View from 30,000 feet and Introduction to Graph Problems
Lecture 19
CS 312

View from 30,000 feet

- Greedy
  - simple, but need a selection function
- Divide and Conquer
  - solve subproblems and recombine

View from 30,000 feet

- Dynamic programming
  - solving the same subproblems many times
  - bottom-up: iterative but may not need all subsolutions
  - memory functions: recursive but only solve the subproblems you need.
- Exploring graphs
  - unordered graph exploration

Objective

- Describe a simple game using a graph
- Explain why only some of the graph can be stored in memory

Nim (or the Marienbad game)

- Player that removes the last match wins
- Rules:
  - in the first turn: must take at least one and leave at least one
  - each following turn: must take at least one and may take at most 2x what your opponent took.

Game Graph
Finding winning positions

- (0,0) is a losing position
- A winning position has at least one transition to a losing position
- (n,n) has a transition to (0,0)
- A losing position has no transitions to a losing position

Finding winning positions

function recwin(i,j)
for k := 1 to j do
  if not recwin(i-k, min(2k, i-k))
    then return true
  return false

Will recwin terminate? When?

Dynamic Programming?

- Bottom up
  - Compute 121 for (15,14) but only need 28
  - Compute >30,000 for (248,247) but only need 1,000
- Top down
  - Use a NxN table
  - Trade space for time

Harder games

- Chess, proofs, life (the real one)
- Can't build the entire graph.
- Only have an implicit representation
  - Know how to build the graph, but don't have the graph
- Backtracking and Branch-and-bound.