© 2016 IJSRSET | Volume 2 | Issue 2 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

VRF: A Novel Algorithm for optimized Sorting

Sunny Sharma[1], Vinay Kumar[2], Prithvipal Singh[3], Amritpal Singh[4]

[1,3,4] Department of Computer Science, Guru Nanak Dev University, Amritsar, Punjab, India
[2] Department of Computer Engineering and Technology, Guru Nanak Dev University, Amritsar, Punjab, India

ABSTRACT

An algorithm is a well-defined way that takes some input in the form of certain values, processes them and gives certain values as output. Although there is a large variety of sorting algorithms, sorting problem has appealed a great deal of research; because effective sorting is important to enhance the use of other algorithms.A novel sorting algorithm namely 'V-Re-Fr (VRF) Sorting Algorithm' is proposed to address the limitations of the current popular sorting algorithms. The goal of this paper is to propose a new algorithm which will provide improved functionality and reduce algorithm complexities. The observations backed by literature survey indicates that proposed algorithm is much more efficient in terms of number of swaps or iterations than the other algorithms having O(n2) complexity, like insertion, selection and bubble sort algorithms.

Keywords: V-Re-Fr (VRF), Sorting, algorithms, selection sort, swaps, time complexity

I. INTRODUCTION

Algorithms have a crucial role in resolving the computational difficulties. It is a tool to solve the computational problems [1]. Here we discuss about the various sorting algorithms. In the event of sorting, it will be required to organize an arrangement of numbers under a provided order. The formal meaning of the sorting issue is as per the following:

Input: A sequence having n numbers in some random order (b1, b2, b3, …., bn)
Output: A permutation (b'1, b'2, b'3, … , b'n) of the input sequence such that b'1 ≤ b'2 ≤ b'3 ≤ ….. b'n.

For instance, if the given input of numbers is (43, 41, 92, 32, 17, 75), then the output sequence returned by a sorting algorithm will be (17, 32, 41, 43, 75, 92). In practical, some records in the data which are to be sorted according to their keys. An n records sequence (R1, R2,..., Rn), whose corresponding sequence of keywords is (KEY1, KEY2,...,KEYn) is required to be sorted to identify a permutation Per1, Per2,..., Pern of the current subscript sequence 1,2,...,n, so that the appropriate keywords meet the decreasing or increasing relationship, that is (KPer1≤ KPer2 ≤... ≤ KPern) in order to get an ordered record sequence by their keywords. Such process is known as sorting [3].

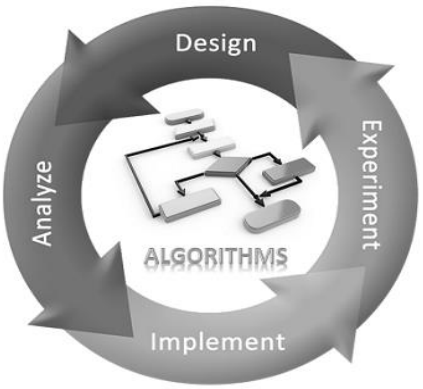

Figure 1: Flow of Algorithms

Large numbers of sorting algorithms are available at our disposal. Out of these available algorithms, which one is considered as the best one for a particular application further has its dependency on various other factors which include:

- The span of the sequence to be sorted.
- The degree up to which the given info grouping is now sorted.
- The probable restrictions on the data values.

- The framework engineering on which the sorting operation will be performed.
- The kind of capacity gadgets to be utilized: primary memory or disks [2].

The investigation of algorithm characterizes that the estimation of resources required for an algorithm to tackle an issue. Running time and space required are of primary concern for the aim that algorithms that needed a short or long time to solve a problem [14]. Finding an appropriate algorithm for an exact problem there is need to analyse numerous potential algorithms. Figure 1 represents one of the ways to recognize the finest suitable algorithm for a mentioned problem is to deploy both algorithms and find out their running time and space needed for executing a program.

In the event that the observed running time coordinates that anticipated running time of the examination furthermore beat the other calculation that would be the best appropriate calculation for the given issue. Different components influence the running time of a program in which the measure of information is the essential concern. Also, the greater part of the fundamental algorithm performs extremely well in little exhibits and sets aside more time for greater size.

The association of this proposal paper is as follows. In Section 2 (**Literature Work**), the work associated with sorting algorithms was studied. In Section 3 (**Existing Algorithms**), some existing sorting algorithms were explored. In Section 4 (**Proposed Algorithms**), a sorting algorithm is proposed and in Section 5 (**Conclusion**) there are some concluding explanations.

II. METHODS AND MATERIAL

1. Related Work

Khalid Suleiman Al-Kharabsheh [5][13] et al suggested a GCS (Grouping Comparison Sorting) algorithm which makes further comparison with others conventional sorting algorithms like Bubble sort, Merge sort, Insertion sort, Quick sort and Selection sort, to show how these algorithms reduce time of execution.

