5. Werkcollege 5

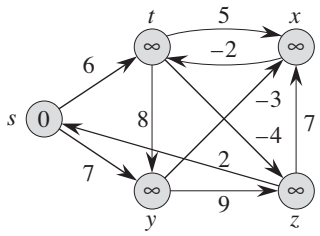
You are allowed to answer in Dutch. Whenever an algorithm is required, it can be given in pseudocode or plain English (or Dutch), and its running time and correctness must always be justified (even informally, but in a clear way!).

5.1. A *d-ary heap* is like a binary heap, but (with one possible exception) non-leaf nodes have d children instead of 2 children.

a) How would you represent a d-ary heap in an array?
b) What is the height of a d-ary heap of n elements in terms of n and d?
c) Give an efficient implementation of Extract-Max in a d-ary max-heap. Analyze its running time in terms of d and n.
d) Give an efficient implementation of Insert in a d-ary max-heap. Analyze its running time in terms of d and n.
e) Give an efficient algorithm of Increase-Key(A, i, k), which flags an error if $k < A[i]$, but otherwise sets $A[i] = k$ and then updates the d-ary max-heap structure appropriately. Analyze its running time in terms of d and n.

5.2. There is a network of roads $G = (V, E)$ connecting a set of cities V. Each road in E has an associated length. There is a proposal to add one new road to this network, and there is a list E' of pairs of cities between which the new road can be built. Each such potential road $e \in E'$ has an associated length l_e. As a designer for the public works department you are asked to determine the road in E' whose addition to the existing network G would result in the maximum decrease in the driving distance between two fixed cities s and t in the network. Give an efficient algorithm for solving this problem.

5.3. Run the Bellman-Ford algorithm on the following directed graph, using vertex z as the source. In each pass, relax edges in the the following order: (t, x), (t, y), (t, z), (x, t), (y, x), (y, z), (z, x), (z, s), (s, t), (s, y). Show the d and π values after each pass.



5.4. Shortest path algorithms can be applied in currency trading. Let $c_1, c_2, \ldots, c_n$ be various currencies; for instance c_1 might be dollars, c_2 pounds, and c_3 euros. For any two currencies c_i and c_j, there is an exchange rate $r_{i,j}$; this means that you can purchase $r_{i,j}$ units of currency c_j in exchange for one unit of c_i. These exchange rates satisfy the condition that $r_{i,j} \cdot r_{j,i} < 1$, so that if you start with a unit of currency c_i, change it into currency c_j and then convert it back to currency c_i, you end up with less than one unit of currency c_i (the difference is the cost of the transaction).

Give an efficient algorithm for the following problem: Given a set of exchange rates $r_{i,j}$, and two currencies s and t, find the most advantageous sequence of currency echanges for converting currency s into currency t. Toward this goal, you should represent the currencies and rates by a graph whose edges are real numbers.

5.5. Run the Ford-Fulkerson algorithm on the following flow network. For each iteration, pick the shortest augmenting path from s to t and give the residual network. Finally, give the value of the maximum flow and a minimum cut (S, T).