07-E

BST Application

Interval Tree

Your instinct, rather than precision stabbing, is more about just random bludgeoning.

邓俊辉
deng@tsinghua.edu.cn

Stabbing Query

❖ Given a set of intervals in general position

on the **x**-axis:

$$S = \{s_i = [x_i, x_i'] \mid 1 \le i \le n\}$$

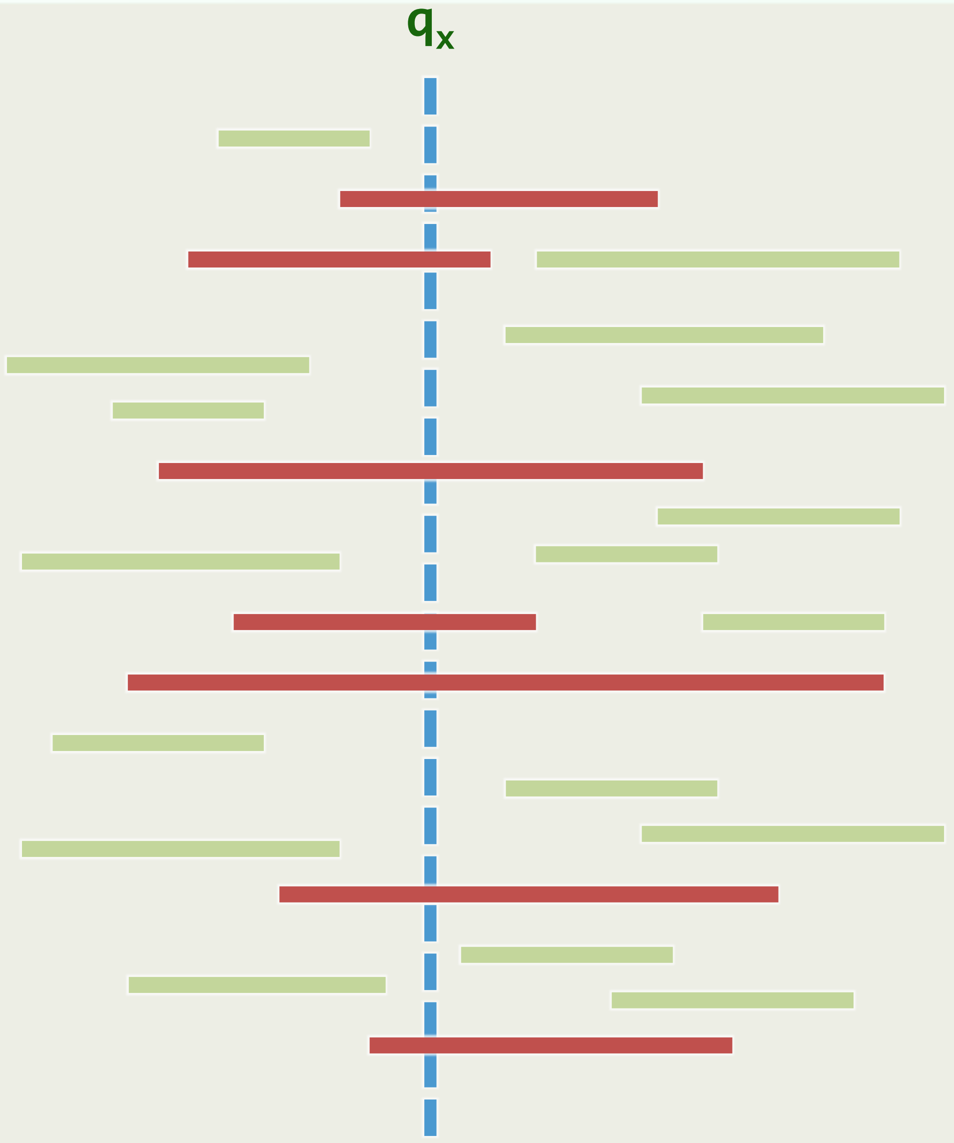
and a query point q_x

❖ Find all intervals that contain q_x

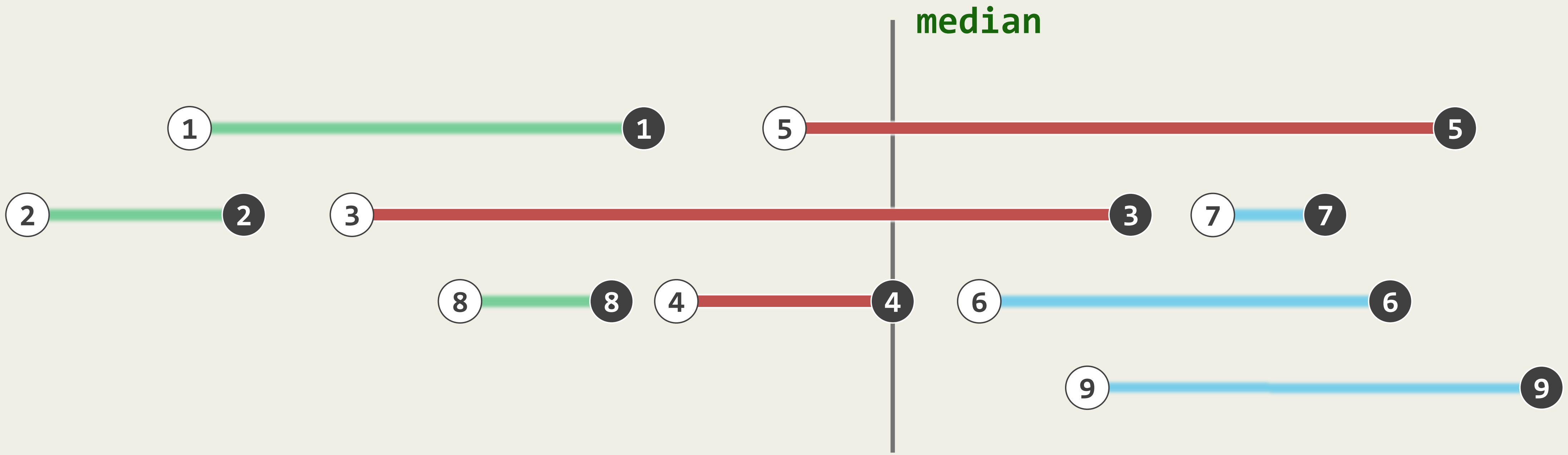
$$\{s_i = [x_i, x_i'] \mid x_i \le q_x \le x_i'\}$$

❖ To solve this query,

we will use the so-called interval tree ...



