

莫比乌斯反演

积性函数

数论函数是一类定义域在正整数上的函数。若对数论函数 f 且 $\forall a, b$ 使得 $(a, b) = 1$ 都满足

$$f(ab) = f(a) \cdot f(b)$$

则称 f 是积性函数。如果条件弱一些，不需要 $(a, b) = 1$ 也有上式成立，则称 f 是完全积性函数。只要 f 是一个积性函数，同时能够快速求出 f 在质数 p 幂次上的取值 $f(p^a)$ 那么就可以用线筛求 f

性质

若 f, g 都是积性函数，则以下函数也是积性函数：

$$h(n) = f(n^p) \quad h(n) = f^p(n) \quad h(n) = f(n)g(n) \quad h(n) = \sum_{d \mid n} f(n) g\left(\frac{n}{d}\right)$$

常见积性函数

定义艾佛森括号

$$[P] = \begin{cases} 1, & P \text{ is true} \\ 0, & P \text{ is false} \end{cases}$$

- 单位函数 $\varepsilon(n) = [n = 1]$
- 恒等函数 $\mathrm{id}(n) = n$ 下文中一般会直接用 n 代替。
- 常数函数 $1(n) = 1$ 下文中一般会直接用 1 代替。
- 欧拉函数 $\varphi(n) = \sum_{1 \leq i \leq n} [\gcd(i, n) = 1]$
- 约数函数 $d(n) = \sum_{d \mid n} 1$
- 约数和函数 $\sigma(n) = \sum_{d \mid n} d$

狄利克雷卷积

对数论函数 f, g 定义它们的狄利克雷卷积

$$(f * g)(n) = \sum_{d \mid n} f(n) \cdot g\left(\frac{n}{d}\right)$$

性质

狄利克雷卷积满足：

- 交换律
- 结合律
- 分配律

- 单位元

同时，两个积性函数的狄利克雷卷积还是积性函数。

常用狄利克雷卷积关系

一个常用的卷积式子 $\sum_{d|n} \varphi(d) = n$ 简单证明如下：

首先枚举约数，每个约数求出小于他且与他互质的个数，即求这个约数为分母的真分数个数，它们的和必为 n 例如 $n = 12$ 时的真分数有

$$\frac{1}{12}, \frac{2}{12}, \frac{3}{12}, \frac{4}{12}, \frac{5}{12}, \frac{6}{12}, \frac{7}{12}, \frac{8}{12}, \frac{9}{12}, \frac{10}{12}, \frac{11}{12}, \frac{12}{12}$$

可以化简为

- $\frac{1}{12}, \frac{5}{12}, \frac{7}{12}, \frac{11}{12}$ $\varphi(12) = 4$
- $\frac{1}{6}, \frac{5}{6}$ $\varphi(6) = 2$
- $\frac{1}{4}, \frac{3}{4}$ $\varphi(4) = 2$
- $\frac{1}{3}, \frac{2}{3}$ $\varphi(3) = 2$
- $\frac{12}{12}$ $\varphi(2) = 1$
- $\frac{11}{12}$ $\varphi(1) = 1$

整数格上的莫比乌斯反演

莫比乌斯函数

莫比乌斯反演是偏序集上的一个反演，不过在此处我们只讨论整数格上的莫比乌斯反演。

定义莫比乌斯函数 $\mu(n)$

$$\mu(n) = \begin{cases} 1, & n = 1 \\ (-1)^m, & n = \prod_{i=1}^m p_i \\ 0, & \text{if } n \text{ is not square-free} \end{cases}$$

$\mu(n)$ 有两个性质：

- μ 是积性函数
- $\sum_{d|n} \mu(d) = [n = 1]$

第一条性质说明 $\mu(n)$ 可以线性筛；第二条性质提供了我们一个当且仅当 $n = 1$ 时计数的函数，因此在遇到对 $\gcd(i, j) = 1$ 的计数问题中通常会用到它。

