**浅谈一类积性函数的前缀和**

置顶 2016年01月12日 14:53:45 [skywalkert](https://me.csdn.net/skywalkert" \t "_blank) 阅读数：34455

版权声明：本文为博主原创文章，未经博主允许不得转载。http://blog.csdn.net/skywalkert https://blog.csdn.net/skywalkert/article/details/50500009

**写在前面**

笔者在刷题过程中遇到一些求积性函数前缀和的问题，其中有一类问题需要在低于线性时间复杂度的算法，今天就来浅析一下这类问题的求解方法，当作以后讲课使用的讲义。若之后有了新的研究，再来继续完善这篇文章。   
**本文会随时更新内容，建议以链接形式转载，或者与笔者保持联系。爬虫转载不标明出处必究。**

author: skywalkert   
original article: <http://blog.csdn.net/skywalkert/article/details/50500009>   
last update time : 2017-04-01

**前置技能**

**积性函数的定义**

1. 若f(n)f(n)的定义域为正整数域，值域为复数，即f:Z+→Cf:Z+→C，则称f(n)f(n)为**数论函数**。
2. 若f(n)f(n)为数论函数，且f(1)=1f(1)=1，对于互质的正整数p,qp,q有f(p⋅q)=f(p)⋅f(q)f(p⋅q)=f(p)⋅f(q)，则称其为**积性函数**。
3. 若f(n)f(n)为积性函数，且对于任意正整数p,qp,q都有f(p⋅q)=f(p)⋅f(q)f(p⋅q)=f(p)⋅f(q)，则称其为**完全积性函数**。

**积性函数的性质与例子**

1. 常见的积性函数有   
   * 除数函数σk(n)=∑d|ndkσk(n)=∑d|ndk，表示nn的约数的kk次幂和，注意σk(n)σk(n)与σk(n)σk(n)是不同的。
   * 约数个数函数τ(n)=σ0(n)=∑d|n1τ(n)=σ0(n)=∑d|n1，表示nn的约数个数，一般也写为d(n)d(n)。
   * 约数和函数σ(n)=σ1(n)=∑d|ndσ(n)=σ1(n)=∑d|nd，表示nn的约数之和。
   * 欧拉函数φ(n)=∑ni=1[(n,i)=1]⋅1φ(n)=∑i=1n[(n,i)=1]⋅1，表示不大于nn且与nn互质的正整数个数，另外∑ni=1[(n,i)=1]⋅i=n⋅φ(n)+[n=1]2∑i=1n[(n,i)=1]⋅i=n⋅φ(n)+[n=1]2，且对于正整数n>2n>2来说φ(n)φ(n)是偶数。
   * 莫比乌斯函数μ(n)μ(n)，在狄利克雷卷积的乘法中与恒等函数互为逆元，μ(1)=1μ(1)=1，对于无平方因子数n=∏ti=1pin=∏i=1tpi有μ(n)=(−1)tμ(n)=(−1)t，对于有平方因子数nn有μ(n)=0μ(n)=0。
   * 元函数e(n)=[n=1]e(n)=[n=1]，狄利克雷卷积的乘法单位元，完全积性。
   * 恒等函数I(n)=1I(n)=1，完全积性。
   * 单位函数id(n)=nid(n)=n，完全积性。
   * 幂函数idk(n)=nkidk(n)=nk，完全积性。
2. 关于莫比乌斯函数和欧拉函数有两个经典的公式   
   * [n=1]=∑d|nμ(d)[n=1]=∑d|nμ(d)，将μ(d)μ(d)看作是容斥的系数即可证明。
   * n=∑d|nφ(d)n=∑d|nφ(d)，将in(1≤i≤n)in(1≤i≤n)化为最简分数统计个数即可证明。
3. 若f(n)f(n)为积性函数，则对于正整数n=∏ti=1pkiin=∏i=1tpiki有f(n)=∏ti=1f(pkii)f(n)=∏i=1tf(piki)；若f(n)f(n)为完全积性函数，则对于正整数n=∏ti=1pkiin=∏i=1tpiki有f(n)=∏ti=1f(pi)kif(n)=∏i=1tf(pi)ki。

