Finding spanning trees, various algorithms

Let G be a graph. Then it has a spanning tree if and only if it is connected.

How to find a spanning tree / show that G is connected?

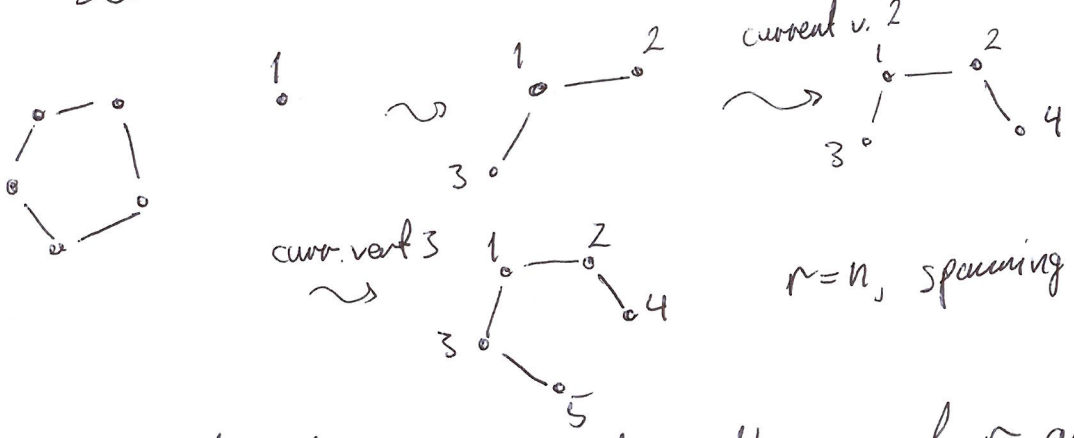
Algorithm: "Breadth first search", build a tree as follows

Initial step: Choose some vertex, call it vertex 1, say "current vertex" is 1.

Then, repeat the following step: Say current vertex is i, & we have added $1, 2, \ldots, r$ to our tree.

- If $r = |V|$, we have a spanning tree.
- Otherwise, add all neighbours of i which are not already added to tree. Label these $r+1, r+2, \ldots, r+j$, & add edges $i(r+1), i(r+2), \ldots, i(r+j)$.
- Set current vertex $= i+1$, & repeat.

Ex



Note, if current vertex is r, all neighbours of r are in tree, $r<n$, then G is disconnected

Proof that if G is connected, then this gives a spanning tree

Proof that this produces a tree: A vertex is added

Every

Proof that this algorithm gives a tree: Starting from 1, every step adds a vertex & an edge. Thus the graph is connected with $|V| - |E| = 1$, so a tree.

Proof that if G is connected, we get a spanning tree:
Let T be the tree generated, let $u = 1 \in V(T)$, and assume for a contradiction $v \in V(G) \setminus V(T)$. Then since G is connected, there is a path $u u_1 u_2 \dots u_k v$.
Let u_i be the first vertex of the path s.t. $u_i \notin V(T)$.
Then u_i is neighbour of u_{i-1}, hence added when u_{i-1} is current vertex. Contradiction.

Depth-first search algorithm: Initial step: Choose $u \in V$, label it 1, current vertex

Then, repeat the following: Current vertex is i

Labelled vertices $1, \dots, r$

If $r = n$, stop

Else

If i has unlabelled neighbour, label it $r+1$, add edge $i(r+1)$.

Parent of $(r+1)$ is i

Current vertex is $r+1$

If i has no unlabelled vertex & $i \neq 1$

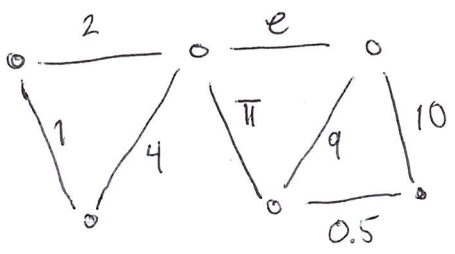
Set current vertex = parent of i

Pf that this works is similar to breadth-first search.

