CSCI 2824: Discrete Structures

Lecture 38: Basic Trees and Structural Induction

Rachel Cox

Department of Computer Science

Section 001 (9am w/ Rachel Cox):
Location: FLMG 155 (our normal classroom)
Time: Wednesday Dec. 18th, 1:30-4:00 PM

Section 002 (11am w/ Ioana Fleming):
Location: HUMN 1B50 (our normal classroom)
Time: Sunday Dec. 15th, 1:30-4:00 PM

You are permitted a calculator and one 8.5x11 inch page of handwritten notes (both sides).

Check out this week on Moodle for the following study resources:
-Final Exam Concept Guide
-All Moodle HW problem set
-All Moodle Quiz problem set
-HW13 (not collected) and solutions

TABLE 6 Logical Equivalences.

Equivalence	*Name*
$p \wedge \mathbf{T} \equiv p$ $p \vee \mathbf{F} \equiv p$	Identity laws
$p \vee \mathbf{T} \equiv \mathbf{T}$ $p \wedge \mathbf{F} \equiv \mathbf{F}$	Domination laws
$p \vee p \equiv p$ $p \wedge p \equiv p$	Idempotent laws
$\neg(\neg p) \equiv p$	Double negation law
$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Commutative laws
$(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan's laws
$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Absorption laws
$p \vee \neg p \equiv \mathbf{T}$ $p \wedge \neg p \equiv \mathbf{F}$	Negation laws

TABLE 1 Rules of Inference.

Rule of Inference	*Tautology*	*Name*
p $p \to q$ $\therefore q$	$(p \wedge (p \to q)) \to q$	Modus ponens
$\neg q$ $p \to q$ $\therefore \neg p$	$(\neg q \wedge (p \to q)) \to \neg p$	Modus tollens
$p \to q$ $q \to r$ $\therefore p \to r$	$((p \to q) \wedge (q \to r)) \to (p \to r)$	Hypothetical syllogism
$p \vee q$ $\neg p$ $\therefore q$	$((p \vee q) \wedge \neg p) \to q$	Disjunctive syllogism
p $\therefore p \vee q$	$p \to (p \vee q)$	Addition
$p \wedge q$ $\therefore p$	$(p \wedge q) \to p$	Simplification
p q $\therefore p \wedge q$	$((p) \wedge (q)) \to (p \wedge q)$	Conjunction
$p \vee q$ $\neg p \vee r$ $\therefore q \vee r$	$((p \vee q) \wedge (\neg p \vee r)) \to (q \vee r)$	Resolution

$p \to q \equiv \neg p \vee q$	Relation by Implication (RBI)
$p \to q \equiv \neg q \to \neg p$	Contraposition
$p \Leftrightarrow q \equiv (p \to q) \wedge (q \to p)$	Definition of Biconditional
$p \oplus q \equiv (p \vee q) \wedge \neg(p \wedge q)$	Alternate Definition of xor

