[Java/Python 3] Sliding Window O(n) code w/ brief explanation and analysis.

rock ★ 5276 Last Edit: August 12, 2019 11:51 AM 976 VIEWS

15

1. Maintain a sliding window of `width`, which is the number of `1`'s in `data`;
2. Find the max number of `1`'s, `maxWin`, in that window;
3. Then the remaining `0`'s in the window with max number of `1`'s, `width - maxWin`, is the minimum swaps to get a window of all `1`'s.

Java:

```
public int minSwaps(int[] data) {
    int maxWin = 0, width = Arrays.stream(data).sum(); // count the number of 1's in data.
    for (int l = -1, r = 0, cntWin = 0; r < data.length; ++r) {
        cntWin += data[r];
        if (r - l > width) { cntWin -= data[++l]; } // wider than width, shrink the lower bound to maintain its wi
        maxWin = Math.max(cntWin, maxWin);
    }
    return width - maxWin;
}
```

Update: credit to @venendroid for correction of last statement.

Python 3:

```
def minSwaps(self, data: List[int]) -> int:
    width, maxWin, cntWin, l = sum(data), 0, 0, -1
    for r, d in enumerate(data):
        cntWin += d
        if r - l > width:
            l += 1
            cntWin -= data[l]
        maxWin = max(maxWin, cntWin)
    return width - maxWin
```

Analysis:
Time: O(n), space: O(1).

Comments: 1 Best Most Votes Newest to Oldest Oldest to Newest

Type comment here... (Markdown is supported)

Post

venendroid ★ 35 Last Edit: August 11, 2019 12:30 AM

@rock I think, last statement should be "width - maxwin"?

1 Reply