积性函数求和

visit_world

目录

- 引入
- 科技
- 例题

一个小问题...

- 记 f(n) 表示 n 的所有约数之和
- 求 ∑{i=1}^{n} f(i)
- n <= 1e12 吧

小问题的解决

- 转换枚举对象!
- 原式要求我们对每个数 i，枚举它的所有约数 d，并求和
- 我们可以倒过来，先枚举每个 d，然后考虑它能作为多少个 i 的约数
- 那么就能得到式子：ans = ∑{d=1}^{n} d * [n / d]
- 到这一步，就能得到一个 O(sqrt(n)) 的做法了

小问题蕴含的小技巧

- 记 f 和 g 的Dirichlet 卷积为 f * g
- 刚才给我们要求那个函数实际上是 id * 1
- 而这个式子实际上有三种求法
- 普通青年： $\sum_{i=1}^{n}\sum_{ab=i} f(a)\cdot g(b)$
- 文艺青年： $\sum_{a=1}^{n} f(a)\sum_{b=1}^{\left\lfloor \frac{n}{a} \right\rfloor} g(b)$ $\sum_{b=1}^{n} g(b)\sum_{a=1}^{\left\lfloor \frac{n}{b} \right\rfloor} f(a)$
- 理解了这个就很容易理解杜教筛了

杜教筛

- 给出 n，求 $\sum(i <= n)\ \boldsymbol{\varphi}(i)$
- 不难为你们，让 n <= 1e9 好了

杜教筛

- 一些神奇的变换！
- 首先我们知道 $\varphi * 1 = id$
- id的前缀和很好算 = = 等差数列求和嘛
- 我们令 $\phi(n) = \sum(1<=i<=n)\ \varphi(i)$
- $\sum(1<=i<=n)\ \sum(d|i)\ \varphi(d) = \sum(1<=d<=n)\ \phi(n / d)$
- $\phi(n) = \sum(i<=n)\ i - \sum(2<=i<=n)\ \phi(n / i)$
- 这样一来，只要算出 O(sqrt(n)) 个较小的 ϕ，就能得到 $\phi(n)$ 辣

另一种解释

- 考虑 $\phi(n)$ 的一种组合意义：
    - 对 x<=y<=n, 且 (x,y)=1 的二元组计数
- 共有 C(n+1,2) 个可能的二元组，枚举不合法的 gcd=d, 令 $\phi(n)$ -= $\phi([n/d])$
- $\phi(n) = C(n+1,2) - \sum(2<=i<=n)\ \phi([n/i])$
- 这个思想在很多题目中都很有用

杜教筛

- 递归地计算

- 把已经算过的值用哈希表或者 map 存下来，算法的复杂度是 $O(n ^ {(3/4)})$

- 预处理出 $<= n ^ {(2/3)}$ 的值，可以进一步把复杂度优化到 $O(n ^ {(2/3)})$

杜教筛

- 总结一下：
- 如果需要求数论函数 f(n) 的前缀和，它并不好求
- 考虑找到另外一个合适的数论函数 g，使得 g 和 (f*g)的前缀和 容易计算
- 我们有
  - $\sum(i<=n) \sum(d|i) f(d) * g(i/d) = \sum(i<=n) g[i] * F(n/i)$
  - $F(n) = \sum(i) (f*g)(i) - \sum(2<=i<=n) g[i] * F(n / i)$
  - 其中 F(n) 是 f(n) 的前缀和
- 我们把已经算过的 F 值记忆化，预处理出 <= n ^ (2/3) 的 F 的值
- 算法复杂度为 O(n ^ (2/3))
- 上面这个例子中，我们选的函数是 g(n) = 1 (好多题目都用到这个)

这个 n^{2/3} 怎么来的

为了计算 $F(n)$，我们需要计算 $O(\sqrt{n})$ 个更小的 $F(i)$，设计算 $F(n)$ 的时间消耗为 $T(n)$，我们有：

$$T(n) = \sum_{i=1}^{\sqrt{n}} T(i) + \sum_{i=1}^{\sqrt{n}} T(\frac{n}{i}) + O(\sqrt{n})$$

