DP 选讲

wlx

2025 年 3 月 28 日

CF1188D

- 给定 n 个数字 $a_1, a_2, \cdots, a_n$。
- 每次，你可以将某个 a_i 加上 2^x，要求 $x \in \mathbb{N}$。
- 问使 a_i 全部相等所需的最小操作次数。
- $n \leq 10^5, a_i \leq 10^{17}$

CF1188D

先将 a 从小到大排序。假设最后 a_i 全变成了 $a_n + x, x \geq 0$，则总操作次数为 $\sum_{i=1}^{n} \text{popcount}(x + a_n - a_i)$。显然存在最优的 x 满足 $x < 2^{60}$。

考虑从低往高进行数位 DP。记 $b_j = a_n - a_j \geq 0$，则在 2^i 位，1 的个数除了受 x 和 b_j 的第 i 位影响，还受 $x + b_j$ 是否进位影响。而 DP 时直接记录产生了进位的 b_j 是不现实的。

注意到一点：如果只看 b_j 的前 $i-1$ 位，则会产生进位的一定是最大的若干个 b_j。于是，我们只用设 $f_{i,j}$ 表示考虑到 x 的前 i 位，前 i 位最大的 j 个 b 发生了进位的情况下，操作次数最小是多少。

正常转移即可，时间复杂度 $O(n \log n \log V)$。使用基数排序可做到 $O(n \log V)$。

CF933E

- 给定一个长为 n 的序列 p。
- 每次，你可以选择两个相邻的数 p_i, p_{i+1}，将它们同时减去 $\min\{p_i, p_{i+1}\}$。这一操作的代价也是 $\min\{p_i, p_{i+1}\}$。
- 问使得 p 中不存在相邻正整数的最小代价。你需要给出方案。
- $n \leq 3 \times 10^5, 0 \leq p_i \leq 10^9$

CF933E

考虑一个新的问题：每次你可以花费 x 代价将 p_i, p_{i+1} 同时减小 x，求最终不出现相邻正整数的最小代价。

仅考虑 $[l, r]$ 内的所有 p_i，先计算使 $p_i \le 0, \forall i \in [l, r]$ 的最小代价 $w_{l,r}$。

递归定义 $c_l = p_l, c_i = \max\{p_i - c_{i-1}\}, \forall i \in [l+1, r]$，则有：

$$\begin{aligned} w_{l,r} &= \sum_{i=l}^{r} c_i = \sum_{i=1}^{r-2} c_i + c_{r-1} + \max\{p_r - c_{r-1}, 0\} \\ &= w_{l,r-2} + \max\{p_r, c_{r-1}\} \ge w_{l,r-2} + w_{r,r} \end{aligned}$$

CF933E

由于在确定最终非正的位置之后，最小代价恰为 $\sum w_{l,r}$，其中 $[l,r]$ 为极长非正连续段。于是，我们可以断言段长均 ≤ 2。

在这种情况下，不难证明，最优解一定可以通过原操作完成。又由于新问题相较原问题更松，简单 DP 求出新问题的最小代价即为答案。方案构造也很简单。时间复杂度 $O(n)$。

CF708E

- 有 $(n+2) \times m$ 个方块排成 $n+2$ 行 m 列。
- 每一天，除了第一行和最后一行，每行最左边和最右边的方块都有 p 的概率被摧毁。特别地，若某行只有一个方块，则该方块被摧毁的概率为 $1-(1-p)^2$。
- 求 k 天后所有剩下的方块四连通的概率。
- $1 \leq n, m \leq 1500$，$0 \leq k \leq 10^5$。

