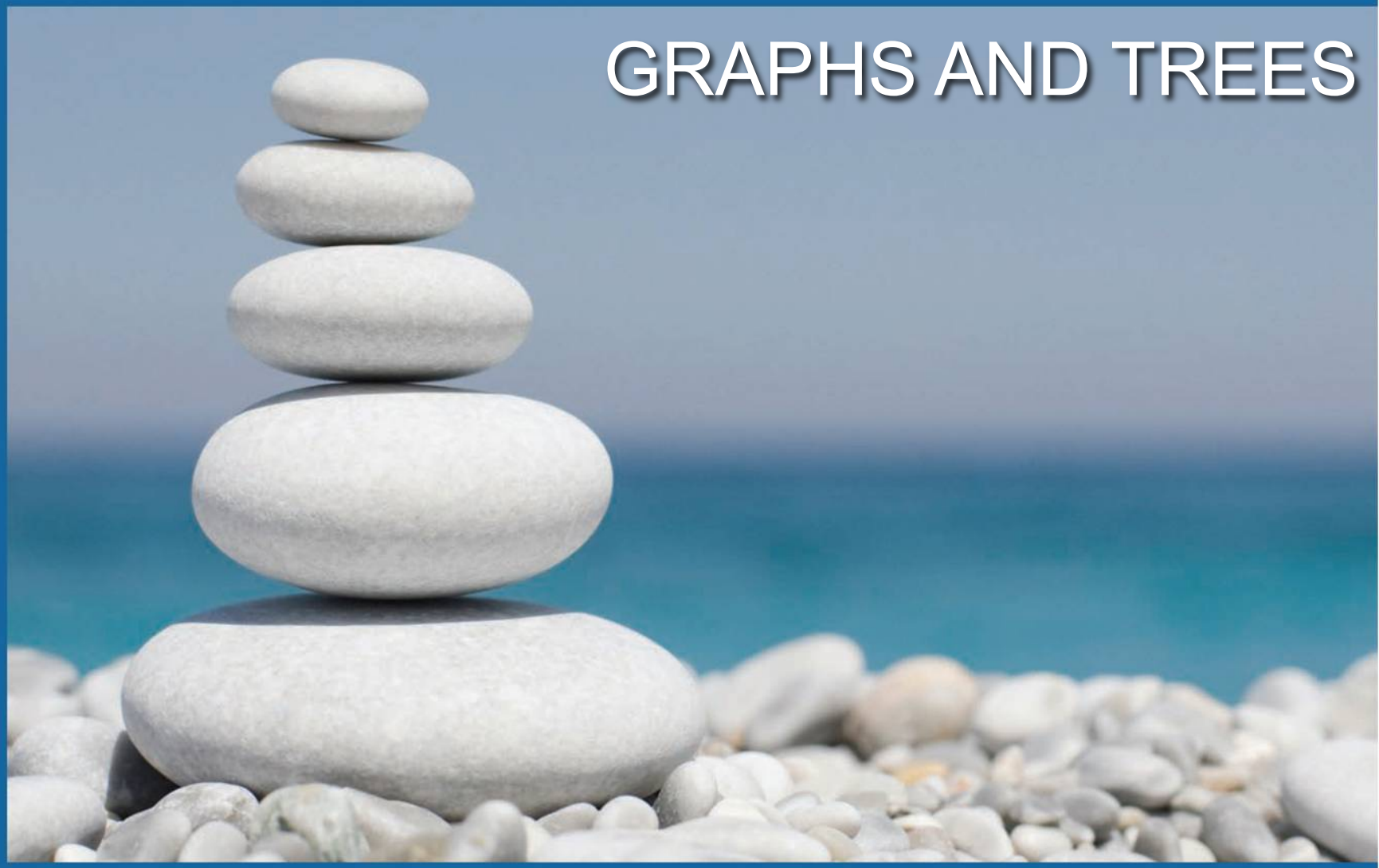
CHAPTER 10

GRAPHS AND TREES



Alessandro Artale – UniBZ - http://www.inf.unibz.it/~artale/

SECTION 10.5

Trees

Trees

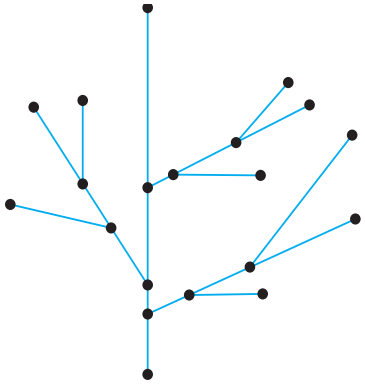
In mathematics, a **tree** is a connected graph that does not contain any circuits.

