Topic 10: CDQ, Mo.

Offline vs Online Algorithm.

Online Algorithm

Treat queries as user input

- Answer after every input

Offline Algorithm

Treat queries as object

- Answer only after reading the batch

CDQ Divide & Conquer

Example 1: Range Set & Query

(1) Set l r v : $a_l = a_{l+1} = \cdots = a_r = v$

(2) Query d : ouputs a_d.

Solution 1 : Use a segment tree

Solution 2 : Treat queries as objects

```
Set [1,3] 2
Query 2
Set [2,4] 5
Query 2
Set [2,3] 1
```

← This is an array of queries!

CDQ Divide & Conquer

Example 1: Range Set & Query

(1) Set l r v : $a_l = a_{l+1} = \cdots = a_r = v$

(2) Query d : ouputs a_d.

Set [1, 3] 2

Set [2, 4] 5

Query 2

Subproblem: What if all 'Set' happens before 'Query'?

CDQ Divide & Conquer

Example 1: Range Set & Query

(1) Set l r v : $a_l = a_{l+1} \cdots = a_r = v$

(2) Query d : ouputs a_d.

Set [1, 3] 2

Set [2, 4] 5

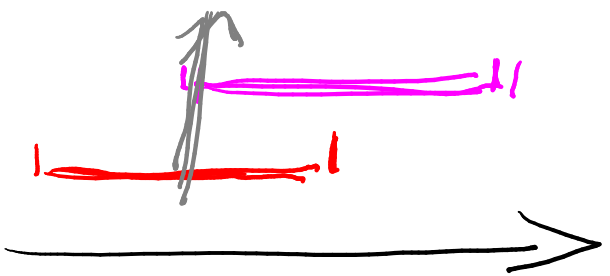
Query 2

Subproblem: What if all 'Set' happens before 'Query'?

Sol: Use a sweep line

(1) priority-queue on height

(2) Monotonic Queue on height

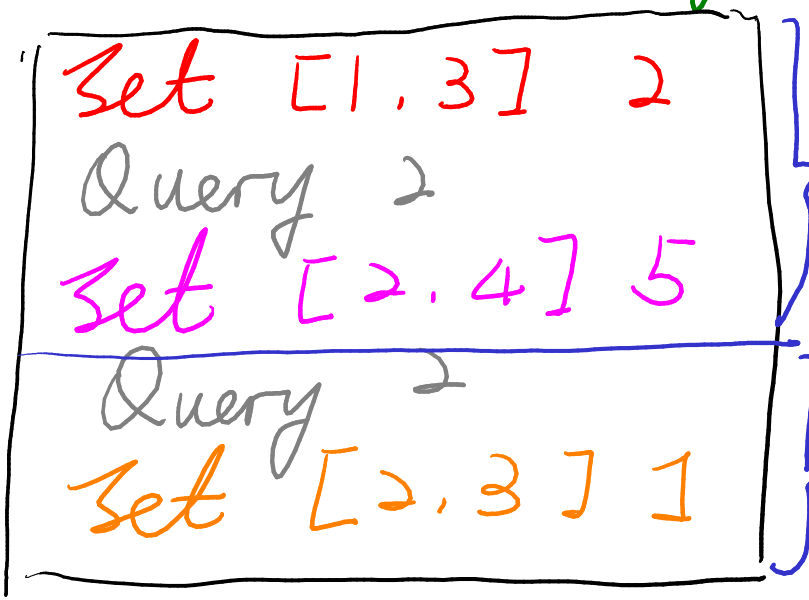


CDQ Divide & Conquer

Example 1: Range Set & Query

(1) Set l r v : $a_l = a_{l+1} = \cdots = a_r = v$

(2) Query d : ouputs a_d.



D&C on this object

```
CDQ(l, r) {
    m ← (l+r)/2
    CDQ(l, m)
    CDQ(m+1, r)
```

CDQ Divide & Conquer

Example 1: Range Set & Query

(1) Set l r v : $a_l = a_{l+1} = \cdots = a_r = v$

(2) Query d : ouputs a_d.

