1 Minimum Spanning Trees (MST)

A graph $H(U, F)$ is a subgraph of $G(V, E)$ if $U \subseteq V$ and $F \subseteq E$. A subgraph $H(U, F)$ is called spanning if $U = V$.

Let $G$ be a graph with weights assigned to the edges. Then the weight of a subgraph $H$ of the $G$ is the sum of the weights of the edges of $H$.

A minimum spanning tree, MST, is a spanning tree of the minimum weight.

A cut $(V_1, V - V_1)$ of a graph $G(V, E)$ is a partitioning of its vertices into two subsets; $[V_1, V - V_1]$ denotes the set of edges $(u, v)$ of the form

$$ u \in V_1 \quad \& \quad v \in V - V_1. $$
Machinery for constructing MST’s

1. All algorithms use the Greedy Algorithmic Strategy: a tree is constructed as a sequence of steps; each step is **locally optimal**; no step is ever reversed.

2. Main step of every algorithm:
   an edge is colored either BLUE (≡ accepted) or RED (≡ rejected)
   every step is done so that the **color invariant** holds true:
   \[ \exists \text{MST containing all blue edges and none of the red ones.} \]

3. The coloring rules
   **blue rule:**
   - construct a cut of the graph without blue edges (if it exists);
   - select a **shortest** uncolored edge from the cut and color it blue.
   **red rule:**
   - construct a simple cycle without red edges (if it exists);
   - select the **longest** uncolored edge in the cycle and color it red.
**Greedy Method:** Initialize by coloring all edges white. At every moment, apply either rule until all edges are colored red or blue.

**Comment:** For the method to succeed, it is necessary that the color invariant holds after every coloring step.
Problem 1

Construct a minimum spanning tree in the network below by applying, first, blue rules (resp. red rules) and then applying red rules (resp. blue rules).
Problem 2

Let \((u, v)\) be a minimum-weight edge in a graph \(G\). Show that \((u, v)\) belongs to some minimum spanning tree.

Problem 3

Call an edge light, if there is a cut of \(G\) for which the edge is a minimal weight edge crossing the cut.

1. Show that every edge of a minimal spanning tree is light.

2. Construct an example of a weighted graph such that the set of all its light edges does not form an MST.
Problem 4

Let $T$ be an MST of a graph $G$ and let $L(T)$ be the sorted list of the edge weights of $T$. Show that for any two MST’s $S$ and $T$, $L(S) = L(T)$.

Problem 5

Let $T$ be an MST of a graph $G(V, E)$ and let $V' \subseteq V$. Let $T'$ be the subgraph of $T$ induced by $V'$ and let $G'$ be the subgraph of $G$ induced by $V'$. Show that if $T'$ is connected, then $T'$ is an MST of $G'$.
**Theorem 1**

At every step of the red-blue coloring, the color invariant holds true.

**Proof:** We want to prove that, at every step of the coloring there is a minimum spanning tree which uses all blue edges and does not use any of the red ones.

Initially, no edge is colored either blue or red, so the statement is correct and any minimum spanning tree would do. Let $T$ be a minimum spanning tree satisfying the color invariant and let $e$ be the next colored edge.

**Case 1: $e$ was colored blue.** If $e$ is in $T$ already, we are done. Otherwise, $T$ contains a path, $P$, connecting the end-points of $e$. If $(X, \overline{X})$ is the cut used to select $e$, then $P$ contains edges from the cut. Because of the color invariant and the condition of the blue rule, every edge which belongs to $P$ and to $[X, \overline{X}]$ is uncolored. Let $e'$ be such an edge. We build a new tree $T^*$ by removing $e'$ and adding $e$ to $P$. Obviously, the resulting spanning tree can only be shorter than $T$.

**Case 2: $e$ was colored red.** If $e$ is not in $T$, we are done. Let $e$ be an edge of $T$ and let $C$ be the cycle using which $e$ was selected. Obviously, at least one edge of the cycle, $e'$, does not belong to $T$. Since $T$ contains all blue edges, $e'$ is not blue; furthermore, because of the color invariant, it is not red. By the red rule, the weight of $e$ is not less than that of $e'$. This proves that $T - \{e\} \cup \{e'\}$ is a minimum spanning tree satisfying the required conditions.
2 Ideas for min-span algorithms

A blue tree is a tree in the forest defined by blue edges.

All three algorithms below are mainly blue-rule algorithms; every coloring step selects a blue tree, finds a minimum-cost edge incident to the tree and colors it blue.

Kruskal’s algorithm builds blue trees according to the edge costs; Prim’s algorithm builds only one non-trivial tree; Boruvka’s algorithm builds blue trees uniformly throughout the graph.

**Boruvka’s** algorithm (1926):
- initialize the forest of $n$ blue trees, each with one vertex and no edges.
- repeatedly consider the blue trees and for each one select the shortest incident edge;
- color all selected edges blue.

**Kruskal’s** algorithm (1956):
- reorder the edges in the nondecreasing order of their costs;
- for each next edge, if both ends of the edge belong to the same blue tree, color the edge red, else color it blue.

**Prim’s** algorithm (Jarnic (1930); Prim (1957); Dijkstra (1959))
- start with an arbitrary vertex $x$;
- repeatedly, if $T$ is a blue tree containing $x$, then select the shortest uncolored edge incident to $T$ (light blue);
- color the edge blue.
Prim’s Algorithm (example)
Theorem 2

Let $U \subseteq V$ and $e$ be of the minimum length among the edges with one endpoint in $U$ and the other in $V - U$. Then there exists a minimum spanning tree $T$ such that $e$ is in $T$.

**Proof:** Let $T_0$ be a minimum spanning tree. If $e$ is not in $T$, add $e$ to $T$. There is a path comprised of $T$-edges that connects the endpoints of $e$. Together with $e$ the path forms a cycle.

This cycle must have at least one more edge connecting $U$ and $V - U$. Let $e'$ be the edge; add $e$ to $T$ and remove $e'$.

The resulting graph $T'$ is connected and is a tree; the length of the new tree does not exceed that of $T$, thus $T'$ is minimal.
Problem 6

Let $G$ be a weighted undirected graph with all weights distinct; is it true that $G$ has a unique minimum spanning tree?
Application to the TSP on the plane

For this case of the general TSP, the triangle inequality holds true: for any three points $x$, $y$, and $z$,

$$w(x, y) + w(y, z) \geq w(x, z).$$

Approximate-TSP($G, w$)

1. select a vertex $r \in V[G]$ to be the root vertex;
2. construct a minimum spanning tree $T$ for $G$ from $r$ using MST-Prim($G, w, r$);
3. let $L$ be the list of vertices visited in a preorder tree walk of $T$;
4. use shortcuts to eliminate revisits; let $H$ be the resulting tour;
5. return the cycle $H$;