數論

AY

October 26, 2021

Contents

0.0 簡介

0.0.1 符號定義

Definition 1 (Iverson bracket). 給定一個命題 P，定義 $[P] = \begin{cases} 1 & P \text{ is true} \\ 0 & P \text{ is false} \end{cases}$

Definition 2 (asymptotic notations).

$$\limsup_{x\to\infty} \frac{|f(x)|}{g(x)} < \infty \iff f(x) = O(g(x))$$

$$\limsup_{x\to\infty} \frac{|f(x)|}{g(x)} = 0 \iff f(x) = o(g(x))$$

$$\liminf_{x\to\infty} \frac{f(x)}{g(x)} > 0 \iff f(x) = \Omega(g(x))$$

$$f(x) = O(g(x)) = \Omega(g(x)) \iff f(x) = \Theta(g(x))$$

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 1 \Leftrightarrow f(x) \sim g(x)$$

另外，本章的每一份程式碼前面都需要加上以下程式碼

```cpp
#include<vector>
#include<array>
#include<utility>

#include<chrono>
#include<random>

using std::vector;
using std::array;
using std::pair;

namespace Random {
    // returns a random integer betwenn [a, b]
    int randInt(int a, int b){
        static std::mt19937 rng(
            std::chrono::steady_clock::now().time_since_epoch().count()
        );
        return std::uniform_int_distribution<int>(a, b)(rng);
```

```
19      }
20  }
21
22  // returns a^b mod p
23  int fpow(int a, int b, int p){
24      int ans = 1;
25      do if(b&1) ans = ans*a%p;
26      while(a=a*a%p, b>>=1);
27      return ans;
28  }
```

Listing 1: Prelude

0.0.2 質數

Definition 3 (整除). 若存在整數 x 使得 $b = ax$，則我們稱 a **整除** b，記做 $a|b$。

Definition 4. 如果 $a|b$，則我們稱 a 為 b 的**因數**；b 為 a 的**倍數**。若 $a \neq 1$ 且 $a \neq b$，則我們稱 a 是 b 的**嚴格因數**

Definition 5 (因數集). 定義 $I(n) = \{x \in \mathbb{Z} | x|n\}$ 為 n 的因數集。

Problem 1. 若 a, b 互質，則我們有一個雙射

$$\begin{aligned} I(a) \times I(b) &\xrightarrow{\sim} I(ab) \\ (x, y) &\mapsto xy \end{aligned}$$

Proof. 我們有反函數

$$\begin{aligned} I(ab) &\to I(a) \times I(b) \\ x &\mapsto (\gcd(a, x), \gcd(b, x)) \end{aligned}$$

□

Definition 6 (質數). 若一個正整數 n 的正因數僅有 $1, n$ 兩個，則我們稱 n 為**質數**。大於 1 且不是質數的數字稱為**合數**。所有質數的集合記做 $\mathbb{P}$

Definition 7. $\pi(n)$ 定義為 n 以下的質數個數。第 i 個質數記為 p_i。

Definition 8. $\gcd(a, b) = \max I(a) \cap I(b)$ 定義為 a, b 的最大公因數。

Definition 9 (互質). 若 $\gcd(a, b) = 1$，則我們稱 a, b 互質，記做 $a \perp b$

相信大家都知道輾轉相除法。寫成程式碼就是

```cpp
int gcd(int x, int y){
    return y?gcd(y, x%y):x;
}
```

Listing 2: Euclid's algorithm

Lemma 1. 利用歐幾里德演算法計算 a, b 的最小公因數是 $O(\log \min(a, b))$ 的。

Proof. 第一次遞迴後，兩者皆小於 $\min(a, b)$。之後每次遞迴，其中一個大小會至少減半，直到有一個人達到 0 就中止。 □

0.0.3 篩法

注意到一個正整數是質數若且唯若它恰有兩個因數 2。以下是最直接計算因數並判別質數的方法：

```cpp
vector<int> factors(int n){
    vector<int> v;
    for(int i = 1; i <= n; i++)
        if(n % i == 0)
            v.push_back(i);
    return v;
}

bool isPrime(int n){
    return factors(n).size() == 2;
}
```

Listing 3: Finding factors naively

可以發現，這個演算法的複雜度是 $O(n)$ 的，但我們有更快的方法嗎？

Observation. 若 $ab = n$，則 $a \leq \sqrt{n}, b \leq \sqrt{n}$ 中至少有一個成立。

於是枚舉因數時，我們只需要枚舉至 $\sqrt{n}$ 即可

```cpp
vector<int> factors(int n){
    vector<int> v1, v2;
    for(int i = 1; i*i <= n; i++)
        if(n % i == 0){
```

```
5               v1.push_back(i);
6               if(i*i != n)
7                   v2.push_back(n/i);
8           }
9       reverse(v2.begin(), v2.end());
10      return v1+v2;
11  }
```

Listing 4: Finding factors in $O(\sqrt{n})$

注意當 n 是完全平方數時，$\sqrt{n}$ 不要重複計算到。

0.0.3.1 埃式篩法

首先觀察到一個重要的事實

Observation.

$$\sum_{i=1}^{n} \frac{1}{i} = O(\log n)$$

Proof. 因為 $1/n$ 遞減，所以我們有

$$\sum_{i=1}^{n} \frac{1}{i} = \Theta\left(\int_{1}^{n} \frac{1}{t}\, dt\right) = \log n$$

□

於是，對於 1 到 n 的每個數字以上述方法枚舉因數要花 $\Theta(n\sqrt{n})$ 的時間，但對每個數字枚舉倍數只要 $\Theta(n \log n)$。這件事非常重要，將枚舉因數轉成枚舉倍數常常是解題時減少複雜度的關鍵。在這裡運用這個概念，得到的演算法就是

```
1   array<vector<int>, N> factors;
2   void sieve(int n){
3       for(int i = 1; i <= n; i++)
4           for(int j = 0; j <= n; j += i)
5               factors[j].push_back(i);
6   }
```

Listing 5: Naive sieving

如果我們想要直接將每個數字的因數存下的話，這是最好的複雜度了，但如果我只想判別每個數字是否是質數，我們事實上可以做得更好。若只在判別質數，那上述的演算法就像篩子一樣，一個一個的將每個數字的倍數篩去，因此又稱為埃

式篩法 (sieve of Eratosthenes)。如果我們想要控制篩法的複雜度，我們事實上就必須控制每個數字被篩掉幾次。一個簡單的優化是只拿質數做篩法，這樣的複雜度會進化到 $\Theta(n \log\log n)$（因為Theorem 4告訴你質數的倒數和是 $\Theta(\log\log n)$ 的）。

0.0.3.2 線性篩

前述的優化似乎已經足夠好了，但我們事實上可以讓每個數字只被篩掉一次。若我們想讓每個合數被「最小嚴格因數」與「最大嚴格因數」的組合刪去，那我們可以如何避免多餘組合的嘗試呢？

Definition 10. 對於 $n > 1$，$p(n)$ 定義為 n 的最小質因數。為了敘述方便，我們將 $p(1)$ 定義為 ∞。

Observation. 一個合數 n 的最小嚴格因數是質數 $p(n)$。

於是

Observation. 若 $n = ab$，則 a 是 n 的最小嚴格因數若且唯若 a 是質數且 $a \le p(b)$。

因此我們只要對每個 b，枚舉所有比 $p(b)$ 小的質數即可，亦即

```
vector<int> primes;
array<bool, N> isPrime;
void sieve(int n){
    for(auto& i: isPrime)i = true;
    isPrime[0] = isPrime[1] = false;
    for(int i = 1; i <= n; i++){
        if(isPrime[i])
            primes.push_back(i);
        for(int p: primes){
            if(i*p > n) break;
            isPrime[i*p] = false;
            if(i%p == 0) break;
        }
    }
}
```

Listing 6: Linear sieving

根據以上分析，上述演算法只需要 $\Theta(n)$ 個運算。但事實上加法是比乘除法快的；一個 b 位元的數字只要 $\Theta(b)$ 的時間做加法，但需要 $\Theta(b \log b)$ 的時間做乘除法。這樣算下來，線性篩與埃式篩事實上都是 $\Theta(n \log n \log\log n)$ 的。

0.0.3.3 次線性篩

想要優化這個複雜度，一個可行的方法是減少一開始可能是質數的候選人。

Observation. 整數 $y > p_i\#$ 滿足 $p(y) > p_i$ 若且唯若 $p(y - p_i\#) > p_i$。

這件事告訴我們，如果我們維護 $S_i = \{x \le p_i\# | p(x) > p_i\}$ 了，則 p_{i+1} 就是 $\min\{x \in S_i | x \neq 1\}$，且想獲得 S_{i+1}，只要把所有 $\{kp_i\# + d \le p_{i+1}\# | d \in S_i\}$ 集合起來，再拔掉所有 p_{i+1} 的倍數即可。這導出了以下的演算法

```
dlist S;
vector<int> primes;
void sieve(int n){
    S.push(1);
    primes.push_back(2);
    for(int p = 3, pm = 2; p*p <= n || pm < n; p = S.next(1)){
        // add k*pm + d into S
        int nextpm = min(n, p*pm);
        for(int d = 1; d && pm+d <= nextpm; d = S.next(d))
            S.push(pm+d);
        pm = nextl;

        // remove all multiples of p
        int f = p;
        while(f && p*f <= pm) f = S.next(f);
        while(f > 1) f = S.prev(f), S.erase(p*f);

        primes.push_back(p);
    }
    for(int i = S.next(1); i; i = S.next(i))
        primes.push_back(i);
}
```

Listing 7: Sublinear sieving

其中 `dlist` 是一個支援 $O(1)$ 插入刪除尋找下一個的資料結構。一個可能的實作方法是

```
#include<unordered_map>

struct dlist{
    std::unordered_map<int, pair<int, int>> l;
    dlist(){ l[0] = {0, 0}; }
    int next(int i){ return l[i].first; }
```

```
7       int prev(int i){ return l[i].second; }
8       void push(int i, int j){ // add i directly after j
9           l[i] = {l[j].first, j};
10          l[j].first = l[l[j].first].second = i;
11      }
12      void push(int i){ push(i, l[0].second); }
13      void erase(int i){
14          l[l[i].first].second = l[i].second;
15          l[l[i].second].first = l[i].first;
16          l.erase(i);
17      }
18  };
```

Listing 8: Double linked list

以下證明此演算的複雜度是好的

Theorem 1 ([12]). Listing 7 的時間複雜度是 $\Theta(n/\log\log n)$ 個乘法。

Proof. 註：建議閱讀至Subsection 0.1.3再回來閱讀本證明。
因為迴圈只跑了 $O(\sqrt{n})$ 次，我們只需要估計 `S.push` 與 `S.erase` 發生幾次即可。前者只比後者多 $\pi(n) = O(n/\log n)$ 次，因此重點就是後者的次數。令 p_k 表示滿足 $p_k\# \le n$ 中最大的質數。在前 $k-1$ 次刪除時，`f` 會跑遍所有 $p_{i-1}\#$ 以下且不被 $p_1, \ldots, p_{i-1}$ 整除的數字，也就是刪除 $\varphi(p_{i-1}\#)$ 個數字，一共

$$\sum_{i=1}^{k-1} \varphi(p_{i-1}\#) \le (k-1)\varphi(p_{k-1}\#) \le \varphi(p_k\#) = O\left(\frac{n}{n\log\log n}\right)$$

而剩下刪除的是其餘 2 至 N 之間，且不被任何 $p_i \le p_k$ 整除的數字。不被 $p_i \le p_k$ 整除的數字平均上有 $\frac{\varphi(p_k\#)}{p_k\#}$ 個，因此共有

$$n\frac{\varphi(p_k\#)}{p_k\#} = \Theta\left(\frac{n}{n\log\log n}\right)$$

兩個估計皆使用到了Lemma 8的結果。 □

因此考慮乘法的花費後，該演算法的複雜度是 $\Theta(n\log n)$ 的。另外，原論文亦有提到如何不用雜湊達到該空間複雜度，亦有說明如何利用預處理小數字的乘法表，並透過差分的一些估計做到 $\Theta(n/\log\log n)$ 個加法，因而考慮運算的花費後會是 $\Theta(n\log n/\log\log n)$ 的。

0.0.4 其他質數檢查與因式分解演算法

0.0.4.1 Miller 質數判別法

Property 1. 若 $p = 2^k d + 1$ 是質數，則 $a^d \equiv 1 \pmod p$ 或存在 d' 使得 $a^{2^k d'} \equiv -1 \pmod p$。

注意這是費馬小定理與「1 的平方根只有 1 或 -1」的直接結論（詳情請見Subsection 0.2.1）。

Definition 11. 若 $n = 2^k d + 1$ 不是質數，則一個不滿足「$a^d \equiv 1 \pmod p$ 或存在 d' 使得 $a^{2^k d'} \equiv -1 \pmod p$」的 a 稱為 n 的合數證據。

Property 2 (Miller). 若 n 是合數，則 n 的最小合數證據是 $O(N^{1/4})$ 的。如果擴展黎曼猜想是對的，則最小合數證據會小於 $2\log^2 n$。

Proof. 本證明篇幅過大，從略。有興趣的讀者請參閱 Miller 的 [10]，以及他使用到的估計 [4] 與 [2]。 □

Corollary 1. 我們有一個 $O(N^{1/4}\log^2 N)$ 的演算法檢查 N 是不是質數。

```cpp
#include<cmath>

bool isPrime(int n){
    int a = 1, b = n-1;
    while(b%2 == 0)b/=2, a++;
    for(int i = 0; i <= 2*log(n); i++){
        int x = fpow(i, b, n);
        if(x == 1 || x == n-1) goto failed;
        for(int i = 0; i < a; i++){
            x = x*x%n;
            if(x == n-1) goto failed;
        }
        return false;
        failed:;
    }
    return true;
}
```

