VERIFYING BALANCED TREES

Zohar Manna[1] Henny B. Sipma[1] Ting Zhang[2]

[1]Department of Computer Science
Stanford University

[2]Theory Group
Microsoft Research Asia

Logical Foundations of Computer Science
June 5, 2007

Outline

Outline

Verifying High-level Data Structures

☞ What?

- Complex data structure: Trees ...
- High-level properties: Being Balanced ...
- Intricate Operations: Self-balancing ...

Verifying High-level Data Structures

☞ **What?**

- Complex data structure: Trees . . .
- High-level properties: Being Balanced . . .
- Intricate Operations: Self-balancing . . .

☞ **Why?**

- Ubiquitous in advanced programming languages
- But hard to get it right

Verifying High-level Data Structures

☞ **What?**

- Complex data structure: Trees . . .
- High-level properties: Being Balanced . . .
- Intricate Operations: Self-balancing . . .

☞ **Why?**

- Ubiquitous in advanced programming languages
- But hard to get it right

☞ **Difficulty?**

- Lost in Translation

Verifying High-level Data Structures

- What?
  - Complex data structure: Trees ...
  - High-level properties: Being Balanced ...
  - Intricate Operations: Self-balancing ...
- Why?
  - Ubiquitous in advanced programming languages
  - But hard to get it right
- Difficulty?
  - Lost in Translation
- Approach?
  - Develop decidable logics to model them directly

Verifying High-level Data Structures

- What?
  - Complex data structure: Trees . . .
  - High-level properties: Being Balanced . . .
  - Intricate Operations: Self-balancing . . .
- Why?
  - Ubiquitous in advanced programming languages
  - But hard to get it right
- Difficulty?
  - Lost in Translation
- Approach?
  - Develop decidable logics to model them directly

Get High, Stay High ☺

Outline

Our Contributions

- ☞ Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic

Our Contributions

- Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic
- Show how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples

Our Contributions

- Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic
- Show how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples
- Provide a decision procedure for automatically checking validity of the resulting verification conditions

Our Contributions

- Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic
- Show how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples
- Provide a decision procedure for automatically checking validity of the resulting verification conditions
- Generalizable to model other balanced tree structures, such as AVL trees and B-trees

OUTLINE

1. INTRODUCTION
   - Motivation
   - Our Contributions
   - Related Work and Comparison
2. MAIN TALK
   - Decidable Logic of R-B Trees
   - Analyze Algorithms on R-B Trees
3. CONCLUSION
   - Our Contributions
   - Future Work

Related Work

- Quantitative Shape Analysis [Rugina 04]
  Abstract Interpretation Performs forward propagation in an abstract heap

Related Work

- **Quantitative Shape Analysis** [Rugina 04]
  Abstract Interpretation Performs forward propagation in an abstract heap
- **Tree Automata with Size Constraints** [Habermehl et al 06]
  Automata Transformation Encodes transition relations, pre- and post-conditions as tree languages

Related Work

- Quantitative Shape Analysis [Rugina 04]
  Abstract Interpretation Performs forward propagation in an abstract heap
- Tree Automata with Size Constraints [Habermehl et al 06]
  Automata Transformation Encodes transition relations, pre- and post-conditions as tree languages
- Hypergraph Rewriting [Baldan et al 05]
  Rewriting Techniques Uses approximate unfolding to compute the reachable states of a graph rewriting system

Related Work

- Quantitative Shape Analysis [Rugina 04]
  ABSTRACT INTERPRETATION Performs forward propagation in an abstract heap
- Tree Automata with Size Constraints [Habermehl et al 06]
  AUTOMATA TRANSFORMATION Encodes transition relations, pre- and post-conditions as tree languages
- Hypergraph Rewriting [Baldan et al 05]
  REWRITING TECHNIQUES Uses approximate unfolding to compute the reachable states of a graph rewriting system
- Context Logic [Calcagno et al 05]
  DEDUCTIVE SYSTEM Proved sound and complete

Comparison

Related Work

- ✔ Express updates at an arbitrary pointed location
- ✘ Verification of Hoare triples is not fully automatic
- ✘ Lack of intuitive connections between low level program statements and the high level formalism

Comparison

Related Work

- ✔ Express updates at an arbitrary pointed location
- ✘ Verification of Hoare triples is not fully automatic
- ✘ Lack of intuitive connections between low level program statements and the high level formalism

Our Work

- ✘ Cannot express updates at an arbitrary pointed location
  Resort to induction
- ✔ Verification of Hoare triples is fully automatic
- ✔ Clear connections between low level program statements and the high level formalism

