



A Comparison and Selection on Basic Type of Searching Algorithm in Data Structure

Sapna Mittal

Assistant Professor, Computer Science,
Post Graduate Department of Computer Science and Applications,
R.S.D College, Ferozepur, Punjab, India

Abstract: A lot of problems in different practical fields of Computer Science, Database Management System, Networks, Data Mining and Artificial intelligence. Searching is common fundamental operation and solve to searching problem in a different formats of these field. These research papers are presents the basic type of searching algorithms of data structure like Linear search, binary search, and hash search. We have tried to cover some technical aspects to this searching algorithm. This research is provides a detailed study of searching algorithms working process, select on Searching Algorithm according To Problem and compares them on the basis of different parameters like total number of comparison, type of data Structure, time complexity and space complexity.

Keywords: Sequential Search, Binary Search, Hashing, Hash function, Comparisons, Time Complexity.

I. INTRODUCTION

Sorting and Searching are two fundamental operations in a computer science. Sorting means arranging on data in given order such that increment or decrement. Searching means find out location or find out an element of a given item in a collection of item. Many data structures are used to store information but Arrays, linked lists and tree are basic data structures used to sorting and searching operation. Searching element is any type of a numerical data, alphabet, String, character data. A number of searching algorithms have been developed like that sequential search, binary search, tree search and hashing etc. Every searching algorithm depends on specific problem, property of data and algorithm complexity. This research paper gives brief introduction about searching algorithm, we define to which type of searching algorithm used to which type of Problem and we compare to different type of searching algorithm in a different important parameter like time complexity, space complexity, associated key, no of comparisons etc. A group of element is originated to a file or table. Each element is a called to record. Search technique is applying to a file or table and find to a specific record with location. Each record is associated to a key and this key is separated to different record. If key are store on start of a record so this type of key is called to internal key. In other case key are store on separate table including pointer to the record so this type of key is called external key. Every file or tables have a two set of key. First set are define a uniquely data this key is a called primary key and this set have an internal and external key. Second set are define none uniquely data this key is a called secondary key. We search to an element with location we need to first set after that need to second set.

The searching technique can be divided into two categories:

1. Internal Searches: if file or tables are kept in main memory this type of searching is called internal search.
2. External Searches if file or table is kept in auxiliary memory (hard disk, Floppy, tape etc) this type of searching is called External Search.

II. WORKING PROCEDURE OF SEARCHING ALGORITHMS

1. Sequential Search Simplest form of searching technique is a sequence search or linear search. This search is applicable to a small size of array or linked list data structure. Find out on searching element in sequential manner on unordered list or ordered list. In this method searching process are started at begins to end, scans the elements one by one of the list from left to right until the searching record/element is found. If we taken on ordered list or table then time it given to fast searching and good efficient as a comparison on unordered list.

Algorithm:

Here L is a location variable, I is a variable which value is a element position and A is Array/List[0-N]. SE is search element.

LINEAR_SEARCH (A, I, SE)

1. Initialize L=0
2. For I=0 to length[A] //scan on element start to end
3. If (SE == A[I]) //check on searching element
4. L=L+1 // increment on location variable
5. Return A[I] //print on finding element
6. END If
7. END For LOOP
8. If L=0 then return "not find out element"
9. Exit

Example:

Binary Search: - another simplest form of searching technique and best efficient method is a binary search. It can be used if the table is stored as an array and mid size of array. If we use to linked list and insert/delete on data in linked list during searching time then problem are generated.

This algorithm is also based on Divide-and-Conquer algorithm. In sorted list is divided into two parts.

Conquer: We first compare to search/input element with the mid element of the list. If do not match then we check search element and mid element if search element is less then to mid element then go to first part/left part otherwise go to second/right part.

Combine: we apply to divide and conquer part whenever our search element is not found. When our search element is found then list are automatic combined and return.

Algorithm: Here L is a location variable, LOW is start index and HIGHT is last index element position of Sorted Array/List A[0-N]. SE is search element

BINARY_SEARCH (A, LOW, HIGHT, MID, SE)

1. Initialize LOW=0, HIGHT=N, L=0
2. While (LOW<=HIGHT)//scan on element start to end
3. MID=(LOW+HIGHT)/2 //divide on list
4. IF (SE==A[MID]) //check on searching element
5. L=L+1 // increment on location variable
6. Return A[MID] //print on finding element
7. Else If (SE<A[MID])
8. HIGHT=MID-1 // go to Left/first part
9. Else
10. LOW=MID+1 // go to right/second part
11. END If Statement
11. ENDS While LOOP
12. If L=0 then return, "not find out element"
13. Exit

Example:

2. Hashing: -Best searching Technique and most efficient method is a hash search. It searches technique whose search times can be independent of the number of entries in a table. Hashing has a worst-case behavior that is linear for finding a target Value, but with some case, hashing can be dramatically fast in the average-case. With this approach, the key for every record determines the position of a particular entry in the table. This association is realized through the use of a hashing function. If H is a hash function and K is key, H (K) is called hash of key. So a hash function H (K) transforms a key into an address/hash table index position. if key is a non integer value then we convert to a integer value. Some hashing function is

- (i) Division Method (table size = prime):-choose a prime number M which are equal to a table size and K is a key. The hash function H is define as $H(K) = K \text{ MOD } M$ or $H(K) = (K \text{ MOD } M) + 1$. First function denotes the remainder when K is divided by M. second function is used when we want the hash table to range from 1 to M rather than from 0 to M-
- (ii) Midsquare Method (tablesize = $2n$): -The multiplication method may be used for a HashTableSize that is a power of 2. In this method a key is multiplied by itself and the

- address is obtained by selecting a number from the middle of the square. The hash function H is defined as $H(K) = L$ is obtained K
- (iii) Folding Method: -in this method key is portioned into a equal number of parts. This part are added together and ignoring the final carry, to form an address. The hash function H is defined as $H(K) = K_1 + K_2 + \dots + K_n$

III. CONCLUSIONS

This paper discusses three basic type of searching algorithms and their example. Searching is the process to finding to location of the given data elements in the data structure. Hashing are faster for large lists as a compare Binary search and Binary search are faster for mid size lists or small list as a compare Linear search. We have compared the searching algorithm on the basis of various factors like complexity, data structure, searching type, application etc. we introduced which type of problem solve through which type of searching algorithm it help to selection on best searching algorithm.

REFERENCES

- [1] Data structures using c and c++ by Yedidyah angam, Aaron M.Tenenbaum second Indian printing (Prentice Hall of India Private limited), New Delhi-110001
- [2] An Introduction to Data Structures with Application by Jean-paul Tremblay Tata McGraw Hill [3] Data Structures by Seymour Lipschutz and G A Vijayal
- [3] Akshmi Pai (Tata McGraw Hill companies), Indian adapted edition-2006, 7 west Patel nagar, New Delhi-110063