

浅谈算法竞赛中的数论

温昕岳

复旦大学白学研究会

2019.04.25

正整数的唯一分解定理

- 定理内容：每个大于1的自然数，要么本身就是质数，要么可以写为2个或以上的质数的积，而且这些质因子按大小排列之后，写法仅有一种方式。
- 举例： $6936 = 2^3 \times 3 \times 17^2$

单次质因数分解

- 对于任意大于1的整数 n ，至多有一个大于 \sqrt{n} 的质因子。
- 暴力试除 $2 \sim \sqrt{n}$ 的所有整数。若最终剩下的数不为1，则其也为 n 的一个质因子。

多次质因数分解

- 如果值域比较大，次数比较少：
- 沿用单次时的方法，试除 $2 \sim \sqrt{n}$ 中的所有**质数**。
- 质数定理： $1 \sim n$ 中的质数个数是 $O(\frac{n}{\log n})$ 的。
- 时间复杂度 $O(\text{预处理质数} + \text{询问次数} \times \frac{\sqrt{x}}{\log x})$
- 如何获得 \sqrt{n} 以内的所有质数？

暴力筛质数

- 枚举 $2 \sim n$ 中的每个整数，用它们去筛掉非本身的所有倍数。
- 时间复杂度： $O(n \log n)$
- 调和级数： $1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = O(\log n)$

埃拉托色尼筛法

- 从小到大枚举 $2 \sim n$ 中的每个**质数**，用它们去筛掉非本身的所有倍数。
- 时间复杂度： $O(n \log \log n)$

欧拉筛法

- 思路：令每个合数被最小的质因子筛去，且仅被筛去这一次。
- 实现方式：设合数 $x = i \times p$ ，其中 p 是 x 的最小质因子。先枚举从小到大枚举 i ，再从小到大枚举质数 p 。若 i 是 p 的倍数，则更大的质数 p_2 不可能是 $i \times p_2$ 的最小质因子，因为 $i \times p_2$ 含有质因子 p 。
- 时间复杂度： $O(n)$

```
for (int i=2;i<=n;i++)
{
    if (!vis[i])
        prime[cnt++]=i;
    for (int j=0;;j++)
    {
        int x=i*prime[j];
        if (x>n)
            break;
        vis[x]=true;
        if (i%prime[j]==0)
            break;
    }
}
```

多次质因数分解

- 值域比较小，次数比较多的情况：
- 考虑埃拉托色尼筛法和欧拉筛法，在用质数 x 筛去 y 时，记录 y 是被 x 筛掉的。回答 y 的质因数分解时，已知 x 是其一个质因子，然后递归回答子问题 y/x 。
- 时间复杂度 $O(\text{最大值} + \sum \text{质因子个数})$

值非常大时的质因数分解

- OI中一般使用Pollard's rho算法。
- 若假设其算法中使用的某个函数是随机的，则其期望时间复杂度为 $O(n^{\frac{1}{4}})$ 。严谨的时间复杂度分析现在还是open的。

乘法逆元

- 若 $xy \equiv 1 \pmod{P}$ ，则在模 P 意义下， x, y 互为对方的乘法逆元。
- 乘法逆元的性质：
- 模 P 意义下， x 有乘法逆元当且仅当 x 与 P 互质。
- 若 x 有逆元，则逆元唯一。

求逆元

- 费马小定理
- 若 p 是一个质数, 且 a, p 互质, 则 $a^{p-1} \equiv 1 \pmod p$
- 求 a 的逆元: $a \cdot a^{p-2} \equiv 1 \pmod p$
- 时间复杂度 $O(\log p)$

求逆元

- 扩展欧几里得算法
- $ax \equiv 1 \pmod P$ 等价于 $ax + Py = 1$
- 时间复杂度 $O(\log a)$, 常数比快速幂小
- 为什么扩展欧几里得算法的时间复杂度是 $O(\log a)$ 的?