Median

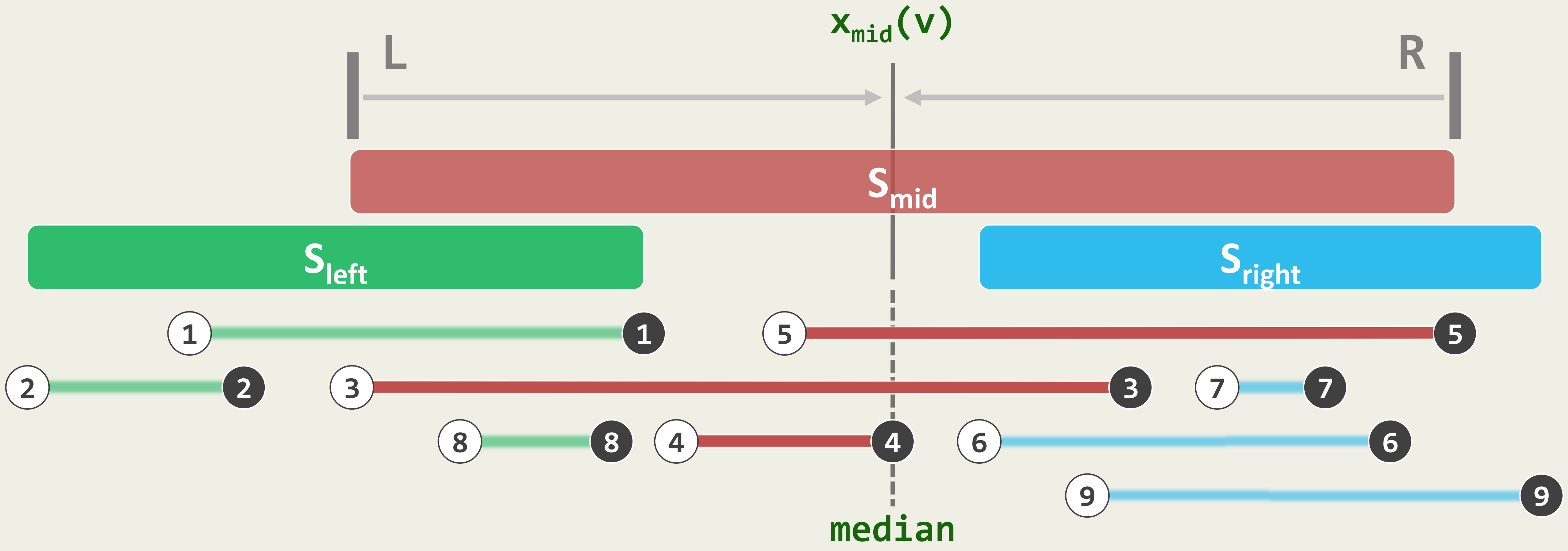


- Let $P = \partial S$ be the set of all endpoints

  (By general position assumption, $|P| = 2n$)

- Let $x_{mid} = median(P)$ be the median of P

Partitioning



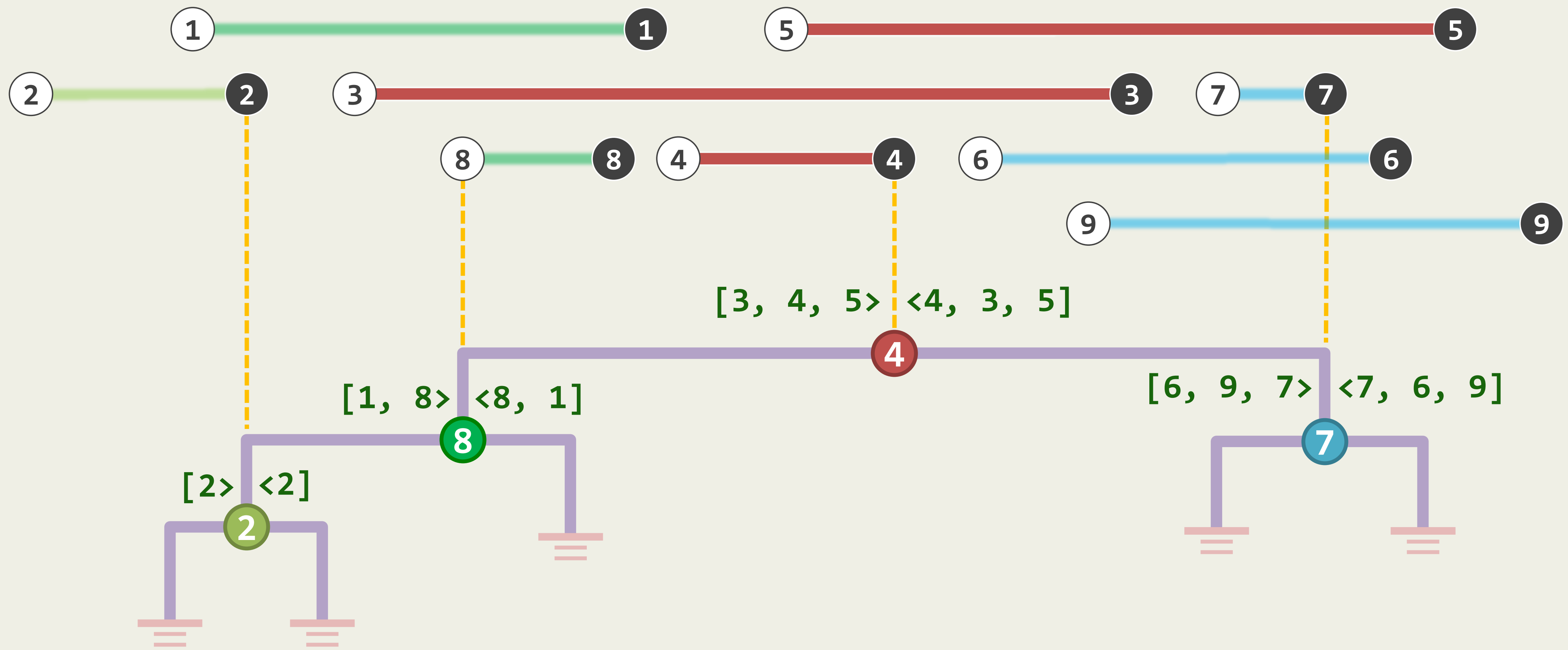
❖ All intervals can be then categorized into 3 subsets:

