COT 5407: Introduction to Algorithms

Giri NARASIMHAN

www.cs.fiu.edu/~giri/teach/5407S19.html

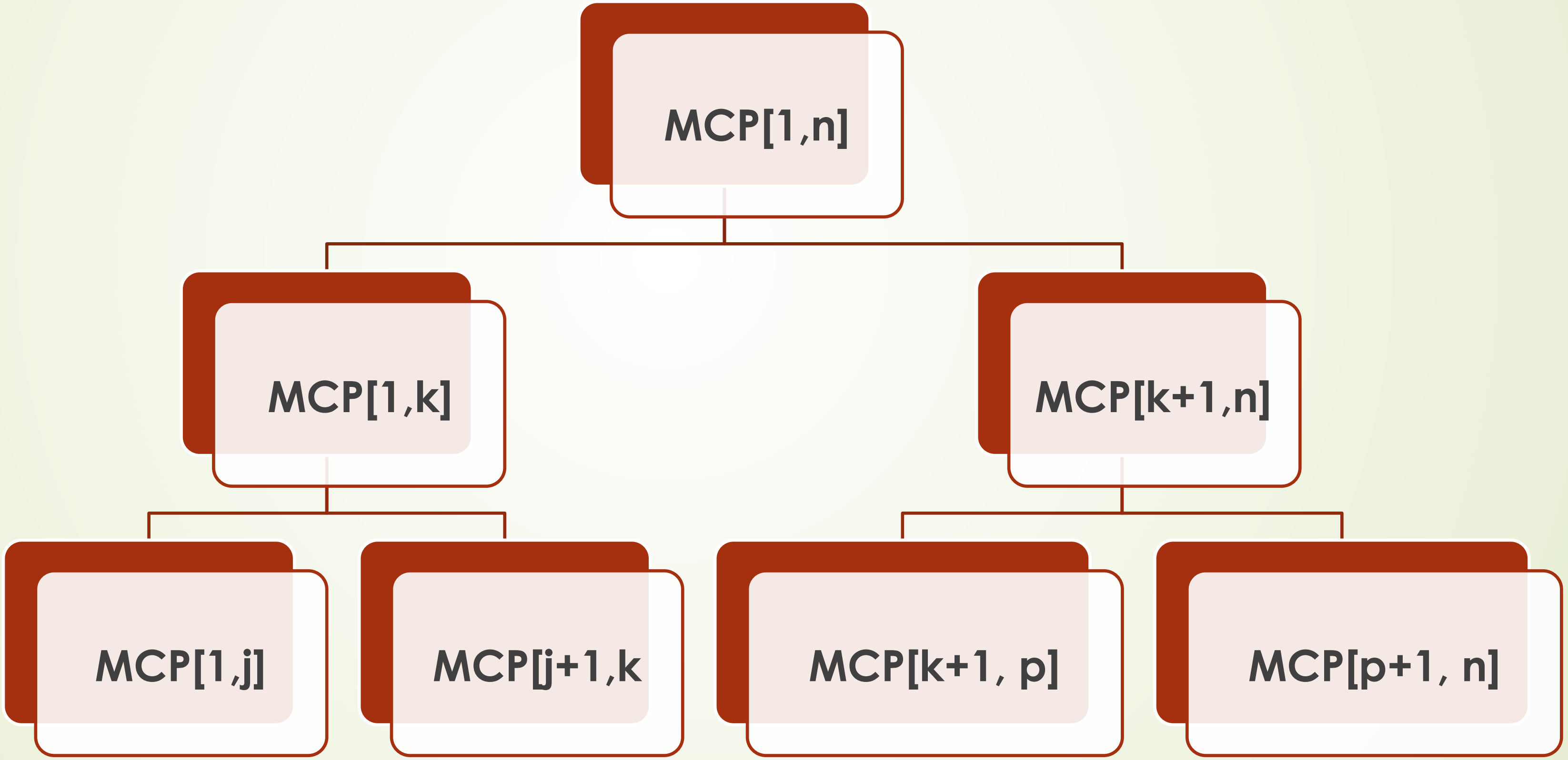
Approach to DP Problems

- 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

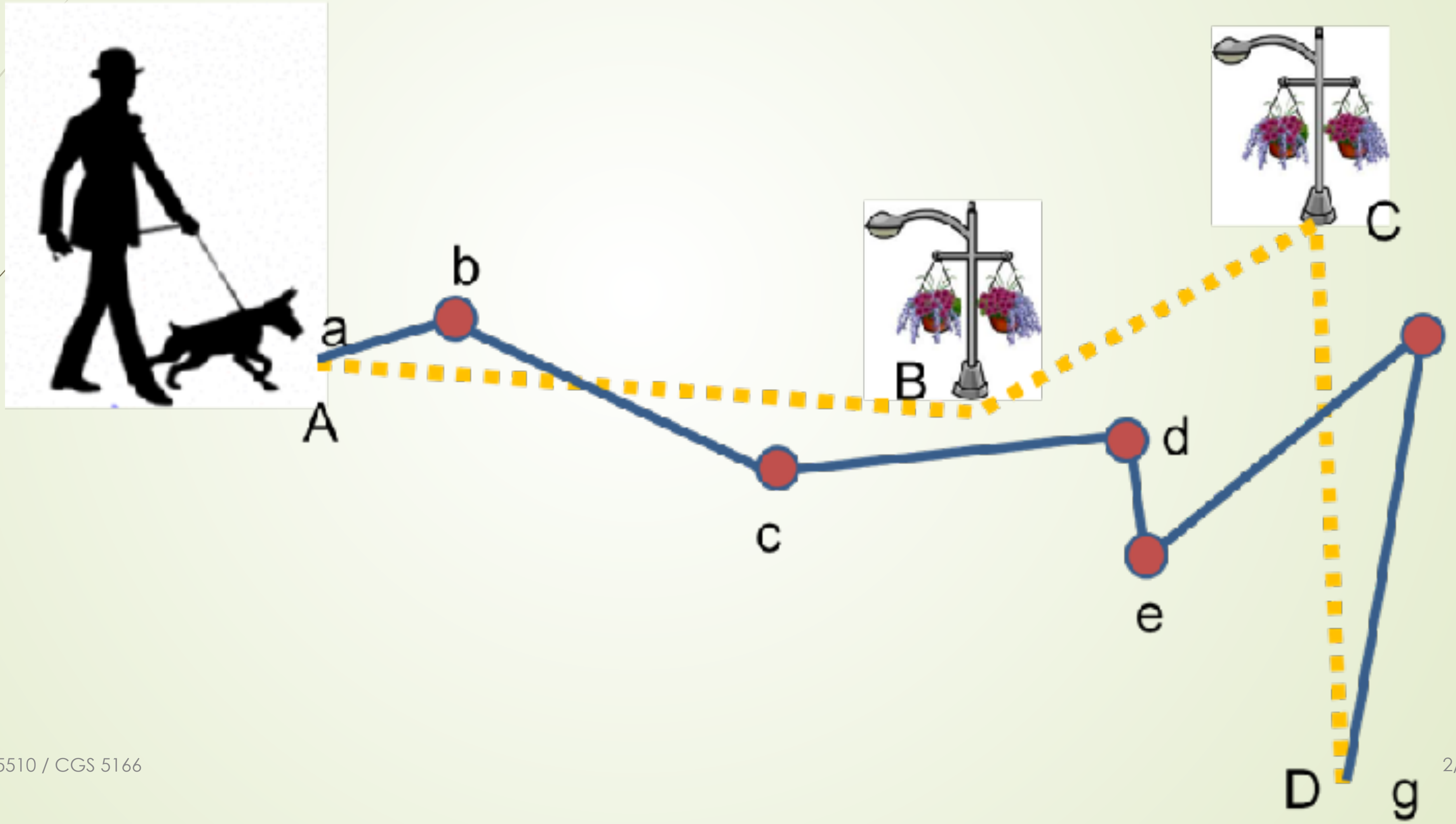
Location of parentheses in chain



Matrix Chain Product

- MCP[1,n] = Min
  - MCP[1,k] + MCP[k+1,n] + cost(1,k,n)
  - Since we don't know the value of k
    - We try every possible value of k

Shortest Leash Problem



Shortest Leash Problem … 1

- L[k,j] = shortest leash for a walk from start to k-th stop for dog walker and j-th stop for dog
- L[k,j] = Min of 2 possibilities
  - Max{ L[k-1, j-1], ssd[k-1, j-1]}
  - Max{L[k-1, j], spd[k-1, j]}
  - Max{L[k, j-1], psd[k, j-1]}