# 积性函数前缀和

吴一岐

2020 年 5 月 9 日

# 1 积性函数前缀和

## 1.1 问题描述

对于积性函数 $f(n)$, 求

$$S(n)=\sum_{i=1}^{n}f(i)(1\le n\le 10^{10})$$

## 1.2 解法

我们发现即便 $f(n)$ 可以 $O(1)$ 计算，这个前缀和也是比较难求的
解法是先找到一个合适的数论函数 $g(n)$ (以后会讲如何去构造这个 $g(n)$)
现在尝试着计算 $g\times f$ 的前缀和

$$\begin{aligned}\sum_{i=1}^{n}(f\times g)(i)&=\sum_{i=1}^{n}\sum_{d|i}g(d)f(\tfrac{i}{d})\\&=\sum_{d=1}^{n}g(d)\sum_{d|i}f(\tfrac{i}{d})\\&=\sum_{d=1}^{n}g(d)\sum_{i=1}^{\lfloor\frac{n}{d}\rfloor}f(i)\\&=\sum_{i=1}^{n}g(i)S(\lfloor\tfrac{n}{i}\rfloor)\end{aligned}$$

所以我们可以得到

$$g(1)S(n)=\sum_{i=1}^{n}(f\times g)(i)-\sum_{i=2}^{n}g(i)S(\lfloor\frac{n}{i}\rfloor)$$

由整除分块我们知道 $\lfloor\frac{n}{i}\rfloor$ 一共有 $\sqrt{n}$ 个不同的值，因此一个 $S(n)$ 的值依赖于 $\sqrt{n}$ 个 $S(i)$ 的值, 因此可以采用记忆化搜索，计算一个 $S(n)$ 的复杂度是 $O(\sqrt{n})$

现在我们来计算一下上面式子的复杂度，假设 $(f\times g)(i)$ 和 $g(i)$ 均可以快速求和，那么我们主要计算右半部分的复杂度, 对于 $\lfloor\frac{n}{i}\rfloor$, 根据 $i>\sqrt{n}$ 与 $i\le\sqrt{n}$ 可以分为两部分计算, 计算量为

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

为了更方便的计算复杂度，我们可以假设 $k$ 是连续的, 利用积分可以得到复杂度为

$$O\left(\int_0^{\sqrt{n}}(\sqrt{k}+\sqrt{\frac{n}{k}})dk\right)=O(n^{3/4})$$

这个复杂度是可以继续优化的，因为我们可以用空间换时间，利用线性筛筛选出前面一部分 $S(i)$ 的值，那么，筛选多少最优呢?

忘掉前面的分析，我们重新回到求解 $\sum\limits_{i=1}^{n} S(\lfloor\frac{n}{i}\rfloor)$

假设我们利用线性筛预处理前 $x$ 个前缀和，那么当 $i>\frac{n}{x}$ 的时候，$\frac{n}{i}\leq x$，查询的复杂度为 $O(1)$, 当 $i<\frac{n}{x}$ 的时候 $\frac{n}{i}>x$，需要进行递归计算，所以总计算量大概为如下式子

$$x+\sum_{k=1}^{\lfloor\frac{n}{x}\rfloor}\sqrt{\frac{n}{k}}=\left(x+\int_0^{n/x}\sqrt{\frac{n}{k}}dk\right)=\left(x+\frac{2n}{\sqrt{x}}\right)$$

对这个东西求导后的结果为

$1-nx^{-\frac{3}{2}}$

所以当 $x=n^{\frac{2}{3}}$ 时取到最小值

最后总复杂度为 $O(n^{\frac{2}{3}})$

我们发现最终只有 $n^{\frac{1}{3}}$ 个不同的 $S(x)(x>n^{\frac{2}{3}})$ 需要求, 因为 $\frac{n}{i}$ 大于 $n^{\frac{2}{3}}$ 的话, i 就小于 $n^{\frac{1}{3}}$

所以从此处也可以看出如果采用记忆化搜索，假设查询 $q$ 次前缀和，总复杂度为 $O(q+n^{\frac{2}{3}})$, 也就是花在查询上的复杂度始终是均摊 $O(n^{\frac{2}{3}})$

# 2 第二部分例题

## 2.1 例题一

**莫比乌斯函数前缀和**

求

$$S(n)=\sum_{i=1}^{n}\mu(i)$$

$n\leq 10^{10}$

### 2.1.1 问题求解

这个题我们先找到一个合适的 $g(n)=I(n)$ 即恒等函数

$I(n) = 1$
然后我们计算 $I$ 与 $\mu$ 函数的狄利克雷卷积的前缀和

$$\sum_{i=1}^{n}(\mu \times I)(i) = \sum_{i=1}^{n} I(i)S(\lfloor\frac{n}{i}\rfloor)$$

$$I(1)S(n) = \sum_{i=1}^{n}(\mu \times I)(i) - \sum_{i=2}^{n} I(i)S(\lfloor\frac{n}{i}\rfloor)$$

