Comparative Study of Different Sorting Algorithms used in Data Structures

Article · March 2022

DOI: 10.2017/IJRCS/202203021

2 authors, including:

Hermehar Pal Singh Bedi
Chandigarh Engineering College
3 PUBLICATIONS **2** CITATIONS

SEE PROFILE

INTERNATIONAL JOURNAL OF RESEARCH CULTURE SOCIETY
Monthly Peer-Reviewed, Refereed, Indexed Journal
Volume - 06, Issue - 03, March - 2022

ISSN: 2456-6683
[Impact Factor: 5.743]
Publication Date: 31/03/2022

DOIs:10.2017/IJRCS/202203021 --:-- Research Article

Comparative Study of Different Sorting Algorithms used in Data Structures

[1]Hermehar Pal Singh Bedi **[2]Amandeep Kaur**
[1] B.Tech student, [2] Assistant Professor
Department of Computer Science and Engineering, Chandigarh Engineering College, Mohali, Punjab, India
Email – [1] hermeharbedi@gmail.com, [2] amandeep.2008@cgc.edu.in

Abstract: *All the programs we see work on some or the other algorithms. Algorithm is the basic formula behind the working of any program. In Data Structures, there are several sorting techniques, which are used in arranging the given data in ascending or descending order. Different Sorting algorithms have been discussed and a comparative study has been formed, which concludes that for sorting large lists, Merge Sort shall be given preference.*

Key Words: *Quick Sort, Merge Sort, Insertion Sort, Bubble Sort, Sorting Algorithms.*

1. INTRODUCTION:

Algorithm can be defined as the set of rules, which are to be followed to solve a given set of problems. Sorting is a way of arranging the given elements of any data structure in an ascendingor descending order. Sorting is a sequence of data that is an important pillar in computer science (1). Depending upon the need, we use different algorithms to sort any given set of data. Sorting consists of different steps, which include comparing the given data elements with each other and then swapping or rearranging them according to our needs. Sorting is of two types: Internal Sorting and External Sorting.

Internal Sorting: In the internal sorting, the data is sorted directly in the main memory.

External Sorting: In the external sorting, the data is sorted in the auxiliary memory (such as the hard disks, pen drives, floppy disks etc) and not the main memory.

Once the data is sorted, then it can be classified into two types: Stable Sorting and the Unstable Sorting.

Stable Sorting: In this type of sorting, the position of similar contents is not changed after the sorting.

Unstable Sorting: In this type of sorting, the position of similar contents is changed after the sorting.

All the algorithms have certain complexities. Complexity of an algorithm can be defined as the measure of the amount of time or space required by an algorithm for an input of given size n.

2. DIFFERENT SORTING TECHNIQUES AVAILABLE:

2.1 Quick Sort: Quick sort was developed by Sir Charles Antony Richard Hoare. Quick Sort is also known as the divide and conquer algorithm or partition exchange sort.

2.1.1 Working of Quick Sort: The given list of items is firstly divided into two halves by choosing the pivot element. The pivot element divides the list in such a way, that on one side of the pivot element, all the elements are smaller and on the other side, all the elements are greater than the pivot element.

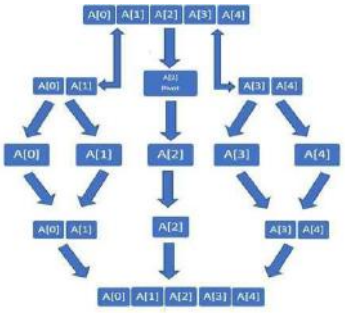

Fig 1: Flowchart of Quick Sort Technique

ISSN: 2456-6683
[Impact Factor: 5.743]
Publication Date: 31/03/2022

2.1.2 Algorithm:

QUICKSORT (A, N, BEGIN, END, LOCATION)

Let us consider an array A with N elements. The parameters BEGIN and END contain the boundary values of sub list A, to which this algorithm is applied. LOCATION keeps track of the position of the first element A[BEGIN] of the sub list during the procedure. The local variables LEFT and RIGHT will contain the boundary values of the unscanned elements.

