The minimum spanning tree Problem

Bakh Khoussainov

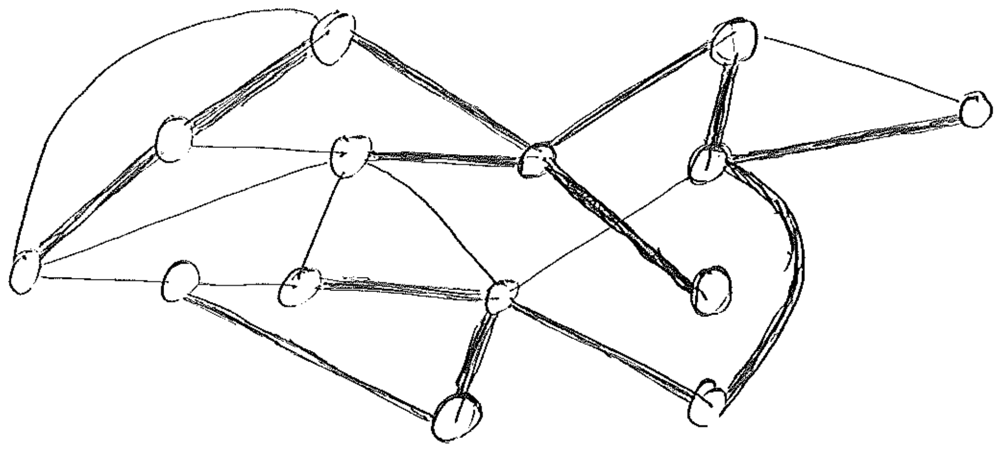
The minimum spanning tree problem.

Let $G=(V,E)$ be a graph.

A subset $T \subseteq E$ is a spanning tree of G if (V,T) is a tree.

Every connected graph has a spanning tree.

Example:



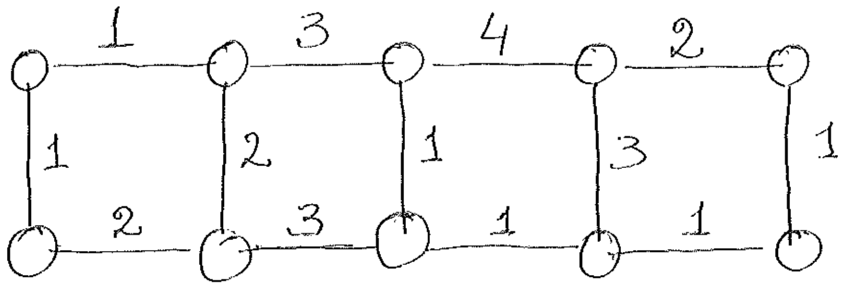
Suppose each edge e in G has a cost $c(e) > 0$.

For a spanning tree T of G its cost is

$$c(T) = \sum_{e \in T} c(e)$$

Goal: Find a spanning tree with the least cost.

Example:



T_1: Keep the top edges and vertical edges

$$c(T_1) = 10 + 8 = 18$$

T_2: keep the bottom edges and vertical edges

$$c(T_2) = 7 + 8 = 15$$

Prim's algorithm G, v:

Initially $S = \{v\}$, $T = \emptyset$.

While $S \neq V$

Among all edges $e = \{x, y\}$ such that $x \in S$ and $y \notin S$ find an edge $e' = \{a, b\}$ with the minimum cost.

Add b to S,

Add e' to T.

Kruskal's algorithm(G):

Initially $T = \emptyset$.

While (V, T) is disconnected

Find an edge $e \notin T$ with the smallest cost such that adding e to T does not produce a cycle

Add e to T.

To analyze these two algorithms we assume the following:

All edge costs are distinct from one another.

Cut property.

