CS 473: Algorithms, Fall 2009
HW 4 (due Tuesday, September 29 in class)

This homework contains three problems. **Read the instruction for submitting homework on the course web page**. In particular, *make sure* that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: For this home work, students can work in groups of up to 3 members each. Each group submits only one written solution (some groups will do an oral presentation. Indicate your group members on the homework (netids are needed)).

1. (20 pts) Let $G = (V, E)$ be a DAG. Each edge $e \in E$ has a length $\ell(e)$ and each node has a length $\ell(v)$. These lengths could be negative. Give a linear time (that is $O(n + m)$ time), algorithm for finding the shortest path distances from a given node s to all the nodes in V. The length of a path P is now the sum of the lengths of the edges and nodes on the path.

2. (40 pts) Suppose we want to run Dijkstra's algorithm on a graph with edge lengths that are integers in the range $0, 1, \ldots, L$.

   (a) Show how to implement Dijkstra's algorithm to run in time $O(nL + m)$ where n and m are the number of nodes and edges in G.

   (b) Show how to implement Dijkstra's algorithm to run in time $O((n+m) \log L)$ time. *Hint:* Use a priority queue that only stores L elements.

   You only need to prove why your implementation correctly follows the steps of Dijkstra's algorithm.

3. (40 pts) You are given a directed graph $G = (V, E)$ where each edge e has a length/cost c_e and you want to find shortest path distances from a given node s to all the nodes in V. Suppose there is exactly one edge $f = (u, v)$ that has a negative length and the rest have non-negative lengths. The Bellman-Ford algorithm for shortest paths with negative length edges takes $O(nm)$ time where $n = |V|$ and $m = |E|$. Show that you can take advantage of the fact that there is only one negative length edge to find shortest path distances from s in $O((m+n) \log n)$ time — effectively this is the running time for running Dijkstra's algorithm. Your algorithm should output the following for each node $v \in V$: *either* that the shortest path distance to v is undefined because of a negative length cycle C that can be reached from s and can reach v *or* the correct value of $dist(s, v)$. *Hint:* Consider shortest path distances in $G - f$. How can you check if G has a negative length cycle?