线性求出 $1\sim n$ 的逆元

- 如何用 $O(n)$ 的时间求出 $1\sim n$ 中的每个数在模 P 意义下的逆元?
- 法1:
- $1\sim n$ 都与 P 互质: 求出 $n!$ 的逆元, $inv(n) = inv(n!) * (n - 1)!$
- $1\sim n$ 不一定与 P 互质: 跳过与 P 不互质的数。
- 法2: 欧拉筛, 可见数论函数前缀和一节
- 法3:
- $1\sim n$ 都与 P 互质: $inv(i) \equiv -\left\lfloor \frac{P}{i} \right\rfloor \times inv(P \% i) \bmod P$

dp题中的数论

- NWERC2015 Debugging
- 有一份包含一个bug的 n ($1 \leq n \leq 10^6$)行代码，运行一次到崩溃需要的时间为 r ($1 \leq r \leq 10^9$)。
- 你可以任意行添加printf语句来输出调试，即你知道是否在执行printf语句前就崩溃了。每设置一个printf语句需要花费 p ($1 \leq p \leq 10^9$)时间，但是运行时不额外消耗时间。
- 问在最坏情况下，最少需要多少时间可以定位BUG在哪一行。

解法

- 设 $f(n)$ 表示 n 行代码debug需要的最少时间。显然 f 是个不降的序列。
- 假如我们选择第一次运行时添加 x 行printf语句，则最少花费 $f\left(\left\lceil \frac{n}{x+1} \right\rceil\right) + r + x \times p$ 的时间。
- 枚举 x ，有dp转移式 $f(n) = \min_{1 \leq i < n} f\left(\left\lceil \frac{n}{i+1} \right\rceil\right) + r + i \times p$
- 显然当 $\left\lceil \frac{n}{i+1} \right\rceil$ 相等时，只需要考虑最小的 i
- $\left\lceil \frac{n}{i} \right\rceil$ 只有 $O(\sqrt{n})$ 种取值
- 记忆化搜索+暴力转移dp

- 因为 $\left\lfloor \frac{n}{ab} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{n}{a} \right\rfloor}{b} \right\rfloor$, 所以记忆化搜索时遍历到的所有值都等于某个 $\left\lfloor \frac{n}{i} \right\rfloor$
- 之前提到了 $\left\lfloor \frac{n}{i} \right\rfloor$ 只有 $O(\sqrt{n})$ 种取值, 那么就意味着我们遍历到了 $O(\sqrt{n})$ 种值, 每种值我们又用了 $O(\sqrt{n})$ 的时间去枚举转移, 总时间复杂度是 $O(n)$ 的。
- 可以有更好的分析, 时间复杂度为 $O\left(\sum_{i=1}^{\sqrt{n}} \sqrt{i} + \sqrt{\left\lfloor \frac{n}{i} \right\rfloor}\right) = O\left(\int_1^{\sqrt{n}} \sqrt{\frac{n}{i}} \, di\right) = O(n^{\frac{3}{4}})$ 。

一些定义和性质

- $\left\lceil \frac{n}{ab} \right\rceil = \left\lceil \frac{\left\lceil \frac{n}{a} \right\rceil}{b} \right\rceil$, $\left\lfloor \frac{n}{ab} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{n}{a} \right\rfloor}{b} \right\rfloor$
- $\left\lfloor \frac{n}{i} \right\rfloor$ 只有 $O(\sqrt{n})$ 种取值
- **数论函数**: 定义域为正整数, 陪域为复数的函数。今天我们主要研究定义域为正整数, 值域为整数的函数。
- **积性函数**: 满足“若 a, b 互质, 则 $f(ab) = f(a)f(b)$ ”的数论函数称为积性函数。
- **完全积性函数**: 满足“ $f(ab) = f(a)f(b)$ ”的数论函数称为完全积性函数。
- **狄利克雷卷积**: 设 f, g 为两个数论函数, 则满足 $h(n) = \sum_{d|n} f(d)g\left(\frac{n}{d}\right)$ 的函数称为 f 与 g 的狄利克雷卷积。也可以理解为 $h(n) = \sum_{ij=n} f(i)g(j)$ 。
- 两个积性函数的狄利克雷卷积仍为积性函数。

一些定义和性质

- 狄利克雷卷积满足交换律和结合律
- 考虑使用 $h(n) = \sum_{ij=n} f(i)g(j)$ 这个定义式，那么多个函数的狄利克雷卷积实际上就是 $h(n) = \sum_{i_1 i_2 \cdots i_k = n} f_1(i_1) f_2(i_2) \cdots f_k(i_k)$ ，由乘法的交换律和结合律可以直接得到狄利克雷卷积的交换律和结合律。

一些常见的积性函数

- 单位函数 $e(x) = \begin{cases} 1, & x = 1 \\ 0, & x > 1 \end{cases}$
- 常函数 $I(x) = 1$
- 幂函数 $id(x) = x^k$
- 欧拉函数 $\varphi(x) = 1 \sim x$ 中与 x 互质的数的个数 $= x \prod_{p|x, p \text{ is a prime}} 1 - \frac{1}{p}$
- 莫比乌斯函数 $\mu(x) = \begin{cases} 1, & x = 1 \\ (-1)^k, & x = p_1 p_2 \cdots p_k \\ 0, & x = \text{others} \end{cases}$

