IOI2019 中国国家集训队第一阶段作业

IOI 练习赛 08 报告

袁无为[*1], 苏凯[†1], and 郑钧天[‡1]

[1] 广州市第二中学

命题总结

数学是解决信息学问题的基本工具，是信息学竞赛中考查的重点。本场练习赛分别以数论函数求和、群论、生成函数等数学角度切入，综合考查了选手对搜索、状态压缩动态规划、（扩展）欧几里得算法、常见数论函数、容斥原理、"洲阁筛"、有限阿贝尔群、线性基、在线转离线、指数级生成函数、快速傅里叶变换、牛顿迭代法、线性微分方程等知识点的理解和熟练程度。除此之外，选手需要具备一定的恒等变换技巧、观察能力、代码阅读能力、数据结构设计能力、问题转化能力、代码实现能力、时间复杂度分析能力、与命题人的心理博弈能力以及健康稳定的心态，才能在赛场上顺利完成所有题目。

这三道题目难度均匀，综合性强，子任务丰富且难度、分值设置合理，有利于提高区分度。题目的解法十分自然；以第三题为例，依照题目的描述推出生成函数的转移式，然后考虑使用牛顿迭代法解多项式方程，顺利得到线性微分方程，使用快速傅里叶变换加速实现即可。

本场练习赛的选题属近年信息学竞赛的热门，紧跟潮流，盼能弥补部分选手在新技巧上的不足。除此之外，三道题也有各自的作用和目的：

1. 小水题：第一题是一道洲阁筛练习题，旨在考查选手数论的扎实程度；本题作为练习赛的开端，用于热身，为选手增添信心；
2. 中水题：这道放在中间的披着群论外壳的简单题，要求选手对现有数据结构（线性基）进行扩展，意在抛砖引玉，盼能给选手一点启迪，使选手逐步进入状态；
3. 大水题：本题既考查了常见多项式算法，又用微积分相关的知识引领着大家走进美妙的高等数学花园，为比赛画上圆满的句号。

正如题目名称所述，本场模拟赛的题目偏重基础，也许不能让选手们充分展现高超的解题能力。又因命题人能力有限，命题工作或有疏漏，请多包涵。出题人盼望，这套美妙的题目，可以给拼搏于 AK IOI2019 的逐梦之路上的你，提供一个有力的援助。

命题组组长 苏凯

[*] weeerrr720@qq.com

[†] i_sukai@live.com

[‡] 930234984@qq.com

1 小水题

命制：袁无为

1.1 题目大意

给你 n,m,k,x_i,y_i，
求

$$\sum_{a_1=1}^{m}\sum_{a_2=1}^{m}\cdots\sum_{a_n=1}^{m}\left(\sigma_0(\gcd(a_1,a_2,\ldots,a_n)^3)\right)^3\prod_{i=1}^{k}[a_{x_i}\le a_{y_i}] \tag{1}$$

子任务 1[5pts]：$n\le 5,m\le 10$；
子任务 2[15pts]：$n\le 13,m\le 13$；
子任务 3[30pts]：$m\le 10^7$；
子任务 4[30pts]：$k=0$；
子任务 5[20pts]：无特殊限制；
对于所有数据：$1\le n\le 20,1\le m\le 10^{10},0\le k\le n(n-1)$。

1.2 参考算法

1.2.1 算法一

对于第一个子任务，$n\le 5,m\le 10$。
直接暴搜就好了。
时间复杂度 $O(m^n)$，可以通过第一个子任务获得 5 分。

1.2.2 算法二

对于第二个子任务，$n\le 13,m\le 13$。
考虑状压 DP。
记 $f_{i,j,S}$ 为当前 S 中的点都已经选好了数，这些数都 $\le i$，且 $\gcd=j$ 的方案数。
转移的时候枚举哪些数为 $i+1$，然后转移。
时间复杂度 $O(n^2 3^n)$，可以通过前两个子任务获得 20 分。

