37. Merge Overlapping Intervals

Tuesday, April 25, 2023 5:16 PM

56. Merge Intervals

Medium 18670 634 Add to List Share

Given an array of `intervals` where `intervals[i] = [start_i, end_i]`, merge all overlapping intervals, and return *an array of the non-overlapping intervals that cover all the intervals in the input.*

Example 1:

```
Input: intervals = [[1,3],[2,6],[8,10],[15,18]]
Output: [[1,6],[8,10],[15,18]]
Explanation: Since intervals [1,3] and [2,6] overlap, merge them into [1,6].
```

Example 2:

```
Input: intervals = [[1,4],[4,5]]
Output: [[1,5]]
Explanation: Intervals [1,4] and [4,5] are considered overlapping.
```

Merge Overlapping Subintervals ?

[(1,3) (2,6) (8,9) (9,11) (8,10) (2,4) (15,18) (16,17)]

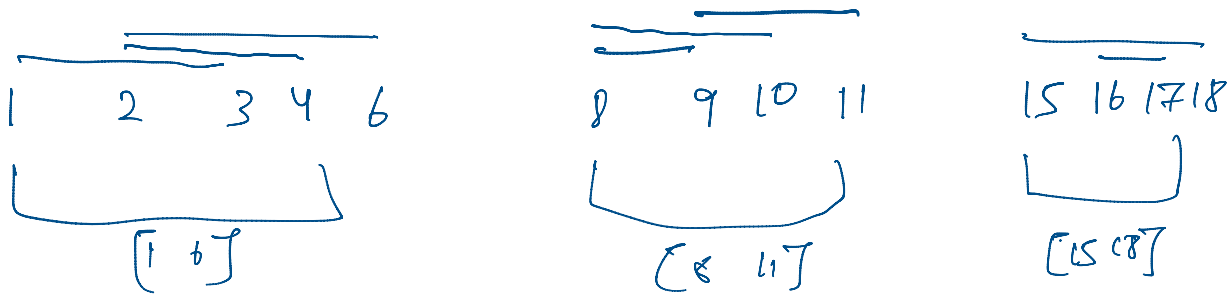
Subintervals => (1, 3)

↓

start with 1 and end with 3

start with 1 and end with 3

{1 6} {8 11} {15, 18}



1) Brute force:

- Sort then

(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)

- Start iterating over

(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)

- (1,3) now in (2,4) if 2 comes b/w (1,3) then it's overlapping

- so now it expans to (1, 9)
- same for (2,6) ⇒ (1, 6)
- (8,9) is not overlapping as 8 ≥ 6, so (1,6) is one overlapping as it's sorted so stop ione ans is (1,6)

→ again take one interval (8,9)

- (8,10) is part of (8, 9) ⇒ (8, 10)
- (9, 11) is part of (8,10) ⇒ [(8, 11)]

one more interval

- (15, 17) another interval start,

(16, 18) can be part of (15, 18)

so (15, 18) is one more interval.

```cpp
class Solution {
public:
    vector<vector<int>> merge(vector<vector<int>>& arr) {
      int n = arr.size();
      sort(arr.begin(), arr.end());

      vector<vector<int>> ans;
      for (int i = 0; i < n; i++) {
        int start = arr[i][0];
        int end = arr[i][1];
        if (!ans.empty() && end <= ans.back()[1]) {
          continue;
        }
        for (int j = i + 1; j < n; j++) {
          if (arr[j][0] <= end) {
            end = max(end, arr[j][1]);
          } else {
            break;
          }
        }
        ans.push_back({start, end});
      }
      return ans;
    }
};
```

TC => $N \log N + O(2N)$

SC => $O(N)$

2) Optimal Solution :

(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)

Try to do in single iteration.

(1,3) => (1,4)

[2,4]

2 < 3 (So attach)

[2,6]

2 < 4

(1,3) => (1,7)

2 < 3 (So attach) 2 < 4

=> (1,6)

(8,9) => (8,10) => (8,11) (8,10) (9,11)

8 < 9 (So attach) 9 < 10

(15,18) => (16,17)

dg it inside keep it.

T.C => O(nlogn)

C++ Autocomplete

```cpp
class Solution {
public:
    vector<vector<int>> merge(vector<vector<int>>& arr) {
      int n = arr.size();
      sort(arr.begin(), arr.end());

      vector<vector<int>> ans;
      for (int i = 0; i < n; i++) {
        if(ans.empty() || arr[i][0] > ans.back()[1]){
            ans.push_back(arr[i]);
        } else {
            ans.back()[1] = max(ans.back()[1], arr[i][1]);
        }
      }
      return ans;
    }
};
```