具体数学中级班第二周参考答案

517

2020 年 5 月 10 日

1 序列变换

1.1 题意

给你两个序列 a, b, 求有多少对的 (x, y) 满足

1: $gcd(x, y) = 1$

2: $a_{b_x} = b_{a_y}$

$1 \le n \le 10^5, 1 \le a_i, b_i \le n$

1.2 题解

首先，假如不考虑第一个条件，就是先枚举 x，用个桶记录一下 $cnt[a[b[x]]]$, 再枚举 y，对于每个 y，统计 $cnt[b[a[y]]]$ 的出现次数

然后考虑第一个条件，我们可以设 $f(n)$ 表示 $gcd(x, y) \ge n$ 的满足条件的对数，设 $g(n)$ 表示 $gcd(x, y) = n$ 的满足条件的对数，先求出 $f(n)$, 然后反演出 $g(n)$

序列变换

```
#include <bits/stdc++.h>
using namespace std;

const int N = 100010;

long long f[N];
int cnt[N], a[N], b[N], mu[N];
bool flag[N];
int pn, p[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]] = true;
      if (i % p[j] == 0) {
        mu[i * p[j]] = 0;
        break;
      } else {
        mu[i * p[j]] = -mu[i];
      }
    }
  }
}

int main() {
  init ();
  int n;
  scanf("%d", &n);
  for (int i = 1; i <= n; i++) {
    scanf("%d", &a[i]);
  }

  for (int i = 1; i <= n; i++) {
    scanf("%d", &b[i]);
  }
```

```
  for (int d = 1; d <= n; d++) {
    for (int x = d; x <= n; x += d) {
      cnt[a[b[x]]]++;
    }

    for (int y = d; y <= n; y += d) {
      f[d] += cnt[b[a[y]]];
    }

    for (int x = d; x <= n; x += d) {
      cnt[a[b[x]]]--;
    }
  }

  long long ret = 0;
  for (int i = 1; i <= n; i++) {
    ret += f[i] * mu[i];
  }
  printf("%lld\n", ret);
  return 0;
}
```


2 数表

2.1 题意

Q 组数据, 每组数据给你三个整数 n, m, a, 求 $\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m} f(gcd(i,j))[f(gcd(i,j)) \le a]$

$f(d)$ 表示 d 的约数之和,$1 \le n, m \le 10^5, 1 \le Q \le 20000, a \le 10^9$

2.2 题解

假设 $n \le m$, 先不考虑 a 的影响, 原式 $= \sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m} f(gcd(i,j))$

$= \sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m}\sum\limits_{d=1}^{n} f(d)[gcd(i,j) == d]$ (枚举 gcd)

$$= \sum_{d=1}^{n} f(d) \sum_{i=1}^{\lfloor \frac{n}{d} \rfloor} \sum_{j=1}^{\lfloor \frac{m}{d} \rfloor} [gcd(i,j) == 1]$$
$$= \sum_{d=1}^{n} f(d) \sum_{t=1}^{\lfloor \frac{n}{d} \rfloor} \mu(t) \lfloor \frac{n}{dt} \rfloor \lfloor \frac{m}{dt} \rfloor$$
$$= \sum_{Q=1}^{n} \lfloor \frac{n}{Q} \rfloor \lfloor \frac{m}{Q} \rfloor \sum_{d|Q} f(d) \mu(\frac{Q}{d})$$

再考虑 a 的影响, 很自然想到离线算法, 具体策略为:

将 a 升序排序, 将 $f(d)$ 按值升序排序, 因此我们预处理的时候需要预处理出 f 和 μ , 然后随着 a 的增大动态的去枚举小于等于 a 的那些 f 值, 每次需要枚举所有的 $f \times u$ 插入树状数组即可.

代码设计:

- 预处理部分

预处理 μ 函数, 预处理 f 函数,f 函数应该设计成一个结构体, id 表示下标,$value$ 表示函数值,$value$ 可以 $nlogn$ 预处理, 枚举每一个数，再枚举所有的倍数, 然后将 f 按照 $value$ 从小到大排序

数据结构

```
struct data{
 int id;
 int value;
}f[100010];
```


- 回答询问部分:

对于当前的 n, m, a, 前半部分可以 $\sqrt{n}$ 利用整除分块枚举, 后半部分看成一个整体,$g(Q) = \sum_{d|Q} f(d)\mu(\frac{Q}{d})$, 相当于是求 g 函数的区间和, 利用类似 two-pointer 技巧找到所有小于等于当的 a 的 $f(d)$, 枚举所有的 d 的倍数 t, 在树状数组的 dt (相当于 Q) 位置插入 $f(d) \times \mu(t)$ 这个值, 最后利用树状数组区间求和回答当前询问

