2	3	4	→ Lengths
↓	↓	↓	↓	
2	5	7	8	→ profits

		0	1	2	3	4	5
(2)	1	0	(2)	4	6	8	10
(5)	2	0	(2)	(5)	(7)	10	(12)
(7)	3	0	2	(8)	7	10	(12)
(8)	4	0	2	5	7	10	(12) → optimal

2-2=0 +5 = (5 and 4) = 5 (greater)

Backtracking →

{1, 2, 2} ~

{2, 3} ✓

Algorithm

Length = 5

```
if (j ≥ i)
{
    Mat[i,j] = max { Mat[i-1,j],
                     Mat[i,j-i] + A[i]
}
else
{
    Mat[i,j] = Mat[i-1,j]
}
```