dp 专题 II

A_zjzj

Quzhou No.2 High School Zhejiang

Nov 2024

Part I

The 2023 ICPC Asia Macau Regional Contest H. Random Tree Parking

题目描述

给定一棵 n 个点的随机树（第 i 号点的父亲在 $[1, i-1]$ 中均匀随机）。

对于一个满足 $a_i \in [1, n] \cap \mathbb{Z}$ 的序列 $\{a_i\}_{i=1}^{n}$ 进行如下过程：

- 初始树上每个节点都是未标记的；
- 依次遍历 $i = 1, 2, \cdots, n$，找到 a_i 的祖先中深度最大的未标记节点 u 并标记 u。

求出有多少种可能的序列使得进行上述过程中不存在找不到节点 u 的情况，对 998244353 取模。

$2 \le n \le 10^5$。

思路

- 热身题，大家应该都能爆切。
- 容易发现，最终每个点一定都被标记了。
- 所以只考虑所有在 u 子树中的 a_i，此时 u 子树一定全被标记，但可能会有一些操作需要向上找。
- 由于树随机，故 u 的深度期望 $O(\log n)$，所以向上找的次数最多是 $O(\log n)$ 的。
- 故设 $f_{u,i}$ 表示 u 子树中，有 $siz_u + i$ 个节点，即向上找 i 次，合并时枚举一下两边的次数即可。
- 可以不需要用到指数生成函数知识。

The 2023 ICPC Asia East Continent Final Contest C. Equal Sums

题目描述

给定正整数 $l^{(x)}_{1\sim n}, r^{(x)}_{1\sim n}, l^{(y)}_{1\sim m}, r^{(y)}_{1\sim m}$，有 $n+m$ 个整形变量 $x_{1\sim n}, y_{1\sim m}$，每个变量有限制：$l^{(x)}_i \le x_i \le r^{(x)}_i, l^{(y)}_j \le y_j \le r^{(y)}_j$

对于每个 $1 \le a \le n, 1 \le b \le m$，求出有多少种选择 $x_{1\sim a}, y_{1\sim b}$ 的方案，对 998244353 取模，使得：

$$\sum_{i=1}^{a} x_i = \sum_{j=1}^{b} y_j$$

$1 \le n, m \le 500$，$1 \le l^{(x)}_i \le r^{(x)}_i \le 500$，$1 \le l^{(y)}_j \le r^{(y)}_j \le 500$。

思路

- 显然有个背包的暴力，复杂度为 $O(n^3V)$，考虑如何优化。
- 我们考虑一个状态 $(a, b, \sum_{i=1}^{a} x_i, \sum_{j=1}^{b} y_j)$：
  - 若 $\sum_{i=1}^{a} x_i < \sum_{j=1}^{b} y_j$，则转移到 $(a+1, b, x_{a+1} + \sum_{i=1}^{a} x_i, \sum_{j=1}^{b} y_j)$；
  - 若 $\sum_{i=1}^{a} x_i \geq \sum_{j=1}^{b} y_j$，则转移到 $(a, b+1, \sum_{i=1}^{a} x_i, y_{b+1} + \sum_{j=1}^{b} y_j)$。
- 如此，$\sum_{i=1}^{a} x_i - \sum_{j=1}^{b} y_j \in [-V, V)$，转移可以使用差分/前缀和做到 $O(1)$。
- 时间复杂度：$O(n^2V)$，空间复杂度可以做到 $O(nV)$。

The 3rd Universal Cup. Stage 13: Sendai D. And DNA

题目描述

给定非负整数 N, M，求出满足如下条件的环形序列 $A = (A_1, A_2, \cdots, A_N)$ 的个数模 998244353 的结果：

- $A_i \in [0, M] \cap \mathbb{Z}$;
- 对于 $i = 1, 2, \cdots, N$，$A_i + (A_{i-1} \text{ and } A_{i+1}) = M$（其中 A_0 代表 A_N，A_{N+1} 代表 A_1）。

$3 \le N \le 10^9$，$0 \le M \le 10^9$。

思路

- ~~这道题其实不完全是 dp 题。~~
- 首先，由于存在与运算、加法运算，故考虑从低位到高位考虑确定每个 A_i，首先考虑 A_i 的最低位：
  - 若 M 最低位为 0，则 A_i 的最低位全相等；
  - 若 M 最低位为 1，当且仅当，A_i 的最低位 0 的连续段长为 1，1 的连续段长度不超过 2。
- 此时，对于 $A_i + (A_{i-1} \text{ and } A_{i+1})$ 更高位的贡献，要么都没有进位，要么都有 1 的进位。
- 接着，考虑 $2^1, 2^2 \cdots$ 位的方式是一样的。

第一步 dp

- 于是，我们设 $f_{k,0/1}$ 表示确定了 $A_{1\sim n}$ 的低 k 位，进位为 $0/1$ 的方案数，可得到转移：
  - 若 M 的 2^k 位为 0，则：

$$
\begin{aligned}
f_{k+1,0} &= f_{k,0} \\
f_{k+1,1} &= f_{k,1} \times w_N + f_{k,0}
\end{aligned}
$$