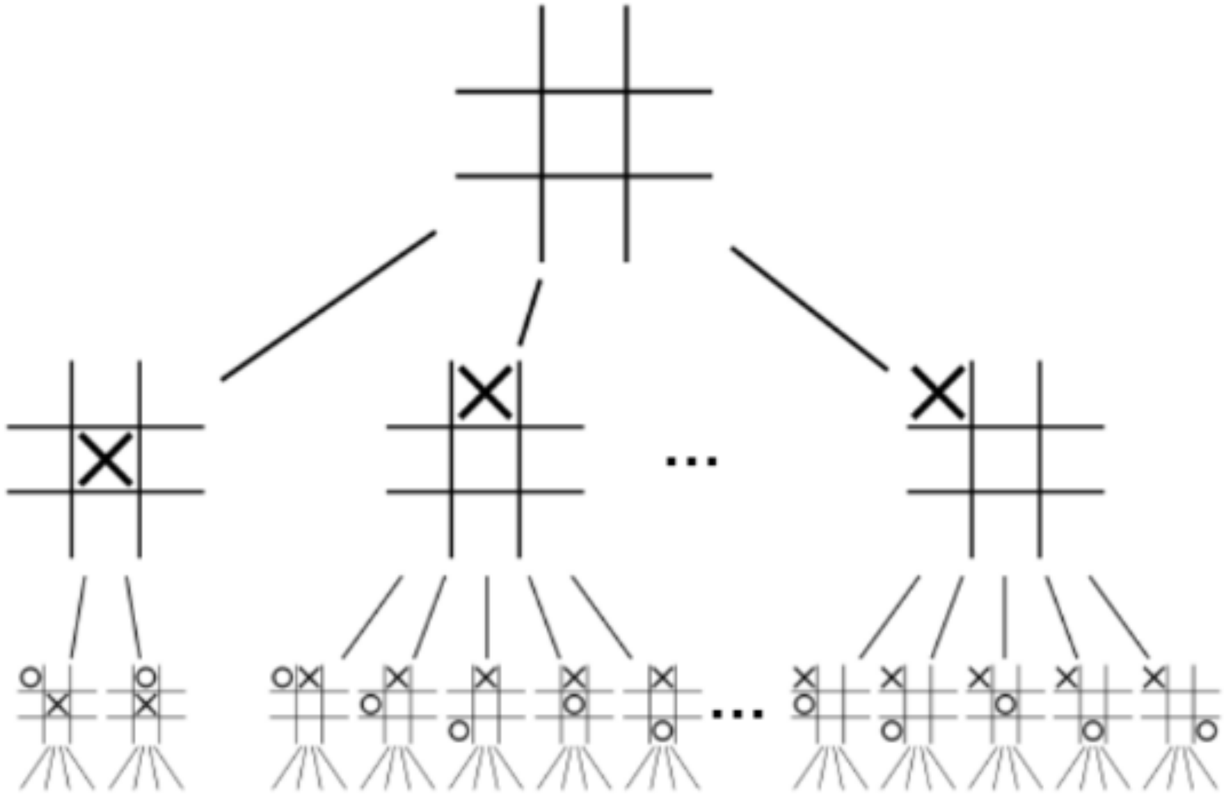
Permutations with repetition	n^r	Permutations without repetition	$\frac{n!}{(n-r)!}$
Combinations with repetition	$\frac{(r+n-1)!}{r!\,(n-1)!}$	Combinations without repetition	$\frac{n!}{(n-r)!\,r!}$
Binomial theorem	$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k}\, y^k$	Pigeonhole Principle	$\left\lceil \frac{N}{k} \right\rceil$
Bayes' thoerem	$p(A \mid B) = \frac{p(B \mid A)p(A)}{p(B)}$	Law of total probability	$p(E) = \sum_{i=1}^{N} p(E \mid F_i)p(F_i)$
Cardinality of union of sets	$\|A \cup B\| = \|A\| + \|B\| - \|A \cap B\|$	Conditional probability	$p(A \mid B) = \frac{p(A \cap B)}{p(B)}$

TABLE 1 Set Identities.

Identity	*Name*
$A \cap U = A$ $A \cup \emptyset = A$	Identity laws
$A \cup U = U$ $A \cap \emptyset = \emptyset$	Domination laws
$A \cup A = A$ $A \cap A = A$	Idempotent laws
$\overline{(\overline{A})} = A$	Complementation law
$A \cup B = B \cup A$ $A \cap B = B \cap A$	Commutative laws
$A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$	Associative laws
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$	Distributive laws
$\overline{A \cap B} = \overline{A} \cup \overline{B}$ $\overline{A \cup B} = \overline{A} \cap \overline{B}$	De Morgan's laws
$A \cup (A \cap B) = A$ $A \cap (A \cup B) = A$	Absorption laws
$A \cup \overline{A} = U$ $A \cap \overline{A} = \emptyset$	Complement laws

