



K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch:C2 Roll No.:16010122257

Experiment No. 1

Grade: AA / AB / BB / BC / CC / CD / DD

Signature of the Staff In-charge with date

Title: Implementation of selection sort/ Insertion sort

Objective: To analyse performance of sorting methods

CO to be achieved:

CO 1 Analyze the asymptotic running time and space complexity of algorithms.

Books/ Journals/ Websites referred:

1. Ellis horowitz, Sarataj Sahni, S.Rajsekaran," Fundamentals of computer algorithm", University Press
2. T.H.Cormen ,C.E.Leiserson,R.L.Rivest and C.Stein," Introduction to algortihms",2nd Edition ,MIT press/McGraw Hill,2001
3. http://en.wikipedia.org/wiki/Insertion_sort
4. <http://www.sorting-algorithms.com/insertion-sort>
5. http://www.princeton.edu/~achaney/tmve/wiki100k/docs/Insertion_sort.html
6. <http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/Sorting/insertionSort.htm>
7. http://en.wikipedia.org/wiki/Selection_sort
8. <http://www.sorting-algorithms.com/selection-sort>
9. <http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/Sorting/selectionSort.htm>
10. <http://courses.cs.vt.edu/~csonline/Algorithms/Lessons/SelectionCardSort/selectioncardsort.html>

Pre Lab/ Prior Concepts:

Data structures, sorting techniques.

Historical Profile:

There are various methods to sort the given list. As the size of input changes, the performance of these strategies tends to differ from each other. In such case, the priori analysis can helps the engineer to choose the best algorithm.



K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

New Concepts to be learned:

Space complexity, time complexity, size of input, order of growth.

Topic: Sorting Algorithms

Theory: Given a function to compute on n inputs the divide-and-conquer strategy suggests splitting the inputs into k distinct subsets, $1 < k \leq n$, yielding k sub problems. These sub problems must be solved and then a method must be found to combine sub solutions into a solution of the whole. If the sub problems are still relatively large, then the divide-and-conquer strategy can possibly be reapplied. Often the sub problems resulting from a divide-and-conquer design are the same type as the original problem. For those cases the reapplication of the divide-and-conquer principle is naturally expressed by a recursive algorithm. Now smaller and smaller sub problems of the same kind are generated until eventually sub problems that are small enough to be solved without splitting are produced.

Algorithm Insertion Sort

INSERTION_SORT (A, n)

//The algorithm takes as parameters an array $A[1..n]$ and the length n of the array.

//The array A is sorted in place: the numbers are rearranged within the array

// $A[1..n]$ of eltype, n : integer

FOR $j \leftarrow 2$ TO length[A]

DO $key \leftarrow A[j]$

{Put $A[j]$ into the sorted sequence $A[1..j-1]$ }

$i \leftarrow j - 1$

WHILE $i > 0$ and $A[i] > key$

DO $A[i+1] \leftarrow A[i]$

$i \leftarrow i - 1$

$A[i+1] \leftarrow key$

Algorithm Selection Sort

SELECTION_SORT (A, n)

//The algorithm takes as parameters an array $A[1..n]$ and the length n of the array.



K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

//The array A is sorted in place: the numbers are rearranged within the array
// $A[1..n]$ of eleteype, n : integer

FOR $i \leftarrow 1$ **TO** $n-1$ **DO**

$\min j \leftarrow i$;

$\min x \leftarrow A[i]$

FOR $j \leftarrow i + 1$ **to** n **do**

IF $A[j] < \min x$ **then**

$\min j \leftarrow j$

$\min x \leftarrow A[j]$

$A[\min j] \leftarrow A[i]$

$A[i] \leftarrow \min x$

The space complexity of Insertion sort:

Space complexity = $O(1)$ (in place sorting)

The space complexity of Selection sort:

Space complexity = $O(1)$ (in place sorting)

Time complexity for Insertion sort:

Time complexity =
 $O(n)$ — best
 $O(n^2) \times \frac{n(n+1)}{2}$ — Average
 $O(n^2)$ — worst



K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

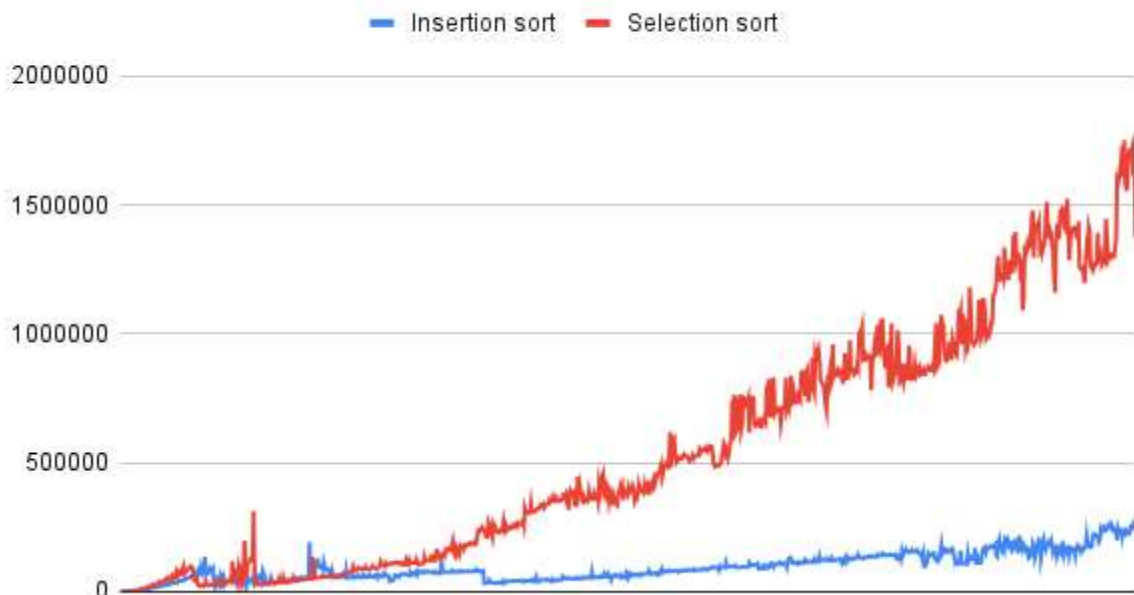
Time complexity for selection sort:

Time complexity :

$$n \times \frac{n}{2} \Rightarrow O(n^2)$$

Graphs for varying input sizes: (Insertion Sort & Selection sort)

Sorting comparing



CONCLUSION: So, we've grasped the methods for evaluating the time complexity of different sorting algorithms. We've also conducted time measurements on a specific dataset and generated graphs. In our comparison between Insertion Sort and Selection Sort, we determined the time and space complexities for both algorithms.