Cullen Watson

Dr. Fontenot

CS2341-801

9 October 2021

Solving the Sorting Mystery

The task of this assignment was to relate five functions to five different sorting algorithms. However, I had no access to the implementation of the functions, so therefore, my only tool to figure out the corresponding sorting algorithm of each function was through execution time. The five different sorting algorithms that the functions undertook were optimized bubble sort, insertion sort, merge sort, quicksort (with last element as pivot), and selection sort.

The first step to decoding the functions was to split the functions into two major categories. Bubble, selection, and insertion sort all had an average time complexity of $O(n^2)$, while merge and quick sort had an average time complexity of $O(n \log(n))$. To get the average time complexity of each function, I passed an array of randomized integers of size 100,000 and 1,000,000. Because the size increase is a factor of 10, you would expect a function with average time complexity of $O(n^2)$ to have an execution time that increased by a factor of 100. The results showed that Mystery 1 and 4 did not increase by a factor of 100 but something closer to that of $O(n \log(n))$, while Mystery 2, 3, 5 did increase by a factor of around 100. Therefore, I concluded that Mystery 1 and 4 were the $O(n \log(n))$ functions (quick and merge sort), while Mystery 2, 3, 5 were the $O(n^2)$ functions (bubble, insertion, and selection sort).

Mystery01	Random
Input Size	Time
100000	0.00853775
1000000	0.08328229
O(n*log(n))	9.75459483

Mystery04	Random
Input Size	Time
100000	0.00653367
1000000	0.05565575
O(n*log(n))	8.5183022

O(n log(n)) functions

Mystery02	Random
Input Size	Time
100000	10.38180642
1000000	1105.706583
$O(n^2)$	106.5042574

Mystery03	Random
Input Size	Time
100000	0.834168833
1000000	81.82667933
$O(n^2)$	98.09366653

Mystery05	Random
Input Size	Time
100000	9.46219108
1000000	940.373131
$O(n^2)$	99.3821751

$O(n^2)$ functions

After I had grouped the functions into two categories, I began with the $O(n^2)$ category. I knew the best case for insertion and bubble was O(n), leaving selection sort as an outlier. To test this, I created a sorted integer array. I then compared the execution times for arrays of size 100,000 and 1,000,000. I noticed that only Mystery 5 was $O(n^2)$, while the other two mysteries had a time complexity of O(n). Therefore, I was able to conclude that Mystery 5 was selection sort.

Mystery02	Sorted
Input Size	Time
100000	0.00003625
1000000	0.000359167
O(n)	9.908055172

Mystery03	Sorted
Input Size	Time
100000	0.000075
1000000	0.000541875
O(n)	7.225

Mystery05	Sorted
Input Size	Time
100000	9.50424754
1000000	940.536232
$O(n^2)$	98.9595681

Outlier: Mystery05

To distinguish between bubble and insertion sort for the last two functions in this category, I knew that insertion sort was a much more efficient algorithm than bubble sort, so I compared the actual execution times for a randomized integer array of size 1,000,000 and found

Mystery 3 to be much faster. Therefore, I was able to conclude that Mystery 3 is insertion sort and Mystery 2 is bubble sort.

Mystery02	Random
Input Size	Time
1000000	1105.706583

Mystery03	Random
Input Size	Time
1000000	81.82667933

Mystery02's execution time is significantly longer

For the O(n log(n)) category of the last two mysteries, I looked at the worse case scenario for quick sort. Because quick sort's pivot was the last element, I chose an ascending array. By having an ascending array, the pivot was also the largest element in the array, which creates the worst case scenario for quick sort. Running sorted arrays of size 10,000 and 100,000. I noticed the time complexity of Mystery 4 grew by a factor of 100, or in other words, a time complexity of $O(n^2)$. Therefore, I could now conclude that Mystery 4 is quick sort due to its worse case being $O(n^2)$, while Mystery 1 is merge sort because the time complexity remained O(n log(n)) for all test cases.

Mystery01	Sorted
Input Size	Time
10000	0.0002255
100000	0.00238883
O(n*log(n))	10.5519747

Mystery04	Sorted
Input Size	Time
10000	0.02466242
100000	2.38646767
$O(n^2)$	96.7653603

Mystery04 worse case is $O(n^2)$

RESULTS	
Mystery01	Merge Sort
Mystery02	Bubble Sort
Mystery03	Insertion Sort
Mystery04	Quick Sort
Mystery05	Selection Sort