单调性与数据结构优化 DP

wlx

2025 年 7 月 25 日

P1 UOJ285

- 有一个长为 $n-1$ 的数组和 q 个区间求和的查询 $[l_i, r_i) \subset [1, n)$。
- 你需要选择若干个切分点 $1 = a_0 < a_1 < a_2 < \cdots < a_k = n$，将这个数组切分为 k 段，第 i 段为 $[a_{i-1}, a_i)$，k 由你决定。
- 这 q 次查询的代价如下：对于任意一对有交的 $[a_i, a_{i+1}), [l_j, r_j)$，若 $[a_i, a_{i+1}) \subset [l_j, r_j)$，则代价为 1；若 $[a_i, a_{i+1}) \supset [l_j, r_j)$，则代价为 $(l_j - a_i) + (a_{i+1} - r_j)$；其余情况下，记 $L = [a_i, a_{i+1}) \cap [l_j, r_j)$，则代价为 $\min(|L|, a_{i+1} - a_i - |L|)$。
- 请计算所有可能划分的最小代价。
- $n \le 5 \times 10^4, q \le 10^5$

P1 UOJ285

一个简单的 DP 如下：记 f_i 表示已经划分好 $[1,i)$ 且最后分割点在 i 前的最小代价。

记 $w(l,r)$ 表示区间 $[l,r)$ 的贡献，显然有 DP

$$f_i = \min_{j<i}\{f_j + w(j,i)\}$$

先考虑如何快速计算 $w(l,r)$，不难发现所有代价都可以转化为二维限制下的矩阵查，可以用主席树做到单次 $O(\log n)$。

w 看上去没有明显的性质，只能猜想 w 满足四边形不等式。

大力讨论一下发现确实满足，于是可以直接套用二分法，时间复杂度 $O(n\log^2 n)$。

P2 ARC066F

- 有 n 道题，做出第 i 题需要 T_i 的时间。
- 你可以任选题目作答，假如你做的题目集合为 S，则你的分数为

$$\sum_{1\le l\le r\le n} [[l,r]\subset S] - \sum_{i\in S} T_i$$

- 有 q 次询问，每次询问给出 u,w，问假如将初始的 T_u 改为 w，你的最大分数是多少。询问之间两两独立。
- $n,q\le 3\times 10^5$

P2 ARC066F

这种单点独立修改的题有个套路，就是维护 f_i 表示考虑前 i 题且不选 i 的最大收益，g_i 表示考虑后 $n-i+1$ 题且不选 i 的最大收益，和 h_i 表示全局选 i 的最大收益，然后 $O(1)$ 回答询问。

先考虑 f 的转移，形如

$$f_i = \max_{j<i} \left\{ f_j + \binom{i-j}{2} - (s_{i-1} - s_j) \right\}$$

显然是斜率优化的形式，可以用单调栈直接做。g 的处理也是类似的。

P2 ARC066F

h 的求法可以写作

$$h_i = \max_{l<i<r} \left\{ f_l + g_r - (s_{r-1} - s_l) + \binom{r-l}{2} \right\}$$

有一个套路，考虑分治，要转移左半边时就建右半边的凸包，转移右半边时就建左半边的凸包，这样又可以斜率优化了。

时间复杂度 $O(n \log n)$。

P3 无来源

- 有 n 家商店，它们一共售卖 m 个商品，每个商品有价格 v_i。
- 如果想购买第 i 商店的商品，需要先交 c_i 的入场费。
- 已知每家商店售卖的物品和价格，询问购买 $1, 2, \ldots, k$ 件商品的最小代价。
- $n, m \le 2 \times 10^6, k \le 1000$

P3 无来源

肯定优先买便宜的商品，设商品前缀和为 $s_{i,j}$，其中第一个商品加上入场费。

设 $f_{i,j}$ 表示前 i 个商店买了 j 个物品的最小代价，有转移

$$f_{i,j} = \min_{k \le j}\{f_{i-1,k} + s_{i,j-k}\}$$

如果不考虑入场费，直接有 s_i 上凸，然后满足四边形不等式和决策单调性。

考虑入场费也没关系，多做一个 $f_{i,j} = f_{i-1,j}$ 的转移即可，这样可以二分或分治地做到 $O(nk \log k)$。单看转移很难继续优化了。

