Tree Structures and Tree Traversals

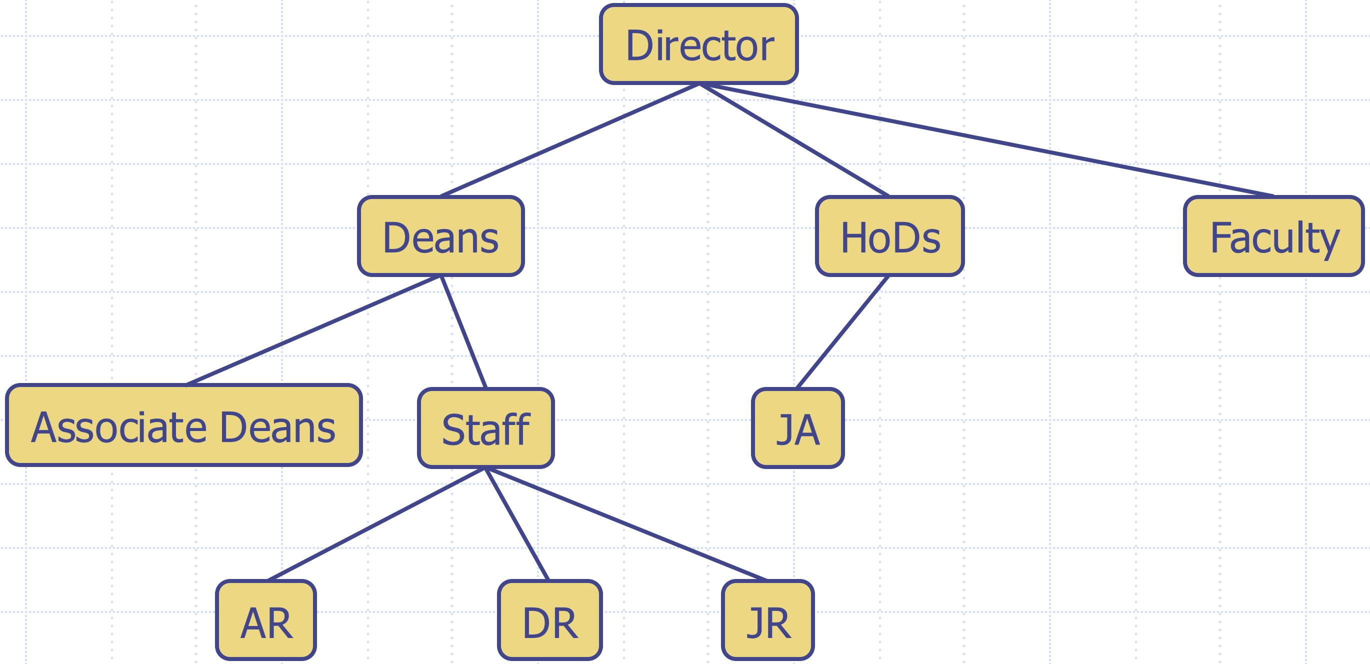
Readings - Chapter 8

Trees

- Abstract model of a hierarchical structure
- A tree consists of nodes with a parent-child relation

google images

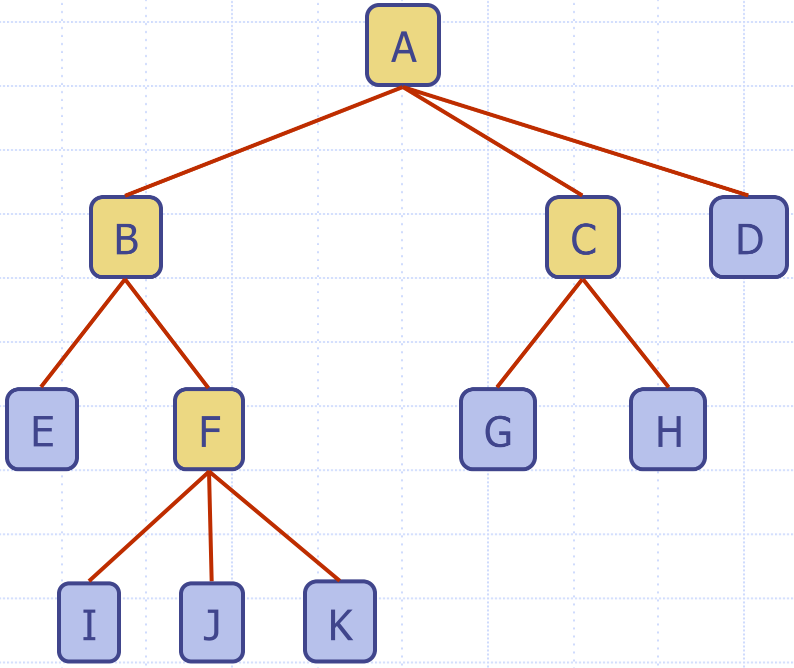
Trees - Examples



organization structure of a corporation

Trees - Terminology

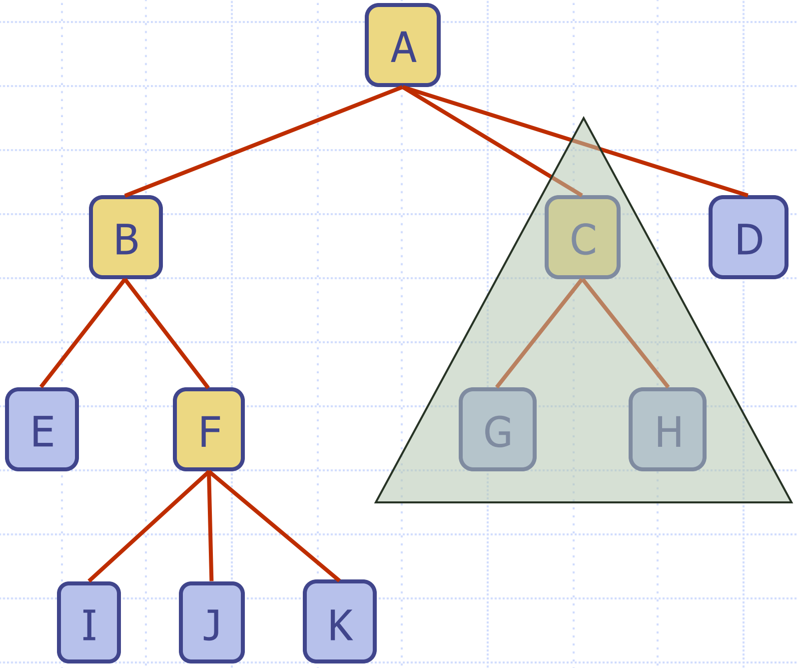
- *A* is the *root* node
- *B* is *parent* of *E* and *F*
- *A* is *ancestor* of *E* and *F*
- *E* and *F* are *descendants* of *A*
- *C* is the *sibling* of *B*
- *E* and *F* are *children* of *B*
- *E, I, J, K, G, H,* and *D* are *leaves*
- *A, B, C,* and *F* are *internal nodes*



Trees - Terminology (2)

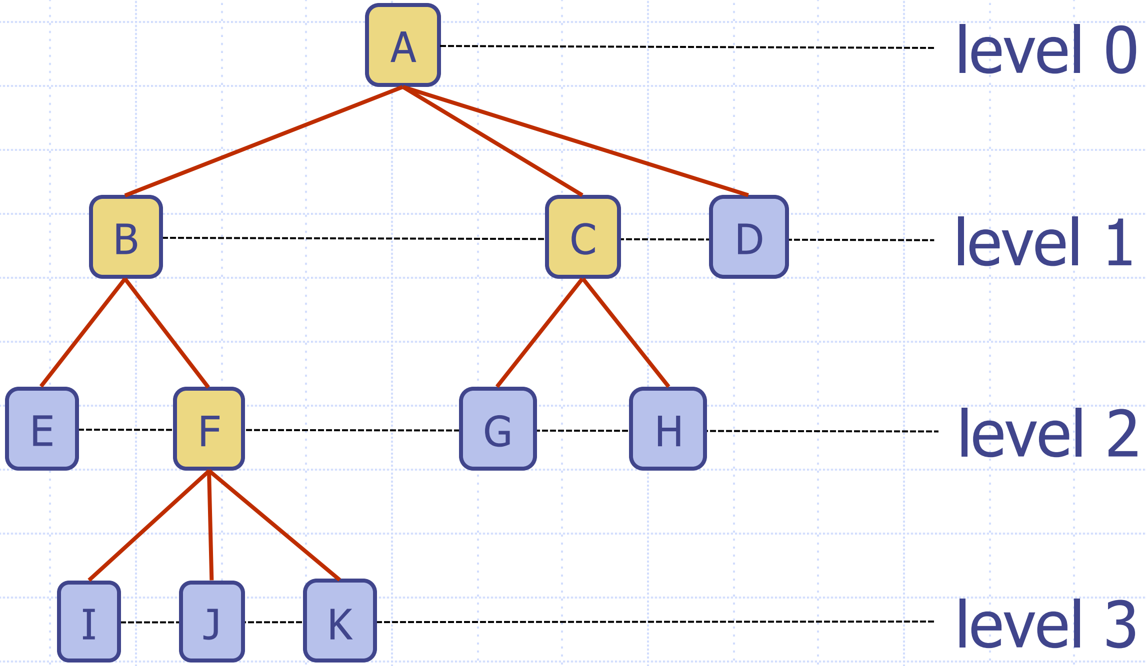
- *A* is the *root* node
- *B* is *parent* of *E* and *F*
- *A* is *ancestor* of *E* and *F*
- *E* and *F* are *descendants* of *A*
- *C* is the *sibling* of *B*
- *E* and *F* are *children* of *B*
- *E, I, J, K, G, H,* and *D* are *leaves*
- *A, B, C,* and *F* are *internal nodes*

- *Subtree:* tree consisting of node and its descendants



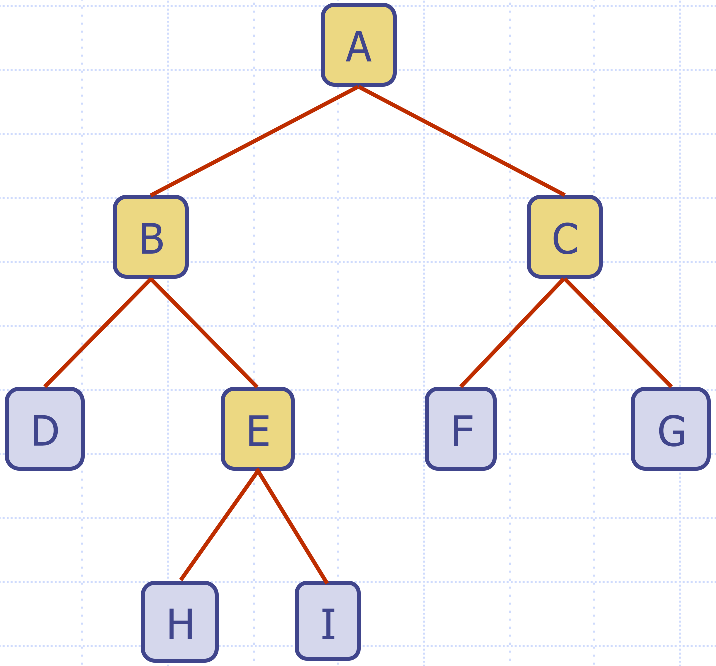
Trees - Terminology (3)

- The *depth* (*level*) of *E* is 2
- The *height* of the tree is 3
- The *degree* of node F is 3



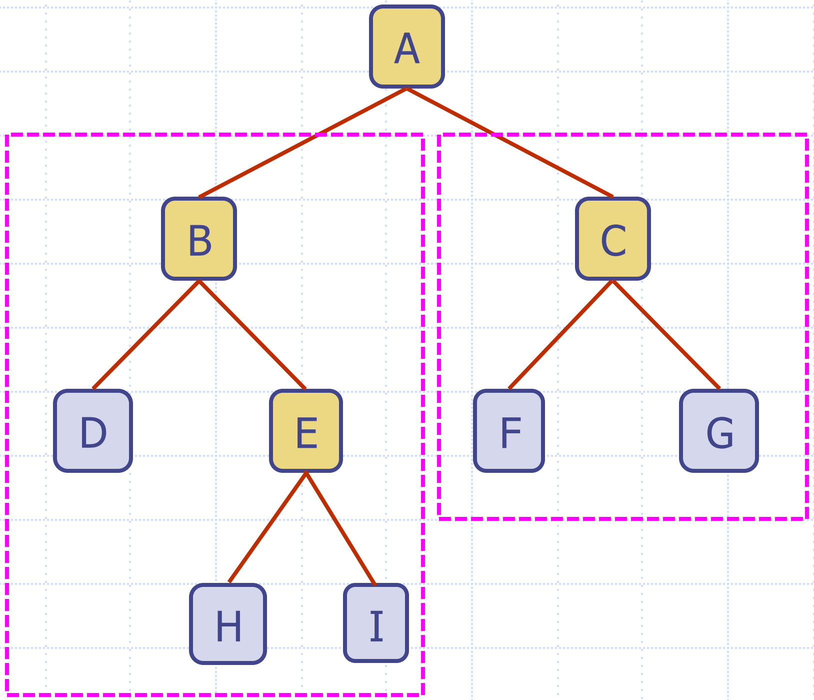
Binary Trees

- An **ordered tree** is one in which the children of each node are ordered
- **Binary tree**: ordered tree with all nodes having at most 2 children
  - left child and right child



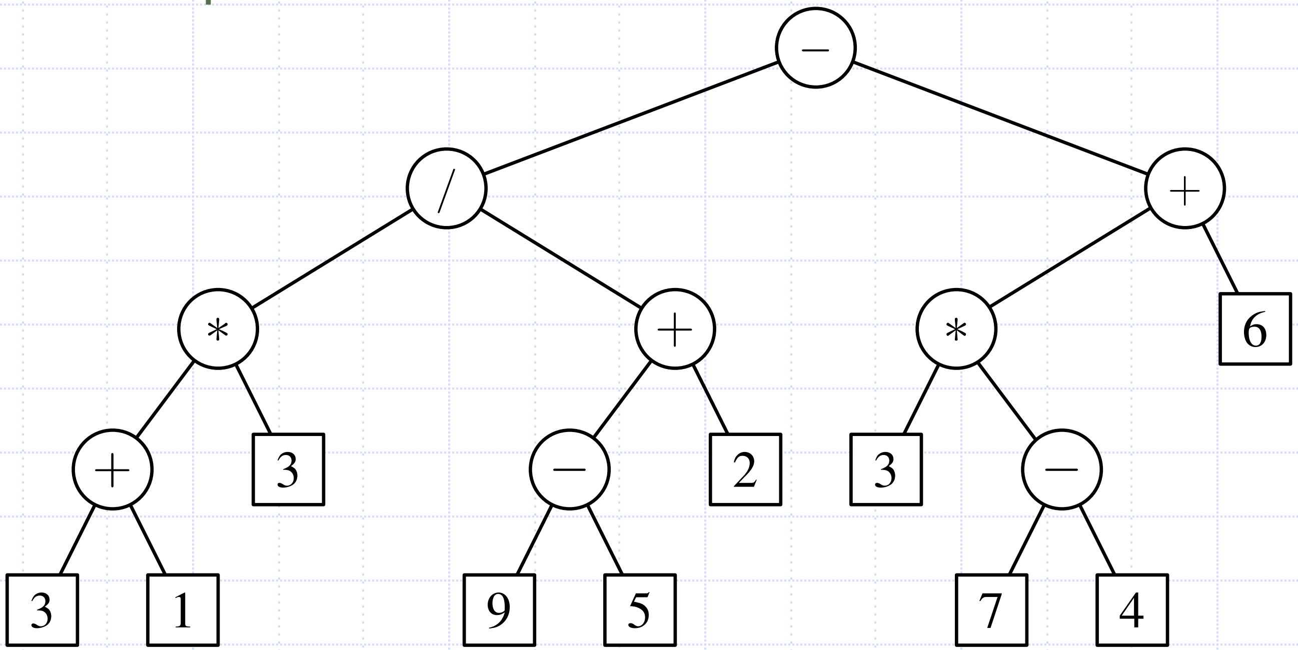
Binary Trees

- Recursive definition of binary tree
  - either a leaf or
  - an internal node (the root) and one/two binary trees (left subtree and/or right subtree)



