DP 优化

Tea

Yali High School

2019 年 8 月 15 日

- 由于本人水平有限，不免出现纰漏，请各位大佬多多指正。
- 由于时间有限，部分大家较熟悉的部分将较少涉及。
- 选题较水，请放心食用。
- 欢迎大家积极参与讨论。

几种常见的 dp 优化类型

- 状态数优化
- 维护决策集合
- 优化决策点

套路

先来一道经典题。

套路

先来一道经典题。

给定两个字符串 S, T, 求最长公共子串长度。
$|S| \leq 10^6, \ |T| \leq 10^3$，字符集大小：26

套路

一般会想设状态 $F[i,j]$ 表示 S 的前 i 位，T 的前 j 位的最长公共子串多长，然而 $|S|$ 太大了，状态爆炸。注意 $|T|$ 很小，并且答案也 ≤ 1000 ，考虑**将值与状态互换**。

套路

设状态 $F[i,j]$ 表示到 T 的第 i 位，到 S 的第 $F[i,j]$ 位，最长公共子串长度为 j，不难发现这个状态取最小值最优，记 $nxt[i][c]$ 为从 S 的第 i 位向后最近字符 c 在哪，有转移:

$$F[i+1,j] = min\{F[i+1,j],\ F[i][j]\}$$
$$F[i+1,j+1] = min\{F[i+1,j+1],\ nxt[F[i,j]][S[i+1]]\}$$

时间复杂度 $O(26|S|+|T|^2)$

费用提前计算

将一个长为 n 的序列 a 划分为若干段，每一段代价为 **当前段数** × **该段和**，要求最小化代价。

$n \leq 10^7$

费用提前计算

设 $F[i][j]$ 为前 i 个数，分为 j 段的最小代价之和。

$$F[i][j] = \min_{k<i}\{F[k][j-1] + j(\sum_{l=k+1}^{i} a_l)\}$$

$O(n^3)$

费用提前计算

如果计算贡献，那么 j 这一维是不需要的。设 $F[i]$ 为前 i 个数分成若干段且包括对后续贡献的最小代价。

$$F[i] = \min_{j<i}\{F[j] + \sum_{k=j+1}^{n} a_k\}$$

再用个变量记一下 $F[j] + \sum_{k=j+1}^{n} a_k$ 的最小值转移即可，$O(n)$ 。

基本方法

对于转移

$$F[i] = (\text{限制条件})\{F[j] + val(j,i)\}$$

$val(j,i)$ 为一个和 i,j 有关的多项式。

基本方法

对于转移

$$F[i] = (\text{限制条件})\{F[j] + val(j, i)\}$$

$val(j, i)$ 为一个和 i, j 有关的多项式。

如果 $val(j, i)$ **中的每一项都只跟** i **或** j **有关，**那么大多都可以将 $val(j, i)$ 拆成 $g(i)$ 与 $h(j)$，即

$$F[i] - g(i) = (\text{限制条件})\{F[j] + h(j)\}$$

基本方法

对于转移

$$F[i]=(\text{限制条件})\{F[j]+val(j,i)\}$$

$val(j,i)$ 为一个和 i,j 有关的多项式。

如果 $val(j,i)$ **中的每一项都只跟** i **或** j **有关，**那么大多都可以将 $val(j,i)$ 拆成 $g(i)$ 与 $h(j)$，即

$$F[i]-g(i)=(\text{限制条件})\{F[j]+h(j)\}$$

那么就用数据结构维护下 $F[j]+h(j)$，每次求出 $F[i]$ 后，将 $F[i]+h(i)$ 加入维护的决策集合。

基本方法

对于转移

$$F[i] = (\text{限制条件})\{F[j] + val(j,i)\}$$

$val(j,i)$ 为一个和 i,j 有关的多项式。

如果 $val(j,i)$ **中的每一项都只跟** i **或** j **有关，**那么大多都可以将 $val(j,i)$ 拆成 $g(i)$ 与 $h(j)$，即

$$F[i] - g(i) = (\text{限制条件})\{F[j] + h(j)\}$$

那么就用数据结构维护下 $F[j] + h(j)$ ，每次求出 $F[i]$ 后，将 $F[i] + h(i)$ 加入维护的决策集合。

最长上升子序列就是最简单也最典型的例子。

Cleaning Shifts

link

你有 n 个纸条，每条可以覆盖 $[l_i, r_i]$ 的区间，每个纸条有一个费用 c_i 你的任务是覆盖区间 $[L, R]$ 并且花费最少。

$n \le 10^5,\ 1 \le L \le R \le 10^9$

Cleaning Shifs

很简单的一道题。

设 $f[i]$ 为覆盖 $[L, i]$ 的最小代价。

$$f[r_i] = \min_{l_i \le x \le r_i} \{f[x-1]\} + c_i$$

先离散化，然后把所有区间按右端点排序，依次枚举，用线段树维护 min 那一块即可。

Fence

link

有 n 块木板排成一行，有 M 个工匠对这些木板进行粉刷，每块木板至多被粉刷一次。第 i 个工匠要么不粉刷，要么粉刷包含木板 S_i 的、长度不超过 L_i 的连续的一段木板，每粉刷一块可以得到 P_i 的报酬。求如何安排使工匠们获得的总报酬最多。

