Minimum Spanning Trees

Goal : interconnect a set of objects in the cheapest possible way

example : computers – cable

fundamental problem, studied since at least 1920s

Definition :

Input : a graph $G = (V, E)$ undirected, connected, and weighted

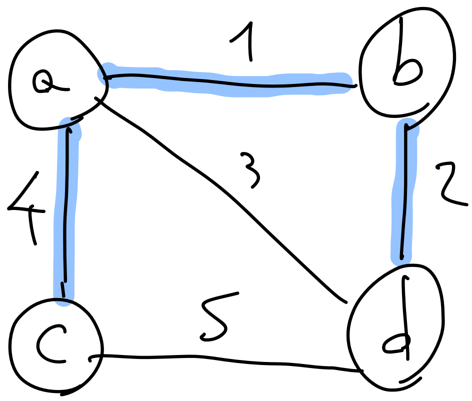
$$w : E \to \mathbb{R}$$

$$w(u,v) = \text{cost of edge } (u,v)$$

Output : a spanning tree $T \subseteq E$ of G s.t.

$$w(T) = \sum_{(u,v) \in T} w(u,v) \text{ is minimized}$$

example :



MST(G) ?

Observations:

1) minimum-~~weight~~ spanning tree

2) connected assumption is without loss of generality. (wlog)

if not, MST → MSF
↳ "forest"

Applications:

- networks (computers, sensors, electrical, ...)
  ↳ e.g. for broadcast
- machine learning (clustering)
- computer vision (object detection)
- data mining
- subroutine in other (approximation) algorithms

How difficult is it?

How many spanning trees can a graph have?

Complete graph: has all the $\binom{n}{2}$ possible edges

a complete graph has n^{n-2} different spanning trees!
↳ exponential

However, MST can be solved in near-linear time!
↳ $O((n+m)\log^{\alpha(1)} n)$

Not only: greedy algorithms ⟹ simple to implement in practice

→ Prim
→ Kruskal

They both apply (in ≠ ways) the same <u>generic</u> greedy <u>algorithm</u>:

Invariant maintained:

- at each iteration, A is a subset of edges of some MST

At each iteration the algorithm adds an edge that does not violate the invariant

↳ "safe" edge for A

GENERIC-MST(G)

$A = \emptyset$

while A does not form a spanning tree

find an edge (u,v) that is safe for A // crucial step

$A = A \cup \{(u,v)\}$

return A // A is an MST

How to find a safe edge? Luckily, MSTs enjoy the following structural property (see Theorem next)

First, some definitions:

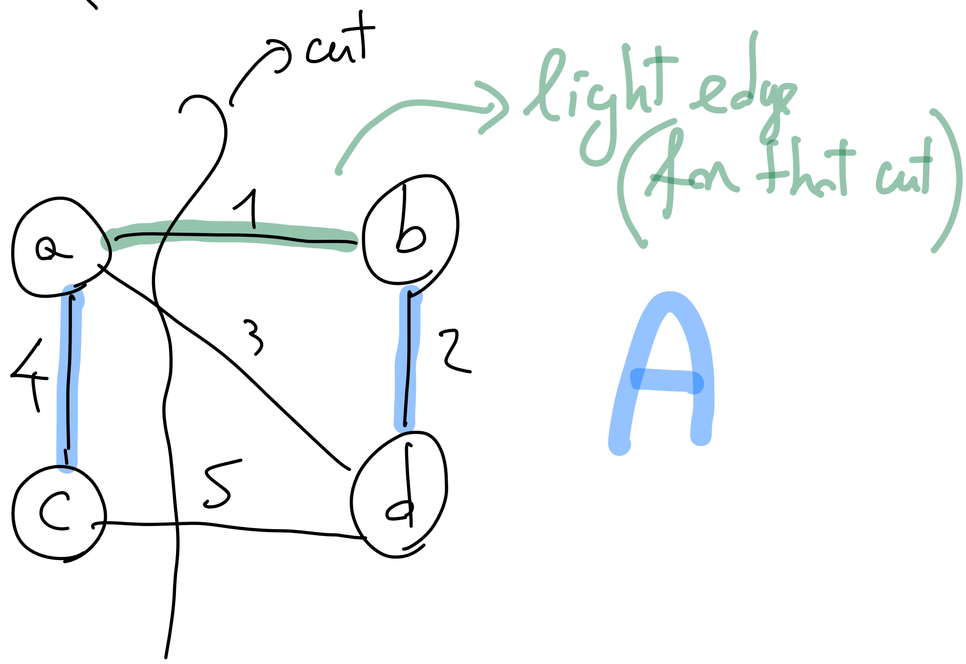
- A <u>cut</u> in a graph $G=(V,E)$ is a partition of V $\rightarrow$ $(S, V \setminus S)$

