Leetcode 87: Scramble String

Posted on August 16, 2018 · 1 minute read

Question

Given a string s1, we may represent it as a binary tree by partitioning it to two non-empty substrings recursively.

Below is one possible representation of s1 = “great”:

```
    great
   /    \
  gr    eat
 / \    /  \
g   r  e   at
           / \
          a   t
```

To scramble the string, we may choose any non-leaf node and swap its two children.

For example, if we choose the node “gr” and swap its two children, it produces a scrambled string “rgeat”.

```
    rgeat
   /    \
  rg    eat
 / \    /  \
r   g  e   at
           / \
          a   t
```

We say that “rgeat” is a scrambled string of “great”.

Similarly, if we continue to swap the children of nodes “eat” and “at”, it produces a scrambled string “rgtae”.

```
    rgtae
   /    \
  rg    tae
 / \    /  \
r   g  ta  e
       / \
      t   a
```

We say that “rgtae” is a scrambled string of “great”.

Given two strings s1 and s2 of the same length, determine if s2 is a scrambled string of s1.

Example 1:

```
Input: s1 = "great", s2 = "rgeat"
Output: true
```

Example 2:

```
Input: s1 = "abcde", s2 = "caebd"
Output: false
```

Solution

Divide-and-Conquer.

```
class Solution {
    public boolean isScramble(String s1, String s2) {
        if(s1.length() != s2.length())
            return false;
        else {
            int[] count = new int[26];
            Arrays.fill(count, 0);
            for(int i = 0; i < s1.length(); i++) {
                count[s1.charAt(i) - 'a']++;
            }
            for(int i = 0; i < s2.length(); i++) {
                if(--count[s2.charAt(i) - 'a'] < 0)
                    return false;
            }
            if(s1.equals(s2)) return true;
            for (int i = 1; i < s1.length(); i++) {
                if (isScramble(s1.substring(0, i), s2.substring(0, i))
                 && isScramble(s1.substring(i), s2.substring(i))) return true;
                if (isScramble(s1.substring(0, i), s2.substring(s2.length() - i))
                 && isScramble(s1.substring(i), s2.substring(0, s2.length() - i))) retu
            }
            return false;
        }
    }
}
```

Tags: Leetcode Hard

← PREVIOUS POST

NEXT POST →