$1 \leq N \leq 16000,\ 1 \leq M \leq 100$

Fence

先按 S_i 排序。

设 $F[i][j]$ 表示前 i 个工匠粉刷了前 j 个木板 (可以有空的)，有转移:

$$F[i][j] = max\{F[i-1][j], F[i][j-1]\}$$
$$F[i][j] = \max_{j-L_i \le k \le S_i - 1}\{F[i-1][k] + P_i \cdot (j-k)\}, \text{其中} j \ge S_i$$

i 可以看做定值，j 为枚举的状态，k 在枚举决策。

Fence

先按 S_i 排序。

设 $F[i][j]$ 表示前 i 个工匠粉刷了前 j 个木板 (可以有空的)，有转移:

$$F[i][j] = max\{F[i-1][j], F[i][j-1]\}$$
$$F[i][j] = \max_{j-L_i \le k \le S_i-1} \{F[i-1][k] + P_i \cdot (j-k)\}, \text{其中} j \ge S_i$$

i 可以看做定值，j 为枚举的状态，k 在枚举决策。

令 $h(k) = -P_i \cdot k$，$g(j) = P_i \cdot j$

$$F[i][j] - g(j) = \max_{j-L_i \le k \le j-1} \{F[i-1][k] + h(k)\}$$

那么就可以直接用线段树维护转移了，于是我们得到了 $O(nm\log(n))$ 的做法，尽管比较优秀了，但还是无法通过此题。

Fence

$$F[i][j]-g(j)=\max_{j-L_i\leq k\leq j-1}\{F[i-1][k]+h(k)\}$$

Fence

$$F[i][j]-g(j)=\max_{j-L_i\le k\le j-1}\{F[i-1][k]+h(k)\}$$

不难发现**随着 j 的增加，决策集合的上界 S_i-1 不变，下界 $j-L_i$ 在递增。**

Fence

$$F[i][j]-g(j)=\max_{j-L_i\le k\le j-1}\{F[i-1][k]+h(k)\}$$

不难发现**随着 j 的增加，决策集合的上界 S_i-1 不变，下界 $j-L_i$ 在递增。**

对于在决策集合中的 $k_1<k_2$ 若

$$F[i-1][k_1]+h(k_1)<F[i-1][k_2]+h(k_2)$$

Fence

$$F[i][j] - g(j) = \max_{j-L_i \le k \le j-1} \{F[i-1][k] + h(k)\}$$

不难发现**随着 j 的增加，决策集合的上界 $S_i - 1$ 不变，下界 $j - L_i$ 在递增。**

对于在决策集合中的 $k_1 < k_2$ 若

$$F[i-1][k_1] + h(k_1) < F[i-1][k_2] + h(k_2)$$

说明**不仅 k_2 作为决策点优于 k_1 且在决策集合中待的”时间”会更长，k_1 是对于后续的转移完全没有用的决策点。**

Fence

$$F[i][j]-g(j)=\max_{j-L_i\le k\le j-1}\{F[i-1][k]+h(k)\}$$

不难发现**随着 j 的增加，决策集合的上界 S_i-1 不变，下界 $j-L_i$ 在递增。**

对于在决策集合中的 $k_1<k_2$ 若

$$F[i-1][k_1]+h(k_1)<F[i-1][k_2]+h(k_2)$$

说明**不仅 k_2 作为决策点优于 k_1 且在决策集合中待的" 时间" 会更长，k_1 是对于后续的转移完全没有用的决策点。**

Fence

$$F[i][j] - g(j) = \max_{j-L_i \le k \le j-1}\{F[i-1][k] + h(k)\}$$

Fence

$$F[i][j] - g(j) = \max_{j-L_i \le k \le j-1} \{F[i-1][k] + h(k)\}$$

于是我们就可以维护一个随着 k 增大，$F[i-1][k]+h(k)$ 单调递减的队列，每次取队头决策转移，从队尾加入时保证单调性，均摊下来每次操作 $O(1)$ 这就是著名的单调队列优化 dp 的方法。

