PKUWC2020 题解

吉如一
北京大学信息科学与技术学院

排列

- 给出长度为 n 的排列 P
- 定义 f(P) 为依次连接所有比 P 字典序小的排列得到的数列。f([2,1,3]) = [1,2,3,1,3,2,2,1,3]
- 求 f(P) 本质不同的子序列个数
- n <= 50，取模

排列

- 前置技能：已知字符集大小为 K 的序列 A,B，通过对 A,B 的预处理，可以 $O(K^3)$ 求出 AB 的本质不同的子序列个数

- 令 w(A)[i,j] 表示满足下列条件的子序列 t 的个数：
  - t不是 A 的子序列
  - t 删去最后一个字符后是 A 的子序列
  - t 的第一个字符是 i 最后一个字符是 j

排列

- 添加一个字符集外的字符 0，那么 w(A) 本质不同的子序列个数为 Sum w(A)[i,0]，其中 $1 \le i \le K$
- 已知 w(A), w(B)，考虑求 w(AB)
  - 考虑 w(AB)[i,j] 中的串 t，它一定存在一个极长的前缀是 A 的子序列，令 n 为该子序列长度
  - 那么，t 的前 n+1 个字符组成的串在w(A)[i, t[n+1]]中，从第 n+1个字符开始的后缀在 w(B)[t[n+1],j] 中
  - 所以 w(AB)[i,j] = Sum w(A)[i,k] × w(B)[k,j]

排列

- 回到这题，令 A[i] 表示 f([1,2,..,n-i, n, n-1,..,n-i+1])，即最小的 i! 个排列拼接起来的数组
- 其中 A[1] = [1,2,..,n]，所以w(A[1])可以快速得到
- 考虑从 w(A[i]) 推 w(A[i+1]):
  - 考虑第 (i!, 2i!] 个排列拼接起来的数组 B，它们相当于在 A[i] 中交换了所有的 n-i 与 n-i+1
  - 因此 w(B) 的特征数组可以从 w(A[i]) 交换一些行列得到
  - 其它段同理，从而 A[i+1] 被拆分成了 (i+1) 段 w 已知的数组

排列

- 当 P 是任意排列的时候同理，先预处理所有的 w(A[i])

- f(P) 可以被拆分成 O(n^2) 段数组，每一段 C[k] 都是从一个排列出发，按照字典序连接这个排列最后 i 个字符的所有排列顺序

- 所有的 w(C[k]) 都可以从 w(A[i]) 交换行列得到

- 总的时间复杂度为 O(n^5)

数论结构

- 给出一个 n*n 的数组，先 q1 次修改，再 q2 次询问
- 修改：把所有编号与 s 互质的行的区间 [l,r] 加上 x
- 询问：求所有编号与 s 互质的行 [l,r] 区间和的和
- n, q1 ≤ 50000，q2 ≤ 100000，s 随机

数论结构

- 考虑所有修改 si, li, ri, xi 对询问 s, l, r 的贡献

$$\sum_{i=1}^{q} \mathrm{overlap}(l_i,r_i,l,r)\,x_i \sum_{j=1}^{n} \overbrace{[(s_i,j)=1][(s,j)=1]}^{\text{反演}}$$

$$= \sum_{i=1}^{q} d(l_i,r_i,l,r)\,x_i \sum_{j=1}^{n} \left(\sum_{d_1|(s_i,j)} \mu(d_1) \sum_{d_2|(s,j)} \mu(d_2) \right).$$

$$= \sum_{d_2|s} \mu(d_2) \sum_{d_1=1}^{n} \left\lfloor \frac{n}{\mathrm{lcm}(d_1,d_2)} \right\rfloor \mu(d_1) \sum_{i\in Q_{d_1}} d(l_i,r_i,l,r)\,x_i \quad (Q_t\text{表示所有 } t|s_i \text{ 的修改})$$

