



# Recursive Data Types

## Ordered Recursive Binary Trees



## Ordered Recursive Binary Trees

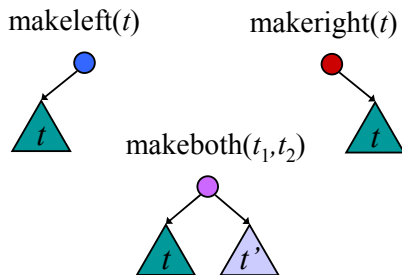
Defined recursively as follows:

- A single node ● is an RBT

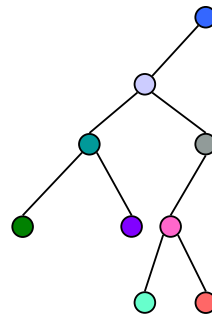


## Recursive Binary Trees

If  $t, t'$  are RBTs, then so are:



## Recursive Binary Tree



## Recursive Functions on RBT

Recursive def of set of nodes

- $\text{nodes}(\bullet) ::= \{ \bullet \}$ ,
- $\text{nodes}(\text{makeleft}(t)) ::= \text{nodes}(t) \cup \{ \circ \}$
- $\text{nodes}(\text{makeright}(t)) ::= \text{nodes}(t) \cup \{ \circ \}$
- $\text{nodes}(\text{makeboth}(t_1, t_2)) ::= \text{nodes}(t_1) \cup \text{nodes}(t_2) \cup \{ \circ \}$

where  $\circ$  is the "new" root node.



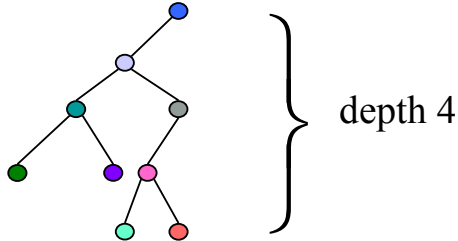
## Recursive Functions on RBT

Def. of depth

- $\text{depth}(\bullet) ::= 0$
- $\text{depth}(\text{makeleft}(t)) ::= 1 + \text{depth}(t)$
- $\text{depth}(\text{makeright}(t)) ::= 1 + \text{depth}(t)$
- $\text{depth}(\text{makeboth}(t_1, t_2)) ::= 1 + \max \{ \text{depth}(t_1), \text{depth}(t_2) \}$



## Tree Depth



## Nodes versus Depth

*Lemma:*  $|\text{nodes}(t)| + 1 \leq 2^{\text{depth}(t)+1}$

Proof by **Structural Induction**

Base Case:  $t = \bullet$

$$\begin{aligned} |\text{nodes}(t)| + 1 &= 1 + 1 = 2 = 2^{0+1} \\ &= 2^{\text{depth}(t)+1} \quad \text{OK!} \end{aligned}$$



## Nodes versus Depth

**Induction Case:**  $\text{makeleft}(t)$

**Assume:**  $|\text{nodes}(t)| + 1 \leq 2^{\text{depth}(t)+1}$

**To Prove:**

$$|\text{nodes}(\text{makeleft}(t))| + 1 \leq 2^{\text{depth}(\text{makeleft}(t))+1}$$



## Nodes versus Depth

$$|\text{nodes}(\text{makeleft}(t))| + 1$$

$$= (|\text{nodes}(t)| + 1) + 1 \quad \text{by def. of nodes}$$

$$\leq (2^{\text{depth}(t)+1} + 1) + 1 \quad \text{by ind. hyp.}$$

$$= 2^{\text{depth}(t)+1} + 2 \leq 2^{\text{depth}(t)+1} + 2^{\text{depth}(t)+1}$$

