Introduction to Tree

Introduction

There are many situations in which information has a hierarchical or nested structure like that found in family trees or organization charts. The abstraction that models hierarchical structure is called a tree and this data model is among the most fundamental in computer science. It is the model that underlies several programming languages, including Lisp.

Definition:

A tree is an undirected connected graph with no simple circuits, no multiple edges, no loops. Therefore, any tree must be a simple graph.

Ex 1: Which of the graphs shown in Figure 1 are trees?

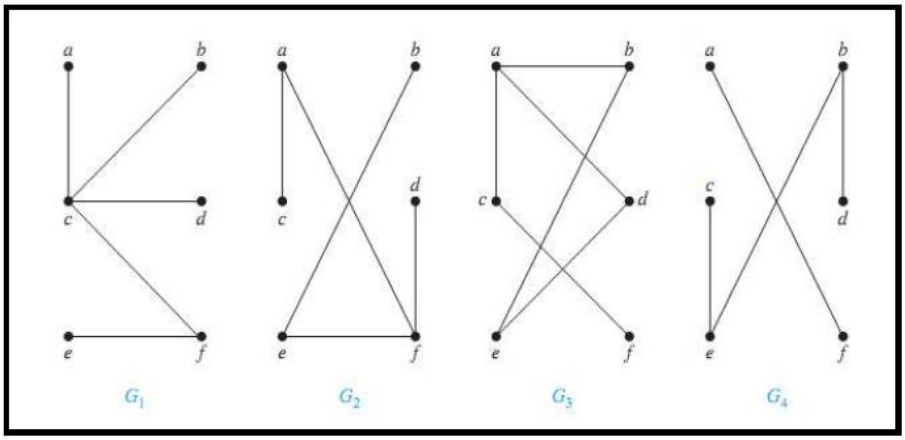


Figure1: Examples of Trees and Graphs That Are Not Trees.

Sol:

G1 and G2 are trees, because both are connected graphs with no simple circuits.
G3 is not a tree because e, b, a, d, e is a simple circuit in this graph.
G4 is not a tree because it is not connected.

The nodes are the elements of a tree.
The branches are the lines between nodes.
The root is the top element in the tree.

A rooted tree is a tree in which one vertex has been designated as the root and every edge is directed away from the root

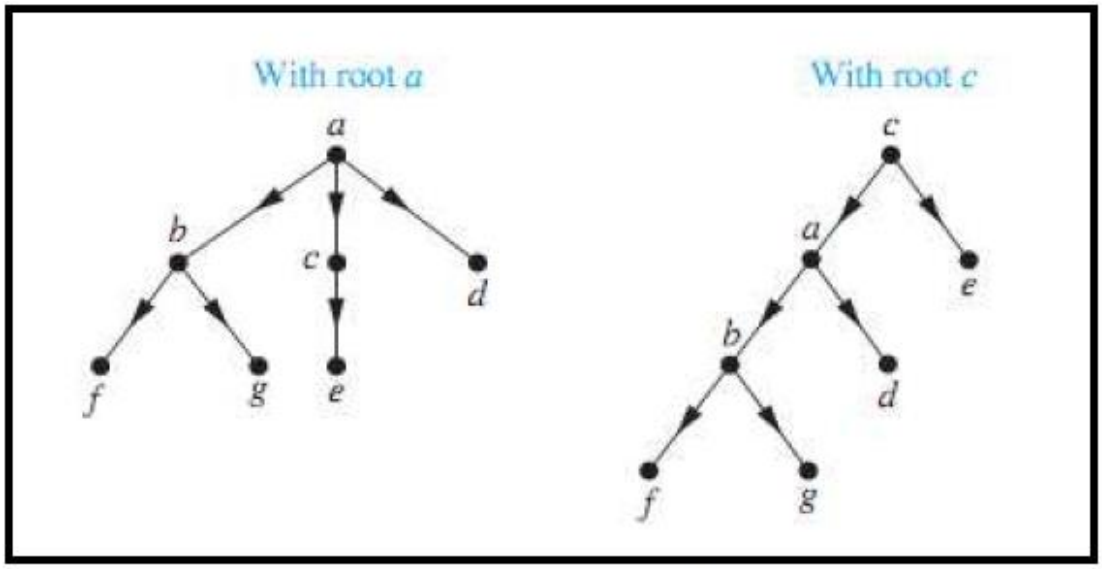


Figure 2: Examples of Rooted Trees.