Fence

$$F[i][j]-g(j)=\max_{j-L_i\le k\le j-1}\{F[i-1][k]+h(k)\}$$

于是我们就可以维护一个随着 k 增大，$F[i-1][k]+h(k)$ 单调递减的队列，每次取队头决策转移，从队尾加入时保证单调性，均摊下来每次操作 $O(1)$ 这就是著名的单调队列优化 dp 的方法。

单调队列优化 dp 的主要思想就是**及时排除不可能的决策，保证决策集合的高度有序性与有效性。**

斜率优化

先看一道题。

任务安排超级弱化版

机器上有 N 个需要处理的任务，它们构成了一个序列。这些任务被标号为 1 到 N ，因此序列的排列为 $1, 2, 3...N$ 。这 N 个任务被分成若干批，每批包含相邻的若干任务。从时刻 0 开始，这些任务被分批加工，第 i 个任务单独完成所需的时间是 T_i 。在每批任务开始前，机器需要启动时间 S ，而完成这批任务所需的时间是各个任务需要时间的总和。注意，同一批任务将在同一时刻完成。每个任务的费用是它的完成时刻乘以一个费用系数 C_i 。请确定一个分组方案，使得总费用最小。$1 \le N \le 5000$, $1 \le S \le 50$, $1 \le T_i, C_i \le 100$

任务安排超级弱化版

求出 T, C 的前缀和 $sumC$, $sumT$，设 $F[i]$ 为把前 i 个任务分成若干批执行的最小费用。

任务安排超级弱化版

求出 T, C 的前缀和 $sumC$, $sumT$，设 $F[i]$ 为把前 i 个任务分成若干批执行的最小费用。

$$F[i] = \min_{0 \le j < i} \{F[j] + sumT[i](sumC[i] - sumC[j]) + S(sumC[N] - sumC[j])\}$$

$S(sumC[N] - sumC[j])$ 表示机器启动时间 S 对后续的费用贡献，所以在算这一批费用的时候没考虑启动时间。时间复杂度 $O(n^2)$

任务安排弱化版

link

题意同上，$1 \leq N \leq 3 \times 10^5$，$1 \leq S, T_i, C_i \leq 512$

任务安排弱化版

$O(N^2)$ 跑不过了。

尝试按之前的方法将后面的多项式**拆成只根 i 或 j 有关的项，**然后你会发现 $-sumT[i] \cdot sumC[j]$ 拆不开，这是**和状态有关的项与决策有关的项相乘的形式**。

任务安排弱化版

$O(N^2)$ 跑不过了。

尝试按之前的方法将后面的多项式**拆成只根 i 或 j 有关的项，**然后你会发现 $-sumT[i] \cdot sumC[j]$ 拆不开，这是**和状态有关的项与决策有关的项相乘的形式**。

所以我们要运用脑髓，放出眼光，大胆优化！

任务安排弱化版

$$F[i] = \min_{0 \le j < i} \{F[j] + sumT[i](sumC[i] - sumC[j]) + S(sumC[N] - sumC[j])\}$$

任务安排弱化版

$$F[i] = \min_{0 \leq j < i} \{F[j] + sumT[i](sumC[i] - sumC[j]) + S(sumC[N] - sumC[j])\}$$

去掉 min 函数，观察决策点 j 的特点，有

任务安排弱化版

$$F[i] = \min_{0 \le j < i} \{F[j] + sumT[i](sumC[i] - sumC[j]) + S(sumC[N] - sumC[j])\}$$

去掉 min 函数，观察决策点 j 的特点，有

$$F[j] = (S + sunT[i])sumC[j] + F[i] - sumT[i] \cdot sumC[i] - S \cdot sumC[N]$$

任务安排弱化版

$$F[i] = \min_{0 \le j < i} \{F[j] + sumT[i](sumC[i] - sumC[j]) + S(sumC[N] - sumC[j])\}$$