积性函数性质例题1

- 定义 $f(n)$ =选两个 $[0,n)$ 的整数 a,b , 且 ab 不是 n 的倍数的方案数。
(2015 ICPC 长春 B)
- 求 $g(n) = \sum_{d|n} f(d)$ 。
- 数据组数 $1 \leq T \leq 20000$, $1 \leq n \leq 10^9$ 。

解法

- $f(n) = n^2 - \sum_{d|n} \varphi\left(\frac{n}{d}\right) * d$
- 设 $h(n) = \sum_{d|n} \varphi\left(\frac{n}{d}\right) * d$, 则 $f(n) = n^2 - h(n)$ 。
- $g(n) = \sum_{d|n} f(d) = \sum_{d|n} d^2 - \sum_{d|n} h(d)$ 。
- 设 $P(n) = \sum_{d|n} d^2$, $Q(n) = \sum_{d|n} h(d)$ 。
- 根据积性函数的性质, 我们只需要计算 $P(p^k)$ 和 $Q(p^k)$, 乘起来就可以得到 $P(n)$ 和 $Q(n)$ 。而这是很容易计算的, 因为 p^k 的因数只有 p^0, p^1, \dots, p^k 。
- 只剩下了质因数分解的时间复杂度。

一个相对简化的解法

- 设 $id(n) = n$, 则有 $I * \varphi = id$, 即 $\sum_{d|n} \varphi(d) = n$
- 证明: $1 \leq i \leq n$ 且 $\gcd(n, i) = d$ 的 i 有 $\varphi(\frac{n}{d})$ 个, 枚举 $1 \sim n$ 中每个数与 n 的最大公因数, 则有 $\sum_{d|n} \varphi(\frac{n}{d}) = \sum_{d|n} \varphi(d) = n$
- $Q(n) = \sum_{d|n} h(d) = \sum_{d|n} \sum_{w|d} \varphi(w) \times \frac{d}{w} = I * \varphi * id = (I * \varphi) * id = id * id = \sum_{d|n} d \times \frac{n}{d} = n \times n$ 的因数个数

积性函数性质例题2

- 定义 $f_0(n) =$ 满足 $pq = n$ 且 $\gcd(p, q) = 1$ 的二元组 (p, q) 个数
- 定义 $f_{r+1}(n) = \sum_{uv=n} \frac{f_r(u) + f_r(v)}{2}$
- 若干组询问，每次给出 r, n ，询问 $f_r(n)$ 的值，对 $10^9 + 7$ 取模。
- 询问次数 $1 \leq q \leq 10^6$
- $0 \leq r \leq 10^6, 1 \leq n \leq 10^6$
- CF 757 E

解法

- 定义 $\omega(n) = n$ 的不同质因子个数，则 $f_0(n) = 2^{\omega(n)}$ ，由积性函数的定义可验证 f_0 为积性函数。
- 若 f_r 为积性函数，则由 $f_{r+1}(n) = \sum_{d|n} f_r(d)$ 得 f_{r+1} 也为积性函数。
- 由积性函数性质，我们只需要求 $f_r(p^k)$ ，由于 $\forall p, f_0(p) = 2$ ，所以对于固定的 k, r ， $\forall p, f_r(p^k)$ 相等。
- 又注意到 k 是 $O(\log n)$ 的，前缀和优化求 $f_{r+1}(p^k) = \sum_{i=0}^k f_r(p^i)$ ，使用 $O(r \log n)$ 的时间预处理出所有可能的 $f_r(p^k)$ 的询问。
- 每组询问只剩下了质因数分解的复杂度。

数论函数的前缀和

- 给出一个积性函数 f ，求 $\sum_{i=1}^n f(i)$ 。
- 最简单的想法是分别求出来 $f(1), f(2), \dots, f(n)$ ，然后求和。显然这种思路的时间复杂度不可能低于 $O(n)$ 。
- 如果 $f(p)$ 可以在 $O(\log n)$ 的时间内求出来，求出质数项的总时间是 $O(n)$ 的；通常， $f(p^k)$ 可以比较容易的由 $f(p^{k-1})$ 等值递推出来，其他项可以直接由积性函数的性质由 $f(x) = f(d) * f(\frac{x}{d})$ 得到。因此，很多积性函数都可以在欧拉筛的过程中顺便递推出前 n 项的值，时间复杂度为 $O(n)$ 。

低于线性时间对数论函数求和

- 2018 四川省赛GRISAIA
- 求 $\sum_{i=1}^n \sum_{j=1}^i n \% (i \times j)$, $1 \leq n \leq 10^{11}$

