时空穿梭

清华大学交叉信息研究院　贯志鹏

题意简述

在一个 n 维欧式空间中，选定 c $(c \geq 2)$ 个点，要求满足下面三个条件：

1. 每个点的每一维坐标均为正整数，且第 i 维坐标不超过 m_i。
2. 第 $i+1$ $(1 \leq i < c)$ 个点的每一维坐标都必须严格大于第 i 个点。
3. 存在一条直线经过所选的所有点。

计算方案数 $\bmod 10,007$ 后的值。每个测试点包含 T 组数据。

数据范围

测试点编号	T	n	c	m_i
1	$= 1,000$	$= 1$	≤ 20	$\le 100,000$
2	$= 3$	≤ 4	≤ 20	≤ 30
3	$= 3$	$= 2$	$= 3$	$\le 100,000$
4	$= 1,000$	$= 2$	$= 3$	$\le 100,000$
5	$= 20$	≤ 5	$= 3$	$\le 100,000$
6	$= 100$	≤ 11	$= 3$	$\le 100,000$
7	$= 1$	≤ 5	≤ 20	$\le 100,000$
8	$= 20$	≤ 5	≤ 20	$\le 100,000$
9	$= 100$	≤ 11	≤ 20	$\le 100,000$
10	$= 100$	≤ 11	≤ 20	$\le 100,000$

得分情况

- 集训队
  - 平均值：14.38、中位数：10、标准差：17.97
  - 70 分：徐寅展（浙江）
  - 60 分：黄志翱（湖南）、张恒捷（浙江）
  - 50 分：匡正非（湖南）、俞鼎力（浙江）
  - 40 分：吕可凡（江苏）、何琦（天津）
  - 30 分：岑若虚（浙江）
- 非集训队
  - 平均值：4.55、中位数：0、标准差：8.97
  - 60 分：吕凯风（湖北）、张羿翔（上海）、杜瑜皓（浙江）、金策（浙江）
  - 50 分：孔彦（吉林）

$n = 1$ 的情况

在一行 m_1 个点中选 c 个，无论如何选都一定共线，因此答案为 $\binom{m_1}{c}$。

$n = 1$ 的情况

在一行 m_1 个点中选 c 个，无论如何选都一定共线，因此答案为 $\binom{m_1}{c}$。

注意到 $m_i \le 10^5$，$c \le 20$，因此能利用 $\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}$ 预处理出所有需要的组合数。

$n = 1$ 的情况

在一行 m_1 个点中选 c 个，无论如何选都一定共线，因此答案为 $\binom{m_1}{c}$。

注意到 $m_i \le 10^5$，$c \le 20$，因此能利用 $\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}$ 预处理出所有需要的组合数。

期望得分：10 分。难度水平：NOIP 普及组。

$n = 2$ 的情况

如果选择的点中，最左下角点的坐标为 (x_1, y_1)、最右上角的点坐标为 (x_2, y_2)，那么在这两个点中间的整数坐标点个数为 $\gcd(x_2 - x_1, y_2 - y_1) - 1$，在这些点中选择剩下的 $c - 2$ 的点。

$n = 2$ 的情况

如果选择的点中，最左下角点的坐标为 (x_1, y_1)、最右上角的点坐标为 (x_2, y_2)，那么在这两个点中间的整数坐标点个数为 $\gcd(x_2 - x_1, y_2 - y_1) - 1$，在这些点中选择剩下的 $c - 2$ 的点。

注意到这里仅与最左上和最右下点坐标的差值相关，因此可以只枚举差值，由此可得答案为

$$\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\binom{\gcd(x,y)-1}{c-2}(m_1-x)(m_2-y)$$

$n > 2$ 的情况

对于 $n = 2$ 的计算方法，能发现可以扩展到 $n > 2$ 的情况，因此答案为

$$\sum_{x_1=1}^{m_1} \sum_{x_2=1}^{m_2} \cdots \sum_{x_n=1}^{m_n} \binom{\gcd(x_1, x_2, \dots, x_n) - 1}{c - 2} (m_1 - x_1) \cdots (m_n - x_n)$$

