min_25筛的推导/实现与复杂度分析.

这是(af)oier spinach退役前的笔记与解题记录,不能保证内容的严谨性,仅供参考如果您发现文章存在可能的错误,请立刻联系我.

作者正在学文化课,不能保证及时修正,抱歉.

参考文献:

- 1. zzt: 2018集训队论文.
- 2. zzt: WC2019讲课课件.
- 3. zza blog
- 4. zhb2047 blog
- 5. xehoth blog
- 6. cz xuyixuan blog
- 7. 简单易懂的质数筛法-过时数论魔法.ppt,一个APIO的讲课内容,出门左转UOJ用户群.

min_25筛

 min_25 提出.复杂度 $O(n^{1-\epsilon})$.是近线性复杂度.

在 $n \leq 10^{13}$ 时可以认为是 $O(\frac{n^{\frac{3}{4}}}{\log n})$ 的.

是OI中最好用的筛法(洲阁筛常数太大...而且不好写不好推导...可以说只有理论价值了).

例子

$$egin{aligned} n &= \prod_{i=1}^m {p_i}^{c_i} \ \phi(n,d) &= \prod_{i=1}^m (p^{c_i} + d) \ \phi(1,d) &= 1 \ \ \phi(p) &= G(p) = p + d \ \phi(p^c) &= T(p,c,d) = p^c + d \end{aligned}$$

求 $S(n) = \sum_{i=1}^{n} f(i)$.

- 满足 $f(p^c)$ 是一个关于p, c的**低次**多项式多项式.
- f(n)是积性函数

考虑 提取最小质因子 加速计算.

$$\sum_{i=1}^n f(i) = 1 + \sum_{\substack{2 \leq p^c \leq n \ p \in prime}} f(p^c) (1 + \sum_{\substack{2 \leq x \leq \lfloor rac{n}{p^c}
floor} \ minprime_x > p}} f(x))$$

对于合数x必定有. $minprime_x \leq \sqrt{x} \leq \sqrt{n}$ 继续分类.

$$1 + \sum_{\substack{2 \leq p^c \leq n \ p \in prime \ p < \sqrt{n}}} f(p^c) (1 + \sum_{\substack{minprime_x > p \ p \leq x \leq \lfloor rac{n}{p^c} \rfloor}} f(x)) + \sum_{\substack{p \in prime \ \sqrt{n}$$

构造辅助函数 $g_{n,m}$ 与 h_n 定义如下:

- $g_{n,m}$ 为n以内,最小质因子大于m的贡献.
- h_n 为n以内素数的贡献.

$$egin{aligned} g_{n,m} &= \sum_{\substack{minprime_x > m \ 2 \leq x \leq n}} f(x) \ h_n &= \sum_{\substack{p \in prime \ 2$$

寻找g的递归式

$$egin{align*} g_{n,m} &= \sum_{\substack{minprime_x > m \ 2 \leq x \leq n}} f(x) \ &= \sum_{\substack{p^c \leq n \ p \in prime \ m p \ m \leq x \leq \lfloor rac{n}{p^c} \rfloor}} f(x)) + \sum_{\substack{p \in prime \ m$$

根据定义 $\sum_{i=1}^n f(i) = S(n) = g_{n,1} + 1$,如果求出了所有需要用到的 $h_m(\exists i \quad \lfloor \frac{n}{i} \rfloor = m)$ 即可递归求出g解释:观察递归式,根据 $\lfloor \frac{\lfloor \frac{n}{a} \rfloor}{b} \rfloor = \lfloor \frac{n}{ab} \rfloor$ 用到的 h_m 必然是 $\lfloor n/i \rfloor$ 的形式.或者有 $m \leq \sqrt{n}$,需要求解的 h_m 是 $O(\sqrt{n})$ 量级的

$$h_n = \sum_{\substack{p \in prime \ 2$$

由于 $f(p^c)$ 是G(p,c),一个低次多项式,把f(p)拆成 $\sum_{i=0}^k c_i p^i$ 的形式.我们只需要计算这个.(素数k次幂和)

$$h_n = \sum_{\substack{p \in prime \ 2$$

定义

$$L_{n,m} = \sum_{i=2}^n [(i \in prime) \, or \, (minprime_i > prime_m)] \, \, i^k$$

即范围(1,n] 内满足质数k次幂与 $,minprime_x>p_m$ (最小质因子足够大,大于第m个素数)的 数的k次幂和. 也可以说是埃氏筛第m轮筛除后剩余的数字的k次方和.