Example of Binary Trees - Arithmetic Expression Tree

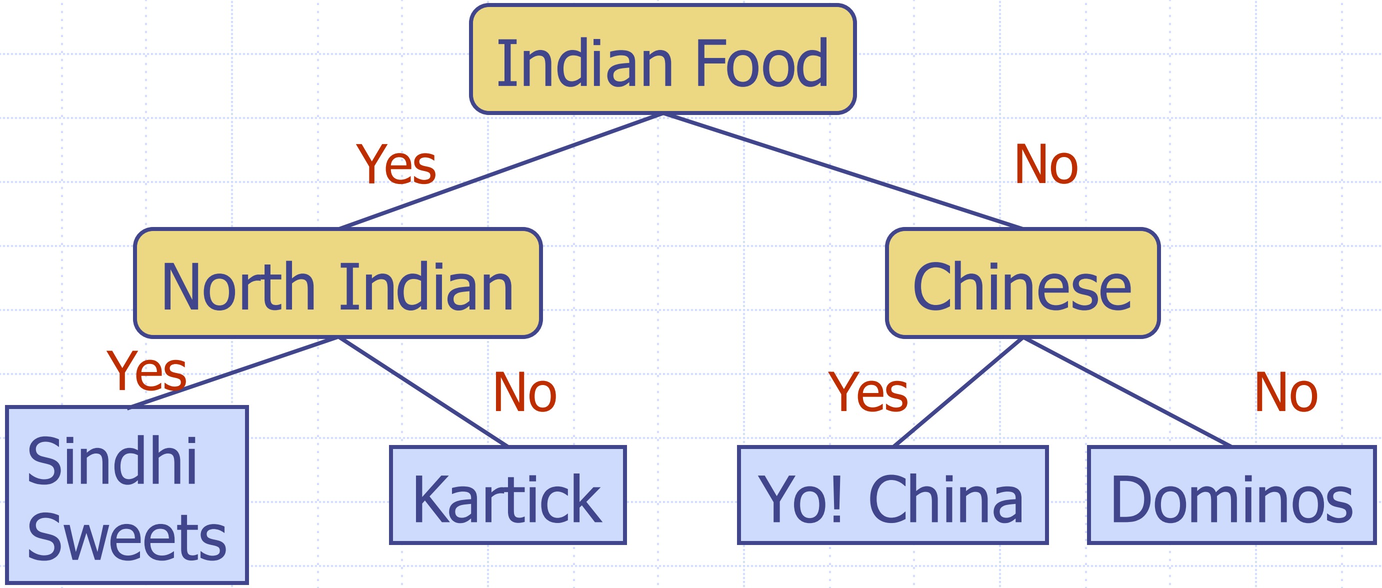
- Binary tree associated with an arithmetic expression
  - internal nodes: operators
  - external nodes: operands



((((3 + 1) *3) / (9 - 5)+2))-((3*(7 - 4)) +6))

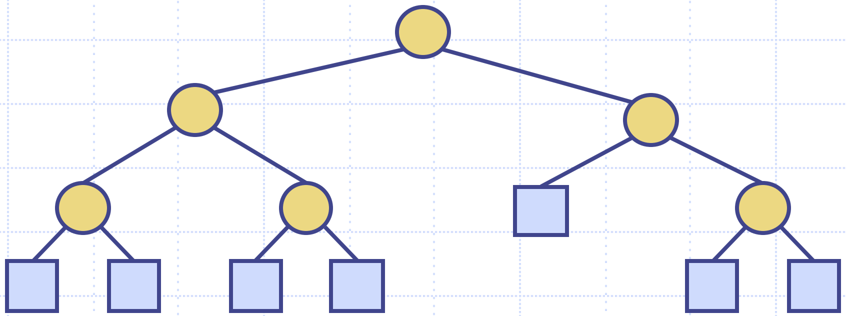
Example of Binary Trees - Decision Tree

- Binary tree associated with a decision process
  - internal nodes: questions with yes/no answer
  - external nodes: decisions
- Example: dining decision

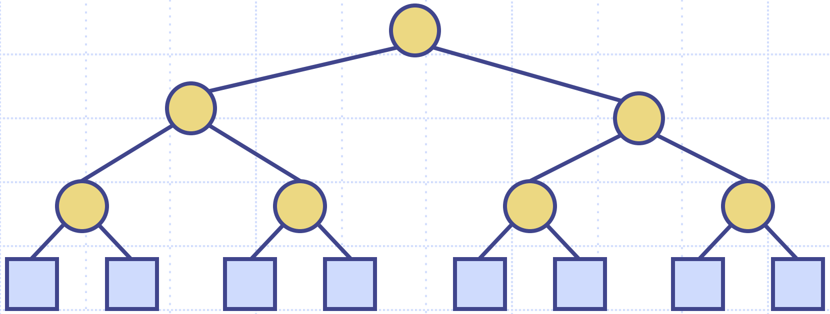


Proper, Full, Complete Binary Trees

- Proper/Full - Every node has either zero or two children

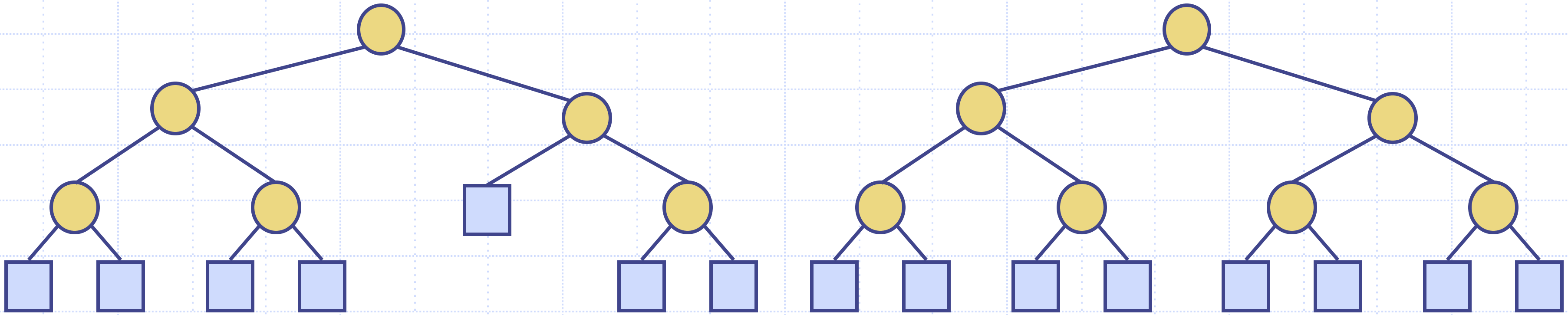


- Complete - every level except possibly the last is completely filled and all leaf nodes are as left as possible.



Binary tree from a complete binary tree

- A binary tree can be obtained from appropriate complete binary tree by pruning.



Properties of a Binary Tree

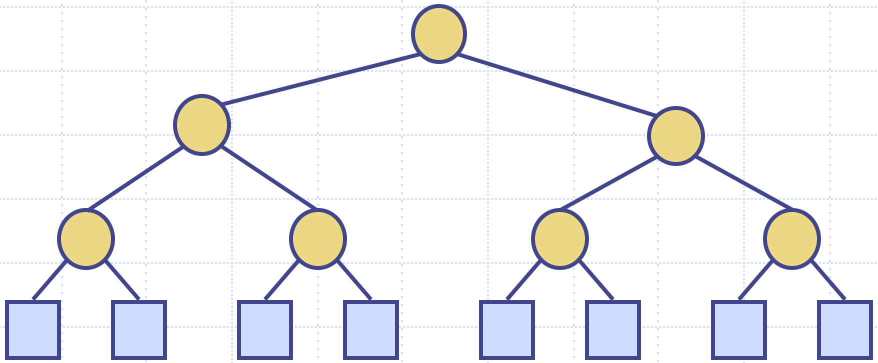
- Notations
  - n - number of nodes
  - n_E - number of leaves (external nodes)
  - n_I - number of internal nodes
  - h - height of the tree
- $h+1 \leq n \leq 2^{h+1} -1$
- $1 \leq n_E \leq 2^h$
- $h \leq n_I \leq 2^h -1$
- $\log(n+1) -1 \leq h \leq n-1$

Properties of Binary Trees (2)

- $n_E \leq n_I + 1$
- proof by induction on n_I
  - Tree with 1 node has a leaf but no internal node
  - Assume $n_E \leq n_I + 1$ for tree with $k\text{-}1$ internal nodes
  - A tree with k internal nodes has k_1 internal nodes in the left subtree and $k\text{-}k_1\text{-}1$ internal nodes in the right subtree
  - By induction $n_E \leq (k_1+1) + (k-k_1-1+1) = k+1$

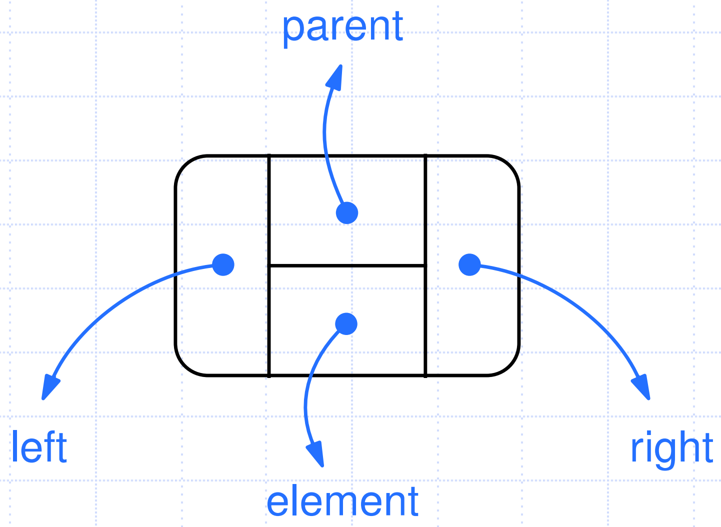
Complete Binary Tree

- level i has 2^i nodes
- In a tree of height h
  - leaves are at level h
  - $n_E = 2^h$
  - $n_I = 1 + 2 + 2^2 + \ldots + 2^{h-1} = 2^h - 1$
  - $n_I = n_E - 1$
  - $n = 2^{h+1} - 1$
- In a tree of n nodes
  - n_E is $(n+1)/2$
  - $h = \log_2 (n_E)$

