Comparison of Different Sorting Algorithm

Hsiang Yun Lu

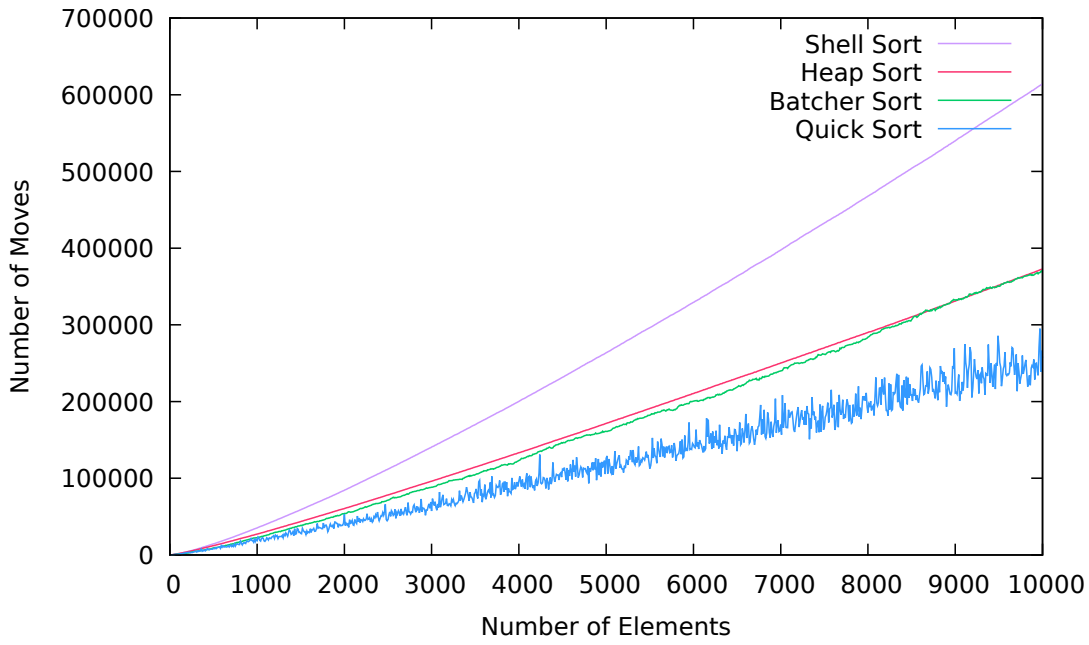
February 6, 2023

1 Sorts Performance on Different Numbers of Input Elements

This graph shows the performance of four different sorting algorithms using various sizes of input arrays. The elements in the input array are randomly generated. The x axis is the number of elements in the input (size of the input array), while the y axis shows the number of moves that each algorithm performs.

The performance lines of Shell Sort and Heap Sort are smooth, while the lines for Batcher Sort and Quick Sort show up and downs. The non smooth lines of Batcher Sort and Quick Sort are the results of the algorithms. Quick Sort partitions arrays into two sub-arrays (elements larger than the pivot and smaller than the pivot) based on the pivot. Similarly, Batcher Sort sorts the even and odd subsequences of the array and merging the sorted even and odd sequences in the end. These characteristics make the performance of Batcher Sort and Quick Sort highly depend on the input elements, which leads to the not so smooth lines in the graph. However, generally all the sorts we discuss here use more moves as the number of elements increases.

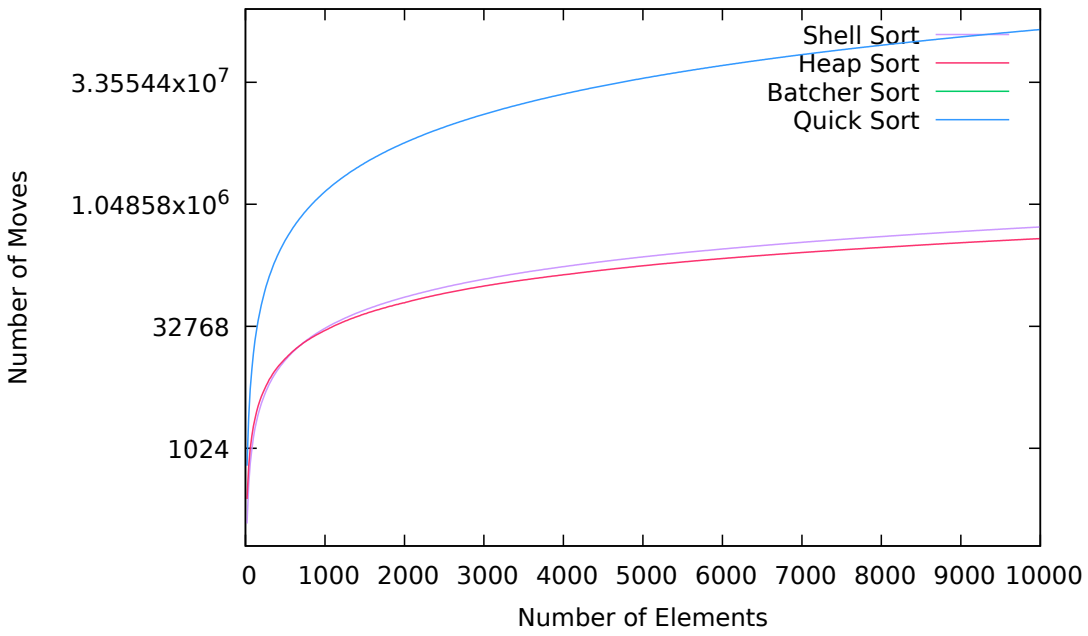
The increase in Shell Sort is the largest among all sorts, while Quick Sort has less increase compared to the others. Quick Sort is faster than other sorts because it breaks an array into smaller ones and swaps the smaller ones. Shell Sort gets slower when the number of input elements increase because as the gaps decreases, Shell Sort will get closer to Insertion Sort – which is slower and needs more moves.



2 Sorts Performance on Input Arrays with reverse Order

This graph as well shows the performance of four different sorting algorithms using various sizes of input arrays. But this time the elements in the input array are already reverse ordered. The x axis is the number of elements in the input (size of the input array), while the y axis shows the number of moves that each algorithm performs.

There is no visible line for Batcher Sort because it performs 0 moves for reverse ordered arrays of all sizes. Quick Sort surprisingly performs much more moves than the other sorts, since it basically needs to exchange all the elements in each partition.



3 Conclusion

In this assignment, I gained more insights on each sorting algorithm and learnt to pay attention to more sophisticated factors that might affect their performance. Also, I understood more about the applications of bitwise operation and memory manipulation.