FOI2023 数论练习赛题解

张志心

Zhejiang University

2023 年 7 月 14 日

简要题意:
q 次询问，每次询问两个数的 LCA，其中数的“父亲”定义为它的最大约数。
$q \leq 10^6$, 所有数 $\leq 10^7$。

$O(n)$ 预处理所有数的最大约数：相当于求所有数的最小约数，考虑欧拉筛就是用一个数的最小约数来筛掉这个数的，那么直接在欧拉筛的同时记录一下这个数是被哪个数筛的即可。

$O(n)$ 预处理所有数的最大约数：相当于求所有数的最小约数，考虑欧拉筛就是用一个数的最小约数来筛掉这个数的，那么直接在欧拉筛的同时记录一下这个数是被哪个数筛的即可。
求 LCA 的过程：每次让 x, y 中较大的数变成它的“父亲”，直到两个数相等。

T2. 加三十三

简要题意:
n 个人，a 个前场 b 个后场 c 个全能剩下的无特长。选 m 个人，x 个前锋（从前场或全能选）y 个后卫（从后场或全能选）剩下的替补（随意），问选人方案数。
$n \leq 14, m \leq 50$

考虑 a 个前场队员的选择情况。枚举这 a 个人中有 i 个人被选为前锋，则从 a 个人中选 i 个人当前锋，有 $\binom{a}{i}$ 种方案；

考虑 a 个前场队员的选择情况。枚举这 a 个人中有 i 个人被选为前锋，则从 a 个人中选 i 个人当前锋，有 $\binom{a}{i}$ 种方案；再从 c 个全能中选 $x-i$ 个人当剩下的前锋，有 $\binom{c-i}{x-i}$ 种方案；

T2. 加三十三 - Solution

考虑 a 个前场队员的选择情况。枚举这 a 个人中有 i 个人被选为前锋，则从 a 个人中选 i 个人当前锋，有 $\binom{a}{i}$ 种方案；再从 c 个全能中选 $x-i$ 个人当剩下的前锋，有 $\binom{c-i}{x-i}$ 种方案；再从剩下的 $c-x+i$ 个全能和 b 个后场中选 y 个人当后卫，有 $\binom{b+c-x+i}{y}$ 种方案。

T2. 加三十三 - Solution

考虑 a 个前场队员的选择情况。枚举这 a 个人中有 i 个人被选为前锋，则从 a 个人中选 i 个人当前锋，有 $\binom{a}{i}$ 种方案；再从 c 个全能中选 $x-i$ 个人当剩下的前锋，有 $\binom{c-i}{x-i}$ 种方案；再从剩下的 $c-x+i$ 个全能和 b 个后场中选 y 个人当后卫，有 $\binom{b+c-x+i}{y}$ 种方案。最后从剩下的 $n-x-y$ 个人中选 $m-x-y$ 个人当替补，有 $\binom{n-x-y}{m-x-y}$ 种方案。

T2. 加三十三 - Solution

考虑 a 个前场队员的选择情况。枚举这 a 个人中有 i 个人被选为前锋，则从 a 个人中选 i 个人当前锋，有 $\binom{a}{i}$ 种方案；再从 c 个全能中选 $x-i$ 个人当剩下的前锋，有 $\binom{c-i}{x-i}$ 种方案；再从剩下的 $c-x+i$ 个全能和 b 个后场中选 y 个人当后卫，有 $\binom{b+c-x+i}{y}$ 种方案。最后从剩下的 $n-x-y$ 个人中选 $m-x-y$ 个人当替补，有 $\binom{n-x-y}{m-x-y}$ 种方案。
故答案就是

$$\sum_{i=0}^{a}\binom{a}{i}\binom{c-i}{x-i}\binom{b+c-x+i}{y}\binom{n-x-y}{m-x-y}$$

