CS 2336
Discrete Mathematics

Lecture 16

Trees: Introduction

Outline

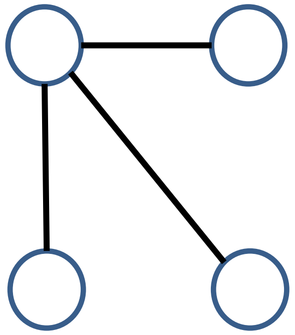
- What is a Tree?
- Rooted Trees
- Properties of Trees
- Decision Trees

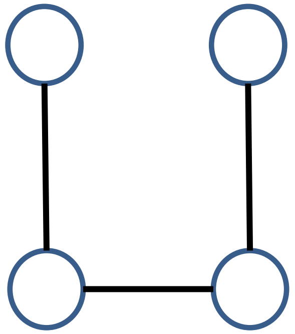
What is a Tree?

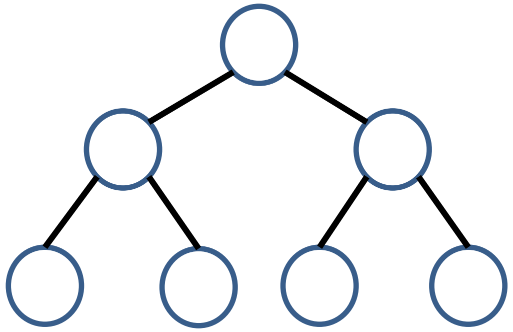
- A path or a circuit is simple if it does not contain the same edge more than once.

> Definition : A tree is a connected undirected graph that does not contain a simple circuit.

- Ex :







What is a Tree ?

- By definition, a tree must be a simple graph

  That is, with no loops, no multiple edges (why?)

- Let G be an undirected graph

Theorem : G is a tree $\Leftrightarrow$ there is a unique simple path between any two vertices.

- Proof (see the following pages)

What is a Tree ?

- Proof (“if” case) :

  If there is a path between any two vertices, then

  (i) G must be connected.

  Further, we can see that

  (ii) G cannot have any simple circuits.

  Otherwise, there are two vertices such that more than one simple path exist between them.

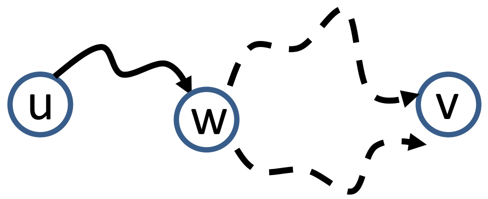
  ➔ By (i) and (ii), G is a tree

What is a Tree ?

- Proof ("only if" case) :

  G is a tree, so that it is connected. This implies for any two vertices u and v, there must be a simple path between them.

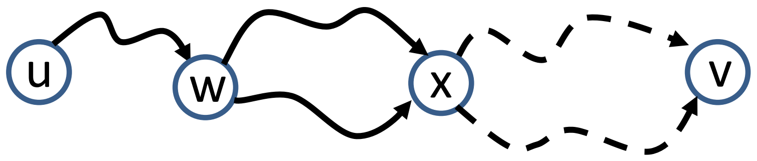
  Further, we can see that no other simple paths can exist. Otherwise, consider moving along two of those paths from u to v. Let w be the first node where the two paths split.



What is a Tree ?

- Proof ("only if" case continued) :

  Node w exists since the two paths are different. Similarly, we can find the first node x where the two paths meet again.



