ADSA Assignment 1

Gulshan Hatzade

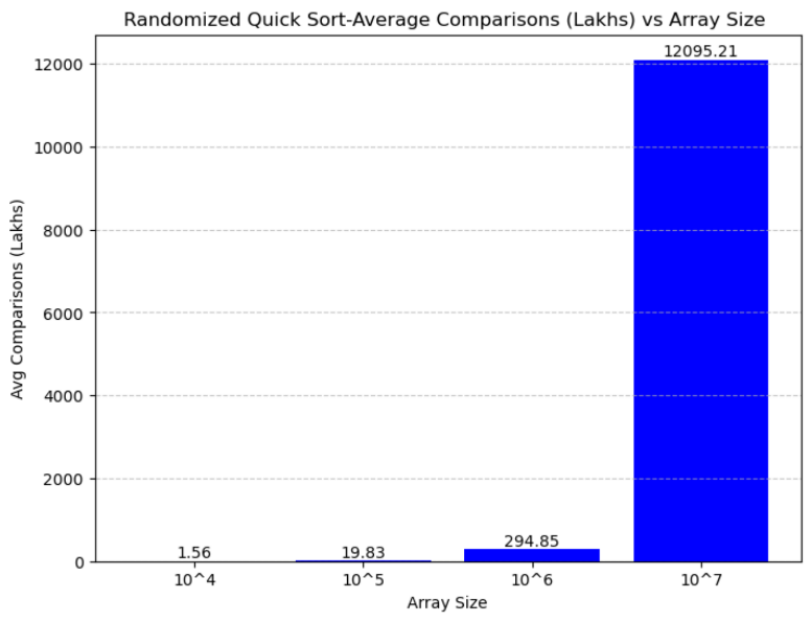
October 2024

1 Randomized Quicksort

The following results were obtained for the randomized quicksort algorithm across different array sizes-

- **Array size 10,000:** Average Comparisons: 155,982.80, Standard Deviation: 4,855.56
- **Array size 100,000:** Average Comparisons: 1,983,168.44, Standard Deviation: 45,869.03
- **Array size 1,000,000:** Average Comparisons: 29,485,141.55, Standard Deviation: 1,175,563.43
- **Array size 10,000,000:** Average Comparisons: 1,209,521,403.58, Standard Deviation: 53,523,516.80

Plot of average no. of comparisons with size of array-



As expected, avg no. of comparisons increases as the array size grows, following $\mathcal{O}(n \log n)$ complexity of the algorithm.

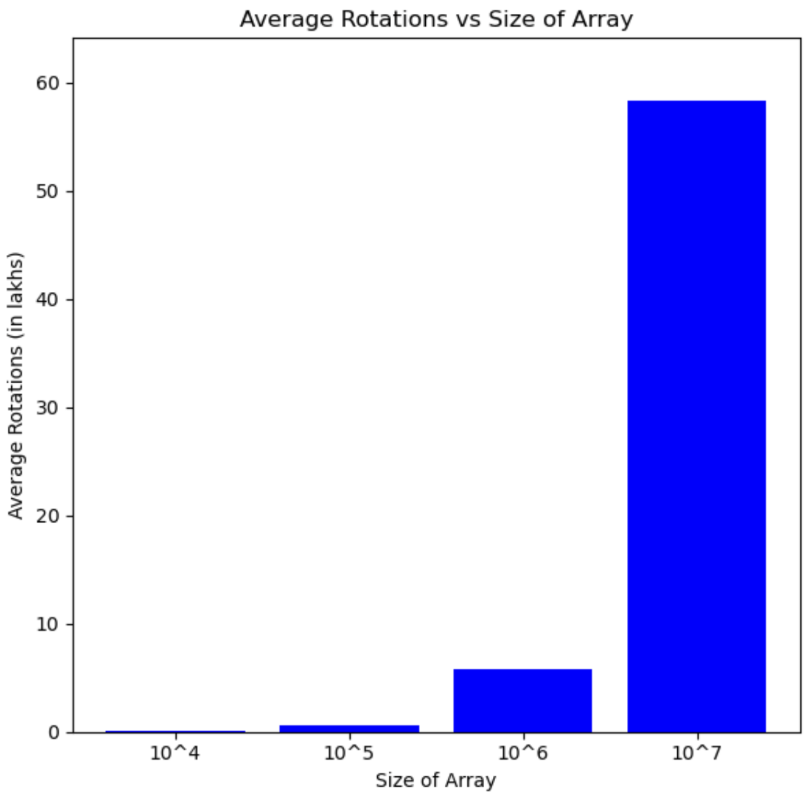
Standard deviation also increases with array size which reflects variability in the algorithm's performance across random inputs.