$n > 2$ 的情况

对于 $n = 2$ 的计算方法，能发现可以扩展到 $n > 2$ 的情况，因此答案为

$$\sum_{x_1=1}^{m_1} \sum_{x_2=1}^{m_2} \cdots \sum_{x_n=1}^{m_n} \binom{\gcd(x_1, x_2, \ldots, x_n) - 1}{c - 2} (m_1 - x_1) \cdots (m_n - x_n)$$

预处理出需要的组合数后，用上面的式子直接计算，可以通过前两个测试点。

期望得分：20 分。难度水平：NOIP 提高组。

$n > 2$ 的情况

对于 $n = 2$ 的计算方法，能发现可以扩展到 $n > 2$ 的情况，因此答案为

$$\sum_{x_1=1}^{m_1}\sum_{x_2=1}^{m_2}\cdots\sum_{x_n=1}^{m_n}\binom{\gcd(x_1,x_2,\ldots,x_n)-1}{c-2}(m_1-x_1)\cdots(m_n-x_n)$$

预处理出需要的组合数后，用上面的式子直接计算，可以通过前两个测试点。

期望得分：20 分。难度水平：NOIP 提高组。可惜的是，总共只有 31 $(13+18)$ 名同学分数大于等于 20 分。

预备知识

对于布尔表达式 $stat$，$[stat]$ 当 $stat$ 为真时取值 1，否则取值 0。

预备知识

对于布尔表达式 $stat$，$[stat]$ 当 $stat$ 为真时取值 1，否则取值 0。

用 μ 表示 Mobius 函数，有 $[n=1]=\sum_{d|n}\mu(d)$。

预备知识

对于布尔表达式 $stat$，$[stat]$ 当 $stat$ 为真时取值 1，否则取值 0。

用 μ 表示 Mobius 函数，有 $[n=1]=\sum_{d|n}\mu(d)$。

Mobius 函数为积性函数，令 n 的质因数分解形式为 $p_1^{c_1}p_2^{c_2}\cdots p_k^{c_k}$。若 c_i 中存在大于 1 的元素，$\mu(n)=0$，否则 $\mu(n)=(-1)^k$。

预备知识

对于布尔表达式 $stat$，$[stat]$ 当 $stat$ 为真时取值 1，否则取值 0。

用 μ 表示 Mobius 函数，有 $[n=1]=\sum_{d|n}\mu(d)$。

Mobius 函数为积性函数，令 n 的质因数分解形式为 $p_1^{c_1}p_2^{c_2}\cdots p_k^{c_k}$。若 c_i 中存在大于 1 的元素，$\mu(n)=0$，否则 $\mu(n)=(-1)^k$。

使用 Euler 筛法可以在 $O(n)$ 的时间内求出 $\mu(1),\mu(2),\ldots,\mu(n)$。

进一步分析 $n = 2$ 的情况

$$\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\binom{\gcd(x,y)-1}{c-2}(m_1-x)(m_2-y)$$

进一步分析 $n=2$ 的情况

$$
\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\binom{\gcd(x,y)-1}{c-2}(m_1-x)(m_2-y)
$$

$$
=\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\sum_{k=1}^{\max(m_1,m_2)}[\gcd(x,y)=k]\binom{k-1}{c-2}(m_1-x)(m_2-y)
$$

进一步分析 $n=2$ 的情况

$$
\begin{aligned}
&\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\binom{\gcd(x,y)-1}{c-2}(m_1-x)(m_2-y)\\
=&\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\sum_{k=1}^{\max(m_1,m_2)}[\gcd(x,y)=k]\binom{k-1}{c-2}(m_1-x)(m_2-y)\\
=&\sum_{k=1}^{m}\sum_{x=1}^{\lfloor m_1/k\rfloor}\sum_{y=1}^{\lfloor m_2/k\rfloor}[\gcd(x,y)=1]\binom{k-1}{c-2}(m_1-kx)(m_2-ky)
\end{aligned}
$$

