Algorithms – Searching, Greedy Algorithms

**SEARCHING ALGORITHMS**

In computer science, a search algorithm is an algorithm that retrieves information stored within some data structure. Here we are going to look at linear search and binary search.

**Linear Search**

The name itself says that we are going to go in a linear fashion. This algorithm is also called **sequential search.**

It sequentially checks each element of the list for a target value until a match is found or until all the elements have been searched.

*It works on both sorted and unsorted data.*

**Problem Statement:** Given a list L of n elements with values or records and target value T, the following subroutine uses linear search to find the index of the target T in L.

**Basic algorithm:**

1. Set i to 0
2. If Li = T, the search terminates successfully; return i
3. Increase i by 1
4. If i < n, go to step 2. Otherwise, the search terminates unsuccessfully

**Time complexity**

1. **Approach 1**
   1. **Best case:** number of iterations = 1
   2. **Worst case:** number of iterations = n+1 (we didn’t find the element at all)
   3. **Average case :** number of iterations = best + worst / 2 = O(n+2/2) = O(n)
2. **Approach 2: Using recurrence relation**

T(n) = T(n-1) + 1 if n > 1

T(1) = 1

Then by back substitution T(n) = O(n)

**Implementation of linear search on a linked list or array is going to give the same worst case time which is O(n)**

Let’s see the next searching algorithm called Binary Search

**Binary Search**

*Binary search can be only applied when:*

1. *The data is stored in an array*
2. *The data is sorted*

*Binary search cannot be applied when:*

1. *The data is stored in a linked list and sorted*
2. *The data is unsorted*

Binary search, also known as **half-interval search** or **logarithmic search,** is a search algorithm that finds the position of a target value within a sorted array. It does not work on unsorted array.

* Binary search works on the sorted arrays, it begins by comparing the middle element of the array to the target value
* If target value matches the middle element, its position in the array is returned
* If the target value is less than the middle element, then, the search is continued on the left half of the array otherwise the search is continued on the right half of the array

**Problem Statement:** Given an array A of n elements with values or records , sorted such that , and target value T, the following subroutine uses binary search to find the index of T in A.

**Algorithm:**

1. Set L to 0 and R to n-1
2. If L > R, the search terminates as unsuccessful
3. Set m (the position of the middle element) to the floor (the largest previous integer) of (L+R)/2
4. If T = Am, the search is done; return m
5. If T > Am set L to m+1 and go to step 2
6. If T < Am set R to m-1 and go to step 2

**Time complexity:**

Recurrence relation: T(n) = T(n/2) + 1 if n > 1

T(n) = 1 if n = 1

* **T(n) = O(log2n)**