1.2.3 算法三

对于第三个子任务，$m\le 10^7$。
推一下式子。

$$\sum_{a_1=1}^{m}\sum_{a_2=1}^{m}\cdots\sum_{a_n=1}^{m}\sigma_0(\gcd(a_1,a_2,\dots,a_n)^3)\prod_{l=1}^{k}[a_{x_l}\le a_{y_l}] \tag{2}$$

$$=\sum_{i=1}^{m}\sigma_0(i^3)^3\sum_{a_1=1}^{m}\sum_{a_2=1}^{m}\cdots\sum_{a_n=1}^{m}[\gcd(a_1,a_2,\dots,a_n)=i]\prod_{l=1}^{k}[a_{x_l}\le a_{y_l}] \tag{3}$$

$$=\sum_{i=1}^{m}(\sum_{j|i}\sigma_0(j^3)^3\mu(\frac{i}{j}))\sum_{a_1=1}^{m}\sum_{a_2=1}^{m}\cdots\sum_{a_n=1}^{m}[i\mid\gcd(a_1,a_2,\dots,a_n)]\prod_{l=1}^{k}[a_{x_l}\le a_{y_l}] \tag{4}$$

$$=\sum_{i=1}^{m}g(i)f(\lfloor\frac{m}{i}\rfloor) \tag{5}$$

$g(n)=\sum_{i|n}\sigma_0(i^3)^3\mu(\frac{n}{i})$，是一个积性函数，可以线性筛求。

$f(x)$ 为当 $m=x$ 的时候不同的情况数。

当 $x\le n$ 时可以状压 DP。对于一个限制 $a_{x_i}\le a_{y_i}$，连一条 $x_i\to y_i$ 的有向边。先把强连通分量缩成一个点，然后按照拓扑序 DP。记 $f_{i,j,S}$ 为当前 S 中的点都已经选好了数，选了的数都 $\le i$，当前正在决策拓扑序中第 j 个点的方案数。那么当前这个点能选 i 当且仅当图中有边连向 i 的都已经选了。

然后对于 $x>n$ 的情况，可以通过插值得到。

时间复杂度 $O(n^2 2^n+n\sqrt{m}+m)$，可以通过前三个子任务获得 50 分。

1.2.4 算法四

对于第四个子任务，$k=0$。

所以 $f(x)=x^n$。

容易发现 $g(n)$ 是一个积性函数，且

$$g(1)=1 \tag{6}$$

$$g(p)=1\times(-1)+4^3\times 1=63 \tag{7}$$

$$g(p^c)=(3(c-1)+1)^3(-1)+(3c+1)^3(1) \tag{8}$$

$$=(27c^3+27c^2+9c+1)-(27c^2-54c^2+36c-8) \tag{9}$$

$$=81c^2-27c+9 \tag{10}$$

所以可以用洲阁筛筛出对于任意的 i，在 $\lfloor\frac{n}{i}\rfloor$ 处 g 的前缀和。

时间复杂度 $O(\frac{m^{\frac{3}{4}}}{\log m}+\sqrt{m}\log n)$，可以通过第四个子任务获得 30 分。

1.2.5 算法五

对于第五个子任务，$n\le 20, m\le 10^{10}$。

把算法三中求 $f(x)$ 的方法和算法四中求 $g(x)$ 的方法结合在一起即可。

时间复杂度 $O(\frac{m^{\frac{3}{4}}}{\log m}+n\sqrt{m}+n^2 2^n)$，可以通过所有子任务获得 100 分！

2 中水题

命制：苏凯

2.1 题目大意

给定一个阿贝尔群和一个序列，每次询问序列一个区间的生成子群的大小。

2.2 参考算法

这是一道简单的披着群论外壳的数论数据结构设计题。部分子任务（2、3）需要一定的群论基础，但只要观察给出的辅助代码就可以得到正解。

以下时间复杂度省略辅助代码的 $\mathrm{O}(n^2)$。

2.2.1 算法一

由于 n, m, q 都很小，可以暴力地进行 BFS：每次往当前子群里面加入一个新的元素时，扫描一遍当前子群中的元素，看能不能运算得到新的元素。

时间复杂度为 $\mathrm{O}(nmq)$。

可通过子任务：子任务 1

期望得分：5 分

2.2.2 算法二

往当前子群里面加入新的元素 a 时，由于当前子群 H 一定是正规子群，所以一次性加入若干个陪集 $a^k H$，可以做到每次询问中每个元素只在加入的时候被扫描一次。

时间复杂度为 $\mathrm{O}(q(m+n))$。

可通过子任务：子任务 1、子任务 2

期望得分：15 分

2.2.3 算法三

由于题目允许离线，所以可以用一些套路来优化一下子任务 2 的做法。按询问的左端点排序，从右到左枚举询问左端点，同时维护一个每一个元素都能使子群大小增加的链表，链表中的元素是关键元素，把它和询问右端点做双指针即可；因为每次加入若干陪集，所以每加入一个元素子群大小至少翻倍，链表中的元素个数不超过 $\mathrm{O}(\log n)$，不是复杂度瓶颈。