考虑一个素数筛的过程(欧拉筛的过程,用最小质因子进行筛除)

解释:考虑最小质因子为 p_m 的数 $x=p_m\cdot y\leq n\Rightarrow y\leq \lfloor \frac{n}{p_m} \rfloor$

但是 $p_m^k L_{\lfloor \frac{n}{p_m} \rfloor, m-1}$ 里面包含了满足 $x=q\cdot p_m \le n, minprime_q < p_m$ 的x,它们并不应该在这一轮中筛掉,而应该在之前用更小的质数筛除.

再考虑这种q,必定是 $q=p_i < p_m, p_i \in prime$ 的,不然q已经被筛掉了.所以补上一个 pre_{m-1} ,这样找到的 $x=p_m \cdot y$ 就是我们本轮需要筛除的数了.

这样做扣除且仅扣除了所有满足 $x \notin prime, x \leq n, minprime_x = p_m$ 的x所以筛除的正确性得到了保证.

$$egin{aligned} L_{n,m} &= L_{n,m-1} - p_m^k (L_{\lfloor rac{n}{p_m}
floor, m-1} - pre_{m-1}) \ pre_m &= \sum_{i=1}^m prime_i^k \end{aligned}$$

如果 $p_m^2 > n$ 了那么不需要转移...有 $L_{n,m} = L_{n,m-1}$ 理由如下.

设最小质因子大于 p_j 的最小合数为x必定有 $x\geq p_j^2>$ n使用 p_j 不能再不重复地筛掉[1,i]内的非质数.(可以考虑埃试筛的过程.这时,用 p_j 去筛[1,n]中的倍数.这些合数已经被更小的质数筛过了,应该从 p_j^2 开始筛)

复杂度

村教筛的复杂度.

$$egin{aligned} \sum_{i=1}^n (f*g)(i) &= \sum_{i=1}^n \sum_{d \mid i} f(d)g(rac{i}{d}) = \sum_{d=1}^n f(d)\sum_{i=1}^{\lfloor rac{n}{d}
floor} g(i) \ &let \, S(n) = \sum_{i=1}^n g(i) \ &\sum_{i=1}^n (f*g)(i) = \sum_{d=1}^n f(d)S(\lfloor rac{n}{d}
floor) \ &S(n) = \sum_{i=1}^n (f*g)(i) - \sum_{d=2}^n f(d)S(\lfloor rac{n}{d}
floor) \end{aligned}$$

由于采用了记忆化搜索,每个状态只会向下递归一次,我们可以考虑总转移次数来估计复杂度.即考虑每个状态会依赖多少状态,这会导致算法到hashmap中查询之前的计算结果.考虑x依赖 $S_x=\{d_1,d_2,d_3\dots d_n\}$.那么对复杂度的贡献为 $|S_x|$,而 S_x 是 $\{\lfloor \frac{x}{d}\rfloor \mid d \leq n,d \in N\}$ 从而 $|S_x|=O(\sqrt{x})$.然后进行分类计算, $d,\lfloor \frac{n}{d}\rfloor$ 中至少有一个不超过 \sqrt{n} ,枚举这个数进行计算.

$$T(n) = \sum_{i=1}^{\sqrt{n}} \sqrt{i} + \sum_{i=1}^{\sqrt{n}} \sqrt{rac{n}{i}} \leq \int_0^{\sqrt{n}} \sqrt{x} dx + \int_0^{\sqrt{n}} \sqrt{rac{n}{x}} dx \ \int_0^{\sqrt{n}} \sqrt{x} dx + \int_0^{\sqrt{n}} \sqrt{rac{n}{x}} dx = O(n^{rac{1}{2} imes rac{3}{2}}) + O(n^{rac{1}{2} imes rac{1}{2} imes rac{1}{2}}) = O(n^{rac{3}{4}})$$

