Last week:

- Bridges (edges that affect connectedness).
- Trees (connected graphs with no cycles).
  - Examples (stars, double stars)
  - Leaves exist. Pruning leaves allows induction within trees.
  - $|E| = |V| - 1$

This week:

- Spanning trees.
- Counting trees.

Recall that a subgraph H of a graph G is spanning if $V(H) = V(G)$.

Definition. *A **spanning tree** of a connected graph G is a spanning subgraph $H \subseteq G$ that is also a tree.*

Example (Two spanning trees in a graph)**.**

Theorem (4.10)**.** *If G is a connected graph, then G contains a spanning tree.*

Exercise (HW 5.1 Alternative proof of the previous theorem)**.** *Let* G *be a connected graph. Consider the family* $\mathcal{F}$ *of all spanning subgraphs of* G *that contain no cycle. Show that* $\mathcal{F}$ *is non empty and that an element of* $\mathcal{F}$ *with* ***maximum*** *number of edges is a spanning tree of* G.

Exercise (A short proof for Theorem 4.7)**.** *Use Theorem 4.10 to prove that if* G *is a connected graph, then*

$$|E(G)| \geqslant |V(G)| - 1.$$

Note: A spanning tree constitutes a minimal set of paths in a connected graph (if we discard any edge in the tree, it disconnects.) Sometimes some edges are more *expensive* than others.

Definition. *Fix a connected graph* G*.*

- *We say that* G *is a* ***weighted graph*** *if it comes with a* ***weight function*** $w\colon E(G) \to \mathbb{R}^{>0}$.
- *The* ***weight*** *of a subgraph* H *of* G *is defined as* $w(H) = \sum_{e \in E(H)} w(e)$.
- *A* ***minimum spanning tree (msp)*** *of* G *is is spanning tree* T *such that* $w(T)$ *is minimum among all spanning trees.*

Example (A weighted graph)**.**

Algorithm: (Kruskal, 1956)

(1) Order $E(G)$ by increasing weight according to w.

(2) Start with T being the empty spanning subgraph of G.

(3) Add edges one-by-one according to the ordering if the edge creates no cycle.

(4) End if adding the next edge would create a cycle.

Algorithm: (Prim–Jarník, 1957–1930)

(1) Start with T being a single vertex in G.

(2) Find an edge e of smallest weight that has only one endpoint in T. Add e to T.

(3) End when every edge has two endpoints in T.

Exercise. *For the following graph, use the algorithms above to construct minimal spanning trees..*

Algorithm: (Kruskal, 1956)

(1) Order $E(G)$ by increasing weight according to w.
(2) Start with T being the empty spanning subgraph of G.
(3) Add edges one-by-one according to the ordering if the edge creates no cycle.
(4) End if adding the next edge would create a cycle.

Theorem (4.11)**.** *Kruskal's algorithm produces a minimum spanning tree in a connected weighted graph.*

Theorem (4.12)**.** *Prim's algorithm produces a minimum spanning tree in a connected weighted graph.*

Theorem. *If all edge weights in a connected graph* G *are distinct, then* G *has a unique minimum spanning tree.*