去掉 min 函数，观察决策点 j 的特点，有

$$F[j] = (S + sunT[i])sumC[j] + F[i] - sumT[i] \cdot sumC[i] - S \cdot sumC[N]$$

于是设

$$K(i) = S + sumT[i]$$
$$B(i) = F[i] - sumT[i] \cdot sumC[i] - S \cdot sumC[N]$$

任务安排弱化版

$$F[i]=\min_{0\le j<i}\{F[j]+sumT[i](sumC[i]-sumC[j])+S(sumC[N]-sumC[j])\}$$

去掉 min 函数，观察决策点 j 的特点，有

$$F[j]=(S+sunT[i])sumC[j]+F[i]-sumT[i]\cdot sumC[i]-S\cdot sumC[N]$$

于是设

$$K(i)=S+sumT[i]$$
$$B(i)=F[i]-sumT[i]\cdot sumC[i]-S\cdot sumC[N]$$

$K(i)$ 与 $B(i)$ 是只根 i 有关的多项式，$K(i)$ 是目前已知的。

$$F[j]=K(i)\cdot sumC[j]+B(i)$$

任务安排弱化版

$$F[j] = K(i) \cdot sumC[j] + B(i)$$

可以看做一条 $sumC[j]$-o-$F[j]$ 坐标系上的一条直线，斜率 $K(i)$ 是确定的。

任务安排弱化版

$$F[j] = K(i) \cdot sumC[j] + B(i)$$

可以看做一条 $sumC[j]$-o-$F[j]$ 坐标系上的一条直线，斜率 $K(i)$ 是确定的。

$$F[i] = B(i) + sumT[i] \cdot sumC[i] - S \cdot sumC[N]$$

任务安排弱化版

$$F[j] = K(i) \cdot sumC[j] + B(i)$$

可以看做一条 $sumC[j]$-o-$F[j]$ 坐标系上的一条直线，斜率 $K(i)$ 是确定的。

$$F[i] = B(i) + sumT[i] \cdot sumC[i] - S \cdot sumC[N]$$

所以对所有可能的决策点 $1 \leq j < i$，在平面上是一个个散乱分布的点 $(sumC[j], F[j])$，一个点若使经过它斜率为 $K(i)$ 的直线的截距 $B(i)$ 最小那么这个点就为 $F[i]$ 的最优决策点。

任务安排弱化版

这题要求截距最小，那么最优决策点是在下凸壳上的点。

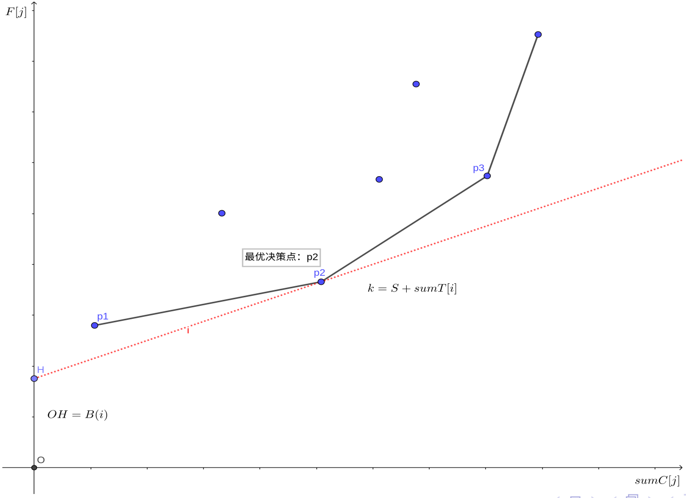
任务安排弱化版

这题要求截距最小，那么最优决策点是在下凸壳上的点。

且对于 $F[i]$ 的决策点 j，设它在凸壳上是 p_2，前一个点是 p_1，后一个点是 p_3 该点 p_2 满足

$$slope(p_1, p_2) \le K(i) \le slope(p_2, p_3)$$

任务安排弱化版



任务安排弱化版

这题要求截距最小，那么最优决策点是在下凸壳上的点。

且对于 $F[i]$ 的决策点 j，设它在凸壳上是 p_2，前一个点是 p_1，后一个点是 p_3 该点 p_2 满足

$$slope(p_1, p_2) \leq K(i) \leq slope(p_2, p_3)$$

由于 $sumT[i] + S$ 是单增的，所以我们用一个斜率单调递增的队列来维护凸壳上的点，并满足队头两点的斜率刚好 $\geq S + sumT[i]$，那么队头就是决策点。

任务安排弱化版

这题要求截距最小，那么最优决策点是在下凸壳上的点。

且对于 $F[i]$ 的决策点 j ，设它在凸壳上是 p_2 ，前一个点是 p_1，后一个点是 p_3 该点 p_2 满足

$$slope(p_1,p_2) \le K(i) \le slope(p_2,p_3)$$

由于 $sumT[i]+S$ 是单增的，所以我们用一个斜率单调递增的队列来维护凸壳上的点，并满足队头两点的斜率刚好 $\ge S+sumT[i]$ ，那么队头就是决策点。

时间复杂度 $O(n)$ 。

任务安排弱化版

这题要求截距最小，那么最优决策点是在下凸壳上的点。

且对于 $F[i]$ 的决策点 j，设它在凸壳上是 p_2，前一个点是 p_1，后一个点是 p_3 该点 p_2 满足

$$slope(p_1, p_2) \leq K(i) \leq slope(p_2, p_3)$$

由于 $sumT[i] + S$ 是单增的，所以我们用一个斜率单调递增的队列来维护凸壳上的点，并满足队头两点的斜率刚好 $\geq S + sumT[i]$，那么队头就是决策点。

