海亮高级中学
HAILIANG SENIOR HIGH SCHOOL

线段树II

任飞宇

海亮教育
HAILIANG EDUCATION

图的优化

1. Legacy

在宇宙中一共有 n 个星球标号为 $1 \sim n$。Rick 现在身处于标号为 s 的星球(地球)但是他不知道 Morty 在哪里。

众所周知，Rick 有一个传送枪，他用这把枪可以制造出一个从他所在的星球通往其他星球(也包括自己所在的星球)的单行道路。但是由于他还在用免费版，因此这把枪的使用是有限制的。

默认情况下他不能用这把枪开启任何传送门。在网络上有 q 个售卖这些传送枪的使用方案。每一次你想要实施这个方案时你都可以购买它，但是每次购买后只能使用一次。每个方案的购买次数都是无限的。

网络上一共有三种方案可供购买：

- 开启一扇从星球 v 到星球 u 的传送门；
- 开启一扇从星球 v 到标号在 $[l, r]$ 区间范围内任何一个星球的传送门。(即这扇传送门可以从一个星球出发通往多个星球)
- 开启一扇从标号在 $[l, r]$ 区间范围内任何一个星球到星球 v 的传送门。(即这扇传送门可以从多个星球出发到达同一个星球)

Rick 并不知道 Morty 在哪儿，但是 Unity 将要通知他 Morty 的具体位置，并且他想要赶快找到通往所有星球的道路各一条并立刻出发。因此对于每一个星球(包括地球本身)他想要知道从地球到那个星球所需的最小钱数。

$1 \le n, q \le 10^5$，$1 \le w \le 10^9$

1. Legacy

边的数量非常多，不过我们可以用线段树优化。

建立一颗线段树用来优化第二类边，从v出发向$[l,r]$对应的线段树里$\log n$个节点连边，线段树每个节点向儿子节点连边。

同理，建立另一颗线段树用来优化第三类边，然后求最短路。

2. Journeys

一个星球上有 n 个国家和许多双向道路，国家用 $1 \sim n$ 编号。

但是道路实在太多了，不能用通常的方法表示。于是我们以如下方式表示道路：$(a, b), (c, d)$ 表示，对于任意两个国家 x, y，如果 $a \le x \le b, c \le y \le d$，那么在 x, y 之间有一条道路。

首都位于 P 号国家。你想知道 P 号国家到任意一个国家最少需要经过几条道路。保证 P 号国家能到任意一个国家。

$1 \le n \le 5 \times 10^5$，$1 \le m \le 10^5$，$1 \le a \le b \le n$，$1 \le c \le d \le n$。

2. Journeys

$(a,b),(c,d)$要连的边数很多，我们可以新建辅助节点e，f，作为中间节点优化连边。
(a,b)所有点往e连有向边，e往(c,d)所有点连有向边，
(c,d)所有点往f连有向边，f往(a,b)所有点连有向边。

使用线段树优化，然后01BFS求最短路，复杂度$n+mlogn$。

01BFS用来$O(V+E)$时间求解边权为0或1的图的最短路。虽然叫01BFS，不过把它理解为dijkstra的特殊情况更不容易误解。

类似dij，用双端队列deque维护备选点集，每次取出队首，用队首尝试relax相邻的边，如果成功则加入deque中，边权为0加前面，边权为1加后面。
其实deque相当于一个堆，只不过这里情况特殊，堆里最多有相邻的两种值。

一般BFS使用vis数组记每个点是否入队过，遇到vis=1的点就不再入队，这里不能这样做，因为一个点可能入队两次。

3. Pustynia

给定一个长度为 n 的正整数序列 a，每个数都在 1 到 10^9 范围内，告诉你其中 s 个数，并给出 m 条信息，每条信息包含三个数 l, r, k 以及接下来 k 个正整数，表示 $a_l, a_{l+1}, \ldots, a_{r-1}, a_r$ 里这 k 个数中的任意一个都比任意一个剩下的 $r-l+1-k$ 个数大（严格大于，即没有等号）。