Saleh Abdel-hafeezl [6] et al proposed a hardware based comparison-free sorting algorithm which controls Hamming memory (SRAM) for storing data elements in a serial shift buffer and using (ANDING) operation of matrix multiplication between memory and the buffer produces sorted N elements in 2N cycles of clock.

Ashok Kumar Karunanithi [7] presented a review on sorting algorithms and makes comparison on three critical aspects related to efficiency such as memory, number of swaps and running time used for sorting algorithms. It takes into consideration various performance behavior like linear class or non-comparison class (Radix, Bucket ,Counting Sort) and O(n log n) class (Quick Sort) and O(n^2) class (Selection, Insertion).

Susumu Horiguchi [8] et al proposed a table-lookup sorting approach namely; Noisy sort for memory intensive calculations. For special data classes, it develops high parallel method for approximation in sorting by using associative memory.

V.P.Kulalvaimozhi [9] et al deals to analyze various sorting algorithms to calculate their performance analysis and for sorting list of data elements, it analyze problem type like large, small numbers and then compare all algorithms according to their performance.

2. Existing Sorting Algorithms

A. Bubble Sort

Bubble Sort is a simple sorting algorithm that repetitively steps through the list to be sorted matches each pair of adjacent items and swaps them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted.

'A' as an array with N elements in the bubble sort algorithm is as follows [4][10]:

Algorithm: BUBBLE (A, N)

```
Repeat Steps II and III for R=1 to N-1
Set C = 1
Repeat while C<= N-R
    If A[C] >A[C+1], then
    Swap A[C] and A[C+1]
    Set C = C+1
Exit.
```

B. Insertion Sort

Insertion sort repeats, consuming one input element each repetition, and growing a sorted output list. Every cycle, insertion sort expels one element from the input information, finds the location inside the sorted list, and add it there. . It repeats until no input elements remain.
'A' as an array with N elements in the insertion sort algorithm is as follows [4][11]:

Algorithm: INSERTION (A, N)

Set A[0] = -∞
Repeat Steps III to V for R = 2 to N
Set TMP = A[R] and C = R-1
Repeat while TMP <A[C]
Set A[C+1] = A[C]
Set C =C – 1
Set A[C+1] = TMP
Exit.

C. Selection Sort

Selection sort is an algorithm used for sorting, also called as an in-place comparison sort. Time complexity of this algorithm is $O(n^2)$. It starts by finding the smallest or largest element and swapping that element with the leftmost unsorted element and placed it in ordered list. 'A' as an array with N elements in the selection sort algorithm is as follows [4][12]:

Algorithm: SELECTION SORT (A, N)

Repeat Steps II & III for R=1 to N-1
Set MIN = A[R] and POS = R
Repeat for C= R+1 to N
If MIN >A[C] then
MIN= A [C]
POS = A [C]
POS = C
Set TMP = A[R]
A [R] = A[POS]
A[POS] = TMP
Exit

III. RESULTS AND DISCUSSION

Proposed Sorting Algorithm

V-RE-FR (VRF) Sort

There have been several authors who had made regular efforts for improving the effectiveness of the sorting method. There is a novel algorithm named as friends sort algorithm and OSSA[2] which are based on the selection sort. The proposed algorithm is based on bubble sort as its method in the second step of the operation is somewhat, similar to the bubble sort.

Figure 2, the proposed algorithm works in two steps:

1) In first step, the first and the last element of the array is compared. If the first element is larger than the last element, then exchange of the elements is required. The position of the element from front end and element from the rear end of the array are stored in variables which are increased (front end) and decreased (rear end) as the algorithm progresses.

2) In the second step, two adjacent elements from the front and rear end of the array are taken and are compared. Swapping of elements is done if required according to the order 4 variables are taken which stores the position of two front elements and two rear elements to be sorted.

Step No.	Elements						Pointer
1.	**97**	43	58	84	23	**76**	x=1, y=6
2.	76	**43**	58	84	**23**	97	x=2, y=5
3.	76	23	**58**	**84**	43	97	x=3, y=4
4.	**76**	**23**	58	84	**43**	**97**	p=1, q=2 x=6, y=5
5.	23	**76**	**58**	**84**	**43**	97	p=2, q=3 x=5, y=4
6.	23	58	**76**	**43**	84	97	p=3, q=4 x=4, y=3
7.	23	**58**	**43**	**76**	**84**	97	p=4, q=5 x=3, y=2
8.	**23**	**43**	58	76	**84**	**97**	p=5, q=6 x=2, y=1
9.	23	**43**	**58**	**76**	**84**	97	p=2, q=3 x=5, y=4
10.	23	43	**58**	**76**	84	97	p=3, q=4 x=4, y=3
11.	23	**43**	**58**	**76**	**84**	97	p=4, q=5 x=3, y=2
Sorted:	23	43	58	76	84	97	

