Forests, trees, leaves . . .

A graph with no cycle is called acyclic.
An acyclic graph is also called a forest.

A connected, acyclic graph is called a tree.

Examples. Paths, stars

Theorem (Characterization of trees) For an n-vertex graph G, the following are equivalent

1. G is a tree
2. G is connected and has $n-1$ edges.
3. G has $n-1$ edges and no cycles.
4. $\forall\, u, v \in V(G)$, G has exactly one u, v-path.

A leaf (or pendant vertex) is a vertex of degree 1.

Properties of trees

Lemma. If T is a tree with $n(T) \geq 2$ then T contains at least two leaves.
Deleting a leaf from a tree produces a tree.

A spanning subgraph of G is a subgraph with vertex set $V(G)$.

A spanning tree is a spanning subgraph which is a tree.

Corollary.

(i) Every edge of a tree is a cut-edge.

(ii) Adding one edge to a tree forms exactly one cycle.

(iii) Every connected graph contains a spanning tree.

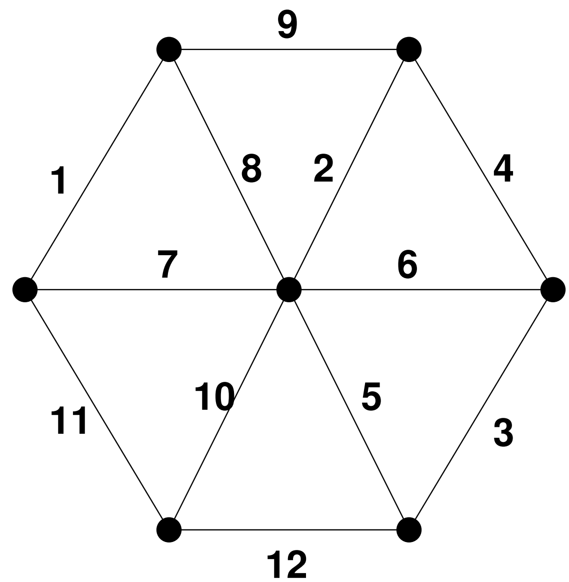
How to build the cheapest road network?

G is a weighted graph if there is a weight function $w : E(G) \to \mathbb{R}$.

Weight $w(H)$ of a subgraph $H \subseteq G$ is defined as

$$w(H) = \sum_{e \in E(H)} w(e).$$

Example:



Kruskal's Algorithm

Input: connected graph G, weight function $w : E(G) \to \mathbb{R}$, $w(e_1) \le w(e_2) \le ... \le w(e_m)$.

Idea: Maintain a spanning forest H of G. At each iteration try to enlarge H by an edge of smallest weight.

Initialization: $V(H) \leftarrow V(G)$, $E(H) \leftarrow \emptyset$, $i \leftarrow 1$

WHILE $i \le n$

    $e \leftarrow e_i$

    IF e goes between two components of H THEN

        update $H \leftarrow H + e$

        IF H is connected THEN

            stop and return H

    $i \leftarrow i + 1$

Theorem. In a connected weighted graph G, Kruskal's Algorithm constructs a minimum-weight spanning tree.

Proof of correctness of Kruskal's Algorithm

Proof. T is the graph produced by the Algorithm.
$E(T) = \{f_1, \ldots, f_{n-1}\}$ and $w(f_1) \leq \cdots \leq w(f_{n-1})$.

Easy: T is spanning (already at initialization!)
T is a connected (by termination rule) and has no cycle (by iteration rule) $\Rightarrow$ T is a tree.

But **WHY** is T min-weight?

Let T^* be an arbitrary min-weight spanning tree. Let j be the largest index such that $f_1, \ldots, f_j \in E(T^*)$.

If $j = n - 1$, then $T^* = T$. Done.