请任意构造出一组满足条件的方案，或者判断无解。

第一行包含三个正整数 n, s, m（$1 \le s \le n \le 10^5$，$1 \le m \le 2 \times 10^5$）。接下来 s 行，每行包含两个正整数 p_i, d_i,表示已知 $a_{p_i} = d_i$，保证 p_i 递增。

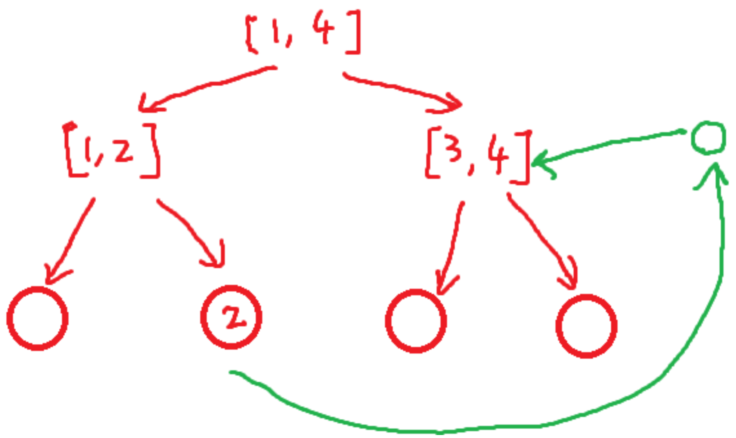
接下来 m 行,每行一开始为三个正整数 l_i, r_i, k_i）$1 \le l_i < r_i \le n$，$1 \le k_i \le r_i - l_i$），接下来 k_i 个正整数 $x_1..x_2...x_{k_i}$（$l_i \le x_1 < x_2 < ... < x_{k_i} \le r_i$），表示这 k_i 个数中的任意一个都比任意一个剩下的 $r_i - l_i + 1 - k_i$ 个数大。（$\sum k \le 3 \times 10^5$）

3. Pustynia

题目给出一些大小关系，那么可以建图然后根据拓扑序构造。

对于每条信息，新增一个辅助节点，k个点往辅助节点连边，辅助节点往最多k+1个区间连边。往区间连边可以使用线段树优化。

需要注意这里连的边不能全部表示“大于”关系，否则跟实际情况不符。



我们令k个点往辅助节点连边权1表示“大于”关系，辅助节点往线段树及线段树内部连边权0表示“大于等于”关系。如果有环则无解，如果没有环则根据拓扑序填数。

从出度为0的点开始填。如果当前点u没有固定值，则$a_u = \max(a_v + e_{u,v})$；
如果当前点u有固定值，要判断这个固定值是否合法。

4. Antennas

有n个站点等距离地排成直线，第i个站点的功率值为p_i，站点i和站点j可以交流当且仅当$|i-j| \le \min(p_i, p_j)$，在这样的两个站点之间传递信息耗时1秒。从起点a传递信息到终点b最短耗时几秒？

$1 \le a, b \le n \le 200\,000$ $1 \le p_i \le n$

4. Antennas

有n个站点等距离地排成直线，第i个站点的功率值为p_i，站点i和站点j可以交流当且仅当$|i-j| \le \min(p_i, p_j)$，在这样的两个站点之间传递信息耗时1秒。从起点a传递信息到终点b最短耗时几秒？

$1 \le a, b \le n \le 200\,000 \qquad 1 \le p_i \le n$

使用BFS求最短路径。但是边数非常多，需要优化。

在BFS的过程中，我们取出当前队首i，然后找所有出边i->j，若j没访问过就入队。
如何快速地寻找满足条件的j呢？

若$i<j$，条件可表示为$j-i \le p_i$且$j-i \le p_j$，
也就是我们要在$[i+1, i+p_i]$里找满足$j-p_j \le i$的j。线段树记$j-p_j$的最小值，树上二分就能找到满足条件的j。

若$i>j$，条件可表示为$i-j \le p_i$且$i-j \le p_j$，
也就是我们要在$[i-p_i, i-1]$里找满足$j+p_j \ge i$的j。线段树记$j+p_j$的最大值，树上二分就能找到满足条件的j。

