CSE 202 Homework 2
Fall, 2020
Due October 23, 11:59 PM
Graph search, using and modifying algorithms, fitting data structures to algorithms.
All graded parts are worth 20 points.

Exercises

Example of graph search Consider a directed graph with vertices A, B, C, D, E, F, and edges $(A, B), (A, C), (B, C), (B, E), (C, D), (C, F), (D, F), (D, A), (D, E), (E, F), (F, E)$. Show a depth-first search tree starting at A, and a breadth-first search tree starting at A. What are the strongly connected components of this graph?

Round trip times The round trip distance between s and t in a strongly connected weighted directed graph is the minimum total lengths of edges of a cycle that contains both s and t. Give an equivalent definition in terms of the shortest path distances between vertices. Then use this to give as efficient an algorithm as possible to, given s and t, compute the round trip distance between them.

Finding the largest k values in an array Give an $O(n \log k)$ time algorithm to find the largest k values in an unsorted array.

Ungraded problems

Nearest k points Give an $O(k|V|)$ time algorithm that given a vertex s in a directed weighted graph, finds the k points that are closest to s. You can refer to other algorithms presented in class and use properties of these algorithms proved in class to modify them. (5 points correct algorithm , 5 points correctness argument, 5 points efficiency, 5 points time analysis, incl.data structures)

Matrix sizes You can only multiply $r_1 \times c_1$ and $r_2 \times c_2$ dimensional matrices if $c_1 = r_2$, and the result is a $r_1 \times c_2$ dimensional matrix. Say that you are given a list of pairs of integers representing dimensions of matrix variables $M_1, ...M_n$, where M_i is a $r_i \times c_i$ dimensional matrix. You want to find a list of all possible dimensions (r, c) of products of sequences of matrices from among $M_1...M_n$. Give an efficient algorithm for this problem. (10 points correct algorithm, 10 points efficiency).

Top k elements in heap: 20 pts Give an efficient algorithm that, given a binary max-heap H of size n and a number $1 \leq k \leq n$, returns the k largest elements of H. Analyze the time in terms of both k and n (although, for the best algorithm I know, the time does not depend on n at all, just k.) (5 points correct algorithm, 15 points efficiency).

Problems

Monster hunting In a role-playing game, your character can choose among many monsters to fight in some order. For each of n monsters, you have a list of tools or weapons that can defeat the monster, and a list of tools or weapons found in the monster's lair after defeating it. You start the game with a rusty knife. You wish to know whether you can defeat the "big boss" monster through a series of monster battles. Consider two versions of the problem. In the first, you can only carry one weapon or tool at a time, and so can only pick up a single item from each monster, and need to trade your current item to do so. In the other, you can carry any number of items with you as you go. For each version, give an efficient algorithm to determine, if possible, a sequence of monsters to defeat and items to pick up, ending in defeating the boss monster. Analyse the running time in terms of n the number of monsters, and m, the total length of all lists of usable and available weapons and tools. (14 points correct algorithm and correctness proof, 6 points efficiency and time analysis. My best time is $O(n+m)$ for each version. Hint: the versions part is a trick question.)

Comparison constraints You are given n non-negative real-valued variables $x_1, ..x_n$ and a list of m comparison constraints, each either of the form $c \leq x_i$ for some given positive real number c, or $x_i \leq x_j$. (Since the variables take on non-negative values, we can assume we have constraints $0 \leq x_i$ for each $1 \leq i \leq n$. This increases the number of constraints by at most n.) You wish to find the assignment of real number values to variables, $a_1..a_n$, so that $x_i = a_i$ satisfies all constraints, that minimizes $\sum_i a_i$. Give the most efficient algorithm you can think of for this problem. (10 points correct poly-time algorithm; 10 points efficiency).

Top k elements in binary search tree Give an $O(k+d)$ time algorithm to find the k largest elements in a binary search tree of maximum depth d. (5 points clear algorithm description, 5 points correctness argument, 10 points efficiency and completeness of time analysis).

Implementation problem: arrays vs heaps in Dijkstra's algorithm Consider Dijkstra's algorithm for random directed graphs where each edge is present with probability p, and edges have weights that are uniform reals between 0 and 1. For a wide variety of numbers of vertices n, experimentally determine the value of p where you should switch from the array implementation to the heap implementation. How does p grow with n? Be sure to present your data clearly.