P3 无来源

考虑将所有商店按 $s_{i,1}$ 排序，并取前 k 个商店。注意到剩余商店不能恰好买一个商品，否则不如在前面的空商店买。

再将剩下的商店按 $s_{i,2}$ 排序，取前 $\frac{k}{2}$ 个商店，类似的可以推出剩余商店不能恰好买一个或两个商品。重复此过程，则商店数变为至多 $O(k\log k)$ 个。

再套用之前的做法，则时间复杂度为 $O(k^2\log^2 k)$。

P4 UOJ672

- 平面上有一个正 n 边形，顶点编号顺时针依次为 $1, 2, \ldots, n$。
- 这 n 个顶点之间有 $2n-3$ 条边，恰好构成了这个多边形的所有边和一个三角剖分，每条边有一个长度。
- 初始时，图上有两枚棋子。接下来有 q 个请求，在第 i 个请求中，你需要选择两个棋子中的一个，移动到 p_i 上。假设你选择的棋子现在在 u 上，则收益为 u, p_i 之间的最短路长度。棋子可以同时位于同一个结点上。
- 你不知道图的结构，只知道请求和棋子的初始位置。你可以进行 L 次查询，每次获取两个点之间的最短路长度。
- 你需要按顺序满足所有要求，并最大化收益。
- $n \leq 5 \times 10^4, q \leq 3 \times 10^4, L \leq 2 \times 10^6$

P4 UOJ672

如果不考虑查询次数限制，有个暴力的 DP：记 $f_{i,j}$ 表示第 i 次请求完成后，一个棋子在 p_i，另一个在 j 上的最大收益，转移为

$$f_{i,j} + \mathrm{dis}(p_i, p_{i+1}) \to f_{i+1,j}, \quad f_{i,j} + \mathrm{dis}(j, p_{i+1}) \to f_{i+1,p_i}$$

由于 d 只能通过查询得到，难以优化，我们需要考察 dis 的性质。

尝试后发现 dis 满足四边形不等式。且进一步，对于任意顺时针排列的四个点 u, v, x, y，成立

$$\mathrm{dis}(u, x) + \mathrm{dis}(v, y) \geq \mathrm{dis}(u, y) + \mathrm{dis}(v, x)$$

为方便书写，我们将 $[1, n]$ 复制到 $[n+1, 2n]$ 上，这样上述不等式的限制就成了 $u \leq v \leq x \leq y < u + n$

P4 UOJ672

是不是可以直接转移了呢？并不行，因为此时的 dis 矩阵并不直接满足决策单调性。我们需要保证 $i > j$ 的 (i, j) 上没有值才行

所以，只对 $j \in [i, i+n-1]$ 的 (i, j) 进行转移即可满足单调性。

下面考虑实现，先滚动掉 i 这一维。如果只有 $f_{i,j} + \text{dis}(p_i, p_{i+1}) \to f_{i+1,j}$ 的转移，那只需要维护一个全局加的标记，这个修改不改变最大值的位置

