Design and Analysis of Optimized Stooge Sort Algorithm

Amit Kishor, Pankaj Pratap Singh

ABSTRACT: Data sorting has many advantages and applications in software and web development. Search engines use sorting techniques to sort the result before it is presented to the user. The words in adictionary are in sorted order so that the words can be found easily. There are many sorting algorithms that are used in many domains to perform some operation and obtain the desired output. But there are some sorting algorithms that take large time in sorting the data. This huge time can be vulnerable to the operation. Every sorting algorithm has the different sorting technique to sort the given data, Stooge sort is a sorting algorithm which sorts the data recursively. Stooge sort takes comparatively more time as compared to many other sorting algorithms. Stooge sort works recursively to sort the data element but the Optimized Stooge sort does not use recursive process. In this paper, we propose Optimized Stooge sort to reduce the time complexity of the Stooge sort. The running time of Optimized Stooge sort is very much reduced as compared to the Stooge sort algorithm. The existing research focuses on reducing the running time of Stooge sort. Our results show that the Optimized Stooge sort is faster than the Stooge sort algorithm.

***INDEX TERMS** Arrays, Algorithms, Data structures, Optimization, Sorting.*

I. INTRODUCTION

Sorting is generally defined as the arrangement of data items into ascending or descending order. From the beginning of the computing, sorting has attracted the research work and has been studied extensively. The reason behind this is not only the need of solving very common task but also the challenge of solving the complex task in the most efficient manner. Sorting a list is one of the pillars of Computer Science and Information Technology. Mostly sorting algorithms are comparison based. There too many number of sorting algorithms, but in practical implementation some of the Algorithms predominate.

Sorting algorithms are used depending on the need. Not only in computer science but various sorting tasks are essential in industrial processes. A typical serial sorting algorithm have a good behavior of O(nlogn). Sorting algorithms are recursive, non -recursive or both. Many sorting algorithms are comparison based sorts. A comparison sort compares two data with the help of comparison operator and places them in order. Depending on the working and sorting approach, sorting algorithms are classified. Generally sorting algorithms use insertion, exchange, merging etc.

Revised Manuscript Received on October 30, 2019.
***** Correspondence Author

Amit Kishor*, Assistant Professor in the department of Computer Science and Engineering, Subharti Institute of Engineering and Technology, Swami Vivekanand Subharti University, Meerut, India.

Pankaj Pratap Singh, Assistant Professor in the Department of CSE, Subharti Institute of Engineering and Technology, Swami Vivekanand Subharti University, Meerut, India.

Before using the sorting algorithms on sensitive data, it is usually recommended to confirm whether the sorting algorithm is stable or non-stable. Some sorting algorithms are considerably stable by nature like Insertion Sort [1], Bubble Sort [1], etc.

And some sorting algorithms are not, like Heap Sort [1], Quick Sort [2], etc. Stable Sort algorithms helps in sorting the identical elements in the order in which they appear at the time of input.

An algorithm is a mathematical procedure which serves the computation or construction of the solution to the given problem [5]. There are some algorithm like Hill Climbing algorithms which cannot be predicted [16]. Hill Climbing algorithm can find the solution faster in some cases or else it can also take a large time. Sorting algorithms are prominent in introductory computer science classes, where the number of algorithms for the typical problems provides an introduction to a variety of core algorithm with several concepts, such as divide and conquer algorithms, big O notation, data structures such as binary trees, heaps, randomized algorithms, best case, average case, worst case analysis, lower bounds, upper bounds and time –space trade-offs. Computational complexity analysis and behavior of an algorithm is prominent for the real -time system .Sorting is a very useful operation which is usually performed on the array. The performance of an algorithm is good on small array but if it is used for large array in real time then it may give results in an inadequate time. In the past few years too many attempts have been make and studied to develop a better theory of data make structure [9]. Providing a mathematically rigorous proof of correctness for a small program is a very typical task [10]. In 2001 a new concept of harmonic Search has been introduced, the new solution in generated by three operators [13].There are several factors to be considered in choosing a sorting algorithm, including the programming effort, the size of disk or memory, the size of the array to be sorted, the words count available in the main memory, up to which extent list is already ordered, the repetition of values in the array. Efficient algorithms usually use hybrid algorithm they use several combination of asymptotically efficient algorithm for the sorting of the data with insertion sort for small list. Highly optimized implementations use sophisticated variants. Most of the time in most rendering algorithms are taken by searching and sorting methods [11]. For more sensitive and restricted data, distribution sorts are used. Sometimes, multithreaded algorithm are also, because it is generally expected that multithreading in any algorithm will improve the performance of the algorithm [12]. Any number of practical applications require things in order. Underestimating the need of sorting can lead to undesirable output.

