# 1272. Remove Interval

Dec. 15, 2019 | 2K views

Average Rating: 5 (2 votes)

Given a **sorted** list of disjoint `intervals`, each interval `intervals[i] = [a, b]` represents the set of real numbers `x` such that `a <= x < b`.

We remove the intersections between any interval in `intervals` and the interval `toBeRemoved`.

Return a **sorted** list of `intervals` after all such removals.

**Example 1:**

```
Input: intervals = [[0,2],[3,4],[5,7]], toBeRemoved = [1,6]
Output: [[0,1],[6,7]]
```

**Example 2:**

```
Input: intervals = [[0,5]], toBeRemoved = [2,3]
Output: [[0,2],[3,5]]
```

**Constraints:**

- `1 <= intervals.length <= 10^4`
- `-10^9 <= intervals[i][0] < intervals[i][1] <= 10^9`

## Solution

### Approach 1: Sweep Line, One Pass.

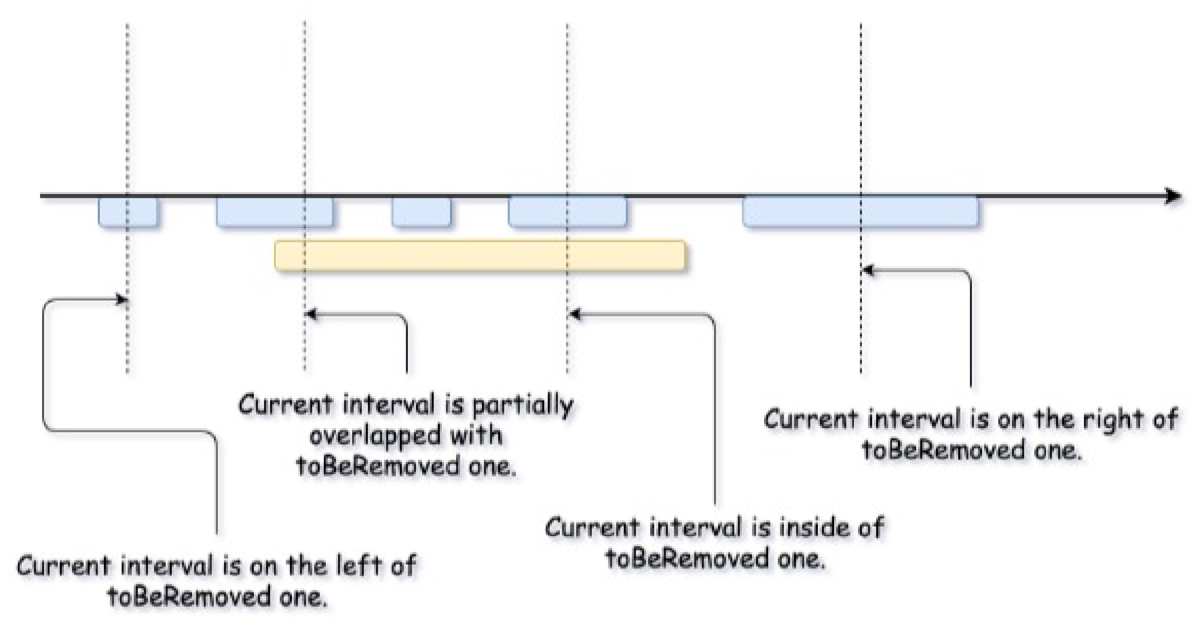
**Best Possible Time Complexity**

> What is the best possible time complexity here?

The input is sorted, that usually means *at least* linear time complexity. Is it possible to do $\mathcal{O}(\log N)$? No, because to copy input elements into output still requires $\mathcal{O}(N)$ time.