解法1

- $\sum_{i=1}^n \sum_{j=1}^i n \% (i \times j) = \sum_{i=1}^n \sum_{j=1}^i n - \lfloor \frac{n}{ij} \rfloor ij$, 现在只需要求 $\sum_{i=1}^n \sum_{j=1}^i \lfloor \frac{n}{ij} \rfloor ij$ 。
- $\sum_{i=1}^n \sum_{j=1}^i \lfloor \frac{n}{ij} \rfloor ij = \frac{\sum_{i=1}^n \sum_{j=1}^n \lfloor \frac{n}{ij} \rfloor ij + \sum_{i=1}^n \lfloor \frac{n}{i^2} \rfloor i^2}{2}$, 重点是如何计算 $\sum_{i=1}^n \sum_{j=1}^n \lfloor \frac{n}{ij} \rfloor ij$ (以后简记为 ans)。
- 定义 $f(n) = \sum_{i=1}^n \lfloor \frac{n}{i} \rfloor i$, 则 $ans = \sum_{i=1}^n \sum_{j=1}^n i \times \left\lfloor \frac{\lfloor \frac{n}{i} \rfloor}{j} \right\rfloor \times j = \sum_{i=1}^n i \sum_{j=1}^{\lfloor \frac{\lfloor \frac{n}{i} \rfloor}{i} } \left\lfloor \frac{\lfloor \frac{n}{i} \rfloor}{j} \right\rfloor \times j = \sum_{i=1}^n i \times f(\left\lfloor \frac{n}{i} \right\rfloor)$

解法1

- 定义 $f(n) = \sum_{i=1}^n \lfloor \frac{n}{i} \rfloor i$, 则 $ans = \sum_{i=1}^n \sum_{j=1}^n i \times \left\lfloor \frac{\lfloor \frac{n}{i} \rfloor}{j} \right\rfloor \times j = \sum_{i=1}^n i (\sum_{j=1}^{\lfloor \frac{n}{i} \rfloor} \left\lfloor \frac{\lfloor \frac{n}{i} \rfloor}{j} \right\rfloor \times j) = \sum_{i=1}^n i \times f(\lfloor \frac{n}{i} \rfloor)$ 。
- $O(\sqrt{n})$ 地枚举 $\lfloor \frac{n}{i} \rfloor$ 的值, 然后 $O(\sqrt{\lfloor \frac{n}{i} \rfloor})$ 地计算 $f(\lfloor \frac{n}{i} \rfloor)$, 时间复杂度为 $O\left(\sum_{i=1}^{\sqrt{n}} \sqrt{\lfloor \frac{n}{i} \rfloor}\right) = O(n^{\frac{3}{4}})$ 。
- $O(1)$ 读写 f 的技巧: 使用两个 \sqrt{n} 大小的数组, 一个存储 $f(1) \sim f(\sqrt{n})$, 一个存储 $f(\frac{n}{1}) \sim f(\frac{n}{\sqrt{n}})$ 。

解法2

- 考虑优化解法1, 定义 $g(n) = f(n) - f(n-1)$
- $g(n) = \sum_{i=1}^n i \times \left(\left\lfloor \frac{n}{i} \right\rfloor - \left\lfloor \frac{n-1}{i} \right\rfloor \right) = \sum_{i|n} i$
- 使用欧拉筛求出 g 的前 $n^{\frac{2}{3}}$ 项, 再求一下前缀和, 就可以得到 f 的前 $n^{\frac{2}{3}}$ 项, 这部分的时间复杂度是 $O(n^{\frac{2}{3}})$ 。
- 现在只需要在 $\left\lfloor \frac{n}{i} \right\rfloor > n^{\frac{2}{3}}$ 时暴力计算 $f\left(\left\lfloor \frac{n}{i} \right\rfloor\right)$, 时间复杂度为 $O\left(\sum_{i=1}^{n^{\frac{1}{3}}} \sqrt{\left\lfloor \frac{n}{i} \right\rfloor}\right) = O(n^{\frac{2}{3}})$ 。

莫比乌斯反演

- $\forall n, f(n) = \sum_{d|n} g(d)$ 与 $\forall n, g(n) = \sum_{d|n} \mu(d) f(\frac{n}{d})$ 等价。
- 即 $f = g * I \Leftrightarrow g = \mu * f$ 。
- 证明：先证明 $\mu * I = e$
- $\forall n > 1, \sum_{d|n} \mu(d) = \sum_{i=0}^k C(k, i) \times (-1)^i = (1 - 1)^k = 0$
- 证明 $f = g * I \Rightarrow g = \mu * f$
- 利用狄利克雷卷积的交换律和结合律，由 $f = g * I$ 得 $\mu * f = g * (\mu * I) = g * e = g$
- $g = \mu * f \Rightarrow f = g * I$ 也可以类似地证明。