进一步分析 $n = 2$ 的情况

$$\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\binom{\gcd(x,y)-1}{c-2}(m_1-x)(m_2-y)$$

$$=\sum_{x=1}^{m_1}\sum_{y=1}^{m_2}\sum_{k=1}^{\max(m_1,m_2)}[\gcd(x,y)=k]\binom{k-1}{c-2}(m_1-x)(m_2-y)$$

$$=\sum_{k=1}^{m}\sum_{x=1}^{\lfloor m_1/k\rfloor}\sum_{y=1}^{\lfloor m_2/k\rfloor}[\gcd(x,y)=1]\binom{k-1}{c-2}(m_1-kx)(m_2-ky)$$

$$=\sum_{k=1}^{m}\sum_{x=1}^{\lfloor m_1/k\rfloor}\sum_{y=1}^{\lfloor m_2/k\rfloor}\binom{k-1}{c-2}(m_1-kx)(m_2-ky)\sum_{d|x\text{且}d|y}\mu(d)$$

进一步分析 $n = 2$ 的情况

$$\sum_{k=1}^{m} \sum_{x=1}^{\lfloor m_1/k \rfloor} \sum_{y=1}^{\lfloor m_2/k \rfloor} \binom{k-1}{c-2} (m_1 - kx)(m_2 - ky) \sum_{d|x \text{且} d|y} \mu(d)$$

进一步分析 $n=2$ 的情况

$$\sum_{k=1}^{m}\sum_{x=1}^{\lfloor m_1/k\rfloor}\sum_{y=1}^{\lfloor m_2/k\rfloor}\binom{k-1}{c-2}(m_1-kx)(m_2-ky)\sum_{d|x\text{且}d|y}\mu(d)$$

$$=\sum_{k=1}^{m}\binom{k-1}{c-2}\sum_{d=1}^{m}\mu(d)\sum_{x=1}^{\left\lfloor\frac{m_1}{kd}\right\rfloor}\sum_{y=1}^{\left\lfloor\frac{m_2}{kd}\right\rfloor}(m_1-kdx)(m_2-kdy)$$

进一步分析 $n = 2$ 的情况

$$\sum_{k=1}^{m}\sum_{x=1}^{\lfloor m_1/k\rfloor}\sum_{y=1}^{\lfloor m_2/k\rfloor}\binom{k-1}{c-2}(m_1-kx)(m_2-ky)\sum_{d|x\text{且}d|y}\mu(d)$$

$$= \sum_{k=1}^{m}\binom{k-1}{c-2}\sum_{d=1}^{m}\mu(d)\sum_{x=1}^{\left\lfloor\frac{m_1}{kd}\right\rfloor}\sum_{y=1}^{\left\lfloor\frac{m_2}{kd}\right\rfloor}(m_1-kdx)(m_2-kdy)$$

$$= \sum_{i=1}^{m}\left(\sum_{x=1}^{\lfloor m_1/i\rfloor}\sum_{y=1}^{\lfloor m_2/i\rfloor}(m_1-ix)(m_2-iy)\right)\left(\sum_{k|i}\binom{k-1}{c-2}\mu(i/k)\right)$$

