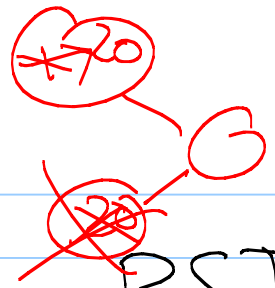
CS180 - AVL trees (3)

Note Title 10/31/2012

Announcements

- Test in 1 week
- HW due Monday
  (download new BinaryTree.h)
- AVL tree will be done today
  Today: delete

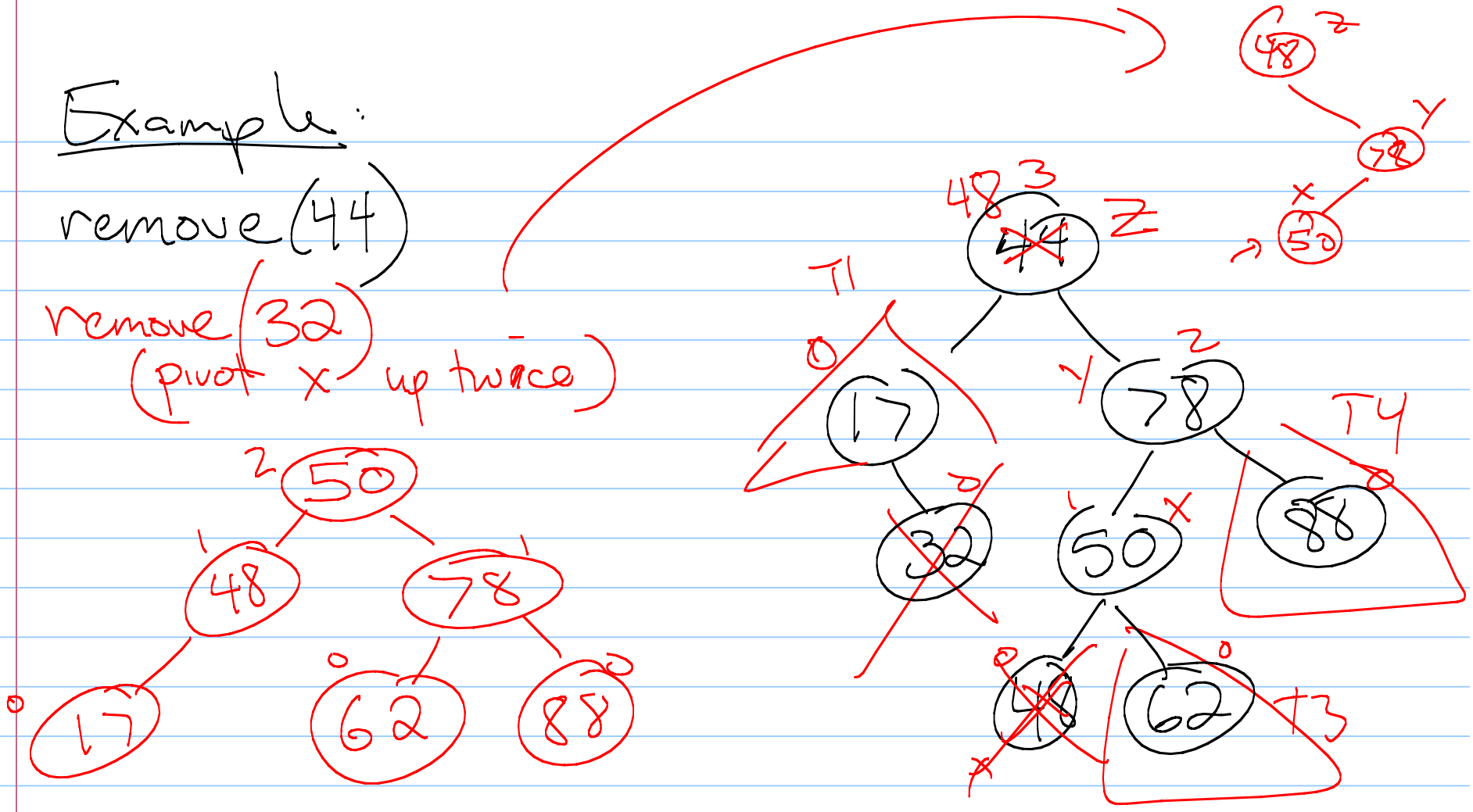
Removing in AVL trees



Step 1: Remove - just like in BST

Step 2: Re-balance (if removal violated H-B property.)

Note: Unlike insert, remove could actually unbalance all the way to the root.