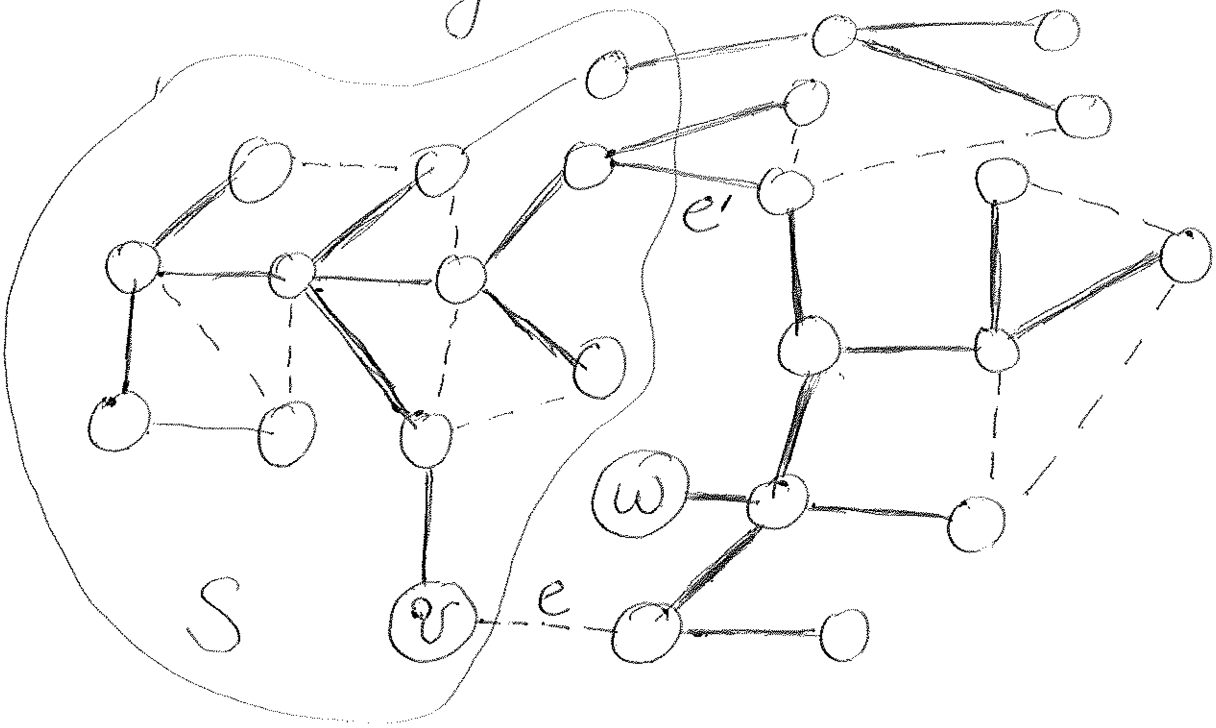
Let $S \subseteq V$ such that $S \neq \emptyset$, $S \neq V$. Let $e = \{v, w\}$ be the minimum cost edge with $v \in S$ and $w \notin S$. Then every minimum spanning tree contains e.

Let T be a minimum spanning tree that does not contain the edge e.

Want to find an edge e' in T such that $c(e') > c(e)$ and we can replace e' with e.

Since T is a tree, T has a path P from v to w. Let $e' = \{v', w'\}$ be the first edge in P such that $v' \in S$, $w' \notin S$.

Pictorially:



We replace e' with e and obtain T'. It is easy to see that T' is a spanning tree. Moreover,

$$c(T') < c(T).$$

Contradiction.

Fact. Prim's algorithm produces a minimum spannig tree for G.

At each step the algorithm has a partial spanning tree S. A new edge $e = \{v, w\}$ is added with minimum cost such that $v \in S$, $w \notin S$. So e is in every minimum spanning tree.

Hence the output of the algorithm is a minimum spanning tree.

Fact. Kruskal's algorithm produce a minimum spanning tree.

Let $e = \{v, w\}$ be an edge added at step i of the algorithm.

Set

$$S = \{ x \mid v \text{ has a path to } x \text{ before } e \text{ is added} \}$$

So $v \in S$ and $w \notin S$.

The edge e is the cheapest among edges between S and $V \setminus S$.

Hence, by the cut property, e belongs to every minimum spanning tree.

Each iteration of the algorithm guarantees that (V, T) has no cycles.

It is easy to see that the output of the algorithm is a spanning tree.

It must be a minimum spanning tree.