Assignment 3

Sayak Sen

2023CSB047

Introduction

- Sorting is a fundamental operation in computer science, used in various applications such as searching, data analysis, and optimization problems. QuickSort is one of the most widely used sorting algorithms due to its average-case efficiency. However, its worst-case performance can degrade to **$O(n^2)$** when an unbalanced pivot is chosen. This report compares **Normal QuickSort** and **QuickSort using the Median-of-Medians pivot selection technique**, analyzing their efficiency in terms of the number of comparisons.

Algorithm Descriptions

- **1. Normal QuickSort**
- QuickSort is a **divide-and-conquer** algorithm that selects a pivot element and partitions the array such that elements smaller than the pivot are on the left and larger elements on the right and then recursively applies the same operation on the partitions.
- **Pivot Selection:** The first element of the array is chosen as the pivot.
- **Partitioning Strategy:** The partitioning is done using two pointers, moving elements smaller than the pivot to the left and larger ones to the right.
- **Time Complexity:**
    - **Best/Average Case:** O(n log n)
    - **Worst Case:** $O(n^2)$ (occurs when the pivot consistently selects the smallest or largest element)

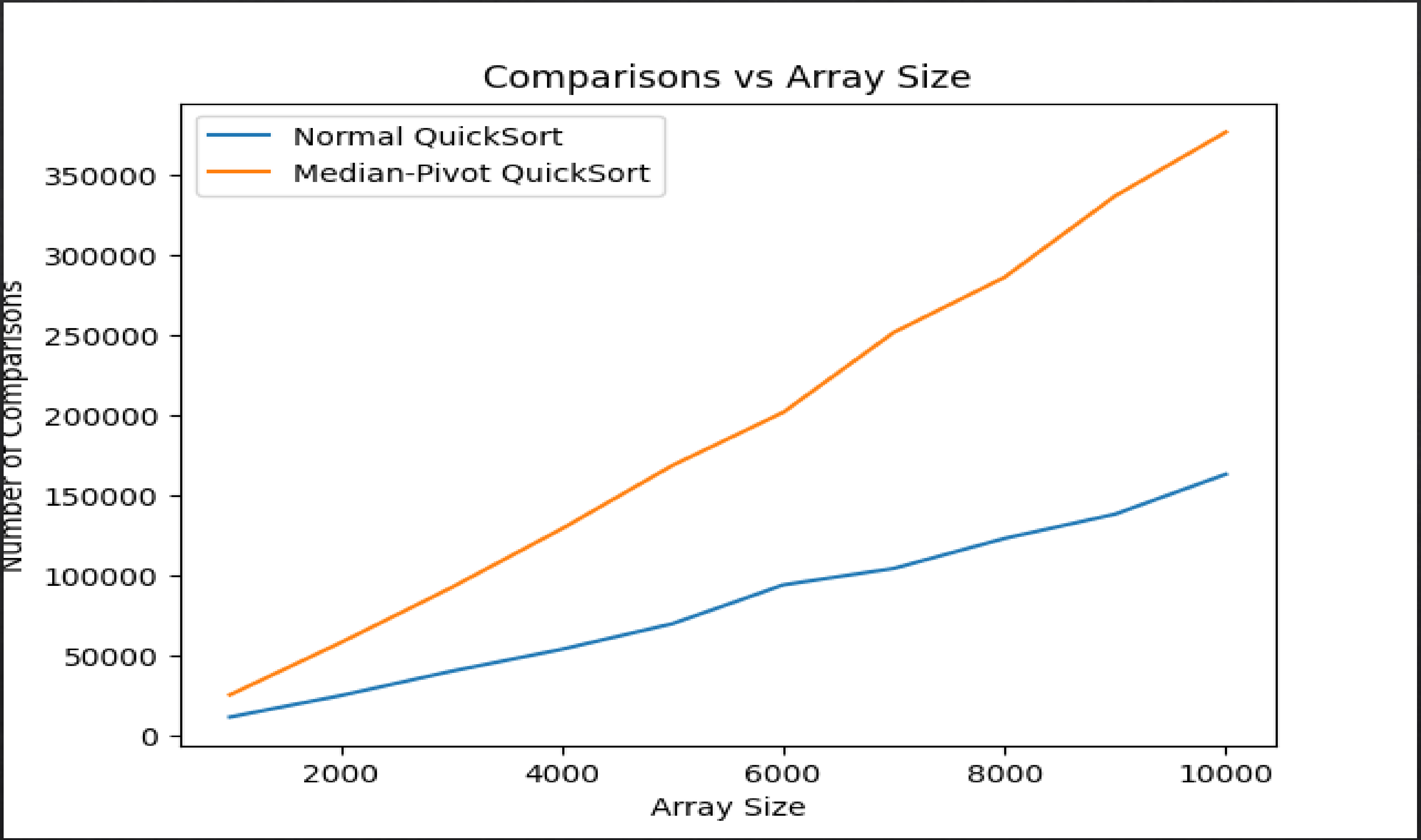
- **2. QuickSort with Median-of-Medians Pivot**
- This variation improves pivot selection by using the **Median-of-Medians** algorithm to find a better pivot reducing the likelihood of worst-case performance.
- **Pivot Selection:** The pivot is determined by first dividing the array into groups of five elements, sorting each group and selecting the median of each group. The median of these medians is then used as the pivot.
- **Partitioning Strategy:** Similar to normal QuickSort elements are rearranged around the selected pivot.
- **Time Complexity:**
  - **Best/Average Case:** O(n log n)
  - **Worst Case:** O(n log n) (avoids O(n^2) degradation)

Results

Array Size	Normal QuickSort Comparisons	QuickSort (Median Pivot) Comparisons
1,000	11,618	25,483
2,000	25,005	58,137
3,000	40,242	92,528
4,000	53,971	129,329
5,000	69,986	168,977
6,000	94,208	202,066
7,000	104,491	252,095
8,000	123,251	286,516
9,000	138,439	337,356
10,000	163,429	377,285

Observations

- **Normal QuickSort is faster in practice:** Despite the risk of $O(n^2)$ performance, the first-element pivot is often efficient in random datasets.
- **Median-of-Medians QuickSort makes more comparisons:** It guarantees $O(n \log n)$ worst-case performance but the overhead of computing the pivot increases the total number of comparisons.
- **Performance Gap Increases with Array Size:** The difference in comparisons grows as the array size increases.


Comparisons vs Array Size
Normal QuickSort
Median-Pivot QuickSort
Number of Comparisons
Array Size
0
50000
100000
150000
200000
250000
300000
350000
2000
4000
6000
8000
10000

Conclusion

- Both algorithms have their strengths and weaknesses:
- **Normal QuickSort** is more practical and performs well on average, making fewer comparisons.
- **Median-of-Medians QuickSort** provides theoretical guarantees against worst-case scenarios but incurs additional computation overhead.
- For real-world applications where worst-case scenarios are rare, **Normal QuickSort** is usually preferable. However, if **deterministic worst-case performance is crucial**, the **Median-of-Medians approach** is the better choice.