Today:

Dijkstra

Trees

Minimal spanning trees (Kruskal's algorithm)

Problem (minimal path / Dijkstra)

Given a connected graph G, vertices v & w and "length" function

$$l : E \longrightarrow \mathbb{R}_{\geq 0}$$

want to find a path from v to w of "minimal length"

i.e. so that the sum of lengths of edges in path is as small as possible.

Procedure:

Inductively construct a set Q

at each stage the vertices in Q are closer to v than all other vertices.

Keep adding to Q until get to w.

Also construct a function

$$b : V \longrightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$$

At each stage, $b(u) = \begin{cases} \infty & \text{if } u \text{ not adjacent to a vertex in } Q \\ \text{length of a minimal path to } u \text{ from } v \text{ of the form } v\, v_1, v_2 \cdots v_m\, u \text{ where } v_j \in Q & \end{cases}$

also $p(u)$ = "previous vertex in optimal path"

Procedure at each pt:

Know (previous) conversation: if u is closest ($\leq$ distance) to v of any vertex not in Q then

$$d(v,u) = d(v,x) + l(x\,u) \quad \text{some } x \in Q$$

so next u to be added satisfies

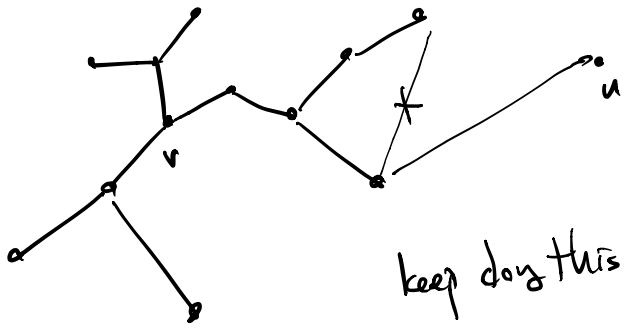
$$d(v,u) = \min_{x \text{ adj to } u \text{ in } Q} \{d(v,x) + l(x\,u)\}$$

but by assumption, $d(v,u) = b(u)$

So can choose u via making $b(u)$ minimal,

then update b's by $b(x) = \min\{\text{old } b(x),\ b(u) + l(u,x)\}$ if u is adj to x.

and $p(x) = u$



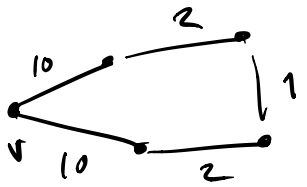
keep doing this - get "tree"

connected acyclic graph

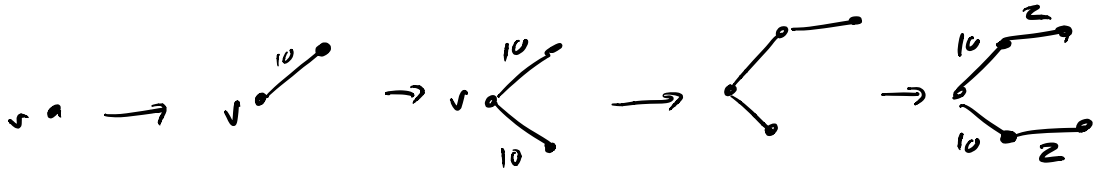
! unique path from v to every other vertex.

in this process - we make a subgraph connecting each vtex to another (unique)

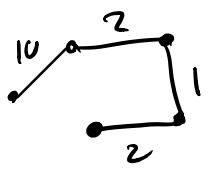
But this is generally not the most efficient way to do it.



→ Dijkstra



minimal to just connect:



Trees (and other things)

Def If G is a pseudograph, we define

$\delta(G)$ = minimum degree of a vertex

$\Delta(G)$ = max degree of a vertex

Def A vtex w/ degree 1 is called a "leaf"

Def if u,v vertices, $d(u,v) = \min_{v}$ length of a walk from u to v

Def $\text{diam}(G) = \max_{u,v} d(u,v)$

Note? A walk from u to v of minimal length must be a path.

Def $k(G)$ (aka $\omega(G)$) is the number of connected components of G.

Def An edge e is called a **bridge** if $k(G-e) > k(G)$

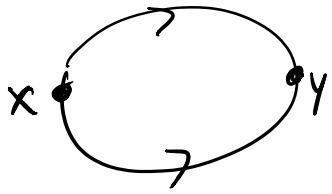
Def A tree is a connected graph with no cycles

Def A forest is a graph with no cycles (aka "acyclic")

Thm A graph G is a tree if and only if there is a unique path between any two vertices

"Pf:" if tree, connected, so $\exists$ paths between any 2 vrtxes.
if not unique $\Rightarrow$ cycle $\Rightarrow$ contradiction.

Conversely, if $\exists!$ path between any 2 vrtxes
$\Rightarrow$ connected, acyclic, since if a cycle, would be two paths from same pair of vertices



□

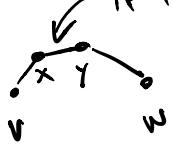
Thm trees with at least 2 vertices have at least two leaves.

Pf: ends of path of max'l length.

Prop Every connected graph has a spanning tree

Pf: start with all vertices, at each stage, choose an edge such that adding it doesn't create a cycle.
stop when you can't find one

connected?



if this edge isn't in subgraph, since if we add it to subgraph get a cycle, have an xy path in subgraph.

↑ minimally distant in G, not connected in subgraph

but can also get from u to x & y to w in subgraph

What this argument actually shows:

if H is a subgraph of G which is max'l with respect to being acyclic (i.e. any additional edges create cycles)
then H is a spanning tree.

Thm: If T is a tree with n vertices, then T has $n-1$ edges.

Pf: "prony"