18.7
Dynamic Programming: Postscript

Dynamic Programming: Postscript

Dynamic Programming = Smart Recursion + Memoization

1. How to come up with the recursion?
2. How to recognize that dynamic programming may apply?

Dynamic Programming: Postscript

Dynamic Programming = Smart Recursion + Memoization

1. How to come up with the recursion?
2. How to recognize that dynamic programming may apply?

Some Tips

1. Problems where there is a natural linear ordering: sequences, paths, intervals, DAGs etc. Recursion based on ordering (left to right or right to left or topological sort) usually works.
2. Problems involving trees: recursion based on subtrees.
3. More generally:
    1. Problem admits a natural recursive divide and conquer
    2. If optimal solution for whole problem can be simply composed from optimal solution for each separate pieces then plain divide and conquer works directly
    3. If optimal solution depends on all pieces then can apply dynamic programming if interface/interaction between pieces is limited. Augment recursion to not simply find an optimum solution but also an optimum solution for each possible way to interact with the other pieces.

Examples

1. Longest Increasing Subsequence: break sequence in the middle say. What is the interaction between the two pieces in a solution?
2. Sequence Alignment: break both sequences in two pieces each. What is the interaction between the two sets of pieces?
3. Independent Set in a Tree: break tree at root into subtrees. What is the interaction between the subtrees?
4. Independent Set in an graph: break graph into two graphs. What is the interaction? Very high!
5. Knapsack: Split items into two sets of half each. What is the interaction?