Trees

CSE 122 ~ Algorithms & Data Structures

Trees ~ Terminology

- Trees ~ non-linear data structure
  - Nodes do not form a simple sequence
  - Each node points to a number of other nodes
- Parent
  - The node linked above it.
- Root
  - The node of the tree with no parent(s)
- Edge
  - Refers to the link from parent to child
- Leaf
  - A node with no children
- Siblings
  - Children nodes of the same parent node

Trees ~ Terminology

- ➔ Ancestor
  - A node *p* is an ancestor of a node *q* if there exists a path from the root to *q* and *p* appears on that path
- ➔ Subtree
  - Any node in a tree can be viewed as the root of a smaller tree, or subtree of the original
- ➔ Depth
  - Length of the path from the root to the node
- ➔ Height
  - Length of the path from that node to the deepest node
- ➔ Size
  - The number of descendents it has including itself
- ➔ Level
  - Set of all nodes at a given depth in the tree

Binary Tree

- A *binary tree* is a finite set of nodes. The set may be empty, but if it is not empty, it will obey the following rules:
  - There exists a root node
  - Each node can be associated with zero, one, or two different nodes (children)
  - Each node, except the root, has exactly one parent. The root has no parents
  - If you start at a node and move to the node's parent, and keep moving to that node's parent, and keep moving to that node's parent, you will eventually reach the root. It is **connected**.

Binary Tree ~ Cont.

- Subtrees
  - For a node in a binary tree, the nodes beginning with its:
    - left child and below are its left subtree
    - Right child and below are its right subtree
- Full Binary Tree
  - Every leaf has the same depth and every non-leaf has two children
  - I.e. Each node has exactly zero or two children.

Binary Tree ~ Cont.

- Complete Binary Tree
  - Suppose you take a full binary tree and start adding new leaves at a new depth from left to right
  - All new leaves have the same depth - one more than where we started - and we always add leftmost nodes first
  - The tree is no longer full, instead we call it a **complete binary tree**
  - To be complete:
    - Every level except the deepest must contain as many nodes as possible
    - All the nodes are as far left as possible
    - To label often start at 1 with the root and label from left to right.

Trees ~ Formal Definition

- A *tree* is a finite set T of one or more nodes such that
  - There is one specially designated node called the *root* of the tree
  - The remaining nodes (excluding the root) are partitioned into $m \geq 0$ disjoint sets T_1, T_2, …, ™ and each of these sets are in turn a tree (subtrees of the root)

Trees ~ Recursive Definition

- Trees of one node consist of only the root.
- Trees with n > 1 node are defined in terms of trees with fewer than n nodes
- A **binary tree** is a finite set of nodes that either is empty, or consists of a root and the elements of two disjoint binary trees called the left and right subtrees

Properties of Binary Trees

- Assume you have a tree with depth *n*
    - The number of nodes *n* in a full binary tree is $2^{h+1}-1$, where *h* is the height of the tree. Since *h* is the height, and adding all nodes $2^0+2^1+2^2+...+2^h = 2^{h+1}-1$
    - The number of nodes *k* in a complete binary tree is between (min) $2^n \leq k < 2^{n+1}-1$ (max)
    - The number of leaf nodes in a full binary tree are 2^n
    - The number of empty nodes in a complete binary tree with *k* nodes is $2^{n+1}-1-k$