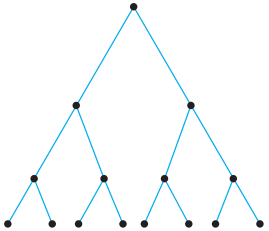
• Definition

A graph is said to be **circuit-free** if, and only if, it has no circuits. A graph is called a **tree** if, and only if, it is circuit-free and connected. A **trivial tree** is a graph that consists of a single vertex. A graph is called a **forest** if, and only if, it is circuit-free and not connected.

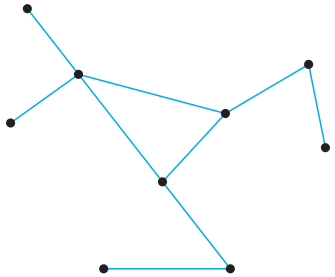
Example of Trees

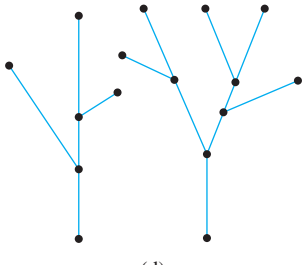
Example of Trees





The following are not trees (the last is a forest):





Example 3 – *A Parse Tree*

In the last 30 years, Noam Chomsky and others have developed grammars to describe the syntax of natural languages such as English.

This work has proved useful in constructing compilers for high level computer languages.

In the study of grammars, trees are often used to show the derivation of grammatically correct sentences from certain basic rules. Such trees are called **derivation trees** or **parse trees**.

Example 3 – *A Parse Tree*

cont' d

The rules of a grammar are called **productions**. It is customary to express them using the shorthand notation illustrated below.

$$\langle \text{sentence} \rangle \rightarrow \langle \text{noun phrase} \rangle \langle \text{verb phrase} \rangle$$

$$\langle \text{noun phrase} \rangle \rightarrow \langle \text{article} \rangle \langle \text{noun} \rangle \mid \langle \text{article} \rangle \langle \text{adjective} \rangle \langle \text{noun} \rangle$$

$$\langle \text{verb phrase} \rangle \rightarrow \langle \text{verb} \rangle \langle \text{noun phrase} \rangle$$

$$\langle \text{article} \rangle \rightarrow \text{the}$$

$$\langle \text{adjective} \rangle \rightarrow \text{young}$$

$$\langle \text{noun} \rangle \rightarrow \text{man} \mid \text{ball}$$

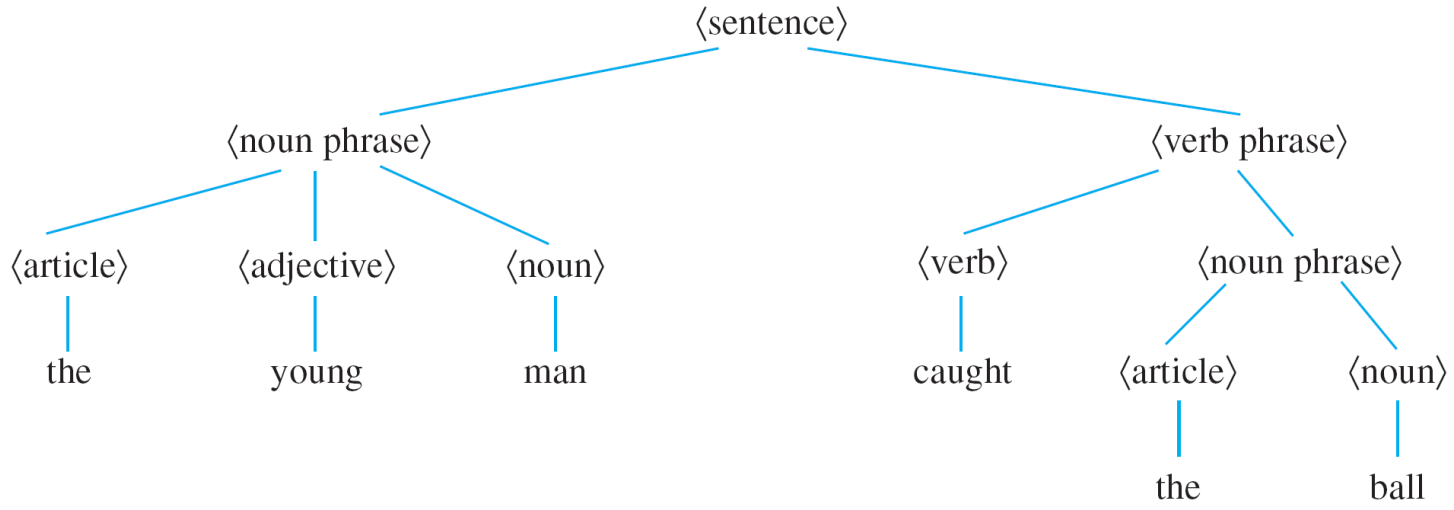
$$\langle \text{verb} \rangle \rightarrow \text{caught}$$

Example 3 – *A Parse Tree*

cont' d

In the notation, the symbol | represents the word *or*, and angle brackets ⟨ ⟩ are used to enclose terms to be defined (such as a sentence or noun phrase).

