线段树

XJTU

Information and Computational Science

mg

xjtumg.me

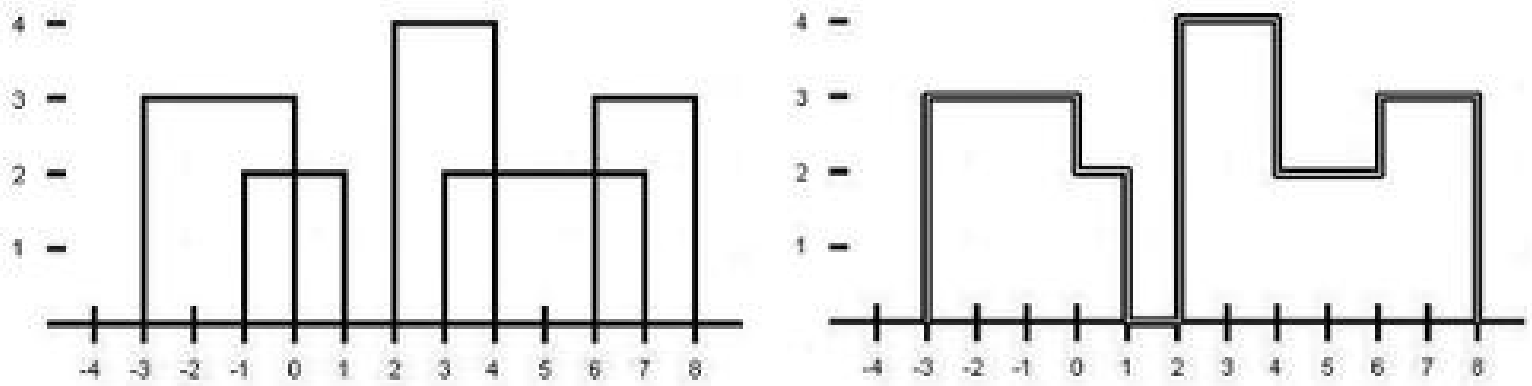
xjtumg1007@gmail.com

- Codeforces 35E Parade

- 给定n个底边均在X轴上的矩形，输出矩形覆盖后的所有转折点
- $1 \le n \le 1e5$, $1 \le hi \le 1e9$, $-1e9 \le li < ri \le 1e9$

- 离散化 + 扫描线 + 线段树

- 维护最大值即可，最大值发生改变即为转折点



- multiset可以解决问题

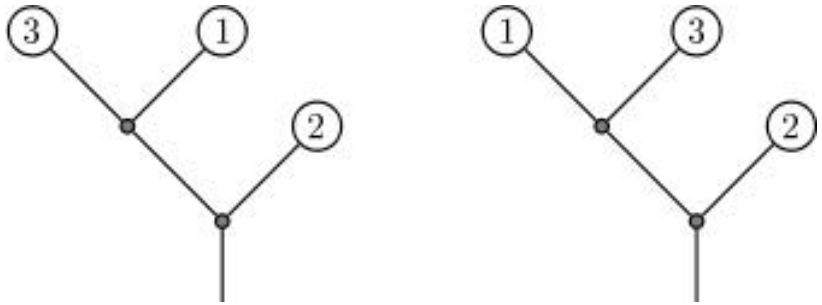
```
int n; scanf("%d", &n);
for(int i = 0; i < n; i++) {
    int y, l, r; scanf("%d%d%d", &y, &l, &r);
    event.push_back(make_pair(l, y));
    event.push_back(make_pair(r, -y));
}

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

S.insert(0);
int hmax = 0;
for(int i = 0, j; i < n * 2; i = j) {
    for(j = i; j < n * 2 && event[i].first == event[j].first; j++) {
        if(event[j].second > 0) S.insert(event[j].second);
        else S.erase(S.lower_bound(-event[j].second));
    }
    if((*S.rbegin()) != hmax) {
        ans.push_back(make_pair(event[i].first, hmax));
        ans.push_back(make_pair(event[i].first, hmax = (*S.rbegin())));
    }
}
```

- POI 18 Tree Rotations

- http://main.edu.pl/en/archive/oi/18/rot
- 给定一棵2n-1个节点的二叉树，每个叶子上有1~n的数字，保证每个数字出现且仅出现一次
- 允许任意次交换某两棵兄弟子树
- 对交换完毕的树求先序遍历，形成1~n的一个排列
- 求这个排列最小的逆序对个数
- 1 ≤ n ≤ 2 * 1e5 (1e6)



- T (不是叶子)的逆序对由三部分组成

- 1. 左子树的逆序对个数
- 2. 右子树的逆序对个数
- 3. 集合 {(a, b) | a > b, a属于左子树, b属于右子树} 的大小

- 1 & 2可以递归处理变为3
- 于是，自底向上，对每个子树的根判断交换左右子树是否会更优

- 平衡树 (Splay) 维护子树内数字的有序排列
- 自底向上对Splay进行启发式合并, 合并时计算出3
- O(n log^2(n))
- 复杂度仍然不足以AC (1e6)