Trees

- Game trees in AI (tic-tac-toe, chess, go, …)
- Decision trees in Machine Learning
- Spanning trees in network routing
- Parse trees for compilers
- Organization of real-world data (family tree)
- Data Structures – rooted trees and binary search trees

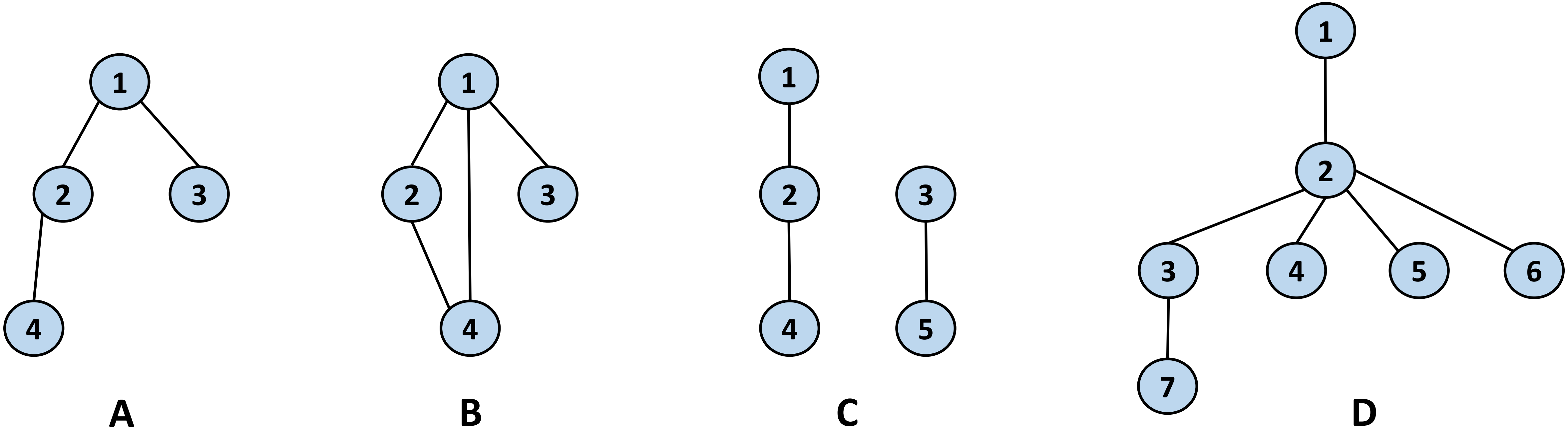


Trees

An (unrooted) ***tree*** T is an undirected graph $T = (V, E)$, such that

- T is fully connected.
- T is acyclic (no cycles).

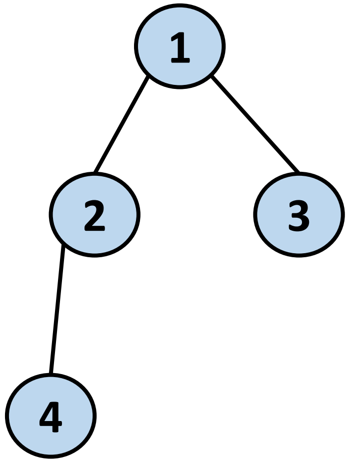
<u>**Example**</u>: Which of these graphs are trees? Why/why not?

