一些题

Y25t

2024 年 7 月

QOJ1427. Flip

有 $2n$ 个 bot，编号分别为 $1 \sim 2n$，你想把他们分为两组，每组 n 个 bot，分法如下：

- 所有 bot 按编号从小到大顺序抛一枚完全均匀（正反面朝上概率均为 0.5）的硬币。
- 如果硬币正面朝上就被分进 A 组，除非 A 组已经有 n 个 bot（这时他被分进 B 组）。
- 如果硬币反面朝上就被分进 B 组，除非 B 组已经有 n 个 bot（这时他被分进 A 组）。

有 q 次相互独立的询问，每次询问给定 k 个 bot 的编号 $b_1 \sim b_k$，求这 k 个 bot 被分进同一组的概率。答案对 998244353 取模。
$2 \le n \le 10^5, 1 \le q \le 10^5, 1 \le b_i \le 2n, b_i > b_{i-1}, 2 \le k \le n, \sum k \le 2 \times 10^5$。

题解

首先，给定一种分组状态，它出现的概率是多少？设最后一个分到 A 组的人编号为 i_A，最后一个分到 B 组的人编号为 i_B，则概率为 $2^{-\min(i_A,i_B)}$。

对于每组询问，不妨假设全分在 A 组，枚举 $\min(i_A, i_B) = p$ 的值，计算分组方案数：

- $p = b_k$，此时分组方案数为 $\binom{b_k-k}{n-k}$。
- p 位置分在 B 组，不妨设 $b_i < p < b_{i+1}$，则分组方案数为 $\binom{p-i-1}{n-1}$。对于固定的 i，$p-i-1$ 构成区间，恰当预处理前缀和可以 $O(1)$ 求出。
- $p > b_k$ 且 p 位置在 A 组，则分组方案数为 $\binom{p-k-1}{n-k-1}$。对于每种出现的 k，分别 $O(n)$ 预处理该式与概率的乘积的后缀和即可。

因为不同的 k 只有 $\sqrt{S}$ 个，所以时间复杂度 $O(n\sqrt{S}+S)$，期望得分 100。注意边界。

CF917D Stranger Trees

给定一张 n（$2 \le n \le 100$）个节点的无向完全图和这个图的一棵生成树。对于 $i = 0, 1, \cdots, n-1$，求出有多少棵这个完全图的生成树，使得这些生成树与给定的生成树恰好有 i 条边重复。答案对 (10^9+7) 取模。

题解

矩阵树定理 + 插值.

QOJ2568. Mountains

求有多少 $n \times m$, 元素均为非负整数的矩阵使得所有从左上角到右下角的路径 (每次只能往下或右走一格) 的元素和均不超过 k。答案对 10^9+7 取模。

$n, m, k \le 100$。

题解

设 $f_{i,j}$ 表示从 $(1,1)$ 走到 (i,j) 的最大路径元素和, 那么满足向两个方向分别递增的 f 和原矩阵有一个双射.
对于每个 $0 \le x \le k$, 所有满足 $f_{i,j} \le x$ 的 (i,j) 构成一个阶梯型. 划出阶梯的轮廓线, 轮廓线会随着 x 的增大 (非严格地) 不断向右下扩张. 而轮廓线组和原矩阵也有双射.
于是只需求出从 $(1,n)$ 到 $(n,1)$ 划出 k 条轮廓线的方案即可, 这可以转化后用 LGV 引理.

QOJ3082. Ascending Matrix

求有多少 $N\times M$ 的整数矩阵 $(a_{i,j})$ 使得:

- $1\le a_{i,j}\le K$.
- $a_{i,j}\le a_{i+1,j}$.
- $a_{i,j}\le a_{i,j+1}$.
- $a_{R,C}=V$.

答案对 998244353 取模。
$N,M\le 200, K\le 100$. $R\le N, C\le M, V\le K$.

考虑每个值的轮廓线，如果忽略最后一个限制时，就是求从左下到右上的 $k-1$ 条不穿过路径的条数。做法是把起点和终点平移一下转成不相交然后用 LGV 引理。
现在加上了 $a_{R,C}=V$ 的限制，那么相当于从上往下数的第 $V-1$ 条路径在 (R,C) 的下方。于是可以把 (R,C) 也平移一下然后给所有在其下方的路径乘上 x，最后求的就是行列式的 x^{V-1} 的系数。
可以代点值进去做行列式然后再插值回来，复杂度 $O(K^2(N+M)+K^4)$。

LOJ3706. 「ZJOI2022」树

对所有 $2 \le n \le N$, 求出满足以下条件的 n 个节点的有根树对 (T_1, T_2) 的个数:

- T_1 的根为 1, 其它节点的父亲节点编号均小于自己的编号.
- T_2 的根为 n, 其它节点的父亲节点编号均大于自己的编号.
- 不存在 i 使得节点 i 在 T_1 和 T_2 中均为叶子.