  - 若 M 的 2^k 位为 1，则：

$$
\begin{aligned}
f_{k+1,0} &= f_{k,0} \times w_N + f_{k,1} \\
f_{k+1,1} &= f_{k,1}
\end{aligned}
$$

- 其中，w_N 为长度为 N 的 01 环形序列，满足 0 的连续段长度为 1，1 的连续段长度不超过 2。

第二步 dp

- 现在考虑如何求 w_N。
- 对于 $N \geq 4$，此时 $0/1$ 的连续段个数都不少于 2，考虑找到第一个 0 的位置。
  - 若接下来 1 的连续段长度为 1，则删去 01 后剩余部分的方案数为 w_{N-2}；
  - 若接下来 1 的连续段长度为 2，则删去 011 后剩余部分的方案数为 w_{N-3}。
- 故当 $N \geq 4$ 时，$w_N = w_{N-2} + w_{N-3}$。边界条件：$w_1 = 0, w_2 = 2, w_3 = 3$。使用矩阵快速幂即可在 $O(\log N)$ 复杂度内求出。
- 实际上，$\{w\}$ 在 OEIS 中也能找到：A001608。

Part II

2024“钉耙编程”中国大学生算法设计超级联赛（3）1005. 数论

题目描述

给定长为 n 的正整数序列 $\{a_i\}_{i=1}^n$。

定义不交区间集为若干不交的区间 $[l_1,r_1],[l_2,r_2],\cdots,[l_k,r_k]$ 的集合，其中所有元素 $[l_i,r_i]$ 满足 $1\le l_i\le r_i\le n$。

称一个不交区间集为好的，当且仅当：

$$\gcd_{i=l_1}^{r_1}\{a_i\}=\gcd_{i=l_2}^{r_2}\{a_i\}=\cdots=\gcd_{i=l_k}^{r_k}\{a_i\}$$

对于每个 $x=1,2,\cdots,n$，求出有多少个好的不交区间集，存在 $[l_i,r_i]$ 包含 x，对 998244353 取模。

$1\le n\le 10^5$，$1\le a_i\le 10^9$。

思路

- ~~这道题重点并不在 dp 上。~~
- 首先，先使用经典结论，在固定左端点 l，右端点向右的过程中，$\gcd_{i=l}^{r}\{a_i\}$ 只会改变 $O(\log V)$ 次。
- 则外层枚举所有区间 gcd 的值 v，找到所有三元组 (l, r_1, r_2) 表示所有 $[l, r_1], [l, r_1+1], \cdots, [l, r_2-1]$ 的 gcd 都为 v。
- 接下来肯定要先将所有 l, r_1, r_2 离散化。
- 方便起见，接下来的 l, r_1, r_2 都为离散化之后的值，离散化后为 $i = 0, 1, \cdots, k-1$ 的原值为 w_i，不妨令 $w_k = n+1$。

做法

- 考虑对答案贡献的三元组 (l, r_1, r_2)，容易发现，对于相同的 $i \in [r_1, r_2)$，所有 $r \in [w_i, w_{i+1})$，区间 $[w_l, r]$ 选入好的不交区间集的方案数相同。
- 故设 f_i 为区间右端点小于 w_i 的 $\gcd = v$ 的不交区间集的方案数；类似地，设 g_i 为区间左端点不小于 w_i 的 $\gcd = v$ 的不交区间集的方案数。
- 则对于所有 $j \in [w_l, r]$，$ans_j \leftarrow f_l \times g_{i+1}$。

计算答案

- 于是，考虑计算对于答案数组 ans 的差分 ans'，而其中一部分贡献，对于 $[w_i, w_{i+1})$ 是相同的，所以考虑计算出 b_i 表示对于 $ans'_{w_i+1\sim w_{i+1}}$ 的贡献。
- 则对于所有 $ans'_{w_l} \leftarrow f_l \times g_{i+1}, b'_{i+1} \leftarrow -f_l \times g_{i+1}$。
- 所以，对于三元组 (l, r_1, r_2)，用前缀和、差分求出：

$$ans'_l \leftarrow f_l \times \sum_{i=r_1}^{r_2-1} g_{i+1} \times (w_{i+1} - w_i)$$

$$b_i \leftarrow -f_l \times g_{i+1} \qquad i \in [r_1, r_2)$$

求解 $\{f_i\}_{i=0}^{k}$

- 边界条件：$f_0 = 1$。考虑从小到大枚举 i，求出 f_i。
- 当求出 f_i 时，枚举左端点落在 i 的三元组 (i, r_1, r_2)，考虑其对于 f 的贡献：