CF708E

先预处理出一个长为 i 的前缀/后缀被摧毁的概率 q_i 及其前缀和 sq_i。显然每行会被删到只剩一个区间 $[l_i, r_i]$。

比较传统的方法是使用 f_{i,l_i,r_i} DP，然后对转移式和状态进行优化。这里介绍另一种方法。

我们发现状态多的原因是考虑了区间的两个端点，如果是计算路径，状态数就可以减少一维。

因此，我们将每个连通状态映射到以下路径：从第 0 行最左边开始，能向下走就向下走，反之向左/右走到最近能向下走的位置。

CF708E

设 $f_{i,j}$ 表示从第 $i-1$ 行进入第 i 行的位置为 j，且在第 $i-1$ 行向右走了，仅考虑前 $i-1$ 行的概率；$g_{i,j}$ 表示向左走了的概率；$h_{i,j}$ 表示直接向下走的概率。转移形如：

$$f_{i,j} = \sum_{l=1}^{j-1} (f_{i-1,l} \times q_{l-1} + h_{i-1,l} \times sq_{l-1}) \times sq_{m-j}$$

$$g_{i,j} = \sum_{l=j+1}^{m} (g_{i-1,l} \times q_{m-l} + h_{i-1,l} \times sq_{m-l}) \times sq_{j-1}$$

$$h_{i,j} = q_{j-1} \times f_{i-1,j} \times sq_{m-j} + sqj - 1 \times g_{i-1,j} \times q_{m-j} + sq_{j-1} \times h_{i-1,j} \times sq_{m-j}$$

前缀和优化一下，时间复杂度 $O(nm)$。

CF1608F

- 给定 n, k 和长度为 n 的整数序列 b。
- 求有多少个长度为 n 的整数序列 a 满足：$0 \le a_i \le n$ 且 $|\operatorname{mex}(a_1, a_2, \ldots, a_i) - b_i| \le k, \forall i \in [1, n]$。
- $n \le 2000$，$k \le 50$，$-k \le b_i \le n + k$。对 998244353 取模。

CF1608F

先记 $f_{i,j}$ 为填到了 a_i，前面数的 mex 为 j 的方案数。直接确定 $> j$ 的数的具体值不利于转移，故我们考虑延后确定。这样只用新增一维 l，表示前面共有 l 种 $> j$ 的数。

接下来考虑转移。若 $a_i \neq j$，则对 mex 没有影响，转移形如：

$$f_{i,j,l} \leftarrow f_{i-1,j,l} \times (j+l) + f_{i-1,j,l-1} \times l$$

若 $a_i = j$，我们可以枚举 mex 增加多少，再将之前 l 种数中最小的几个依次赋为 $j+1, j+2, \cdots$，转移形如：

$$f_{i,j,l} \leftarrow \sum_{p=1}^{j} f_{i-1,j-p,l+p-1}$$

转移可以前缀和优化，时间复杂度 $O(n^2k)$。

LOJ3276

- 有 $2n$ 根石柱排成一行。$\forall k \in [1, n]$，恰有 2 根石柱的高度为 k。
- 随后发生了 n 次地震。每次地震时，$\forall k \in [1, n]$，人们将保护编号最大的高度为 k 的石柱。所有未被保护的石柱高度减 1，被保护的高度不变。
- n 次地震后，只有 n 根石柱的高度至少为 1。给定这 n 根石柱的下标，求初始时所有可能的石柱高度序列的数量。
- $n \leq 600$。对 $10^9 + 7$ 取模。

LOJ3276

设高度序列为 h，需要一种有前景的方式来求出 n 次地震后 h 的变化。一种方法是，从后往前依次考虑每根石柱 i，若存在 $j>i, h_j=h_i$，就将 h_i 减一，并重复此过程。

如果记 t_i 满足对于所有 $l\le t_i$，最后都存在 $j>i, h_j=l$ 且不存在 $j>i, h_j=t_i+1$，则石柱 i 消失当且仅当 $h_i\le t_i$。

类似的，设 $f_{i,j}$ 表示考虑到倒数第 i 个石柱，且 $t_i=j$，后面最终高度 $\ge j+2$ 的石柱的具体高度未定的方案数。考虑如何转移。记 c_0 表示后 $i-1$ 根石柱中最后消失的数量，c_1 表示最后存在的数量。为便于计算，下面均区分相同的初始高度。

若石柱 i 最后消失，则其高度 $h_i\le j$，显然后面最终高度 $\le j$ 的石柱，初始高度也 $\le j$。则转移为 $f_{i,j}\leftarrow f_{i-1,j}\times(j-c_0)$。