筛法求 h_n 的复杂度分析.这里由于涉及了对数积分,比较复杂,我们进行一些估计并且利用Wolfram Alpha来计算.

$$egin{aligned} \pi(n) &= \sum_{x \leq n} [x \in prime] = O(rac{n}{log\,n}) \ &T(n) = \sum_{i=1}^{\sqrt{n}} \pi(\sqrt{i}) + \sum_{i=1}^{\sqrt{n}} \pi(\sqrt{rac{n}{i}}) \ &\int_0^{\sqrt{n}} rac{x}{log\,x} dx + \int_0^{\sqrt{n}} rac{\sqrt{rac{n}{x}}}{\log\sqrt{rac{n}{x}}} = O(rac{n^{rac{3}{4}}}{log\,n}) \end{aligned}$$

- 筛出 h_n 的复杂度为 $O(\frac{n^{\frac{3}{4}}}{\log n})$.
- DFS计算g这部分是玄学的 $O(n^{1-\epsilon})$ 近线性复杂度,不过 $n \leq 10^{13}$ 时可以认为是 $O(\frac{n^{\frac{3}{4}}}{\log n})$

代码实现

如何正确地实现 Sieve of Eratosthenes, Euler's Sieve

```
1
   int vis[N],prime[N],cnt;
2
3
   void sieve(int n){
4
       vis[1]=1;
5
        for(int i=2;i*i<=n;i++) if(!vis[i]){</pre>
6
            prime[++cnt]=i;
7
            for(int j=i*i;j<=n;j+=i) vis[j]=1;</pre>
8
       }
9
   }
```

```
int vis[N],prime[N],cnt,minp[N],pk[N];
 1
 2
    void sieve(int n){
 3
        vis[1]=1;
        for(int i=2;i<=n;i++){
 4
 5
             if(!vis[i]){
 6
                 prime[++cnt]=i;
 7
                 minp[i]=i;
                 pk[i]=1;
 8
 9
             }
10
             for(int j=1;j<=cnt&&i*prime[j]<=n;j++){</pre>
11
                 vis[i*prime[j]]=1;
12
                 minp[i*prime[j]]=prime[j];
13
                 if(i%prime[j]==0){
14
                     pk[i*prime[j]]=pk[i]+1;
15
                     break;
16
                 }
17
                 pk[i*prime[j]]=1;
18
19
        }
    }
20
```

```
预处理\sqrt{n}以内的质数与pre_x=\sum_{i=1}^x prime_x^k. 找到m,\exists i\lfloor \frac{n}{i}\rfloor=m,这些m的h是需要计算的.共O(\sqrt{n})项,初始化h(m)=\sum_{i=2}^m i^k从小到大枚举prime_j,从m=n开始到m\geq p_j^2的m=\lfloor \frac{n}{d}\rfloor向下枚举,进行筛除操作 h(m)=h(m)-p_j^k(h(\lfloor \frac{m}{p_j}\rfloor)-pre_{m-1})
```

下面是 SPOJ DIVCNT κ的代码,朴素实现,没有使用上面提到的技巧去掉hashmap,也没有什么常数优化.

