MTH6105 – Algorithmic Graph Theory **Spring 2024**

Problem Sheet 6 **F. Fischer**

You are expected to **attempt all exercises** before the seminar and to **actively participate** in the seminar itself.

---

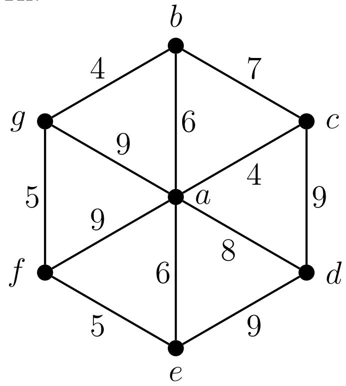
1. Consider a network (G, w), and let T be a minimum spanning tree of (G, w). Show that (G, w) has a unique minimum spanning tree if and only if the following condition is satisfied: for every edge $e \in E(G) \setminus E(T)$ with endpoints $u, v \in V(G)$ and every edge $d \in E(T)$ contained in the unique u–v-path in T, $w(e) > w(d)$.

   Solution: For the direction from left to right, assume that T is the unique minimum spanning tree of (G, w). Assume for contradiction that there exists an edge $e \in E(G) \setminus E(T)$ with endpoints $u, v \in V(G)$ and an edge $d \in V(T)$ contained in the unique u–v-path in T such that $w(e) \leq w(d)$. Let T' be the graph with $V(T') = V(T)$ and $E(T') = (E(T) \setminus \{d\}) \cup \{e\}$. Then T' is a spanning tree of G, and its weight is no greater than that of T, contradicting the assumption that T is the unique minimum spanning tree of (G, w).

   For the direction from right to left, assume that $w(e) > w(d)$ for every edge $e \in E(G) \setminus E(T)$ with endpoints $u, v \in V(G)$ and every edge $d \in V(T)$ contained in the unique u–v-path in T. Consider an arbitrary edge $e \in E(G) \setminus E(T)$, and let $F \subseteq E(G)$ contain edge e as well as the edges contained in the unique u–v-path in T. Then the edges in F form a cycle, and $w(e) > \max_{d \in F \setminus \{e\}} w(d)$. Thus, by Theorem 5.11 in the lecture notes, e is not contained in any minimum spanning tree of (G, w). Since this is true for every $e \in E(G) \setminus E(T)$, T is the unique minimum spanning tree of (G, w).

---

2. Consider the following network.



   (a) Use Prim's algorithm starting from vertex b to find a minimum spanning tree of the network.

   (b) Give another minimum spanning tree of the network.

Solution:

(a) Prim's algorithm may for example add edges in the order

$$bg, \ gf, \ fe, \ ba, \ ac, \ ad$$

to obtain the spanning tree T with $E(G) = \{ab, ac, ad, bg, ef, fg\}$.

(b) The edge ae has the same weight as the edge ab, which is contained in the unique a–e-path in T. By Exercise 1 there must thus exist another minimum spanning tree, and the spanning tree S with $E(S) = (E(T) \setminus \{ae\}) \cup \{ab\}$ is such a spanning tree.

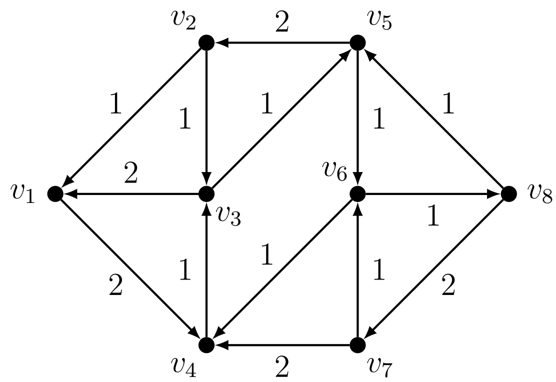
---

3. Consider a directed network (D, w) such that $w(e) \geq 0$ for all $e \in A(D)$. Let $v \in V(D)$.

(a) Give an algorithm with running time $O(|V(D)| \cdot |A(D)|)$ that finds a shortest directed v–u-path in (D, w) for every $u \in V(D)$ for which such a path exists. Provide a brief justification for the claimed running time.

(b) Give an algorithm with running time $O(|V(D)| \cdot |A(D)|)$ that finds a shortest directed u–v-path in (D, w) for every $u \in V(D)$ for which such a path exists.

(c) Find shortest directed v_8–u-paths and shortest directed u–v_1-paths in the following directed network for all $u \in \{v_1, v_2, \ldots, v_8\}$.



Solution:

(a) To find shortest paths from v to all vertices reachable from v along a directed path, we adapt Dijkstra's algorithm such that the set F of arcs considered for addition to the current tree T are those with tail in $V(T)$ and head in $V(D) \setminus V(T)$. The algorithm then constructs a tree on the set of vertices reachable from v in which all arcs are directed away from v. To obtain an upper bound on the running time of the algorithm we can argue in the same way as for Dijkstra's algorithm. The algorithm adds a vertex to T in every iteration where it doesn't stop, and thus stops after at most $|V(D)| - 1$ iterations. The arc uv added to the tree, which is selected to minimize $\delta(u) + w(uv)$ among all arcs in F, can be selected in time $O(|A(D)|)$ assuming that we have stored $\delta(x)$ for all $x \in V(T)$. The running time of the algorithm is thus $O(|V(D)| \cdot |A(D)|)$.

(b) To find shortest paths to v from all vertices from which v is reachable along a directed path, we adapt Dijkstra's algorithm such that the set F of arcs considered for addition to the current tree T are those with head in $V(T)$ and tail in $V(D) \setminus V(T)$. The algorithm then constructs a tree on the set of vertices from which v is reachable in which all arcs are directed towards v.

(c) The two algorithms described in Parts (a) and (b), respectively started from v_8 and v_1, may construct the following two spanning trees.