Enhanced Novel Sorting Algorithm

By R. Srinivas
Surampalem University, India

Abstract - In computer science and mathematics; we can formulate a procedure for sorting unordered array or a file. Such procedure is always governed by an algorithm; which is called as Sorting Algorithm. Sorting is generally understood to be the process of rearranging a given set of objects in a specific order. The purpose of sorting is to facilitate the later search for members of the sorted set.

GJCST-C Classification : E.5

Strictly as per the compliance and regulations of:

© 2013. R. Srinivas. This is a research/review paper, distributed under the terms of the Creative Commons Attribution-Noncommercial 3.0 Unported License http://creativecommons.org/licenses/by-nc/3.0/, permitting all non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
Enhanced Novel Sorting Algorithm

R. Srinivas

Abstract - In computer science and mathematics, we can formulate a procedure for sorting unordered array or file. Such procedure is always governed by an algorithm, which is called a Sorting Algorithm. Sorting is generally understood to be the process of rearranging a given set of objects in a specific order. The purpose of sorting is to facilitate the later search for members of the sorted set.

I. INTRODUCTION

The intelligent and most intuitive way to do this is to analyze the items in the array or list with each other and adapt them according to their relative order, moving an element forward or backward in the list depending on whether items next to it are greater or smaller. This is called a comparison sort.

Bubble sort algorithm[13], which is a simple sorting algorithm, works by repeatedly stepping through the list to be sorted, comparing each pair of adjacent items and swapping them if they are in the wrong order.

We considered three elements and move one element towards left or right and while moving this element other element moved one or two position towards opposite direction.

The main drawback of the proposed algorithm is that if smallest element is at last location then it requires n/2 iterations to move to first location. To overcome this drawback we proposed a modification to the sorting algorithm, in this for each iteration in first half the largest in the first half moved to first location of second half and in the second half, iteration start from the last element and in this iteration smallest in second half moved to the last location of first half of the elements. This procedure is repeated for n/2 times to arrange the elements in sorted order.

II. PROPOSED MODIFICATION

In each iteration in first half the largest in the first half moved to first location of second half and in the second half iteration start from the last element and in this second half iteration smallest in second half moved to the last location of first half of the elements. In successive iterations the smallest element moved to first half may be moved towards left if it is smaller than other elements in the first half same is true for the element moved to second half of the elements. This procedure is repeated to arrange the elements in sorted order.

For example consider set of elements 9 4 6 7 8 3 9 5 2

Novel sorting algorithm

Author: SSAIST, Surampalem, India.
E-mail: rayudu_srinivas@rediffmail.com

First iteration

Pass One

Element 4 compared with 9 and 6 arrange these elements in order. The order of elements after arrangement 4 6 7 8 9 3 5 2

Element 7 compared with 9 and 8 arrange the elements. The order of elements after arrangement 4 6 7 8 9 3 5 2

Element 3 compared with 9 and 9 arrange the elements. The order of elements after arrangement 4 6 7 8 9 3 5 2

Element 5 compared with 9 and 2 arrange the elements. The order of elements after arrangement 4 6 7 8 9 3 2 5 9

Pass Two

Pass Three

3 4 6 7 2 8 5 9 9
3 4 2 6 7 8 5 9 9
3 4 2 6 5 7 8 9 9

Pass Four

2 3 4 6 5 7 8 9 9
2 3 4 5 6 7 8 9 9

Using proposed modification
Consider same set of elements 9 4 6 7 8 3 9 5 2

Pass One

for first half

4 6 7 8 9 3 5 2
4 6 7 8 9 3 5 2

Second half

4 6 7 8 9 3 2 5 9
4 6 7 8 2 3 9 5 9

Pass Two

for first half

4 6 7 8 2 3 9 5 9
4 6 2 7 8 3 9 5 9

Second half

4 6 2 7 8 3 5 9 9
4 6 2 7 3 5 8 9 9

Pass three

for first half

2 4 6 7 3 5 8 9 9
2 4 3 6 7 5 8 9 9

Global Journal of Computer Science and Technology (C) Volume XIII Issue XI Version 1

© 2013 Global Journals Inc. (US)
III. Algorithm

Input: List of elements a[0..n-1] where n is number of elements.

Step 1: swap=0
Step 2: repeat step 3 to 6 for j=0 to n/2 where step size=1
Step 3: repeat step 4 for i=1 to n/2 where step size=2
Step 4: compare elements a[i-1], a[i] and a[i+1]. If they are not in order arrange them in order. Set swap=1;
Step 5: repeat step 6 for k=n-1 to n/2 where step size=-2
Step 6: compare elements a[k-1], a[k] and a[k+1]. If they are not in order arrange them in order. Set swap=1;
Step 7: If swap=0 then given elements are in order break the outer loop else set swap=0.

a) Algorithm Analysis

The time for most sorting algorithms depends on the amount of data or size of the problem.[4]

Worst Case

The outer loop repeats for n/2 times.

In first pass of outer loop, the first inner loop repeats for (n/4) times and performs 3n/4 comparison operations and second inner loop repeats for (n/4) times and perform 3n/4 comparisons. The number of assignments performed depends on the order of elements. The maximum number of assignments performed is 2n.

In the second pass (n/2) in third pass (n/2)……

Inner loop repeats n/2+n/2+………n/2 terms.

= n^2/4

Therefore the worst time complexity is O(n^2)

IV. Results

The time complexity of this algorithm in in worst case O(n^2) same as bubble sort but their actual run time differ. For better understanding the actual performance we conducted some experiments.

The run times are measured on a PC, (AMD athlon 220xd) processor and 1G.B. RAM under Microsoft XP operating system. These algorithms are compiled using the sun java platform compiler and run under the java interpreter. The run time shown is CPU execution time measured using object of Date class. The class Date available in java util package. The elements are generated using nextInt method of Random class. The same set of elements is used for algorithms.

Table 1 : For average execution times

<table>
<thead>
<tr>
<th></th>
<th>1000</th>
<th>5000</th>
<th>10000</th>
<th>20000</th>
<th>30000</th>
<th>40000</th>
<th>50000</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Sorting With Left-Right Sorting</td>
<td>1.58</td>
<td>32.2</td>
<td>118.22</td>
<td>502.52</td>
<td>1206.58</td>
<td>2114.72</td>
<td>3279.4</td>
</tr>
<tr>
<td>New Sorting Algorithm</td>
<td>2.84</td>
<td>45.74</td>
<td>167.76</td>
<td>692.56</td>
<td>1635.94</td>
<td>2918.8</td>
<td>4513.8</td>
</tr>
<tr>
<td>Bubble Sort</td>
<td>21.9</td>
<td>103.66</td>
<td>342.2</td>
<td>1345.84</td>
<td>3240.56</td>
<td>5814.6</td>
<td>9010.04</td>
</tr>
</tbody>
</table>

Graph 1 : Graph drawn for execution times
V. Conclusion

We have proposed modification to a novel sorting algorithm to sort given elements. The Proposed method uses three elements at a time to compare based on the result it arranges the elements. The proposed algorithm is easy to understand and easy to implement. We also proposed a modification of considering iterations to increase the speed of execution. The proposed novel algorithm has similarity like bubble sort that is in every phase one element moved to its correct location.

References Références Referencias