Listing 9: Miller test

Remark 1. 注意到你其實只需要檢查質數即可，因為兩個非證據的乘積仍然不是一個證據。而若 $N \leq^9$，所有合數都可以用 $2, 3, 7$ 檢查出，而若 $N \leq 10^{18}$，則所有合數的最小合數證據會小於等於 37，也就是你只要檢查 12 個質數。

0.0.4.2 Strassen 因式分解法

令 $d = \lceil \sqrt[4]{N} \rceil$，則 N 若是合數，一定不會與 $d^2!$ mod N 互質。考慮多項式

$$f(x) = (x+1)(x+2)\dots(x+d)$$

則 $f(0), f(d), f(2d), \dots f(d^2)$ mod N 一定有一個與 N 不互質。注意到

Property 3. 將 d 個一次式相乘只需要 $O(n\log^2 N)$ 的時間。

Proof. 每次拆兩半去算，將兩個 d 次的多項式相乘需要 FFT 的 $O(N\log N)$。(若 N 過大，你需要拿 $\log N$ 個質數各做一遍 NTT，最後 CRT 起來，會多一個 $\log N$ 的倍數）複雜度 T 滿足 $T(N)=2T(N/2)+O(N\log N)$。因此總時間是 $O(N\log^2 N)$。 □

Property 4. 將一個 d 次式對 n 個點求值需要 $O(d\log d+n\log^2 n)$ 的時間

Proof. 對 $x_1,\dots x_n$ 求值事實上就是在求 f 對 $x-x_1,\dots,x-x_n$ 求餘數。首先我們先花 $O(d\log d)$ 計算 $f \mod (x-x_1)\cdots(x-x_n)$，得到一個 n 次式所以我們每次將 x_i 拆成兩半，計算 $f \mod (x-x_1)\cdots(x-x_{n/2})$ 與 $f \mod (x-x_{n/2+1})\cdots(x-x_n)$，並遞迴下去。時間複雜度滿足 $T(n)=2T(n/2)+O(n\log n)$，總複雜度是 $O(n\log^2 n)$ 的 □

因此我們可以在 $O(d\log^2 d)$ 的時間算出 $f \mod N$，再花 $O(d\log^2 d)$ 的時間算出 $f(0),\dots,f(d^2)$。注意我們可以先暴力檢驗 N 有沒有小於 d 的因數，因此我們可以假設 N 的所有因數皆大於 d。特別的，N 只有至多三個因數。
找到跟 N 不互質的 $f(kd)$ 後，我們再對 $\{kd+1,\cdots,kd+d\}$ 做二分搜，尋找哪一個人整除 N。做一次二分搜是 $O(d\log^2 d)$ 的，至多做三次。於是

Corollary 2. 可以在 $O(N^{1/4}\log^2 N)$ 的時間把 N 因式分解。

Remark 2. 目前已知最快的因數分解演算法是 $O(N^{1/4}\log^2 N/\sqrt{\log\log N})$ 的，是本演算法的優化。若考慮隨機演算法，則存在一些次指數的方法，有興趣的讀者請自行翻閱文獻。

0.1 數論函數

0.1.1 捲積

Definition 12 (（狄利克雷）捲積). 令 $f,g:\mathbb{Z}_+\to R$ 為兩個函數。
定義 $f*g:\mathbb{Z}_+\to R$ 為

$$(f*g)(n)=\sum_{d|n}f(d)g(n/d)=\sum_{i=1}^{n}\sum_{j=1}^{n}[ij=n]f(i)g(j)$$

利用前面所敘述的思路，我們也可以簡易的在 $O(n\log n)$ 內求出兩函數的捲積

```
array<int, N> convolute(array<int, N> a, array<int, N> b){
    array<int, N> res;
    for(int i = 1; i <= N; i++)
        for(int j = 1; j*i <= N; j++)
            res[i*j] += a[i]*b[j];
    return res;
}
```

Listing 10: Calculating dirichlet convolution

Property 5. $(f*g)*h = f*(g*h)$

Proof. 兩者的第 n 項皆為 $\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{k}[ijk=n]f(i)g(j)h(k)$ □

Definition 13 (ε 函數)**.** $\varepsilon : \mathbb{Z}_+ \to \mathbb{Z}$，$\varepsilon(n) = [n=1]$

Property 6. 對於所有函數 f，我們有 $\varepsilon * f = f$

Definition 14 (恆等函數)**.** $\mathrm{id} : \mathbb{Z}_+ \to \mathbb{Z}$，$\mathrm{id}(n) = n$

Definition 15 (次方函數)**.** $\mathrm{id}_k : \mathbb{Z}_+ \to \mathbb{Z}$，$\mathrm{id}_k(n) = n^k$

Definition 16 ($\mathbb{1}$ 函數)**.** $\mathbb{1} : \mathbb{Z}_+ \to \mathbb{Z}$，$\mathbb{1}(n) = 1$

Definition 17 (莫比烏斯函數)**.** $\mu : \mathbb{Z}_+ \to \mathbb{C}$，$\mu(n) = \sum_{i=1}^{n}[i \perp n]\zeta_n^i$

Property 7. $\mu * \mathbb{1} = \varepsilon$

Proof.

$$\begin{aligned}
\varepsilon(n) = \sum_{i=1}^{n}\zeta_n^i &= \sum_{d|n}\sum_{i=1}^{n}[\gcd(i,n)=d]\zeta_n^i \\
&= \sum_{d|n}\sum_{i=1}^{n/d}[\gcd(i,n/d)=1]\zeta_n^{id} \\
&= \sum_{d|n}\sum_{i=1}^{n/d}[i \perp n/d]\zeta_{n/d}^{i} \\
&= \sum_{d|n}\mu(n/d) = (\mathbb{1}*\mu)(n)
\end{aligned}$$

□

這其實提供了我們另一個定義 μ 的方式，也就是

Property 8. $\mu(n) = \begin{cases} 1 & n=1 \\ -\sum_{i=1}^{n-1}[i|n]\mu(i) & n>1 \end{cases}$

所以其實 μ 的取值都是整數。於是我們有

Property 9 (莫比烏斯反演)**.** 對於函數 f, g，

$$f * \mathbb{1} = g \iff f = g * \mu$$

其實不只 $\mathbb{1}$，其他函數也有這種反函數

Definition 18 (迪利克雷反函數)**.** 對函數 f，若 $f * g = 1$，則 g 稱為 f 的迪利克雷反函數

Property 10. 函數 $f : \mathbb{Z}_+ \to R$ 有迪利克雷反函數若且唯若 $f(1)$ 可逆

Proof. f 的反函數是 $f^{-1}(n) = \begin{cases} (f(1))^{-1} & n = 1 \\ -(f(1))^{-1} \sum_{i=1}^{n-1} [i|n] f(i) g(n/i) & n > 1 \end{cases}$ □

0.1.2 積性函數

Definition 19 (積性函數)**.** 給定一個函數 $f : \mathbb{Z}_+ \to R$ 。若對於所有互質的 a, b 都有 $f(ab) = f(a)f(b)$，則我們稱 f 是積性的。若對所有 a, b 都滿足 $f(ab) = f(a)f(b)$，則我們稱 f 是積性的。同樣稱滿足 $f(ab) = f(a) + f(b)$ 的函數為加性或完全加性的。

Property 11. 若 f 是積性，則 $f(1) = 1$。

Property 12. 若 f 是積性的，則 g 是積性的若且唯若 $f * g$ 是積性的

Proof. 假設 g 與 $h = f * g$ 只有一個是積性的，並假設 $n = ab$ 是最小的組合使 $g(a)g(b) \neq g(ab)$ 或 $h(a)h(b) \neq h(ab)$。則

$$\begin{aligned} h(a)h(b) &= \left(\sum_{i|a} f(i)g(a/i) \right) \left(\sum_{j|b} f(j)g(b/j) \right) \\ &= \sum_{i|a} \sum_{j|b} f(i)f(j)g(a/i)g(b/j) \\ &= f(1)g(a)g(b) + \sum_{\substack{k|ab \\ k \neq 1}} f(k)g(ab/k) \end{aligned}$$

故

$$h(a)h(b) - h(ab) = g(a)g(b) - g(ab)$$

跟假設矛盾。 □

Property 13. ε 是積性的

Corollary 3. 積性函數的迪利克雷反函數是積性的

Property 14. $\mathbb{1}$ 是積性的

Corollary 4. 莫比烏斯函數是積性的

Definition 20 (歐拉函數; Euler's totient function)**.** $\varphi(n) = \sum_{i=1}^{n} [n \perp i]$

Property 15. $\mathrm{id} = \mathbb{1} * \varphi$

Proof.

$$\begin{aligned} n = \sum_{i=1}^{n} 1 &= \sum_{d|n} \sum_{i=1}^{n} [\gcd(i,n) = d] \\ &= \sum_{d|n} \sum_{i=1}^{n/d} [\gcd(id,n) = d] \\ &= \sum_{d|n} \sum_{i=1}^{n/d} [\gcd(i,n/d) = 1] = \sum_{d|n} \varphi(n/d) = \sum_{d|n} \varphi(d) \end{aligned}$$

□

Corollary 5. φ 是積性的

於是我們可以好好計算 φ 了

Property 16. $\varphi(n) = n \prod_{p|n} \left(1 - \frac{1}{p}\right)$

最後介紹一些有時會用到的函數

Definition 21. $\sigma_k(n)$ 定義為 n 的所有因數的 k 次方和。有時我們會把 σ_0 寫作 d，為質數個數函數，並把 σ_1 的 1 省略，稱為質數和函數。

Property 17. $\sigma_k = \mathrm{id}_k * \mathbb{1}$

所以 σ_k 是積性的。

Definition 22. $\Omega(n)$ 為 n 的質因數個數 (可重複)；$\omega(n)$ 為相異質因數個數。

Property 18. $\omega(n)$ 是加性的；$\Omega(n)$ 是完全加性的。

Definition 23 (劉維爾函數)**.** $\lambda(n) = (-1)^{\Omega(n)}$

因為 $\Omega(n)$ 是完全加性的，所以劉維爾函數是完全積性的

Definition 24 (特徵函數)**.** 若 $S \subseteq \mathbb{Z}$ 是一個集合，則 $\chi_S : \mathbb{Z} \to [0,1], x \mapsto [x \in S]$。

Definition 25 (梅滕斯函數)**.** $M(n) = \sum_{i=1}^{n} \mu(i)$ 是莫比烏斯函數的前綴和。

Definition 26. $v_p(n)$ 定義為最大的 $k \in \mathbb{N}$ 使得 $p^k | n$。

Definition 27 (質數階乘; primorial)**.** $n\# = \prod_{p \le n} p$ 是比 n 小的質數的乘積。

以及推導式子偶爾會用到的一些卷積關係

Property 19.

1. $\lambda * \mu^2 = \varepsilon$
2. $\lambda * \mathbb{1} = \chi_{\{x^2 | x \in \mathbb{Z}\}}$
3. $\mathbb{1} * \mu^2 = 2^\omega$
4. $\mathbb{1} * \chi_{\mathbb{P}} = \omega$
5. $(\mathrm{id}_k f) * (\mathrm{id}_k g) = \mathrm{id}_k (f * g)$

0.1.3 關於一些函數增長的估計

本節證明大多修改自 [3] 與 [7]。
本節中，所有求和符號底下的 p 跑遍的是所有質數；所有 log 皆為自然對數。

0.1.3.1 切比雪夫函數與質數個數函數

Definition 28 (Chebyshev functions)**.**

$$\psi(n) = \sum_{p^k \le n} \log p$$

$$\vartheta(n) = \sum_{p \le n} \log p = \log n\#$$

Property 20.

$$\psi(n) = \sum_{i \le \log_2 n} \vartheta(\sqrt[i]{n})$$

Proof. 若 $i > \log_2 n$，則 $\sqrt[i]{n} < 2$，因此 $\vartheta(\sqrt[i]{n}) = 0$。 □

Property 21.

$$\vartheta(n) \le \psi(n) \le \pi(n) \log n$$

Proof.

$$\psi(n) = \sum_{p^k \le n} \log p = \sum_{p \le n} \left\lfloor \frac{\log n}{\log p} \right\rfloor \log p \le \sum_{p \le n} \log n = \pi(n) \log n$$

□

所以事實上切比雪夫函數與質數個數函數的增長有一定的關聯性。事實上利用以下的定理，我們可以估計 $\pi(n)\log n$ 與 $\vartheta(n)$ 的差距。

Theorem 2 (Abel summation)**.** 若 a_i 是一個數列，$A_i = \sum_{j=0}^{i} a_j$，且 $f \in C^1([l,r])$。則

$$\sum_{i=l+1}^{r} a_i f(i) = A_r f(r) - A_l f(l) - \int_l^r A_{\lfloor t \rfloor} f'(t)\,dt$$

Proof. 我們只需要證明 $r = l+1$ 的情形即可，因為在兩段區間合併時左右皆可直接相加。此時很簡單的就可以算出

$$\begin{aligned} &A_r f(r) - A_l f(l) - \int_l^r A_l f'(t)\,dt \\ &= (A_l + a_r) f(r) - A_l f(l) + A_l(f(r) - f(l)) \\ &= a_r f(r) \end{aligned}$$

□

Lemma 2.

$$\vartheta(n) = \pi(n)\log n - \int_1^n \frac{\pi(t)}{t}\,dt$$

Proof. 令數列 $a_i = [i \in \mathbb{P}]$，則其前綴和為 $\pi(i)$。對函數 $f(r) = \log r$ 使用阿貝爾求和定理即可。 □

接著利用以下這個兩個引理，我們可以給出這三個函數的上下界

Lemma 3.

