5760. Minimum Number of Swaps to Make the Binary String Alternating

My Submissions (/contest/weekly-contest-241/problems/minimum-number-of-swaps-to-make-the-binary-string-alternating/submissions/)

Back to Contest (/contest/weekly-contest-241/)

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

Given a binary string `s`, return *the **minimum** number of character swaps to make it **alternating**, or* `-1` *if it is impossible.*

The string is called **alternating** if no two adjacent characters are equal. For example, the strings `"010"` and `"1010"` are alternating, while the string `"0100"` is not.

Any two characters may be swapped, even if they are **not adjacent**.

Example 1:

```
Input: s = "111000"
Output: 1
Explanation: Swap positions 1 and 4: "111000" -> "101010"
The string is now alternating.
```

Example 2:

```
Input: s = "010"
Output: 0
Explanation: The string is already alternating, no swaps are needed.
```

Example 3:

```
Input: s = "1110"
Output: -1
```

Constraints:

- `1 <= s.length <= 1000`
- `s[i]` is either `'0'` or `'1'`.

JavaScript

```
const minSwaps = (s) => {
    let n = s.length;
    let r1 = create(n, '0');
    let r2 = create(n, '1');
    // pr(r1, canMake(s, r1), r2, canMake(s, r2));
    // if (r1 == s || r2 == s) return 0;
    if (canMake(s, r1)) {
        if (canMake(s, r2)) {
            let cnt1 = cal(s, r1);
            let cnt2 = cal(s, r2);
            // pr(cnt1, cnt2)
            return Math.min(cnt1, cnt2);
        } else {
            return cal(s, r1);
        }
    } else {
        if (canMake(s, r2)) {
            return cal(s, r2);
        } else {
            return -1;
        }
    }
};

const cal = (s, r) => {
    let n = s.length;
    let cnt = 0;
    for (let i = 0; i < n; i++) {
        if (s[i] != r[i]) cnt++;
    }
    // pr(cnt, s, "origin", r)
    return cnt / 2;
};

const canMake = (s, r) => {
    let n = s.length;
    let ms = counter(s);
    let mr = counter(r);
    // pr(ms, mr);
    if (ms.get('0') != mr.get('0') || ms.get('1') != mr.get('1')) return 0;
    return 1;
};

const counter = (a_or_s) => { let map = new Map(); for (const i of a_or_s) map.set(i, map.get(i) + 1 ||
1); return map; };
const create = (n, start) => {
    let res = start;
    for (let i = 1; i < n; i++) {
        res += (start ^= 1)
    }
    return res;
};
```

Custom Testcase Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/493756968/) More Details (/submissions/detail/493756968/)

Share your acceptance!