怎么把j标记为已入队呢？把这个位置的$j-p_j$改为∞，$j+p_j$改为$-\infty$即可。

动态dp

一些动态规划的递推可以表示成矩阵的形式，且仍然满足结合律。

一些问题需要修改操作，然后得到动态规划的结果。

可以使用数据结构维护矩阵，以支持修改操作。

5. 动态最大子段和

给定一个长度为 n 的序列，你需要维护两种操作：

①查询一个区间的最大子段和；

②单点修改(即将一个位置上的数改成另一个数)

$n, q \le 10^5$

5. 动态最大子段和

令f_i表示i为结尾的区间的最大和，则 $f_i = \max(f_{i-1}, 0) + a_i$

令m_i表示$[1, i]$的最大子段和，则 $m_i = \max(m_{i-1}, f_i)$

令矩阵乘法的+运算为max， ×运算为加法，用矩阵乘法表示动态规划：

$$\begin{bmatrix} m_{i-1} & f_i & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & -\infty & -\infty \\ 0 & a_{i+1} & -\infty \\ -\infty & a_{i+1} & 0 \end{bmatrix} = \begin{bmatrix} m_i & f_{i+1} & 0 \end{bmatrix}$$

用线段树维护每个节点区间内矩阵的乘积。

6. New Year and Old Subsequence

定义一个数字串满足性质$nice$当且仅当：该串包含子序列2017，且不包含子序列2016。

定义一个数字串的函数$ugliness$为：该串至少删去几个字符，可以使得剩余串满足性质$nice$；如果该串没有满足性质$nice$的子序列，则该串的$ugliness$是-1。

给定一个长度为n的字符串t，和q次询问，每次询问用(l,r)表示。对于每次询问，回答$ugliness(t[l,r])$

6. New Year and Old Subsequence

考虑用动态dp解决，那么我们要用逐位dp来表示一个串的求解过程。

设$f_{i,0\backslash1\backslash2\backslash3\backslash4}$分别表示在$[1,i]$串里最少删几个字符，使得从左往右贪心找2017，最后找到的结果是Ø\2\20\201\2017。整个过程不允许2016的出现，我们在转移中确保这一点。

$$\begin{cases} f_{i,0} = f_{i-1,0} + [s_i = 2] \\ f_{i,1} = \min(f_{i-1,1} + [s_i = 0], f_{i-1,0}[s_i = 2]) \\ f_{i,2} = \min(f_{i-1,2} + [s_i = 1], f_{i-1,1}[s_i = 0]) \\ f_{i,3} = \min(f_{i-1,3} + [s_i = 7 \vee s_i = 6], f_{i-1,2}[s_i = 1]) \\ f_{i,4} = \min(f_{i-1,4} + [s_i = 6], f_{i-1,3}[s_i = 7]) \end{cases}$$

对于乘的[], []内条件成立则为1，否则为∞。

把转移表示为矩阵的形式。+为min，×为加法

不同的数字代表了不同的转移矩阵，用线段树维护区间内转移矩阵乘起来的结果即可。

扫描线

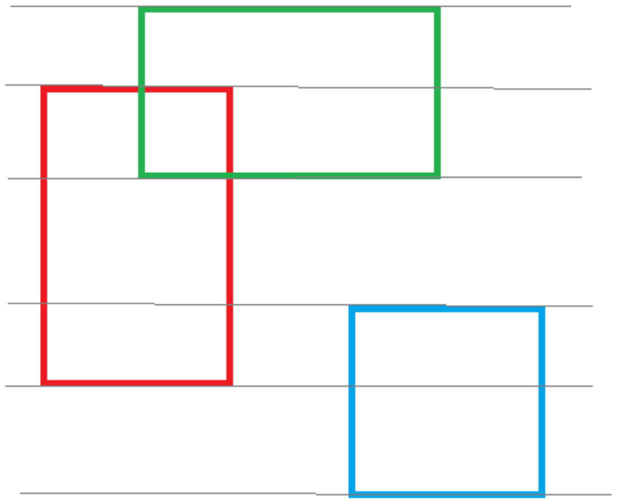
某些问题可以使用离线思想，把一个东西拆分为添加/删除两部分，扫描的过程中维护信息，被形象地称为扫描线。