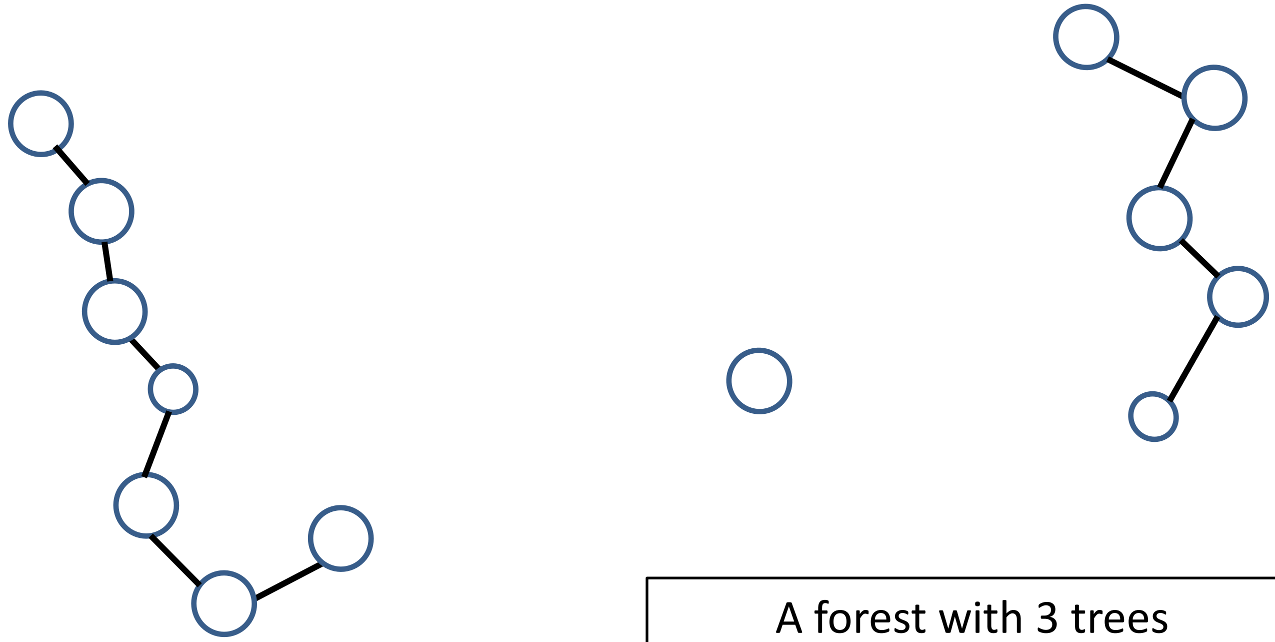
  It is easy to check that there is a simple circuit containing w and x ➔ contradicting G is a tree!

  Thus, only a unique simple path between u and v

What is a Tree?

Definition : If each component of a graph G is a tree, G is called a forest.

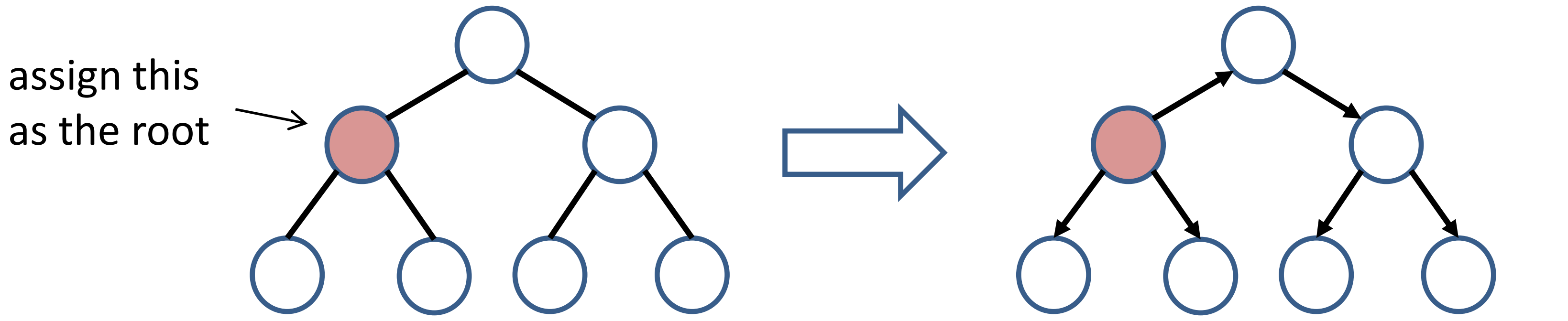
- Ex :