时间复杂度为 $\mathrm{O}(\min(q, m)n)$。

可通过子任务：子任务 1、子任务 2、子任务 3

期望得分：25 分

2.2.4 算法四

观察辅助代码可以发现，n 为输入的 t 个数的乘积。当 n 为质数时，输入的 t 必然为 1。再观察 g 的生成方式，可以发现此时 G 就是个循环群（更为人熟知的是模域 F_P 上的加法）。根

据同余相关理论，只要 $[l,r]$ 之间有一个不是单位元的元素，那就可以生成整个群。简单的求个区间和就可以了。

时间复杂度因实现差异略有不同，最优为 $O(m+q)$。

可通过子任务：子任务 4

期望得分：5 分

2.2.5 算法五

经过简单的观察可以发现，每一个元素对每一个组成 n 的质数都形成了一个循环群，而且不同质数之间的循环群加法是互相独立的。那这个子任务的做法就和子任务 4 十分类似了，只是要多维护几个质数的信息。

时间复杂度为 $O(m\log n+q\log n)$。

可通过子任务：子任务 4、子任务 5

期望得分：10 分

2.2.6 算法六

这个群的运算此时就构成了普通自然数的按位异或运算。那就用子任务 3 的优化套路，用线性基维护当前的生成子群，答案即为 2 的线性独立的元素个数次方。

时间复杂度为 $O(\min(m,q)\log n+m+q)$。

可通过子任务：子任务 6

期望得分：5 分

2.2.7 算法七

复杂度瓶颈在于快速维护子群。标程是我乱搞的类似于线性基的东西。我所识甚少，不知道有没有把什么东西重新造了一遍。

有限阿贝尔群可以被分解成若干个循环群（或更为人熟知的模域 F_P）的直积，也就是每个元素用一个 k 元组 $(a_1,a_2,...,a_n)$ 表示，元素的乘法就是每一维对应相加，然后每一维模一下那一维的模数。[1]

如何分解？请参阅刘承奥同学的 2018 年国家集训队论文。为了不增加题目的毒瘤/困难程度，输入数据中已经给出了分解的结果。

然后把所有模数不是质数的幂的维拆开来，就是 $M=\prod p_i^{k_i}$，可以拆成很多维。这样每一维的模数都是质数幂了。这一步的合理性由中国剩余定理保证。

现在我们建一个“扩展线性基”：每一维可以存一个元素。这个线性基支持求出，用一些给定的元素，能线性组合出其他什么元素。一开始，每一维存的元素是单位元 e。

令函数 $\mathrm{ins}(g)$ 为插入了一个元素 g 到“扩展线性基”中。$\mathrm{ins}(g)$ 的简要流程如下：

- 如果 g 已经是单位元，则直接返回插入失败；
- 找 g 的非零最高维（插入次序随意，但需要在初始时确定），并尝试用在该维上存储的元素来消去 g 的这一维：

[1]The fundamental theorem of finite abelian groups

- 消去成功：设 g' 为消去这一维之后的元素，返回 $\text{ins}(g')$；
- 消去失败：
  * 设这一维存了一个元素叫 h，那么 g 一定能把 h 这一维消掉，设 h 这一维被消掉之后的元素是 h'，$\text{ins}(h')$；
  * 把 g 存到这一维上；
  * 找到最小的幂 t 使得 g^t 的这一维是 0，$\text{ins}(g^t)$；
  * 返回插入成功。

正确性是比较显然的。如何分析时间复杂度？假设模的是 r 元组 $(p_1^{k_1}, p_2^{k_2}, \cdots, p_r^{k_r})$，第 i 维里存的元素最多变 $k_i - 1$ 次，每变一次复杂度最多额外增加 $\mathrm{O}(\log n)$，那总体增加的复杂度是 $\mathrm{O}(\log^2 n)$，故总复杂度不变，均摊每次插入 $\mathrm{O}(\log n)$。

虽然这个做法的实现常数略大，但是复杂度常数应该很小，很多地方都不太好卡满。比如那个 log 在底为 2 的时候最大，但是在底为 2 的时候，插入的分支会立刻消失。

一个基于 Sylow-p 子群的优化：对于不同的质数，他们可以分别讨论。想提取出 g 模 p^k 的那些维，让其他变成单位元，只需要取

$$g^{\frac{n}{p^k}\left(\left(\frac{n}{p^k}\right)^{-1} \bmod p^k\right)} \tag{11}$$

