Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 12, December 2015, pg.148 – 155

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

Divya Kothari[1], B.L.Pal[2]

M.Tech Scholar, CSE, Mewar University, Chittorgarh, India[1]

Assistant Professor, CSE, Mewar University, Chittorgarh, India[2]

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.

Step1: i ← 1;

Step2: j ←n;

Step3: p←0;

Step4: while(i<j)

Step5: if (L[i] ≠ x and L[j] ≠ x)

then i ← i + 1; j ← j - 1; p← 0;

Step6: else

then p ← 1; //end while;

Step7: if p ==1

return 1;

Step8: 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:

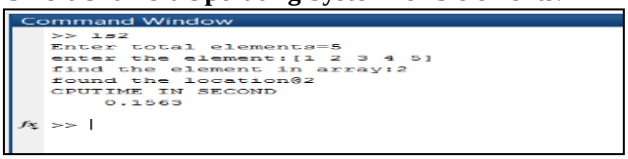

Fig 1: CPU time calculation for existing linear search algorithm for32-bit operating system for 5 elements

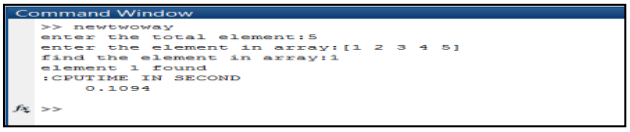

Fig 2: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 5 elements

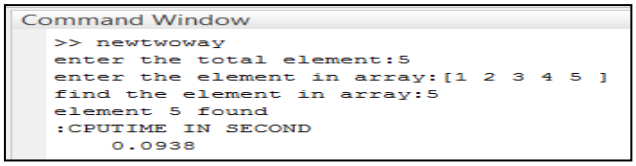

Fig 3: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 5 elements

B. CPU time calculation for 32-bit & 64-bit Operating System for 10 elements:

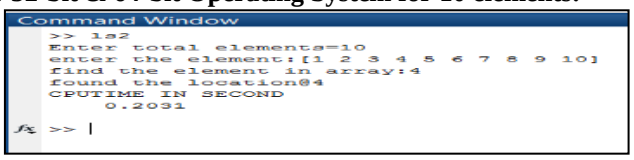

Fig 4: CPU time calculation for existing linear search algorithm for32-bit operating system for 10 elements

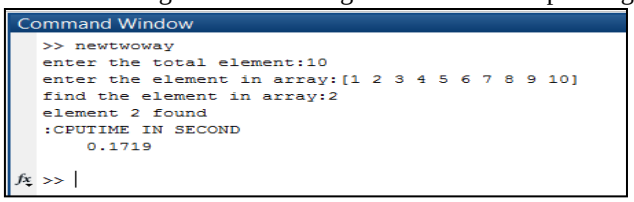

Fig 5: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 10 elements

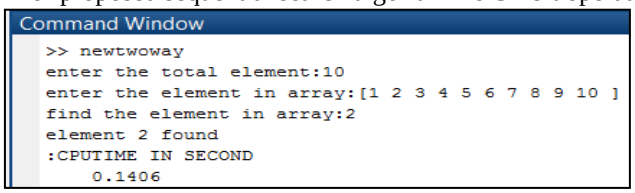

Fig 6: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 10 elements

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

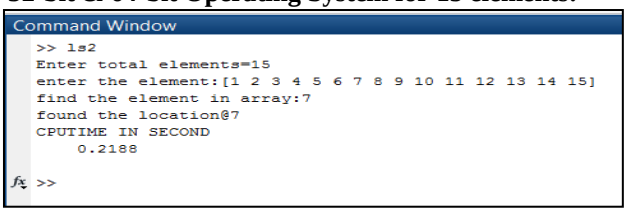

Fig 7: CPU time calculation for existing linear search algorithm for32-bit operating system for 15 elements

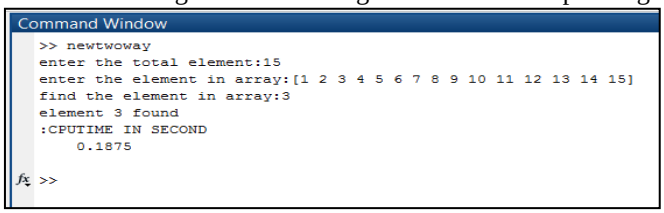

Fig 8: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 15 elements

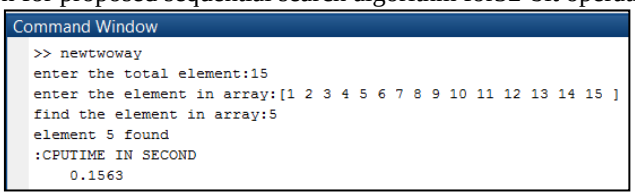

Fig 9: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 15 elements

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

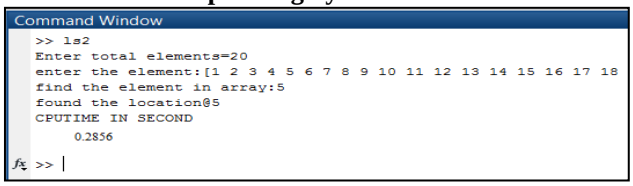

Fig 10: CPU time calculation for existing linear search algorithm for32-bit operating system for 20 elements