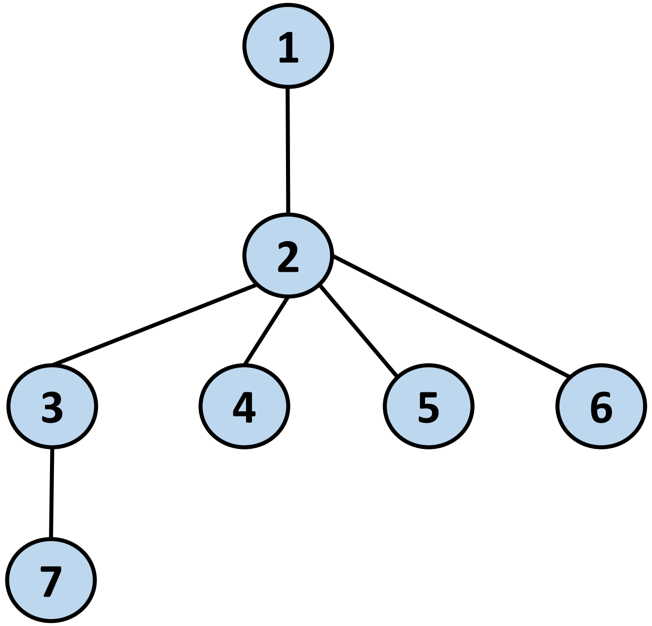


Trees

A ***leaf*** of an unrooted tree is a vertex with degree 1.

Example: What are the leaves of each of the trees below?

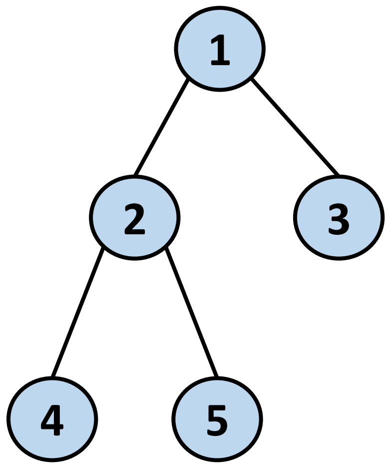




Trees

A ***rooted tree*** T can be thought of as a directed graph.

- T is fully connected and acyclic (no cycles).
- The sole root v has no incoming edges.
- Leaves have no outgoing edges.



If v is any vertex besides the root, then the ***parent*** of v is the unique vertex u such that there is a directed edge from u to v.

When u is the parent of v, we say that v is the ***child*** of u.

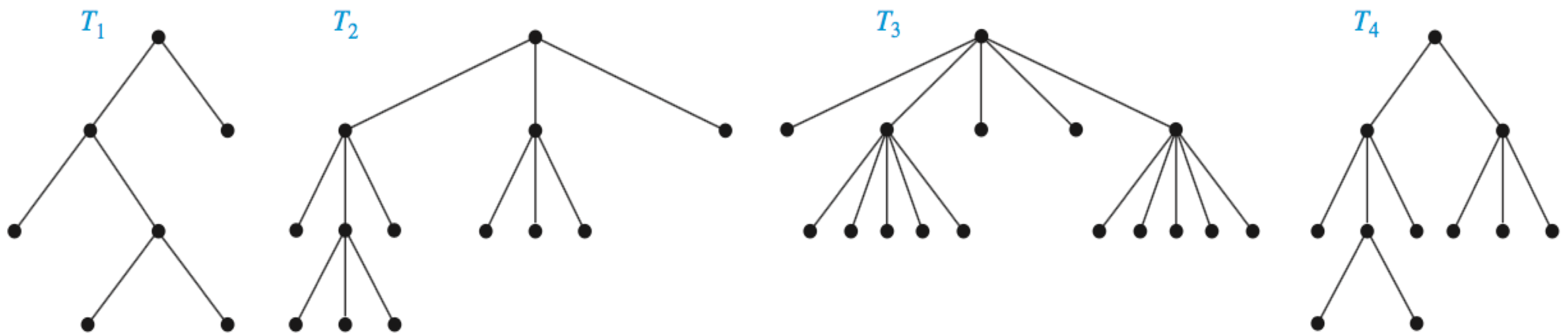
If vertices v_1 and v_2 have the same parent, we say that v_1 and v_2 are ***siblings***.

Vertices with children are called ***internal*** vertices.