直接给出代码。

```
void InitMu() {
    mu[1] = 1;
    for (int i = 2; i < N; i++) {
        if (!notPri[i]) pri[siz++] = i, mu[i] = -1;
        for (int j = 0; j < siz && i * pri[j] < N; j++) {
            int nxt = i * pri[j];
            if (i % pri[j] == 0) mu[nxt] = 0;
            else mu[nxt] = -mu[i];
        }
    }
}
```

```

        notPri[nxt] = 1;
        if (i % pri[j])
            mu[nxt] = -mu[i];
        else {
            mu[nxt] = 0;
            break;
        }
    }
}
}
}

```

当出现平方因子就退出筛法保证了每个数只会被最小的因子筛去，因此时间复杂度线性。\$\mu(i) = 0\$ 的情况是由最小因子筛掉的，而其他情况都是由 \$\mu(i) = -\mu(j)\$ 得到的。

莫比乌斯反演

若函数 \$f(n)\$ 与 \$g(n)\$ 为数论函数，则

$$f(n) = \sum_{d \mid n} g(d) \Leftrightarrow g(n) = \sum_{d \mid n} \mu\left(\frac{n}{d}\right) f(d)$$

这其实是说

$$g = f * 1 \Leftrightarrow f = g * \mu$$

一种理解的方法如下：

狄利克雷卷积中，\$1\$ 的逆是 \$\mu\$ 即 \$\varepsilon = 1 * \mu\$ 这很容易理解：对 \$(\mu * 1)(n)\$ 作出贡献的仅有 \$n\$ 的质因数的乘积和 \$1\$。

对于 \$n\$ 的质因数，如果 \$n\$ 有 \$m \geq 1\$ 个质因数，那它就有 \$\binom{m}{1}\$ 个“一个质因数的积” \$\binom{m}{2}\$ 个“两个质因数的积……他们卷起来的和是

$$(-1) \cdot \binom{m}{1} + (-1)^2 \cdot \binom{m}{2} + \cdots + (-1)^m \cdot \binom{m}{m} = [1 + (-1)]^m - 1$$

加上 \$1\$ 的贡献，即为 \$0\$。所以只有当 \$n=1\$ 的时候 \$(\mu * 1)(n)\$ 才为 \$1\$，故 \$\varepsilon = 1 * \mu\$

莫比乌斯反演的另一种不太常用的形式是，若函数 \$f(n)\$ 与 \$g(n)\$ 为数论函数，则

$$f(n) = \sum_{n \mid d} g(d) \Leftrightarrow g(n) = \sum_{n \mid d} \mu\left(\frac{d}{n}\right) f(d)$$

给出一些常用反演：

- \$\varepsilon = \mu * 1\$
- \$n = \varphi * 1 \Leftrightarrow \varphi = n * \mu\$

莫比乌斯反演与前缀、差分的联系

莫比乌斯反演实际上是一个高维前缀和与高维差分。

对 n 进行唯一分解 $n = \prod_{i=1}^m p_i^{k_i}$ 则 n 代表了一个高维空间的点 $(k_1, k_2, \dots, k_m)$ 且 n 的所有约数 d 代表的点与 $(k_1, k_2, \dots, k_m)$ 的关系都是一个高维偏序关系。因此若有 $f = g * 1$ 则 f 实际是 g 的一个高维前缀和。类似地，与 μ 的狄利克雷卷积可以将这个高维前缀和还原，因此 μ 实际上是与之相对的高维差分。

利用这个性质，在计算类似 $g(n) = (f * \mu)(n)$ 时不需要枚举每个 n 的因数 d 而只用枚举其质因数 p_i 依次利用差分将每个 p_i 对应的维度降下来，可以略微降低复杂度。