An exchange tool for the proof

Proposition. If T and T' are spanning trees of a connected graph G and $e \in E(T) \setminus E(T')$, then **there is** an edge $e' \in E(T') \setminus E(T)$, such that $T - e + e'$ is a spanning tree of G.

Proposition. If T and T' are spanning trees of a connected graph G and $e \in E(T) \setminus E(T')$, then **there is** an edge $e' \in E(T') \setminus E(T)$, such that $T' + e - e'$ is a spanning tree of G.

Proof of Kruskal, cont'd

If $j < n-1$, then $f_{j+1} \notin E(T^*)$.
There is an edge $e \in E(T^*)$, such that
$T^{**} = T^* - e + f_{j+1}$ is a spanning tree.

(i) $w(T^*) - w(e) + w(f_{j+1}) = w(T^{**}) \geq w(T^*)$
So $w(f_{j+1}) \geq w(e)$.

(ii) Key: When we selected f_{j+1} into T, e was also available. (The addition of e wouldn't have created a cycle, since $f_1, \ldots, f_j, e \in E(T^*)$.)
So $w(f_{j+1}) \leq w(e)$.

Combining: $w(e) = w(f_{j+1})$, i.e. $w(T^{**}) = w(T^*)$.
Thus T^{**} is min-weight spanning tree and it contains a *longer* initial segment of the edges of T, than T^* did. Contradiction.

Remark. Repeating this procedure at most $(n-1)$-times transforms any min-weight spanning tree into T.

Counting labeled trees

How many trees are there on vertex set $[n]$?

Example: $n = 1, 2, 3, 4, 5$... Conjecture?

Theorem The number of trees on $[n]$ is n^{n-2}.

Proof. (Prüfer code)

Bijection p from family of n-vertex trees to $[n]^{n-2}$.
Define $p(T) \in [n]^{n-2}$:

Let $T_0 = T$. Iteratively for $i = 1, \ldots n-2$ **do**

(1) $p(T)_i$:= (unique) neighbor of the smallest leaf ℓ_i of T_{i-1}

(2) $T_i := T_{i-1} - \ell_i$

This is a bijection!

Inverse: Given vector $(p_1, \ldots, p_{n-2}) \in [n]^{n-2}$, for $1 \leq i \leq n-1$, iteratively define:

$b_i := \min \left([n] \setminus \{b_1, \ldots, b_{i-1}, p_i, \ldots, p_{n-2}\} \right)$

Observation (1) $b_i \neq b_j$ for $i \neq j$

Define p_{n-1} by $[n] \setminus \{b_1, \ldots, b_{n-1}\} := \{p_{n-1}\}$

Observation (2) $b_i \neq p_j$ for $j \geq i$

Corollary $[n] = \{b_1, \ldots, b_{n-1}, p_{n-1}\}$

Define graph G_i: $V(G_i) := \{b_i, \ldots, b_{n-1}, p_{n-1}\}$
$E(G_i) := \{p_j b_j : j = i, \ldots, n-1\}$

G_i is well defined: $p_j \in V(G_{j+1}) \subseteq V(G_i)$, $\forall j \geq i$

G_i is a tree (contradiction to (2) if cycle $C \subseteq G_i$)

Claim The set of leaves of G_i:
$[n] \setminus \{b_1, \ldots, b_{i-1}, p_i, \ldots, p_{n-2}\} = \{b_i, \ldots, b_{n-1}, p_{n-1}\} \setminus \{p_i, \ldots, p_{n-2}\}$
In particular, b_i is the smallest leaf of G_i, $G_{i+1} = G_i - b_i$ and $p(G_i) = (p_i, \ldots, p_{n-2})$.

Proof: Reverse induction on $i = n-1, n-2, \ldots, 1$.

leaf-situation in G_i compared in G_{i+1}

- $b_i \in V(G_i) \setminus V(G_{i+1})$ is new leaf
- p_i is not a leaf in G_i (had a neighbor in G_{i+1}, received new neighbor in G_i)