Bidirectional Linear Search Algorithm and Its Comparative Analysis with Linear Search Algorithm

Divya Kothari¹, B.L.Pal²

M.Tech Scholar, CSE, Mewar University, Chittorgarh, India¹
Assistant Professor, CSE, Mewar University, Chittorgarh, India²

Abstract: Searching is a process of checking and finding an element from a list of elements. There are two basic searching algorithms. Linear Search and Bidirectional Linear Search. Linear search starts by scanning the elements sequentially one by one from the beginning from the given list. If search key elements found in the given list then it returns true else false. But Linear Search Algorithm takes maximum time. That is why we designed Proposed Bidirectional Linear Search Algorithm to minimize the execution time. It is based on bidirectional search. New developed Bidirectional Linear Search Algorithm is tested on MATLAB 8.0 for both 32-bit as well as 64-bit Operating System to find search key elements and it works properly.

Keywords: Searching, Algorithms, Linear Search, New Bidirectional Linear Search.

I. INTRODUCTION

Linear Search is a method of searching an element in a list in sequence. Linear Search is the basic algorithm used in data structure. The array is searched for the required element from the beginning of the array. It is used to find the particular element in an array. It is not compulsory to arrange the array in ascending or descending order [1]. In this search, compare each element of the array with the element to be searched in sequence. The search is said to be successful if the required element found otherwise it is said to be unsuccessful [2]. An operating system is a program that manages the computer hardware. It also provides the basis for the application programs and act as an interact between the computer user and the computer hardware. [3].

II. LITERATURE SURVEY

Ghani Khan et al., 2014, Compared binary search, linear search and interpolation search algorithm on the basis of time complexity and space complexity in C language [4]. Nitin Arora, 2014, Calculated CPU time of the linear search algorithm using MATLAB 8.0 [5]. Ayush Pathak, 2015, Compared and analysis of the searching algorithm on the basis of time complexity and efficiency [6]. Sapinder, 2012, Compared and analysis of the linear search algorithm and binary search algorithm on the basis of line of code, program length and time using visual basic 6.0 [7]. Kamlesh Kumar Panday et al., 2014, It is only comparison analysis of all searching algorithm like Binary Search, Linear Search and Interpolation Search.[8]
III. OBJECTIVE

The main objectives are:
- Designing Bidirectional Linear Search Algorithm.
- Implementing and testing Proposed Bidirectional Linear Search Algorithm on both 32-bit and 64-bit operating system.
- Analysing Proposed Bidirectional Linear Search Algorithm on the basis of CPU time.

IV. PROPOSED ALGORITHM

Let A be the linear array of n elements. The algorithm searches for element ‘x’. ‘p’ represents the location of the element ‘x’ in the array. The algorithm returns the values p=0 if the element ‘x’ is not present in the array. The algorithm returns the values p=1 if the element ‘x’ is present in the array.

**Input:** Array A, value x to be search and n total

**Output:** 1 if x belongs to A, or 0 if x does not belongs A.

**Step 1:** i ← 1;

**Step 2:** j ← n;

**Step 3:** p ← 0;

**Step 4:** while (i < j)

**Step 5:** if (L[i] ≠ x and L[j] ≠ x)

- then i ← i + 1; j ← j - 1; p ← 0;

**Step 6:** else

- then p ← 1; //end while;

**Step 7:** if p == 1

- return 1;

**Step 8:** else

- return 0;

V. EXPERIMENTAL RESULT & ANALYSIS

We calculated the CPU time for existing linear search algorithm and proposed Bidirectional Linear Search Algorithm based on 64-bit operating system as well as 32-bit operating system using MATLAB 8.0. New designed algorithm is tested on different different number of elements of the array or size of elements.

A. CPU time calculation for 32-bit & 64-bit Operating System for 5 elements:

- Fig 1: CPU time calculation for existing linear search algorithm for 32-bit operating system for 5 elements

- Fig 2: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 5 elements

- Fig 3: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 5 elements
B. CPU time calculation for 32-bit & 64-bit Operating System for 10 elements:

Fig 4: CPU time calculation for existing linear search algorithm for 32-bit operating system for 10 elements

Fig 5: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 10 elements

Fig 6: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 10 elements

C. CPU time calculation for 32-bit & 64-bit Operating System for 15 elements:

Fig 7: CPU time calculation for existing linear search algorithm for 32-bit operating system for 15 elements

