<https://leetcode.ca/all/1257.html>

You are given some lists of regions where the first region of each list includes all other regions in that list.

Naturally, if a region X contains another region Y then X is bigger than Y. Also by definition a region X contains itself.

Given two regions region1, region2, find out the smallest region that contains both of them.

If you are given regions r1, r2 and r3 such that r1 includes r3, it is guaranteed there is no r2 such that r2 includes r3, it's guaranteed the smallest region exists.

Example 1:

Input:

regions = [["Earth","North America","South America"],

["North America","United States","Canada"],

["United States","New York","Boston"],

["Canada","Ontario","Quebec"],

["South America","Brazil"]],

region1 = "Quebec",

region2 = "New York"

Output: "North America"

Constraints:

* 2 <= regions.length <= 10^4
* region1 != region2
* All strings consist of English letters and spaces with at most 20 letters.

**Attempt 1: 2023-05-29**

**Solution 1:  HashMap + DFS (10 min)**

class Solution {

public String findSmallestRegion(List<List<String>> regions, String region1, String region2) {

Map<String, String> parent = new HashMap<>();

Set<String> ancestors = new HashSet<>(); // region1's ancestors

for (List<String> region : regions)

for (int i = 1; i < region.size(); ++i)

parent.put(region.get(i), region.get(0));

// Add all of region1's ancestors

while (region1 != null) {

ancestors.add(region1);

region1 = parent.get(region1); // Region1 becomes null in the end

}

// Go up from region2 until meet any of region1's ancestors

while (!ancestors.contains(region2))

region2 = parent.get(region2);

return region2;

}

}

Time Complexity : O(N^2)

Space Complexity : O(N)

**Refer to**

<https://blog.51cto.com/u_15127506/3427658>

e.g

regions = [["Earth","North America","South America"],

["North America","United States","Canada"],

["United States","New York","Boston"],

["Canada","Ontario","Quebec"],

["South America","Brazil"]],

region1 = "Quebec",

region2 = "New York"

Output: "North America"

-----------------------------------------------------

Earth

/ \

North America South America

/ \ /

US Canada Brazil

/ \ / \

NY Boston Ontario Quebec

这道题跟235，236题很像，找的是两个 region 的最小公共祖先。那么这里我们需要一个hashmap<String, String> 记录每个 region 和他的父节点。同时我们还需要一个 hashset 记录有哪些 region 被访问过。首先我们用 hashmap 记录每个 region 和他的父节点，记录好了之后，对于 region1 ，我们开始执行 DFS，用 hashset 记录好 region1 的每一个父节点，直到无法遍历为止。

此时我们再从 region2 开始做 DFS 遍历去找 region2 的父节点。如果当前找到的这个父节点不存在于 hashset，就接着再往上一层找；如果在 hashset 里找到了 region2 的父节点，说明这个父节点就是 region1 和 region2 的最小公共祖先。

时间O(n^2)

空间O(n)

Java实现

class Solution {

public String findSmallestRegion(List < List < String >> regions, String region1, String region2) {

HashMap < String, String > map = new HashMap < > ();

//build tree

for (List < String > region: regions) {

String value = region.get(0);

for (int i = 1; i < region.size(); i++) {

String key = region.get(i);

// this map is about the child point to the parent

map.put(key, value);

}

}

//start from region1, use a hashset to store the parent path

HashSet < String > parents = new HashSet < > ();

while (region1 != null) {

//add it to the set for future search

parents.add(region1);

// get region1's parent

String parent = map.get(region1);

// update region1 to parent

region1 = parent;

}

while (!parents.contains(region2)) {

String parent = map.get(region2);

region2 = parent;

}

return region2;

}

}