2493. Divide Nodes Into the Maximum Number of Groups

My Submissions (/contest/weekly-contest-322/problems/divide-nodes-into-the-maximum-number-of-groups/submissions/)

Back to Contest (/contest/weekly-contest-322/)

User Accepted:	403
User Tried:	1249
Total Accepted:	447
Total Submissions:	2892
Difficulty:	Hard

You are given a positive integer `n` representing the number of nodes in an **undirected** graph. The nodes are labeled from `1` to `n`.

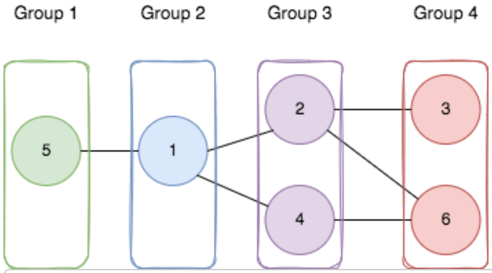
You are also given a 2D integer array `edges`, where `edges[i] = [`a_i`, `b_i`]` indicates that there is a **bidirectional** edge between nodes a_i and b_i. **Notice** that the given graph may be disconnected.

Divide the nodes of the graph into `m` groups (**1-indexed**) such that:

- Each node in the graph belongs to exactly one group.
- For every pair of nodes in the graph that are connected by an edge `[`a_i`, `b_i`]`, if a_i belongs to the group with index `x`, and b_i belongs to the group with index `y`, then `|y - x| = 1`.

Return *the maximum number of groups (i.e., maximum* `m`*) into which you can divide the nodes*. Return `-1` *if it is impossible to group the nodes with the given conditions*.

Example 1:



```
Input: n = 6, edges = [[1,2],[1,4],[1,5],[2,6],[2,3],[4,6]]
Output: 4
Explanation: As shown in the image we:
- Add node 5 to the first group.
- Add node 1 to the second group.
- Add nodes 2 and 4 to the third group.
- Add nodes 3 and 6 to the fourth group.
We can see that every edge is satisfied.
It can be shown that that if we create a fifth group and move any node from the third or fourth group to it, at least on of
```

Example 2:

```
Input: n = 3, edges = [[1,2],[2,3],[3,1]]
Output: -1
Explanation: If we add node 1 to the first group, node 2 to the second group, and node 3 to the third group to satisfy the
It can be shown that no grouping is possible.
```

Constraints:

- `1 <= n <= 500`
- `1 <= edges.length <=` 10^4
- `edges[i].length == 2`
- `1 <=` a_i`,` b_i `<= n`
- a_i `!=` b_i
- There is at most one edge between any pair of vertices.

Discuss (https://leetcode.com/problems/divide-nodes-into-the-maximum-number-of-groups/discuss)

JavaScript

```javascript
////////////////////// Template ////////////////////////////////////////
const initializeGraph = (n) => { let g = []; for (let i = 0; i < n; i++) { g.push([]); } return g; };
const packUG = (g, edges) => { for (const [u, v] of edges) { g[u].push(v); g[v].push(u); } };

function DJSet(n) {
    // parent[i] < 0, -parent[i] is the group size which root is i. example: (i -> parent[i] -> parent[parent[i]] -> parent[parent[parent[i]]] ...)
    // parent[i] >= 0, i is not the root and parent[i] is i's parent. example: (... parent[parent[parent[i]]] -> parent[parent[i]] -> parent[i] -> i)
    let parent = Array(n).fill(-1);
    return { find, union, count, equiv, par }
    function find(x) {
        return parent[x] < 0 ? x : parent[x] = find(parent[x]);
    }
    function union(x, y) {
        x = find(x);
        y = find(y);
        if (x != y) {
            if (parent[x] < parent[y]) [x, y] = [y, x];
            parent[x] += parent[y];
            parent[y] = x;
        }
        return x == y;
    }
    function count() { // total groups
        return parent.filter(v => v < 0).length;
    }
    function equiv(x, y) { // isConnected
        return find(x) == find(y);
    }
    function par() {
        return parent;
    }
}

const isBipartite = (g) => {
    let n = g.length, start = 1, visit = Array(n).fill(false), q = [], color = Array(n).fill(0); // 0: no color, 1: red  -1: blue
    for (let i = start; i < n; i++) {
        if (color[i] != 0) continue;
        q.push(i);
        color[i] = 1;
        if (visit[i]) continue;
        while (q.length) {
            let cur = q.shift();
            if (visit[cur]) continue;
            for (const child of g[cur]) {
                if (color[child] == color[cur]) return false;
                if (color[child]) continue;
                color[child] = -color[cur];
                q.push(child);
            }
        }
    }
    return true;
};
////////////////////////////////////////////////////////////////////

const magnificentSets = (n, edges) => {
    let g = initializeGraph(n + 1), ds = new DJSet(n + 1);
    packUG(g, edges);
    if (!isBipartite(g)) return -1;
    let res = Array(n + 1).fill(0);
    for (const [u, v] of edges) ds.union(u, v);
    let d = floyd_warshall(n + 1, edges);
    for (let i = 1; i <= n; i++) {
        let max = 0;
        for (let j = 1; j <= n; j++) {
            if (d[i][j] >= Number.MAX_SAFE_INTEGER) continue;
            max = Math.max(max, d[i][j]);
        }
        let par = ds.find(i);
        res[par] = Math.max(res[par], max + 1);
    }
    let ans = 0;
    for (let i = 1; i <= n; i++) ans += res[i];
    return ans;
};

const floyd_warshall = (n, edges) => {
    let d = [...Array(n)].map(() => Array(n).fill(Number.MAX_SAFE_INTEGER)), start = 1;
    for (let i = start; i < n; i++) d[i][i] = 0;
    for (const [u, v, weight] of edges) {
        let w = weight || 1;
        d[u][v] = w;
        d[v][u] = w;
    }
    for (let k = start; k < n; k++) {
        for (let i = start; i < n; i++) {
            for (let j = start; j < n; j++) {
                if (d[i][j] > d[i][k] + d[k][j]) d[i][j] = d[i][k] + d[k][j];
            }
        }
    }
    return d;
};
```

□ **Custom Testcase** Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/856522357/) More Details (/submissions/detail/856522357/)

Share your acceptance!

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)

United States (/region)