7. 矩形面积并

平面上有1e5个矩形，问它们的并的面积有多少？

7. 矩形面积并

扫描线转化为一维上的问题。需要实现：插入线段/删除线段/询问有多少被覆盖的位置



被覆盖位置的数量不好维护，不如转而维护没被覆盖的位置数量，即0的个数。

然而0的个数也不好直接维护。
但是在这个问题里，任意位置的数字都≥ 0。

用线段树维护最小值和最小值的数量。

7. 矩形面积并

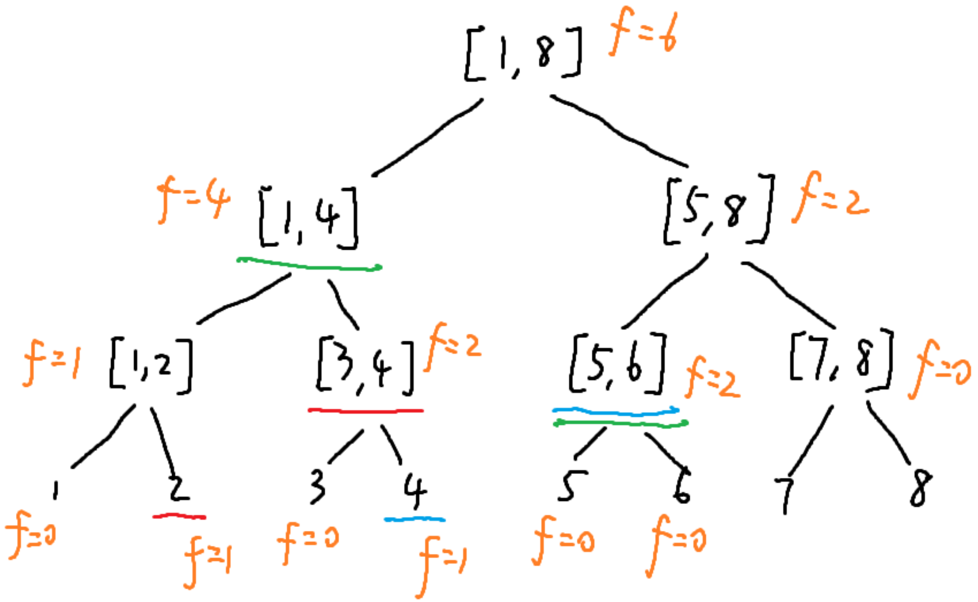
一个比较灵活的思路：

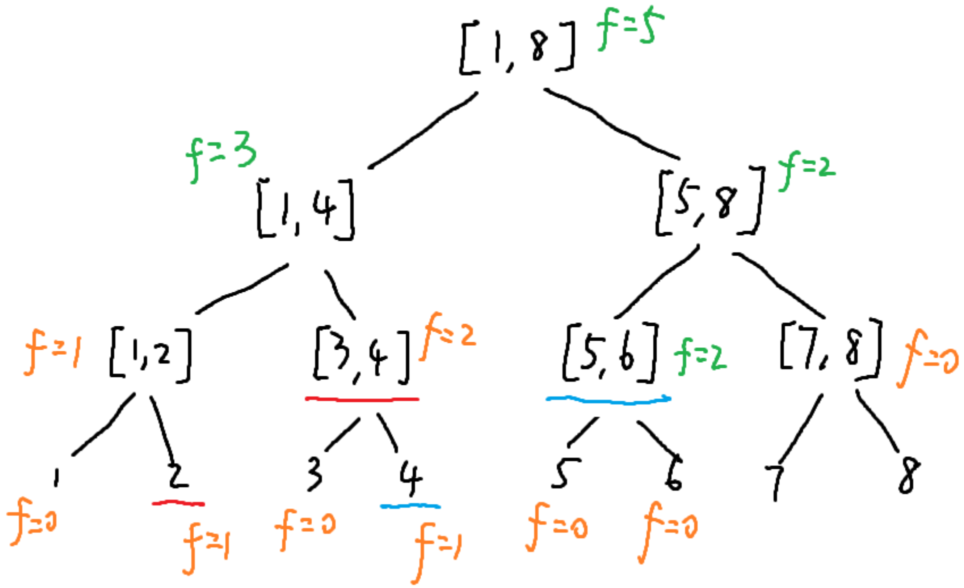
不使用延迟标记，而是把插入的线段直接挂到节点上。

那么怎么获取被覆盖位置的数量呢？
类似树形dp，从下往上对每个节点计算f_x，
表示如果只考虑以x为根的子树内挂着的线段，
x代表的范围有多少被覆盖位置。

如果当前节点上挂着线段，f值就是区间长度；
否则f值从两个儿子转移得来。

