AN ENHANCED SELECTION SORT ALGORITHM

Ramesh M. Patelia¹, Shilpan D. Vyas², Parina S. Vyas³, Nayan S. Patel⁴

¹Department of M.Sc. (IT) & HOD of BCA, AMCOST, Anand, Gujarat, (India)
²Department of M.Sc. (IT) & BCA, AMCOST, Anand, Gujarat, (India)
³Department of E&C, Shree P.M. Patel College of E&C, Anand, Gujarat, (India)
⁴Department of M.Sc. (IT), B.N. Institute of Paramedical and Science, Anand, Gujarat, (India)

ABSTRACT
Sorting is a commonly used operation in field of computer science. Sorting is the process to arrange the data in either ascending or descending order. Sorting technique is a technique that does the sorting. The names of sorting techniques are Selection sort, Insertion sort, Bubble sort, Shell sort, Merge sort, Quick sort and many more. There is no one sorting method that is best for every situation. In this paper, an enhanced version of the selection sort algorithm is presented. It is analyzed and tested before presenting here. Enhanced selection sort reduced the iteration by half times.

Keywords – Enhanced Selection Sort, Permutation, Selection Sort, Sorting, Time Complexity.

I. INTRODUCTION

A sorting algorithm is an algorithm that puts elements of a list in a certain order. The output must satisfy two conditions:
1. The output is in a particular order either ascending or descending.
2. The output is a permutation (reordering) of the input.

Two main factors on which the performance of a program depends are the amount of computer memory consumed and time required to execute it successfully. Time complexity of a program is the amount of time required to execute successfully.

II. SELECTION SORT

2.1 Concept
It is the simplest sorting technique. A selection sort selects the element with the lowest value and exchanges it with the first element. Then, from the remaining n-1 elements with the smallest key is found and exchanged with the second element, and so forth. The exchange continues to the last two elements.

2.2 Algorithm
The following algorithm is to arrange the data in ascending order of an array A which is consisting of N elements. The variable PASS and MIN_INDEX are pass index and position of smallest element in an array, respectively. The variable I is used to index of an array. All variables are integer.
1. Repeat step 2 to 4 for PASS = 1 to N-1
2. Set MIN_INDEX = PASS
3. Repeat for I = PASS+1 to N
   Then
       MIN_INDEX = I
4. If ( PASS != MIN_INDEX )
   Then
       Interchange A[ PASS ] and A[ MIN_INDEX ]
5. EXIT

2.3 Tracing of a Data

For example, if the Selection sort were used on array, 90, 80, 100, 20, 30, 10, 40, 50, 70, 60, each pass would be like as shown in Table-1.

<table>
<thead>
<tr>
<th>Unsorted</th>
<th>Initial</th>
<th>90</th>
<th>80</th>
<th>100</th>
<th>20</th>
<th>30</th>
<th>10</th>
<th>40</th>
<th>50</th>
<th>70</th>
<th>60</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pass</td>
<td>Pass1</td>
<td>10</td>
<td>80</td>
<td>100</td>
<td>20</td>
<td>30</td>
<td>90</td>
<td>40</td>
<td>50</td>
<td>70</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Pass2</td>
<td>10</td>
<td>20</td>
<td>100</td>
<td>80</td>
<td>30</td>
<td>90</td>
<td>40</td>
<td>50</td>
<td>70</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Pass3</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>80</td>
<td>100</td>
<td>90</td>
<td>40</td>
<td>50</td>
<td>70</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Pass4</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>100</td>
<td>90</td>
<td>80</td>
<td>50</td>
<td>70</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Pass5</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>90</td>
<td>80</td>
<td>100</td>
<td>70</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Pass6</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>80</td>
<td>100</td>
<td>70</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td>Pass7</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>100</td>
<td>80</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td>Pass8</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>100</td>
<td>90</td>
</tr>
<tr>
<td>Sorted</td>
<td>Pass9</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>100</td>
</tr>
</tbody>
</table>

In above table, the underline elements are indicates that only those elements are arranged in their proper order in corresponding iteration. The array is sorted at the end of the 9th iteration of an outer pass loop.