LOJ3276

若石柱 i 最后存在，其初始/最终高度均 $> j$。由于要在此计算方案，我们对最终的 h_i 进行讨论：若 $h_i \geq j+2$，我们延后其贡献的计算，转移为 $f_{i-1,j} \leftarrow f_{i,j}$。

若 $h_i = j+1$，枚举 $t_i = k$，并考虑 $f_{i-1,j}$ 向 $f_{i,k}$ 的贡献系数。显然初始 $h_i \in [j+1, k]$，i 后面共有 $c_1 - j$ 个石柱最终高度待定。先算上选出 $k-j-1$ 个最终高度 $\in [j+2, k]$ 的石柱方案数 $\binom{c_1-j}{k-j-1}$，它们的初始高度同样 $\in [j+2, k]$。于是初始可能的 h_i 有 $k-j+1$ 种选择。

唯一的限制是，选出来的 $k-j-1$ 的石柱的最终高度要恰好覆盖 $j+2, j+3, \cdots, k$。我们还需要这部分的方案数 g_{k-j-1}。转移为 $f_{i,k} \leftarrow f_{i-1,k} \times (k-j+1)\binom{c_1-j}{k-j-1} g_{k-j-1}$。

LOJ3276

考虑计算 g。显然，g_i 表示为 i 根石柱选择 $[1,i]$ 的高度，每种高度出现不超过两次，最终没有石柱消失的方案数。

枚举最左侧石柱的最终高度 k，其初始高度 $\in [i,k]$，共有 $i-k+2$ 种选择。后面 $>k$ 和 $<k$ 的石柱互不干扰，于是 $g_i=\sum_{k=1}^{i}(i-k+2)\binom{i-1}{k-1}g_{k-1}g_{i-k}$。

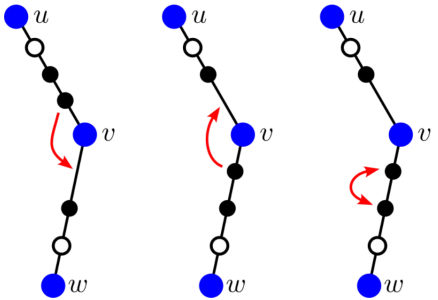
总时间复杂度 $O(n^3)$。

CF2081E

- 给定一棵 $n+1$ 个结点的树，根节点为 0，其唯一子节点为 1。
- 有 m 个黑白棋子，可以在这棵树的边上移动。第 i 个棋子的颜色为 c_i，移动范围限制在根结点到点 d_i 的路径上。
- 初始时，所有棋子顺序放置在 0 到 1 的边上。你可以通过如下操作移动棋子：
- 交换一条边上相邻两个同色棋子的位置；
- 对于结点 u，设 v 是 u 的父亲，w 是 v 的父亲。将边 (v,w) 上最靠近 v 的棋子移动到边 (u,v) 最靠近 v 的位置上；
- 类似的，将边 (v,u) 上最靠近 v 的棋子移动到边 (w,v) 最靠近 v 的位置上。
- 最终，你需要将所有棋子移动回边 $(0,1)$ 上。这些棋子从 0 到 1 按顺序形成一个排列。

CF2081E

- 问总共有多少种不同的排列，可以在有限次操作内得到。
- $n \leq 5000$。对 998244353 取模。三种操作示例如下：



CF2081E

第一步是将所有棋子尽可能向下移。不难证明，当每个棋子都到达它可能到达的最深的边上时，可以只通过上移操作和交换操作，得到所有可能的排列。

下面考虑计数。设 $f_{u,x,c}, c = 0/1$ 表示 u 子树内的所有棋子移动到边 (u, fa_u) 上时，最顶端为长度为 x，颜色为 c 的极长段的方案数。这样，在与原本位于 (u, fa_u) 上的棋子合并时，只用乘上 $\binom{x+y}{x}$ 的系数，y 表示 (u, fa_u) 最底端颜色为 c 的棋子数量。

考虑加入新子树 v 时如何转移。设 s_u 为当前 u 子树内的棋子数，s_v 同理。分两种情况讨论：最顶端颜色相同和颜色不同。

对于颜色不同的情况，假设先上移 u 子树的棋子（先上移 v 同理），则转移形如

$$f_{u,i,c} \leftarrow \left(\binom{s_u - i + s_v}{s_v} f_{u,i,c} + \binom{s_u - i + s_v - 1}{s_v - 1} \sum_{j=i+1}^{s_u} f_{u,j,c} \right) \left(\sum_{j=1}^{s_v} f_{v,j,1-c} \right)$$