The derivation of the sentence "The young man caught the ball" from the mentioned rules is described by the tree shown below.



Characterizing Trees

Characterizing Trees

There is a somewhat surprising relation between the number of vertices and the number of edges of a tree. It turns out that if *n* is a positive integer, then any tree with *n* vertices (no matter what its shape) has *n* – 1 edges.

Perhaps even more surprisingly, a partial converse to this fact is also true—namely, any *connected* graph with *n* vertices and *n* – 1 edges is a tree.

It follows from these facts that if even one new edge (but no new vertex) is added to a tree, the resulting graph must contain a circuit.

Characterizing Trees

Also, from the fact that removing an edge from a circuit does not disconnect a graph, it can be shown that every connected graph has a subgraph that is a tree.

It follows that (if n is a positive integer) any graph with n vertices and *fewer* than $n - 1$ edges is not connected.

Characterizing Trees

An important fact necessary to derive the first main theorem about trees is that any nontrivial tree must have at least one vertex of degree 1.

Lemma 10.5.1

Any tree that has more than one vertex has at least one vertex of degree 1.

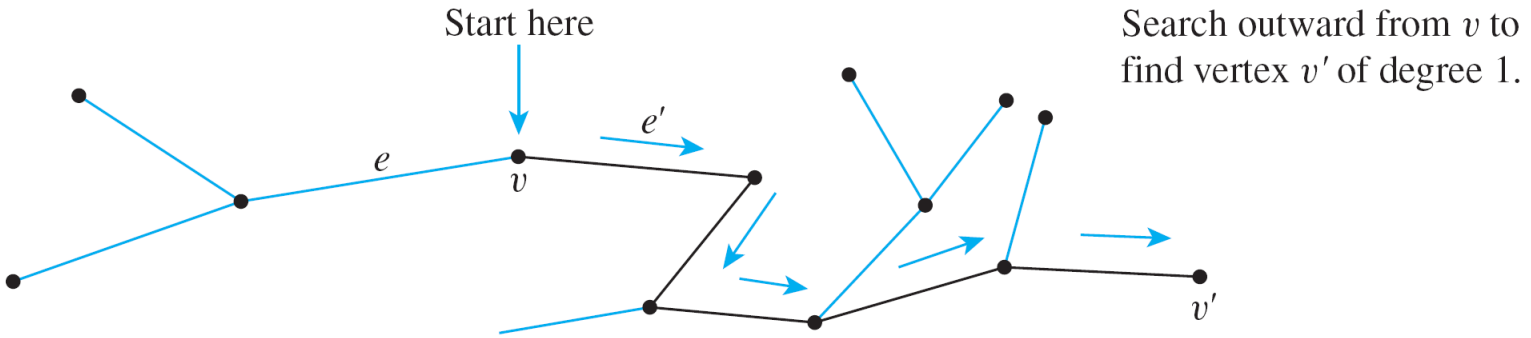
A constructive way to understand this lemma is to imagine being given a tree *T* with more than one vertex. You pick a vertex *v* at random and then search outward along a path from *v* looking for a vertex of degree 1.

Characterizing Trees

As you reach each new vertex, you check whether it has degree 1. If it does, you are finished. If it does not, you exit from the vertex along a different edge from the one you entered on.

Because T is circuit-free, the vertices included in the path never repeat. And since the number of vertices of T is finite, the process of building a path must eventually terminate.

When that happens, the final vertex v' of the path must have degree 1. This process is illustrated below.



Characterizing Trees

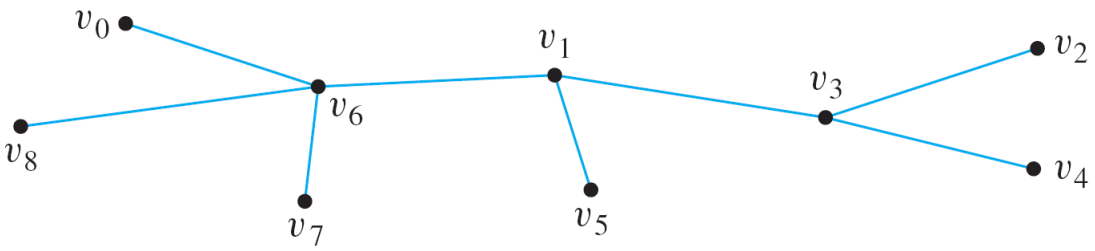
Using Lemma 10.5.1 it is not difficult to show that, in fact, any tree that has more than one vertex has at least *two* vertices of degree 1.