A forest with 3 trees

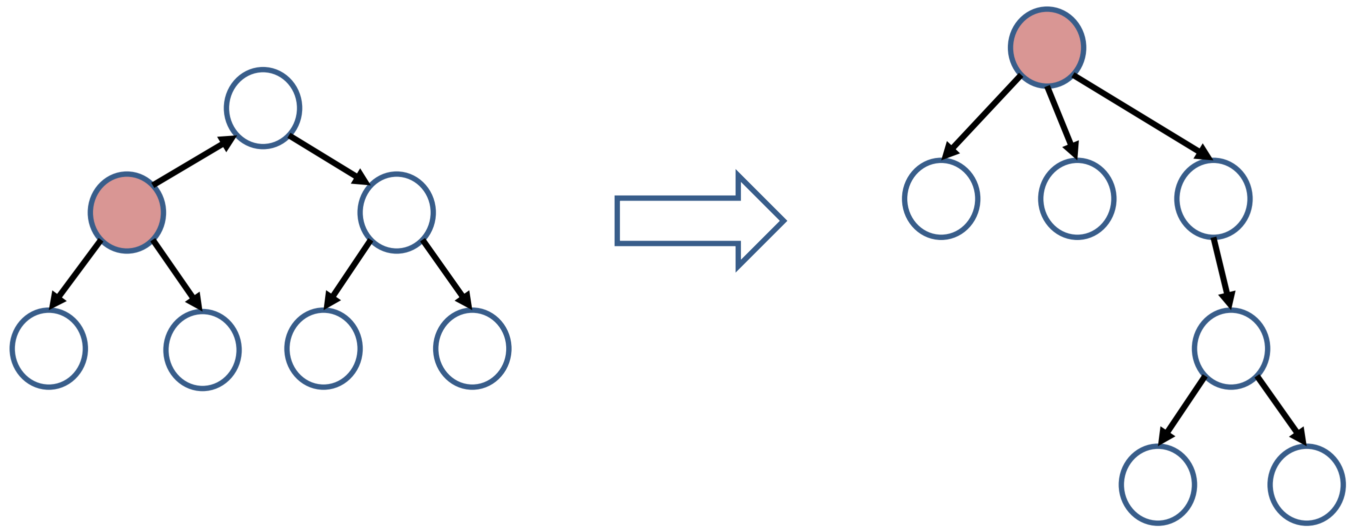
Rooted Trees

- In many applications of trees, a particular vertex is designated as the root
- Once we specify the root, we can direct each edge away from the root and get a rooted tree
- Ex :



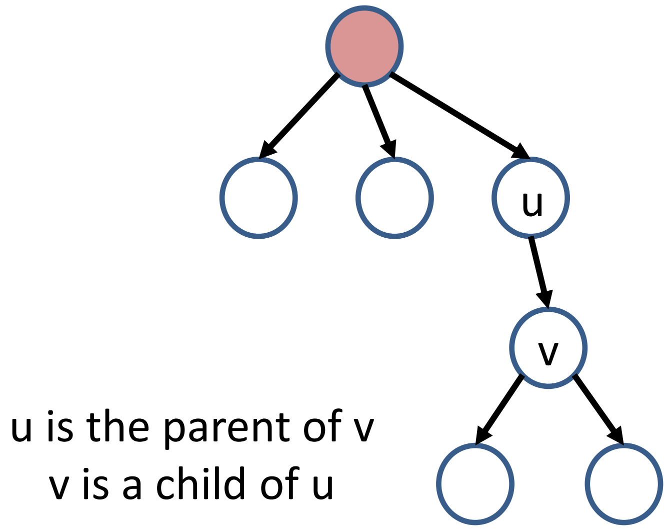
Rooted Trees

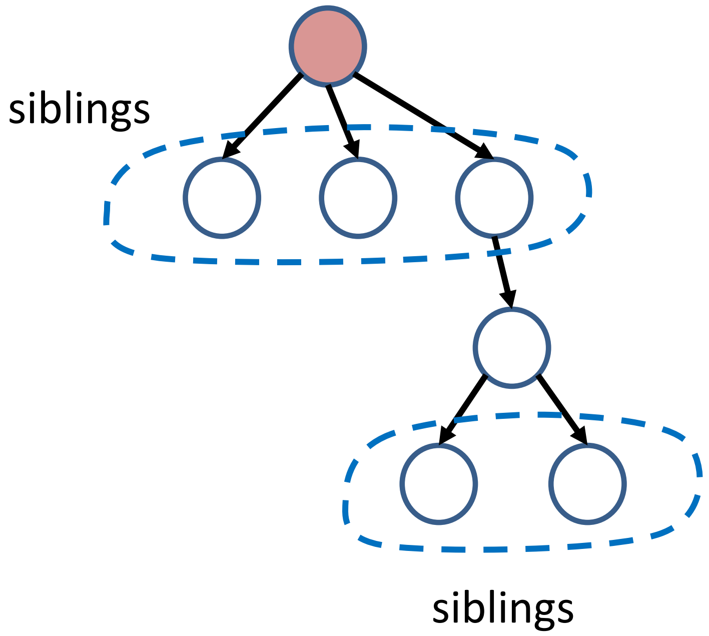
- When we draw a rooted tree, we usually put the root at the top, and point each edge downwards
- Ex :



