CS314 - Minimum Spanning Trees

Note Title 2/19/2010

Announcements

- Midterm will be Friday after break
- HW- due next Friday

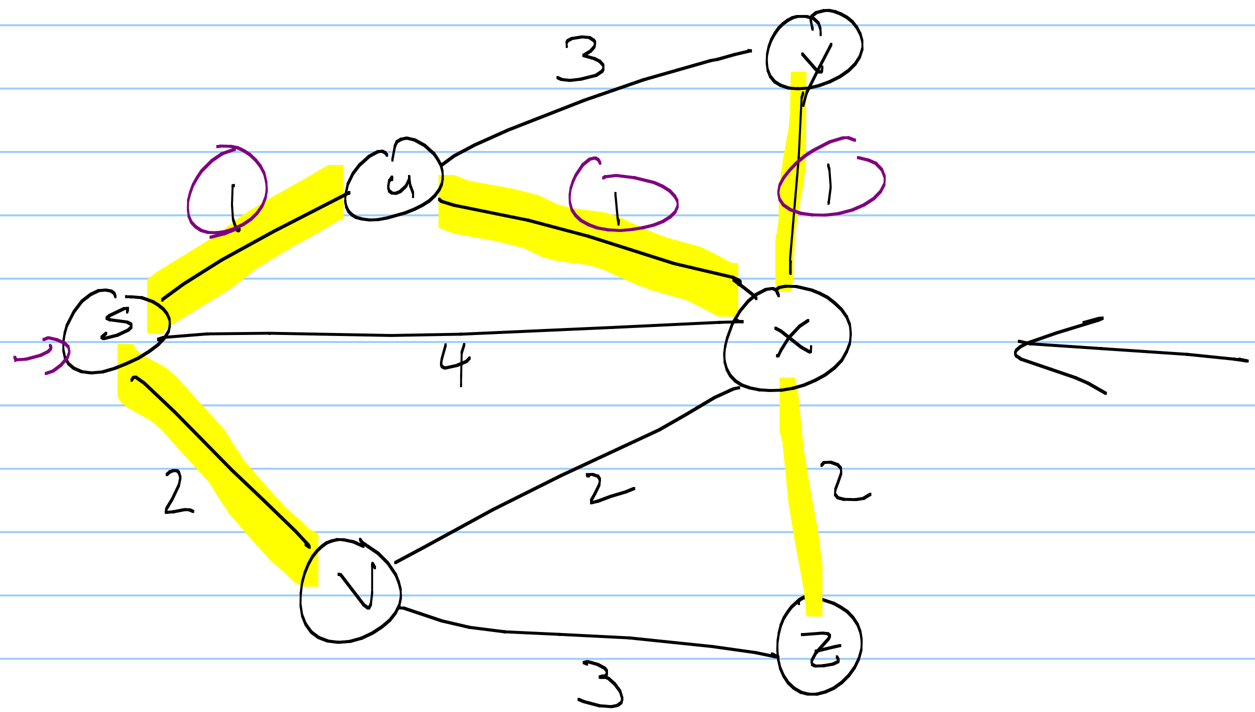
Minimum Spanning Tree

Idea: • Have a set of nodes + want to build communications network on them.
• Have distances (or costs) for each possible connection

