Appendix B5 Trees

Similar to graphs, there are a variety of trees with their own properties and definitions.

B.5.1 Free Trees

Def: (Free) Tree: A connected, acyclic, undirected graph.
Note: Free is often dropped as an adjective when referring to a graph as a tree.
Def: Forest: If a tree is a disconnected, it is considered a forest.
Note: A forest can also be thought as a collection of trees.

Theorem: Properties of Free Trees

Let $G = (V, E)$ be an undirected graph. The following statements are equivalent,

1. G is a free tree.
2. Any two vertices in G are connected by a unique simple path.
3. G is connected, but if any edge is removed from E, the resulting graph is now disconnected.
4. G is connected so $|E| = |V| - 1$
5. G is acyclic so $|E| = |V| - 1$
6. G is acyclic, but if any edge is added to E, the resulting graph now contains a cycle.

Proof

> $(1) \implies (2)$
> Since a tree is connected, any two vertices in G are connected by at least one simple path.
> Suppose, by contradiction, that $u, v \in V$ by two distinct paths p_1, p_2.
> Let w be the vertex at which the paths diverge.
> Let z be the vertex at which the paths reconverge.
> Let p' be the subpath of p_1 from w through x to z.
> Let p'' be the subpath of p_2 from w through y to z.
> As p_1 and p_2 are distinct, p' and p'' share no vertices except their endpoints.
> Since p_1 and p_2 share two vertices, they form a cycle which contradicts our assumption that G is a tree.
> $\therefore$ If G is a tree, there are at most one simple path between two vertices.

> $(2) \implies (3)$
> If any two vertices in G are connected by a distinct simple path, then G is connected.
> Let $(u, v) \in E$.
> Note (u, v) must be the unique path as proven by $(1) \implies (2)$.
> If we remove (u, v) from E, there will no longer be a path from $u \to v$.
> $\therefore$ The removal of (e, v) from E would disconnect G.

> $(3) \implies (4)$
> As G is connected, we know that $|E| \geq |V| - 1$ by a skipped exercise (B.4-3).
> We shall prove that $|E| \leq |V| - 1$ by induction.
> Note: A connected graph with $n = 1$ vertices has 0 edges and $n = 2$ vertices has 1 edge.
> Base Case: We can express this relation as a graph of n vertices has $n - 1$ edges.
> Suppose that G has $n \geq 3$ vertices and that all graphs satisfying (3) with fewer than n also satisfy $|E| \geq |V| - 1$.
> Removing an arbitrary edge from G separates the graph into $k \geq 2$.
> Let each subcomponent $G_i = (V_i, E_i)$ be a tree satisfying (3). If a G_i does not satisfy (3), then G was never a tree.
> As we know that $|V_i| < |V|$, by the inductive hypothesis we have that $|E_i| \leq |V_i| - 1$.
> Thus, $|V| - k \leq |V| - 2$ where k is also the number of removed edges.
> If we add back in the removed edge, we result in $|E| \leq |V| - 1$.
> $\therefore$ As $|E| \leq |V| - 1$ and $|E| \geq |V| - 1$, then $|E| = |V| - 1$.

> $(4) \implies (5)$
> Suppose that G is connected and that $|E| = |V| - 1$.
> To show G is acyclic, $suppose for contradiction that \$G$ has a simple cycle containing k vertices $v_1, \dots v_k$.
> Let $G_k = (V_k, E_k)$ be the subgraph of G consisting of the cycle.
> Note that $|V_k| = |E_k| = k$.
> If $k < |V|$, then $\exists v_{k+1} \in V - V_k$ that is adjacent to some $v_i \in V_k$ as G is connected.
> Define $G_{k+1} = (V_{k+1}, E_{k+1})$ to be the subgraph of G with $V_{k+1} = V_k \cup \{v_{k+1}\}$ and $E_{k+1} = E_k \cup \{e_{k+1}\}$.
> Note, this process can continue until we reach $G_n = (V_n, E_n)$ where $n = |V|$, $V_n = V$, and $|E_n| = |V_n| = |V|$.
> As G_n is a subgraph of G, we have that $E_n \subseteq E$.
> With the value of $|E_n|$ and its subset relationship, it follows that $|E| \leq |V|$ which contradicts the assumption that $|E| = |V| - 1$.
> $\therefore G$ is acyclic.

> $(5) \implies (6)$
> Suppose that G is acyclic and that $|E| = |V| - 1$.
> Let k be the number of connected components of G.
> Each connected component is a free tree by definition.
> As $(1) \implies (5)$, the sum of all edges in all connected components of G is $|V| - k$.
> Note: k must be $k = 1$ as G is a tree and there is 1 connected component in a tree.
> As $(1) \implies (2)$, any two vertices in G are connected by a unique path.
> $\therefore$ Adding any edge to G would create a cycle.

> $(6) \implies (1)$
> Suppose that $G = (V, E)$ is acyclic and that adding any edge to E creates a cycle.
> We must show that G is connected.
> Let $u, v \in V$.
> If u, v are not adjacent, adding the edge (u, v) creates a cycle in which all edges but (u, v) belong to G.
> Thus, the cycle minus edge (u, v) must contain a path from $u \to v$.
> $\therefore G$ is connected.

■

B.5.2 Rooted and Ordered Trees

Def: Rooted Tree is a free tree with a vertex distinguished from the others as the ***root***.
In the context of a rooted tree, the vertices are referred to as ***nodes***.
Visually, the root node is depicted as being the top most node.

Def: Parent: Given a non-root node, the parent node is the node directly above it.
Def: Ancestor: Let x be a node, any node in the simple path from the root to x is an ancestor of x.
Def: Descendant: Let x be a node, if a node has x as its ancestor, then that node is an ancestor of x.
  Note: A node x is both its ancestor & descendent by definition. Ancestors & descendents not of x are ***proper***.
Def: Child: Given a non-leaf node, a child node is the node directly below it. A node may have multiple children
Def: Siblings: Nodes that share the same parent.
Def: Leaf (external node): A node is a leaf or external node if it does not have any children.
Def: Internal Node: A nonleaf node.

Given a tree, a subtree can be induced from a given node x with x as the root.

Def: Degree: The number of children of a node.
Def: Depth: The length of the simple path from the root r to a node x is the depth of x in T.
Def: Level: The collection of nodes at the same depth.
Def: Height of a Node: The number of edges on the longest simple downward path from the node to a leaf.
Def: Height of a Tree: The largest depth of any node in the tree.

Def: Ordered Tree: A rooted tree where the children of each node are ordered.

B.5.3 Binary and Positional Trees

Def: Binary Tree: A recursive ordered tree structure defined on a finite set of nodes that either:
  - Contains no nodes, denoted as the ***empty tree***.
  - A three disjoint sets of nodes: a root node (parent), a left subtree (left child), and a right subtree (right child).

Def: Missing Child: A child connection is considered missing if it is the empty tree.
Def: Full Binary Tree: Each node is either a leaf or has degree exactly 2.

Def: K-ary Tree: An extension of the binary tree where each node has k children.
Def: Complete K-ary Tree: A k-ary tree in which all leaves have the same depth and all internal nodes have degree k.
  The number of leaves at depth h of a complete K-ary tree can be computed as k^h.
  The number of internal nodes of a tree of height h can be computed as:

$$1 + k^1 + k^2 + \cdots + k^{h-1} = \sum_{i=0}^{h-1} k^i = \frac{k^h - 1}{k - 1}\,.$$

Following this, a complete binary tree has $2^h - 1$ internal nodes.