$$f_{j+1} \leftarrow (w_{j+1} - w_j) \times f_i \quad j \in [r_1, r_2)$$

- 这里对于 f_{j+1} 的贡献是上一个区间右端点落在 $[w_j, w_{j+1})$ 的情况，对于上一个区间右端点 $< w_j$ 的情况，增加转移：

$$f_i \leftarrow f_{i-1} \quad i \in (0, k]$$

- 只需差分即可轻松解决。

求解 $\{g_i\}_{i=0}^{k}$

- 边界条件：$g_k = 1$。考虑从大到小枚举 i，求出 g_i。
- 在求 g_i 时，对于左端点 $\geq w_{i+1}$ 的情况，有转移：

$$g_i \leftarrow g_{i+1} \quad i \in [0, k)$$

- 对于左端点为 w_i 的情况，枚举左端点落在 i 的三元组 (i, r_1, r_2)，考虑其对于 g_i 的贡献，使用前缀和即可。

$$g_i \leftarrow \sum_{j=r_1}^{r_2-1} g_{j+1} \times (w_{j+1} - w_j)$$

总结

- 至此，本题已经完全解决。
- 本题的难点在于，离散化后出现的一系列系数，需要考虑清楚。
- 通过合理的设置前后缀和/差分的方式，使得不会访问到数组未定义的位置。

总结

- 例如，差分/求和的方向是前缀还是后缀，前后缀和是否包含当前位置。
- 若原数据为 $\{a_i\}_{i=1}^{n}$，则前缀和 $\{s_i\}_{i=0}^{n}$ 设置为 $s_i = \sum_{j=1}^{i} a_j$ 更为方便；
- 若原数据为 $\{a_i\}_{i=0}^{n-1}$，则前缀和 $\{s_i\}_{i=0}^{n}$ 设置为 $s_i = \sum_{j=0}^{i-1} a_j$ 更为方便；
- 写代码时注意这些细节，熟练后可以使代码更加优雅。

2024 牛客暑期多校训练营 4 D. Plants vs. Zombies (Sunflower Edition)

题目描述

你有两种向日葵可以种植，第 i 种向日葵需要 p_i 个阳光，种下之后每轮会生产 q_i 的阳光。

初始时你有 s 个阳光，游戏共有 n 轮。在每轮开始时，你可以种植向日葵，但每种向日葵至多只能种一株。在每轮结束时，你可以收获现有的所有向日葵产生的阳光。

请你求出最终 n 轮结束后，最多能够剩下多少阳光。

$1 \le n, s \le 2 \times 10^7$，$1 \le p_i, q_i \le 1023$。

设计 dp

- 首先可以非常 easy 地设计出 dp。
- 设 $f_{i,j}$ 表示在第 i 轮结束时，种下的向日葵每轮能够产生 j 个阳光时，目前最多的阳光数。
- 容易发现，j 不超过 $n(q_1+q_2)$，且转移是简单的，时间复杂度：$O(n^2q)$。

优化 1

- 发现当 j 这一维超过 $p_1 + p_2$ 时，接下来就不用考虑缺阳光的问题了，直接贪心即可。
- 具体地，对于接下来的每一轮，如果种下来获得的收益大于 0 就种，否则不种。
- 求出范围后，做一遍等差数列求和，$O(1)$ 计算。
- 所以，j 这一维降为 p，时间复杂度：$O(np)$。

优化 2

- 若第一轮无法种下任何一株向日葵，则答案为 s。
- 否则，第一轮一定会种下一株，由于 $q_i \geq 1$，所以，接下来最多需要 $\frac{p}{1}$ 轮攒足一次种向日葵的钱。
- 接下来，类似地，需要 $\frac{p}{2}$ 轮攒足第三次，$\frac{p}{3}$ 轮攒足第四次……
- 所以，种下 p 株向日葵最多需要 $\frac{p}{1}+\frac{p}{2}+\cdots+\frac{p}{p}=O(p\ln p)$ 轮。
- 所以，dp 中 i 这一维是 $O(p\ln p)$ 级别的。
- 至此，总复杂度降为 $O(p^2\ln p)$。

The 3rd Universal Cup. Stage 3: Ukraine F. Formal Fring

题目描述

定义 $\text{highest\_bit}(n) \neq \max_{2^i \geq n, i \in \mathbb{N}}\{i\}$，特别地，定义 $\text{highest\_bit}(0) = -1$。

给定一个正整数 X，求出满足以下条件的多重集 S 的方案数对 998244353 取模的结果：

- 所有 S 中的元素都是 2 的非负整数幂；
- S 中所有元素的和为 X；
- 不存在一种将 S 的所有元素划分为两个多重集的方案，使得两个多重集元素和的 highest_bit 相等（即 $\text{highest\_bit}(S_1) = \text{highest\_bit}(S_2)$，其中 S_1, S_2 分别为两个多重集的元素和）。

对于 $X = 1, 2, \cdots, n$ 求出对应的答案。$1 \leq n \leq 10^6$。

思路

- 首先，容易发现，判断多重集 S 是否合法，需要完成一个背包。
- 但是，还有 S 元素是 2 的幂的性质没有用。
- 此时，我们发现，背包只需要贪心就行了。
- 具体地，我们记录一个变量 x，然后从大到小枚举 S 中的元素 y。若 $x > 0$，则 $x \leftarrow x - y$，否则 $x \leftarrow x + y$。S 符合条件当且仅当最终的 x 满足 $\text{highest\_bit}(\frac{X+x}{2}) = \text{highest\_bit}(\frac{X-x}{2})$。
- 这个贪心的正确性是比较显然的，分别考虑充分性和必要性即可。

