线段树优化dp与二维数点

前言

- 在现在的很多的竞赛中，动态规划的优化越来越重要了。
- 如果不加优化的动态规划几乎是过不了几个点的。
- 对动态规划上，时间复杂的度的优化相比于空间复杂度的优化尤为重要。

DP常用优化

- DP的复杂度 = 状态数 * 状态的转移数
- 优化的方向
- 一.减少状态的表示
- 二.减少状态的转移数
- 1.根据本身转移状态的单调性优化
- 2.用树结构优化转移的查找复杂度
- 等等

例题1：BZOJ1672. Cleaning Shifts 清理牛棚

- 题目大意：
- 给出一个区间[M, E], 给你N条带权线段，使用这些线段覆盖原区间的最小花费
- 概括：带权最小区间覆盖问题

例题1： BZOJ1672. Cleaning Shifts 清理牛棚

- 思路:
- 朴素的DP是 O(n^2) 的，状态和转移方程如下:

$$dp[cow[i].e] = min_{cow[i].b-1 \leq j}(dp[j] + cow[i].s)$$

- 化简后:

$$dp[cow[i].e] = min_{cow[i].b-1 \leq j \leq cow[i].e-1}(dp[j]) + cow[i].s$$

- 基本转换为区间最值问题

例题2：「ABC339E」Smooth Subsequence

- 题目大意：

给定长度为序列 a 和阈值 D。

我们称一个序列 s 是「光滑序列」，当且仅当其相邻两项之差的绝对值都不超过 D。换言之，即 $\forall i \in [1,n) \cap \mathbb{Z}, |s_i - s_{i+1}| \leq D$。

你要做的是求序列 a 的最长「光滑子序列」。

注意： 子序列可能不连续。

- $1 \leq N \leq 5 \times 10^5$
- $0 \leq D \leq 5 \times 10^5$
- $1 \leq A_i \leq 5 \times 10^5$

例题3： CF115E Linear Kingdom Races

- 题目大意:
- 有 n ($n \leq 10^5$)个物品，编号为1~ n。选取第i个物品需要 c_i 的代价。另外有 m($m \leq 10^5$)个条件，
- 表示若 $l_i \sim r_i$ 间的物品全部选择，可以获得 p[i] 的收益。求最大收益。

例题3： CF115E Linear Kingdom Races

- 分析：

- 用f[i]表示考虑完前i个物品是否选取能获得的最大收益。

- 朴素的转移方程： $f[i] = \max\{f[j] - cost(j+1,i) + profit(j+1,i)\}$

例题3： CF115E Linear Kingdom Races

- 分析:
- 朴素的转移方程： f[i] = max{f[j] − cost(j+1,i) + profit(j+1,i)}
- 令 g[j] = f[j] − cofit(j+1,i) + profit(j+1,i)
- 第一部分：f[j] − cost(j+1,i) 把这部分看成一个整体，那么 随着 i 的增加每个 g[i] 花费都会增加 -c[i]

例题3： CF115E Linear Kingdom Races

- 分析:
- 朴素的转移方程： f[i] = max{f[j] − cost(j+1,i) + profit(j+1,i)}
- 第二部分：protift(j+1,i) 对于这部分 如果存在一个区间[l, r] 全选择了 会增加 p， 那么当 i == r 时，
- 我们可以给i属于 [1, l - 1] 这部分的g[i] 每个增加 p
- 以上两部分均可以在线段树上维护。

例题4： CF474E Pillars

- 题目大意：
- 有n个柱子，每个柱子有一个高度hi，每个柱子可以跳到它后面高度与它相差大于d的柱子（$\left|h_i - h_j\right| \geq d$）
- 求最多可以跳多少个柱子

例题4： CF474E Pillars

- 分析：

$$f_i = max_{|h_i - h_j| \geq d}(f_j + 1)$$

- 尝试优化，建立权值线段树
- 每次查询到一个数，f[i] 等于1 ~ h[i]−d 和 h[i] + d 中最大数的最大值 + 1
- 线段树维护最大值和最大值是哪个点

例题5 [ZJOI2010] base 基站选址

- 题目大意：

有 N 个村庄坐落在一条直线上，第 i $(i > 1)$ 个村庄距离第 1 个村庄的距离为 D_i。需要在这些村庄中建立不超过 K 个通讯基站，在第 i 个村庄建立基站的费用为 C_i。如果在距离第 i 个村庄不超过 S_i 的范围内建立了一个通讯基站，那么该村庄就被覆盖了。如果第 i 个村庄没有被覆盖，则需要向他们补偿，费用为 W_i。现在的问题是，选择基站的位置，使得总费用最小。

