17 APRIL 2012 MA375

TREE

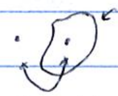
Def: Connected graphs without simple circuits

*simple circuit := a circuit that does not ~~have~~ reuse vertices or edges except that the last vertex is equal to the first.

Note: trees are planar

what are the problems in drawing planar graphs anyway?

simple circuit



two disconnected graphs

since a tree does not have a simple circuit, need not worry abt

Lemma Every tree T has at least one leaf, a vertex of degree at most 1.

Proof:

Pick a vertex of T. Go on a walk not reusing the edges. At some point, this will stop because the number of edges is finite. At this moment, you are standing on a vertex . . .

(i) No edge extends from this vertex. All edges involving this vertex are already part of the path

(ii) the only edge involving this vertex that was ever used is the one we used just now (othrwise, you'd have completed a simple graph at some pt.)

$\Rightarrow \deg(\text{vertex}) = 1$ (or 0 $\overset{\text{e.g.}}{\cdot}$) $\square$

Remark: if you snip off a leaf, T will stay connected

$\Rightarrow$ T − (leaf) has leaf

<u>Thm</u> In a tree $e = v - 1$

<u>Proof</u>

using planarity: $v + f = e + 2$ but tree has no circuits

$v + 1 = e + 2$, $f = 1$

$e = v - 1$ $\square_1$

<u>Proof 2</u>

Inductively Method:

If you remove a leaf, you change vertex and edge by 1. Iterate leaf removal until you are left with an isolates. Then, $v = 1$ $e = 0$ $\square_2$

<u>Def.</u> A graph w/o simple circuits (but potentially not connected) is a <u>forest</u>.

<u>Thm</u> If a forest has c components, then

$e = v - c$

<u>Prf</u>.

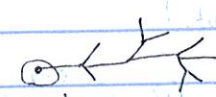
In a forest, e.g.

$$\begin{matrix} T_1 & T_2 & T_3 \\ e_1 = v_1 - 1 & e_2 = v_2 - 1 & e_3 = v_3 - 1 \end{matrix} \Bigg\} \begin{matrix} \Sigma e_i = \Sigma v_i - \Sigma 1 \\ e = v - c \quad \square. \end{matrix}$$

<u>Kinds of Trees</u>

→ <u>rooted tree</u>: one vertex designated root.

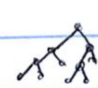
e.g.



root

→ <u>binary tree</u>: rooted tree + each branching is 0/1

e.g.



Note : In binary tree, 3 types of vertex exists :

= root (degree 2), branch (degree 3), leaf (degree 1)

In particular,

$$2e = \sum \deg(\text{vert}) \quad \text{handshake lemma}$$

$$= \sum_{\text{leaf}} \deg(\text{vert.}) + \sum_{\text{branch}} \deg(\text{vert}) + \sum_{\text{root}} \deg(\text{vert})$$

$$= 1(\text{\# of leaves}) + 3(\text{\# of branches}) + 2(1 \text{ root})$$

and $v = (\text{\# of leaves}) + (\text{\# of branches}) + (\text{\# of root} = 1)$

e.g. T is a binary tree w/ 8 vertices & 7 edges.

What does T look like

A:

we have $2e = L + 3B + 2$

$v = L + B + 1$

trees : $v = e + 1$

then,

$14 = L + 3B + 2 \qquad 5 = 2B$

$8 = L + B + 1$

but B should be an integer. Such binary tree cannot exist

e.g. 2. T is a binary rooted tree.

$v = 9$

$e = 8$ what is T like?

$$\left.\begin{aligned} 2e = 16 &= L + 3B + 2 \\ 9 &= L + B + 1 \end{aligned}\right\} \quad 6 = 2B \Rightarrow B = 3, \ L = 5$$

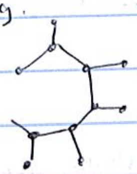
visualizing ...

Ternary Tree

→ rooted → 

→ unrooted → the degree of all none leaves is 3 k-regular tree : all non-leaves have degree k.

e.g.

 $\Longleftrightarrow$ 3-regular tree

2-regular tree $\Longleftrightarrow$ 

for k-reg tree, one has the relationship

$$2e = \sum_{\text{leaves}} 1 + \sum_{\text{branch}} k \quad \text{AND} \quad v = (\#\text{leaves}) + (\#\text{ of nonleaves})$$

<u>spanning trees</u>

Def: G connected. A subgraph T is a spanning tree if : → T is a tree

→ it involves all e. vertex

Ex. 1

$G = \Delta$

$v = 3 \Rightarrow e$ tree $= 3 - 1 = 2$ (tree, ~~v=e-1~~ $e = v - 1$)

choices:

Ex 2

 $\Longleftrightarrow$ 

$v = 4 \Rightarrow e = 4 - 1 = 3.$

choices

The number of spanning trees inside G is truly enormous. In particular, there is no good way of enumerating the possibilities.

<u>Def</u> G is weighted if to each edge an ~~un~~ number (usually non-negative real) has been attached.

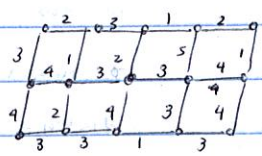
<u>Problem</u>

Given a weighted G, find a spanning tree in tree G of minimum weight

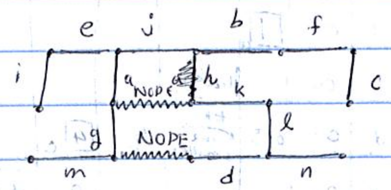
= this is a huge problem with many checks
(even when the # of vert of spanning tree is low)

Algorithm (Kruskal)
(1) Find the lightest edge, and make it part of T
(2) Repeat: add to T the lightest edge of G. that will not close the circuit. until no such edge can be found.

e.g.



$\Rightarrow$



1) abcd = weight 1   3)

2) efgh = — 2   DONE.

Greedy Algorithm

Problem: Accomplish something with minimum
→ Knapsack problem: back pack volume V is given
List of items w/ volume v_i and value r_i
want to select of total vol $\leq V$

greedy idea: (1) Take the most valuable item fills in first
(2) Repeat ~~on~~ until nothing fits

This sometimes work, sometimes doesn't.

Alternative idea:

take only the smallest → only look @ the size.

short term improvements → by order $\frac{r_i}{v_i}$ ~

sometimes good.

but the same problem.