时间复杂度 $O(n)$。

其实，当斜率单调，$sumC[j]$ 也单调时，它满足决策单调性，后面会讲。

任务安排

link

题意同上

$1 \le N \le 3 \times 10^5,\ 0 \le S, C_i \le 512,\ -512 \le T_i \le 512$

任务安排

link

题意同上

$1 \leq N \leq 3 \times 10^5,\ 0 \leq S, C_i \leq 512,\ -512 \leq T_i \leq 512$

斜率不单调了，怎么办？(不满足决策单调性)

任务安排

link

题意同上

$1 \leq N \leq 3 \times 10^5,\ 0 \leq S, C_i \leq 512,\ -512 \leq T_i \leq 512$

斜率不单调了，怎么办? (不满足决策单调性)

在凸壳上二分！

任务安排加强版

link

题意同上

$1 \le N \le 3 \times 10^5,\ -512 \le S, C_i \le 512,\ -512 \le T_i \le 512$

任务安排加强版

link

题意同上

$1 \leq N \leq 3 \times 10^5, \ -512 \leq S, C_i \leq 512, \ -512 \leq T_i \leq 512$

横坐标也不单调了，怎么办?

任务安排加强版

link

题意同上

$1 \leq N \leq 3 \times 10^5,\ -512 \leq S, C_i \leq 512,\ -512 \leq T_i \leq 512$

横坐标也不单调了，怎么办?

用平衡树动态维护凸壳！[1]（你写 cdq 分治也没问题）

[1] 见 NOI2007 货币兑换

这里简单讲下 cdq 分治 (细节少)。

这里简单讲下 cdq 分治 (细节少)。

把所有询问的斜率和应插入的点离线下来，求出斜率，按斜率大小排序，保证斜率单调。

这里简单讲下 cdq 分治 (细节少)。

把所有询问的斜率和应插入的点离线下来，求出斜率，按斜率大小排序，保证斜率单调。

cdq 分治回溯时按横坐标排序 (归并排序)。

这里简单讲下 cdq 分治 (细节少)。

把所有询问的斜率和应插入的点离线下来，求出斜率，按斜率大小排序，保证斜率单调。

cdq 分治回溯时按横坐标排序 (归并排序)。

每次 cdq 时先 cdq 左区间，然后左边的 x 单调，用单调栈或单调队列维护凸壳，右边的斜率单调，于是直接可以 O(区间长度) 转移。

这里简单讲下 cdq 分治 (细节少)。

把所有询问的斜率和应插入的点离线下来，求出斜率，按斜率大小排序，保证斜率单调。

cdq 分治回溯时按横坐标排序 (归并排序)。

每次 cdq 时先 cdq 左区间，然后左边的 x 单调，用单调栈或单调队列维护凸壳，右边的斜率单调，于是直接可以 O(区间长度) 转移。

再 cdq 右区间。

时间复杂度 $O(n\log(n))$。

Cats Transport(CF311B)

题面太长了，放个 *link*

Cats Transport

对于每只猫，设 $A[i]=T[i]-\sum_{j=1}^{H[i]}D[j]$ ，一名饲养员如果想接到这只猫那么就要在 $>=A[i]$ 的时刻出发。如果在 t 时刻出发，则猫要等 $t-A[i]$ 。

Cats Transport

对于每只猫，设 $A[i] = T[i] - \sum_{j=1}^{H[i]} D[j]$ ，一名饲养员如果想接到这只猫那么就要在 $>= A[i]$ 的时刻出发。如果在 t 时刻出发，则猫要等 $t - A[i]$ 。

将 $A[]$ 排序，最优方案一个饲养员接的猫一定是按照 $A[]$ 排序后连续的若干只。

Cats Transport

证明如下：

有猫 i, k, j 且 $A[i] \leq A[k] \leq A[j]$，假设选了 i, j 而不选 k，那么需要在 $>= A[j]$ 的时间出发，那这时必然可以顺带上 k，答案不会更差，可能会更优。