Outline

1. Introduction
   - Motivation
   - Our Contributions
   - Related Work and Comparison
2. Main Talk
   - Decidable Logic of R-B Trees
   - Analyze Algorithms on R-B Trees
3. Conclusion
   - Our Contributions
   - Future Work

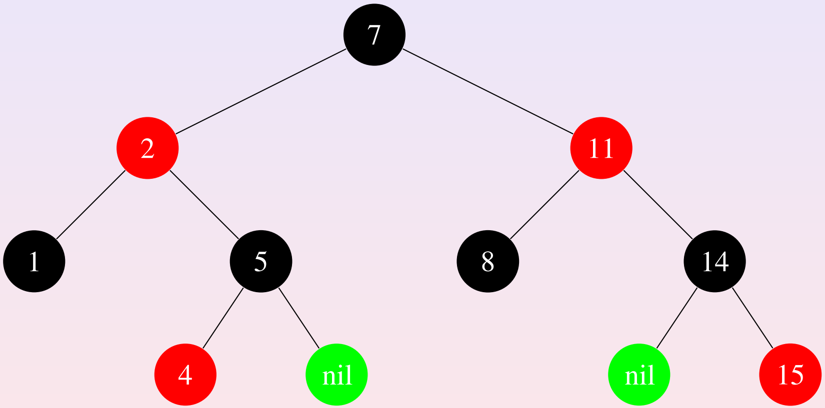
Red-black Trees

Definition (Red-black Trees)

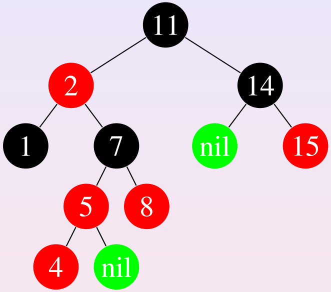
A binary tree with the following coloring properties:

1. Every node is either red or black.
2. Every leaf node is black.
3. The root is black.
4. Every red node has two black children.
5. All paths from the root to leaf nodes contain the same number of black nodes.