Trees

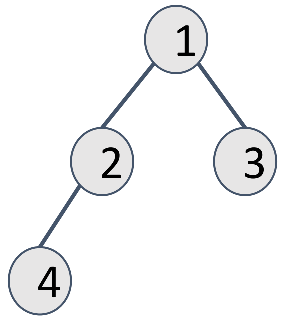
A rooted tree is called an ***m*-ary tree** if every internal vertex has no more than m children. An m-ary tree is called a **full *m*-ary tree** if every internal vertex has exactly m children.

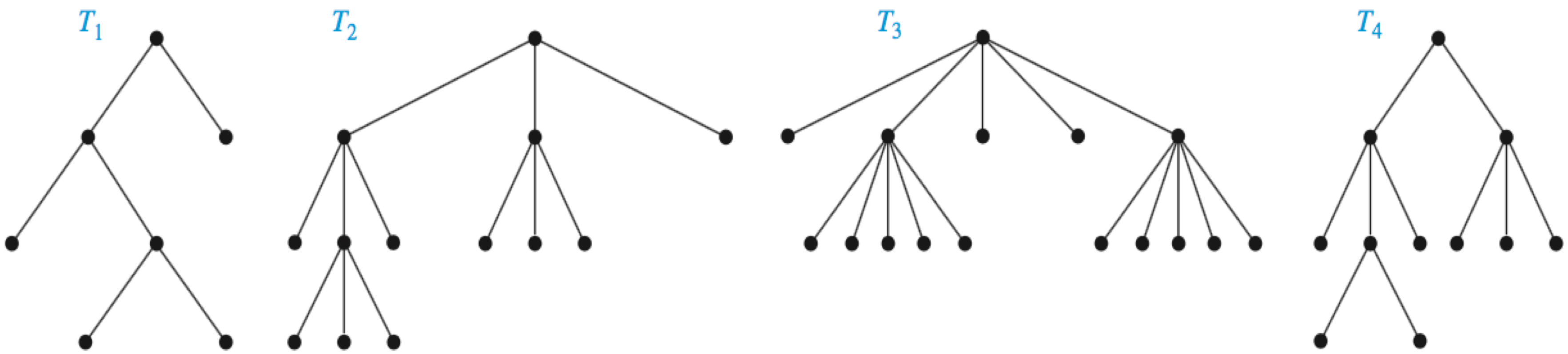
Example: Which of the following are full *m*-ary trees?