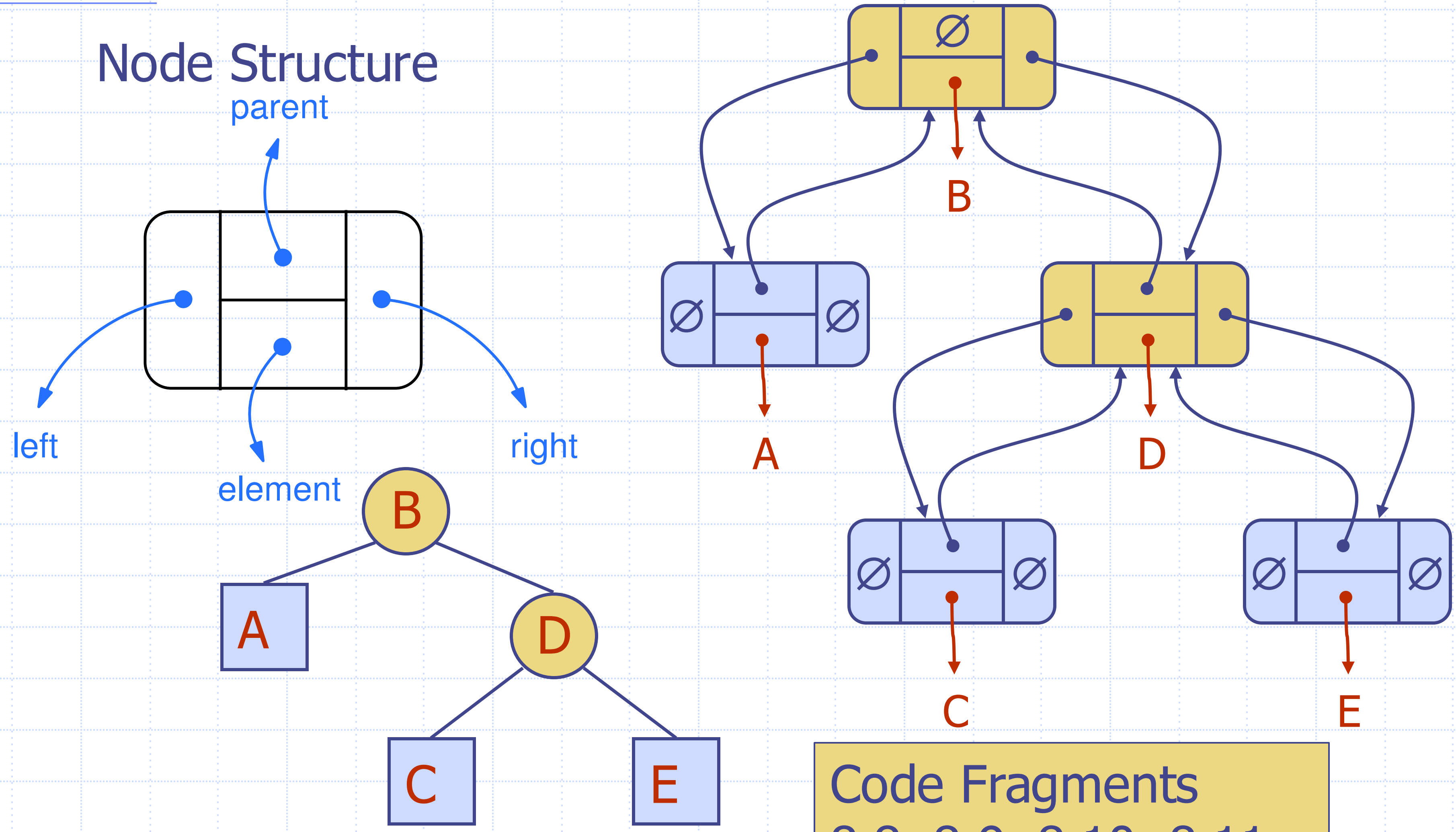


Linked Structure for Binary Trees

Node Structure



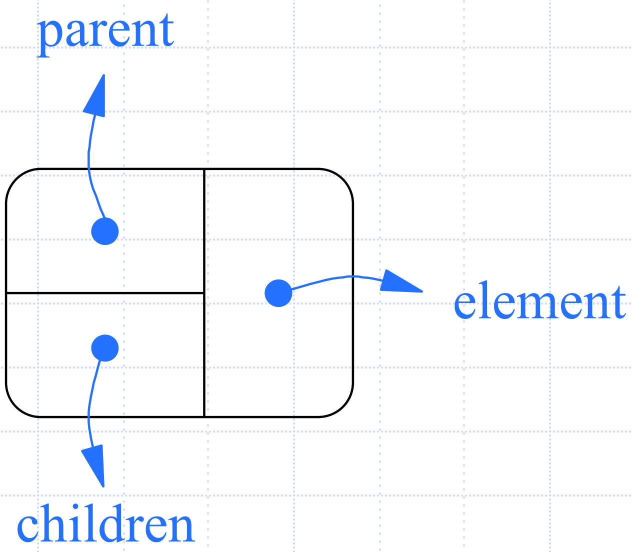
Linked Structure for Binary Trees



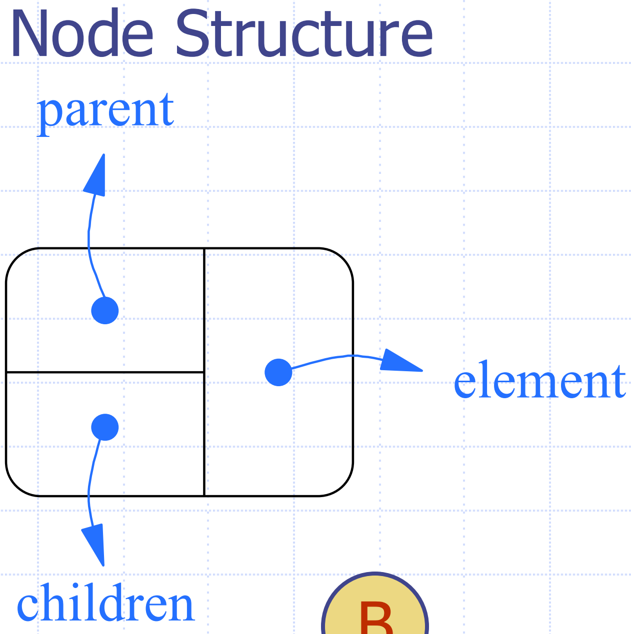
Code Fragments
8.8, 8.9, 8.10, 8.11

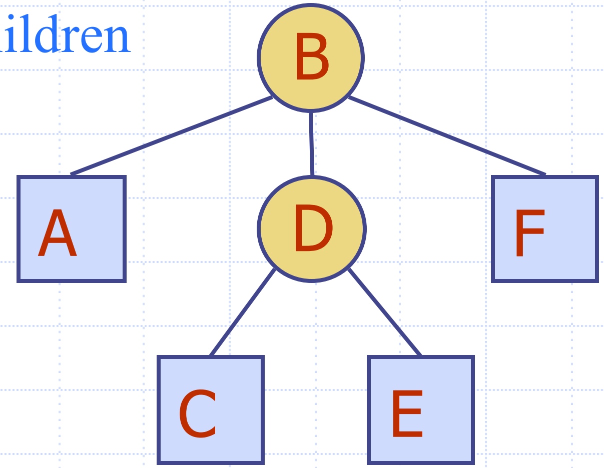
Linked Structure for General Trees

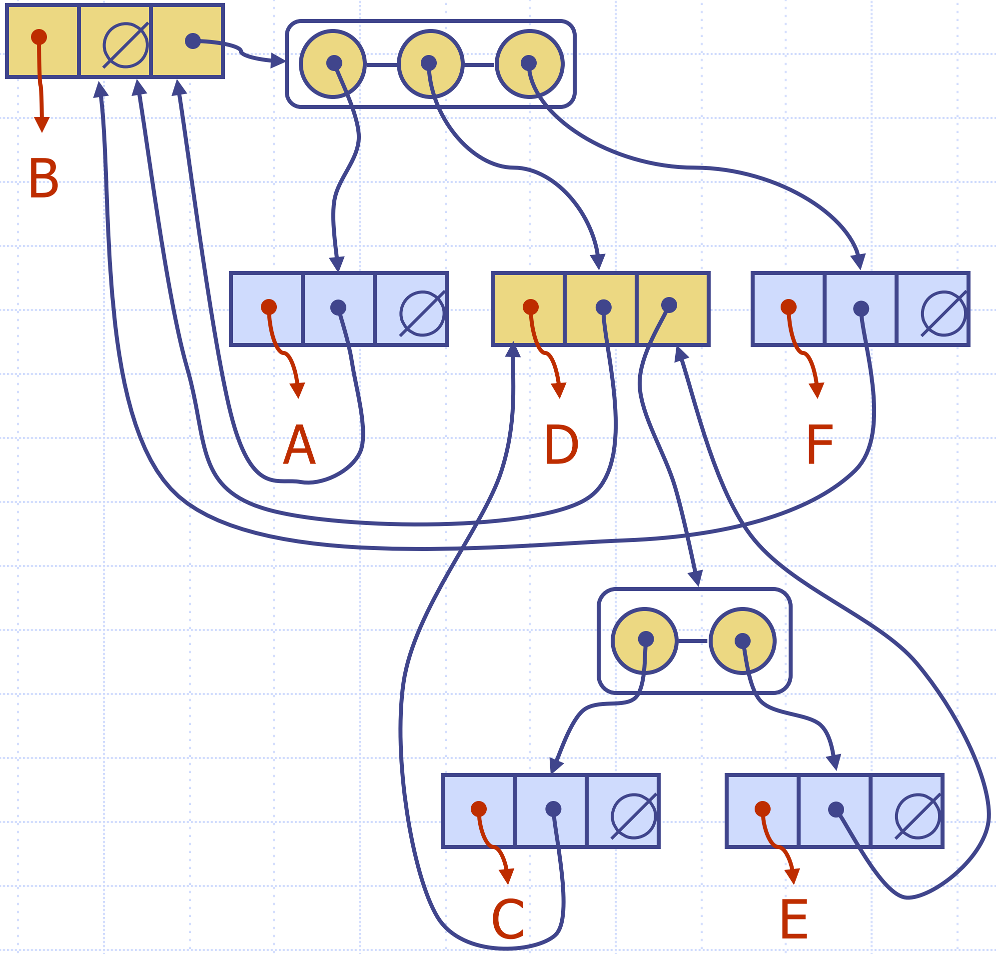
Node Structure



Linked Structure for General Trees



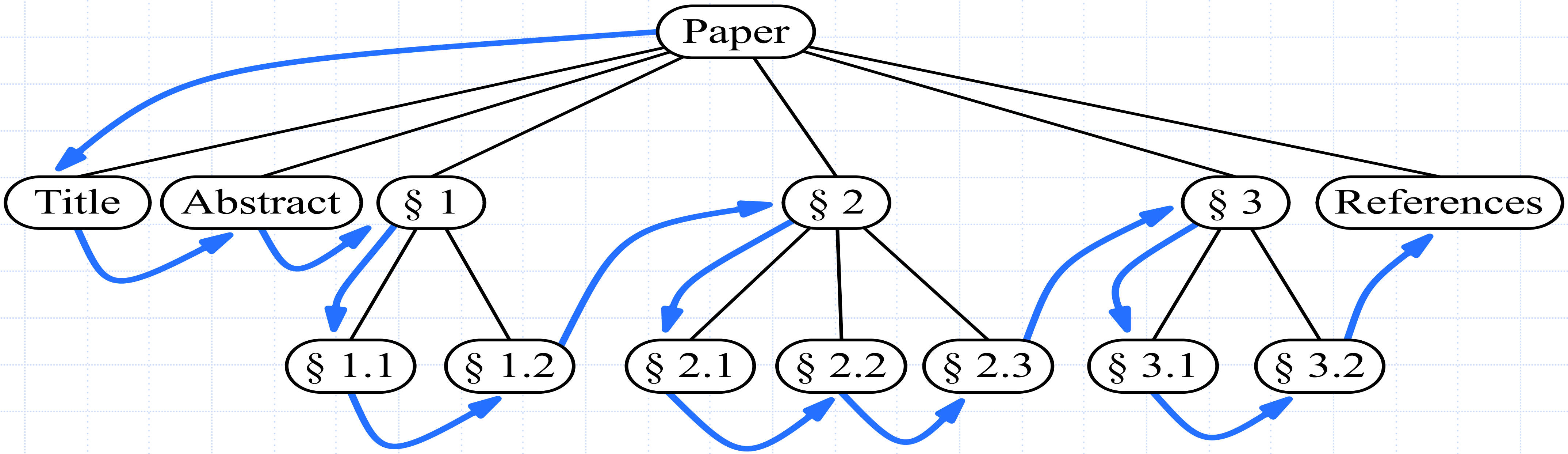




Tree Traversals

- Systematic way of visiting all nodes in a tree in a specified order
  - preorder - processes each node before processing its children
  - postorder - processes each node after processing its children

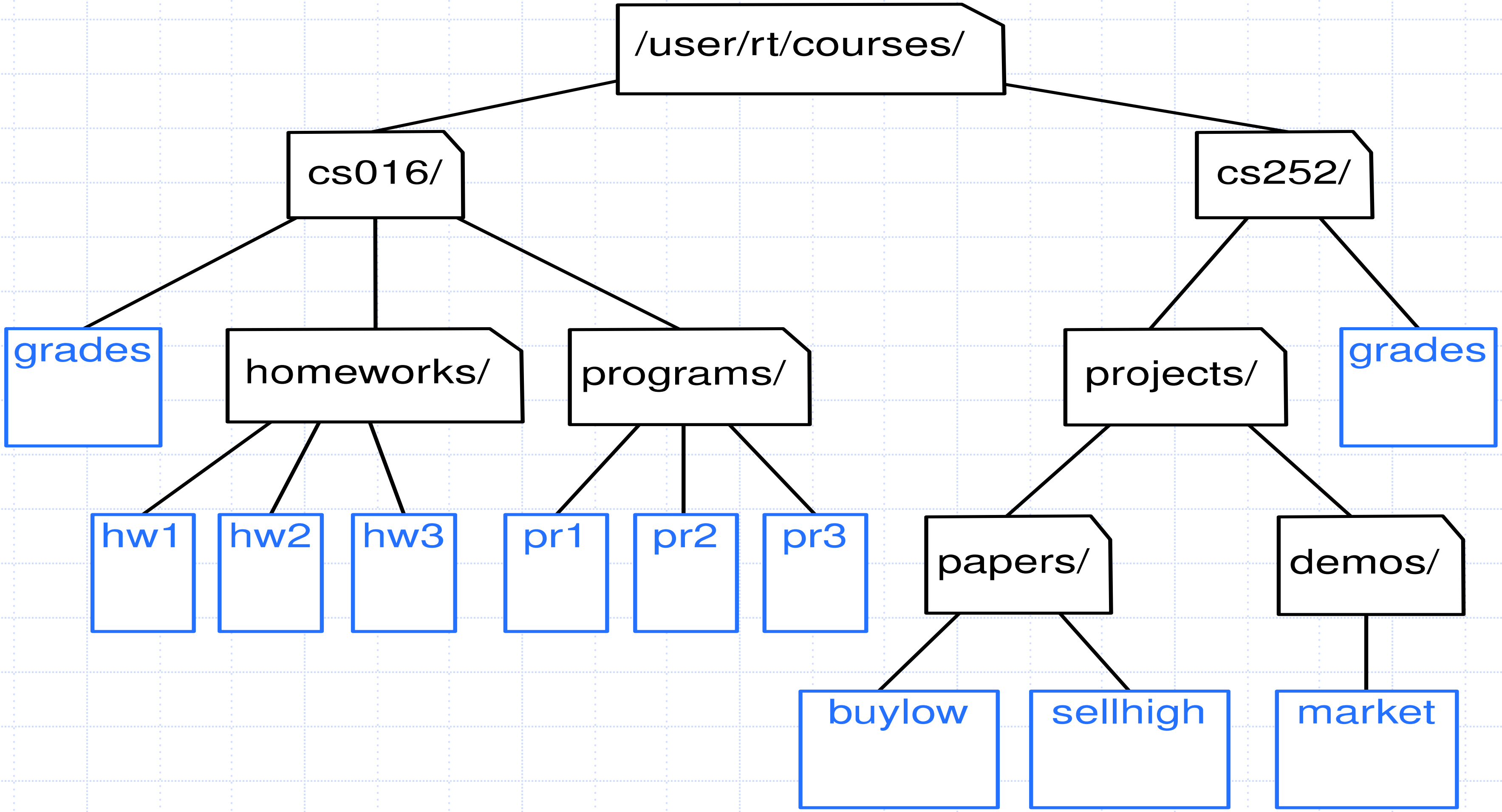
Preorder Traversal



Preorder Traversal - Algorithm

- Algorithm preorder(p)
  - perform the “visit” action for position p
  - for each child c in children(p) do
    - preorder(c)
- Example:
  - reading a document from beginning to end

Postorder Traversal



Postorder Traversal - Algorithm

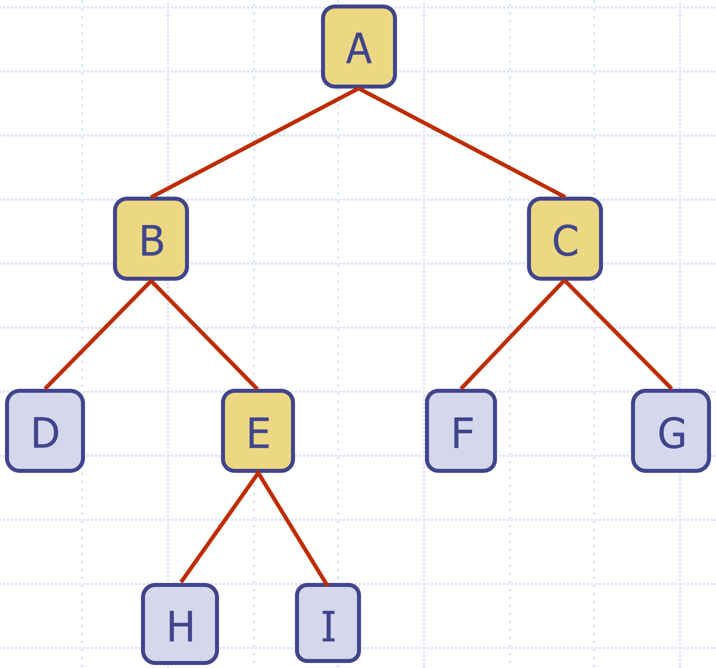
- Algorithm postorder(p)
  - for each child c in children(p) do
    - postorder(c)
  - perform the “visit” action for position p
- Example
  - du - disk usage command in Unix

Traversals of Binary Trees

- preorder(v)
  - visit(v)
  - preorder(v.leftchild())
  - preorder(v.rightchild())
- postorder(v)
  - postorder(v.leftchild())
  - postorder(v.rightchild())
  - visit(v)

