杜教筛 学习总结

看了看唐老师的blog，照猫画虎的做了几道题目，感觉对杜教筛有些感觉了

但是稍微有一点难度的题目还是做不出来，放假的时候争取都A掉(挖坑ing)

这篇文章以后等我A掉那些题目之后再UPD上去就好啦

由于懒得去写怎么用编辑器写公式，所以公式就准备直接copy唐老师的啦

首先积性函数和完全积性函数什么的就不再多说了

列举常见的积性函数：

1、约数个数函数和约数个数和函数

2、欧拉函数phi

3、莫比乌斯函数mu

4、元函数e 其中e(n)=[n==1]

5、恒等函数I 其中I(n)=1

6、单位函数id 其中id(n)=n

有关狄利克雷卷积

定义：$(f*g)(n)=\sum_{d|n} f(d)\cdot g(\frac{n}{d})$

我们简单记为f*g

注意，狄利克雷卷积满足交换律，结合律，对于加法满足分配率

两个很显然的事情是：1、f*e=e*f=f

2、设h=f*g，若f和g是积性函数，则h是积性函数

积性函数的一些性质：

1、$\sum_{i=1}^{n}[(n,i)=1]\cdot i=\frac{n\cdot\varphi(n)+[n=1]}{2}$

2、e=mu*I

3、id=phi*I

之后我们由第二条性质可以得到一些很有趣的结论：

设存在函数g=f*I　　(通过f求g）

那么我们可以得到g*mu=f*I*mu

进而化简得到f=g*mu (通过g求f)

唐老师的博客里还举了一些更为有趣的例子

好了，前置技能都说完了，我们来说正题

杜教筛基本可表示成如下形式：

求F(n)=sigma(f(i))

存在g=f*I，定义G(n)=simga(g(i))

就可以得到F(n)=G(n)-sigma(F(n/i)) (这里i从2开始)

如果G(n)是可以在一定时间内求解(不一定要求O(1))

那么我们就可以做到O(n^(3/4))求解F(n)

如果加一些预处理我们可以做到O(n^(2/3))求解F(n)

下面通过几个例子来说一说为什么是这样的：

1、求sigma(phi(i))

由于我们知道id=phi*I

而且id的前缀和是可以O(1)的计算的

所以我们可以做以下推导

$$\frac{n\cdot(n+1)}{2}=\sum_{i=1}^{n}i=\sum_{i=1}^{n}\sum_{d}[d|i]\cdot\varphi(d)=\sum_{\frac{i}{d}=1}^{n}\sum_{d=1}^{\left\lfloor\frac{n}{\frac{i}{d}}\right\rfloor}\varphi(d)=\sum_{i=1}^{n}\phi(\lfloor\frac{n}{i}\rfloor)$$

对，没错，推完了之后就是我们上面提到过的形式，那么至于证明上面的形式的正确性就可以用这种推导方式啦

时间复杂度的分析:

假设计算出$\phi(n)$的复杂度为$T(n)$，则有$T(n)=O(\sqrt{n})+\sum_{i=1}^{\sqrt{n}}T(i)+T(\frac{n}{i})$，**只展开一层就可以了，更深层的复杂度是高阶小量**，所以有$T(n)=\sum_{i=1}^{\sqrt{n}}O(\sqrt{i})+O(\sqrt{\frac{n}{i}})=O(n^{\frac{3}{4}})$。

由于$\phi(n)$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k个正整数的$\phi(n)$，且$k\geq\sqrt{n}$，则复杂度变为$T(n)=\sum_{i=1}^{\frac{n}{k}}\sqrt{\frac{n}{i}}=O(\frac{n}{\sqrt{k}})$，当$k=O(n^{\frac{2}{3}})$时可以取到较好的复杂度$T(n)=O(n^{\frac{2}{3}})$。

这个结论非常的神奇，因为看上去复杂度像是sqrt(n)*sqrt(n)=n的，有兴趣的话可以自己化简一下，时间复杂度的确是上述所说

2、求sigma(mu(i))

有了上面的经验，我们又知道e=mu*I

那么我们就可以直接写了

$$1=\sum_{i=1}^{n}[i=1]=\sum_{i=1}^{n}\sum_{d|i}\mu(d)=\sum_{i=1}^{n}\sum_{d=1}^{\lfloor\frac{n}{i}\rfloor}\mu(d)=\sum_{i=1}^{n}M(\lfloor\frac{n}{i}\rfloor)$$

