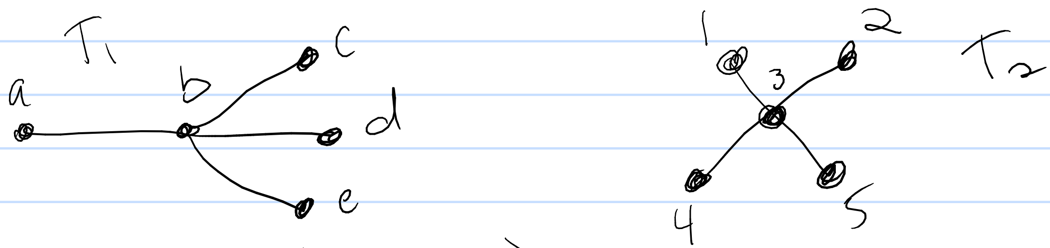
9.8 Isomorphisms of Trees

We will analyze the graph isomorphism problem in the special case where G is a tree.

Ex)



$V(T_1)$ $V(T_2)$

$a \mapsto 1$

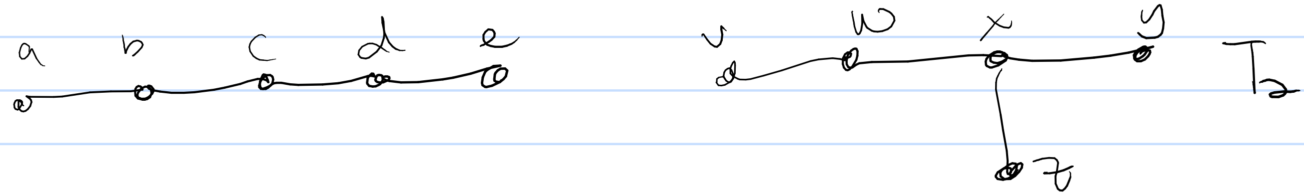
$b \mapsto 3$

$c \mapsto 2$

$d \mapsto 4$

$e \mapsto 5$

Ex)

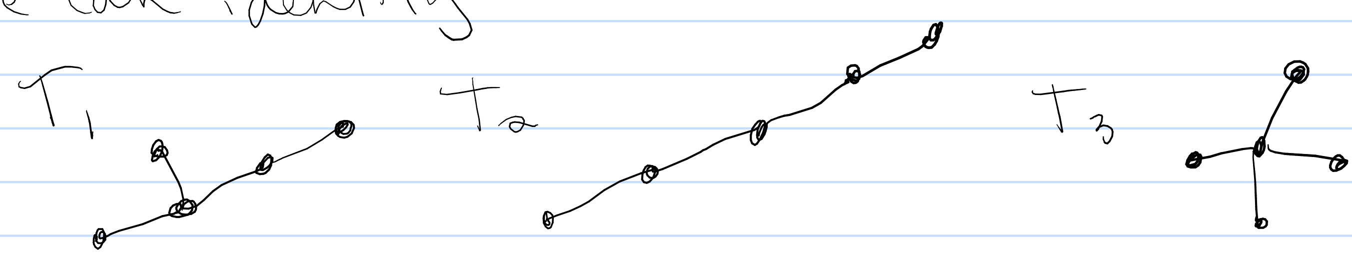


T_1 is not iso to T_2 because T_2 has 3 leaves + T_1 only has 2

Theorem There are 3 nonisomorphic trees with 5 vertices

Pf

We can identify



These are clearly nonisomorphic.
We will show any other such T is isomorphic to T_1, T_2, or T_3.

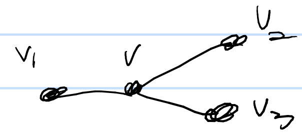
We know $|V(T)| = 5 \Rightarrow |E(T)| = 4$ since T is a tree

Since T is a tree, T is simple $\Rightarrow \delta(v) \leq 4$ for each v

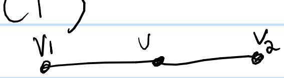
<u>Case 1</u>: T has a degree 4 vertex v.

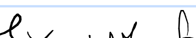
Then there are 4 edges incident to v.
$\Rightarrow$ since $|E(T)| = 4 \Rightarrow T$ is isomorphic to T_3

<u>Case 2</u>: T has a degree 3 vertex v.
Then by the above argument, T has no degree 4 vertex.
We know v is incident on 3 edges by assumption $\Rightarrow T$ has the subgraph