CF2081E

对于颜色相同的情况，先特判只有一种颜色的可能，然后假设上移的第一个异色棋子在 u 子树内（v 子树同理），则转移形如

$$f_{u,i+j,c} \leftarrow \binom{i+j}{i}\binom{s_u-i-1+s_v-j}{s_v-j} f_{u,i,c} \sum_{l=j}^{s_v} f_{v,l,c}$$

总时间复杂度 $O(n^2)$。

LOJ3124

- 给定一棵 n 个点的树，第 i 条边为 (u_i, v_i)。第 i 个点的权值为 $W_i \in [1, 3]$。
- W_i 按如下方式确定：每个点 i 有三个参数 $p_{i,1}, p_{i,2}, p_{i,3}, p_{i,1} + p_{i,2} + p_{i,3} = 1$。对于 $j = 1, 2, 3$，W_i 有 $p_{i,j}$ 的概率为 j
- 所有 W_i 确定后，一个人开始随机选点。每次选到点 i 的概率为 $\frac{W_i}{\sum_j W_j}$，直到所有点都被选中过至少一次。记点 i 第一次被选中的时间为 T_i。
- 求对于所有边 (u_i, v_i) 都有 $T_{u_i} < T_{v_i}$ 的概率。
- $n \leq 1000$。对 998244353 取模。

LOJ3124

先假设边 (u_i, v_i) 的方向是 $u_i \to v_i$。考虑外向树的情况怎么做。

设 $f_{u,j}$ 表示考虑 i 的子树，子树内所有点的 W 的和为 $s_u = j$，满足条件的概率。

由于 u 要比子树内的其他点先被选，概率为 $\frac{W_u}{s_u}$。u 的不同儿子的子树间没有别的限制，朴素转移即可。

考虑正常情况，即有儿子指向父亲的边怎么办。正常算很困难，所以考虑容斥。注意到一条反向边不满足等价于正向边的满足，所以容斥相当于枚举不满足的反向边集 E，再将不在 E 中的反向边断掉。

显然容斥系数为 -1，然后就变成和外向树一样的树形背包了。时间 $O(n^2)$。

AGC058F

- 对于一棵树 T，定义函数 $f(\mathrm{T})$ 如下：
- 若 $|\mathrm{T}|=1$，则 $f(\mathrm{T})=1$。
- 若 $|\mathrm{T}|>1$，则 $f(\mathrm{T})=\frac{1}{|\mathrm{T}|}\sum_{e\in \mathrm{T}} f(\mathrm{T}_e^1)\times f(\mathrm{T}_e^2)$，其中 $\mathrm{T}_e^1,\mathrm{T}_e^2$ 分别表示 T 删掉边 e 得到的两棵子树。
- 给定一棵大小为 n 的树 T_0，请你求出 $f(\mathrm{T}_0)$ 模 998244353 的值。
- $n\le 5000$

AGC058F

如果式子中是 $\frac{1}{|\mathrm{T}|-1}$ 而不是 $\frac{1}{|\mathrm{T}|}$，那 $f(\mathrm{T})$ 就恒等于 1 了。

这启发我们将边数转为点数。对 T 中的每条边拆成两条，中间加一个“边点”，点数就变成了 $2n-1$，不太对。

怎么办呢，我们可以给每个边点再挂一个大小为 -1 的儿子，这样总点数就为 n 了。当然，在模意义下，这相当于挂 $p-1$ 个正常的儿子。

然后原式有一个非常漂亮的组合意义：随机一个点排列作为每个点的权值，问所有边点的权值小于它连接的所有点的概率。

容斥 + 树形背包即可，时间复杂度 $O(n^2)$。

LOJ3687

- 对于长度为 n 的小写字母字符串 S，记 T_i 表示从 S 中删除第 i 个字符得到的字符串。
- 给定 m 个条件，每个条件给定 x, y，要求 T_x 的字典序小于等于 T_y。
- 求满足所有条件的 S 的个数。
- $n, m \leq 5 \times 10^5$。对 $10^9 + 7$ 取模。

