搜索树应用

区间树

07-A

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

邓俊辉

deng@tsinghua.edu.cn

穿刺查询/Stabbing Query

❖ 沿x轴给定一组区间：

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

对于任何点 q_x，找出所有包含它的区间：

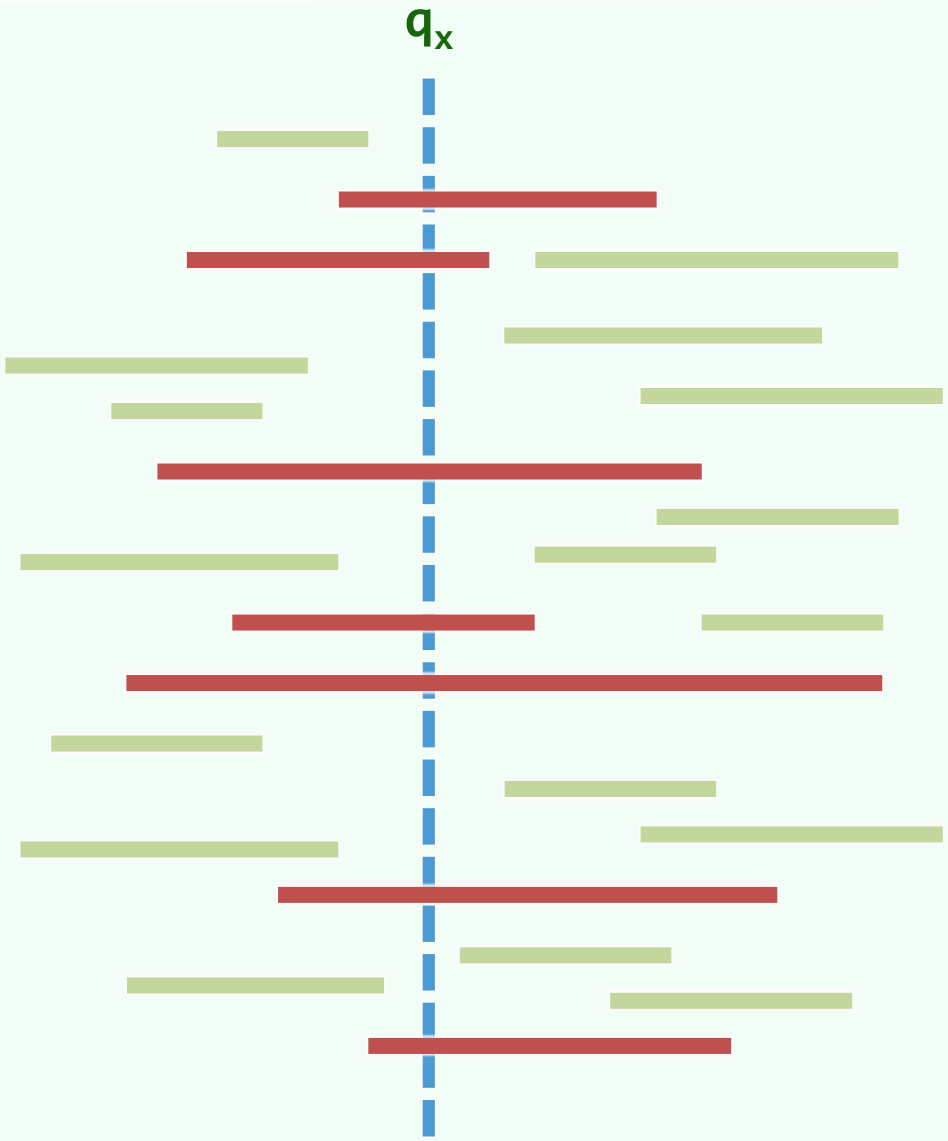
$$S(q_x) = \{s_i = [x_i, x_i^{'}] \mid x_i \le q_x \le x_i{'}\}$$