Example:
remove(44)
remove(32)
(pivot x up twice)
50
48
78
17
62
88
48
44
50
78
17
32
50
88
48
62
T1
T4
T3

Fixing the tree

Algorithm to remove

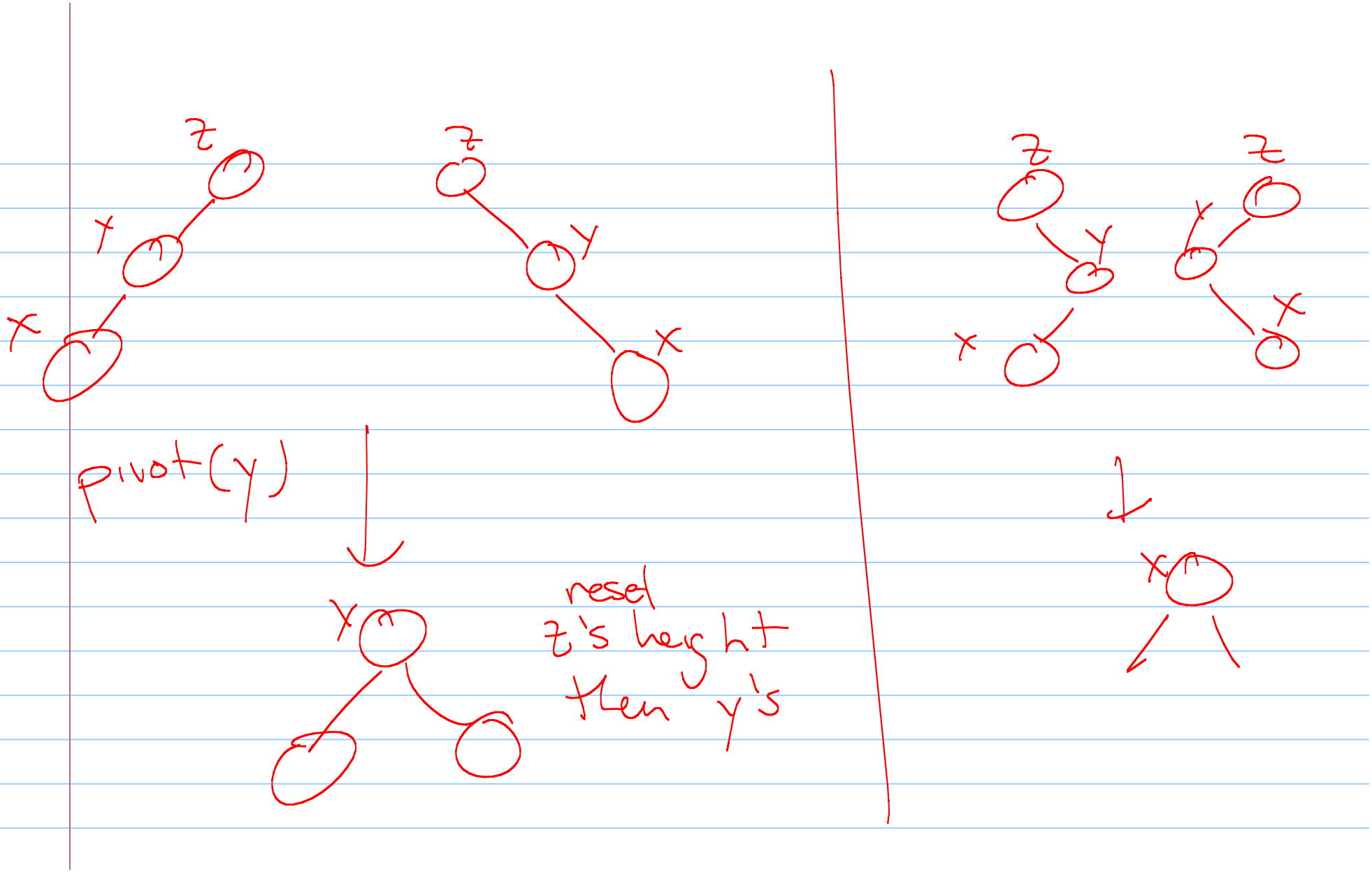
- Remove as in BST
- Track lower node that was removed.
  - Travel up tree, searching for unbalanced nodes (& fixing) until you reach the root.

Performance

For insert & delete, follow root to leaf path at most 3 times:
- find
- next in inorder (for remove)
- travel back up tree balancing

At each node:
- 1 comparison
- 2 getHigherChild, 1 setHeight, 2 ifs, & 2 pivots
  - ≤ 11 pointer updates

Total time: $O(\lg n)$


z
y
x
pivot(y)
reset
z's height
then y's

Testing insert:

insert: 10, 5, 15, 21, 35, 42