All Sorts of Sorting Algorithms

Arnav Nepal

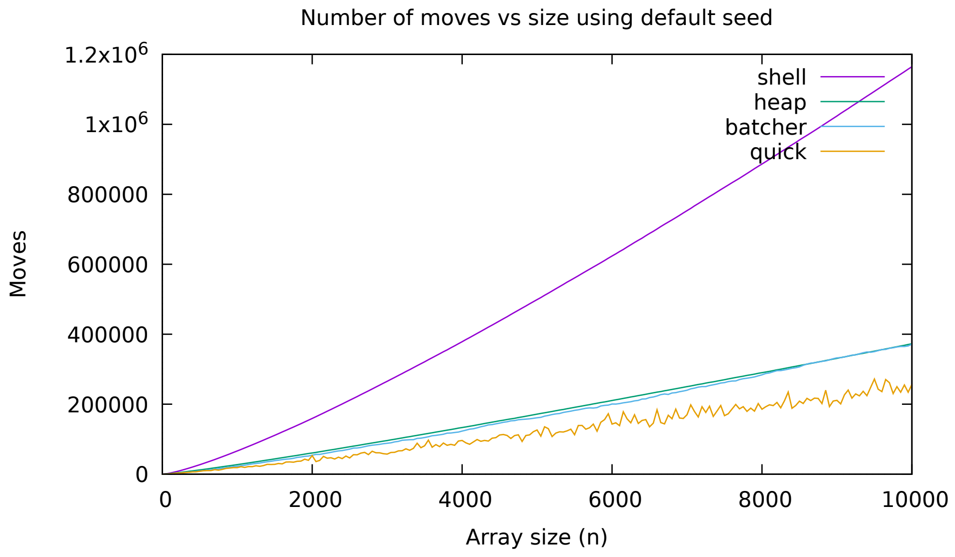
February 6, 2023

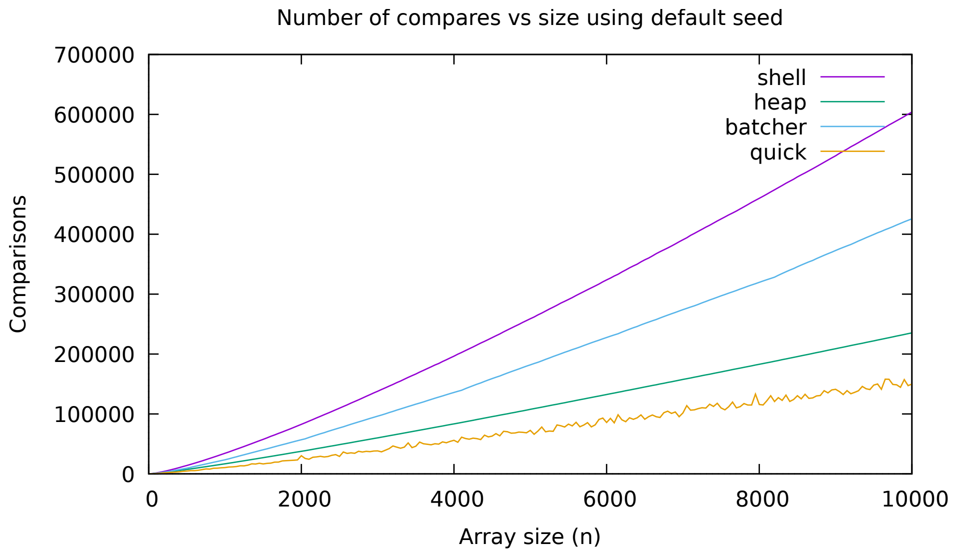
1 Introduction

In this assignment, we were tasked with implementing 4 sorting algorithms. The sorting algorithms we designed include: Quicksort, Heapsort, Shell Sort, and Batcher's Odd-Even Merge Sort. Furthermore We were also tasked with gathering statistics from those sorting methods, including data on the number of compares and moves required by each algorithm. Lastly, we implemented a small set creation and manipulation program for keeping track of command line options and other use in the future. We utilized all these functions and programs in our test harness "sorting.c", where we ran each of the sorts on an array with randomly generated numbers.