Rooted Trees

- Each edge is from a parent to a child
- Vertices with the same parent are siblings

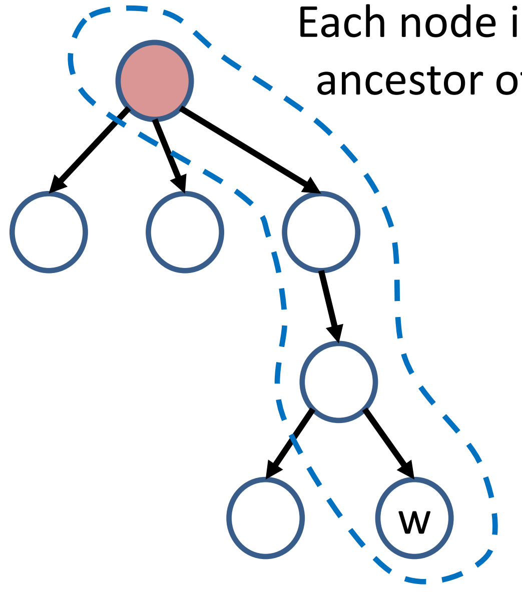




- Remark: The parent of a vertex is unique (why?)

Rooted Trees

- The ancestors of a vertex w include all the nodes in the path from the root to w
- The proper ancestors of a vertex w are the ancestors of w, but excluding w

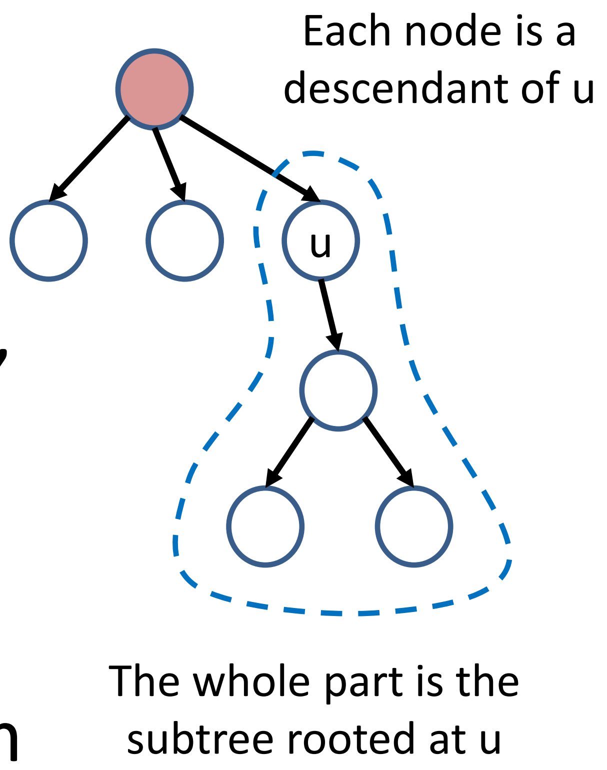


Each node is an ancestor of w

The whole part forms a path from root to w

Rooted Trees

- The descendants of a vertex u include all the nodes that have u as its ancestor
- The proper descendants of a vertex u are the descendants of u, but excluding u
- The subtree rooted at u includes all the descendants of u, and all edges that connect between them