• Definition

Let T be a tree. If T has only one or two vertices, then each is called a **terminal vertex.** If T has at least three vertices, then a vertex of degree 1 in T is called a **terminal vertex** (or a **leaf**), and a vertex of degree greater than 1 in T is called an **internal vertex** (or a **branch vertex**).

Example 5 – *Terminal and Internal Vertices*

Find all terminal vertices and all internal vertices in the following tree:



Solution:

The terminal vertices are v_0, v_2, v_4, v_5, v_7, and v_8.

The internal vertices are v_6, v_1, and v_3.

Characterizing Trees

The following is the **first** of the two main theorems about trees:

Theorem 10.5.2

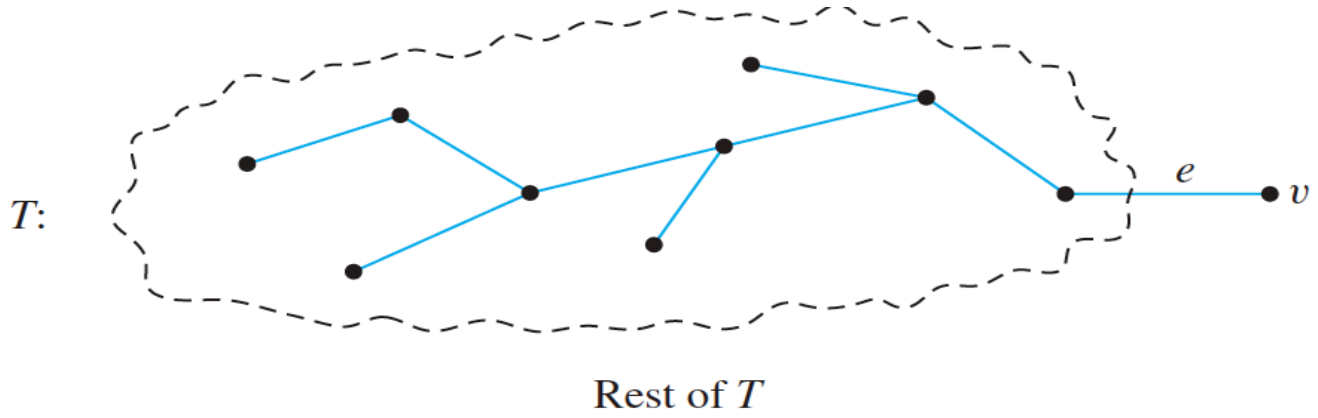
For any positive integer n, any tree with n vertices has $n - 1$ edges.

Proof by Induction.

Basis case: For n=1 the property holds since there are no loops.

Inductive Step: Let T be an arbitrarily chosen tree with k + 1 vertices, for k ≥ 1. Since k+1>1, by Lemma 10.5.1, T has a vertex *v* of degree 1. Since T is connected, *v* is attached to the rest of T by a single edge *e* as sketched in the following figure.

Theorem 10.5.2



Now if *e* and *v* are removed from T, what remains is a tree T' with (k + 1) − 1 = k vertices and, by inductive hypothesis, T' has k − 1 edges. But the original tree T has one more vertex and one more edge than T'. Hence T must have (k − 1) + 1 = k edges, as was to be shown.

Example 7 – *Finding Trees Satisfying Given Conditions*

Find all non-isomorphic trees with four vertices.

Solution:

- By Theorem 10.5.2, any tree with 4 vertices has 3 edges;
- Thus, by the Handshake Theorem (10.1.1), the total degree of a tree with four vertices must be 6;
- Also, every tree with more than one vertex has at least two vertices of degree 1.

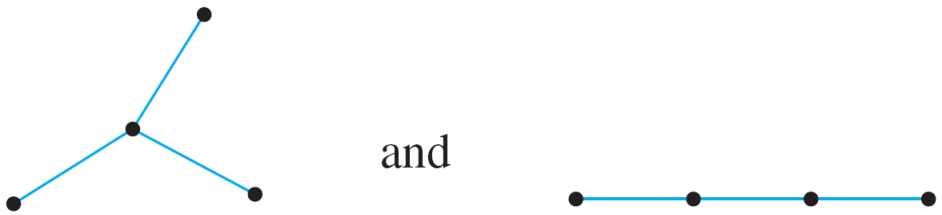
Example 7 – *Solution*

cont' d

Thus the following combinations of degrees for the vertices are the only ones possible:

1, 1, 1, 3 and 1, 1, 2, 2.