$S_{left} = \{ S_i \mid x_i' < x_{mid} \}$ $\quad$ $S_{mid} = \{ S_i \mid x_i \le x_{mid} \le x_i' \}$ $\quad$ $S_{right} = \{ S_i \mid x_{mid} < x_i \}$

❖ $S_{left/right}$ will be **recursively** partitioned until they are empty (leaves)

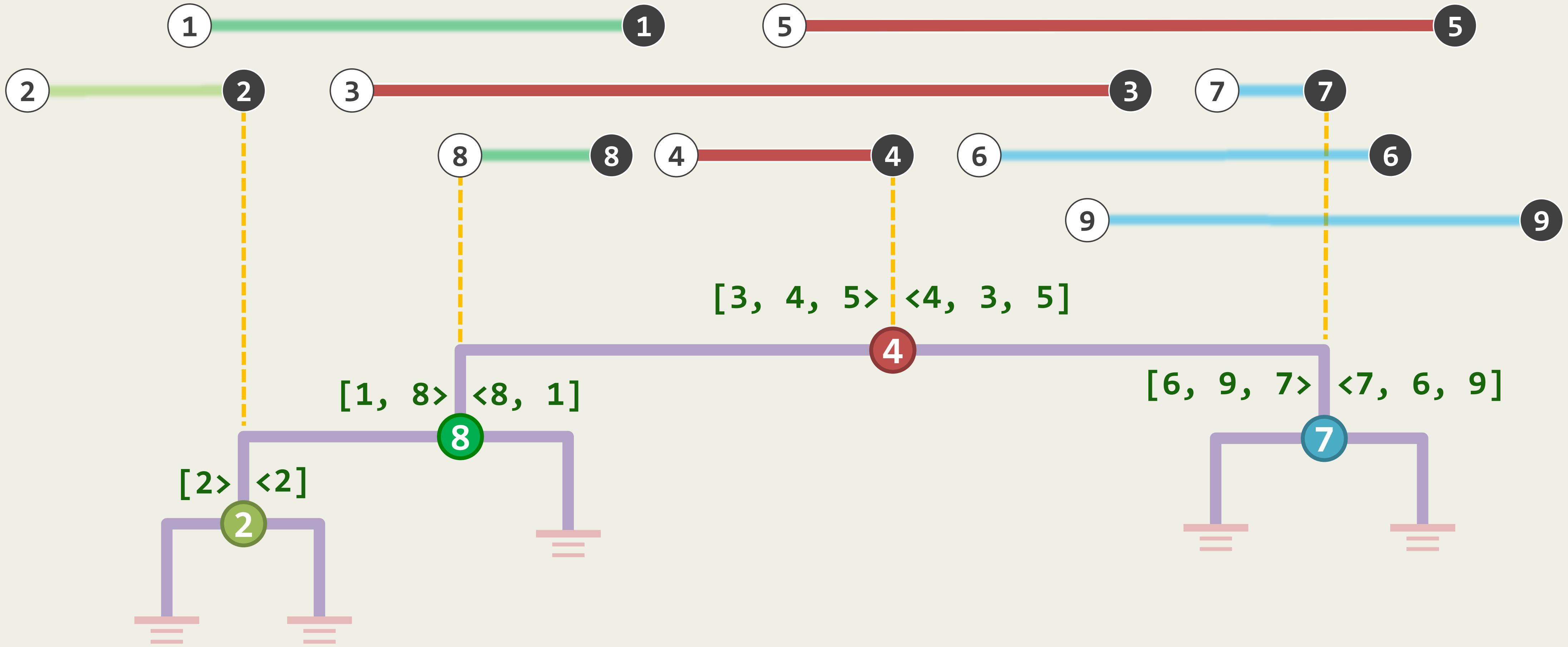
Balance & $\mathcal{O}(\text{logn})$ Depth