“莫比乌斯反演”例题

- 求 $\sum_{i=1}^n \sum_{j=1}^m [\gcd(i, j) = \text{质数}]$, $T \leq 10^4, n, m \leq 10^7$ (BZOJ2820)

- 解法:

- $ans = \sum_p \sum_{i=1}^{\lfloor \frac{n}{p} \rfloor} \sum_{j=1}^{\lfloor \frac{m}{p} \rfloor} [\gcd(i, j) = 1]$

- 容斥原理:

$$|\bigcup_{i=1}^n A_i| = \sum_{i=1}^n (-1)^{i-1} \sum_{1 \leq j_1 < j_2 < \dots < j_i \leq n} |A_{j_1} \cap A_{j_2} \cap \dots \cap A_{j_i}|$$

- $ans = \sum_p \sum_{g=1}^{\lfloor \frac{\min(n, m)}{p} \rfloor} \mu(g) \left\lfloor \frac{n}{pg} \right\rfloor \left\lfloor \frac{m}{pg} \right\rfloor = \sum_{i=1}^{\min(n, m)} \left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor \sum_{p|i} \mu\left(\frac{i}{p}\right)$

“莫比乌斯反演”例题

- 定义 $f(i) = \sum_{p|i} \mu\left(\frac{i}{p}\right)$, 则 $ans = \sum_{i=1}^{\min(n,m)} \left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor f(i)$, 预处理出 f 就可以 $O(\sqrt{n} + \sqrt{m})$ 地回答每组询问。
- 求 f 的一个方式是按照定义暴力求和, 复杂度 $O(n \log \log n)$
- 观察之后可以得到 f 的一个递推式, 设 x 为 n 的最小质因子, 若 $x^2 | n$, 则 $f(n) = \mu\left(\frac{n}{x}\right)$, 否则 $f(n) = \mu\left(\frac{n}{x}\right) + \mu(x)f\left(\frac{n}{x}\right) = \mu\left(\frac{n}{x}\right) - f\left(\frac{n}{x}\right)$, 使用欧拉筛可以做到 $O(n)$ 。

杜教筛

- 设 $S(n) = \sum_{i=1}^n f(i)$, 下面我们将求 $S(n)$ 的过程一般化
- \forall 数论函数 g , 设 $h = f * g$, 有 $\sum_{i=1}^n h(i) = \sum_{i=1}^n g(i) S(\lfloor \frac{n}{i} \rfloor)$
- 移项, 得 $g(1)S(n) = \sum_{i=1}^n h(i) - \sum_{i=2}^n g(i) S(\lfloor \frac{n}{i} \rfloor)$
- 如果我们可以 $O(\sqrt{n})$ 计算 $\sum_{i=1}^n h(i)$, $O(1)$ 计算 g 的前缀和, 就可以快速把问题递归为同类子问题, 时间复杂度为 $O\left(\sum_{i=1}^{\sqrt{n}} \sqrt{\lfloor \frac{n}{i} \rfloor}\right) = O(n^{\frac{3}{4}})$ 。
- 如果 f 有一些比较好的性质 (如是积性函数), 我们可以用欧拉筛求出前 $n^{\frac{2}{3}}$ 项, 更后面的项再递归, 时间复杂度为 $O(n^{\frac{2}{3}})$ 。

杜教筛例题1

- 求 $\sum_{i=1}^n \varphi(i)$, $1 \leq n \leq 10^{11}$, 答案取模
- 设 $S(n) = \sum_{i=1}^n \varphi(i)$, $g(n) = 1$, $h = \varphi * g$
- $g(1)S(n) = \sum_{i=1}^n h(i) - \sum_{2 \leq d \leq n} g(d)S\left(\left\lfloor \frac{n}{d} \right\rfloor\right)$
- 代入 g 的值, 有 $h(n) = \sum_{d|n} \varphi(d) = n$
- 代入 g, h 的值, 得 $S(n) = \sum_{i=1}^n i - \sum_{2 \leq d \leq n} S\left(\left\lfloor \frac{n}{d} \right\rfloor\right)$
- 用欧拉筛求出欧拉函数的前 $n^{\frac{2}{3}}$ 项, 求前缀和得到 S 的前 $n^{\frac{2}{3}}$ 项。 $\left\lfloor \frac{n}{i} \right\rfloor$ 较大时递归计算, 总时间复杂度 $O(n^{\frac{2}{3}})$ 。

