

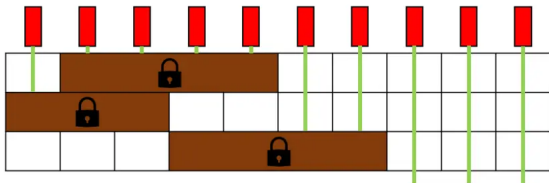
# DP 杂题选讲

张乐涛

2025 年 4 月 23 日

# P12196 [NOISG 2025 Prelim] Lasers 2

- 有一个  $h$  行  $w$  列组成的网格。每一行恰好包含一个上锁的滑动墙。
- 初始时，第  $i$  行的墙覆盖第  $l_i$  列到第  $r_i$  列，解锁它需要花费  $c_i$  美元。
- 墙一旦解锁，就可以在第  $i$  行上水平滑动到任意位置，只要它对齐在网格的边缘内即可。墙不能超出网格的左边缘或右边缘。
- 你可以至多花费  $k$  美元解锁并滑动激光墙，求空白列（所有格子都没有被激光墙覆盖）的最大数量。
- $1 \leq h, w \leq 2000, 0 \leq k \leq 10^9$ 。



洛谷

# P12196 [NOISG 2025 Prelim] Lasers 2

- 考虑将最终的局面划分成，若干个极长的，每一列都至少有一个滑动墙的列连续段。
- 设划分出来的连续段是  $[l_1, r_1], [l_2, r_2], \dots, [l_t, r_t]$ ，那么不用被解锁的滑动墙的花费是  $\sum_{i=1}^t g(l_i, r_i)$ ，其中  $g(l, r)$  表示所有满足  $l \leq l_i \leq r_i \leq r$  的滑动墙的  $c_i$  之和。
- 一种方案合法，当且仅当：
  - $\sum_{i=1}^h c_i - \sum_{i=1}^t g(l_i, r_i) \leq k$ ，即花费不能超过  $k$ ；
  - 令  $x = \max_{p=1}^h r_p - l_p + 1$ ，则  $\exists i \in [1, t], r_i - l_i + 1 \geq x$ ，因为要给最长的滑动墙留够位置放。

# P12196 [NOISG 2025 Prelim] Lasers 2

- 于是我们转而求满足条件的划分方案中， $\sum_{i=1}^t g(l_i, r_i)$  的最大值。
- 接下来的 DP 是自然的。设  $f_{i,j,0/1}$  表示，考虑到  $[1, i]$  这个前缀，其中有  $j$  列没有滑动墙，当前是否出现长度  $\geq x$  的连续段。
- 有两种转移：
  - 第  $i$  位留空，有  $f_{i,j,k} \leftarrow f_{i-1,j-1,k}$ ；
  - 第  $i$  位是一个连续段的右端点，枚举这个连续段的左端点  $l$ ，有  $f_{i,j,k \vee [i-l+1 \geq x]} \leftarrow f_{l-1,j,k} + g(l, i)$ 。
- 朴素转移时间复杂度为  $O(w^3)$ ，无法通过。

# P12196 [NOISG 2025 Prelim] Lasers 2

- 考虑优化。发现瓶颈在第二种转移，考虑在最外层枚举  $j$ ，那么转移形如  $f_i \leftarrow f_{j-1} + g(j, i)$ 。
- 考虑线段树优化。线段树每个下标  $j$  维护  $f_{j-1} + g(j, i)$ 。
- 枚举到  $i$  时加入所有  $r_k = i$  的滑动墙，即对  $j \in [1, l_k]$  区间加上  $c_k$ 。
- 计算  $f_i$  时求线段树区间最大值即可。

