0424. 替换后的最长重复字符

ITCharge 大约 2 分钟

- 标签：哈希表、字符串、滑动窗口
- 难度：中等

题目链接

- 0424. 替换后的最长重复字符 - 力扣

题目大意

描述：给定一个仅由大写英文字母组成的字符串 s，以及一个整数 k。可以将任意位置上的字符替换成另外的大写字母，最多可替换 k 次。

要求：在进行上述操作后，找到包含重复字母的最长子串长度。

说明：

- $1 \leq s.length \leq 10^5$。
- s 仅由大写英文字母组成。
- $0 \leq k \leq s.length$。

示例：

- 示例 1：

```py
输入：s = "ABAB", k = 2
输出：4
解释：用两个'A'替换为两个'B',反之亦然。
```

- 示例 2：

```py
输入：s = "AABABBA", k = 1
输出：4
解释：
将中间的一个'A'替换为'B',字符串变为 "AABBBBA"。
子串 "BBBB" 有最长重复字母，答案为 4。
可能存在其他的方法来得到同样的结果。
```

解题思路

先来考虑暴力求法。枚举字符串 s 的所有子串，对于每一个子串：

- 统计子串中出现次数最多的字符，替换除它以外的字符 k 次。
- 维护最长子串的长度。

但是这种暴力求法中，枚举子串的时间复杂度为 $O(n^2)$，统计出现次数最多的字符和替换字符时间复杂度为 $0(n)$，且两者属于平行处理，总体下来的时间复杂度为 $O(n^3)$。这样做会超时。

思路 1：滑动窗口

1. 使用 counts 数组来统计字母频数。使用 left、right 双指针分别指向滑动窗口的首尾位置，使用 max_count 来维护最长子串的长度。
2. 不断右移 right 指针，增加滑动窗口的长度。
3. 对于当前滑动窗口的子串，如果当前窗口的间距 > 当前出现最大次数的字符的次数 + k 时，意味着替换 k 次仍不能使当前窗口中的字符全变为相同字符，则此时应该将左边界右移，同时将原先左边界的字符频次减少。

思路 1：代码

```py
class Solution:
    def characterReplacement(self, s: str, k: int) -> int:
        max_count = 0
        left, right = 0, 0
        counts = [0 for _ in range(26)]
        while right < len(s):
            num_right = ord(s[right]) - ord('A')
            counts[num_right] += 1
            max_count = max(max_count, counts[num_right])
```

```
            right += 1
            if right - left > max_count + k:
                num_left = ord(s[left]) - ord('A')
                counts[num_left] -= 1
                left += 1

        return right - left
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$，其中 n 为字符串的长度。
- **空间复杂度**：$O(|\sum|)$，其中 $\sum$ 是字符集，本题中 $|\sum| = 26$。