该做法总时间复杂度为 $\mathrm{O}(n^2 + (m+q)\log n + \min(m,q)\log^2 n)$。需要注意的是，为了达到这个复杂度，实现时需要预处理扩展欧几里得算法的结果：可以使用 $\mathrm{O}(n^2)$DP 模拟，也可以 $\mathrm{O}(n \log n)$ 对每一维的每一个数进行暴力计算。

可通过子任务：全部子任务

期望得分：100 分

3 大水题

命制：郑钧天

3.1 题目大意

有一个 01 序列，初始时序列为空。你可以对序列进行两种操作：

1. 在序列末端插入一个 0。
2. 在序列中删去一个子序列，并在序列末端插入一个 1。这里对子序列的选取有一定限制，设子序列中包含 x 个 0，y 个 1，则你选取的子序列必须满足：
   - 子序列不可为空，即 $x + y > 0$
   - 当 $y > 0$ 时，$x \in A$，这里 A 为给定集合
   - 当 $y = 0$ 时，$x \in B$，这里 B 为给定集合

现在，你需要对序列执行 n 次操作。请你求出在所有不同的操作方案中，最终序列长度为 1 的方案有多少种。两种操作方案被视为不同，当且仅当某一次操作的种类不同，或某个第二类操作中选取的子序列不同。

3.2 部分分设置

- 子任务 1[5pts]：$n \le 10$
- 子任务 2[20pts]：$n \le 2000$
- 子任务 3[5pts]：$|A| = |B| = n$
- 子任务 4[30pts]：$A = B$
- 子任务 5[40pts]：无特殊限制

对于所有数据，$1 \le n \le 114514, 0 \le a_i, b_i < n$，$a_i$ 互不相同，b_i 互不相同。

3.3 针对较简单子任务的解法

3.3.1 算法一

直接 DP，记 $f_{i,j,k}$ 表示当前执行了 i 次操作，序列中存在 j 个 0、k 个 1 的方案数。转移时可以直接枚举 x, y，时间复杂度为 $O(n^5)$。

可通过子任务：子任务 1

期望得分：5 分

3.3.2 算法二

当 $|A| = |B| = n$ 时，操作 2 中选取任意子序列均为合法。我们记 t_i 表示在第 i 次操作中被加入的元素会在哪一次操作中被删去。只要对于 $i < n$ 均有 $t_i > i$，我们就能对应构造出唯一的操作方案。

因此，t_i 的取值有 $n - i$ 种，总方案数即为 $\prod_{i=1}^{n-1}(n-i) = (n-1)!$。

可通过子任务：子任务 3

期望得分：5 分

3.4 初步分析

我们把第 i 次操作加入的数看作编号为 i 的节点，并把操作 2 中删除的节点看作自己的儿子。这样整个操作序列就形成了一棵树，满足父亲的编号大于儿子，原本的 0 对应了叶子节点，1 对应了非叶子节点。对于任何非叶子节点，当该节点所有儿子均为叶子时，儿子个数在集合 B 中，否则叶子儿子的个数在集合 A 中。

为了处理起来更加方便，我们可以把 0 加入 B 中，这样叶子节点也能符合上述条件。

3.4.1 算法三

建立起树结构后，DP 起来就方便很多了。记 f_i 为 i 个节点的合法有根树个数，g_i 为 i 个节点的合法森林个数，且森林中每棵树节点个数不少于 2。DP 的转移方程为：

$$g_i = \sum_{j \geq 2} g_{i-j} f_j \binom{i-1}{j-1} \tag{12}$$

$$f_i = [i-1 \in B] + \sum_{j=0}^{i-2} [j \in A] \binom{i-1}{j} g_{i-1-j} \tag{13}$$

时间复杂度为 $O(n^2)$。

可通过子任务：子任务 1、子任务 2

期望得分：25 分

3.5 进一步分析

直接 DP 的时间复杂度过高，无法通过更大的测试点。我们尝试建立生成函数，用多项式技巧解出系数。

为了方便用生成函数处理，我们重写转移方程：

$$f_n = [n-1 \in B] + \sum_{i \in A} \sum_{k \geq 1} \sum_{a_1+a_2+\cdots+a_k = n-1-i, a_j \geq 2} \frac{1}{k!} f_{a_1} f_{a_2} \cdots f_{a_k} \frac{(n-1)!}{i! a_1! a_2! \cdots a_k!} \tag{14}$$

