ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

VARIATIONS OF QUICK SORT

ASHWINI KAMATH
Asst. Professor
IS&E Department
AIT Chikkamagaluru

DEEPASHRI K S
Asst. Professor
IS&E Department
AIT Chikkamagaluru

ABSTRACT

Sorting techniques provides a way to arrange the data in a particular order, ascending or descending if data contains numbers, or alphabetical order if data contains letters. This paper gives the description, implementation and analysis of the variations of quicksort algorithm. Variations of quicksort considered for the paper is based on the selection of pivot element at different positions like first element, last element, middle element and selecting element randomly.

I. INTRODUCTION

There are different sorting techniques available. Depending on the information available like size of the input data, type of the input data and time for sorting available one can choose any sorting technique. Among different sorting techniques quick sort is considered as faster as its average efficiency is in O(n log n). The variations in this algorithm are depending on the pivot element position we consider for sorting like first element, last element, middle element and selecting element randomly in the list of input data.

II. RELATED WORK

Many works have been done on quick sort. In paper titled "A Historical Perspective and Empirical Study" a study on the different variations of quick sort including initially developed quick sort has been considered by comparing based on time and number of comparisons [1]. In paper titled "Improving of Quicksort Algorithm Performance by Sequential Thread or Parallel Algorithms" the performance of the algorithm is compared with theoretical results. It also has comparison between sequential and parallel quick sort [2]. In paper titled "Performance Comparison of Sequential Quick Sort and Parallel Quick Sort Algorithms" three versions of quick sort: sequential, parallel and hyper quicksort are compared based on the time statistics [3].

III. OVERVIEW

Quick sort uses a divide and conquer technique of algorithm design. Input list is generated randomly using rand () function and quick sort program is implemented using C language. The input list is divided into two halves based on the pivot element. Selection of pivot element makes difference in the efficiency of the quick sort algorithm. The pivot element can be selected based on the positions like first element, last element, middle element or any random position. In all cases algorithm divides input into two halves and pivot element is placed in such a way that all the elements before the pivot are smaller than it and all elements after the pivot are larger than it. The algorithm continues by again applying the same method on two divided list until the input list is sorted. The recursive quick sort algorithmis as shown in figure 1.

```
Algorithm QuickSort(array[], low, high)
        if (low < high)
                pi = partition(array, low, high) // get position of pivot element
                quickSort(array, low, pi - 1) // apply sort on first half
                quickSort(array, pi + 1, high) // apply sort on second half
```

Figure 1: Quick sort algorithm

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

The partition function is different depending on the pivot element position selection. The pivot element can be first element (figure 2), or last element (figure 3), or middle element (figure 4) or it can be selected randomly (figure 5). The partition algorithm scans the input list from two sides: from first element to last element and from last element to first element. While scanning from first element it will compare each element with pivot and if the element(j^{th} element)is greater than pivot is found scan is stopped. Similarly while scanning from last element if the element (j^{th} element) which is lesser than pivot is found scan is stopped. If position i is lesser than position j i^{th} and j^{th} elements are swapped. Otherwise pivot element and j^{th} elements are swapped. After this swap, the position of pivot element is final and list is divided into two halves. The quick sort algorithm is called with the divided sub-lists. This continues until the list is sorted.

```
Algorithmpartition (array[], low, high)
        i=low; j=high
        pivot=array[low] // first element as pivot
        while(i<j)
        while(array[i]<=pivot&&i<last)
                i++;
        while(array[j]>pivot)
                j--;
        if(i<j)
                        swap(array[i], array[j])
 swap(array[low],array[j])
        return j
```

Figure 2. Selecting first element as pivot

```
Algorithmpartition (array[], low, high)
        pivot = array[high]    // last element as pivot
        i = (low - 1)
        for (j = low; j <= high- 1; j++)
              if (arr[j] <= pivot)
                    i++;    // increment index of smaller element
                    swap(arr[i], arr[j]);
        swap(arr[i + 1], arr[high]);
        return (i + 1);
```

Figure 3. Selecting last element as pivot

```
Algorithmpartition (array[], low, high)
        i=low; j=high
        pivot = array[(low + high) /2];
        // partition
        while(i <= j)
                while(array[i] < pivot)
                        i++;
                while(array[j] > pivot)
                        j--;
                if(i <= j)
                        swap(array[i],array[j])
                        i++; j--;

      // call to QuickSort: QuickSort (array,low,j); QuickSort (array,i,high);
```

Figure 4. Selecting middle element as pivot

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

```
Algorithmpartition (array[], low, high)
        pivotIndex = low + rand()%(high - low + 1); //generates a random number as a pivot
        i = low - 1;
        pivot = arr[pivotIndex];
        swap(array[pivotIndex], array[high]);
        for (j = p; j < r; j++)
                if (arr[j] < pivot)
                i++;
                swap(array[i], array[j]);

        swap(array[i+1], array[high]);
        return i + 1;
```

Figure 5. Selecting pivot randomly

IV. RESULTS

Table 1: Time taken for quick sort based on different positions for pivot element.

