《月亮的背面是粉红色的》解题报告

焦作市第一中学　焦思源

2024 年 10 月 3 日

1 题目描述

小 L 终于见到了月球的背面，可这里一片荒芜，冷漠乏味。

他想要把这里染成热情的粉红色，为此他翻阅数学书找到了一个函数 $f_t(n) = 2^{\omega(n)}n^t$，他要根据这个函数决定染色的过程。

这里的 $\omega(n)$ 为 n 的不同质因子个数，例如 $\omega(1) = 0, \omega(2) = 1, \omega(8) = 1, \omega(6) = 2$。

小 L 先把这里划分成了 $n \times n$ 片区域，每个区域倒入不同数量的粉色颜料。具体来说，他会在第 i 行第 j 列的区域内倒入 $f_t(\gcd(i,j))f_t(\mathrm{lcm}(i,j))$ 桶颜料。

不过他已经没有精力去计算了，因此请你直接告诉他总共需要多少桶粉色颜料。

更进一步的，如果上面的答案记成 $F_t(n)$，小 L 会告诉你一个整数 $m \in \{0,1\}$：

- 如果 $m = 0$，请你输出 $F_0(n)$。
- 如果 $m = 1$，请你输出 $F_0(n), F_1(n)$。

由于答案可能很大，请输出答案对 $10^9 + 7$ 取模的值。

2 数据规模

- 子任务一 (3 分)：$1 \le n \le 5000, m \in \{0,1\}$。
- 子任务二 (3 分)：$1 \le n \le 10^7, m \in \{0,1\}$。
- 子任务三 (8 分)：$0 \le n \le 10^{10}, m = 0$。
- 子任务四 (8 分)：$0 \le n \le 10^{10}, m \in \{0,1\}$。
- 子任务五 (8 分)：$0 \le n \le 10^{12}, m \in \{0,1\}$。
- 子任务六 (10 分)：$0 \le n \le 10^{13}, m \in \{0,1\}$。
- 子任务七 (13 分)：$0 \le n \le 10^{14}, m = 0$。
- 子任务八 (14 分)：$0 \le n \le 10^{14}, m \in \{0,1\}$。
- 子任务九 (16 分)：$1 \le n \le 10^{16}, m = 0$。
- 子任务十 (17 分)：$1 \le n \le 10^{15}, m \in \{0,1\}$。

3 解题过程

3.1 算法一

注意到 $f_t(n)$ 是一个积性函数，那么显然有 $f_t(\gcd(i,j))f_t(\mathrm{lcm}(i,j)) = f_t(i)f_t(j)$，故答案就是

$$(\sum_{i=1}^{n} f_t(i))^2$$

设 $S_t(n)$ 为 $f_t(n)$ 的前缀和即 $\sum_{i=1}^{n} f_t(i)$，我们只需要求 S_t 在某个位置的值即可。

f_t 在 $p^k(k \geq 1)$ 处的值为 $f_t(p^k) = 2p^{tk}$，用线性筛即可得到所有的 $f_t(n)$ 的值。

复杂度 $\Theta(n)$，可以通过子任务一，二获得 6 分。

3.2 算法二

考虑一下 $t = 0$ 的特殊情况，此时 $f(p^k) = 2$。

3.2.1 性质：

$$\sum_{i|n} f(i)\mu(\frac{n}{i}) = \mu^2(i)$$

证明：因为对于积性函数 f, g，$n = p_1^{c_1}p_2^{c_2}\dots$，其狄利克雷卷积

$$(f*g)(n) = \sum_{i_1 \leq c_1}\sum_{i_2 \leq c_2}\cdots\sum_{i_k \leq c_k} f(p_1^{i_1}p_2^{i_2}\cdots p_k^{i_k})g(p_1^{c_1-i_1}p_2^{c_2-i_2}\dots) \tag{1}$$

$$= \sum_{i_1 \leq c_1} f(p_1^{i_1})g(p_1^{c_1-i_1}) \sum_{i_2 \leq c_2} f(p_2^{i_2})g(p_2^{c_2-i_2})\cdots\sum_{i_k \leq c_k} f(p_k^{i_k})g(p_k^{c_k-i_k}) \tag{2}$$