之后会写求更复杂度的积性函数前缀和以及部分并非积性的函数(典型的是最大/最小/次大次小质因子)

```
1 #include <iostream>
 2
   #include <cstdio>
 3 #include <ctime>
 4 #include <algorithm>
   #include <unordered_map>
 5
   #include <bitset>
 6
 7
    using namespace std:
    typedef unsigned long long Int;
 8
    const int N=10000000+10;
 9
10
    int vis[N],prime[N],cnt,pre[N];
    inline void init(){
11
12
        for(int i=2;i<N;i++){</pre>
             pre[i]=pre[i-1]+(vis[i]==0);
13
14
             if(!vis[i]) prime[++cnt]=i;
             for(int j=1;j<=cnt&&1LL*i*prime[j]<N;j++){</pre>
15
16
                 vis[i*prime[j]]=1;
17
                 if(i%prime[j]==0) break;
18
            }
19
        }
20
    }
21
    namespace SPINACH{
22
        Int n,k,h[N],m;
23
        unordered_map<Int,int> id;
        inline Int geth(Int x){ return (k+1)*(x<N?pre[x]:h[id[x]]); }</pre>
24
25
        Int g(Int n,Int m){
26
             m++; if(prime[m]>n) return 0;
27
             Int s=0;
28
             while(m<=cnt&&1LL*prime[m]*prime[m]<=n){</pre>
29
                 Int p=prime[m],pc=p,c=1;
30
                 while(pc<=n){
31
                     s = (k*c+1)*(1+g(n/pc,m));
32
                     pc=pc*p;c++;
33
                 }
34
                 m++;
35
36
             return s+(geth(n)-geth(prime[m-1]));
37
         }
38
        Int solve(Int n,Int _k){ id.clear();
39
             k=_k;
```

```
40
             Int l=1, q=0; m=0;
41
             while(1 \le n) \{
42
                  q=n/1; l=n/q+1;
                  h[id[q]=++m]=(q-1);
43
44
             }
             for(int i=1;i<=cnt&llL*prime[i]*prime[i]<=n;i++){</pre>
45
46
                  Int p=prime[i]; l=1;q=0;m=0;
47
                  while(1 \le n){
                      q=n/1; l=n/q+1;
48
49
                      if(p*p>q) break;
50
                      h[++m]-=(h[id[q/p]]-1LL*(i-1));
                  }
51
52
53
             return 1+g(n,0);
54
         }
55
    int main(){
56
57
         ios::sync_with_stdio(0);
58
         cin.tie(0); cout.tie(0);
59
60
         init();
61
         int T;cin>>T; Int n,k;
62
         while(T--){
63
             cin>>n>>k;
             cout<<SPINACH::solve(n,k)<<'\n';</pre>
64
65
         }
66
         return 0;
    }
67
```

这里是使用了小技巧去掉hashmap的写法.效率足够对付大多数题目.多组询问记得遍历 $\left| \frac{n}{i} \right|$ 清空.

$$at(x) = egin{cases} pre_0(x) & x \leq \sqrt{n} \ pre_1(x) & x > \sqrt{n} \end{cases}$$

```
1 #include <iostream>
      2
                    #include <cstdio>
      3 #include <ctime>
      4 #include <algorithm>
      5
                    #include <bitset>
                    using namespace std;
      7
                       typedef unsigned long long Int
      8
                        const int N=1000000+10;
     9
                       int vis[N],prime[N],cnt,pre[N];
 10
                         inline void init(){
 11
                                                 for(int i=2;i<N;i++){
                                                                        pre[i]=pre[i-1]+(vis[i]==0);
12
13
                                                                       if(!vis[i]) prime[++cnt]=i;
                                                                       for(int j=1; j \leftarrow t_{k-1} = t_{j-1} = t_{k-1} = t_{j-1} = t_{j-1}
14
15
                                                                                              vis[i*prime[j]]=1;
 16
                                                                                              if(i%prime[j]==0) break;
 17
```

```
18
19
    }
20
    namespace SPINACH{
21
         Int n,k,h[N];
22
         Int pre0[N],pre1[N];
23
         inline Int& at(Int x){ return (x*x \le n)?pre0[x]:pre1[n/x]; }
24
         inline Int geth(Int x){ return (k+1)*(x<N?pre[x]:at(x)); }</pre>
25
         Int g(Int n,Int m){
26
             m++; if(prime[m]>n) return 0;
27
             Int s=0;
28
             while(m<=cnt&&1LL*prime[m]*prime[m]<=n){</pre>
29
                  Int p=prime[m],pc=p,c=1;
30
                  while(pc<=n){</pre>
                      s += (k*c+1)*(1+g(n/pc,m));
31
32
                      pc=pc*p;c++;
33
                  }
34
                  m++;
35
36
             return s+(geth(n)-geth(prime[m-1]));
37
38
         Int solve(Int _n,Int _k){
39
             n=_n; k=_k;
40
             Int 11, q=0;
41
             while(1 \le n) \{
42
                  q=n/1; l=n/q+1;
43
                  at(q)=(q-1);
44
             }
             for(int i=1;i<=cnt&&1LL*prime[i]*prime[i]<=n;i++){</pre>
45
46
                  Int p=prime[i]; l=1;q=0;
47
                  while(1 \le n){
48
                      q=n/1; l=n/q+1;
49
                      if(p*p>q) break;
50
                      at(q)=(at(q/p)-1LL*(i-1));
51
                  }
52
53
             return 1+g(n,0);
54
         }
55
    }
    int main(){
56
57
         ios::sync_with_stdio(0);
         cin.tie(0); cout.tie(0);
58
59
60
         init();
61
         int T;cin>>T; Int n,k;
         while(T--){
62
63
             cin>>n>>k;
64
             cout<<SPINACH::solve(n,k)<<'\n';</pre>
         }
65
66
         return 0;
67
    }
```