$$= 2 \cdot 2^{\text{depth}(t)+1} = 2^{(\text{depth}(t)+1)+1}$$

$$= 2^{\text{depth}(\text{makeleft}(t))+1} \quad \text{by def. of depth}$$



## Nodes versus Depth

**Induction Case:**  $\text{makeright}(t)$

**Same**



## Nodes versus Depth

**Induction Case:**  $\text{makeboth}(t_1, t_2)$

**Assume:**  $|\text{nodes}(t_1)| + 1 \leq 2^{\text{depth}(t_1)+1}$

$|\text{nodes}(t_2)| + 1 \leq 2^{\text{depth}(t_2)+1}$

**To Prove:**

$$|\text{nodes}(\text{makeboth}(t_1, t_2))| + 1$$

$$\leq 2^{\text{depth}(\text{makeboth}(t_1, t_2))+1}$$



## Nodes versus Depth

$$\begin{aligned}
& |\text{nodes}(\text{makeboth}(t_1, t_2))| + 1 \\
&= (|\text{nodes}(t_1)| + |\text{nodes}(t_2)| + 1) + 1 \quad \text{def. of nodes} \\
&= (|\text{nodes}(t_1)| + 1) + (|\text{nodes}(t_2)| + 1) \\
&\leq 2^{\text{depth}(t_1)+1} + 2^{\text{depth}(t_2)+1} \quad \text{induction hyp.} \\
&\leq 2^{\max(\text{depth}(t_1), \text{depth}(t_2))+1} + 2^{\max(\text{depth}(t_1), \text{depth}(t_2))+1} \\
&= 2^{(\max(\text{depth}(t_1), \text{depth}(t_2))+1)+1} \\
&= 2^{\text{depth}(\text{makeboth}(t_1, t_2))+1} \quad \text{def. of depth}
\end{aligned}$$

Copyright © Albert R. Meyer, 2002.

L6-2.13



## Class Problems

# Problems 1&2

Copyright © Albert R. Meyer, 2002.

L6-2.14



## Structural vs Ordinary Induction

Could **replace Structural Induction** on RBT's by **Strong Induction on size** of an RBT. (Not recommended.)

What about 18.01 Functions (F18's)?  
What is the "size" of *cosine*?

Copyright © Albert R. Meyer, 2002.

L6-2.15



## Structural vs Ordinary Induction

Could **replace Structural Induction** on F18's by **Strong Induction on size** of *derivation trees*.

(Still not recommended.)

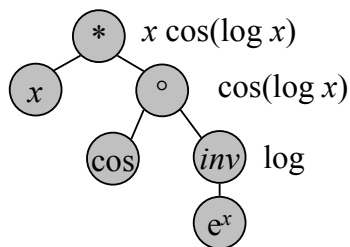
Copyright © Albert R. Meyer, 2002.

L6-2.16



## Structural vs Ordinary Induction

**derivation tree** for  $x \cdot \cos(\log(x))$



Copyright © Albert R. Meyer, 2002.

L6-2.17



## Structural vs Ordinary Induction

But Structural Induction is **also good on infinite trees**, and **can't be replaced** by induction on size.  
(... a *Meta-Theorem* of Formal Logic.)

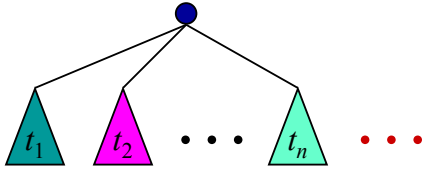
Copyright © Albert R. Meyer, 2002.

L6-2.18

6	9	13	7
12	10	5	8
3	4	14	11
15	1	16	2

### Recursive Ordered Countable Trees

- Single node is a RecCT.
- If  $t_1, t_2, \dots, t_n \dots$  are RecCT's, so is:



May have **infinitely** many subtrees

6	9	13	7
12	10	5	8
3	4	14	11
15	1	16	2

### Recursive Ordered Countable Trees

Theorem. **Every RecCT is Finite-path.**

Proof: By **structural induction** on  $t \in \text{RecCT}$ .

**Base Case** ( $t$  is one node):

- Has only a 0 length path.

6	9	13	7
12	10	5	8
3	4	14	11
15	1	16	2

### RecCT $\subseteq$ Finite-Path Trees

**Induction Step** ( $t$  has subtrees):

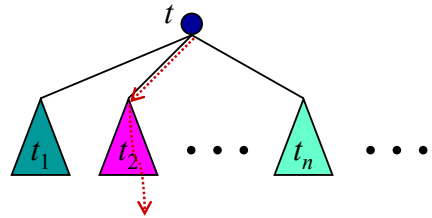
Suppose was **infinite path** from root of  $t$ .

Then would have infinite path in subtree.

6	9	13	7
12	10	5	8
3	4	14	11
15	1	16	2

### RecCT $\subseteq$ Finite-Path Trees

**Induction Step** ( $t$  has subtrees):



6	9	13	7
12	10	5	8
3	4	14	11
15	1	16	2

### RecCT $\subseteq$ Finite-Path Trees

**Induction Step** ( $t$  has subtrees):

Suppose was infinite path from root of  $t$ .

Then would have infinite path in subtree.

*By hypothesis* subtrees are Finite-Path, a contradiction. So  $t$  is F-P. **QED.**

6	9	13	7
12	10	5	8
3	4	14	11
15	1	16	2

### Class Problem

# Problem 3