$$\log \binom{2n}{n} = \Theta(n)$$

Proof.

$$\binom{2n}{n} \le \sum_{i=0}^{2n} \binom{2n}{i} = 2^{2n} = 4^n$$

另外

$$\binom{2n}{n} \ge \frac{1}{2n+1} \sum_{i=0}^{2n} \binom{2n}{i} = \frac{4^n}{2n+1}$$

取 log 便有

$$n\log 4 - \log(2n+1) \le \log \binom{2n}{n} \le n \log 4$$

故

$$\log \binom{2n}{n} \sim n \log 4$$

□

Lemma 4.

$$\log n! = \sum_{p^k} \left\lfloor \frac{n}{p^k} \right\rfloor \log p$$

Proof. 計算 p 在左右兩邊出現的次數即可。 □

首先是上界

Lemma 5.

$$\vartheta(n) = O(n)$$

Proof. 首先注意到 $(n, 2n]$ 間的質數都會整除 $\binom{2n}{n}$。因此

$$\vartheta(2n) - \vartheta(n) = \sum_{p \in (n,2n]} \log p \leq \log \binom{2n}{n} = O(n)$$

$$\vartheta(n) = O(n + n/2 + n/4 + ...) = O(n)$$

□

Corollary 6.

$$\psi(n) = O(n), \quad \pi(n) = O\left(\frac{n}{\log n}\right)$$

Proof.

$$\psi(n) = \vartheta(n) + \sum_{2 \leq i \leq \log_2 n} \vartheta(\sqrt[i]{n}) = O(n) + O(\sqrt{n} \log n) = O(n)$$

$$\pi(n) \log n = \vartheta(n) + \int_1^n \frac{\pi(t)}{t}\, dt \leq O(n) + \int_1^n dt = O(n)$$

□

而下界的部分也可以得到相似的結果

Lemma 6.

$$\psi(n) = \Omega(n)$$

Proof.

$$\ln 2n! - 2 \ln n! = \ln \binom{2n}{n} = \Omega(n)$$

另外，利用Lemma 3可以得到

$$\ln 2n! - 2 \ln n! = \sum_{p^k} \left(\left\lfloor \frac{2n}{p^k} \right\rfloor - 2 \left\lfloor \frac{n}{p^k} \right\rfloor \right) \log p \leq \sum_{p^k \leq 2n} \log p = \psi(n)$$

□

Corollary 7.

$$\vartheta(n) = \Omega(n), \quad \pi(n) = \Omega\left(\frac{n}{\log n}\right)$$

Proof. 注意到

$$\begin{aligned}\int_1^n \frac{\pi(t)}{t}\,dt &= \int_2^n \frac{\pi(t)}{t}\,dt \\ &= O\left(\int_2^n \frac{1}{\log t}\,dt\right) \\ &= O\left(\int_2^n \frac{\log t - 1}{\log^2 t}\,dt\right) \\ &= O\left(\frac{n}{\log n}\right) = o(n)\end{aligned}$$

因此

$$\Omega(n) = \psi(n) \le \pi(n)\log n = \vartheta(n) + o(n)$$

□

於是本節的結論是

Theorem 3.

$$\psi(n) = \Theta(n), \quad \vartheta(n) = \Theta(n), \quad \pi(n) = \Theta\left(\frac{n}{\log n}\right), \quad p_n = \Theta(n\log n)$$

Proof. 唯一還沒證明的就是關於 p_n 的估計。注意到

$$n = \pi(p_n) \ge C\frac{p_n}{\log p_n}$$

故

$$Cp_n \le n\log p_n \le n\log n$$

而

$$n = \pi(p_n) \le C\frac{p_n}{\log p_n} \le Cp_n$$

有

$$Cp_n \ge n\log p_n \sim n\log n$$

□

Remark 3. 事實上，我們可以做到

$$\psi(n) \sim n, \quad \vartheta(n) \sim n, \quad \pi(n) \sim \frac{n}{\log n}, \quad p_n \sim n\log n$$

這四個敘述是等價的，稱為質數定理 (Prime number theorem; PNT)，但證明需要相當長的篇幅，或較難的複分析工具，因此在此略過。但這個性質在估計複雜度時也有時會派上用場，因此也可以簡單記著。

0.1.3.2 梅滕斯定理

本章的三個定理是由梅滕斯提出的，有時會被稱作梅滕斯第一至第三定理。這可以簡單的透過 PNT 推得，但這裡給出的是（類似）當初梅滕斯的初等證明。當時 PNT 僅是一個猜想。

Lemma 7.

$$\sum_{p\le n}\frac{\log p}{p}=\log n+O(1)$$

Proof. 注意到 p 在 $n!$ 的因式分解會出現 $\sum_k\left\lfloor\frac{n}{p^k}\right\rfloor$ 次。因此

$$\frac{1}{n}\log n!=\frac{1}{n}\sum_{p^k\le n}\left\lfloor\frac{n}{p^k}\right\rfloor\log p\ge\frac{1}{n}\sum_{p\le n}\left\lfloor\frac{n}{p}\right\rfloor\log p>\sum_{p\le n}\frac{\log p}{p}-\frac{1}{n}\sum_{p\le n}\log p\ge\sum_{p\le n}\frac{\log p}{p}+O(1)$$

另外

$$\frac{1}{n}\log n!=\frac{1}{n}\sum_{p^k\le n}\left\lfloor\frac{n}{p^k}\right\rfloor\log p\le\frac{1}{n}\sum_{p^k\le n}\frac{n}{p^k}\log p\le\sum_{p\le n}\frac{\log p}{p}+\sum_{\substack{p^k\le n\\k\ge 2}}\frac{\log p}{p^k}$$

注意到

$$\sum_{\substack{p^k\le n\\k\ge 2}}\frac{\log p}{p^k}\le\sum_{p=2}^{\infty}\log p\left(\sum_{n=2}^{\infty}\frac{1}{p^n}\right)=\sum_{p=2}^{\infty}\frac{\log p}{p^2-1}<\infty$$

且斯特靈近似又告訴你 $\log n!=n\log n+O(n)$，因此

$$\log n+O(1)=\frac{\log n!}{n}=\sum_{p\le n}\frac{\log p}{p}+O(1)$$

□

Theorem 4.

$$\sum_{p\le n}\frac{1}{p}=\log\log n+O(1)$$

Proof. 令數列 $a_i=[i\in\mathbb{P}]\frac{\log i}{i}$，則其前綴和為 $A_i=\sum_{p\le i}\frac{\log p}{p}=\log i+O(1)$。對函數 $f(r)=1/\log(r)$ 使用阿貝爾求和定理得到

$$\begin{aligned}\sum_{p\le n}\frac{1}{p}&=\frac{\log n+O(1)}{\log n}+\int_2^n\frac{\log t+O(1)}{t\log^2 t}\,dt\\&\le 1+O(1/\log n)+\log\log n+O(1)\int_0^{\infty}\frac{1}{t\log^2 t}\,dt\\&=\log\log n+O(1)\end{aligned}$$

□

Corollary 8.

$$\prod_{p\le n}(1-1/p)=\Theta\left(\frac{1}{\log n}\right)$$

Proof. 首先，由泰勒展開我們可以得到

$$0\le -\log\left(1-\frac{1}{p}\right)-\frac{1}{p}\le\sum_{i=2}^{\infty}\frac{1}{ip^i}\le\sum_{i=2}^{\infty}\frac{1}{2p^i}=\frac{1}{2p(p-1)}\le\frac{1}{p^2}$$

但 $\sum_p p^{-2}\le\sum_n n^{-2}<\infty$，故

$$\sum_{p\le n}-\log\left(1-\frac{1}{p}\right)-\sum_{p\le n}\frac{1}{p}=\sum_{p\le n}\left(\log\left(1-\frac{1}{p}\right)-\frac{1}{p}\right)=O(1)$$

亦即

$$\prod_{p\le n}(1-1/p)=e^{-\log\log n+O(1)}=\Theta\left(\frac{1}{\log n}\right)$$

□

Remark 4. 事實上梅滕斯第三定理敍述的是

$$\prod_{p\le n}(1-1/p)\sim\frac{e^{\gamma}}{\log n}$$

其中 $\gamma=\lim_{n\to\infty}\left(\left(\sum_{k=1}^{n}\frac{1}{k}\right)-\ln(n)\right)$ 是歐拉-馬斯刻若尼常數。但這個界需要一些比較麻煩的複分析技巧所以被跳過了。

0.1.3.3 其他函數的估計

首先是 φ 的估計。由於 $\varphi(p)=p-1$，故 $\varphi(n)=O(n)$ 是最好的估計了。對於下界，從 φ 的乘積表達式中很清楚可以看出在 $n=p_i\#$ 時達到下界。

Lemma 8. 當 $n=p_i\#$ 時，

$$\varphi(n)=\Theta\left(\frac{n}{\log\log n}\right)$$

Proof.

$$\varphi(p_i\#)=p_i\#\prod_{p\le p_i}(1-1/p)=\Theta\left(\frac{p_i\#}{\log p_i}\right)$$

而當 $n=p_i\#$，則 $\log n=\vartheta(p_i)=\Theta(p_i)$，故 $p_i=\Theta(\log n)$。故

$$\varphi(n)=\Theta\left(\frac{p_i\#}{\log p_i}\right)=\Theta\left(\frac{n}{\log\log n}\right)$$

□

Corollary 9.

$$\varphi(n) = \Omega\left(\frac{n}{\log\log n}\right)$$

接著是質因數個數的估計。顯然下界是質數的 1。Ω 的上界發生在 2 的冪次時。故

Property 22. $\Omega(n) \leq \log_2(n)$

而 ω 的上界如下。

Property 23. $\omega(n) = O\left(\frac{\log n}{\log\log n}\right)$

Proof. ω 的上界發生在 $n = p_i\#$ 時。此時 $p_i = \Theta(\log n)$，同時 $p_i = \Theta(i \log i)$。我們有 $\log i + \log\log i \sim \log\log n$，因此 $\log i = \Theta(\log\log n)$。因此此時

$$\omega(n) = i = \frac{\Theta(\log n)}{\log i} = \Theta\left(\frac{\log n}{\log\log n}\right)$$

□

不過對於一般的題目範圍來說，查表還是比較準的。

N 上界	$\omega(N)$ 上界	$\Omega(N)$ 上界	$2^{\omega(N)}$ 上界	$d(N)$ 上界
10^6	7	19	128	240
10^9	9	29	512	1344
10^{12}	11	39	2048	6720
10^{15}	13	49	8192	26880
10^{18}	15	59	32768	103680

Table 1: Numerical limits of some functions

0.1.4 預處理其他函數

讓我們再回去看看線性篩。他在篩去數字時，恰好分成 $i \perp p$ 與 $p|i$ 兩種情形，正好適合我們預處理一些積性函數。

```
vector<int> primes;
array<bool, N> isPrime;
array<int, N> f;
void sieve(int n){
    for(auto& i: isPrime)i = true;
    isPrime[0] = isPrime[1] = false;
    f[1] = 1;
    for(int i = 1; i <= n; i++){
        if(isPrime[i]){
            primes.push_back(i);
            f[i] = /* i is prime */;
```

```
12          }
13          for(int p: primes){
14              if(i*p > n) break;
15              isPrime[i*p] = false;
16              if(i%p == 0){
17                  f[i*p] = /* i is a multiple of p */;
18                  break;
19              }
20              f[i*p] = f[i] * f[p];
21          }
22      }
23  }
```

Listing 11: Preprocessing multiplicative functions

在上圖的程式碼中，f 是一個積性函數，則我們只要知道如何填入裡面的兩個空格，就可以在 $\Theta(n)$ 的時間預處理出 f。以下我們來看看一些熟悉的積性函數要如何這樣預處理

函數	/* prime */	/* p | i */
μ	-1	0
φ	p-1	f[i]*p
λ	-1	-f[i]
2^{Ω}	2	f[i]*2
2^{ω}	2	f[i]

Table 2: Some functions that can be preprocessed in Listing 11

注意到完全機性函數總是可以這樣預處理出來。因此

Theorem 5. 若 f 是一個完全積性函數，且對於所有 p，$f(p)$ 可以在 $O(T(n))$ 算出，則所有 $1 \le x \le n$ 的 $f(x)$ 值可以在 $O\left(n + \frac{nT(n)}{\log n}\right)$ 得到。

但對於更一般的函數，我們似乎需要尋找更好的辦法。最直接的方法就是維護 $p(n)$ 與 $p(n)^{v_{p(n)}(n)}$，因此我們就可以把每個數字拆成一個質數冪次及一個與他互質的數。

```
1   vector<int> primes;
2   array<bool, N> isPrime;
3   array<int, N> f, pvp, lp, vp;
4   void sieve(int n){
5       for(auto& i: isPrime)i = true;
6       isPrime[0] = isPrime[1] = false;
7       f[1] = 1;
8       for(int i = 1; i <= n; i++){
```

```
 9          if(isPrime[i]){
10              primes.push_back(i);
11              lp[i] = pvp[i] = i;
12              vp[i] = 1;
13              f[i] = /* i is prime */;
14          }
15          for(int p: primes){
16              if(i*p > n) break;
17              isPrime[i*p] = false;
18              if(i%p == 0){
19                  lp[i*p] = p;
20                  pvp[i*p] = pvp[i]*p;
21                  vp[i*p] = vp[i]+1;
22                  if(i*p == pvp[i*p])
23                      f[i*p] = /* i*p = pow(p, vp[i*p]) */;
24                  else
25                      f[i*p] = f[i/pvp[i]] * f[pvp[i*p]];
26                  break;
27              }
28              f[i*p] = f[i] * f[p];
29          }
30      }
31  }
```