例题

SPOJ divcnt

系列题目, divcnt即为(counting divisor),约数个数计数问题.有四题,均为min_25提供.

$$egin{aligned} \sum_{i=1}^n \sigma_0(i) & n \leq 2^{64} \ \sum_{i=1}^n \sigma_0(i^2) & n \leq 10^{12} \ \sum_{i=1}^n \sigma_0(i^3) & n \leq 10^{11} \ \sum_{i=1}^n \sigma_0(i^k) & n \leq 10^{10} \end{aligned}$$

第一个是Stern Brocot Tree的题,具体做法可以在参考文献1中学习,是一个典型的论文题.

第二个和第三个,是推式子+反演+卷积优化求和(杜教筛),但是可以使用本文提供的min 25筛通过.

最后一个是 \min_25 筛的板子题,事实上,这个题也有 $O(n^{\frac{2}{3}})$ 的做法,不过国内没有参考资料,可以自行找 \min_25 提供的算法或者查阅相关论文学习.

• LOI6053 简单的函数

给定一个积性函数 $f(p^c) = p \oplus c$,求 $\sum_{i=1}^n f(i)$ $n \leq 10^{10}$,那个 \oplus 是位运算xor

$$f(p) = egin{cases} p+1 & p=2 \ p-1 & p>2 \end{cases}$$
 $f(p^c) = p \oplus c$

直接套公式(雾)