Cats Transport

设 $F[i][j]$ 表示前 i 个饲养员带走了前 j 个猫的最小等待时间，记 $S[i]=\sum_{j=1}^{i}A[j]$ 。

Cats Transport

设 $F[i][j]$ 表示前 i 个饲养员带走了前 j 个猫的最小等待时间，记 $S[i]=\sum_{j=1}^{i}A[j]$ 。

$$F[i][j]=\min_{1\le k\le j}\{F[i-1][k]+(j-k)A[j]-(S[j]-S[k])\}$$

Cats Transport

设 $F[i][j]$ 表示前 i 个饲养员带走了前 j 个猫的最小等待时间，记 $S[i]=\sum_{j=1}^{i}A[j]$ 。

$$F[i][j]=\min_{1\le k\le j}\{F[i-1][k]+(j-k)A[j]-(S[j]-S[k])\}$$

把 i 看为定值，j 为状态，k 为枚举的决策点，发现式子中有和 j 有关的项与和 k 有关项的乘积，考虑用斜率优化。

Cats Transport

$$F[i-1][k]+S[k]=A[j]*k+F[i][j]-A[j]*j$$

决策点是平面上 $(k,\ F[i-1][k]+S[k])$ 的点，斜率为 $A[j]$ 要使截距最小化，维护一个下凸壳，由于 $A[j]$ 单增且 k 单增，用单调队列维护即可。

总结

形如 $F[i]=(\text{限制条件})\{F[j]+val(j,i)\}$ 的转移，$val(j,i)$ **包含了** i **与** j **乘积项**，那么大多可以写成直线的斜截式考虑斜率优化。

四边形不等式

若定义在整数域上的二元函数 $w(x,y)$，满足 $a \le b \le c \le d$ 且

$$w(a,d)+w(b,d) \ge w(a,c)+w(b,d)$$

那么称 $w(x,y)$ 满足四边形不等式。

另一种定义

若 $a < b$ 且

$$w(a, b+1) + w(a+1, b) \geq w(a, b) + w(a+1, b+1)$$

那么 $w(x, y)$ 满足四边形不等式。

另一种定义

若 $a < b$ 且

$$w(a, b+1) + w(a+1, b) \geq w(a, b) + w(a+1, b+1)$$

那么 $w(x, y)$ 满足四边形不等式。
当

$$w(a, b+1) + w(a+1, b) \leq w(a, b) + w(a+1, b+1)$$

时姑且叫它为" 反四边形不等式" ~~我也不知道叫什么~~，后面会用到，这里先定义清楚。

定义

对于形如 $F[i]=\min_{1\le j<i}\{F[j]+w(j,i)\}$ 的转移，令 $p[i]$ 为 $F[i]$ 取到最小值的 j 的值，若有 $\forall i'>i,\ p[i]\le p[i']$ ，则称 F 具有决策单调性。

定义

对于形如 $F[i]=\min_{1\le j<i}\{F[j]+w(j,i)\}$ 的转移，令 $p[i]$ 为 $F[i]$ 取到最小值的 j 的值，若有 $\forall i'>i$，$p[i]\le p[i']$，则称 F 具有决策单调性。

定理

若有 $F[i] = \min_{1 \le j < i}\{F[j] + w(j, i)\}$ 且函数 $w(j, i)$ 满足四边形不等式，则 F 具有决策单调性。

证明

$$F[i] = \min_{1 \le j < i} \{F[j] + w(j, i)\}$$

$$F[i] = F[p[i]] + w(p[i], i)$$

$w(i, j)$ 满足四边形不等式

证明

$$F[i] = \min_{1 \le j < i} \{F[j] + w(j, i)\}$$

$$F[i] = F[p[i]] + w(p[i], i)$$

$w(i, j)$ 满足四边形不等式

$$\begin{aligned}
&\forall j \in [1, p[i] - 1],\ \forall i' \in [i + 1, n], \\
&j < p[i] < i < i' \\
&F[j] + w(j, i) \ge F[p[i]] + w(p[i], i) - -1 \\
&w(j, i') + w(p[i], i) \ge w(j, i) + w(p[i], i') \\
&w(j, i') - w(j, i) \ge w(p[i], i') - w(p[i], i) - -2
\end{aligned}$$

证明

$$
\begin{aligned}
&\forall j \in [1, p[i]-1],\ \forall i' \in [i+1, n],\\
&j < p[i] < i < i'\\
&F[j] + w(j,i) \geq F[p[i]] + w(p[i], i) --1\\
&w(j,i') + w(p[i], i) \geq w(j,i) + w(p[i], i')\\
&w(j,i') - w(j,i) \geq w(p[i], i') - w(p[i], i) --2
\end{aligned}
$$