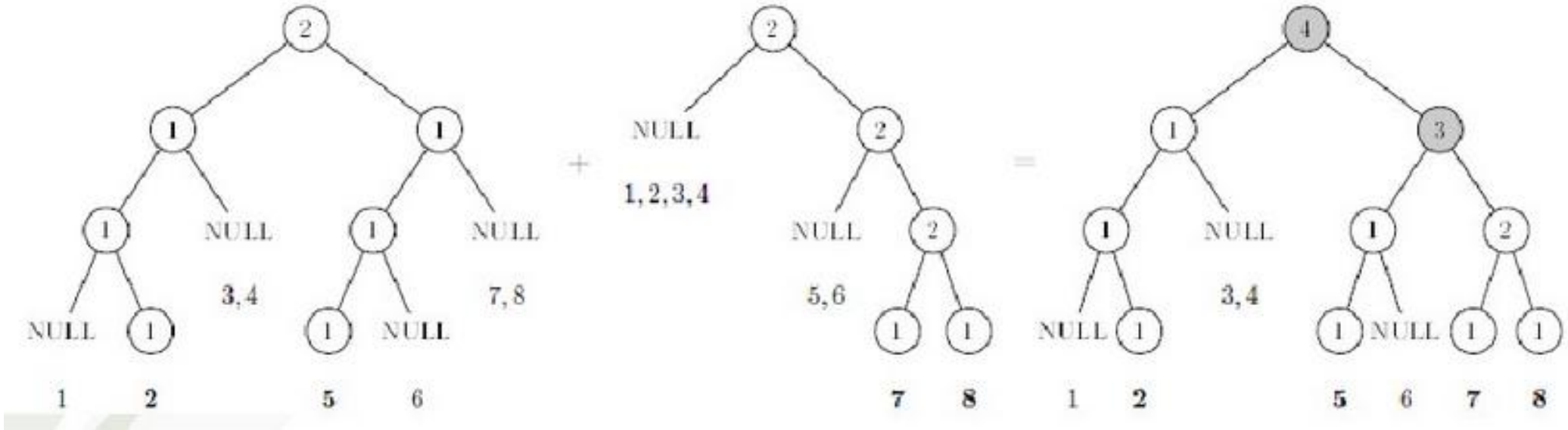
- 线段树的合并

- 前提，两个线段树的范围相同

- merge(a, b):
  - 如果a, b中有一个为空，返回另一个
  - 如果a, b都是叶子节点，merge_leaf(a, b)
  - 返回merge(a->l, b->l)与merge(a->r, b->r)连接形成的树的根

- 动态开点

- Example


2
1
1
1
NULL
1
NULL
NULL
1
3,4
7,8
NULL
1
5
6
1,2,3,4
NULL
2
NULL
2
5,6
1
1
7
8
4
1
3
1
NULL
1
2
NULL
1
2
3,4
NULL
1
1
NULL
1
1
1
2
5
6
7
8

- 关于时间复杂度
- 整个过程的代价≤将n棵只有单个节点的线段树合并成同一棵树代价
- 复杂度为O (n logn)

- 回到此题，可以用一些统计区间内数字个数 (cnt) 的线段树的合并来解决

- 在merge的过程中统计交换与不交换产生的逆序对数
- a属于T的左子树，b属于T的右子树，ans0为不交换产生的逆序对数，ans1为交换产生的逆序对数

- merge(a, b):
  - 如果a, b中有一个为空，就返回另一个
  - ans0 += cnt(a -> r) * cnt(b -> l)
  - ans1 += cnt(a -> l) * cnt(b -> r)
  - 返回merge(a->l, b->l)与merge(a->r, b->r)连接形成的树的根

- “如果a, b都是叶子节点，merge_leaf(a, b)”？

- 不存在两个相同元素，a, b不可能同为叶子节点

- 时间复杂度O(n logn)
- 空间复杂度O(n)

- MLE?
- 注意回收内存
- 先递归较大的子树

- ONTAK2010 aut

- 给一棵n个点的树以及m条额外的双向边
- q次询问，统计满足以下条件的u到v的路径数量：
  - 恰经过一条额外的边
  - 不经过树上u到v的路径上的边

- 对u到v树上的路径分类
  - ^型
  - /型
  - \型

- 只考虑^型，另两种类似
- 所求路径数量=有多少条额外的边是从以u为根的子树中的某点到以v为根的子树中的某点
- “子树”->“dfs序”
- 等价于询问两端点分别在两指定区间内的额外边的数量

- 对于单个询问(u, v)，将以u为根的子树中所有额外的边(一个端点在这个子树内即可)的另一个端点的dfs序号插入线段树

- 如何处理多个询问？

- 注意到回答(u, ?)所需的线段树是u的所有儿子线段树的并，再插入一些以u为端点的额外的边的另一个端点的dfs序号

- 离线按照dfs序从后向前处理

- 时间复杂度O((m + q) logn)