关键在于单点修改，由于这等价于给矩阵的某一行加上一个值，会导致一个区间上的最大值移动到该位置上。

我们可以用 set 维护最大值区间，在每次修改时二分出左右端点

这样只需要 $O(q \log 2n)$ 次查询，时间复杂度 $O(q \log^2 n)$

P5 LOJ3919

- 给定一个长为 n 的 (,) 括号串，你需要将其划分为恰好 k 段，并最小化每段内合法括号子串数量之和。
- $k \leq n \leq 10^6$

P5 LOJ3919

先将括号序列转为 ± 1 序列，并求出前缀和 S_i，于是 $[l,r]$ 合法当且仅当

$$S_{l-1} = S_r = \min_{l\le i\le r} S_i$$

min 的存在启发我们建出广义笛卡尔树（多叉树），每个结点 u 对应一个区间 $[l_u, r_u]$，内部的最小值位置记为 A_u。

下面考虑如何计算区间 $[L,R]$ 的权值 $w(L,R)$。考虑 $[l_u, r_u]$ 对 $w(L,R)$ 的贡献

$$\binom{\sum_{i\in A_u}[L\le i\le R]}{2}$$

不难发现单个 $[l_u, r_u]$ 的贡献满足四边形不等式，于是 $w(L,R)$ 满足四边形不等式，于是答案有凸性，可以使用 WQS 二分。

P5 LOJ3919

接下来将问题转化为树上 DP。注意到在结点 u 处，我们只关心儿子对应的区间有没有被切，而不关心切的次数和位置。

对于每个 u，记 f_i 表示考虑前 i 个儿子，在 v_i 内切了至少一刀的最小代价。有转移

$$f_i = \min_{j<i}\left\{f_j + sh_{i-1} - sh_j + \binom{pre_i - pre_j}{2}\right\} + g_{v_i}$$

g_u 表示 u 内至少切一刀的最小代价，sh_i 表示不切的最小代价的前缀和。注意最小值位置要特殊考虑。

直接上斜率优化即可。时间复杂度 $O(n\log n)$

P6 P8864

- 给定一个长度为 n 的 01 序列 a 和参数 k。
- 有 q 次询问，每次给定 L, R，你可以若干次选择某个 $i \in (L, R]$，将 a_{i-1} 修改为 $a_{i-1} \oplus a_i$，a_{i+1} 修改为 $a_{i+1} \oplus a_i$（如果 $i+1 \le R$）。询问之间相互独立。
- 求使得 $[L, R]$ 内至多有 k 个 1 的最小操作数。
- $n \le 3000, k \le \min(n, 1000), q \le 5 \times 10^5$

P6 P8864

先对 a 做前缀和，这样变成每次交换相邻的 a_{i-1} 和 a_i

不妨先假设 $a_{L-1}=0$，现在要求 $[L-1,R]$ 内相邻且不同的数对个数 $\le k$

我们将要求改为 $[L,R]$ 内至多有 d 段连续的 1

考虑区间 DP，设 $f_{l,r,i}$ 表示将 $a[l,r]$ 中的 1 划分为至多 i 段 1 的最小操作数，则有转移

$$f_{l,r,i}=\min_{l\le p<r}\{f_{l,p,i-1}+w(p,r)\}$$

这里用 $w(l,r)$ 表示将 $(l,r]$ 中的所有 1 移到同一段的最小操作数。计算出第 i 个 1 前 0 的个数后不难用前缀和优化 $O(n^2)$ 预处理

之所以写成 $w(p,r)$，是因为 $(\min,+)$ 类的矩阵操作有结合律，可以快速幂优化。

P6 P8864

接下来就要考虑 w 的性质。注意到 w 等价于到中位数的距离和，这是比较经典的四边形不等式形式。

于是运算得到的中间矩阵均满足四边形不等式，可以 $O(n^2)$ 完成 $(\min,+)$ 矩乘

那么就可以 $O(n^2\log k)$ 求出 f 数组

考虑如何得出答案。当 k 为偶数时，答案为 $f_{l,r,k/2}$；当 k 为奇数时，答案为

$$\min_{l\le i<r}\{f_{l,i,(k-1)/2}+w'(i,r)\}$$

其中 $w'(l,r)$ 表示将所有 $(l,r]$ 中的 1 移到区间末尾的操作数。显然这也可以用四边形不等式优化。

如果 $a_{L-1}=1$ 怎么办？将 a 全部取反做一遍即可。时间复杂度 $O(n^2\log k)$

P7 CF1534G

- 一个棋子初始在二维平面的原点上，只能向上和向右移动。
- 给定平面上的 n 个点，第 i 个点为 (x_i, y_i)。
- 当棋子在 (x, y) 上时，它给第 i 个点打标记需要 $\max(|x - x_i|, |y - y_i|)$ 的代价。
- 求给所有点都打上标记的最小代价。
- $n \le 8 \times 10^5$

