ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 11, November 2013

Performance Analysis of Different Approach of Adaptive Sorting

Maulik Patel[1], Shruti Yagnik[2]

***Abstract*—with the advent of processing speed, sorting algorithms have been studied and optimized for decades. Different algorithms devised to solve sorting problem often differ in their efficiency. Various techniques have been tested against previous sorting techniques to reduce complexity and cost. Since no one sorting method is always efficient for all situations. However along with speed advantage, algorithm brings new challenges in reducing space. To address how to make basic algorithm efficient we present a Survey paper. In this survey paper, we describe different concepts of adaptive sorting used by researcher to improve performance of algorithms.**

***Index Terms*— Average case, Best case, Complexity, Worst case.**

I. INTRODUCTION

Algorithm for Sorting is process of organizing data items into a particular ascending and descending order. Data items may be in form of numeric or character. Complexity of sorting algorithm is correlated with swapping, comparison and storage space. Many researchers have attempted to optimize sorting algorithm with respect to time complexity and produced new approach by comparing existing methods. Adaptive sorting is usually performed by reforming concepts of available sorting technique. To make sorting algorithm adaptive several factors are analyzed, such as data structure, asymptotic notation, comparison method, best, average and worst case analysis. The ultimate intention of researcher is to reduce complexity and cost. We observed all sorting algorithms are problem specific. Some sorting algorithms work better in large number of data items, some are used only for small no of data items, and some are used for ordering repeated data items.

The main objective is to survey all adaptive sorting techniques along with time and space complexity.

***Manuscript received Nov, 2013*.**

***Maulik Patel**, Computer Engineering Department, L.J Institute of Engineering and Technology, Gujarat University, India.*

***Shruti B Yagnik** is Assistant professor at L.J Institute of Engineering and Technology, Gujarat University, India.*

II. BASIC SORTING ALGORITHMS

There are many sorting algorithms have been developed and used in field of computer science .In this section we discuss some basic sorting algorithm .These are following:

A. Bubble sort

This is simple sorting technique among all sorting techniques. It compares adjacent items and exchanges those that are out of order. The pass through the list is repeated until no swaps are needed .This means that on each pass ,the largest item that is left unsorted bubbles to its proper place at the end .Major disadvantage is that bubble sort keeps sorting even if it is in sorted order. Average and worst case time complexity of bubble sort is o (n^2).

B. Selection sort

This is algorithm is an improvement of bubble sort. It repeatedly finds the smallest/largest item in the array and moves it to the first/last location of list. This algorithm is called selection sort because it works by selecting a minimum element in each step of the sort. For each value of element it needs (n-1) comparisons. Average and worst case complexity is o (n^2).A bidirectional variant of selection sort called cocktail sort. Cocktail sort finds both the minimum and maximum values in every pass.

C. Insertion sort

Insertion sort is an efficient sorting technique for small no of data items. It is much less efficient on large data sets. It selects one element in every pass and figuring out where it should place. This process is repeated until all the elements in the correct sequences. Best case complexity is o (n) and average and worst-case time complexity is o (n^2).

D. Merge sort

Merge sort is well known sorting algorithm and based on divide and conquer paradigm. It divides given array into sub arrays and conquers by recursively sorting the sub arrays. Finally combines the elements by merging sorted sequences. Average and worst case time complexity is o (nlogn).

E. Quick sort

Quick sort is also use divide and conquer paradigm to split. It first divides a large list into smaller sub lists by picking an element, called pivot. After that it reorders the list to fix pivot

at proper location. This process continues recursively. Average running time is o(nlogn).it is option best practical choice for sorting.

The efficiency of a sorting algorithm depends on speed and accuracy. Selection of better algorithm for given situation is also a problem. Analysis of these algorithms on comparison basis is required to identify their performance [1].

III. EXISTING ADAPTIVE SORTING ALGORITHMS

A. Novel sort

The proposed novel sort [2] provides adaption over bubble sort .This novel sort makes pair of three elements for comparison. Bubble sort use two element for comparison but novel sort use three element for comparison during each pass. It moves one element towards left or right and other element is moved in opposite direction of other to make proper ordering of elements. By comparing three elements we can minimize the number of iterations. No of swap operation is minimized by comparing odd location elements. The time complexity of this algorithm in best case is O(n) and in worst case $O(n^2)$ same as bubble sort. The advantage phase of this sorting algorithm is it takes actual execution time smaller than time taken by bubble sort for more than 5000 elements. This proposed method is only suitable for large number of elements because for small number of elements (lower than 1000), it takes more execution time than bubble sort. Space complexity of this algorithm is same as bubble sort.