Listing 12: Preprocessing any multiplicative function

因此

Theorem 6. 若 f 是一個完全積性函數，且對於所有 p，$f(p)$ 可以在 $O(T_1(n))$ 算出；其餘 $f(p^k)$ 可以在 $O(T_2(n))$ 算出，則所有 $1 \le x \le n$ 的 $f(x)$ 值可以在 $O\left(n + \frac{n}{\log n}T_1(n) + \sqrt{n}T_2(n)\right)$ 得到。

Proof. 小於 n 不是質數的質數冪次個數是

$$\sum_{2 \le i \le \log_2 n} \pi(\sqrt[i]{n}) = O\left(\frac{\sqrt{n}}{0.5 \log n} \log_2 n\right) = O(\sqrt{n})$$

的 □

因此，只要積性函數在質數的取值可以在 $O(\log n)$ 內做出，其餘質數冪次的值能在 $O(\sqrt{n})$ 內算出，則該函數就可以在線性被預處理完成。加性函數與完全加性函數也同理。至於其他函數，就需要照著不同的情況各自發揮創意了。
另外，維護好 $p(n)$ 與 $p(n)^{v_{p(n)}(n)}$ 後，也可以得到以下的結論

Theorem 7. 可以在 $\Theta(n)$ 的預處理後，對每次詢問任何 $1 \le x \le n$，在 $\Theta(\omega(x))$ 內的時間回答出 x 的質因數分解。

根據先前對 $\omega(n)$ 的估計，這其實是非常快的。

0.1.5 關於整除的求和

首先先介紹一些有力的技巧。

0.1.5.1 數論分塊

Definition 29. $D_N = \{\lfloor N/i \rfloor | 1 \le i \le N\}$

Lemma 9. $|D_N| = O(\sqrt{N})$

Proof. 若 $\lfloor N/i \rfloor \ge \sqrt{N}$，僅有 $\lfloor N/1 \rfloor, \lfloor N/2 \rfloor, \dots \lfloor N/\lceil \sqrt{N} \rceil \rfloor$ 等 $\sqrt{N}$ 種。否則只有 $\sqrt{N}$ 種。一共是 $O(\sqrt{N})$ 的。 □

因此若求和的內容只有 $O(\sqrt{N})$ 種，我們就可以把時間壓到 $O(\sqrt{N})$。另一個會用到的事情是

Property 24. $\max\{i | \lfloor N/i \rfloor = \lfloor N/x \rfloor\} = \left\lfloor \frac{N}{\lfloor N/x \rfloor} \right\rfloor$

Proof. 首先令 $y = \lfloor N/x \rfloor$，並令 $N = xy + r_1, r_1 = qy + r_2$，則 $N = y(x+q) + r_2$。因為 $r_2 < y$ 且 $r_2 < x \le x + q$，因此 $\lfloor N/y \rfloor = x + q, \lfloor N/(x+q) \rfloor = y$。另外 $\lfloor N/i \rfloor = y \Rightarrow iy \le N$，因此 $\lfloor N/y \rfloor$ 是該集合裡面最大的。 □

Example 1. 給定 N，試計算 $\sum_{i=1}^{N} \sigma_0(N)$。

Solution. 所求是

$$\sum_{i=1}^{N} \sigma_0(N) = \sum_{i=1}^{N} \sum_{d|i} 1 = \sum_{d=1}^{N} \sum_{i=1}^{N} [d|i] = \sum_{d=1}^{N} \sum_{i=1}^{\lfloor N/d \rfloor} 1 = \sum_{d=1}^{N} \lfloor N/d \rfloor$$

利用數論分塊可以在 $O(\sqrt{N})$ 做出。(事實上可以做到 $O(N^{1/3})$[13])。

```
1 int divisor_prefix_sum(int N){
2     int ans = 0;
3     for(int i = 1; i <= N; i++){
4         int nexti = n/(n/i);
5         ans += (nexti - i + 1) * (n/i);
6         i = nexti;
7     }
8     return ans;
9 }
```

Listing 13: Calculating the prefix sum of the divisor function

□

0.1.5.2 杜教篩

Definition 30 (前綴和). $S_f(n) = \sum_{x \le n} f(x)$

Lemma 10. $S_{f*g}(n) = \sum_{d=1}^{n} f(d) S_g(\lfloor n/d \rfloor)$

Proof.

$$S_{f*g}(n) = \sum_{i=1}^{n} \sum_{d|i} f(d) g(i/d) = \sum_{d=1}^{n} f(d) \sum_{i=1}^{n} [d|i] g(i/d)$$
$$= \sum_{d=1}^{n} f(d) \sum_{i=1}^{\lfloor n/d \rfloor} g(i) = \sum_{d=1}^{n} f(d) S_g(\lfloor n/d \rfloor)$$

□

Corollary 10. 若我們知道 S_f, S_g 在 D_N 的取值，則我們可以 $O(\sqrt{N})$ 算出 $S_{f*g}(N)$。

Corollary 11. 若我們可以 $O(N)$ 算出 $f * g$ 的前 N 項，則已知 S_f, S_g 在 D_N 的取值，我們可以 $O(N^{2/3})$ 算出 $S_{f*g}(n)$ 在 D_N 的取值。

Proof. 前 $K \ge \sqrt{N}$ 項線性做，後面 $O(\sqrt{n})$ 做。則總時間是

$$K + \int_1^{N/K} \sqrt{N/x}\, dx = O(K + \sqrt{N^2/K})$$

取 $K \sim N^{2/3}$ 的時候有 $O(N^{2/3})$。 □

```cpp
#include<unordered_map>
using std::unordered_map;

int convolute_prefix_sum(int N, unordered_map<int, int> Sf,
        unordered_map<int, int> Sg){
    int ans = 0;
    for(int i = 1; i <= N; i++){
        int nexti = n/(n/i);
        ans += (Sf[nexti] - Sf[i-1]) * Sg[n/i];
        i = nexti;
    }
    return ans;
}
```

Listing 14: Calculating $S_{f*g}(N)$ given S_f and S_g

Lemma 11 (杜教篩). 給定 S_f, S_{f*g} 在 D_N 的取值，可以 $O(N^{4/3})$ 算出 S_g 在 D_N 的取值。若可以 $O(N)$ 求出 g 的前 N 項（例如當 g 是積性），這可以被優化到 $O(N^{2/3})$。

Proof. 考慮 $S_g(N) = \frac{1}{f(1)}\left(S_{f*g}(N) - \sum_{d=2}^{N} f(d)S_g(\lfloor N/d \rfloor)\right)$ 所以有了 S_f 就可以對右側做數論分塊並遞迴下去。時間是

$$\sum_{i=1}^{\sqrt{n}} \sqrt{n/i} + \sqrt{i} \sim \int_1^{\sqrt{n}} \sqrt{n/i} + \sqrt{i}\, di = O(n^{4/3})$$

若前 $K > \sqrt{N}$ 項可以 $O(K)$ 做，剩下的項可以在

$$\sum_{i=1}^{\lfloor N/K \rfloor} \sqrt{N/i} \sim \int_1^{N/K} \sqrt{N/i}\, di = O(\sqrt{N^2/K})$$

做完。取 $K \sim N^{2/3}$ 得到 $O(N^{2/3})$。 □

```cpp
#include<unordered_map>
using std::unordered_map;

unordered_map<int, int> du_sieve(int N, unordered_map<int, int> Sf,
        unordered_map<int, int> Sfg, const vector<int> &Sg){

    unordered_map<int, int> ret;
    for(int i = 1; i <= N; i++){
        int nexti = n/(n/i);
        if(nexti < Sg.size()) ret[nexti] = Sg[nexti];
        else {
            int ans = Sfg[nexti];
            for(int j = 1; j <= nexti; j++){
                int nextj = nexti/(nexti/j);
                ans -= (Sf[nextj] - Sf[j-1]) * ret[nexti/j];
            }
            ret[nexti] = ans/Sf[1];
        }
        ans += (Sf[nexti] - Sf[i-1]) * Sg[n/i];
        i = nexti;
    }
    return ret;
}
```

Listing 15: Du's sieve

Corollary 12. 若 f 是積性，給定 S_f 在 D_N 的取值，可以在 $O(N^{2/3})$ 做出 $S_{f^{*-1}}$ 在 D_N 的取值。

Corollary 13. 以下函數的前綴和在 D_n 的取值都可以透過杜教篩在 $O(N^{2/3})$ 篩出

1. μ （已知 $S_{\mathbb{1}}$ 與 $S_{\mu*\mathbb{1}} = S_\varepsilon$）
2. φ （已知 $S_{\mathbb{1}}$ 與 $S_{\varphi*\mathbb{1}} = S_{\mathrm{id}}$）
3. λ （已知 $S_{\mathbb{1}}$ 與 $S_{\lambda*\mathbb{1}}(n) = \lfloor n^2 \rfloor$）
4. μ^2 （已知 $S_{\mathbb{1}}$ 與 $S_{\varphi*\mathbb{1}} = S_{\mathrm{id}}$）

0.1.5.3 π 的計算

本節主要介紹 Meissel–Lehmer 演算法 [9]，但注意一些地方的定義與原論文不同。

Definition 31. $\varphi(n, m) = \sum_{i=2}^{n} [p(i) > m]$

注意到其實只需要存 m 是質數的時候的 $\varphi(n, m)$ 即可。我們有以下兩件事

Property 25. 若 $\sqrt{n} < m \le n$，則 $\varphi(n, m) = \pi(n) - \pi(m)$

Property 26. $\varphi(n, p) = \varphi(n, p-1) - \varphi(\lfloor n/p \rfloor, p-1)$

Corollary 14. 我們可以在 $O(N^{3/4})$ 打完 $\varphi(i, p)$ 在 $i \in D_N, p \le \sqrt{i}$ 的表

因為我們有 $\pi(n) = \varphi(n, \sqrt{n}+1) - \pi(\sqrt{n}+1)$，所以

Corollary 15. 我們可以在 $O(N^{3/4})$ 算出 $\pi(x)$ 在 D_N 的取值。

事實上這個做法可以再優化。

Property 27. 可以在 $O((N \log N \log\log N)^{2/3})$ 算出所有 $\pi(n)$ 在 D_N 的值。

Proof. 假設 $K \ge \sqrt{N}$ 是一個常數，我們可以用 $O(K \log\log K)$ 的線性篩維護一個持久化線段樹，讓我們可以 $O(\log K)$ 詢問到 $\varphi(n, p), p \le n \le K$。剩下的部分好好打表，複雜度是 $\int_1^{N/K} \sqrt{N/i}\, di = \sqrt{N^2/K}$。$\sqrt{N^2/K} + K \log N \log\log N$ 取 $K = N^{1/3}/(\log N \log\log N)^{2/3}$ 得到 $O((N^2 \log N \log\log N)^{1/3})$。 □

Remark 5. 在 [6] 中，這個想法可以透過更詳細的分析各種狀況的 $\phi(n, p)$，將 $\pi(N)$ 切成了九個不同的求和，使打表只需要打到 $p = O(n^{1/4})$，進一步優化到 $O(N^{2/3} \log^{-2} N)$。有興趣的讀者可以自行參閱。

Remark 6. 透過處理 $\sum_{i=2}^{n} [p(i) \le m] f(i)$，此方法亦可以求某些積性函數的前綴和。有興趣的讀者請參閱 [11] 或 [14]。

0.1.5.4 範例題目

Example 2. 給定 N, M，求滿足 $1 \le a \le N, 1 \le b \le M$, 且 a, b 互質的數對個數。

Solution. 題目要求 $\sum_{i=1}^{N}\sum_{j=1}^{M}[\gcd(i,j)=1]$。我們有

$$\begin{aligned}\sum_{i=1}^{N}\sum_{j=1}^{M}[\gcd(i,j)=1] &= \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{g\mid \gcd(i,j)}\mu(g)\\ &= \sum_{g=1}^{\min(N,M)}\mu(g)\sum_{i=1}^{N}\sum_{j=1}^{M}[g|i][g|j]\\ &= \sum_{g=1}^{\min(N,M)}\mu(g)\lfloor N/g\rfloor\lfloor M/g\rfloor\end{aligned}$$

假設 $N \le M$，多筆詢問可以做 $O(N)$ 的線性篩預處理，再對每次詢問做一次 $O(\sqrt{N})$ 的數論分塊；單筆詢問只要對 N, M 各做一次杜教篩，複雜度是 $O(N^{2/3}+M^{2/3})$。 □

Example 3. 給定 N, M，求滿足 $1 \le a \le N, 1 \le b \le M$, 且 $\gcd(a,b) \in \mathbb{P}$ 的數對個數。

Solution. 枚舉 $p = \gcd(a,b)$，我們得到答案是

$$\begin{aligned}\sum_{p\le N,M}\sum_{i=1}^{N/p}\sum_{j=1}^{M/p}[\gcd(i,j)=1] &= \sum_{p\le N,M}\sum_{g=1}^{\min(N/p,M/p)}\mu(g)\lfloor N/pg\rfloor\lfloor M/pg\rfloor\\ &= \sum_{m\le N,M}\left(\sum_{p\mid m}\mu(m/p)\right)\lfloor N/m\rfloor\lfloor M/m\rfloor\\ &= \sum_{m\le N,M}(\chi_{\mathbb{P}}*\mu)(n)\lfloor N/m\rfloor\lfloor M/m\rfloor\end{aligned}$$

注意到

$$\sum_{p\mid n}\mu(n/p)=\begin{cases}0 & \Omega(n)>\omega(n)+1 \text{ or } n=1\\ (-1)^{\omega(n)} & \Omega(n)=\omega(n)+1\\ (-1)^{\omega(n)-1}\omega(n) & \Omega(n)=\omega(n)\end{cases}$$

