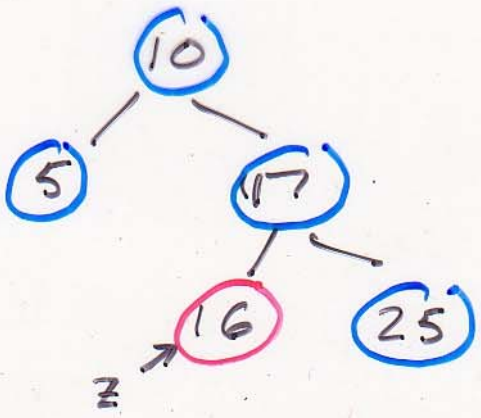
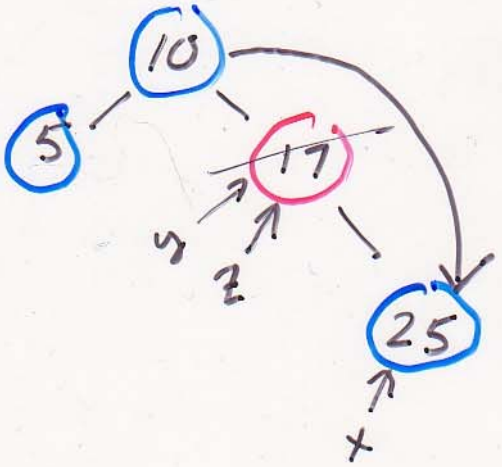
CS 223 - Lec 9

Deleting a node from a binary search tree...