## 时间复杂度

设  $h, w$  同阶，时间复杂度  $O(w^2 \log w)$ 。

# AGC062B Split and Insert

- 有一个长度为  $n$  的排列  $a$ , 初始  $a_i = i$ 。
- 你要进行  $K$  次操作, 第  $i$  次操作选择一个整数  $0 \leq k_i < n$ , 将排列最后  $k_i$  个元素插入到前面的  $n - k_i$  个元素中, 要求这  $k_i$  个元素保持原来的相对顺序。
- 给定一个长度为  $K$  的序列  $c$ , 定义一个操作方案的花费为  $\sum_{i=1}^K k_i c_i$ 。
- 给定一个长度为  $n$  的排列  $p$ , 你需要保证进行  $K$  次操作后  $a$  和  $p$  相同。求最小花费。
- $2 \leq n \leq 100, 1 \leq K \leq 100$ 。

# AGC062B Split and Insert

- 考虑逆序操作。每次相当于取出一个子序列，放到后面。最终目标是将数组排成升序。
- 考虑区间 DP，设  $f_{k,i,j}$  为进行完第  $k \sim K$  次操作，已经把数组中值为  $i \sim j$  的数排好序。有转移：
  - $f_{k,i,j} \leftarrow f_{k+1,i,j}$ ，表示这次操作可以不动；
  - $f_{k,i,j} \leftarrow f_{k+1,i,x} + f_{k+1,x+1,j} + c_k \times (j-x), x \in [i,j]$ ，表示选权值在  $[x+1, j]$  之中的数，拎到最后面，花费是  $c_k \times (j-x)$ 。
- 初值是  $f_{K+1,i,j} = 0$ ，其中权值在  $[i,j]$  之中的数已经在原排列中排好序。答案就是  $f_{1,1,n}$ 。

## 时间复杂度

$O(n^3 K)$ 。

# CF2062F Traveling Salescat

- 有一个  $n$  个点的无向完全图，每个点有两个权值  $a_i, b_i$ 。 $i, j$  之间的边权为  $\max(a_i + b_j, b_i + a_j)$ 。一条路径的权值为其经过的边权之和。
- 对于  $k = 2, 3, \dots, n$ ，求经过恰好  $k$  个不同点的简单路径的最小权值。
- $2 \leq n \leq 3000$ 。



# CF2062F Traveling Salescat

- 首先先把边权变成  $b_i + b_j + \max(a_i - b_i, a_j - b_j)$ 。令  $a_i \leftarrow a_i - b_i$ ，边权变成  $b_i + b_j + \max(a_i, a_j)$ 。
- 考虑若确定了路径经过的点  $p_1, p_2, \dots, p_k$ ，如何排列它们能取得最小权值。
- 若所有  $b_i = 0$ ，那么显然是按  $a_i$  排序最优。
- 对于一般情况，发现  $\sum_{i=1}^{k-1} b_{p_i} + b_{p_{i+1}} = 2 \sum_{i=2}^{k-1} b_{p_i} + b_{p_1} + b_{p_k}$ ，所以  $p_1$  和  $p_k$  是比较重要的。

# CF2062F Traveling Salescat

- 考虑选出了起点  $p_1$  和终点  $p_k$  (不妨设  $a_{p_1} \leq a_{p_k}$ ), 那么  $\sum_{i=1}^{k-1} \max(a_{p_i}, a_{p_{i+1}})$  取到最小值时,  $p_2$  到  $p_{k-1}$  之间一定是按照  $a_i$  升序排序最优。
- 如果  $a_{p_k}$  都不是  $a_{p_i}$  中的最大值, 此时每个点对  $\sum_{i=1}^{k-1} \max(a_{p_i}, a_{p_{i+1}})$  的贡献为:  $a_i$  第二小值到最大值 (除了  $a_{p_k}$ ) 贡献 1 次, 最大值贡献 2 次。
- 如果  $a_{p_1}$  或  $a_{p_k}$  是最大值, 那么  $a_i$  第二小值到最大值都贡献 1 次。