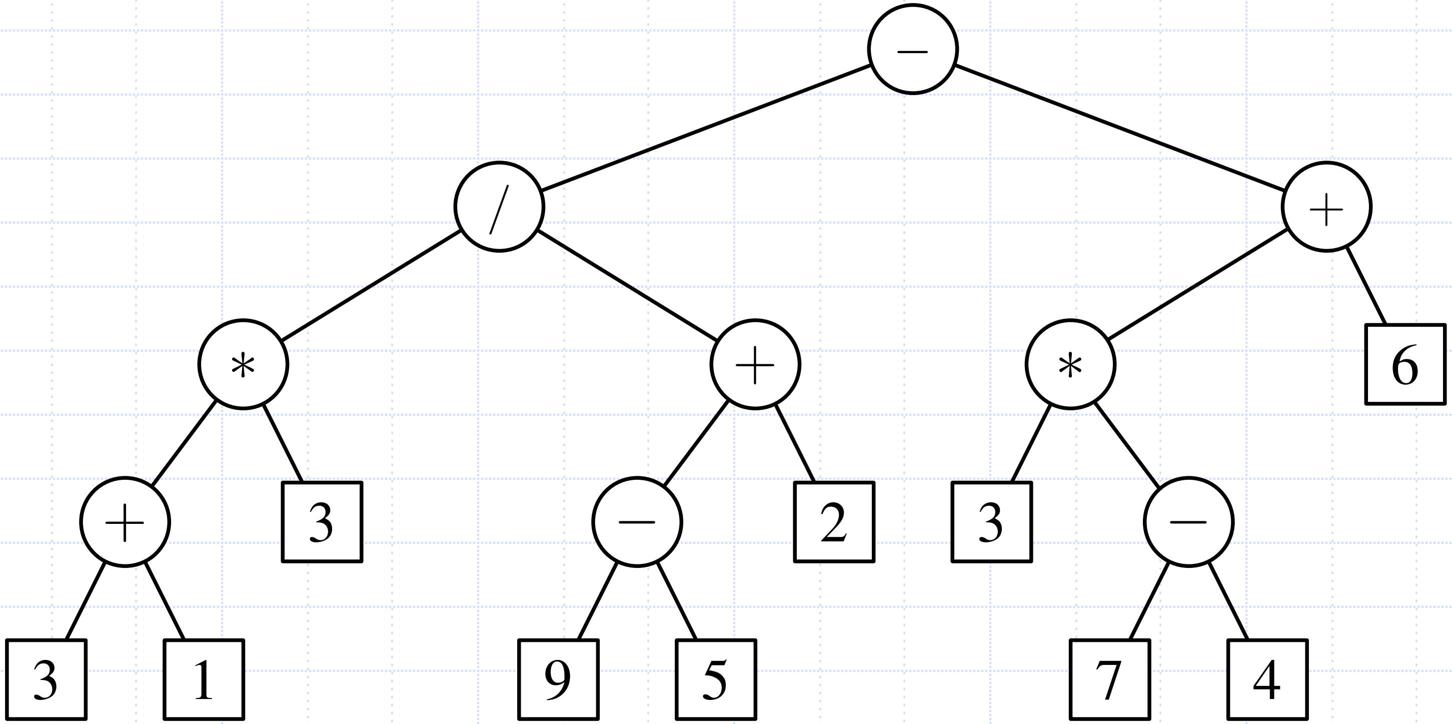
More Example of Traversals

- Visit - printing the data in the node
- Preorder traversal
  - a b d e h i c f g
- Postorder traversal
  - d h i e b f g c a



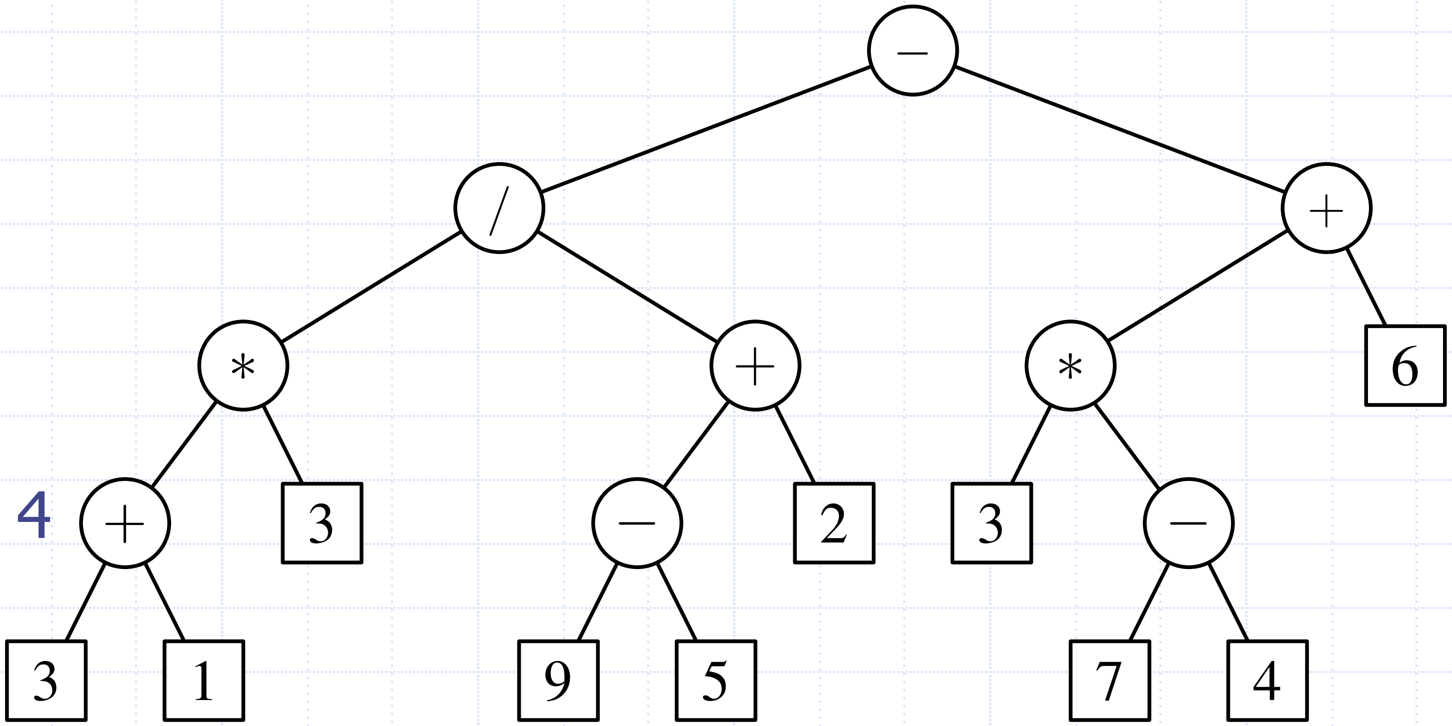
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



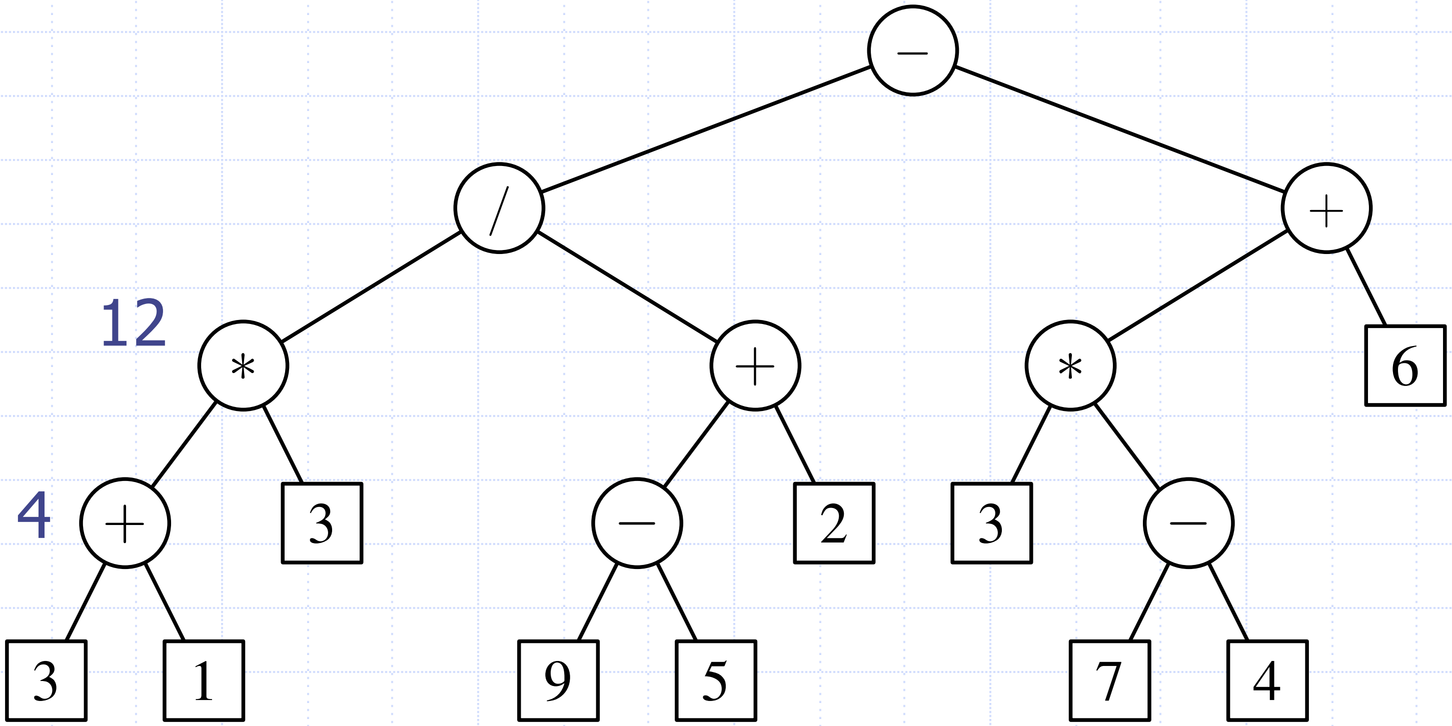
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



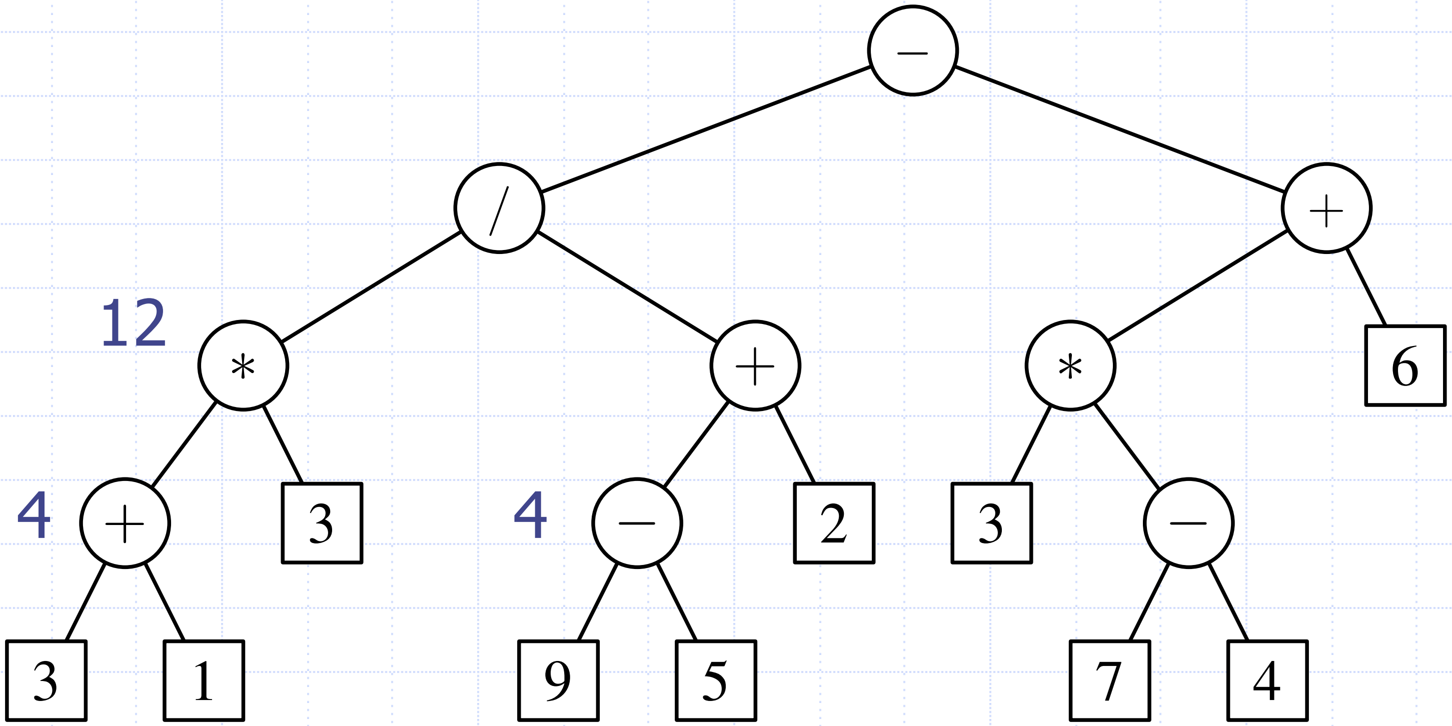
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



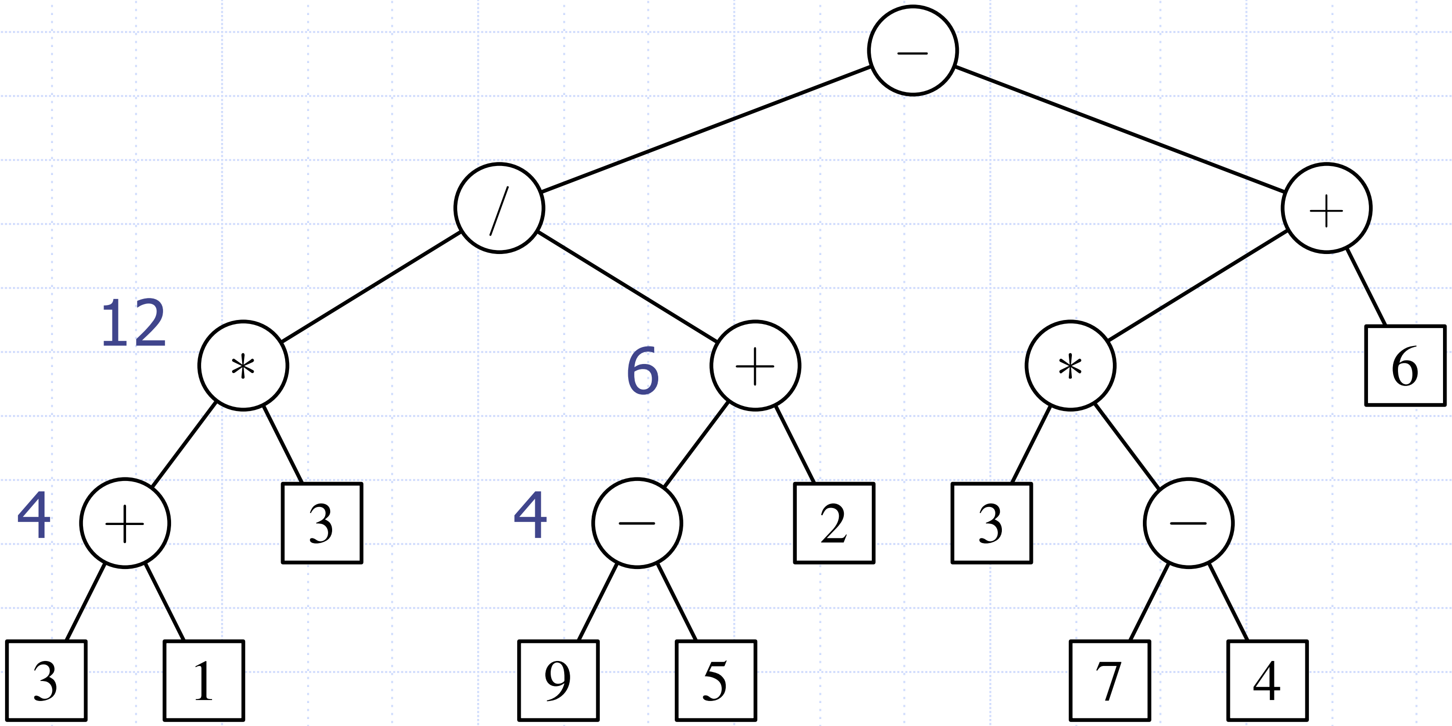
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