**狄利克雷卷积与莫比乌斯反演**

1. 数论函数ff和gg狄利克雷卷积定义为(f∗g)(n)=∑d|nf(d)⋅g(nd)(f∗g)(n)=∑d|nf(d)⋅g(nd)，狄利克雷卷积满足交换律、结合律，对加法满足分配律，存在单位元函数e(n)=[n=1]e(n)=[n=1]使得f∗e=f=e∗ff∗e=f=e∗f，若ff和gg为积性函数则f∗gf∗g也为积性函数。
2. 狄利克雷卷积的一个常用技巧是对于积性函数ff与恒等函数II的卷积的处理，例如n=∏ti=1pkii,g(n)=∑d|nf(d)n=∏i=1tpiki,g(n)=∑d|nf(d)，则有g(n)=∏ti=1∑kij=0f(pji)g(n)=∏i=1t∑j=0kif(pij)。
3. 莫比乌斯反演也是对于g(n)=∑d|nf(d)g(n)=∑d|nf(d)的讨论，但是不要求ff是积性函数，适用于已知g(n)g(n)求f(n)f(n)的情况，由于I∗μ=eI∗μ=e，则g∗μ=f∗I∗μ=f∗e=fg∗μ=f∗I∗μ=f∗e=f，即f(n)=∑d|ng(d)⋅μ(nd)f(n)=∑d|ng(d)⋅μ(nd)，类似地有g(n)=∑n|df(d)⇒f(n)=∑n|dg(d)⋅μ(dn)g(n)=∑n|df(d)⇒f(n)=∑n|dg(d)⋅μ(dn)，二项式反演也是类似的技巧。有一个例子可以看出欧拉函数和莫比乌斯函数之间的关系，由于∑d|nφ(d)=id(n)∑d|nφ(d)=id(n)，所以φ(n)=∑d|nμ(d)ndφ(n)=∑d|nμ(d)nd，也即φ(n)n=∑d|nμ(d)dφ(n)n=∑d|nμ(d)d。

**正文：黑科技**

这种黑科技大概起源于Project Euler这个网站，由xudyh引入[中国](https://www.baidu.com/s?wd=%E4%B8%AD%E5%9B%BD&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "_blank)的OI、ACM界，目前出现了一些OI模拟题、OJ月赛题、ACM赛题是需要这种技巧在低于线性时间的复杂度下解决一类积性函数的前缀和问题。

首先看一个简单的例子，求前nn个正整数的约数之和，即∑ni=1σ(i)∑i=1nσ(i)，其中n≤1012n≤1012。   
显然不能直接做了，但是我们可以推导一番：

∑i=1nσ(i)=∑i=1n∑j=1n[j|i]⋅j=∑i=1ni⋅∑j=1n[i|j]=∑i=1ni⋅⌊ni⌋∑i=1nσ(i)=∑i=1n∑j=1n[j|i]⋅j=∑i=1ni⋅∑j=1n[i|j]=∑i=1ni⋅⌊ni⌋

当i≤n−−√i≤n时，⌊ni⌋⌊ni⌋显然只有O(n−−√)O(n)个取值；当i>n−−√i>n时，⌊ni⌋<n−−√⌊ni⌋<n显然也只有O(n−−√)O(n)个取值；对于固定的⌊ni⌋⌊ni⌋，ii的取值是一段连续的区间，这段区间是[⌊n⌊ni⌋+1⌋+1,⌊n⌊ni⌋⌋][⌊n⌊ni⌋+1⌋+1,⌊n⌊ni⌋⌋]，因此可以O(n−−√)O(n)[计算所](https://www.baidu.com/s?wd=%E8%AE%A1%E7%AE%97%E6%89%80&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)求。   
同样地，求前nn个正整数的约数个数之和也可以这样计算，留给读者练习。   
另外需要说明的是，∑ni=1⌊ni⌋⋅i=∑ni=1⌊ni⌋⋅(⌊ni⌋+1)2∑i=1n⌊ni⌋⋅i=∑i=1n⌊ni⌋⋅(⌊ni⌋+1)2，这也是一种常见的表示形式。

现在我们来加大一点难度，求前nn个正整数的欧拉函数之和，即∑ni=1φ(i)∑i=1nφ(i)，其中n≤1011n≤1011。   
目前本文提到的有关欧拉函数的公式只有几个，是否能用它们来帮助化简呢？答案是肯定的，接下来我们就利用∑d|nφ(d)=n∑d|nφ(d)=n来化简这个式子。   
这个公式也可以看成是φ(n)=n−∑d|n,d<nφ(d)φ(n)=n−∑d|n,d<nφ(d)，设ϕ(n)=∑ni=1φ(i)ϕ(n)=∑i=1nφ(i)，则有

