Week 3 Lecture 2 Worksheet Interval Trees

Collections of Intervals

Scenario. You have a set of *time intervals* representing when TA's have office hours.

Closed time intervals: $\{x \in \mathbb{R} \mid l \leq x \leq h\} = [l,h]$.

Representation: Just use l and h.

Operations:

- *insert*(l,h): Store $[l,h]$ in the collection.
- *delete*(l,h): Delete $[l,h]$.
- *search*(l,h): Return a *stored interval* that *overlaps* with $[l,h]$.

includes overlapping boundaries.

[9, 11] [11, 13]

overlaps [11, 13] ✓

Search represents finding when a TA is available when you are.

Goal. Want $O(\lg n)$ time each.

The data structure

Q. How can we do this?

A. Use a *balanced binary search tree* (AVL, Red Black Tree, weight balanced tree ...) to store the intervals.

Q. For BST order, how do we *compare* $[l,h]$ with $[l',h']$?

- If $l < l'$, then $[l,h] < [l',h']$.
- If $l = l'$ and $h < h'$, then $[l,h] < [l',h']$.

$[2,4] < [3,6]$ b.c. $2 < 3$

Q. Is this *sufficient*?

$[2,4] < [2,5]$ because $2 = 2$, and $4 < 5$

A. for insert and delete. What about search....

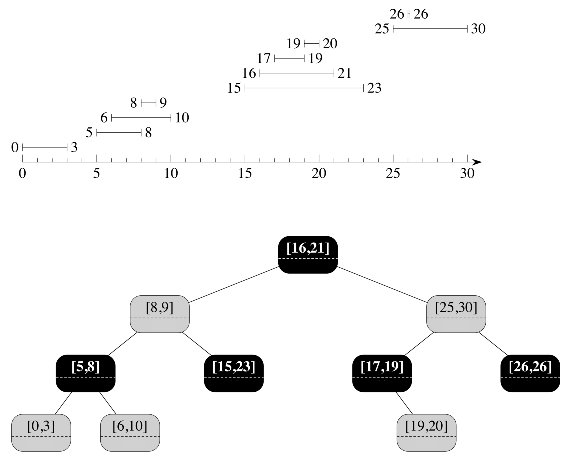
Each node x_i stores:

- l_i and h_i: interval's two ends, and the key

Example (from textbook)

(a)

(b)



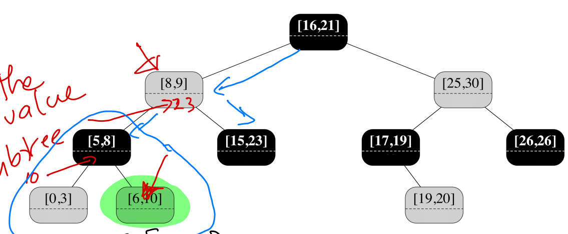
Suppose we call *search(10, 12)*. Problems? What about *search(2, 4)*.

Searching for [h, l] = [10, 12]

We need *more information*.

$10 \leq H$
l searching for.

store the largest high value in this subtree



[6,10] [10,12]

Observations about [h, l] = [10, 12]

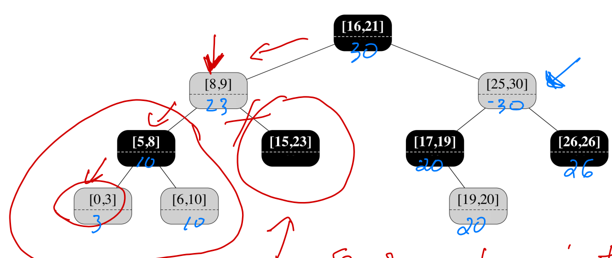
- Consider node *[8,9]*. *[10,12]* is *not* overlapping. It lies to the *right* of *[8,9]*.
- If *[10, 12]* were to overlap an interval in the left subtree of *[8, 9]*, what must be true?
- If there is some $h \geq 10$ in the left subtree - does that *guarantee* an overlap with *[10, 12]*? Why?
- if I know that in a left subtree there is a high value larger or equal to my low value (i.e 10 in [10,12]).