进一步分析

- 我们考虑最终 x 的取值情况，发现 x 至多有 $\log n$ 种取值。
- 就是考虑 x 最后一次为 0 时，接下来一定是先加上 2^k，再不断减，但剩余的和也不够把 x 减到 0。
- 此时最后 x 的值一定为 $2^k - (X \bmod 2^k)$。

构建 dp

- 设 f_i 表示总和为 i 的可重集个数，设 g_i 表示总和为 i 且贪心结果 x 为 0 的可重集个数。
- 于是，在求 X 的答案时，枚举 k，若满足条件，则贡献为：

$$ans \leftarrow f_{X \bmod 2^k} \times g_{\lfloor \frac{X}{2^k} - 1 \rfloor},$$

$$\text{highest\_bit}(X + x) = \text{highest\_bit}(X - x)$$

- 前者是好理解的，后者是要保证 k 之前的步长 $\geq 2^k$。

求解 f, g

- f 是好求的，直接做多重背包即可。
- 求 g_i 时，考虑枚举贪心过程最近一次 x 为 0 的下一步加上的是 2^k，则有转移：

$$g_i \leftarrow f_{2^k} \times g_{\frac{i}{2^k}-2} \quad i \geq 2^{k+1} \wedge i \bmod 2^k = 0$$

- 至此，我们用 $O(n)$ 的空间，在 $O(n \log n)$ 的时间复杂度内解决了该问题。

Part III

The 2023 ICPC Asia Jinan Regional Contest L. Ticket to Ride

题目描述

有 n 个元素和 m 个区间 $[l_i, r_i]$，你需要找到序列 $\{p_i\}_{i=0}^{k}$ 满足 $0 = p_0 < p_1 < \cdots < p_k = n$，最大化：

$$\sum_{i=1}^{k}\sum_{j=1}^{m}[p_{i-1} < l_j \land r_j \le p_i]$$

对于每个 $k = 1, 2, \cdots, n$，求出上述答案。

$1 \le n, m \le 10^4$。

时间限制：3s，空间限制：128MB。

思路

- 首先，该题是一个区间划分型 dp 题，而代价函数 $w(l, r)$ 显然满足四边形不等式。但是，这个四边形不等式的似乎反了，所以并不能使用四边形不等式。
- 另一方面，使用四边形不等式，不仅复杂度不能接受，同时记录区间代价时的空间也需要 $O(n^2)$，否则就要用主席树，没有前途。

构建 dp

- 容易想到，我们一层一层 dp，设 $f_{i,j}$ 表示将 $1 \sim j$ 划分为 i 段区间的权值最大值。
- 在外层枚举 i 之后，我们显然有一个 $O((n+m)\log n)$ 的转移所有 $f_{i,j}$ 的方法：
  - 从小到大枚举 j，找到右端点落在 j 的区间 $[l,j]$，将线段树上 $1 \sim l-1$ 的点权值加一；
  - 求出线段树上 $1 \sim i-1$ 的最大值即为 $f_{i,j}$，并将 $f_{i,j-1}$ 的权值加入线段树中。

优化 dp

- 但是，线段树利用到的性质不够多，难以进一步优化。
- 我们发现，若某一时刻，对于 $x < y$，且 x 的权值不少于 y 的权值，那么之后 y 一定无法优于 x。
- 依据此性质，我们可以使用单调栈维护，需要支持前缀加一并删除无用元素。
- 于是，用链表维护单调栈，对于 $[1, r]$ 前缀加一，用并查集查询出单调栈中不超过 r 的最后一个元素，然后打上前缀加一的懒惰标记，并尝试删除它之后的若干元素，注意删除/插入元素时，需要维护好懒惰标记和并查集。

优化 dp

- 并查集维护的是未加入栈的元素和目前还在栈中的元素，当某个元素从栈中被删除时，在并查集中同样删除，查询时只需查询 r 之前第一个未删除的元素即可。
- 这样，我们就可以做到 $O(n+m\alpha(n))$ 转移一层了。
- 时间复杂度：$O(nm\alpha(n)+n^2)$，空间复杂度：$O(n+m)$。
- 题外话：QOJ 上最快的提交是 $O(nm\log n)$ 的。

The 2023 ICPC Asia Xi'an Regional Contest M. Random Variables

题目描述

对于所有满足 $a_i \in [1, m] \cap \mathbb{Z}$ 的 m^n 种序列 $\{a_i\}_{i=1}^{n}$，该序列的权值为众数的出现次数。