2 Red-Black Tree Insertion

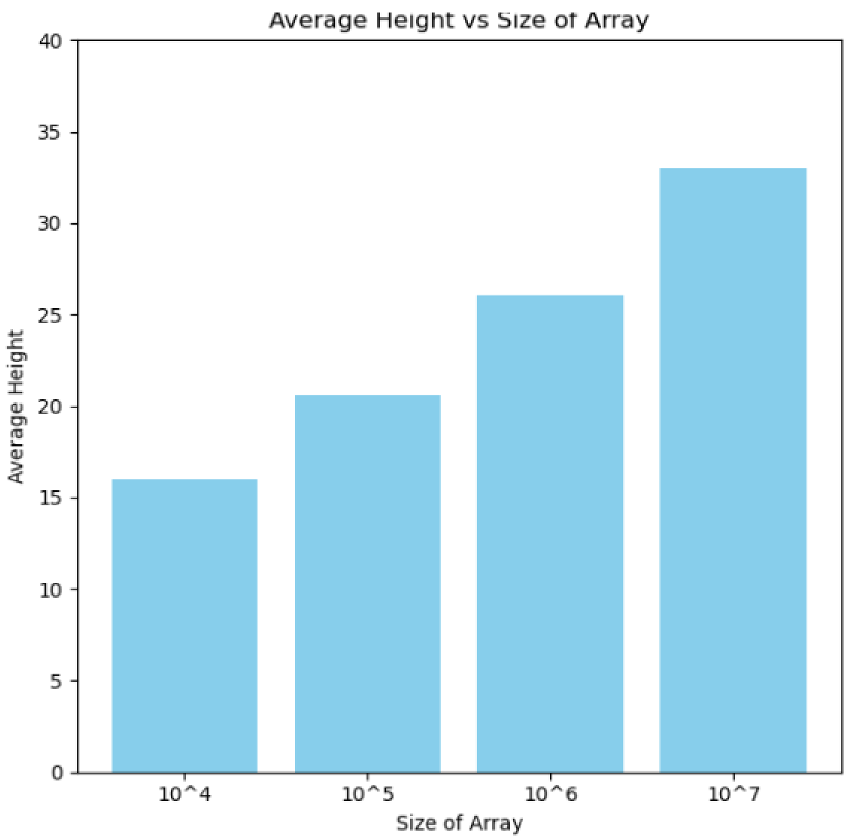
The results for red-black tree insertion across different array sizes are as follows-

- **Array size 10,000:** Average Rotations: 5,815.27, Average Height: 16.01
- **Array size 100,000:** Average Rotations: 58,254.72, Average Height: 20.63
- **Array size 1,000,000:** Average Rotations: 582,578.31, Average Height: 26.08
- **Array size 10,000,000:** Average Rotations: 5,825,133.00, Average Height: 33.00

