Data Structures
Homework #6

Due: Jan 2, 2024

Mark your calendar: Final exam is scheduled on Jan 8, 2024 (in class).

1. (20 pts) Let D be an ordered dictionary with n entries. Show how to modify the AVL tree to implement the following operation for D in time $O(\log n)$:

   countAllInRange(k_1, k_2): Compute and return the number of entries in D with key k such that $k_1 \leq k \leq k_2$.

   You also need to provide the modified insertion and deletion operations.

   HINT: Note that this operation returns a single integer. You will need to extend the AVL tree data structure, adding a new field to each internal node and the ways of maintaining this field during updates.

2. (20 pts) Prove that the number of (single or double) rotations done in removing a key from an AVL tree can not exceed half the height of the tree.

3. (30 pts) A **B*-tree** is a B-tree in which every node, except possibly the root, is at least two-third full, rather than half full. Insertion into a B*-tree moves entries between sibling nodes (as done during deletion) as needed, thereby delaying splitting a node until two sibling nodes are completely full. These two nodes can then be split into three, each of which will be at least two-third full. Discuss the relative advantages and disadvantages of B*-trees compared to ordinary B-tree.

4. (30 pts) Recall the red-black tree structure we discuss in class. It manages n entries, of which each is a pair of *key* and *value*, (key, value) and supports, in O($\log n$) time, the operations `search(x)`, `insert(x)` and `delete(x)`, where x is a key for some entry.
   Describe how to augment a red-black tree to support, in O($\log n$) time, the operations `increase(x, y, q)` and `decrease(x, y, q)`. That is to have additional fields (or attributes) used in the structure. In `increase(x, y, q)`, the value ($w.value$) of every entry w in the tree with key, $x \leq w.key \leq y$, is increased by q (the `decrease` operation does the corresponding decrease). One cannot explicitly update the entries as this may take O(n) time. Describe how **all** operations can be executed in O($\log n$) time.