There are two non-isomorphic trees corresponding to both of these possibilities, as shown below.



Characterizing Trees

Before proving the second Theorem on Trees let's consider the following lemma.

Lemma 10.5.3

If G is any connected graph, C is any circuit in G, and any one of the edges of C is removed from G, then the graph that remains is connected.

Essentially, the reason why Lemma 10.5.3 is true is that any two vertices in a circuit are connected by two distinct paths.

It is possible to draw the graph so that one of these goes "clockwise" and the other goes "counterclockwise" around the circuit.

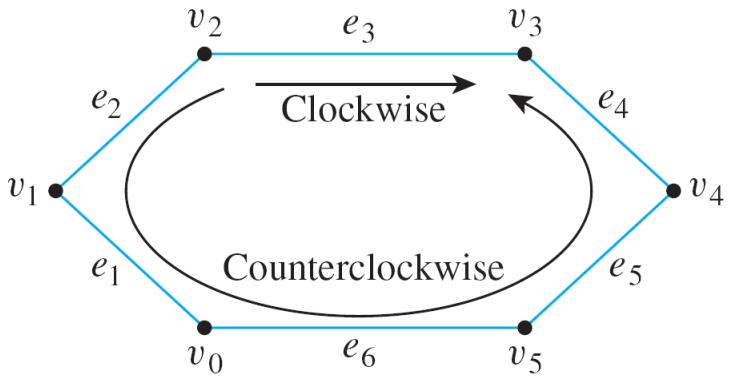
Characterizing Trees

For example, in the circuit shown below, the clockwise path from v_2 to v_3 is

$$v_2 e_3 v_3$$

and the counterclockwise path from v_2 to v_3 is

$$v_2 e_2 v_1 e_1 v_0 e_6 v_5 e_5 v_4 e_4 v_3 .$$



Characterizing Trees

The **second** major theorem about trees is a modified converse to Theorem 10.5.2.

Theorem 10.5.2

For any positive integer n, any tree with n vertices has $n - 1$ edges.

Theorem 10.5.4

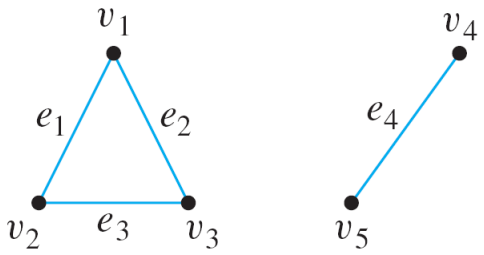
For any positive integer n, if G is a connected graph with n vertices and $n - 1$ edges, then G is a tree.

Proof by Contradiction. Since G is connected, it is enough to show that G is circuit-free. If, **by absurd**, it is not, then we can remove edges from each circuit obtaining a new graph G' which, by Lemma 10.5.3, is still connected. At the end of this process, we obtain a new connected and circuit-free graph, G", with *n* vertices but no more that n-2 edges.

Example 8 – *A Graph with n Vertices and n – 1 Edges That Is Not a Tree*

Theorem 10.5.4 is not a full converse of Theorem 10.5.2. Although it is true that every *connected* graph with n vertices and $n - 1$ edges ($n>0$) is a tree, it is not true that *every* graph with n vertices and $n - 1$ edges is a tree.

The following is an example of a graph with five vertices and four edges that is not a tree. By Theorem 10.5.4, such a graph cannot be connected.



SECTION 10.6

Rooted Trees

Rooted Trees

An outdoor tree is rooted and so is the kind of family tree that shows all the descendants of one particular person. The terminology and notation of rooted trees blends the language of botanical trees and that of family trees.

In mathematics, a **rooted tree** is a tree in which one vertex has been distinguished from the others and is designated the ***root***.

Given any other vertex *v* in the tree, there is a unique path from the root to *v*. (After all, if there were two distinct paths, a circuit could be constructed.)