# CF2062F Traveling Salescat

- 据此可以设计 DP。
- 将所有点按  $a_i$  升序排序，这样方便考虑每个点对答案的贡献系数。
- 设  $f_{i,j,0/1/2}$  表示考虑了  $[1, i]$  中的点，选出  $j$  个路径的点，路径的起点和终点都没被钦定 / 钦定了起点，未钦定终点 / 起点和终点都已被钦定，路径权值的最小值。
- 转移大力分类讨论即可，这里不展开讲了。


## 时间复杂度

$O(n^2)$ 。

# QOJ1250 Tokens<sup>1</sup>

- 给定一个  $A \times B \times C$  的立体棋盘。每个格子可以用三元组  $(i, j, k)$  描述，其中  $1 \leq i \leq A$ ,  $1 \leq j \leq B$ ,  $1 \leq k \leq C$ 。
- 起初， $(i, j, k)$  上有  $a_{i,j,k}$  个棋子。
- 每次操作，可以选择一个格子  $(i, j, k)$ ，满足  $(i, j, k)$  上至少有一个棋子，然后将这枚棋子移动到  $(i+1, j, k)$  或  $(i, j+1, k)$  或  $(i, j, k+1)$  中的一个。棋子不能移出棋盘边界。
- 目标是让  $(i, j, k)$  上有  $b_{i,j,k}$  个棋子。判断是否能够达成目标。
- $1 \leq A \leq 10^4$ ,  $1 \leq B, C \leq 6$ ,  $1 \leq a_{i,j,k}, b_{i,j,k} \leq 10^{12}$ 。

---

<sup>1</sup>一道双倍经验是 P11816 [PA 2019 Final] 摆棋 / Pionki. 

# QOJ1250 Tokens

- 考虑建二分图,  $(i, j, k)$  向全部满足  $i' \geq i, j' \geq j, k' \geq k$  的点  $(i', j', k')$  连边, 求所有初始棋子是否都能匹配一个目标棋子。
- 考虑 Hall 定理判定, 相当于选一些位置  $(i, j, k)$ , 求  $a_{i,j,k}$  的和减去所有  $b_{i',j',k'}$  的和是否有可能  $> 0$ , 其中  $(i', j', k')$  满足至少存在一个选的位置  $(i, j, k)$  满足  $i' \geq i, j' \geq j, k' \geq k$ 。
- 考虑求  $a_{i,j,k}$  的和减去所有  $b_{i',j',k'}$  的和的最大值。观察到若选了一个  $(i, j, k)$ , 那么把全部位置  $(x, y, z)$  (满足  $x \geq i, y \geq j, z \geq k$ ) 都选了一定不劣。
- 所以若设  $p_{i,j}$  表示  $(i, j, p_{i,j}), (i, j, p_{i,j} + 1), \dots, (i, j, C)$  都被选, 那么  $p$  满足  $p_{i,j} \geq p_{i+1,j}$  且  $p_{i,j} \geq p_{i,j+1}$ 。

# QOJ1250 Tokens

- 发现  $B$  很小，考虑轮廓线 DP。
- 设  $f_{i,j,p_1,p_2,\dots,p_C}$  表示考虑到格子  $(i,j)$ ,  $(i,1),\dots,(i,j),(i-1,j+1),\dots,(i-1,C)$  的  $p$  的取值分别为  $p_1,p_2,\dots,p_C$  的方案数。
- 转移枚举  $(i,j)$  的  $p$  的取值即可。

## 时间复杂度

$p$  的状态数是  $\binom{B+C}{B}$ ，时间复杂度  $O(ABC\binom{B+C}{B})$ 。

# ARC159F Good Division

- 定义一个序列是好的当且仅当可以每次删去一对相邻不同的数把序列删空。
- 现在给定一个长度为  $2n$  的序列  $a$ ，求有多少种划分方式，使得每一段都是好的。
- 答案对 998244353 取模。
- $1 \leq n \leq 5 \times 10^5$ 。