求出所有可能的序列的权值和模 p 的结果。

多测 T 组数据，这 T 组数据的 p 均相同，**64MB** 空间。

$1 \le T \le 10^4$，$2 \le p \le 10^9 + 7$，$1 \le n \le 10^3$，$1 \le m \le 10^9$，$\sum n \le 10^4$。

思路

- 考虑枚举众数的出现次数 $\leq k$ 计算贡献，则需要求：

$$\left[\frac{x^n}{n!}\right]\left(\sum_{i=0}^{k}\frac{x^i}{i!}\right)^m$$

- 但是，这样看上去似乎没有前途。

dp 做法

- 然而，这道题有意思的地方就在于，如果直接 dp，设 $f_{n,m}$ 表示众数的出现次数 $\le k$ 的贡献，则：

$$f_{n,m} = m \times \left(f_{n-1,m} - f_{n-k-1,m-1} \times \binom{n-1}{k} \right)$$

- 容易发现，当 m 减一的时候，n 至少减少了 $k+1$，故此 dp 中，m 的范围是 $O(\frac{n}{k+1})$，复杂度为 $O(\frac{n^2}{k+1})$。
- 故总复杂度为 $O(n^2 \ln n)$，常数较小，足以通过。

2024 牛客暑期多校训练营 8 F. Haitang and Diameters

题目描述

给定一棵 n 个点的树。定义两个点的距离为简单路径上边权之和。定义一棵树的直径为距离最大的两个点之间的距离。

定义一棵树的直径数为满足如下条件的二元组 (u, v) 的个数：

- $1 \le u < v \le n$;
- u, v 的距离等于该树的直径。

现在，你可以把 $n-1$ 条边任意赋权 $0/1$，对于所有 2^{n-1} 种赋权的方案，求出所有方案的直径数之和模 998244353 的结果。

$2 \le n \le 2000$。

思路

- 首先，对于一个给定边权的树，考虑如何求出直径数；
- 想到找到所有可能的直径中点：
  - 若中点只能落在边 (u, v) 上，则需满足 u, v 两边的子树深度最大值相等且 (u, v) 边权为 1；
  - 否则，直径中点可能的情况是一个连通块，找到该连通块的所有叶子，求出该叶子对应子树深度最大值的端点个数。
- 于是，考虑将这个做法扩展到一般情况。

扩展做法

- 第一种情况相对好处理，考虑第二种情况，很容易想到连通块的 $|V|-|E|$ 容斥。
- 对于 $|V|$ 的贡献，枚举点 u，需要满足 u 至少两个子树取到相等的最大深度。
- 对于 $|E|$ 的贡献，枚举边 (u,v)，需要满足 u,v 两边的子树的最大深度相等。
- 至此，我们有两种方向：
  1. 对于每个点进行长剖优化深度为下标的 dp；
  2. 换根 dp。
- 对于这道题，两种方法都是可行的，但是，出题人 dXqwq 在题解课件中声称，长剖需要处理除法，所以要考虑乘 0 的次数。
- 于是，我们考虑使用较为方便的换根 dp。

自下而上 dp

- 设 $f_{u,i}=(x,y)$ 表示在 u 子树中，最大深度为 i，方案数为 x，端点个数和为 y。
- 这一部分 dp 是方便的，需要注意，若二元组 (x,y) 在合并时不为最大深度，则需要用 $(x,0)$ 合并。

自上而下 dp

- 设 $g_{u,i} = (x, y)$ 表示在 u 子树外，到 u 最大深度为 i，方案数为 x，端点个数和为 y。
- 容易发现，合并两个子树的复杂度可以用前缀和做到 $O(n)$。
- 而从 g_u 转移到 g_v，需要合并 g_u 和 u 的其他所有儿子 w 的 f_w。
- 于是，考虑对于儿子序列，处理一下前缀合并的信息，后缀合并的信息，则每个 g_v 只需把前缀后缀再合并一次即可。

计算答案

- 最终，计算答案时，对于 $|E|$ 的贡献和直径中点在一条边上的情况都是简单的，下面考虑 $|V|$ 的贡献。
- 枚举一个点 u 的贡献，考虑枚举最大深度 d，然后再枚举 u 的所有出边 (u,v)，讨论 v 是否取到最大深度，若取到最大深度，则贡献为端点数，否则贡献为方案数；最终要求至少两个 v 取到最大深度即可。
- 总时间复杂度：$O(n^2)$，空间复杂度：$O(n^2)$，和 std 一致；
- 参考代码：link，由于是赛时代码，不够精简，仅供参考。

Part IV

The 2024 ICPC World Finals Astana H. Maxwell's Demon

题目描述

宽为 $2w$，高为 h 的容器，正中间一个竖直挡板把容器划分为两个 $w \times h$ 的容器。容器中有 r 个红色和 b 个蓝色的小球，小球在容器中运动（给定初始位置 p_x, p_y 和初速度 v_x, v_y），遇到容器壁或挡板的时候反弹。