Set [1,3] 2
Query 2
Set [2,4] 5
Query 2
Set [2,3] 1

D&C on this object

```
CDQ(l, r) {
    m ← (l+r)/2
    CDQ(l, m)
    CDQ(m+1, r)
    For every 'Q' not answered in [m+1, r]
        Answer it using 'S' in [l, m+1]
}
```

Complexity:

$T(n) = 2T(n/2) + O(n)$

$= O(n \log n)$

CDQ Divide & Conquer

Summary: (1) Do left.

(2) Do right

(3) Analyze how left affects the right.

Example 2: Inverse Pairs

$(a_1, a_2, \cdots, a_n)$

count (i, j) s.t. $i < j \wedge a_i > a_j$

CDQ Divide & Conquer

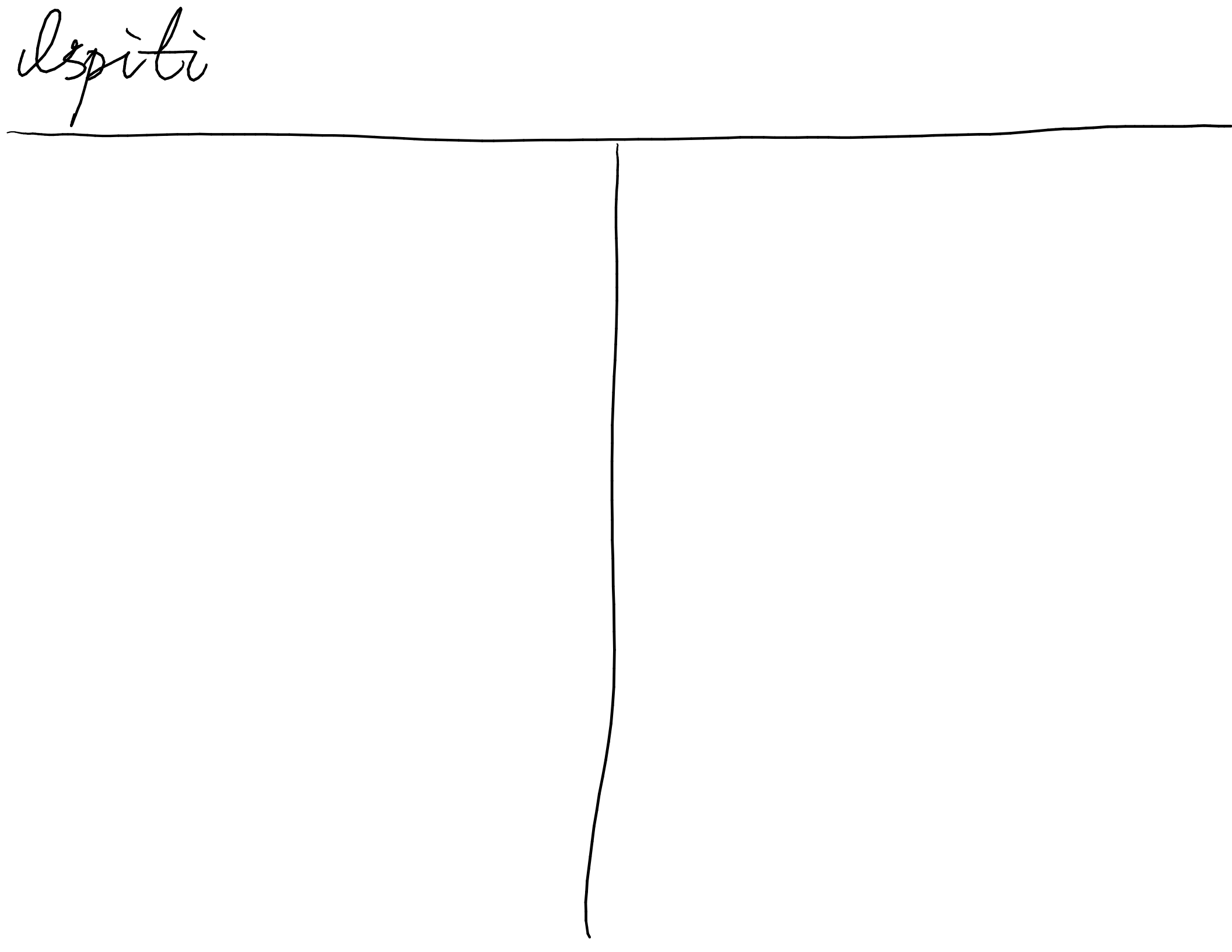
Example 2: Inverse Pairs

$(a_1, a_2, \cdots, a_n)$

count (i, j) s.t. $i < j \wedge a_i > a_j$

Sol 1: Fenwick Tree

Sol 2: Merge Sort - ish


ospiti

Mo's Algorithm

Segtree wants us to merge stuff

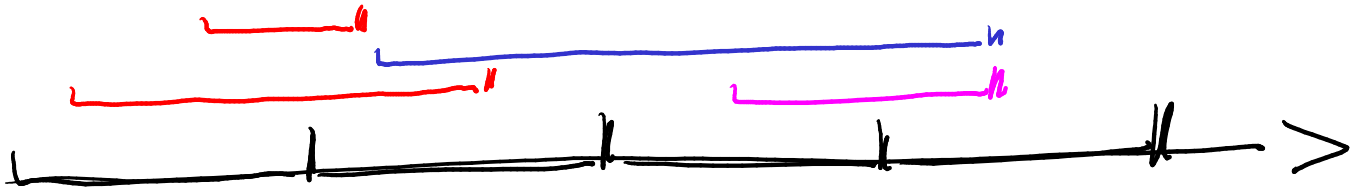
Unfortunately, some stuff refuses to be merged

Example : Cupid

