CS 2230
CS II: Data structures

Meeting 25: AVL trees, in detail
Brandon Myers
University of Iowa
Today’s big ideas

• The **AVL tree** is balanced binary tree and rebalances itself using “tri-node restructuring”

• Tri-node restructuring looks at three roots involved in the new imbalance and re-arranges them to restore the balance property
Insert messes up the heights

$\text{Insert } d$
Peer instruction

After the insert, which node now doesn’t satisfy the AVL balance property?
   a) a
   b) b
   c) c
   d) d
   d) multiple nodes

https://b.socrative.com/login/student/
CS2230A ids 1000-4999
CS2230B ids 5000+
To get the answer

```
\text{insert}(d)
```
Tri-node restructuring

Goal of tri-node restructuring: rearrange 3 roots to rebalance the tree
Tri-node restructuring

Intuition: we want to **make the middle node** (here, it is b) **the new root** of the subtree

How do we pick the 3 nodes in the restructuring? Root is the node that now has unbalanced children (here, a), then include two nodes below it on the path to d (here, b and c)

© 2014 Goodrich, Tamassia, Goldwasser
Another example
Peer instruction

We might need to rebalance the tree. What will be the result?

a) no change
b) a remains the root but left=b
c) b is the new root with left=a, right=c
d) c is the new root with left=a, right=b
e) d is the new root with left=b, right=c

https://b.socrative.com/login/student/
CS2230A ids 1000-4999
CS2230B ids 5000+
We might need to rebalance the tree. What will be the result?

a) no change
b) a remains the root but left=b
c) b is the new root with left=a, right=c
d) c is the new root with left=a, right=b
e) d is the new root with left=b, right=c
We might need to rebalance the tree. What will be the result?

a) no change
b) a remains the root but left=b
c) b is the new root with left=a, right=c
d) c is the new root with left=a, right=b
e) d is the new root with left=b, right=c
Implementing tri-node restructuring

• Step 1: traverse up from the inserted node, checking the AVL balance property at each node. Once you find once, you have your 3 nodes involved in the tri-node

check d,
check its parents...
check b,
check c,
check a (aha! imbalanced)
Peer instruction

Node 54 was just inserted, what three nodes should be included in the tri-node restructuring?

https://b.socrative.com/login/student/
CS2230A ids 1000-4999
CS2230B ids 5000+
Answer
Implementing tri-node restructuring

- Step 1: traverse up from the inserted node, checking the AVL balance property at each node. Once you find once, you have your 3 nodes involved in the tri-node

- Step 2: do either 1 rotation or 2 rotations depending on which of four cases...
Implementing tri-node restructuring

Single rotations

Case 1: right, right

Case 2: left, left

© 2014 Goodrich, Tamassia, Goldwasser
Implementing tri-node restructuring

**Double rotations**

- Case 3: right, left

- Case 4: left, right

© 2014 Goodrich, Tamassia, Goldwasser
Which case is needed to perform either 1 or 2 rotations?

a) case 1: right, right
b) case 2: left, left
c) case 3: right, left
d) case 4: left, right
Answer

left rotate (50)

right rotate (78)
Question for next time...

after an insert imbalances a BinarySearchTree of height $H$, how many tri-node restructurings do we have to perform to balance the whole tree?

(give answer in big-oh)
Today’s big ideas

• The **AVL tree** is balanced binary tree and re-balances itself using “tri-node restructuring”

• Tri-node restructuring looks at three roots involved in the new imbalance and re-arranges them to restore the balance property