[ZJOI2010] base 基站选址

算法一分析

状态定义

我们定义 $f[i][j]$ 表示在前 i 个村庄中已经建立了 j 个基站，并且第 j 个基站是在第 i 个村庄建立的最小费用。

状态转移方程

状态转移方程如下：

$$f[i][j] = \min_{k<i}(f[k][j-1] + \text{cost}(k,i))$$

其中，$\text{cost}(k,i)$ 表示在第 k 和第 i 村庄之间建立基站的补贴费用。

例题5 [ZJOI2010] base 基站选址

预处理 $\mathrm{cost}(k,i)$

由于 $\mathrm{cost}(k,i)$ 需要在 k 和 i 之间计算补偿费用，可以通过预处理来提高效率。预处理的复杂度为 $O(n^3)$。

算法复杂度

1. **预处理** $\mathrm{cost}(k,i)$:
   复杂度为 $O(n^3)$。

2. **动态规划转移**:
   对于每个 i 和 j:

$$f[i][j] = \min_{k<i}(f[k][j-1] + \mathrm{cost}(k,i))$$

   这里的复杂度为 $O(n^2 \cdot k)$。

综上，整个算法的复杂度为 $O(n^2 \cdot k + n^3)$，这能过 40% 的数据。

例题5 [ZJOI2010] base 基站选址

算法二分析

算法二通过优化决策过程中的状态转移来降低时间复杂度。具体的优化方法包括利用线段树来处理区间的最小值和加和操作。以下是详细的分析：

状态定义

- $f[i][j]$：在前 i 个村庄中建立了 j 个基站，并且第 j 个基站是在第 i 个村庄建立的最小费用。

优化方法

1. **线段树优化**：
    - 在决策过程中，状态转移的最小值计算可以利用线段树来优化，以达到更高的效率。
2. **补贴费用的计算**：
    - 通过补贴费用的递增性质，可以简化计算过程。

例题5 [ZJOI2010] base 基站选址

补贴费用处理：

- **补贴的递增性**：由于 D 数组是递增的，如果一个村庄 p 需要补贴，那么所有在 p 之后的村庄 h 也会需要补贴。
- **计算补贴费用**：对于每个决策 k 和村庄 h，如果第 k 个基站在村庄 h 处建立，补贴费用可以通过 $\text{cost}(k, i)$ 计算，并更新 $f1$ 数组。

例题6： IOI2009 旅行商人

- 题目大意：在一条线段上有若干点，点上有一些数值，一个人从坐标s出发，在线段上走，
- 每顺走一米和你走一米都有不同的费用，这些点只有在某一天可以取，同一天可以去的点无先后顺序。
- 求此人可以得到的最大利益。
- 题目链接：https://www.luogu.com.cn/problem/P5902

例题6： IOI2009 旅行商人

- 如果本题只考虑在一天中只有一个展览，那么显然按天数划分阶段，就是一个很经典的DP了，
- f[j] = max(f[i] - cost（i，j）+ 获利)，这样的的时间复杂度是O（n^2）。
- 显然需要继续优化

例题6： IOI2009 旅行商人

- 对于状态一维已经很优秀了。
- 考虑用线段树优化转移。
- 因为发现在转移时 如果只考虑顺流情况,转移的比较是

$$f_k - (L_i - L_k) * d \geq f_j - (L_i - L_j) * d$$

$$f_k + L_k * d \geq f_j + L_j * d$$

- 逆流情况也是同理，建一棵顺流线段树和逆流线段树。

例题6： IOI2009 旅行商人

- 增加两个数组FU[i],和FD[i] 表示同一天中的展销会按L的远近从近到远排序，
- FU表示从上一个逆流而上走过来的最大值，
- FD表示从上一个逆流而下走过来的最大值。
- 初始时将起点s插进去和每次枚举会s点后的最大获利求出最优解就可以。
- 因为每个展销会只枚举一次，解转移的复杂度为logn所以最后的复杂度也为O（nlogn）

例题7 CF1485F Copy or Prefix Sum

- 题目大意：

给定一个 b 数组，一个 a 是合法的指对于每一个 i 都有 $b_i = a_i$ 或 $b_i = \sum_{j=1}^{i} a_j$ 。问合法的 a 有多少个。

答案对 $10^9 + 7$ 取模。

$1 \le t \le 10^4, 1 \le \sum n \le 2 \times 10^5, -10^9 \le b_i \le 10^9$

例题7 CF1485F Copy or Prefix Sum

- 题目大意：
- 最朴素的DP：f[i][j] 表示前i个数和为j的情况下 有多少个不同的方案
- 考虑转移：
- f[i][j + b[i]] += f[i - 1][j]
- f[i][b[i]] += sum{ dp[i - 1][k] } -inf <= k <= inf
- 注意 j = 0 的时候会重复转移

例题7 CF1485F Copy or Prefix Sum

- f[i][j + b[i]] += f[i - 1][j]
- f[i][b[i]] += sum{ dp[i - 1][k] } -inf <= k <= inf （此处使用map转移）
- 进一步思考：
- 第一个转移相当于平移DP数组，第二个转移相当于在某一特定位置加上dp数组的总和，因此我们可以用一个变量 tot记录数组整体偏移量，再用一个变量sum记录当前DP数组的和, 再结合一下map
- 时间复杂度O(nlogn)

例题8 P9400 「DBOI」Round 1 三班不一般

- 题目大意：

有一个序列 S，第 i 个位置可以取 $[l_i, r_i]$ 中的数，但是不能存在一个长度大于等于 a 且所有数大于 b 的子串。求满足条件的 S 数量，对 $998,244,353$ 取模。

