Assignment 3 WRITEUP.pdf

Julian Chop

February 6, 2023

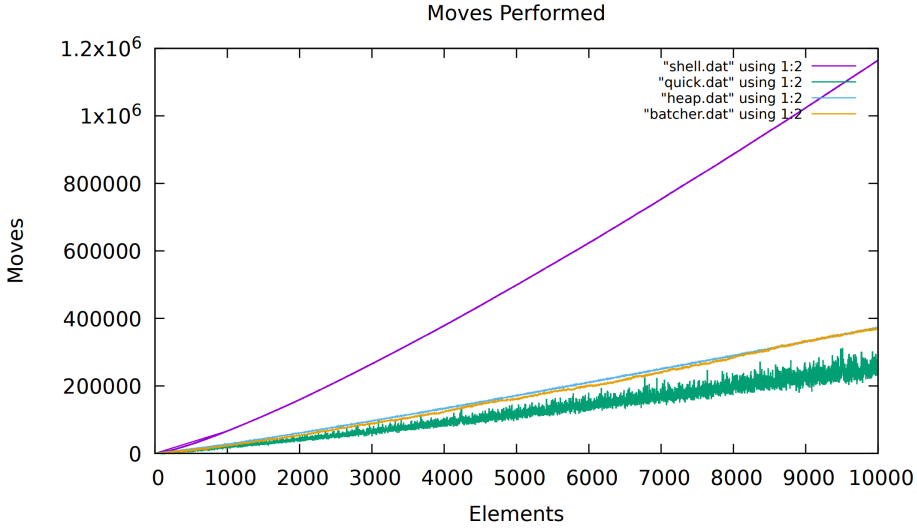
1 We Learning!

In this assignment, I learned about different sorting algorithms as well as how they compare to each other. I looked 4 different sorting functions: shell sort, quick sort, heap sort, and Batcher sort.

Efficiency was observed through 2 variables that we would keep track off while running each of these sorting algorithms. One was the amount of times the algorithm would move the elements of the array around to sort it. The second was the number of times the algorithm would compare two elements of the array. The fewer moves and comparisons, the more efficient the algorithm.

2 Moves Performed

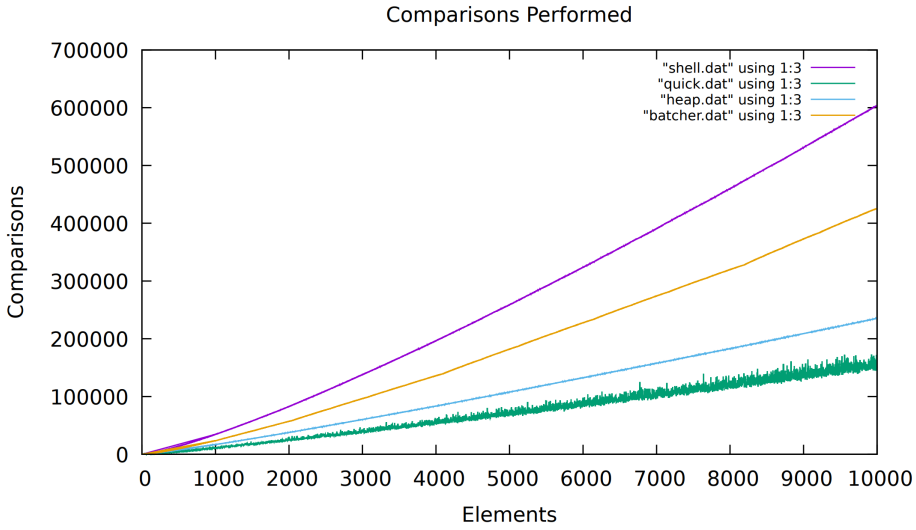
This first graph compares each sorting algorithms moves performed depending on how many elements. The max number of elements shown in this graph is 10000 elements.



Looking at it quickly, you can see the shell sort algorithm taking more moves than any of the others no matter the number of elements in the array. Next up is Batcher and heap which are very similar. Even as the elements increase, both algorithms seem to take the same amount of moves. And then lastly is quick sort which stands out from the rest. You can see how quick sort's line goes up and down as the number of elements increases. This means that there are times when an array with more elements can be sorted with fewer moves than an array with less elements, and vice versa. Regardless, quick sort uses the fewest moves out of the 4 sorting algorithms we looked at.

3 Comparisons Performed

This graph compares each sorting algorithms comparisons depending on how many elements they are sorting. The max number of elements shown in this graph is 10000 elements.



Comparing this graph to the Moves Performed graph, you can see a lot of similarities. For one, shell sort is also leading in number of comparisons on every amount of iterations. After shell sort, we can start to see some differences between graphs. In comparisons, it looks like batcher sort uses more comparisons than heap sort. It actually looks like it uses twice as much as heap sort. Then, at the bottom, using the least amount of comparisons out of the 4 algorithms, is quick sort again. I guess it's called quick sort for a reason. Just like its moves used, quick sort has this zig-zag pattern on the graph. At some points, it uses less comparisons as elements increase, and at some points, it uses more comparisons as elements increase.

4 What Do These Graphs Represent?

Well yes, these graphs do represent the number of moves and comparisons the sorting algorithms used as the number of elements increased, but this is only for a single instance. Every algorithm was given the same 10000 element array which was created through srandom(). This means that these graphs could've looked a lot different if we used a different seed. What if we gave the sorting algorithms an already sorted array? What if we gave the sorting algorithms an array sorted in the reverse order? How long would it take these algorithms to sort these other kinds of arrays?

5 Conclusion

This assignment was a challenge. Not only were the programs we needed to code more difficult than the previous assignment, it also introduced topics that I haven't explored with C yet. Two of these things being pointers/arrays, and the second being sets and bit wise operations. Sets at first were pretty hard to understand, but after a few sections, it began to click. Learning about sets gave me a better way to implement my command options in my test cases; this will definitely be using onward. Pointers on the other hand, still needs some more studying. Overall, this assignment was great in showing me more about c as well as the time complexities of algorithms.

6 Citations/Resources used

- All of the sorting algorithms coded in this assignment was strongly based on the pseudo code given in resources
- I went to office hours and sections to better understand pointers, memory allocation, and bit wise operations. I also went to Varun's tutoring session on Saturday and asked about how I can implement sets into my command-line options in main.