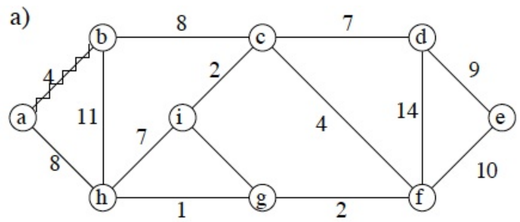
Algorithms Lab 10

In lab



1. Show how Prim's and Kruskal's algorithms run on an example graph.

   - What is the role of checking whether $v \in PQ$?
   - How many INSERT operations are performed by the algorithm?
   - How many DELETE-MIN operations are performed by the algorithm?
   - How many DECREASE-KEY operations are performed by the algorithm?
   - Assuming the priority is implemented as a heap, what is the complexity of the algorithm?

2. Show how Dijkstra's algorithm will run on the example graph from source vertex a. What will happen if you run Dijkstra's algorithm on a graph with negative edge weights?

Homework

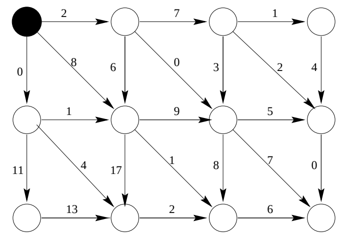
1. (CLRS 23.1-1) Show that a minimum-weight edge in G belongs to some MST of G.

2. (CLRS 24.3-6) 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.

3. (GT C-7.7) Suppose you are given a diagram of a telephone network, which is a graph G whose vertices represent switching centers, and whose edges represent communication links between the two centers. The edges are marked by their bandwidth. The

bandwidth of a path is the *minimum* bandwidth along the path. Give an algorithm that, given two switching centers a and b, will output a maximum bandwidth path between a and b.

4. 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).