因此我们只需要验证 $f * \mu = \mu^2$ 在 p^k 处成立即可。

因为 $\mu(p^i)$ 只有在 $i \leq 1$ 是不为 0，所以卷积结果为 $f(p^k) - f(p^{k-1}) = [k \leq 1] = \mu^2(p^k)$，成立。

3.2.2 杜教筛

由于具有上面的性质，我们可以使用杜教筛求解 f 的前缀和，具体的：

$$\sum_{i=1}^{n}\mu^2(i) = \sum_{i=1}^{n}\sum_{j|i}\mu(j)f(\frac{i}{j}) \tag{3}$$

$$= \sum_{i=1}^{n}\mu(i)S(\lfloor\frac{n}{i}\rfloor) \tag{4}$$

$$S(n) = \sum_{i=1}^{n}\mu^2(i) - \sum_{i=2}^{n}\mu(i)S(\lfloor\frac{n}{i}\rfloor) \tag{5}$$

预处理 $n \leq B$ 的 $S(n)$,并使用整除分块优化递归过程,记忆化之后的复杂度可以视为 $\sum_{1}^{\lfloor\frac{n}{B}\rfloor}\sqrt{\frac{n}{i}} = \Theta(\frac{n}{\sqrt{B}})$，取 $B = n^{\frac{2}{3}}$ 即可做到 $\Theta(n^{\frac{2}{3}})$ 的复杂度。

对于 $\mu(n)$ 的前缀和，类似的可以证明下面的式子：

$$\sum_{i|n}\mu(i)=[n=1]$$

之后得到递归式：

$$S_\mu(n)=1-\sum_{i=2}^{n}S_\mu(\lfloor\frac{n}{i}\rfloor)$$

对于 $\mu^2(n)$ 的前缀和，其求的就是 $\le n$ 的，不含平方因子的数的个数，考虑容斥，钦定若干质因子的指数至少为 2，因为容斥系数恰好就是莫比乌斯函数，所以可以得到式子：

$$\sum_{i=1}^{n}\mu^2(i)=\sum_{i=1}^{\sqrt{n}}\mu(i)\lfloor\frac{n}{i^2}\rfloor$$

记忆化之后，三部分复杂度独立，复杂度为 $\Theta(n^{\frac{2}{3}})$。

结合算法一可以通过前三个子任务获得 14 分。

3.3 算法三

当 $t>0$ 时，我们考虑如下做法：

3.3.1 Min25 筛

记 p_k 为从小到大第 k 个质数，$mn(i)$ 为 i 的最小质因子。

设 $S(n,k)$ 为 $\sum\limits_{i=1}^{n}[mn(i)>p_k]f_t(i)$，我们要求出 $S(n,0)$。

$S(n,k)$ 的贡献分成质数和合数两部分，对于合数，我们枚举最小质因子 $p_j(j>k)$ 及其次数 c，显然这里 $p_j\le\sqrt{n}$。

对于质数的贡献，考虑求出所有质数的贡献的和减掉前 k 个质数的贡献，后者因为 $k\le\sqrt{n}$ 所以可以预处理，对于前者，考虑 $f_t(p)=2p^t$ 是一个单项式，因此设 $g(n)=n^t,G_k(n)$ 为用 $p_1\to p_k$ 做完线性筛后剩下的数的 $g(i)$ 之和，也就是 $G_k(n)=\sum_{j=1}^{k}g(p_j)+\sum_{i=p_k+1}^{n}[mn(i)>p_k]g(i)$。

我们让 k 从 0 依次增大，这样枚举完 $\le\sqrt{n}$ 的质数后就是要求的了。

$G_0(n)$ 就是一个自然数幂和，相比于 $G_{k-1}(n)$，$G_k(n)$ 筛掉了所有 $mn(i)=p_k$ 的 i，我们从 i 中提出一个 p_k，然后一定有 $mn(\frac{i}{p})>p_{k-1}$，这个东西就是 $G_{k-1}(n)$ 去掉前边质数的贡献，因此我们有：

$$G_k(n)=G_{k-1}(n)-g(n)(G_{k-1}(\lfloor\frac{n}{p_k}\rfloor)-sum_{k-1}))$$

其中 sum_k 是前 k 个质数的贡献。

这样就完成了递推。

完整的 $S(n,k)$ 的表达式就是

$$S(n,k)=G_*(n)-sum_k+\sum_{j>k}\sum_{c}f(p_j^c)(S(\lfloor\frac{n}{p_j^c}\rfloor,j)+[c\ne1])$$

