Lecture 3
Array Searching Algorithm

PREPARED BY:

AFSANA BEGUM
SENIOR LECTURER,
SOFTWARE ENGINEERING DEPARTMENT,
DAFFODIL INTERNATIONAL UNIVERSITY.

Contents

- Linear Search Algorithm and complexity
- Binary search Algorithm and complexity

Linear Search and Binary Search Algorithm

How Linear Search works:

Searching refers to the operation of finding the location LOC of ITEM in DATA, or printing some message that ITEM does not appear there.

DATA is a linear array with n elements. The most intuitive way to search for a given ITEM in DATA is to compare ITEM with each element of DATA one by one. That is first we test whether DATA[1]=ITEM, and then we test whether DATA[2]=ITEM, and so on. This method, which traverses DATA sequentially to locate ITEM, is called **linear search** or **sequential search**.

Linear Search and Binary Search Algorithm

Linear Search Algorithm : A linear array DATA with N elements and a specific ITEM of information are given. This algorithm finds the location LOC of ITEM in the array DATA or sets LOC = Null.

1. Set Variable=ITEM.
2. Set LOC:=0.
3. Repeat while DATA[LOC]! =ITEM :
   Set LOC := LOC +1.
   [End of loop]
4. If LOC = N+1, then :
   Write : ITEM is not in the array DATA.
   Else :
   Write : LOC is the location of ITEM.
5. Exit.

Linear Search Complexity

O(n)

Worst Case Analysis (Usually Done)
In the worst case analysis, we calculate upper bound on running time of an algorithm. We must know the case that causes maximum number of operations to be executed. For Linear Search, the worst case happens when the element to be searched (x in the above code) is not present in the array. When x is not present, the search() functions compares it with all the elements of arr[] one by one. Therefore, the worst case time complexity of linear search would be Θ(n).

Write down how:

Follow Class Lecture

Sample Code (the main part of the code)

```
for (c = 0; c < n; c++)
   {
      if (array[c] == search)
/* if required element found */
      {
         printf("%d is present at location %d.\n", search,
   c+1);
         break;
      }
   }
```

Linear Search and Binary Search Algorithm

How Binary Search Works:

It must be sorted array before applying binary search.

In binary search, we first compare the key with the item in the middle position of the array. If there's a match, we can return immediately. If the key is less than the middle key, then the item should lie in the lower half of the array; if it's greater then the item so it must lie in the upper half of the array. So we repeat the procedure on the lower (or upper) half of the array.

Linear Search and Binary Search Algorithm

Binary Search Algorithm :

BINARY(DATA, LB, UB, ITEM, LOC)

1. **Set BEG=LB; END=UB; and MID=INT((BEG+END)/2).**
2. **Repeat step 3 and 4 while BEG ≤ END and DATA[MID] ≠ ITEM**
3. **If ITEM < DATA[MID] then**
   Set END= MID - 1
   Else:
   Set BEG= MID+1
   [end of if structure]
4. **Set MID= INT((BEG+END)/2)**
   [End of step 2 loop]
5. **If ITEM = DATA[MID] then**
   Set LOC= MID
   Else:
   Set LOC= NULL
   [end of if structure]
6. **Exit.**

Binary Search Complexity

O(Log2N)

How??

n

n/2

n/4

n/8

.....1 or 0

n/2k

Follow Class Lecture

Sample Code (the main part of the code)

```
first = 0;
last = n - 1;       //n=total  length
middle = (first+last)/2;

while (first <= last) {
  if (array[middle] < search)
    first = middle + 1;
  else if (array[middle] == search)
    {
    printf("%d found at location %d.\n", search, middle+1);
    break;
    }
  else
    last = middle - 1;

  middle = (first + last)/2;
}
if (first > last)
  printf("Not found! %d is not present in the list.\n", search);
```

Questions?