Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre

Lab I: Sorting

COMP 314

Submitted by:

Manish Shivabhakti
Roll no: 63
CE, III year/ II Semester

Date: 2nd May, 2024

1. Introduction

We analyze the performance of two sorting algorithms: Insertion Sort and Selection Sort. The performance analysis is conducted based on the execution time of each algorithm for inputs of different sizes. We generate random inputs and consider best and worst-case scenarios to understand how the algorithms behave under various conditions.

2. Experimental Setup

We implemented both the Insertion Sort and Selection Sort algorithms in Python. We generated random input sequences of increasing sizes ranging from 1000 to 14000 elements. For each input size, we recorded the execution time of both sorting algorithms. Additionally, we evaluated the performance of each algorithm for best-case scenarios (sorted input) and worst-case scenarios (reverse sorted input).

3. Results and Analysis

Insertion Sort

Average Case Performance: In the average case, Insertion Sort exhibits a quadratic time complexity, as expected. The execution time increases quadratically with the increase in input elements.

Best Case Performance: Insertion Sort performs efficiently when the input is sorted. In the best-case scenario, its time complexity reduces to linear. As observed, the execution time increases linearly with the input size. Though it looks like the best time case in O(1) in the graph, the graph is drawn on a large scale that the linear time seems like constant time (as shown in Figure 1).

Worst Case Performance: When the input is in reverse sorted order, Insertion Sort demonstrates its worst-case time complexity. The execution time grows quadratically, like the average case, indicating poor performance.

	n	time
0	1000	0.000086
1	2000	0.000172
2	3000	0.000266
3	4000	0.000361
4	5000	0.000444
5	6000	0.000533
6	7000	0.000615
7	8000	0.000703
8	9000	0.000802
9	10000	0.000879
10	11000	0.000976
11	12000	0.001054
12	13000	0.001151
13	14000	0.001259

Figure 1. Best Case Insertion Sort is linear time

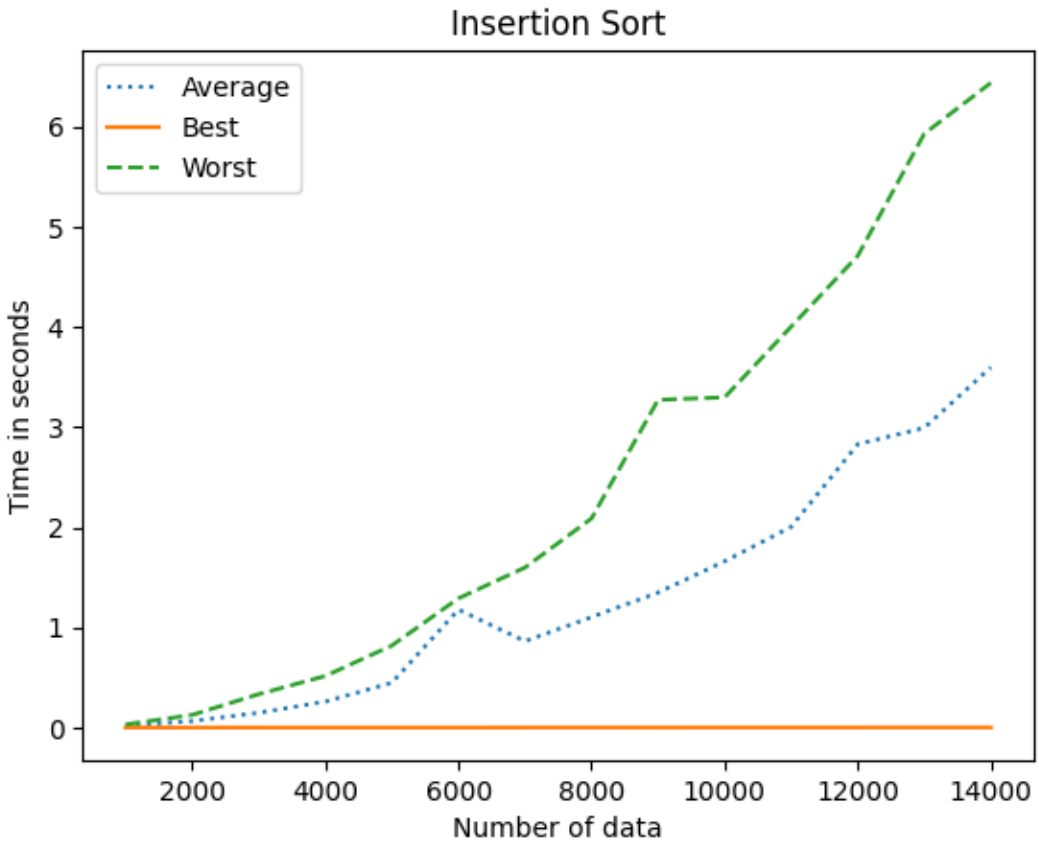

Figure 2. Insertion Sort Against Number of data and time

Selection Sort

Average Case Performance: Selection Sort also demonstrates a quadratic time complexity in the average case. The execution time increases quadratically with the input size, like Insertion Sort.

