0487. 最大连续1的个数 II

ITCharge 大约 2 分钟

- 标签：数组、动态规划、滑动窗口
- 难度：中等

题目链接

- 0487. 最大连续1的个数 II - 力扣

题目大意

描述：给定一个二进制数组 $nums$，可以最多将 1 个 0 翻转为 1。

要求：如果最多可以翻转一个 0，则返回数组中连续 1 的最大个数。

说明：

- 1 <= nums.length <= 105 nums[i] 是 0 就是 1.

示例：

- 示例 1：

```py
输入：nums = [1,0,1,1,0]
输出：4
解释：翻转第一个 0 可以得到最长的连续 1。当翻转以后，最大连续 1 的个数为 4。
```

- 示例 2：

```py
输入：nums = [1,0,1,1,0,1]
输出：4
```

解题思路

思路 1：滑动窗口

暴力做法是尝试将每个位置的 0 分别变为 1，然后统计最大连续 1 的个数。但这样复杂度就太高了。

我们可以使用滑动窗口来解决问题。保证滑动窗口内最多有 1 个 0。具体做法如下：

设定两个指针：$left$、$right$，分别指向滑动窗口的左右边界，保证滑动窗口内最多有 1 个 0。使用 $zero\_count$ 统计窗口内 1 的个数。使用 ans 记录答案。

- 一开始，$left$、$right$ 都指向 0。
- 如果 $nums[right] == 0$，则窗口内 1 的个数加 1。
- 如果该窗口中 1 的个数多于 1 个，即 $zero\_count > 1$，则不断右移 $left$，缩小滑动窗口长度，并更新窗口中 1 的个数，直到 $zero\_count \leq 1$。
- 维护更新最大连续 1 的个数。然后右移 $right$，直到 $right \geq len(nums)$ 结束。
- 输出最大连续 1 的个数。

思路 1：代码

py

```
class Solution:
    def findMaxConsecutiveOnes(self, nums: List[int]) -> int:
        left, right = 0, 0
        ans = 0
        zero_count = 0

        while right < len(nums):
            if nums[right] == 0:
                zero_count += 1
            while zero_count > 1:
                if nums[left] == 0:
                    zero_count -= 1
```

```
            left += 1
        ans = max(ans, right - left + 1)
        right += 1

    return ans
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$，其中 n 为数组 $nums$ 的长度。
- **空间复杂度**：$O(1)$。