CS325H: Analysis of Algorithms, Fall 2020

Practice Problems 3

Problem 1.

(a) Find a graph that has multiple minimum spanning trees.

(b) Prove that any graph with distinct edge weights has a unique minimum spanning tree.

(c) Find a graph with non-distinct edge weights that has a unique minimum spanning tree (can you generalize (b)?).

Problem 2. Let $G = (V, E)$ be an arbitrary connected graph with weighted edges.

(a) Prove that for any cycle in G, the minimum spanning tree of G excludes the maximum-weight edge in that cycle.

(b) Prove or disprove: The minimum spanning tree of G includes the minimum-weight edge in every cycle in G.

Problem 3. Recall the job scheduling problem. The input is composed of the starting and finishing times of n jobs. We would like to find the maximum set of pairwise disjoint jobs. Consider the following alternative greedy algorithms for the job scheduling problem. For each algorithm, either prove or disprove (by presenting a counter example) that it always constructs an optimal schedule.

(a) Choose the job that ends last, discard all conflicting jobs, and recurse.

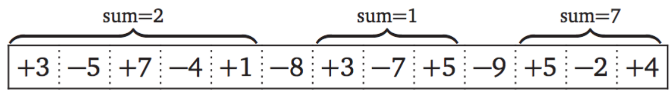
(b) Choose the job that starts first, discard all conflicting jobs, and recurse.

(c) Choose the job that starts last, discard all conflicting jobs, and recurse.

(d) Choose the job with shortest duration, discard all conflicting jobs, and recurse.

Problem 4. What is an optimal Huffman code for n characters whose frequencies are the first n Fibonacci numbers?

Problem 5. Suppose you are given an array $A[1 \ldots n]$ of integers, each of which may be positive, negative, or zero. A contiguous subarray $A[i \ldots j]$ is called a positive interval if the sum of its entries is greater than zero. Describe and analyze an algorithm to compute the minimum number of positive intervals that cover every positive entry in A. For example, given the following array as input, your algorithm should output 3. If every entry in the input array is negative, your algorithm should output 0.



Problem 6. A graph (V, E) is bipartite if the vertices V can be partitioned into two subsets L and R, such that every edge has one vertex in L and the other in R.

(a) Prove that every tree is a bipartite graph.

(b) Prove that a graph G is bipartite if and only if every cycle in G has an even number of edges.

(c) Describe and analyze an efficient algorithm that determines whether a given undirected graph is bipartite.

Problem 7. Describe and analyze an algorithm to compute an optimal ternary prefix-free code for a given array of frequencies $f[1 \dots n]$. Don't forget to prove that your algorithm is correct for all n.

Problem 8. Describe in detail how to implement the Gale-Shapley stable matching algorithm, so that the worst-case running time is $O(n^2)$, as claimed earlier in this chapter.