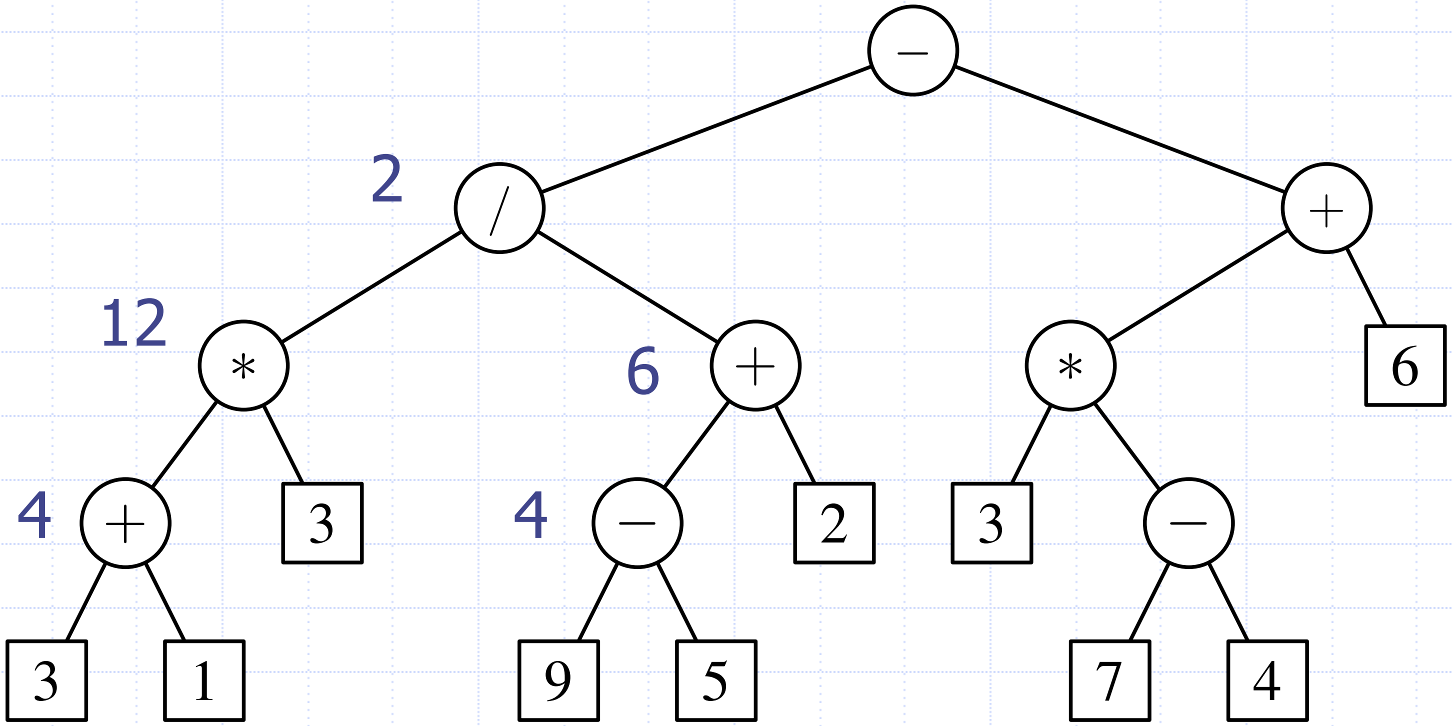
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



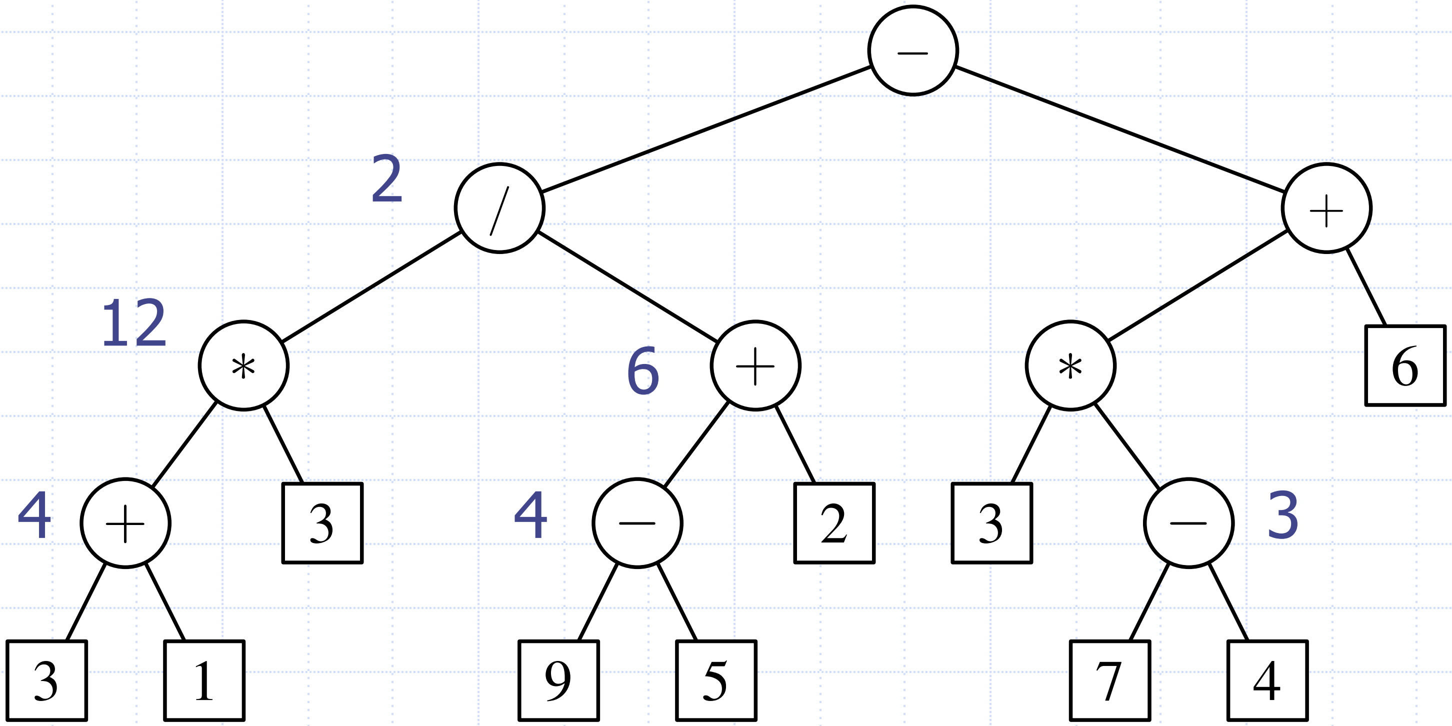
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



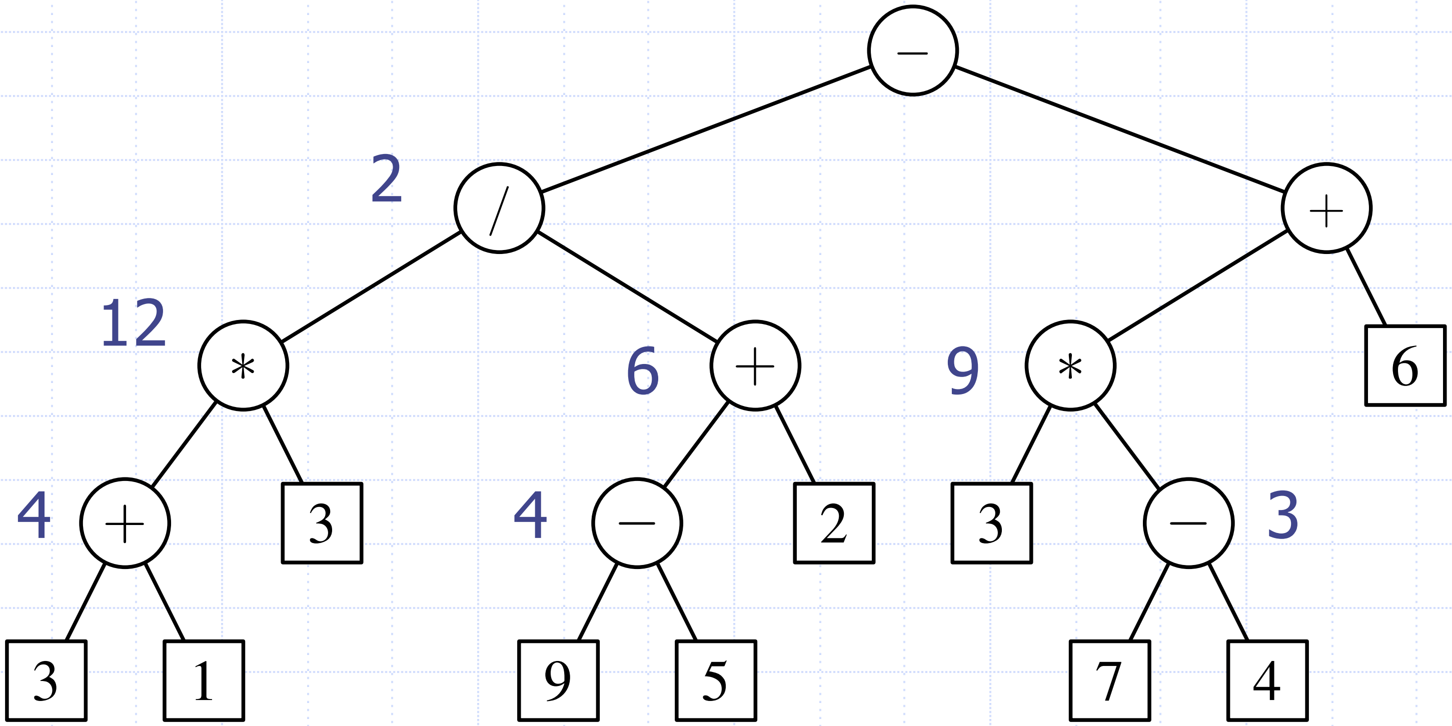
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



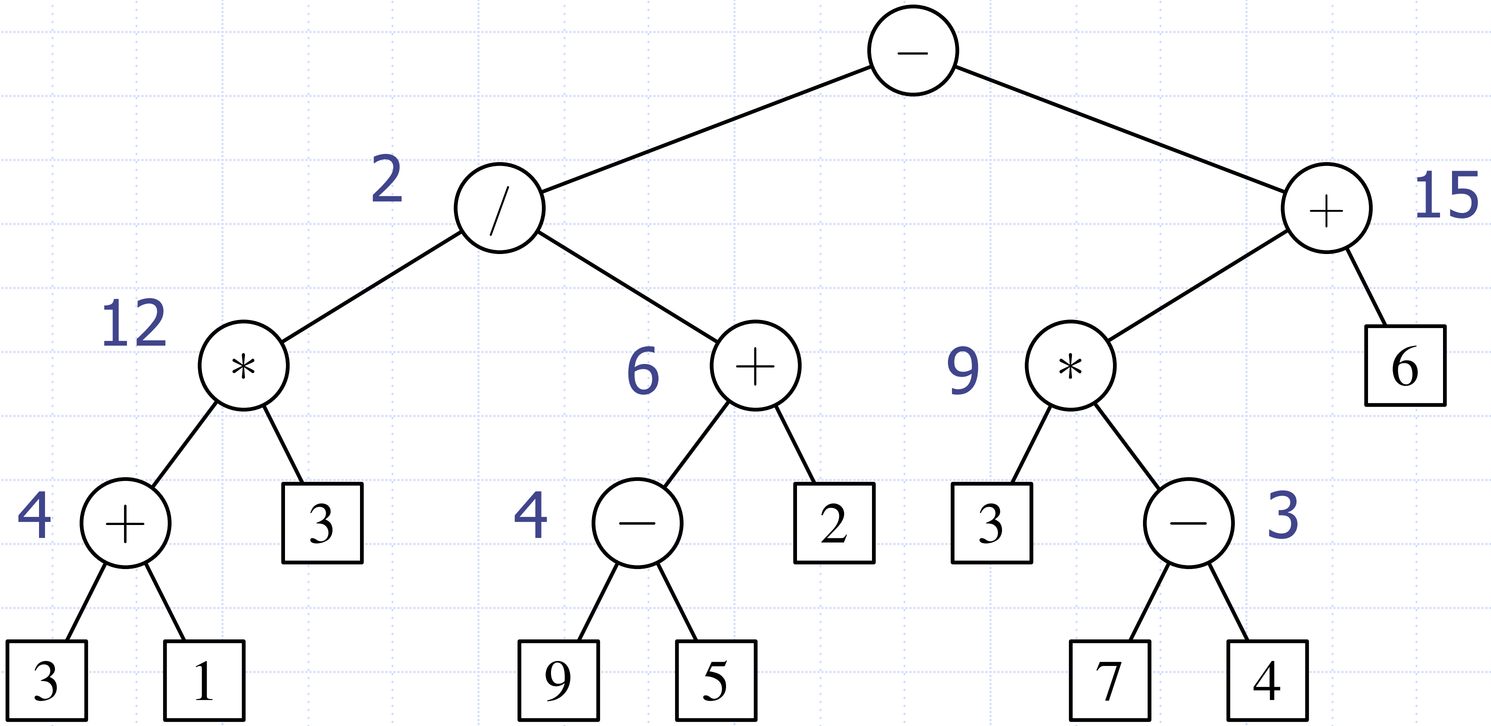
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