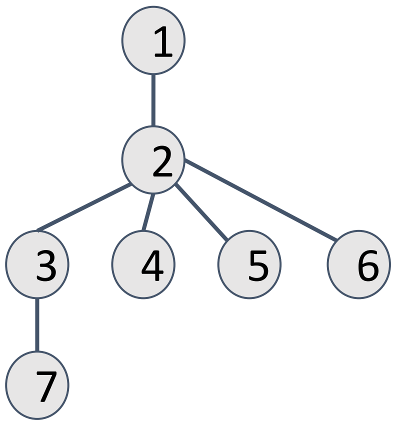


Induction and Trees

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







Induction and Trees

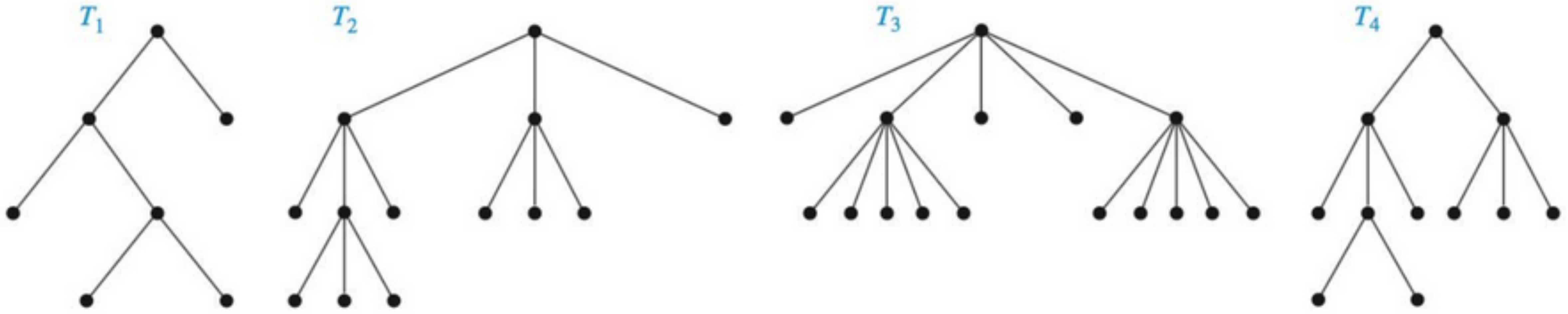
Example: Use induction to prove the following theorem: A tree with n vertices has $n - 1$ edges.

Induction and Trees

Let $n(T)$ denote the number of vertices in tree T.

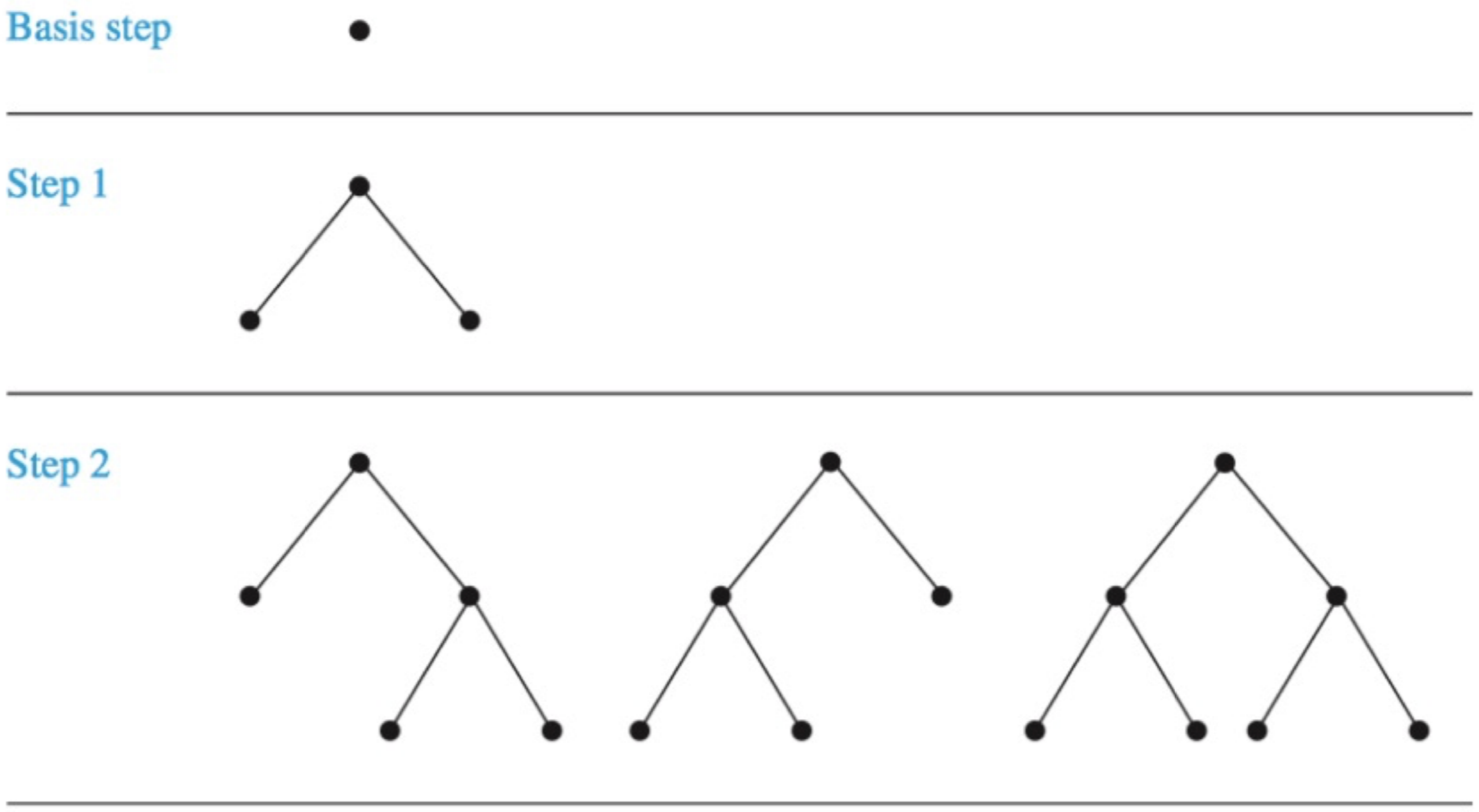
Let $h(T)$ denote the ***height*** of a rooted tree T. The height of a tree is defined to be the length of the longest path from the root to a leaf.