当然你可以把最后一项提出来（它与 i 无关）。

考虑 a 个前场队员的选择情况。枚举这 a 个人中有 i 个人被选为前锋，则从 a 个人中选 i 个人当前锋，有 $\binom{a}{i}$ 种方案；再从 c 个全能中选 $x-i$ 个人当剩下的前锋，有 $\binom{c-i}{x-i}$ 种方案；再从剩下的 $c-x+i$ 个全能和 b 个后场中选 y 个人当后卫，有 $\binom{b+c-x+i}{y}$ 种方案。最后从剩下的 $n-x-y$ 个人中选 $m-x-y$ 个人当替补，有 $\binom{n-x-y}{m-x-y}$ 种方案。
故答案就是

$$\sum_{i=0}^{a}\binom{a}{i}\binom{c-i}{x-i}\binom{b+c-x+i}{y}\binom{n-x-y}{m-x-y}$$

当然你可以把最后一项提出来（它与 i 无关）。
预处理阶乘及其逆元，复杂度 $O(n)$。

写在题后

对于组合计数的问题，确定选择顺序是非常重要的部分。顺序错误很可能导致你的程序从 $O(n)$ 变成 $O(n^2)$ 甚至更离谱。

简要题意:
定义 $f(n)$ 以下问题的答案:
从 $1, 2, \ldots, n$ 中选出一些数使得选出的任意两个数之积都不是立方数,
问最多能选出多少个。
求 $f(1), f(2), \ldots, f(n)$ 的值。
$1 \le n \le 10^7$。
$n \le 5 \times 10^5$.

T3. 未命名一 - Solution

算法一：手玩，暴力，打表。

算法一：手玩，暴力，打表。
算法二：记

$$[x] = \{k^3x | k \in \mathbb{N}^*, 1 \leq k^3x \leq n\}$$
$$\overline{x} = \min\{y \in \mathbb{N}^* | xy = k^3, k \in \mathbb{N}^*\}$$

算法一：手玩，暴力，打表。
算法二：记

$$[x] = \{k^3x | k \in \mathbb{N}^*, 1 \leq k^3x \leq n\}$$
$$\overline{x} = \min\{y \in \mathbb{N}^* | xy = k^3, k \in \mathbb{N}^*\}$$

其中规定 x 无立方因子（即不能被 1 以外任何立方数整除）。

算法一：手玩，暴力，打表。
算法二：记

$$[x]=\{k^3x|k\in\mathbb{N}^*,1\le k^3x\le n\}$$
$$\overline{x}=\min\{y\in\mathbb{N}^*|xy=k^3,k\in\mathbb{N}^*\}$$

其中规定 x 无立方因子（即不能被 1 以外任何立方数整除）。
容易发现：
1. [1] 中只能选择一个数。

算法一：手玩，暴力，打表。
算法二：记

$$[x] = \{k^3x | k \in \mathbb{N}^*, 1 \le k^3x \le n\}$$
$$\overline{x} = \min\{y \in \mathbb{N}^* | xy = k^3, k \in \mathbb{N}^*\}$$

其中规定 x 无立方因子（即不能被 1 以外任何立方数整除）。
容易发现：
1. $[1]$ 中只能选择一个数。
2. 若 $x \neq 1$，$[x]$ 要么不选，要么全选。

算法一：手玩，暴力，打表。
算法二：记

$$[x] = \{k^3x | k \in \mathbb{N}^*, 1 \le k^3x \le n\}$$
$$\overline{x} = \min\{y \in \mathbb{N}^* | xy = k^3, k \in \mathbb{N}^*\}$$

其中规定 x 无立方因子（即不能被 1 以外任何立方数整除）。
容易发现：
1. $[1]$ 中只能选择一个数。
2. 若 $x \neq 1$，$[x]$ 要么不选，要么全选。
3. $[x]$ 和 $[\overline{x}]$ 只能二选一。

因此我们有一个大体思路：

T3. 未命名一 - Solution

因此我们有一个大体思路：
预处理出 $[x]$ 和 $\overline{x}$，并维护所有 $[x]$ 当前的大小，贪心选择 $[x]$ 和 $[\overline{x}]$ 中较大者即可。

因此我们有一个大体思路：
预处理出 $[x]$ 和 $\overline{x}$，并维护所有 $[x]$ 当前的大小，贪心选择 $[x]$ 和 $[\overline{x}]$ 中较大者即可。
如果直接暴力求 $\overline{x}$，复杂度为 $O(n^2)$，可以得到 60 分的高分。