LOJ3687

T_x 与 T_y 的大小关系可以只用相邻字符的大小关系描述。我们在所有相邻字符放上 $< / = / >$ 来描述这一关系，那么约束就是：一个区间内必须全为 $=$，或第一个不等号是 $< / >$。

将限制记录在区间左端点上，仅保留右端点最大的一个。下面从右往左填写字符。

设 $f_{i,c}$ 表示已经填完了 $S_i \sim S_n$，$S_i = c$，且满足了 i 后面所有限制的方案数。如果 i 处没有限制，则 $f_{i,c} = \sum_j f_{i+1,j}$。

如果 i 处有限制，形如 $[i, r]$ 间的第一个不等号为 $<$，则不合法的情况形如 $S[i, r] = cc \cdots [> c] \cdots$。我们枚举 $k = i+1, i+2, \cdots, r$，将 $f_{i,c}$ 减去 $\sum_{j=c+1}^{26} f_{k,j}$，来删掉这些方案。

LOJ3687

显然，需要考虑的 k 要求 $S[i,r]=cc\cdots[>c]\cdots$ 能满足所有 $i+1,i+2,\cdots,k-1$ 上的限制。所以，我们需要对每个 k，求出 k 处出现 $>/<$ 时，k 前面第一个不满足的限制的位置。这部分可用单调栈求出。

于是总时间为 $O(n|\Sigma|)$。

LOJ2396

- 一辆车出发时刻为 0，到达时刻为 X，途中会经过 n 个服务站，经过第 i 个服务站的时刻为 s_i。
- 车上有一台容量无限的饮水机。在出发前或在每一个服务站，你都可以给饮水机加整数升的水，每升 W 元。
- 车上有 m 名乘客和一名司机。乘客 i 在 $kT+d_i$ 时刻需要装 1 升水，司机在 kT 时刻需要装 1 升水，$k\in\mathbb{N}$。若乘客 i 需要装水时没水了，他就会下车，你需要给他退 c_i 元。你不能让司机没水喝。
- 问到达终点后，水费加退掉车费的最小值。
- $n,m\le 2\times 10^5, T\le X\le 10^{12}, W\le 10^6, c_i\le 10^9$。保证 $d_i>0$ 且互不相同，到达服务站/终点时不会有人喝水。

LOJ2396

先将乘客按照 d_i 从小到大排序，则在 $kT \sim (k+1)T-1$ 的时间段内，司机和乘客会依次进行一轮喝水。我们记这一段时间为第 k 个周期。

若我们想让乘客 i 下车，则在某周期中的 d_i 时刻，饮水机没有水。这会导致 i 之后的乘客也跟着下车。之后，在该周期内必须要遇到一个服务站或到达终点，来补水，否则司机将没水喝。

设 f_i 为仅考虑了前 i 个乘客时的最小花费（不能强制使后面的人下车），并取 $s_{n+1} = X$。

若乘客 i 没有下车，则 $f_i = f_{i-1} + \lceil \frac{X-d_i}{T} \rceil W$。

LOJ2396

若乘客 i 中途下车了，假设和他一同下车的是 $[j+1,i]$ 的人，那么

$$f_i=\min_{j<i} f_j+S_i-S_j+W(i-j)\times k_i$$

其中 k_i 表示在哪一轮将 $[j+1,i]$ 的人赶下车，其可能的取值为 $\lfloor\frac{s_l}{T}\rfloor, d_i<s_l \bmod T<d_{i+1}$，且越小越好。

不难对上述转移构造出一组对应的补水方案。答案即为 f_m。

k_i 是可以 $O(n)$ 预处理得到的。

将转移式写作 $f_i-iWk_i-S_i=\min_{j<i} f_j-S_j-jWk_i$。不难发现这个式子可以斜率优化，于是对 (j,f_j-S_j) 维护凸壳即可，并在其上二分即可。时间复杂度 $O(m\log m)$。