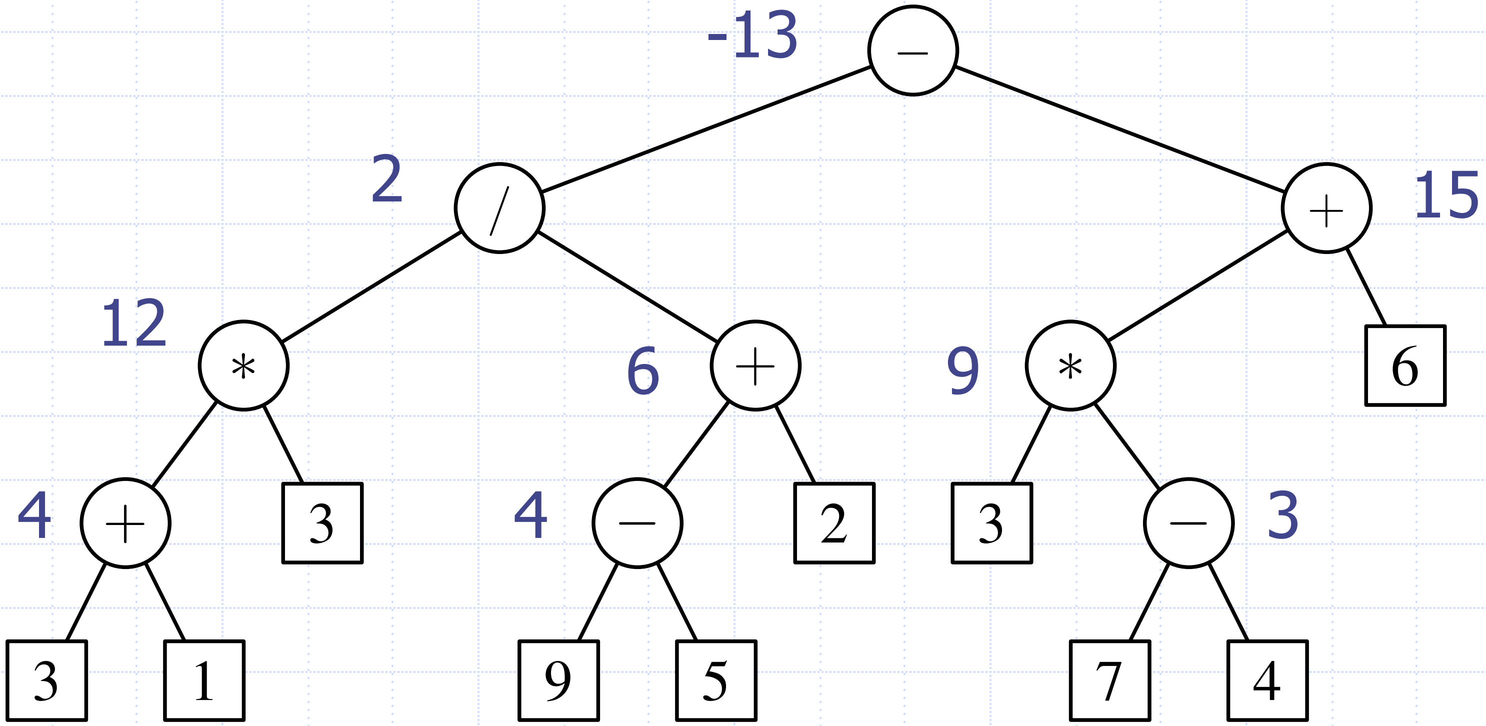
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



Application of Postorder Traversal

- Evaluating Arithmetic Expressions

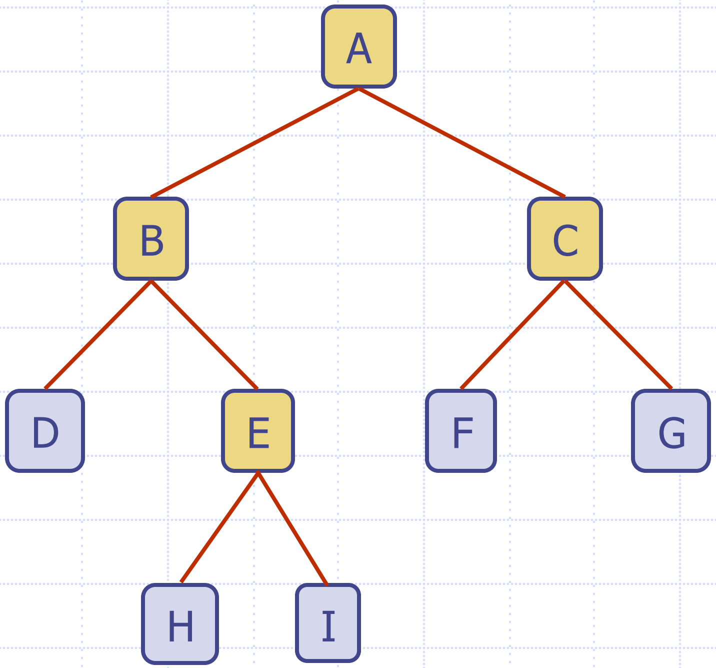


Inorder traversals

- Visit the node between the visit to the left and right subtree
- Algorithm inorder(p)
  - If p has a left child lc then
    - inorder(lc)
  - perform "visit" action for position p
  - If p has a right child rc then
    - inorder(rc)

Example - Inorder Traversal

- Inorder
  - d b h e i a f c g



Building Tree from Preorder Traversal

- Given the preorder traversal, can we uniquely determine the binary tree?

Preorder

a b d e h i c f g

Building Tree from Postorder Traversal

- Given the postorder traversal, can we uniquely determine the binary tree?

Postorder

d h i e b f g c a

Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder	Inorder
a b d e h i c f g	d b h e i a f c g

Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder	Inorder
a b d e h i c f g	d b h e i a f c g

Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

a	b	d	e	h	i	c	f	g

Inorder

d	b	h	e	i	a	f	c	g

A

Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

a	b	d	e	h	i	c	f	g
						c	f	g

Inorder

d	b	h	e	i	a	f	c	g
						f	c	g

A

Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder								
a	b	d	e	h	i	c	f	g
						c	f	g

Inorder								
d	b	h	e	i	a	f	c	g
						f	c	g

A

Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

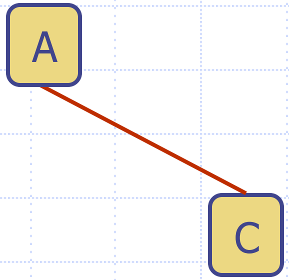
a b d e h i c f g

c f g

Inorder

d b h e i a f c g

f c g



Building Tree from Pre- and In- Order Traversals

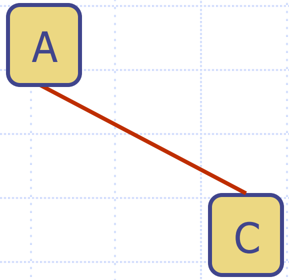
- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

a	b	d	e	h	i	c	f	g
						c	f	g

Inorder

d	b	h	e	i	a	f	c	g
						f	c	g



Building Tree from Pre- and In- Order Traversals

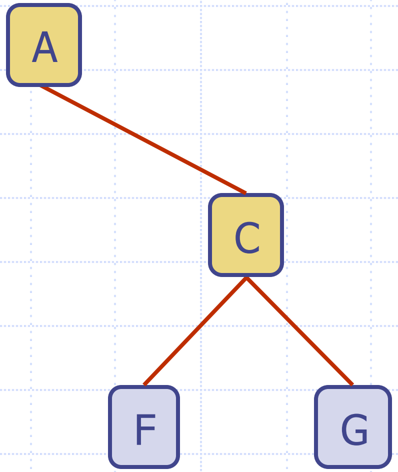
- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

a	b	d	e	h	i	c	f	g
						c	f	g

Inorder

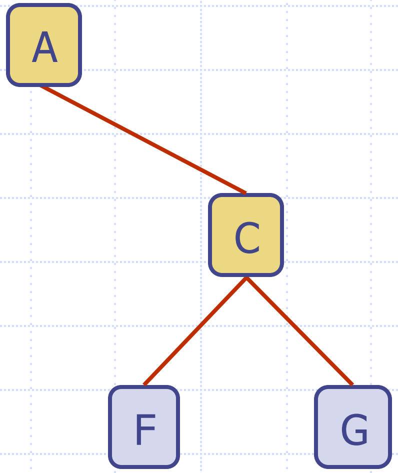
d	b	h	e	i	a	f	c	g
						f	c	g



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

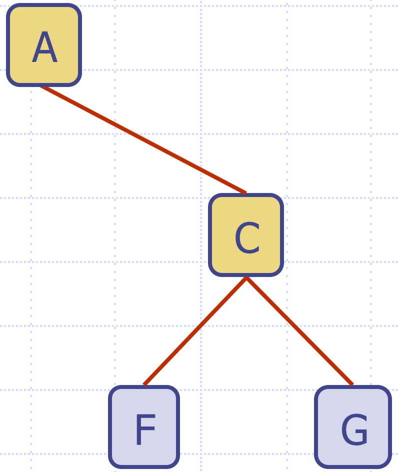
Preorder	Inorder
a b d e h i c f g	d b h e i a f c g
c f g	f c g
b d e h i	d b h e i



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

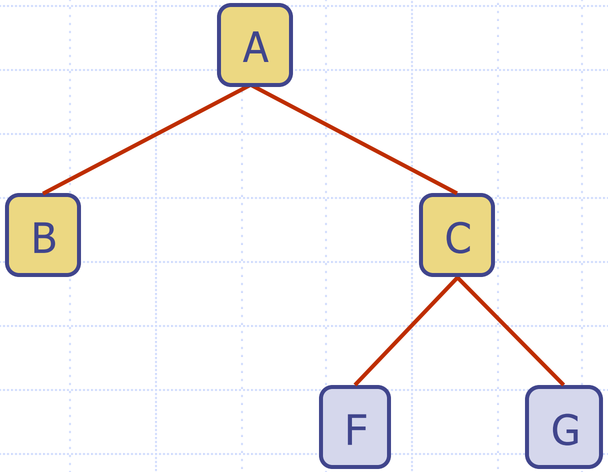
Preorder	Inorder
a b d e h i c f g	d b h e i a f c g
c f g	f c g
b d e h i	d b h e i



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

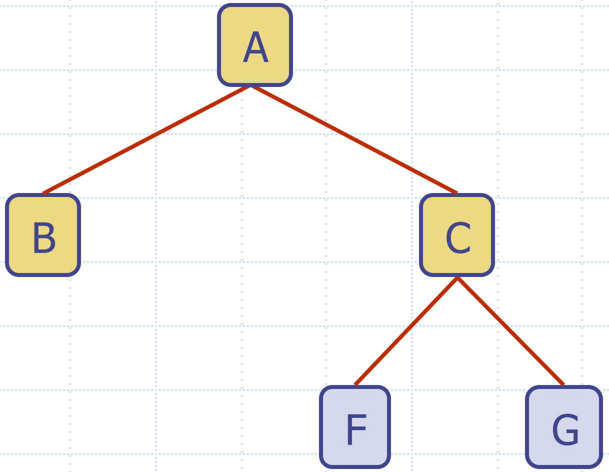
Preorder	Inorder
a b d e h i c f g	d b h e i a f c g
c f g	f c g
b d e h i	d b h e i



Building Tree from Pre- and In- Order Traversals

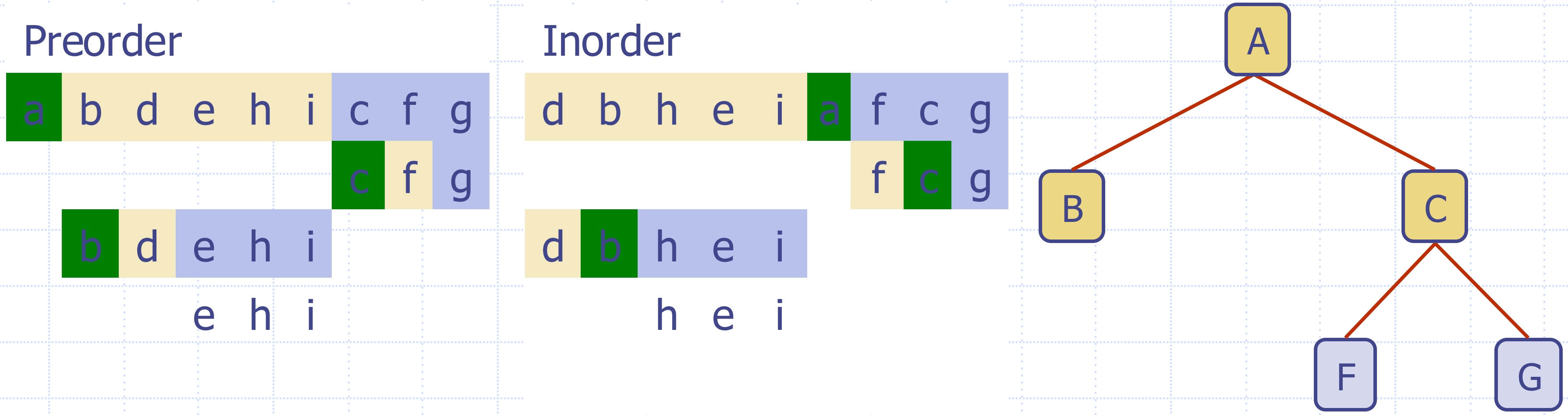
- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder	Inorder
a b d e h i c f g	d b h e i a f c g
c f g	f c g
b d e h i	d b h e i



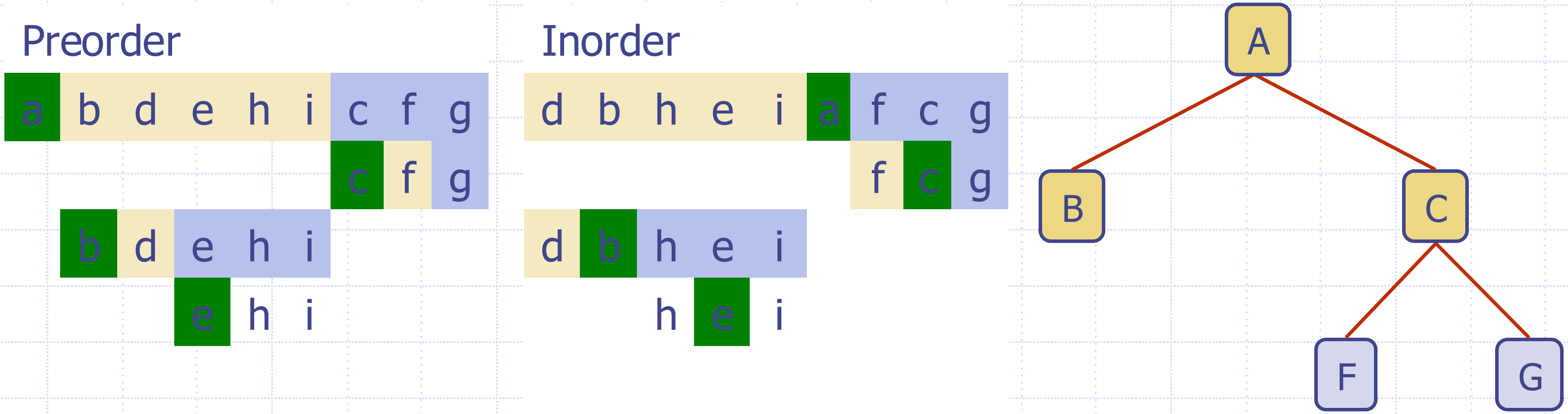
Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree



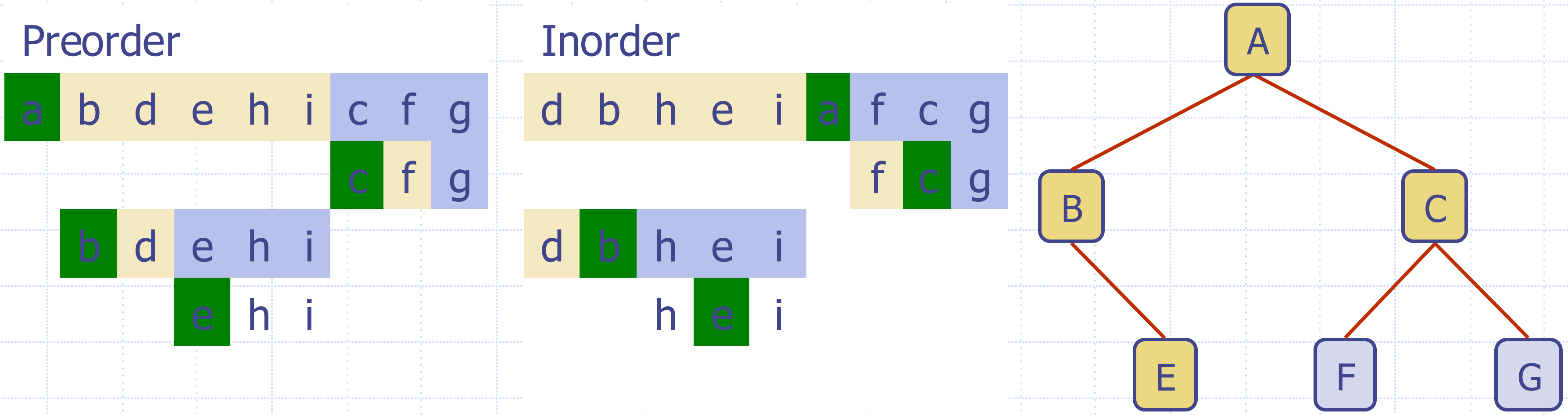
Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree



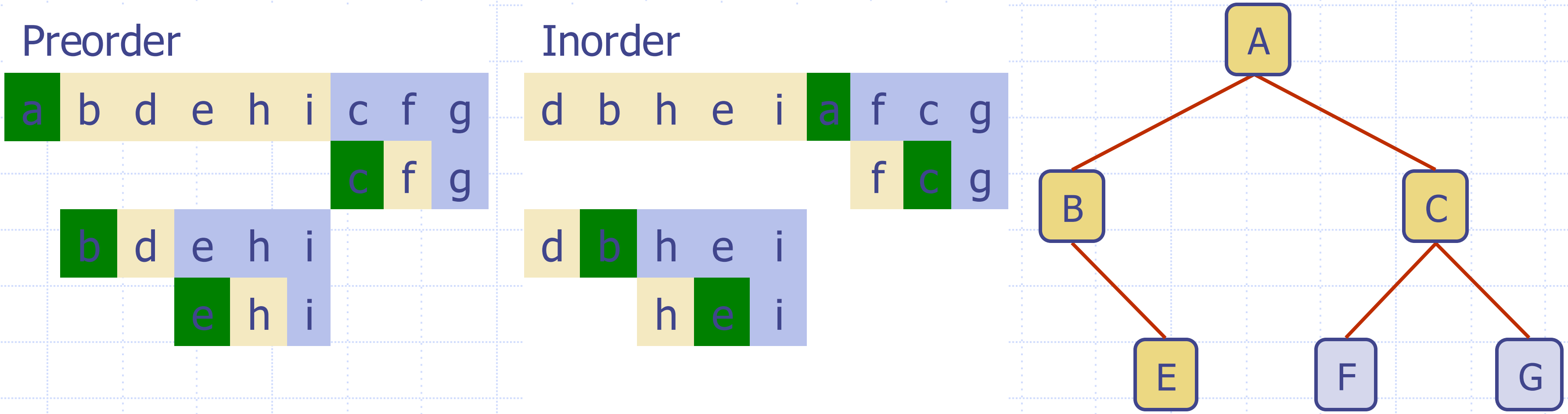
Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

a b d e h i c f g

c f g

b d e h i

e h i

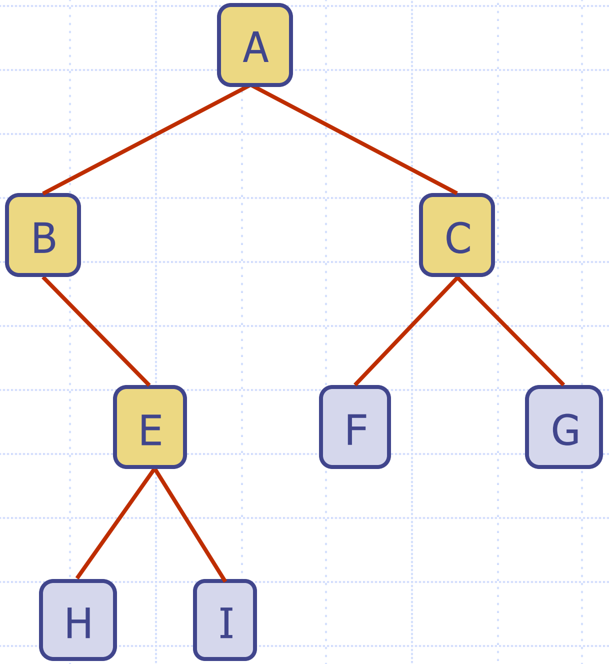
Inorder

d b h e i a f c g

f c g

d b h e i

h e i



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree

Preorder

a b d e h i c f g

c f g

b d e h i

e h i

d

Inorder

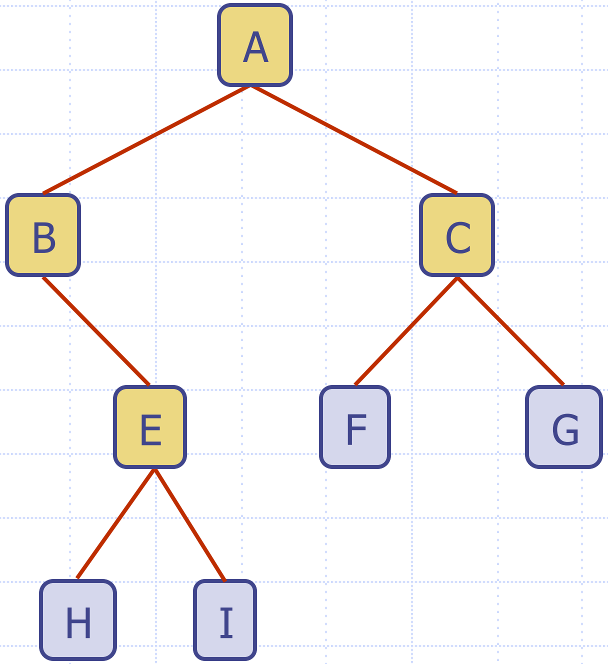
d b h e i a f c g

f c g

d b h e i

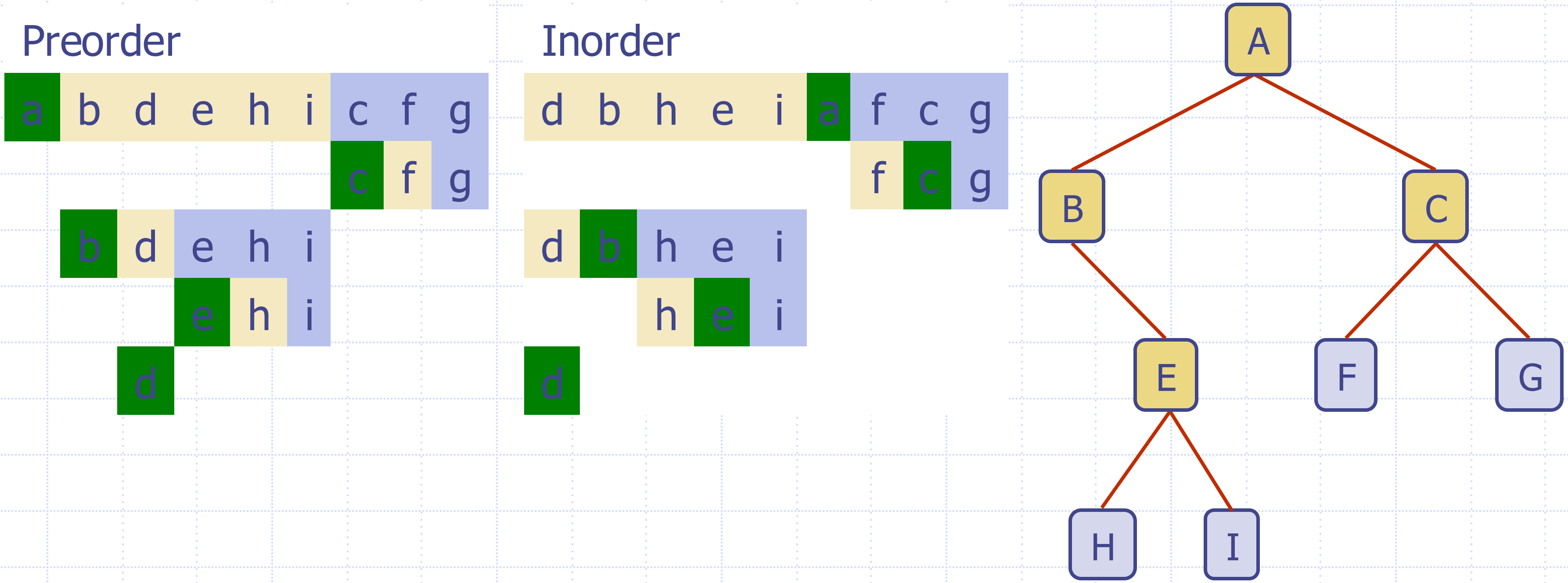
h e i

d



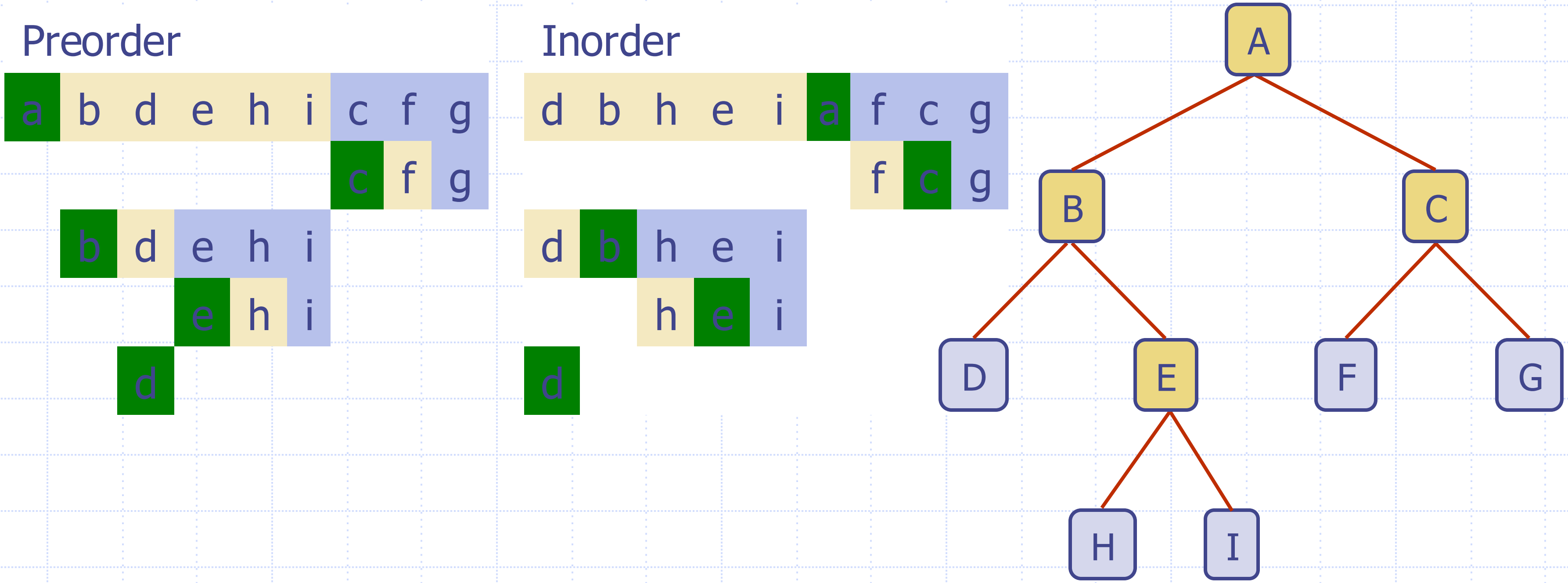
Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree



Building Tree from Pre- and In- Order Traversals

- Given the preorder and inorder traversals of a binary tree we can uniquely determine the tree



Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder	Inorder
d h i e b f g c a	d b h e i a f c g

Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder	Inorder
d h i e b f g c a	d b h e i a f c g

A

Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

d	h	i	e	b	f	g	c	a

Inorder

d	b	h	e	i	a	f	c	g

A

Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

d	h	i	e	b	f	g	c	a
					f	g	c	

Inorder

d	b	h	e	i	a	f	c	g
						f	c	g

A

Building Tree from Post- and In- Order Traversals

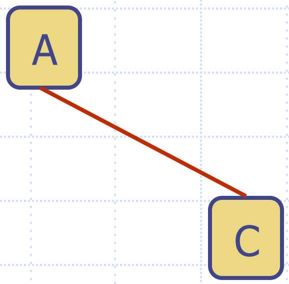
- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

d	h	i	e	b	f	g	c	a
					f	g	c	

Inorder

d	b	h	e	i	a	f	c	g
						f	c	g



Building Tree from Post- and In- Order Traversals

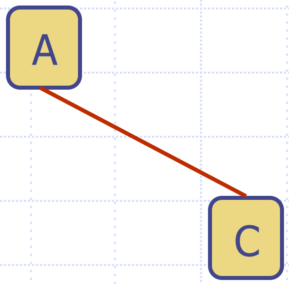
- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

d	h	i	e	b	f	g	c	a
					f	g	c	

Inorder

d	b	h	e	i	a	f	c	g
						f	c	g



Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

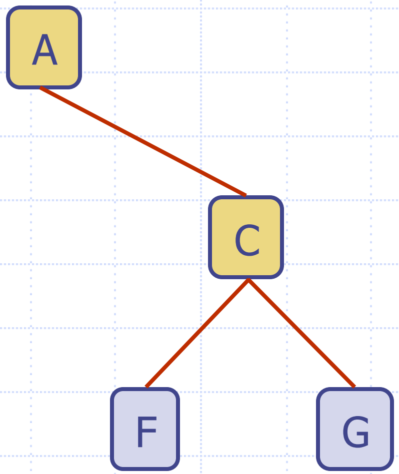
d h i e b f g c a

f g c

Inorder

d b h e i a f c g

f c g



Building Tree from Post- and In- Order Traversals

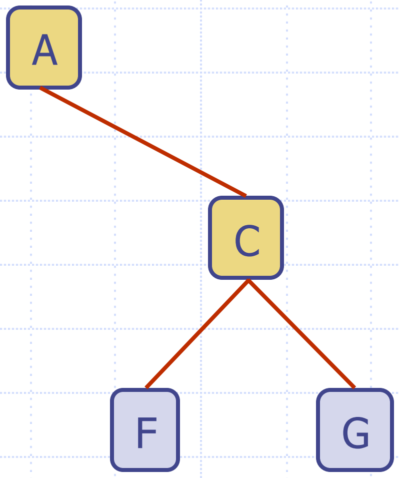
- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

