Problem 2322: Minimum Score After Removals on a Tree

Problem Information

Difficulty: Hard
Acceptance Rate: 76.35%
Paid Only: No
Tags: Array, Bit Manipulation, Tree, Depth-First Search

Problem Description

There is an undirected connected tree with `n` nodes labeled from `0` to `n - 1` and `n - 1` edges.

You are given a **0-indexed** integer array `nums` of length `n` where `nums[i]` represents the value of the `ith` node. You are also given a 2D integer array `edges` of length `n - 1` where `edges[i] = [ai, bi]` indicates that there is an edge between nodes `ai` and `bi` in the tree.

Remove two **distinct** edges of the tree to form three connected components. For a pair of removed edges, the following steps are defined:

1. Get the XOR of all the values of the nodes for **each** of the three components respectively. 2. The **difference** between the **largest** XOR value and the **smallest** XOR value is the **score** of the pair.

* For example, say the three components have the node values: `[4,5,7]`, `[1,9]`, and `[3,3,3]`. The three XOR values are `4 ^ 5 ^ 7 = _**6**_`, `1 ^ 9 = _**8**_`, and `3 ^ 3 ^ 3 = _**3**_`. The largest XOR value is `8` and the smallest XOR value is `3`. The score is then `8 - 3 = 5`.

Return _the**minimum** score of any possible pair of edge removals on the given tree_.

Example 1:

![](https://assets.leetcode.com/uploads/2022/05/03/ex1drawio.png)

Input: nums = [1,5,5,4,11], edges = [[0,1],[1,2],[1,3],[3,4]] **Output:** 9 **Explanation:** The diagram above shows a way to make a pair of removals. - The 1st component has nodes [1,3,4] with values [5,4,11]. Its XOR value is 5 ^ 4 ^ 11 = 10. - The 2nd component has node [0] with value [1]. Its XOR value is 1 = 1. - The 3rd component has node [2] with value [5]. Its XOR value is 5 = 5. The score is the difference between the largest and smallest XOR value which is 10 - 1 = 9. It can be shown that no other pair of removals will obtain a smaller score than 9.

Example 2:

![](https://assets.leetcode.com/uploads/2022/05/03/ex2drawio.png)

Input: nums = [5,5,2,4,4,2], edges = [[0,1],[1,2],[5,2],[4,3],[1,3]] **Output:** 0 **Explanation:** The diagram above shows a way to make a pair of removals. - The 1st component has nodes [3,4] with values [4,4]. Its XOR value is 4 ^ 4 = 0. - The 2nd component has nodes [1,0] with values [5,5]. Its XOR value is 5 ^ 5 = 0. - The 3rd component has nodes [2,5] with values [2,2]. Its XOR value is 2 ^ 2 = 0. The score is the difference between the largest and smallest XOR value which is 0 - 0 = 0. We cannot obtain a smaller score than 0.

Constraints:

* `n == nums.length` * `3 <= n <= 1000` * `1 <= nums[i] <= 108` * `edges.length == n - 1` * `edges[i].length == 2` * `0 <= ai, bi < n` * `ai != bi` * `edges` represents a valid tree.

Code Snippets

C++:

```
class Solution {
public:
int minimumScore(vector<int>& nums, vector<vector<int>>& edges) {

}
};
```

Java:

```
class Solution {
public int minimumScore(int[] nums, int[][] edges) {
}
}
```

Python3:

```
class Solution:
def minimumScore(self, nums: List[int], edges: List[List[int]]) -> int:
```