B. A New Friends sort

Friends sort [3] comparing each element with the rest of elements in sequential pair. Friends sort is based on the idea of selection sort. Elements those are on left side of middle element are compared with all next elements which are on right side of them .On the other hand those elements on right side of middle element are compared with previous elements which are on left side of them. After comparing both sides at every pass we get first and last element at proper place. This comparing process is repeated for remaining elements until all elements in correct positions. Performance of Friends Sort depicts that it is clearly efficient than Bubble sort and Cocktail sort. For the small data elements it is equally efficient with selection sort and insertion sort.

C. Qureshi sort

In [4], the author Proposed qureshi sort that uses the difference of each element from the largest value to sort the repeated sequential list. This algorithm is also remembering the repetition of that element. This algorithm starts with extracting maximum value from the list. Difference array is used to store difference of each element from maximum value with the same index as the difference. Number of repetition is stored in repeating array with the same index as difference using increment command. Then it subtracts the difference from maximum value and saves in the answer array and also checks its repetition. If repetition is found for element, it repeats it in the answer array otherwise no change is done in answer array. Finally answer array becomes sorted array. To enhance performance Qureshi Sort can combine with bubble sort making complexity of bubble sort from O (N^2) to O (N), when the difference of the maximum and minimum value is less than or equal to number of element. However, it does not work on random elements. Space complexity is increased because of difference and repetition array.

D. Rapid partition sort

The paper [5] proposes an algorithm that partitions data elements in two partitions .first partition contain all the elements which are lesser than corresponding element and second partition contains all the elements which are higher than those corresponding element. It recursively returns partition to subdivide them into two partitions and the process will continue till all elements come in one partition or in worst case to the single element. The corresponding partitions will be merged after getting the required condition. By making partition it follow Divide and conquer paradigm. This sort is efficient than Bubble Sort, Selection Sort, Insertion Sort. On an average it perform better than Quick sort and proved better than merge sort when data are partially sorted in any of the order including both an ascending and a descending order. It provide O(n) complexity when half elements are in descending order and half in ascending order.

E. A Bi-partitioned Insertion Algorithm for sorting

This proposed technique [6] partitions the data set into two equal partitions. All the elements in one partition are smaller than all the elements of the other partitions .In each passes both partitions are sorted by swapping. It suggests new methodology to maintain two or more sorted partition. It also reduces the time complexity of the algorithm by reducing the number of iterations required to half. This algorithm removes the redundant data by using select procedure. The running time complexity is O (n^2).This proposed method give idea to maintain two or more sorted partition and apply effective sorting technique that will increase performance. This algorithm performs better in worst case in comparison with bubble and insertion sort when data elements are in thousands.

F. Enhanced Insertion sort

In [7], the authors have discussed new way for boosting sorting process. This algorithm is enhancement of insertion sort algorithm. Instead of comparing all elements, it compares only required i^{th} element with $(i-1)^{th}$ element. If sub array elements are already sorted than it compare with only first element of sorted list. Hit method of this algorithm prevents to scan all elements. It makes comparison and swaps according to specified condition. It focuses on reducing worst case time complexity of basic insertion sort. Enhanced insertion sort is more efficient for large number of data list as its worst case time complexity is $O(n^{1.585})$.

G. Enhanced Bubble sort

In paper [8] author proposed an algorithm that sorts elements by finding the minimum and the maximum elements and exchanging the minimum with the first element and maximum with the last element. This sorting algorithm works by remembering past index value. After exchanging it decreases the size of the array by two elements .For that it decreases the value of last index and increases the value of first index by one. It also keeps last values of the first index and last index of array. This process continues recursively until the size of the array remains one. This algorithm decreases comparisons by two makes the complexity O(nlogn) .