在正中间挡板高度为 d 的位置，有一个筛选器，在每一时刻，你可以任意决定筛选器是否开启：若开启，则所有此时到达筛选器的小球都会穿过挡板，否则都会反弹。

求出最快能够在何时把所有红色小球筛选到左边容器，蓝色小球筛选到右边容器，或判断无解。

$2 \le w, h \le 200$，$0 \le d \le h$，$1 \le r + b \le 200$，$0 < |p_x| < w$，$0 < p_y < h$，$|v_x| < w$，$|v_y| < h$，$(v_x, v_y) \neq (0, 0)$。

思路

- 此题显然不是 dp 题，~~防止大家形成惯性思维~~，实际上是我开始以为题目不够，就选了这题，后来就保留下来了。
- 首先，对于每个小球，我们只需要求出其到达筛选器的时间集合。因为无论是否开启筛选器，不会改变小球到达筛选器的时间。
- 此时，需要一个经典的反射法，我们不让小球反弹，转为把容器翻转，这样，我们就可以求出小球到达筛选器的时间 t 的集合。

推式子

- 我们可以列出如下式子：

$$\begin{cases} p_x + t \times v_x \equiv 0 \pmod{2w} \\ p_y + t \times v_y \equiv d \pmod{2h} \end{cases}$$

$$\begin{cases} p_x + t \times v_x \equiv 0 \pmod{2w} \\ p_y + t \times v_y \equiv 2h - d \pmod{2h} \end{cases}$$

推式子

- 两种情况本质是一样的，我们考虑前者：

$$\begin{cases} p_x + t \times v_x = 2w \times p \\ p_y + t \times v_y = 2h \times q + d \end{cases}$$

$$\Rightarrow t = \frac{2wp - p_x}{v_x} = \frac{2hq + d - p_y}{v_y}$$

$$\Rightarrow (2wv_y)p - v_y p_x = (2hv_x)q + v_x d - v_x p_y$$

$$\Rightarrow (2wv_y)p - (2hv_x)q = (v_y p_x + v_x d - v_x p_y)$$

推式子

- 解出其中一组满足条件的 (p_0, q_0)，则所有满足的 p 可写为：

$$\left\{ p = p_0 + k \times \frac{hv_x}{\gcd(hv_x, wv_y)} \middle| k \in \mathbb{Z} \right\}$$

推式子

- 反过来，$t=\frac{2wp-p_x}{v_x}$，代入可得：

$$\left\{t=\frac{2w\left(p_0+k\times\frac{hv_x}{\gcd(hv_x,wv_y)}\right)-p_x}{v_x}\middle|k\in\mathbb{Z}\right\}$$

$$\Rightarrow\left\{t=t_0+\frac{2wh}{\gcd(hv_x,wv_y)}\times k\middle|k\in\mathbb{Z}\right\}$$

分析

- 容易发现，所有小球的 t，循环节都可以写为 $\frac{2wh}{x}$，其中 $x \in \mathbb{N}^*$，故 $2wh$ 一定是循环节。
- 所以，对于每个 $t \in (0, 2wh] \cap \mathbb{Q}$，找到此时到达筛选器的小球集合 S_t。
- 接下来，就非常容易想到使用线性基解决剩余问题。
- 总时间复杂度：$O(\frac{wh(r+b)^3}{\omega})$。由于是 6s 时限，加上上界并卡不满，所以能够通过。
- 参考代码：link。

The 2024 ICPC World Finals Astana K. Tower of noiHa

题目描述

在朴素的 n 个圆盘的汉诺塔问题中，存在一种唯一的 2^n-1 步将所有圆盘从 0 号柱子移动到 2 号柱子的方案。则考虑移动 k 步后的状态，将此时 0 号柱子的所有圆盘**整体**移动到 2 号柱子上，在现在的状态上，求出最少的把所有圆盘以**正确**顺序摆放在 2 号柱子上的步数。

注意，此时的移动需满足：若把大小为 A 的圆盘移动到大小为 B 的圆盘上，要求 $A<B$。

输入的 k 和输出的步数均用二进制表示（无前导零）。

$1\le n\le 2\times 10^5$，$0\le k\le 2^n-1$。

思路

- 此问题，大致有几种解决方案：对于所有可能的移动方式计算贡献，设计子问题递归解决。
- 对于这道题，显然需要设计子问题。我们用 012 字符串 S 表示一个状态，意思是 n 个圆盘分成三垒，分别编号为 $0, 1, 2$。
- 对于初始状态，我们可以设计 $g(S, k = 0/1/2)$ 表示 0 号柱子上有 $0, 1$ 两垒，0 在下，1 在上，1 号柱子上有一垒 2。目标是所有圆盘排序在 k 号柱子上。需要多少步。
- 则答案即为 $g(S, 0)$。

f 的转移

- 另外，手玩一下发现，我们还需要其他状态。
- 对于 01 字符串 S，设 $f(S, k = 0/1/2)$ 表示 0 号柱子上有一垒 0，1 号柱子上有一垒 1。目标是 k 号柱子。需要多少步。
- 接下来考虑 $f(S, 0/1/2)$ 如何转移。