因此可以很簡單的透過線性篩預處理。假設 $N \le M$，多筆詢問可以做 $O(N)$ 的預處理，再對每次詢問做一次 $O(\sqrt{N})$ 的數論分塊；單筆詢問只要對 N, M 各做一次卷積（注意到 $S_{\chi_{\mathbb{P}}} = \pi$ 在 D_N, D_M 的值可以好好算），複雜度是 $O(M^{2/3}(\log M \log\log M)^{1/3})$。 □

Example 4. 給定 N, M，求滿足 $1 \le a \le N, 1 \le b \le M$ 的 $\gcd(a,b)$ 的和。

Solution. 所求是

$$\begin{aligned}\sum_{i=1}^{N}\sum_{j=1}^{M}\gcd(i,j) &= \sum_{g=1}^{\min(N,M)} g\sum_{i=1}^{N}\sum_{j=1}^{M}[\gcd(i,j)=m]\\ &= \sum_{g=1}^{\min(N,M)} g \sum_{h=1}^{\min(N/g,M/g)} \mu(h)\lfloor N/gh\rfloor\lfloor M/gh\rfloor\\ &= \sum_{k=1}^{\min(N,M)} \left(\sum_{g|k} g\mu(k/g)\right)\lfloor N/k\rfloor\lfloor M/k\rfloor\\ &= \sum_{k=1}^{\min(N,M)} \varphi(k)\lfloor N/k\rfloor\lfloor M/k\rfloor\end{aligned}$$

注意到 $\mu * \mathrm{id} = \varphi$。複雜度一樣是 $O(N + Q\sqrt{N})$ 或 $O(M^{2/3})$ 的。 □

Example 5. 給定 N, M，求滿足 $1 \le a \le N, 1 \le b \le M$ 的 $\mathrm{lcm}(a, b)$ 的和。

Solution. 所求是

$$\begin{aligned}\sum_{i=1}^{N}\sum_{j=1}^{M}\mathrm{lcm}(i,j) &= \sum_{g=1}^{\min(N,M)}\sum_{i=1}^{N}\sum_{j=1}^{M}[\gcd(i,j)=g]ij/g\\ &= \sum_{g=1}^{\min(N,M)}\sum_{i=1}^{N/g}\sum_{j=1}^{M/g}[\gcd(i,j)=1]ijg\\ &= \sum_{g=1}^{\min(N,M)}\sum_{h=1}^{\min(N/g,M/g)}\sum_{i=1}^{N/gh}\sum_{j=1}^{M/gh}\mu(h)ijh^2g\\ &= \sum_{k=1}^{\min(N,M)}\left(\sum_{h|k}(k/h)h^2\mu(h)\right)S_{\mathrm{id}}(\lfloor N/k\rfloor)S_{\mathrm{id}}(\lfloor M/k\rfloor)\\ &= \sum_{k=1}^{\min(N,M)}(\mathrm{id}*\mu\mathrm{id}_2)(k)S_{\mathrm{id}}(\lfloor N/k\rfloor)S_{\mathrm{id}}(\lfloor M/k\rfloor)\end{aligned}$$

注意到 $(\mathrm{id} * \mu\mathrm{id}_2) * \mathrm{id}_2 = \mathrm{id}$ 可以杜教篩。複雜度一樣是 $O(N + Q\sqrt{N})$ 或 $O(M^{2/3})$ 的。 □

Example 6. 給定 N, M，求滿足 $1 \le a \le N, 1 \le b \le M$ 的 $\sigma_1(ab)$ 的和。

Solution. 所求是

$$\begin{aligned}
\sum_{i=1}^{N}\sum_{j=1}^{M}\sigma_1(ij) &= \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{d|ij} d \\
&= \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{d'|i}\sum_{k|j}[\gcd(d'k,i)=d']d'k \\
&= \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{d''|i}\sum_{k|j}[\gcd(k,d'')=1]ik/d'' \\
&= \sum_{g=1}^{\min(N,M)}\mu(g)\sum_{i=1}^{N/g}\sum_{j=1}^{M/g}\sum_{d''|i}\sum_{k|j} gik/d'' \\
&= \sum_{g=1}^{\min(N,M)}\mu(g)g\sum_{i=1}^{N/g}\sum_{j=1}^{M/g}\left(\sum_{d'''|i} d\right)\left(\sum_{k|j} k\right) \\
&= \sum_{g=1}^{\min(N,M)}\mu(g)gS_{\sigma_1}(N/g)S_{\sigma_1}(M/g)
\end{aligned}$$

注意到 $(\mu\mathrm{id})*\mathrm{id}=\varepsilon$ 可以杜教篩。複雜度一樣是 $O(N+Q\sqrt{N})$ 或 $O(M^{2/3})$ 的。更一般的，如果 $F=f*g$ 是兩個完全機性函數的卷積，則

$$\sum_{i=1}^{N}\sum_{j=1}^{M} = \sum_{d=1}^{\min(N,M)}\mu(d)f(d)g(d)S_F(N/g)S_F(M/g)$$

□

Problem 2. 給定 N,M，求滿足 $1\le a\le N, 1\le b\le M$, 且 $\gcd(a,b)$ 是平方數的數對個數。

Problem 3. 給定 N,M，求滿足 $1\le a\le N, 1\le b\le M$ 的 $ab\gcd(a,b)$ 的和。

Problem 4. 給定 N,M，求滿足 $1\le a\le N, 1\le b\le M$ 的 $F_{\gcd(a,b)}$ 的積。其中 F_n 為費氏數列，$F_0=0, F_1=1$。

Problem 5 (OJDL 7055)**.** 給定 N,M，求滿足 $1\le a\le N, 1\le b\le M$ 的 $\mathrm{lcm}(a,b)^{\gcd(a,b)}$ 的積。

0.2 模運算

0.2.1 $\mathbb{Z}$ 與 $\mathbb{Z}/\langle n\rangle$ 的結構

本節假設讀者了解基本的交換抽象代數。沒有接觸過的讀者可以選擇跳過本章或選擇先看完附錄再回來或選擇以假設附錄的定理都是對的的前提閱讀。

Property 28. $\mathbb{Z}$ 是一個 ED。

Corollary 16 (貝祖定理)**.** $\mathbb{Z}$ 是一個 PID。

Corollary 17. $\mathbb{Z}$ 的理想就恰好是 $\langle n\rangle$。

Corollary 18. $\mathbb{Z}$ 的質理想就恰好是 $\langle p\rangle$。

Corollary 19 (算術基本定理)**.** $\mathbb{Z}$ 是一個 UFD。

Property 29 (中國剩餘定理; Theorem 20)**.** 若 $a_1, \dots a_n$ 兩兩互質，則 $\mathbb{Z}/\langle a_1a_2\dots a_n\rangle \to \mathbb{Z}/\langle a_1\rangle \times \mathbb{Z}/\langle a_2\rangle \times \cdots \times \mathbb{Z}/\langle a_n\rangle$ 是一個同構。

Property 30 (Corollary 31)**.** $\mathbb{Z}/\langle p\rangle$ 是一個體。

Property 31 (Corollary 34)**.** $(\mathbb{Z}/\langle p\rangle)^\times$ 是一個循環群。

Property 32 (Lemma 28)**.** $(\mathbb{Z}/\langle n\rangle)^\times = \{a + \langle n\rangle \,|a \perp n\}$

Corollary 20 (歐拉定理)**.** 若 $x \in (\mathbb{Z}/\langle n\rangle)^\times$，則 $x^{\varphi(n)} = \overline{1}$。

Proof. Corollary 28 □

Corollary 21 (費馬小定理)**.** 若 $x \in \mathbb{Z}/\langle n\rangle$，則 $x^p = x$。

Lemma 12 (Lifting the exponent Lemma)**.** 給定質數 p，且 $p|a-b, p\not|a$（若 $p = 2$ 則要求 $4|a-b$），則 $v_p(a^n - b^n) = v_p(a-b) + v_p(n)$

Proof. 假設 $v = v_p(a-b)$，且 $a = p^vc - b$，則

$$a^k - b^k = \cdots + \frac{k(k-1)}{2}(p^vc)^2b^{k-2} + k(p^vc)b^{k-1}$$

前面的每一項都被 p^{3v} 整除。若 k 跟 p 互質則第二項至少被 p^{2v} 整除，第一項恰被 p^v 整除。因此 $v_p(a^k - b^k) = v$。若 $k = p$ 則第一項至少被 p^{2v+1} 整除（這裡若 $p = 2$ 會用到 $v > 1$），第一項恰被 p^{v+1} 整除。因此 $v_p(a^k - b^k) = v + 1$。利用歸納法，我們可以慢慢把指數的 p 搬下來（$v_p(a^{kp} - b^{kp}) = v_p(a^k - b^k) + 1 = \cdots$）。 □

Lemma 13. 若 p 是奇質數，則 $\mathbb{Z}/\langle p^n\rangle \simeq C_{p^{n-1}p}$。

Proof. 考慮 $f : (\mathbb{Z}/\langle p^n\rangle)^\times \mapsto (\mathbb{Z}/\langle p\rangle)^\times \simeq \langle x\rangle$。注意到

$$\begin{aligned}(1 + p + \langle p^n\rangle)^k = 1 + \langle p^n\rangle &\iff p^n|(1+p)^k - 1\\ &\iff n \le v_p((1+p)^k - 1) = v_p(k) + 1 \iff p^{n-1}|k\end{aligned}$$

因此 $\mathrm{ord}(1 + p + \langle p^n\rangle) = p^{n-1}$。注意到 $1 + p + \langle p^n\rangle \in \ker f$ 且 $|\ker f| = p^{n-1}$。另外，任挑一個 $y \in f^{-1}(x)$，其階會被 $\mathrm{ord}(x) = p - 1$ 整除。因此 $(\mathbb{Z}/\langle p^n\rangle)^\times$ 有兩個階各為 $p-1$ 和 p^{n-1} 的乘法子群。他們的階互質，且相乘為 $|G| = \varphi(p^n) = (p-1)p^{n-1}$。故 $(\mathbb{Z}/\langle p^n\rangle)^\times \simeq C_{p-1} \times C_{p^{n-1}} \simeq C_{(p-1)p^{n-1}}$。 □

將證明中的 $p+1$ 改成 5，我們可以得到 $p=2$ 的版本

Lemma 14. $(\mathbb{Z}/\langle 2^{n+2}\rangle)^\times \simeq C_2 \times C_{2^n}$

Proof. 同樣的，我們有 $\mathrm{ord}(5) = 2^n$。因為 $(\mathbb{Z}/\langle 2^{n+2}\rangle)^\times$ 不是循環群 ($\{\pm 1, 2^{n+1} \pm 1\} \simeq {C_2}^2$ 是一個不是循環群的子群)，因此你有該同構。 □

注意根據 $(\mathbb{Z}/\langle 2\rangle)^\times$ 是平凡群，不適用上面的討論。
至此，根據 CHT，我們有所有 $(\mathbb{Z}/\langle n\rangle)^\times$ 的結構了。

Example 7.
$(\mathbb{Z}/\langle 7122222\rangle)^\times \simeq (\mathbb{Z}/\langle 2\rangle)^\times \times (\mathbb{Z}/\langle 27\rangle)^\times \times (\mathbb{Z}/\langle 131893\rangle)^\times \simeq \times\mathbb{Z}/\langle 181392\rangle$

Lemma 15 (Hensel's lemma)**.** 若 $f \in \mathbb{Z}[x]$ 且 $a \in \mathbb{Z}$ 使得 $f(a) \equiv 0 \pmod{p^n}$, $f'(a) \neq 0 \pmod{p}$，則 $b = a - f(a)f'(a)^{-1}$ 是唯一的整數（模 p^{2n}）使得 $b \equiv a \pmod{p^n}$ 且 $f(b) \equiv 0 \pmod{p^{2n}}$

Proof. 將 f 泰勒展開得 $f(x) = f(a) + f'(a)(x-a) + (x-a)^2 g(x)$。則 $f(b) \equiv f(a) + f'(a)(b-a) \equiv 0 \pmod{p^{2n}}$。顯然這是唯一的。 □

0.2.2 擴展歐幾里得演算法

回憶前文提及的貝祖定理，它事實上告訴你，對於所有的 x, y，都存在 $a, b \in \mathbb{Z}$ 使得 $ax + by = \gcd(x, y)$。對前文提及的歐幾里德演算法做一些修改，我們亦可以算出一組 a, b。

```
1  //returns (gcd, (a, b)) such that a*x + b*y = g
2  pair<int, pair<int, int>> extgcd(int x, int y){
3      if(y){
4          auto res = extgcd(y, x%y);
5          return {res.first, {res.second.second,
6                  res.second.first - (x/y)*res.second.second}
7          };
8      }
9      else return {x, {1, 0}};
10 }
```

Listing 16: Extended Euclid's algorithm

Property 33. 上述演算法是正確的

Proof. 注意到呼叫`extgcd(y, x%y)`回傳的 $(g, (a, b))$ 滿足

$$g = ay + b(x - \langle x/y\rangle y) = bx + (a - \langle x/y\rangle b)y$$

而最一開始回傳亦滿足 $g = x = 1x + 0y$。因此該演算法是正確的。 □

Problem 6 (CF 982E). 給定一個 $n \times m$ 的矩形，以及其中的一個點 (x, y)。從 (x, y) 往 45° 往右上以每秒 $\sqrt{2}$ 格的速度移動，遇到邊界會反彈，則多久後會碰到角落？(或永不碰到？)

0.2.2.1 模逆元

前一節提到，只要 x, n 互質，就存在 y 使得 $xy = 1 \pmod{n}$。我們把該 y 稱為 x 的模逆元。注意到如果 $1 = \gcd(x, y) = ax + by$，則模 y 得到 $1 \equiv ax \pmod{p}$，因此 a 就是我們要找的模逆元。則利用上述演算法，我們得到

```
const int mod = /* mod */;
int invmod(int x, int y = mod, int a = 1, int b = 0){
    return y? invmod(y, x%y, b, a-(x/y)*b):a;
}
```

