Lecture 24: Balanced Binary Search Trees

CS 62

Fall 2017

Kim Bruce & Alexandra Papoutsaki

Friday Quiz

- Ordered Structures
- Binary Search Trees
- Splay trees from today!

Removing nodes in BSTs

- Calling `remove(E val)` removes node with value `val`
- Predecessor of root becomes new root
  - Predecessor is in left subtree
  - Predecessor has no right subtree

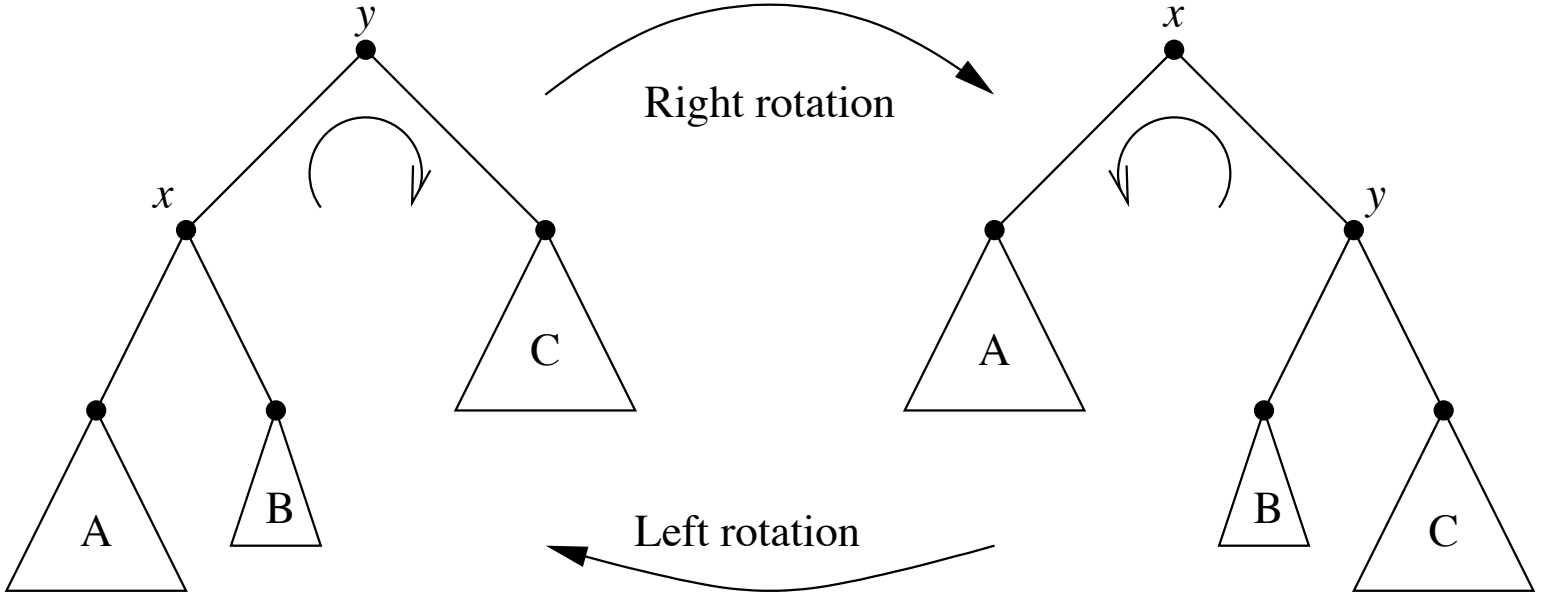
  - Complexity is O(h) where h is height of tree
    - Worst-case O(h) to locate
    - Worst-case O(h) to find predecessor

Complexity

- `locate`, `add`, `contains`, `remove` are all O(h)
- Can we guarantee that h is O(log n)?
  - Only if tree stays balanced!!
- Binary search trees that stay balanced
  - AVL trees
  - Red-black trees
- We'll do splay tree, which doesn't guarantee balance
    - but guarantees good average behavior
    - easier to understand than alternatives
    - better than others if likely to go back to recent nodes