# ARC159F Good Division

- 先考虑判定一个序列是否是好的。

## 结论

一个序列  $b$  是好的，当且仅当  $|b|$  为偶数，且不存在出现次数  $> \frac{|b|}{2}$  的数（即不存在绝对众数）。



# ARC159F Good Division

- 考虑 DP。设  $f_i$  表示前缀  $[1, 2i]$  有多少种划分方式使得每一段都是好的。
- 转移是：

$$f_i = \sum_{j=1}^i [a_{2j-1}, a_{2j}, \dots, a_{2i-1}, a_{2i} \text{ 不存在绝对众数}] f_{j-1}$$

- 考虑 cdq 分治优化 DP。
- 设当前分治区间为  $[l, r]$ ,  $mid = \lfloor \frac{l+r}{2} \rfloor$ 。
- 我们要计算  $j \in [l, mid]$  对  $i \in [mid+1, r]$  的贡献。

# ARC159F Good Division

- 考虑先加上全部  $f_j$ ，再减去存在绝对众数的部分。
- 显然一个数要成为  $[j, i]$  的绝对众数，必须满足它在  $[j, mid]$  是绝对众数或在  $[mid + 1, i]$  是绝对众数。

## 结论

对于任意一个长度为  $m$  的序列，它的所有不同的前缀绝对众数数量是  $O(\log m)$  的。

## 证明

每次出现一个不同的绝对众数，所需要的区间长度翻倍。

# ARC159F Good Division

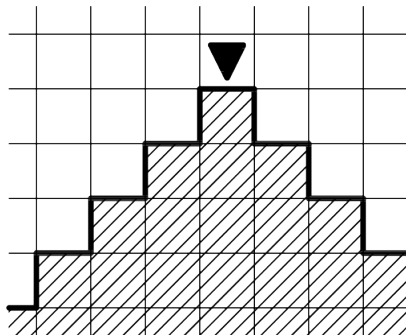
- 所以我们求出  $[l, mid]$  所有后缀的绝对众数和  $[mid + 1, r]$  所有前缀的绝对众数。
- 枚举这  $O(\log(r - l + 1))$  个绝对众数，将序列中等于这个数的权值赋为 1，其他数权值赋为  $-1$ 。
- 做个前缀和，设前缀和数组为  $s$ ，那么转移条件变成  $s_i > s_{j-1}$ 。
- 容易前缀和优化做到单次线性。

## 时间复杂度

cdq 分治一个  $\log$ ，绝对众数数量一个  $\log$ ，总时间复杂度  $O(n \log^2 n)$ 。

# Gym102443E Hide-and-Seek for Robots

- 一个  $n \times m$  的网格上有一些波特，每个波特有一个上下左右的朝向。
- 一个朝下的波特的视野范围如图：



- 三角形是波特的所在位置。阴影部分是它的探测范围。

# Gym102443E Hide-and-Seek for Robots

- 你可以转动一个波特的朝向，只能正好转动一个直角，且一个波特最多转一次。
- 转完之后，要求两个波特不能互相看到。
- 求最少转动多少个波特的朝向使得要求被满足。
- 无解输出  $-1$ ，有解则要构造一个转的方案。
- $1 \leq n, m \leq 2000$ 。

# Gym102443E Hide-and-Seek for Robots

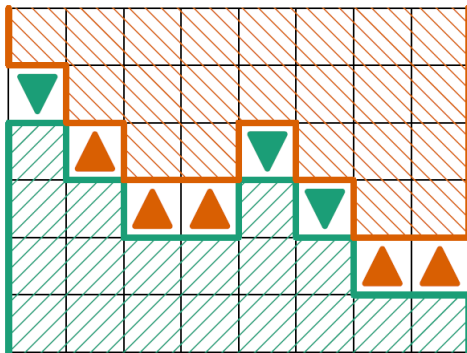
## 关键的观察

若一个波特被转了，那么一定存在一个方向合法。所以我们可以把转了的波特当做被删了。

- 所以现在的问题变成：保留最多的波特，使得波特不会互相看见。
- 然后注意到向上下和向左右不会存在互相看见的情况。而上下，向下看见向上的，那么这个向上的肯定也能看见向下的。它们是对称的。
- 我们先考虑上下的问题。我们需要找出，最大的不会相互影响的上下子集。