STEP 1: Start

STEP 2: Set LEFT: = BEGIN and RIGHT: = END

STEP 3: Repeat while A[LOCATION] <= A[RIGHT] and LOCATION! = RIGHT:

Set RIGHT: = RIGHT – 1

[end of loop]

If LOCATION = RIGHT

Then Return

[end of if statement]

If A[LOCATION] > A[RIGHT],

TEMP: = A[LOCATION]

A[LOCATION]: = A[RIGHT]

A[RIGHT]: = TEMP

Set LOCATION: = RIGHT

Go to step 4

[end of if statement]

STEP 4: Repeat while A[LOCATION] >= A[LEFT] and LOCATION! = LEFT:

Set LEFT: = LEFT +1

[end of loop]

If LOCATION = LEFT

Then Return

[end of if statement]

If A[LEFT] > A[LOCATION], then

TEMP: = A[LOCATION]

A[LOCATION]: = A[LEFT]

A[LEFT]: = TEMP

Set LOCATION: = LEFT

Go to step 3

[end of if statement]

STEP 5: Exit

ISSN: 2456-6683
[Impact Factor: 5.743]
Publication Date: 31/03/2022

2.2 Merge Sort: Merge sort was developed by John Von Newmann in 1945. Merge sorting is similar to Quick sort as it also works on the principle of divide and conquer. Merge sort is an efficient comparison-based sorting algorithm.

2.2.1 Working of Merge Sort: In merge sort, the given data set is divided into two equal halves. Merge sorting uses the concepts of swapping of the data elements (if necessary) to sort the elements. Merge sorting requires sequential access rather than random access, due to which it can easily be applied on the lists and the arrays (1).

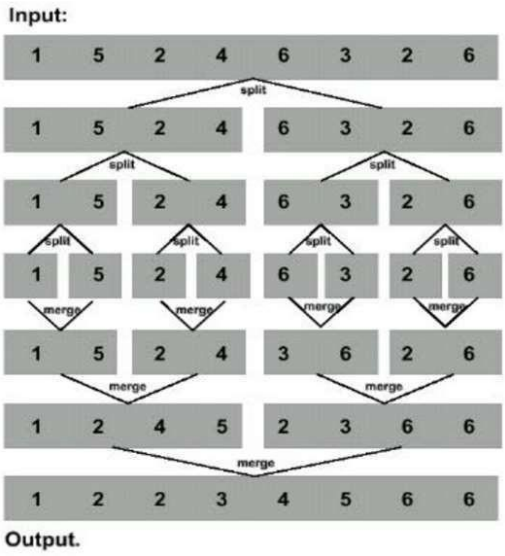

Fig 2: Example showcasing the Merge Sort Technique

2.2.2 Algorithm: In Merge sort, the list is firstly divided into two equal halves. The list keeps on dividing till the time, only one element is left in a set. This step is known as Divide. Once, the stage where a set consisting of one element is reached, then the elements start merging with each other in the same manner as they were divided, but in ascending or descending order depending upon the condition provided. This step is known as Conquer.

2.3 Insertion Sort: Insertion sort is the simplest sorting algorithm. In insertion sort, the finalarray or list is built, by comparing one item at a time.

2.3.1 Working of Insertion Sort: In insertion sort, sorting is done by repeatedly taking the next item and inserting it into the final data by comparing it with the already existing data items.

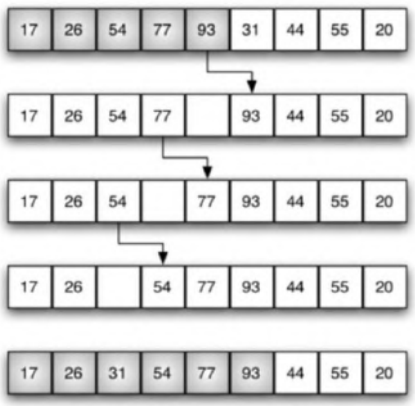

Fig 3: Example Showcasing the Insertion Sort Technique (2)

2.3.2 Algorithm:

Insertion (A, N)

Let us suppose an array A with N elements. Point is the variable, which stores the address of the current data element

ISSN: 2456-6683
[Impact Factor: 5.743]
Publication Date: 31/03/2022

STEP 1: Start

STEP 2: Set A[0]:= - INFINITE

STEP 3: Repeat steps 3 to 5 for K: = 2, 3, …. N

STEP 4: Set TEMP: =A[K] & POINT: =K - 1

STEP 5: Repeat while TEMP<A[POINT]

Set A[POINT+1]: = A[POINT]

Set POINT: = POINT – 1

[end of loop]

STEP 6: Set A[POINT+1]: = TEMP

[end of step 3 loop]

STEP 7: Exit

2.4 Bubble Sort: It is one of the simplest sorting techniques. The term "bubble sort" was introduced by Iverson in 1962 (3). Earlier, bubble sort was known as "Sorting by Exchange" (4) and then by the name of "Exchange Sorting" (5).

2.4.1 Working of Bubble Sort: Bubble sort compares an element with its adjacent elements and swaps the elements if necessary. This process repeats till until no more swapping is possible.

