CS 332 Exam 2 - Review Problems

Fall, 2002

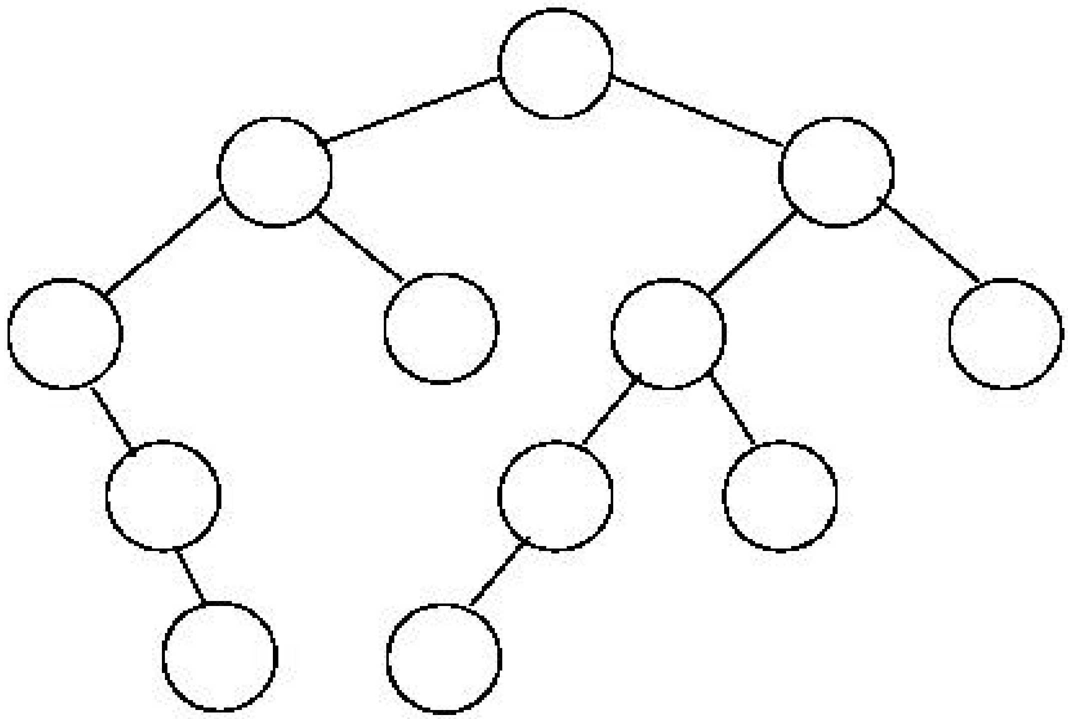
For this exam you are still responsible for the basic material from the first exam. So, for example, you should know basic facts about Big-O notation, the basic sorting algorithms, and the three basic algorithm design techniques, greedy, divide and conquer, and dynamic programming.

Here is a list of what we have covered since the first exam. (The second exam is mostly about this material.)

- Trees and tree traversals (pre-order, in-oreder, post-order, and level-order).
- Graphs, both directed and undirected, and their representations using adjacency lists or an adjacency matrix
- DFS (both recursively and explicitly using a stack) and BFS (using a queue).
- Various spanning trees, e.g., DFS spanning tree, BFS spanning tree, MST, shortest path spanning tree.
- Prim's, Kruskal's, and Dijkstra's algorithms.
- Knapsack problems.
- Greedy algorithm for the fractional knapsack problem.
- Dynamic programming algorithm for the integer 0-1 knapsack problem.
- Approximation algorithms for the 0-1 knapsack problem and the vertex cover problem.
- Reductions
- NP-Completeness
- You should be familiar with the following NP-Complete problems: Traveling Salesman Problem, Hamiltonian Cycle, Vertex Cover, Independent Set, Clique, Satisfiability and 3-SAT. The graph problems are described in the textbook in Sections 8.5.1-8.5.5, pages 311-325. Satisfiability is described in the textbook in Section 8.3.10, pages 266-268.

Problem 1: Insert keys into the binary tree structure so that a postorder traversal will give the following ordering of the keys.

8	3	5	11	14	4	2	1	9	7	6	0



Problem 2: Suppose you have a tree with vertices $V = \{0, 1, 2, 3, 4, 5, 6, 7\}$ and with the adjacency list equal to:

0: 1, 3, 4, 7
1: 0
2: 6
3: 0
4: 0, 5, 6
5: 4
6: 2, 4
7: 0

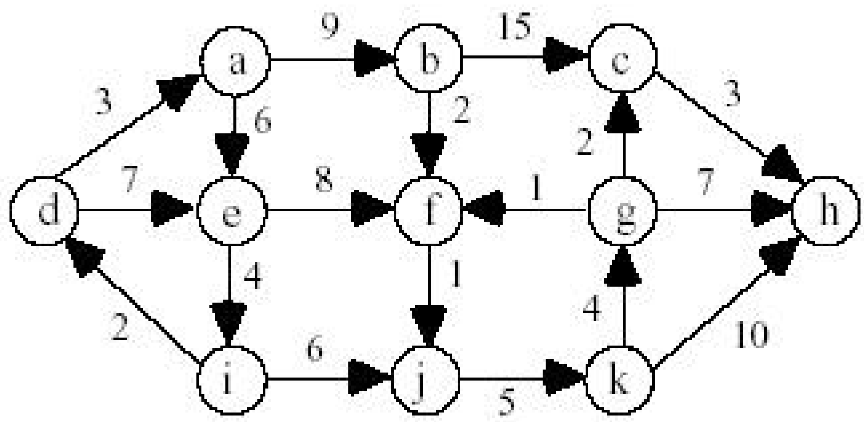
Show how this tree could be represented in an array of size 8. Explain what you are doing. (Hint: Look at the tree from the bottom up.)

0	1	2	3	4	5	6	7

Problem 3: Explain why an undirected graph with n vertices can have at most $\frac{n(n-1)}{2}$ edges. (Hint: Think of the adjacency matrix representation for the graph.)

Problem 4: Consider the undirected graph G with vertices $V = \{a, b, c, d, e, f, g\}$ and edges $E = \{(a, b), (a, c), (a, d), (a, f), (a, g), (b, c), (b, g), (d, e), (d, f), (e, f)\}$.
(a) Draw a picture of the graph. Then draw an adjacency list representation of the graph. Here each vertex sees its neighbors in alphabetical order.
(b) Draw the BFS spanning tree achieved by starting at vertex a, numbering each node as it is inserted and visiting neighbors in alphabetic order. Show cross-edges.
(c) Draw the DFS spanning tree achieved by starting at vertex a, numbering each node as it is inserted and visiting neighbors in alphabetic order. Show back-edges.

Problem 5: Draw the DFS spanning tree that is induced by performing a depth-first-search starting at node a in the following directed graph. Iterate through adjacent vertices in alphabetical order. Label each vertex by its discovery and finish times (as in the book by Cormen, Leiserson, and Rivest (CLR), p. 479). Using dotted lines, draw the edges of G that are not in the DFS spanning tree and label them as forward edges (F), back edges (B), or cross edges (C).



Problem 6: Draw a single directed graph G that has all of the following features:

5 vertices $V = \{A, B, C, \mathrm{D}, E\}$
8 edges
no self edges (edges from a vertex to itself)
2 weakly connected components
3 strongly connected components

See Section 8.4.1, pages 270-271, of the course textbook for a definition of weakly and strongly connected components of a graph.

Problem 7: (a) If all of the edges in a graph have different lengths, then there is only one Minimum Spanning Tree. Prove that the MST must contain the shortest and the second shortest edges (assume that the graph has at least two edges, i.e., $2 \le |E|$).
(b) Give an example of a graph with at least three edges in which the edges all have different lengths and the MST does not include the third shortest edge.
(c) True or False: In a graph with at least three edges where all of the edges have different lengths, the MST cannot contain the longest edge in the graph.

Problem 8: (A modified version of CLR 25.2-2, p. 531)
Dijkstra's algorithm assumes that the edge weights of the given graph are non-negative. Here we consider some of the consequences of allowing negative edge weights.
a) A negative-weight cycle of a weighted graph is a cyclic path whose path weight is negative. Suppose that *G* is a directed graph with a negative-weight cycle. Explain why the single-source shortest path problem may not be well-defined on such a graph.
b) If a graph *G* has some edges with negative weights but no negative-weight cycles, then the single-source shortest path problem is still well-defined. However, Dijkstra's algorithm is not guaranteed to correctly solve the problem in the presence of negative
edge weights. Construct a connected, directed, weighted graph with four vertices such that Dijkstra's algorithm gives an incorrect solution to the single-source shortest path problem.

Problem 9: Let $S = \{ a, b, c, d, e, f, g \}$ be a collection of objects with weight-value pairs as follows: *a* :(4,12), *b*:(6,10), *c*:(5,8), *d*:(7,11), *e*:(3,14), *f*:(1,7), *g*:(6,9). Assume that we have a knapsack that can hold objects with total weight 18.
Part (a) Find the optimal solution to the fractional knapsack problem.
Part (b) Find the optimal solution to the 0-1 knapsack problem.

Problem 10: Sally is hosting an auction to sell *n* widgets. She receives *m* bids, each of the form "I want k_i widgets for d_i dollars", for $i = 1 .. m$. Characterize her optimization problem as a knapsack problem. Under what conditions is this a 0-1 versus a fractional knapsack problem?

Problem 11: Draw an example of a graph with 10 vertices and 20 edges that has a Hamiltonian cycle. Also, draw an example of a graph with 10 vertices and 20 edges that does not have a Hamiltonian cycle.

Problem 12: Give an example of a graph *G* with at least 10 vertices such that the greedy 2-approximation algorithm for the Vertex Cover problem given in Section 6.8.1, page 157, of the textbook, is guaranteed to give a suboptimal vertex cover.