杜教筛例题2

- 已知函数 f 满足 $\sum_{d|n} f(d) = n^2 - 3n + 2$, 求 $\sum_{i=1}^n f(i)$, 答案取模。
 $T \leq 500, n \leq 10^9$, 只有5组数据中 $n > 10^6$ (HDOJ 5608)
- $f(n) = n^2 - 3n + 2 - \sum_{d|n \text{ and } d \neq n} f(d)$, 从小到大递推 f 的前 $n^{\frac{2}{3}}$ 项, 时间复杂度 $O(n^{\frac{2}{3}} \log n)$ 。
- 题设条件实际是直接告诉了 f 与 $g(n) = n$ 的狄利克雷卷积, 因此剩下的部分直接套用杜教筛即可。唯一的问题可能出在如何快速计算 $\sum_{i=1}^n i^2 - 3i + 2$, 用公式分别求 $\sum_{i=1}^n i^2$, $\sum_{i=1}^n 3i$, $\sum_{i=1}^n 2$, 即可 $O(1)$ 计算。

Tips

- 杜教筛用到的等式对所有数论函数都成立，只是通常当 f 为积性函数时，我们才能方便地用欧拉筛求值，从而实现 $O(n^{\frac{2}{3}})$ 的时间复杂度。但正如例题2，有时大于 $O(n^{\frac{2}{3}})$ 的时间复杂度我们仍然可以接受，因此杜教筛仍然可以使用。
- 杜教筛实际上求出来了所有的 $S\left(\left\lfloor \frac{n}{i} \right\rfloor\right)$ ，我们需要的 g 的前缀和也正是 $S_g\left(\left\lfloor \frac{n}{i} \right\rfloor\right)$ 这些项，因此有需要时可以先对 g 使用一次杜教筛以便于快速计算 g 的前缀和。
- 不是所有的函数 f （即使它是积性的）都能找到合适的函数 g 满足 g 与 $f * g$ 的前缀和都能很快地计算，这种情况下杜教筛的时间复杂度可能非常大，不适合使用杜教筛。

rng_58-clj等式

- $\sum_{i=1}^a \sum_{j=1}^b \sum_{k=1}^c d(ijk) = \sum_{\gcd(i,j)=\gcd(j,k)=\gcd(k,i)=1} \left\lfloor \frac{a}{i} \right\rfloor \left\lfloor \frac{b}{j} \right\rfloor \left\lfloor \frac{c}{k} \right\rfloor$
- 证明：设 $f(a, b, c) = \sum_{i=1}^a \sum_{j=1}^b \sum_{k=1}^c d(ijk)$, $g(a, b, c) = \sum_{\gcd(i,j)=\gcd(j,k)=\gcd(k,i)=1} \left\lfloor \frac{a}{i} \right\rfloor \left\lfloor \frac{b}{j} \right\rfloor \left\lfloor \frac{c}{k} \right\rfloor$
- 由数学归纳法，只需要证 $d(abc) = f(a, b, c) - f(a-1, b, c) - f(a, b-1, c) - f(a, b, c-1) + f(a-1, b-1, c) + f(a-1, b, c-1) + f(a, b-1, c-1) - f(a-1, b-1, c-1) = g(a, b, c) - g(a-1, b, c) - g(a, b-1, c) - g(a, b, c-1) + g(a-1, b-1, c) + g(a-1, b, c-1) + g(a, b-1, c-1) - g(a-1, b-1, c-1) = \sum_{\gcd(i,j)=\gcd(j,k)=\gcd(k,i)=1} [i|a][j|b][k|c]$

rng_58-clj等式

- $d(abc) = \sum_{\gcd(i,j)=\gcd(j,k)=\gcd(k,i)=1} [i|a][j|b][k|c]$
- 设 x_p, y_p, z_p 分别为质数 p 在 a, b, c 中的幂次
- 则 $d(abc) = \prod_p (x_p + y_p + z_p + 1) = \sum_{\gcd(i,j)=\gcd(j,k)=\gcd(k,i)=1} [i|a][j|b][k|c]$
- 由证明方法可知这个等式可扩展至任意维