Time taken in seconds based on position of pivot element				
Number of elements	Pivot=first element	Pivot=last element	Pivot=middle element	Pivot=randomly selected element
200000	2.70	1.84	**0.03**	0.58
400000	2.79	7.27	**0.07**	2.26
600000	6.23	16.32	**0.11**	5.05
800000	11.05	28.99	**0.16**	8.93
1000000	17.28	45.33	**0.20**	13.90

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

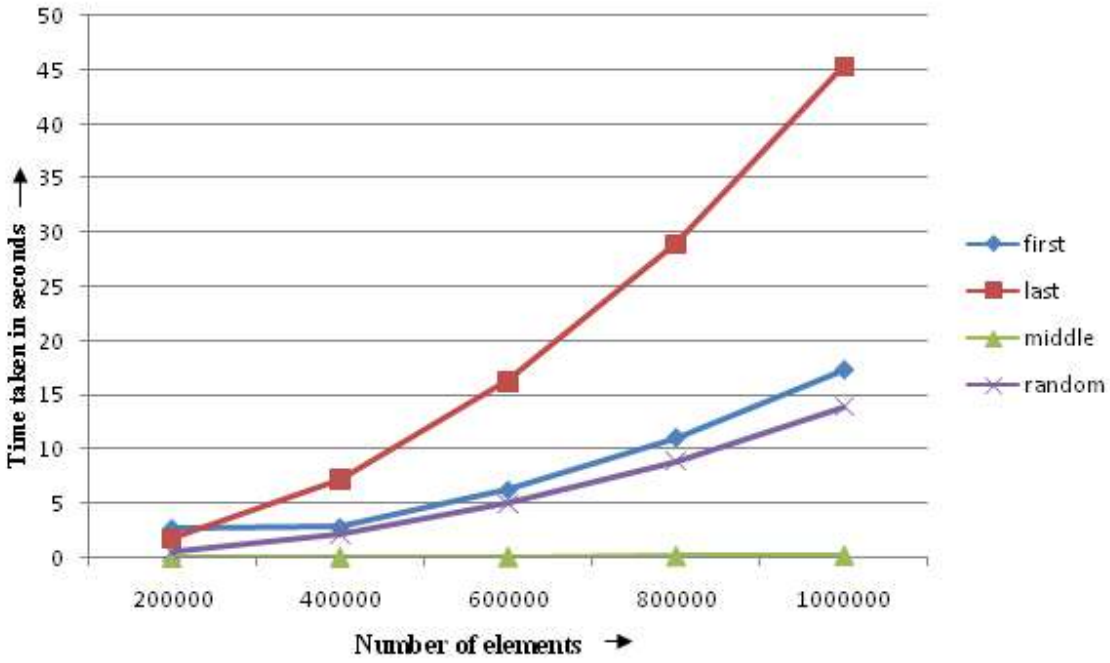

Figure 6: Graph showing difference in timings for variations in quick sort

The table 1 shows the timing statistics of all the four cases of quick sort considered. According to this table selecting middle element as pivot gives best result and selecting last element as pivot takes more time among four cases. The graph in figure 6 also shows the variations in timings for all cases of quick sort. [4] proposed a system which is an innovative congestion control algorithm named FAQ-MAST TCP (Fast Active Queue Management Stability Transmission Control Protocol) is aimed for high-speed long-latency networks. Four major difficulties in FAQ-MAST TCP are highlighted at both packet and flow levels. The architecture and characterization of equilibrium and stability properties of FAQ-MAST TCP are discussed. Experimental results are presented comparing the first Linux prototype with TCP Reno, HSTCP, and STCP in terms of throughput, fairness, stability, and responsiveness. FAQ-MAST TCP aims to rapidly stabilize high-speed long-latency networks into steady, efficient and fair operating points, in dynamic sharing environments, and the preliminary results are produced as output of our project. The Proposed architecture is explained with the help of an existing real-time example as to explain why FAQ-MAST TCP download is chosen rather than FTP download.

CONCLUSION

The quick sort algorithm is implemented in four variations based on the pivot element position. That is first element, last element, middle element and selecting random position. The list of elements for sorting are generated using random number generator rand () function. The time taken, to sort the list of randomly generated numbers are compared based on the pivot element selection. The quick sort algorithm in which middle element considered as pivot is efficient compared to the remaining three cases.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

REFERENCES

[1] Laila Khreisat ,Dept. of Computer Science, Math and Physics, Fairleigh Dickinson University- "QuickSort - A Historical Perspective and Empirical Study", IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.12, December 2007

[2] Abdulrahman Hamed Almutairi & Abdulrahman Helal Alruwaili- "Improving of Quicksort Algorithm Performance by Sequential Thread or Parallel Algorithms" , Global Journal of Computer Science and Technology Hardware & Computation Volume 12 Issue 10 Version 1.0, Online ISSN: 0975-4172

[3] Ishwari Singh Rajput, Bhawnesh Kumar & Tinku Singh - "Performance Comparison of Sequential Quick Sort and Parallel Quick Sort Algorithms", International Journal of Computer Applications (0975 – 8887) Volume 57– No.9, November 2012

[4] Christo Ananth, S.Esakki Rajavel, I.AnnaDurai, A.Mydeen@SyedAli, C.Sudalai@UtchiMahali, M.Ruban Kingston, "FAQ-MAST TCP for Secure Download", International Journal of Communication and Computer Technologies (IJCCTS), Volume 02 – No.13 Issue: 01 , Mar 2014, pp 78-85