COT 5407: Introduction to Algorithms

Giri NARASIMHAN

Approach to DP Problems

- Write down a recursive solution
- Use recursive solution to identify list of subproblems to solve (there must be overlapping subproblems for effective DP)
- Decide a data structure to store solutions to subproblems (MEMOIZATION)
- Write down Recurrence relation for solutions of subproblems
- Identify a hierarchy/order for subproblems
- Write down non-recursive solution/algorithm
1-d, 2-d, 3-d Dynamic Programming

- Classification based on the dimension of the table used to store solutions to subproblems.

1-dimensional DP
- Activity Problem

2-dimensional DP
- LCS Problem
- 0-1 Knapsack Problem
- Matrix-chain multiplication

3-dimensional DP
- All-pairs shortest paths problem
Location of parentheses in chain

- MCP[1,n]
  - MCP[1,k]
    - MCP[1,j]
    - MCP[j+1,k]
  - MCP[k+1,n]
    - MCP[k+1, p]
    - MCP[p+1, n]
Matrix Chain Product

- $\text{MCP}[1,n] = \text{Min}$
  - $\text{MCP}[1,k] + \text{MCP}[k+1,n] + \text{cost}(1,k,n)$
  - Since we don’t know the value of $k$
    - We try every possible value of $k$
Shortest Leash Problem ... 1

- $L[k,j] = \text{shortest leash for a walk from start to } k\text{-th stop for dog walker and } j\text{-th stop for dog}$

- $L[k,j] = \text{Min of 2 possibilities}$
  - $\text{Max}\{ L[k-1, j-1], ssd[k-1, j-1] \}$
  - $\text{Max}\{ L[k-1, j], spd[k-1, j] \}$
  - $\text{Max}\{ L[k, j-1], psd[k, j-1] \}$