

原文链接:

<https://github.com/abcwuhang/Algorithms/blob/master/multiplicative/multiplicative.pdf>

今天与大家分享一些数论知识和新(?)科技(与广为人知的杜教筛相得益彰~)

感谢 [zzq](#) 对 [pe](#) 某数论问题论坛里(我的)解法的扩充。

先介绍一下OI中常见的积性函数(multiplicative function)及性质。(不想看入门介绍的可直接跳到最后读正文干货☺)

对一个函数 $f$ ,若它满足以下几个条件,则称之为积性函数:

- 1、定义域为正整数集合
- 2、 $f(1) = 1$
- 3、对任意两个互素的正整数 $x$ 和 $y$ ,有 $f(xy) = f(x)f(y)$

由唯一分解定理可知,若 $n$ 的因式分解为 $n = \prod_{i=1}^k p_i^{e_i}$ (下同),则 $f(n) = \prod_{i=1}^k f(p_i^{e_i})$ 。可见 $f$ 完全由其在素数幂次处的值唯一确定。

常见的积性函数包括:

- 1、欧拉函数 $\varphi(n) = \prod_{i=1}^k p_i^{e_i-1} (p_i - 1)$ ,含义为1到 $n$ 中与 $n$ 互素的数的个数
- 2、约数个数函数 $\sigma_0(n) = \prod_{i=1}^k (e_i + 1)$ ,含义为 $n$ 的约数个数(即因子个数)
- 3、约数和函数 $\sigma_1(n) = \prod_{i=1}^k \sum_{j=0}^{e_i} p_i^j$ ,含义为 $n$ 的所有约数之和
- 4、莫比乌斯函数 $\mu(n) = \prod_{i=1}^k (-\lfloor \frac{1}{e_i} \rfloor)$ ,含义为:若 $n$ 有平方因子,则 $\mu(n) = 0$ ;若 $n$ 为奇数个素数之积,则 $\mu(n) = -1$ ;否则 $\mu(n) = 1$ 。

更进一步,完全积性函数(completely multiplicative function)的定义为:

- 1、定义域为正整数集合
- 2、 $f(1) = 1$
- 3、对任意两个正整数 $x$ 和 $y$ (无需互素),都有 $f(xy) = f(x)f(y)$

常见的完全积性函数包括:

- 1、全0函数
- 2、全1函数
- 3、 $f_a(x) = x^a$ ,其中 $a$ 为给定常数

4、 $f(1) = 1, f(x) = 0, \forall x > 1$  (即单位元函数 $\epsilon(n)$ )

接下来介绍一下常见的狄利克雷卷积。定义两个函数 $f$ 和 $g$ 的狄利克雷卷积为 $(f * g)(n) = \sum_{d|n} f(d)g(\frac{n}{d})$ 。

狄利克雷卷积算子满足：

1、交换律  $f * g = g * f$  (显然)

2、分配率  $f * (g + h) = f * g + f * h$  (这里加法指的是两个函数单点求和： $(f + g)(n) = f(n) + g(n)$ )

证明： $(f * (g + h))(n) = \sum_{d|n} f(d)(g(\frac{n}{d}) + h(\frac{n}{d})) = \sum_{d|n} f(d)g(\frac{n}{d}) + \sum_{d|n} f(d)h(\frac{n}{d}) = f * g + f * h$

3、结合律  $(f * g) * h = f * (g * h)$

证明： $((f * g) * h)(n) = \sum_{ab=n} (f * g)(a)h(b) = \sum_{ab=n} \sum_{cd=a} f(c)g(d)h(b) = \sum_{bcd=n} f(c)g(d)h(b)$ ，而 $(f * (g * h))(n) = \sum_{ab=n} f(a)(g * h)(b) = \sum_{ab=n} \sum_{cd=b} f(a)g(c)h(d) = \sum_{acd=n} f(a)g(c)h(d)$ ，对比式子可发现二者相同。

4、单位元  $f * \epsilon = f$  (显然)

接下来介绍一下莫比乌斯反演。如果对两个定义在正整数上的函数 $f$ 和 $g$ 满足 $\forall n, g(n) = \sum_{d|n} f(d)$ ，那么有 $\forall n, f(n) = \sum_{d|n} \mu(d)g(\frac{n}{d})$ 。

证明： $\sum_{d|n} \mu(d)g(\frac{n}{d}) = \sum_{d|n} \mu(\frac{n}{d})g(d) = \sum_{d|n} \sum_{i|d} \mu(\frac{n}{i})f(i) = \sum_{i|n} f(i) \sum_{k|\frac{n}{i}} \mu(k)$   
 $= \sum_{i|n} f(i) [\sum_{k|\frac{n}{i}} \mu(k) = 1] = f(n)$

