

《区间切割》解题报告

宋佳兴
成都外国语学校

一 题目大意

有 n 个区间 $[L_i, R_i]$ ，然后有 m 次切割操作，每次操作由三元组 (x, l, r) 描述，对 $l \sim r$ 号区间在 x 处切割，如果一个区间被切成了两段，只保留较长的一段，长度相等时保留左边。

求出每个区间在所有操作后的左右端点，保证所有 x 构成 $1 \sim m$ 的排列。

二 数据范围

子任务	分值	$n \leq$	$m \leq$	特殊性质
1	5	5000	5000	
2	15	16000	2×10^5	AB
3	15	10^5	10^6	A
4	15	16000	2×10^5	B
5	10	10^5	10^6	B
6	20	16000	2×10^5	
7	20	10^5	10^6	

特殊性质 A：所有 x 构成 $1 \sim m$ 的随机排列。

特殊性质 B：每次操作的 $l = 1, r = n$ 。

时间限制：3s。

空间限制：512 MiB。

三 解题过程

3.1 SubTask 1

按题意模拟即可。

3.2 SubTask 2 (性质 AB)

分析性质 A，当一个区间 $[L_i, R_i]$ 被切割成两段的时候，切点等概率落在 $[L_i + 1, R_i - 1]$ ，直觉上一个区间只会经历 $O(\log m)$ 次有效切割。

结论. 一个区间期望会经历 $O(\log m)$ 次有效切割。

证明. 一次切割有 $\frac{1}{2}$ 的概率落在区间中部的 $\frac{1}{2}$ ，然后长度减少至少 $\frac{1}{4}$ 。假设 $\frac{1}{2}$ 的概率长度不变， $\frac{1}{2}$ 的概率长度乘 $\frac{3}{4}$ ，那么期望 $2\log_{\frac{4}{3}} m$ 次长度变为 1，因此期望切割次数的上界是 $2\log_{\frac{4}{3}} m$ 。□

每次进行切割操作 $(x, 1, n)$ 时，只需要找到所有的区间 i 满足 $L_i < x < R_i$ ，然后直接模拟复杂度就有保证。一种做法是使用线段树，对于每个区间 $[L_i, R_i]$ ，把 $[L_i + 1, R_i - 1]$ 拆分成 $O(\log m)$ 个线段树节点，把 i 挂到这些节点上，进行操作时遍历 x 对应叶子节点到根的路径，找到需要修改的区间，修改它们后再重新挂到线段树上，复杂度 $O(n \log^2 m + m \log m)$ 。

更简单的做法是离线。记 $time_i$ 表示位置 i 被切割的时间，那么区间 $[L_i, R_i]$ 第一次被切割的时间为 $\min time_{L_i+1 \sim R_i-1}$ ，假设第一次切割后区间变成了 $[L'_i, R'_i]$ ，那么第二次切割的时间为 $\min time_{L'_i+1 \sim R'_i-1}$ ，以此类推。使用 ST 表解决区间最小值查询，复杂度为 $O((n + m) \log m)$ 。

3.3 SubTask 3 (性质 A)

延续 SubTask 2 的离线做法，考虑一个区间 $[L_i, R_i]$ ，它第一次被切割的时间不再是 $\min time_{L_i+1 \sim R_i-1}$ ，因为有些操作 (x, l, r) 不包含 i ，那么

$time_x$ 就应当视为无穷大。我们以 $i = 1 \sim n$ 的顺序求出区间 i 的答案, $time$ 数组会在这个过程中变化。初始时 $time_{1 \sim m} = \infty$, 对于操作 (x, l, r) , 在处理区间 l 之前把 $time_x$ 修改为操作本来的时间, 在处理区间 r 之后把 $time_x$ 修改为 ∞ 。用线段树维护 $time$ 数组来支持单点修改和区间最小值查询, 复杂度 $O(n \log^2 m + m \log m)$ 。