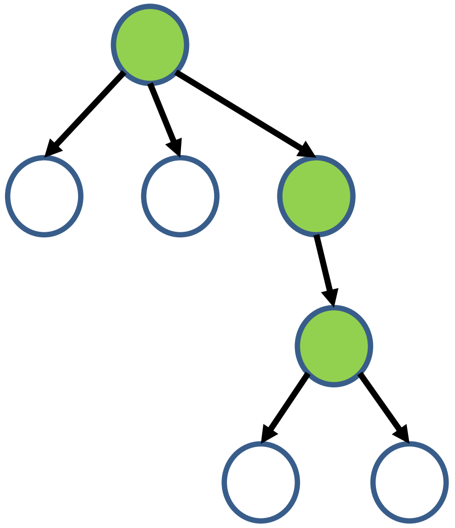

Each node is a descendant of u
u
The whole part is the subtree rooted at u

Rooted Trees

- Vertices with no children are called leaves ;

  Otherwise, they are called internal nodes
- If every internal node has no more than m children, the tree is called an m-ary tree
  - Further, if every internal node has exactly m children, the tree is a full m-ary tree

All internal nodes are colored



The tree is ternary (3-ary), but not full

Properties of Trees

Theorem : A tree with n nodes has $n - 1$ edges.

- Proof :

[Method 1 : By Induction (Try yourself)]

[Method 2 : Via Rooted Trees]

Pick a vertex u in the tree and make u the root. Each edge now links a parent and a child.

The root has no parent, but every other node has exactly one parent $\rightarrow$ # of edges = $n - 1$

Properties of Trees

Theorem : A full m-ary tree with i internal nodes has $mi + 1$ nodes.

- Proof : Every node, apart from the root, is a child of an internal node

  ➔ excluding the root : mi nodes

  ➔ including the root : $mi + 1$ nodes

Properties of Trees

Corollary : A full m-ary tree with i internal nodes has $mi - i + 1$ leaves.

- Ex : Peter starts out a chain mail. Each person receiving the mail is asked to send it to four other people. Some people do this, and some don't

  Now, there are 100 people who received the letter but did not send it out

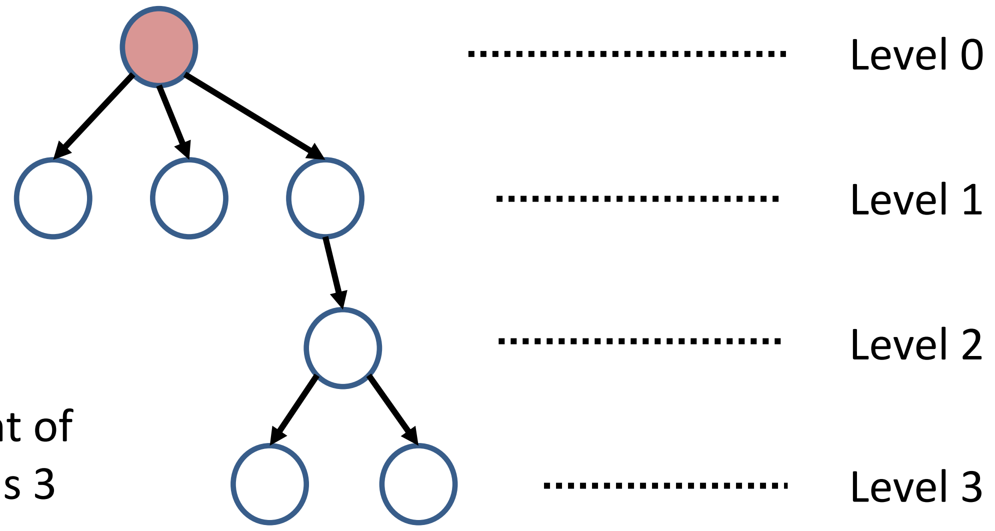
  Assuming no one receives more than one mail. How many people have sent the letter ?

Properties of Trees

Definition : The level of vertex v in a rooted tree is the length of the path from root to v

The maximum level of all the vertices is called the height of the tree

- Ex :