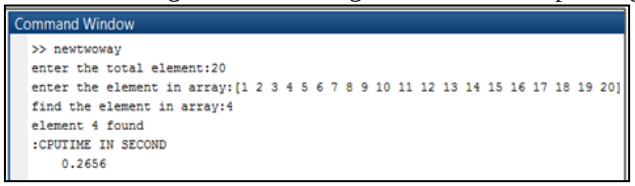

Fig 11: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 20 elements

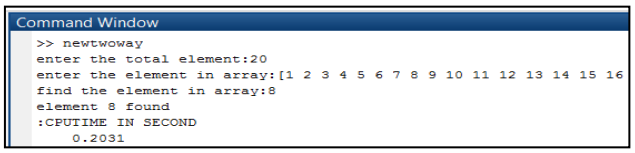

Fig 12: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 20 elements

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

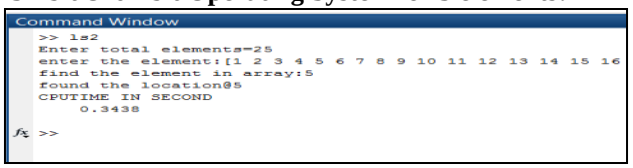

Fig 13: CPU time calculation for existing linear search algorithm for32-bit operating system for 25 elements

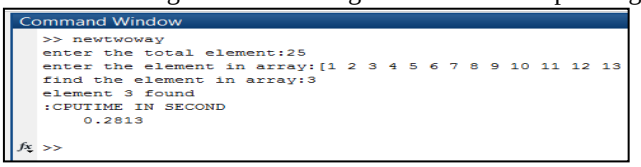

Fig 14: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 25 elements

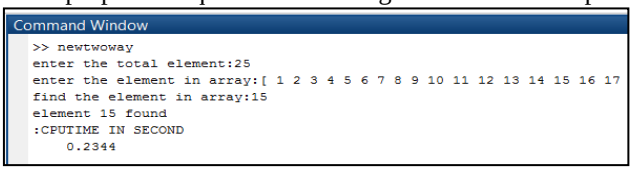

Fig 15: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 25 elements

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

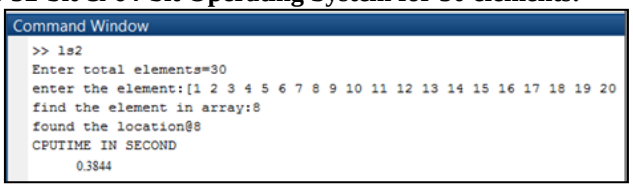

Fig 16: CPU time calculation for existing linear search algorithm for32-bit operating system for 30 elements

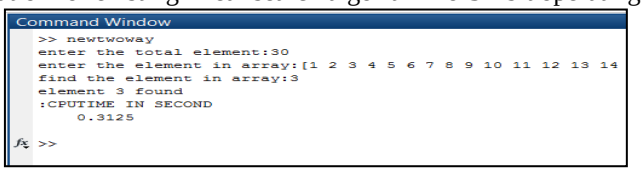

Fig 17: CPU time calculation for proposed sequential search algorithm for 32-bit operating system for 30 elements

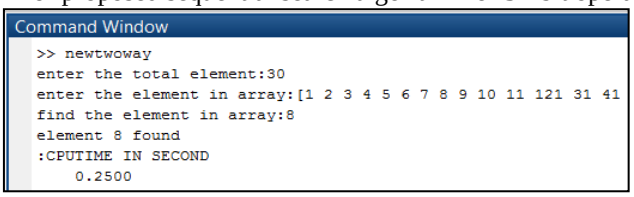

Fig 18: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 30 elements

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

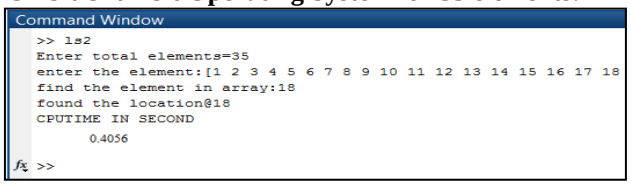

Fig 19: CPU time calculation for existing linear search algorithm for32-bit operating system for 35 elements