rng_58-clj等式例题1

- 求 $\sum_{i=1}^n \sum_{j=1}^m d(ij)$ 。 $T, n, m \leq 50000$ 。 (SDOI2015 约数个数和)
- $\sum_{i=1}^n \sum_{j=1}^m d(ij) = \sum_{\gcd(i,j)=1} \left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{j} \right\rfloor = \sum_{g=1}^{\min(n,m)} \mu(g) \sum_{i=1}^n \sum_{j=1}^m \left\lfloor \frac{n}{ig} \right\rfloor \left\lfloor \frac{m}{jg} \right\rfloor$
- 定义 $f(n) = \sum_{i=1}^n \left\lfloor \frac{n}{i} \right\rfloor$
- $\sum_{i=1}^n \sum_{j=1}^m d(ij) = \sum_{g=1}^{\min(n,m)} \mu(g) f\left(\left\lfloor \frac{n}{g} \right\rfloor\right) f\left(\left\lfloor \frac{m}{g} \right\rfloor\right)$
- 预处理时间复杂度 $O(n)$ ， 每组时间复杂度 $O(\sqrt{n} + \sqrt{m})$ 。

rng_58-clj等式例题2

- 求 $\sum_{i=1}^n \sum_{j=1}^n d(ij)$ 。 $n \leq 10^9$ 。 (BZOJ4176)
- 类似地推出 $\sum_{i=1}^n \sum_{j=1}^n d(ij) = \sum_{g=1}^n \mu(g) \sum_{i=1}^n \sum_{j=1}^n \left\lfloor \frac{n}{ig} \right\rfloor \left\lfloor \frac{n}{jg} \right\rfloor = \sum_{g=1}^n \mu(g) f^2 \left(\left\lfloor \frac{n}{g} \right\rfloor \right)$
- 注意到我们需要求的 μ 的前缀和总是 $\left\lfloor \frac{n}{g} \right\rfloor$ 变化处，正好是杜教筛求 μ 的前 n 项和时求出来的值。
- 时间复杂度 $O(n^{\frac{2}{3}})$

数论在计数问题中的应用

- 现在有一个长度为 n 的整数序列 A ，定义 $f(m)$ = 有多少个长度为 n 的整数序列 C 满足以下条件：
 - 1. 若 $A_i = -1$ ，则 $C_i \in [0, m)$ ；否则 $C_i = A_i \% m$ 。
 - 2. $\sum_{i=1}^n C_i x_i \equiv 1 \pmod{m}$ 有整数解。
- 求 $\sum_{i=1}^m f(i)$ ，答案对 $10^9 + 7$ 取模。（HDOJ 6439）
- $1 \leq n \leq 50, 1 \leq A_i, m \leq 10^9$

