Calc 解题报告

杭州外国语学校 陈立杰

Contents

1 题目大意

一个有序序列$a_1, ..., a_n$是合法的,当且仅当:

- 长度为给定的n.
- 其中的数两两不同.
- $a_i \in [1, A]$对所有i.

一个序列的值$value(a) = \prod_{i=1}^{i=n} a_i$.
求所有不同序列的值的和.
两个序列a, b不同,当且仅当存在一个i,使得$a_i \neq b_i$.
输出答案 $\bmod p$的值,p为素数

2 数据范围

- 5% $A, n \leq 10$
- 20% $A \leq 1000, n \leq 20$
- 50% $A \leq 10^9, n \leq 20$
- 100% $A, p \leq 10^9, n \leq 500, p > A > n + 1$
- For Fun $A \leq 10^9, p \leq 10^5, n \leq 20000$

3 本文中需要的一些基础知识

- 关于素数的逆元 我相信参加NOI的选手应该都会,故不作介绍.
- FFT 这仅仅是解决这个问题的强化版需要的知识,在解决此问题中并不需要,有兴趣的人可以自主学习.

4 5%的算法

我们枚举所有可能的序列并计算出它们值的和即可.
复杂度$O(A!)$

5 20%的算法

注意到因为所有元素互不相同,故我们如果将所有合法序列都从小到大排序,
那么对于每一种,都出现了$n!$次.
我们不妨先算出排个序之后的不同合法数列的和,再乘上$n!$.
令$dp[i][j]$表示已经考虑了$[1,i]$的数,选了j 个数的所有合法序列的和.
只需要考虑$i+1$这个数选不选,就能进行转移.
复杂度$O(An)$.

6 进一步的分析

由于A的值可以很大,我们希望复杂度最好不要跟A挂钩,或者是$\log A$级别的. 不妨让我们进行一些尝试.

7 100%的算法

让我们计算排个序之后的不同合法数列的和,再乘上$n!$. 让我们用$dp[A][n]$,表示答案.
注意到

$$\prod(a_i+A)=\sum_{s\subset\{0..n-1\}}A^{n-|s|}*\prod_{i\in s}a_i. \tag{1}$$

那么不妨考虑计算$dp[2A][n]$.
不妨考虑在$[1,A]$中选了a个,在$[A+1,2A]$中选了$n-a$个数,
不妨令a_i表示$[1,A]$中选了i个的序列的和,即为$dp[A][i]$.
同时令b_i表示$[A+1,2A]$中选了i个的序列的和.根据之前的式子,可以推出

$$b_i=\sum_{j=0}^{j=i}A^{i-j}*a_j*\binom{A-j}{i-j} \tag{2}$$

不妨考虑1的右边,考虑计算a_j对b_i的贡献,A^{i-j}是共有的系数, 同时可以发现有$\binom{A-j}{i-j}$个长度为i且包含某给定长度为j序列的串, 故系数是$\binom{A-j}{i-j}*A^{i-j}$
同时考虑化简$\binom{A-j}{i-j}$

$$\binom{A-j}{i-j}=\prod_{k=1}^{k=i-j}\frac{A-j+1-k}{k}=\frac{\prod_{k=j}^{k=i-1}(A-k)}{(i-j)!} \tag{3}$$

不妨令$m_i=\prod_{k=0}^{k=i-1}A-i, rm_i=(m_i)^{-1}$ 那么$\binom{A-j}{i-j}=(i-j)!^{-1}*m_i*rm_j$
那么我们进一步推出

$$b_i=m_i\sum_{j=0}^{j=i}A^{i-j}*a_j*rm_j*(i-j)!^{-1} \tag{4}$$

那么$dp[2A][n] = \sum_{i=0}^{i=n} a_i * b_{n-i}$.

m_i求出之后,我们可以使用逆元求出rm_i. 同时预处理$i \in [0, n]$的$i!^{-1}$

那么我们可以通过$dp[A][*]$,在$O(n^2)$时间内算出$dp[2A][*]$. 当然也可以通过$dp[A][*]$,在$O(n)$时间内算出$dp[A+1][*]$.

那么考虑计算$dp[A][*]$,如果A是偶数,归结到$dp[A/2][*]$,否则归结到$dp[A-1][*]$,每两步就至少将A减小一半,

总复杂度为$O(\log An^2)$.可以通过全部数据.

8 For Fun的算法

因为这个做法理论知识要求较高,就没有出到比赛中.

首先注意到如果要计算两个序列的卷积,可以使用FFT.

可以发现在上一个做法中,无论是算b_i,还是算$dp[2A][*]$,都可以转化成卷积,来使用FFT来加速,这样复杂度就降为了. p $O(\log An \log n)$.

9 一些其它的算法

这个题目也有$O(n^2)$的通过容斥原理来计算的做法,但是牵扯到快速计算$\sum_{i=1}^{i=A} i^x$这个式子,这需要比较高等的数学知识,如Bernoulli Number.在这里不做介绍,可以留个大家自己思考.