1,2 相加

$$F[p[i]] + w(p[i], i') \leq F[j] + w(j, i')$$

不难发现小于 $p[i]$ 的决策点对 i 后面的答案没有任何贡献，故 F 满足决策单调性。

当转移是取 max，且 w(i,j) 满足"反四边形不等式"时，它也有决策单调性。

诗人小 G

link

有一个长度为 n 的序列 A 与常数 L，P，你需要把它划分成若干段，每一段的代价为

$$\left|\left(\sum_{i=k+1}^{k+cnt} A_i\right) + cnt - 1 - L\right|^P$$

请你最小化代价之和。 $n \le 10^5$，$A[i] \le 30$，$L \le 3 \times 10^6$，$P \le 10$

诗人小 G

设 $F[i]$ 为前 i 个，划分为若干段的最小总代价，sum 为 A 数组的前缀和。

诗人小 G

设 $F[i]$ 为前 i 个，划分为若干段的最小总代价，sum 为 A 数组的前缀和。

$$F[i] = \min_{0 \le j < i} \left\{ F[j] + |sum[i] - sum[j] + i - j - 1 - L|^P \right\}$$

诗人小 G

设 $F[i]$ 为前 i 个，划分为若干段的最小总代价，sum 为 A 数组的前缀和。

$$F[i] = \min_{0 \leq j < i} \left\{ F[j] + |sum[i] - sum[j] + i - j - 1 - L|^P \right\}$$

令 $w(i, j) = |sum[i] - sum[j] + i - j - 1 - L|^P$，由于 P 过大，**关于 i，j 乘积的多项式次数过高**，斜率优化也不再适用，于是考虑决策单调性。

诗人小 G

$w(i,j)=|sum[i]-sum[j]+i-j-1-L|^P$

尝试判断 $w(i,j)$ 是否满足四边形不等式。

诗人小 G

$w(i,j) = |sum[i] - sum[j] + i - j - 1 - L|^P$
尝试判断 $w(i,j)$ 是否满足四边形不等式。
即证明 $\forall j < i,\ w(j,i+1) + w(j+1,i) \geq w(j,i) + w(j+1,i+1)$

诗人小 G

$w(i,j) = |sum[i] - sum[j] + i - j - 1 - L|^P$
尝试判断 $w(i,j)$ 是否满足四边形不等式。
即证明 $\forall j < i,\ w(j,i+1) + w(j+1,i) \geq w(j,i) + w(j+1,i+1)$

$$w(j,i+1) - w(j,i) \geq w(j+1,i+1) - w(j+1,i)$$

诗人小 G

$w(i,j) = |sum[i] - sum[j] + i - j - 1 - L|^P$
尝试判断 $w(i,j)$ 是否满足四边形不等式。
即证明 $\forall j < i,\ w(j,i+1) + w(j+1,i) \geq w(j,i) + w(j+1,i+1)$

$$w(j,i+1) - w(j,i) \geq w(j+1,i+1) - w(j+1,i)$$

记 $u = sum[i] - sum[j] + i - j - 1 - L$
记 $v = sum[i] - sum[j+1] + i - j - 1 - 1 - L =$
$sum[i] - sum[j] + i - j - 1 - L - a[j] - 1 = u - a[j] - 1$

诗人小 G

$u > v$, 只需证明 $|u + a[i]|^P - |u|^P \geq |v + a[i]|^P - |v|^P$ ，即证明离散函数 $f(x) = |x + c|^P - |x|^P$ 是增函数，$c > 0$ 。

诗人小 G

$u > v$, 只需证明 $|u+a[i]|^P - |u|^P \geq |v+a[i]|^P - |v|^P$ ，即证明离散函数 $f(x) = |x+c|^P - |x|^P$ 是增函数， $c > 0$ 。

当 $x+c \leq 0$, P 为偶数时， $f(x) = (x+c)^P - x^P$ ，$f'(x) = P(x+c)^{P-1} - Px^{P-1} > 0$ ，故 $f(x)$ 为增函数。

诗人小 G

$u>v$, 只需证明 $|u+a[i]|^P-|u|^P \geq |v+a[i]|^P-|v|^P$ ，即证明离散函数 $f(x)=|x+c|^P-|x|^P$ 是增函数， $c>0$ 。

当 $x+c \leq 0$, P 为偶数时，$f(x)=(x+c)^P-x^P$，$f'(x)=P(x+c)^{P-1}-Px^{P-1}>0$ ，故 $f(x)$ 为增函数。

其余情况类似可证明。

诗人小 G

- 如何实现?

诗人小 G

- 如何实现?
- 由于决策是单调的，于是我们想到了什么?