❖ 蛮力解法不值一提

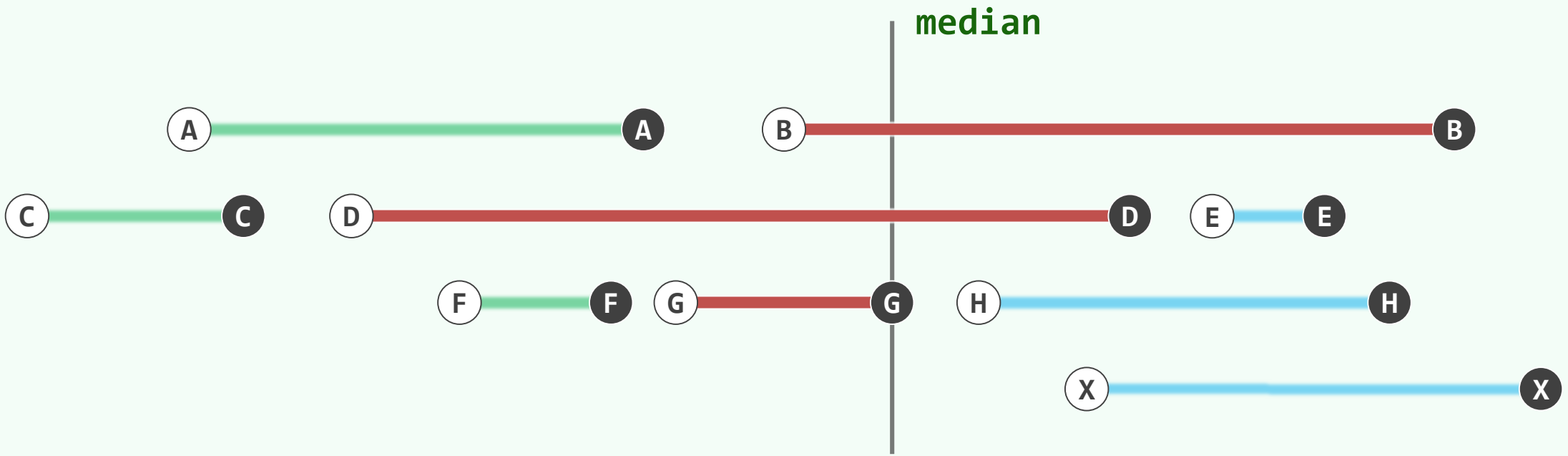
没有利用区间集相对固定的条件

❖ 将区间集预处理成一棵区间树（Interval Tree）

可以得到一个高效的在线查询算法...