Rooted Trees

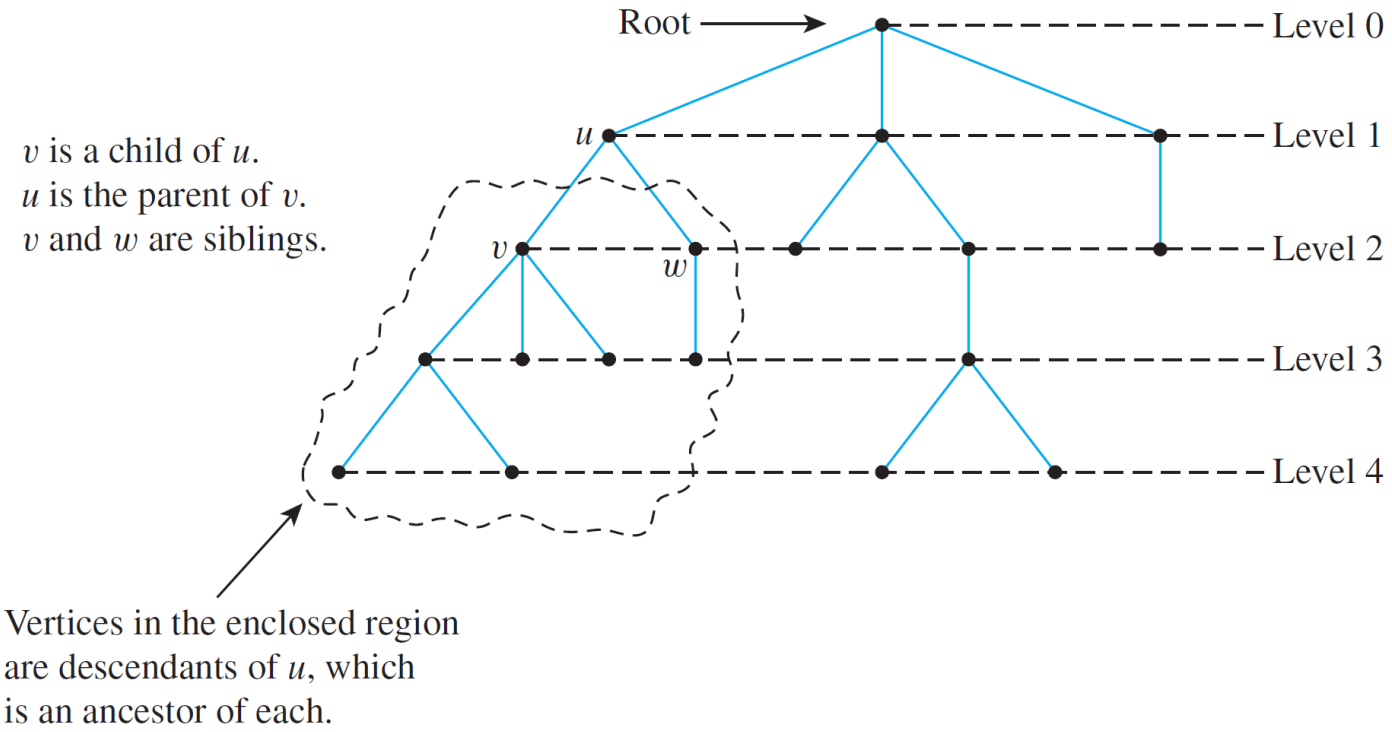
The number of edges in such a path is called the *level* of *v*, and the *height* of the tree is the length of the longest such path. It is traditional in drawing rooted trees to place the root at the top and show the branches descending from it.

• Definition

A **rooted tree** is a tree in which there is one vertex that is distinguished from the others and is called the **root.** The **level** of a vertex is the number of edges along the unique path between it and the root. The **height** of a rooted tree is the maximum level of any vertex of the tree. Given the root or any internal vertex v of a rooted tree, the **children** of v are all those vertices that are adjacent to v and are one level farther away from the root than v. If w is a child of v, then v is called the **parent** of w, and two distinct vertices that are both children of the same parent are called **siblings.** Given two distinct vertices v and w, if v lies on the unique path between w and the root, then v is an **ancestor** of w and w is a **descendant** of v.

Rooted Trees

These terms are illustrated in Figure 10.6.1.



A Rooted Tree

Figure 10.6.1

Example 1 – *Rooted Trees*

Consider the tree with root v_0 shown on the right.

a. What is the level of v_5?

b. What is the level of v_0?

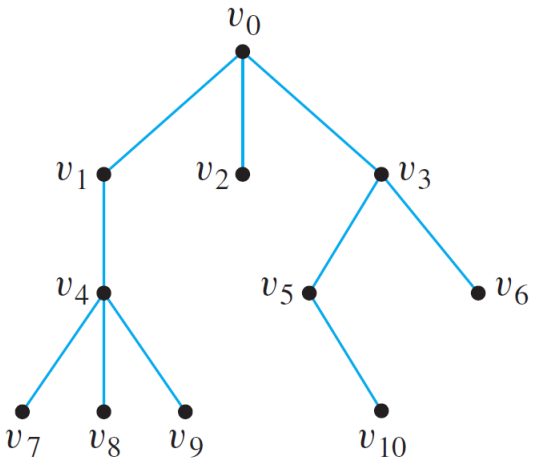
c. What is the height of this rooted tree?

d. What are the children of v_3?

e. What is the parent of v_2?

f. What are the siblings of v_8?

g. What are the descendants of v_3?



