CS325: Analysis of Algorithms, Winter 2024

Practice Assignment 4

Due: Tue, 3/7/24

Homework Policy:

1. Students should work on practice assignments individually. Each student submits to CANVAS one set of *typeset* solutions in pdf format.
2. Practice assignments will be graded on effort alone and will not be returned. Solutions will be posted.
3. The goal of the assignments is for you to learn solving algorithmic problems. So, I recommend spending sufficient time thinking about problems individually before discussing them with your friends.
4. You are allowed to discuss the problems with others, and you are allowed to use other resources, but you *must* cite them. Also, you *must* write everything in your own words, copying verbatim is plagiarism.

Problem 1. An undirected graph $G = (V, E)$ is bipartite if the vertices V can be partitioned into two subsets L and R, such that every edge has exactly one endpoint in L and one endpoint 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.

Try to work on this problem as you read about graph search algorithms.

Problem 2. Describe and analyze an algorithm to compute an optimal ternary prefix-free code for a given array of frequencies $f[1 \ldots n]$. Don't forget to prove that your algorithm is correct for all n. This is a good exercise to ensure that you understand Huffman codes. What you get here should be very similar to the Huffman code; try to modify each step/proof of Huffman codes to work for ternary codes instead of binary codes.

Problem 3. For each of the following statements, respond *Ture*, *False*, or *Unknown*.

(a) If a problem is decidable then it is in P.

(b) For any decision problem there exists an algorithm with exponential running time.

(c) $P = NP$.

(d) All NP-complete problems can be solved in polynomial time.

(e) If there is a reduction from a problem A to CIRCUIT SAT then A is NP-hard.

(f) If problem A can be solved in polynomial time then A is in NP.

(g) If problem A is in NP then it is NP-complete.

(h) If problem A is in NP then there is no polynomial time algorithm for solving A.