答案对给定模数 M 取模.

$2 \le N \le 500, 10 \le M \le 2^{30}$.

题解

所以就 dp，决定每个点是哪个叶子的同时，另一颗树可以选择也视为叶子容斥。

考虑按编号从小到大决定，$f_{x,y}$ 表示目前位置（包括现在这个）之前第一棵树前面有 x 非叶子，目前位置之后（不包括这个）第二棵树有 y 个非叶子。

初始 $f_{1,i} = i$，终止 $ans = \sum_i f_{i,1} \times i$

每次迭代加一个，设之前的是 f'，考虑对当前 f 的贡献：

- 考虑当前叶子给第一棵树
  - 钦定第二棵树是叶子容斥：$-f'_{x,y} \times x \times y \to f_{x,y}$
  - 不钦定第二棵树是叶子：$f'_{x,y} \times x \times (y-1) \to f_{x,y-1}$
- 考虑第二棵树叶子集合是这个
  - 钦定第一棵树是叶子容斥：$-f'_{x,y} \times x \times y \to f_{x,y}$ （哈哈，和刚才是一样的，神奇哦
  - 不钦定：$f'_{x,y} \times x \times y \to f_{x+1,y}$

复杂度 $O(n^3)$。

LOJ3714. 「AHOI2022」山河重整

求 $S \subseteq \{1, \dots, N\}$ 的个数满足对所有 $1 \le k \le N$, S 存在和为 k 的子集.
答案对给定模数 M 取模.
$1 \le N \le 5 \times 10^5, 2 \le M \le 1.01 \times 10^9$.

题解

首先有一个很显然的 $O(n^2)$ dp：

令 $f_{i,j}$ 为使用了前 i 个数字，目前最多可以凑出前缀 $[1,j]$ 的方案数，转移只需要新加入的区间拼的上就好了。

实际上，转移时的限制可以写作“集合中 $\leqslant k$ 的数之和要 $\geqslant k$”。于是我们考虑容斥，我们找到第一个不满足的位置 p 进行转移，并乘上 -1 的系数。

此时有一个很好的性质，由于 p 之前的位置都满足性质，所以小于等于 $p-1$ 的数之和一定是 $p-1$。

那么就可以把 dp 变成一维的，令 f_i 为 $[1,i]$ 内的数，和恰好为 i 的方案的容斥系数之和。它的计算可以用正常的整数拆分减去 f 带来的容斥转移。

这一类型的 dp 有一个很经典的优化到背包的方法，我们可以类似 P6189 [NOI Online #1 入门组] 跑步，一次要么加入一个钦定不满足的数字，要么给全局加若干次一，可以发现只会加入至多根号个数字，所以做这个 dp 的复杂度是 $O(n\sqrt{n})$ 的。

但是我们在转移前必须把转移过来的 dp 值提前计算完成，可以使用类似半在线卷积，每次先求出前一半的 dp 值，再转移给后一半。

复杂度是 $n\sqrt{n}+\frac{n\sqrt{n}}{2}+\frac{n\sqrt{n}}{4}+\cdots=O(n\sqrt{n})$ 的。

LOJ3627. 「2021 集训队互测」这是一道集训队胡策题

给出一个 $n \times n$ 的 01 矩阵 c, 求有多少长度为 n 的 01 序列 a, b, 满足 $c_{i,j} = a_i$ 或 $c_{i,j} = b_j$, 答案对 998244353 取模.
$1 \le n \le 5000$.

题解

注意到 $\sum_{i,j}([a_i = c_{i,j}] + [b_j = c_{i,j}] - [a_i = b_j]) \le n^2$ 且取等当且仅当 a, b 合法.

题解

注意到 $\sum_{i,j}([a_i = c_{i,j}] + [b_j = c_{i,j}] - [a_i = b_j]) \le n^2$ 且取等当且仅当 a, b 合法.

而设 $x = \sum_i a_i, y = \sum_j b_j$ 则有 $\sum_{i,j}[a_i = b_j] = xy + (n-x)(n-y)$. 于是固定 x, y 后一定会贪心地选取 a, b 使得 $\sum_{i,j}([a_i = c_{ij}] + [b_j = c_{i,j}])$ 尽可能大.

题解

注意到 $\sum_{i,j}([a_i = c_{i,j}] + [b_j = c_{i,j}] - [a_i = b_j]) \le n^2$ 且取等当且仅当 a, b 合法.
而设 $x = \sum_i a_i, y = \sum_j b_j$ 则有 $\sum_{i,j}[a_i = b_j] = xy + (n - x)(n - y)$. 于是固定 x, y 后一定会贪心地选取 a, b 使得 $\sum_{i,j}([a_i = c_{ij}] + [b_j = c_{i,j}])$ 尽可能大.
那么枚举 x, y 计算就能 $O(n^2)$.

