Problem Set 2—Due Friday , February 3, 3:45PM

(22) Problem 1. Let $d[i, v]$ be defined to be the length of the shortest simple path from v to t that uses at most i arcs ($d[i, v]$ is infinity if no such path exists). In our discussion of the Bellman-Ford shortest path algorithm (a similar approach is also discussed in section 6.8 of the book), we argued that if G had no negative cycles, then after iteration i of the main loop, $M[v] \leq d[i, v]$.

Even if G has negative cycles, we can still sensibly talk about the shortest (not necessarily simple) path from v to t using i or fewer arcs, and we let $d'[i, v]$ denote the shortest path from v to t using at most i arcs (which may have cycles in it).

For the following, assume that G has negative cycles, and we consider the Bellman-Ford algorithm we described.

a) Is it still true that after iteration i of the main loop, $M[v] \leq d[i, v]$ for all v? Give a proof or a counter-example.

b) After iteration i of the main loop, is $M[v] \leq d'[i, v]$ for all v? Give a proof or a counter-example.

(12) Problem 2. We noted that one possible approach to implementing the Ford-Fulkerson algorithm is to find for each G_f the path P that can send the most additional flow from s to t. Describe how to modify Dijkstra's algorithm to find such a path P (in the book's notation, we find the path P such that $bottleneck(P, f)$ is maximum). Discuss the running time of your solution to find P. Note that you are not being asked to analyze how many such paths the flow algorithm will need to find.

(22) 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 $l_{ij} \geq 0$ such that the flow in arc (i,j) cannot be lower than l_{ij} (you should assume an initial feasible flow, where, the flow $f(i, j)$ on each arc satisfies $l_{ij} \leq f(i, j) \leq c_{ij}$ in addition to the balance constraints).

Justify the correctness of your algorithm by proving a version of the max-flow min-cut theorem for this setting (in particular, what is the capacity of a cut here?)

Explain why in this setting it may be tricky to find a feasible flow.

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

(18) Problem 4 Consider the better method of finding augmenting paths discussed in class to improve the capacity scaling algorithm to $O(mn)$ per scaling phase, and similar to a method used in 7.4 as part of the preflow-push algorithm.

Prove the claim we made that the distances of the nodes to t never go down. In particular, prove that when we execute relabel(v), $d(v)$ always increases.

(26) Problem 5 Problem 7 page 417. In addition to the problem given, suppose that you can't meet the requirements for L. Find the smallest integer value L' such that all the clients can be connected to a base station.