Arrays contain data, these data needed to be arranged in optimal time and hence sorting algorithms are used for this purpose.

Sorting the data makes the problem easy to solve. Arranging the list of items is one of the basic problems in computer science. There are many solutions for sorting data elements. Sorting some tuples can be done based on one or more of the components and the objects are sorted based on the property.

There are many algorithms for sorting data elements. Some sorting algorithms require extra space for temporary storage and matching of data elements. Some sorting algorithms are stable sorting algorithms, and some are non-stable sorting algorithms. Some are easy, such as Selection Sort and Bubble sort. Others, such as Merge Sort and Quicksort are complex to understand. In several application programs, quicksort Implementation are used, these are based on an abstract compare operation, and searching usually use hashing or binary search tree [8]. If we compare the complexity of Bubble sort, Insertion sort, Heapsort, Quicksort, Selection sort [3], and Shell sort [4], we found that although all algorithms have a worst -case runtime of $O(n^2)$, only Shell Sort and Quicksort has a best and average runtime of O(nlog(n)).The efficiency and cost of an algorithm which deals values and words depends on two factors: the strategy of algorithm and the mechanism the algorithm uses to perfect the task [14]. Serial sorting lies very close to the statement serial sorting phase is used to order the element of each in a very less [15]. There are several problems related algorithm, one problem can be solved in many ways with many different algorithm [7]. All the sorting algorithms are problem specific. Some algorithms work better on the partially sorted array, some work better on numeric data, some work better on less number of data. Most studies of sorting algorithms derive an expression for the average or expected sorting effort for an arbitrary unsorted list. Stooge sort is a recursive sorting algorithm. Stooge sort is slower compared to the other sorting algorithms, such as Quick sort, Shell sort, Radix sort and Insertion sort.

II. RELATED WORK

Stooge sort is a very slow sorting algorithm. The algorithm of Stooge sort is given below:

- If the value at the start is larger than the value at the end, swap them.
- Perform Stooge sort the initial 2/3 of the list.
- Perform Stooge sort the final 2/3 of the list.

It is a very inefficient sorting algorithm and has a time complexity of O(N 2.71) which is lower than other sorts [5]. It is important to take the ceiling value of 2/3 otherwise the sort may fail for certain data. Stooge sort uses the recursive technique to arrange the data. Asymptotic analysis is a method to describe the behavior, limits and the performance of the algorithm which is used for a considerable number of values or data sets [6].

A. ANALYSIS OF STOOGE SORT

The running time complexity of Stooge sort is written as,

$$T(n) = 3T\left(\frac{2n}{3}\right) + \theta(1) \text{ ———————— } (1)$$

To find the solution, Master Theorem is used [1].
From the recurrence (1),

a=3 and b=1.5

Applying Master Theorem:

f(n) =1,

$PO(n^{\log_b a})$, where a=3, b=1.5
$O(n^{\log_b a}) = n^{2.7095}$

On applying Case-1 of Master Theorem. Solution of above recurrence is $O(n^{2.7095})$.
$T(n) = O(n^{2.7095})$
Stooge sort is slower than Selection Sort and Bubble Sort.
The algorithm checks whether the array contains 3 or more elements. If more than 3 elements are present, then the recursive call is performed for the initial 2/3 of the list, final 2/3 of the list and again initial 2/3 of the list.