中位数

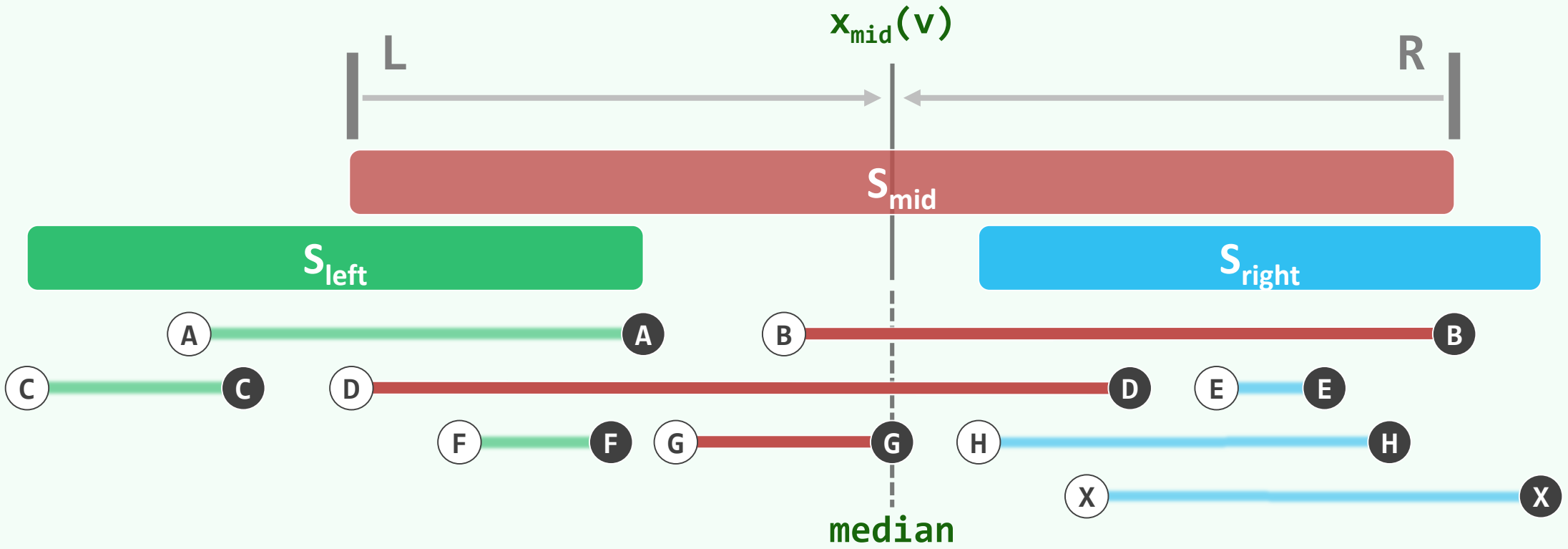


❖ 令所有区间的端点构成集合 $P = \partial S$

不失一般性，假定 $|P| = 2n$

❖ 取其中的中位数 $x_{mid} = median(P)$

分而治之



❖ 所有区间于是分为三类：

$$S_{mid} = \{ S_i \mid x_i \le x_{mid} \le x_i' \}$$

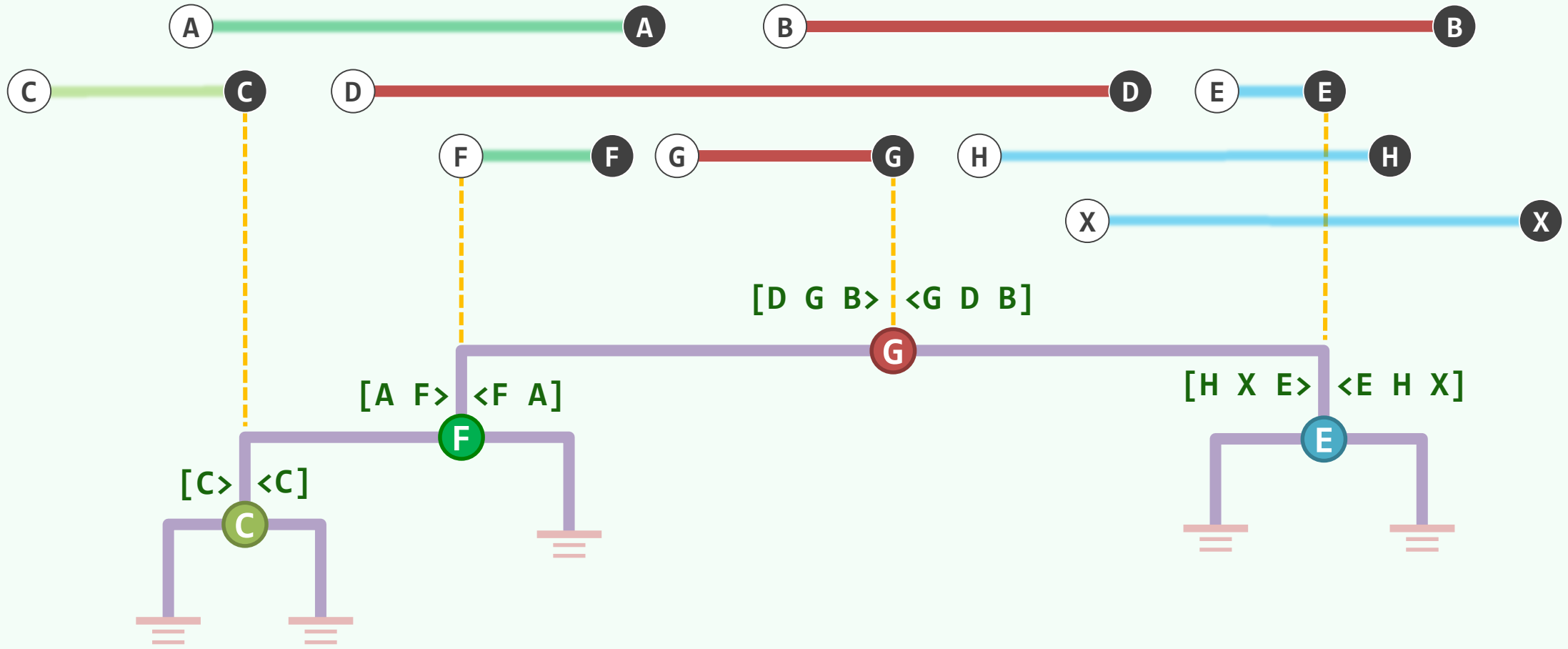
$$S_{left} = \{ S_i \mid x_i' < x_{mid} \}$$