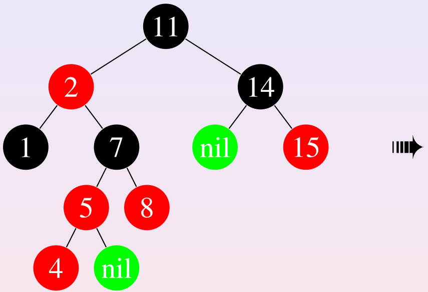
Example: Red-black Trees



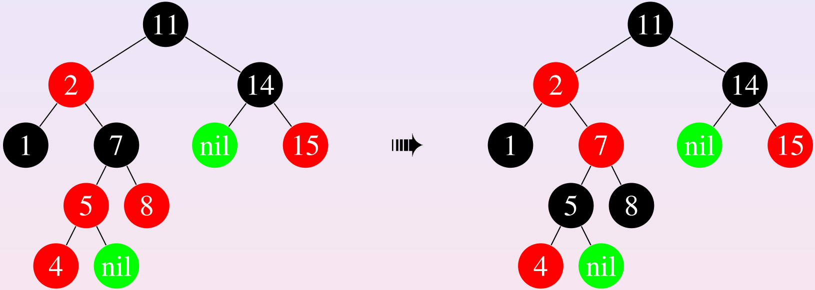
Color Flipping



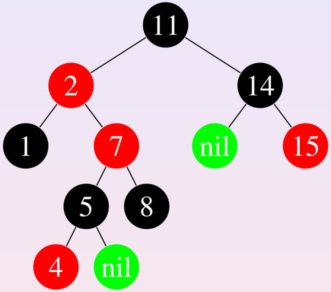
Color Flipping



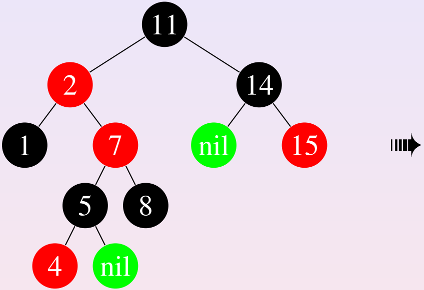
Color Flipping



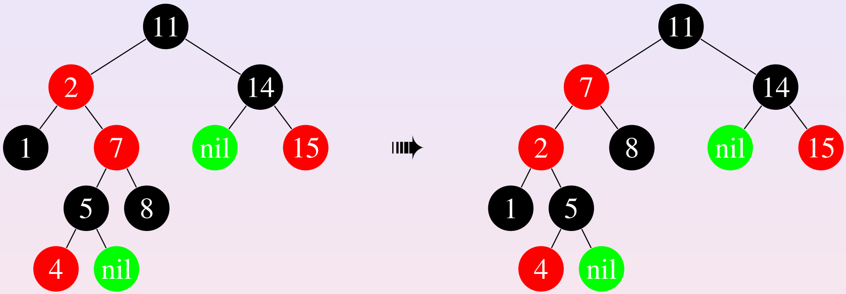
Left Rotation



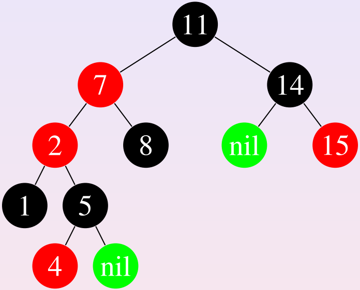
Left Rotation



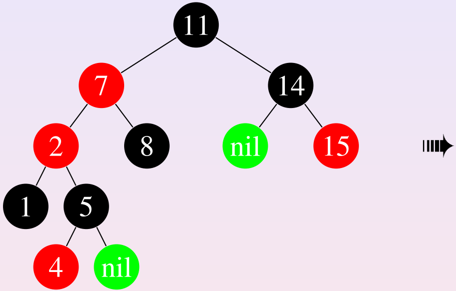
Left Rotation



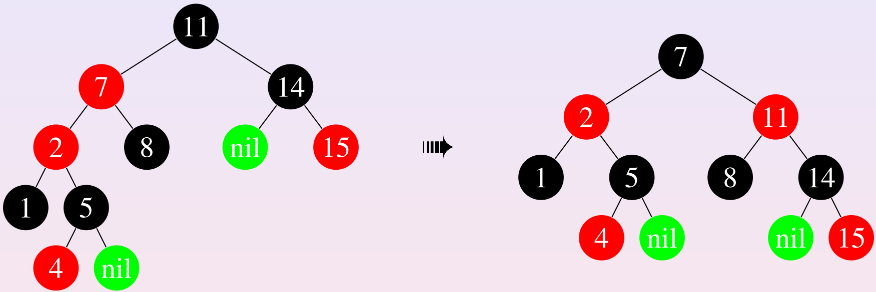
Right Rotation



Right Rotation



Right Rotation



Term Algebras

Definition (Term Algebras)

A term algebra $\mathrm{TA} : \langle \mathbb{T}; \mathcal{C}, \mathcal{A}, \mathcal{S}, \mathcal{T} \rangle$ consists of

1. $\mathbb{T}$: The term domain called $\mathcal{C}$-terms
2. $\mathcal{C}$: A set of constructors: α, β, γ, ...
3. $\mathcal{A}$: A set of constants: a, b, c, ... We require $\mathcal{A} \neq \emptyset$ and $\mathcal{A} \subseteq \mathcal{C}$.
4. $\mathcal{S}$: A set of selectors. For a constructor α with arity $k > 0$, there are k selectors $\mathsf{s}_1^\alpha, \ldots, \mathsf{s}_k^\alpha$ in $\mathcal{S}$.
5. $\mathcal{T}$: A set of testers. For each constructor α there is a corresponding tester Is_α.

Colored Trees

$$\mathrm{RB} = \langle\ \mathbb{T}_{\mathrm{rb}};\ \{\mathrm{red}, \mathrm{black}, \mathrm{nil}\},\ \ \{\mathrm{nil}\},\ \{\mathrm{car}^{\mathrm{red}}, \mathrm{cdr}^{\mathrm{red}}, \mathrm{car}^{\mathrm{black}}, \mathrm{cdr}^{\mathrm{black}}\},\ \ \{\mathrm{Is}_{\mathrm{red}}, \mathrm{Is}_{\mathrm{black}}, \mathrm{Is}_{\mathrm{nil}}\}\ \rangle\ ,$$

where

- $\mathbb{T}_{\mathrm{rb}}$ denotes the domain
- nil denotes a leaf,
- red and black are binary constructors
- $\mathrm{car}^{\sharp}$ and $\mathrm{cdr}^{\sharp}$ are the left and the right $\sharp$-selectors ($\sharp \in \{\mathrm{red}, \mathrm{black}\}$).

