Balanced Binary Search Trees

definition of "balanced" should:

1. guarantee height is $O(\lg n)$

   "regular" BST violates this

2. work for any size collection
   (any size tree can be made legal somehow)

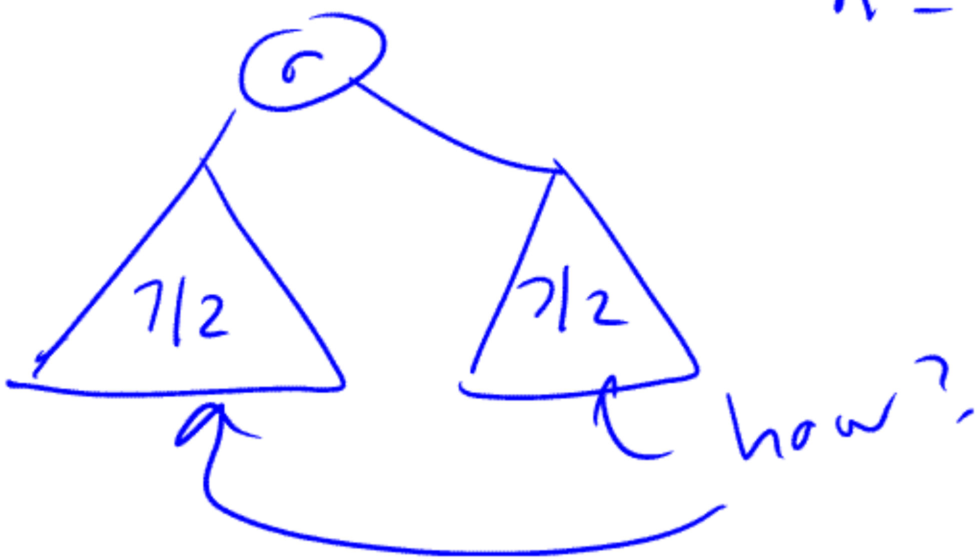
3. maintenance must be $O(\lg n)$ time

   insert & maintenance
   remove & maintenance
   $O(\lg n)$

Attempt #1

BST where at every node the subtrees of that node are the same size

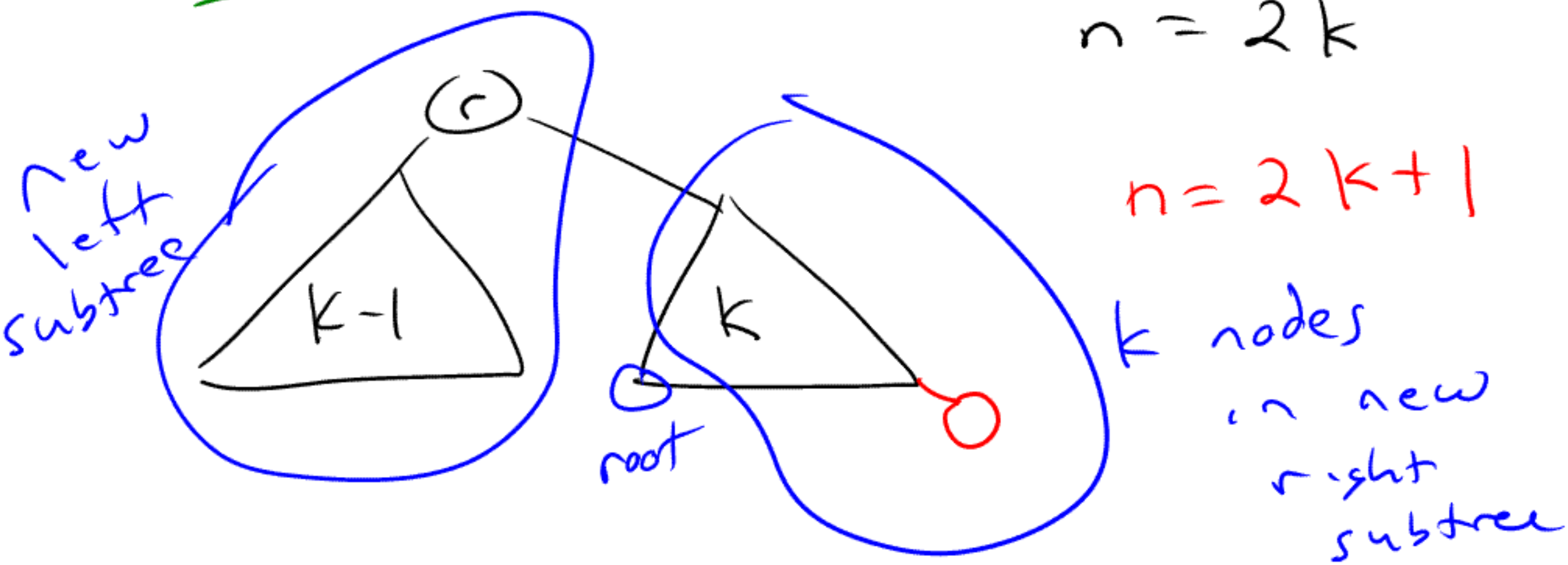
$n = 8$



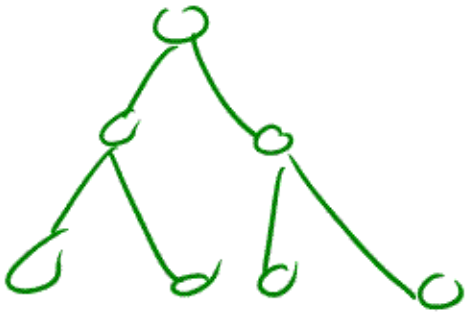
Attempt #2:

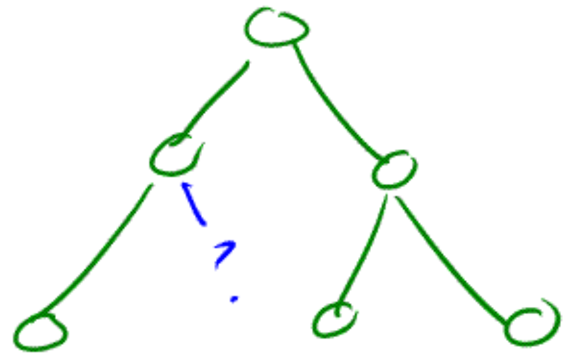
BST where at every node, the subtrees are equal in size <u>or</u> differ by one in size

$n = 2k$



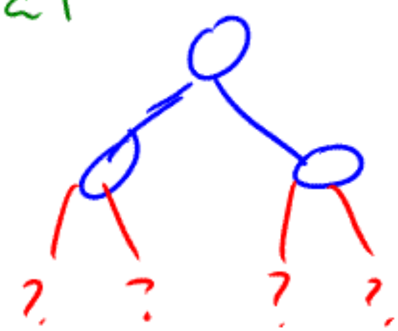
$n = 2k+1$





Attempt #3:

BST where at every node, heights of subtrees of that node are equal



$n = 4$

Attempt #4: BST where at every node, heights of that node's subtrees are equal or differ by one

1. $h \le 1.44 \lg n$ (proof in blue book)

2. <u>will</u> work for every collection

3. <u>is</u> maintainable efficiently

AVL tree - BST where at every node, heights of node's subtrees differ by at most one