$\max\{\ |S_{left}|, |S_{right}|\ \} \le n/2$ **Best case:** $|S_{mid}| = n$ **Worst case:** $|S_{mid}| = 1$



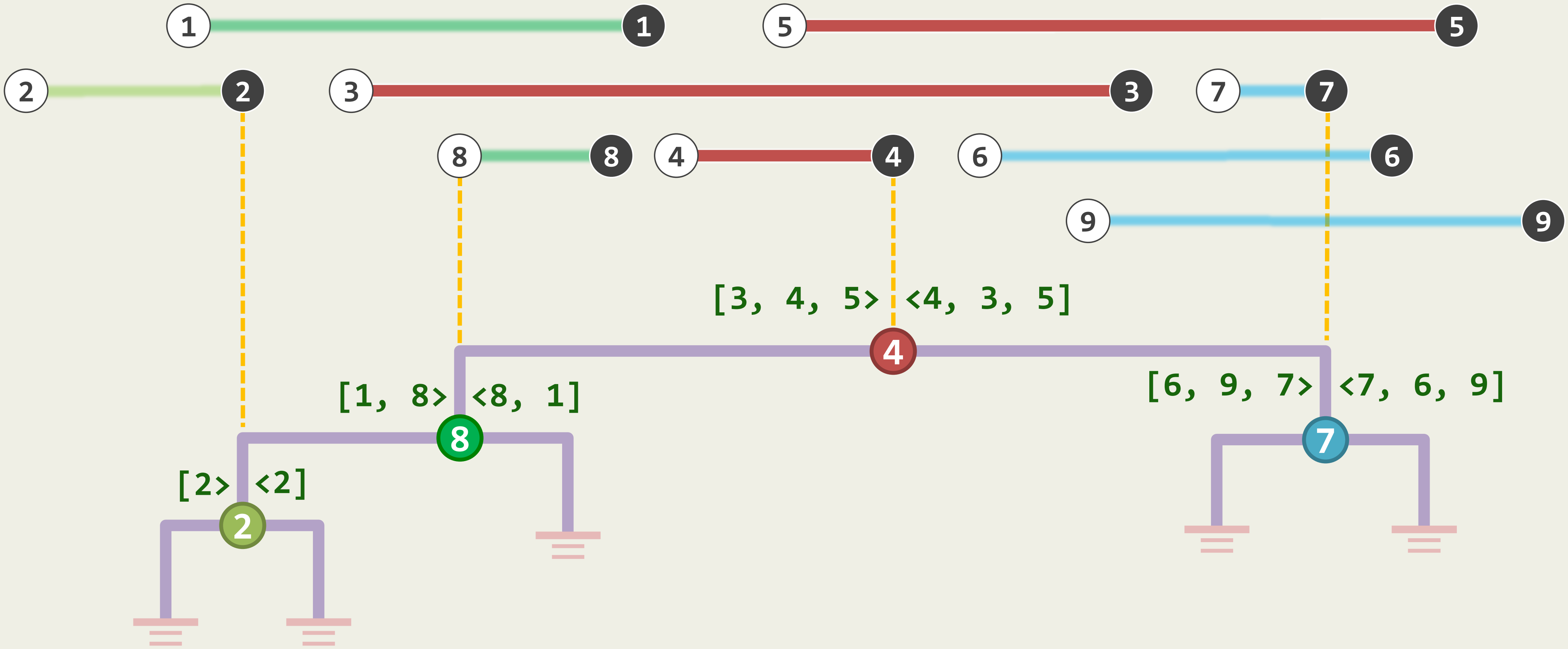
Associative Lists

- $L_{left/right}$ = all intervals of S_{mid} sorted by the **left/right** endpoints



