Student name: VUONG QUANG PHAT

Student number: AC4933

Student Group: TIC22S1

Title: Comparison of List and Binary Search Tree Data Structures in Sorting Algorithm

Introduction:

In this project, the aim is to compare the effieciency of two different data structures, namely a list and a binary search tree (BST), in the context of a sorting algorithm. Sorting algorithm are fundamental in computer science and are widely used in various applications. The choice of data structure can significantly impact the effeciency and performance of sorting algorithms. We choose a list and a BST for comparison because they represent different approaches to data storage and manipulation.

Data Structure used in the project:

1. **List:** A list is a dymamic array-like data structure that allows for flexible resizing and modification of elements. In Python, lists are implemented as dynamic arrays, making them suitable for storing and manipulating collections of items.
2. **Binary Search Tree (BST):** A BST is a binary tree data structure where each node has at most two children, and the left children is less than the parent, while the right child is greater. BSTs are commonly used for efficient searching, insertion and deletion operations, and they can also facilitate sorting operations through inorder traversal.

Algorithm:

- The algorithm we implemented ultilizes both the list and BST data structures to perform sorting. We constructed a binary search tree from a given list of integers and then performed an inorder traversal of the BST to store the elements in a list. The resulting list contains the elements in sorted order. This algorithm allows us to compare the efficiency of sorting using a list versus a BST.

Efficiency Comparison:

- To Compare the efficiency of the list and BST data structures in sorting, we measured the execution time of the sorting algorithm using both data structures with different input sizes. We conducted experiements with varying input sizes, ranging from small to large datasets, to observe how the efficiency of each data structure scales with the sizes of the input.

Analysis of Results:

- Our results showed that the efficiency of the sorting algorithm using a BST was generally better than using a list, especially for larger datasets. This is because the BST allows for efficient insertion and retrieval of elements, resulting in faster sorting times compared to lists, which may require linear-time operations for insertion and retrieval.
- However, it's important to note that the efficiency of a data structure can also depend on various factors such as the distribution of data, the specific implemetation details, and the operations being performed. While BSTs performed better in our experiments, list may be more suitable for certain scenarios, especially when the overhead of maintaining a balanced BST is not justified.

Conclusion:

In conclusion, the choice of data structure can significantly impact the efficiency and performance of algorithms. In this project, we compared the efficiency of sorting using a list versus and a binary search tree. While BSTs generally outperformed lists in our experiments, the choice of data structure should be carefully considered based on the specific requirements and characteristics of the problem at hand. Further research and experimentation could explore additional factors affecting the efficiency of different data structures in sorting algorithms.