单次查询复杂度为 $O(\sqrt{n} \times log(n))$

3 DZY Loves Math VI

3.1 题意

给定正整数 n, m, 求
$$\sum_{i=1}^{n} \sum_{j=1}^{m} lcm(i,j)^{gcd(i,j)}$$

对 $1e9+7$ 取模

$1 \le n, m \le 500000$

3.2 题解

假设 $n \le m$

$\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m} lcm(i,j)^{gcd(i,j)} = \sum\limits_{d=1}^{n}\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m}[gcd(i,j)==d](\frac{ij}{d})^d$

$= \sum\limits_{d=1}^{n}\sum\limits_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum\limits_{i=1}^{\lfloor\frac{m}{d}\rfloor}[gcd(i,j)==1](\frac{idjd}{d})^d$

$= \sum\limits_{d=1}^{n} d^d \sum\limits_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum\limits_{i=1}^{\lfloor\frac{m}{d}\rfloor}\sum\limits_{k|i,k|j}\mu(k)(ij)^d$

$= \sum\limits_{d=1}^{n} d^d \sum\limits_{k=1}^{\lfloor\frac{n}{d}\rfloor}\mu(k)k^{2d}\sum\limits_{i=1}^{\lfloor\frac{n}{kd}\rfloor} i^d \sum\limits_{j=1}^{\lfloor\frac{m}{kd}\rfloor} j^d$

化简到这一步，其实就可以暴力算了，枚举 d，再枚举 k，后面是两个前缀和，可以预处理之后 $O(1)$ 得到，总复杂度为调和级数 $O(nlogn)$

4 DZY Loves Math

4.1 题意

多组数据

定义 $f(n)$ 为 n 所含质因子的最大幂指数, 给定正整数 n, m, 求 $\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m} f(gcd(i,j))$

$1 \le n, m \le 1e7$

4.2 题解

假设 $n \le m$

$\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m} f(gcd(i,j)) = \sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m}\sum\limits_{d=1}^{n} f(d)[gcd(i,j)==d]$ (枚举最大公约数)

$= \sum\limits_{d=1}^{n} f(d)\sum\limits_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum\limits_{j=1}^{\lfloor\frac{m}{d}\rfloor}[gcd(i,j)==1]$ (交换枚举顺序)

$= \sum\limits_{d=1}^{n} f(d)\sum\limits_{t=1}^{\lfloor\frac{n}{d}\rfloor}\mu(t)\lfloor\frac{n}{dt}\rfloor\lfloor\frac{m}{dt}\rfloor$

$= \sum\limits_{q=1}^{n}\lfloor\frac{n}{q}\rfloor\lfloor\frac{m}{q}\rfloor\sum\limits_{t|q}\mu(t)f(\frac{q}{t})$

然后我们分析一下后面这个卷积, 由于 f 不像是积性的, 我们可以找找规律

我们可以假设

$q = p_1^{a_1} \cdot p_2^{a_2} ... p_k^{a_k}$

$g(q) = \sum\limits_{t|q} \mu(t) f(\frac{q}{t})$

由于 t 的质因子中幂次超过 1 的时候 $\mu(t)$ 的值都为 0, 所以我们只需考虑

$t = p_1^{b_1} \cdot p_2^{b_2} ... p_k^{b_k}$ $(0 \leq b_i \leq 1)$

的情况, 也就是说一共有 2^k 种情况, 每种质因子要么选，要么不选

此时分两种情况

- 1: 存在 $a_i \neq a_j$

假设 a_i 里面有多个最大值 $a_{i1}, a_{i_2} ... a_{im}$, 令这些幂次对应的质因子集合为 A 集合

其他的 a 对应的质因子集合构成 B 集合, A 和 B 集合并起来构成了所有的质因子集合

$f(\frac{q}{t})$ 的取值被 t 中 A 集合的选取方案决定, 比如如果 t 没有选择全部 A 集合中的质因子，那么 $f(\frac{q}{t})$ 就一定能取到 a 中的最大值

但是由于 B 集合中选奇数与选偶数个是相等的，所以对于 t 中任意一种 A 集合的选法，都会对应 B 集合中相同数量的选奇数个与选偶数个的选法，最终全部抵消 $g(q) = 0$

- 2: 所有的 a 值相等

这个时候只有所有的 b_i 都取 1 的时候才会使得 $f(\frac{q}{t})$ 的取值减 1, 其他所有的情况 $f(\frac{q}{t}) = f(q)$

反过来看, 如果所有的 f 值取值都一样, 那么答案还是 0, 现在相当于有一个 f 的取值减了 1, 那么此时要看 $\mu(t)$ 是-1 还是 1, 因此当所有的 a 值相等的时候 $g(q) = (-1)^{k+1}$ (与 $\mu(t)$ 的取值相反)

所以求 g 函数可以在线性筛的时候维护一个数最小质因子的幂次，以及所有最小质因子的乘积, 具体细节见代码.