LOJ3972

- 给定 n, k 和长为 $2n$ 的字符串 S。S 恰含 n 个 A 和 n 个 B。
- 每次可以交换 S 中的两个相邻字符。求最小交换次数，满足：S 可以被划分为恰好 k 个子序列，每个子序列形如 $AA\cdots ABB\cdots B$，且 A, B 数量相等。
- $1 \le k \le n \le 10^6$。

LOJ3972

先考虑如何判定一个序列是否合法。一个基本条件是，第 i 个 A 要在第 i 个 B 之前。同时，通过调整法可知，存在最优划分方案使得第 i 个 A 和第 i 个 B “匹配”，每次将前缀的 A 划入一个子序列。

注意到第 i 个 A 和第 i 个 B 在开始就已经确定，所以我们考虑朴素 DP: 设 $f_{i,l}$ 表示目前划分出了 l 个子序列，当前子序列开头为第 $i+1$ 个 A 所需的最小交换次数。

枚举当前子序列的末尾 j，则转移为 $f_{j,l+1} = \min\{f_{j,l+1}, f_{i,l} + \sum_{k=i+1}^{j} \max\{b_k - i, 0\}\}$，此处 b_k 表示第 k 个 A 之前 B 的数量。

注意到 $w(i,j) = \sum_{k=i+1}^{j} \max\{b_k - i, 0\}$ 满足四边形不等式，于是 DP 有凸性，可用 WQS 二分优化掉 l。新的转移为 $f_j = W + \min_{i<j}\{f_i + \sum_{k=i+1}^{j} \max\{b_k - i, 0\}\}$。

LOJ3972

接下来处理 $w(i,j)$：当 $b_j \le i$ 时，$w(i,j)=0$，直接用单调栈维护。

当 $b_j > i$ 时，对于每个 i 都存在最大的 $t \ge i$，使得 $b_t \le i$，那么贡献式变为 $f_i + s_j - s_t - (j-t)i$。这是斜率优化的形式，可以在扫 j 时维护一个关于 i 的凸包做到线性。

加上 WQS 二分，总时间为 $O(n \log n)$。

AGC022F

- 数轴上有 n 个点，第 i 个点坐标为 x^i，$x = 10^{100}$ 足够大。
- 每次选择两个点 A, B，将 A 的位置关于 B 对称，并删去 B。
- 问 $n-1$ 次操作后，剩下的点可能出现的位置有几处。
- $n \leq 50$。模 998244353。

AGC022F

由于对称操作为 $A=2B-A$，每个点的坐标都可表示为 $\sum_i 2^{a_i}(-1)^{b_i}x^i$。显然任意一项 $2^{a_i}(-1)^{b_i}$ 不同，求和结果就不同。下面考虑最后剩下点的坐标对应的 a 和 b。

如果在 A 关于 B 对称时将结点 A 向结点 B 连边，则每个点入度为 1，$n-1$ 次操作后会形成一颗外向树，根节点即为最后剩下的点。

这样做的好处是，a_i 必定等于 dep_i，即只和树的形态有关。为每条边赋上边权 1 或 -1，表示儿子是否被取反，则 b_i 与边权的选取有关。

接下来研究边权需要满足的条件。不难发现，点 u 连向 $v\in son_u$ 的边中，恰有 $\lfloor\frac{|son_u|}{2}\rfloor$ 条边边权为 -1。这样 b_i 等于 i 到根路径上边权的乘积，乘上 $(-1)^{|son_i|}$。我们设 c_i 表示 i 到根路径上边权的乘积。

AGC022F

下面开始 DP。先设 $f_{i,j}$ 表示目前填了 i 个结点，最后一层有 j 个节点钦定有奇数个儿子。为了不重复计算方案，这 j 个节点的 c_i 相同。在这种情况下，当前层 $c_i=1$ 和 $c_i=-1$ 的点个数之差绝对值为 j。

与另一种设法比较：设 $g_{i,j}$ 表示目前填了 i 个结点，当前层 $c_i=1$ 比 $c_i=-1$ 的点多 j 个 (j 可以为负数) 的方案数。$g_{i,j}$ 的转移需要枚举当前层填入点的个数 k，同时枚举这 k 个节点中 $b_i=1$ 的结点个数 x。