Rotating Trees

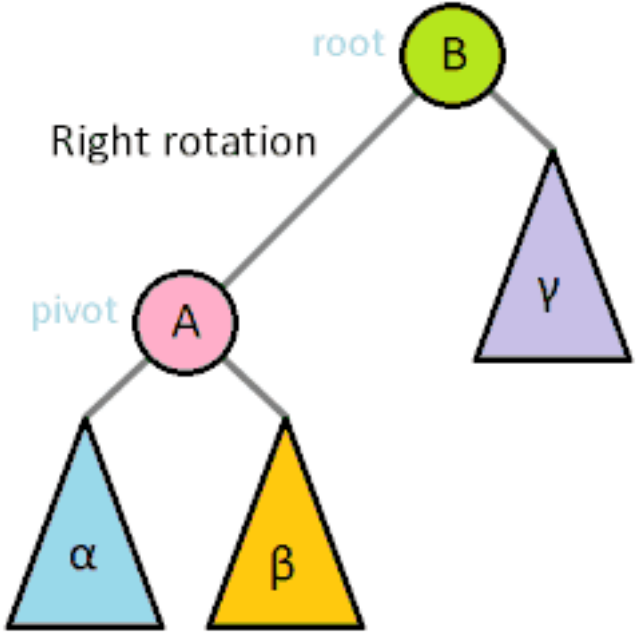
Key idea: Rotate node higher in tree while keeping it in order.



Rotating Trees

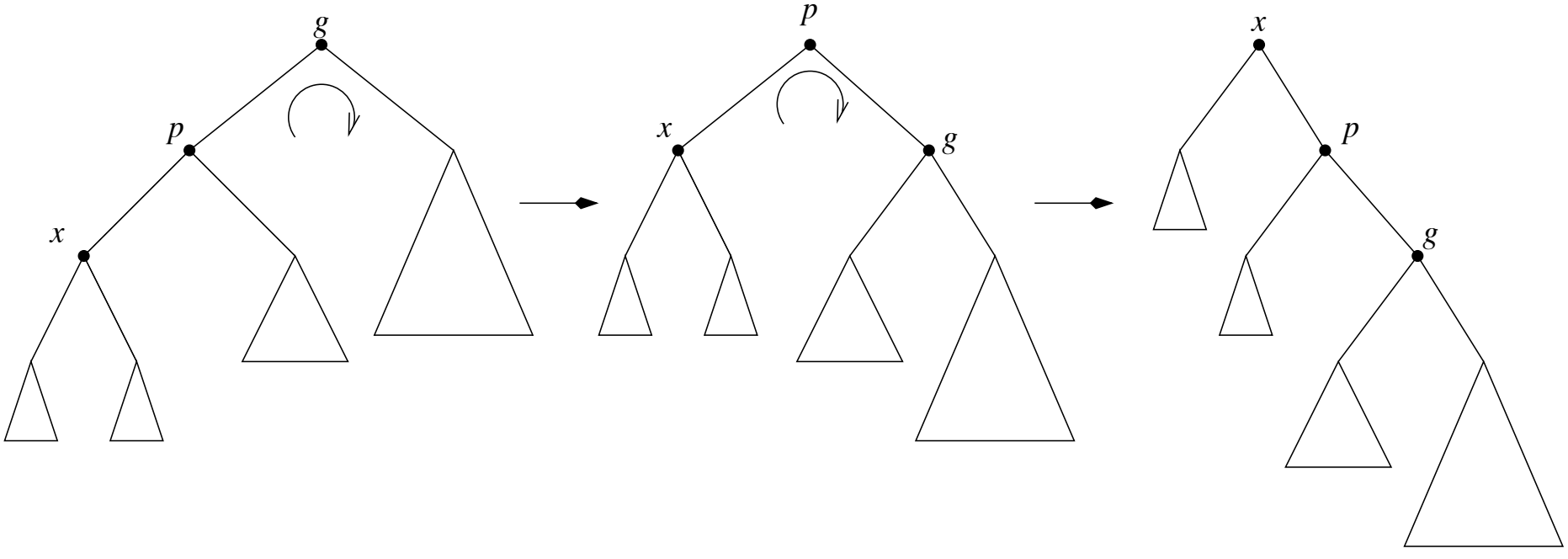
Rotate x to root, while maintain BST structure

