Assignment3 - WRITEUP.pdf

Ning Jiang

January 23, 2022

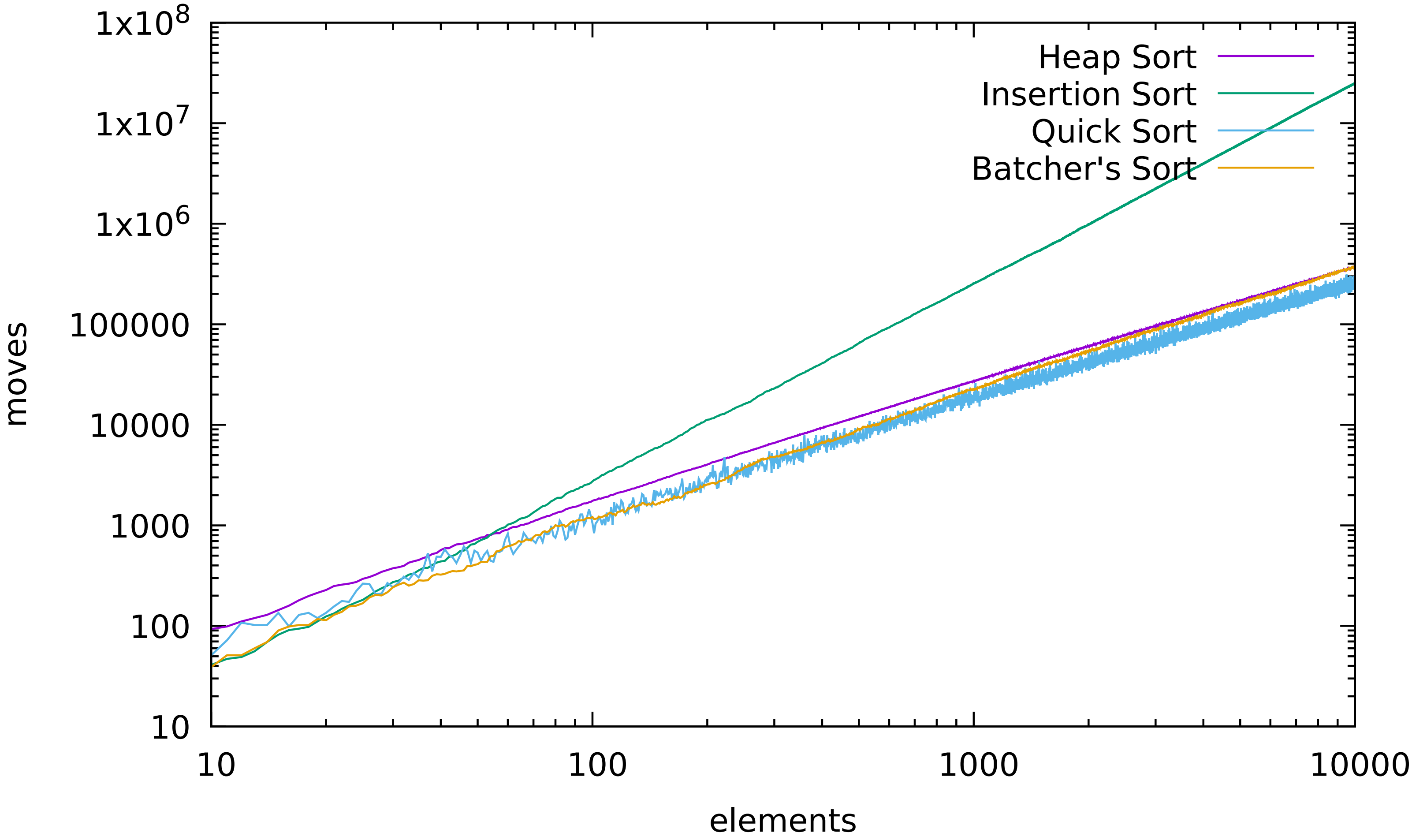
Abstract

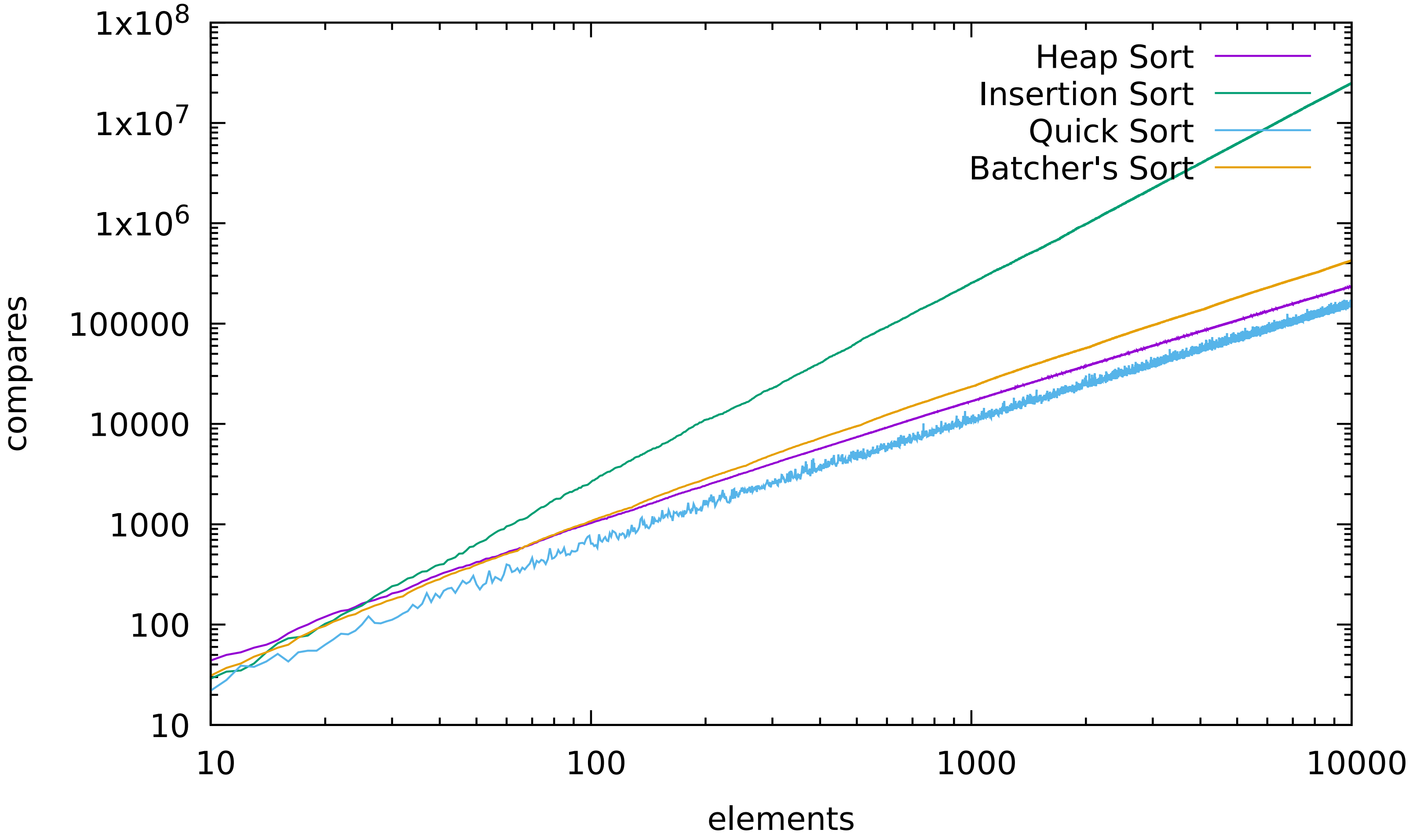
This writeup will include the things I learned from the different sorting algorithms, graphs explaining the performance of the sorts on a variety of inputs, Graphs explaining the performance of the sorts on a variety of inputs, analysis of the graphs I produce.

1 Introduction for each sort (lessons that I learned)

- **Insertion Sort**
  - It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort. Worst case performance: O(n^2).
  - It is a simple implementation and it is Efficient for small data sets (fewer than 10 to 20 elements).
- **Heapsort**
  - Heapsort is a comparison-based sorting algorithm. Although it is slower than a quicksort, it has the advantage of a worst-case performance O($nlogn$).
- **Quicksort**
  - Quicksort is fast.
  - It has worst-case performance O($nlogn$) too.
- **Batcher's Odd-Even Merge Sort**
  - For Batcher's Sort, asmall list will take fewer steps to sort than a large list.
  - Worst case performance: O($nlogn$)

2 Figures


1x10^8
1x10^7
1x10^6
100000
10000
1000
100
10
moves
10
100
1000
10000
elements
Heap Sort
Insertion Sort
Quick Sort
Batcher's Sort


1x10^8
1x10^7
1x10^6
100000
10000
1000
100
10
compares
10
100
1000
10000
elements
Heap Sort
Insertion Sort
Quick Sort
Batcher's Sort

3 Analysis of Figures

- **Insertion Sort**: we can see that when the size of the list increases, the moves and compares of the Insertion Sort increase rapidly which is much more larger than other sorts.

- **Quicksort**: not a stable sort. When size is larger, the compares of Quicksort is smaller than other sorts.

- **Heapsort**: we can see that when the the size of the list is small, the moves of Heapsort is larger than other sorts, however, when the size is large (maybe more than 1000), the moves are very close to other sorts.