Listing 17: Computing modular Inverse

注意到乘法有時候會爆 `long long`，而最後回傳值不一定介在 0 與 `mod` 之間。

0.2.2.2 中國剩餘定理

另外，擴展歐幾里得也可以拿來做 CRT 的實際計算。若 $n \equiv r_1 \pmod{x)_1, n \equiv r_2 \pmod{x)_2$，則 $r_1 + x_1 q_1 = r_2 + x_2 q_2$，亦即 $r_1 - r_2 = -x_1 q_1 + x_2 q_2$。所以這有解若且唯若 $r_1 \equiv r_2 \pmod{\gcd(x_1, x_2)}$。此時若 $a_1 x_1 + a_2 x_2 = \gcd(x_1, x_2)$，則 $q_1 = -a_1 \frac{r_1 - r_2}{\gcd(x_1, x_2)}, q_2 = a_2 \frac{r_1 - r_2}{\gcd(x_1, x_2)}, n = r_2 + a_2 x_2 \frac{r_1 - r_2}{\gcd(x_1, x_2)}$

```
// a pair (a, m) represents "= a mod m"
pair<int, int> cht(pair<int, int> a, pair<int, int> b){
    int g = gcd(a.second, b.second), m = a.second*b.second/g;
    if((a.first - b.first) % g != 0) throw -1;
    int d = (a.first - b.first) / g;
    return {((b.first + invmod(b.second, a.second)*b.second*d)%m+m)%m, m};
}
```

Listing 18: Chinese remainder theorem

0.2.3 二次剩餘

以下的討論中，在一般情形以前，p 都是一個奇質數，且 n 跟模互質。

Definition 32. 給定 n，若存在 x 使得 $x^2 \equiv n \pmod{m}$，則我們稱 n 為（模 m）的二次剩餘。

0.2.3.1 模 p

Property 34 (歐拉判別法)**.** n 是二次剩餘若且唯若

$$n^{(p-1)/2} \equiv 1 \pmod{p}$$

Proof. ($\Rightarrow$): 若 $n = x^2$，則根據費馬小定理，

$$n^{(p-1)/2} \equiv x^{p-1} \equiv 1 \pmod{p}$$

($\Leftarrow$): 因為 $(\mathbb{Z}/\langle p\rangle)^\times$ 是循環群，二次剩餘恰有 $(p-1)/2$ 個。但滿足 $x^{(p-1)/2} = 1$ 的 x 分別至多只有 $(p-1)/2$ 個，因此這 $(p-1)/2$ 個數字恰好就是二次剩餘。 □

Lemma 16 (Cipolla)**.** 若 n 是一個二次剩餘，且 $a^2 - n$ 不是一個二次剩餘，現在考慮在 $\mathbb{Z}/\langle p\rangle\left[\sqrt{a^2-n}\right]$ 中，並令 $b = \left(a + \sqrt{a^2-n}\right)^{(p+1)/2}$。則 $b \in \mathbb{Z}/\langle p\rangle$ 滿足 $b^2 = n$。

Proof. 首先注意到 $F_{p^2} = \mathbb{Z}/\langle p\rangle\left[\sqrt{a^2-n}\right] \simeq (\mathbb{Z}/\langle p\rangle[x])/(x^2 - a^2 + n)$ 是一個體。因此只要存在 b 滿足 $b^2 = n$，則 $b, -b$ 就是在 F_{p^2} 中 n 唯二的平方根。亦即 $b \in F_p$ 就是所求的平方根。

$$\begin{aligned} b^2 = \left(a + \sqrt{a^2-n}\right)^{p+1} &= \left(a + \sqrt{a^2-n}\right)\left(a + \sqrt{a^2-n}\right)^p \\ &= \left(a + \sqrt{a^2-n}\right)\left(a^p + \sqrt{a^2-n}^p\right) \\ &= \left(a + \sqrt{a^2-n}\right)\left(a + \sqrt{a^2-n}(a^2-n)^{(p-1)/2}\right) \\ &= \left(a + \sqrt{a^2-n}\right)\left(a - \sqrt{a^2-n}\right) = n \end{aligned}$$

□

Lemma 17. 令 n 是一個給定的二次剩餘，則共有 $(p-1)/2$ 個 a 使得 $a^2 - n$ 不是二次剩餘。

Proof. 首先注意

$$\sum_{i=0}^{p-1} i^k \equiv \begin{cases} -1 & p-1 | i \\ 0 & \text{otherwise} \end{cases} \pmod{p}$$

故

$$\sum_{i=0}^{p-1} (i^2 - n)^{(p-1)/2} \equiv \sum_{i=0}^{p-1} i^{p-1} - pn \equiv -1 \pmod{p}$$

在所有的 i 中，恰有兩個使得 $(i^2-n)^{(p-1)/2} = 0$，剩下 $p-2$ 個 ± 1 的總和要是 $-1 \pmod{p}$，因此一定是 $(p-3)/2$ 個 1，$(p-1)/2$ 個 -1。也就是使 $i^2 - n$ 是二次剩餘的 i 有 $(p+1)/2$ 個 □

因此以上定理保證隨機選取 a，可以在平均兩次的時間找到非二次剩餘的 $a^2 - n$。每次檢查需要歐拉判別法中快速冪的 $O(\log p)$，而找到後需要花 $O(\log p)$ 得到答案。故有

Theorem 8 (Cipolla). 給定 n 為模 p 的二次剩餘，可以在最差 $O(p \log p)$，期望 $O(\log p)$ 的時間複雜度內找出一個 x 使得 $x^2 \equiv n \pmod{p}$。

以下是一個可行的實作

```
using Random::randInt

struct Fp2{
    static int p, c;
    int a, b; // a + b sqrt(c) mod p
    Fp2(int aa, int bb = 0) : a{aa%p}, b{bb%p} {
        if(a < 0)a += p;
        if(b < 0)b += p;
    }
    Fp2 friend operator+(Fp2 n, Fp2 m){
        return Fp2(n.a+m.a, n.b+m.b);
    }
    Fp2 friend operator*(Fp2 n, Fp2 m){
        return Fp2(n.a*m.a + n.b*m.b%p*c, n.a*m.b + n.b*m.a);
    }
    bool operator==(int m){ return a == m%p && b == 0; }
};

int Fp2::p = 0, Fp2::c = 0;

Fp2 fpow(Fp2 a, int b){
    Fp2 ans = 1;
    do if(b&1) ans = ans * a;
    while(a = a*a, b >>= 1);
    return ans;
}

bool is_residue(int n, int p){
    return fpow(n, (p-1)/2, p) == 1;
}

int cipolla(int n, int p){
    Fp2::p = p;
    assert(n > 0);
    int a = 0;
    do a = randInt(1, p-1);
    while(is_residue(a*a-n, p));
    if(a*a%p == n)return a;

    Fp2::c = (a*a-n+p)%p;
    auto res = fpow(Fp2(a, 1), (p+1)/2);
```

```
42      if(res.b) throw -1;
43      else return res.a;
44  }
```

Listing 19: Cipolla's algorithm

在這裡再提另一個比較不直覺的演算法

```
1   using Random::randInt;
2
3   bool is_residue(int n, int p){
4       return fpow(n, (p-1)/2, p) == 1;
5   }
6
7   int tonelli_shanks(int n, int p){
8       int a = 0, b = p-1, c;
9       while(b%2 == 0) b/=2, a++;
10      do c = randInt(2, p-1);
11      while(is_residue(c, p));
12
13      int d = a, e = fpow(c, b, p), f = fpow(n, b, p), g = fpow(n, (b+1)/2, p);
14      while(true){
15          if(f == 1) return g;
16          int i = 0;
17          for(int t = f; i < d && t != 1; i++, t=t*t%p);
18          if(i == d) throw -1;
19
20          int h = fpow(e, 1<<d-i-1, p);
21          d = i, e = h*h%p, f=f*e%p, g=g*h%p;
22      }
23  }
```

Listing 20: Tonelli-Shanks algorithm

Lemma 18. 上述演算法的輸出 g 滿足 $g^2 \equiv n \pmod{p}$。

Proof. 我們有以下不變量：$e^{2^{d-1}} \equiv -1, f^{2^{d-1}} \equiv 1, g^2 \equiv fn$。
注意到每次選取的 i 是使 $f^{2^i} \equiv 1$ 的最小正整數。 □

Corollary 22. 給定 n 為模 p 的二次剩餘，Tonelli-Shanks 演算法可以在最差 $O(p \log p)$，期望 $O(\log^2 p)$ 的時間複雜度內找出一個 x 使得 $x^2 \equiv n \pmod{p}$。

Proof. 一開始亂戳恰有 1/2 的機率戳到非二次剩餘，因此該步是最差 $O(p\log n)$，平均 $O(\log n)$ 的。後面的迴圈每次 d 會從 a 開始嚴格遞減，且每個迴圈需要快速冪的 $O(\log n)$。因此總期望複雜度是 $O(\log n + a\log n) = O(\log^2 n)$ 的。 □

Remark 7. 這個最差複雜度在多筆詢問相同 p 的詢問是可以被均攤掉的。

Remark 8. 若亂戳改為從 $2, 3, \ldots$ 開始往上戳，最差複雜度是 $O(\sqrt[4]{p}\log p)$ 的。而且如果擴展黎曼猜想是對的，這個最差複雜度可以被壓到 $O(\log^3 p)$。

Remark 9. 目前是否有（不倚靠猜想的）決定性多項式的開根號演算法仍然是公開問題。

於是我們會做奇質數的情形了。

0.2.3.2 模 p^k

現在我們考慮質數冪次。根據其乘法群的性質，二次剩餘必有兩個平方根，而由 Hensel's lemma，我們有

Property 35. $y^2 \equiv n \pmod{p^k}$ 有解若且唯若 $x^2 \equiv n \pmod{p}$ 有解。

Property 36. 若 $x^2 \equiv n \pmod{p}$，則 $y = x^s n^{(q-2s+1)/2}$ 是 $y^2 \equiv n \pmod{p^k}$ 的一個解，其中 $q = p^k, s = p^{k-1}$。

Proof. 首先，透過 LTE，$x^2 \equiv n \pmod{p}$ 可以推得 $(x^2)^s \equiv n^s \pmod{p^k}$。故

$$(x^s n^{(q-2s+1)/2})^2 = (x^2)^s n^{q-2s+1} \equiv n^{q-s+1} \equiv n \pmod{p^k}$$

注意 $\varphi(p^k) = q - s$，因此最後一個等號成立。 □

0.2.3.3 模 2^k

最後是模 2^k 的情形。我們只考慮 $k \geq 3$。此時 x^n 的微分是 $2x$，因此可能有非零根，不適用 Hensel's lemma。而從 $\mathbb{Z}/\langle 2^k\rangle$ 的乘法單位群結構可以看出。

Property 37. 在模 $2^k \geq 8$ 的情況下，共有 2^{n-3} 個奇數是二次剩餘，而且每個奇二次剩餘恰有 4 個平方根。

Property 38. 在模 2^k 的情況下，n 是奇二次剩餘若且唯若 $n \equiv 1 \pmod{8}$

Property 39. 在模 $2^k \geq 8$ 的情況下，若 $x^2 \equiv n \pmod{2^k}$，則 $y = x - (x^2 - n)/2$ 滿足 $y^2 \equiv n \pmod{2^{k+1}}$。

Proof.

$$y^2 = \left(x - \frac{x^2-n}{2}\right)^2 = x^2 - x(x^2-n) + \left(\frac{x^2-n}{2}\right)^2 = a - (x^2-a)(x-1) + \left(\frac{x^2-a}{2}\right)^2$$

因為 $x-1$ 是奇數，第二項被 x^{k+1} 整除，且第二項被 2^{2k-2} 整除。因為 $k \geq 3$，所以 $y^2 \equiv a \pmod{2^{k+1}}$。 □

所以我們可以花 $O(k)$ 的時間從模 8 的情形慢慢往上推。

0.2.3.4 一般情形

注意這裡的 p 可以是 2。

Property 40. n 在模 p^k 底下是二次剩餘，若且唯若 $v_p(n) \geq k$ 或 $n = n'p^{v_p(n)}$ 滿足 n' 是模 p 的二次剩餘且 $2|v_p(n)$。

Proof. 若 $x^2 \equiv n \pmod{p^k}$，則 $p|x^2$，故 $p|x$，亦即 $p^2|n^2$，且我們有 $(x/p)^2 \equiv (n/p)^2 \pmod{p^k}$。利用歸納法得證。 □

現在如果模任意整數，則我們可以先對他的每個因數做事，再 CRT 回來，得到以下的演算法：

```
int quadratic_residue_2k(int n, int k){
    n%=(1<<k);
    if(n == 1) return 1;
    if(k <= 3) throw -1;
    int x = quadratic_residue_2k(n, k-1);
    return ((x - (x*x-n)/2)%(1<<k) + (1<<k))%(1<<k);
}

int quadratic_residue_pk(int n, int p, int k, int q){
    if(n%q == 0) return 0;
    int t = 1;
    while(n%p == 0){
        n/=p;
        if(n%p) throw -1;
        n/=p, t*= p;
    }

    if(p == 2) return t*quadratic_residue_2k(n, k);
    int x = quadratic_residue_p(n%p, p), r = q/p;
    return t*(fpow(x, r, q)*fpow(n, (q-2*r+1)/2), q).a%q;
}

// returns [{p, v_p(n), p^{v_p(n)}},... ]
vector<array<int, 3>> factorize(int n);

// returns -1 if n is not a residue
int quadratic_residue(int n, int m){
    auto ps = factorize(m);
    pair<int, int> ans = {0, 1};
    try{
        for(auto p: ps)
            ans = cht(ans, {quadratic_residue_pk(n, p[0], p[1], p[2]), p[2]});
```

```
33          return ans.first;
34      }catch(signed e){
35          return -1;
36      }
37  }
```

