Illinois Institute of Technology
Department of Computer Science

# Homework Assignment 8

CS 430 Introduction to Algorithms
Fall Semester, 2014

**Due: Monday, November 24**

1. A *looped tree* is a weighted, directed graph built from an $n$-node binary tree by adding an edge from every leaf back to the root of the tree. All edges have non-negative weight

   (a) How long would it take Dijkstra's algorithm to compute the shortest path between two vertices in a looped tree?

   (b) Describe and analyze a faster algorithm.

2. As mentioned in the lecture of November 12, Dijkstra's algorithm can be used to determine shortest paths on graphs with some negative edge weights, as long as there are no negative cycles; however, the worst-case running time can be much worse than the $O(|E| + |V| \log |V|)$ given in for non-negative edge weights in lecture and CLRS because vertices that have come off the priority queue may have to be put back on.

   (a) Give a modified version of Dijkstras algorithm that works for negative edge weights, as long as there are no negative cycles.

   (b) Construct an infinite family of graphs (with negative edge weights) for which the asymptotic running time of this algorithm is $\Omega(2^{|V|})$.

3. The Floyd-Warshall all-pairs shortest path algorithm (section 25.2 of CLRS) computes, for each pair of vertices $u$, $v$, the shortest path from $u$ to $v$. However, if the graph has negative cycles, the algorithm fails. Describe a modified version of the algorithm (with the same asymptotic time complexity) that correctly returns shortest-path distances, even if the graph contains negative cycles. That is, if there is a path from $u$ to some negative cycle, and a path from that cycle to $v$, the algorithm should output $\infty$ as the length of the shortest path from $u$ to $v$. For other pairs of vertices the algorithm should correctly find the length of the shortest directed path.