IV. COMPARISON OF THE VARIOUS ADAPTIVE SORTING ALGORITHMS

Sorting Algorithm	**Best case time complexity**	**Average case time complexity**	**Worst case time complexity**	**Performance**
Bubble Sort	O(n)	$O(n^2)$	$O(n^2)$	Simple sorting algorithm.
Cocktail Sort	O(n)	$O(n^2)$	$O(n^2)$	Bidirectional bubble sort.
Novel Sort	O(n)	$O(n^2)$	$O(n^2)$	Execution time of bubble sort is higher than novel sorting method for more than 5000 element.
A New Friends Sort	O(n)	$O(n^2)$	$O(n^2)$	Execution time of friends sort is better than bubble sort for more than 25000 elements and better than cocktail sort for more than 50000 elements. Equally efficient with selection sort and insertion sort for the small no of data item.
Qureshi Sort	O(3n) when D<=N and only one element is repeated O(4n) when D=N and all elements are different	O(3n) to O(4n) when D<=N and half elements are repeated .	$O(n^2)$ D >> N*N	Better performance than bubble sort when D<=N.
	Here D is difference between maximum and minimum element and N is no of elements.			
Rapid Partition Sort	O(n) when half elements are in descending order and half in ascending order.	O(nlogn) because of recursive nature.	$O(n^2)$	Efficient than bubble sort, selection sort, insertion sort .On average better than quick sort and better than merge sort when data are partially sorted.
Bi-partition insertion sort	O(n)	$O(n^2)$	$O(n^2)$	In worst case actual running time is better than bubble and insertion sort. Suggest methodology to maintain two or more partitions
Enhanced Insertion Sort	O(n)	$O(n^{1.585})$	$O(n^{1.585})$	Efficient for bigger list. Its worst case time complexity is $O(n^{1.585})$.
Enhanced Bubble Sort	O(nlogn)	O(nlogn)	O(nlogn)	Number of swapping always remains N/2. Time complexity is remain O(nlogn).

V. CONCLUSION

In this survey paper, we provided an overview about classification of the employed adaptive techniques along with their performance. Some proposed sorting techniques are simple and easy to implement. Some techniques can not able to reduce complexity but provide new classical way for improvement. This substantial improvement is achieved by different approaches. It shows that minimizing certain swaps and comparison can enhance performance of algorithm.

REFERENCES

[1] Ms. Nidhi Chhajed, Mr. Imran Uddin, Mr. Simarjeet Singh Bhatia, "A Comparison Based Analysis of Four Different Types of Sorting Algorithms in Data Structures with Their Performances", International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X ,Volume 3, Issue 2 ,2013.
[2] R.Srinivas, A.Raga Deepthi,"Novel Sorting Algorithm", International Journal on Computer Science and Engineering (IJCSE), ISSN : 0975-3397 ,Volume 5- No. 01, 2013.
[3] Sardar Zafar Iqbal, Hina Gull, Abdul Wahab Muzaffar, **"**A New Friends Sort Algorithm", IEEE, pp. 326-329, 2009.
[4] Muhammad Anjum Qureshi,"Qureshi Sort: A new Sorting Algorithm",AERO (PVT) LTD.IEEE,2009 .
[5] Mr. Naveen Kumar Goswami, " Rapid partition sort (rps) –a new sorting algorithm ", International Conference on Information System , Computer Engineering & Applications,2011. (Article in conference proceedings)
[6] Tarun tiwari, Sweetesh singh, Rupesh srivastava and Neerav kumar, "A bi-partitioned insertion algorithm for sorting",IEEE ,pp. 139-143,2009.
[7] Tarundeep Singh Sodhi,Surmeet Kaur,Snehdeep kaur,"Enhanced insertion Sort Algorithm", International Journal of Computer Applications ISSN:0975 – 8887, Volume 64– No.21, February 2013.
[8] Jehad Alnihoud ,Rami mansi, "An Enhanced of Major Sorting Algorithms", The International Arab Journal of Information Technology, Volume 7, No. 1, January 2010.

Maulik Patel received B.E Degree in Computer Engineering from Kalol Institute of Technology and Research centre. He is pursuing M.E in Computer Engineering from L.J Institute of Engineering and Technology, Gujarat. His research area includes Data Structure and Cryptography.

Shruti B. Yagnik completed Bachelors in Information Technology from L.J Institute of Engineering and Technology from Gujarat University and Masters in Computer Engineering specialization in IT Systems and Network Security from Gujarat Technological University. She is currently working as Assistant Professor at L.J Institute of Engineering and Technology carrying out research in Cyber forensics and Network Security and Artificial Intelligence.