Red-black Trees

$$\mathrm{RB}_{\mathbb{Z}} = \langle\, \mathrm{RB};\ \mathrm{PA};\ |\cdot|_{\max}, |\cdot|_{\min} : \mathbb{T}_{\mathrm{rb}} \to \mathbb{N}\,\rangle$$

with

$|\cdot|_{\max}$: *length of maxiaml black path*

$|\cdot|_{\min}$: *length of mimimal black path*

Maximal Black Path

$$
|x|_{\max} = \begin{cases} 1 & \text{if } x \text{ is nil} \\ 0 & \text{if } x \text{ has two consecutive red nodes} \\ \max(|x_1|_{\max}, |x_2|_{\max}) + 1 & \text{if } x \text{ is a well-formed black tree} \\ \max(|x_1|_{\max}, |x_2|_{\max}) & \text{if } x \text{ is a well-formed red tree} \end{cases}
$$

Minimal Black Path

$$
|x|_{min} = \begin{cases} 1 & \text{if } x \text{ is nil} \\ 0 & \text{if } x \text{ has two consecutive red nodes} \\ \min(|x_1|_{min}, |x_2|_{min}) + 1 & \text{if } x \text{ is a well-formed black tree} \\ \min(|x_1|_{min}, |x_2|_{min}) & \text{if } x \text{ is a well-formed red tree} \end{cases}
$$

Predicates for Well-formed Trees

x is a well-formed black tree:

$$\mathrm{GB}(x, x_1, x_2) \stackrel{\text{def}}{=\!=} x = \mathrm{black}(x_1, x_2) \ \wedge\ |x_1|_{\max} \neq 0 \ \wedge\ |x_2|_{\max} \neq 0$$

x is a well-formed red tree:

$$\mathrm{GR}(x, x_1, x_2) \stackrel{\text{def}}{=\!=} x = \mathrm{red}(x_1, x_2) \ \wedge\ |x_1|_{\max} \neq 0 \ \wedge\ |x_2|_{\max} \neq 0$$

x has two consecutive red nodes:

$$\mathrm{Vio}(x) \stackrel{\text{def}}{=\!=} x \neq \mathrm{nil} \ \wedge\ \forall x_1 \forall x_2 \left(\neg \mathrm{GB}(x, x_1, x_2) \ \vee\ \neg \mathrm{GR}(x, x_1, x_2)\right)$$

Red-black Properties

x is a red black tree if

φ_1 :	$\lvert x\rvert_{max} = \lvert x\rvert_{min}$	any maximal path of x contains the same number of black nodes
φ_2 :	$\lvert x\rvert_{max} > 0$	any red node of x must have two black children
φ_3 :	$\mathrm{Is}_{black}(x)$	the root of x is black

Red-black Properties

Subdomain predicate:

$$\varphi_{RB}(x) \stackrel{\text{def}}{=\!=} \varphi_1 \ \wedge \ \varphi_2 \ \wedge \ \varphi_3$$

Theory of the subdomain obtained by relativization:

$\forall x \, (\varphi_{RB}(x) \rightarrow \Phi(x))$ *for universal properties*

$\exists x \, (\varphi_{RB}(x) \wedge \Phi(x))$ *for existential properties*

Decidability of $RB_{\mathbb{Z}}$

Theorem (Decidability of $RB_{\mathbb{Z}}$)

1. $\mathrm{Th}^{\exists}(RB_{\mathbb{Z}})$ *is NP-complete.*
2. $\mathrm{Th}(RB_{\mathbb{Z}})$ *is decidable and admits quantifier elimination.*

Proof Sketch.

1. Reduce term constraints to integer constraints
2. Reduce term quantifiers to integer quantifiers

Outline

1. Introduction
   - Motivation
   - Our Contributions
   - Related Work and Comparison
2. Main Talk
   - Decidable Logic of R-B Trees
   - Analyze Algorithms on R-B Trees
3. Conclusion
   - Our Contributions
   - Future Work

Transition Relation

Notation

- ☞ $\bar{v}$: variables in the current state
- ☞ $\bar{v}'$: the corresponding variables in the next state.
- ☞ $\rho_q(\bar{v}, \bar{v}')$: transition relation of a statement q
- ☞ $post(q, \varphi)$: post-condition of $\varphi(\bar{v})$ after executing a statement q

Composition

The transition relation of the composite statement $\langle q; r \rangle$ is

$$(\exists \bar{v}^1) \left(\rho_q(\bar{v}, \bar{v}^1) \ \wedge \ \rho_r(\bar{v}^1, \bar{v}') \right)$$

Verification Conditions

Hoare Triples

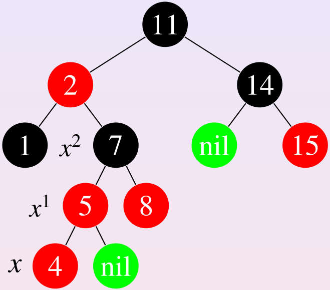
- $\{\varphi\}q\{\psi\}$: state ψ reached after executing q at state φ
- $\{\varphi\}q\{\psi\}$: equivalent to $post(q,\varphi) \rightarrow \psi$