因此我们有一个大体思路：
预处理出 $[x]$ 和 $\overline{x}$，并维护所有 $[x]$ 当前的大小，贪心选择 $[x]$ 和 $[\overline{x}]$ 中较大者即可。
如果直接暴力求 $\overline{x}$，复杂度为 $O(n^2)$，可以得到 60 分的高分。
如果直接暴力分解质因数，复杂度为 $O(n\sqrt{n})$，可以得到 70 分的高分。

T3. 未命名一 - Solution

算法三：考虑如何高效预处理 $[x]$。

算法三：考虑如何高效预处理 $[x]$。
类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。

算法三：考虑如何高效预处理 $[x]$。
类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。
这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。

算法三：考虑如何高效预处理 $[x]$。
类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。
这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。
下面考虑预处理 $\overline{x}$。

T3. 未命名一 - Solution

算法三：考虑如何高效预处理 $[x]$。
类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。
这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。
下面考虑预处理 $\overline{x}$。
注意到 $\bar{\ }$ 是积性函数，因此可直接线性筛。

算法三：考虑如何高效预处理 $[x]$。

类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。

这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。

下面考虑预处理 $\overline{x}$。

注意到 $\bar{}$ 是积性函数，因此可直接线性筛。

根据线性筛的原理，我们需要知道 $\overline{p^k}$。

T3. 未命名一 - Solution

算法三：考虑如何高效预处理 $[x]$。
类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。
这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。
下面考虑预处理 $\overline{x}$。
注意到 $\overline{\cdot}$ 是积性函数，因此可直接线性筛。
根据线性筛的原理，我们需要知道 $\overline{p^k}$。
由定义，$\overline{p}=p^2,\overline{p^2}=p$。

T3. 未命名一 - Solution

算法三：考虑如何高效预处理 $[x]$。

类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。

这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。

下面考虑预处理 $\overline{x}$。

注意到 $\overline{\cdot}$ 是积性函数，因此可直接线性筛。

根据线性筛的原理，我们需要知道 $\overline{p^k}$。

由定义，$\overline{p}=p^2,\overline{p^2}=p$。

有立方因子的数的 $\overline{x}$ 没有定义。因此我们直接令 $\overline{p^k}=0(k\geq 3)$。

T3. 未命名一 - Solution

算法三：考虑如何高效预处理 $[x]$。

类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。

这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。

下面考虑预处理 $\overline{x}$。

注意到 $\bar{\cdot}$ 是积性函数，因此可直接线性筛。

根据线性筛的原理，我们需要知道 $\overline{p^k}$。

由定义，$\overline{p} = p^2, \overline{p^2} = p$。

有立方因子的数的 $\overline{x}$ 没有定义。因此我们直接令 $\overline{p^k} = 0 (k \geq 3)$。

这样我们就能线性求出 $\overline{x}$ 了。

T3. 未命名一 - Solution

算法三：考虑如何高效预处理 $[x]$。
类似于枚举倍数，我们枚举 k，将 k^3x 加入 $[x]$ 中。
这一步预处理的复杂度仅为 $\sum_{k=1}^{\infty}\lfloor\frac{n}{k^3}\rfloor$。
下面考虑预处理 $\overline{x}$。
注意到 $\bar{\cdot}$ 是积性函数，因此可直接线性筛。
根据线性筛的原理，我们需要知道 $\overline{p^k}$。
由定义，$\overline{p} = p^2, \overline{p^2} = p$。
有立方因子的数的 $\overline{x}$ 没有定义。因此我们直接令 $\overline{p^k} = 0 (k \geq 3)$。
这样我们就能线性求出 $\overline{x}$ 了。
总复杂度 $O(n)$。

1 T1. 数上 LCA

2 T2. 加三十三

3 T3. 未命名一

4 T4. 虚空之梦

简要题意：
给一个 $n \times m$ 的矩阵，求所有子矩阵和的 k 次方和。
$1 \leq n \times m \leq 2 \times 10^5, 0 \leq k \leq 10$。