**Sweep Line**

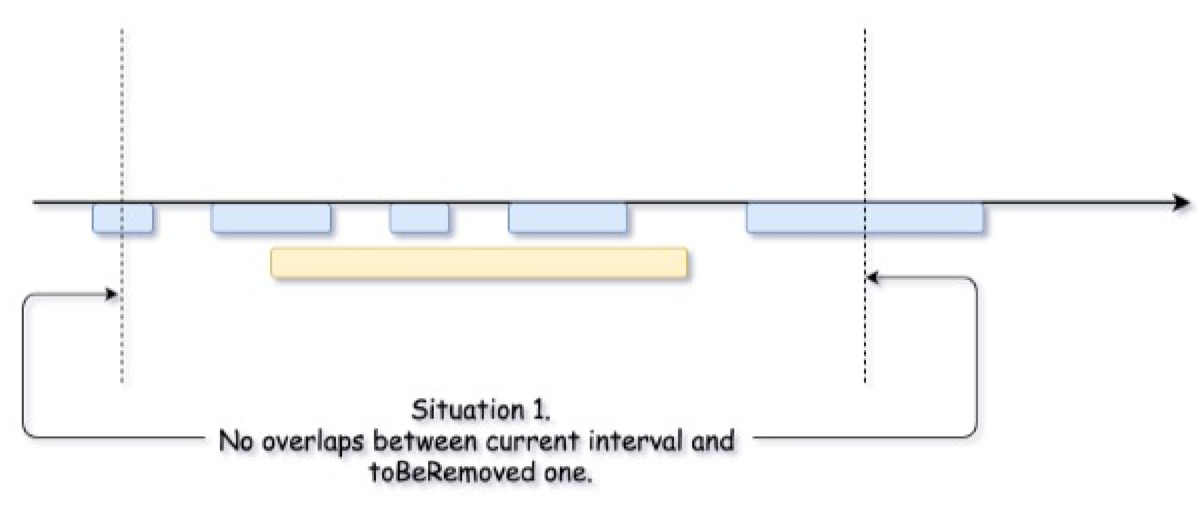
Sweep Line algorithm is a sort of geometrical visualisation. Let's imagine a vertical line which is swept across the plane, stopping at some points. That could create various situations, and the decision to make depends on the stop point.


Current interval is partially overlapped with toBeRemoved one.
Current interval is on the right of toBeRemoved one.
Current interval is inside of toBeRemoved one.
Current interval is on the left of toBeRemoved one.

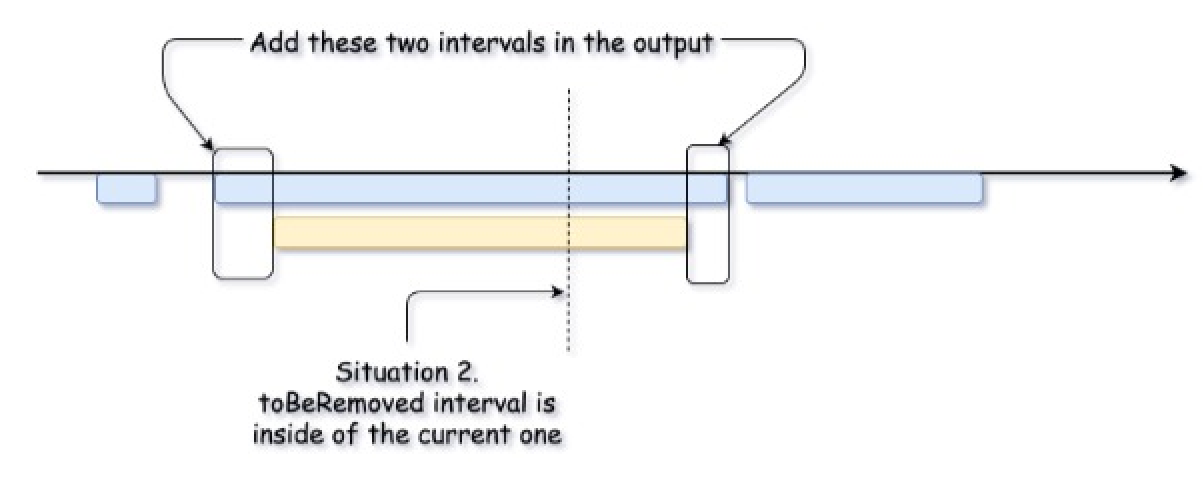
**Algorithm**

Let's sweep the line by iterating over input intervals and consider what it could bring to us.

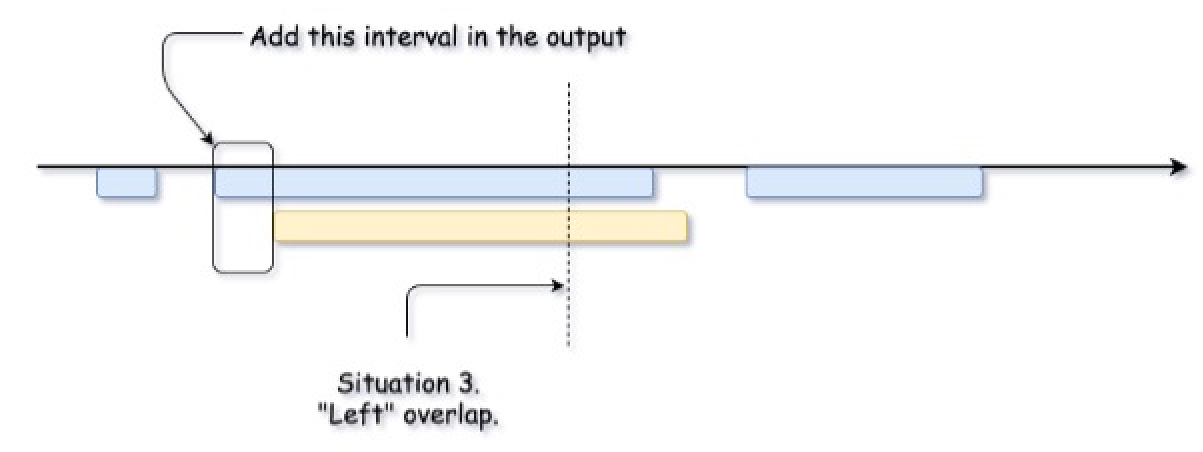
- Current interval has no overlaps with toBeRemoved one. That means there is nothing to take care about, just update the output.