<u>**Example**</u>: What is the height of a tree T consisting of a single node?
What are the respective heights of the trees below?



Induction and Trees

Example Create the set of all full binary trees by recursion



Induction and Trees

Def: The number of vertices in a full binary tree T, $n(T)$, can be defined recursively as

Basis Step: If T is a FBT made up just a root then $n(T) = 1$

Recursive Step: If T_1 and T_2 are FBTs and $T = T_1 \cdot T_2$ then $n(T) = 1 + n(T_1) + n(T_2)$.

Def: The height of a a full binary tree T, $h(T)$, can be defined recursively as

Basis Step: If T is a FBT made up just a root then $h(T) = 0$

Recursive Step: If T_1 and T_2 are FBTs and $T = T_1 \cdot T_2$ then $h(T) = 1 + \max(h(T_1), n(T_2))$.

Structural Induction

When we use induction (strong or weak), we prove that a proposition P, indexed by a natural number n, is true for all values of n.

When we want to prove things about a structure that is defined recursively, we use a version of induction called ***structural induction***.

1. Base step: Show that the result (P) holds for all elements defined in the base step of the recursive definition for the structure.

2. Recursive step: Show that if the result holds for all elements used in the recursive step to construct new elements, then the result must hold for each new element.

Structural Induction

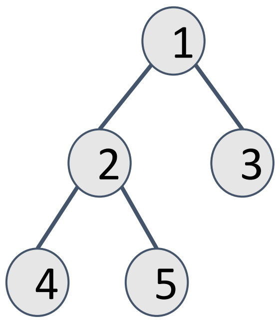
Example: If T is a full binary tree (FBT), then T has an odd number of vertices.

Structural Induction

Example: If T is a full binary tree (FBT), then $l(T) = i(T) + 1$

Trees

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



Proof:

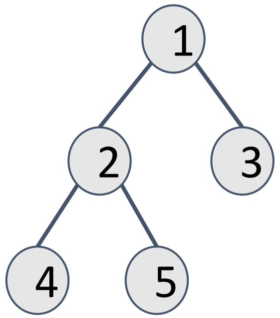
Trees

Theorem: A full m-ary tree with . . .

i. n vertices has $i = \frac{n-1}{m}$ internal vertices and $l = \frac{(m-1)n+1}{m}$ leaves,

ii. i internal vertices has $n = mi + 1$ 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.



Proof of (i):

Trees

Example: Suppose that someone starts a chain letter. Each person who receives the chain letter is asked to send it out to 4 other people. Some people send it out and some do not. How many people have seen the letter (including the first person) if no one receives the letter twice and the letter stops after 100 people receive it but do not send it out?


That's all Folks!