Plot of average no. of rotations with size of array-



Plot of average height with size of array-



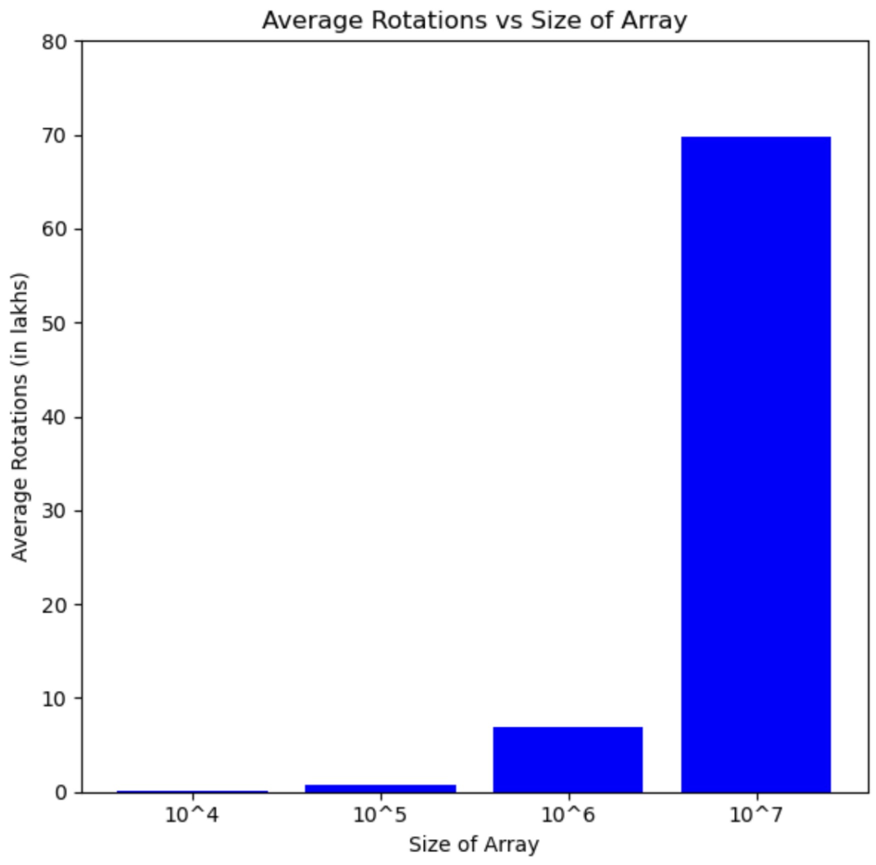
Results shows that both no. of rotations and tree height increase as the array size grows. This is expected due to the need for more balancing operations in larger trees, as clearly observed from the results.

3 AVL Tree Insertion

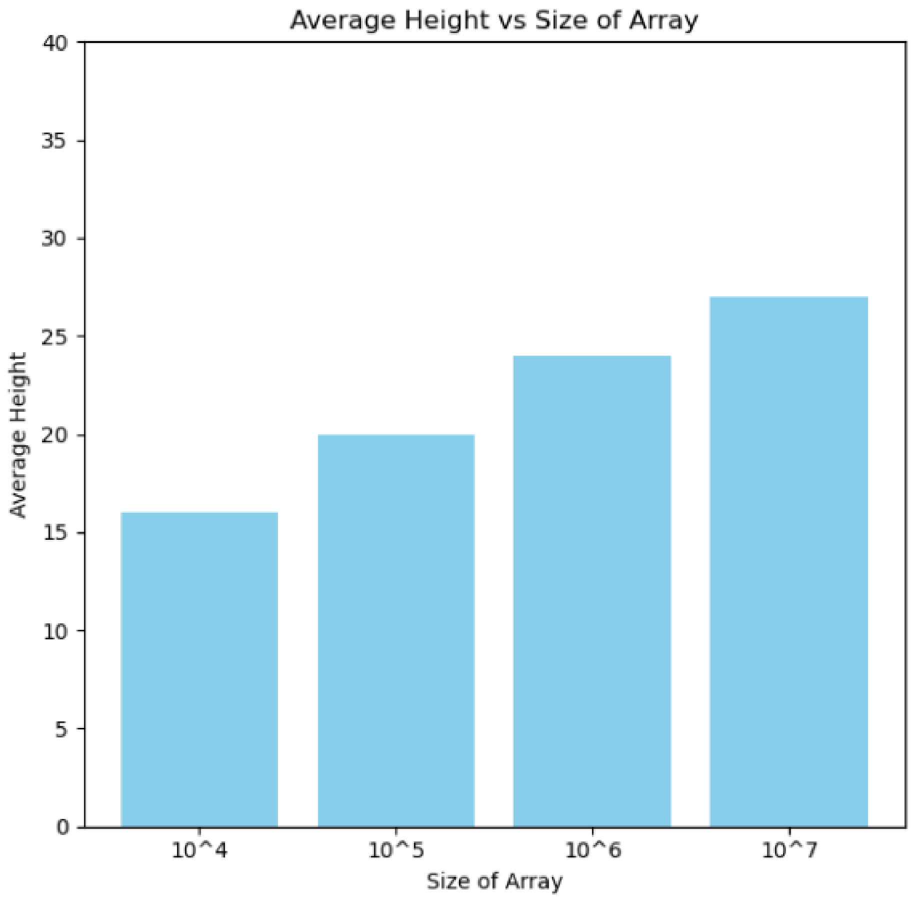
Results for AVL tree insertion across different array sizes are-

