Lecture 01/30/17

Lecturer: Xiaodi Wu

Reading Assignment: Chapter 10.4, 12.1-3
A **traversal** of a tree \( T \) is a systematical way of "visiting" all nodes in \( T \).
A **traversal** of a tree $T$ is a systematical way of "visiting" all nodes in $T$.

- Pre-order: Root first and then visit each sub-tree in order.
- Post-order: Visit each sub-tree first and then the root.
Traversals of Trees

**Algorithm** preorder($T, v$)

"visit" the node $v$

for each child $w$ of $v$ do

preorder($T, w$)

end for

**Algorithm** postorder($T, v$)

for each child $w$ of $v$ do

post-order($T, w$)

end for

"visit" the node $v$
Traversal of Trees

**Algorithm preorder** $(T, v)$

"visit" the node $v$

for each child $w$ of $v$ do

preorder$(T, w)$

end for

**Algorithm postorder** $(T, v)$

for each child $w$ of $v$ do

post-order$(T, w)$

end for

"visit" the node $v$

**Complexity**

$O(n)$: similar counting as the analysis in height().
Trees: Pre-order

Pre-order: 33, 15, 10, 5, 12, 20, 18, 47, 38, 36, 39, 51, 49
Trees: Pre-order

Pre-order: 33, 15, 10, 5, 12, 20, 18, 47, 38, 36, 39, 51, 49
Trees: Post-order

Post-order: 5, 12, 10, 18, 20, 15, 36, 39, 38, 49, 51, 47, 33
Post-order: 5, 12, 10, 18, 20, 15, 36, 39, 38, 49, 51, 47, 33
A **binary tree** is an ordered tree in which each node has at most two children. It is called **proper** if each internal node has two children (**left** and **right child**).
A **binary tree** is an ordered tree in which each node has at most two children. It is called **proper** if each internal node has two children (**left** and **right child**).

**Methods**

- leftChild(v): return the left child of v if v is internal.
- rightChild(v): return the right child of v if v is internal.
Binary Tree

A **binary tree** is an ordered tree in which each node has at most two children. It is called **proper** if each internal node has two children (**left** and **right child**).

Methods

- **leftChild(v)**: return the left child of $v$ if $v$ is internal.
- **rightChild(v)**: return the right child of $v$ if $v$ is internal.

A third traversal order: **inorder**.
Traversal of Binary Trees

**Algorithm** bPreorder\((T, v)\)

"visit" the node \(v\)

if \(v\) is internal then

bPreorder\((T, T.leftChild(v))\)

bPreorder\((T, T.rightChild(v))\)

end if

**Algorithm** bPostorder\((T, v)\)

if \(v\) is internal then

bPostorder\((T, T.leftChild(v))\)

bPostorder\((T, T.rightChild(v))\)

end if

"visit" the node \(v\)
Algorithm blnorder\((T, \nu)\)

if \(\nu\) is internal then
  bPostorder\((T, T.leftChild(\nu))\)
end if

"visit" the node \(\nu\)

if \(\nu\) is internal then
  if \(\nu\) is internal then
    bPostorder\((T, T.rightChild(\nu))\)
  end if
end if
In-order: 5, 10, 12, 15, 18, 20, 33, 36, 38, 39, 47, 49, 51
Identify Trees from preorder, inorder, postorder visits

- **Pre-order:** 33, 15, 10, 5, 12, 20, 18, 47, 38, 36, 39, 51, 49
- **Post-order:** 5, 12, 10, 18, 20, 15, 36, 39, 38, 49, 51, 47, 33
- **In-order:** 5, 10, 12, 15, 18, 20, 33, 36, 38, 39, 47, 49, 51
Identify Trees from preorder, inorder, postorder visits

- Pre-order: 33, 15, 10, 5, 12, 20, 18, 47, 38, 36, 39, 51, 49
- Post-order: 5, 12, 10, 18, 20, 15, 36, 39, 38, 49, 51, 47, 33
- In-order: 5, 10, 12, 15, 18, 20, 33, 36, 38, 39, 47, 49, 51

How?
- Use Pre-order or Post-order to identify the root.
Identify Trees from preorder, inorder, postorder visits