算法一：$n, m \le 100$
【模板】二维前缀和。
复杂度 $O(n^2 m^2 \log k)$，期望得分 14。

T4. 虚空之梦 - Solution

算法一：$n, m \leq 100$
【模板】二维前缀和。
复杂度 $O(n^2m^2\log k)$，期望得分 14。
算法二：$k = 0$
简单排列组合问题。
答案就是 $\dfrac{nm(n-1)(m-1)}{4}$。
期望得分 6。

T4. 虚空之梦 - Solution

算法三：$n = 1$
前缀和 + 二项式定理展开，简单数学推导。

算法三：$n = 1$

前缀和 + 二项式定理展开，简单数学推导。

$$
\begin{aligned}
&\sum_{i=0}^{n}\sum_{j=i}^{n}(s_j - s_i)^K \\
=&\sum_{i=0}^{n}\sum_{j=i}^{n}\sum_{k=0}^{K}(-1)^k\binom{K}{k}s_j^k s_i^{K-k} \\
=&\sum_{k=0}^{K}(-1)^k\binom{K}{k}\sum_{i=0}^{n}s_i^{K-k}\sum_{j=i}^{n}s_j^k
\end{aligned}
$$

算法三：$n = 1$
前缀和 + 二项式定理展开，简单数学推导。

$$
\begin{aligned}
&\sum_{i=0}^{n}\sum_{j=i}^{n}(s_j - s_i)^K \\
=&\sum_{i=0}^{n}\sum_{j=i}^{n}\sum_{k=0}^{K}(-1)^k\binom{K}{k}s_j^k s_i^{K-k} \\
=&\sum_{k=0}^{K}(-1)^k\binom{K}{k}\sum_{i=0}^{n}s_i^{K-k}\sum_{j=i}^{n}s_j^k
\end{aligned}
$$

复杂度 $O(mk)$，期望得分 16。

算法三：$n = 1$
前缀和 + 二项式定理展开，简单数学推导。

$$\begin{aligned}&\sum_{i=0}^{n}\sum_{j=i}^{n}(s_j - s_i)^K\\=&\sum_{i=0}^{n}\sum_{j=i}^{n}\sum_{k=0}^{K}(-1)^k\binom{K}{k}s_j^k s_i^{K-k}\\=&\sum_{k=0}^{K}(-1)^k\binom{K}{k}\sum_{i=0}^{n}s_i^{K-k}\sum_{j=i}^{n}s_j^k\end{aligned}$$

复杂度 $O(mk)$，期望得分 16。
算法四：
注意到 $nm \le 2 \times 10^5$，因此 $\min\{n, m\} \le 450$。

算法三：$n=1$
前缀和 + 二项式定理展开，简单数学推导。

$$
\begin{aligned}
&\sum_{i=0}^{n}\sum_{j=i}^{n}(s_j-s_i)^K \\
=&\sum_{i=0}^{n}\sum_{j=i}^{n}\sum_{k=0}^{K}(-1)^k\binom{K}{k}s_j^k s_i^{K-k} \\
=&\sum_{k=0}^{K}(-1)^k\binom{K}{k}\sum_{i=0}^{n}s_i^{K-k}\sum_{j=i}^{n}s_j^k
\end{aligned}
$$

复杂度 $O(mk)$，期望得分 16。
算法四：
注意到 $nm \le 2\times 10^5$，因此 $\min\{n,m\}\le 450$。
我们直接枚举子矩阵在短边一侧的两个端点，在另一侧进行算法三即可。

算法三：$n = 1$
前缀和 + 二项式定理展开，简单数学推导。

$$\sum_{i=0}^{n}\sum_{j=i}^{n}(s_j - s_i)^K$$

$$=\sum_{i=0}^{n}\sum_{j=i}^{n}\sum_{k=0}^{K}(-1)^k\binom{K}{k}s_j^k s_i^{K-k}$$

$$=\sum_{k=0}^{K}(-1)^k\binom{K}{k}\sum_{i=0}^{n}s_i^{K-k}\sum_{j=i}^{n}s_j^k$$

复杂度 $O(mk)$，期望得分 16。
算法四：
注意到 $nm \le 2\times 10^5$，因此 $\min\{n, m\} \le 450$。
我们直接枚举子矩阵在短边一侧的两个端点，在另一侧进行算法三即可。
复杂度 $O(\min\{n^2mk, nm^2k\})$，期望得分 100。