P7 CF1534G

可以转成曼哈顿距离，不过没必要。注意到直线 $x+y=x_i+y_i$ 与路径的交点一定是打标记最优处，所以先将所有点按 x_i+y_i 排序

记 $f_{i,x}$ 表示考虑到第 i 个点，当前横坐标为 x 的最小代价，则有

$$f_{i,x} = |x-x_i| + \min_{x-D_i \le x' \le x} f_{i-1,x'}$$

其中 $D_i = (x_i+y_i)-(x_{i-1}+y_{i-1})$。

由于 $|x-x_i|$ 是下凸的，下凸函数的区间 min 还是下凸的，可归纳得出 $f_{i,x}$ 关于 x 下凸。于是我们只需要维护下凸包。

P7 CF1534G

区间 min 相当于将最低点右侧的部分再向右平移区间长度。

由于 $|x - x_i|$ 斜率只有 ± 1，我们可以使用一个经典技巧，拿可重集维护斜率变化 1 的位置。

用两个可重集分别维护最低点左侧和右侧的部分，那么平移只需要打标记即可。

时间复杂度 $O(n \log n)$。

P8 LOJ2537

- 给定一颗 n 个节点的二叉树，初始给定所有叶子节点的点权，保证点权各不相同。
- 对于非叶子结点 u，它的权值有 p_u 的概率是子节点权值的较大值，$1-p_u$ 的概率是较小值。
- 假设根节点的权值有 m 种可能性，第 i 小可能性的权值是 V_i，对应的概率为 D_i，求 $\sum_i i \cdot V_i \cdot D_i^2 \bmod 998244353$
- $n \le 3 \times 10^5$

P8 LOJ2537

先离散化，肯定是要求所有的 D_i，考虑如何对每个节点维护。

u 只有一个儿子的情况是简单的，考虑有两个儿子时的转移：记左儿子的概率数组为 L，右儿子的为 R，转移形如

$$D_i = L_i \left[(1-p_u) \sum_{j>i} R_j + p_u \sum_{j<i} R_j \right] + R_i \left[(1-p_u) \sum_{j>i} L_j + p_u \sum_{j<i} L_j \right]$$

转移基本只需要前缀/后缀和，同时叶子节点的 D 只有一个位置有值，考虑线段树合并维护 D

假设到某个位置只有一方有结点，那只需要一个区间乘标记。如果两边都有结点的话就往下做，动态维护当前位置的前后缀和即可。

时间复杂度 $O(n\log n)$。本题还有动态 DP 做法，大家也可以思考一下。

P9 LOJ3044

- 给定一颗以 1 为根，有 n 个结点的树，初始所有叶子节点的点权等于它们的编号。
- 对于非叶子结点，若其深度为奇数，则其点权为子节点点权的较大值，反之则为较小值。
- 将叶子节点 i 的点权修改为 w_i 所需的代价为 $|i - w_i|$。对于一个叶子节点的集合 S，$w(S)$ 定义为：所有修改 S 中点的点权，使得根节点点权改变的方案的最小代价。此处一个方案的代价定义 $\max_{i \in S} |i - w_i|$。
- 特别的，如果无论如何也无法使根节点点权改变，则定义 $w(S) = n$
- 给出 L, R，请你对每个 $k \in [L, R]$，求出使得 $w(S) = k$ 的 S 的数量。
- $2 \le n \le 2 \times 10^5$，模 998244353

P9 LOJ3044

不妨求 $w(S) \le k$ 的 S 的数量，先将每个点的初始点权 W_i 求出。

若 $W_1 \in S$，直接将点 W_1 的权值 $+1$ 即可，这是平凡的。下面讨论 $W_1 \notin S$ 的情况。

显然 $W_u = W_1$ 的 u 构成了从 1 到 W_1 的一条链，修改任意一个 W_u 都会使 W_1 变化。

考虑链上的某一点 u，假设它是第一个权值改变的点。若 u 的深度为奇数，则 W_u 只可能增大，为偶数时是类似的。

将链上的边断掉后，考虑 u 所在连通块（子树）的叶子节点。由于我们只关心 $\max|i - w_i|$，我们可直接确定出叶子节点的 f，然后 $O(n)$ DP。

P9 LOJ3044

不难发现转移式和 k 无关，k 增加时恰会影响一个叶子节点处 f 的初值，且转移满足结合律，所以可以使用动态 DP 维护，时间 $O(n\log^2 n)$，细节很多。

更快的方法是直接使用线段树合并。设 $f_{u,k}$ 表示代价不超过 k 时的 f_u，则叶子节点处的 f 只有两段。合并的时候只有点积和区间加，直接做即可。计算答案时只需要把链上每个点不改变的方案数乘起来，在减掉即可，也可用线段树合并做。

