CS 3343 Analysis of Algorithms – Fall 2011

11/22/11

11. Homework

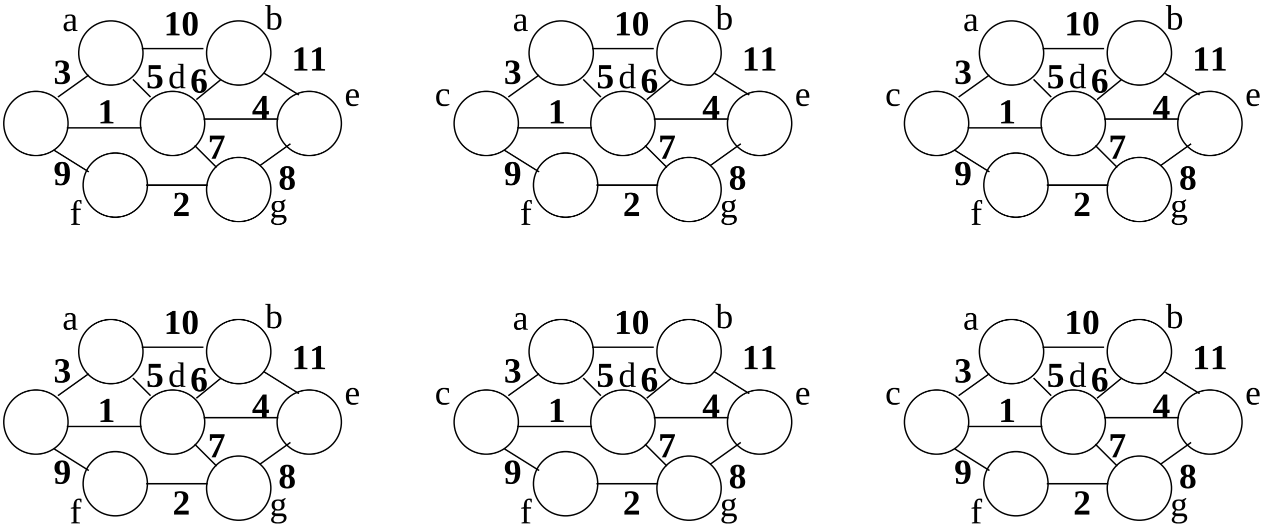
Due **12/6/11** before class

1. **Dijkstra and negative edge weights (4 points)**

   Give an example of a directed connected graph with negative edge weights, but without a negative weight cycle, for which Dijkstra's algorithm produces incorrect answers. Justify your answer.

2. **Dijkstra (7 points)**
   Run Dijkstra's algorithm on the graph below, with start/source vertex a. (Assume that each undirected graph edge $\{u, v\}$ is represented using two directed edges (u, v) and (v, u) with the same weight.)

   (a) Show all the different stages of the algorithm (vertex weights, tree edges stored in the predecessor array, and the priority queue).



   (b) List the shortest paths from a to all other vertices.

3. **Floyd-Warshall (7 points)**
   During the Floyd-Warshall all-pairs shortest paths algorithm, the shortest paths can be stored in a predecessor matrix. This is similar to storing a predecessor array for Dijkstra's algorithm, just that there is such an array for every vertex. (Page 695 / 632 in the textbook covers this topic, however it is possible to express the formula in a simpler way.)

   (a) (3 points) Modify Floyd-Warshall's algorithm to include the computation of the predecessor matrix.

   (b) (4 points) Write a method to use the predecessor matrix to print a shortest path between two vertices i and j.

FLIP OVER TO BACK PAGE $\Longrightarrow$

Practice Problems

(Not required for homework credit.)

1. **Dijkstra and negative edge weights**

   Give an example of a directed connected graph with a negative weight cycle for which Dijkstra's algorithm produces incorrect answers. Justify your answer.

2. **Bellman-Ford**
   Given a weighted, directed graph $G = (V, E)$ that possibly has negative weights but that has no negative weight cycles. For given two vertices u, v, let $l(u, v)$ be the minimum number of edges in a shortest path from u to v (where the shortest path is of course based on the edge weights). Now, define k to be the maximum, over all pairs of vertices $u, v \in V$, of $l(u, v)$. Suggest a simple change to the Bellman-Ford algorithm that allows it to terminate in $k + 1$ passes. **Do not assume that k is known in advance.**

3. **Floyd-Warshall**

   Run the Floyd-Warshall algorithm on the weigthed directed graph below. Show the matrix $c^{(k)}$ that results for each iteration of the outer loop (for k=0,...,6).