$1 \le n \le 2 \times 10^5, 1 \le l_i, r_i, b \le 10^9$

例题8 P9400 「DBOI」Round 1 三班不一般

- 初步想法：考虑一个朴素的dp
- f[i][j] 表示 前i个数已经考虑完毕最后结尾大于 b的数有 j 个的方案数
- 考虑转移方程：

$$f[i][0] = \sum_{j=0}^{a-1} f[i-1][j] \times (min(r, b-1) - l + 1)$$

$$f[i][j] = \sum_{j=0}^{a-1} f[i-1][j-1] \times (r - max(b, l) + 1)$$

例题8 P9400 「DBOI」Round 1 三班不一般

- 考虑转移方程：

$$f[i][0] = \sum_{j=0}^{a-1} f[i-1][j] \times (min(r, b-1) - l + 1)$$

$$f[i][j] = \sum_{j=0}^{a-1} f[i-1][j-1] \times (r - max(b, l) + 1)$$

- 第一个式子显然可以前缀和一下得到。
- 考虑第二式子的式子，整体平移后，乘以一个数值。
- 设计单点插入，单点删除，区间移动，区间求和，区间乘。显然 平衡树支持这个操作
- 线段树做法?

例题9 gym103439 G. Replace Sort

- 题目大意：
- 给出一个数组A和一个集合B，保证A和B的所有元素都是不重复的，B里面的元素每个只能使用一次，现在使用B里面的元素替换A中的，使得A是一个排序的数组。
- 问最少的替换次数
- A和B的元素个数不超过5 * 10^5

例题9 gym103439 G. Replace Sort

- 初步思路:
- 显然 B 中的数组需要排序（很套路），这样B中的元素以此从小到大替换。

gym103439 G. Replace Sort

- 初步思路:
- 显然 B 中的数组需要排序（很套路），这样B中的元素以此从小到大替换。

- 进一步思路:
- 我们可以令 a[0] = -inf, a[n + 1] = inf
- f[i] 表示前 i 个元素，在a[i]不替换的情况下的满足要求的最少次数

例题9 gym103439 G. Replace Sort

- 初步思路：
- 显然 B 中的数组需要排序（很套路），这样B中的元素以此从小到大替换。
- 进一步思路：
- 我们可以令 a[0] = -inf, a[n + 1] = inf
- f[i] 表示前 i 个元素，在a[i]不替换的情况下的满足要求的最少次数
- 考虑 a[i] 替换：
- 引入 g[i][j] 表示前 i 个元素，在a[i] 替换成 b[j] 的情况下满足要求的最少次数

例题9 gym103439 G. Replace Sort

- f[i] 表示前 i 个元素，在a[i] 不替换的情况下的满足要求的最少次数
- 考虑 a[i] 替换：
- g[i][j] 表示前 i 个元素，在 a[i] 替换成 b[j] 的情况下满足要求的最少次数
- 先考虑一个O(n^2)的转移：
- f[i] = f[i - 1] if a[i - 1] < a[i]
- f[i] = min{ g[i - 1][j] } 满足 b[1]...b[j] < a[i]

例题9 gym103439 G. Replace Sort

- 在考虑 g 的转移:
- g[i][j] = f[i - 1] + 1　　a[i - 1] < b[j]
- g[i][j] = g[i - 1][j - 1] + 1

例题9 gym103439 G. Replace Sort

- 考虑一个O(n^2)的转移：
- f[i] = f[i - 1] if a[i - 1] < a[i]
- f[i] = min{ g[i - 1][j] } 满足 b[1]...b[j] < a[i]
- 在考虑 g 的转移:
- g[i][j] = f[i - 1] + 1 a[i - 1] < b[j]
- g[i][j] = g[i - 1][j - 1] + 1
- 仔细观察上述方程：
- f[i] = f[i - 1] if a[i - 1] < a[i] **Easy**
- g[i][j] = f[i - 1] + 1 a[i - 1] < b[j] **Easy**

例题9 gym103439 G. Replace Sort

- f[i] = min{ g[i - 1][j] } 满足 b[1]...b[j] < a[i] **区间取min**

- g[i][j] = g[i - 1][j - 1] + 1 **整体平移 + 1**

例题10 P9871 [NOIP2023] 天天爱打卡

题目大意

一共有 n 天，大 Y 可以花费 d 的能量在任何一天跑步，但大 Y 不能在连续的 k 天跑步。

此外，将给出 m 段任务区间 $[l_i, r_i]$。如果在这段区间的每一天都跑了步，则会获得 v_i 的能量。

大 Y 的初始能量为 0，请你最大化跑完步后大 Y 最终的能量。

例题10 P9871 [NOIP2023] 天天爱打卡

题目大意：

有 n 天，每天可以选择跑步或者不跑步。若跑步，能量会减少 d（能量中途可以为负）。不能连续跑超过 k 天。题目给定 m 个任务：如果在 $[L, R]$ 天连续跑步，那么就会得到 v 个能量。求最后能量的最大值。

