Balanced Binary Search Trees

Calvin Higgins

Department of Computer Science and Statistics
University of Rhode Island

November 11, 2025

Balancing via Pattern Matching Transformations

Red-Black Tree Insert:

1. Binary search tree insert (red node).
2. Fix red-black tree property violations.

Which properties could be violated after step 1?

1. Every node is either red or black.
2. Red nodes do not have red children.
3. Every path from a given node to any of its leaf nodes goes through the same number of black nodes.
4. Every node obeys the binary search tree property.
5. The root is black.

Balancing via Pattern Matching Transformations

Red-Black Tree Insert:

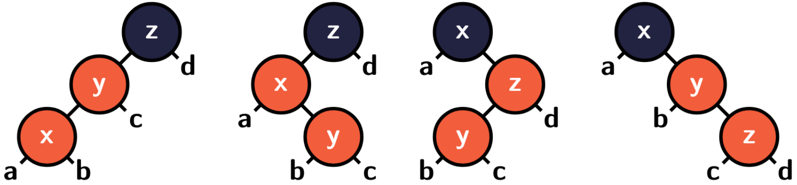
1. Binary search tree insert (red node).
2. Fix red-black tree property violations.

Which properties could be violated after step 1?

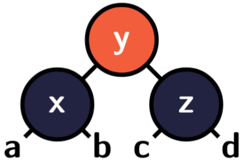
1. Every node is either red or black.
2. **Red nodes do not have red children.**
    1. **Hard. Fixing a red-red violation might introduce new violations.**
3. Every path from a given node to any of its leaf nodes goes through the same number of black nodes.
4. Every node obeys the binary search tree property.
5. **The root is black.**
    1. **Easy. Just make the root black.**

Balancing via Pattern Matching Transformations

Potential red-red violations:



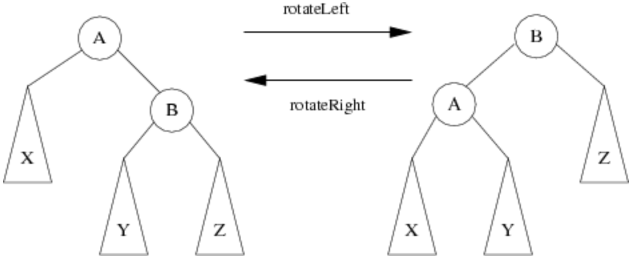
Solution (y might be a red-red violation):



`https://ccs.neu.edu/~camoy/pub/red-black-tree.pdf`

Implementing Pattern Matching Transformations

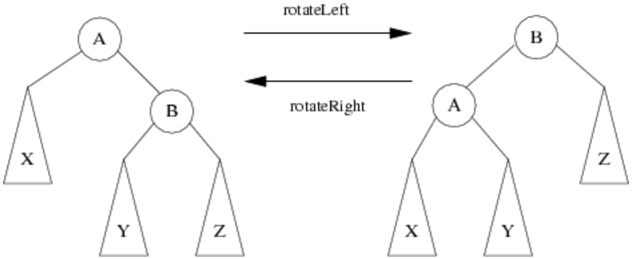
Idea: Disassemble, Then Reassemble The Tree



```
Node* rotate_left(Node* root) {
    auto a = root;
    auto x = a->left;    auto b = a->right;
    auto y = b->left;    auto z = b->right;
    a->left = x;         a->right = y;
    b->left = a;         b->right = z;
    return b;
}
```

Implementing Pattern Matching Transformations

Idea: Remove Redundant Code



```
Node* rotate_left(Node* a) {
    auto b = a->right;
    auto y = b->left;
    a->right = y;
    b->left = a;
    return b;
}
```

Warmup Problem

With Your Group:

- **Program the `fix_red_left_left` function in `pseudoset.cpp`.**
  - If needed, lab slides are available on GitHub.
- **Analyze the time complexity:**
  - Assume an input of size n.
  - Give an asymptotic bound on the time complexity of your algorithm.

Write your group's work on the whiteboards!

Lab Directions

Begin These Now:

- **Consider attending the review session:**
    - This Friday at 2:00-4:00PM in Engineering 040
    - Ask your group members if they are going!
- **Create a new project in your IDE for Lab 10**
    - If you aren't sure how to do this
        - Ask a group member
        - Search for documentation
        - Chat with AI
    - If you are still stuck, call over a staff member
- **Work through the lab handout**
    - Available on GitHub under labs/lab-10
    - `https://github.com/URI-CSC/212-fall-2015`
        - Yes, it is 2015 not 2025
    - All directions available in the lab handout