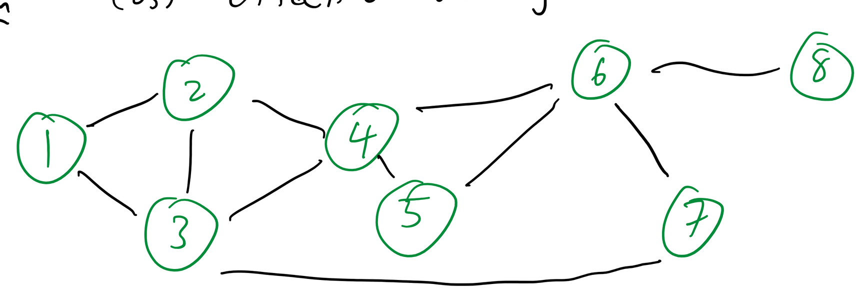
Lec01-MST

Sunday, August 23, 2026 4:36 PM

Minimum spanning trees (MST)

Problem: cost-effective wiring of a network



locations

possible connections

Other applications: (1) DNA sequences + similarity
└ recapitulate evolutionary tree

(2) Clustering by removing long edges

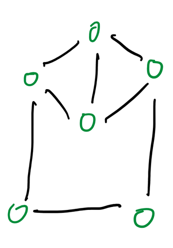
Graphs An undirected graph $G = (V, E)$ is a pair of sets

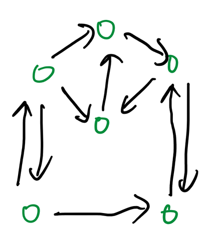
V = set of vertices / leaves $|V| = n$

E = set of 2-element subsets of V $|E| = m$

$e \in E \Rightarrow e = \{u, v\}$, with $u, v \in V$.

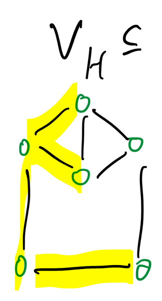
A graph is directed if E is a set of ordered pairs (u,v), $u, v \in V$.

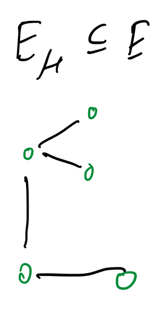




A graph $H = (V_H, E_H)$ is a subgraph of $G = (V_G, E_G)$ if

$V_H \subseteq V$ and $E_H \subseteq E$





A connected component is a maximal connected subgraph

Graphs naturally model many concepts
└ anything that has objects + relationships b/t pairs of objects

1. social networks
2. geographic adjacency
3. polyhedra
4. chemical molecules
5. assigning jobs to applicants
6. food webs
7. finite-state machines
8. Markov processes
9. project dependencies
10. WWW
11. telephone networks
12. roads + cities
13. neural networks
14. phylogenetic relationships
15. mesh approximation to surfaces

Def. A cycle of a graph $G = (V, E)$ is a sequence of distinct vertices $v_1, \dots, v_k \in V$ s.t. $\{v_i, v_{i+1}\} \in E \ \forall i = 1, \dots, k-1$ and $\{v_k, v_1\} \in E$.

Def. A tree is a graph that is connected + has no cycles

Vote: If I add an edge between two nodes in a tree that don't already have an edge between them, will I get a cycle?

Always Sometimes Never No clue

Minimum Spanning Tree problem (MST)

Given an undirected, connected graph G, and nonnegative weights $d(u,v)$ = cost of using edge $\{u,v\}$,

Find the subgraph T that connects all vertices + minimizes

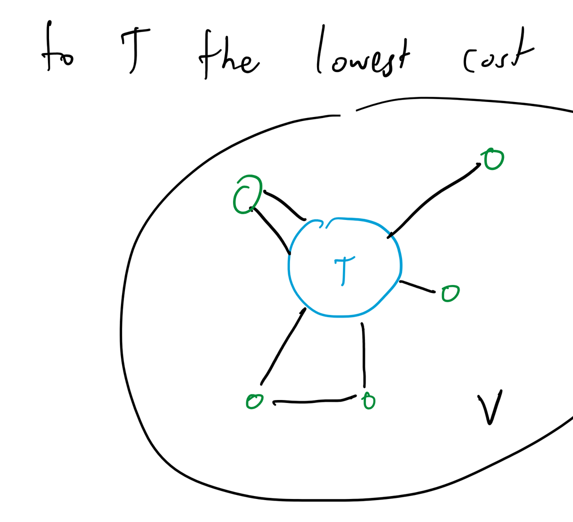
$$\text{cost}(T) = \sum_{\{u,v\} \in T} d(u,v).$$

Why will T be a tree? If not, ...