Figure 2: Working Example of VRF Sort

Algorithm: VRF(A, N)
R, C, N, TMP, FLAG←1, R←1, C←N
while(R<C)

```
{
        if(A[R]>A[C])
        {
            TMP←A[R]
            A[R]←A[C]
            A[C]←TMP
        }
        R←R+1
        C←C-1
}
for (R←1 to N/2 && FLAG)
{
   FLAG←0
   for (C←R to N-R)
  {
        if(A[C]>A[C+1])
        {
            TMP←A[C]
            A[C]←A[C+1]
            A[C+1]←TMP
            FLAG←1
        }
        if(A[N-C]>A[N-C+1])
        {
            TMP←A[N-C]
        A[N-C]←A[N-C+1]
        A[N-C+1]←TMP
            FLAG←1
        }
    }
}
```

IV. CONCLUSION

An algorithm is a group of instruction which takes input from user, executes them and gives output to user. Sorting is an essential operation in IT field that arrange elements either in descending or ascending order. In the proposal , a novel sorting algorithm is presented namely VRF sorting algorithm and it was analytically compared with others sorting algorithms like selection, insertion and bubble sort which have $O(n^2)$ complexity and indicate better and somewhere similar complexity as the other sorting algorithms. As per the literature study it can be concluded that proposed algorithm will show improvements in terms of number of swaps or iterations as compared to other sorting algorithm and it will act as a catalyst to develop and analyse other problem specific algorithms.

In future this algorithm can be implemented and tested for its potency in a variety of application domains. Further improvements can be made keeping in view the application specific nature of algorithms.

V. REFERENCES

[1] Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, "Fundamentals Of Computer Algorithms", Second Ed. USA, 2008.

[2] Sultanullah Jadoon, Salman Faiz Solehria, Prof. Dr. Salim ur Rehman, and Prof. Hamid Jan, "Design and Analysis of Optimized Selection Sort Algorithm", International Journal of Electric & Computer Sciences (IJECS-IJENS), Vol. 11, No. 01,February 2011.

[3] Charles E. Leiserson, Thomas H. Cormen, Ronald L. Rivest, Clifford Stein, "Introduction To Algorithms", 3rd Ed. MIT Press, p.5-7, 147-150, 2009.

[4] Seymour Lpischutz, G A Vijayalakshmi Pai, "Data Structures", 3rd Ed., Tata McGraw-Hill Publishing Company Limited, p.4.11, 9.6, 9.8, 2006.

[5] Khalid Suleiman Al-Kharabsheh, Ibrahim Mahmoud AlTurani, Abdallah Mahmoud Ibrahim AlTurani, and Nabeel Imhammed Zanoon, "Review on Sorting Algorithms A Comparative Study", International Journal of Computer Science and Security (IJCSS), Vol. 7, No. 3, 2013.

[6] Saleh Abdel-hafeez, and Ann Gordon-Ross, "A Comparison-Free Sorting Algorithm", IEEE International SoC Design Conference (ISOCC), pp. 214-215, 2014.

[7] Ashok Kumar Karunanithi, "A Survey, Discussion and Comparison of Sorting Algorithms", Department of Computing Science, Umea University, June 2014.

[8] Susumu Horiguchi, and Willard L. Miranker, "Noisy Sort, A Memory-Intensive Sorting Algorithm", Elsevier Science Publishing Co., Inc, pp. 641-658, 1989.

[9] V.P.Kulalvaimozhi, M.Muthulakshmi, R.Mariselvi, G.Santhana Devi, C.Rajalakshmi, and C. Durai, " Performance Analysis Of Sorting Algorithm", International Journal of Computer Science and Mobile Computing (IJCSMC), Vol. 4, No. 1, pg.291 – 306, January 2015.

[10] Anthony LaMarca, and Richard E. Ladner, “The Influence of Caches on the Performance of Sorting”, Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms,. pp. 370–379, January1997.
[11] C.Cook, and D.Kim, “Best sorting algorithm for nearly sorted lists", Commun. ACM, pp.620-624.
[12] Deepak Garg, “Selection O. Best Sorting Algorithm”, International Journal of Intelligent Information Processing, pp.363-368.
[13] I. trini, k. kharabsheh, and A. trini, "Grouping Comparison Sort", Australian Journal of Basic and Applied Sciences, pp. 221-228, May 2016.
[14] Kronrod, M. A., "Optimal ordering algorithm without operational field",Soviet Mathematics – Doklady, pp. 744, 1969.