当然也可和上题一样，将 f 定义改为恰好为 k 然后做前后缀和，细节是类似的。时间复杂度 $O(n\log n)$。

P10 LOJ3711

- 给定一棵 n 个结点的树，每个点有点权。
- 我们定义 (x, y) 合法当且仅当 x 是 y 的祖先。此时 $f(x, y)$ 定义为从 x 出发，对 x 的子树进行 DFS 寻找 y，期间访问过的所有结点点权的最小值的期望。此处规定 DFS 时对未访问的子节点的选取时等概率随机的。
- 请求出所有合法的 (x, y) 的 $f(x, y)$ 之和，对 998244353 取模。
- $1 \leq n \leq 4 \times 10^5$

P10 LOJ3711

先离散化，对于权值 V，我们计算 $\min \geq V$ 的概率，然后求和即可。

将 $\geq V$ 的位置记为黑点，$< V$ 的位置为白点，则 $x \to y$ 如果走到白点或进入链外某棵存在白点的子树一定不合法。

设结点 u 除含 y 的子树外共有 c_u 棵含 y 的子树，不难说明 u 对概率的贡献是 $(c_u+1)^{-1}$，当然这里要求 u 是黑点。将 u 的贡献记作 w_u

当 y 是黑点时，$f(x,y) = \prod_{u\in[x\to y)} w_u$

P10 LOJ3711

下一步考虑 $g_x = \sum_y f(x, y)$，我们有转移式

$$g_u = 1 + \sum_v \frac{g_v}{c_{u,v} + 1}$$

这里的 $c_{u,v}$ 是指 y 在 v 子树内时的 c_u。显然 u 为白点时 $c_u = 0$。

现在从大往小扫描 V，则每个点会从白变黑，每颗子树也会从含白点变成不含白点，总共 $2n$ 次修改。

显然转移式符合矩阵形式，于是可以使用动态 DP 维护。两种修改都不难通过维护重儿子、轻儿子信息实现。

由于树是静态的，可以套用全局平衡二叉树，做到 $O(n \log n)$

P11 QOJ5171

- 有 n 个人参加 m 场比赛，第 i 个人只参加 $[l_i, r_i]$ 中的比赛。
- 初始时第 i 个人分数为 w_i，每场有人参加比赛会使某一名参赛者分数 $+1$。
- 记所有比赛结束后分数 $\leq v$ 的选手的集合为 S。
- 问 $|S|$ 的最大值，以及有多少种可能的 S 可以取到最大值。
- $n, m \leq 2 \times 10^5$，满足 $\forall i, j$ s.t. $l_i < l_j$ 都有 $r_i \leq r_j$，答案模 $10^9 + 7$

P11 QOJ5171

先解决第一个问题，求 $|S|$ 的最大值。

不难想到我们有个基于贪心的做法。先将初始分数 $\leq v$ 的人的分数上限设作 v，$> v$ 的设作 ∞。

依次考虑每场比赛，每次选择最靠左的、分数未达到上界的人，使其分数 $+1$。若这样的人不存在，则将最靠右的人分数上限改为 ∞。

对于 S 数量的计算，我们还需要一个判定算法。对于某个 S，先将有 S 外的人参加的比赛删去，这样剩余的比赛和参赛者一定构成若干个连续段，且每个段对应的参赛者两两不交。我们可以对每个段跑一遍贪心算法进行判定。

P11 QOJ5171

设 f_i 表示考虑完前 i 个选手，钦定 $i \notin S$ 时，$|S|$ 的最大值和对应的方案数。转移可枚举上一个删掉的选手 j，并判定区间 $[r_j+1, l_i-1]$。

注意到，对于某个连续段 $[l,r]$，它对应的参赛者恰为比赛区间与其有交的人。所以判定结果可表示为固定的 $a_{l,r}$。进一步易得，对于某个 l，满足 $a_{l,r}=1$ 的 r 恰为从 l 开始的一段前缀。如果能求出 $R(l)$，就可以直接线段树优化了。

