Today’s Topics
Introduction
Terminology and Characterization of Trees
Spanning Trees
Minimal Spanning Trees
Binary Trees
Tree Traversals
Decision Trees and the Minimum Time for Sorting
Isomorphisms of Trees

TREES
Binary Trees

• Definition
  – A binary tree is a rooted tree in which each vertex has either no children, one child, or two children.
  – If a vertex has a child, that child is designated as either a left child or a right child (but not both).
  – If a vertex has two children, one child is designated a left child and the other child is designated a right child.

• Example
  – the binary tree of Figure 9.5.1
  – a tree that defines a Huffman code
Binary Trees

• Note
  – A full binary tree is a binary tree in which each vertex has either two children or zero children.

• Theorem
  – If $T$ is a full binary tree with $i$ internal vertices, then $T$ has $i + 1$ terminal vertices and $2i + 1$ total vertices.

• Example
  – a single-elimination tournament
Binary Trees

• Theorem
  – If a binary tree of height $h$ has $t$ terminal vertices, then $\lg t \leq h$.
  – Proof sketch.
    • Prove the equivalent inequality $t \leq 2^h$ by induction on $h$.
    • Basis: $h = 0$.

• Example
  – the binary tree of Figure 9.5.3
Binary Trees

• Definition
  – A binary search tree is a binary tree $T$ in which data are associated with the vertices. The data are arranged so that, for each vertex $v$ in $T$, each data item in the left subtree of $v$ is less than the data item in $v$, and each data item in the right subtree of $v$ is greater than the data item in $v$.

• Example
  – Construct a binary search tree from the following words.
    • OLD PROGRAMMERS NEVER DIE
    • THEY JUST LOSE THEIR MEMORIES
Binary Trees

Algorithm 9.5.10: Constructing a Binary Search Tree

Input: A sequence $w_1, \ldots, w_n$ of distinct words and the length $n$ of the sequence
Output: A binary search tree $T$

\begin{verbatim}
make_bin_search_tree(w, n) {
    let $T$ be the tree with one vertex, root
    store $w_1$ in root
    for $i = 2$ to $n$ {
        $v = root$
        search = true // find spot for $w_i$
        while (search) {
            $s = \text{word in } v$
            if ($w_i < s$)
                if ($v$ has no left child) {
                    add a left child $l$ to $v$
                    store $w_i$ in $l$
                    search = false // end search
                }
                else
                    $v = \text{left child of } v$
            else
                search = false // end search
        }
    }
}
\end{verbatim}
Tree Traversals

• Preorder Traversal

Algorithm 9.6.1: Preorder Traversal

Input: \( PT \), the root of a binary tree
Output: Dependent on how “process” is interpreted in line 3

\[
\text{preorder}(PT) \{ \\
1. \quad \text{if } (PT \text{ is empty}) \\
2. \quad \text{return} \\
3. \quad \text{process } PT \\
4. \quad l = \text{left child of } PT \\
5. \quad \text{preorder}(l) \\
6. \quad r = \text{right child of } PT \\
7. \quad \text{preorder}(r) \\
\}
\]
Tree Traversals

• Inorder Traversal

Algorithm 9.6.3: Inorder Traversal

Input:  $PT$, the root of a binary tree
Output: Dependent on how “process” is interpreted in line 5

$$inorder(PT) \{$$
1. if ($PT$ is empty)
2. return
3. $l = \text{left child of } PT$
4. $inorder(l)$
5. process $PT$
6. $r = \text{right child of } PT$
7. $inorder(r)$
$$}$$
Tree Traversals

• Postorder Traversal

Algorithm 9.6.5: Postorder Traversal

Input:  \( PT \), the root of a binary tree
Output: Dependent on how “process” is interpreted in line 7

\[
\text{postorder}(PT) \{
1. \quad \text{if (PT is empty)}
2. \quad \text{return}
3. \quad l = \text{left child of } PT
4. \quad \text{postorder}(l)
5. \quad r = \text{right child of } PT
6. \quad \text{postorder}(r)
7. \quad \text{process } PT
\}
\]
Decision Trees and the Minimum Time for Sorting

• Examples
  – A decision tree for restaurants
  – Five-Coins Puzzle
    • Five coins are identical in appearance, but one coin is either heavier or lighter than the others, which all weigh the same.
    • The problem is to identify the bad coin and determine whether it is heavier or lighter than the others using only a pan balance, which compares the weights of two sets of coins.
Decision Trees and the Minimum Time for Sorting