Goal: Build cheapest network which connects each pair

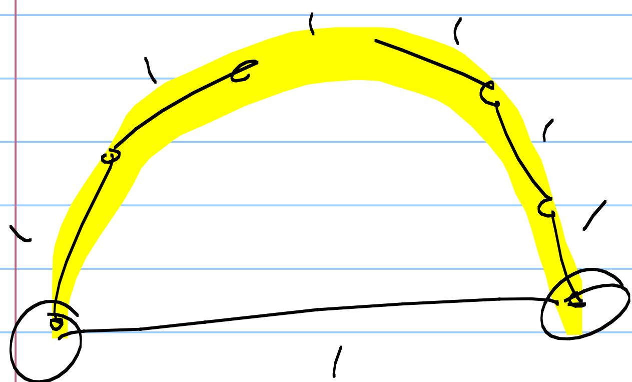
Called MST - minimum spanning tree

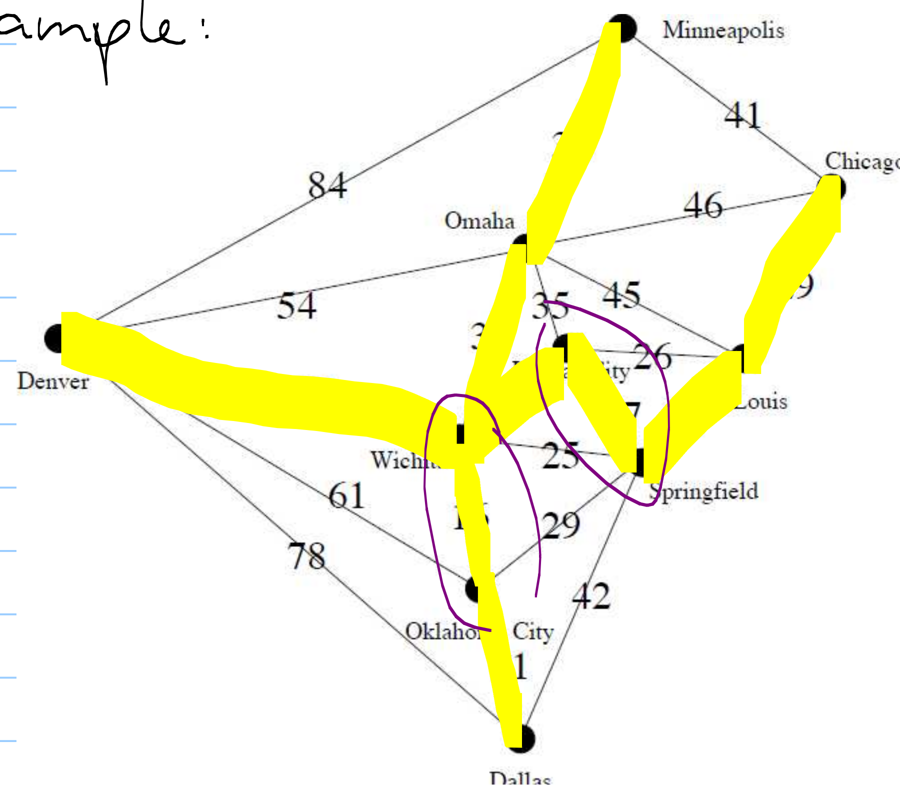
Ex:



weight = 7

More complex example:





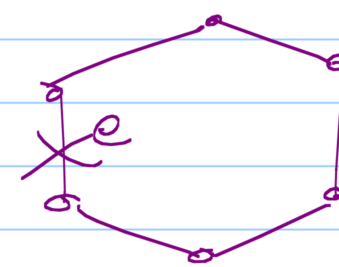
<u>Lemma</u>: Let T be a min cost set of edges connecting the vertices. Then (V,T) is a tree.

<u>pf</u>: Must be <u>connected</u>.

Suppose T had a cycle. (pf by contradiction)

Delete any edge on cycle.

Any path between two nodes which used e can now use rest of cycle instead, so still connected + is smaller. ↯



Greedy ideas:

Prim's — Start at a node v. Add shortest edge uv out of v, & add u to a set S.
At each step, take shortest edge out of S & add it to our tree (& add new vertex to S).

Backwards Kruskal's — Start from V, & order edges longest to shortest. Delete longest edge if it is on a cycle. Recurse.

Kruskal's — Start w/ no edges & process edges, smallest to largest. Add edge to T if it doesn't create a cycle.

MST – the ideal greedy structure

All of these work!

Proofs of correctness can be given easily using a few key lemmas.