$$S_{right} = \{ S_i \mid x_{mid} < x_i \}$$

❖ S_{left} 和 S_{right} 可递归地同样划分，直至无需划分

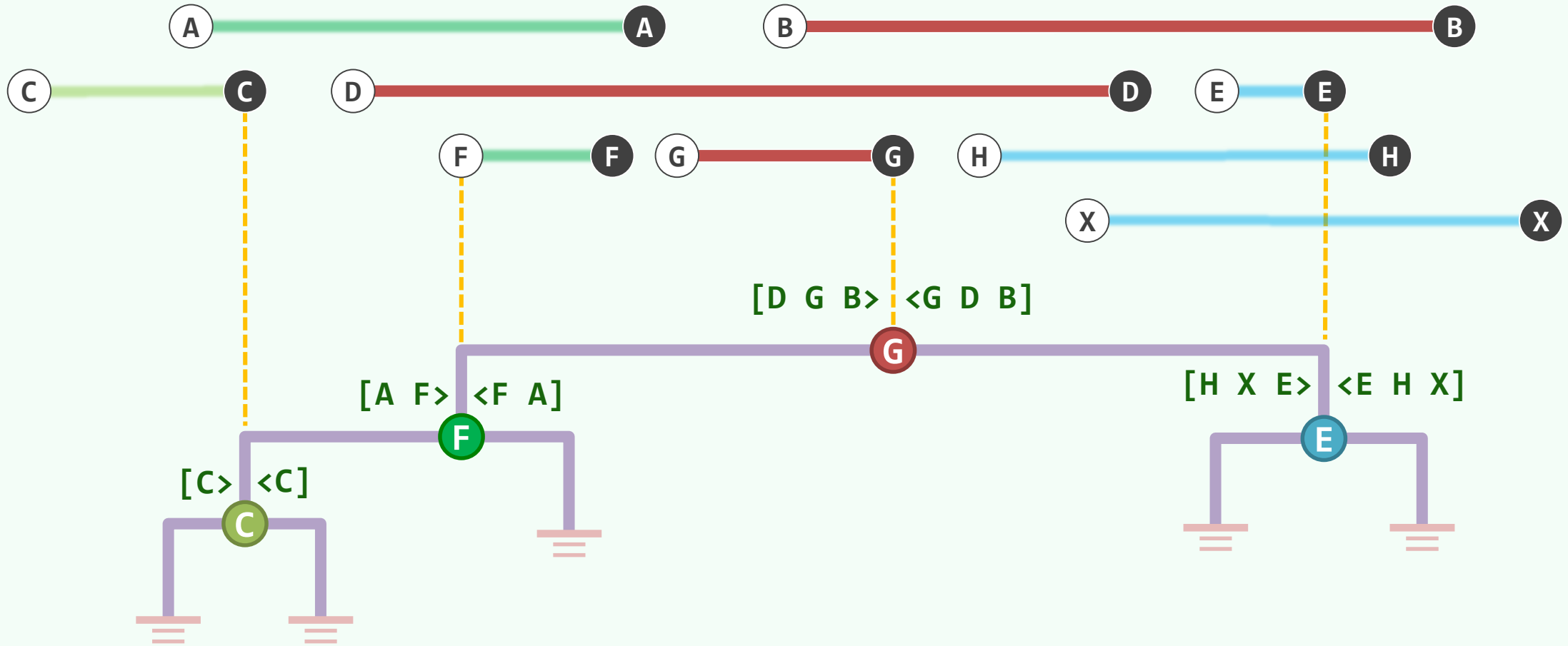
渐近平衡：O(logn)深度

