Week 6 - Balanced Search Trees

AD 325 - Brenden West

Contents

Learning Outcomes

- overview of balanced trees
- AVL trees
- 2-3 trees
- Red-Black trees
- m-Way trees

Reading & Videos

- Carrano & Henry: Chapter 28
- https://www.coursera.org/learn/algorithms-part1/home/week/5

Reference

- https://algs4.cs.princeton.edu/33balanced/
- https://www.geeksforgeeks.org/2-3-trees-search-and-insert/
- https://www.geeksforgeeks.org/red-black-tree-set-1-introduction-2/
- https://www.geeksforgeeks.org/m-way-search-trees-set-1-searching/

Balanced Search Trees

Operations on a balanced binary search tree (BST) have O(log n) performance, but add/remove operations don’t ensure the treem remains balanced.

Balanced search tree algorithms maintain balance while adding or removing entries.

AVL Trees

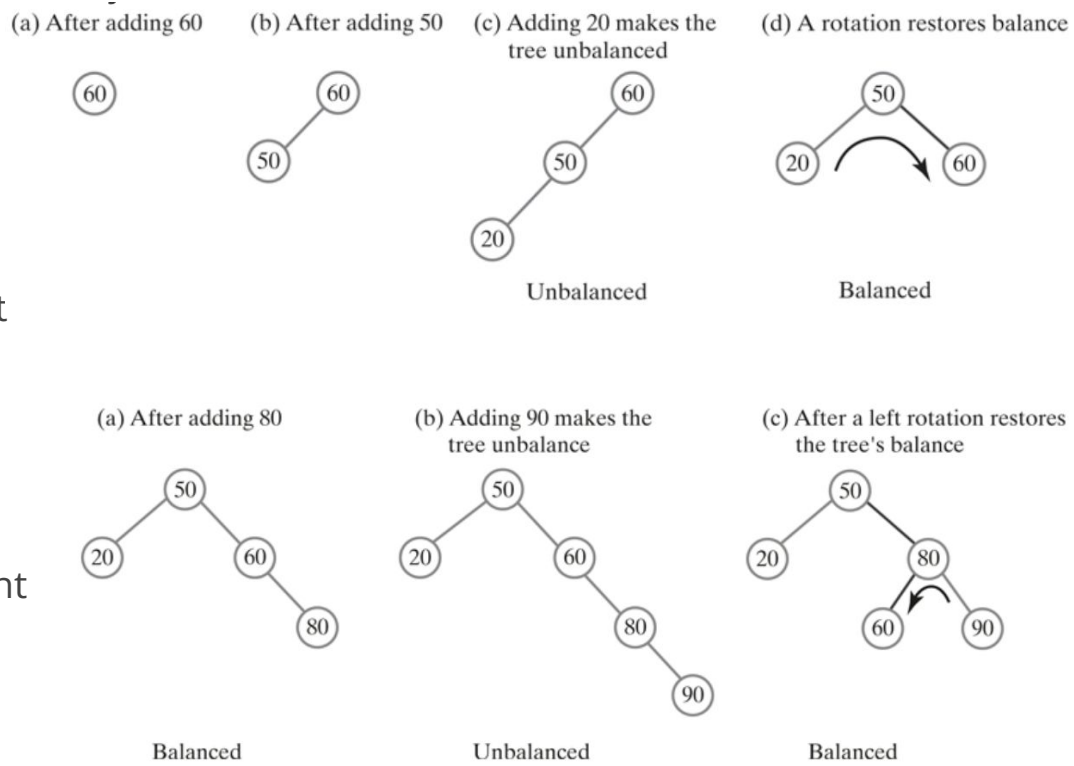
A binary search tree that re-arranges its nodes during add / remove operations to maintain balance.

A node is balanced if the height of its subtrees differ by no more than one level.

Subtree nodes are rotated around the first unbalanced node between the inserted node & the root.

A **right rotation** restores balance in the left subtree.

A **left rotation** restores balance in the right subtree.



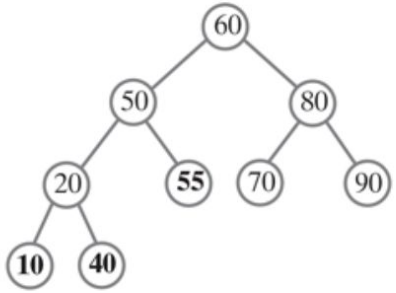
AVL - Double Rotations

Sometimes a single rotation doesn't restore balance and a second rotation in the opposite direction is required.

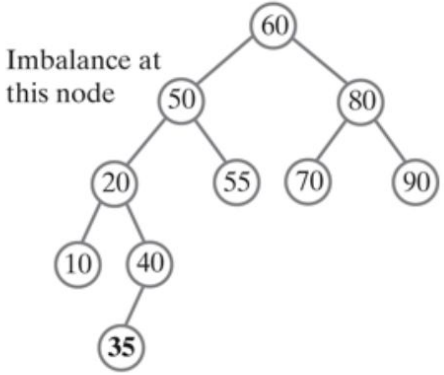
Imbalance at node N can be corrected by double rotation if

- The addition occurred in the left subtree of N's right child
- The addition occurred in the right subtree of N's left child