考虑递归调用的过程，如果计算 F(n) 时用到了 F(i)，而计算 F(i) 时用到了 F(j)，那么 F(n) 的计算也用到了 F(j)。我们把计算过程记忆化，让每个 F(n) 只在第一次被调用的时候计算，那么上面的递归式只需展开一层，即：

$$T(n) = O(\sqrt{n}) + \sum_{i=1}^{\sqrt{n}} O(\sqrt{i}) + \sum_{i=1}^{\sqrt{n}} O\left(\sqrt{\frac{n}{i}}\right)$$

n^{2/3}？

大O意义下，只需要考虑最后一项，我们来用积分近似一下：

$$T(n)=O\left(\int_0^{\sqrt{n}}\sqrt{\frac{n}{x}}\mathrm{dx}\right)=O\left(n^{\frac{3}{4}}\right)$$

如果我们预处理了函数的前 $n^{\frac{2}{3}}$ 项，复杂度可以更优秀一点：

$$T(n)=O\left(\int_0^{n^{\frac{1}{3}}}\sqrt{\frac{n}{x}}\mathrm{dx}\right)=O\left(n^{\frac{2}{3}}\right)$$

杜教筛

- 练习：
  - 试求 1~n 的莫比乌斯函数之和
- 就让 n <= 1e9 好了
- 与上一例合在一起就是 BZOJ 3944

例

- 令 d(n) 表示 n 的约数个数
- 求 ∑(i<=n) d(i^2)
- n <= 10^11

例的解答

- 有一个非常经典的公式：
- $d(x*y)=\sum(a|x)\sum(b|y)\ [(a,b)=1]$
- 这个可以使用一种组合技巧证明
- 那么就有：
- $d(x^2)=\sum(a|x)\sum(b|x)[(a,b)=1]$
    - $=\sum(T|x)\sum(a|T)[(a,T/a)=1]$
    - $=\sum(T|x)\ 2^{(T\text{的质因子种数})}$

例的解答

- 如果我们定义 g(x) = 2^(x的不同质因子种数)
- 那么有 f = g * 1
- 我们又有 g(x)=∑(d|x) (mu(d))^2
- 定义 h(x) = mu(x)^2
- 那么有 g = h * 1
- 所有 f = h * 1 * 1

例的解答

- 如果能快速求 g=h * 1 的前缀和，就能求 f 的前缀和
- 如果能快速求 h 的前缀和，就能求 g=h * 1 的前缀和
- h 的前缀和可以用容斥来求，复杂度 O(sqrt(n))
- 因为有 [f / xy] = [[f / x] / y]，所以：
  - 我们只需要求 O(sqrt(n)) 个不同的 g 的前缀和
  - 为了求这 O(sqrt(n)) 个 g 的前缀和，我们用到的 h 的前缀和总共有 O(sqrt(n)) 个
- 所以：在分析复杂度的时候仍然只需展开一层，复杂度 O(n ^ {2/3})

一个有趣的事

- 如果我们在上文中的那个函数上再卷积一个恒等函数，计为 $w = f * 1$
- $w(x) = (d(x))^2$
- 也就是说：$(d(x))^2 = \sum_{k|x} d(k^2)$
- 感兴趣的同学们可以自己验证一下
- 当然这个你也可以用上文介绍的技巧做啦，类似地，如果有函数 $f = g * 1 * 1 * 1\ldots$，只要这里的卷积层数是一个常数，那么使用记忆化递推，复杂度都能保证是 $O(n^{2/3})$
- 不过常数会增大多少就不好说了）

小练习

- $\sum(i<=n)\sum(j<=n)\ gcd(i, j)$
- $n<=10^{10}$

小练习

- ∑(k<=n) phi(k) * [n/k]^2
- 剩下的事情非常显然了

大练习

- $\sum(i<=n)\sum(j<=n)$ lcm(i, j)
- n<=10^10

大练习

- 设 A(n)=∑(i<=n) lcm(i, n)
- ans = 2 * ∑(i<=n) A(i) - ∑(i<=n) i
- 那么 A(n) 怎么算呢?
- 枚举 gcd(i, n) 试试吧!

