Longest common Prefix (Divide & Conquer)

Presented by:

AASTHA SINGH - IIT2019078
VAIBHAV - IIT2019079
RANVEER SINGH - IIT2019080

Department of Information Technology
Indian Institute of Institution Technology, Allahabad

11th March

Content

I. Abstract
II. Introduction
III. Algorithm Description and Analysis
IV. Example
V. Pseudo Code
VI. Time Complexity
VII. Space Complexity
VIII. Conclusion
IX. Reference

I. Abstract

In this paper ,we have devised an algorithm to find the longest common Prefix from a set of strings. We have used the divide and conquer approach to solve the problem and also analysed the time and space complexity of the same.

II. INTRODUCTION

In this problem we have to find the longest common prefix for given ‘n’ strings.

We know that the common prefix for any two strings will be the consecutive common character from the start of the strings.

For example: Let string1 is “India” and string2 is “Indore”, so the common prefix for string1 and string2 will be “Ind”.

III. Algorithm Description and Analysis

1. The algorithm asks for the number of strings ‘N’ as an input.
2. Now the algorithm asks for N strings which are now stored in an array.
3. Now we use the divide and conquer approach to solve this problem.
4. The array of strings is divided into two parts. The same is done for the left part and then the right part.
5. We keep on dividing the arrays until we get a single string.
6. Now we start returning the longest common prefix between the left and the right strings.
7. The longest common prefix between two strings is calculated by simply comparing the strings by iterating through them.
8. At the end we will get the longest common prefix of the array of strings.

IV. EXAMPLE

Let StringArr[ ] = {“**India**”, “**Indonesia**”, “**Indiana**”, “**Indian**” , “**IndianOcean**”}

The array is first divided into 2 parts -

Arr1[ ] = { “**India**”, “**Indonesia**” }
and
Arr2[ ]={ “**Indiana**” ,“**Indian**”, “**IndianOcean**”}

Arr11[ ] = {“**India**” }
Arr12[ ] = { “**Indonesia**” }

Arr21[ ] = { “**Indiana**” }
Arr22[]={“**Indian**”,“**IndianOcean**” }

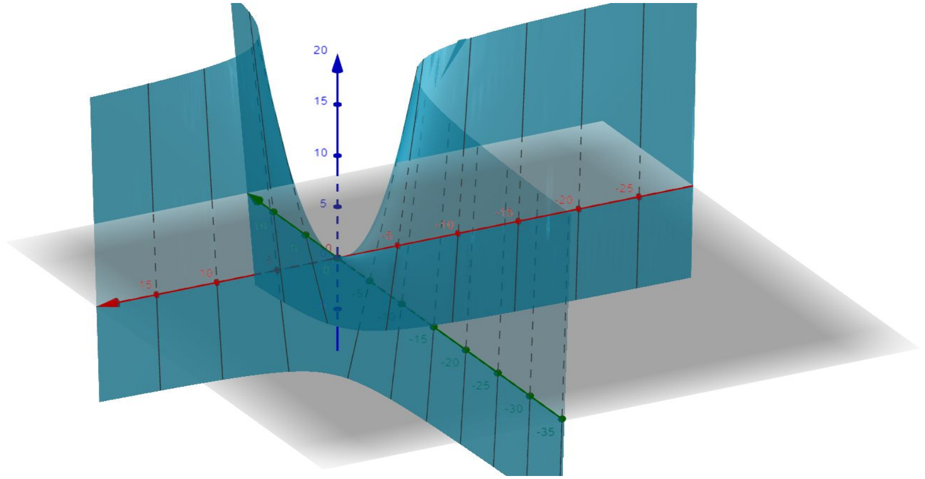
We get “**Ind**” as the answer.

V. PSEUDO CODE

```
Function lcp
Pass In: string s, int lo, int hi
    If lo=hi then
        return s[hi]
    If hi>lo
        mid⇐lo+(hi-lo)/2
        str1⇐function(s,lo,mid)
        str2⇐function(s,mid+1,hi)
        res ⇐ “”
        n1⇐str1.size(), n2⇐str2.size()
        If n1<=n2 then
            sz=n1
        else
            sz=n2
        for i⇐3 to (sz) incrSize 1 do
            If str1[i]=str2[i] then
                res.pushback(str1[i])
            Else
                break
Pass Out: res
```

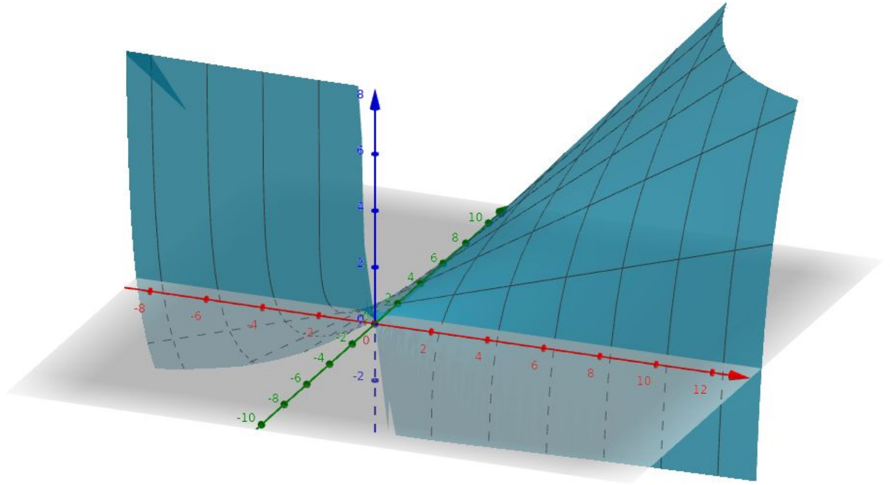
VI. Time Complexity

Since in the lcp function, we are iterating through all the strings in the string array so the complexity of the function will be O(n*m) where n is the total number of strings in the array of strings and m is the length of the longest string among them.



VII. Space Complexity

The Space complexity of the program is O(m*log(n)). This is because we allocate space to the resultant string or the longest common prefix string. There will be a maximum of log(n) divisions and each string returned will have a maximum size of m.



VIII. Conclusion

We have arrived at the solution of the problem by the divide and conquer method, where we are iterating through all the characters of all the strings, so we can say that the time complexity is O(N*M) where,

N = Number of strings

M = Length of the largest string

IX. Reference

1. Introduction_to_Algorithms_Third_Edition_(2009)
2. StackOverflow
3. GeeksforGeeks

THANK YOU