Best Case Performance: Unlike Insertion Sort, Selection Sort's performance remains quadratic even in the best-case scenario. This is because Selection Sort always scans the entire remaining unsorted portion of the array to find the minimum element.

Worst Case Performance: Like the average case, Selection Sort's performance is quadratic in the worst-case scenario, where the input is in reverse sorted order.

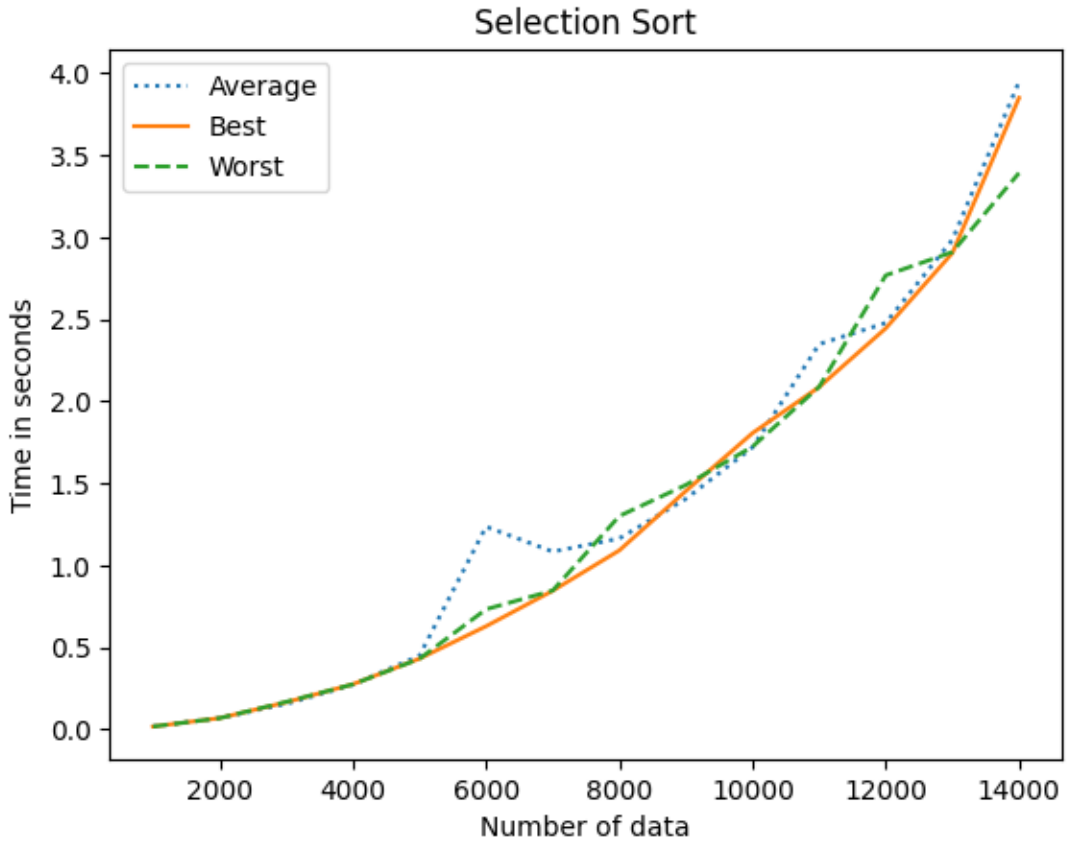

Figure 3. Selection Sort Against Number of data and time

4. Conclusion

a. Both Insertion Sort and Selection Sort exhibit quadratic time complexity in the average and worst-case scenarios except for the best case, in which time complexity for insertion sort is O(n) while selection sort has $O(n^2)$.
b. Insertion Sort outperforms Selection Sort in the best-case scenario due to its linear time complexity when the input is already sorted.

5. Code

GitHub link: https://github.com/TheManysh/algorithm_and_complexity_lab/tree/main/lab-1