Notice

1- An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.
2- The tree is set of straight line segments connected at theor ends containing no closed loops (cycles)
3- the tree with n vertices has n-1 edges
4- the trees have their own terminology

Tree Terminologies

Suppose the following rooted tree. The common tree terminologies are as follows:

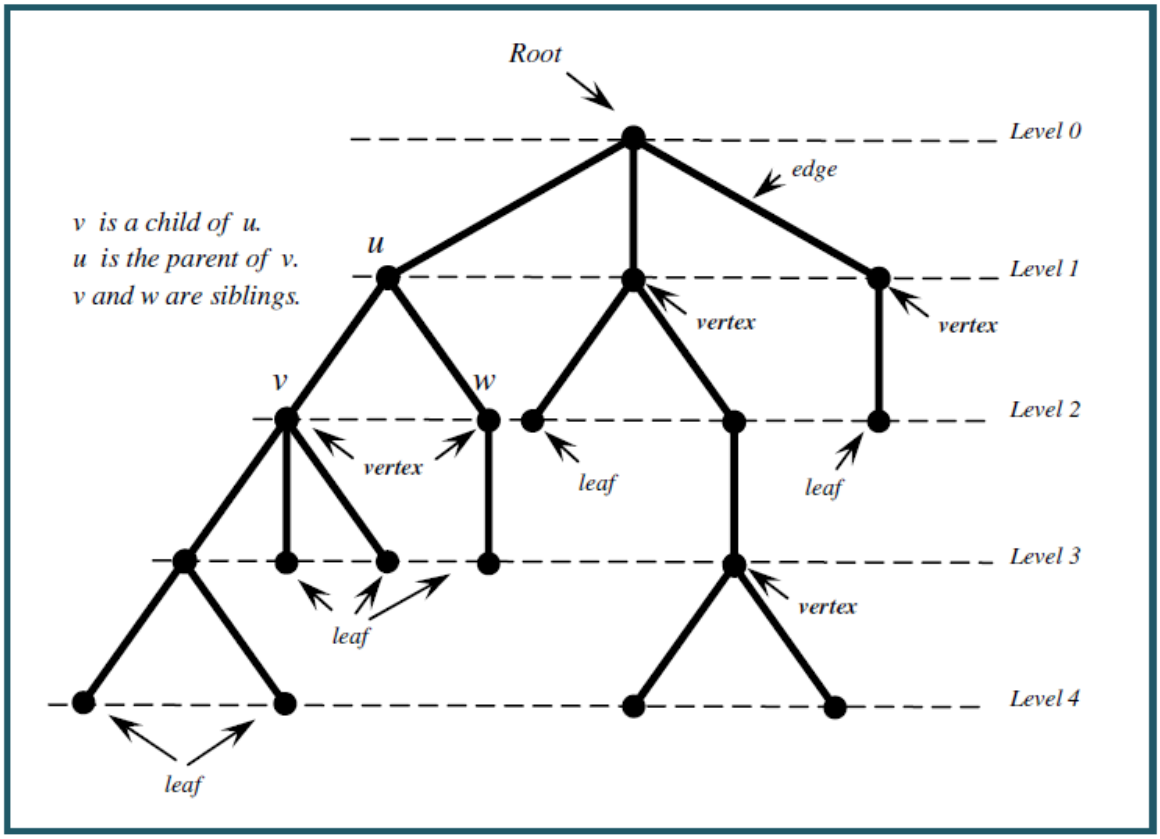
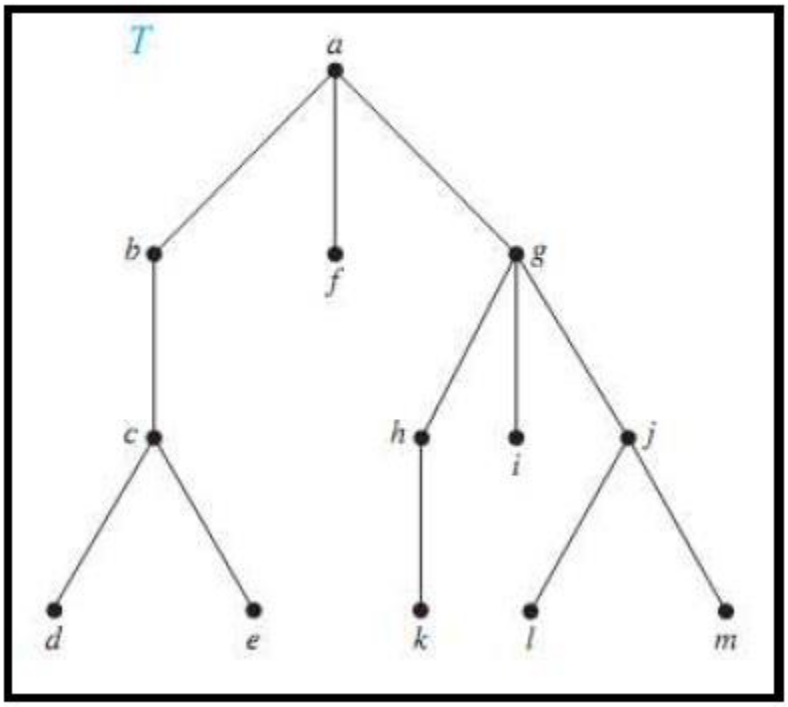