f 的转移

- $f(S,0/1/2)$ 的转移较为简单（其中 S' 为 S 取出 n 的剩余字符串）:

$$f(S,0)=\begin{cases} f(S',0) & S_n=0 \\ f(S',2)+2^{n-1} & S_n=1 \end{cases}$$

$$f(S,1)=\begin{cases} f(S',1) & S_n=1 \\ f(S',2)+2^{n-1} & S_n=0 \end{cases}$$

$$f(S,2)=\begin{cases} f(S',1)+2^{n-1} & S_n=0 \\ f(S',0)+2^{n-1} & S_n=1 \end{cases}$$

g 的转移

- 接下来考虑 g 的转移:

$$g(S,0)=\begin{cases} g(S',0) & S_n=0 \\ f(w_{1,2}(S'),1/2)+2^{n-1}+f(w_{0,12}(S'),1) & S_n=1 \\ g(S',2)+2^{n-1} & S_n=2 \end{cases}$$

$$g(S,1)=\begin{cases} g(S',2)+2^{n-1} & S_n=0 \\ f(w_{1,2}(S'),2)+1+f(w_{0,12}(S'),2) & S_n=1 \\ g(S',1) & S_n=2 \end{cases}$$

g 的转移

- $g(S,2)$ 还有没有讨论：
  - $S_n=0$： $g(S',1)+2^{n-1}$；
  - $S_n=1$： $f\left(w_{1,2}(S'),1\right)+1+f(w_{0,12}(S'),2)$；
  - $S_n=2\wedge w_{1,2}(S')[-1]=1$：

$$\min\left\{g(S',0)+2^{n-1},f(w_{1,2}(S'),0)+1+h(w_{0,12}(S'),1)\right\}$$

  - $S_n=2\wedge w_{1,2}(S')[-1]=0$：

$$g(S',0)+2^{n-1}$$

- 其中，我们发现仍然需要一个状态 $h(S,1)$。

h 的转移

- 对于 01 字符串 S，$h(S,1)$ 表示 $0,1$ 两垒都在 0 号柱子上，0 垒在下，1 垒在上。目标为 1 号柱子。最少需要多少步。
- 设 $t_1(S)$ 表示 S 中 1 的个数，则：

$$h(S,1)=\begin{cases} h(S',1)+2^{n-1} & S_n=0 \\ 2^{t_1(S')}+f(S',2) & S_n=1 \end{cases}$$

- 其中 $w_{1,2}(S')$ 表示将 S' 中的 $1,2$ 提取出来分别作为 $0,1$，$w_{0,12}(S')$ 表示将 S' 中的 2 改为 1。

优化

- 现在，唯一的问题在于 $g(S,2)$ 有一种情况需要分别递归两个子问题，考虑优化这一部分；
- 一方面，由实际意义可以发现，$f(w_{1,2}(S'),0) \le g(S',0)$；
- 另一方面，归纳可以证明，$h(S',1) \le 2^n - 1$；
- 所以，当 $S_n = 2 \wedge w_{1,2}(S')[-1] = 1$ 时，就无需递归，即：

$$g(S,2) = f(w_{1,2}(S'),0) + 1 + h(w_{0,12}(S'),1)$$

实现

- 最后，只需要把上面的过程实现一下就行（tu）了。
- 一些注意：
  - 需要实现一下二进制下的长整数加法和比较；
  - 加 2^{n-1} 的部分，把一次递归的所有加起来，然后再加上其余贡献即可；
  - 细节较多，每个地方都要考虑清楚才行；
- 参考代码：link。

Part V

The 3rd Universal Cup. Stage 13: Sendai N. 0100 Insertion

题目描述

对于 01 字符串 S，每次操作为：

- 找到 S 中的任意一个子串 0100，删去该子串，剩余部分拼接起来。

给定一个长度为 n 包含 01? 的字符串，求出有多少种把每个 ? 分别替换为 0/1 的方案，使得得到的字符串能够经过若干次操作变为空串。

$1 \leq n \leq 500$，n 为 4 的倍数。

思路

- 首先，我们必然需要解决判定问题。
- 可以先列出一些显然的必要条件：
  - 1 恰好有 $\frac{n}{4}$ 个，0 有 $\frac{3n}{4}$ 个；
  - 1 连续段长度不超过 1；
  - 若将 0 看成 $+1$，1 看成 -1，则前缀和均大于等于 0；
  - 若将 0 看成 $+1$，1 看成 -2，则后缀和均大于等于 0。
- 但是，很明显，这些性质并不是充分的。
- 反例：001010100000。

探究反例

- 考虑一下 001010100000 为什么不满足。
- 由于把 0 看成 -1，1 看成 $+3$，使得总和为 0，性质会更优秀。
- 发现 001010100000 的其中两个 1 可以匹配成功：001[01[0100]00]0。
- 容易发现，第一个 1 是一定无法匹配的，可以通过折线图理解。