数论结构

令 $D=(d_1,d_2)$, $d_1=$ 原 d_1/D, $d_2=$ 原 d_2/D

$$=\sum_{D|S}\sum_{d_2|\frac{S}{D}}\mu(d_2D)\sum_{d_1=1}^{\lfloor\frac{n}{D}\rfloor}\mu(d_1D)\left\lfloor\frac{n}{d_1Dd_2}\right\rfloor\underbrace{[(d_1,d_2)=1]}_{\text{反演}}\sum_{i\in Q_{d_1D}}\alpha(l_i,r_i,l,r)x_i$$

$$=\sum_{D|S}\sum_{d_2|\frac{S}{D}}\mu(d_2D)\sum_{d_1=1}^{\lfloor\frac{n}{D}\rfloor}\mu(d_1D)\left\lfloor\frac{n}{d_1Dd_2}\right\rfloor\sum_{d|(d_1,d_2)}\mu(d)\sum_{i\in Q_{d_1D}}\alpha(l_i,r_i,l,r)x_i$$

令 $D=$ 原 $D\times d$

$$=\sum_{D|S}\sum_{d|D}\mu(d)\sum_{d_2|\frac{S}{D}}\mu(d_2D)\sum_{d_1=1}^{\lfloor\frac{n}{D}\rfloor}\mu(d_1D)\left\lfloor\frac{n}{d_1Dd\,d_2}\right\rfloor\sum_{i\in Q_{d_1D}}\alpha(l_i,r_i,l,r)x_i$$

- 考虑如何用数据结构来维护这东西

数论结构

最外层枚举D，Q_t为S_i为t倍数的修改集合，A_t为S_i为t倍数的询问集合.

一次处理Q_D对A_D的贡献

对于修改：枚举$\sum_{d|D}$，$\sum_{d_1=1}^{\lfloor \frac{n}{D} \rfloor}$．把每个区间加$[l_i, r_i]+x_i$记录在第$\left\lfloor \frac{n}{d_1 D d} \right\rfloor$行

复杂度：$\sum_{D=1}^{n} \sum_{d|D} \sum_{d_1=1}^{\lfloor \frac{n}{D} \rfloor} |Q_{Dd_1}| = \sum_{D=1}^{n} \sum_{d|D} \sum_{d_1=1}^{\lfloor \frac{n}{D} \rfloor} \frac{n}{Dd_1}$　　只有$O(\sqrt{\frac{n}{D}})$种取值

$$= \sum_{D=1}^{n} \sum_{d|D} \frac{n}{D} \log n = \sum_{d=1}^{n} \sum_{D=1}^{\lfloor n/d \rfloor} \frac{n}{Dd} \log n = \sum_{d=1}^{n} \frac{n}{d} \log^2 n = n \log^3 n.$$

数论结构

预处理：修改 中在一个 $O(\sqrt{\frac{n}{D}})$ 行；n多)数组中进行了3次询问D

① 只有 $|Q_D|+|A_D|$ 行有用．先离散化

② 作二维前缀和

复杂度：$\sum_{D=1}^{n}\sqrt{\frac{n}{D}}\left(|Q_D|+|A_D|\right)=\sum_{D=1}^{n}\left(\frac{n}{D}\right)^{1.5}\approx\int_{1}^{n}\left(\frac{n}{x}\right)^{1.5}dx=O(n\sqrt{n})$

离散化：$\sum_{D=1}^{n}\left(|Q_D|+|A_D|\right)\log n=\sum_{D=1}^{n}\left\lfloor\frac{n}{d}\right\rfloor\log n=n\log^2 n$

数论结构

询问：枚举 d_1，那么第 i 行的贡献就为 $\lfloor \frac{i}{d_1} \rfloor$ 倍

枚举 $\lfloor \frac{i}{d_1} \rfloor$，注意 $i=\lfloor \frac{n}{d_2 D d} \rfloor$ ∵ $\lfloor \frac{i}{d_1} \rfloor$ 有 $\lfloor \lfloor \frac{n}{D d_1} \rfloor / x \rfloor$ 的形式 ∴ 有 $O(\sqrt{\frac{n}{D d_1}})$ 段