Figure 3: Tree

Parent: the parent of v is the unique vertex u such that there is a directed edge from u to v.
Child: When u is the parent of v, v is called a child of u.
Siblings: siblings are vertices with the same parent.
Ancestors: the ancestors of a vertex other than the root are the vertices in the path from the root to this vertex, excluding the vertex itself and including the root.
Descendants: The descendants of a vertex v are those vertices that have v as an ancestor.
Leaf: A vertex that has no children.
Internal Vertices: they are the vertices that have children.
Level of a Vertex: the level of vertex in a rooted tree is the length of the unique path from the root to this vertex. The level of the root is defined to be zero.

The Height of a Rooted Tree: the height is the maximum level of vertices

Ex: In the rooted tree T shown in the figure below, find the root, the parent of c, the children of g, the siblings of h, all ancestors of e, all descendants of b , all internal vertices, and all leaves.



Sol:
The root is a.
The parent of c is b.
The children of g are h, i, and j.
The siblings of h are i and j.
The ancestors of e are c, b, and a.
The descendants of b are c, d, and e.
The internal vertices are a, b, c, g, h, and j.
The leaves are d, e, f , i, k, l, and m.

An m-ary Tree

A rooted tree is called an m-ary tree if every internal vertex has no more than m children. The tree is called a full m-ary tree if every internal vertex has exactly m children.