- An edge $(u,v) \in E$ <u>crosses</u> a cut $(S, V\setminus S)$ if $u \in S$ and $v \in V \setminus S$ (or vice versa)

- A cut <u>respects</u> a set of edges A if no edge of A crosses the cut

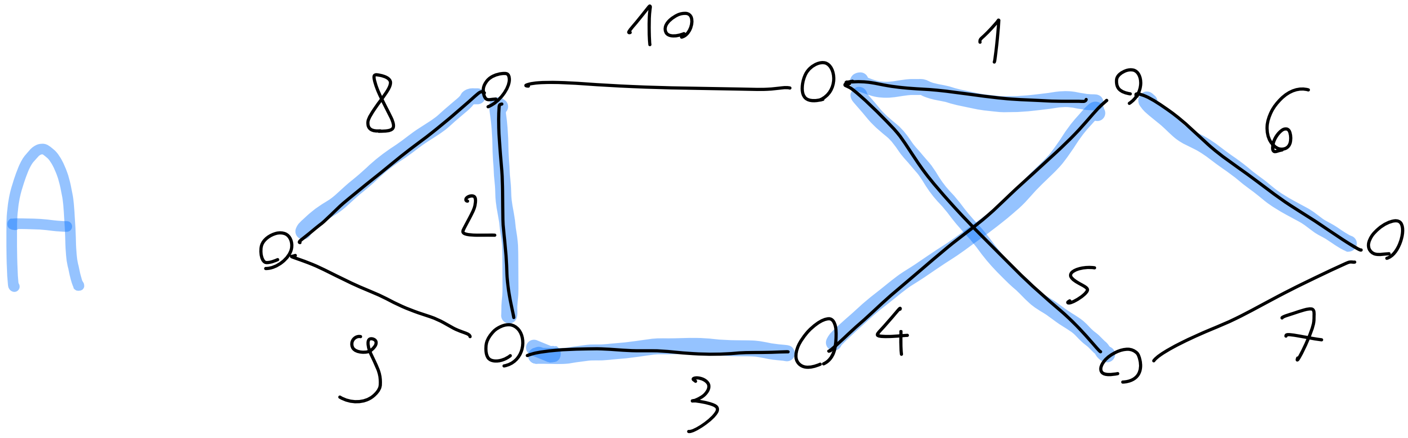
- Given a cut, an edge that crosses the cut and is of minimum weight is called <u>light edge</u> (for that cut)

example:



Theorem: Let $G = (V, E)$ be an undirect, connected, and weighted graph. Let A be a subset of E included in some MST of G, let $(S, V \setminus S)$ be a cut that respects A, and let (u, v) be a light edge for $(S, V \setminus S)$. Then (u, v) is safe for A.

Example of GENERIC-MST:



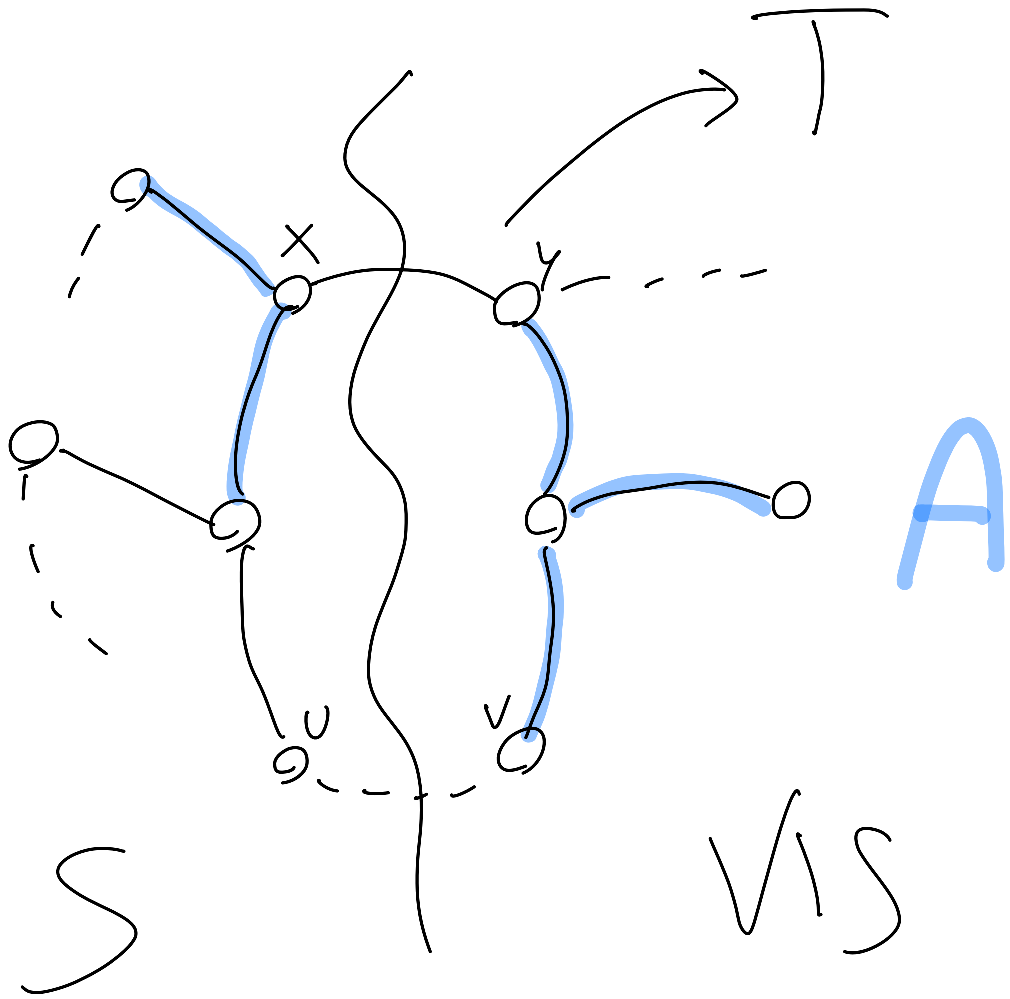
Proof of Theorem:

technique: "cut & paste", standard for greedy algorithms

Let T be an MST that includes A.

Assume $(u, v) \notin T$ (otherwise, we'd be done)

We'll build a new MST T' that includes $A \cup \{(u,v)\}$.



By hypothesis (u,v) crosses $(S, V \setminus S)$

$\Rightarrow \exists$ another edge of T that crosses that cut $\rightarrow (x,y)$

By hypothesis $(S, V \setminus S)$ respects $A \Rightarrow$ $(x,y) \notin A \Rightarrow$ removing (x,y) from T and adding (u,v) we obtain a new spanning tree $T' = T \setminus \{(x,y)\} \cup \{(u,v)\}$ that includes $A \cup \{(u,v)\}$.

Now we need to show that T' is an MST.

(x,y) and (u,v) both cross $(S, V \setminus S)$, and (u,v) by hypothesis is light $\Rightarrow$

$$w(u,v) \leq w(x,y)$$

$$\Rightarrow w(T') = w(T) \underbrace{- w(x,y) + w(u,v)}$$

$$\leq w(T)$$

but T is an MST $\Rightarrow w(T') = w(T)$

We'll now see two MST algorithms that organize the choices of those "respectful" cuts.