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



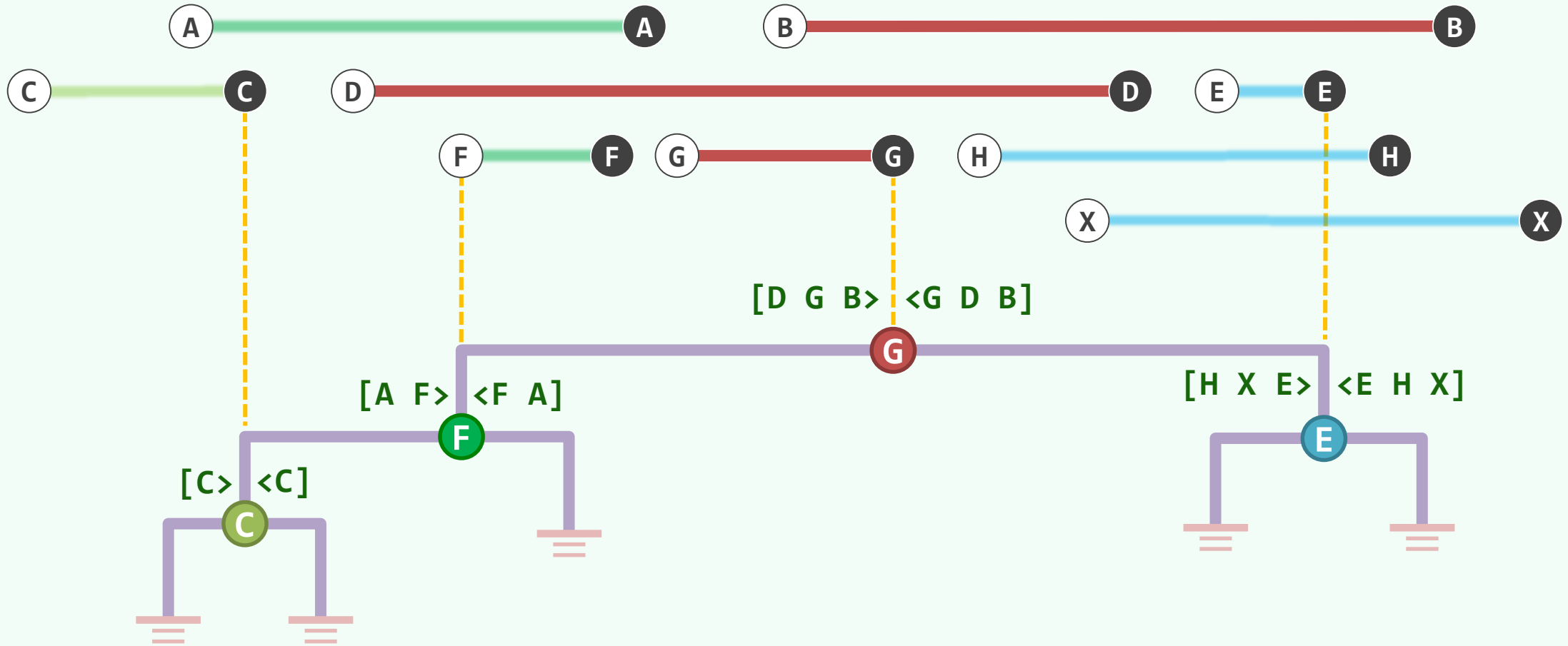
关联列表/Associative Lists

❖ S_{mid}中的区间，分别按左、右端点的次序，由外向内地排成一对列表：L-list和R-list



