Problem Set 6—Due Monday., June 4 4PM

15 Problem 1 Problem 16.3-7 in the text (P. 436). You can simply argue that your algorithm gives an optimal ternary prefix code rather than giving a formal proof.

20 Problem 2 For the input string: CABAACAABAACACAABAACBAA

(a) Give a huffman code for this string (describe both your tree and the actual binary encoding of the above string).

(b) Give the "high-level" Lempel-Ziv encoding of the above string where each match of length 3 or longer is represented by a length, pointer pair, and otherwise by the actual input character.

(c) Now give the actual bit string representation for the encoding of part b) that would be produced by encoding literals, lengths and pointers using two huffman trees. Again give the huffman trees used followed by the encoding.

(15) Problem 3. Describe how to modify the Ford-Fulkerson algorithm (using BFS to find augmenting paths) if we add the restriction that for each arc (i,j) there is a nonnegative lower bound l_{ij} such that the flow in arc (i,j) cannot be lower than l_{ij}. **You should assume you are given an initial feasible flow, where, the flow f_{ij} on each arc satisfies $l_{ij} \leq f_{ij} \leq c_{ij}$ in addition to the balance constraints**.

Hint: modify how the residual graph is built.

Extra credit (10) To help argue that your program works correctly, prove a version of the max-flow min-cut theorem for this new setting (with lower bounds). First define what the capacity of an S-T cut is in this new setting, then argue that your flow matches the capacity of a (minimum) S-T cut.

(30) Problem 4.

Use network flows to solve each of the following problems; Give the run time of your solution (try and make it as fast as possible)

a) You are given a directed strongly connected graph. Each arc has a cost associated with it. We want to find a minimum cost set of arcs such that removing that set of arcs makes the graph no longer strongly connected.

b) In a computer network there are n processors $P_1, P_2, .., P_n$, and m communication lines $C_1, C_2, ..., C_m$. Each processor i has the ability to test t_i lines per day and there is a list L_i which contains the communication lines that processor i is able to test. Subject to these constraints we would like to be able to test all the communication lines every day. A testing schedule determines for each processor the lines it should test. Find a testing schedule or determine that no schedule can test all lines in a single day.

c) Same as b) but find the minimum number of days to test all lines (and a schedule that achieves it). Careful. Be sure your approach actually finds the minimum number of days (some simple greedy solutions fail).

(30) Problem 5 The *yes/no clique*— problem is: given an undirected graph G=(V,E) and a target integer k, is there a clique of size k? A *clique* is a set of vertices C in V such that each pair of vertices (u, v) in C, is also an edge in E (thus every pair of vertices in a clique are connected by an edge).

a) Show that the yes/no clique problem is in NP (note, this problem is NP-C but you are NOT being asked to prove that).

b) Show that you can use a program which solves the yes/no clique problem to actually find a clique of size k (when one exists). You should find the clique using a polynomial number of calls to the yes/no clique routine, plus polynomial additional work. Thus you are showing that the problem of finding a clique of a given size is polynomialy reducible to yes/no clique.

c) Give a polynomial-time algorithm for *yes/no clique*— when $k < c$ for a constant c. Would your algorithm still run in polynomial time if we restrict k so $k < \log n$, with n the number of vertices in the graph?