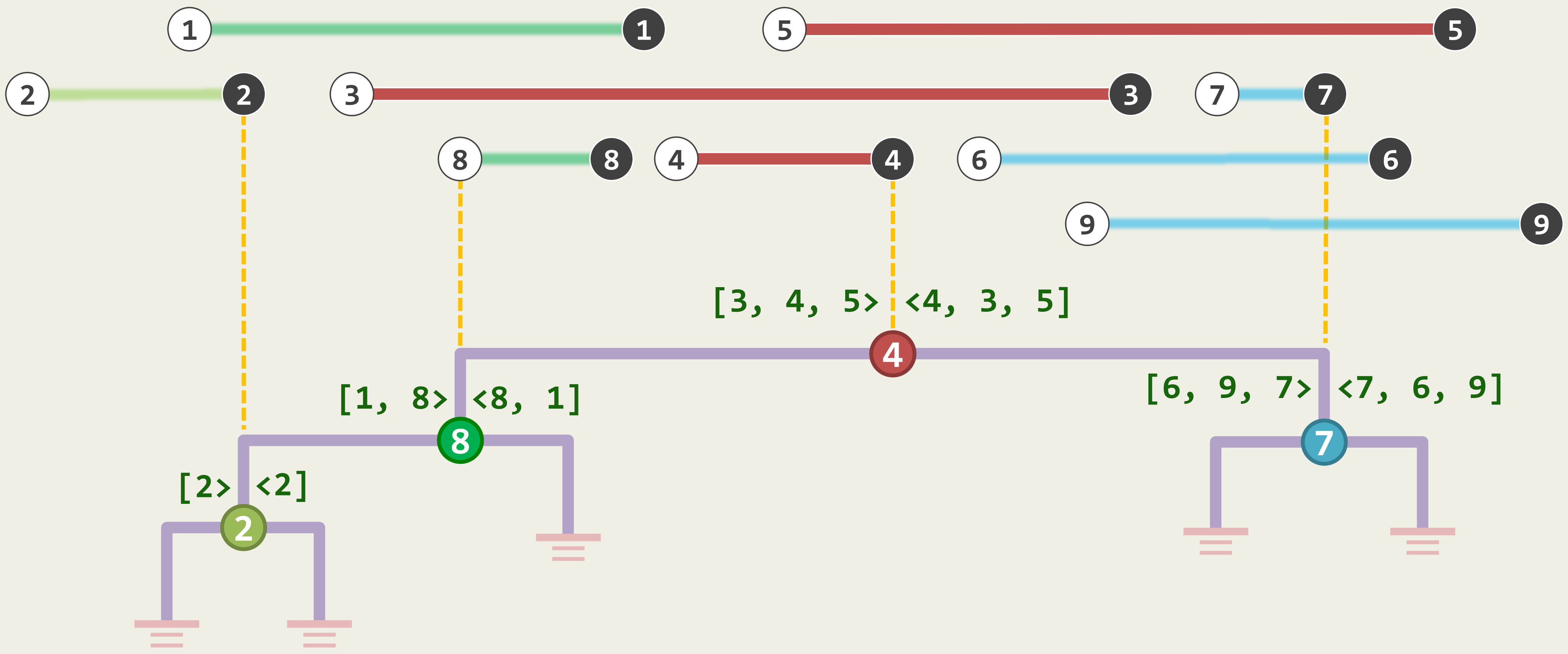
$\mathcal{O}(n)$ Size

❖ Each segment appears twice (one in each list)