两边除以 $\frac{1}{n!}$：

$$\frac{f_n}{n!}=\frac{1}{n}[n-1\in B]\frac{1}{n-1}!+\frac{1}{n}\sum_{i\in A}\sum_{k\geq 1}\sum_{a_1+a_2+\cdots+a_k=n-1-i,a_j\geq 2}\frac{1}{k!}f_{a_1}f_{a_2}\cdots f_{a_k}\frac{1}{i!a_1!a_2!\cdots a_k!} \tag{15}$$

$$n\frac{f_n}{n!}=[n-1\in B]\frac{1}{n-1}!+\sum_{i\in A}\frac{1}{i!}\sum_{k\geq 1}\sum_{a_1+a_2+\cdots+a_k=n-1-i,a_j\geq 2}\frac{1}{k!}\frac{f_{a_1}}{a_1!}\frac{f_{a_2}}{a_2!}\cdots\frac{f_{a_k}}{a_k!} \tag{16}$$

记 $F(x)$ 为 f_n 的指数生成函数：$F(x)=\sum_{k\geq 0}\frac{f_k}{k!}x^k$，再记 $A(x)=\sum_{k\geq 0}[k\in A]\frac{1}{k!}x^k$，$B(x)=\sum_{k\geq 0}[k\in B]\frac{1}{k!}x^k$。从转移方程可以得到：

$$xF'(x)=xB(x)+xA(x)(e^{F(x)-x}-1) \tag{17}$$

化简得到：

$$F'(x)=A(x)e^{-x}e^{F(x)}+B(x)-A(x) \tag{18}$$

为了方便计算，我们记 $C(x)=A(x)e^{-x},D(x)=B(x)-A(x)$。

3.5.1 算法四

对于 $A=B$ 的情况，方程为 $F'(x)=C(x)e^{F(x)}$。直接对方程进行求解：

$$\frac{dF}{dx}=Ce^F \tag{19}$$

$$e^{-F}dF=Cdx \tag{20}$$

$$\int e^{-F}dF=\int Cdx \tag{21}$$

$$-e^{-F}=\int Cdx \tag{22}$$

$$F=-\ln(-\int Cdx) \tag{23}$$

直接对 $C(x)$ 积分并取对数即可。为使多项式 ln 有意义，我们可以钦定积分的常数项为 -1。

可通过子任务：子任务 4

期望得分：30 分

3.5.2 算法五

对于一般情况，我们要解的方程形如 $F'(x)=C(x)e^{F(x)}+D(x)$。由于非线性方程不便于直接求解，我们考虑使用牛顿迭代。

首先，我们有 $F(x)\equiv 0 \pmod{x^1}$。假设已经求出 $F_0(x)\equiv F(x) \pmod{x^{n/2}}$，将其代入方程并作泰勒展开，可以得到：

$$F' \equiv Ce^{F_0} + D + Ce^{F_0}((F - F_0) + (F - F_0)^2 + \cdots) \pmod{x^n} \tag{24}$$

我们记 $G(x) = Ce^{F_0}, H(x) = (1 - F_0)Ce^{F_0} + D$。由于 $(F - F_0)^2$ 及之后的项在模 x^n 意义下为 0，我们将其舍去并化简：

$$F' \equiv GF + H \pmod{x^n} \tag{25}$$

考虑先解出 U 满足 $U' = GU$：

$$\frac{dU}{dx} \equiv GU \pmod{x^n} \tag{26}$$

$$U^{-1}dU \equiv Gdx \pmod{x^n} \tag{27}$$

$$\int U^{-1}dU \equiv \int Gdx \pmod{x^n} \tag{28}$$

$$\ln |U| \equiv \int Gdx \pmod{x^n} \tag{29}$$

$$U \equiv e^{\int Gdx} \pmod{x^n} \tag{30}$$

记 $V = FU^{-1}$。将 $F = UV$ 代入方程，可得：

$$(UV)' \equiv GUV + H \pmod{x^n} \tag{31}$$

$$UV' + U'V \equiv GUV + H \pmod{x^n} \tag{32}$$

$$UV' + GUV \equiv GUV + H \pmod{x^n} \tag{33}$$

$$V' \equiv HU^{-1} \pmod{x^n} \tag{34}$$

$$V \equiv \int HU^{-1}dx \pmod{x^n} \tag{35}$$

$$F \equiv UV \equiv U\int HU^{-1}dx \pmod{x^n} \tag{36}$$

每一层迭代都只进行了 $O(1)$ 次多项式乘法，积分，求逆以及求 exp，因此单层的复杂度为 $O(n\log n)$。总时间复杂度为 $T(n) = T(n/2) + O(n\log n) = O(n\log n)$。

可通过子任务：全部子任务

期望得分：100 分