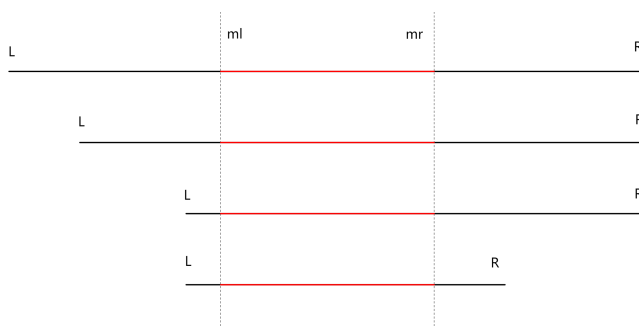
3.4 SubTask 4,5 (性质 B)

没有了性质 A, 会面临一个很大的困难: 一个区间会经历 $O(n)$ 次有效切割, 快速处理一系列操作不是一件容易的事。考虑一个区间会经历 $O(n)$ 次有效切割的原因是什么, 就是切点可以很靠近区间的端点, 让区间长度减少得很缓慢, 反过来, 只要切点位于区间中间的 80%, 就能让长度按比例减少, 从而只会发生 $O(\log m)$ 次。我们的想法是找出并模拟切点不靠近两端的情况, 剩下的情况就是切左边改左端点, 切右边改右端点, 左右端点的变化就独立了。

结论. 把一个区间 $[L, R]$ 等分成三段 $[L, ml], [ml, mr], [mr, R]$, 如果有一系列操作 $x_1, x_2, \dots, x_k$ 都不在 $[ml, mr]$ 中, 那么单个 x_i 的效果就是:

$$\begin{cases} L = \max(L, x_i) & (x_i < ml) \\ R = \min(R, x_i) & (x_i > mr) \end{cases}$$

注意这里 ml, mr 不随 L, R 变化, 如下图所示。



证明. 观察图片可知: 当 $L < x_i < ml$ 时 $[L, x_i]$ 更短, 当 $mr < x_i < R$ 时 $[x_i, R]$ 更短。

□

有了这个结论，思路就很明显了，对于区间 $[L_i, R_i]$ ，设 $[ml, mr]$ 是它中间的 $\frac{1}{3}$ ，等到切点落在 $[ml, mr]$ 的时候再处理这个区间，这样就只需要处理 $O(\log m)$ 次。

延续 SubTask 2 的离线做法，记 $time_i$ 表示位置 i 被切割的时间，对区间 $[L_i, R_i]$ ，计算 t 表示 $\min time_{ml \sim mr}$ 的位置，先求出进行 t 操作前一刻的左右端点 L'_i, R'_i 。

记 tl 表示 t 左边第一个比 $time_t$ 小的位置（不存在为 0），记 tr 表示 t 右边第一个比 $time_t$ 小的位置（不存在为 $m+1$ ），那么 $L'_i = \max(L_i, tl)$, $R'_i = \min(R_i, tr)$ ，然后直接模拟 t 操作。使用 ST 表解决区间最小值查询，单调栈预处理每个 t 的 tl 和 tr ，复杂度 $O((n+m)\log m)$ ，可以通过 SubTask 4,5。

延续 SubTask 2 的线段树做法，可以做到 $O(n \log^2 m + m \log m)$ ，也可以通过 SubTask 4,5。

3.5 正解

把 SubTask 3 和 5 中的离线做法结合起来就可以得到正解，具体地，需要维护一个 $time$ 数组，支持单点修改，查询区间最小值，以及 t 左边和右边第一个比 $time_t$ 小的位置，用线段树即可，每种操作都是 $O(\log m)$ ，因此总复杂度为 $O(n \log^2 m + m \log m)$ 。

如果求“ t 左边和右边第一个比 $time_t$ 小的位置”时使用二分 + 区间最小值，那么总复杂度为 $O(n \log^3 m + m \log m)$ ，只能通过 SubTask 1,2,4,6。