该算法的复杂度被证明为 $\Theta(n^{1-\epsilon})$，这里不过多表述。

可以通过前四个子任务获得 22 分。

3.4 算法四

考虑 $2^{\omega(n)}$ 的实际意义：

选出两个整数 i,j，满足如下条件的方案数。

- $ij=n$
- $\gcd(i,j)=1$

那么可以推出下面的式子：

$$\sum_{i=1}^{n}2^{\omega(i)}i^t=\sum_{ij\le n}i^tj^t[\gcd(i,j)=1] \tag{6}$$

$$=\sum_{ij\le n}i^tj^t\sum_{u|i,u|j}\mu(u) \tag{7}$$

$$=\sum_{u=1}^{n}\mu(u)u^{2t}\sum_{i=1}^{\lfloor\frac{n}{u}\rfloor}\sum_{j=1}^{\lfloor\frac{n}{u}\rfloor}[iju^2\le n]i^tj^t \tag{8}$$

$$=\sum_{u=1}^{\sqrt{n}}\mu(u)u^{2t}\sum_{i=1}^{\lfloor\frac{n}{u^2}\rfloor}i^t\sum_{j=1}^{\lfloor\frac{n}{u^2i}\rfloor}j^t \tag{9}$$

$$\tag{10}$$

后面的 $\sum_{i=1}^{\lfloor\frac{n}{u^2}\rfloor}i^t\sum_{j=1}^{\lfloor\frac{n}{u^2i}\rfloor}j^t$ 可以用整除分块做到 $\Theta(\sqrt{\lfloor\frac{n}{u^2}\rfloor})$ 的复杂度，这样子总复杂度约为 $\sum_{u=1}^{\sqrt{n}}\sqrt{\lfloor\frac{n}{u^2}\rfloor}=\Theta(\sqrt{n}\ln n)$。

可以通过前六个子任务获得 40 分。

3.5 算法五

我们考虑优化上述做法中计算 $\sum_{i=1}^{\lfloor\frac{n}{u^2}\rfloor}i^t\sum_{j=1}^{\lfloor\frac{n}{u^2i}\rfloor}j^t$ 的部分，我们依旧从 $t=0$ 的情况入手。

3.5.1 双曲线下整点数量

以下记 $T=\lfloor\frac{n}{u^2}\rfloor$，那么等价于计算双曲线 $f(x)=\frac{T}{x}$ 下的整点个数。

由于双曲线的对称性，我们记 $B=\sqrt{T}$，并且计算出 $y\le B$ 的点的个数 C，那么答案就是 $2C-B^2$。

考虑用若干段一次函数构成的折线拟合双曲线，具体来说，我们维护处当前拟合到的位置 $P(x,y)$，每次选取一个向量 $(\Delta x,-\Delta y)$，满足 $Q(x+\Delta x,y-\Delta y)$ 不在双曲线内，且 $\frac{\Delta y}{\Delta x}$ 最大。

注意到我们需要进行一个对有理分数 $\frac{\Delta y}{\Delta x}$ 二分的过程，考虑直接在 Stern-Brocot Tree 上二分维护这个过程，具体的，维护两个分数 $L=(1,-1),R=(1,0)$，每次选取 $M=(L_x+R_x,L_y+R_y)$ 作为中间向量。

- 如果 $P+M$ 没有走进双曲线内，令 $R=M$。
- 如果 $P+M$ 走进了双曲线内，此时需要判断是否应该终止二分。

- 如果 $\frac{R_y}{R_x} \le f'(P_x + M_x)$，那么下一步的中间向量 $M' = (M_x + R_x, M_y + R_y)$ 一定也满足 $P + M'$ 在双曲线内，且因为 $M'_x > M_x$ 所以 $f'(P_x + M'_x) > f'(P_x + M_x)$ ，因此接下来的过程一定都满足该条件，故此时我们可以停止二分。
- 否则，我们令 $L = M$，继续二分。