- Pre-order: 33, 15, 10, 5, 12, 20, 18, 47, 38, 36, 39, 51, 49
- Post-order: 5, 12, 10, 18, 20, 15, 36, 39, 38, 49, 51, 47, 33
- In-order: 5, 10, 12, 15, 18, 20, 33, 36, 38, 39, 47, 49, 51

How?
- Use Pre-order or Post-order to identify the root.
- Use In-order to identify both sub-trees.
Identify Trees from preorder, inorder, postorder visits

- Pre-order: 33, 15, 10, 5, 12, 20, 18, 47, 38, 36, 39, 51, 49
- Post-order: 5, 12, 10, 18, 20, 15, 36, 39, 38, 49, 51, 47, 33
- In-order: 5, 10, 12, 15, 18, 20, 33, 36, 38, 39, 47, 49, 51

How?

- Use Pre-order or Post-order to identify the root.
- Use In-order to identify both sub-trees.
- Apply the above procedure recursively.
Arithmetic Expression

\[
\begin{align*}
\frac{(3 + 1) \times 3}{(9 - 5) + 2}\end{align*}
\begin{align*}
- (3 \times (7 - 4) + 6)
\end{align*}
\]
Arithmetic Expression

\(((3 + 1) \times 3)/((9 - 5) + 2)) - ((3 \times (7 - 4)) + 6)
Arithmetic Expression: Preorder

- /
  /  
× +  
 +  
  
+ 3 2 3  
  
3 1 9 5 7 4 

6
Arithmetic Expression: Postorder

\[ \frac{(3 + 1) \times 3}{((9 - 5) + 2) + \frac{3 \times (7 - 4) 	imes 6}{7 - 4}} \]
Arithmetic Expression: Inorder

\[ (((3 + 1) \times 3) \div ((9 - 5) + 2)) - ((3 \times (7 - 4)) + 6) \]
Arithmetic Expression: Inorder

\[ (((3 + 1) \times 3)/((9 - 5) + 2)) - ((3 \times (7 - 4)) + 6) \]
Properties about Binary Trees

Theorem

Let $T$ be a (proper) binary tree with $n$ nodes, $h$ the height of $T$. We have

- # external nodes of $T$ is between $h + 1$ and $2^h$.
- # internal nodes of $T$ is between $h$ and $2^h - 1$.
- The height of $T$ is between $\log(n + 1) - 1$ and $(n - 1)/2$. 
Properties about Binary Trees

Theorem

In a (proper) binary tree $T$, the number of external nodes is 1 more than the number of internal nodes.
Theorem

In a (proper) binary tree $T$, the number of external nodes is 1 more than the number of internal nodes.

Proof.

By induction,

- If $T$ only has one node, it must be external. Thus, no internal node. The statement holds.
Properties about Binary Trees

Theorem

In a (proper) binary tree $T$, the number of external nodes is 1 more than the number of internal nodes.

Proof.

By induction,

- If $T$ only has one node, it must be external. Thus, no internal node. The statement holds.
- Otherwise, $T$ has at least one external node with its parent. Remove any external node $w$ and its parent $v$, then connect $w$’s sibling to $v$’s parent. The tree remains proper and binary, but smaller.
Properties about Binary Trees

Let $e$, $i$ be external/internal nodes of a (proper) binary tree.

- $e = i + 1$ and $e + i = n$. 
Let $\#e, \#i$ be $\#$ external/internal nodes of a (proper) binary tree.

- $\#e = \#i + 1$ and $\#e + \#i = n$.
- $n \geq 2h + 1$. What is this case?
Properties about Binary Trees

Let \( e \), \( i \) be external/internal nodes of a (proper) binary tree.

- \( e = i + 1 \) and \( e + i = n \).
- \( n \geq 2h + 1 \). What is this case?
- \( n \leq 2^{h+1} - 1 \). What is this case?
Properties about Binary Trees

Let $e$, $i$ be external/internal nodes of a (proper) binary tree.

- $e = i + 1$ and $e + i = n$.
- $n \geq 2h + 1$. What is this case?
- $n \leq 2^{h+1} - 1$. What is this case?
- $(n - 1)/2 \leq h \leq \log(n + 1) - 1$.
- $h + 1 \leq e \leq 2^h$.
- $h \leq i \leq 2^h - 1$. 