CSci 231 Homework 4

Graphs

Write each problem on a separate page. You will be graded on the clarity of the solution as well as correctness and completeness.

1. Problem 3.7 from textbook.

2. Problem 3.11 from textbook.

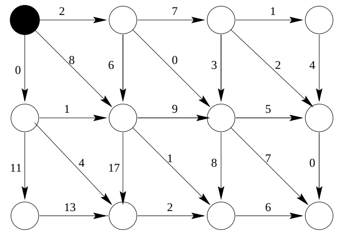
3. The *square* of a directed graph $G = (V, E)$ is a graph $G^2 = (V, E^2)$ such that $(u, w) \in E^2$ if and only if, for some $v \in V$, both $(u, v) \in E$ and $(v, w) \in E$. Thus, G^2 contains an edge between u and w whenever G contains a path with exactly two edges between u and w. Describe efficient algorithms for computing G^2 from G from both the adjacency-list and the adjacency-matrix representatiosn of G. Analyze the running time of your algorithms.

   Dijkstra'a SSSP algorithm works on general graphs with non-negative weights. The running time of Dijkstra's algorithm is $O(|E| + |V| \times$ INSERT $+ |V| \times$ DELETE-MIN $+ |E| \times$ CHANGE-KEY). Assuming the graph is connected and the priority queue is implemented as a heap the running time is $O(|E| \log |V|)$. The running time can be improved to $O(|E| + |V| \log |V|)$ using improved versions of priority queue (for instance the Fibonacci heap, which supports INSERT and CHANGE-KEY in $O(1)$ time amortized, and DELETE-MIN in $O(\lg n)$ amortized). While Dijkstra's algorithms gives the best known upper bounds for general SSSP with general non-negative weights and linear space, improved algorithms are known for special classes of graphs. In the following problems you will investigate several examples and derive improved bounds for computing SSSP.

4. *Shortest path for Directed Acyclic Graphs (DAGs):* Let $G = (V, E)$ be a DAG and let s be a vertex in G. Find a linear time $O(|V| + |E|)$ algorithm for computing SSSP(s). What vertices are reachable from s? Sketch a proof that your algorithm is correct. Does your algorithm need the constraint that the edge weights are non-negative?

5. Consider a directed weighted graph with non-negative weights and V vertices arranged on a rectangular grid. Each vertex has an edge to its southern, eastern and southeastern

neighbours (if existing). The northwest-most vertex is called the root. The figure below shows an example graph with V=12 vertices and the root drawn in black:



Assume that the graph is represented such that each vertex can access **all** its neighbours in constant time.

(a) How long would it take Dijkstra's algorithm to find the length of the shortest path from the root to all other vertices?

(b) Describe an algorithm that finds the length of the shortest paths from the root to all other vertices in $O(V)$ time.

(c) Describe an efficient algorithm for solving the all-pair-shortest-paths problem on the graph (it is enough to find the length of each shortest path).

6. We are given a directed graph $G = (V, E)$ on which each edge (u, v) has an associated value $r(u, v)$, which is a real number in the range $[0, 1]$ that represents the reliability of a communication channel from vertex u to vertex v. We interpret $r(u, v)$ as the probability tht the channel from u to v will not fail, and we assume that these probabilities are independent. Give an efficient algorithm to find the most reliable path between two given vertices.

7. Let (u, v) be a minimum-weight edge in a graph G. Show that (u, v) belongs to some minimum spanning tree of G.

8. Suppose that all edge weights in a graph are integers in the range from 1 to $|V|$. How fast can you make Kruskal's algorithm run? What if the edge weights are integers from 1 to W for some constant W?

9. Suppose that all edge weights in a graph are integers in the range from 1 to $|V|$. How fast can you make Prim's algorithm run? What if the edge weights are integers from 1 to W for some constant W?