Nevertheless, we observe that
it's possible to add/remove
one element to a status.

Mo's Algo gives a $O(n\sqrt{n})$
sol on query-only scenarios.

Mo's Algorithm

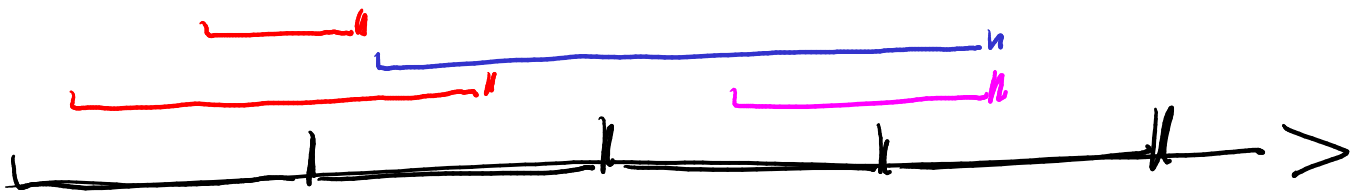


Intuition: SQRT decomp.

We put queries into diff. buckets based on its left endpoints

Within the same bucket, we sort by right endpoint.

Mo's Algorithm



Intuition: SQRT decomp.

We put queries into diff. buckets based on its left endpoints

Within the same bucket, we sort by right endpoint.

Claim: If we sort queries in this order we can brute-force in $O(n\sqrt{n})$.

Mo's Algorithm

We put queries into diff. buckets based on its left endpoints

Within the same bucket, we sort by right endpoint.

Claim: If we sort queries in this or we can brute-force in $O(n\sqrt{n})$.

Proof: Cross-bucket comp. happens $O(\sqrt{n})$ times.