DZY Loves Math

```
#include <bits/stdc++.h>
using namespace std;

const int N = 10000010;

int p[1000010], pn, g[N];
int minimum_pow[N], minimum_prod[N];
bool flag[N];
```

```
void init() {
  pn=0;
  for(int i = 2; i < N; i++) {
    if(!flag[i]) {
      p[pn++] = i;
      minimum_pow[i] = 1;
      minimum_prod[i] = i;
      g[i] = 1;
    }
    for(int j = 0; j < pn && i*p[j] < N; j++) {
      flag[i * p[j]] = true;
      if (i % p[j] == 0) {
        minimum_pow[i * p[j]] = minimum_pow[i] + 1;
        minimum_prod[i * p[j]] = minimum_prod[i] * p[j];
        int tmp = i / minimum_prod[i];
        if (tmp == 1) {
          g[i * p[j]] = 1;
        } else {
          if(minimum_pow[i * p[j]] == minimum_pow[tmp]) {
            g[i * p[j]] = -g[tmp];
          } else {
            g[i * p[j]] = 0;
          }
        }
        break;
      }
      else {
        minimum_pow[i*p[j]] = 1;
        minimum_prod[i*p[j]] = p[j];
        if (minimum_pow[i] == 1) {
          g[i * p[j]] = -g[i];
        } else {
          g[i * p[j]] = 0;
        }
      }
```

```
    }
  }

  for (int i = 1; i < N; i++) {
    g[i] += g[i - 1];
  }
}

long long query(int n, int m) {
  if (n > m) swap(n, m);
  long long ret = 0;
  for (int i = 1, j; i <= n; i = j + 1) {
    j = min(n / (n / i), m / (m / i));
    ret += 1LL * (g[j] - g[i - 1]) * (n / i) * (m / i);
  }
  return ret;
}

int main() {
  init();
  int t, n, m;
  scanf("%d", &t);
  while (t--) {
    scanf("%d%d", &n, &m);
    printf("%lld\n", query(n, m));
  }
  return 0;
}
```

5 数字表格

5.1 题意

T 组数据，求 $\prod_{i=1}^{n}\prod_{j=1}^{m} f(gcd(i,j))$,$f(x)$ 表示 fib 数列的第 x 项，前两项是 0 和 1，答案对 1e9+7 取模

$1 \le n, m \le 10^6, T \le 1000$

5.2 题解

连乘操作是本题的亮点，不过通过枚举 gcd 之后可以变成加法

假设 $n \le m$

$\prod_{i=1}^{n}\prod_{j=1}^{m} f(gcd(i,j)) = \prod_{d=1}^{n} f(d)^{\sum_{i=1}^{n}\sum_{j=1}^{m}[gcd(i,j)==d]}$

此时我们发现右上角是一个我们熟悉的套路，化简之后可得

$= \prod_{d=1}^{n} f(d)^{\sum_{k=1}^{\lfloor \frac{n}{d} \rfloor} \mu(k)\cdot\lfloor\frac{n}{kd}\rfloor\cdot\lfloor\frac{m}{kd}\rfloor}$

$= \prod_{q=1}^{n}\prod_{d|q} f(d)^{\lfloor\frac{n}{q}\rfloor\lfloor\frac{m}{q}\rfloor\mu(\frac{q}{d})}$

$= \prod_{q=1}^{n} \cdot \left(\prod_{d|q} f(d)^{\mu(\frac{q}{d})}\right)^{\lfloor\frac{n}{q}\rfloor\lfloor\frac{m}{q}\rfloor}$

我们可以令 $g(q) = \left(\prod_{d|q} f(d)^{\mu(\frac{q}{d})}\right)$

可以预处理出 g 函数，顺便预处理出前缀积，最后整除分块即可，注意计算过程中逆元的使用

数字表格

```
#include <bits/stdc++.h>
using namespace std;

const int N = 1000010;
const int md = 1000000007;

int p[N], pn, mu[N];
bool flag[N];
int f[N], g[N];
```

```
int pow_mod(int a, int b, int c) {
  int ret = 1;
  while (b) {
    if (b & 1) {
      ret = 1LL * ret * a % c;
    }
    b >>= 1;
    a = 1LL * a * a % c;
  }
  return ret;
}

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