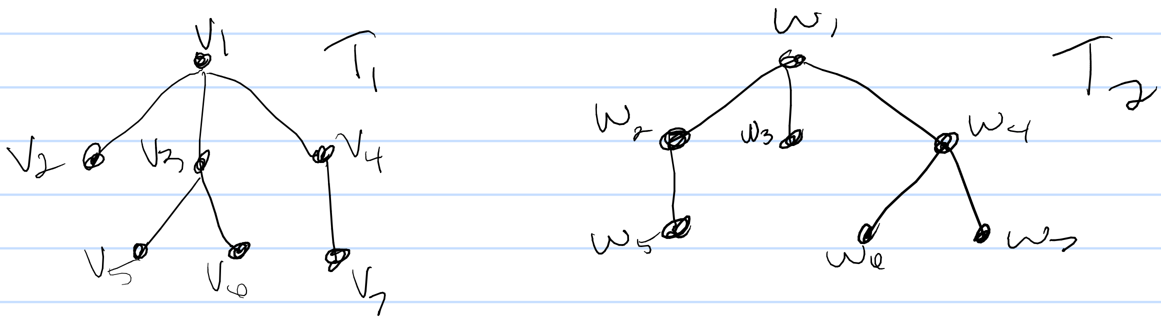
Then there is one vertex w that is not in this subgraph. Since T is a tree $(v_i, v_j) \notin E(T)$ for each i, j so there must be an edge (v_i, w) for one of these v_i. Then this accounts for all edges + vertices
$\Rightarrow T$ is isomorphic to T_1

<u>Case 3</u>: $\delta(v) \leq 2$ for each $v \in V(T)$
Then T has subgraph 

There can be no other edges incident to v.
There are 2 more edges
$\Rightarrow$ must be an edge (v_1, v_3) or (v_2, v_3)
Then the graph looks like  so far.
To add the last edge + vertex we follow the same logic above. $\therefore T$ is isomorphic to T_2

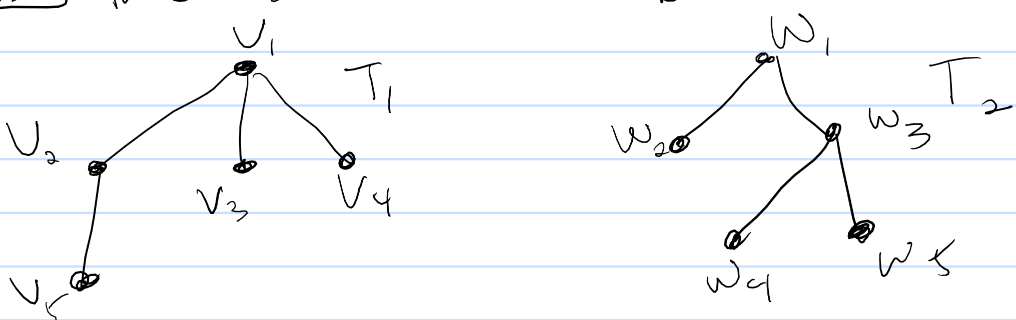
Def Rooted trees T_1, T_2 are isomorphic if there is a bijection $f: V(T_1) \to V(T_2)$ such that $(v_1, v_2) \in E(T_1) \Leftrightarrow (f(v_1), f(v_2)) \in E(T_2)$ where if r_1 is the root of T_1 & r_2 is the root of T_2, $f(r_1) = r_2$

Ex The rooted trees below are isomorphic



$v_1 \mapsto w_1$
$v_2 \mapsto w_3$
$v_3 \mapsto w_4$
$v_4 \mapsto w_2$
$v_5 \mapsto w_6$
$v_6 \mapsto w_7$
$v_7 \mapsto w_5$

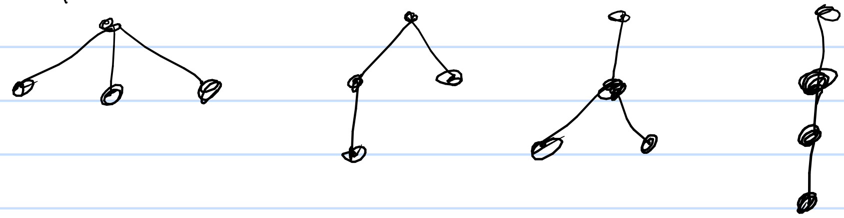
Ex The rooted trees below are not isomorphic



The root of T_1 has degree 3 but the root of T_2 has degree 2

(Note: these are isomorphic as free trees (i.e. the non-rooted version))

Theorem There are 4 non-isomorphic rooted trees with 4 vertices. These are listed below



Pf sketch

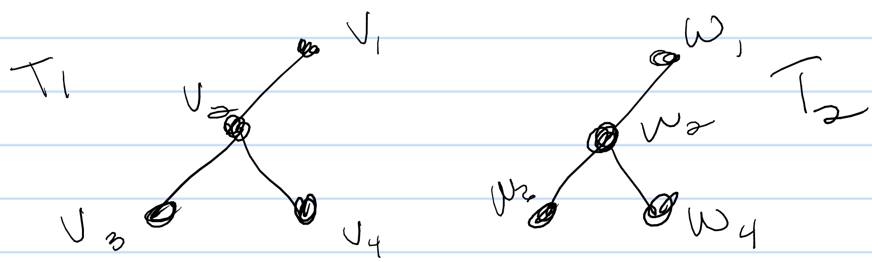
Similar to previous theorem, consider cases based on the degree of vertices at each level.

