Length of the union of segments

Table of Contents

Given n segments on a line, each described by a pair of coordinates (a_{i1}, a_{i2}). We have to find the length of their union.

The following algorithm was proposed by Klee in 1977. It works in $O(n \log n)$ and has been proven to be the asymptotically optimal.

Solution

We store in an array x the endpoints of all the segments sorted by their values. And additionally we store whether it is a left end or a right end of a segment. Now we iterate over the array, keeping a counter c of currently opened segments. Whenever the current element is a left end, we increase this counter, and otherwise we decrease it. To compute the answer, we take the length between the last to x values $x_i - x_{i-1}$, whenever we come to a new coordinate, and there is currently at least one segment is open.

Implementation

```
int length_union(const vector<pair<int, int>>
    int n = a.size();
    vector<pair<int, bool>> x(n*2);
    for (int i = 0; i < n; i++) {
        x[i*2] = {a[i].first, false};
        x[i*2+1] = {a[i].second, true};
    }

    sort(x.begin(), x.end());

    int result = 0;
    int c = 0;
    for (int i = 0; i < n * 2; i++) {
        if (i > 0 && x[i].first > x[i-1].first
            result += x[i].first - x[i-1].firs
        if (x[i].second)
            c++;
        else
            --c;
    }
    return result;
}
```