Summary on Lecture 23, March 2, 2016

Optimal spanning trees: Kruskal's Algorithm in more detail

For a given finite connected graph $G = (V(g), E(G))$, we are looking for a spanning tree $T \subset G$ of minimal weight. Let $|E(G)| = m$. We assume that the edges $e_1, \dots, e_m$ have been initially sorted so that

$$\mathsf{wt}(e_1) \le \mathsf{wt}(e_2) \le \cdots \le \mathsf{wt}(e_m).$$

Recall Krushkal's algorithm:

Kruskal's Algorithm $(G = (V(G), E(G)),\ \mathsf{wt} : E(G) \to (0, \infty))$
`Input:` A finite weighted connected graph (G, wt) with edges listed in order of increasing weight
`Output:` A set E of edges of an optimal spanning tree for G)
`Set` $E = \emptyset$`, for` $j = 1$ `to` $|E(G)|$ `do`
  `if` $E \cup \{e_j\}$ `is acyclic then`
  `Put` e_j `in` E`.`
`return` E

Theorem 2. *Let G be a finite connected weighted graph. Then Kruskal's algorithm produces an optimal spanning tree.*

Proof. We consider the statement:

$$\mathbf{S} := \text{``The set of edges } E \text{ is contained in an optimal spanning tree of } G\text{''}$$

This statement is clearly true initially when the set E is empty. We assume the statment $\mathbf{S}$ is true at the start of the j-th pass through the loop, so that E is contained in some optimal spanning tree T, i.e., $E \subset E(T)$. There are two cases here:

(1) The graph $E \cup \{e_j\}$ is not acyclic.

(2) The graph $E \cup \{e_j\}$ is acyclic.

In the case (1), we do not change E, and the statement $\mathbf{S}$ holds. Then we move to the next iteration.

Consider the case (2). We would like to find an optimal tree T^* such that $E \cup \{e_j\} \subset T^*$. If e_j is in T, then we can take $T^* = T$. Now assume that e_j is not in T. Recall that since T is a spanning tree for G, $V(T) = V(G)$. Thus if e_j is not in T, the graph $T \cup \{e_j\}$ is not a tree anymore, and the edge e_j must be a part of some cycle C in $T \cup \{e_j\}$. By construction, the graph $E \cup \{e_j\}$ is acyclic, the cycle C must contain some edge f in T with f in $T \setminus (E \cup \{e_j\})$. Indeed this is true, otherwise all edges of the cycle C are in $E \cup \{e_j\}$, which is acyclic.

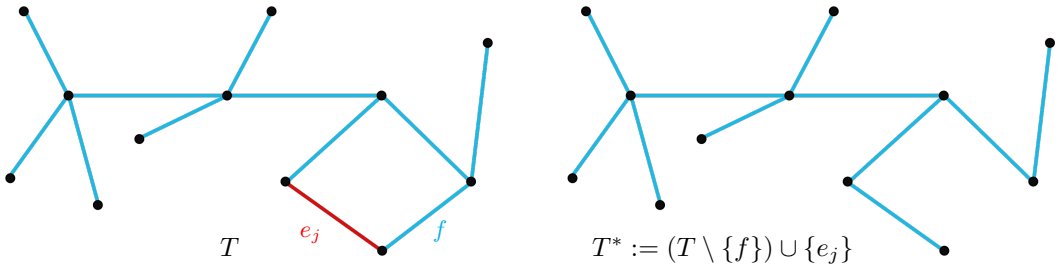

Fig. 4. Changing an optimal tree T to T^*

We remove the edge f from the tree T and construct the tree

$$T^* := (T \setminus \{f\}) \cup \{e_j\}.$$

We notice that T^* is connected, it spans G and it is a tree since $|E(T^*)| = |V(T^*)| + 1$ (we delete an edge and than add an adge to the tree T). Clearly, T^* is a spanning tree. Since the edge f has not yet been picked to be adjoined to E, it must be that e_j has first chance; i.e., $\mathsf{wt}(e_j) \leq \mathsf{wt}(f)$. Since

$$W(T^*) = W(T) + \mathsf{wt}(e_j) - \mathsf{wt}(f) \leq W(T),$$

and T is an optimal spanning tree, in fact we have $W(T^*) = W(T)$. Thus T^* is, indeed, an optimal spanning tree, as desired.

Since E is always contained in an optimal spanning tree, it only remains to show that the graph with edge set E and vertex set $V(G)$ is connected when the algorithm stops. Let u and v be two vertices of G. Since the original graph G is connected, there is a path from u to v in G. If some edge f on that path is not in E, then the graph $E \cup \{f\}$ contains a cycle. Indeed, otherwise f would have been chosen in its turn. Thus the edge f can be replaced in the path by the part of the cycle that's in E. Making necessary replacements in this way, we obtain a path from u to v lying entirely in E. □

Remark. We notice that Kruskal's algorithm works even if G has loops or parallel edges. It never chooses loops, and it will select the first edge listed in a collection of parallel edges. It is not even necessary for G to be connected in order to apply Kruskal's algorithm. In the general case the algorithm produces an optimal spanning forest made up of minimum spanning trees for the various components of G. ◊

Remark. In the process of attaching one more edge, Kruskal's algorithm has to check if the graph $E \cup \{e_j\}$ is acyclic or not. Here we can use the algorithm **Forest**(H) to produce a spanning forest of a graph $H = E \cup \{e_j\}$. If the reasulting forest contains the same number of edges as $E(H)$, then H is acyclic, and it does contain a cycle otherwise. ◊