例题10 P9871 [NOIP2023] 天天爱打卡

36分做法：

动态规划状态定义

我们定义 $\mathrm{dp}[i]$ 为截止到第 i 天的能量最大值。

转移方程

我们需要根据是否选择在第 i 天跑步来决定 $\mathrm{dp}[i]$ 的值：

1. **不在第 i 天跑步**：

    - 这种情况下，能量值等于前一天的最大能量值，即：

$$\mathrm{dp}[i] = \mathrm{dp}[i-1]$$

例题10 P9871 [NOIP2023] 天天爱打卡

2. **在第 i 天跑步**：

- 我们需要找一个适当的开始天数 j，使得从第 j 天开始到第 i 天连续跑步，同时满足不能连续跑超过 k 天的限制。
- 假设 j 是跑步的开始天数，则有 $i-j+1 \leq k$，表示从第 j 天到第 i 天之间跑步天数不能超过 k 天。
- 跑步会减少能量，减少的能量为 $(i-j+1) \times d$。
- 我们需要在 $[j,i]$ 区间内获取任务奖励的总和，记作 $\text{cost}(j,i)$。
- 因此，这种情况下的能量值为：

$$\text{dp}[i] = \max(\text{dp}[i], \text{dp}[j-2] - (i-j+1) \times d + \text{cost}(j,i))$$

- 注意 $\text{dp}[j-2]$ 是为了确保 j 到 i 天之前还有一个不跑步的间隔日。

例题10 P9871 [NOIP2023] 天天爱打卡

对于 56 分 考虑一个优化：

$$\mathrm{dp}[i] = \max(\mathrm{dp}[i], \mathrm{dp}[j-2] - (i-j+1) \times d + \mathrm{cost}(j,i))$$

- 其中，如果 $j \geq 2$，那么：

$$\mathrm{tmp} = \mathrm{dp}[j-2]$$

- 更新临时变量 tmp：

$$\mathrm{tmp} += \mathrm{total} - \mathrm{query}(j-1) - d \times (i-j+1)$$

合并公式：

- 计算能量总和并更新 $\mathrm{dp}[i]$：

$$\mathrm{dp}[i] = \max(\mathrm{dp}[i], \mathrm{total} - d \times (i+1) + \mathrm{dp}[j-2] - \mathrm{query}(j-1) + d \times j)$$