Prim's algorithm

(also will say $s \in T$)

- Given graph $G = (V, E)$, select arbitrary start node s. Set $V_T = \{s\}$, $E_T = \emptyset$
- Repeat $|V| - 1$ times:
  Add to T the lowest cost edge $\{u, v\}$ where $u \in T$, $v \notin T$ (and add v to T)



Questions:

Correctness

Data structure / implementation ← how to quickly find lowest cost edge

Worst-case runtime ← of our algorithm

Complexity ← is it possible to be faster

Proof of correctness:

Theorem (characterization of trees)

The following are equivalent

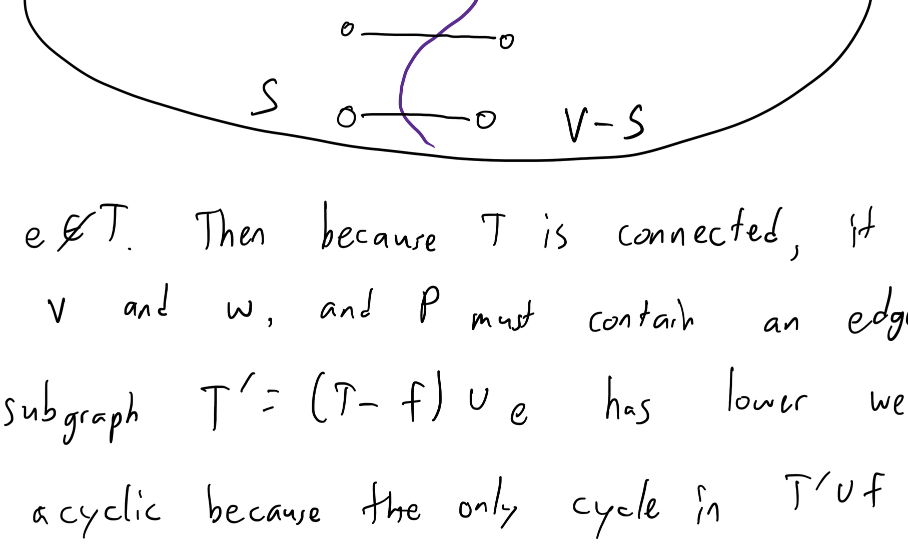
1. T is a tree
2. T contains no cycles and $n-1$ edges.
3. T is connected and has $n-1$ edges.
4. T is connected and removing any edge disconnects it.
5. Any two nodes in T are connected by exactly one path.
6. T is acyclic, and adding any new edge creates exactly one cycle.

Assumption No two edges in G have the same cost.
(else, can add small random ε_e to weight of every edge e.)

Thm (MST cut property)

Let S be a subset of nodes with $|S| \geq 1$ and $|S| < |V|$.

Every MST contains the edge $e = \{v, w\}$ with $v \in S$ and $w \in V - S$ of minimum weight.



A pair $(S, V-S)$ is a cut of the graph

proof. Suppose $e \notin T$. Then because T is connected, it must contain a path P between v and w, and P must contain an edge f that crosses the cut

Then, subgraph $T' = (T - f) \cup e$ has lower weight than T.

T' is acyclic because the only cycle in $T' \cup f$ is eliminated by removing f.

Thm Prim's correctness

At termination, Prim's always returns a MST.

proof. At any point, $T = (V_T, E_T)$ is a subgraph that is a tree.

T grows by 1 vertex + 1 edge at each step, so it will stop after $|V_G| - 1$ steps, + will be a spanning tree. ← a tree that connects all vertices in G

The pair $(V_T, V_G - V_T)$ is a cut of G.

By the cut property, the MST contains the lowest cost edge crossing this cut, but that is exactly the next edge Prim's always adds

So, Prim's always adds edges in the MST. ☐