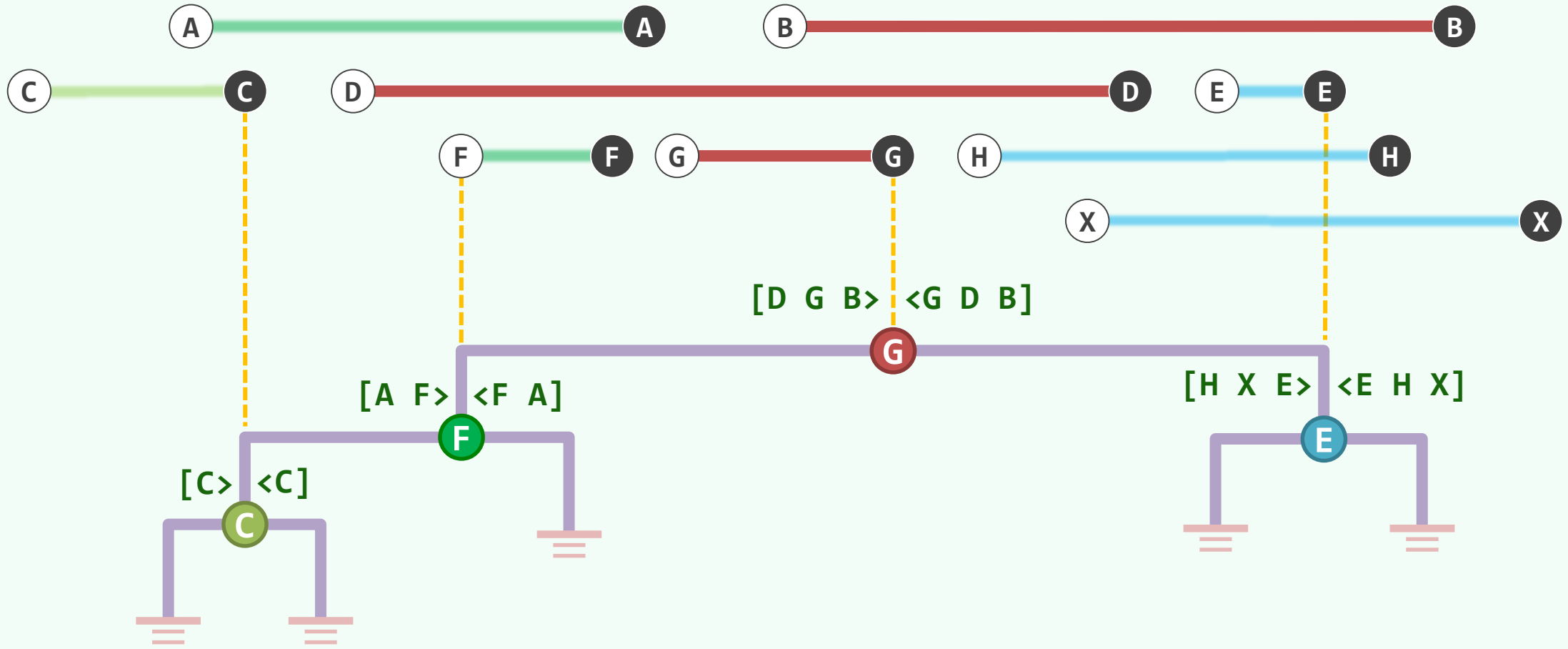
总共占用O(n)空间

❖ 该树自身占用O(n)空间；输入的每个区间各存放了两份，总共也是O(n)



构造只需O(nlogn)时间

❖ 为此，需要避免反复地对端点排序...