III. PROBLEM DEFINITION

Several number of practical application in computer science many things to be in order. While Software development, various types of problem occurs, various algorithms solve these problem. Searching and sorting algorithm are used to solve these problem In case of searching, linear search, Binary search, Bisecting Linear search are used as problem solvers [17]. Even before the evolution of computer science, there were too many uses and examples of sorting .Stooge sort may be used to accomplish several objectives. The most common uses of sorted sequences are making search operations efficient, enable processing of data in a defined order, merging of sequences etc. When we comparing various sorting algorithms, there are several things to be consider .The first is usually run -time .While dealing with immensely large sets of data, inefficient sorting algorithms can make the application very slow.

The second and prominent Consideration is memory space and in-place algorithm works well in this consideration and avoids the excessive copy of data .Real time machines and software's such as rocket software , airplane software avoid using very slow sorting algorithms .In this paper, the optimized algorithm is proposed for achieving the objectives faster .The goal of this paper is to reduce the overall execution time .The execution time depends on the size of the dataset ,but the time taken to sort the dataset by Stooge sort is reduced when performing the operations of the Optimized Stooge sort on the same dataset.

IV. THE PROPOSED OPTIMIZED ALGORITHM

The Optimized Stooge sort comprises of three methods. First is the Exchange method, second is the Forward method and third is the backward method. In the Exchange method, the first element is compared and swapped with the last element. Then the second element is compared and swapped with the second last element of the array. Similarly, in the next iteration, the third data element is compared and swapped with the third last element of the array.
This swapping depends upon the sorting need (ascending or descending). This process continues till all the elements are compared and swapped. In case of an odd size array, the middle value is not compared with any other value. Depending on the sorting need (ascending or descending), the swapping is performed. The Forward method is invoked after the Exchange method. In the first pass, the array compares and swaps the first and the second last element.

The second and the third last element is compared and so on. Similarly, in the next pass, the first and the third last element is compared and so on. The Backward method is invoked after the Forward method. In the first pass, the array compares and swaps the last and the second data element. The second last and the third data element is compared and so forth. Similarly, in the next pass, the last and the third data element is compared and swapped.

The Optimized Stooge sort is given below:

```
int[ ] exchange(int A[ ])
        Length← A.length
        i← 0
        j ← Length-1
        while(i<j)
        if (A[i]>A[j])

temp← A[i]
        A[i]  ← A[j]
        A[j]  ← temp
        i← i+1
        j← j-1
A=forward(A,0, Length -2)
A=backward(A,1, Length -1)
return A
int[] forward(int A[],int i,int j)
while(i<j)
        x← j
while(i<x)
if (A[i]>A[x])
        temp← A[i]
        A[i]  ← A[x]
        A[x] ← temp
        i← i+1
        x← x-1
i← 0
j←j- 1 return A int
[] backward(int A[],
int i, int j)
        while(i<j)
c ← i
while(c<j)
if (A[c]>A[j])
        temp ← A[c]
        A[c] ← A[j]
        A[j]  ←  temp
        c← c+1;
        j← j-1;
        i← i+1
j← Length-1
return A
```

In the Exchange method, i is initialized to zero and j is initialized to Length-1. After each iteration i is incremented and j is decremented by one. Both i and j position is matched and swapped. The loop continues till i is less than j. The method Exchange takes only one parameter and that is the array in which data elements are present. The Forward method takes three parameters. First is the array which is modified by the Exchange method. Initially i has zero and j has Length -2, but after each iteration i is initialized to zero and j is decremented by one. When the internal loop terminates, i is again initialized to zero and j is decremented by one. The first loop continues, but when i is equal to j or i is greater than j the loop terminates. The method backward takes three parameters. First is the array which is modified by the forward method. The second parameter is i, in this method i has the value one. The value of i is the index of the second cell of the array. The third parameter is j, in this method the value of j is Length(A) -1. When the value of i position is larger than the j value, Swapping is performed. The loop terminates when i is greater than or equal to j.

A. WORKING

Fig. 1 shows the working and control flow steps of Exchange method. The 0th index element is compared and swapped with the 6th index element. Similarly, the 1st index element is compared and swapped with the 5th index element. Swapping of elements will be performed according to the sorting requirement (ascending or descending). Fig.1 is an odd size array, so the middle cell remains, but in case of an even sized array, there is no middle cell. In Fig. 2 the working and control flow of Forward method is shown. In the first pass, the first element is compared and swapped with the second last element. The second element is compared and swapped with the third last element and so on. In the second pass, the first element is compared with the third last element. The second element is compared and swapped with the fourth last element and so forth.

