

521. Longest Uncommon Subsequence I

April 1, 2017

|

33.2K views

★★★★★

Average Rating: 3.53 (40 votes)

Given two strings, you need to find the longest uncommon subsequence of this two strings. The longest uncommon subsequence is defined as the longest subsequence of one of these strings and this subsequence should not be **any** subsequence of the other string.

A **subsequence** is a sequence that can be derived from one sequence by deleting some characters without changing the order of the remaining elements. Trivially, any string is a subsequence of itself and an empty string is a subsequence of any string.

The input will be two strings, and the output needs to be the length of the longest uncommon subsequence. If the longest uncommon subsequence doesn't exist, return -1.

Example 1:

Input: a = "aba", b = "cdc"

Output: 3

Explanation: The longest uncommon subsequence is "aba", because "aba" is a subsequence of "aba", but not a subsequence of the other string "cdc". Note that "cdc" can be also a longest uncommon subsequence.

Example 2:

Input: a = "aaa", b = "bbb"

Output: 3

Example 3:

Input: a = "aaa", b = "aaa"

Output: -1

Constraints:

- Both strings' lengths will be between `[1 - 100]`.
- Only letters from a ~ z will appear in input strings.

Solution

Approach #1 Brute Force [Time Limit Exceeded]

In the brute force approach we will generate all the possible 2^n subsequences of both the strings and store their number of occurrences in a hashmap. Longest subsequence whose frequency is equal to 1 will be the required subsequence. And, if it is not found we will return -1 .

Java

Copy

```
1 public class Solution {
2     public int findLUSlength(String a, String b) {
3         HashMap < String, Integer > map = new HashMap < > ();
4         for (String s: new String[] {a, b}) {
5             for (int i = 0; i < (1 << s.length()); i++) {
6                 String t = "";
7                 for (int j = 0; j < s.length(); j++) {
8                     if (((i >> j) & 1) != 0)
9                         t += s.charAt(j);
10                }
11                if (map.containsKey(t))
12                    map.put(t, map.get(t) + 1);
13                else
14                    map.put(t, 1);
15            }
16        }
17        int res = -1;
18        for (String s: map.keySet()) {
19            if (map.get(s) == 1)
20                res = Math.max(res, s.length());
21        }
22        return res;
23    }
24 }
25 }
```

Complexity Analysis

- Time complexity : $O(2^x + 2^y)$, where x and y are the lengths of strings a and b respectively . Number of subsequences will be $2^x + 2^y$.
- Space complexity : $O(2^x + 2^y)$. $2^x + 2^y$ subsequences will be generated.

Approach #2 Simple Solution[Accepted]

Algorithm

Simple analysis of this problem can lead to an easy solution.

These three cases are possible with string a and b :-

- $a = b$. If both the strings are identical, it is obvious that no subsequence will be uncommon. Hence, return -1.
- $length(a) = length(b)$ and $a \neq b$. Example: abc and abd . In this case we can consider any string i.e. abc or abd as a required subsequence, as out of these two strings one string will never be a subsequence of other string. Hence, return $length(a)$ or $length(b)$.
- $length(a) \neq length(b)$. Example $abcd$ and abc . In this case we can consider bigger string as a required subsequence because bigger string can't be a subsequence of smaller string. Hence, return $max(length(a), length(b))$.

Java

Copy

```
1 public class Solution {
2     public int findLUSlength(String a, String b) {
3         if (a.equals(b))
4             return -1;
5         return Math.max(a.length(), b.length());
6     }
7 }
```

Complexity Analysis

- Time complexity : $O(\min(x, y))$. where x and y are the lengths of strings a and b respectively. Here equals method will take $\min(x, y)$ time .
- Space complexity : $O(1)$. No extra space required.

Rate this article: ★★★★★

Comments: 60

Sort By ▼

- 🔍

Type comment here... (Markdown is supported)

👁 Preview

Post
- 🔍

supwy ★114 🕒 November 6, 2018 8:30 PM

I feel I'm a fool.

112 📈 📉 | 📄 Share | 🗨 Reply

SHOW 1 REPLY

🔍

echomingzhu ★59 🕒 December 22, 2018 3:27 PM

mdzz

59 📈 📉 | 📄 Share | 🗨 Reply

📄

Tangcodes ★267 🕒 October 30, 2018 2:54 AM

Popular candidate of the "Most Dislike Question".

55 📈 📉 | 📄 Share | 🗨 Reply

🔍

wanzongmu ★20 🕒 August 19, 2018 10:06 AM

mdzz

20 📈 📉 | 📄 Share | 🗨 Reply

🌐

kevincongcc ★492 🕒 January 19, 2018 11:18 AM

ma de zhi zhang.hahahahahaha

13 📈 📉 | 📄 Share | 🗨 Reply

📄

zhoubowei ★871 🕒 May 24, 2018 6:30 AM

mdzz

39 📈 📉 | 📄 Share | 🗨 Reply

SHOW 7 REPLIES

🔍

leetcode_deleted_user ★275 🕒 May 30, 2017 8:30 AM

OMG, are you kidding me?

11 📈 📉 | 📄 Share | 🗨 Reply

🐱

Jojozzc ★11 🕒 November 2, 2018 12:42 PM

...mdzz. Brain teasers?

10 📈 📉 | 📄 Share | 🗨 Reply

SHOW 1 REPLY

📄

sandro101 ★117 🕒 November 1, 2018 1:48 AM

Why is the question written in such a way that at all points during the processing of reading it there are plurals used when referring to only 1 string? Question seems reasonable to me its just incomprehensible the way it is written

8 📈 📉 | 📄 Share | 🗨 Reply

📄

bodziozet ★97 🕒 January 20, 2019 6:05 AM

I think it's time for shortest uncommon subsequence problem next :D

7 📈 📉 | 📄 Share | 🗨 Reply

SHOW 1 REPLY

<

1

2

3

4

5

6

>