Minimum Spanning Trees (Notes for MCS-236)

Max Hailperin

Fall 2011

Generic algorithm to find a minimum spanning tree

Given a connected weighted graph, G, of order n, the following algorithm will find the edges of a minimum spanning tree:

1. Initialize A_0 to $\{\}$.
2. Repeat for k from 0 to $n-2$:
   (a) Select a component, C, of the forest that has A_k as its edges but all of G's vertices.
   (b) Select a minimum-weight edge, e, from those edges of G that connect C to $G - C$.
   (c) Let $A_{k+1} = A_k \cup \{e\}$.
3. Return A_{n-1}.

The algorithm can't get stuck

Each of two steps in this algorithm, steps 2a and 2b, is a command to select an element from a set. Whenever we include such a command in an algorithm, we have a responsibility to show that the set in question can't be empty. Otherwise, the algorithm would be stuck with nothing to select.

In step 2a, we know that every graph has at least one component. The set of components can never be empty, and so selecting one is always possible.

In step 2b, we can take advantage of the fact that A_k contains k edges. As such, no component can have more than $k+1$ vertices. Since the loop control ensures that k is at most $n-2$, it follows that C can contain at most $n-1$ vertices, leaving at least one for $G - C$. Thus, in looking for an edge connecting C with $G - C$, we are looking for a connection between two nonempty subgraphs of G. As G is connected, it must have at least one such edge.

Prim's and Kruskal's algorithms

Prim's and Kruskal's algorithms for finding minimum spanning trees are specializations of the generic algorithm. Prim's algorithm is as follows:

1. Select a vertex, u, from G.
2. Run the generic algorithm, at each iteration of the loop choosing C to be the component that contains u.

Kruskal's algorithm consists of running the generic algorithm, each time through the loop choosing C to be a component that minimizes the weight of the corresponding minimum-weight edge e.

Correctness theorem for the generic algorithm

For all integers k such that $0 \leq k \leq n-1$, A_k is a subset of the edges of some minimum spanning tree of G.

Note that as a particular consequence of this theorem, A_{n-1} must be the edge set of a minimum spanning tree. From the theorem, it is a subset of the edges of some minimum spanning tree. But we know that A_{n-1} has size $n-1$, which is the same as the size of any spanning tree. So A_{n-1} must be the entire edge set of the minimum spanning tree.

Proof of the correctness theorem

We will prove the theorem by induction on k. For the base case, $A_0 = \{\}$, which surely is a subset of the edges of any minimum spanning tree of G. For the inductive step, we can take our induction hypothesis to be that some minimum spanning tree exists, call it T, such that $A_k \subseteq E(T)$. We need to show that A_{k+1}, which equals $A_k \cup \{e\}$, is a subset of $E(T')$, for some minimum spanning tree T' that may in general be different from T. We consider two cases:

Case 1, $e \in E(T)$: In this case, we can let $T' = T$.

Case 2, $e \notin E(T)$: Consider $T+e$. Because it has n edges, it must contain a cycle. In particular, because T is acyclic, $T+e$ must contain a cycle that includes e. Because e connects C with $G-C$, the cycle must include at least one other edge, call it e', that also connects C with $G-C$. Let $T' = T-e'+e$. Because e' connects C with $G-C$, we know that $e' \notin A_k$. Thus $A_k \subseteq E(T-e')$ and so $A_k \cup \{e\} \subseteq E(T-e'+e)$, that is, $A_{k+1} \subseteq E(T')$. All that remains is to show that T' is a minimum spanning tree.

To start with, T' is a spanning tree of G, because adding e reconnects the components of T severed by removing e'. The weight of T' can be calculated as $w(T') = w(T) - w(e') + w(e)$. But e was chosen as a minimum-weight edge connecting C to $G-C$, so $w(e) \leq w(e')$. Thus $w(T) - w(e') + w(e) \leq w(T) - w(e') + w(e')$; that is, $w(T') \leq w(T)$. Because T is of minimum weight and T' is no heavier, T' is also a minimum spanning tree.