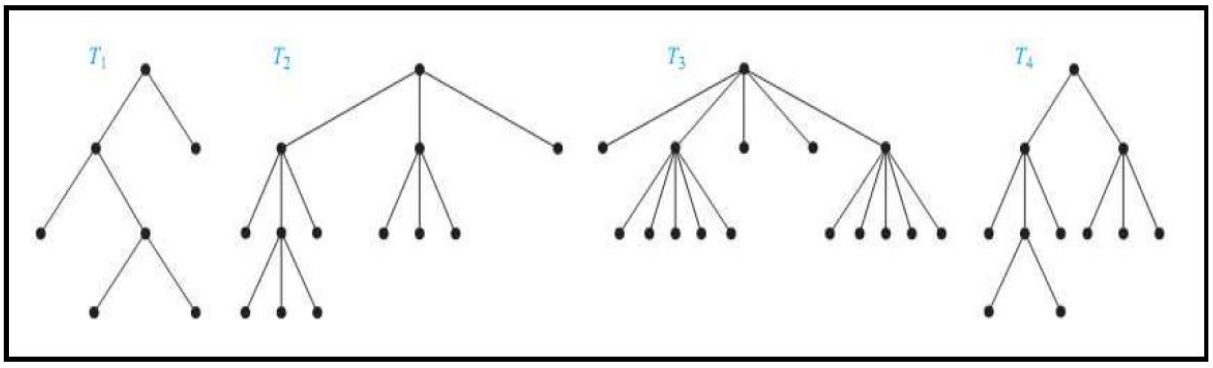
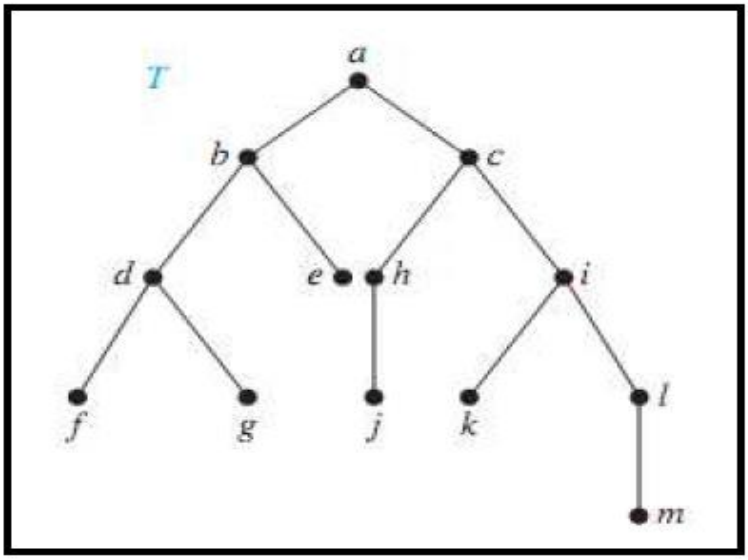


Figure 4: Examples on m-ary trees

From figure 4, we notice that it includes different m-ary trees as follows:
T1: is a full 2-ary tree; because each of its internal vertices has two children (m=2).
T2: is a full 3-ary tree; because each of its internal vertices has three children (m=3).
T3: is a full 5-ary tree; because each internal vertex has five children (m=5).
T4: is not a full m-ary tree for any m; because some of its internal vertices have two children and others have three children. it is just 3-ary tree.

Ordered rooted Tree

An ordered rooted tree is a rooted tree where the children of each internal vertex are ordered from left to right.



Tree Traversal

Tree Traversal is a procedure for visiting each vertex of an ordered rooted tree in order to process the data in that vertices and building a list of the results. Ordered rooted trees can also be used to represent various types of expressions, such as arithmetic expressions involving numbers, variables, and operations.

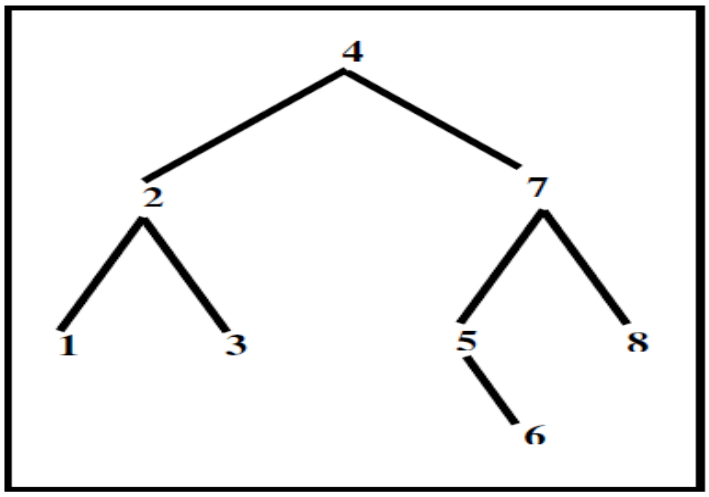
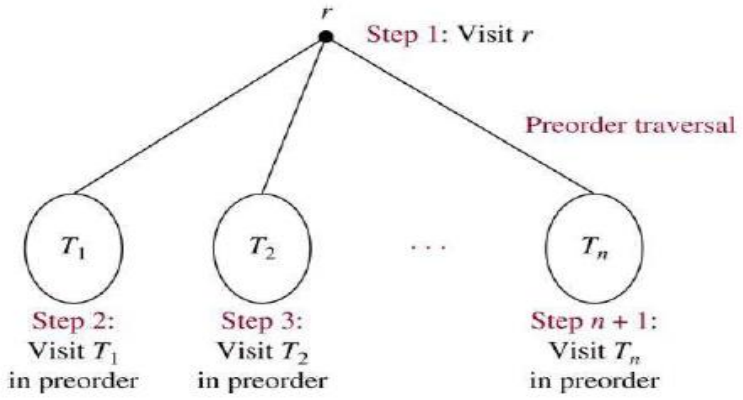