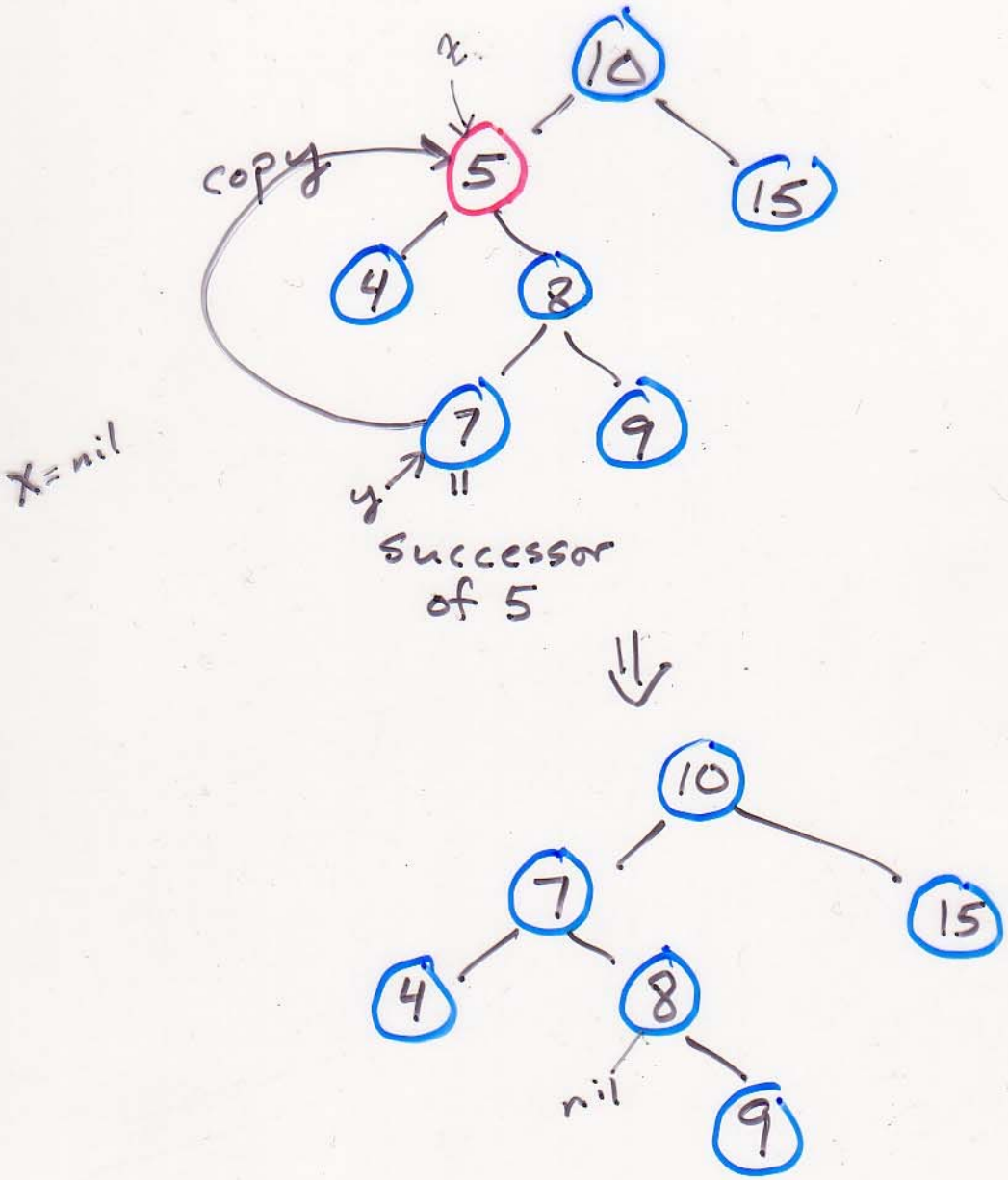
Case 1



Case 2

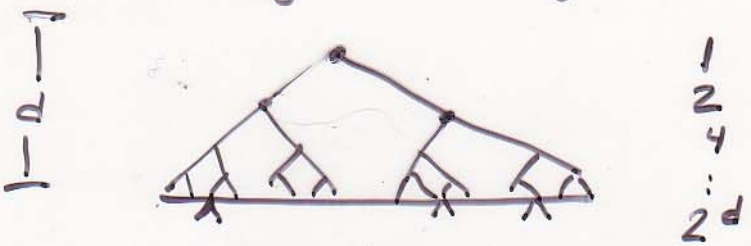


Case 3 (deleting an internal node that has two children)



Balanced Binary Search Trees.

- storing n objects



$$1 + 2 + 2^2 + \dots 2^d$$

$$n = 2^{d+1} - 1$$

$$n + 1 = 2^{d+1}$$

$$\lg(n+1) = d + 1$$

$\log_2$

$$\Rightarrow \quad d = \lceil \lg(n+1) - 1 \rceil$$

Common Balanced Search Trees

AVL trees

AA trees

- simpler implementation than R-B
- but more mysterious

* Red Black trees ← simple, efficient, provably correct

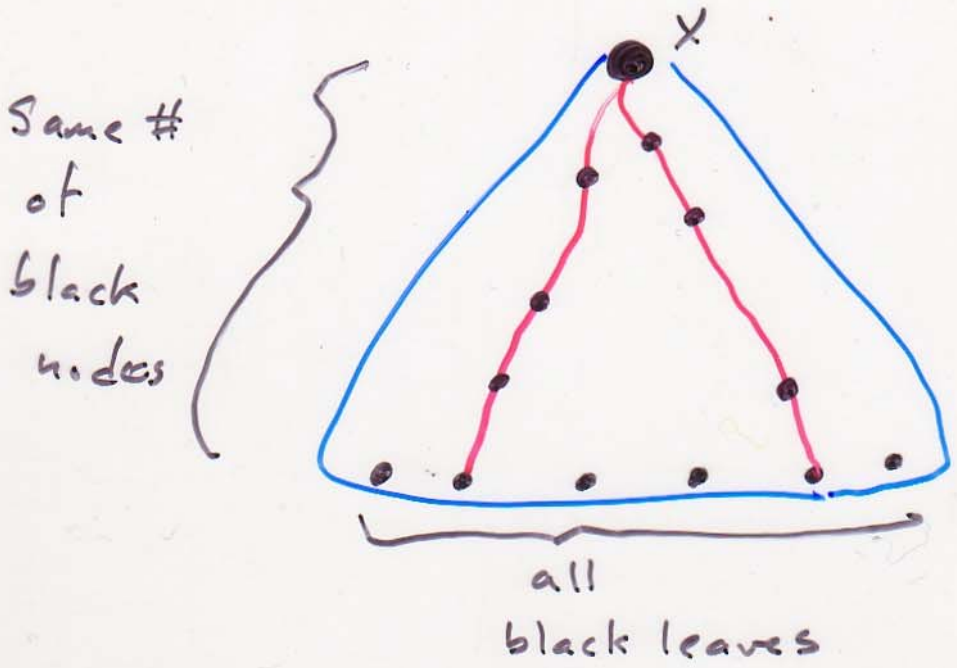
Splay Trees

Properties of R-B trees

- every node is either red or black
- the root is black
- every leaf (nil) is black

- if a node is red, then both its children are black

- for any node x, all paths from x to a descendant leaf of x must contain the same # of black nodes


x
Same # of black nodes
all black leaves