- **Array size 10,000:** Average Rotations: 6,977.83, Average Height: 16.00
- **Array size 100,000:** Average Rotations: 69,795.64, Average Height: 20.00
- **Array size 1,000,000:** Average Rotations: 698,254.38, Average Height: 24.00
- **Array size 10,000,000:** Average Rotations: 6,982,610.50, Average Height: 27.00

Plot of average no. of rotations with size of array-



Plot of average height with size of array-



As expected, height of AVL tree grows as the array size increases. Stricter balancing criteria ensures the height remains lower compared to red-black trees, even though it requires more rotations for achieving this balance.

Due to stricter balancing criteria of AVL trees, they performed more rotations than red black trees across all array sizes, ensuring lower tree height.

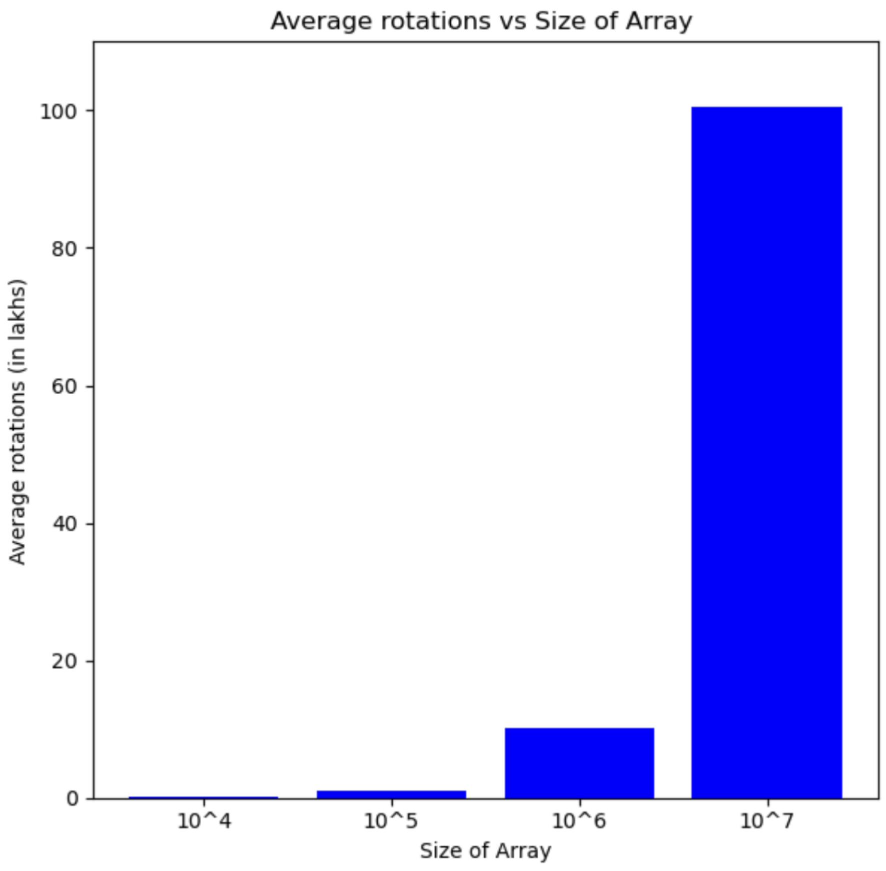
4 AVL Tree deletion

The results for AVL tree deletion across different array sizes are:

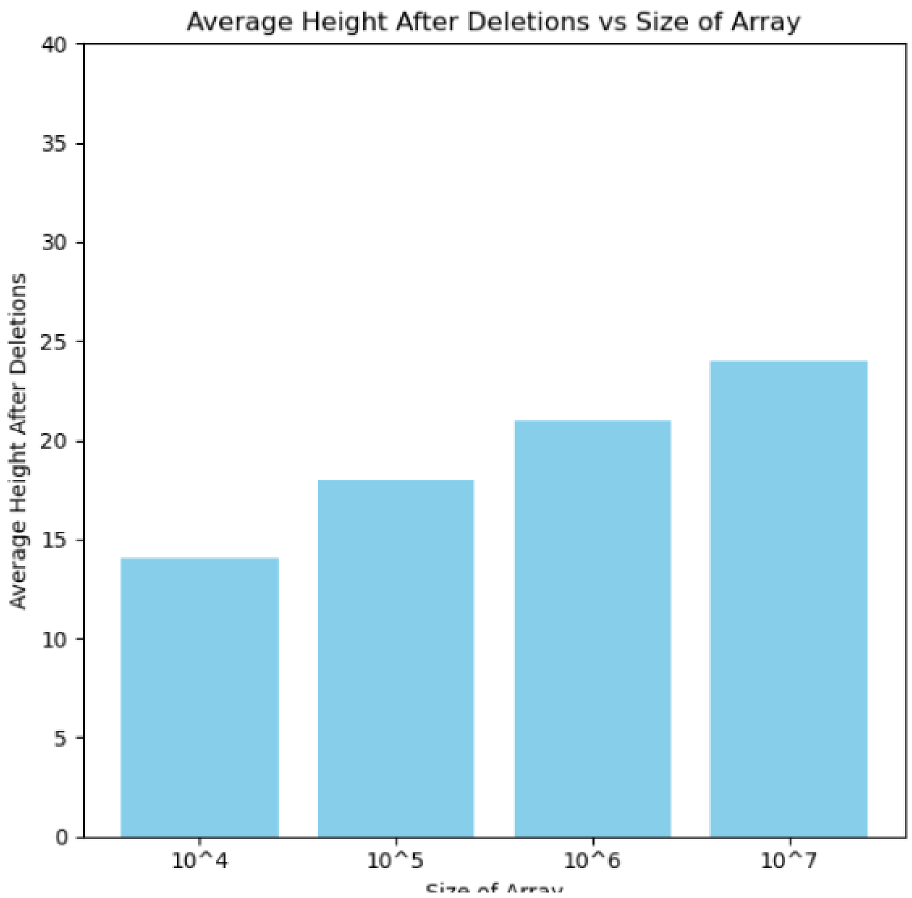
- **Array size 10,000:** Average Rotations: 10,035.42, Average Height: 14.00
- **Array size 100,000:** Average Rotations: 102,543.45, Average Height: 18.00
- **Array size 1,000,000:** Average Rotations: 1,014,155.44, Average Height: 21.00

- **Array size 10,000,000:** Average Rotations: 10,040,960.00, Average Height: 24.00

