Advanced Sorting Algorithms

James Dungan

August 31st 2023

Contents

1 Merge Sort

Merge Sort is the first sort we'll be looking into. Merge sort takes a base list of values and essentially divides and conquers, it takes that base list and splits the list into 2 smaller lists and those lists will split into smaller lists until each list is a single value. Merge sort then sorts each of these values from least to greatest or however you determine. These smaller lists will combine sorted and then combine with more smaller lists until we reconstuct the original list with sorted values.

2 Quick Sort

Quick sort is a highly efficient sorting algorithm that like Merge Sort uses the divide and conquer technique, meaning the original list breaks down into smaller lists that are easier to sort. Unlike Merge Sort though, Quick Sort uses a pivot. The pivot can be placed anywhere on the list and will move the numbers greater than the pivot to the right and numbers less than the pivot to the left. The Sort uses recursion to sort the list, this sort is very quick given the name, it is often favored above other sorts due to the sheer speed and ability to handle large amounts of elements to be sorted.

3 Heap Sort

A Heap sort is a sorting algorithm based on the binary heap data structure. Essentially Heap sort is finding the highest value and moving it again and repeating that process until its all sorted. The Sort starts by working from the top down just like a binary tree meaning it will split the numbers based on who is greater and the greater the number the farther down they will go on the tree until they are each there own value with there own identifier or value, this value is just a counter for each value starting from 0 at the top of the tree and increasing as you make your way farther down the tree. We then use there values to create an array and the original list should be sorted from least to greatest.

4 Time Complexity Comparison

Time Complexity is defined as the number of times a particular instruction set is executed rather than the total time taken. So lets compare the sorts Time Complexity. Lets start With Merge sort the Time Complexity of Merge Sort is O(N log(N)). The Time Complexity of Quick Sort is O(N log(N)). And finally the Time Complexity of Heap Sort is O(N log(N)). Now if we look at these the Time complexity is the same for all 3 sorts. This means that all 3 of the sorts are exexecuted the same number of times.

5 Space Complexity Comparison

Space Complexity is the total memory space required by the program for its execution. So lets compare the different sorts space complexity. We'll start with Merge Sort, Merge Sort has a Space complexity of O(n). The Space Complexity of Quick sort is O(n). And Finally The Space Complexity of Heap Sort is O(1). Heap sort has a value of 1 because it is a in place sorting algorithm meaning we need no extra data structure. Once we compare Quick Sort and Merge Sorts Space Complexity you might see that they have the same Space complexity. Meaning they need the same amount of memory space for program execution.