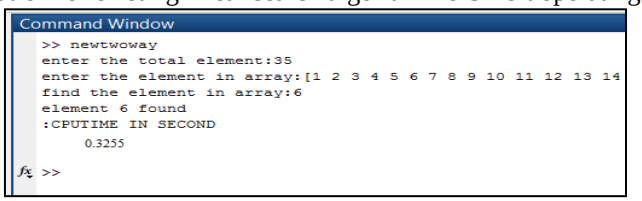

Fig 20: CPU time calculation for proposed sequential search algorithm for32-bit operating system for35 elements

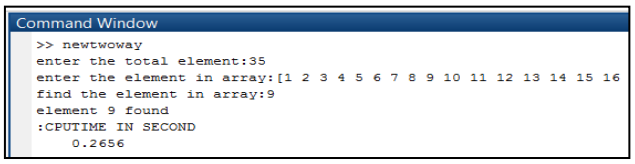

Fig 21: CPU time calculation for proposed sequential search algorithm for64-bit operating system for 35 elements

H. CPU time calculation for 32-bit & 64-bit Operating System for 40 elements:

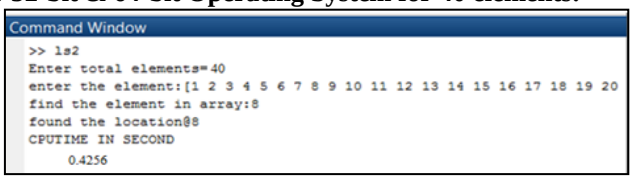

Fig 22: CPU time calculation for existing linear search algorithm for32-bit operating system for 40 elements

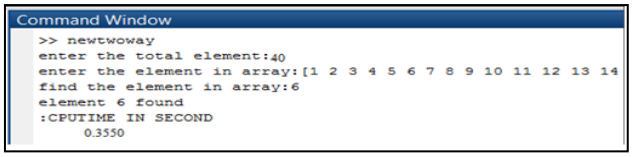

Fig 23: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 40 elements

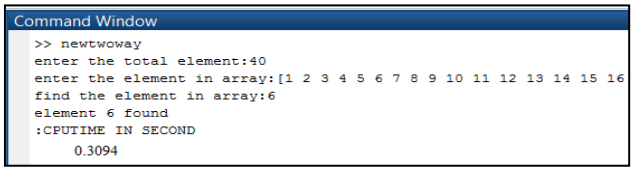

Fig 24: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 40 elements

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

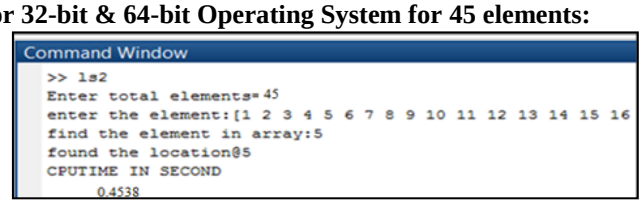

Fig 25: CPU time calculation for existing linear search algorithm for32-bit operating system for 45 elements

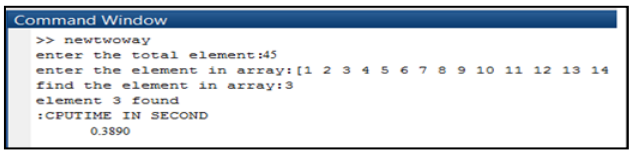

Fig 26: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 45 elements

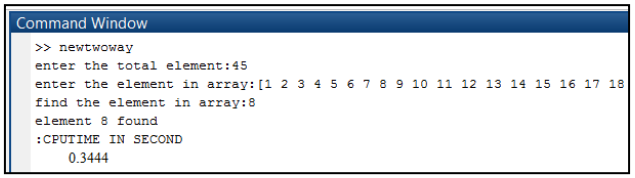

Fig 27: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 45 elements

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

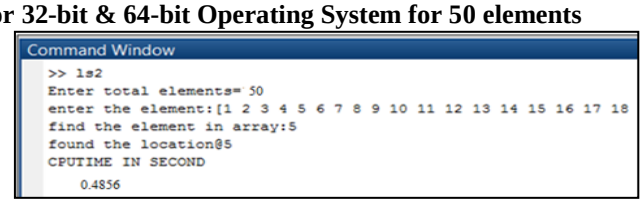

Fig 28: CPU time calculation for existing linear search algorithm for32-bit operating system for 50 elements

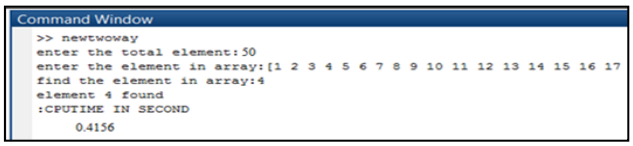

Fig 29: CPU time calculation for proposed sequential search algorithm for32-bit operating system for 50 elements