ϕ(n)=∑i=1nφ(i)=∑i=1ni−∑d|i,d<iφ(d)=n⋅(n+1)2−∑i=2n∑d|i,d<iφ(d)=n⋅(n+1)2−∑id=2n∑d=1⎢⎣⎢⎢nid⎥⎦⎥⎥φ(d)=n⋅(n+1)2−∑i=2n∑d=1⌊ni⌋φ(d)=n⋅(n+1)2−∑i=2nϕ(⌊ni⌋)ϕ(n)=∑i=1nφ(i)=∑i=1ni−∑d|i,d<iφ(d)=n⋅(n+1)2−∑i=2n∑d|i,d<iφ(d)=n⋅(n+1)2−∑id=2n∑d=1⌊nid⌋φ(d)=n⋅(n+1)2−∑i=2n∑d=1⌊ni⌋φ(d)=n⋅(n+1)2−∑i=2nϕ(⌊ni⌋)

那么只要计算出O(n−−√)O(n)个ϕ(⌊ni⌋)ϕ(⌊ni⌋)的值，就可以计算出ϕ(n)ϕ(n)，这样的复杂度又如何呢？   
假设计算出ϕ(n)ϕ(n)的复杂度为T(n)T(n)，则有T(n)=O(n−−√)+∑n√i=1T(i)+T(ni)T(n)=O(n)+∑i=1nT(i)+T(ni)，**这里只展开一层就可以了，更深层的复杂度是高阶小量**，所以有T(n)=∑n√i=1O(i√)+O(ni−−√)=O(n34)T(n)=∑i=1nO(i)+O(ni)=O(n34)。   
由于ϕ(n)ϕ(n)是一个积性函数的前缀和，所以筛法也可以预处理一部分，假设预处理了前kk个正整数的ϕ(n)ϕ(n)，且k≥n−−√k≥n，则复杂度变为T(n)=∑nki=1ni−−√=O(nk√)T(n)=∑i=1nkni=O(nk)，当k=O(n23)k=O(n23)时可以取到较好的复杂度T(n)=O(n23)T(n)=O(n23)。   
之前利用φ(n)=n−∑d|n,d<nφ(d)φ(n)=n−∑d|n,d<nφ(d)的地方是怎么想到的呢？不妨来看一下这个

n⋅(n+1)2=∑i=1ni=∑i=1n∑d[d|i]⋅φ(d)=∑id=1n∑d=1⎢⎣⎢⎢nid⎥⎦⎥⎥φ(d)=∑i=1nϕ(⌊ni⌋)n⋅(n+1)2=∑i=1ni=∑i=1n∑d[d|i]⋅φ(d)=∑id=1n∑d=1⌊nid⌋φ(d)=∑i=1nϕ(⌊ni⌋)

**如果能通过狄利克雷卷积构造一个更好计算前缀和的函数，且用于卷积的另一个函数也易计算，则可以简化计算过程。**例如上题就是利用了φ∗I=idφ∗I=id的性质，但一定注意，不是所有的这一类题都只用配个恒等函数II就可以轻松完事的，有时需要更细致的观察。

定义梅滕斯函数M(n)=∑ni=1μ(i)M(n)=∑i=1nμ(i)，给定正整数nn，计算M(n)M(n)，其中n≤1011n≤1011。   
有了欧拉函数的经验，这次似乎就[轻车熟路](https://www.baidu.com/s?wd=%E8%BD%BB%E8%BD%A6%E7%86%9F%E8%B7%AF&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "_blank)了吧，使用[n=1]=∑d|nμ(d)[n=1]=∑d|nμ(d)来试试？

1=∑i=1n[i=1]=∑i=1n∑d|iμ(d)=∑i=1n∑d=1⌊ni⌋μ(d)=∑i=1nM(⌊ni⌋)1=∑i=1n[i=1]=∑i=1n∑d|iμ(d)=∑i=1n∑d=1⌊ni⌋μ(d)=∑i=1nM(⌊ni⌋)