诗人小 G

- 如何实现?
- 由于决策是单调的，于是我们想到了什么?
- 单调队列

诗人小 G

- 如何实现?
- 由于决策是单调的，于是我们想到了什么?
- 单调队列
- 用单调队列维护决策点。

诗人小 G

每次求出 $F[i]$ 后，考虑 i 可能是哪些点 $i' > i$ 的决策点。

诗人小 G

每次求出 $F[i]$ 后，考虑 i 可能是哪些点 $i' > i$ 的决策点。

单调队列中存的是三元组 $(j,\ l,\ r)$ 表示 $[l, r]$ 在 $[1, i]$ 的决策点可能是 j 。

诗人小 G

每次求出 $F[i]$ 后，考虑 i 可能是哪些点 $i' > i$ 的决策点。

单调队列中存的是三元组 $(j,\ l,\ r)$ 表示 $[l, r]$ 在 $[1, i]$ 的决策点可能是 j 。

并时刻保证队列中的决策点由优越至劣，那么对于每个 $i \in [1, n]$，执行：

诗人小 G

1. 检查队头 (j_0, l_0, r_0)，若 $r_0 \le i-1$，那么显然这个决策对 i 无用，从队首弹出，否则令 $l_0 = i+1$。

2. 去队头 j 作为决策，计算 $F[i]$。

3. 尝试插入新决策 i，步骤如下：

(1) 拿出队尾 (j_t, l_t, r_t)

(2) 若 $F[j_t] + w(j_t, l_t) \ge F[i] + w(i, l_t)$，即对于 $F[l_t]$ 决策 i 比决策 j_t 优，记 $pos \leftarrow l_t$，那么弹出队尾，执行 (1)。

(3) 若 $F[j_t] + w(j_t, r_t) \le F[i] + w(i, r_t)$，即对于 $F[r_t]$ 决策 j_t 比 i 优，去步骤 (5)。

(4) 在 $[l_t, r_t]$ 上二分，求出位置 pos，再此之前决策 j_t 最优，在此之后决策 i 更优，去往步骤 (5)。

(5) 把三元组 (i, pos, N) 插入队尾。

然而单调队列维护需要能快速求出 $w(i,j)$ 不然复杂度还是承受不了，而且代码细节较多。

然而单调队列维护需要能快速求出 $w(i,j)$ 不然复杂度还是承受不了，而且代码细节较多。

在一些题目不要求转移在线时，即必须求出 i 才能继续转移时，可以使用分治等方法转移。

小 Q 的书架

将长度为 n 的排列划分为 k 段，代价为各段逆序对数量之和，最小化代价。
$n \le 40000,\ k \le min(10, n)$

小 Q 的书架

设 $F[i][j]$ 表示前 i 个分成 j 段的最小代价。

小 Q 的书架

设 $F[i][j]$ 表示前 i 个分成 j 段的最小代价。

$$F[i][j] = \min_{k<i}\{F[k][j-1] + w(k+1, i)\}$$

$w(k+1, i)$ 表示 $k+1$ 到 i 这一段的逆序对数量。

小 Q 的书架

设 $F[i][j]$ 表示前 i 个分成 j 段的最小代价。

$$F[i][j] = \min_{k<i}\{F[k][j-1] + w(k+1, i)\}$$

$w(k+1, i)$ 表示 $k+1$ 到 i 这一段的逆序对数量。
发现 $w(i, j)$ 满足四边形不等式。

小 Q 的书架

但是 $w(i,j)$ 的求解是 $O((j-i)\cdot\log(n))$ 的，单调队列不再适用。

小 Q 的书架

但是 $w(i, j)$ 的求解是 $O((j-i) \cdot \log(n))$ 的，单调队列不再适用。
而 $F[i][j]$ 的转移是不强制在线的，因为 $F[][j-1]$ 都已经求出来了。

小 Q 的书架

考虑整体二分，设计函数 $solve(L, R, l, r, j)$ 表示当前处理的状态为 $F[L \cdots R][j]$，它们的决策点分布在 $[l, r]$ 中。

小 Q 的书架

考虑整体二分，设计函数 $solve(L,R,l,r,j)$ 表示当前处理的状态为 $F[L\cdots R][j]$，它们的决策点分布在 $[l,r]$ 中。

每次二分 L,R 的 mid，然后求出 $F[mid][j]$ 的决策点 p，这一操作复杂度为 $O((r-l)\log(n))$，由于决策单调性，$[mid+1,R]$ 的决策点就在 $[p,r]$

小 Q 的书架

考虑整体二分，设计函数 $solve(L, R, l, r, j)$ 表示当前处理的状态为 $F[L \cdots R][j]$，它们的决策点分布在 $[l, r]$ 中。

每次二分 L, R 的 mid，然后求出 $F[mid][j]$ 的决策点 p，这一操作复杂度为 $O((r-l)\log(n))$，由于决策单调性，$[mid+1, R]$ 的决策点就在 $[p, r]$

接着递归左右 $solve(L, mid-1, l, p, j)$ 与 $solve(mid+1, R, p, r, j)$ 。

小 Q 的书架

递归 log 层，每层 $O(n\log_2(n))$

时间复杂度 $O(nk(\log_2(n))^2)$.

The End

Thanks!