2.4 Analysis

The time required to sort the data does not depend upon the order of the elements. There is no difference in the number of iteration for a list which is sorted already or which has its elements in reverse order. The only difference will be with the number of comparison and with the assignment statements. Their execution frequency will be varying. Total number of comparison is n(n-1)/2. The best case, average case and worst case are same and they are

\[ F(n) = (n-1) + (n-2) + (n-3) + \ldots + 1 \]

\[ = \frac{n(n-1)}{2} \]

\[ = O(n^2) \]
III. ENHANCED SELECTION SORT

3.1 Concept
The main idea behind the enhanced selection sort is that successive elements are selected on both the side of an array or file and placed in their proper position. In this technique, we sort the data in a single pass by two ways as shown below:
1. To find the minimum element from the array and replaced with first element.
2. To find the maximum element from the array and replaced with last element.
This will place smallest element at the top and largest element at the bottom of an array after completion of first cycle. The repetition of above work successively gets the desired result.

3.2 Algorithm
The following algorithm is to arrange the data in ascending order of an array A which is consisting of N elements. The variable MIN_INDEX and MAX_INDEX are position of smallest & largest element in an array, respectively. The variables FIRST and LAST are position of starting & ending element in an unsorted array, respectively. The variable I is used to index of an array. All variables are integer.

1. Set FIRST = 1 and LAST = n
2. Repeat step 3 to 10 for PASS = 1 to N/2
3. Set MIN_INDEX = FIRST
4. Repeat for I = FIRST+1 to LAST
   Then
       MIN_INDEX = I
5. If ( PASS != MIN_INDEX )
   Then
       Interchange A[ FIRST ] and A[ MIN_INDEX ]
6. Increment FIRST = FIRST + 1
7. Set MAX_INDEX = FIRST
8. Repeat for I = FIRST+1 to LAST
   Then
       MAX_INDEX = I
9. If ( PASS != MAX_INDEX )
   Then
       Interchange A[ LAST ] and A[ MAX_INDEX ]
10. Decrement LAST = LAST - 1
11. EXIT

3.3 Tracing of a Data
For example, if the Enhanced Selection sort were used on array, 90, 80, 100, 20, 30, 10, 40, 50, 70, 60, each pass would be like as shown in Table-2.
### Table-2 Tracing Of Data for Enhanced Selection Sort

<table>
<thead>
<tr>
<th>Pass Number(I)</th>
<th>Unsorted</th>
<th>Pass1</th>
<th>Pass2</th>
<th>Pass3</th>
<th>Pass4</th>
<th>Sorted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initial</td>
<td>90</td>
<td>80</td>
<td>100</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>60</td>
<td>20</td>
<td>30</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>80</td>
<td>30</td>
<td>70</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>60</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>50</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>30</td>
<td>60</td>
<td>70</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>60</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>50</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>60</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>60</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>60</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>60</td>
<td>80</td>
</tr>
</tbody>
</table>

In above table, the underline elements are indicates that only those elements are arranged in their proper order in corresponding iteration. The array is sorted at the end of the 5th iteration of an outer pass loop.

### 3.4 Analysis

The enhancement of the proposed algorithm is by reducing number of pass iteration, which does not necessary reduce any comparisons. Hence, the time complexity of the enhanced selection sort is exactly same as the original version, which are $O(n^2)$, $O(n^2)$ and $O(n^2)$ for the best, average and worst case respectively.

### IV. CONCLUSION

The number of pass iteration in enhanced selection sort is half of the number of pass iteration in original selection sort algorithm. The proposed enhancement algorithm has a significant improvement on reducing the number of pass iteration. The other popular sorting techniques such as Bubble sort, Insertion sort, Quick sort, Heap sort can be enhanced by using the proposed approach. This method and mention algorithm deserves future research.

### REFERENCES

**Books:**


**Websites:**


**Journal Papers:**


Biographical Notes

Mr. Ramesh M. Patelia has cleared National Eligibility Test (NET) in Computer Applications and is working as an Assistant Professor in B.C.A. (HOD) and M.Sc. (IT), Anand Mercantile College of Science, Management and Computer Technology, Anand, Gujarat, India.

Dr. Shilpan D. Vyas has been awarded Doctorate Degree (Ph.D.) in Computer Science and Engineering from Sai Nath University and is working as a Lecturer in B.C.A. and M.Sc. (IT), Anand Mercantile College of Science, Management and Computer Technology, Anand, Gujarat, India.

Ms. Parina S. Vyas is working as an Assistant Professor in Electronics and Communication Department, Shri P. M. Patel College of Electronics and Communication, Anand, Gujarat, India.

Mr. Nayan Patel is working as an Assistant Professor in M.Sc. (IT) Department, B. N. Patel Institute of Paramedical and Science, Anand, Gujarat, India.