由于 $c_i=1$ 的节点个数为 $p=\frac{k+j}{2}$，我们需要选择 $|p-x|$ 个结点钦定为奇数个儿子。这样将转移到 $g_{i+k,p-x}$，形如 $g_{i+k,p-x}\leftarrow g_{i,j}\binom{n-i}{k}\binom{k}{x}$。

AGC022F

由于 (j,k,x) 不同时，$(p-x,x,k-x)$ 也不同，上述过程不会使答案相同的方案被多次计算。

注意到 $g_{i,j}$ 和 $g_{i,-j}$ 共享相同的转移式，我们可以用 $f_{i,j}=g_{i,j}+g_{i,-j}$ 代替。转移为 $f_{i+k,|p-x|}\leftarrow f_{i,j}\binom{n-i}{k}\binom{k}{x}$。

初值 $f_{1,0}=f_{1,1}=n$，答案即为 $f_{n,0}$，时间复杂度 $O(n^4)$。

LOJ3967

- 一个问题的输入为 n 个区间 $[l_i, r_i]$，所有 $l_i, r_i \in [1, 2n] \cap \mathbb{N}$ 且两两不同，并满足 $l_i < l_{i+1}$。
- 为求出最多能选出多少个两两不交的区间，有如下算法：依次令 $i = 1, 2, \cdots, n$，若区间 i 与当前已选的区间均不交，则选择区间 i。输出选择的区间数量。
- 显然，该算法是错误的。给定 n，求有多少种该问题的合法输入，会让该算法得出错误结果。
- $n \leq 3000$。对给定的大质数 p 取模。

LOJ3967

输入总数为 $(2n-1)!!$，所以我们可以去求能使算法能得到正确结果的输入数量。

假做法是按 l_i 排序贪心取，而真做法是按 r_i 排序贪心取，分别设两种做法取到的区间集合为 $A, B, |A| = |B|$，A_i 可能等于 B_i。将所有不在 A, B 中的区间加入集合 C。

现在考虑 A, B 满足的限制。首先，A 满足任意区间不被 $[A_{i-1}.r, A_i.r]$ 包含，B 满足不存在区间 j 使得 $l_j \in (B_{i-1}.l, B_i.l)$。容易验证，上述条件均充要。同时，我们还有 $B_{i-1}.r < B_i.l < A_i.r < B_i.r$。

限制已经基本足够，开始 DP。一个自然的想法是，不断向空区间集中从前往后加入 A_i, B_i 和对应的 C。于是设 f_i 表示已经加入了 i 个区间，最后一个 A 区间右端点前的 C 区间已全部加入的方案数。

LOJ3967

简单写一下转移式发现，由于加入的 C 区间的左端点要满足 B 的限制，还需要增设一维 k 表示有多少个位置可以插入左端点。这样很不划算，时间复杂度要到 $O(n^3)$。

注意到 A, B 集合对左右端点的限制并不等同，右端点受到的限制更弱。于是考虑从后往前加入区间。现在 f_i 表示已经加入了 i 个区间，第一个 A 区间右端点后的 C 区间已全部加入的方案数。

转移枚举新加入了 j 个 C 区间，形如 $f_{i+j+2} \leftarrow f_i\binom{j+1}{2}(2i-2)^{\bar{j}}$。这里延后计算了第一个 A 左端点的选择。

注意还有 $A_i = B_i$ 的情况，所以还需要加一位状态 $0/1$ 表示上一个 A_i 是否等于 B_i。

LOJ3967

总的转移如下

$$f_{i+j+2,0} \leftarrow f_{i,0}\binom{j+1}{2}(2i-2)^{\overline{j}} + f_{i,1}(j+1)(2i-1)^{\overline{j}}$$

$$f_{i+j+1,1} \leftarrow f_{i,0}(j+1)(2i-2)^{\overline{j}} + f_{i,1}(2i-1)^{\overline{j}}$$

时间复杂度 $O(n^2)$。

原题数据范围是 $n \leq 2 \times 10^4$。注意到转移式可以写成卷积形式，所以可使用 MTT 或 Karatsuba 乘法优化。但 $O(n^2)$ 随便卡卡常就能过。

The End

谢谢！