# Gym102443E Hide-and-Seek for Robots

- 感受一下发现找到的上下子集，存在一条线把他们分开，上面的区域是向上的部分，下面的部分是向下的部分。
- 并且线在每一列的位置与前一列的位置差不会超过一格。



# Gym102443E Hide-and-Seek for Robots

- 更形式化的表述就是，对于每一列存在一个分界点  $p_i$ ，使得位于  $(1, i), (2, i), \dots, (p_i, i)$  的波特都是向上的，位于  $(p_i, i), (p_i + 1, i), \dots, (n, i)$  的波特都是向下的。
- 并且  $p$  满足  $|p_i - p_{i+1}| \leq 1$ 。
- 考虑 DP。设  $f_{i,j}$  为考虑了  $[1, i]$  中的列， $p_i = j$ ，最多保留多少个波特。
- 转移直接枚举  $p_{i+1}$  的  $O(1)$  个取值即可。
- 对于左右的问题也是同理，对所有行再做一遍同样的 DP 即可。



# Gym102443E Hide-and-Seek for Robots

- 至于构造方案，若一个波特需要被转，假设这个波特原本是上下朝向，现在要变成向左或向右。
- 通过 DP 值倒推出每一行的分界点，若它所在列比分界点小那么变成向左，否则变成向右。
- 其余情况同理。

## 时间复杂度

$O(nm)$ 。

# ARC150F Constant Sum Subsequence

- 有一个长度为  $n^2$  的序列  $a$ 。给定  $a_1, a_2, \dots, a_n$ ，序列  $a$  满足  $\forall i \in [1, n^2 - n], a_i = a_{i+n}$ 。
- 再给定一个正整数  $S$ 。你要求出最小的整数  $p$ ，满足所有和为  $S$  的正整数序列是  $[a_1, a_2, \dots, a_p]$  的子序列。
- $1 \leq n \leq 1.5 \times 10^6$ ,  $1 \leq S \leq \min(n, 2 \times 10^5)$ ,  $1 \leq a_i \leq S$ ，保证  $1 \sim S$  中的所有整数在  $a_1, a_2, \dots, a_n$  中都出现至少一次。

# ARC150F Constant Sum Subsequence

## 一些定义

定义  $\text{nxt}(i, x)$  为最小的  $j$  满足  $a_j = x$  且  $j > i$ ,  $\text{pre}(i, x)$  为最大的  $j$  满足  $a_j = x$  且  $j < i$ 。

- 考虑 DP, 设  $f_i$  为  $S = i$  时的最小的  $p$ 。
- 转移枚举下一个数填什么, 有:

$$f_{i+j} \leftarrow \max(f_{i+j}, \text{nxt}(f_i, j))$$

- 暴力 DP 时间复杂度  $O(S^2 \log n)$ , 无法通过。

# ARC150F Constant Sum Subsequence

- 还是考虑 **cdq** 分治优化。设当前区间为  $[l, r]$ ，令  $mid = \lfloor \frac{l+r}{2} \rfloor$ 。
- 考虑  $[l, mid]$  对  $[mid+1, r]$  的贡献。
- 还是先枚举一个  $j$ ，由  $f$  的转移式可得  $f$  有单调性，则  $\forall i \in [mid+1, r], f_i > f_{mid}$ 。
- 则我们只需要考虑可能对右区间有贡献的  $i$ ，即  $i$  满足  $nxt(f_i, j) = nxt(f_{mid}, j)$ 。这些  $i$  形成了一段区间，而这个区间的左端点就是最小的  $k$  满足  $f_k \geq pre(f_{mid}, j)$ 。
- 于是对这段区间内的  $i$ ，令  $f_{i+j} \leftarrow \max(f_{i+j}, nxt(f_{mid}, j))$ 。
- 我们需要一个区间取  $\max$ ，单点查询的数据结构。可以用线段树。

