To find an element X in an array using divide and conquer algorithm

Indian Institute of Information Technology, Allahabad

Aditya Aggarwal
iit2019210@iiita.ac.in

Divy Agrawal
iit2019211@iiita.ac.in

Aman Rubey
iit2019212@iiita.ac.in

***Abstract*—In this paper we have devised an algorithm to find an element X in an array of given elements solely using the concept of divide and conquer. Also in the later part of the paper we have discussed the space and time complexity of the devised algorithm.**

***Index Terms*—Algorithm, Divide and Conquer, Searching, complexity**

I. Introduction

Divide and conquer algorithm is an efficient algorithm that decompose a given problem into small sub-problems recursively until it is relatively easier to solve those small sub-problems. Then the solution to the original problem is computed by combining the solutions to the sub-problems. Implementation of this algorithm could be seen in sorting algorithms such as merge-sort and quick-sort.

II. Algorithm Design

We have taken elements (Integer) as input from user and stored them in a vector. We will be using this vector to check whether an element X is present in this vector.

An important concept used in this algorithm is that if vector contains a singleton element only and if this element is equal to the required element X the answer would be true else it would be false.

- Here the main problem is to check whether an element is present in a vector or not.
- This main problem could be divided into two sub-problems by dividing this initial vector (parent vector) into two vectors (child vectors) of approximately same size and checking whether the required element is present in these two vectors formed(child vectors).
- We will divide these new sub-vectors formed until the new vectors comprise of singleton element only.
- As discussed earlier in this section, answer for these vectors comprising of single elements will be either true or false depending on whether this single element is equal to required element or not.
- Suppose, we know that whether the required element X is present in two vectors or not. Then the answer to the problem-vector comprising of the elements of both the vectors simply depends upon the answer for the two child vectors.
  - If element is present in either or in both of the child vectors then the element will be present in the parent vector comprising of these two child vectors.
  - If element is not present in both of the child vectors then the element will not be present in the parent vector comprising of these two child vectors.
- Similarly, we will be checking whether the required element is present in all of the sub-vectors (child vectors) previously created or not. And, this then would be used to compute whether element is present in the parent vector or not.

These 6 steps of the algorithm would become more clear by the following example:

1) Example 1
   - Consider a vector V.
     $$V \equiv \{34, 4, 56, 999\}$$
     We want to check whether the element X=999 is present or not in this vector using divide and conquer.
   - We will now divide this vector V into two sub-vectors V_1 and V_2 which is equivalent to dividing problem-1 into two sub-problems problem-1a and problem-1b.
     $$V_1 \equiv \{34, 4\}$$
     $$V_2 \equiv \{56, 999\}$$
     Now we will check whether X is present in these sub-problems formed or not.
   - We will now divide vector V_1 into two sub-vectors V_{11} and V_{12} which is equivalent to dividing problem-1a further into two sub-problems problem-1a1 and problem-1a2.
     $$V_{11} \equiv \{34\}$$
     $$V_{12} \equiv \{4\}$$
     Similarly, we will now divide vector V_2 into two sub-vectors V_{21} and V_{22} which is equivalent to dividing problem-1b further into two sub-problems problem-1b1 and problem-1b2.
     $$V_{21} \equiv \{56\}$$
     $$V_{22} \equiv \{999\}$$
     Now we will check whether X is present in these sub-problems formed or not.