应用

二维GCD计数前缀和

给定 n, m, k 求 $\sum_{i=1}^n \sum_{j=1}^m [\gcd(i, j) = k]$

不使用函数变换的方法

不难发现：

$$\begin{aligned} \sum_{i=1}^n \sum_{j=1}^m [(i, j) = k] &= \sum_{i=1}^n \sum_{j=1}^m [\gcd(i/k, j/k) = 1] \\ &= \sum_{i=1}^{\lfloor n/k \rfloor} \sum_{j=1}^{\lfloor m/k \rfloor} \mu(\gcd(i, j)) \\ &= \sum_{i=1}^{\lfloor n/k \rfloor} \sum_{j=1}^{\lfloor m/k \rfloor} \sum_{g \mid \gcd(i, j)} \mu(g) \\ &= \sum_{g=1}^{\lfloor \min(n, m)/k \rfloor} \mu(g) \sum_{i=1}^{\lfloor n/(kg) \rfloor} \sum_{j=1}^{\lfloor m/(kg) \rfloor} 1 \end{aligned}$$

而 $\lfloor \frac{n}{kg} \rfloor$ 只有不超过 $\sqrt{n}$ 种取值， $\lfloor \frac{m}{kg} \rfloor$ 只有不超过 $\sqrt{n} + \sqrt{m}$ 种取值，因此可以将 $[1, n]$ 分成 $\sqrt{n} + \sqrt{m}$ 块，每一块的 $\lfloor \frac{n}{kg} \rfloor$ 和 $\lfloor \frac{m}{kg} \rfloor$ 取值都不变，则我们预处理 μ 后可以对一块区间进行 $O(1)$ 的统计，总时间复杂度为 $O(\sqrt{n} + \sqrt{m})$

使用函数变换的方法

令 $f(k) = \sum_{i=1}^n \sum_{j=1}^m [(i, j) = k]$ ， $g(k) = \sum_{i=1}^n \sum_{j=1}^m [k \mid \gcd(i, j)]$ 则 $f(k)$ 就是我们要求的答案。很明显 $k \mid \gcd(i, j) \Leftrightarrow k \mid i \text{ 且 } k \mid j$ 因此 $g(k) = \lfloor \frac{n}{k} \rfloor \lfloor \frac{m}{k} \rfloor$

发现 $g(k) = \sum_{d=1}^{\lfloor n/k \rfloor} f(d * k)$ 因此有：

$$\begin{aligned} f(k) &= \sum_{d=1}^{\lfloor n/k \rfloor} g(d * k) \mu(d) \\ &= \sum_{d=1}^{\lfloor n/k \rfloor} \lfloor \frac{n}{dk} \rfloor \lfloor \frac{m}{dk} \rfloor \mu(d) \end{aligned}$$

令 $n' = \lfloor \frac{n}{k} \rfloor, m' = \lfloor \frac{m}{k} \rfloor$ 则

$$f(k) = \sum_{d=1}^{\lfloor n/k \rfloor} \lfloor \frac{n'}{d} \rfloor \lfloor \frac{m'}{d} \rfloor \mu(d)$$

$\left\lfloor \frac{m}{d} \right\rfloor \left\lfloor \frac{n}{d} \right\rfloor \mu(d)$

类似上面可以证明 $\sum_{d|n} \sum_{d|m} \mu(d)$ 的取值个数，因此求解也是 $O(\sqrt{n} + \sqrt{m})$ 的。

好了，那求了一个区间后，怎么寻找下一个区间？假设我们当前区间开头为 i 并假设下一个区间为 j 则：

$$\left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor \mu(i) \leq \left\lfloor \frac{n}{j} \right\rfloor \left\lfloor \frac{m}{j} \right\rfloor \mu(j) \Rightarrow \left\lfloor \frac{n}{j} \right\rfloor \left\lfloor \frac{m}{j} \right\rfloor \mu(j) \leq \left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor \mu(i) \Rightarrow j \leq \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor \mu(i)}$$

同理可得 m ，因此 $j = \min\left(\left\lfloor \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor \mu(i)} \right\rfloor, \left\lfloor \frac{m}{\left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{i} \right\rfloor \mu(i)} \right\rfloor\right)$ 这个技巧在很多莫比乌斯反演的题目都得上。

求约数个数和

直接给出结论：

$$d(ij) = \sum_{x|i} \sum_{y|j} [(x, y) = 1]$$

以下给出一个简单的证明：

上式显然先决定 x 的取值，再决定 y 的取值。对于一个质因子 p 若 $p^a \mid i, p^b \mid j$ 且 $p^{a+b} \mid ij$ 则由于 $(x, y) = 1$ 一定有 $\min(a, b) = 0$ 故