小心爆long long...小心爆ull,小心爆int128...我这份代码重写三次才过的.

```
1 #include <iostream>
   #include <algorithm>
 3 #include <cmath>
 4 using namespace std;
 5
   typedef unsigned long long UInt;
 6 typedef long long Int;
 7
   const int N=300000+10;
 8 const Int mod=(Int)(1e9)+7LL;
 9 Int qpow(Int a,Int p){
10
       if(p==0) return 1;
11
        Int r=qpow(a,p>>1);
12
        r=r*r%mod;
13
        return (p&1)?(r*a%mod):r;
   }
14
15 | Int inv2;
16 | UInt n;
17 int vis[N],prime[N],cnt;
18 | Int pre0[N],pre1[N];
19
   UInt qwq[N];
20 double SQRTN=0;
21 void init(){
       for(int i=2;i<N;i++){
22
23
            pre0[i]=pre0[i-1];
24
            pre1[i]=pre1[i-1];
```

```
25
             qwq[i]=qwq[i-1];
26
             if(!vis[i]){
27
                 prime[++cnt]=i;
28
                 pre0[i]++;
29
                 pre1[i]=(pre1[i]+i)%mod;
30
             }
31
             for(int j=1; j <= cnt \& prime[j] * i < N; j++){
32
                 vis[i*prime[j]]=1;
                 if(i%prime[j]==0) break;
33
             }
34
35
        }
36
    }
37
    // f(p) = p-1 (p>2)
    // f(p=2) = (p=2)+1
38
39
    Int pre0A[N],pre1A[N];
40
    Int preOB[N],pre1B[N];
    inline Int& at(UInt x,int y){ // x*x <= n,overflow uint64!}
41
42
        if(x<=SQRTN) return y?pre1A[x]:pre0A[x];</pre>
43
         return y?pre1B[n/x]:pre0B[n/x];
44
    inline Int geth(UInt x){
45
46
        Int ret=2*(x>=2);
47
        if(x<N) ret+=( (pre1[x]-pre0[x])\%mod);
48
        else ret+=( (at(x,1)-at(x,0))%mod );
        return (ret%mod+mod)%mod;
49
50
51
    Int g(UInt n,int m){
        m++; if(1ULL*prime[m]>n) return 0;
52
53
54
        while(m<=cnt&&1ULL*prime[m]*prime[m]<=n){</pre>
55
             UInt p=prime[m],pc=p,c=1;
56
             while(pc<=n){
57
                 s=(s+((p\land c)\mbox{mod})*(1+g(n/pc,m))\mbox{mod})\mbox{mod};
58
                 c++;pc=pc*p;
59
             }
60
             m++;
        }
61
62
        Int tmp=( (geth(n)-geth(prime[m-1]))%mod+mod)%mod;
63
        return (s+tmp)%mod;
64
    inline Int s1(Int x){ x%=mod; return x*(x+1)%mod*inv2%mod; }
65
    inline Int sub(Int a,Int b){
66
67
        a=(a\%mod+mod)\%mod;
68
        b=((-b)\%mod+mod)\%mod;
69
        return (a+b)%mod;
70
71
    inline Int mul(Int a,Int b){
72
        a=(a\%mod+mod)\%mod;
73
        b=(b%mod+mod)%mod;
        return a*b%mod;
74
75
76
    Int solve(){
77
        UInt l=1,q=0;
```

```
78
          while(1 \le n){
 79
              q=n/1; l=n/q+1;
 80
              at(q,0)=(q-1);
              at(q,1)=(s1(q)+mod-1)%mod;
 81
 82
          for(int i=1;i<=cnt&&1ULL*prime[i]*prime[i]<=n;i++){</pre>
 83
 84
              l=1;q=0;Int p=prime[i];
 85
              while(1 \le n) \{
 86
                  q=n/1; l=n/q+1;
 87
                  if(p*p>q) break;
                  //at(q,0) = (at(q/p,0) - (i-1));
 88
 89
                  //at(q,1) -= p*(at(q/p,1) - pre1[i-1]);
 90
                  at(q,0)=sub(at(q,0),sub(at(q/p,0),pre0[p-1]));
                  at(q,1)=sub(at(q,1),mul(p,sub(at(q/p,1),prel[p-1])));
 91
              }
 92
 93
          }
 94
          Int ret=1+g(n,0);
 95
          return ret%mod;
     }
 96
 97
 98
     int main(){
 99
          cin>>n; SQRTN=sqrt(n);
100
          inv2=qpow(2,mod-2);
101
          init();
102
          cout<<solve()<<endl;</pre>
103
          return 0;
     }
104
```

• UOJ Round 13,C:sanrd $求 \sum_{i=1}^n f(i), \not\exists \text{ $p \cdot k$}$ 即质因子可重集合的次大元素. 定义 f(p) = 0, f(1) = 0

```
1 #include <iostream>
 2
   #include <algorithm>
 3 #include <ctime>
   #include <cmath>
 5
   #include <cassert>
   using namespace std;
 6
 7
   typedef long long Int;
    const int N=1000000+10;
 8
 9
    int vis[N],prime[N],cnt;
    void init(){
10
        for(int i=2;i<N;i++){</pre>
11
            if(!vis[i]) prime[++cnt]=i;
12
13
            for(int j=1;j<=cnt&i*prime[j]<N;j++){}
14
                 vis[i*prime[j]]=1;
15
                if(i%prime[j]==0) break;
            }
16
```

```
17 }
    }
18
19
    Int pre0[N],pre1[N],n;
    Int& at(Int m){ return 1.0*m*m<=n?pre0[m]:pre1[n/m]; }</pre>
20
21
     Int g(Int n,int m){
22
         m++; if(n<=2||prime[m]>n) return 0;
23
         Int s=0;
24
         while(prime[m]*prime[m]<=n&&m<=cnt){</pre>
25
             Int p=prime[m], c=1, pc=p;
26
             while(pc*p<=n){</pre>
27
                  s+=(g(n/pc,m)+p*(at(n/pc)-(m-1)));
28
                  pc=pc*p;c++;
29
             }
30
             m++;
31
         }
32
         return s;
33
    }
34
    Int solve(Int _n){
35
         ::n=_n;
36
         Int l=1, q=0;
37
         while(1 <= n){
38
             q=n/1; l=n/q+1;
39
             at(q)=q-1;
40
         }
41
         for(int i=1;i<=cnt&&1LL*prime[i]*prime[i]<=n;i++){</pre>
42
             Int p=prime[i];
43
             l=1; while(l<=n){</pre>
                  q=n/1; 1=n/q+1;
44
45
                  if(p*p>q) break;
46
                  at(q) = (at(q/p) - (i-1));
47
             }
         }
48
49
         return g(n,0);
50
    }
    int main(){
51
52
         init();
53
         Int 1,r;cin>>1>>r;
54
         cout<<solve(r)-solve(l-1)<<endl;</pre>
55
         return 0;
56 }
```