Fig. 3 shows the working of backward method. In the first pass, the second element is compared and swapped with the last element. The third element is compared with the second last element and so on. In the second pass, the last element is compared and swapped with the third element. The fourth element is compared and swapped with the second last element and so forth.

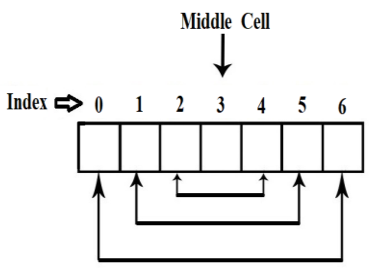

FIGURE 1: Exchange Method

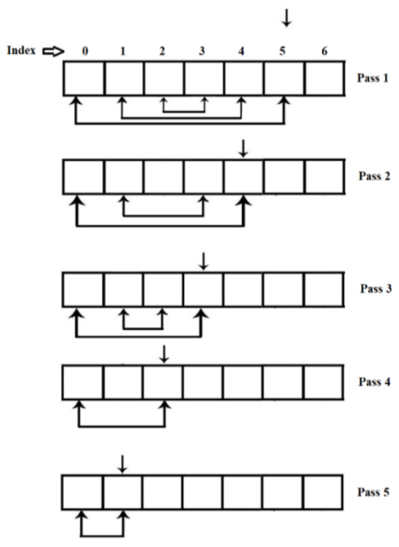

FIGURE 2: Forward Method

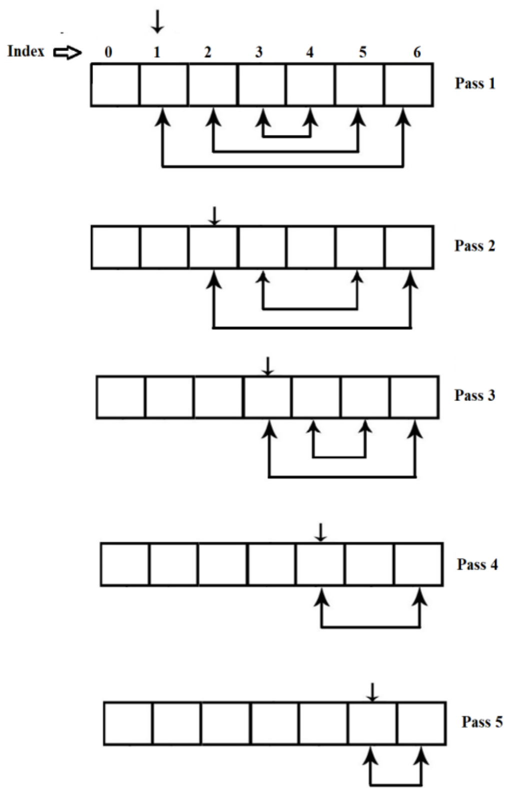

FIGURE 3: Backward Method

B. ANALYSIS

Analysis of the Optimized Stooge sort is easy to understand. There are 3 methods to perform the sorting operation. Forward method and Backward method are called inside the exchange method. So, first we analyze the Forward method and Backward method. In first pass of Forward method, (n-2)/2 comparisons are made as the array length is n and loop runs for n-1 times only. The second pass makes (n-3)/2 comparisons as the loop runs for n-2 times. The third pass makes (n-4)/2 comparisons as now the loop runs for n-3 times and so on. Here the number of iterations reduces each time a pass complete its execution. Therefore, there is a total of

$(n-2)/2 + (n-3)/2 + (n-4)/2 +. . + 1= n^2$----(2)

The above series is in Arithmetic progression. On solving above, n^2 is obtained. The number of comparisons in Backward method is similar to Forward method. The first pass makes (n-2)/2 comparisons as the length of array is n but the

loop runs only for n-1 times only. Second pass makes (n-3)/2 comparisons as the loop runs for n-2 times. Therefore, there is a total of

$(n-2)/2 + (n-3)/2 + (n-4)/2 +. . . + 1= n^2$—(3)