## 时间复杂度

$O(n + S \log S (\log S + \log n))$ 。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 塞一点私货←
- 对于一个长度为  $k$  的正整数序列  $c$ ，如下定义  $f(c)$ ：
  - A 和 B 在这个序列上玩游戏。A 先手，交替进行操作。
  - 游戏过程如下：
    - 设序列当前的长度为  $m$ 。若  $m = 1$ ，则游戏结束。
    - 如果游戏没有结束且轮到 A 操作，那么 A 必须选择一个  $1 \leq i \leq m - 1$ ，将  $c_i$  赋值为  $\min(c_i, c_{i+1})$ ，并且删除  $c_{i+1}$ 。
    - 如果游戏没有结束且轮到 B 操作，那么 B 必须选择一个  $1 \leq i \leq m - 1$ ，将  $c_i$  赋值为  $\max(c_i, c_{i+1})$ ，并且删除  $c_{i+1}$ 。
  - A 希望最大化最后  $c_1$  的值，而 B 希望最小化最后  $c_1$  的值。
  - $f(c)$  为当双方都采取最优策略时，游戏结束后  $c_1$  的值。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 给定一个  $1 \sim n$  的排列  $a$ ，其中有一些位置丢失了，变成了 0。
- 你需要补充完整这个排列，并且使得

$$\sum_{i=1}^n \sum_{j=i}^n f([p_i, p_{i+1}, \dots, p_j])$$

- 最大化。
- 输出这个最大值。
- $1 \leq n \leq 5 \times 10^5$ 。 **Easy Version:**  $1 \leq n \leq 5000$ 。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 首先，通过打表或观察可以发现：

$$f(c) = \begin{cases} \min(c_1, c_k) & k \bmod 2 = 0 \\ \max(c_1, c_k) & k \bmod 2 = 1 \end{cases}$$

- 对称地设  $g(c)$  为调换先后手顺序后的答案，那么：

$$g(c) = \begin{cases} \max(c_1, c_k) & k \bmod 2 = 0 \\ \min(c_1, c_k) & k \bmod 2 = 1 \end{cases}$$

## 证明

考虑归纳。 $k \leq 2$  显然成立。对于  $k \geq 3$ ，考虑  $f(c)$  的计算。假设  $k$  为奇数。若先手选择  $2 \leq i \leq k-2$  递归到  $g(c')$ ，那么答案为  $\max(c_1, c_k)$ ；若先手选择  $i=1$  或  $i=k-1$ ，那么答案  $\leq \max(c_1, c_k)$ 。其余情况同理。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 所以对于一个排列  $p_1, p_2, \dots, p_n$ :

$$\begin{aligned} & \sum_{i=1}^n \sum_{j=i}^n f([p_i, p_{i+1}, \dots, p_j]) \\ &= \sum_{i=1}^n \sum_{j=i}^n \left( [i \bmod 2 = j \bmod 2] \max(p_i, p_j) + [i \bmod 2 \neq j \bmod 2] \min(p_i, p_j) \right) \\ &= \sum_{i=1}^n \sum_{j=i}^n \max(p_i, p_j) - [i \bmod 2 \neq j \bmod 2] (\max(p_i, p_j) - \min(p_i, p_j)) \\ &= \sum_{i=1}^n i^2 - \sum_{i=1}^n \sum_{j=i+1}^n [i \bmod 2 \neq j \bmod 2] |p_i - p_j| \end{aligned}$$



