CMSC 341
Lecture 8

Announcements

My office hours changing:
M 12:15-2
W 12:15-1

Proj 1 grading
– make sure your program compiles and runs in submit directory
– all future projects named “Proj#”
Tree ADT

Tree definition
- A tree is a set of nodes.
- The set may be empty.
- If not empty, then there is a distinguished node r, called root and zero or more non-empty subtrees $T_1, T_2, \ldots T_k$, each of whose roots are connected by a directed edge from r.

Basic Terminology
- Root of a subtree is a child of r. R is the parent.
- All children of a given node are called siblings.
- A leaf (or external) node has no children.

More Tree Terminology

A path from node $V_i$ to node $V_k$ is a sequence of nodes such that $V_i$ is the parent of $V_{i+1}$ for $1 \leq i \leq k$.
The length of this path is the number of edges encountered ($k-1$).
The depth of any node in a tree is the length of the path from root to the node.
All nodes of the same depth are at the same level.
The depth of a tree is the depth of its deepest leaf.
The height of any node in a tree is the length of the longest path from the node to a leaf.
The height of a tree is the height of its root.
Yet More Tree Terminology

The *internal path length* of a rooted tree is the sum of the depths of all of its nodes.

The *external path length* of a rooted tree is the sum of the depths of all the null pointers.

If there is a path from \( n_1 \) to \( n_2 \), then \( n_1 \) is an *ancestor* of \( n_2 \) and \( n_2 \) is a *descendent* of \( n_1 \).

Tree Storage

A tree node contains:
- Element
- Links
  - to each child
  - to sibling and first child
Binary Trees

A binary tree is a rooted tree in which no node can have more than two children AND the children are distinguished as left and right.

A full BT is a BT in which every node either has two children or is a leaf (every interior node has two children).

Traversals

- Inorder
- Preorder
- Postorder
Proof of FBT

A FBT with n internal nodes has n+1 leaf nodes:
Base case: BT of one node (the root) has:
- zero internal nodes
- one external node (the root)

Inductive Assumption (assume for n):
- All FBT of up to and including n internal nodes have n+1 external nodes.

Proof (cont)

Inductive Step (prove for n+1):
- Let T be a FBT of n internal nodes.
- It therefore has n+1 external nodes.
- Enlarge T by adding two nodes to some leaf. These are therefore leaf nodes.
- Number of leaf nodes increases by 2, but the former leaf becomes internal.
- So,
  - # internal nodes becomes n+1,
  - # leaves becomes (n+1)+1 = n+2
Perfect BT

A *perfect BT* is a full BT in which all leaves have the same depth.

Proof of PBT

The number of nodes in a PBT is $2^{h+1} - 1$, where $h$ is height.

Proof:

Notice that the number of nodes at each level is $2^l$.

So the total number of nodes is:

$$
\sum_{l=0}^{h} 2^l = 2^{h+1} - 1
$$

Prove this by induction:

Base case:

$$
\sum_{l=0}^{0} 2^l = 2^0 = 1 \iff 2^{0+1} - 1 = 2 - 1 = 1
$$
Proof of PBT (cont)

Assume true for all \( h \leq H \)

Prove for \( H+1 \):

\[
\sum_{l=0}^{H+1} 2^l = \sum_{l=0}^{H} 2^l + 2^{H+1}
\]

\[
= 2^{H+1} - 1 + 2^{H+1}
\]

\[
= 2^{H+2} - 1
\]

Complete BT

A complete BT is a perfect BT except that the lowest level may not be full. If not, it is filled from left to right.
Augmented BT

An *augmented binary tree* is a BT in which every unoccupied child position is filled by an additional “augmenting” node.