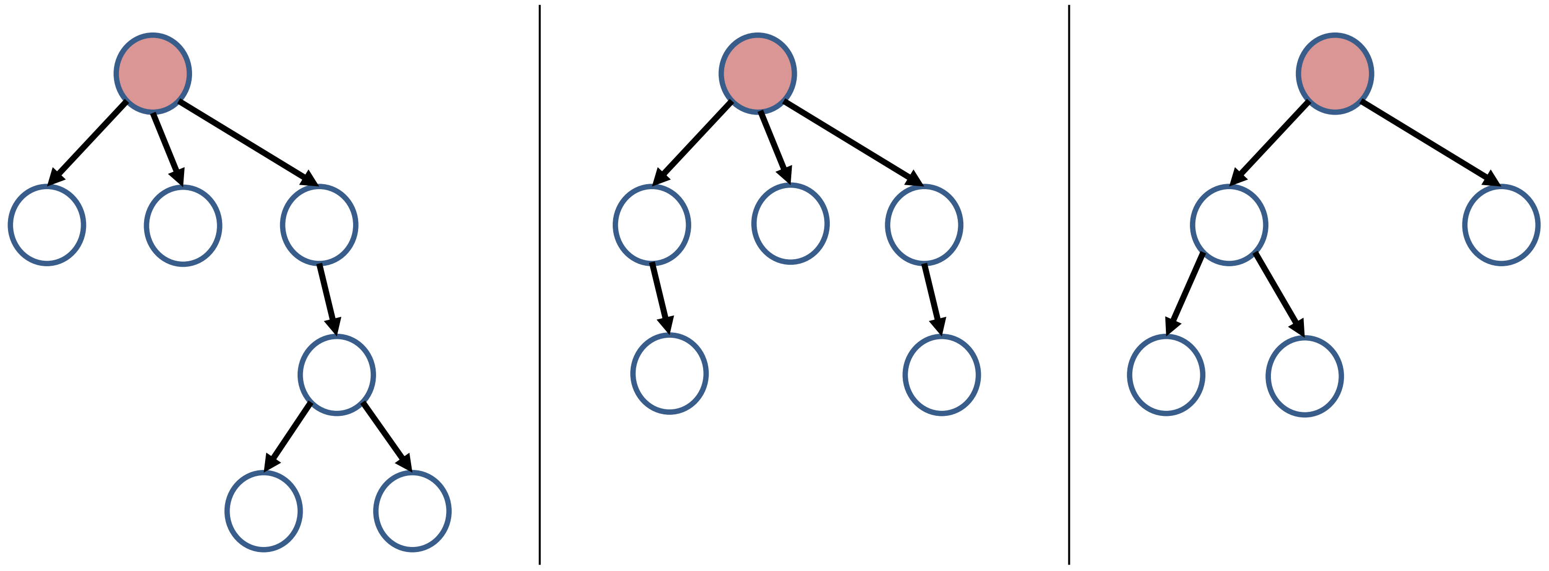


This height of the tree is 3

Properties of Trees

Definition : A rooted tree of height h is balanced
if all the leaves are at level h or h – 1

- Ex : Which of the following trees are balanced ?



Properties of Trees

Theorem : In an m-ary tree with height h, there are at most m^h leaves.

- Proof : Make the tree into a full tree T by adding leaves (if necessary). To prove the theorem, it is sufficient to show that T has at most m^h leaves.

  In tree T, at most m^{k-1} internal nodes at level k

  ➔ # internal nodes $\leq 1 + m + \ldots + m^{h-1}$

  ➔ # leaves $\leq (m - 1)(1 + m + \ldots + m^{h-1}) + 1 = m^h$

Properties of Trees

Corollary : If a full and balanced m-ary tree ($m \geq 2$) has height h and x leaves, $h = \lceil \log_m x \rceil$.

- Proof :

(i) By the previous theorem $\rightarrow$ $x \leq m^h$

(ii) Since the tree is full, removing all nodes at level h gives a tree T with fewer leaves ($m \geq 2$) ;