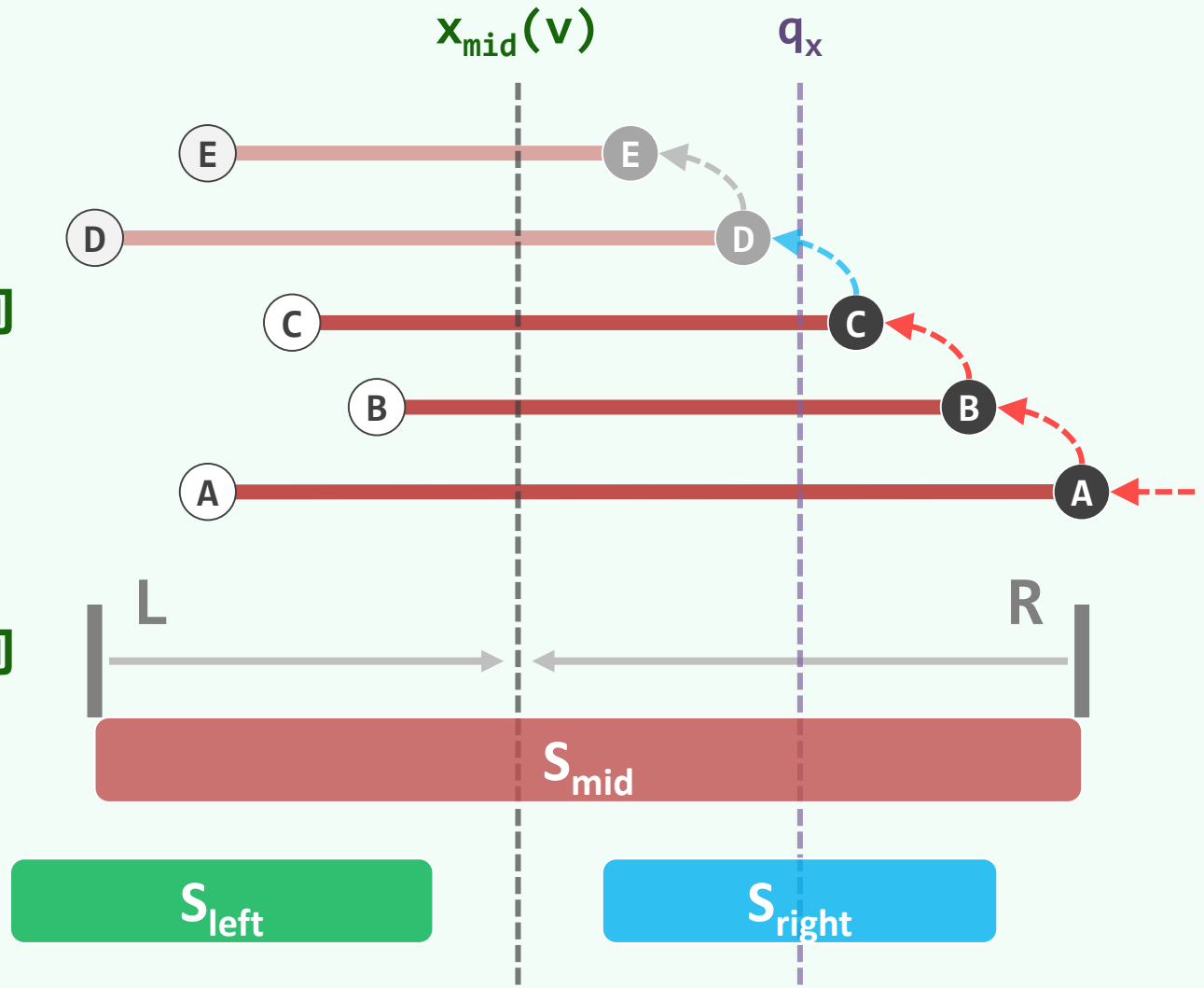
查询算法：queryIntTree(v, q_x)

```
if ( ! v ) return;   //递归基

if ( qx < xmid(v) )
   借助L-list，从Smid(v)中分拣出包含qx的区间
   queryIntTree( lc(v), qx )

else if ( xmid(v) < qx )
   借助R-list，从Smid(v)中分拣出包含qx的区间
   queryIntTree( rc(v), qx )

else   //概率无限接近于0
   直接报告Smid(v)
```



O(r + logn)查询时间

❖ 线性递归，只会访问O(logn)个节点；对L/R-List的访问：累计O(r)个元素成功，O(logn)个失败