Figure 5: An ordered Tree

The three common traversal Algorithms are as follows:

1- Preorder traversal:

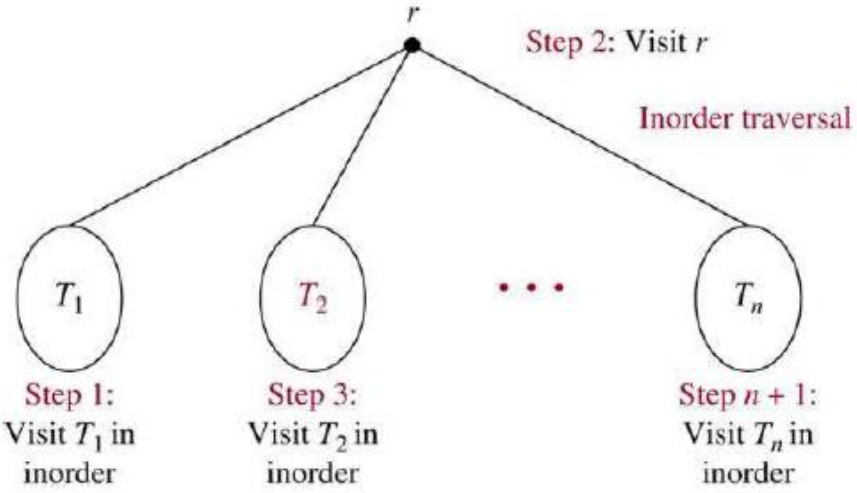
In this algorithm, the root is firstly visited, then traverse the left subtree, then traverse the right subtree.



A preorder traversal of the tree in figure 5 is: 4, 2, 1, 3, 7, 5, 6, 8.

2- Inorder traversal:

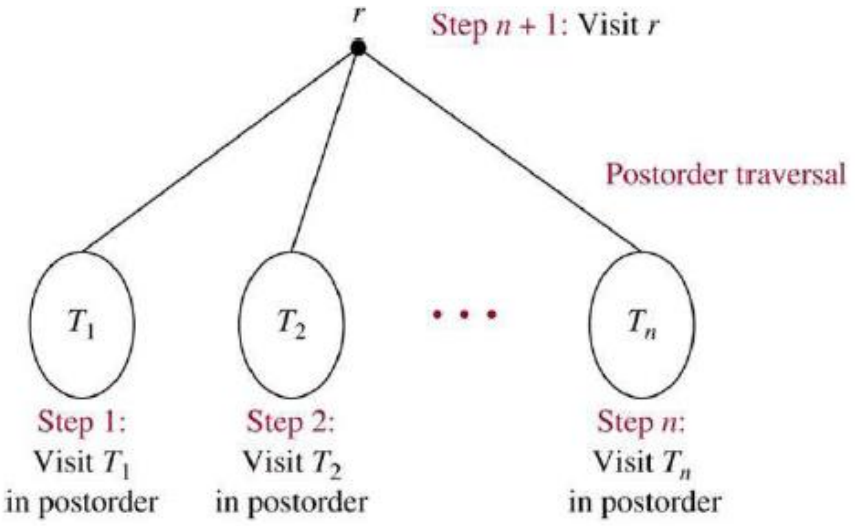
In this algorithm, the left subtree is firstly visited, then the root, then the right subtree.