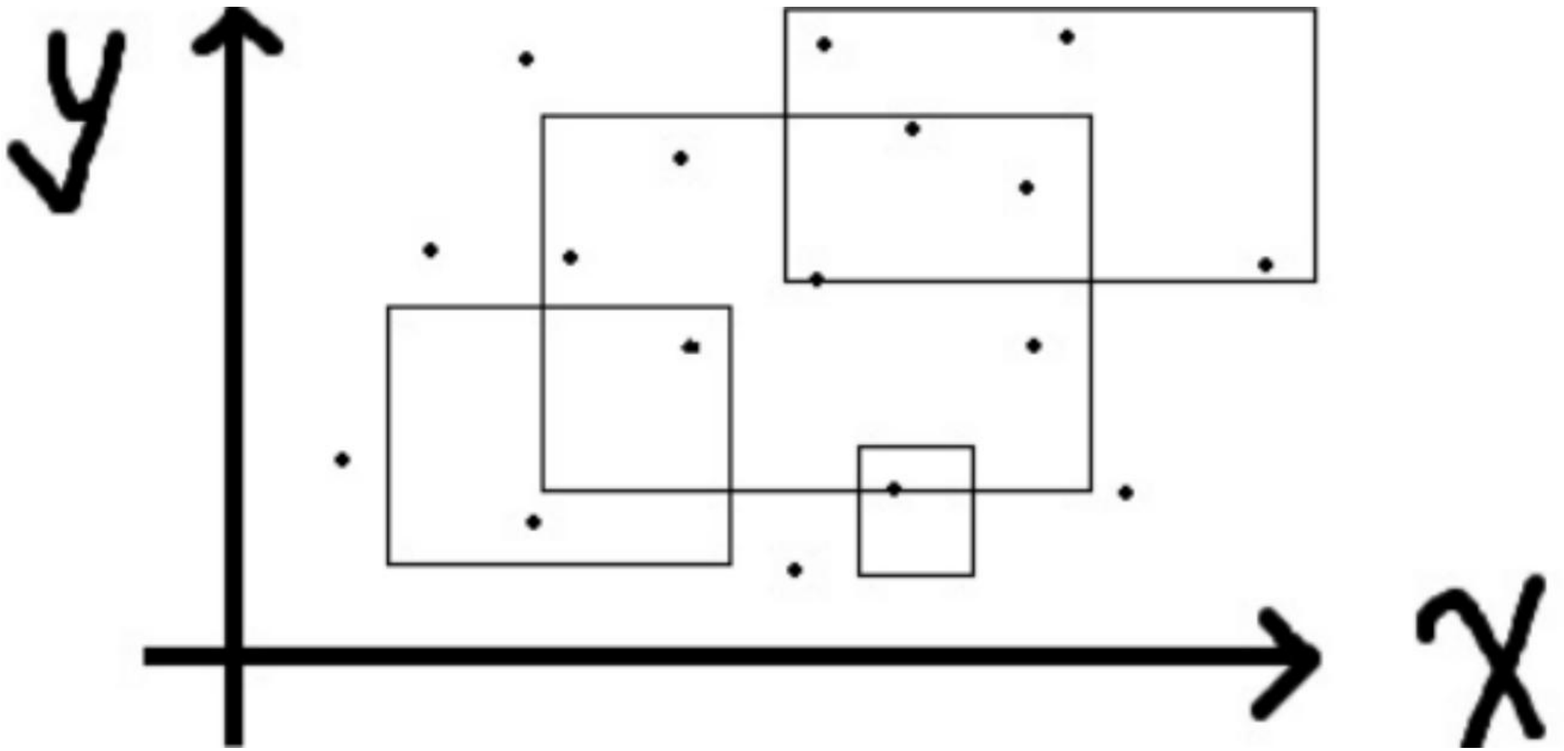
例题10 P9871 [NOIP2023] 天天爱打卡

100 分做法：

离散化一下，把需要转移的关键点取出来 or 动态开点线段树？

什么是二维数点

- 二维数点是什么?
- 二维数点又称二位偏序，形象化的解释就是在一个二维平面上有许多的点，让你数出在一个区间内，一共有几个点。



什么是二维数点

- 二维数点可以干什么?
- 二维数点通常可以处理类似给你一个序列，让你求出在区间 [l, r] 里，有多少数在 [a, b] 之间。
- 其实就是有两个限制的情况下，让你求出有多少符合要求的元素。

怎么解决二维数点

- 二维数点的问题如何转换+处理?
- 我们可以发现，二维数点就是二维偏序，所以有很多的解法，比如：扫描线思想+线段树/树状数组，主席树，CDQ分治，K-D Tree（后两个还可以解决三维偏序问题）。

例题11. P2163[Shoi2007] Tree 园丁的烦恼

- 题目大意：多次询问矩阵和，无修改。

- 对于一个查询我们可以把它拆成四个，也就是用二维前缀和的方式来查询
- 我们发现其实前缀和的定义就是多少个点的横纵坐标都小于这个点

例题11. P2163[Shoi2007] Tree 园丁的烦恼

- 题目大意：多次询问矩阵和，无修改。

第一行有两个整数 n, m，分别表示树木个数和询问次数。

接下来 n 行，每行两个整数 x, y，表示存在一棵坐标为 (x, y) 的树。有可能存在两棵树位于同一坐标。

接下来 m 行，每行四个整数 a, b, c, d，表示查询以 (a, b) 为左下角，(c, d) 为右上角的矩形内部（包括边界）有多少棵树。

例题11. P2163[Shoi2007] Tree 园丁的烦恼

- 题目大意：多次询问矩阵和，无修改。

第一行有两个整数 n, m，分别表示树木个数和询问次数。

接下来 n 行，每行两个整数 x, y，表示存在一棵坐标为 (x, y) 的树。有可能存在两棵树位于同一坐标。

接下来 m 行，每行四个整数 a, b, c, d，表示查询以 (a, b) 为左下角，(c, d) 为右上角的矩形内部（包括边界）有多少棵树。

例题12. P4390 [BalkanOI 2007] Mokia 摩基亚

有三种命令，意义如下：

命令	参数	意义
0	w	初始化一个全零矩阵。本命令仅开始时出现一次。
1	$x\ y\ a$	向方格 (x, y) 中添加 a 个用户。a 是正整数。
2	$x_1\ y_1\ x_2\ y_2$	查询 $x_1 \le x \le x_2$，$y_1 \le y \le y_2$ 所规定的矩形中的用户数量。
3	无参数	结束程序。本命令仅结束时出现一次。

输入共若干行，每行有若干个整数，表示一个命令。

例题12. P4390 [BalkanOI 2007] Mokia 摩基亚

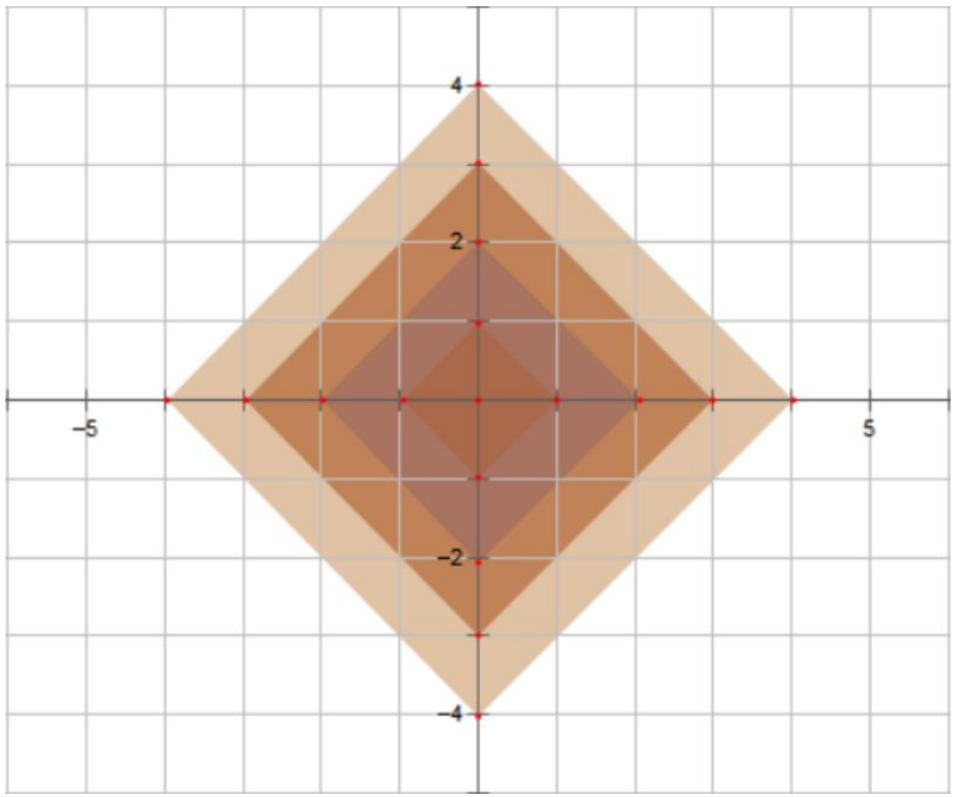
- 思路：考虑CDQ分治，首先将子矩阵查问拆分为4个查询，然后容斥搞一下，
- 将所有询问按照 x 轴排序，然后将所有操作按照时间一分为二。
- （1）先处理掉前半边区间的修改对后半边询问造成的影响，由于已经按照 x 轴排序了，所以用一颗树状数组就可以维护点权和。
- （2）然后还原前半边区间的影响。
- （3）将前半边操作放在左边，后半边操作放在右边，然后分治处理两边。