先将 $w_i > v$ 的选手处理掉，然后对 r 扫描线。维护 $b_{l,r}$ 表示考虑完 $[l,r]$ 的比赛后，参加 r 的选手最多还能赢几场比赛。现在 $b_{l,r}$ 关于 l 是单调不降的。

加入区间和比赛是简单的，现在要考虑如何删去区间。事实上，我们只需要考虑右端点 $>r$ 的选手的容量之和 V，再将 $b_{l,r}$ 和 V 取 min 即可。由于 b 的单调性，这实质上是区间赋值。

用线段树维护即可，时间复杂度 $O(n \log n)$。

P12 GYM102155J

- 给定一个长为 n 的序列 a 。你需要将 a 划分成两个子序列（可为空），使得它们的权值和最小。
- 一个序列 b 的权值定义为 $\sum_i \max\{b[1,i]\} - b_i$
- $n \leq 10^5, a_i \leq 10^9$

P12 GYM102155J

记 $f_{i,j}$ 表示考虑完前 i 个数，其中一个序列的最大值为 $\max\{a[1,i]\}$，另一个为 j 的最小权值。

记 $m_i = \max\{a[1,i]\}$，有一个简单的转移

$$f_{i,j} = f_{i-1,j} + \begin{cases} m_i - a_i & j < a_i \\ j - a_i & j > a_i \end{cases}$$

$$f_{i,a_i} = \min_{l \le a_i} f_{i-1,l}$$

将 f_{i,a_i} 的转移视作单点修改，考虑其他的转移怎么维护

先把 $-a_i$ 用全局 tag 搞掉，现在要支持前缀加定值和后缀加下标，同时查询前缀最小值，不好做。

P12 GYM102155J

但注意到一点：如果某个时刻存在 $j < l$ 使得 $f_{i,j} < f_{i,l}$，那么 $f_{i,l}$ 没有任何用处。进一步的，在下一次单点修改之前，l 位置上的值都可以删掉。

如何维护这个过程呢？先离散化，假设有相邻的两个位置 $j, k, j > k$，前缀加不会改变 $f_k - f_j$，后缀加每次会使其减少 $j - k$。

我们可以直接计算使得 $f_k - f_j \leq 0$ 所需的后缀加次数，每次删除和单点加更新前后位置的次数，这样的操作次数显然是 $O(n)$ 的。

可以用平衡树或线段树做到 $O(n \log n)$。分块实现更简单，$O(n\sqrt{n})$ 也可通过。

P13 P10181

- 有一个长为 n 的序列 a。定义区间 $[l,r]$ 的权值为 $a[l,r]$ 中不同数的个数。
- 有 m 个询问，每次给出 x,k，问将区间 $[1,x]$ 划分为 k 段的最大区间权值和。
- $a_i \le n \le 10^5, m \le 10^6$

P13 P10181

先考虑单次询问，设状态 $f_{i,k}$，有转移 $f_{i,k} = \max_{j \le i}\{f_{j-1,k-1} + w(j,i)\}$

不难发现 $w(j+1,i)$ 有四边形不等式，于是 k 可以 WPS 二分掉

设斜率为 c，转移为 $f_i = \max_{j \le i}\{f_{j-1} + w(j,i) - c\}$

不难注意到，当 c 很大时，切的段数就相对很少。不难说明

$$c \le \frac{n}{L(c) - 1}$$

其中 $L(c)$ 表示最优时的切分段数。

P13 P10181

考虑取阈值 B，当 $k \le B$ 时正常 DP，当 $k > B$ 时 WQS 二分，取 $B = \sqrt{n}$

注意到 $f_{j-1} + w(j, i) - c$ 在从左往右扫时要支持三种操作：末尾加数、后缀加 1 和全局求 max。

线段树可以 $O(\log n)$ 做，但不够快。考虑单调栈，维护相邻两个数的差分，这样除了定位的所有操作都可以 $O(1)$ 处理。

定位可以直接用并查集维护栈内的点，做到 $O(n\alpha(n))$ 甚至 $O(n)$。

两种情况都可以使用此算法，总时间 $O(n\sqrt{n})$

拓展：SMAWK 算法

模板题：CF1423M Milutin's Plums

几乎没什么用，极少出现，可以 $O(n)$ 处理一部分决策单调性问题，感兴趣的可以自行学习。