Combinatorial optimization 1
Winter term 2016/2017
2nd working sheet[1]

12.* Let $G = (V, E)$ be a graph and let $F = (F_1, F_2, \dots, F_k)$ be a k-tuple of pairwise edge-disjoint forests in G, i.e. $\forall\ i \in \{1, 2, \dots, k\}$, F_i is a forest in G and $E(F_i) \cap E(F_j) = \emptyset$, for all $i \neq j$, $i, j \in \{1, 2, \dots, k\}$. Assume moreover that $F = (F_1, F_2, \dots, F_k)$ is chosen so as to maximize $|E(F)|$, where $E(F) = \cup_{i=1}^{k} E(F_i)$, among all k-tuples of pairwise edge-disjoint forests in G. Let $e \in E(G) \setminus E(F)$. Show that there exists $X \subseteq V(G)$ with $e \subseteq X$ such that $F_i[X]$ is connected, for all $i \in \{1, 2, \dots, k\}$, and illustrate this statement by a concrete example.

13.* Let $G = (V, E)$ be a graph. A *multicut* in G is a set of edges $\delta(X_1, X_2, \dots, X_p) := \delta(X_1) \cup \delta(X_2) \cup \dots \cup \delta(X_p)$, where p is a natural number, $p \geq 2$, and $(X_1, X_2, \dots, X_p)$ is a partition of the vertex set $V(G)$, i.e. $V(G) = \cup_{i=1}^{p} X_i$, $X_i \cap X_j = \emptyset$ for all $i \neq j$, $i, j \in \{1, 2, \dots, p\}$, and $X_i \neq \emptyset$, for all $i \in \{1, 2, \dots, p\}$. Show that G contains k pairwise edge-disjoint spanning trees if and only if $|\delta(X_1, X_2, \dots, X_p)| \geq k(p-1)$ for every multicut $\delta(X_1, X_2, \dots, X_p)$ in G with $p \in \{2, 3, \dots, |V(G)|\}$, and illustrate this statement by a concrete example.

14. The shortest path arborescence
Let G be a digraph with edge weights $c\colon E(G) \to \mathbb{R}$ and $s \in V(G)$. A vertex v is called *reachable* from s in G iff there exists some s-v path P_v in G. An arborescence A with root s is called a *shortest path arborescence with root s in G* iff (i) $V(A)$ consists of all vertices v reachable from s in G, and (ii) for all $v \in V(A) \setminus \{s\}$ the unique s-v path in A is a shortest s-v-path in G.

    (a) Show that if the edge weight function c is conservative, then there exists a shortest path arborescence with root s for every $s \in V(G)$. Is the shortest path arborescence with root s uniquely determined?
    (b) Assume that every $v \in V(G)$ is *reachable from s* and that the edge weights are non-negative. Let A_P be a shortest path arborescence with root s in G and let A_S be a minimum spanning arborescence with root s in G. Show that $c(A_P) \leq (|V(G)| - 1)c(A_S)$ holds.

15. Solve the single source shortest paths problem with source at vertex s for the digraph in Figure 1. The numbers on the edges are the edge weights.

16. Consider the digraph in Figure 1. Solve the single source shortest paths problem with source at vertex 6 in the (non-directed) graph obtained from $G[\{3, 6, 7, 8\}]$ by ignoring the directions of the edges.

17. Modify Dijkstra's algorithm so as to obtain an efficient algorithm for *the bottleneck shortest path problem* defined as follows. In a given digraph G with non-negative edge weights $c\colon E(G) \to \mathbb{R}_+$ and two vertices $s, t \in V(G)$, $s \neq t$, determine a (directed) s-t-path P^* such that

$$\max_{e \in E(P^*)} c(e) = \min\{\max_{e \in E(P)} c(e) \colon P \text{ is an } s\text{-}t\text{-path in } G\}.$$

Such a path P^* is called *bottleneck shortest s-t-path* in G. Does your algorithm also work correctly if the weights $c(e)$ are allowed to take negative values? Argue upon your answer!

18. Use the Moore-Bellman-Ford algorithm to solve the single source shortest paths problem with source at vertex s in the digraph in Figure 2. What happens if the weight of edge $(2, 3)$ is changed to 4? Apply the algorithm in this case and comment on its output.

19. Consider a digraph G with conservative edge weights $C\colon E(G) \to \mathbb{R}$ and two vertices $s, t \in V(G)$, $s \neq t$. Assume that there is only one shortest s-t-path P in G. Can you determine *the second shortest s-t-path in G*, i.e. the shortest among all s-t-paths in G except for P, in polynomial time?

[1] Exercises marked with an $*$ are more demanding than others.

20. (Shortest) paths in acyclic graphs

Let $G = (V, E)$ be a digraph with $n := |V(G)|$. G is called *acyclic* or *cycle-free* if there are no (directed) cycles in G. A *topological sorting* of the vertices in G is a bijective mapping $f: V(G) \to \{1, 2, \dots, n\}$ such that $f(i) < f(j)$ for all $(i, j) \in E(G)$.

(a) Let G be an acyclic digraph. Formulate a *depth first search* (DFS) based algorithm to determine a topological sorting f in G in linear time. Show that the mapping f determined by DFS is not a topological sorting if G is not cycle-free. Hence your linear-time algorithm can also be used to detect the existence of cycles in G.

(b) Modify the Morre-Bellman-Ford algorithm for the single source shortest paths problem so that it runs in linear time in acyclic graphs.

(c) Consider the so-called *longest path problem* in G defined as follows. For a given digraph G and two vertices $s, t \in V(G)$, $s \neq t$, determine a longest s-t-path in G. Can you specify a polynomial-time algorithm for this problem? What happens if G is acyclic?

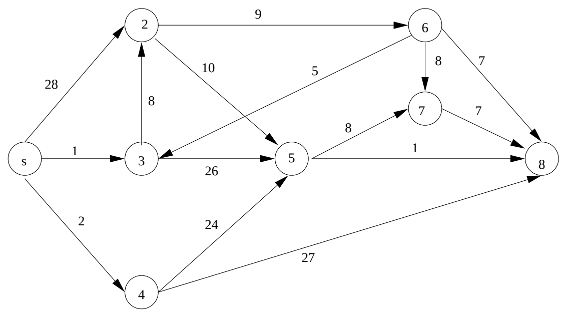

Figure 1: Digraph for Example 15, 16

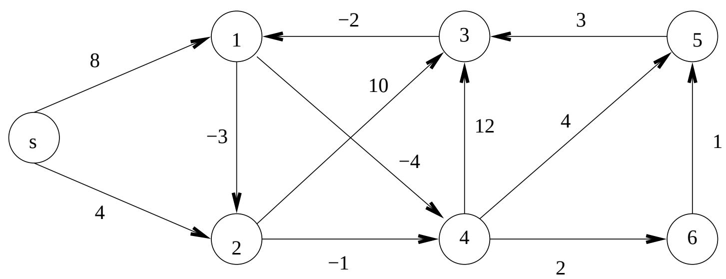

Figure 2: Digraph for Example 18