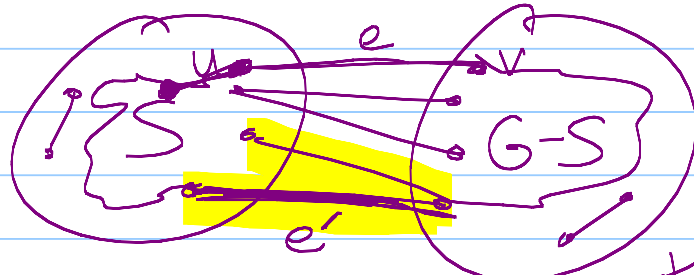
Note: – Graphs are undirected.

– Edge weights are unique.

Cut Property

Lemma: Let S be any subset of V and let e be min-cost edge from S to V−S. Every MST must contain e.

pf:



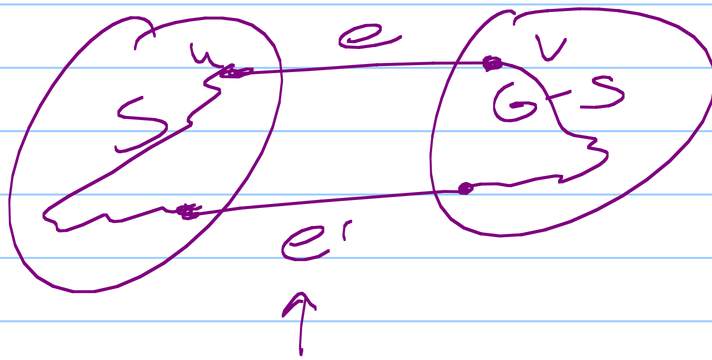
Let $e = \{u, v\}$.

[Suppose have MST T without e.

T is connected, so there is a path from u to v.

$u \in S$ and $v \in G-S$, so path must contain some edge going between S & V−S. Call this edge e'.

pf cont:



Consider $T' = T + e - e'$

T' is still connected, since any path using e' can now reroute through e.

T' is smaller, since e was mincost edge between 2 sets.

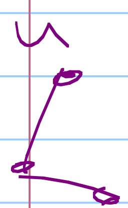
So T is not a MST, since $\text{cost}(T') < \text{cost}(T)$. ∎

Thm: Kruskal's alg works

pf: Consider edge $e=(u,v)$ added by the algorithm.

Let $S = \{u\}$ + anything connected to u just before e was added

Know $u \in S$, $v \in G-S$ (since otherwise adding e creates a cycle).



So e is cheapest edge from S to $G-S$, + therefore belongs in MST.

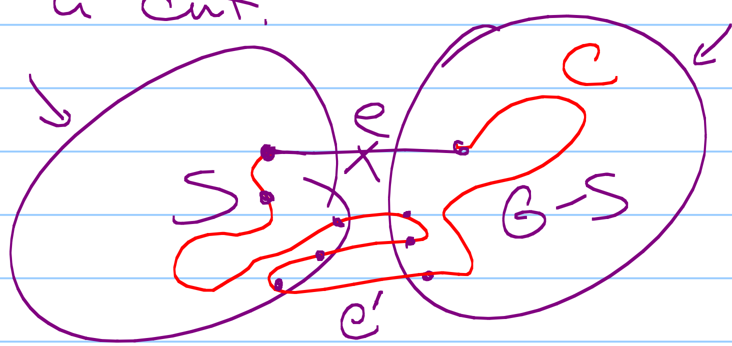
$\blacksquare$

Cycle Property

Lemma: Let C be a cycle in G and let e be the most expensive edge on C. Then e is not in any MST.

pf: Start with a tree T containing e & show T is not an MST.

Take T & delete edge e. This gives a cut.



Want to find an edge to add which is cheaper & reconnects graph.

Create T' by adding e', which is an edge on C going between S & G−S.

Smaller weight, acyclic, & connects everything

So T could not have been MST.