例题13. BZOJ2989. 数列

- 题目大意：
- 给定一个长度为n的正整数数列a[i]。
- 定义2个位置的graze值为两者位置差与数值差的和，即graze(x,y)=|x-y|+|a[x]-a[y]|。
- 2种操作（k都是正整数）：
- 1.Modify x k：将第x个数的值修改为k。
- 2.Query x k：询问有几个i满足graze(x,i)<=k。因为可持久化数据结构的流行，询问不仅要考虑当前数列，还要考虑任意历史版本，即统计任意位置上出现过的任意数值与当前的a[x]的graze值<=k的对数。（某位置多次修改为同样的数值，按多次统计）

例题13 BZOJ2989. 数列

- 观察到 |x-y|+|a[x]-a[y]| ,我们想到曼哈顿距离。
- 于是我们把(x,a[x])看作一个二维平面上的点，而查询的是到平面上一定点距离<=k的点的个数。同时这样还避免了可持久化的问题，因为把a[x]修改成k，相当于新建了一个点。



例题13 BZOJ2989. 数列

- 曼哈顿距离转切比雪夫距离

(x_1, y_1)和(x_2, y_2)的切比雪夫距离是$max(|x_1 - x_2|, |y_1 - y_2|)$

考虑把两个点转化成$(x_1 + y_1, x_1 - y_1), (x_2 + y_2, x_2 - y_2)$,容易发现新点的切比雪夫距离和原来点的曼哈顿距离相同.

- 那么我们只要把询问和修改的每个点都转成(x+y,x-y)的形式，然后问题就变成在把原来查询的正方形转换后新的正方形里点的个数。由于边都平行于坐标轴，带修改。

例题14 P11364 [NOIP2024] 树上查询

有一天小 S 和她的朋友小 N 一起研究一棵包含了 n 个结点的树。

这是一棵有根树，根结点编号为 1，每个结点 u 的深度 dep_u 定义为 u 到 1 的简单路径上的**结点数量**。

除此之外，再定义 $\mathrm{LCA}^*(l,r)$ 为编号在 $[l,r]$ 中所有结点的最近公共祖先，即 $l, l+1, \ldots, r$ 的公共祖先结点中深度最大的结点。

小 N 对这棵树提出了 q 个询问。在每个询问中，小 N 都会给出三个参数 l, r, k，表示他想知道 $[l,r]$ 中任意长度大于等于 k 的连续子区间的最近公共祖先深度的最大值，即

$$\max_{l\le l'\le r'\le r \wedge r'-l'+1\ge k} \mathrm{dep}_{\mathrm{LCA}^*(l',r')}$$

你的任务是帮助小 S 来回答这些询问。

例题14 P11364 [NOIP2024] 树上查询

- k = 1, 找区间深度最大值， ST表

- 现在考虑 k > 1

例题14 P11364 [NOIP2024] 树上查询

- k = 1, 找区间深度最大值， ST表
- 现在考虑 k > 1, 考虑区间问题转换成 相邻2个元素之间的问题。

1. **关键结论修正**:

   原结论应为:

$$\mathrm{dep}(LCA^*(l,r)) = \min_{i=l}^{r-1} \mathrm{dep}(LCA^*(i,i+1)).$$

   即区间 $[l,r]$ 的 LCA 深度等于该区间内所有相邻节点对的 LCA 深度的最小值。

例题14 P11364 [NOIP2024] 树上查询

- k = 1, 找区间深度最大值， ST表
- 现在考虑 k > 1, 考虑区间问题转换成 相邻2个元素之间的问题。

2. **问题重新形式化**:

   定义 $a_i = \mathrm{dep}(LCA^*(i, i+1))$，则查询问题转化为：

$$\max_{\substack{l \le l' \le r' \le r \\ r'-l'+1 \ge k}} \left(\min_{j=l'}^{r'-1} (a_j) \right).$$