UR 13,sanrd现在还不懂为什么....

陈牧歌@PKU的课件里面对于本题的推导有些问题.UOI官方题解提供了一个类似洲阁筛的做法.

代码抄的ppl blog

UOI提交记录

```
1
    Int q(Int n, int m){
 2
        m++; if(n<=2||prime[m]>n) return 0;
 3
        Int s=0;
        while(prime[m]*prime[m]<=n&&m<=cnt){</pre>
 4
 5
             Int p=prime[m], c=1, pc=p;
             while(pc<=n){</pre>
 6
 7
                 Int r=at(n/pc)-(m-1);
                 if(r>0) s=(g(n/pc,m)+p*(at(n/pc)-(m-1)));
 8
 9
                 else s+=g(n/pc,m);
10
                 pc=pc*p;c++;
11
             }
12
             m++;
13
14
         return s;
15 }
```

这样就清晰了, $f(p^c \cdot q) = p \quad q \in prime, q \geq p$,

对于这种东西需要**乘一个大于等于p的质数**,所以要求 $\lfloor \frac{n}{n^c} \rfloor \geq p \Rightarrow p^c \cdot p \leq n$

如果p不是次大质因子,那么递归去处理,按照 $g_{n,m}$ 的定义,是正确的.

BZOJ 4916

$$\sum_{i=1}^n \mu(i^2) \quad \sum_{i=1}^n arphi(i^2)$$

首先第一个是1...然后考虑第二个.

- 1. 这是个积性函数,质因子贡献独立.
- 2. 这个函数在质数幂出的取值可以快速求出,其中在质数出的取值是关于p的二次多项式.