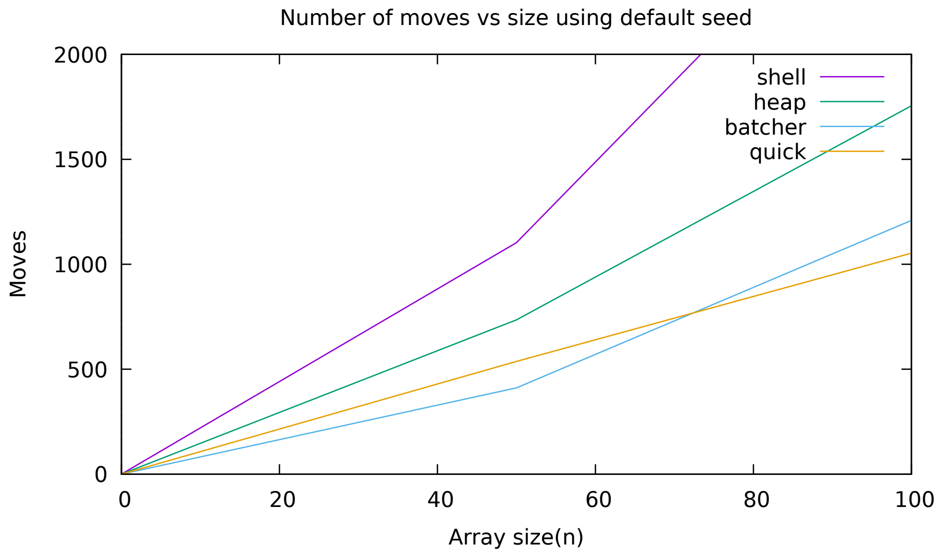
2 The Sorts

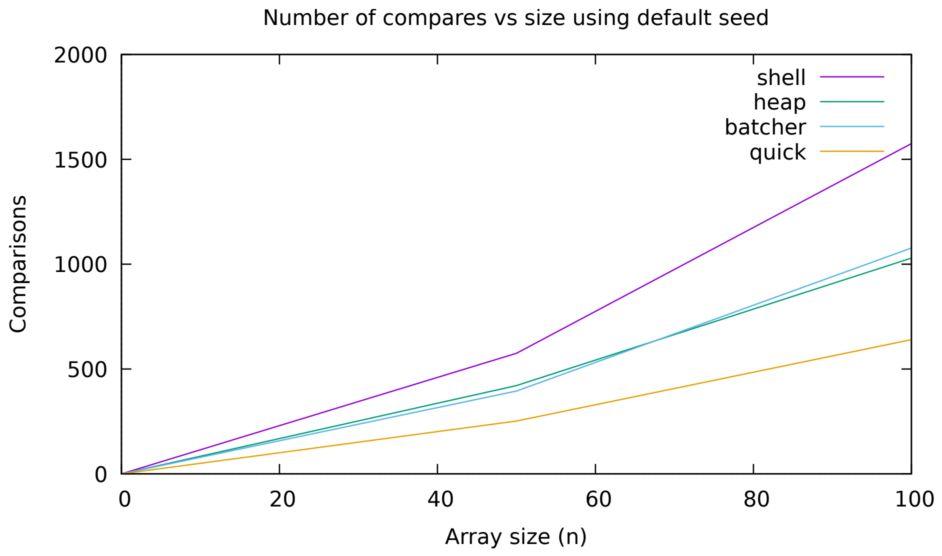
Not all sorts are created equal. Different sorts can have different situations in which they perform well. Of the 4 sorts we created, 3 of them (quick, heap, and batcher sorts) have an average time complexity of $O(n \log n)$. On the other hand, our Shell Sort implementation using the Pratt sequence has a time complexity of $O(n^{\frac{5}{3}})$. The following graphs exemplifies these differences by showing the number of moves and compares required to sort based on the size of the graph:





Based on these graphs, we can see that Shell sort is clearly outclassed by the other sorts in both the amount of moves and comparison, especially when sorting large arrays. This makes sense, since it has a greater average time complexity than the other sorts. On the other hand, quicksort lives up to it's name, performing most efficiently in both moves and comparisons. However, these graphs only show effectiveness at large n. If we limit the x and y range, we can see that:





Based on these graphs, we can come to the conclusion that it is better to use batcher sort on smaller arrays as it performs better in terms of the number of moves. However quicksort quickly overtakes batcher sort at around 70 n. Furthermore, quicksort requires less comparisons than batcher sort even at small n, which might make it the better algorithm to use in case of uncertainty of the size of the array.