Prof. Friedrich Eisenbrand
Dr. Yuri Faenza

Discussions from: September 24, 2013

Combinatorial Optimization

Fall 2013

Assignment Sheet 1

Exercises marked with a $\star$ can be handed in for bonus points. Due date is October 8.

Exercise 1

Recall that in class we started by considering the following *connector problem*

GIVEN: a connected graph $G(V,E)$ with weights $w : E \to \mathbb{R}_+$.

OUTPUT: a connected subgraph of G, i.e. a graph $G'(V,\tilde{E})$ with $\tilde{E} \subseteq E$ such that there is a path in G' between any two vertices from V, of minimum cost.

and we showed that this can be solved by computing a *minimum spanning tree*:

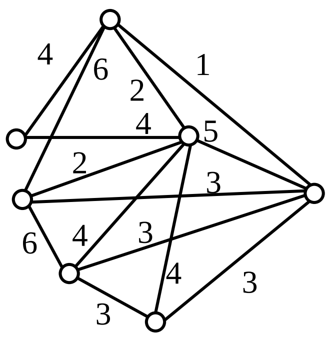
GIVEN: a connected graph $G(V,E)$ with weights $w : E \to \mathbb{R}$.

OUTPUT: a spanning tree of G, i.e. a tree $G'(V,\tilde{E})$ with $\tilde{E} \subseteq E$, of minimum cost.

Give an example showing that, if in the connector problem we do not assume the non-negativity of w, this is no longer true.

Exercise 2

Compute a minimum spanning tree for the instance below.



Exercise 3

Consider the instance (G,w) from the previous exercise. Modify the weights on the edges as to obtain a vector w' so that the minimum spanning tree T_{opt} in (G,w') is unique and is one of the minimum spanning trees of (G,w) (note: T_{opt} is not required to have the *same* cost in (G,w) and in (G,w')).

Exercise 4

Given an instance (G,w) with $G = (V,E)$ and $w : E \to \mathbb{R}$, show how to obtain a new weight vector $w' : E \to \mathbb{R}$ such that no two spanning trees of (G,w') have the same cost, and the minimum spanning tree T_{opt} in (G, w') is one of the minimum spanning trees of (G, w). (note: T_{opt} is not required to have the *same* cost in (G, w) and in (G, w')).

Exercise 5
In class we stated that a graph $G = (V, E)$ is a tree if and only if it is connected and $|E| = |V| - 1$, showing the "only if" part. Show the "if" part.

Exercise 6
Prove that, in a graph G with exactly two nodes of odd degree, there is always a path connecting those two nodes. (The *degree* of a node v is the number of nodes that are connected to v with an edge.)

Exercise 7
Prove that, in each simple graph with at least two nodes, there are at least two nodes with the same degree. (Here *simple* means that copies of the same edge are not allowed.) Is the statement true if the graph is allowed to be non-simple?

Exercise 8
A *connected component* of a graph $G = (V, E)$ is a set $U \subseteq V$ such that there exists a path between any two vertices of U in G, and no path between any vertex in U and any vertex in $V \setminus U$. For instance, a tree has exactly one connected component, while a graph with no edge has $|V|$ connected components. Consider the following algorithm ALGO.

`INPUT:` a connected graph $G = (V = \{v_1, \ldots, v_n\}, E)$, a weight function $w : E \to \mathbb{R}$.
`OUTPUT:` a spanning tree of G.

`SET` $\tilde{E} = \emptyset$.
`WHILE` $G' = (V, \tilde{E})$ is not a tree

Let U be the connected component of v_1 in G'.

Let e be an edge of minimum cost among those that have one endpoint in U and one endpoint in $V \setminus U$.

`SET` $\tilde{E} = \tilde{E} \cup \{e\}$.

(a) Apply ALGO to the instance from Exercise 2.

(b) [$\star$] Prove that ALGO correctly computes a minimum spanning tree.

(c) Show that, given in input the adjacency matrix of a graph, ALGO can be implemented as to run in $O(|V|^2)$ time.

(d) Show that, given in input also the adjacency lists of the nodes, ALGO can be implemented as to run in $O(|E| \log |V|)$ time.
(Hint: use binary heaps.)