[=, ≥10]
[10,12]

Searching for [h, l] = [2, 4]

We need *more information*.

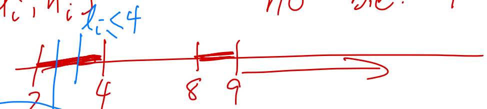
[0,3]
[2,4]



could [2,4] overlap in this subtree.

no b.e. 4<8

$[l_i, h_i]$ $l_i < 4$



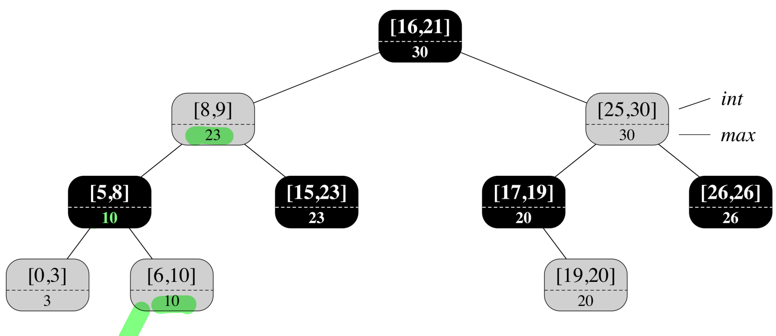
$2 \leq h_i$

Observations about [h, l] = [2, 4]

- Consider node *[8,9]*. *[2,4]* is *not* overlapping. It lies to the *left* of *[8,9]*.
- Could *[2,4]* overlap an interval $[l_i, h_i]$ in the right subtree of *[8, 9]*? Why?
- .
- If *[2, 4]* were to overlap an interval in the left subtree of *[8, 9]*, what must be true?
-
- There seems to be a *common theme*...

Augmenting the Tree

Q. So how should we *augment* our *interval tree*?



Store in node x the largest *high* interval value in the subtree rooted at x.

$[x.l, x.h]$

Searching for $[h, l]$

Let x be the root of our tree.

search(x, h, l)

```
if x = null: return null. i.e. (no overlap in the tree)

# Check if x's interval overlaps [l,h] - how?    interval we are searching for.
if x.l ≤ h and l ≤ x.h:
    return [x.l, x.h]

if h < x.l or x.h < l:
    # How do we decide whether to go left or right?
    if x.left == null:
        return search(x.right, h, l)
    elif l > x.l.max:        x.left.max
        no hope in the left, go right.
        return search(x.right, h, l)
    else: # l ≤ x.l.max
        return search(x.left, h, l)
```

$l \le x.h$ and $x.l \le h$

$[l, h]$

$h < x.l$ or $x.h < l$

↑ fail case

we want ¬fail.

x.left.max

max

looking for [6,8]

Walks just one path. $O(\lg n)$ time.

Correctness of search(x, l, h)

Each part of the **if** clause is straightforward except for the final **elif**.

In the final **elif** we go down the *left subtree*. How do we know that we haven't missed an *interval* in the *right subtree*?

Here is the pseudocode:

```
else:   # l ≤ x.l.max
  return search(x.left, l, h)
```

Claim.

If $l \le x.l.max$, then $[l,h]$ overlaps with an interval in the left subtree, or no interval in the whole tree.

Proof by Contradiction:

Suppose there is *no overlap in the left subtree*:

- Let v be a node in the *left* subtree with $v.h = x.l.max$.
- Then $l \le v.h$. Why?
- By assumption, $[v.l, v.h]$ does *not overlap* with $[l,h]$,
- Draw the possible positions of $[v.l, v.h]$ and $[l,h]$ on a line:
- So either $[l,h]$ lies to the *left* or the *right* of $[v.l, v.h]$.
- Then, we can deduce that (fill in $<$ or $>$) $h\_v.l$ or $v.h\_l$
- Which of the two inequalities is impossible? and why?
-
- Consider every node z in the right subtree of x: $h < z.l$, why?
-

Therefore, If $l \le x.l.max$, then $[l,h]$ overlaps with an interval in the *left subtree*, or *no interval* in the whole tree.