CHAPTER 5

Section 5.2

Trees

- A tree is a special type of graph.
- Definition
  - A tree is an acyclic, connected graph with one node designated as the **root** of the tree.
    - A tree is typically drawn with the root at the top.
  - Some texts call a tree with a root node a **rooted tree**.
  - Examples in Figure 5.31 of book (pg 433).

Trees: Recursive Definition

- A single node is a tree (with that node as the root).
- A tree is made up of subtrees, which are smaller trees.
  - If $T_1, T_2, \ldots, T_t$ are disjoint trees with roots $r_1, r_2, \ldots, r_t$, the graph formed by attaching a new node r by a single arc to each of $r_1, r_2, \ldots, r_t$ is a tree with root r.
- Any nodes $r_1, r_2, \ldots, r_t$, directly below node r, are the **children** of r.
  - r is the **parent** of $r_1, r_2, \ldots, r_t$

Tree Terminology

- **Depth of a node**: the length of the path from the root to the node.
  - The root has depth 0.
- **Depth (height) of the tree**: the maximum depth of any node in the tree.
  - The length of the longest path from the root to any node.
- **Leaf**: A node with no children.
- **Internal node**: a non-leaf.
- **Forest**: A disjoint collection of trees (an acyclic, but not necessarily connected) graph

Tree Terminology

- **Binary tree**: A tree in which each node has at most 2 children.
  - Each child of a node is either the **left child** or the **right child**.
- **Full binary tree**: all internal nodes have 2 children and all leaves are at the same depth.
- **Complete binary tree**: an almost-full binary tree
  - The bottom level may not have all its leaves
  - Any "missing" leaves are on the right of all the other leaves.

Practice 18

- Consider the binary tree shown in Figure 5.36, page 435. Given that node 1 is the root,
  - What is the height (or depth)?
  - What is the left child of node 2?
  - What is the depth of node 5?

Applications of Trees

- Decision Trees
- Binary search trees
- Compiler parse trees
- Organization chart (Figure 5.37)
- File structure

Example 22

- A computer virus is spread via e-mail. Each second, 4 new machines are infected.
  - A 4-ary tree structure (Figure 5.39) represents the spread of the virus.
  - By the multiplication principle, how many machines have been infected after n seconds?

Example 23

- Algebraic expressions involving binary operations can be represented by labeled binary trees.
  - Internal nodes are labeled as operators.
  - Leaves are labeled as operands.
  - Figure 5.40 (pg. 436) is $(2 + x) - (y * 3)$
- Practice 19: What is the expression tree for $(2 + 3) * 5$?

Binary Tree Representation

- Adjacency matrix equivalent
  - 2 column array, where the data for each node is the left child and right child of that node.
- Adjacency list equivalent
  - Collection of records with 3 fields each, the current node, a pointer to the record for the left child and a pointer to the record for the right child.
  - Class in object-oriented languages for a tree node. Then a tree consists of a root. From the root, one can get to the rest of the tree.

Binary Tree Representation

- Practice 20
  - Give the left child-right child array representation of the binary tree in Figure 5.43 (matrix)
  - Give the pointer representation of the binary tree in Figure 5.43 (list)

Tree Traversal

- Traversing a tree involves visiting each of its nodes
- Three common tree traversal algorithms
  - Preorder
  - Inorder
  - Postorder
- They each define a different order in which to visit the nodes, particularly when to visit the root node.

Tree Traversal

- **Preorder** Traversal
  - Visit the root of a tree.
  - Visit the subtrees left to right, using preorder traversal of each.
  - For a binary tree, this means visit the root, then do a preorder traversal of the left subtree, then a preorder traversal of the right subtree.
- Consider the tree in Figure 5.45.
  - List the order in which the nodes are visited if using **preorder** traversal.

Tree Traversal

- **Inorder** Traversal
  - Visit the left subtree, using inorder traversal.
  - Visit the root.
  - Visit the remaining subtree(s), left to right, using inorder traversal.
- Consider the tree in Figure 5.45.
  - List the order in which the nodes are visited if using **inorder** traversal.

Tree Traversal

- **Postorder** Traversal
  - Visit all the subtree(s), left to right, using postorder traversal.
  - Visit the root.
- Consider the tree in Figure 5.45.
  - List the order in which the nodes are visited if using **postorder** traversal.

Practice 21

- Do a preorder, inorder, and postorder traversal of the tree in Figure 5.47.

Example 28

- Consider the expression tree in Figure 5.48 (page 442).
  - Do an inorder traversal.
    - Corresponds to infix notation – the operation symbol appears between the two operations.
  - Do a preorder traversal.
    - Corresponds to prefix notation, or Polish notation, where the operation symbol appears before the operands.
  - Do a postorder traversal
    - Corresponds to postfix notation, or reverse Polish notation, where the operation symbol appears after the operands.

Tree Traversal

- What is an advantage of postfix and prefix over infix?
- Compilers often change algebraic expressions in computer programs from infix to postfix for more efficient processing.

Practice 22

- Write the expression tree for

  $$a + (b * c - d)$$

- Write the expression in prefix and postfix notation.

Section 5.2: Exercises

- Exercise 38: Prove that a binary tree has at most 2^d nodes at depth d.
- Exercise 39
  - Draw a full binary tree of height 2. How many nodes does it have?
  - Draw a full binary tree of height 3. How many nodes does it have?
  - Conjecture how many nodes there are in a full binary tree of height h.
  - Prove your conjecture. (Hint: Use Exercise 38).

Section 5.2: Exercises

- Exercise 42
  - Find an expression for the height of a complete binary tree with n nodes. (Hint: Use Exercise 39.)