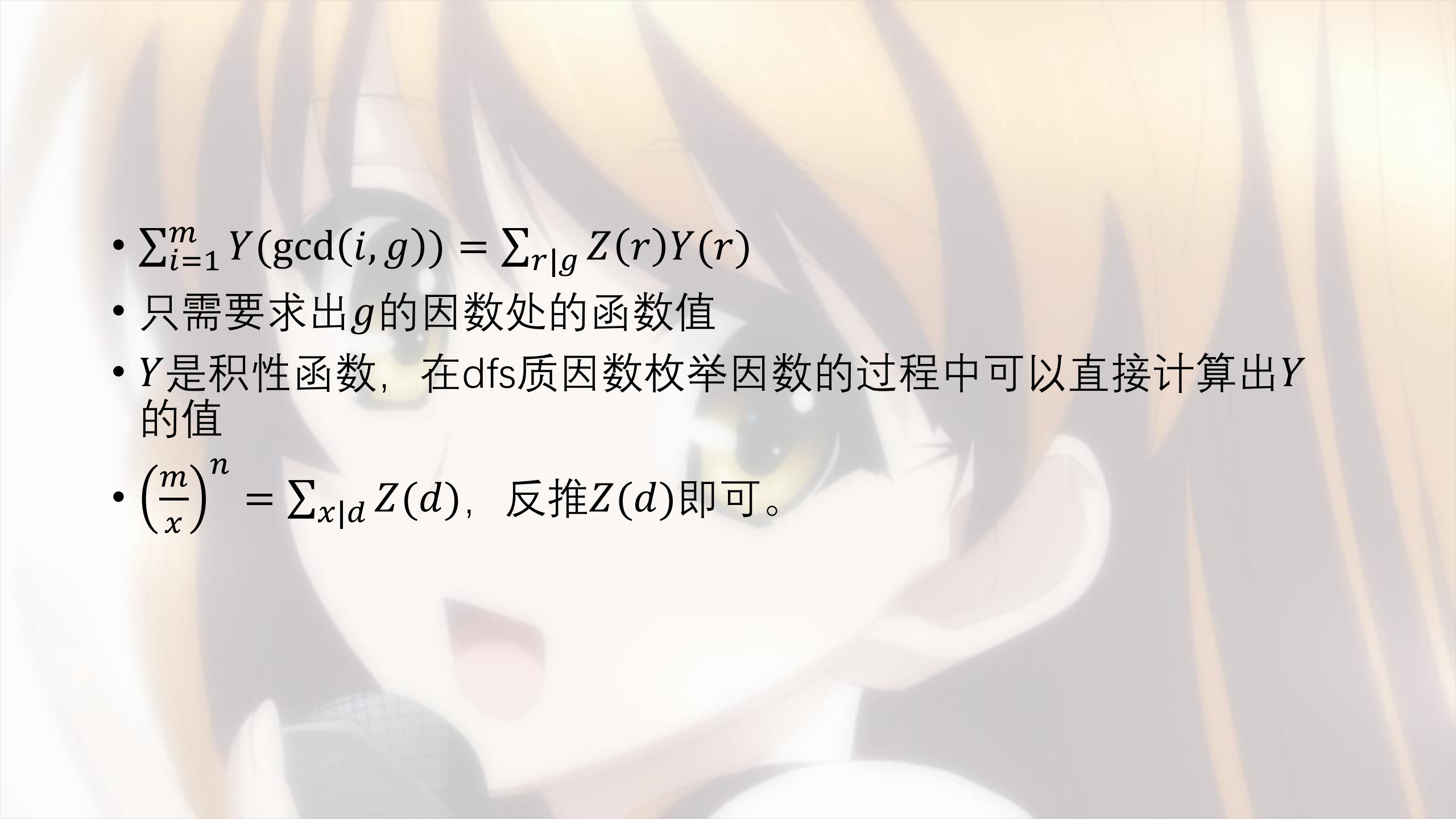
解法

- 条件2的等价条为 $\gcd(c_1, c_2, \dots, c_n, m) = 1$ （裴蜀定理）。
- 如果 A_i 全部是-1:
- $f(m) = \sum_{d|m} \mu(d) \left(\frac{m}{d}\right)^n = \mu * id^n$
- 设 $h(m) = f * id = \mu * id^n * id = (\mu * id) * id^n = id^n$
- 则 $id(1)S(m) = \sum_{i=1}^m h(i) - \sum_{i=2}^m id(i) S\left(\left\lfloor \frac{n}{i} \right\rfloor\right)$
- 即 $S(m) = \sum_{i=1}^m i^n - \sum_{i=2}^m i S\left(\left\lfloor \frac{n}{i} \right\rfloor\right)$
- 若求 h 的前缀和的时间为 T ，则总时间复杂度为 $O(m^{\frac{2}{3}} + T\sqrt{m})$ 。

如何求 h 的前缀和

- k 次幂的前缀和有很多种求法，大部分常用做法可以在 https://blog.csdn.net/infinity_edge/article/details/79464454 这篇博文找到。这篇博文唯一缺少的常见做法是伯努利数，可以另行搜索一下。

- 如果 A_i 不全为-1:
- $\gcd(C_{p_0}, C_{p_1}, \dots, C_{p_k}, m) = \gcd(A_{p_0} \% m, A_{p_1} \% m, \dots, A_{p_k} \% m, m) = \gcd(\gcd(A_{p_0} \% m, m), \gcd(A_{p_1} \% m, m), \dots, \gcd(A_{p_k} \% m, m)) = \gcd(\gcd(A_{p_0}, m), \gcd(A_{p_1}, m), \dots, \gcd(A_{p_k}, m)) = \gcd(A_{p_0}, A_{p_1}, \dots, A_{p_k}, m)$ 。令 $g = \gcd(A_{p_0}, A_{p_1}, \dots, A_{p_k})$ 。
- 令 n 表示值为-1的 A_i 的个数, $Y(x) = \sum_{d|x} \mu(d) \left(\frac{m}{d}\right)^n$, $Z(i)$ 表示满足 $\gcd(x, g) = i$ 且 $x \in [1, m]$ 的 x 的个数,
- $\sum_{i=1}^m Y(\gcd(i, g)) = \sum_{r|g} Z(r)Y(r)$

- 
- $\sum_{i=1}^m Y(\gcd(i, g)) = \sum_{r|g} Z(r)Y(r)$
 - 只需要求出 g 的因数处的函数值
 - Y 是积性函数，在dfs质因数枚举因数的过程中可以直接计算出 Y 的值
 - $\left(\frac{m}{x}\right)^n = \sum_{x|d} Z(d)$ ，反推 $Z(d)$ 即可。