CS 202
Fundamental Structures of Computer Science II

Fall 2017
Assignment 3 Solutions

Section 1
Selim Firat Yilmaz
21502736
firat.yilmaz@ug.bilkent.edu.tr

November 27, 2017

Contents

1 Question 1

1.1 a) AVL Tree Insertion/Deletion

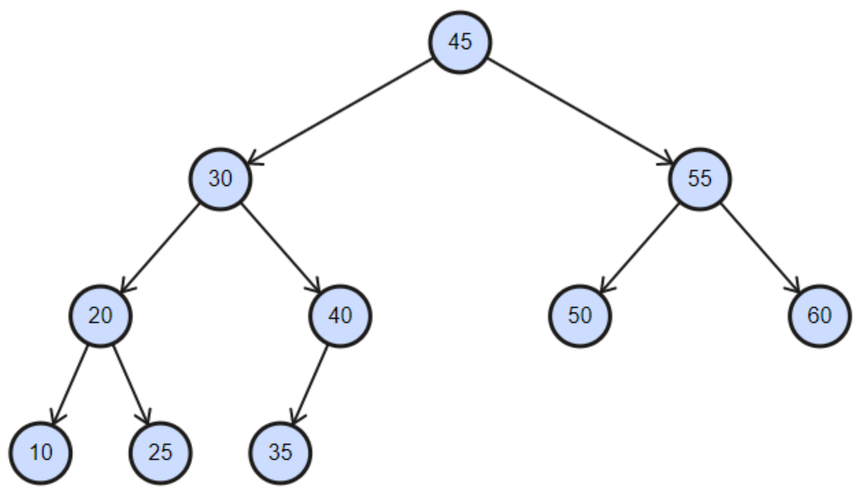

Figure 1: After desired elements are inserted to an empty AVL Tree

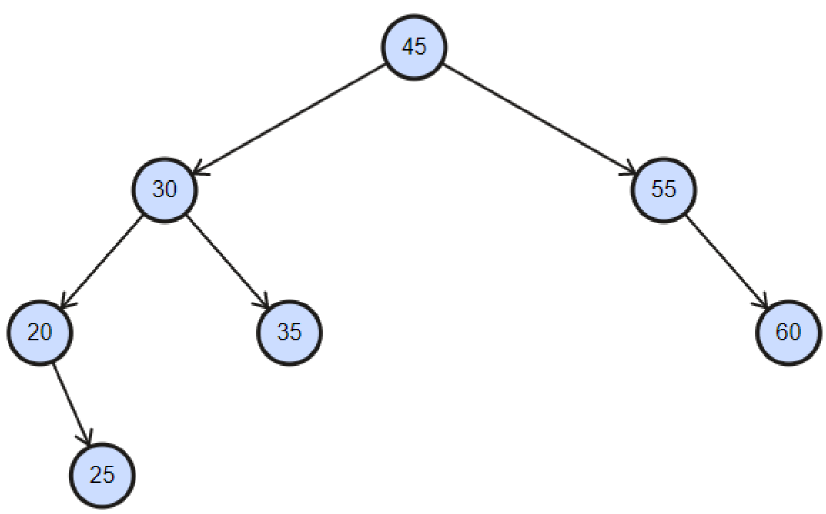

Figure 2: After desired elements are deleted from the AVL Tree

1.2 b) Max-heap Insertion/Deletion

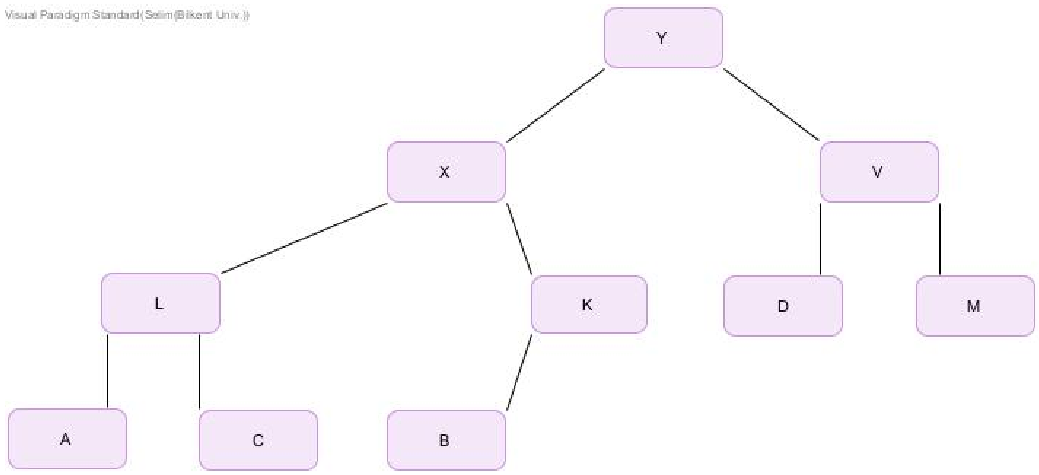

Figure 3: After desired elements are inserted to an empty Max-heap

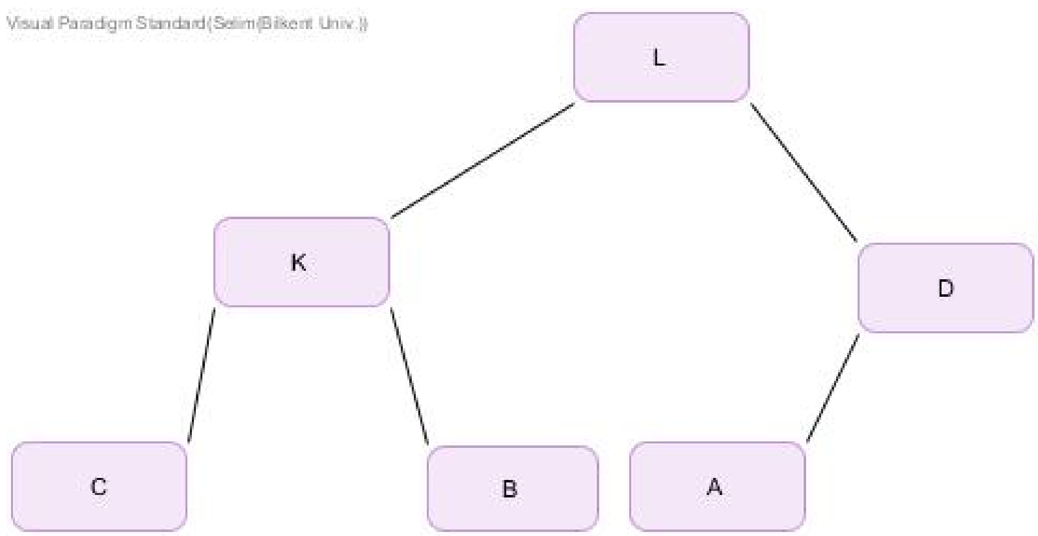

Figure 4: After desired elements are deleted from the Max-heap

1.3 c) Best Fitting Data Structure

1.3.1 (a) finding maximum element quickly

It does not matter because we can obtain the maximum element in both in $O(1)$ time. Root of Max-heap and the last element of ascendingly sorted array corresponds to the maximum element. (It is assumed that we know the size of the ascendingly sorted array. Also it corresponds the first element of the array is descendingly sorted)

1.3.2 (b) finding minimum element quickly

Sorted array is better because it is $O(1)$ in sorted array and corresponds to the first element in ascendingly sorted array.

1.3.3 (c) finding median of elements quickly

Sorted array is better if we know the size of it because we can obtain the median in O(1) time whereas in max-heap we need to extract max repeatedly in $O(n)$ time.

1.3.4 (d) deleting an element quickly

Max-heap is better because we can delete an element in $O(\log n)$ time whereas in sorted array we need to reconstruct the array in $O(n)$ time in worst-case.

1.3.5 (e) forming the structure quickly

