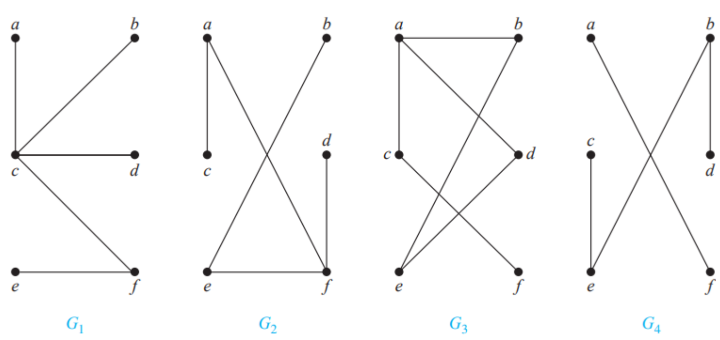
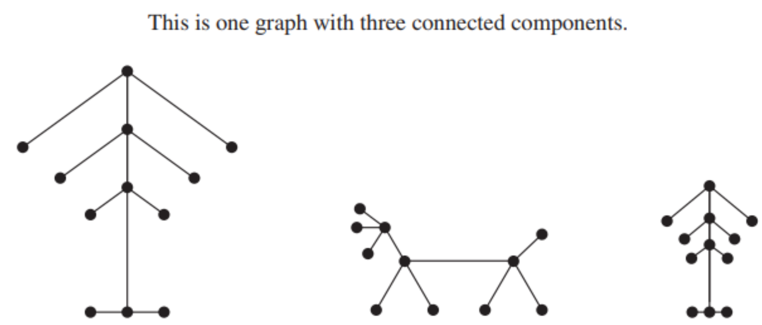
Ch11.1 Introduction to Trees (Week 16)



G_1 & G_2 are trees, G_3 is not (circuit), G_4 is not (connected)



Example of a Forest

- **Forest**: graphs with no simple circuit, not necessarily connected

> **1** A *tree* is a connected undirected graph with no simple circuits.

- No multi-edges, no loops → simple graph

> **1** **Theorem 1** An undirected graph is a tree *iff* there is a unique simple path between any two of its vertices.

Rooted Trees

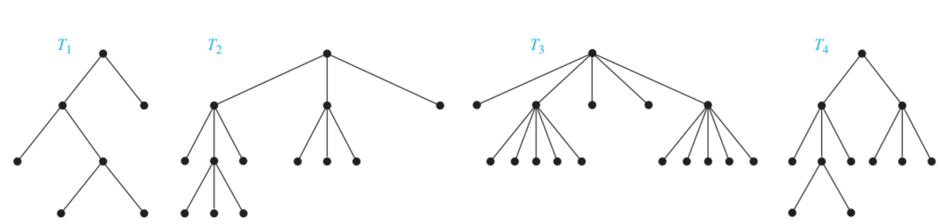
> **2** A *rooted tree* is a tree in which one vertex has been designated as the root and every edge is directed away fron the root.

- **parent:** a unique vertex that there is a directed edge to it
- **child:** the opposite of parent
- **siblings:** vertices with the same parent
- **ancestors:** vertices from root to itself

  (exclude itself, include root)
- **descendants:** vertiecs that have it as an ancestor
- **leaf:** vertex that has no children
- **internal vertices:** vertices that have children
- **subtree:** a vertex in the tree as the root, consisting all its descendants

  [The root is an internal vertex, unless it's the only vertex in the graph (leaf)]

> **3** 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. An m*-ary* tree with $m = 2$ is called a *binary tree*.



T_1 : full binary T_2 : full 3-ary T_3 : full 5-ary T_4 : Not full m-ary

Ordered Rooted Trees

→ the children of each internal vertex are ordered

(binary)

- **left child (subtree):** first child (subtree)
- **right child (subtree):** second child (subtree)

Tree as Models

→ Representing Organizations, Computer File Systems, ...

Properties of Trees

2 **Theorem 2** A tree with n vertices has $n-1$ edges.

3 **Theorem 3** A full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.

4 **Theorem 4** A full m-ary tree with

(i) n vertives has $i = (n-1)/m$ internal vertices and $l = [(m-1)n+1]/m$ leaves,

(ii) i internal vertices has $n = mi + l$ vertices and $l = (m-1)i + 1$ leaves,

(iii) l leaves has $n = (ml-1)/(m-1)$ vertices and $i = (l-1)/(m-1)$ internal vertices.

▼ Example 9 (How to implement Theorem 4)

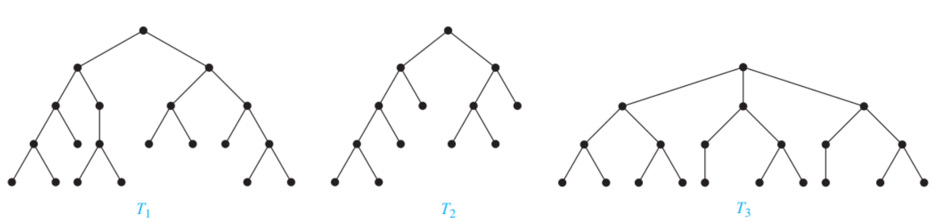
Suppose someone starts a chain letter. Each person who receives the letter is asked to send it on to four other peple. Some people do this, but others don't. How many people have seen the letter, including the first person, if no one receives more than one letter and if the chain letter ends after 100 people who read it but didn't sent it out? How many people sent out the letter?

Solution: The chain letter can be represented using a 4-ary tree. The internal vertices correspond to people who sent out the letter, and the leaves correspond to people who did not send it out. Because 100people didn't sent out the letter, the number of leaves in this rooted tree is $l = 100$. Hence, part (iii) of Theorem 4 shows that the number of people who have seen the letter is $n = (4 \cdot 100 - 1)/(4-1) = 133$. Also, the number of internal vertices is $133 - 100 = 33$, so 33 people sent out the letter.

Balanced *m*-ary Trees

→ a rooted m-ary tree of height h is **balanced** if all leaves are at level h or $h-1$.

- **level:** length of the unique path from root to itself
- **height:** length of longest path from root to any vertex



T_1 : balanced T_2 : not balanced T_3 : balanced

5 **Theorem 5** There are at most m^h leaves in an m-ary tree of height h.

1 **Corollary 1** If any m-ary tree of height h has l leaves, then $h \geq \lceil log_m l \rceil$. If the m-ary tree is full and balanced, then $h = \lceil log_m l \rceil$.