Problem Set 4—Due Tuesday, Nov. 29

(20) Problem 1. We (and the book) proved a $\Theta(mn)$ bound on the number of saturating pushes and a $\Theta(n^3)$ bound on the number of non-saturating pushes for the FIFO preflow-push algorithm. We now want to conclude that we can find a max-flow in $\Theta(n^3)$ time.

Describe and analyze efficient data structures for this problem. You should be able to achieve $O(n^3)$ worst case time (e.g. your structures must have the total time for push, lift and selecting an active vertex meet this bound). In addition, for inputs where the number of non-saturating pushes is $O(mn)$ (which is the norm) your algorithm should run in $O(mn)$ time.

(20) Problem 2. Problem 27-3, page 627.

(20) 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 lower bound l_{ij} such that the flow in arc (i,j) cannot be lower than l_{ij} (you should assume an initial feasible flow, where, the flow x_{ij} on each arc satisfies $l_{ij} \leq x_{ij} \leq c_{ij}$ in addition to the balance constraints).

Extra credit (10): Describe how to find an initial feasible flow.

(20) Problem 4. Suppose that in addition to each arc having a capacity we also have a capacity on each node (thus if node i has capacity c_i then the maximum total flow which can enter or leave the node is c_i). Suppose you are given a flow network with capacities on both arcs and nodes. Describe how to find a maximum flow in such a network (hint: by modifying the network you can use a standard flow algorithm to solve this problem).

(20) Problem 5. Use network flows to solve each of the following problems:

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 which achieves it).