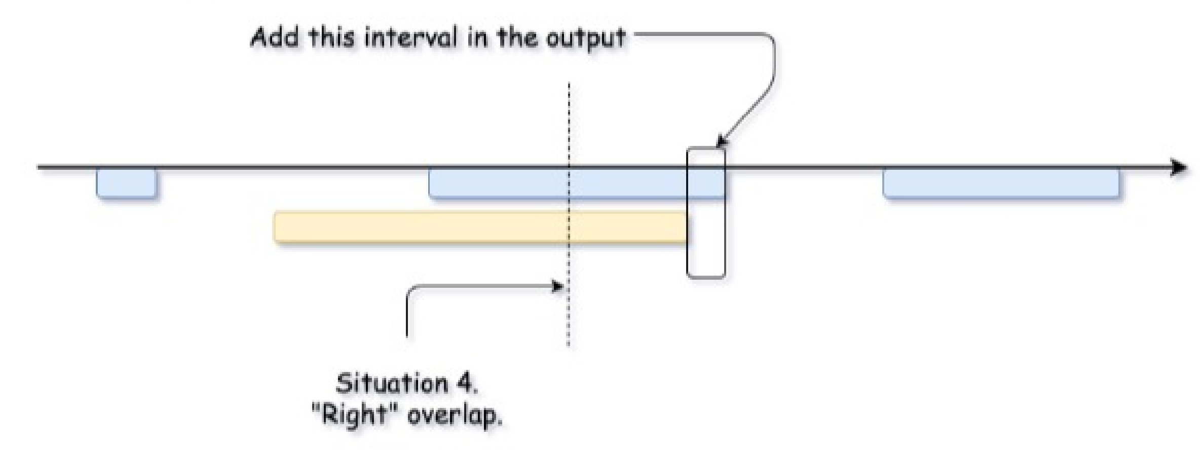
- Second situation is when toBeRemoved interval is inside of the current interval. Then one has to add two non-overlapping parts of the current interval in the output.


Add these two intervals in the output
Situation 2.
toBeRemoved interval is inside of the current one

- "Left" overlap.



- "Right" overlap.



And here we are, all situations are covered, the job is done.

**Implementation**

Java Python Copy

```python
class Solution:
    def removeInterval(self, intervals: List[List[int]], toBeRemoved: List[int]) -> List[List[int]]:
        remove_start, remove_end = toBeRemoved
        output = []

        for start, end in intervals:
            # if current interval ends before toBeRemoved
            # or starts after
            if end <= remove_start or start >= remove_end:
                output.append([start, end])

            # if the interval to be removed is inside
            # of the current interval
            elif start < remove_start and end > remove_end:
                output.append([start, remove_start])
                output.append([remove_end, end])

            # "left" overlap
            elif start < remove_start and end <= remove_end:
                output.append([start, remove_start])

            # "right" overlap
            elif start >= remove_start and end > remove_end:
                output.append([remove_end, end])

        return output
```

**Complexity Analysis**

- Time complexity : $\mathcal{O}(N)$ since it's one pass along the input array.
- Space complexity : $\mathcal{O}(N)$ to keep the output.

Analysis written by @liaison and @andvary

Rate this article: ★ ★ ★ ★ ★

Previous Next

Comments: 1 Sort By

Type comment here... (Markdown is supported)

Preview Post

iliaTalalai ★ 0 December 17, 2019 4:59 AM

if the input is sorted then why do you need to go through the list at all?
you know input[0][0] is min and input [last][last] is max
then you split min and max if the rangeToBeRemoved is within the bounds

```
function removeIntervalsIfSorted(rangeArr, rangeToBeRemoved){
```

Read More

0 Share Reply