大练习

同样的，考虑计算 $\sum_{i=1}^{n}[i,n]$，记为 $A(n)$，同样的枚举 (i,n)：

$$A(n)=n\sum_{d|n}\sum_{(i,\frac{n}{d})=1}i$$

注意到 $n>1$ 时，"小于n且与n互质的数" 的和为 $\frac{n\times\varphi(n)}{2}$，对于 $n=1$ 的情况特殊讨论，上式又可以简化为：

$$A(n)=\frac{n}{2}\sum_{d|n}d\times\varphi(d)+\frac{n}{2}$$

大练习

记 $f(n) = n\sum_{d|n} d \times \varphi(d)$，$f(n)$ 的前缀和为 $F(n)$，考虑如何计算 $F(n)$

把 n "分配" 到和式中去，我们有：

$$f(n) = \sum_{ab=n} a^2\varphi(a) \times b$$

注意到这是 $id^2 \cdot \varphi$ 和 id 的 Dirichlet 卷积，我们进一步地推导：

注意到 $(id^2 \cdot \varphi) \circ id^2 = id^3$，即：

$$\begin{aligned}\sum_{ab=n} a^2\varphi(a) \times b^2 &= n^2 \sum_{a|n} \varphi(a) \\ &= n^3\end{aligned}$$

大练习

那么我们有：

$$id^2 \circ (id^2 \cdot \varphi) \circ id = id^3 \circ id$$

同样地，我们记 $id^3 \circ id$ 的前缀和为 $H(n)$，那么：

$$H(n) = \sum_{i=1} i \sum_{j=1}^{\left\lfloor \frac{n}{d} \right\rfloor} j^3$$

这个也可以分块计算，接下来的处理思路就很经典了。

NOI2016 循环之美

- 这里给出一个简化版题意
- $\sum(x<=n)\sum(y<=m)$ $[(x,y)=1][(y,k)=1]$
- $n,m<=1e9$, $k<=2000$

循环之美

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

$$=\sum_{d=1}^{n}\mu(d)\cdot\left\lfloor\frac{n}{d}\right\rfloor\sum_{d|j}[(k,j)=1]$$

$$=\sum_{d=1}^{n}[(d,k)=1]\mu(d)\left\lfloor\frac{n}{d}\right\rfloor\sum_{j=1}^{\left\lfloor\frac{m}{d}\right\rfloor}[(j,k)=1]$$

（化简到这一步还想反演的同学基本上救不回来了）

循环之美

注意到这里的 k 是固定的，定义：

$$S(n)=\sum_{i=1}^{n}[(i,k)=1]$$

$$T(n)=\sum_{i=1}^{n}[(i,k)=1]\mu(i)$$

那么原式子可以写为：

$$\sum_{d=1}^{n}\left\lfloor\frac{n}{d}\right\rfloor\cdot(T(d)-T(d-1))\cdot S(\left\lfloor\frac{m}{d}\right\rfloor)$$

循环之美

不难看出，我们只需要对 S 和 T 各计算 $O(\sqrt{n}+\sqrt{m})$ 个不同的值，然后这个式子就能算了。

S 和 T 可以通过递推得出。

不难发现，问题只和 k 中含有的质因子有关，取出 k 中的所有质因子，设为 $p_1, p_2, \ldots, p_l$。

定义：

$$S(n,s)=\sum_{i=1}^{n}[(i,p_1p_2\ldots p_s)=1]1$$

$$T(n,s)=\sum_{i=1}^{n}[(i,p_1p_2\ldots p_s)=1]\mu(i)$$

循环之美

接着，使用一些计数技巧得到递推关系：

$$S(n, s) = S(n, s-1) - S\left(\left\lfloor \frac{n}{p_s} \right\rfloor, s-1\right)$$

$$T(n, s) = T(n, s-1) + T\left(\left\lfloor \frac{n}{p_s} \right\rfloor, s\right)$$