Listing 21: Calculating square root mod m

注意到其中的 `cht` 需要用到Listing 18的函式。

Corollary 23. 若已知 n 的因式分解，可以在期望 $O(\log n)$ 的時間找出一個 m 模 n 的平方根。

Proof. 假設 $n = \prod q_i = \prod {p_i}^{k_i}$。若 $p_i = 2$，則處理該部分的問題需要 $O(k_i) = O(\log q_i)$。若 p_i 是奇質數，則需要 Cipolla 的 $O(\log q_i)$ 跟快速冪的 $O(\log q_i)$。CRT 的部分亦是 $O(\log p_i)$ 的。故總時間是 $\sum_{p_i} O(\log q_i) = O(\log n)$ 的。 □

0.2.4 高次剩餘

處理完平方根的情形，我們接著處理高次的根號。首先注意到

Property 41. 如果 r 跟 $\varphi(m)$ 互質，則 $x = n^{r^{-1}}$ 滿足 $x^r \equiv n \pmod m$，其中 r^{-1} 為 r 模 $\varphi(m)$ 的模逆元。

Lemma 19. 若 n, m 互質，則 n 是模 m 的 r 次剩餘若且唯若 n 是模 m 的 $\gcd(r, \varphi(m))$ 次剩餘。

Proof. $\Rightarrow$ 是顯然的。對於 $\Leftarrow$，我們只要檢查質數冪次即可，因為 $p^k | m$ 可以推出 $\gcd(r, \varphi(p^k)) | \gcd(r, \varphi(m))$，而且中國剩餘定理告訴你 n 是模 m 的 r 次剩餘若且唯若 n 是模所有 p^k 的 r 次剩餘。
對於 p^k，$(\mathbb{Z}/\langle p^k \rangle)^\times$ 根本是一個循環群，所以若 g 是生成元，則 $n = g^o$ 是 r 次剩餘若且唯若存在 a, b 使得 $o + b\varphi(p^k) = ar$，若且唯若 $\gcd(\varphi(p^k), r) | o$。因此事情根本只跟 $\gcd(\varphi(p^k), r)$ 有關。 □

從以上證明也可以看出

Property 42. 如果 r, s 互質，且 n, m 互質，則 n 是模 m 的 rs 次剩餘若且唯若 n 同時是模 m 的 r, s 次剩餘。

Property 43. 若 $r | p-1, p \nmid n$，則 n 是模 p^k 的 r 次剩餘若且唯若 $n^{(p-1)/r} \equiv 1 \pmod p$

Proof. 若 $k = 1$，則可以利用與歐拉判別法一樣的證明得到。對 $k > 1$，注意到若 $x \not\equiv 0 \pmod p$，則 $\frac{d}{dx} x^r - n = r x^{r-1}$ 模 p 非零。根據 hensel's lemma 得證。 □

Property 44. 若 $p \nmid n$，則 n 是模 p^k 的 p^l 次剩餘（$l < k$）若且唯若 n 是模 p^{l+1} 的 p^l 次剩餘若且唯若 $n^{p^l} \equiv n \pmod{p^{l+1}}$

Proof. $(1 \Leftrightarrow 2)$: 那個 $\mathbb{Z}/\langle p^k \rangle \to \mathbb{Z}/\langle p^{l+1} \rangle$ 的投影根本就只是把指數取 $\varphi(p^{l+1})$ 的餘數而已，不會改變指數是不是 p^l 的倍數。
$(2 \Leftrightarrow 3)$ 模 p^{l+1} 的 p^l 次剩餘只有 $\varphi(p^{l+1})/p^l = p-1$ 種，其實就是 $1^{p^l}, \dots (p-1)^{p^l}$。所以你只要檢查 n 是不是 n^{p^l} 即可。 □

接著我們開始介紹如何實際計算。首先根據中國剩餘定理，可以把模數拆成質數冪次的乘積並分開做。另外，我們也可以把 r 拆成質數的乘積並依序開根號。而根據以上的討論我們已剩下 r 的質因數整除 p 或等於 p 的情形了。若等於 p，我們可以從 p^2 的情形慢慢往上推，亦即

Property 45. 在模 $p = 2, k > 2$ 或 $p > 2, k > 1$ 的情況下，若 $x^p \equiv n \pmod{p^k}$，則 $y = x - (x^p - n)/p$ 滿足 $y^p \equiv n \pmod{p^{k+1}}$。

Proof.

$$\begin{aligned} y^p &= (x - (x^p - n)/p)^p \\ &\equiv x^p + x^{p-1}(x^p - n) + \frac{p(p-1)}{2}((x^p - n)/p)^2 x^{p-2} \\ &= n + (x^{p-1} - 1)(x^p - n) + \frac{p(p-1)}{2}((x^p - n)/p)^2 x^{p-2} \equiv n \pmod{p^{k+1}} \end{aligned}$$

注意到第二項根據費馬小定理被 $p \cdot p^k$ 整除，第二項被 p^{2k-1} 整除（若 $p = 2$ 則 $2 \nmid \frac{p(p-1)}{2}$，會再少一）。因此只剩第一項。 □

而如果 $r | p-1$，我們有以下 Cipolla 的推廣：

Theorem 9 (Cipolla–Lehmer)**.** 若 n 是一個模 p 的 r 次剩餘，且 $r | p-1$ 是一個質數，$a^r - n$ 不是一個 r 次剩餘，現在考慮在 $\mathbb{Z}/\langle p \rangle \left[\sqrt[r]{a^r - n}\right]$ 中，並令 $b = \left(a - \sqrt{a^2 - n}\right)^{\frac{p^r - 1}{r(p-1)}}$。則 $b \in \mathbb{Z}/\langle p \rangle$ 滿足 $b^r = n$。

Proof. 首先注意 $\mathbb{Z}/\langle p \rangle \left[\sqrt[r]{a^r - n}\right] \simeq \mathbb{Z}/\langle p \rangle [X]/\langle X^r - a^r + n \rangle$ 是一個體，且 $\omega = \sqrt[r]{a^r - n}^{p-1} = (a^r - n)^{(p-1)/r}$ 的階是 r（這裡用到 r 是質數）。因此 $\omega^p = \omega$，且

$$(\sqrt[r]{a^r - n})^{p^i} = \sqrt[r]{a^r - n}\left((\sqrt[r]{a^r - n})^{p-1}\right)^{1+p+\cdots+p^{i-1}} = \omega^i \sqrt[r]{a^r - n}$$

故

$$b^r = \left(a - \sqrt{a^2 - n}\right)^{1+p+p^2+\cdots+p^{r-1}} = \prod_{i=0}^{r-1}(a - \omega^i \sqrt[r]{a^r - n}) = a^r - (\sqrt[r]{a^r - n})^r = n$$

□

Lemma 20 ([8] Theorem 2)**.** 固定 $r | p-1$，則對於任何 n，使 $a^r - n$ 是 r 次剩餘的 a 至多只有 $p/r + O(\sqrt{p})$ 個。

Proof. 細節待補，想法是找一個多項式 $R(x)$ 使得當 $x \in F_p$ 滿足 $x^r - n$ 是 r 次剩餘時（亦即 $(x^r - n)^{(p-1)/r} = x^p - x = 0$ 時），R 在 x 都是一個 M 階零點。（在該處泰勒展開的最低 M 項為零）。因此滿足該條件的 x 至多只有 $\deg R/M$ 個。藉由選取好的 R 我們便可以得到上述上界。 □

綜合以上討論，隨機戳到非 r 次剩餘的機率是 $1 - r^{-1} - O(1/\sqrt{p})$ 的。因此對於每一個 r 都只要試 $O(1)$ 次

Corollary 24. 對固定 r，我們都有一個期望複雜度 $O(\log p)$ 的開 r 次方根演算法。

Remark 10. 注意上述 $O(\log p)$ 的常數與 r 有關。如果不考慮隨機部分，那大約是 $O(r^3 \log p)$ 的。

不只 Cipolla 演算法可以被推廣，Tonelli-Shanks 也可以被推廣，稱為 Adelman-Manders-Miller 演算法 [1]。

```
#include<unordered_map>
using Random::randInt;

int invmod(int a, int mod);

bool is_rth_residue(int n, int p, int r){
    return fpow(n, (p-1)/r, p) == 1;
}

int adelman_manders_miller(int n, int p, int r){
    int a = 0, b = p-1, c;
    vector<int> r_power; r_power.push_back(1);
    while(b%r == 0) b/=r, a++, r_power.push_back(r*r_power.back()%p);
    do c = randInt(2, p-1);
    while(is_rth_residue(c, p, r));

    int d = a, l = (r-invmod(b, r))%r, e = fpow(c, b, p),
        f = fpow(n, l*b%(p-1), p), g = fpow(n, (l*b+1)/r, p),
        omega = fpow(c, (p-1)/r, p);

    std::unordered_map<int, int> omega_power;
    for(int i = 0, o = 1; i < r; i++, o = o*omega%p) omega_power[o] = i;

    while(true){
        if(f == 1) return g;
        int i = 0, prev;
        for(int t = f; i < d && t != 1; i++, prev=t, t=fpow(t, r, p));
        if(i == d) throw -1;

        int j = omega_power[invmod(prev, p)],
```

```
31              h = fpow(e, j*r_power[d-i-1], p);
32          d = i, e = fpow(e, r_power[d-i], p), f=f*h%p*h%p, g=g*h%p;
33      }
34  }
```

Listing 22: Adelman-Manders-Miller algorithm

Lemma 21. 上述演算法的輸出 g 滿足 $g^r \equiv n \pmod{p}$。

Proof. 注意到 $\omega \neq 1$ 滿足 $\mathrm{ord}(\omega) = r$。我們有以下不變量：
$e^{r^{d-1}} \equiv \omega, f^{r^{d-1}} \equiv 1, g^r \equiv fn$。
注意到每次選取的 i 是使 $f^{r^i} \equiv 1$ 的最小正整數。
此時 $\mathrm{ord}(f^{r^{i-1}}) = r$，因此存在 j 使得 $\omega^j f^{r^{i-1}} = 1$。 □

Corollary 25. 給定 n 為模 p 的 r 次剩餘，Adelman-Manders-Miller 演算法可以在最差 $O(p \log p)$，期望 $O\left(r + \frac{\log^2 p}{\log r}\right)$ 的時間複雜度內找出一個 x 使得 $x^r \equiv n \pmod{p}$。

Proof. 一開始需要 $O(\log_r(p) + r)$ 的時間，而迴圈總共跑 $O(\log_r(p))$ 次，每次需要跑 $O(\log_r(p) \log r + \log p)$ 的時間，因此總共是 $O\left(r + \frac{\log^2 p}{\log r}\right)$ 的。 □

Remark 11. 如果 $r = \Omega(\sqrt{p})$ 的話，直接做離散對數再把指數除以 r 會比上述兩種演算法都來得快。

0.2.5 原根

以下的討論中，p 都是一個奇質數。
我們在Subsection 0.2.1中已經證明過，

Property 46. $(\mathbb{Z}/\langle n \rangle)^\times$ 是循環群若且唯若 $n = p^k$ 或 $n = 2p^k$ 或 $n = 4$，其中 p 是任意奇質數。

我們稱該循環群的生成元為原根。首先先考慮 $(\mathbb{Z}/\langle p \rangle)^\times$ 的情形。
根據簡單分析，一個循環群 C_{p-1} 的生成元有 $\varphi p - 1$ 個。因此

Property 47. 平均 $O(\log \log p)$ 個數中有一個是模 p 的原根。

至於原根的檢查，你只要確認 $o(g)$ 是不是 $p-1$ 即可，所以你只要確認對於 $p-1$ 的所有質因數 p_i，我們都有 $g^{(p-1)/p_i} \not\equiv 1 \pmod{p}$ 即可。因此

Theorem 10. 若以知 $p-1$ 的因式分解，則可以在期望 $O(\log^2 p)$ 的時間找到 $\mathbb{Z}/\langle p \rangle$ 的一個原根。

Proof. 上述的時間複雜度是 $O(\log \log p) O(\omega(p-1)) O(\log p) = O(\log^2 p)$ 的。 □

以下是範例實作：

```cpp
using Random::randInt;

// returns list of prime factors of n
vector<int> prime_factors(int n);

int primitive_root(int p){
    if(p < 5) return p-1;
    auto ps = prime_factors(p-1);
    while(true){
        int g = randInt(2, p-2);
        for(auto pi: ps)
            if(fpow(g, (p-1)/pi) == 1)
                goto not_root;
        return g;
        not_root:;
    }
}
```

Listing 23: Obtaining an primitive root

Remark 12. 若 ERH 是對的，則最小原根是 $O(\log^6 p)$ 的，於是我們有一個多項式的暴力演算法。否則目前是否有決定性的多項式演算法仍是一個公開問題。

而質數冪次的情形我們一樣有 $\varphi(p^k) = \Theta(p^k)$，因此以上的估計都是對的。但我們還有以下的刻劃

Lemma 22 ([5] Lemma 1.4.5). 若 g 是 $\mathbb{Z}/\langle p\rangle$ 的原根，則若 $g^{p-1} \not\equiv 1 \pmod{p^2}$，則 g 是模 p^k 的原根。否則 $g+p$ 是模 p^k 的原根