The above series is in Arithmetic progression. On solving above, n^2 is obtained.

```
int[] exchange(int A[])
1.  Length ← A.length
2.  i← 0
3.  j← Length-1
4.  while(i<j)                          }
        4.1. if (A[i]>A[j])             }
            4.1.1.  temp← A[i]          }  n/2
            4.1.2.  A[i] ← A[j]         }
            4.1.3.  A[j] ← temp         }
        4.2. i← i+1                     }
        4.3. j← j-1                     }
5.  A=forward(A,0, Length -2)           } n^2
6.  A=backward(A,1, Length -1)          } n^2
7.  return A
```

FIGURE 4: Analysis of Exchange Method

In the Exchange method, there is only one pass which makes n/2 comparisons as the loop runs for n/2 times.

The method calls Forward method which gives n^2 and Backward method give n^2. The most significant term is taken, hence the sort may therefore categorized as $O(n^2)$.

C. EXAMPLE

An example is given: Some random data are taken in an array.

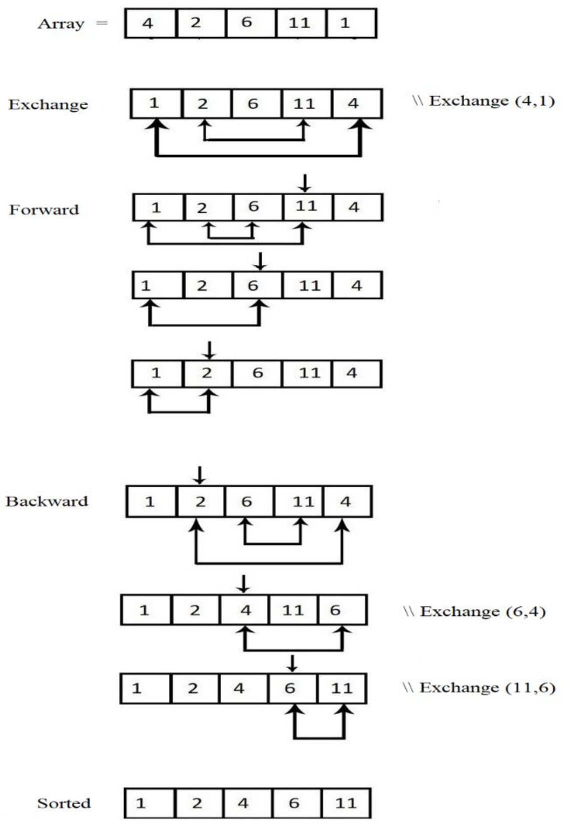

FIGURE 5: Example of Optimized Stooge sort

These data are sorted with the help of Optimized Stooge sort algorithm. The Exchange, Forward and Backward operations are applied to the data. The data are being sorted in ascending order.

D. RESULTS

The algorithms are tested on a machine with 64-bit Operating System having Intel(R) Core(TM) i5-5250U CPU @1.60GHz. Both the algorithms were given the same data, and their processing time was calculated.

TABLE 1: Time taken by Stooge sort and Optimized Stooge sort

Dataset	No. of Instances(n)	Stooge sort(second)	Optimized Stooge sort(seconds)
Dataset 1	100	0.005403832	0.000309835
Dataset 2	200	0.006640606	0.000835849
Dataset 3	300	0.011551138	0.001860935
Dataset 4	400	0.034077365	0.003505049
Dataset 5	500	0.093157072	0.005558428
Dataset 6	600	0.086505561	0.006493065
Dataset 7	700	0.074286	0.008156423
Dataset 8	800	0.21977697	0.009491345

The table shows the time taken by Stooge sort and Optimized Stooge sort. Fig. 5 is the graph plotted in the response to the data given in the table. The graph shows the performance and time consumption of both the algorithm.