Fig 8: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 15 elements

Fig 9: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 15 elements

D. CPU time calculation for 32-bit & 64-bit Operating System for 20 elements:

Fig 10: CPU time calculation for existing linear search algorithm for 32-bit operating system for 20 elements

Fig 11: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 20 elements
Fig 12: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 20 elements

E. CPU time calculation for 32-bit & 64-bit Operating System for 5 elements:

Fig 13: CPU time calculation for existing linear search algorithm for 32-bit operating system for 25 elements

Fig 14: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 25 elements

Fig 15: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 25 elements

F. CPU time calculation for 32-bit & 64-bit Operating System for 30 elements:

Fig 16: CPU time calculation for existing linear search algorithm for 32-bit operating system for 30 elements

Fig 17: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 30 elements

Fig 18: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 30 elements

G. CPU time calculation for 32-bit & 64-bit Operating System for 35 elements:

Fig 19: CPU time calculation for existing linear search algorithm for 32-bit operating system for 35 elements

Fig 20: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 35 elements
**H. CPU time calculation for 32-bit & 64-bit Operating System for 40 elements:**

Fig 21: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 35 elements

Fig 22: CPU time calculation for existing linear search algorithm for 32-bit operating system for 40 elements

Fig 23: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 40 elements

Fig 24: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 40 elements

**I. CPU time calculation for 32-bit & 64-bit Operating System for 45 elements:**

Fig 25: CPU time calculation for existing linear search algorithm for 32-bit operating system for 45 elements

Fig 26: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 45 elements

Fig 27: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 45 elements

**J. CPU time calculation for 32-bit & 64-bit Operating System for 50 elements**

Fig 28: CPU time calculation for existing linear search algorithm for 32-bit operating system for 50 elements
VI. PERFORMANCE ANALYSIS & COMPARISON

Both the searching algorithm (Linear search and Bidirectional Linear Search Algorithm) are implemented in MATLAB 8.0 and tested for 5 to 50 elements. CPU times for the different – different elements are shown in Table 1.1:

<table>
<thead>
<tr>
<th>Number of elements</th>
<th>Existing Linear Search CPU Time (In Second) (32-bit Operating System)</th>
<th>Proposed New Sequential Search CPU Time (In second)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>32-bit Operating System</td>
<td>64-bit operating System</td>
</tr>
<tr>
<td>5</td>
<td>0.1563</td>
<td>0.1094</td>
</tr>
<tr>
<td>10</td>
<td>0.2031</td>
<td>0.1719</td>
</tr>
<tr>
<td>15</td>
<td>0.2188</td>
<td>0.1875</td>
</tr>
<tr>
<td>20</td>
<td>0.2856</td>
<td>0.2656</td>
</tr>
<tr>
<td>25</td>
<td>0.3438</td>
<td>0.2813</td>
</tr>
<tr>
<td>30</td>
<td>0.3844</td>
<td>0.3125</td>
</tr>
<tr>
<td>35</td>
<td>0.4056</td>
<td>0.3255</td>
</tr>
<tr>
<td>40</td>
<td>0.4256</td>
<td>0.3550</td>
</tr>
<tr>
<td>45</td>
<td>0.4538</td>
<td>0.3890</td>
</tr>
<tr>
<td>50</td>
<td>0.4856</td>
<td>0.4156</td>
</tr>
</tbody>
</table>

Table 1.1: Comparison between both Existing Linear Search (32-bit) Algorithm and Proposed Bidirectional Linear Search (32-bit or 64-bit) Algorithm
Comparative CPU times for all cases in graphical formats are given below:

**Fig 31: Analysis of Existing Linear Search Algorithm (32-bit operating system) & Proposed Bidirectional Linear Search Algorithm**

**VII. CONCLUSION**

Proposed Bidirectional Linear search algorithm works on both 32-bit as well as 64 bit operating system and also it takes very less CPU time for searching search key with respect to existing sequential search algorithm.

**VIII. FUTURE SCOPE**

The proposed sequential search algorithm can be implemented in other language like C, C++, JAVA and other language. Analysing the time complexity, space complexity, efficiency, line of code, program length can be computed or compared with other searching algorithm.

**ACKNOWLEDGEMENT**

I take this opportunity to express my cordial gratitude and deep sense of indebtedness to Asst. Prof. B.L.Pal, for the valuable guidance and inspiration throughout the Work. . I express my gratitude and thanks to all the staff members of Computer Science department for their sincere cooperation in furnishing relevant information to complete this Project well in time successfully. I extend a special word to my friends, who have been a constant source of inspiration throughout my Project work.

**REFERENCES**