   这里需要找到区间 $[l, r]$ 中所有长度至少为 k 的子区间的最小值的最大值。

- 参考 CF484E　mlog^2n 的做法。

例题14 P11364 [NOIP2024] 树上查询

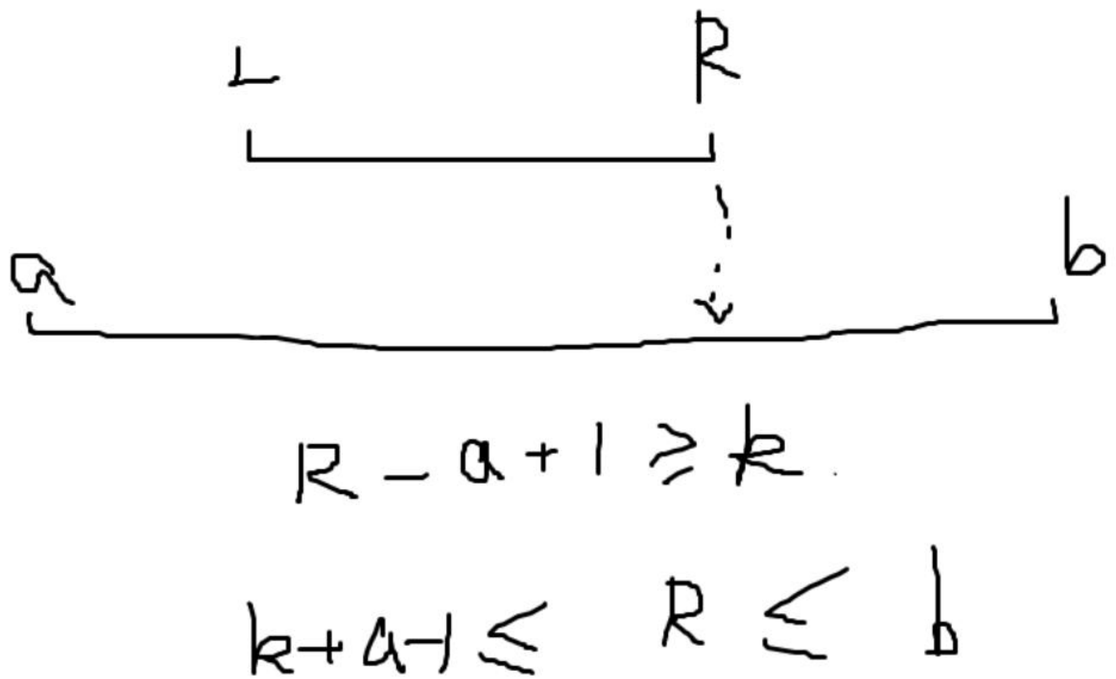
- 单调栈 + 二维数点

- 单调栈：找出最小值为 v 的 极大区间 [L, R, V]， 且极大区间仅有 n 个。

- 转换问题变成：对于查询区间 [a, b] 存在一个极大区间交集为k的情况下的最大区间值。

例题14 P11364 [NOIP2024] 树上查询

- 我们把极大区间按照区间查询从大到小处理。
- 我们把询问也离线按照 k 从大到小处理。



- 对于 查询 [a, b, k]　　对于极大区间 [L, R, v]
- 应当满足情况1：
- R - L + 1 >= k，此时一定满足
- R - a + 1 >= k，　　a <= R < = b

