Analysis of Linear Time Sorting Algorithm (Radix Sort and Bucket Sort)

Atitesh Gaurav
B.E Student
Department of Information Technology
Lakshmi Narain College of Technology Bhopal, India

Abstract— Any algorithm can be analysed in terms of time and space, basically algorithms is a combination of sequence of finite steps to solve a particular problem. Sorting algorithms performs rearranging data in a certain order, either in ascending order or descending, Here we will discuss about Non-comparison based sorting algorithms, these algorithms make assumptions about the input. All elements of the input keys are required to fall within a range of constant length in order to ensure linear time complexity (ie O(n)). While comparison based sorting algorithms are opposite of it these algorithms make no assumption about the input and so they are able to handle any case, while their time complexity varies according to input. Here in this paper we will review the performance and comparison between two most popular linear time sorting algorithm, Radix and Bucket Sort.

Keywords: Algorithm, Complexity, Radix Sort, Bucket Sort, Complexity, Linear Time sorting

I. INTRODUCTION

There are various types of sorting algorithms used in practical applications as per needs, like Quick sort, Merge Sort etc. These algorithms having time complexity of O (n log n), but there exists some sorting algorithms which require certain assumptions about the input sequence to perform sort, Such sorts are Radix sort ,Bucket Sort etc. Bucket sort and RADIX sort are two most popular integer sorting algorithms

II. RADIX SORT

Radix sort is a type of linear sorting algorithm that works without comparing any elements unlike other sorting methods such as Merge sort and Quick sort. It works by rearranging the integer representations based on the processing of individual digits in such a way that the integer representations of it are eventually in desired sorted order .here Keys are mostly binary digits and sometimes it considers an alphabet as keys for strings.

A radix sort is an algorithm that can rearrange integer representations based on the processing of individual digits in such a way that the integer representations are eventually in either ascending or descending order

A. Advantages

It is Stable Sorting, straight forward efficiency not depended on type and size of data Mostly Applicable to data set with multiple fields

B. Disadvantages

Occupy more memory
Algorithm for RADIX-SORT(A, d)

1) for num ← 1 → d do
2) stable sort array A on digit num
3) end for

Radix sort is a stable sort and hence for this it preserves the relative order of element with equal keys.
There are two types of radix sort such as

1) Least significant digit (LSD)
2) Most significant digit (MSD).

C. Least significant digit (LSD)

It is a type of Radix Sort LSD radix sorts process the integer representations starting from the least significant digit and move the processing towards the most significant digit

1) Advantages

1) It's Stable.
2) LSD is efficient and fast sorting method.
3) It's highly efficient for sorting the large data sets.
4) It's having Bad worst-case performance mainly due to data fragmentation.

D. Most significant digit (MSD).

Another type of Radix Sort is MSD radix sorts, Most significant digit works the opposite way of Least Significant Digit. Such that process the integer representations starting from the most significant digit and move the processing towards the least significant digit.

1) Advantages

1) MSD having Bad worst-case performance mainly due to data fragmentation.
2) Its Stable, efficient and fast sorting method.

III. BUCKET SORT

The bucket sort is a non-comparison sorting algorithm in which elements are distributed over the buckets. Bucket sort performs sorting operations by creating a number of buckets then, Each integer is put into a bucket based on the integer value .Mostly it is done by dividing the input element by the integer range, thus it defining an integer range for each of the bucket. All buckets are then sorted as per desired way like either in ascending or descending order. After that, the buckets are concatenated into a single list, which is the result of the algorithm, ie Sorted array.
Algorithm 2 BUCKET-SORT (A)

1) n ← length[A]
2) for i ← 1 → n do
3) insert A[i] into list B[n · A[i]/M]
4) end for
5) for i ← 0 → n − 1 do
6) sort list B[i] with insertion sort
7) end for
8) concatenate lists B[0],B[1], . . ., B[n − 1] together

It is assumed that the range of each integer is between 0 and M .B [1 . . . n] is an array of buckets (for a total number of n buckets, assuming each element mapped to separate bucket) which in this is implemented by using linked lists. Each input element is inserted into a bucket B[n · A[i]/M] where n · A[i]/M is generating kind of hashed key

which is treated as index of the linked lists .They are then sorted with the insertion sort internally in list , and traversing the list will give us Resulted Sorted Output.

It can be exceptionally fast because of the way its elements are assigned to buckets, typically using an array where the index is the value .therefore it required more auxiliary memory for the buckets at the cost of running time than more comparison sorts.

A. Advantages:

1) Efficient whenever input is uniformly distributed over range

B. Disadvantages;

1) Inefficient for large amount of data
2) Requires more memory space

C. Analysis:

T (n) = [time required for inserting n elements in array A] + [iteration through auxiliary array B[0 . . n-1] * (Sorting by INSERTION_SORT)
= O (n) + (n-1) (n)
= O (n)
Time Complexity:
Average time complexity for Bucket Sort is
O (n + k).
The worst time complexity is O (n^2)
The space complexity for Bucket Sort is :
= O (n + k).

The selection of the sorting algorithm varies as per Requirement, Data type, Data size etc.

IV. CONCLUSION

After going through the above analysis, it is analysed that time complexity and space complexity are the two most important factors of sorting algorithms to choose for Solving Real World Problems.In this paper, we saw how radix sort and bucket sort implement and its performance analysis in terms of time and space. These are efficient in terms of time when we compared it with comparison based sorting algorithm as it O(n log n) time , while these Non comparison based like Radix Sort ,Bucket Sort have linear time average .Various implementations of Radix Sort and Bucket Sort have been given till now which reduce the time and space complexity and further more efficient and optimised are yet to come.

REFERENCES

[1] https://en.wikipedia.org/wiki/Radix_sort
[2] Nilsson, Stefan (1 April 2000) "The fastest sorting algorithm" Dr.Dobb's journal 311: 38-45
[3] Panu Horsmalahti, "Comparison of Bucket Sort and RADIX Sort" arXiv:1206.3511v1 [cs.DS] 15 Jun 2012.
[4] http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/Sorting/bucketSort.html