进一步分析 $n=2$ 的情况

$$\sum_{k=1}^{m}\sum_{x=1}^{\lfloor m_1/k\rfloor}\sum_{y=1}^{\lfloor m_2/k\rfloor}\binom{k-1}{c-2}(m_1-kx)(m_2-ky)\sum_{d|x\text{且}d|y}\mu(d)$$

$$=\sum_{k=1}^{m}\binom{k-1}{c-2}\sum_{d=1}^{m}\mu(d)\sum_{x=1}^{\lfloor\frac{m_1}{kd}\rfloor}\sum_{y=1}^{\lfloor\frac{m_2}{kd}\rfloor}(m_1-kdx)(m_2-kdy)$$

$$=\sum_{i=1}^{m}\left(\sum_{x=1}^{\lfloor m_1/i\rfloor}\sum_{y=1}^{\lfloor m_2/i\rfloor}(m_1-ix)(m_2-iy)\right)\left(\sum_{k|i}\binom{k-1}{c-2}\mu(i/k)\right)$$

$$=\sum_{i=1}^{m}\left(\sum_{x=1}^{\lfloor m_1/i\rfloor}(m_1-ix)\right)\left(\sum_{y=1}^{\lfloor m_2/i\rfloor}(m_2-iy)\right)f(i,c)$$

向 $n > 2$ 的情况扩展

$$\sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(\sum_{x=1}^{\lfloor m_t/i \rfloor} (m_t - ix) \right)$$

向 $n > 2$ 的情况扩展

$$
\begin{aligned}
& \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(\sum_{x=1}^{\lfloor m_t/i \rfloor} (m_t - ix) \right) \\
= & \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(m_t \lfloor m_t/i \rfloor - \frac{(\lfloor m_t/i \rfloor + 1) \lfloor m_t/i \rfloor i}{2} \right)
\end{aligned}
$$

向 $n > 2$ 的情况扩展

$$
\begin{aligned}
& \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(\sum_{x=1}^{\lfloor m_t/i \rfloor} (m_t - ix) \right) \\
= \; & \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(m_t \lfloor m_t/i \rfloor - \frac{(\lfloor m_t/i \rfloor + 1) \lfloor m_t/i \rfloor i}{2} \right)
\end{aligned}
$$

能看出，预处理出 $f(1,c), f(2,c), \ldots, f(m,c)$ 后，上式可以在 $O(nm)$ 的时间内计算。

向 $n > 2$ 的情况扩展

$$
\begin{aligned}
& \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(\sum_{x=1}^{\lfloor m_t/i \rfloor} (m_t - ix) \right) \\
= \ & \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(m_t \lfloor m_t/i \rfloor - \frac{(\lfloor m_t/i \rfloor + 1) \lfloor m_t/i \rfloor i}{2} \right)
\end{aligned}
$$

能看出，预处理出 $f(1,c), f(2,c), \ldots, f(m,c)$ 后，上式可以在 $O(nm)$ 的时间内计算。$f(1,c), f(2,c), \ldots, f(m,c)$ 能够在 $O(m \log m)$ 的时间内预处理。

向 $n>2$ 的情况扩展

$$\begin{aligned} & \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(\sum_{x=1}^{\lfloor m_t/i \rfloor} (m_t - ix) \right) \\ = & \sum_{i=1}^{m} f(i,c) \prod_{t=1}^{n} \left(m_t \lfloor m_t/i \rfloor - \frac{(\lfloor m_t/i \rfloor + 1) \lfloor m_t/i \rfloor i}{2} \right) \end{aligned}$$

能看出，预处理出 $f(1,c), f(2,c), \ldots, f(m,c)$ 后，上式可以在 $O(nm)$ 的时间内计算。$f(1,c), f(2,c), \ldots, f(m,c)$ 能够在 $O(m \log m)$ 的时间内预处理。

预处理时间复杂度：$O(mc \log m)$。单次询问时间复杂度：$O(nm)$。期望得分：60 分（包含前两个测试点）。

继续优化

注意到 $\prod_{t=1}^{n}\left(m_t\left\lfloor m_t/i\right\rfloor-\left(\left\lfloor m_t/i\right\rfloor+1\right)\left\lfloor m_t/i\right\rfloor i/2\right)$ 为关于 i 的 n 次多项式，不难在 $O\left(n^2\right)$ 的时间内求出 $n+1$ 个系数。