这里用到了莫比乌斯函数的一个性质： $\sum_{d|n} \mu(d) = [n = 1] = [\frac{1}{n}]$ ，即当 $n = 1$ 时左边求和才1，否则左边为0。证明： $n = 1$ 时显然。若 $n = \prod_{i=1}^k p_i^{e_i} > 1$ ，那么由莫比乌斯函数定义，当 $d$ 含有平方因子时 $\mu(d) = 0$ 对求和没有贡献，非零值只在 $d$ 由互不相同的素因子相乘时取得，所以左边 $= \sum_{i=0}^k \binom{k}{i} (-1)^i = (1 - 1)^k = 0$ 。

入门知识完毕，下面开始讲新筛子（雾）。

给一个普通积性函数 $f$ ，现在希望求它的前缀和： $F(n) = \sum_{i=1}^n f(i)$ 。对任一积性函数 $g$ ，都存在一个积性函数 $h$ 满足 $f = g * h$ （方便起见，可以用除号表示 $h$ ，即 $h = f/g$ ）。将式子展开，有 $f(p^e) = \sum_{i=0}^e g(p^i)h(p^{e-i})$ （ $p$ 为素数， $e$ 为正整数）。特别的，有 $f(p) = g(1)h(p) + g(p)h(1) = g(p) + h(p)$ 。

记 $g$ 的前缀和为 $G(n) = \sum_{i=1}^n g(i)$ ，那么有 $F(n) = \sum_{i=1}^n f(i) = \sum_{i=1}^n \sum_{j|i} h(j)g(\frac{i}{j}) = \sum_{j=1}^n \sum_{k=1}^{\lfloor \frac{n}{j} \rfloor} h(j)g(k) = \sum_{j=1}^n h(j)G(\lfloor \frac{n}{j} \rfloor)$ 。如果我们能构造一个 $g$ ，使得相对应的 $h$ 在素数处的取值都为0（即对任意素数 $p$ ， $f(p) = g(p)$ ），那么我们只要求1到 $n$ 中所有幂数（powerful number）处的 $h$ 和 $G$ 的值即可。（一个数 $n$ 是幂数，当且仅当对 $n$ 的所有素因子 $p$ ， $p^2$ 都能整除 $n$ ）（ $h$ 在素数幂处的值可由上面 $f = g * h$ 表达式展开求出）

原理：观察前缀和表达式，只需要考虑 $h(j)$ 不为0时的贡献即可。由于对任意素数 $p$ 都有 $h(p) = 0$ ，那么 $h(j) \neq 0$ 当且仅当 $j$ 的因式分解表达式为 $\prod_{i=1}^k p_i^{e_i}$ ，其中所有 $e_i$ 均大于等于2。由于不大于 $n$ 的powerful number个数至多有 $O(\sqrt{n})$ 个，如果 $G$ 能在 $O(n^\alpha)$ 时间复杂度内求出，则求 $f$ 前缀和的算法时间复杂度为 $O(\max(\sqrt{n}, n^\alpha \cdot \frac{\zeta(2\alpha)\zeta(3\alpha)}{\zeta(6\alpha)}))$ ，其中 $\zeta$ 为黎曼函数。算法的实际表现非常强劲，上界 $10^{16}$ 都能几秒钟出结果，复杂度常数非常低。

下面看几个（迷之）例子：

- 1、 $f(p^e) = p$ ：取 $g(x) = x$ 即可满足 $f(p) = g(p)$ ，且 $g$ 前缀和非常好求！总时间复杂度为 $O(\sqrt{n})$
- 2、 $f(p^e) = e + 1$ ：取 $g(x) = \sigma_0(x)$ 即可满足 $f(p) = g(p)$ ，且 $g$ 前缀和非常好求！总时间复杂度为 $O(\sqrt{n})$
- 3、 $f(p^e) = p^e - 1$ ：取 $g(x) = \varphi(x)$ 即可满足 $f(p) = g(p)$ ，且 $g$ 前缀和非常好求（套一下杜教筛）！总时间复杂度为 $O(n^{\frac{2}{3}})$
- 4、 $f(p^e) = 2^e$ ：与第二个例子相同！
- 5、 $f(p^e) = p^e + 1$ ：取 $g(x) = \sigma_1(x)$ 即可满足 $f(p) = g(p)$ ，且 $g$ 前缀和非常好求（套一下整除分块）！总时间复杂度为 $O(\sqrt{n})$
- 6、 $f(p^e) = p + \lfloor \frac{3e}{2} \rfloor - 2$ ：与第三个例子相同！
- 7、 $f(p^e) = \varphi(p) \cdot e^e$ ：与第三个例子相同！（算 $e^e$ 会多一个 $\log$ 复杂度，无伤大雅）

总结：与杜教筛相得益彰，又双叒叕有一大堆能“快速求出”前缀和的积性函数了！而且算法非常简单好写，妙手偶得之。