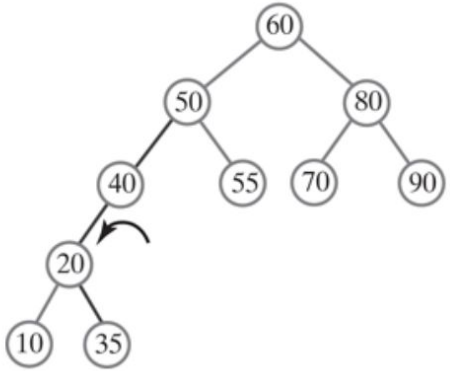
(a) After adding 55, 10, and 40



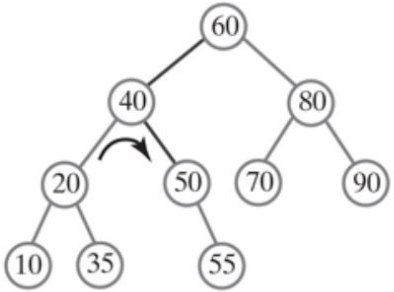
(b) After adding 35



(c) After left rotation about 40



(d) After right rotation about 40

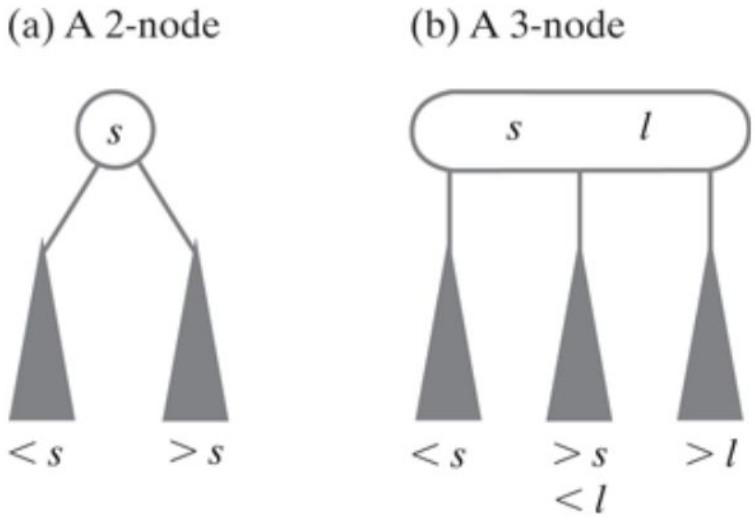


2-3 Trees

A 2-3 tree is a general search tree whose interior nodes must have either 2 or 3 children.

A 2-node contains one data item and has 2 children. The data is larger than any data in the node's left subtree and smaller than any data in the right subtree.

A 3-node contains 2 data items. Data less than the smaller data item will be in the left subtree. Data greater than the larger data item will be in the right subtree. Data between those values will be in the node's middle subtree.



2-3 Operations

A 2-3 tree is **completely balanced** with all leaves on the same level. Maintaining balance is easier than with an AVL tree.

Searching 2-nodes behaves like searching in a BST.

Searching a 3-node involves checking the node's middle subtree if the searched value is between the node's data values.

If an entry is added to a 3-node, the node is split to maintain balance. The middle value propagates up to its parent, splitting the parent node if that already has 2 data entries until reaching the root.

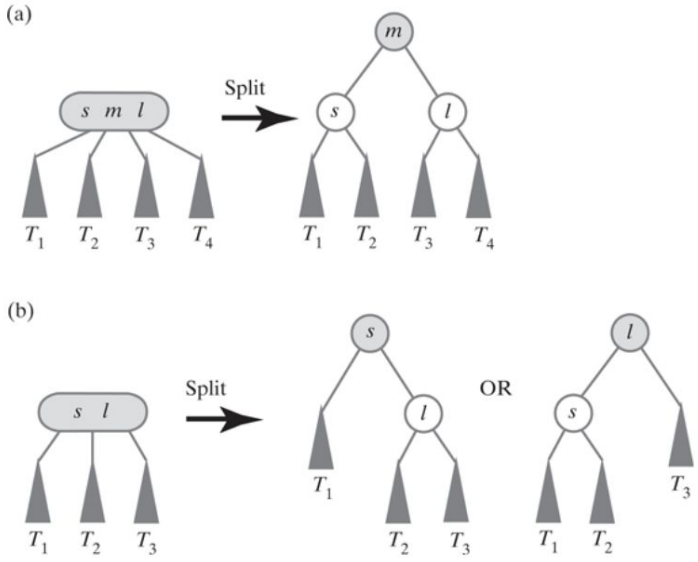
Red-Black Trees

A **red-black tree** is binary tree that's logically equivalent to a 2-4 tree.

Because it uses only 2-nodes, searching a red-black tree can use the same code as for searching a BST.

Red-black trees use **color** (a bit flag) to highlight new nodes that cause tree height to increase.

Additions to red-black tree maintain tree balance with color flips and rotations like those used for AVL trees.



white = red

Red-Black Tree Properties

The root node is black

Every red node has a black parent. Any children of a red node are black. Any new node added to a non-empty red-black tree must be red

Every path from the root to a leaf contains the same number of black nodes

The height of a red-black BST with N nodes is no more than 2 lg N

The average length of a path from the root to a node in a red-black BST with N nodes is ~1.00 lg N

The following operations take logarithmic time in the worst case - search, insertion, finding the minimum, finding the maximum, floor, ceiling, rank, select, delete the minimum, delete the maximum, delete, and range count.

Red-Black, cont.

AVL trees are more balanced than Red-Black Trees, but may cause more rotations during insertion and deletion. So Red-Black trees are preferrable for applications with frequent insertions and deletions.

Applications of red-black trees: - BST map & set functions (e.g. TreeMap & TreeSet) in Java) - MySQL table indexes - K-means clustering in data analysis

m-Way Trees

A multiway search tree whose nodes have up to m children.

- k-node has k-1 data items and k children
- a binary search tree is an m-way search tree of order 2,
- not all multiway search trees are balanced
- A B-tree of order m is a balanced multiway search tree that maintains balance with these properties:
    a. the root has either no children or between 2 and m children
    b. interior nodes have between m/2 and m children
    c. all leaves are on the same level