Problem 3600: Maximize Spanning Tree Stability with Upgrades

Problem Information

Difficulty: Hard
Acceptance Rate: 39.11%
Paid Only: No
Tags: Binary Search, Greedy, Union Find, Graph, Minimum Spanning Tree

Problem Description

You are given an integer `n`, representing `n` nodes numbered from 0 to `n - 1` and a list of `edges`, where `edges[i] = [ui, vi, si, musti]`:

* `ui` and `vi` indicates an undirected edge between nodes `ui` and `vi`. * `si` is the strength of the edge. * `musti` is an integer (0 or 1). If `musti == 1`, the edge **must** be included in the******spanning tree**. These edges **cannot** be **upgraded**.

You are also given an integer `k`, the **maximum** number of upgrades you can perform. Each upgrade **doubles** the strength of an edge, and each eligible edge (with `musti == 0`) can be upgraded **at most** once.

The **stability** of a spanning tree is defined as the **minimum** strength score among all edges included in it.

Return the **maximum** possible stability of any valid spanning tree. If it is impossible to connect all nodes, return `-1`.

Note : A **spanning tree** of a graph with `n` nodes is a subset of the edges that connects all nodes together (i.e. the graph is **connected**) _without_ forming any cycles, and uses **exactly** `n - 1` edges.

Example 1:

Input: n = 3, edges = [[0,1,2,1],[1,2,3,0]], k = 1

Output: 2

Explanation:

* Edge `[0,1]` with strength = 2 must be included in the spanning tree. * Edge `[1,2]` is optional and can be upgraded from 3 to 6 using one upgrade. * The resulting spanning tree includes these two edges with strengths 2 and 6. * The minimum strength in the spanning tree is 2, which is the maximum possible stability.

Example 2:

Input: n = 3, edges = [[0,1,4,0],[1,2,3,0],[0,2,1,0]], k = 2

Output: 6

Explanation:

* Since all edges are optional and up to `k = 2` upgrades are allowed. * Upgrade edges `[0,1]` from 4 to 8 and `[1,2]` from 3 to 6. * The resulting spanning tree includes these two edges with strengths 8 and 6. * The minimum strength in the tree is 6, which is the maximum possible stability.

Example 3:

Input: n = 3, edges = [[0,1,1,1],[1,2,1,1],[2,0,1,1]], k = 0

Output: -1

Explanation:

* All edges are mandatory and form a cycle, which violates the spanning tree property of acyclicity. Thus, the answer is -1.

Constraints:

* `2 <= n <= 105` * `1 <= edges.length <= 105` * `edges[i] = [ui, vi, si, musti]` * `0 <= ui, vi < n` * `ui != vi` * `1 <= si <= 105` * `musti` is either `0` or `1`. * `0 <= k <= n` * There are no duplicate edges.

Code Snippets

C++:

```
class Solution {
public:
int maxStability(int n, vector<vector<int>>& edges, int k) {

}
};
```

Java:

```
class Solution {
public int maxStability(int n, int[][] edges, int k) {

}
}
```

Python3:

```
class Solution:
def maxStability(self, n: int, edges: List[List[int]], k: int) -> int:
```