Sorted array is easier because there are many easily implementable algorithm to sort an array like insertion sort or data structure array implementations like BST.

2 Question 3

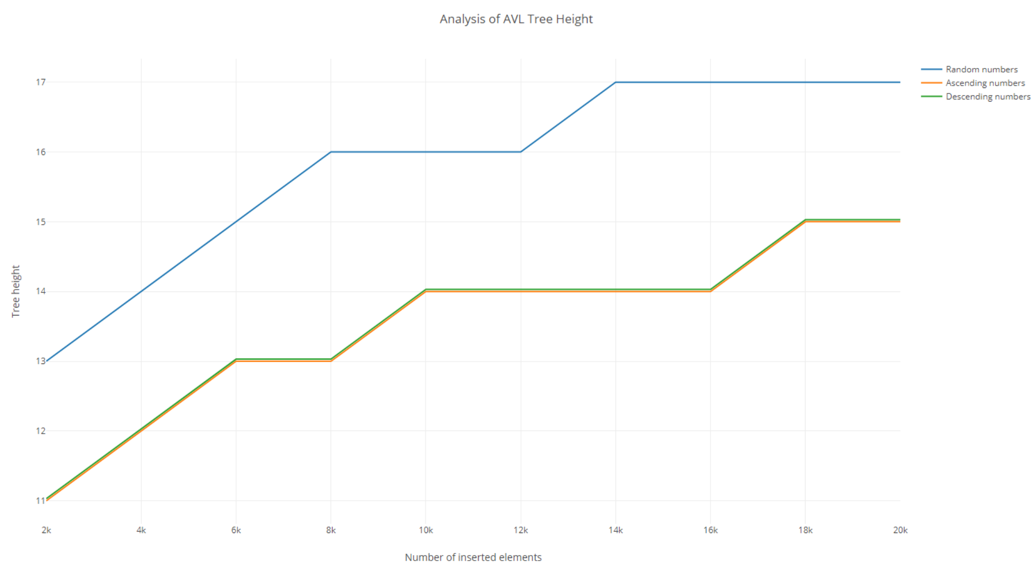

Figure 5: Experiment result for the analysis of AVL Tree Height

2.0.1 Do your findings related to average height of AVL tree agree with the theoretical results?

Yes, they agrees with the theoretical results. The average height of AVL Tree is $O(\log n)$ for n number of items, and we have
$\log 2000 \approx 10.96$, $\log 10000 \approx 13.28$ c, $\log 20000 \approx 14.28$.

These findings are close the $\log n$ for n number of items and thus, we can say that our findings matches with theoretical results.

2.0.2 Do the different patterns of insertion affect the height of AVL tree? If so, explain how. If not, explain why not.

Yes, different patterns affect the height of AVL tree as seen on the plot above since the blue line which represents the random number insertion experiment are not the same with orange & yellow lines.

In the experiment of ascending and descendingly generated numbers, the occurred rotations are symmetric which affect the height equally whereas in the experiment of randomly generated numbers, the occurred rotations are not symmetric and less rotations are happening so it has more height.

2.0.3 How would the result be if you used regular Binary Search Tree instead of AVL tree?

Binary Search tree would result in height n after ascending and descending insertion experiment. For example, in BST, it would result in height of 20000 for 20000 ascendingly inserted elements.

As I showed on assignment 2, insertion of 20000 randomly generated numbers would result in height ≈ 36 in BST whereas it would result in height ≈ 17 in AVL Tree.