继续优化

注意到 $\prod_{t=1}^{n}\left(m_t\left\lfloor m_t/i\right\rfloor-\left(\left\lfloor m_t/i\right\rfloor+1\right)\left\lfloor m_t/i\right\rfloor i/2\right)$ 为关于 i 的 n 次多项式，不难在 $O\left(n^2\right)$ 的时间内求出 $n+1$ 个系数。

继续观察可得 $\left\lfloor m_t/i\right\rfloor$ 最多只有 $2\sqrt{m_t}$ 种不同取值，每种取值对应的 i 是连续的一段。这样可以将 $i=1\ldots m$ 分出最多 $2n\sqrt{m}$ 段，使得每段内对所有 t，$\left\lfloor m_t/i\right\rfloor$ 均相同。这同时也意味着，在一段内前面提及的关于 i 的多项式是相同，可以一起计算他们的和。

继续优化

注意到 $\prod_{t=1}^{n}\left(m_t\left\lfloor m_t/i\right\rfloor-\left(\left\lfloor m_t/i\right\rfloor+1\right)\left\lfloor m_t/i\right\rfloor i/2\right)$ 为关于 i 的 n 次多项式，不难在 $O\left(n^2\right)$ 的时间内求出 $n+1$ 个系数。

继续观察可得 $\lfloor m_t/i\rfloor$ 最多只有 $2\sqrt{m_t}$ 种不同取值，每种取值对应的 i 是连续的一段。这样可以将 $i=1\dots m$ 分出最多 $2n\sqrt{m}$ 段，使得每段内对所有 t，$\lfloor m_t/i\rfloor$ 均相同。这同时也意味着，在一段内前面提及的关于 i 的多项式是相同，可以一起计算他们的和。预处理出 $f(i,c)i^u$ $(u=0\dots n)$ 的部分和后，就能够直接算出一整段的和了。

按照之前的做法，预处理 $f(i,c)i^u$ 的部分和需要 $O(mnc\log m)$ 的时间。

继续优化

按照之前的做法，预处理 $f(i,c)i^u$ 的部分和需要 $O(mnc\log m)$ 的时间。

由于 $f(i,c)=\sum_{k|i}\binom{k-1}{c-2}\mu(i/k)$，其中 $\binom{k-1}{c-2}$ 为关于 k 的 $c-2$ 次多项式。

继续优化

按照之前的做法，预处理 $f(i,c)i^u$ 的部分和需要 $O(mnc\log m)$ 的时间。

由于 $f(i,c) = \sum_{k|i} \binom{k-1}{c-2}\mu(i/k)$，其中 $\binom{k-1}{c-2}$ 为关于 k 的 $c-2$ 次多项式。令 $g(i,j) = \sum_{k|i} k^j\mu(i/k)$，固定 j 后 $g(i,j)$ 为积性函数，可以用 $O(m)$ 的时间预处理。

继续优化

按照之前的做法，预处理 $f(i,c)i^u$ 的部分和需要 $O(mnc\log m)$ 的时间。

由于 $f(i,c)=\sum_{k|i}\binom{k-1}{c-2}\mu(i/k)$，其中 $\binom{k-1}{c-2}$ 为关于 k 的 $c-2$ 次多项式。令 $g(i,j)=\sum_{k|i}k^j\mu(i/k)$，固定 j 后 $g(i,j)$ 为积性函数，可以用 $O(m)$ 的时间预处理。这样预处理的时间复杂度优化为 $O\left(mc(n+c)\right)$。

继续优化

按照之前的做法，预处理 $f(i,c)i^u$ 的部分和需要 $O(mnc\log m)$ 的时间。

由于 $f(i,c)=\sum_{k|i}\binom{k-1}{c-2}\mu(i/k)$，其中 $\binom{k-1}{c-2}$ 为关于 k 的 $c-2$ 次多项式。令 $g(i,j)=\sum_{k|i}k^j\mu(i/k)$，固定 j 后 $g(i,j)$ 为积性函数，可以用 $O(m)$ 的时间预处理。这样预处理的时间复杂度优化为 $O\left(mc(n+c)\right)$。