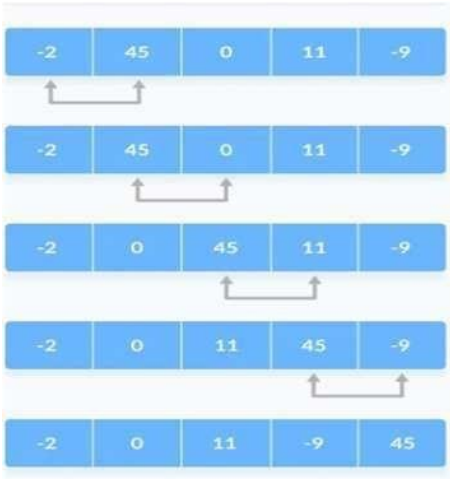

Fig 4: Example showcasing the Bubble Sort Technique (6)

2.4.2 Algorithm:

BUBBLESORT (INFORMATION, N)

Let us consider an array with N elements stored in it. The data available at each location of anarray is represented with INFORMATION.

STEP 1: Start

STEP 2: Repeat the steps 3 and 4 for K = 1 till K = N-1

STEP 3: Set POINTER: = 1

STEP 4: Repeat while POINTER<= N-K:

If INFORMATION [POINTER] > INFORMATION [POINTER +1] then interchange INFORMATION [POINTER] & INFORMATION[POINTER +1]

[end of if statement]

[end of loop started at step 4]

Set POINTER: = POINTER +1

[end of loop started at step 2]

STEP 5: Exit

3. COMPARISONS BASED ON DIFFERENT PARAMETERS:

Quick sort, Merge sort, Insertion sort and Bubble sort can be compared on the basis of different parameters such as the Time Complexity, Space Complexity and Stability.

Table 1: Table Showing Best Case Time Complexity, Worst Case Time Complexity, Worst Case Time Complexity, Space Complexity and Stability of Different Sorting Techniques

Sorting Technique	Best Case Time Complexity	Worst Case Time Complexity	Worst Case Time Complexity	Space Complexity	Stability
Quick Sort	T(N) = O (nlogn)	T(N) = O (nlogn)	T(N) = O (n^2)	O(n)	Unstable
Merge Sort	T(N) = O (nlogn)	T(N) = O (nlogn)	T(N) = O (n^2)	Depends	Stable
InsertionSort	T(N) = O (nlogn)	T(N) = O (nlogn)	T(N) = O (n^2)	O(n)	Stable
Bubble Sort	f (n) = O(n2)	f (n) = O(n2)	f (n) = O(n2)	O(n)	Stable

4. ADVANTAGES AND DISADVANTAGES:

All algorithms have some or the other advantages and disadvantages, which are discussed in Table2.

Table 2: Advantages and Disadvantages of Different Sorting Techniques

Sorting Technique Name	Advantages	Disadvantages
Quick Sort	It is fast and efficient	It is an unstable sort Difficult to choose the pivot element
Merge Sort	It is a fast recursive sorting technique (7)	It consumes more memory space (at least double) (7)
Insertion Sort	Good running time for "almost sorted"arrays Θ(n)	Θ(n2) running time in worst and average case

ISSN: 2456-6683
[Impact Factor: 5.743]
Publication Date: 31/03/2022

5. CONCLUSION:

This paper discussed 4 different sorting algorithms used in Data Structures. Merge sort and Quick sort should be given preference for sorting large lists as compared to the Insertion and Bubble sorts, whereas for short lists, Insertion and Bubble sorts can be given preference. Comparison based on different parameters such as Best-Case Time complexity, Worst Case Time complexity, Worst Case Time complexity, stability and space complexity have been done in this paper.

REFERENCES

Journal Papers:

1. Muhammad Shahzad, Muhammad Shakeel, Um-A-Kalsoom, Atiq Ur Rehman and M Umair Shoukat, (2017): Review on Sorting Algorithms – A Comparative Study. *International Journal ofInnovative Science and Modern Engineering*, 5(1), 17-20.
2. Edward Harry Friend (1956): Sorting on Electronic Computer System. *Journal of the ACM*, 3(3), 134-168.
3. R.C. Bose, R.J. Nelson (1962): A Sorting Problem. *Journal of the ACM*, 9(2), 282-296.
4. Kamlesh Kumar Pandey, Rajesh Kumar Bunkar and Kamlesh Kumar Raghuvanshi (2014): Comparative Study of Different Types of Comparison Based Sorting Algorithms in Data Structure. *International Journal of Advanced Research in Computer Science and Software Engineering*, 4(2), 304-309.

Web Links:

Picture accessed from
https://runestone.academy/runestone/books/published/pythonds/SortSearch/TheInsertionSort.html
Picture accessed from https://www.programiz.com/dsa/bubble-sort

Book:

3. Kenneth E. Iversion (1962). A Programming Language. John Willey.