Review material for Midterm II

1. Lecture notes from 1/26 to 2/23
2. Sections 15.1–15.4, 16.1–16.3, 22.1–22.4
3. Problem sets 4, 5, 6 and solutions (available at canvas/files)
Topics

1. Greedy algorithms
2. Dynamic programming
3. Graph terminology, basic properties
4. Elementary graph algorithms
1. Greedy algorithms

- A greedy algorithm always makes the choice that looks best at the moment, without regard for future consequence “take what you can get now” strategy.

- The proof of the greedy algorithm producing the solution of maximum size of compatible activities is based on the following two key properties:
  - The greedy-choice property
    - a globally optimal solution can be arrived at by making a locally optimal (greedy) choice.
  - The optimal substructure property
    - an optimal solution to the problem contains within it optimal solution to subprograms.

- Greedy algorithms do not always yield optimal solutions, but for many problems they do.
1. Greedy algorithms

Examples:

1. Activity selection problems
2. Job scheduling (homework 4)
3. Huffman coding
4. 0-1 Knapsack problem
5. Money-change problem
2. Dynamic programming

Four-step algorithmic paradigm

1. Characterize the structure of an optimal solution
2. Recursively define the value of an optimal solution
3. Compute the value of an optimal solution in a bottom-up fashion
4. Construct an optimal solution from computed information
2. Dynamic programming

Three key ingredients:

1. **Optimal substructure**
   the optimal solution to the problem contains optimal solutions to subprograms $\rightarrow$ recursive algorithm

2. **Overlapping subproblems**
   There are few subproblems in total, and many recurring instances of each

3. **Memoization**
   after computing solutions to subproblems, store in table, subsequent calls do table lookup.
2. Dynamic programming

Examples:

1. Rod cutting problem (with and without cut cost)
2. Matrix-chain multiplication
3. Longest common subsequence/substring
4. Edit distance
5. 0-1 knapsack problem
6. Money-change problem
3. Graph – intro

- **Definitions:**
  graph, directed/undirected, path, connected, cycle (acyclic), dag, tree, ...

- **Graph representations:**
  adjacency matrix, incidence matrix, adjacency list

- **Basic properties** – lecture notes and homework 6
3. Graph – elementary algorithms

- Two archetype elementary graph algorithms:
  1. Breadth-first search (BFS)
  2. Depth-first search (DFS)

  Issues: features, data structure, output, running time

- Applications of BFS and DFS
  - Cycle
  - Connected component
  - Topological sort
  - Bipartite graph
  - Sink
Extra. Data structures for graph algorithms

1. FIFO queue
2. LIFO stack
Have Fun and Good Luck