In-bucket comp.:

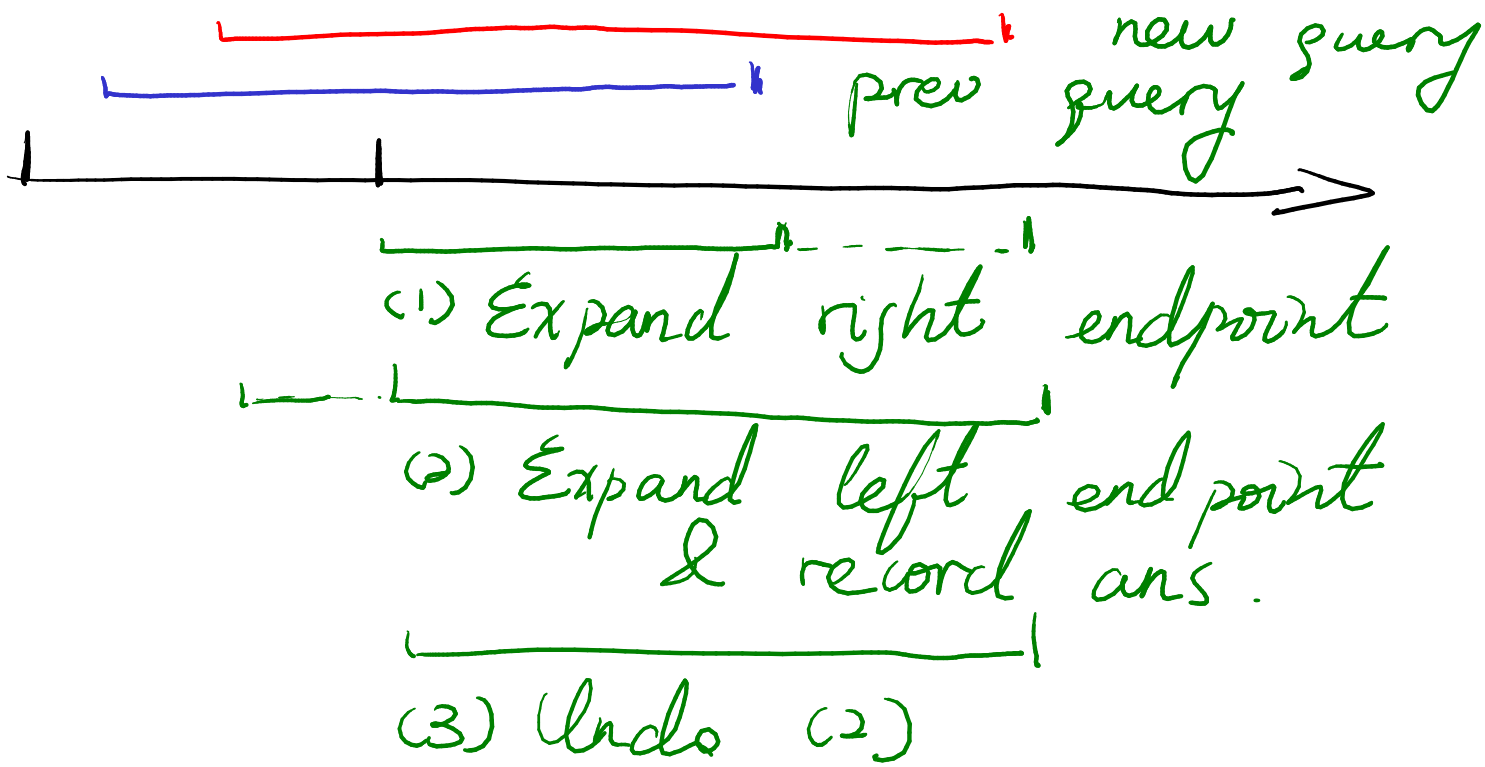
(1) Left endpoints moves $O(\sqrt{n})$ each time

(2) Right endpoints moves $O(n)$ in the buc.

Mo's Algorithm

Expansion: No-removal Mo

What if I cannot remove element?



Claim: This is still $O(n\sqrt{n})$.