Proving Hoare Triples

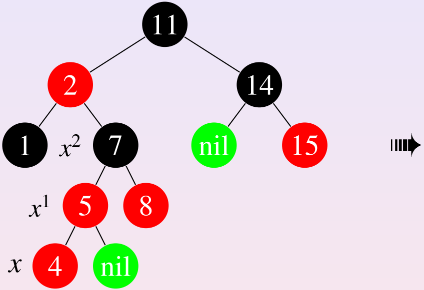
$$post(q,\varphi) \stackrel{\text{def}}{=\!=} (\exists \bar{v}^0)\left(\, \rho_q(\bar{v}^0,\bar{v}) \;\wedge\; \varphi(\bar{v}^0)\,\right)$$

$$\{\varphi\}q\{\psi\} \stackrel{\text{def}}{=\!=} (\exists \bar{v}^0)\left(\, \rho_q(\bar{v}^0,\bar{v}) \;\wedge\; \varphi(\bar{v}^0)\,\right) \;\rightarrow\; \psi(\bar{v})$$

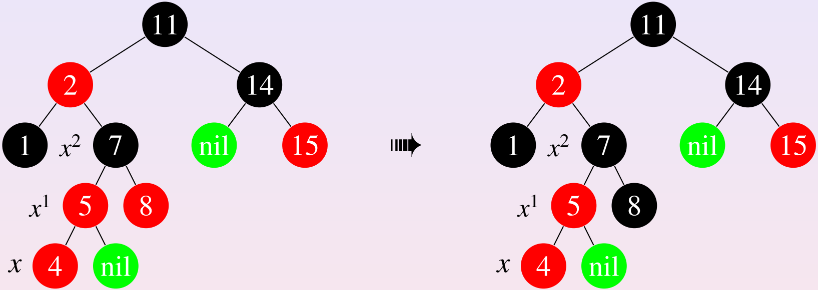
Color Flipping: Step 1



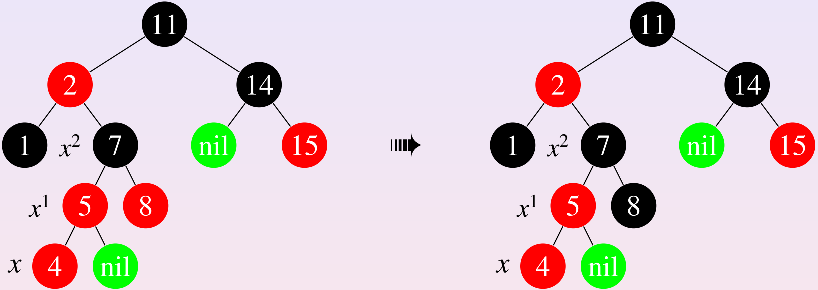
Color Flipping: Step 1



Color Flipping: Step 1

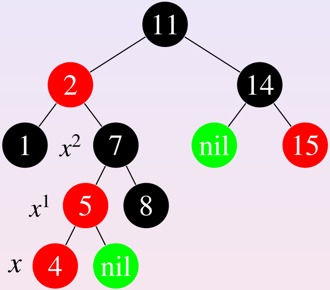


Color Flipping: Step 1

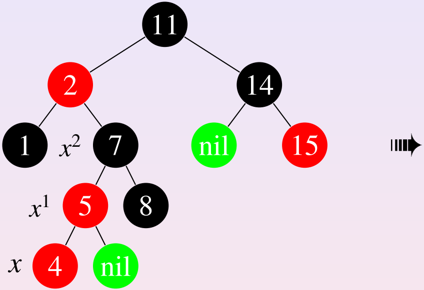


$$\begin{aligned} T'[x-1].tree &= \mathrm{cdr}(T'[x-2]) \\ &= \mathrm{black}(\mathrm{car}(T[x-1].tree), \mathrm{cdr}(T[x-1].tree)) \end{aligned}$$

