International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 5, July - 2012

INDEX SELECTION SORTING ALGORITHEM

Vishweshwarayya C Hallur[1]
Angadi Institute of Technology & Management, Belgaum

Ramesh K[2]
Assistant Professor
Department Computer Science Karnatak University, Dharwad-580003

Basavaraj A Goudannavar[3]
Department of P.G. Studies and Research in Computer Science
Karnatak University, Dharwad-580003

Abstract:

One of the most frequent operations performed on database is searching. To perform this operation we have different kinds of searching techniques. These all searching algorithms work only on data, which are previously sorted. An efficient algorithm is required to make searching algorithm fast and efficient. This paper presents a new sorting algorithm named as "Index Selection Sorting Algorithm (ISSA)". This ISSA is designed to perform sorting quickly and easily and also efficient as existing algorithms in sorting.

Key Words: Algorithm, Sorting, ISSA, Worst Case, Average Case, and Best Case.

I. Introduction

Using a computer to solve problem involves directing it on what step it must follow to get the problem to be solved. The step it must follow is called an algorithm. The common sorting algorithm can be divided into two classes by the difficulty of their algorithms. There is a direct correlation between the complexity and effectiveness of an algorithm [1].

The complexity of an algorithm generally written in the form of Big O(n) notation, where O represents the complexity of the algorithm and value n represents the size of the list. The two groups of sorting algorithm are O(n2), which includes bubble sort, insertion sort, selection sort, and shell sort. And O(n log(n)) which includes the heap sort, merge sort and quick sort[2].

Since the drastic advancement in computing, most of the research is done to solve the sorting problem, perhaps due to the complexity of solving it efficiently dispite its simple and familiar statement. It is always very difficult to say that one sorting technique is better than another. Performance of the various sorting algorithms depends upon the data being sorted. Sorting is used in most of the applications and there have been plenty of performance analyses [3][4].

There has been growing interest on enhancements to sorting algorithms that do not have an effect on their asymptotic complexity but rather tend to improve performance by enhancing data locality [2][3][5].

II. Proposed System

In ISSA technique the first number will be compared with all the elements in the list, at the end of each pass selection of proper index of new list is done and then element is copied it to that position in the new list. And this step will be repeated for n number of times.

The best case time complexity of ISSA is Omega Ω(n2), the average case of the ISSA is theta Θ(n2) and worst case of ISSA is Big O(n2).

Diagrammatic representation of ISSA:

Unsorted List with size a[5]

345	565	49	23	232
a[0]	a[1]	a[2]	a[3]	a[4]

New List i.e. b[5]

a[0]	a[1]	a[2]	a[3]	a[4]

After the first pass, the index of 345 is calculated and then 345 is copied in to that position in the new list. Therefore index of 345 is 3. Hence,

			345	
a[0]	a[1]	a[2]	a[3]	a[4]

After the second pass, the index of the 565 is calculated and then 565 is copied into that position in the new list. Therefore index of 565 is 4. Hence,

			345	565
a[0]	a[1]	a[2]	a[3]	a[4]

After the third pass, the index of 49 is calculated and then 49 is copied into that position in new list. Therefore index of 49 is 1. Hence,

	49		345	565
a[0]	a[1]	a[2]	a[3]	a[4]

After the fourth pass, the index of 23 is calculated and then 23 is copied into that position in the new list. Therefore index of 23 is 0. Hence,

23	49		345	565
a[0]	a[1]	a[2]	a[3]	a[4]

After the last pass, the index of 232 is calculated and then 232 is copied into that position in the new list. Therefore index of 232 is 2. Hence,

23	49		345	565
a[0]	a[1]	a[2]	a[3]	a[4]

III. Algorithm

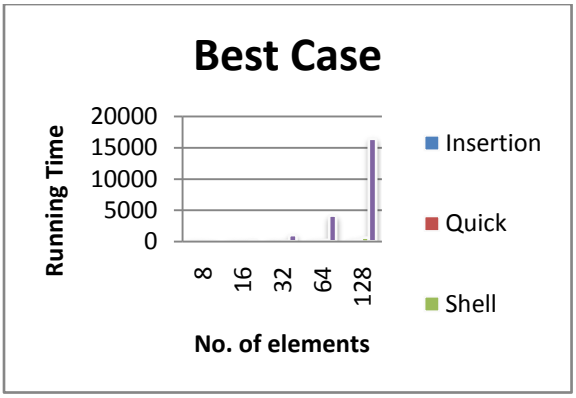
```
Algorithm ISSA(a,b,n)
for i ← 0 to n-1
        k ← 0
        item ← a[i]
        for j ← 0 to n-1
                if (item > a[j]) then
                        increment k
        b[k] ← item
```

IV. Comparisons of ISSA with other sorting technique

Below are the tables representing the calculated running time for n values and their graphs in various cases.