d	h	i	e	b	f	g	c	a
					f	g	c	
d	h	i	e	b				

Inorder

d	b	h	e	i	a	f	c	g
						f	c	g
d	b	h	e	i				



Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree

Postorder

d h i e b f g c a

f g c

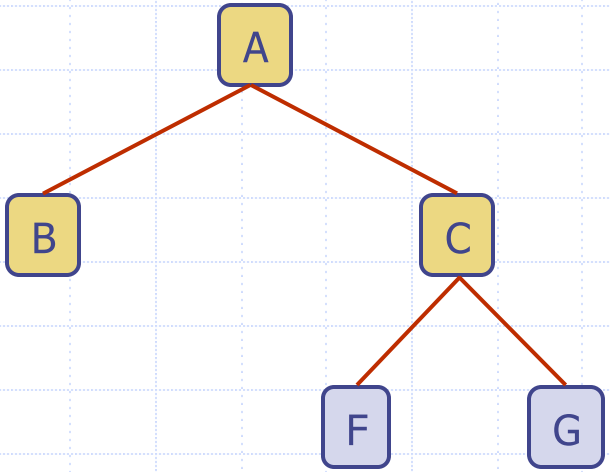
d h i e b

Inorder

d b h e i a f c g

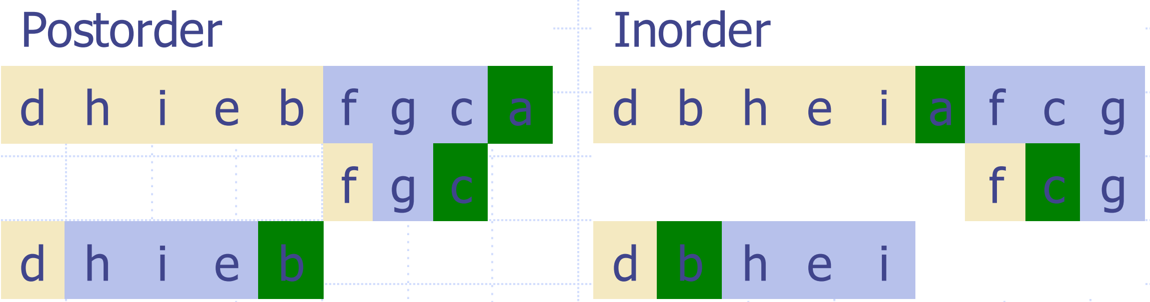
f c g

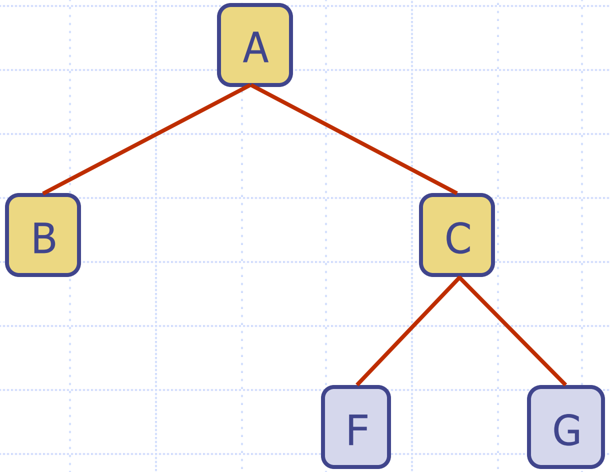
d b h e i



Building Tree from Post- and In- Order Traversals

- Given the postorder and inorder traversals of a binary tree we can uniquely determine the tree
- The last node visited in the postorder traversal is the root of the binary tree





- and so on..

Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder	Postorder
a b d e h i c f g	d h i e b f g c a

Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

a b d e h i c f g

Postorder

d h i e b f g c a

A

Building Tree from Pre- and Post- Order Traversals

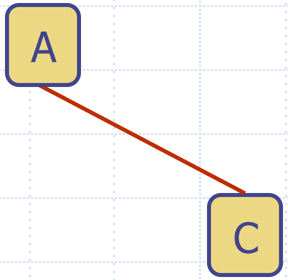
- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder									Postorder								
a	b	d	e	h	i	c	f	g	d	h	i	e	b	f	g	c	a

Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder										Postorder								
a	b	d	e	h	i	c	f	g		d	h	i	e	b	f	g	c	a
						c	f	g							f	g	c	



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

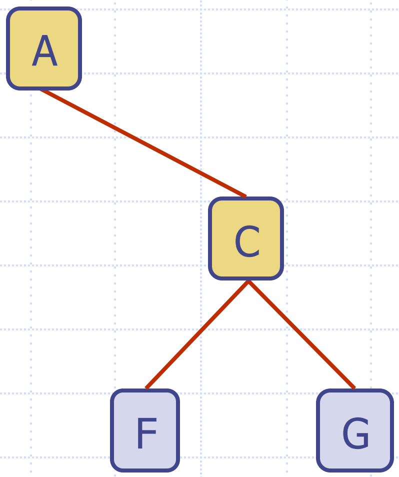
a b d e h i c f g

c f g

Postorder

d h i e b f g c a

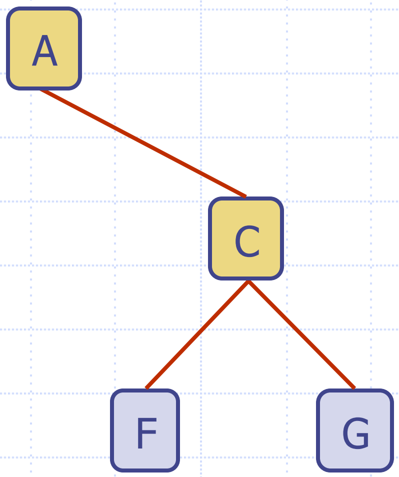
f g c



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder	Postorder
a b d e h i c f g	d h i e b f g c a
c f g	f g c
b d e h i	d h i e b



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

a b d e h i c f g

c f g

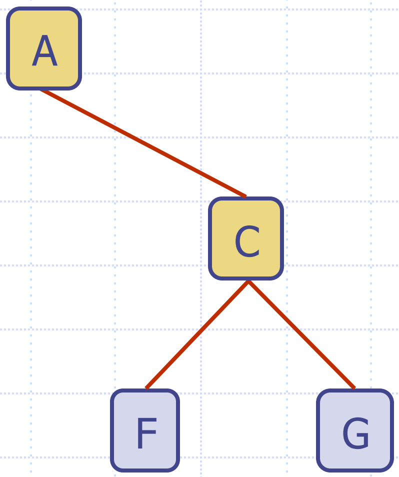
b d e h i

Postorder

d h i e b f g c a

f g c

d h i e b

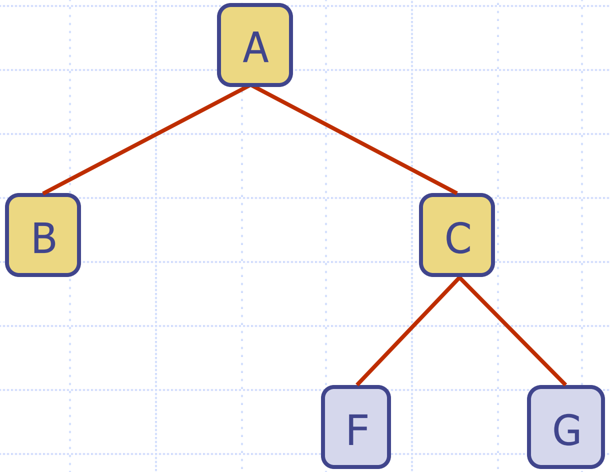


Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder								
a	b	d	e	h	i	c	f	g
						c	f	g
	b	d	e	h	i			

Postorder								
d	h	i	e	b	f	g	c	a
					f	g	c	
d	h	i	e	b				



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

a b d e h i c f g

c f g

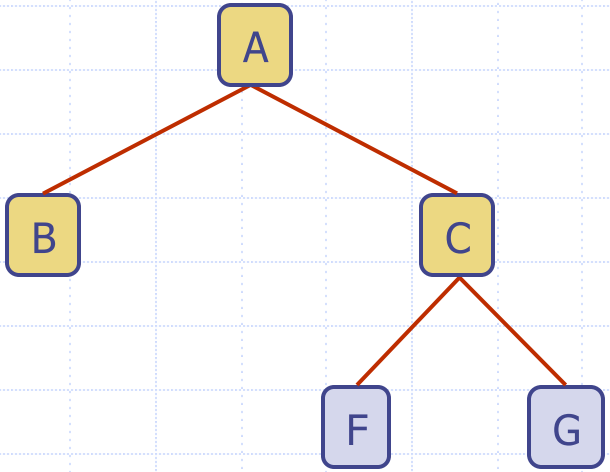
b d e h i

Postorder

d h i e b f g c a

f g c

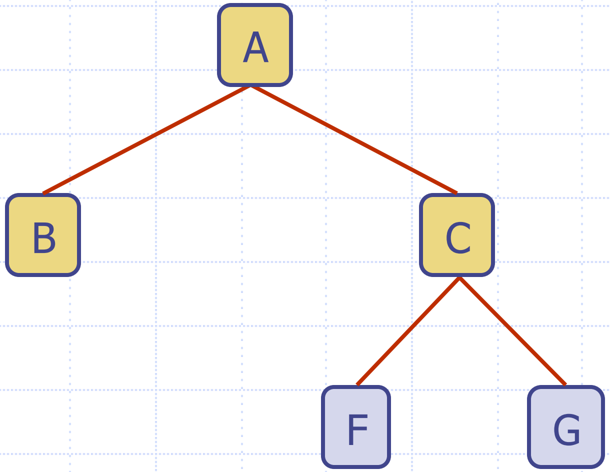
d h i e b



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

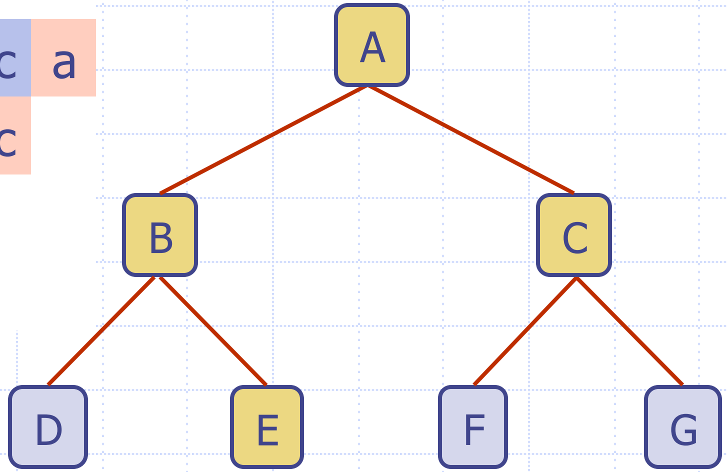
Preorder	Postorder
a b d e h i c f g	d h i e b f g c a
c f g	f g c
b d e h i	d h i e b



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder	Postorder
a b d e h i c f g	d h i e b f g c a
c f g	f g c
b d e h i	d h i e b
e h i	h i e



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

a b d e h i c f g

c f g

b d e h i

e h i

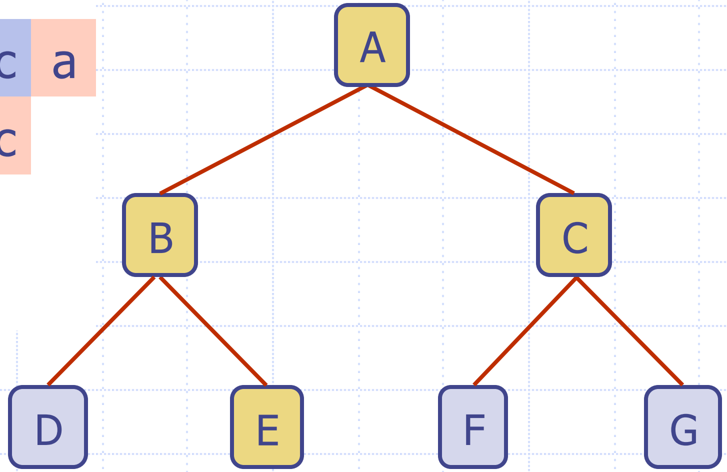
Postorder

d h i e b f g c a

f g c

d h i e b

h i e



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

a b d e h i c f g

c f g

b d e h i

e h i

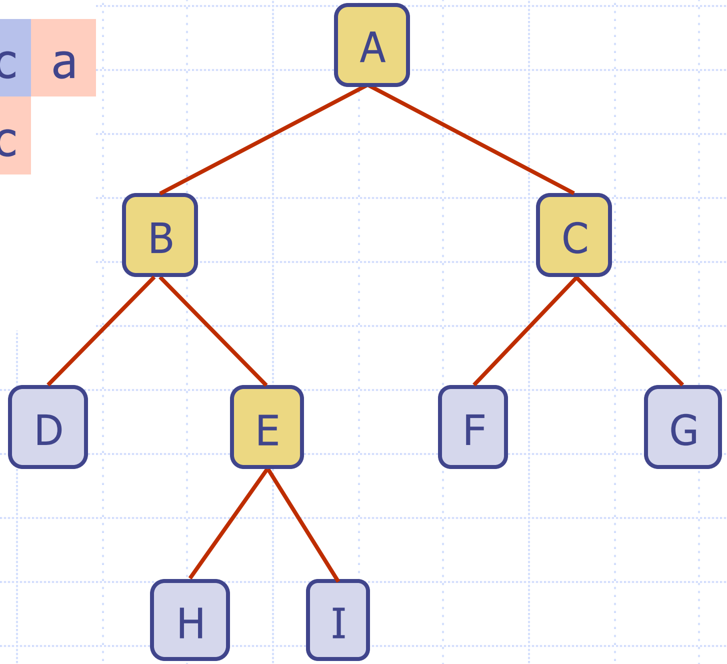
Postorder

d h i e b f g c a

f g c

d h i e b

h i e



Building Tree from Pre- and Post- Order Traversals

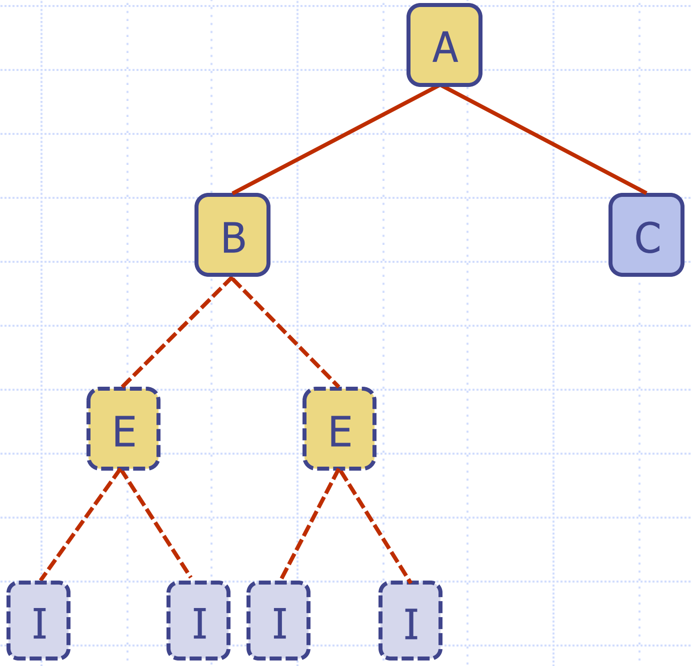
- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder	Postorder
a b e i c	i e b c a

Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder	Postorder
a b e i c	i e b c a



Building Tree from Pre- and Post- Order Traversals

- Given the pre and postorder traversal of a binary tree, can we uniquely reconstruct the tree?

Preorder

a b e i c

Postorder

i e b c a

Only if the internal nodes in a binary tree have exactly two children