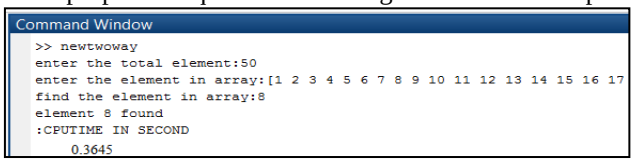

Fig 30: CPU time calculation for proposed sequential search algorithm for 64-bit operating system for 50 elements

VI. PERFORMANCE ANALYSIS & COMPARISION

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:

Number of elements	Existing Linear Search CPU Time (In Second) (32-bit Operating System)	Proposed New Sequential Search CPU Time (In second)	
		32-bit Operating System	64-bit operating System
5	0.1563	0.1094	0.0938
10	0.2031	0.1719	0.1406
15	0.2188	0.1875	0.1563
20	0.2856	0.2656	0.2031
25	0.3438	0.2813	0.2344
30	0.3844	0.3125	0.2500
35	0.4056	0.3255	0.2656
40	0.4256	0.3550	0.3094
45	0.4538	0.3890	0.3444
50	0.4856	0.4156	0.3645

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:

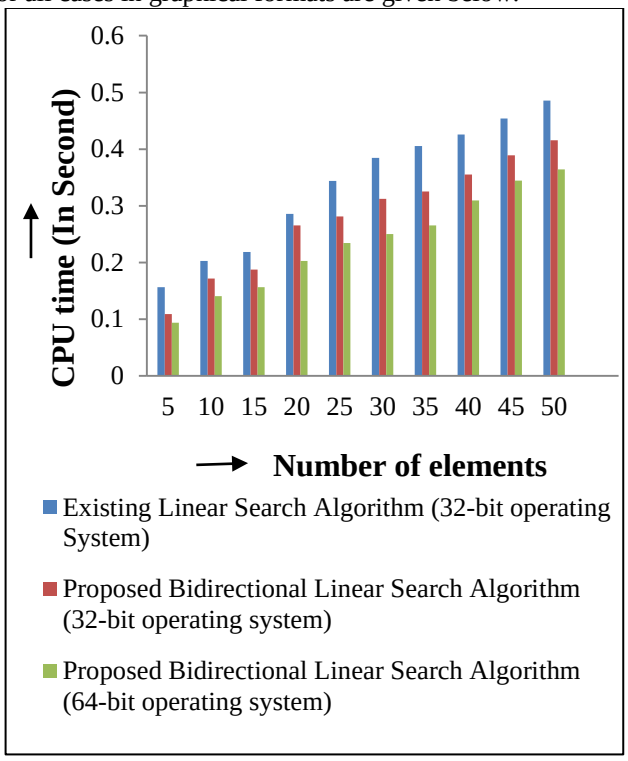

Fig 31: Analysis of Existing Linear Search Algorithm (32-bit operating system) & Proposed Bidirectional Linear Search Algorithm

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

[1] Vinu V Das et al., "Principles of Data Structure using C and C++", New Age International Publisher, First Edition, pg. 207-209, ISBN: 978-81-224-1858-3.
[2] Seymour Lipschutz et al., "Data Structure", Tata McGraw Hill Education Private Limited, Fourth Edition, pg-4.12-4.15, ISBN: 978-0-07- 060168-0.
[3] Silberschatz, A., P.B. Galvin and G. Gagne, 2004 "Operating Systems Concepts". 7th edition., John Wiley and Sons, USA., ISBN: 13: 978-0471694663, pp: 1.
[4] Ghani Khan Chowdhury, Debadrita Roy, Arnab Kundn, "A Comparative Analysis of Three Different Types of Searching Algorithms in Data Structure",International Journal of Advanced Research in Computer and Communication Engineering, Vol. 3, Issue 5, May 2014, pg. 6626-6630.
[5] Nitin Arora "International Journal of Computer Application", vol. 107, No. 21 December 2014.
[6] Ayush Pathak, "Analysis and Comparative Study of Searching Techniques", International Journal of Engineering Science & Research Technology, Vol.1, No.1, March 2014.

[7] Sapinder, Ritu, Heera Lal Singh and Arvinder Singh, "Analysis of Linear Search and Binary Search Algorithm", International Journal of Computers & Distributed Systems Vol.1, No.1, June 2012.
[8] Kamlesh Kumar Pandey and Narendra Pradhan2 , "A Comparison and Selection on Basic type of Searching Algorithm in Data Structure ",International Journal of Computer Science and Mobile Computing, Vol.3 Issue.7, July- 2014, pg. 751-758.