每一段是一个矩形和，预处理后 $O(1)$，复杂度：

$$\sum_{D=1}^{n} \sum_{d_1=1}^{\lfloor n/D \rfloor} \lfloor A d_1 D \rfloor \cdot \sqrt{\frac{n}{d_1 D}} = \sum_{D=1}^{n} \sum_{d_1=1}^{\lfloor n/D \rfloor} \left(\frac{n}{D d_1} \right)^{1.5}$$

$$\approx \int_1^n \int_1^{\frac{n}{x}} \left(\frac{n}{xy} \right)^{1.5} dy\, dx = O(n\sqrt{n}).$$

数论结构

- 总的时间复杂度为 O(nlog^3n+n^1.5)
- 推导部分是非常基础的莫比乌斯反演
- 难点在于如何处理 n/lcm 这个式子还有复杂度计算

最小割

- 给出一张无向图，它的所有 (i,i %n+1) 之前都连有一条边权 INF 的边
- 求两两之间最小割的和
- n <= 7000, m <= 100000

最小割

- 普及 Gomory-hu tree：所有点对间的最小割可以用 O(n) 次最小割求出来
- 当然这个数据范围不用最小割树也能做
- 考虑如何快速求最小割

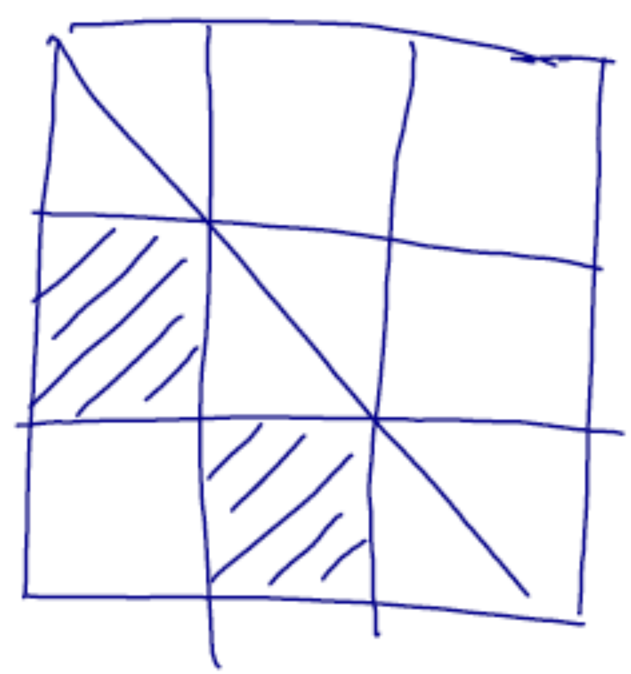
最小割

- 图很特殊：每个最小割肯定割两条 INF 边，最后肯定是割成两个半圆
- 令 A[i,j] 表示割开 [i,j] 与其他点时的额外代价，那么 A[i,j] 等于一端在 [i,j] 内，一端在 [i,j] 外的边权和
- 每一条边对 O(1) 个矩形内的 A[i,j] 产生贡献
- 可以二维前缀和 O(n^2) 处理出来

最小割

- 考虑询问，对于一对 s,t，答案等于所有满足 s,t 中只有一个在 [i,j] 内的所有 A[i,j] 的 min

- 大概是这样的两个矩形：



最小割

- 因此维护一下每一行的前缀 min，每一列的后缀 min
- 询问时枚举一个行/列即可做到 $O(n)$ 询问最小割
- 总的时间复杂度为 $O(n^2+m)$
- 用数据结构维护可以到 $O(n^{1.5}\log n+m)$，但是因为 day2 要降难度所有就削了，感兴趣的话可以想一下