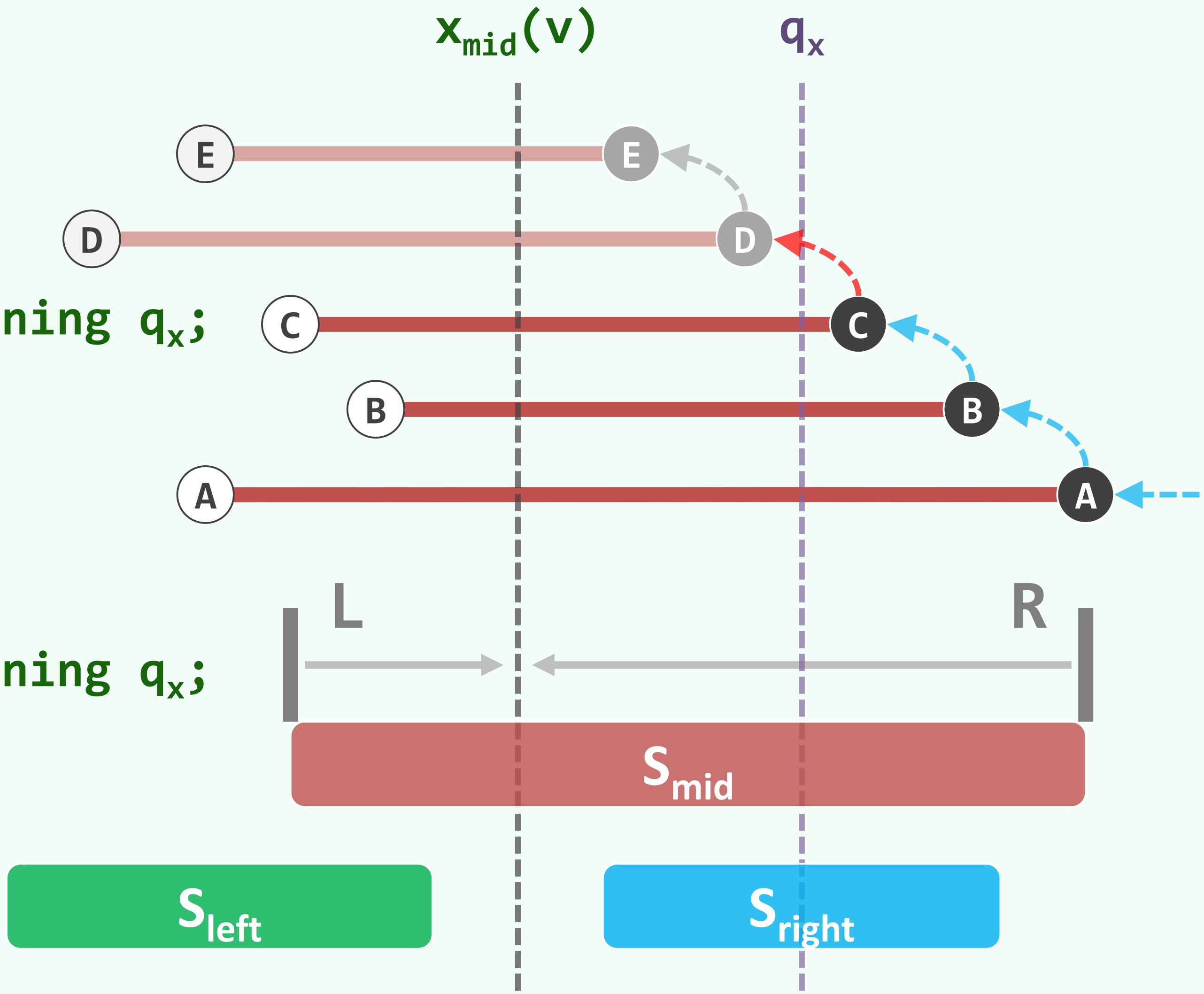
O(nlogn) Construction Time

❖ Hint: avoid repeatedly sorting



queryIntervalTree(v, q_x)

```
if ( ! v ) return;  //base

if ( qx < xmid(v) )

   report all segments of Smid(v) containing qx;

   queryIntervalTree( lc(v), qx );

else if ( xmid(v) < qx )

   report all segments of Smid(v) containing qx;

   queryIntervalTree( rc(v), qx );

else //with a probability ≈ 0

   report all segments of Smid( v ); //both rc(v) & lc(v) can be ignored
```



$\mathcal{O}$(r + logn) Query Time

❖ Each query visits $\mathcal{O}$(logn) nodes //LINEAR recursion