• Example
  • A decision tree for sorting three elements

• Theorem
  – If \( f(n) \) is the number of comparisons needed to sort \( n \) items in the worst case by a sorting algorithm, then \( f(n) = \Omega(n \lg n) \).
Isomorphisms of Trees

• Example
  – Are the following pair of trees isomorphic?
    • the tree $T_1$ of Figures 9.8.1 and the tree $T_2$ of Figure 9.8.2
    • the trees $T_1$ and $T_2$ of Figure 9.8.3

• Theorem
  – There are three nonisomorphic trees with five vertices.
  – Proof.
    • Use an argument on the maximum degree on each vertex.
Isomorphisms of Trees

• Definition
  – Let $T_1$ be a rooted tree with root $r_1$ and let $T_2$ be a rooted tree with root $r_2$. The rooted trees $T_1$ and $T_2$ are isomorphic if there is a one-to-one, onto function $f$ from the vertex set of $T_1$ to the vertex set of $T_2$ satisfying the following:
    (a) Vertices $v_i$ and $v_j$ are adjacent in $T_1$ if and only if the vertices $f(v_i)$ and $f(v_j)$ are adjacent in $T_2$.
    (b) $f(r_1) = r_2$.
  – We call the function $f$ an isomorphism.

• Examples
  – Are the following pair of trees isomorphic?
    • the rooted trees of Figure 9.8.7
    • the rooted trees of Figure 9.8.8: isomorphic only as free trees
Isomorphisms of Trees

• Theorem
  – There are four nonisomorphic rooted trees with four vertices. These four rooted trees are shown in Figure 9.8.9.
  – Proof.
    • Use an argument on the maximum degree of each vertex.
Isomorphisms of Trees

• Definition
  – Let $T_1$ be a binary tree with root $r_1$ and let $T_2$ be a binary tree with root $r_2$. The binary trees $T_1$ and $T_2$ are isomorphic if there is a one-to-one, onto function $f$ from the vertex set of $T_1$ to the vertex set of $T_2$ satisfying the following:
    (a) Vertices $v_i$ and $v_j$ are adjacent in $T_1$ if and only if the vertices $f(v_i)$ and $f(v_j)$ are adjacent in $T_2$.
    (b) $f(r_1) = r_2$.
    (c) $v$ is a left child of $w$ in $T_1$ if and only if $f(v)$ is a left child of $f(w)$ in $T_2$.
    (d) $v$ is a right child of $w$ in $T_1$ if and only if $f(v)$ is a right child of $f(w)$ in $T_2$. 
Isomorphisms of Trees

• Examples
  – Are the following pair of trees isomorphic?
    • the binary trees of Figure 9.8.10
    • the binary trees of Figure 9.8.11
Isomorphisms of Trees

• Theorem
  – There are five nonisomorphic binary trees with three vertices. These five binary trees are shown in Figure 9.8.12.

• Theorem
  – There are $C_n$ nonisomorphic binary trees with $n$ vertices where $C_n = C(2n,n)/(n+1)$ is the $n$th Catalan number.
Isomorphisms of Trees

Algorithm 9.8.13: Testing Whether Two Binary Trees Are Isomorphic

Input: The roots \( r_1 \) and \( r_2 \) of two binary trees. (If the first tree is empty, \( r_1 \) has the special value \( \text{null} \). If the second tree is empty, \( r_2 \) has the special value \( \text{null} \).)

Output: true, if the trees are isomorphic
false, if the trees are not isomorphic

\[
\text{bin_tree_isom}(r_1, r_2) \{
1. \quad \text{if } (r_1 == \text{null} \land r_2 == \text{null})
2. \quad \text{return true}
3. \quad \text{// now one or both of } r_1 \text{ or } r_2 \text{ is not } \text{null}
4. \quad \text{if } (r_1 == \text{null} \lor r_2 == \text{null})
5. \quad \text{return false}
6. \quad \text{// now neither of } r_1 \text{ or } r_2 \text{ is } \text{null}
7. \quad \text{lc}_r_1 = \text{left child of } r_1
8. \quad \text{lc}_r_2 = \text{left child of } r_2
9. \quad \text{rc}_r_1 = \text{right child of } r_1
10. \quad \text{rc}_r_2 = \text{right child of } r_2
11. \quad \text{return } \text{bin_tree_isom} (\text{lc}_r_1, \text{lc}_r_2) \land \text{bin_tree_isom} (\text{rc}_r_1, \text{rc}_r_2)
\}
\]
Summary

• Terminology and Characterization of Trees
• Spanning Trees
• Minimal Spanning Trees
• Binary Trees
• Tree Traversals
• Decision Trees and the Minimum Time for Sorting
• Isomorphisms of Trees