Plot of average no. of rotations with size of array-



Plot of average height with size of array-

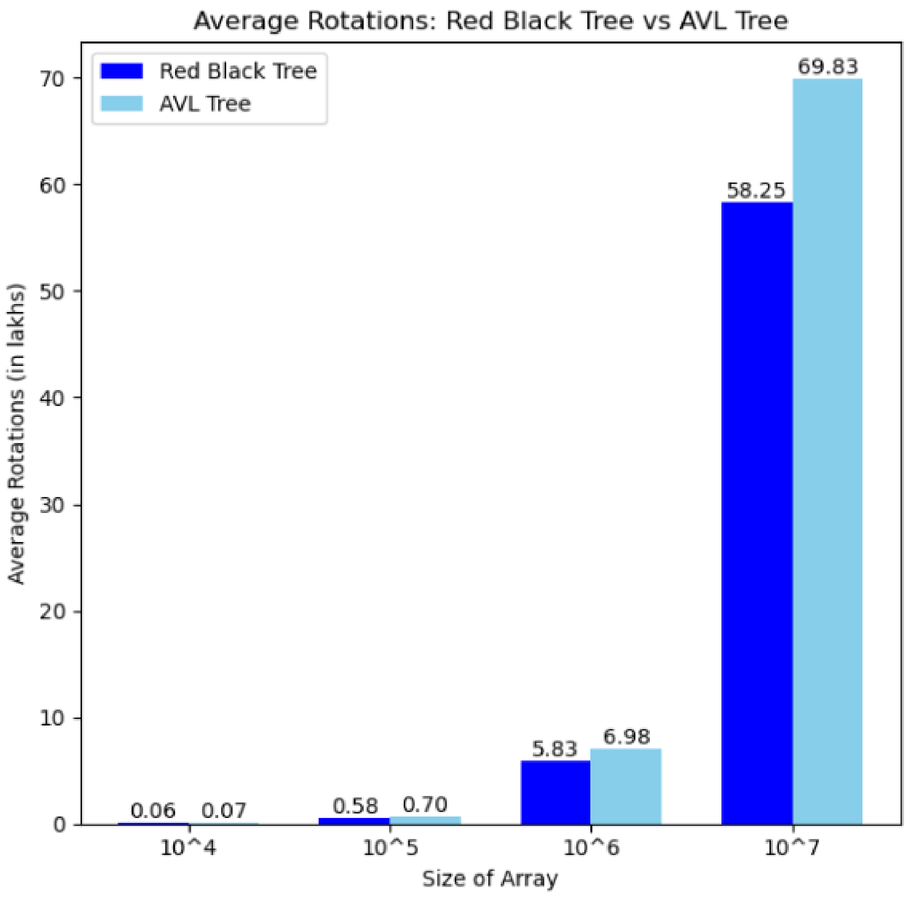


After deletion, height of AVL tree decreases slightly compared to the insertion phase, but it still maintains a well-balanced structure.

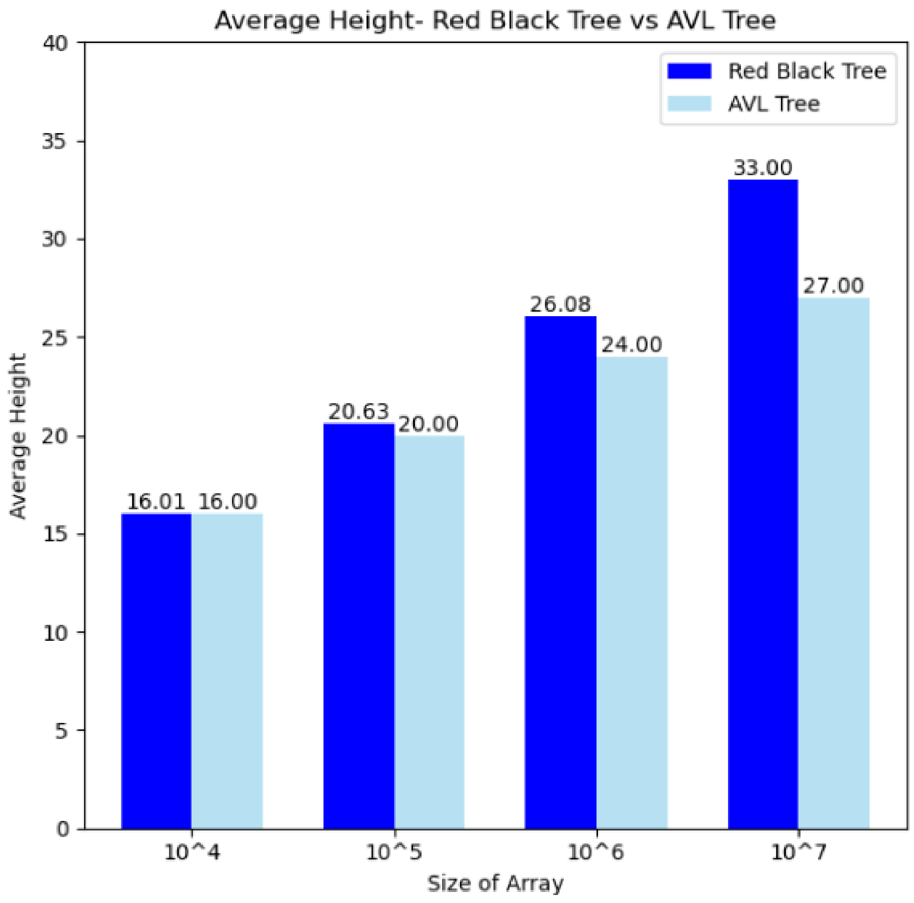
No. of rotations required for rebalancing during deletion is consistent with AVL tree's design, which prioritizes height minimization.