# CF2084G2 Wish Upon a Satellite (Hard Version)

- 所以现在的问题变成了最小化以下式子：

$$\sum_{i=1}^n \sum_{j=i+1}^n [i \bmod 2 \neq j \bmod 2] |b_i - b_j|$$

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 这个问题可以进一步变成：有一个  $1 \sim n$  的数轴，点  $i$  可能是黑色、白色或未确定。
- 若  $i$  在  $a$  中的位置为奇数就是黑色，为偶数就是白色，在  $a$  中未出现则为不确定。
- 我们现在要把未确定的点染成黑色或者白色，使得：
  - 恰好有  $\lceil \frac{n}{2} \rceil$  个黑点；
  - 最小化所有不同颜色点对的距离之和。
- 所有不同颜色点对的距离之和可以变成：对于所有  $1 \leq k < n$ ， $1 \sim k$  中的黑点数量乘  $k+1 \sim n$  的白点数量，加上  $1 \sim k$  中的白点数量乘  $k+1 \sim n$  的黑点数量之和。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 这个简化后的问题可以 DP 了。
- 设  $f_{i,j}$  表示已经确定了  $1 \sim i$  中的点的颜色，其中有  $j$  个黑点，对于所有  $1 \leq k < i$  上述式子的和的最小值。
- 至于转移，考虑先将  $f_{i,j}$  加上  $k=i$  时上述式子的值，即将所有  $f_{i,j}$  加上  $j \times (\lfloor \frac{n}{2} \rfloor - (i-j)) + (i-j) \times (\lceil \frac{n}{2} \rceil - j)$ 。
- 然后考虑第  $i+1$  个点 是黑点或者白点。
  - 若第  $i+1$  个点可以是白点，有转移  $f_{i+1,j} = \min(f_{i+1,j}, f_{i,j})$ ；
  - 若第  $i+1$  个点可以是黑点，有转移  $f_{i+1,j+1} = \min(f_{i+1,j+1}, f_{i,j})$ 。
- 最后的答案即为  $\sum_{i=1}^n i^2 - f_{n, \lceil \frac{n}{2} \rceil}$ 。
- 暴力 DP 时间复杂度  $O(n^2)$ ，可以通过 Easy Version。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 至于 Hard Version, 考虑我们每次要对 DP 数组进行什么操作:
  - 对于所有  $j$ , 将  $f_{i,j}$  加上
$$j \times (\lfloor \frac{n}{2} \rfloor - (i - j)) + (i - j) \times (\lceil \frac{n}{2} \rceil - j) = 2j^2 + (-2i - (n \bmod 2))j + i \cdot \lceil \frac{n}{2} \rceil$$
 即关于  $j$  的一个二次函数;
  - 执行  $f_{i+1,j} = \min(f_{i+1,j}, f_{i,j})$ ;
  - 执行  $f_{i+1,j+1} = \min(f_{i+1,j+1}, f_{i,j})$ 。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 可以归纳地证明  $f_{i,j}$  下凸，即  $f_i$  的差分数组是不降的。
- 考虑经典 slope trick，即维护  $f_{i,j}$  的差分数组即  $g_{i,j} = f_{i,j+1} - f_{i,j}$ 。
- 那么现在操作变成：
  - 对于所有  $j$ ，将  $g_{i,j}$  加上一个关于  $j$  的一次函数；
  - 将  $g_{i,j}$  整体向右平移一位；
  - 向  $g_i$  中插入一个 0，并保持  $g_i$  有序。

# CF2084G2 Wish Upon a Satellite (Hard Version)

- 考虑用 Treap 维护  $g_{i,j}$ 。
- Treap 的每个结点维护对应的  $g_{i,j}$  的值以及下标  $j$  的值。
- 那么操作都可以改写成  $(g_{i,j}, j)$  的线性的变换，可以用向量和矩阵维护。

## 时间复杂度


$O(n \log n)$ 。

# P8984 [北大集训 2021] 末日魔法少女计划<sup>2</sup>

- 给定正整数  $n, k$ , 构造一个只含 0, 1 的矩阵  $A_{i,j}$ ,  $0 \leq i, j \leq n$ , 满足:
  - $A_{i,i} = 1$ ;
  - $A_{i,i+1} = 1$ ;
  - 对  $i > j$  有  $A_{i,j} = 0$ ;
  - 若  $A_{i,j} = 1$ ,  $j - i > 1$ , 则存在  $i < t < j$ , 满足  $A_{i,t} = A_{t,j} = 1$ ;
  - 对  $i \leq j$  有  $(A^k)_{i,j} > 0$ .
- 你需要输出满足  $A_{i,j} = 1$  且  $j - i > 1$  的每个  $(i, j)$ 。设这样的  $(i, j)$  共有  $m$  个, 若  $m \leq nf(k)$  则该测试点获得满分。