一个节点的dp值只依赖该节点及子树内挂着的标记，
所以一次修改只需要重新计算改变标记的节点，
和它们的祖先。

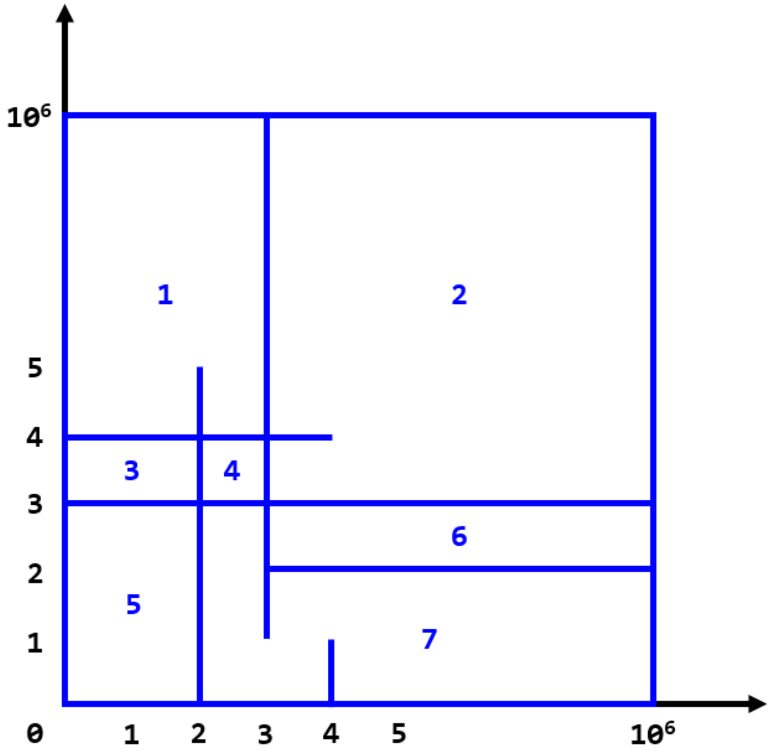




8. Divide Square

给定一个$10^6 \times 10^6$的正方形，n条横线和m条竖线穿过了它，求这些线把正方形分成了多少个部分。

注意: 每条线的两个端点之一一定在正方形的边上 $0 \le n, m \le 10^5$



8. Divide Square

我们把这个图形简化为平面图，根据平面图欧拉定理，$V - E + F = C + 1$，
根据题目条件，平面图的连通块数量$C = 1$，我们需要关注V和E。

分析共$n + m + 4$条线段，设某条线段上的交点数量为v_i，那么这条线段上的边数为$v_i - 1$，
所以$E = \sum(v_i - 1) = 2V - n - m - 4$。现在只需要计算$V$。