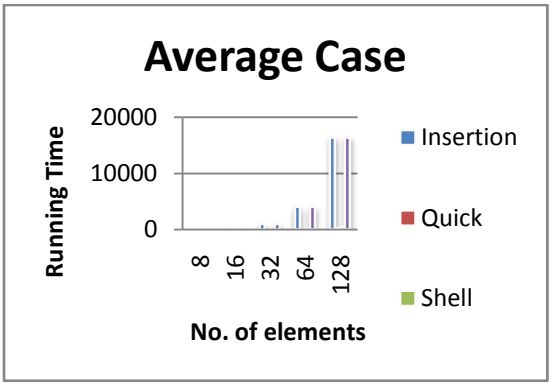
a) Best Case:

Best Case				
	Insertion	Quick	Shell	Index Selection
8	8	7.22	6.52	64
16	16	19.26	23.19	256
32	32	48.16	72.49	1024
64	64	115.59	208.78	4096
128	128	269.72	568.36	16384



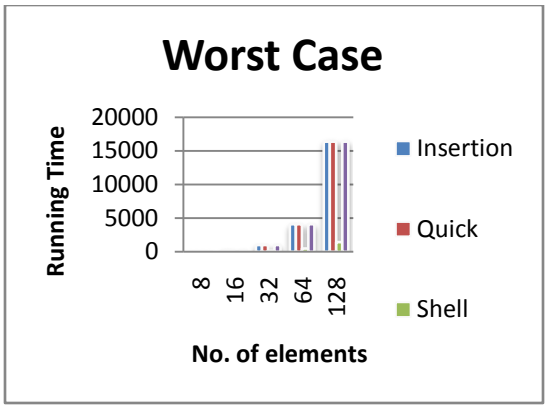
b) Average Case:

Avarage Case				
	Insertion	Quick	Shell	Index Selection
8	64	7.22	13.45	64
16	256	19.26	32	256
32	1024	48.16	76.10	1024
64	4096	115.59	181.01	4096
128	16384	269.72	430.53	16384



c) Worst Case:

Worst Case				
	Insertion	**Quick**	**Shell**	**Index Selection**
8	64	64	22.62	64
16	256	256	64	256
32	1024	1024	181.01	1024
64	4096	4096	512	4096
128	16384	16384	1448.1	16384



From the above graphs one can easily observe that with all the cases it takes same time and in worst case it takes same time like other sorting algorithms except quick sort technique.

V. Conclusion

Logic of ISSA is based on the logic of selection sort and insertion sort. In those techniques either the smallest or largest elements are taken and then placed them in appropriate position but in this ISSA first element, second element, third element and so on from the unsorted list is taken and then it is placed in its appropriate position in new list.

VI. References

[1]. Hore, C.A.R. "Algorithm 64: Quick sort". Comm.ACM 4,7(July 1961), 321.

[2]. Soubhik Chakraborty, Mousami Bose, and Kumar Sushant, A research thesis, On way Parameters of input Distributions Need be Taken Into Account For a more precise Evalution of complexity for certain Algorithms.

[3]. D.S.Malik, C++ Programming: Program Design including Data Structures, Course Technology(Thomson Learning), 2002, www.course.com

[4]. J.L>Bentley and R.Sedgewick. "Fast Algorithms for Sorting and Searching stings", ACM-SIAM SODA, 97,360-369, 1997.

[5]. D.Jim enez-Gonz'alez, J. Navarvo, and Larriba-Pay. CC-Radix: "A catch conscious sorting based on Radix sort". In Euromicro Conference on Parallel Distributed and Network based Processing. Pages 101-108, February 2003.