这样我们就机智的做出来了

唐老师的blog上还有一道题目，不过公式太长了，就不说了

有了上面学习的一些东西，我们就可以刷一些水题了

(感觉还是很好理解和学习的呢）

51Nod 莫比乌斯函数之和

直接套用上面的公式即可，但是真正写起来还是要注意一些事情的

由于杜教筛的时间复杂度的原因，通常情况下给定的n是超过int范围内，这就要求我们在计算过程中要时刻小心算术溢出

第二点是预处理多少比较合适，自己动手实验了一下，唐老师说的不错，大概预处理n^(2/3)较优

第三点就是在写的过程中要把每次递归计算好的函数用哈希表存进来实现记忆化，不然时间复杂度难以保证

+ View Code

51Nod 欧拉函数之和

直接写就可以了

+ View Code

BZOJ 3944 Sum

上面两道题目的混合版，唯一的坑点就是如果你想用int读入，在i=la+1的时候会有爆掉的风险

+ View Code

hdu 5608

QAQ 标准的不能在标准的杜教筛形式

首先n^2-3n+2的前缀和我们可以O(1)的求出来(表告诉我你不会平方前缀和公式）

之后我们考虑如果预处理f，我们设g(n)=n^2-3n+2

可以得到g=f*I，进而得到g*mu=f*I*mu,则f=g*mu

我自己比较懒，并不知道f是不是积性函数(貌似不是)

然后直接O(nlogn)的贡献过去的，这样就可以预处理前100w的f了

之后运用杜教筛就可以O(n^(2/3))求出答案了

+ View Code

51Nod 最大公约数之和

貌似网上并没有题解，那我就来发一份吧，没准能骗骗访问量

注意到这道题是n和n，不是n和m

我们不妨枚举最大公约数k

又因为我们知道(i,j)=k成立当且仅当(i/k,j/k)=1成立

然后我们就可以把原来的式子转化一下变成：

设P(n)=sigma(phi(i))

原式=2*sigma(k*P(n/k))-n*(n+1)/2

式子转化的正确性是显然的

这样我们利用杜教筛就可以求出P(n/k)的值

由于我们知道杜教筛求解所有n的约数k的所有的P(k)的值只需要O(n^(2/3))

所以我们在外面套上一个分块也不会影响时间复杂度

总时间复杂度O(sqrt(n)+n^(2/3))

+ View Code

BZOJ 4176

貌似是SDOI2015某道题的加强版？

接下来是一个悲伤的故事，故事的名称叫做不会用博客园写公式的代价

首先我们有一个结论：

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

好吧，凑合着看吧，结论的证明我知道的有三种，其中最简单的一种是利用数学归纳法

然后式子就变成了(把e函数等价代换成mu函数)

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{x|i}\sum_{y|j}\sum_{k|(x,y)}\mu(k)$$

然后我们枚举k，式子等价转换成

$$\sum_{k=1}^{n}\mu(k)\sum_{k|x}\sum_{x|i}1\sum_{k|y}\sum_{y|j}1$$

我们很容易发现中间那两个sigma连一起的奇怪的东西其实就是这个玩意

$$F(k)=\sum_{k|x}\sum_{x|i}1$$

$$=\sum_{\frac{i}{k}=1}^{\lfloor\frac{n}{k}\rfloor}g\left(\frac{i}{k}\right)=\sum_{i=1}^{\lfloor\frac{n}{k}\rfloor}g(i)$$

好了，终于不用在写这歪歪扭扭的公式了，结束了QAQ

g函数是约数个数函数，我们求约数个数函数的前缀和可以做到O(sqrt(n))

实际做法是枚举每个数i，(n/i)就是这个数对答案的贡献，分块求和即可

然后我们整体枚举k，做分块求和，某一段mu的和我们可以用杜教筛搞定

由于段的起点和终点都是n的约数，所以杜教筛的时间复杂度是O(n^(2/3))的

至于两次分块求和叠加看似是sqrt(n)*sqrt(n)=n的，但是根据上面的时间复杂度分析

时间复杂度是O(n^(3/4))的，到此这道题我们就做完了

(我一定要学会如何用blog写公式(太丑了，捂脸。。挖坑ing))

写这道题的时候我发现一个很逗的事情

就是我之前的题目的哈希表的push其实不用看之前有没有push过

直接push进去就好了，因为之前已经ask过发现并没有放进去了

代码在BZOJ上跑的还是很快的

+ View Code

UPD:最近又做了几道51Nod的题目，公式推导不知道比上面的长到哪里去了

鉴于本人并不会用编辑器写公式，所以就只放代码了

51Nod 最小公倍数之和

+ View Code

51Nod 平均最小公倍数

+ View Code

51Nod 最小公倍数计数

+ View Code

51Nod 约数之和

+ View Code

Em，总结一下吧

杜教筛具有非常明显的特征：

1、数据范围特征:10^9,10^10,10^11

2、能解决的问题是求解某个函数的前缀和或者一段区间的和

但是前提是其和I函数的狄利克雷卷积的前缀和是比较好求的

3、可以在预处理的情况下O(n^(2/3))的时间复杂度内求出O(sqrt(n))个值，也就是n的约数的所有值

因为有记忆化的存在，所以在计算时间复杂度的时候杜教筛不能简单的跟其他东西相乘来计算

4、一定不要忘记记忆化，一定不要忘记预处理

5、由于数据范围的特殊性，所以在计算过程中非常容易整数溢出

在写的时候一定要仔细分析之后在码，码完之后要通过对拍或者测极限数据等方法来检验

留下的一些坑：

51Nod 加权约数之和

BZOJ 3512