套一下式子.没了.范围超小,跑得飞起.不过我们还是来考虑一下正解 $O(n^{\frac{2}{3}})$ 的做法吧.如图.没了

$$egin{aligned} let \, n &= \prod_{i=1}^m p_i^{k_i} \ arphi(n^2) &= arphi(\prod_{i=1}^m p_i^{2k_i}) = \prod_{i=1}^m arphi(p_i^{2k_i}) \ arphi(p^k) &= p^k - p^{k-1} \quad arphi(p^2k) = p^{2k} - p^{2k-1} = p^k \, arphi(p^k) \ arphi(n^2) &= n arphi(n) \ f(n) &= n arphi(n) \quad g(n) = n \ (f st g)(n) &= \sum_{d \mid n} darphi(d) \, (rac{n}{d}) = n \sum_{d \mid n} arphi(d) = n^2 \end{aligned}$$

```
#include <iostream>
    #include <cstdio>
 2
 3
   #include <cctype>
   #include <cassert>
 5
   #include <algorithm>
    const int N=200000+10;
    typedef long long Int;
 8
    const Int mod=(Int)(1e9)+7LL;
 9
    Int qpow(Int a,Int p){
10
        Int r=1; a=(a\%mod+mod)\%mod;
11
        while(p){
12
            if(p\&1) r=r*a%mod;
13
             a=a*a\%mod;
14
             p>>=1;
15
        }
16
        return r;
17
    inline Int inv(Int x){ return qpow(x,mod-2); }
18
19
    int vis[N],prime[N],cnt,phi[N];
20
    struct P{
21
        Int pre1,pre2;
22
        P() { pre1=pre2=0; }
23
        Int get() { return (pre2-pre1+mod)%mod; }
24
    }pre[N];
    void init(){
25
26
        vis[1]=1; phi[1]=1;
        for(int i=2;i<N;i++){</pre>
27
28
             pre[i]=pre[i-1];
29
             if(!vis[i]){
30
                 pre[i].pre1=(pre[i].pre1+i)%mod;
31
                 pre[i].pre2=(pre[i].pre2+1LL*i*i)%mod;
32
                 prime[++cnt]=i;
33
                 phi[i]=i-1;
34
35
             for(int j=1;j<=cnt\&i*prime[j]<N;j++){
36
                 vis[i*prime[j]]=1;
37
                 if(i%prime[j]==0){
```

```
38
                      phi[i*prime[j]]=phi[i]*prime[j];
39
                     break;
                 }
40
                 phi[i*prime[j]]=phi[i]*(prime[j]-1);
41
42
             }
43
        }
44
    }
45
    P A[N], B[N];
46
    int n;
47
    inline P_{\infty}^{k} at(int m){ return (1.0*m*m \le n)?A[m]:B[n/m]; }
    inline Int s1(Int m){
48
49
        static Int inv2=inv(2);
50
        m\%=mod;
        return m*(m+1)%mod*inv2%mod;
51
52
53
    inline Int s2(Int m){
54
        static Int inv6=inv(6);
55
        m\%=mod;
        return (2*m+1)%mod*(m+1)%mod*m%mod*inv6%mod;
56
57
58
    Int geth(int n){
59
        const P &x=at(n);
60
        return (x.pre2-x.pre1+mod)%mod;
61
    }
    Int g(int n,int m){
62
63
        m++;
64
        if(prime[m]>n) return 0;
65
        Int ret=0;
        while(1.0*prime[m]*prime[m]<=n){</pre>
66
67
             Int p=prime[m],pc=p,fpc=p*(p-1)%mod;
             while(pc<=n){</pre>
68
69
                 ret=(ret+fpc*(1+g(n/pc,m))%mod)%mod;
70
                 pc=pc*p;
71
                 fpc=fpc*p%mod*p%mod;
72
             }
73
             m++;
        }
74
75
        Int tmp=(geth(n)-geth(prime[m-1])+mod)%mod;
76
         return (ret+tmp)%mod;
77
    }
78
79
    Int solve(int _n){
80
        ::n=_n;
81
        int l=1, r=0, q=0;
        while(1 \le n){
82
83
             q=n/1; r=n/q;
84
             P &to=at(q);
85
             to.pre1=(s1(q)-1+mod)%mod;
             to.pre2=(s2(q)-1+mod)%mod;
86
             l=r+1;
87
88
        for(int i=1;i<=cnt&&</pre>
89
                 1.0*prime[i]*prime[i]<=n;i++){
90
```

```
91
              Int p=prime[i];
 92
              l=1; r=0; q=0;
 93
              while(1<=n){</pre>
 94
                  q=n/1; r=n/q;
 95
                  if(1.0*p*p>q) break;
 96
                  P &to=at(q), &from=at(q/p);
                  to.pre1-=p*(from.pre1-pre[p-1].pre1)%mod;
 97
 98
                  to.pre1=(to.pre1%mod+mod)%mod;
                  to.pre2-=p*p%mod*(from.pre2-pre[p-1].pre2)%mod;
 99
100
                  to.pre2=(to.pre2%mod+mod)%mod;
101
                  1=r+1;
102
              }
103
104
          return 1+g(n,0);
105
     }
106
107
     int main(){
108
          init();
109
          using namespace std;
          int n; cin>>n;
110
111
          cout<<1<<endl;</pre>
112
          cout<<solve(n)<<endl;</pre>
113
          return 0;
     }
114
115
```