# Dynamic Programming (II)

Anson Ho

# Contents

- DAG DP
- Dimension reduction
- Tree DP
- Bitwise DP

# What is DP

- State(s)
- Transition formula
- Base case(s)

- Memorization

- Time complexity
- Space complexity

# DAG

- Directed acyclic graph
- Visualization of transition

# DAG

- Horner's method
- Fibonacci number
- Number of sequences of strictly including rectangles with integral dimensions

# DAG

- Number of ways to pay $10
- Calculate $nCr = \frac{n!}{r!(n-r)!}$
- Shortest path

# Dimension reduction

- Rolling array

- Reduce dimension for space
- Not for time

# Dimension reduction

- Number of paths in grids
  - From top-left to bottom right

- 0: Only ↓ and → are allowed
- 1: Some blocked grids
- 2: 5000x5000 grid
  5000 blocked  1MB memory
- 3: output “median” path

# Dimension reduction

- Avoid moving a large chunk of memory

- dp[i][j]
- = f(dp[i-1][j],dp[i-2][j+1])
- Only keep the last two dp[i]

- i → i mod 3
- i-1 → i-1 mod 3
- i-2 → i-2 mod 3

Take a break

# Tree DP

- What is tree?

- A Tree is a DAG
- (A DAG is not necessarily a tree)

- Rooted or not?

# Tree DP

- Rooted
- Height of each node
- Size of each subtree
- Minimax in a game tree

# Tree DP

- Not rooted
  - bidirectional edges
- Number of paths passing through each node
- Sum of path weights
  - path weight = sum of edge weights
  - path weight = product of edge weights

# Tree DP

- Number of “k-subtrees” in unrooted tree
- “k-subtree” means a subgraph which is a tree with k nodes

# Tree DP

- Some of the nodes in a tree are coloured
- For every node, find the number of paths with coloured ends passing through it

# (Extra)

- Every node in a tree is coloured

- Queries with a specific node and a specific colour
- Find the number of paths with ends in that colour passing through that node

# Tree DP

- Given two rooted tree
- Find the minimum number of additional nodes to make the two trees “isomorphic”
- Assignment problem

# Bitmask DP

- State
- dp[133][2] → dp[$101100101_2$]

- Bit manipulation
  - bitwise and (&)
  - bitwise or (|)
  - exclusive or (^)

# Bitmask DP

- Assignment problem
- Matching

- Maximum matching
- Matching with minimum cost

- Polynomial time algorithm exists

# Bitmask DP

- Hamiltonian path
- Graph traversal with visiting each node exactly once

# Bitmask DP

- N (<=15) light bulbs
- K (<=30) buttons
  - each control a set of light bulbs

- Find the minimum number of toggling to achieve a specific configuration

# Bitmask DP

- Number of ways to fill a NxM grid with 1x2 (or 2x1) rectangle
  - fully filled
  - no overlapping

- N=2, M=2
- Ans = 2

# Bitmask DP

- Define 2, 8, 9 as lucky digit
- Define numbers containing only lucky digits as lucky number

- Find the least lucky multiple of 369

# Bitmask DP

- 29889 = 369 x 81

- State 1: cur mod 369
- State 2: bitmask of used lucky digit
- DP from leftmost digit

- Ref: HackerEarth Lucky Digit

Thank you