Example 1 – *Solution*

a. 2

b. 0

c. 3

d. v_5 and v_6

e. v_0

f. v_7 and v_9

g. v_5, v_6, v_{10}

Rooted Trees

Note that in the tree with root v_0 shown below, v_1 has level 1 and is the child of v_0, and both v_0 and v_1 are terminal vertices.

Binary Trees

Binary Trees

When every vertex in a rooted tree has at most two children and each child is designated either the (unique) left child or the (unique) right child, the result is a ***binary tree***.

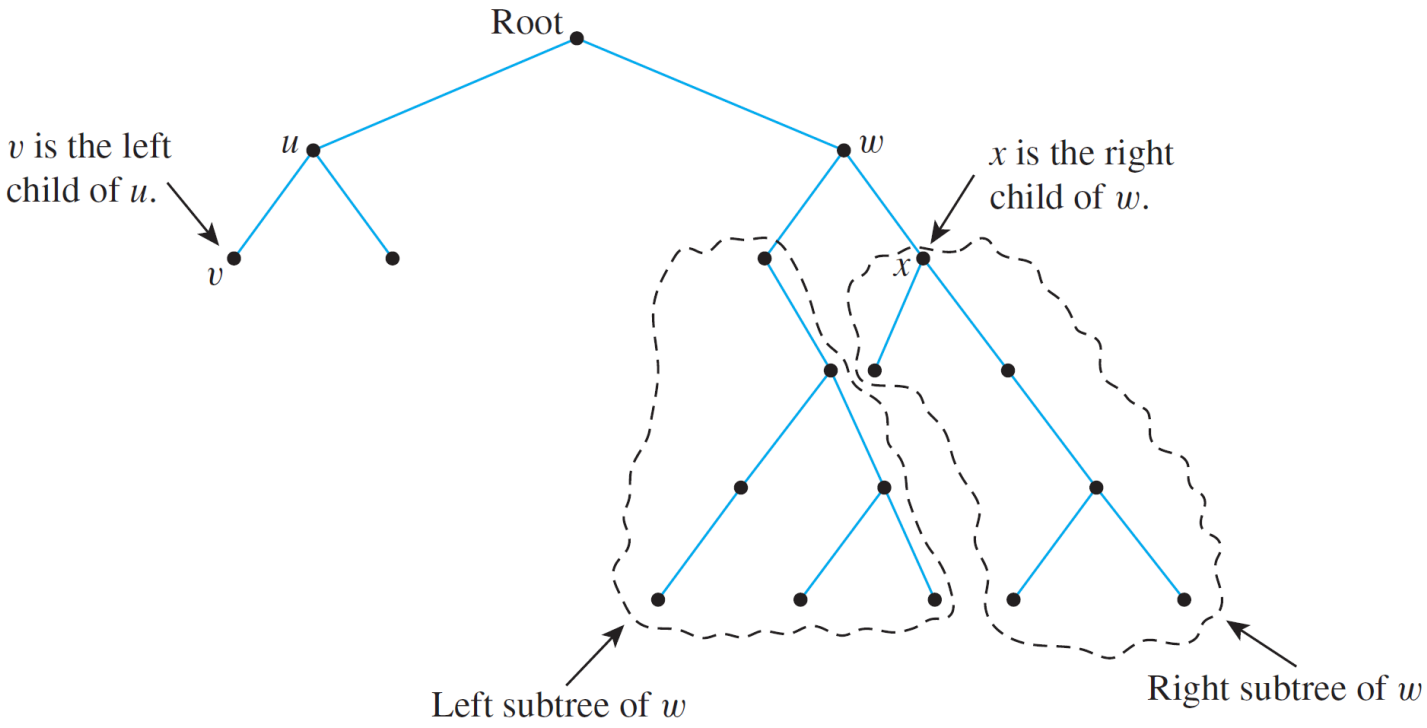
• Definition

A **binary tree** is a rooted tree in which every parent has at most two children. Each child in a binary tree is designated either a **left child** or a **right child** (but not both), and every parent has at most one left child and one right child. A **full binary tree** is a binary tree in which each parent has exactly two children.