Def Let T_1 be a binary tree with root r_1 & T_2 be a binary tree with root r_2.
T_1 and T_2 are isomorphic if there is a bijection $f: V(T_1) \to V(T_2)$
such that ① $(v_1, v_2) \in E(T_1) \Leftrightarrow (f(v_1), f(v_2)) \in E(T_2)$
where if ② r_1 is the root of T_1 &

③ r_2 is the root of T_2, $f(r_1) = r_2$,
v is a left child of w in T_1
$\Leftrightarrow$
$f(v)$ is a left child of $f(w)$ in T_2

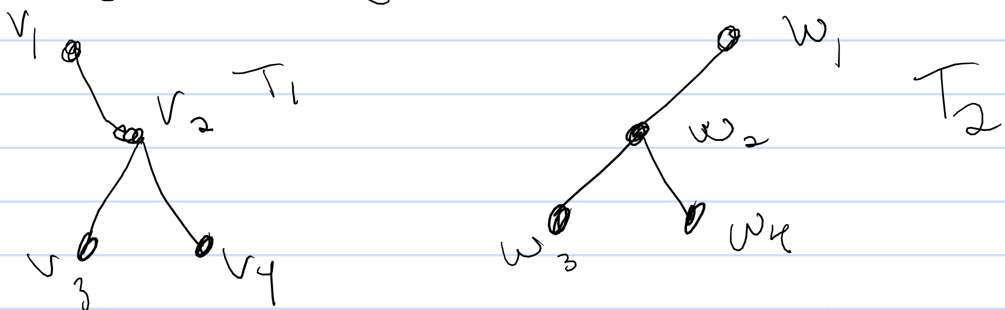
④ v is a right child of w in T_1
$\Leftrightarrow$
$f(v)$ is a right child of $f(w)$ in T_2

Ex The binary trees below are isomorphic



$v_i \mapsto w_i$

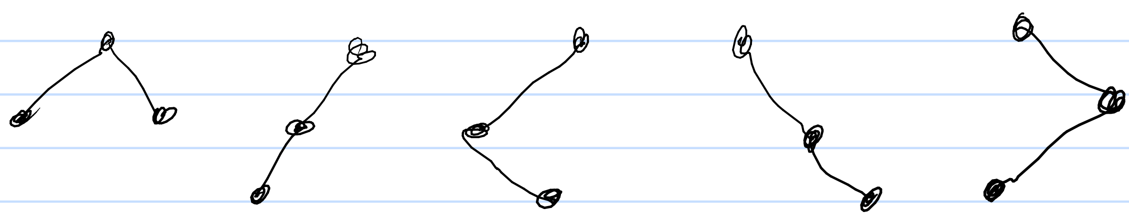
Ex The binary trees below are not isomorphic



The root v_1 in T_1 has no left child but the root w_1 in T_2 does.

T_1 + T_2 are isomorphic as rooted trees + as free trees.

Theorem There are 5 nonisomorphic binary trees with 3 vertices. These are listed below



Theorem There are C_n nonisomorphic binary trees with n vertices where

$$C_n = \frac{1}{n+1} \cdot \binom{2n}{n}$$

Pf

Let a_n = # binary trees on n vertices.
We see $a_0 = 1$

We prove this by deriving a recurrence relation.
The Catalan numbers C_n are defined by the recurrence relation

$$C_n = \sum_{k=0}^{n-1} C_k C_{n-k-1} \quad n \geq 1$$

and initial condition $C_0 = 1$.

Thus since $a_0 = 1 = C_0$, the result will follow if we prove that

$$a_n = \sum_{k=0}^{n-1} a_k a_{n-k-1} \quad n \geq 1.$$

Consider how to construct a binary tree on n vertices for $n > 0$.
One vertex will be the root. There are $n-1$ other vertices.
Say the left subtree has k vertices (so the right subtree has $n-k-1$ vertices)
There are a_k ways to construct the left subtree + a_{n-k-1} ways to construct the right.
By the Mult. Principle, choosing the tree in total will be $a_k a_{n-k-1}$ ways.
By the Addition principle, we sum over all possible k so

$$a_n = \sum_{k=0}^{n-1} a_k a_{n-k-1}$$, so we are done.

Although deciding if graphs G_1, G_2 are isomorphic is hard, doing this for binary trees is easy (linear time):

$\boxed{\text{Algorithm}}$ INPUT: roots r_1, r_2 of binary tree's T_1, T_2
(if $T_1 = \emptyset \Rightarrow r_1 = \text{null}$
if $T_2 = \emptyset \Rightarrow r_2 = \text{null}$)

OUTPUT: True if T_1 is iso. to T_2
False otherwise

```
bin_tree_isom(r1, r2){
    if (r1 = null and r2 = null)
        return true
    if (r1 = null and r2 ≠ null) or (r1 ≠ null and r2 = null)
        return false
    lc_r1 = left child of r1
    lc_r2 = left child of r2
    rc_r1 = right child of r1
    rc_r2 = right child of r2
    if bin_tree_isom(lc_r1, lc_r2) = True and
            bin_tree_isom(rc_r1, rc_r2) = True :
                return True
    Otherwise
            return False
```