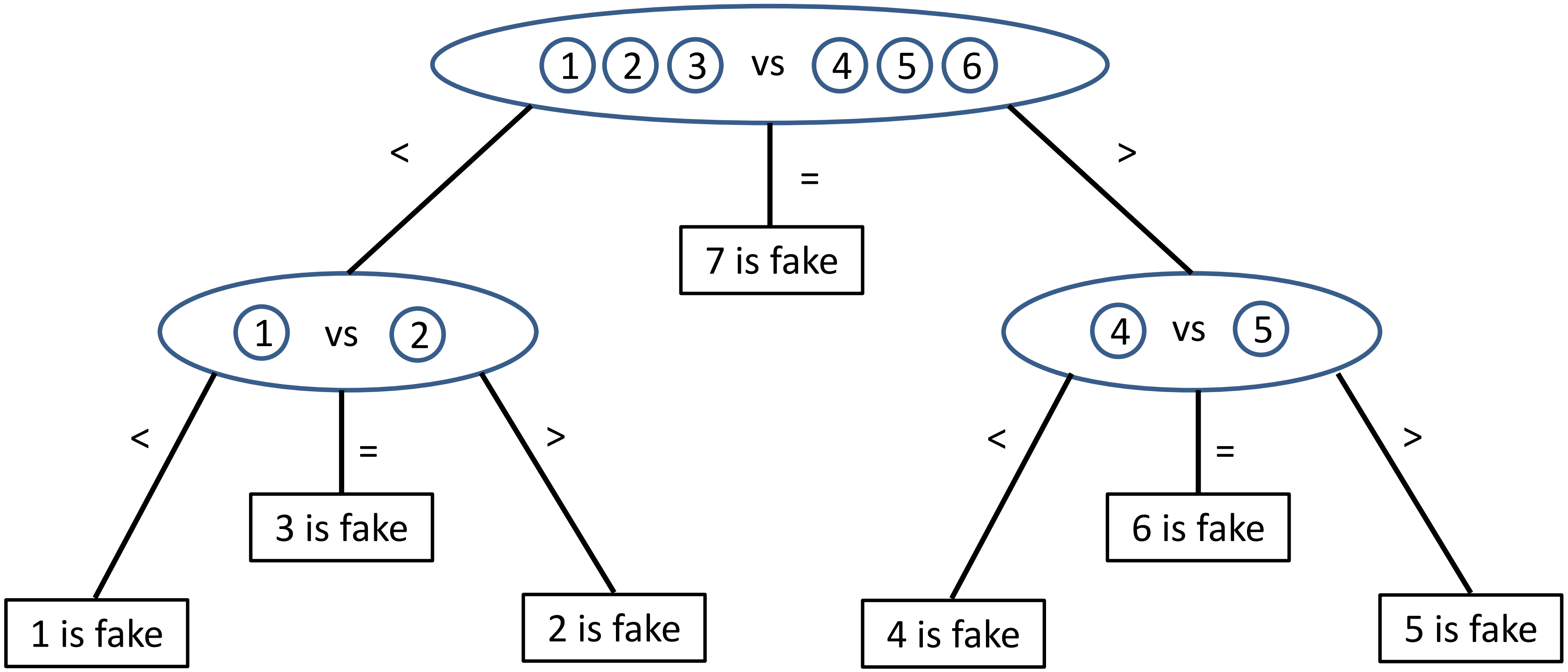
Tree T has exactly m^{h-1} leaves (since original tree is full and balanced) $\rightarrow$ $x > m^{h-1}$

The theorem follows from (i) and (ii)

Decision Trees

- Rooted trees can be used to model problems in which a series of decisions leads to a solution
- Each internal node v corresponds to a decision to be made, and each child of v corresponds to a possible outcome of the decision
- Example 1: There are 6 true coins with the same weight, and a fake coin with less weight. If all the seven coins are mixed, how to use a balance to find the fake one in two weighings?

Decision Trees



Decision Trees

- Example 2 :
  - -- There are 3 distinct numbers : A, B, C
  - -- Our target is to sort these numbers
  - -- No one is going to tell us their exact values, but we can pick any two of them, and compare which is larger

  How many comparisons are sufficient ?

  What if we begin with 4 distinct numbers ?

Decision Trees

- Answer :

  For 3 numbers: 3 comparisons are sufficient

  For 4 numbers: 5 comparisons are sufficient

  Step 1. Sort A, B, C in 3 comparisons

  Step 2. Compare D with B

  Step 3. Compare D with A or C, depending the outcome of Step 2

- We shall show that these methods are optimal !

  (That is, 3 and 5 comparisons are necessary)

Decision Trees

Theorem : In the worst case, sorting n distinct numbers needs $\lceil \log_2 n! \rceil$ comparisons.

- Proof : Any sorting method corresponds to a binary decision tree with at least $n!$ leaves (why?)

$\Rightarrow$ In the worst case,

number of comparisons

= # internal nodes on longest root-leaf path

= height of the tree $\geq \lceil \log_2 n! \rceil$