- 将一个端点在dfs序中的编号 ≤ i的额外边的另一个端点的dfs序号插入一棵线段树T[i]
- 利用可持久化线段树建立T[1~n]
- 若询问为[a, b]和[c, d]，答案即为树T[d]中[a, b]内数字的个数减去T[c - 1]中[a, b]内数字的个数
- 在线做法
- O((m + q) logn)

- BZOJ 4552 排序

- 对一个1~n的全排列，进行m次局部排序
- (0, l, r)将区间[l, r]数字升序排列
- (1, l, r)将区间[l, r]数字降序排列
- 仅一次询问，询问m次局部排序后第q位置上的数字

- $1 \le n, m, q \le 1e5$

- 二分答案
- 对于每一个猜的答案将原数列变为0, 1数列 (比猜的答案小为0，大为1)
- 线段树区间修改以及区间求和即可完成局部排序
- 时间复杂度O(m log^2(n))

- 初始时建立n棵只包含一个点的线段树 (动态开点，范围均为1~n)
- 用线段树的合并完成排序

- 对于每次局部 ([a, b]) 排序
  - 从若干线段树 (将每棵线段树代表的区间插入set，方便每次局部排序快速的找出第一棵包含[a, b]的线段树)中分裂出在[a, b]中的点，相当于分裂出线段树中连续的一段
  - 将分裂出的若干子线段树合并

- 查询相当于查询某个线段树中的第k个值
- 时间复杂度O(n log n) (?)
- 并且支持在线多个询问及局部排序

- HDU 5306 Gorgeous Sequence

- 长度为n的序列A，三种操作
- (0, l, r, t)将区间[l, r]中的每一个数字变为min(Ai, t)
- (1, l, r)输出区间[l, r]中的最大值
- (2, l, r)输出区间[l, r]的和

- $1 \le n, q \le 1e6$ (3s)

- 询问均为正常的线段树询问，关键在于如何解决修改操作

- 在线段树上暴力修改
- 剪枝：如果t不小于当前区间的最大值，则不再递归下去

- 复杂度仍然高达O(n^2 logn) 甚至不如数组模拟
- t只要递减构造就可以轻松卡掉剪枝

- 想办法继续剪枝
- 每个区间存储4个信息：最大值fi，严格次大值se，最大值的个数num，区间和sum

- 1. 如果fi ≤ t，则无需递归无需修改
- 2. 如果se < t < fi，那么可以O(1)的完成修改并且加对当前区间加标记 (传统线段树操作)
- 3. 如果x <= se...不会做了，所以干脆暴力DFS下去好了

- 这里写伪代码理清思路

- 实际上这种暴力是高效的，均摊复杂度O(n logn)

- why？

- 利用势能函数均摊分析复杂度

- 定义一个点的势能为该节点对应区间中不同值的个数
- 那么我们递归的前提条件是此次修改会使得当前节点势能降低

- 势能减少最多次数是O(n)的，且每次复杂度O(logn)
- 对于那些不能减少势能的操作我们用传统打标记的方法可以做到O(m logn)
- 故总复杂度O((m + n) logn)

- BZOJ 1568 Blue Mary开公司

- 对于二维坐标系，两种操作
- 1. 插入一条射线，以x=1为起点
- 2. 查询横坐标为X (整数)时所有射线y值的最大值

- $1 \le n, q \le 1e5$

- 新加入一条射线可能会影响若干区间
- 即最后区间[a,b]的答案可能由”折线”组成
- 这也正是本题难点

- 考虑标记永久化。即对于线段树上的每个点 (代表一个区间)除非由严格更优的射线出现，否则不删除此线段 (新线段直接下传)

- 按照斜率及区间中点处取值分四种情况讨论
- 画图说明

- 如何计算答案？(注意题目是单点询问)

- 答案即为X在线段树中对应的叶子到根路径上所有的点答案取max

- O(n logn)

- Codeforces 794F Leha and security system

- 给n个数，两种操作
- 1. 将[l, r]上所有数中数位为x的都改为y
- 2. 求[l, r]上所有数的和

- $1 \le n, q \le 1e5$
- $1 \le ai \le 1e9$

- 建立10棵线段树，分别维护数位0~9
- 关键思考如何维护lazy标记
- O((n + q) logn) (10为常数不计入复杂度)

- 2016 ACM-ICPC Asia BANGKOK Regional Contest E
- Codeforces Gym 101161 ACM Tax

- 给定一棵带权树，每次询问一个路径上所有边权的中位数

- $1 \le n \le 5 * 1e4$
- $1 \le n \le 1e5$

- 在树上解决路径问题，那么先树链剖分

- 中位数怎么找呢？

- 二分答案 (中位数)
- 现在的问题是如何查询某一区间比x小的数有多少个

- 可持久化线段树 (权值)

- 那么问题”暴力”的解决了O(n log^2(n) + m log^3(n))

- 可以找到更优美，复杂度更低的算法吗?

- Thinking...

- 考虑抛弃树链剖分，对整棵树建立可持久化线段树
- 在持久化线段树上二分答案，复杂度少了一个log
- O(n logn + m log^2(n))

- 二分多余，直接在可持久化线段树上找k大即可

- O(n logn + m logn)