根据上式递推即可，总复杂度 $O((\sqrt{n} + \sqrt{m}) \cdot \log k)$

循环之美

- 推荐一道类似地用到了递推技巧的题目
- BZOJ3512：DZY Loves Math IV

约数之和

- 令 $f(x)$ 表示 x 的约数和
- $\sum(i<=n)\sum(j<=n)\ f(i * j)$
- $n <= 1e9$

约数之和

$$f(xy) = \sum_{a|x} \sum_{b|y} a \cdot \frac{y}{b} \cdot [(a,b)=1]$$

可以得到：

$$\sum_{x=1}^{n} \sum_{y=1}^{n} \sum_{a|x} \sum_{b|y} a \cdot \frac{y}{b} \cdot [(a,b)=1]$$

约数之和

定义 $S(n)=\sum_{i=1}^{n} i$，进行和式变换：

$$=\sum_{a=1}^{n}\sum_{b=1}^{n}\sum_{a|x}\sum_{b|y} a\cdot\frac{y}{b}\cdot[(a,b)=1]$$

$$=\sum_{a=1}^{n}\sum_{b=1}^{n} a\cdot\left\lfloor\frac{n}{a}\right\rfloor\cdot S\left(\left\lfloor\frac{n}{b}\right\rfloor\right)\cdot[(a,b)=1]$$

$$=\sum_{d=1}^{n}\mu(d)\sum_{d|a} a\cdot\left\lfloor\frac{n}{a}\right\rfloor\sum_{d|b} S\left(\left\lfloor\frac{n}{b}\right\rfloor\right)$$

$$=\sum_{d=1}^{n}\mu(d)\cdot d\sum_{a=1}^{\left\lfloor\frac{n}{d}\right\rfloor} a\cdot\left\lfloor\frac{n}{ad}\right\rfloor\sum_{b=1}^{\left\lfloor\frac{n}{d}\right\rfloor} S\left(\left\lfloor\frac{n}{bd}\right\rfloor\right)$$

约数之和

不难发现，a 和 b 两部分的计算本质上都是 $id \circ 1$ 的前缀和，对于 $\mu(d) \cdot d$ 的前缀和，我们也有一个好办法处理 $--$ 注意到 $(mu \cdot id) \circ id = \epsilon$，使用杜教筛计算即可。

总时间复杂度 $O(n^{\frac{2}{3}})$

51nod 1222

- 给定 L，R，求有多少不同的二元组 (x, y)，满足：
  - x <= y
  - $\mathrm{lcm}(x, y) \in [L, R]$
- L, R <= 10^11

51nod 1222

- 首先区间计数转化为前缀相减
- 反演一下，得到一个计算 [1, n] 的答案的公式：

$$\sum_{d=1}^{\lfloor\sqrt{n}\rfloor}\sum_{a}\sum_{b}\sum_{r}[a \le b][abr \le \left\lfloor \frac{n}{d^2} \right\rfloor]$$

51nod 1222

- a <= b 的限制很讨厌，可以通过一些小手段把它抹掉
- 最后，我们需要解决这样一个问题：
    - 统计 xyz <= n 的三元组 (x, y, z) 的数量
    - 我们枚举三个数都不相同、有两个数相同、三个数都相同的情况，然后配上系数即可统计贡献
    - 其中最棘手的部分是 x < y < z 的计算
- 我们枚举不超过 n^{1/3} 的 x，然后枚举大于 x 且不超过 sqrt(n / x) 的 y，计算有多少合法的 z 即可
- 复杂度分析与杜教筛类似，为 n^{2/3}

两个小技巧?

