Multipart Sequential Search Algorithm

[1]Akunuri Satyam, [2]A.Chandu Naik

[1]Assistant Professor, [2]Assistant Professor
[1] Department of CSE,
[1]SreeNidhi Institute of Science and Technology, Hyderabad, India

Abstract : ***A search algorithm plays an important role of many Applications. Some searches involve looking for an entry in a database, such as looking up your record in the IRS database. Other search algorithms travel through a virtual space, such as those hunting for the best chess moves. However programmers can choose from numerous search type methods, they can select the algorithm that best matches the size and structure of the application to provide a user-friendly experience.***

IndexTerms - Searching, Algorithms, Linear Search, Binary Search, MPSS.

I. Introduction

There are many searching algorithms used for searching elements or records from the collection of Data.The linear search is suitable for short lists, because it's simple and easy which requires minimal code to implement. The linear search algorithm is starts at the first list item to see whether you are searching for it and, if so, you are finished. If not, it searches at the next item and on through each entry in the list.
In Linear Search It is not mandatory to arrange an array in any order but as in the case of binary search. Linear search starts by sequentially scanning the elements in the array and if the element has been found. Every searching algorithm has its own merits and demerits based on time complexity as well as space complexity. In this paper we proposed a new algorithm called Multipart Sequential search algorithm.

II. RELATED WORK

Linear search is rarely used practically because other search algorithms such as the binary search algorithm and hash tables allow significantly faster searching comparison to linear search.

Algorithm LinearSearch(num, A, N)

Input: Array A, value num tobe identified, and number N of elements in A

Output: Position i, $0 \le i < N$ such that A[i] = num, or −1 if search does not belongs to A

```
1. i←0
2. while (i < N) and (A[i] != num)
3. do
4. i←i + 1 //end while
5. if i < N then return i
6. else return -1
```

Binary search is a popular algorithm for large applications with many records ordered by numerical key. The algorithm starts at the middle of the list—if your search element is greater than the middle element, the search will continue with the upper half (second portion) of list. If the search element is smaller than the middle, the search will continue with the lower half (first portion) of the database. This process repeats, cutting the list in half each time until it finds the element.
Algorithm: BinarySearch(num,A,N)
Input: Array A, value num to be identify and number N of elements in A
Output: position i,0≤i<N such that A[i]=num,or -1 if num doesnot belong to A

```
1. low ←0
2. high ← N - 1
3. while (low <= high)
4. do
      a. mid ← (low + high) / 2
5. if (A[mid] > num) then
      a. high ← mid - 1
6. else if (A[mid] < num) then
      a. low ← mid + 1
7. else
      a. return mid
8. else
9. return -1
```

This search is more complicated than the linear search but for large number of list it's much faster than a linear search.But, if the condition is unsatisfied or values unequal, the half in which the target cannot lie is eliminated and the search continues on the remaining half until it is successful. If the search ends with an empty half, the condition cannot be fulfilled and target is not found.

III. MULTIPART SEQUENTIAL SEARCH ALGORITHM (MPSS)

In Multipart Sequential Search, we initialize the array with the values (either static or dynamic declaration of values). In that a particular search element to be identified, then Mutlipart sequential search divides the array size into multiple equal parts and the search will start in each portion of the array parallel manner. If the value is found at its position from any one of the portion, then it will return 1. If the values is not found the value returned is -1.

Algorithm *Multi Part Sequential Search(A, x, N,P)*

Input: Array A, value x to be search and N total number elements in A and P is number of portions (Here we divide array into max 4 parts p1,p2,p3and p4) of the Array A

Output: 1 if num belongs to A, or -1 if num does not belongs A

```
1.  p1 ← 0, i=p1;
2.  p2 ← N/p,  j=p2;
3.  p3← p2*2, k=p3;
4.  p4←p3*3/2, l=p4;
5.  a←0
6.  while (p1<p2 or  p2<p3 or p3<p4 or p4<N)
7.  do
8.  if (A[i] ≠ x or A[j] ≠ x or A[k] ≠ x or A[l] ≠x )
9.  then i←i+1,j←j+1,k←k+1,l←l+1
        a.  else
10. then a ← 1; //end while;
        a.  if a ==1 return 1
11. else return -1
```

IV. PROOF OF CORRECTNESS OF MULTIPART SEQUENTIAL SEARCH

To prove that the proposed algorithm is correct, we need to show two things: that the algorithm terminates, and that it produces the correct output

4.1 Algorithm Multipart Sequential Search terminates after a finite number of steps

The maximum number of iterations is equal to the size of the portion. Let an Array of size 100 and if an element to be searched it will take at most 25 comparisons.

V. PERFORMANCE ANALYSIS AND COMPARISON

Both the searching algorithms (Linear Search and Multipart Sequential Search) were implemented in python 3.6 and analyzed that multipart algorithm takes lesser execution time than linear search on various range of values. In this paper we proved that Multipart Sequential Search algorithm is efficient than other two searching algorithms

Table 1. CPU Time (msec) for different lengths of input sequences

Searching Algorith M /Input	0-24999		25000-49999		50000-74999		750000-100000	
	Element present At starting position	Element present At ending position	Element Present At Starting Position	Element present at ending position	Element present at starting position	Element present At ending position	Element present At starting position	Element present at ending position
Linear Search	0.0	0.046875	0.038086	0.125000	0.062500	0.125000	0.133789	0.128906
MPSSearch	0.0	0.015625	0,0	0.015625	0.0	0.062500	0.0	0.112305

V. CONCLUSION AND FUTURE SCOPE

Searching is a process of identifying a particular element in given range of elements. In Linear Search it is not compulsory to arrange an array in any order (Ascending or Descending) as in the case of binary search orderly. Result shows that Multipart Sequential Searching Algorithm is working well for all input values and it takes lesser time if the element can be located any position. This algorithm will enhance text based matching in future to improve efficiency of the system.

REFERENCES

[1]. Arora, N., Tamta, V., and Kumar S. 2012. A Novel Sorting Algorithm and Comparison with Bubble Sort and Selection Sort. International Journal of Computer Applications. Vol 45. No 1. 31-32

[2] Seymour Lipschutz (2009) Data Structure with C, Schaum Series, Tata McGraw-Hill Education.

[3].Herbert Schildt Tata McGraw-Hill [2005], "The Complete Reference C fourth Edition".

[4] Alfred V., Aho J., Horroroft, Jeffrey D.U. (2002) Data Structures and Algorithms.

[5] Frank M.C. (2004) Data Abstraction and Problem Solving with C++. US: Pearson Education, Inc.

[6] Cormen T.H., Leiserson C.E., Rivest R.L. and Stein C. (2003) Introduction to Algorithms MIT Press, Cambridge, MA, 2nd edition.