$k$	2	3	4	5	6	7	8
$f(k)$	7.9870	3.8085	2.3960	1.9610	1.6065	1.4515	1.2540

$k$	9	10	11	12	13	14	15
$f(k)$	1.1980	1.0995	1.0705	1.0345	1.0120	1.0015	0.9940

<sup>2</sup>一道双倍经验是 P6020 [Ynoi2010] Exponential tree. 

# P8984 [北大集训 2021] 末日魔法少女计划

- 可以将题意转化成：
  - 有一个有向图，初始只有边  $i \rightarrow i+1$ 。
  - 需要添加  $m$  条  $u < v$  的有向边  $u \rightarrow v$ ，使得任意两点都可以在  $k$  步内到达；
  - 对于每条边  $u \rightarrow v$  满足  $u+1 < v$ ，都存在  $w \in (u, v)$  使得存在边  $u \rightarrow w$  和  $w \rightarrow v$ 。



# P8984 [北大集训 2021] 末日魔法少女计划

- 考虑  $k \geq 3$ 。
- 拓展  $k = 2$  的思路，设置若干个关键点  $p_1, p_2, \dots, p_c$ 。
- 对于每个段  $(p_i, p_{i+1})$  的所有点，向  $p_i$  和  $p_{i+1}$  连边。
- 对于两端的所有点  $[1, p_1)$  和  $(p_c, n]$  也分别向  $p_1$  和  $p_c$  连边。
- 这样若  $p_1, p_2, \dots, p_c$  两两都可以在  $k - 2$  步内到达，那么我们就满足了跨过至少一个关键点的点对的条件。
- 可以发现  $[1, p_1), (p_i, p_{i+1}), (p_c, n]$  是  $k' = k$  的子问题， $[p_1, p_2, \dots, p_c]$  是  $k' = k - 2$  的子问题。

# P8984 [北大集训 2021] 末日魔法少女计划

- 题目给出的  $f(k)$  没什么规律，所以题目可能是想让我们求出  $m$  的最优解（或者一个很优的解）。
- 观察到子问题的结构后可以设计 DP。
- 设  $f_{k,n}$  为要求  $n$  个点要在  $k$  步内互相到达，且已经连了所有  $i \rightarrow i+1$  的边，还要连的边数量的最小值。
- 处理  $k \leq 2$  的边界。至于转移，发现令关键点之间的间隔  $p_{i+1} - p_i$  全部相等看起来很优。
- 所以我们枚举关键点的个数和间隔，那么关键点到两侧的距离  $p_1 - 1$  和  $n - p_c$  显然均分比较优。
- 仔细地列出连边数量的式子后就可以转移，不要忽略常数。
- 构造方案就直接倒推，模拟一遍递归即可。

# P8984 [北大集训 2021] 末日魔法少女计划

- 于是你按照上述思路写了一发，获得了 99.899979 分。
- 怎么回事呢？关键点的间隔全部相等并不是最优的。
- 于是当  $n \leq 100$  时换一种转移方式：枚举  $p_c - p_1$  和关键点的个数，关键点之间的距离尽量均分，就能过了。

## 时间复杂度

关键点的间隔和数量乘积需要  $\leq n$ ，所以枚举这两个东西的时间复杂度是  $O(n \log n)$ 。总时间复杂度  $O(kn^2 \log n)$ 。

- 做完这题可以去做 P11366 [Ynoi2024] 末日的魔法少女计划，两题的想法非常相似（