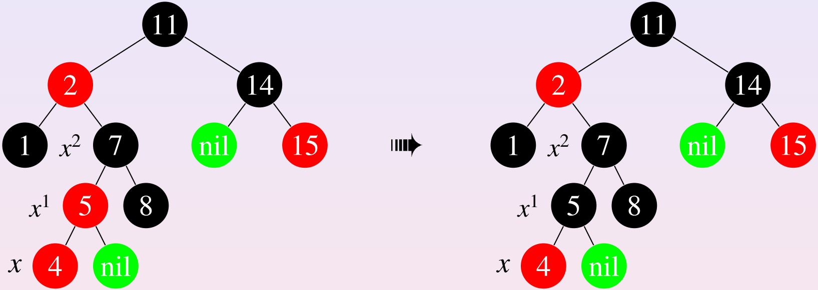
Color Flipping: Step 2



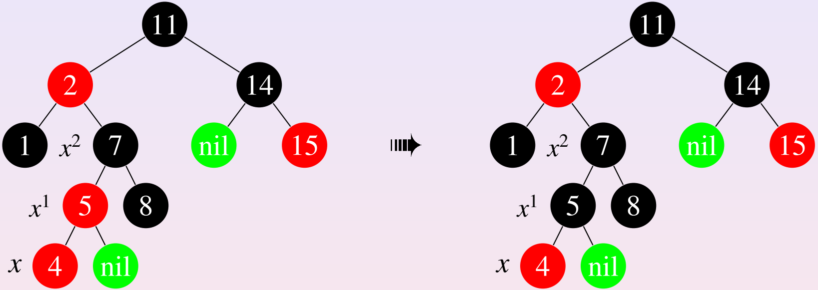
Color Flipping: Step 2



Color Flipping: Step 2

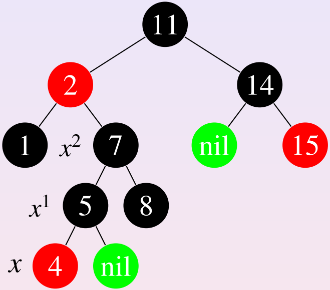


Color Flipping: Step 2

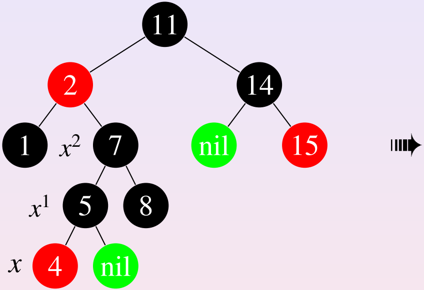


$$\text{car}(T'[x-2]) = T'[x-1] = \text{black}(\text{car}(T[x-1]), \text{cdr}(T[x-1]))$$

