Assignment 4 Sorting Algorithms Analysis

Under is the data for sorting an array with a size of 1000000.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Simple Pivot Quicksort | Median of Three Quicksort | Random Pivot Quicksort | Median of Five Quicksort | Merge Sort |
| Best Min Recursion | 20 | 10 | 10 | 10 | 5 |
| Best Average Runtime  (Per sec and rounds up to the thousands) | 0.2854 | 0.2700 | 0.2550 | 0.2380 | 0.5681 |
| Worst Min Recursion | 160 | 160 | 160 | 160 | 160 |
| Worst Average Runtime  (Per sec and rounds up to the thousands) | 0.4625 | 0.4772 | 0.4515 | 0.4356 | 0.8018 |

*[1] data of the array of size 1000000*

Under is the data for sorting an array with a size of 16000000.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Simple Pivot Quicksort | Median of Three Quicksort | Random Pivot Quicksort | Median of Five Quicksort | Merge Sort |
| Best Min Recursion | 5 | 10 | 20 | 10 | 5 |
| Best Average Runtime  (Per sec and rounds up to the thousands) | 8.3677 | 7.9996 | 8.3900 | 7.7085 | 15.8309 |
| Worst Min Recursion | 160 | 160 | 160 | 160 | 160 |
| Worst Average Runtime  (Per sec and rounds up to the thousands) | 11.6782 | 11.0580 | 11.2126 | 12.0243 | 20.6467 |

*[2] data of the array of size 16000000*

As shown in [1] and [2], Merge sort will always have the worst runtime compared with the other four quick sort algorithms. This is because merge sort always makes an extra copy of the array. It will cause a slower runtime. Also, according to the tables shown above and other output data, the Median of Five Quicksort is always the best over the other four sorting algorithms for all the required array sizes. It’s because this algorithm can decrease the chance of choosing a pivot that is either the smallest or largest element in the array. Therefore, the chance of getting a worse case is low.

My empirical time increases match the asymptotic runtime discussed in the lecture. But I think my result still surprised me. Because min recurses of 160 are always the worst case for every sorting algorithm with any array size. As shown in [1] and [2], the best case for Merge sort is always with a min recurses of 5 for array sizes of 1000000 and 16000000. But for the other 3 array sizes, the best case is always min recuses of 10, which surprised me because I thought the best case should always be min recurses of 5 no matter the array sizes.

From the overall solution from the data, I think the smaller min recurses can always give a faster runtime. And the worse case is always produced by the larger min recurses, which tend to be 160.