探究反例

- 具体地，设 s_i 表示前 i 个字符的权值和。
- 则在 0100 中，容易发现，$s_1 = x - 1, s_2 = x + 2, s_3 = x + 1, s_4 = x$，且无论如何插入 0100，这四个值都不会改变。
- 而在 001010100000 中，$s_3 = 1$，故这个 1 后面的两个 0 的 s 值分别应为 $0, -1$，而这个样例中，后面并不存在 s 值为 -1 的 0。
- 所以，我们似乎找到了比较充分的条件。

充要条件

- 方便起见，把字符串反转，匹配字符串改为 0010。
- 把 0 看成 -1，1 看成 $+3$，s_i 的定义同上。
- 则充要条件为：
  - 无相邻两个 1，最后一个字符不为 1，权值总和 $s_n = 0$；
  - 对于每个 1 出现的位置 i，则一定存在 $0 \leq j < i, s_j = s_i - 1$。

正确性证明

- 必要性是比较好证明的，只需证明：若 $S+T$ 满足上述条件，则 $S+0010+T$ 满足上述条件；
- 接下来考虑一下充分性，我们进行如下构造：
  - 找到 s_i 最大的一个 i，若有多个，取第一个；
  - 容易发现 i 位置一定是 1，$i-1$ 为 0，且 $s_{i-1}=s_i-3, s_{i-2}=s_i-2$；
  - 再找到最大的 j 满足 $0\le j<i, s_j=s_i-1$，容易发现 $j+1, j+2$ 位置一定是 0，则匹配 $(j+1, j+2, i, i+1)$，中间 $[j+3, i-1]$ 显然满足上述性质，而把 $[j+1, i+1]$ 删除后，两边接起来的，同样也满足上述性质；
  - 如此继续归纳即可。

dp 部分

- 至此，我们只需要把上述性质，用 dp 描述出来就行。
- 设 $f_{i,j,k,0/1}$ 表示前 i 个字符，i 字符为 0/1，$s_i = j$，此时 $\max_{x=0}^{i}\{s_x\} = k$ 的方案数。
- 转移时，枚举下一个字符是 $0/1$ 即可 $O(1)$ 转移。
- 总时间复杂度：$O(n^3)$，空间复杂度可以做到 $O(n^2)$。

The 2023 ICPC Asia East Continent Final Contest A. DFS Order 4

题目描述

对于所有点数为 n，每个点父亲编号小于自身编号的有根树，找到其唯一的最小字典序的 DFS 序，求出该 DFS 序有多少种可能，对 P 取模。

$1 \le n \le 800$，$10^8 \le P \le 1.01 \times 10^9$，$P$ 为质数。

前情回顾

- 在九月份的 dp 专题中，我们总结过此类问题：给定函数 $A(x,y)=0/1$，求出有多少 x，$\exists y, A(x,y)=1$。
- 做法大致有三种：转化判定方式、带权计数、对判定过程 dp。
- 这道题，正是转化判定方式的做法。

思路

- 考虑给定一个 DFS 序，找到一棵与之对应的树。
- 我们容易得到如下贪心过程：考虑依次遍历 DFS 序中的每个点，维护当前点到根的路径（类似构建虚树的过程）；在加入一个点 u 时：
  - 若栈顶 v 小于 u，则直接连边 $fa_u = v$；
  - 否则需要弹出所有编号大于 u 的点，因为这些点不可能成为 u 的祖先；另外，还需要弹出当前的栈顶，因为该 DFS 的字典序最小，因此儿子是按照编号依次遍历的，所以需要有一个编号小于 u 的点作为 u 的兄弟。最后再连边 $fa_u =$ 栈顶。
- 容易证明，一个 DFS 满足条件当且仅当可以顺利执行上述过程。

转化判定方式

- 于是，我们考虑对于上述贪心方法生成的树 dp。
- 即有如下两点限制：
  - 每个点的父亲编号小于自身编号；
  - 若 u 存在两个儿子 v_1, v_2，则 v_1 儿子中的编号最大值大于 v_2。

简化问题

- 考虑将编号的大小关系建成一张图，容易发现这个 DAG 并不是很好 dp。
- 所以，考虑对于第二条限制容斥，不考虑第二条限制的贡献为 $+1$，考虑第二条限制的反面的贡献为 -1。
- 此时，重新将编号的大小关系建图，并去除无用边，发现此时 DAG 成为了一棵树。
- 而对于树的拓扑序为：$n!\prod\frac{1}{siz_i}$。

构建 dp

- 这部分比较抽象，需要大量画图解决，此处略去。
- dp 方程为：

$$f_{i,j} = \frac{1}{i}\left(f_{i-1,j-1} - f_{i-1,j+1} + \sum_{k=1}^{i-1} f_{k,1} f_{i-1-k,j-1} + \sum_{k=1}^{i-2} f_{k,1} f_{i-1-k,j}\right)$$

- 边界条件：$f_{0,0} = 1$。答案即为 $(n-1)! f_{n-1,1}$，注意特判 $n = 1$ 的情况。时间复杂度：$O(n^3)$，空间复杂度：$O(n^2)$。

结语

感谢聆听