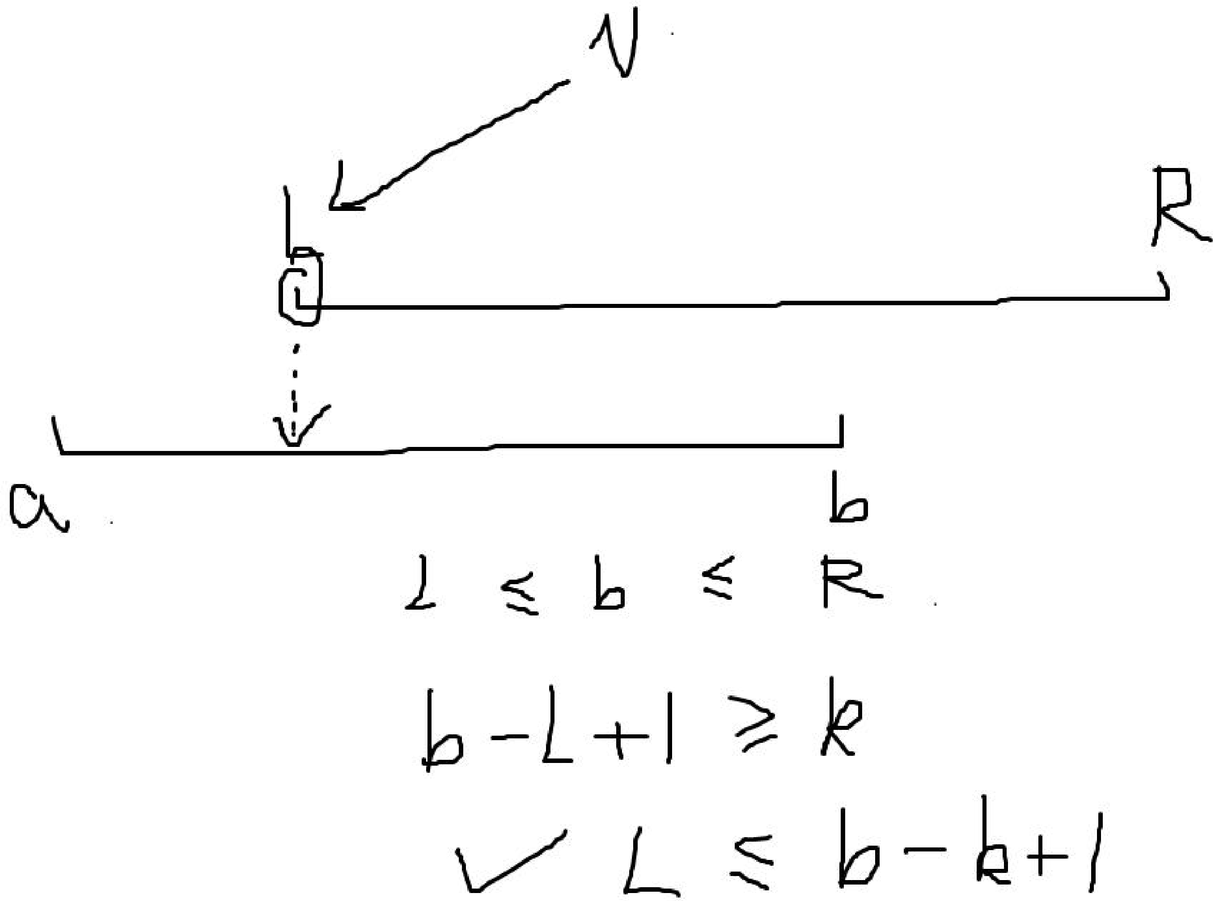
例题14 P11364 [NOIP2024] 树上查询

- 对于应当满足的情况2， 请同学们上来分析。

例题14 P11364 [NOIP2024] 树上查询

- 对于应当满足的情况2， 请同学们上来分析。

- b - L + 1 >= k, L <= b <= R



例题15 CF983E NN country

给定一棵树和若干条路线，每条路线相当于 x, y 之间的路径，途径路径上的每个点

给出若干个询问，每次询问从 u 到 v 至少需要利用几条路线

$N, M, Q \leq 2 \times 10^5$

例题15 CF983E NN country

- 退化成一条链怎么做?

- CF1175E Minimal Segment Cover

例题15 CF983E NN country

- 接下来考虑树怎么做：
- • 两个节点之间的路径可以分为
- • 先向上到 LCA 的第一段
- • 从 LCA 往下的第二段

例题15 CF983E NN country

- 接下来考虑树怎么做：
- • 两个节点之间的路径可以分为
- • 先向上到 LCA 的第一段
- • 从 LCA 往下的第二段

例题15 CF983E NN country

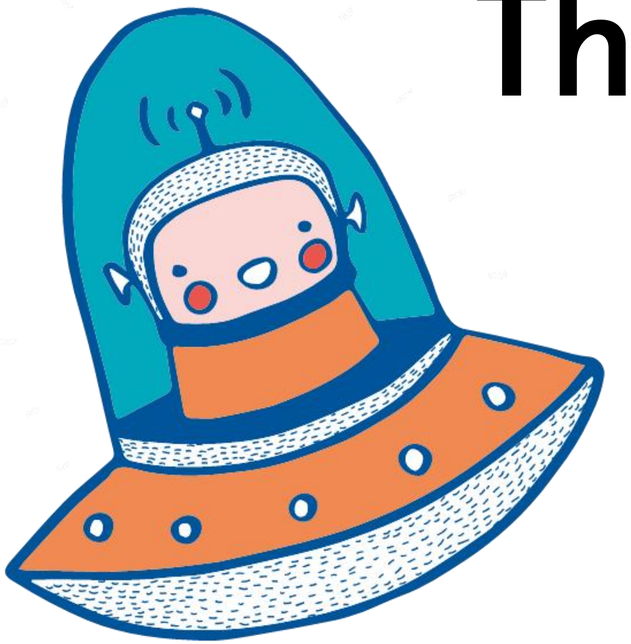
- 两个节点之间的路径可以分为
- 先向上到 LCA 的第一段
- 从 LCA 往下的第二段。
- 对于这两段，我们可以分别用链的做法求出它们最少需要多少步

例题15 CF983E NN country

- 上述显然是不对的，因为可以有一步跨过了这两段，因此我们需要
- 对每组询问 求出它能不能一步跨过这两段，而不是到达了 LCA 以后再往下走

例题15 CF983E NN country

- 如果询问的节点是 (x, y)，假如从 x 往上走到 LCA 的上一步到达
- 了 u，从 y 往上走到 LCA 的上一步到达了 v ，
- • 那么问题就转化成了是否存在一条线路，从 u 所在的子树中出发，
- 到达 v 所在的子树，求出这棵树的 dfs 序后一个节点的子树可以在序列上对应 一个区间，
- 这样就把问题转化成了一个二维数点问题，



Thank You!