Prim's algorithm (1957)

How does Prim's alg. apply GENERIC-MST (G):

1) A is a single tree
2) safe edge: a light edge that connects the tree with a vertex that does not belong to the tree

Prim (G, s) // s = source vertex

$X = \{s\}$

$A = \emptyset$

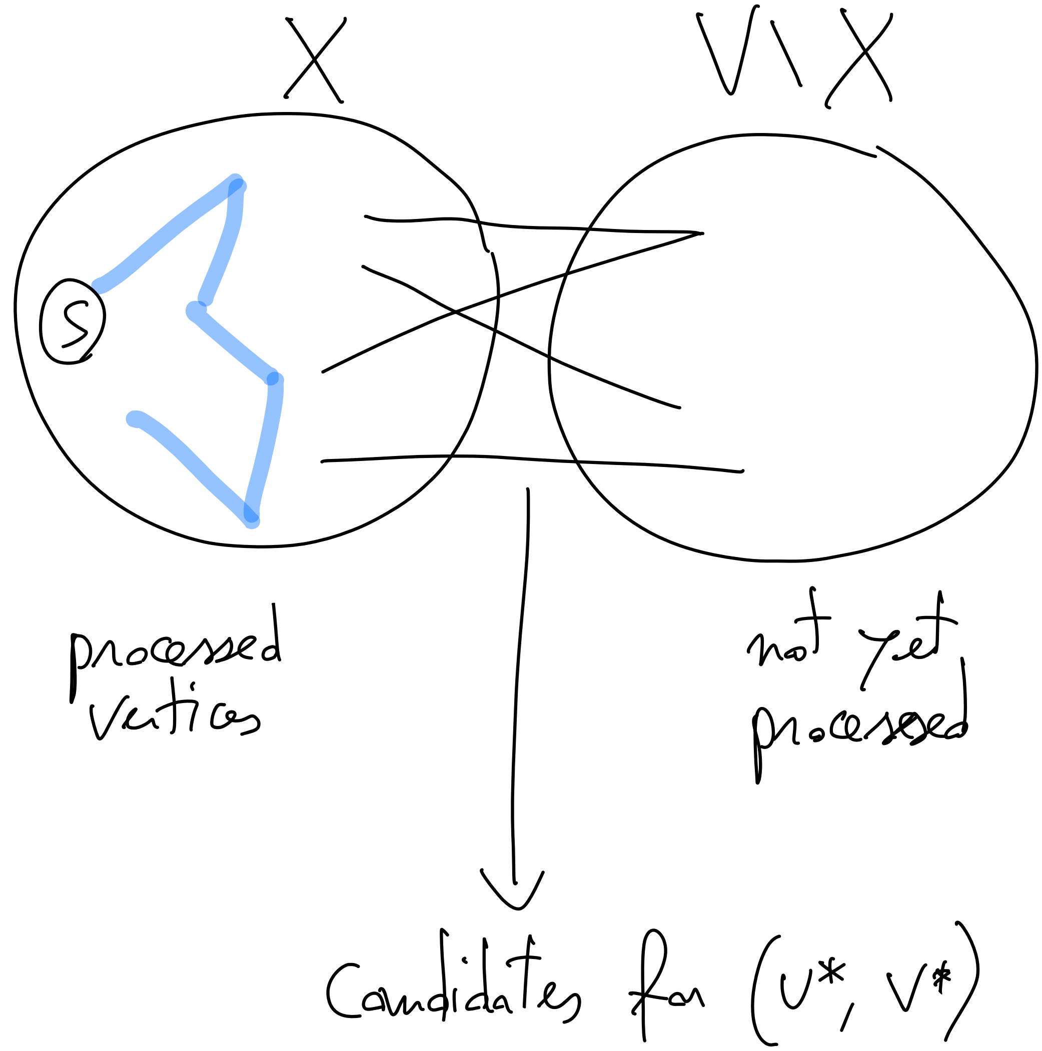
while there is an edge (u, v) with $u \in X$ and $v \notin X$ do

(u^*, v^*) = a minimum-weight such edge // light edge

add vertex v^* to X

add edge (u^*, v^*) to A

return A



this algorithm "grows" a spanning tree from a source vertex s (doesn't matter who s is) by adding one edge at a time