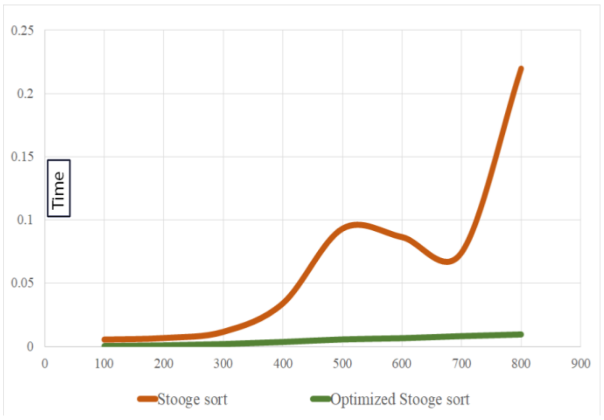

FIGURE 6: Time analysis graph of Stooge sort and Optimized Stooge sort

V. CONCLUSION

To sort large data, the appropriate decision for the selection of sorting algorithm is to be taken to minimize the total execution cost. The Stooge sort algorithm takes huge time when works for large data. The Stooge sort is compared with the Optimized Stooge sort in Table 1. It shows that the Optimized Stooge sort is faster as compared to the Stooge sort. In Fig. 5, it can be observed that the time taken by Stooge sort is more, but the time taken by Optimized Stooge sort is less. The Optimized Stooge sort is not are cursive algorithm, whereas the Stooge sort is a recursive algorithm .The Stooge sort has only one method but the Optimized Stooge sort has three methods. These three methods sort the data faster than the Stooge sort.

REFERENCES

1. Cormen, Thomas H., Leiserson, Charles E., Rivest , Ronald L., Stein, Clifford (2001) [1990]. Introduction to Algorithms ,3rd ed., MIT Press and McGraw-Hill. pp. 161–162. ISBN 0-262-03293-7.
2. Moller F., Analysis of Quicksort, McGraw Hill,2001.
3. Yedidyah Langsam,Moshe J. Augenstein, Aaron M. Tenenbaum., Data structure using C and C++.
4. Janson, S., Knuth, D. E.: Shellsort with three increments. Random Structures and Algorithms 10 (1997), 125âA¸S142. EE 6
5. Peter GAraces and L.At asziar to LovAt asz,... Complexity of Algorithm. Spring 1999.
6. Mari Wahi., Python and Algorithms(2013).
7. Simon Harris, James Rosaa., Beginning Algorithm.
8. Jon I.. Bently, Robert Sedgewick.. Fast Algorithm for Sorting and Searching String.
9. Ben Shneiderman., A Model for Optimizing Indexed File Structures, International Journal of Computer and Information Science, Vol, 3, No 1,1974.
10. Sartaj Sahni.. Data Structure, Algorithm and Applications in C++, Second Edition.
11. Vlastimil Havran,JiEGrAt As Bittner.Efficent Sorting and Searching in Rendering Algorithms,2014.
12. Kevin Jouper,Henrik Nordin,, Performance analysis of multithreaded sorting algorithms,BCS-2015-05
13. Ari Hong,Donghwi Jung,Jiho Choi and Joong Hoon Kim..Sensitivity'.
14. Thu Hien NguyenThi .Toward a realistic analysis of sorting And searching algorithm 2014
15. Andrew Tridgell, Efficient Algorithm for Sorting and Synchronization. February 1999
16. [16] JEFF EDMONDS, HOW TO THINK ABOUT ALGORITHM ,2008.
17. Rahul Sharma, Rohit Kumar, Design and Analysis of Bisecting Linear Search for Sorted Array.e-ISSn: 2395-0056,p-ISSN: 2395-0072, Apr 2018.

AUTHORS PROFILE

Amit Kishor is working as Assistant Professor in the department of Computer Science and Engineering, Subharti Institute of Engineering and Technology, Swami Vivekanand Subharti University, Meerut, India. His area of interest is Cloud Computing, Algorithms, Artificial Intelligence and Data Structure.

Pankaj Pratap Singh received his B. Tech in Computer Science Engineering from U.P.T.U, Lucknow, India and M. Tech in Medical Image and Image Processing from IIT Kharagpur. He is currently working as Assistant Professor in the Department of CSE, Subharti Institute of Engineering and Technology, Swami Vivekanand Subharti University, Meerut, India. His research interests include IOT, Neural Network, Machine Learning, Deep Learning, Image Processing techniques, Cognitive Science, Computer Network and Data Mining Techniques