因此M(n)=1−∑ni=2M(⌊ni⌋)M(n)=1−∑i=2nM(⌊ni⌋)，问题可在O(n23)O(n23)时间复杂度下解决。

看了上面的例子，是不是认为这种做法很naive，很好学啊，再来看一个题吧！   
令A(n)=∑ni=1i(n,i),F(n)=∑ni=1A(i)A(n)=∑i=1ni(n,i),F(n)=∑i=1nA(i)，求F(n)F(n)模(109+7)(109+7)的值，其中n≤109n≤109。   
先做一番化简，变成积性函数前缀和的样子：

A(n)=∑i=1ni(n,i)=∑i=1n∑d|n[(n,i)=d]⋅id=∑d|n∑id=1nd[(nd,id)=1]⋅id=12(1+∑d|nd⋅φ(d))A(n)=∑i=1ni(n,i)=∑i=1n∑d|n[(n,i)=d]⋅id=∑d|n∑id=1nd[(nd,id)=1]⋅id=12(1+∑d|nd⋅φ(d))

设G(n)=2⋅F(n)−nG(n)=2⋅F(n)−n，则

G(n)=2⋅F(n)−n=∑i=1n∑d|id⋅φ(d)=∑i=1n∑d=1⌊ni⌋d⋅φ(d)G(n)=2⋅F(n)−n=∑i=1n∑d|id⋅φ(d)=∑i=1n∑d=1⌊ni⌋d⋅φ(d)

因此要求的是ϕ′(n)=∑ni=1i⋅φ(i)ϕ′(n)=∑i=1ni⋅φ(i)。   
而对于n=∏ti=1pkiin=∏i=1tpiki，有

∑d|nd⋅φ(d)=∏i=1t∑j=0kipji⋅φ(pji)=∏i=1tp2ki+1i+1pi+1∑d|nd⋅φ(d)=∏i=1t∑j=0kipij⋅φ(pij)=∏i=1tpi2ki+1+1pi+1

这并不是什么好算前缀和的函数。   
但是不难发现(id⋅φ)∗id=id2(id⋅φ)∗id=id2，即

∑d|nd⋅φ(d)⋅nd=n⋅∑d|nφ(d)=n2∑d|nd⋅φ(d)⋅nd=n⋅∑d|nφ(d)=n2

，这是一个很好计算前缀和的函数，于是有

n⋅(n+1)⋅(2n+1)6=∑i=1ni2=∑i=1n∑d|id⋅φ(d)⋅id=∑id=1nid⋅∑d=1⎢⎣⎢⎢nid⎥⎦⎥⎥d⋅φ(d)=∑i=1ni⋅ϕ′(⌊ni⌋)n⋅(n+1)⋅(2n+1)6=∑i=1ni2=∑i=1n∑d|id⋅φ(d)⋅id=∑id=1nid⋅∑d=1⌊nid⌋d⋅φ(d)=∑i=1ni⋅ϕ′(⌊ni⌋)

因此ϕ′(n)=n⋅(n+1)⋅(2n+1)6−∑ni=2i⋅ϕ′(⌊ni⌋)ϕ′(n)=n⋅(n+1)⋅(2n+1)6−∑i=2ni⋅ϕ′(⌊ni⌋)，原问题可在预处理前O(n23)O(n23)个值的基础上，在O(n23logn)O(n23log⁡n)的时间复杂度下解决。

但是注意到这种方法的常数与复杂度都可能较高，有时候可能再进行一些推导可以得到一个不使用正文方法的做法，例如[ZOJ 3881 - From the ABC conjecture](http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=3881)，本文的方法与网上一个解法类似，可以用于求解此题，但是可以[这样推导之后](http://blog.csdn.net/skywalkert/article/details/50505260)得到更简单的一个做法。

需要使用此种方法的题目一般数据规模较大，例如n≤109,n≤1011,n≤1012n≤109,n≤1011,n≤1012，但是并不是都要使用此类方法，有时候可能存在其他O(n−−√),O(n23)O(n),O(n23)的做法，例如[51Nod 1222 - 最小公倍数计数](http://www.51nod.com/onlineJudge/questionCode.html#!problemId=1222)，会利用正文复杂度分析的方法即可，再例如[ZOJ 5340 - The Sum of Unitary Totient](http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemId=5340)，笔者不是很懂这题是否有其他做法能过，例如O(n34logn)O(n34log⁡n)**的积性函数求和方法（会在不久后更新）**，可能会因为数据组数较多而超时，网上的一个解法是分段压缩打表，具体问题需要具体分析。