预处理时间复杂度：$O\left(mc(n+c)\right)$。单次询问时间复杂度：$O\left(n^3\sqrt{m}\right)$。期望得分：100 分。

使用 GPU 加速运算

每组数据的运算之间是完全独立的，考虑到现在 GPGPU 十分火热（和比特币关系比较大），我尝试了一下使用 GPU 来加速计算。

使用 GPU 加速运算

每组数据的运算之间是完全独立的，考虑到现在 GPGPU 十分火热（和比特币关系比较大），我尝试了一下使用 GPU 来加速计算。

硬件情况：

CPU: Intel Core i5 4258U @ 2.4GHz

GPU: Intel Iris 5100

- 40 execution units, 704 GFLOPS
- 1GB memory shared with main memory

使用 OpenCL

OpenCL 为一个开放的通用计算标准，其中大致有 kernel、work item、work group 这样几个概念。

使用 OpenCL

OpenCL 为一个开放的通用计算标准，其中大致有 kernel、work item、work group 这样几个概念。

kernel 是一段程序代码，使用和一种和 C 语言几乎一样的语言编写，由编译器翻译成 GPU 可识别的汇编进行执行。kernel 就是我们需要计算的那段代码，它将在多个 work item 上同步执行。

使用 OpenCL

OpenCL 为一个开放的通用计算标准，其中大致有 kernel、work item、work group 这样几个概念。

kernel 是一段程序代码，使用和一种和 C 语言几乎一样的语言编写，由编译器翻译成 GPU 可识别的汇编进行执行。kernel 就是我们需要计算的那段代码，它将在多个 work item 上同步执行。每个 work group 由若干 work item 组成，这些 work item 会尽可能同步执行。

使用 OpenCL

OpenCL 为一个开放的通用计算标准，其中大致有 kernel、work item、work group 这样几个概念。

kernel 是一段程序代码，使用和一种和 C 语言几乎一样的语言编写，由编译器翻译成 GPU 可识别的汇编进行执行。kernel 就是我们需要计算的那段代码，它将在多个 work item 上同步执行。每个 work group 由若干 work item 组成，这些 work item 会尽可能同步执行。

GPU 的运算模式是单指令流多数据流 (SIMD)，简单来说就数据集合是一个个格子排成一行，每个运算单元选择一个格子的数据，所有的运算单元执行的运算指令是相同的。在这样的模型中，如果每个运算单元的运算量、运算逻辑分差几乎一样，效率将是最好的。

使用 OpenCL

经过实践后，对于 10^5 组询问的规模，使用 GPU 加速后的运行时间仅为 CPU 单线程运行的 $1/12$。

使用 OpenCL

经过实践后，对于 10^5 组询问的规模，使用 GPU 加速后的运行时间仅为 CPU 单线程运行的 1/12。

由于我只是简单学习了 OpenCL 相关用法和简单了解了 Intel Iris GPU 的架构，因此我坚信还能有加速的空间。并且我使用的硬件仅为 CPU 内集成的 GPU，如果使用桌面平台的中高端 GPU，理论上看还能有数倍的加速。

使用 OpenCL

经过实践后，对于 10^5 组询问的规模，使用 GPU 加速后的运行时间仅为 CPU 单线程运行的 1/12。

由于我只是简单学习了 OpenCL 相关用法和简单了解了 Intel Iris GPU 的架构，因此我坚信还能有加速的空间。并且我使用的硬件仅为 CPU 内集成的 GPU，如果使用桌面平台的中高端 GPU，理论上看还能有数倍的加速。

Intel Iris 5100: 704 GFLOPS

AMD Radeon R9 280X ($299): 4096 GFLOPS

AMD Radeon R9 290X ($549): 5632 GFLOPS

Nvidia GeForce GTX 770 ($399): 3213 GFLOPS

Nvidia GeForce GTX Titan ($999): 4500 GFLOPS