An inorder traversal of the tree in the figure 5 is: 1, 2, 3, 4, 5, 6, 7, 8.

3- Postorder traversal:

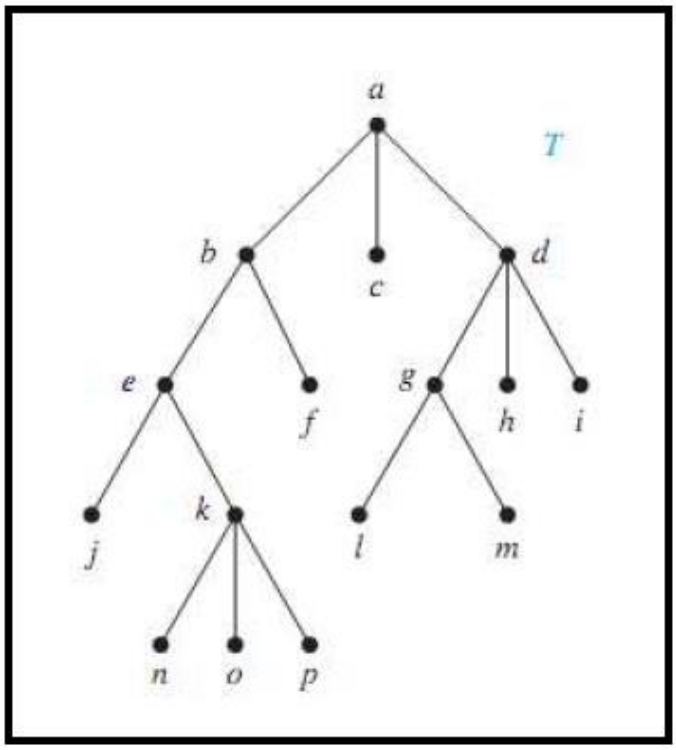
In this algorithm, traverse the left subtree, then the right subtree, and finally visit the root.



A postorder traversal of the tree in figure 5 is: 1, 3, 2, 6, 5, 8, 7, 4.

Tree Traversal
Converts hierarchical data into a linear sequence
Preorder: root, left, right
Inorder: left, root, right
Postorder: left, right, root

Ex: Show the preorder, inorder, postorder traversal for the tree shown below.



Sol:

Preorder: a b e j k n o p f c d g l m h i.
Inorder: j e n k o p b f a c l g m d h i.
Postorder: j n o p k e f b c l m g h I d a.

Ex: Show the preorder, inorder, postorder traversal for the tree shown below.

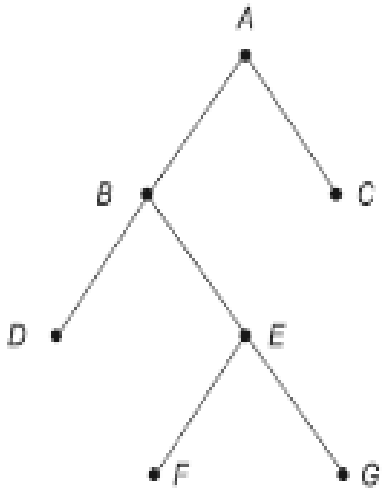


Fig. 13.8

Solution

(i) The inorder traversal of the given binary tree is DBFEGAC.
(ii) The preorder traversal of the given binary tree is ABDEFGC
(iii) The postorder traversal of the given binary tree is DFGEBCA.

Homework

For the following tree, Answer the following:
1. Show the preorder, inorder, postorder traversal.
2. What is the root?
3. What is the parent of m?
4. What are the children of j?
5. What are the siblings of h?
6. What are the ancestors of n?
7. What are the descendants of d?
8. What are the leaves?
9. What is the level of I vertex?
10. What is the height of the tree?
11. What is the type of tree?
12. Is the tree is ordered tree or not? Justify your answer.