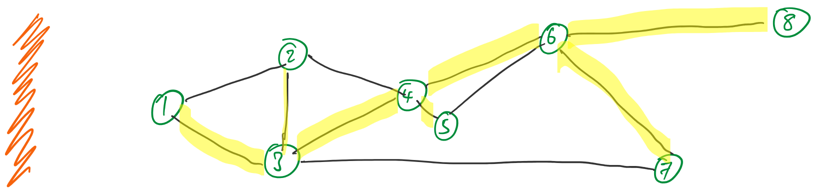
Lec01-MST

Monday, August 28, 2023 12:16 AM

Minimum Spanning Trees (MST) KT 3.1, 4.5, 4.7

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

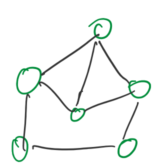
Graphs An undirected graph $G=(V,E)$ is a pair of sets

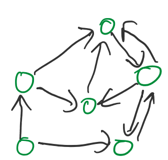
V is a set of nodes/vertices

E is a set of 2-element subsets of V

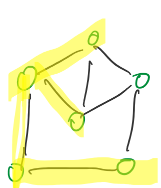
$e \in E \Rightarrow e = \{u,v\}$ with $u,v \in V$.

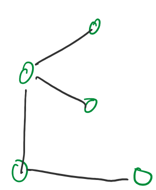
A graph is directed if E is a set of ordered pairs (u,v), $u,v \in V$.





A graph $H=(V_H, E_H)$ is a subgraph of $G=(V_G, E_G)$ iff
$V_H \subseteq V_G$ and $E_H \subseteq E_G$.





A connected component is a maximal connected subgraph

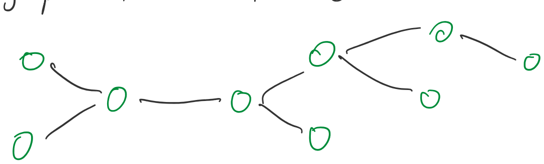
Graphs naturally model many concepts
 ∟ anything that has objects + relationships b/t pairs of objects

1. social networks
2. geographic adjacency

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 network
12. Roads
13. neural networks
14. phylogenetic relationships
15. mesh approximations 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 iff it is connected and has no cycles.



MST problem

Given an undirected connected graph G with vertices for each of n objects and non-negative weights $d(u,v)$ = cost of using edge $\{u,v\}$,

Find the subgraph T that connects all vertices and minimizes

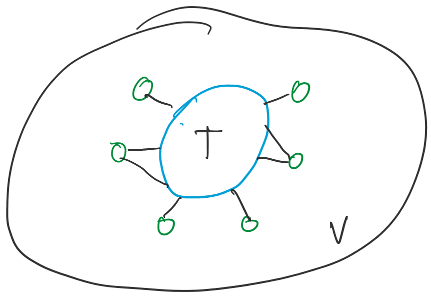
$$\text{cost}(T) = \sum_{\{u,v\} \in T} d(u,v)$$

Why will T be a tree? If $\exists$ cycles can remove any edge on cycle to lower cost

Prim's algorithm

- Given graph $G=(V,E)$, select arbitrary node s as the start of T.

- Repeat $|V|-1$ times:
  Add to T the lowest cost edge $\{u,v\}$ where $u \in T$ and $v \notin T$.



Example run of Prim's on graph above

Pseudocode:

array of distances from T

```
for u ∈ V do    Tdist[u] ← ∞ ,    parent[u] ← NULL
```

encodes edges

```
u ← s
while u ≠ null do
    Tdist[u] ← -∞          (since u in tree)
    for v ∈ Neighbors(u) do
        if d(u,v) < Tdist[v] then
            Tdist[v] ← d(u,v)
            parent[v] ← u
    u ← ClosestVertex(Tdist)

return parent
```

Questions:

Correctness - will we always find the minimum

Choice of data structure / Implementation - how to quickly find next lowest cost edge after updating T

Worst-case running time

Is it possible to be faster / complexity

Proof of correctness:

Theorem (characterization of trees)

The following statements are equivalent:

1. T is a tree.
2. T contains no cycles and $n-1$ edges

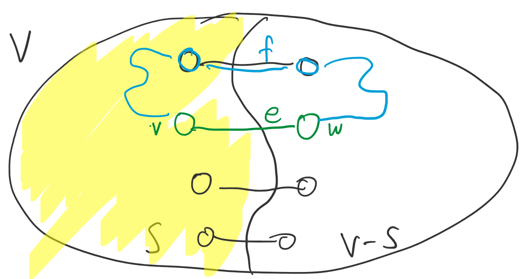
1. T is a tree.
2. T contains no cycles and $n-1$ edges
3. T is connected and has $n-1$ edges
4. T is connected and removing any edge disconnects it.
5. Any two nodes in T are connected by exactly 1 path
6. T is acyclic, and adding any new edge creates exactly one cycle.

Assumption No two edges in G have the same cost

Else, can just add a random small ε_e to weight of every edge e.

Theorem (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

The 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.

Theorem (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 and 1 edge each step, so it will stop after $|V_G| - 1$ steps, and will then be a spanning tree.

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 only adds edges that are in the MST.

$\square$