用扫描线计算交点数量，从左往右扫，维护哪些y坐标有横线，扫到竖线时做区间求和。

9. Arkady and Rectangles

按顺序在坐标轴上画n个颜色为1…n的矩形（数字大的颜色覆盖数字小的颜色），问最后能看到多少种颜色。

$1 \leq n \leq 100\,000$　　坐标范围$[-10^9, 10^9]$

9. Arkady and Rectangles

使用扫描线，问题变为插入一段颜色，删除一段颜色，统计这个过程中哪些颜色被看见过。

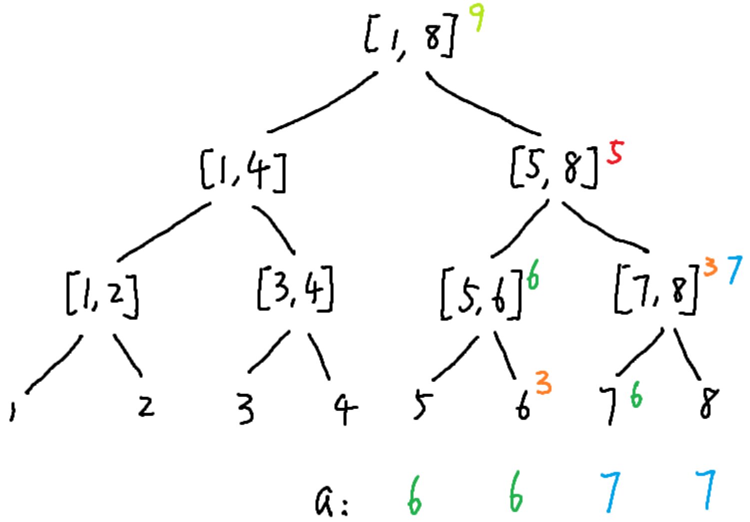
对于当前的扫描线，我们找出一个能被看到的颜色，把这个颜色打上标记；再找一个没有被打上标记的，能被看到的颜色，把它打上标记……重复若干次，直到当前扫描线上所有的颜色都被打上标记后，向前移动扫描线。

用线段树维护扫描线，把插入的颜色段挂到线段树的$\log n$个节点上，每个节点挂的颜色称为$colors$集合。用一个数组$find_i$来记录颜色i是否被找到过。

9. Arkady and Rectangles

考虑找出当前扫描线上每个位置的颜色中还没被找到过，且数字最大的。

线段树上每个节点定义一个值mx。只考虑以当前节点为根的子树内挂着的颜色段，每个位置上可能有多个颜色，但只有最大的能被看见。记这个最大的颜色为a_i。



在当前节点的a数组里排除掉已经被找到过的颜色，剩下的最大颜色记为mx。

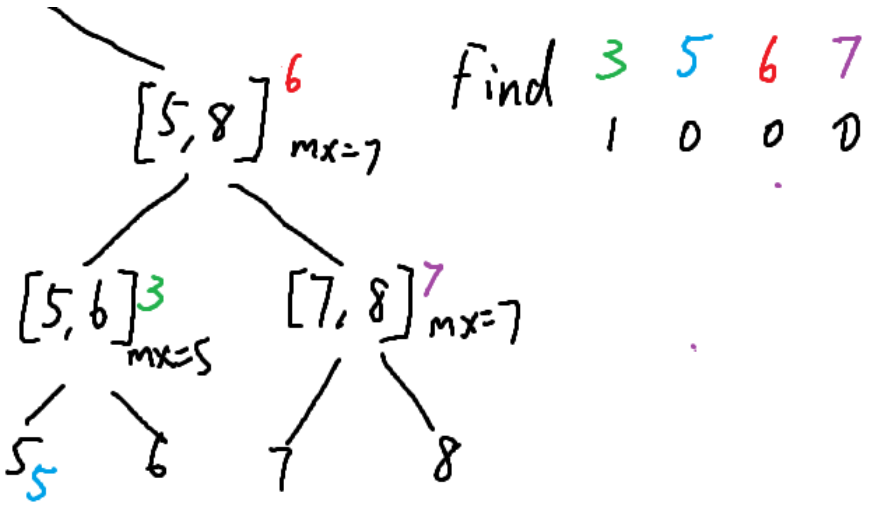
mx_{root}就是要找的颜色。

9. Arkady and Rectangles

现在分析向上转移求mx。每次转移相当于在左右子树的基础上加入当前节点的colors集合。

如果当前节点colors集合为空，那么$mx_x = \max(mx_{ls}, mx_{rs})$；

如果当前节点colors集合的最大值$c_{max} \le \max(mx_{ls}, mx_{rs})$，那么$mx_x = \max(mx_{ls}, mx_{rs})$；