- All nodes in subtree A go up one level, all in C go down one level, all in B stay same.
- See code in BinaryTree.java



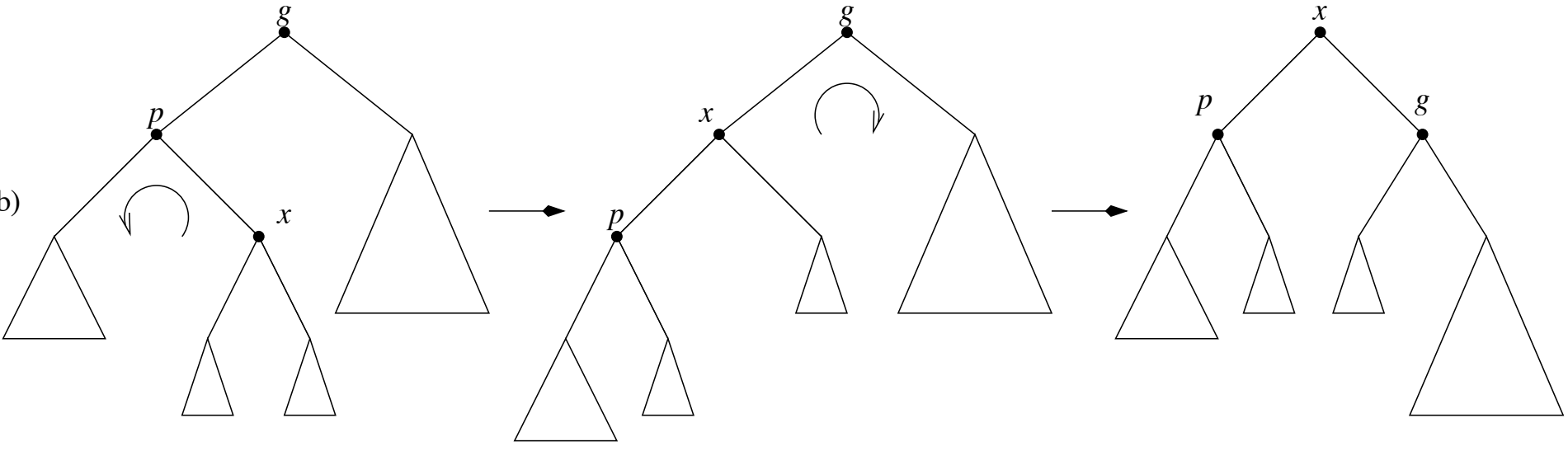
Shifting elements toward root

- Move x up two levels w/ two rotations
- If x is left child of a left child…



Shifting elements toward root

- If x is a right child of a left child…



Symmetric if interchangeable left and right

Splay Tree

- Idea behind splay tree:
  - Every time `contains`, `add` or `remove` an element x, move it to the root by a series of rotations.
  - Other elements rotate out of way while maintaining BST order.
- Splay tree are balanced
  - On average height is O(log n)
  - Worst case height is O(n)
  - All operations are on average O(log n)

Splay Tree – Theory vs Practice

- All that rotation is expensive
- Great theoretical properties
- Simple idea
- Worse performance than other balancing schemes

Fixing Sticks

- Simple “rotate-up” strategy doesn’t fix sticks
- Splay operations:
  - Zig
  - Zig-zig
  - Zig-zag

Splay operations

- Zig: Rotate self once L/R
  (when you have no grandparent)
- Zig-zig: Rotate parent, then self
  (when you're L/L or R/R)
- Zig-zag: Rotate self, then self
  (when you're L/R or R/L)

Splay Tree

Demo