### UE19CS251

### DESIGN AND ANALYSIS OF ALGORITHMS

# Unit 5: Limitations of Algorithmic Power and Coping with the Limitations

## Memory Function Knapsack PES University

#### Concepts covered

- Memory Function Knapsack
  - Motivation
  - Algorithm
  - Example

## 1 Bottom Up Approach

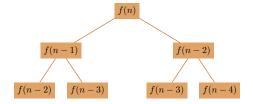
- Advantage of bottom up approach: each value computed only once
- Example computed bottom up:

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

• Disadvantage of bottom up approach: values not required also computed

## 2 Top Down Approach

- Disadvantage of top down approach: same problem solved multiple times
- Example computed top down:



• Advantage of top down approach: only the required subproblems solved

#### 3 Memory Function Dynamic Programming

- Combine the advantages of bottom up and top down approaches:
  - compute each subproblem only once
  - compute only the required subproblems

#### 4 MF-DP Algorithm

```
Algorithm for Memory Function Dynamic Programming
 1: procedure MFKNAPSACK(i, j)
       \triangleright Inputs: i indicating the number items, and
       \triangleright j, indicating the knapsack capacity

ightharpoonup Output: The value of an optimal feasible subset of the first i items
       \triangleright Note: Uses global variables input arrays Weights[1...n],
    Values[1 \dots n],
       \triangleright and table F[0...n,0...W] whose entries are initialized with -1's
 6:
    except
       \triangleright row 0 and column 0 is initialized with 0
 7:
        if F[i,j] < 0 then
 8:
9:
           if j < Weights[i] then
10:
               value \leftarrow MFKnapsack(i-1,j)
           elsevalue \leftarrow max(MFKnapsack(i-1,j),
                                                                 Values[i] +
11:
    MFKnapsack(i-1, j-Weights[i]))
12:
               F[i,j] \leftarrow value
13:
        return F[i,j]
```

## 5 Example

$$F(i,j) = \begin{cases} \max(F(i-1,j), & v_i + F(i-1,j-w_i)) & \text{if } j - w_i \ge 0 \\ F(i-1,j) & \text{if } j - w_i < 0 \end{cases}$$

item $i$	weight $w_i$	value $v_i$					capa	acity $j$		
1	2	12	1	i	0	1	2	3	4	5
2	1	10	(	0	0	$\Box 0$	$\Box 0$	$\Box 0$	$\Box 0$	$\Box 0$
3	3	20		1	$\Box 0$	$\Box 0$	$\Box 12$	$\Box 12$	$\Box 12$	$\Box 12$
4	2	15	6	2	$\Box 0$	_	$\Box 12$	$\Box 22$	_	$\Box 22$
What is the maximum value that can be stored in a knapsack of capacity 5?				3 4 inaj		$ \Box 0 \Box 22 - \Box 0 \Box 2 $ sack problem solved by				□32 □37
• computing 21 out 30 poss subproblems								ssible		

## 6 Complexity

• Constant factor improvement in efficiency

- Space complexity:  $\Theta(nW)$ 

– Time complexity:  $\Theta(nW)$ 

- Time to compose optimal solution: O(n)

• Bigger gains possible where computation of a subproblem takes more than constant time

#### 7 Think About It

• Consider the use of the MF technique to compute binomial coefficient using the recurrence

$$C(n,k) = C(n-1,k-1) + C(n-1,k)$$

- How many table entires are filled?
- How many are reused?