若 f 不好求但是 $f \circ 1$ 的前缀和有办法快速求解，那么：

- 对 $S(n)=\sum_{i=1}^{n} f(i)\cdot g(i)$，若 g 是完全积性函数，那么 $(f\cdot g)\circ g=(f\circ 1)\cdot g$
- 对于 $S(n)=\sum_{i=1}^{n}(f\circ g)(i)$，给它凑上一个 1，有：

$$S(n)=\sum_{i=1}^{n} g(i)\sum_{ij\le n}(f\circ 1)(j)-\sum_{i=2}^{n} S(\left\lfloor \frac{n}{i}\right\rfloor)$$

素数统计

- 给定 n，统计不大于 n 的所有素数的：
  - 个数
  - 求和
  - 平方和
  - …….
- n <= 10^11

素数统计

- 这里以素数个数统计为例
- 我们考虑一种递推 / DP / 筛法 (从哪个角度理解这种算法都是合适的)
- 首先对于不大于 sqrt(n) 的素数可以欧拉筛

递推

- 设不大于 sqrt(n) 的素数有 m 个
- 令 p[i] 表示第 i 个素数
- 我们写一个 dp[i][j]，表示 [1, j] 中有多少数不是 p[1 ~ i] 的倍数
  - i = 0 的情况是 trivial 的
  - i > 0 时，dp[i][j] = dp[i - 1][j] - dp[i - 1][p / p[j]]
- dp[m][n] 即为所求

递推

- 这个递推的第二维有 sqrt(n) 种取值，恰好是 [n / i] 的所有可能的取值
- 朴素实现这个递推，对每个第二维 > sqrt(n) 的状态，需要枚举所有小素数进行转移，小素数可以认为有 sqrt(n) / log(n) 个，总复杂度为 O(n / sqrt(n))
- …….优化?

递推的优化

- 注意到，p[i+1] > j 的时候，一定有 dp[i][j] = 1
- 所以，对 p[i]^2 > j >= p[i] 的 j，我们有
  - [j / p[i]] < p[i]
- 可以得到：
  - dp[i][j]
  - = dp[i - 1][j] - dp[i - 1][j / p[i]]
  - = dp[i - 1][j] - 1

递推的优化

- 因此，j < p[i]^2 时的转移的第二项系数是固定的，我们可以先不显式计算。

- 对每个 j 记录上次显式转移时 i 的值，询问的时候把尚未转移的质数的贡献统一计算即可

- 复杂度?

复杂度

经过这个优化，对每个 $\left\lfloor \frac{n}{x} \right\rfloor$ 只需考虑不超过 $\sqrt{\left\lfloor \frac{n}{x} \right\rfloor}$ 的质数的转移，复杂度可以估计为：

$$\sum_{i=1}^{\sqrt{n}} O\left(\frac{\sqrt{i}}{\log i}\right) + O\left(\frac{\sqrt{\frac{n}{i}}}{\log \frac{n}{i}}\right)$$

大 O 意义下，只需考虑后半部分的贡献，可以近似地认为复杂度为：

$$O\left(\frac{\int_0^{\sqrt{n}} \sqrt{\frac{n}{x}} \mathrm{d}x}{\log n}\right) = O\left(\frac{n^{\frac{3}{4}}}{\log n}\right)$$

素数统计

- 要计算的函数是素数的 k 次幂和（或者其他方便计算前缀和的完全积性函数）时，上述做法都是可以用的，只需要把转移第二项的系数改一下就可以辣
- 这是洲阁筛的前半部分，后半部分原理相似，大家可以自行学习一下QWQ

小技巧

- 注意到在杜教筛 / 州阁筛的技巧中
- 第二维要么 $x \le \mathrm{sqrt}(n)$，要么 $[n / x] \le \mathrm{sqrt}(n)$
- 根据这个可以建立一种索引来访问数组，可以避免写Hash或者map，常数(或许)会小一些。

更多资料请参阅

- 唐老师(tangjz)的博客
- 吉司机(jiry_2)的博客
- 51nod 中的数论题

谢谢大家

- email : 745350128@qq.com
- QQ : 745350128