- $p^0 \mid x$ 则表示 y 可以任意选 $p^1, \dots, p^b$ 等因子，分别映射到因数 $p^{a+1}, p^{a+2}, \dots, p^{a+b}$
- $p^1 \mid x, p^2 \mid x, \dots, p^a \mid x$ 则表示 x 可以任意选 $p^1, \dots, p^a$ 等因子，分别映射到因数 $p^1, p^2, \dots, p^a$
- $p^0 \mid x$ 且 $p^0 \mid y$ 等因子，映射到因数 p^0

综上，因子 $p^0, p^1, \dots, p^{a+b}$ 都能被唯一地表示出来且一一对应（双射），因此等式成立。[这里](#) 提供了另一种关于约数个数和的类似形式证明，但是使用了更合理的映射使得式子易于证明。

练习

莫比乌斯的题目通常能转化为 $(i, j) = 1$ 的计数问题，而转化为计数问题我们就容易通过分块求解了。

POI2007 Zap

二维 GCD 计数前缀和。

HAOI2011 Problem b

POI2007 Zap 的加强版，容斥原理加加减减就好了。

BZOJ2820 YY的GCD

仍然是二维 GCD 计数前缀和，不过需要 (i,j) 为质数。只要预处理质数的 μ 前缀和就好了。

SDOI2008 仪仗队

不被挡住即行列 $(i,j)=1$ (从 0 标号)，因此答案为 $(\sum_{i=1}^{n-1} \sum_{j=1}^{n-1} [(i,j)=1]) + 2$ 个是 $(0,1), (1,0)$ 。最终化为 $(\sum_{g=1}^n \mu(g) \lfloor n/g \rfloor)^2 + 2$ 分块求解。

SDOI2015 约数个数和

是道好题，然而需要结论。

令 $n' = \lfloor \frac{n}{g} \rfloor, m' = \lfloor \frac{m}{g} \rfloor$ 则

$$\begin{aligned} \sum_{i=1}^n \sum_{j=1}^m d(ij) &= \sum_{i=1}^n \sum_{j=1}^m [(i,j)=1] \\ &= \sum_{i=1}^n \sum_{j=1}^m \left\lfloor \frac{n}{i} \right\rfloor \left\lfloor \frac{m}{j} \right\rfloor \sum_{g \mid (i,j)} \mu(g) \\ &= \sum_{g=1}^n \mu(g) \sum_{i=1}^{\lfloor \frac{n}{g} \rfloor} \sum_{j=1}^{\lfloor \frac{m}{g} \rfloor} \left\lfloor \frac{n}{ig} \right\rfloor \left\lfloor \frac{m}{jg} \right\rfloor \\ &= \sum_{g=1}^n \mu(g) \sum_{i=1}^{n'} \sum_{j=1}^{m'} \left\lfloor \frac{n'}{i} \right\rfloor \left\lfloor \frac{m'}{j} \right\rfloor \end{aligned}$$

然后就可以预处理 $f(n) = \sum_{i=1}^n \left\lfloor \frac{n}{i} \right\rfloor$ 的值，每次询问就可以分块解决。之所以要预处理 $f(n)$ 是因为在倒数第二步时如果采用直接计算 $\sum_{i=1}^{n'} \left\lfloor \frac{n'}{i} \right\rfloor \sum_{j=1}^{m'} \left\lfloor \frac{m'}{j} \right\rfloor$ 开销是很大的。但如果我们能预处理，就能做到 $O(1)$ 计算。

预处理时间复杂度 $O(n\sqrt{n})$ 单次询问时间复杂度 $O(\sqrt{n})$

HNMTC2015#5 Lucas的数论

发现是 SDOI2015 约数个数和的单询问加强版本，上面对 μ 前缀和的 $O(n)$ 时间复杂度已经不能满足我们了，因此我们需要用杜教筛求出 $\mu(n)$ 前缀和，在 $O(n^{2/3})$ 时间内完成计算。

