1 Assignment No. 7: Dynamic Order Statistics

Allocated time: 2 hours

1.1 Implementation

You are required to implement **correctly** and **efficiently** the management operations of an **order statistics tree** (*chapter 14.1 from [1]*).

You have to use a balanced, augmented Binary Search Tree. Each node in the tree holds, besides the necessary information, also the *size* field (i.e. the size of the sub-tree rooted at the node).

The management operations of an **order statistics tree** are:

- BUILD_TREE(n)
  - *builds* a **balanced** BST containing the keys 1,2,...n (*hint:* use a divide and conquer approach)
  - make sure you initialize the size field in each tree node
- OS-SELECT(tree, i)
  - selects the element with the i-th smallest key
  - the pseudo-code is available in *chapter 14.1 from the book[1]*
- OS-DELETE(tree, i)
  - you may use the deletion from a BST, without increasing the height of the tree (why don't you need to rebalance the tree?)
  - keep the size information consistent after subsequent deletes
  - there are several alternatives to update the size field without increasing the complexity of the algorithm (it is up to you to figure this out).

Does OS-SELECT resemble anything you studied this semester?

1.2 Minimal requirements for grading

The lack of any of the minimum requirements (even partially) may result in a lower grade through penalties or refusal to accept the assignment resulting in a grade of 0.

- *Demo*: Prepare a demonstration of correctness for each algorithm implemented. The correctness of each algorithm is demonstrated through a simple example (maximum 10 values).

- The charts created must be easy to evaluate as in grouped and added through the Profiler functions as specified by the assignment requirements. The assignment will not be evaluated if it contains a plethora of ungrouped charts. For example, the comparative analysis implies the grouping of the compared algorithms.
- Interpret the chart and write your observations in the header (block comments) section at the beginning of your *main.cpp* file.
- We do not accept assignments without code indentation and with code not organized in functions (for example where the entire code is in the main function).
- *The points from the requirements correspond to a correct and complete solution, quality of interpretation from the block comment and* ***the correct answer to the questions from the teacher.***

1.3 Requirements

1.3.1 BUILD_TREE: correct and efficient implementation (5p)

Demo: You will have to prove your algorithm(s) work on a small-sized input (11)

- pretty-print the initially built tree

1.3.2 OS_SELECT: correct and efficient implementation (1p)

Demo: You will have to prove your algorithm(s) work on a small-sized input (11)

- execute OS-SELECT for a few elements (at least 3) by a randomly selected index

1.3.3 OS_DELETE: correct and efficient implementation (2p)

Demo: You will have to prove your algorithm(s) work on a small-sized input (11)

- execute OS-SELECT followed by OS-DELETE for a few elements (at least 3) by a randomly selected index *and pretty-print the tree after each execution.*

1.3.4 Management operations evaluation - BUILD, SELECT, DELETE (2p)

Once you are sure your program works correctly:

- vary n from 100 to 10000 with a step of 100;

- for each n (don't forget to repeat 5 times),
  - BUILD a tree with elements from *1 to n*
  - perform n sequences of OS-SELECT and OS-DELETE operations using a randomly selected index based on the remaining number of elements in the BST,
  - Evaluate the number of operations needed for each management operation (BUILD, SELECT, DELETE – *resulting in a plot with 3 series*). Evaluate the computational effort as the sum of the comparisons and assignments performed by each individual management operation ofr each value of n.

1.3.5 Bonus: Implementation using AVL / Red black tree (1p)

Demo: You will have to prove your algorithm(s) work on a small-sized input (11)

References

[1] Thomas H. Cormen et al. *Introduction to Algorithms*. 2nd. The MIT Press, 2001. ISBN: 0262032937.