5 Comparative Graphs for RB tree and AVL tree for insertion

Plot for rotation in insertion in reb black tree and avl tree-



Plot for height in insertion in reb black tree and avl tree-



6 Analysis

6.1 Randomized Quicksort Analysis

Results obtained from randomized quicksort algorithm across varying array sizes indicate consistent trend in performance.

As size of array increases from 10,000 to 10,000,000, avg no. of comparisons escalates from 155,982.80 to 1,209,521,403.58. This exponential growth aligns with expected O(n log n) complexity of algorithm, confirming its efficiency for larger datasets.

Standard deviation also rises with array size, from 4,855.56 to 53,523,516.80.

This increase in variability suggests that while algorithm generally performs efficiently, its performance can be sensitive to specific input arrangements which may lead to fluctuations in comparison counts.

6.2 Red-Black Tree Insertion Analysis

As size of dataset increases, both avg no. of rotations tree height increase. For instance, avg rotations increase from 5,815.27 for an array size of 10,000 to 5,825,133.00 for an array size of 10,000,000, while average height rises from 16.01 to 33.00.

This results are expected due to 1need for balancing operations in larger trees.RB trees require fewer rotations compared to AVL trees which results in more efficient insertions but tree height get increased.

6.3 AVL Tree Insertion Analysis

6.3.1 Higher Rotations but Lower Height-

Avg rotations required for AVL trees range from 6,977.83 to 6,982,610.50, while height increases from 16.00 to 27.00. Although AVL trees require more rotations for balancing, but maintain a lower height compared to RB trees

6.3.2 Stricter Balancing-

Stricter balancing criteria for AVL trees ensures that avl trees remain more balanced, which is enhancing search efficiency at expense of increased rotation counts during time of insertion.

6.4 Red-Black Tree Deletion Prediction

RB tree deletions will result in decrease in height as balancing operations are performed, which will ensure tree remains efficient. During these deletions, avg no. of rotations required is will be relatively low, which will show ability of tree for maintaining balance without extensive restructuring. This efficiency contributes to consistent performance across varying input sizes, which will allow trees to handle deletions effectively.But it is imp for noting that while the height decreases, R B trees generally maintain a higher height compared to AVL trees. This characteristic will result in fewer rotations during deletions but may lead to slightly less efficient search operations. Overall, R B tree will make them well suited for dynamic datasets where frequent modifications occur.

6.5 AVL Tree Deletion Analysis

6.5.1 Increased Rotations with Maintained Balance-

Avg rotations for AVL tree deletions start at 10,035.42 reach 10,040,960.00 for largest dataset. Despite higher no. of rotations, tree height remains relatively low (ranging from 14.00 to 24.00) as observed.

6.5.2 Height Minimization-

This supports design of AVL trees, which prioritize minimizing height even during deletion operations which make better v ensuring efficient search operations after deletion process too.

6.6 Comparative Analysis of Insertion and Deletion in RB and AVL tree

6.6.1 Insertion Comparison-

RB trees require fewer rotations during insertion (e.g., 5,815.27 vs. 6,977.83 for 10,000 elements) but they maintain higher height as observed.

AVL trees, while requiring more rotations, achieving lower height,which is result in search efficiency.

6.6.2 Deletion Comparison-

During deletion, RB trees will require fewer rotations but in height can grow significantly.

AVL trees, on other side, will incur more rotations, but avl trees keep height lower which made them advantageous for operations that require frequent lookups.

6.6.3 Insertion vs. Deletion-

Both trees will demonstrate that insertions typically require more rotations compared to deletions (already observed in avl insertion deletion). But AVL tree performance in maintaining a balanced structure during deletions is notable, reinforcing use of avl trees in applications where search efficiency is critical.

7 Conclusion

From the results obtained, we can conclude that-

- Randomized quick sort algorithm performs efficiently across large arrays, with expected $\mathcal{O}(n \log n)$ growth in comparisons.
- Red black trees require fewer rotations than AVL trees but maintain higher tree heightwhich makes them more rotation efficient but slightly less balanced.
- AVL trees, with their stricter balancing requirements, maintain a lower height at cost of more rotations.

Choice between red black and AVL trees depends on whether application prioritizes rotation efficiency (red black trees) or minimized height (AVL trees).