LOJ6627 等比数列三角形

比较有意思的莫比乌斯反演题。

枚举比值 $k = \frac{p}{q} \geq 1$ 中的 p, q 再枚举最短边 $x \leq x+kx < k^2x$ 可以得到约束 $k < \frac{1+\sqrt{5}}{2}$ 即 $q \leq \frac{p}{\phi}$ 同时 x 需要为 q^2 的整数倍，故 $q \leq \sqrt{x}$ 设 $x = iq^2$ 式子化为：

$$\sum_{q=1}^{\sqrt{n}} \sum_{p=q}^{\sqrt{n}} \sum_{(p,q)=1} \sum_{i=1}^{+\infty} [ip^2 \leq n] = \sum_{q=1}^{\sqrt{n}} \sum_{p=q}^{\sqrt{n}} \mu(p/q)$$

$$q \mid (p, q) = 1 \left\lfloor \frac{n}{p^2} \right\rfloor \sum_{q=1}^{\lfloor n/p \rfloor} \mu(q) \left\lfloor \frac{n}{pq} \right\rfloor$$

发现可以将 $\sum_{p=1}^{\sqrt{n}} \sum_{q=1}^{\lfloor n/p \rfloor} \mu(q) \left\lfloor \frac{n}{pq} \right\rfloor$ 原式进一步化为：

$$\sum_{p=1}^{\sqrt{n}} \sum_{q=1}^{\lfloor n/p \rfloor} \mu(q) \left\lfloor \frac{n}{pq} \right\rfloor = \sum_{p=1}^{\sqrt{n}} \sum_{q=1}^{\lfloor n/p \rfloor} \mu(q) \left\lfloor \frac{n}{pq} \right\rfloor$$

此时考虑简化对 $\sum_{q=1}^{\lfloor n/p \rfloor} \mu(q) \left\lfloor \frac{n}{pq} \right\rfloor$ 求和部分。我们发现这部分就是 $\sum_{d \mid p} \mu(d) \left\lfloor \frac{n}{d} \right\rfloor$ 而 $\sum_{d \mid p} \mu(d) \left\lfloor \frac{n}{d} \right\rfloor$ 很容易反演为 $\sum_{d \mid p} \mu(d) \left\lfloor \frac{n}{d} \right\rfloor$ 枚举 p 以及 p 的每个因数 d 由于 $1 \sim n$ 约数个数和是 $O(n \log n)$ 的，故复杂度为 $O(\sqrt{n} \log n)$

这里提供了一个 $O(\sqrt{n} \log \log \sqrt{n})$ 的做法，该做法基于莫比乌斯反演的前缀和和差分意义，使得计算 $g(n) = (f * \mu)(n)$ 时不需要枚举每个 n 的因数 d 而只用枚举质因数 p_i 使复杂度略微下降。

总结

莫比乌斯反演基本上离不开 GCD 和两个累和符号，而且通常通过将式子化为 $\sum_{d \mid n} \mu(d) f\left(\frac{n}{d}\right)$ 的形式，进而反演成 $\sum_{d \mid n} \mu(d) f\left(\frac{n}{d}\right)$ 并提出相关变量的形式，简化式子进行计算。求解一般通过数论分块和预处理 $\sum_{d \mid n} \mu(d)$ 前缀和的方式在 $O(\sqrt{n})$ 时间内求和。

- $\sum_{d \mid n} \mu(d) = 1$ 当 $n = 1$ 时， $\sum_{d \mid n} \mu(d) = 0$ 当 $n > 1$ 时。
- 当待分块函数（如 $\sum_{d \mid n} \mu(d)$ ）可以单独提出预处理时，可以通过此降低时间复杂度。
- 若多次询问中，分块区域下含有 GCD 的枚举值 g 和 i 或 j 之一，可以通过更换枚举变量改为枚举 g 或 j 的值，再枚举 g 加速。（说法很意识流，详见莫比乌斯反演简要笔记 - GCD 的幂）
- 积性函数有时不好证明，可以打表观察。重点观察幂和质数的值。

From:

<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=technique:mobius_inversion

Last update: 2020/05/31 16:43