Given any parent v in a binary tree T, if v has a left child, then the **left subtree** of v is the binary tree whose root is the left child of v, whose vertices consist of the left child of v and all its descendants, and whose edges consist of all those edges of T that connect the vertices of the left subtree. The **right subtree** of v is defined analogously.

Binary Trees

These terms are illustrated in Figure 10.6.2.

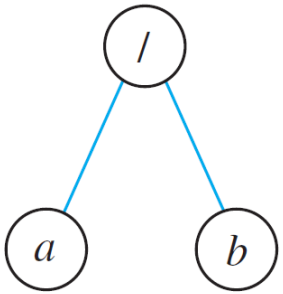


A Binary Tree

Figure 10.6.2

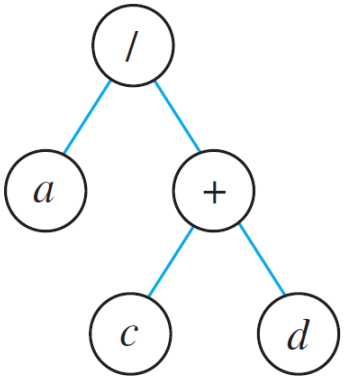
Example 2 – *Representation of Algebraic Expressions*

Binary trees are used in many ways in computer science. One use is to represent algebraic expressions with arbitrary nesting of balanced parentheses. For instance, the following (labeled) binary tree represents the expression *a*/*b*: The operator is at the root and acts on the left and right children of the root in left-right order.



Example 2 – *Representation of Algebraic Expressions* cont' d

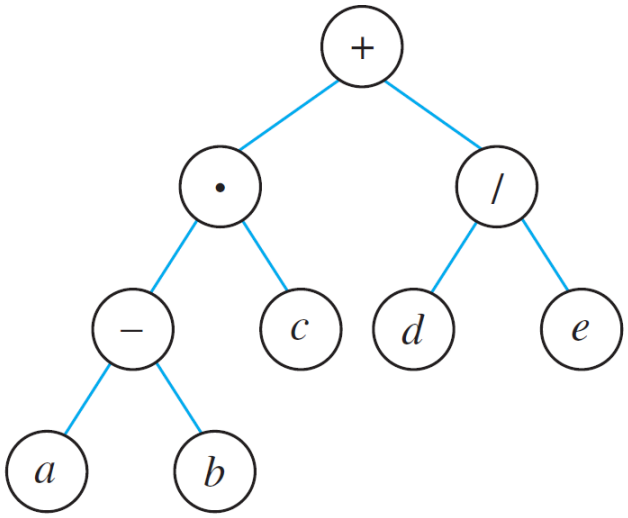
More generally, the binary tree shown below represents the expression $a/(c + d)$. In such a representation, the internal vertices are arithmetic operators, the terminal vertices are variables, and the operator at each vertex acts on its left and right subtrees in left-right order.



Draw a binary tree to represent the expression $((a - b) \cdot c) + (d/e)$.

Example 2 – *Solution*

The binary tree to represent the expression $((a - b) \cdot c) + (d/e)$, is as follows:



Characterizing Binary Trees

An interesting theorem about full binary trees says that if you know the number of internal vertices of a full binary tree, then you can calculate both the total number of vertices and the number of terminal vertices, and conversely. More specifically, a full binary tree with k internal vertices has a total of $2k + 1$ vertices of which $k + 1$ are terminal vertices.

Theorem 10.6.1

If k is a positive integer and T is a full binary tree with k internal vertices, then T has a total of $2k + 1$ vertices and has $k + 1$ terminal vertices.

Proof. See the book, pg.697.

Example 3 – *Determining Whether a Certain Full Binary Tree Exists*

Is there a full binary tree that has 10 internal vertices and 13 terminal vertices?

Solution:

No. By Theorem 10.6.1, a full binary tree with 10 internal vertices has 10 + 1 = 11 terminal vertices, not 13.

Characterizing Binary Trees

Another interesting theorem about binary trees specifies the maximum number of terminal vertices of a binary tree of a given height.

The maximum number of terminal vertices of a binary tree of height h is 2^h. Another way to say this is that a binary tree with t terminal vertices has height of at least $\log_2 t$.

Theorem 10.6.2

For all integers $h \geq 0$, if T is any binary tree with of height h and t terminal vertices, then

$$t \leq 2^h.$$

Equivalently, $\log_2 t \leq h$.

Proof by Induction (see the book, pg.698)

Example 4 – *Determining Whether a Certain Binary Tree Exists*

Is there a binary tree that has height 5 and 38 terminal vertices?

Solution:

No. By Theorem 10.6.2, any binary tree T with height 5 has at most $2^5 = 32$ terminal vertices, so such a tree cannot have 38 terminal vertices.