3394. 「2020-2021 集训队作业」Tour

给定 n 和 n 个整数 $a_1, \ldots, a_n$，你要对满足如下条件的 $1, \ldots, n$ 的排列 p 计数：$\forall 1 \le i < n, a_{p_i} a_{p_{i+1}} \le w$，其中 w 为给定的**非负**整数。答案对 998244353 取模。

$0 \le w, |a_i| \le 10^9, 1 \le n \le 2 \times 10^5$。

（不超纲的）部分分：$a_i \ge 0$ 或 $n \le 2000$。

题解

先考虑 $a_i \geq 0$ 的情况。维护一个队列，执行以下过程直到 a 被删空：考虑 a 中最小元素 x 和最大元素 y，若 $xy \leq w$，就将 x 删去并入队，同时把它标记为好的；否则将 y 删去并入队，同时把它标记为坏的。

题解

先考虑 $a_i \geq 0$ 的情况。维护一个队列，执行以下过程直到 a 被删空：考虑 a 中最小元素 x 和最大元素 y，若 $xy \leq w$，就将 x 删去并入队，同时把它标记为好的；否则将 y 删去并入队，同时把它标记为坏的。

然后据此构造合法的排列 p，它初始只有一个空位。依次弹出队列中的每个元素，将它插入 p 的任意一个空位中，然后：

- 如果该元素为好的，那么后面任何元素填到它旁边都合法，于是给 p 增加了两个空位，使 p 净增加一个空位
- 否则它是坏的，后面任何元素填到它旁边都非法，不增加任何空位，使 p 净减少一个空位。

题解

先考虑 $a_i \geq 0$ 的情况。维护一个队列，执行以下过程直到 a 被删空：考虑 a 中最小元素 x 和最大元素 y，若 $xy \leq w$，就将 x 删去并入队，同时把它标记为好的；否则将 y 删去并入队，同时把它标记为坏的。

然后据此构造合法的排列 p，它初始只有一个空位。依次弹出队列中的每个元素，将它插入 p 的任意一个空位中，然后：

- 如果该元素为好的，那么后面任何元素填到它旁边都合法，于是给 p 增加了两个空位，使 p 净增加一个空位
- 否则它是坏的，后面任何元素填到它旁边都非法，不增加任何空位，使 p 净减少一个空位。

在这个过程中容易维护 p 的剩余空位数并对其计数。

而当 a_i 有正有负时可以按正负分别考虑（不妨将 0 算入正数），因为 $w \geq 0$, 所以正负相邻的情况一定合法。于是可以枚举最终排列中极长正负连续段分别有 i, j 段（需满足 $|i-j| \leq 1$），预处理出 f_i（g_j）表示正（负）数形成 i（j）段的方案数即可线性统计答案。

而当 a_i 有正有负时可以按正负分别考虑（不妨将 0 算入正数），因为 $w \geq 0$, 所以正负相邻的情况一定合法。于是可以枚举最终排列中极长正负连续段分别有 i, j 段（需满足 $|i-j| \leq 1$），预处理出 f_i（g_j）表示正（负）数形成 i（j）段的方案数即可线性统计答案。

以计算 f_i 为例，发现恰好 i 段不太好算，但至多（二项式反演意义下）i 段是容易计算的，就是构造上述 p 的过程中令 p 的初始空位数为 i 得到的方案。可以写成 $F(i)$，其中 $F(x) = \prod_{k\geq 1}(x+s_k)$，$s_k$ 为构造 p 的过程中插入第 k 个元素后空位的净增量，可以线性求出。

现在就是要求出 $F(1),\ldots,F(n)$ 然后二项式反演得到 f_i，暴力做的话是 $O(n^2)$ 的，想要更进一步的话就要一些多项式科技。

现在就是要求出 $F(1),\ldots,F(n)$ 然后二项式反演得到 f_i，暴力做的话是 $O(n^2)$ 的，想要更进一步的话就要一些多项式科技。

一种方法是分治 NTT 求出 $F(x)$ 然后再多点求值，又慢又难写。

现在就是要求出 $F(1),\ldots,F(n)$ 然后二项式反演得到 f_i，暴力做的话是 $O(n^2)$ 的，想要更进一步的话就要一些多项式科技。

一种方法是分治 NTT 求出 $F(x)$ 然后再多点求值，又慢又难写。

比较好的做法是注意到只用求出 $1,\ldots,n$ 处的点值，这对下降幂多项式是可以容易做到 $O(n\log n)$ 的。于是将每一个 $x+s_k$ 看成下降幂多项式然后做的分治下降幂多项式乘法就能求出 $F(x)$ 的下降幂表示了。常数和代码量都比上一个做法少很多。