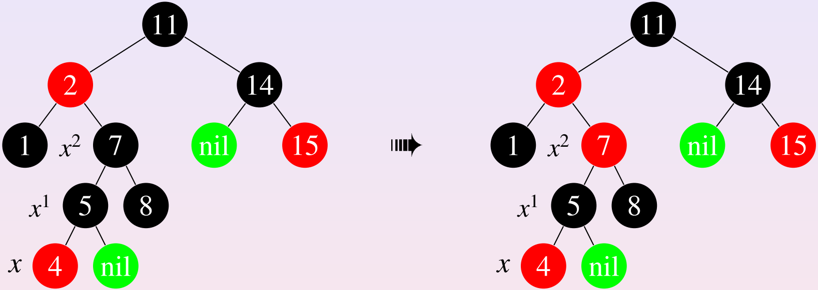
Color Flipping: Step 3



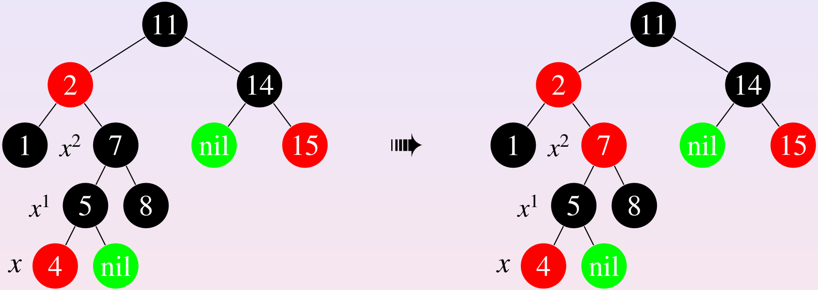
Color Flipping: Step 3



Color Flipping: Step 3

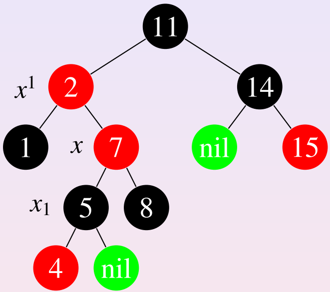


Color Flipping: Step 3

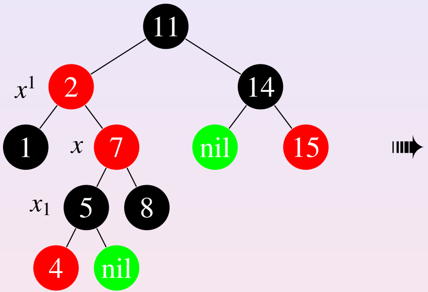


$$T'[x-2] = \text{red}(\text{car}(T[x-2]), \text{cdr}(T[x-2]))$$

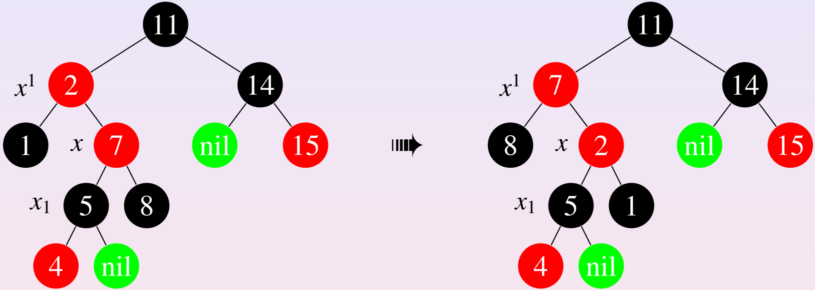
Left Rotation: Step 1



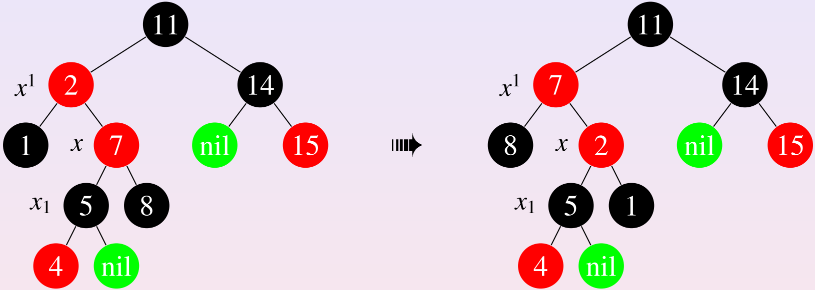
Left Rotation: Step 1



Left Rotation: Step 1

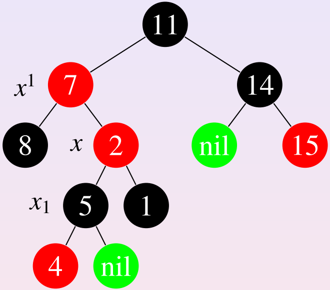


Left Rotation: Step 1

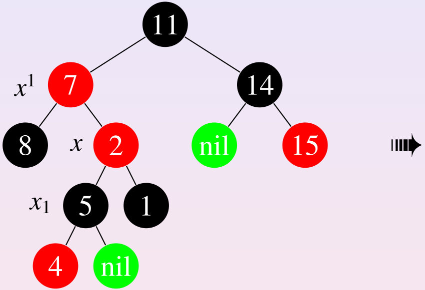


$$
\begin{aligned}
& \mathrm{cdr}(T'[x-1]) = T'[x] \\
\wedge\ & (T'[x+1].\mathit{tree} = \mathrm{cdr}(T'[x]) = T[x].\mathit{tree}) \\
\wedge\ & (T'[x].\mathit{tree} = \mathrm{car}(T'[x-1]) = T[x+1].\mathit{tree})
\end{aligned}
$$

