International Journal of Research (IJR) Vol-1, Issue-11 December 2014 **ISSN 2348-6848**

Linked List With Binary Search

Kartik Singh(16225),Akshat Sharma(16194),Govind Rajput(16212),Devesh Kasturia(16205)
Students- Dronacharya College of Engineering

ABSTRACT*:*

Linked list is a storage programming concept that is implemented as addressing the elements in a memory using the element nodes for every void memory. This paper is trying to combine Binary search using a linked list.

INTRODUCTIONThere are three major areas of research in computer science (CS). There is hardware design with its main goal of building faster computers to store and process larger amounts of data in shorter time. Theoretical CS; part of which deals with the design of efficient algorithms (so often without any considerations about the feasibility and possible effectiveness of implementation on existing hardware). Finally, there is the most underrated area of CS experimental computer science. Experimental CS tries to bridge the gap between hardware and software design by establishing the best ways of implementing theoretically designed software on existing computers. It also tries to establish performance characteristics of the existing computers when running varieties of benchmarks.

A variety of such benchmarks can he found in the large area of searching and sortingalgorithms. In the present paper, we discuss how a simple task of comparing the performanceof different searching techniques can provide valuable insights into computer hardware,compiler design and comparison between programming languages. It is notable that due to the lack of a prior answer such a project can have the flavor of a “real research” project and thus, if only for this reason, attract the attention of students.

<u>Section-1</u>:-describes the recently introduced technique of binary search on linked lists andpresents some implementation details of this algorithm as well as an efficient sequential searchalgorithm.

<u>Section-2</u>:- presents the analysis of the proposed algorithm and shows how thisanalysis leads to sonic questions that need to be answered on experimental basis.

1. Binary Search on Linked List

Until recently, it was generally agreed that nothing can be gained from implementingbinary search algorithms on linked lists. In 1990.Khosraviyani presented a new algorithmand proved that there exists a way of taking advantage of binary search based technique to search linked lists. More recently, a statistical analysis of the new algorithm was presented.

The binary search algorithm can be summarized as follows (for more details, the discussion of assumptions and the pseudo-code see. In order to apply the binary search principle to a linked list the middle of the list must be determined. This means that half of the list has to be sequentially traversed. Assume that the list is ordered and that its current length is known. A counter controlled loop can be used to find the middle of the list. A key field comparison ¡s then used to select the half of the list to which the target key belongs. This process is then repeated on the retained half of the list.

Pointer assignments and key comparisons are the major factors influencing the complexity of the search algorithm. The binary search described above allows to reduce key field comparisons from 0(N) to 0(IogN) but it pays for this reduction by 0(N) counter controlled loop iterations. If the counter controlled loop iterations arc cheaper than key comparisons (which is usually assumed), this algorithm will be more efficient than a purely sequential search.

In the “Programming Algorithms” course one of the authors, ¡t was noted that a sequential search algorithm, (for comparison purposes) would outperform the algorithm as presented. It was only later, when a mathematical analysis of the algorithm allowed us to understand the reasons for the observed results.

2. Algorithm Analysis

It is well known that the sequential search as described above uses an average of N/2 key comparisons for a successful search. The proposed algorithm, on the other hand, will on average have O(logN) key comparisons and N-1 loop iterations. Let us denote by A the time for the key comparison, and by B — time for updating the loop counter.

The average time of a successful search is described by the following formulas:

$$\begin{aligned} Time_{Sequential} &= (N/2)A \\ Time_{Binary} &= A\log N + (N-1)B. \end{aligned}$$

We would like to establish the conditions under which the binary search is more efficient than the sequential search. This leads to solving the following inequality

$$(N/2)A > A\log N + (N-1)B,$$

Which can he reformulated as

$$(N/2 - \log N)A > (N-1)B.$$

Let us now assume that A = B and try to estimate the value of c (c determines the point from which the binary search is more efficient than the sequential search). Simple calculation show that

$$c > \frac{N-1}{N/2 - \log N} \cdot$$

Assuming N-∞, we can see that in the limit c > 2. This result, has a very interesting implication. For the binary search to be more efficient than the sequential search, the cost of key comparison must be at least twice the cost of updating loop counter. It likely that it is because c did not satisfy this condition that the sequential search proved superior to the binary search in the research. This analysis leaves open the question about the actual values of c, which can lead to research in different directions.

3. CONCLUSIONS

We believe that a study of this kind binary search performance can and should be used in teaching Computer science as it expose the student to the realities of programming. Since there are no predetermined answers students may find such a project interacting and challenging. As it is rather simple to program students can concentrate on analyzing the results which is what they often do not but should focus their attention on. Since no particular computer environment is required a project of this type can he completed in almost all schools. Depending on the course and the available computers, such projects may be focused on, for instance, algorithm comparison, hardware comparison, cross language comparison, compiler-related comparisons, etc. Even in the smallest schools it is enough to have a PC and two programming languages to complete the simplest version of a similar project. Ii can tie connected with the introduction of basic statistical techniques (and introduction to Statistical packages) used in the analysis of results. We hope that this paper will develop groups of similar problems that will allow students get exposed to Variety of important issues in the areas of experimental computer science.

REFERENCES

1. Khosraviyani, F., Using Binary Search On A Linki List," S1GCSE Bulletin, Vol. 22, No. 3. Sept..1990, pp. 7-10

2. Khosraiyani, F., Moadab. M,H., Hale, D.F., Time Distribution Analysis for Binary Search of a Linked List," SIGCSE Bulletin, Vol. 23, No. 4, Dec., 1991, pp.7-12

3. Sedgewick, R., Algorithms , Second Edition, Addison Wesley, 1988