由于

$$(\mu \times I)(n) = \sum_{d|n}\mu(d) \times I(\frac{n}{d}) = \sum_{d|n}\mu(d) = e(n)$$

$e(n) = [n == 1]$

所以

$$S(n) = 1 - \sum_{i=2}^{n} S(\lfloor\frac{n}{i}\rfloor)$$

### 2.1.2 代码实现

首先我们利用线性筛先筛前 $n^{\frac{2}{3}}$ 的 $\mu$ 函数

然后利用最终得出的 $S(n)$ 的公式进行记忆化搜索，一旦碰见 $n^{\frac{2}{3}}$ 以内的查询就直接返回结果就行啦.

注意记忆化搜索可以使用 map 或者手写哈希表，也可以采用下面的优化技巧，详情见代码.

莫比乌斯函数前缀和

```
#include <bits/stdc++.h>
using namespace std;
const int N = 4641590;
```

```
int p[N], pn, mu[N];
bool flag[N];
void init() {
  mu[1] = 1;
  for (int i = 2; i < N; i++) {
    if (!flag[i]) {
      p[pn++] = i;
      mu[i] = -1;
    }
    for (int j = 0; j < pn; j++) {
      if (i * p[j] >= N) {
        break;
      }
      flag[i * p[j]] = 1;
      if (i % p[j] == 0) {
        mu[i * p[j]] = 0;
        break;
      } else {
        mu[i * p[j]] = -mu[i];
      }
    }
  }
  for (int i = 2; i < N; i++) {
    mu[i] += mu[i - 1];
  }
}

long long mp[2333];
bool vis[2333];

//用一个now记录现在除过哪些i了
//因为n/(ab) = n/a/b, 所以不会导致不同的x对应同一个now的情况
long long solve (long long x, long long now) {
  if (x < N) {
    return mu[x];
  }
```

```
  //如果x > n^{2/3}, 那么now < n^{1/3}
  if (vis[now]) {
    return mp[now];
  }

  long long ret = 1;
  for (long long i = 2, j; i <= x; i = j + 1) {
    j = x / (x / i);
    ret -= (j - i + 1) * solve(x / i, now * i);
  }
  vis[now] = true;
  mp[now] = ret;
  return ret;
}

int main() {
  init();
  long long a, b;
  cin >> a >> b;
  long long t1 = solve(b, 1);
  //vis必须清空, 因为对应的n不同,
  //因此如果多组数据, 采用哈希表或者map的形式存储会更合适
  //同时存储x 到 ret的映射
  memset(vis, false, sizeof(vis));
  long long t2 = solve(a - 1, 1);
  cout << t1 - t2 << endl;
  return 0;
}
```

## 2.2 例题二

**莫比乌斯函数前缀和**

求

$$\sum_{i=1}^{n} \varphi(i)$$

$1 \le n \le 10^{10}$

### 2.2.1 问题求解

同上题，选取 $g$ 函数为 $I(n) = 1$ 这个恒等函数

最终可以得到

$$S(n) = n * (n + 1)/2 - \sum_{i=2}^{n} S(\lfloor \frac{n}{i} \rfloor)$$

## 2.3 例题三

**莫比乌斯函数前缀和**

有一个函数 $f(x)$ 满足 $N^2 - 3N + 2 = \sum_{d|N} f(d)$

求 $\sum_{i=1}^{N} f(i) \pmod{10^9 + 7}$

$T$ 组数据，$T \le 500, N \le 10^9$

只有 5 组 $N < 10^6$

### 2.3.1 问题求解

假设

$$h(n) = n^2 - 3n + 2 = \sum_{d|n} f(n)$$

$$f(n) = \sum_{d|n} h(d)\mu(\frac{n}{d})$$

设 $g(n) = I(n) = 1$, $S(n) = \sum_{i=1}^{n} f(i)$

推导可得

$$g(1)S(n) = \sum_{i=1}^{n}(f \times g)(i) - \sum_{i=2}^{n} g(i)S(\lfloor\frac{n}{i}\rfloor)$$

$$S(n) = \sum_{i=1}^{n} h(i) - \sum_{i=2}^{n} g(i)S(\lfloor\frac{n}{i}\rfloor)$$

现在就是直接上套路了

$h$ 函数的前缀和为 $n(n+1)(n+2)/6 - 3n(n+1)/2 + 2 = n(n-1)(n-2)/3$