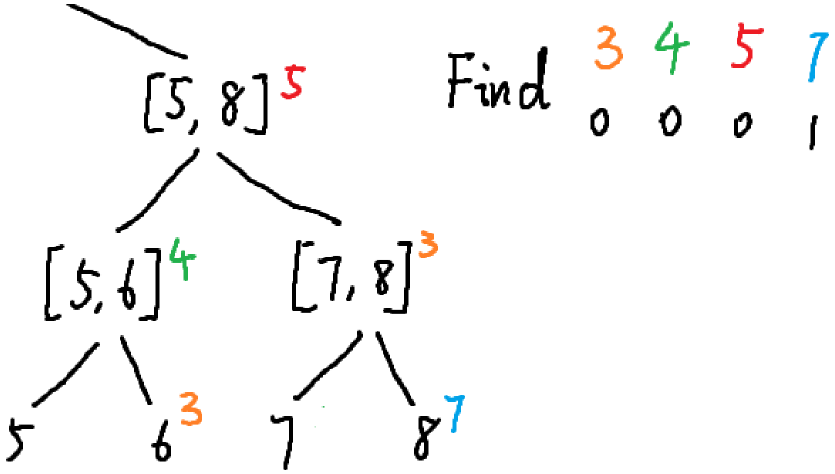
考虑节点[5,8]，只考虑两个子树的话序列为5 3 7 7，加上颜色6后变为6 6 7 7，跟6取max对$\max(mx_{ls}, mx_{rs}) = 7$来说没有影响

9. Arkady and Rectangles

接下来讨论colors集合非空，且colors集合的最大值$c_{max} > \max(mx_{ls}, mx_{rs})$，注意当前隐含了$mx_{ls} = mx_{rs} = -1$的情况，即左右子树的a数组全被找到过了。

1. $\max(mx_{ls}, mx_{rs}) \neq -1$，此时a数组里没被找过的颜色全部变为$c_{max}$，$mx$的值取决于$Find[c_{max}]$

$$mx = Find[c_{max}]?-1: c_{max}$$

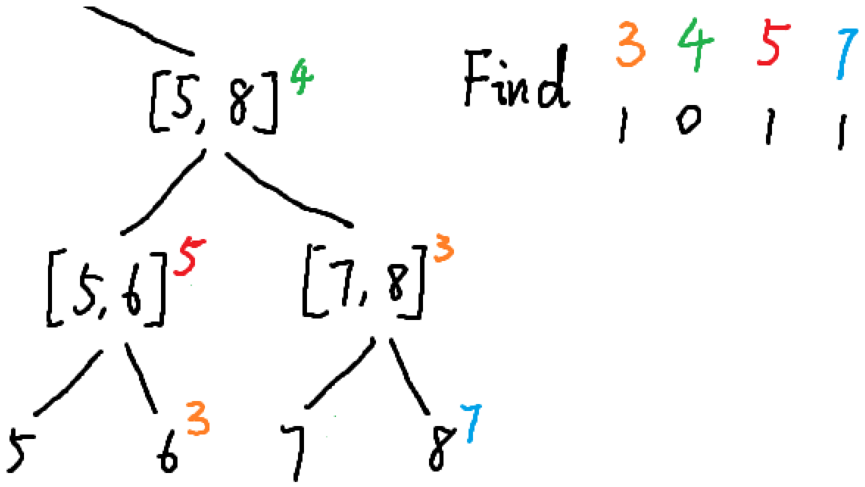


9. Arkady and Rectangles

接下来讨论colors集合非空，且colors集合的最大值$c_{max} > \max(mx_{ls}, mx_{rs})$，注意当前隐含了$mx_{ls} = mx_{rs} = -1$的情况，即左右子树的a数组全被找到过了。