```
  for (int i = 1; i < N; i++) {
    for (int j = 1; i * j < N; j++) {
      if (mu[j] == 1) {
        g[i * j] = 1LL * g[i * j] * f[i] % md;
      } else if (mu[j] == -1) {
        g[i * j] = 1LL * g[i * j] * pow_mod(f[i], md - 2, md) % md;
      }
    }
  }

  for (int i = 2; i < N; i++) {
    g[i] = 1LL * g[i] * g[i - 1] % md;
 //   if(i < 10)cout << g[i] << endl;
  }
}

int calc(int n, int m) {
  if (n > m) swap(n, m);
  int ret = 1;
  for (int i = 1, j; i <= n; i = j + 1) {
    j = min(n / (n / i), m / (m / i));
    int k = 1LL * (n / i) * (m / i) % (md - 1);
    int tmp = 1LL * g[j] * pow_mod(g[i - 1], md - 2, md) % md;
    ret = 1LL * ret * pow_mod(tmp, k, md) % md;
  }
  return ret;
}

int main() {
  init();
  int T, n, m;
  scanf("%d", &T);
  while (T--) {
    scanf("%d%d", &n, &m);
    printf("%d\n", calc(n, m));
```

```
  }
  return 0;
}
```

6 Mophues

6.1 题意

给出 n, m, p, 求有多少对 a, b 满足 gcd(a, b) 的素因子个数 <=p, (其中 1<=a<=n, 1<=b<=m)

5000 组数据，$n, m, p \leq 5e5$

6.2 题解

令 $f(n)$ 表示 gcd 为 n 的对数，$F(n)$ 表示 gcd 为 n 的倍数的对数

令 k 表示质因子个数 $\leq p$ 的数的集合

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

$=\sum\limits_{k}f(k)$

$=\sum\limits_{k}\sum\limits_{k|d}\mu(\frac{d}{k})F(d)$

$=\sum\limits_{k}\sum\limits_{k|d}\mu(\frac{d}{k})\lfloor\frac{n}{d}\rfloor\lfloor\frac{m}{d}\rfloor$

$=\sum\limits_{d}\lfloor\frac{n}{d}\rfloor\lfloor\frac{m}{d}\rfloor\sum\limits_{k|d}\mu(\frac{d}{k})$

$\sum\limits_{k|d}\mu(\frac{d}{k})$ 这个东西我们可以预处理，枚举 k，再枚举 k 的倍数即可，这里再开一维表示素因子的个数，预处理前缀和，这样就可以避免把素因子个数过多的答案给算进来了

参考代码

Mophues

```
void getmu(int n){
  memset(vis, 0, sizeof(vis));
  memset(mu, 0, sizeof(mu));
  memset(num, 0, sizeof(num));
  cnt = 0;
  mu[1] = 1;
```

```
  for(int i = 2; i <= n; i++) {
    if(!vis[i]){
        prime[cnt++] = i;
        mu[i] = -1;
        num[i] = 1;
     }
    for(int j = 0; j < cnt && prime[j] * i <= n; j++){
      vis[i * prime[j]] = 1;
      num[i * prime[j]] = num[i] + 1;
      if(i % prime[j] == 0) break;
      mu[i * prime[j]] = -mu[i];
    }
  }
}

void init(){
   memset(sum, 0, sizeof(sum));
   for(int i = 1; i <= 5e5; i++){
      for(int j = i; j <= 5e5; j += i){
         sum[num[i]][j] += mu[j / i];
      }
   }
   for(int i = 1; i <= 5e5; i++){
      for(int j = 1; j <= 19; j++){
         sum[j][i] += sum[j - 1][i];
      }
   }

   for(int i = 1; i <= 5e5; i++){
      for(int j = 0; j <= 19; j++){
         sum[j][i] += sum[j][i - 1];
      }
   }
}
int main(){
   getmu(5e5);
```