同时，因为拟合的直线斜率显然单调递增，我们可以考虑维护一个单调栈，每次取栈顶作为当前的 L，并且把二分过程中所有 $R \leftarrow M$ 时的 M 都放入单调栈里。

可以证明这样子做的复杂度为 $\Theta(T^{\frac{1}{3}} \log T)$。

3.5.2 原问题

如果直接把算法四和上面的做法结合，复杂度约为 $\sum_{i=1}^{\sqrt{n}} (\frac{n}{i^2})^{\frac{1}{3}} \log n = \Theta(\sqrt{n} \log n)$。

考虑优化该做法，我们把 $\lfloor \frac{n}{i^2} \rfloor$ 相同的 i 一起计算。

取阈值 n^b, 并预处理 $\frac{n}{i^2} \le n^b$ 的答案, 剩下的部分照旧, 此时就得到了一个预处理复杂度 $\Theta(n^b)$, 计算答案复杂度 $\sum_{i=1}^{n^{\frac{1-b}{2}}} (\frac{n}{i^2})^{\frac{1}{3}} \log n = \Theta(n^{\frac{3-b}{6}} \log n)$。

取 $b = \frac{3}{7}$ 就可以得到一个 $\Theta(n^{\frac{3}{7}} \log n)$ 的做法，可以通过子任务九。

3.6 算法六

当 $t > 0$ 时，大致思路不变，唯一不同的是，当用直线拟合出来之后，我们需要快速计算一条直线下方的点 (i, j) 的 $i^t j^t$ 之和。

我们先通过简单的转化可以转化成经过 $(0, 0)$ 和 (A, B) 下方的点 (i, j) 的 $i^p j^q$ 之和，记为 $F(A, B, p, q)$ 。

记 $G(n, t) = \sum_{i=1}^{n} i^t = \sum_{i=1}^{t+1} g_{t,i} n^i$，该系数可以用伯努利数或者拉格朗日插值求解。

考虑使用类欧几里得算法，分成几种情况讨论：

3.6.1 $B \ge A$

记 $K = \lfloor \frac{B}{A} \rfloor$

此时的贡献分成整点三角形内的贡献和剩下的贡献

前者：

$$\sum_{i=1}^{A} \sum_{j=1}^{iK} i^p j^q = \sum_{i=1}^{A} i^p G(Ki, q) \tag{11}$$

$$= \sum_{i=1}^{A} i^p \sum_{j=1}^{q+1} g_{q,j} (Ki)^j \tag{12}$$

$$= \sum_{j=1}^{q+1} g_{q,j} K^j G(A, p + j) \tag{13}$$

$$\tag{14}$$

后者：

$$\sum_{i=1}^{A}\sum_{j=1}^{\lfloor\frac{Bi}{A}\rfloor-Ki} i^p(j+Ki)^q = \sum_{i=1}^{A}\sum_{j=1}^{\lfloor\frac{(B-KA)i}{A}\rfloor} i^p \sum_{k=0}^{q}\binom{q}{k}(Ki)^k j^{q-k} \tag{15}$$

$$= \sum_{k=0}^{q}\binom{q}{k}K^k \sum_{i=1}^{A}\sum_{j=1}^{\lfloor\frac{(B \bmod A)i}{A}\rfloor} i^{p+k}j^{q-k} \tag{16}$$

$$= \sum_{k=0}^{q}K^k\binom{q}{k}F(A, B \bmod A, p+k, q-k) \tag{17}$$

$$\tag{18}$$

3.6.2 B<A

此时考虑交换坐标系，用矩形内的点权和减掉直线上方的点权和，前者是容易计算的，后者根据对称性就是 $F(B, A, q, p)$。

整个递归过程与欧几里得算法类似，复杂度可以被证明为 $\Theta((p+q)^3 \log A)$。

因此，在上述算法的基础上就可以得到一个依旧是 $\Theta(T^{\frac{1}{3}} \log T)$ 的复杂度。

最终我们在 $\Theta(n^{\frac{3}{7}} \log n)$ 的复杂度解决了该问题，可以通过所有子任务获得 100 分。

4 参考资料

命题过程中有与集训队成员赵海鲲、周泽坤同学的讨论。