Divide Intervals Into Minimum Number Of Groups

solved by	Senan
Platform	LeetCode
difficulty	Medium
# Serial	2406
tags	2 Pointers Array
language	C++
solved on	@12/10/2024
link	https://leetcode.com/problems/divide-intervals-into-minimum-number-of-groups/description/
☑ Completion	☐

Intuition

The problem can be thought of as tracking the number of overlapping intervals. Each interval starts and ends at specific points, and our goal is to find the maximum number of intervals that are active at any point in time.

Approach

We use a sweep line technique by marking the start and end points of each interval in a `time` array. For each interval, we increment the start point and decrement the point just after the end of the interval. Then, we perform a cumulative sum across the `time` array to track how many intervals are active at each point. The maximum value in this cumulative sum will give us the answer, i.e., the maximum number of overlapping intervals at any time.

Complexity

Time Complexity:

- **O(n)**: We iterate over the list of intervals once to mark the start and end points.
- **O(m)**: We perform a cumulative sum over the `time` array of size `m`, where `m` is a constant (since we assume the range is bounded by 1,000,001).
- Overall: **O(n + m)**, where `n` is the number of intervals and `m` is the size of the `time` array (which is constant in this case).

Space Complexity:

- **O(m)**: We use a `time` array of size `m = 1,000,002`.
- **O(1)**: Additional space for a few integer variables.
- Overall: **O(m)**, where `m = 1,000,002` is the size of the `time` array.

Code

```
class Solution {
public:
    int minGroups(vector<vector<int>>& intervals) {
        vector<int> time(1000002, 0);
        for (auto& interval : intervals) {
            time[interval[0]]++;
            time[interval[1] + 1]--;
        }
        int maxGrp = 0, currGrp = 0;
        for (int i = 0; i < time.size(); i++) {
            currGrp += time[i];
            maxGrp = max(maxGrp, currGrp);
        }
        return maxGrp;
    }
};
```