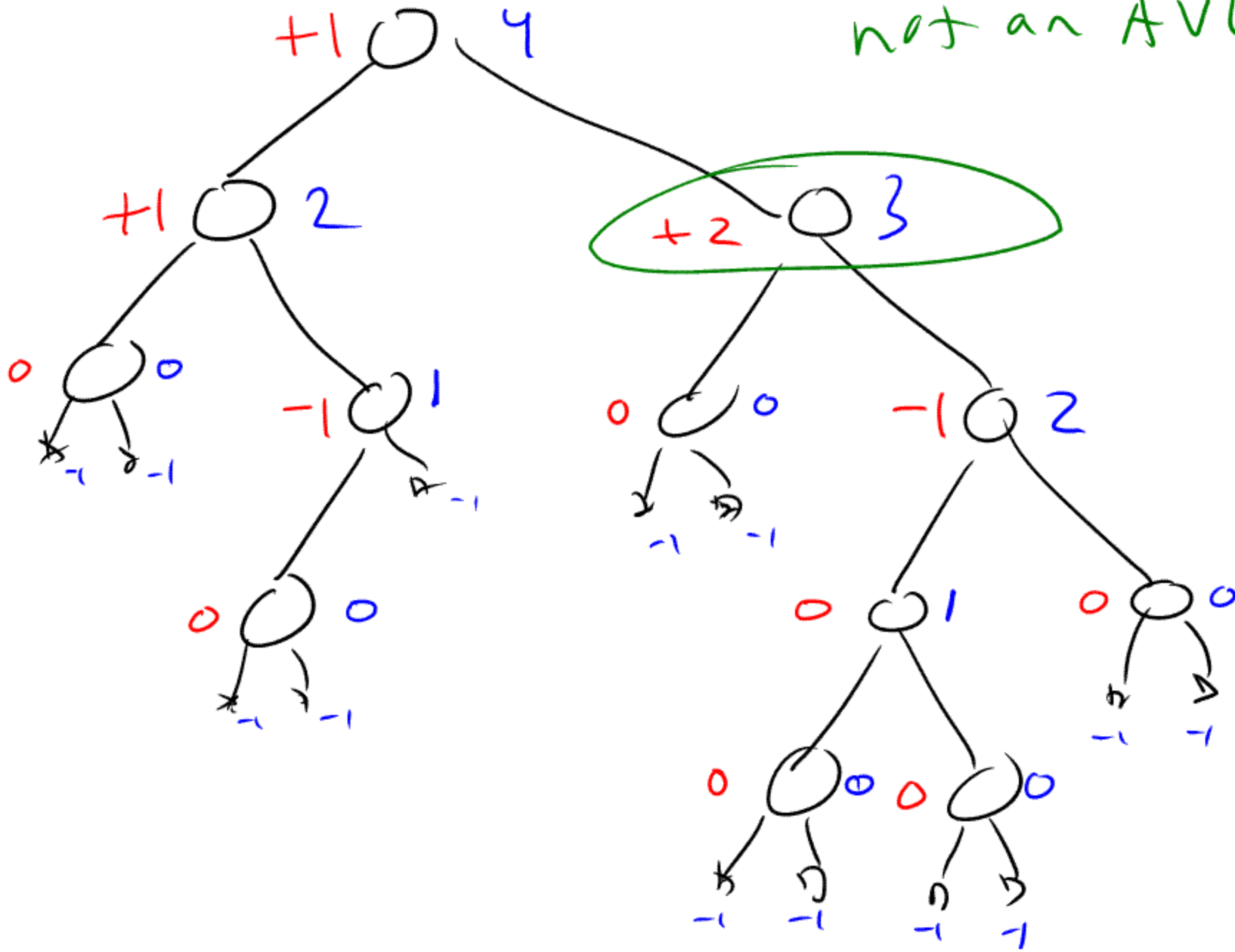
---

Height (nodeptr) → -1 if ptr is NULL

else height is

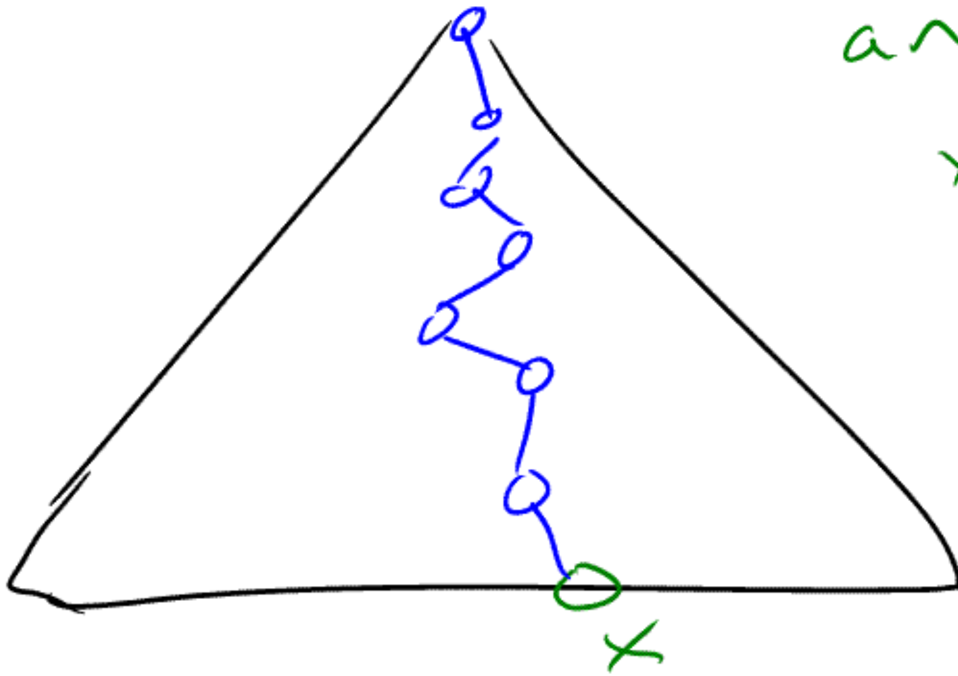
max{left subtree height, right subtree height} + 1

not an AVL



Balance (node) = height of node's right subtree − height of node's left subtree

AVL : BST where every node has balance 0, +1, or −1



ancestors of x have their heights / balances possibly affected by inserting/ removing x

x has at most $O(\lg n)$ ancestors

So → if we spend $O(1)$ per ancestor; $O(\lg n)$ total