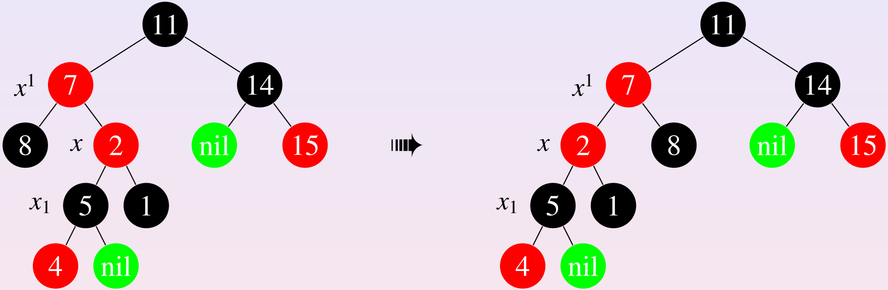
Left Rotation: Step 2



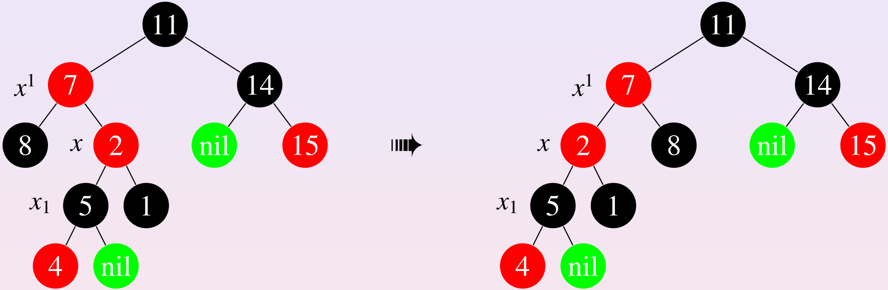
Left Rotation: Step 2



Left Rotation: Step 2

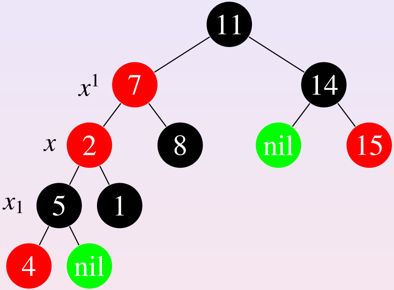


Left Rotation: Step 2

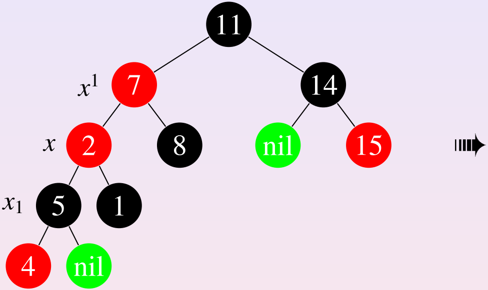


$$T'[x].dir = right \ \wedge \ T'[x-1] = \mathrm{red}(\mathrm{cdr}(T[x-1]), \mathrm{car}(T[x-1]))$$

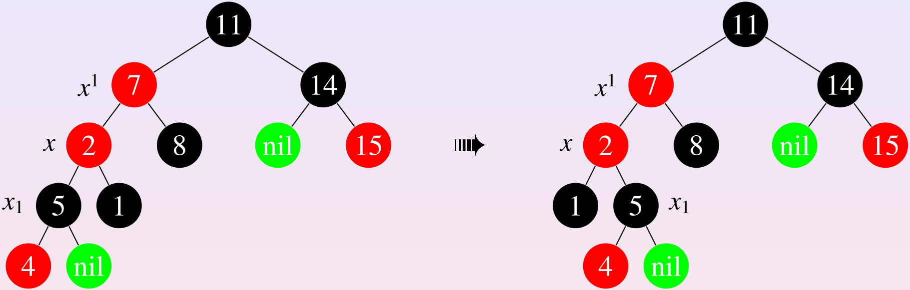
Left Rotation: Step 3



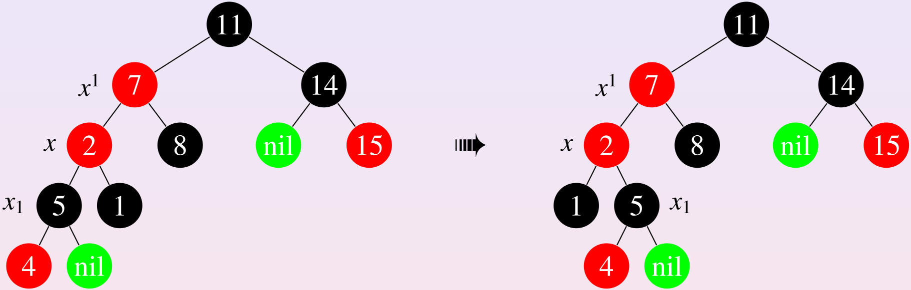
Left Rotation: Step 3



Left Rotation: Step 3



Left Rotation: Step 3



$$T'[x+1].dir = left \ \wedge \ \mathrm{car}(T'[x-1]) = T'[x]$$
$$\wedge \ T'[x] = \mathrm{red}(\mathrm{cdr}(T[x]), \mathrm{car}(T[x]))$$

Outline

Our Contributions

- Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic

Our Contributions

- Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic
- Show how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples

Our Contributions

- Develop a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic
- Show how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples
- Provide a decision procedure for automatically checking validity of the resulting verification conditions

Our Contributions

- **Develop** a first-order theory of red-black trees using the theory of term algebras augmented with Presburger arithmetic
- **Show** how to use this theory to represent the transition relations of the tree operations directly from the program statements, and how to use them to construct Hoare triples
- **Provide** a decision procedure for automatically checking validity of the resulting verification conditions
- **Generalizable** to model other balanced tree structures, such as AVL trees and B-trees

Outline

Future Work

☞ Express more properties:
Tree Orderings

Future Work

- **Express more properties**:
  Tree Orderings
- **Model Destructive Updates**:
  Decidable Logic with Extraction and Assignment

$$T[p] \stackrel{\text{def}}{=\!=} \textit{the subtree of } T \textit{ at position } p$$

$$T \oplus_p T' \stackrel{\text{def}}{=\!=} \textit{the tree obtained from } T \textit{ by substituting } T' \textit{ for the subtree of } T \textit{ at position } p$$

Thank You!