2. $\max(mx_{ls}, mx_{rs}) = -1$，此时$mx$的值得看$c_{max}$能不能冒出来，如果不能冒出来，$mx = -1$，如果能，$mx$的值取决于$Find[c_{max}]$，$mx = Find[c_{max}]?-1:c_{max}$

为了判断c_{max}能不能冒出来，需要记一个值mn表示当前节点的a数组里的最小值。
如果$c_{max} > \min(mn_{ls}, mn_{rs})$就能冒出来。



9. Arkady and Rectangles

现在我们可以基于线段树节点上挂着的colors集合和Find数组自底向上求出mx_{root}，
但是还有修改，修改会改变colors集合，或者把Find数组某位改为1。
每次发生修改后，就对修改的点到根重新自底向上计算mx和mn的值。

时间复杂度为$nlog^2n$。

线段树合并

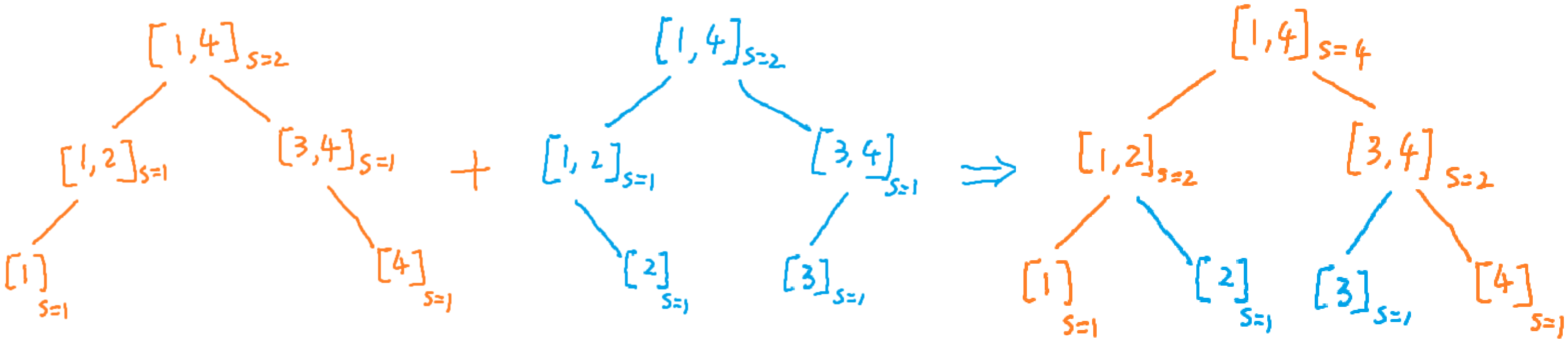
如果有两颗范围相同的动态开点线段树，我们可以进行合并操作：

$merge(a, b)$:

如果a, b中有一个不含任何元素，就返回另一个

如果a, b都是叶子，返回$merge_leaf(a, b)$

返回$merge(a -> l, b -> l)$与$merge(a -> r, b -> r)$连接成的树



每次合并的时间开销为两颗线段树的公共节点部分。

如果有n颗单链的线段树，以任意顺序合并它们的复杂度有上界$O(nlogV)$

证明：一开始有$nlogV$个节点，每次合并时间开销为两颗线段树的公共节点部分，同时也会扔掉这么多的节点。

10. 永无乡

永无乡包含 n 座岛，编号从 1 到 n ，每座岛都有自己的独一无二的重要度，按照重要度可以将这 n 座岛排名，名次用 1 到 n 来表示。某些岛之间由巨大的桥连接，通过桥可以从一个岛到达另一个岛。如果从岛 a 出发经过若干座（含 0 座）桥可以 到达岛 b ，则称岛 a 和岛 b 是连通的。

现在有两种操作：

`B x y` 表示在岛 x 与岛 y 之间修建一座新桥。

`Q x k` 表示询问当前与岛 x 连通的所有岛中第 k 重要的是哪座岛，即所有与岛 x 连通的岛中重要度排名第 k 小的岛是哪座，请你输出那个岛的编号。

线段树合并+线段树二分