Optimisation problems

Let G be a graph, $w : E \longrightarrow \mathbb{R}_{\geq 0}$ a function called the weight function.

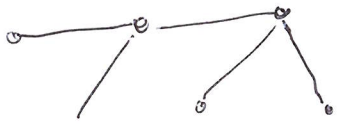
Ex



If T is a spanning subtree, say its weight is

$$w(T) = \sum_{k \in E(T)} w(k)$$

Ex T



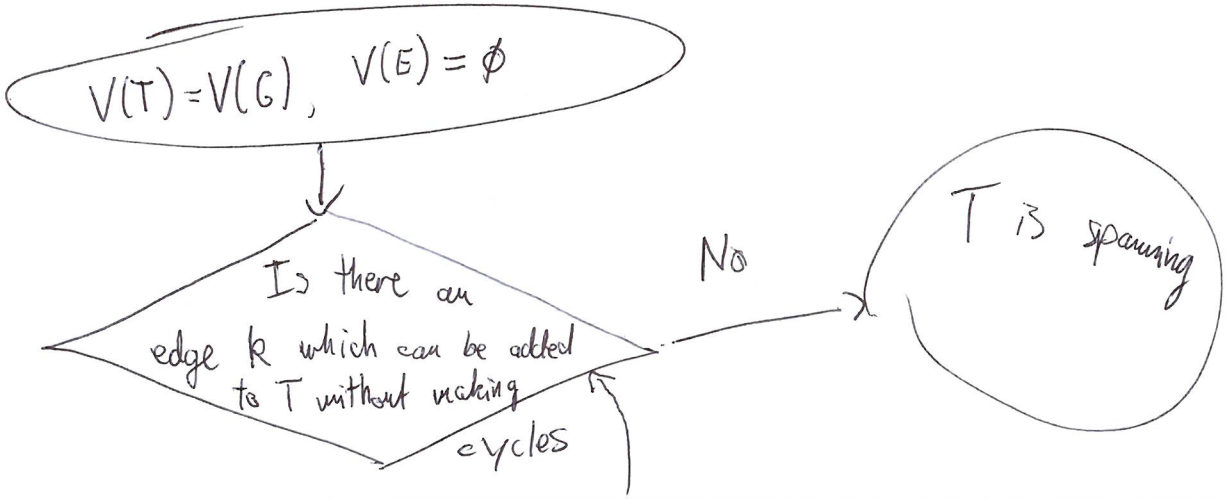
$$w(T) = 2 + 4 + e + 10 + 9$$

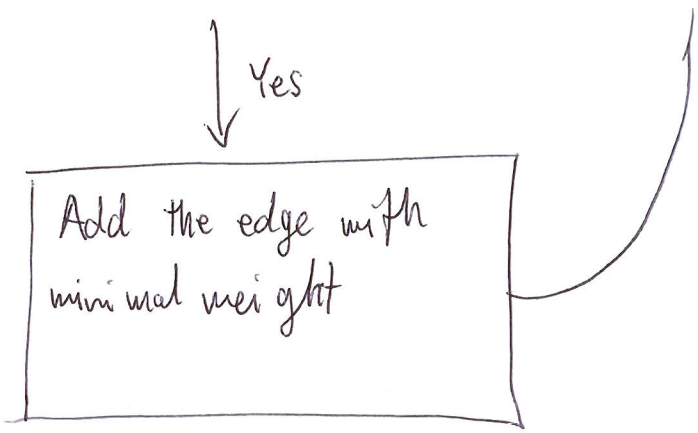
Goal: Find a spanning subtree of minimal weight.

Ex: Vertices are nodes in a power grid, weight of an edge is cost of connecting them. Find cheapest way to connect all nodes.

The greedy algorithm (Kruskal's algorithm)

Assume $G = (V, E)$ is connected with weight function w. To build spanning tree T





Thm This gives a spanning tree of minimal weight.

Matroids Def Let S be a finite set.