```
    init();
    ll n, m;
    int p, T;
    scanf("%d", &T);
    while(T--){
        ll ans = 0;
        scanf("%lld%lld%d", &n, &m, &p);
        if(p > 19){
            printf("%lld\n", n * m);
            continue;
        }
        for(int i = 1; i <= min(n, m);){
            int l, r;
            l = i, r = min(n / (n / i), m / (m / i));
            ans += 1LL * (n / i) * (m / i) * (sum[p][r] - sum[p][l - 1]);
            i = r + 1;
        }
        printf("%lld\n", ans);
    }
    return 0;
}
```


7 怎样跑的更快

7.1 题意

已知 $p = 998244353$, 告诉你 n, c, d, q ，表示 q 个询问，每个询问给出 $b_1, b_2..., b_n$, 让你求出

对于 $1 \leq i \leq n$ 满足

$\sum\limits_{j=1}^{n} gcd(i,j)^c \cdot lcm(i,j)^d \cdot x_j \equiv b_i \pmod p$

的任意一组解 $x_1, x_2..., x_n$

注意：$0 \leq x_1, x_2..x_n, b_1..., b_n < p$

对于所有数据 $nq \leq 3e5$

$0 \leq c, d \leq 10^9$

7.2 题解

以下式子都是在模 p 意义下进行推导

$$b_i=\sum_{j=1}^{n}gcd(i,j)^{c-d}i^dj^dx_j$$

$$=\sum_{j=1}^{n}\cdot f(gcd(i,j))\cdot g(i)\cdot h(j)\cdot x_j$$

$f(n)=n^{c-d}$, 其实 g 和 h 是一回事，不过, 后面我们会发现上面的形式也是可做的

$b_i=\sum\limits_{d|i}\sum\limits_{d|j}[gcd(i,j)==d]f(d)\cdot g(i)\cdot h(j)\cdot x_j$ (枚举 gcd，常规操作)

$$=\sum_{d|i}\sum_{d|j}\sum_{k|\frac{gcd(i,j)}{d}}\mu(k)\cdot f(d)\cdot g(i)\cdot h(j)\cdot x_j$$

$$=\sum_{d|i}\sum_{d|j}\sum_{kd|gcd(i,j)}\mu(k)\cdot f(d)\cdot g(i)\cdot h(j)x_j$$

令 $T=kd$

$$\frac{b_i}{g_i}=\sum_{T|i}\sum_{T|j}\sum_{d|T}\mu(\frac{T}{d})\cdot f(d)\cdot h(j)\cdot x_j$$

此时我们发现 $\sum\limits_{d|T}\mu(\frac{T}{d})\cdot f(d)$ 是两个函数的狄利克雷卷积, 这个显然是可以预处理的，我们设 $R(T)=\sum\limits_{d|T}\mu(\frac{T}{d})\cdot f(d)$

$$\frac{b_i}{g_i}=\sum_{T|i}\sum_{T|j}R(T)\cdot h(j)\cdot x_j$$

$$=\sum_{T|i}R(T)\sum_{T|j}h(j)\cdot x_j$$

设 $Q(T)=\sum\limits_{T|j}h(j)\cdot x_j$

$$\frac{b_i}{g_i}=\sum_{T|i}R(T)\cdot Q(T)$$

所以我们可以先反演出 $R(n)\cdot Q(n)$, 解出 $Q(n)$

$$R(n)\cdot Q(n)=\sum_{d|n}\mu(\frac{n}{d})\cdot\frac{b_d}{g_d}$$

然后反演出 $h(n)\cdot x_n$

$$h(n)\cdot x_n=\sum_{n|d}Q(d)\cdot\mu(\frac{d}{n})$$

所以 x_j 就可以解出来了

代码设计

此题的除法运算基本都用逆元来做

1: 预处理出 $R(n),f(n),g(n)$ 函数

- $R(n)=\sum_{d|n}\mu(\frac{n}{d})\cdot f(d)$
- $f(n)=n^{c-d}$
- $g(n)=h(n)=n^d$

2: 利用莫比乌斯反演求出 $Q(n)$ 函数

- $Q(n)=R(n)^{-1}\sum_{d|n}\mu(\frac{n}{d})\cdot\frac{b_d}{g_d}$
- 判断无解，需要先预处理 $W(n)=\sum_{d|n}\mu(\frac{n}{d})\cdot\frac{b_d}{g_d}$, 在 $R(n)$ 为 0 的时候，判断 $W(n)$ 是否为 0

3: 利用莫比乌斯反演求出 x_j 的值

- $x_n=h(n)^{-1}\sum_{n|d}Q(d)\cdot\mu(\frac{d}{n})$
- 需要先预处理 $S(n)=\sum_{n|d}Q(d)\cdot\mu(\frac{d}{n})$, 注意这是莫比乌斯反演的第二种形式