Proof. 如果 g 是 $\mathbb{Z}/\langle p\rangle$ 的原根，則對於所有 $p_i | p-1$，我們都有

$$g^{\varphi(p^k)/p_i} = (g^{(p-1)/p_i})^{p^{k-1}} \equiv g^{(p-1)/p_i} \not\equiv 1 \pmod{p}$$

因此我們只要檢查 $g^{\varphi(p^k)/p} = g^{p^{k-2}(p-1)}$ 在模 p^k 下是不是 1。注意到

$$a^p \equiv 1 \pmod{p^a} \iff a \equiv 1 \pmod{p^{a-1}}$$

(你可以透過 $(\mathbb{Z}/\langle p^k\rangle)^\times$ 的結構或 LTE 看出。) 因此我們只要檢查 $g^{p-1} \not\equiv 1 \pmod{p^2}$ 即可。注意到

$$(g+p)^{p-1} \equiv g^{p-1} + (p-1)pg^{p-2} \pmod{p^2}$$

因此如果 $g^{p-1} \equiv 1 \pmod{p^2}$，則 $(g+p)^{p-1} \not\equiv 1 \pmod{p^2}$。 □

在模 2^k 底下，是沒有原根的，但是根據Lemma 14的證明，我們可以看出

Property 48. 對於所有奇數 n，存在 b 使得 $5^b \equiv \pm n \pmod{2^k}$。

Proof. 我們有 $\mathrm{ord}(5) = 2^{k-2}$ 且 $-1 \notin \langle 5 \rangle$（LTE 告訴你 $5^{2^{k-3}} \equiv 2^{k-1} + 1 \pmod{2^k}$）。因此 $(\mathbb{Z}/\langle 2^k \rangle) \simeq \langle -1 \rangle \times \langle 5 \rangle$。 □

0.2.6 離散對數

在本節我們將探討以下問題：給定 a, c, m，請找出最小的 b 使得 $a^b \equiv c \pmod{m}$。
首先考慮 a, c 與 m 互質的情形。此時如果存在解，則根據歐拉定理必存在解滿足 $0 \le b < \varphi(m)$。令 K 為一個常數，則我們可以考慮 $b = qK + r$，此時 $(a^K)^q \equiv a^{-r}c \pmod{m}$。因為左右兩側的取值只有 $O(m/K)$ 與 $O(K)$ 種，因此取 $K \sim \sqrt{m}$ 我們得到一個 $O(\sqrt{n})$ 的演算法。
這個演算法稱為大步小步（baby step giant step），你可以把 a^K 想像成大步，a^{-1} 想像成小步。用一個 `unordered_map` 儲存兩側的值即可。這樣的複雜度是 $O(\sqrt{m})$ 的。

```cpp
#include<unordered_map>
#include<cmath>

int invmod(int a, int mod);

int discrete_log(int a, int c, int m){
    a%=m, c%=m;

    std::unordered_map<int, vector<int>> mp, mpk;
    int k = std::sqrt(m), ak = fpow((invmod(a, m)+m)%m, k, m);
    for(int i = 0, j = 1, jk = c; i < k+1; i++, j=j*a%m, jk=jk*ak%m){
        if(mp.find(j) == mp.end())mp[j] = i;
        if(mpk.find(jk) == mpk.end())mpk[jk] = i*k;
    }

    int ans = m;
    for(auto& p: mp)
        if(mpk.find(p.first) != mpk.end())
            ans = std::min(ans, p.second + mpk[p.first]);
    if(ans == m) throw -1;
    return ans;
}
```

Listing 24: Big step giant step when a, c, m are coprime

至於 $\gcd(a, m) > 1$ 的情形，對於每一個 $p|m$，p 必須同時整除 a, c 或同時不整除。且若 $v_p(c) > v_p(m)$，則我們要求 $b \ge \frac{v_p(m)}{v_p(a)}$，並把 m 換成 $m/\gcd(p, m)$。至

於 $v_p(c) < v_p(m)$ 時，會有 $b = \frac{v_p(c)}{v_p(a)}$。若他不是整數，或代入檢查發現錯誤則無解。結束後，我們接著對 $a \equiv c \pmod m$ 做大步小步即可。注意到此時為了找到滿足大於所有 $\frac{v_p(m)}{v_p(a)}$ 的解中最小的 b 需要費一番功夫（例如雙指針掃一下）。實際寫起來長這樣

```cpp
#include<unordered_map>
#include<cmath>

// returns {limit, ans, m', phi(m')}
array<int, 4> check_not_coprime(int a, int c, int m){
    int limit = 0, ans = 0, mx = m, phi = 1;
    for(int i = 2; i && m > 1; i++) {
        if(m%i)continue;
        int va = 0, vc = 0, vm = 0;
        while(m%i == 0)vm++, m/=i, phi*=i;
        while(a%i == 0)va++, a/=i;
        while(c%i == 0)vc++, c/=i;
        phi = phi/i*(i-1);
        if(!va && !vc) continue;
        if(!va || !vc) throw -1;
        if(vc >= vm){
            limit = std::max(limit, (vm-1)/va+1);
            phi = phi*i/(i-1);
            while(mx%i == 0) mx/=i, phi/=i;
        }else{
            va = std::min(va, vm); vc = std::min(vc, vm);
            if(vc%va) throw -1;
            if(!ans) ans = vc/va;
            if(ans != vc/va) throw -1;
        }

        if(i*i>m)i = m-1;
    }
    return {limit, ans, mx, phi};
}

int discrete_log(int a, int c, int m){
    a%=m, c%=m;
    if(c == 1)return 0;
    if(!a)
        if(!c) return 1;
        else throw -1;
    if(!c){
        for(int i = 0, k = 1; (1<<i) <= m; i++, k = k*a%m)
            if(k == c) return i;
```

```
41          throw -1;
42      }
43
44      auto res = check_not_coprime(a, c, m);
45
46      if(res[1])
47          if(fpow(a, res[1], m) == c)
48              return res[1];
49          else throw -1;
50      m = res[2];
51
52      std::unordered_map<int, vector<int>> mp, mpk;
53      int k = std::sqrt(res[3]), ak = fpow(a, res[3]-k, m);
54      for(int i = 0, j = 1, jk = c; i < 2*k+1; i++, j=j*a%m, jk=jk*ak%m)
55          mp[j].push_back(i), mpk[jk].push_back(i*k);
56
57
58      int ans = 3*m, limit = res[0];
59      for(auto& p: mp){
60          if(mpk.find(p.first) == mpk.end())continue;
61          auto& p1 = p.second, &p2 = mpk[p.first];
62          for(int i = 0, j = p2.size()-1; i < p1.size(); i++)
63              while(j>=0 && p1[i]+p2[j]>=limit)
64                  ans = std::min(ans, p1[i]+p2[j]), j--;
65      }
66      if(ans == 3*m) throw -1;
67      printf("%d\n", ans);
68      return ans;
69  }
```

Listing 25: Discrete log

Corollary 26. 離散對數的問題可以在 $O(\sqrt{n})$ 內解開。

Remark 13. 目前最快的離散對數演算法是次指數的。至於有沒有多項式的非決定性演算法仍是公開問題，更遑論決定性的。

0.3 附錄：代數

本節定義的群環體等皆是交換的版本，因此與一般使用的定義皆有出入。在其他地方使用時請小心！

0.3.1 么半群

Definition 33 (么半群；monoid)**.** 一個么半群 $(M, \star)$ 指的是一個非空集合 M 與其上的一個運算 $\star : G \times G \to G$ 若滿足

1. (結合律) $a \star (b \star c) = (a \star b) \star c$。
2. (交換律) $a \star b = b \star a$。
3. (單位元) 存在 $e \in M$ 使得對於所有 $a \in G$，我們有 $e \star a = a$。

以下定義與性質都在一個么半群 $(M, \star)$ 中

Definition 34 (逆元)**.** 若 $x \star y = e$，則我們稱 x 為 y 的逆元。(同樣的 y 也是 x 的逆元）我們把存在逆元的元素稱為可逆元素。

Definition 35 (單位群；group of units)**.** $M^\times$ 表示 M 中可逆元素的集合，稱為 M 的單位群。

Property 49. 可逆元素的逆元唯一。

Proof. 若 y, z 皆為 x 逆元，則 $y = y \star (x \star z) = (y \star x) \star z = z$。 □

我們把 x 的逆元記做 $x^{\star -1}$。

Property 50. $(x^{\star -1})^{\star -1} = x$

Property 51. $(x \star y)^{\star -1} = y^{\star -1} \star x^{\star -1}$

Proof. $(x \star y) \star (y^{\star -1} \star x^{\star -1}) = x \star (y \star y^{\star -1}) \star x^{\star -1} = x \star x^{\star -1} = e$ □

因此

Corollary 27. $(M^\times, \star)$ 是一個么半群。

Proof. 前一個性質告訴你 $\star$ 事實上可以被視為一個 $M^\times \times M^\times \to M^\times$ 的運算。結合律與交換律當然還是會對，而 $e^{\star -1} = e$ 是可逆的，因此 $M^\times$ 亦有單位元。 □

Example 8. 請判斷以下是不是么半群

1. $(\mathbb{N}, +)$
2. $(\mathbb{N}, \cdot)$
3. $(\mathbb{Z}, +)$
4. $(\mathbb{N}^*, +)$
5. $(\varnothing, \varnothing)$
6. $(\mathcal{P}(S), \cup)$
7. $(\mathbb{N}^*, (x, y) \mapsto xy^x)$

8. $(\mathbb{R}, (x,y) \mapsto y)$
9. $(\mathbb{R}, (x,y) \mapsto xy + x + y)$
10. $(M_{n\times n}(\mathbb{R}), \cdot)$

0.3.2 群

Definition 36 (群)**.** 若一個么半群 $(G, \star)$ 滿足 $G^\times = G$（亦即所有元素都有逆元），則我們稱他為一個群。

Property 52. 對於任何么半群的單位群是一個群。

Proof. 我們其實只要證明 $(M^\times)^\times = M^\times$ 即可。但是如果 $x \in M^\times$，則 $x^{\star -1} \in M^\times$。所以在 M 可逆的元素都在 $M^\times$ 可逆。 □

Example 9. 請寫出Example 8中么半群的對應單位群。

Definition 37 (（群的）階；order)**.** $|G|$ 是 G 的集合的大小。

Definition 38 (有限群；finite group)**.** 階有限的群稱為有限群。

Definition 39 (（元素的）階；order)**.** $g \in G$ 的階 $o(g)$ 是最小的正整數 n 使 $g^n = e$。

當然元素的階可能是無限大（不存在）。

Property 53. 有限群的元素都存在階

Proof. $\{x, x^{\star 2}, x^{\star 3}, \dots\}$ 是有限集。根據鴿籠原理存在 $x^{\star i} = x^{\star j}$。則 $x^{\star i-j} = e$。 □

Property 54. 若 $x^{\star n} = e$，則 $\mathrm{ord}(x) | n$

Proof. 令 n 除以 $\mathrm{ord}(x)$ 的餘數是 r。注意到 $x^{\star r} = x^{\star n}(x^{\star \mathrm{ord}(x)})^{\star q}$，因此若 $r \neq 0$，則 $r > \mathrm{ord}(x)$，與餘數的定義矛盾。 □

Definition 40 (子（么半）群；submonoid, subgroup)**.** 給定（么半）群 $(G, \star)$，和 G 的子集 H，若 $\star$ 也是 H 上的運算（$\star$ 對 H 是封閉的），且 $(H, \star)$ 是一個（么半）群，則稱 H 為 G 的子（么半）群，記作 $H \trianglelefteq G$。

Definition 41. $\langle a \rangle$ 是包含 $a \in G$ 的所有子群的交集；$\langle S \rangle$ 是包含 $S \subseteq G$ 的所有子群的交集。稱為被 a（S）生成的子群。

Property 55. $\langle a \rangle = \{e, a, a^{\star 2}, a^{\star 3}, \dots, a^{\star -1}, a^{\star -2}, a^{-\star -3}, \dots\}$

Proof. 首先檢查 $\{e, a, a^{\star 2}, a^{\star 3}, \dots, a^{\star -1}, a^{\star -2}, a^{-\star -3}, \dots\}$ 的確是 G 的子群。接著你發現所有包含 a 的子群都包含他。 □

Property 56. $|\langle a\rangle| = \mathrm{ord}(a)$

Definition 42 (有限生成)**.** 若存在有限的集合 $S \subseteq G$ 使得 $G = \langle S\rangle$，則我們稱 G 是有限生成的

Definition 43 (循環群)**.** 若存在 $a \in G$ 使得 $G = \langle a\rangle$，則稱 G 為一個循環群

Definition 44. $C_n = \langle a | a^n = 1\rangle = \{e, a, a^2, \ldots, a^{n-1}\}$ 是正 n 邊形的旋轉對稱群。

Property 57. $(\mathbb{Z}, +)$ 和所有 C_n 就是所有的循環群

Proof. 考慮 e 的階，如果他沒有階，則他長得像 $(\mathbb{Z}, +)$。否則他長得像 $C_{\mathrm{ord}(e)}$。 □

0.3.3 等價、同餘和商群

Definition 45 (等價關係；equivalence relation)**.** 一個集合上的等價關係是一個滿足以下三點的二元關係。

1. (自反律) $a \sim a$
2. (對稱律) $a \sim b \Leftrightarrow b \sim a$
3. (遞移律) $a \sim b \wedge b \sim c \Rightarrow a \sim c$

Example 10. 給定子集 H，$a \sim_H b \iff a \star b^{\star -1} \in H$ 是等價關係

Definition 46 (等價類；equivalence class)**.** $\overline{x} = \{g \in G | g \sim x\}$

例如 $\sim_H$ 底下 x 的等價類是 $\{x \star h | h \in H\}$。此集合稱為

Definition 47 (陪集；coset)**.** $x \star H := \{x \star h | h \in H\}$

等價關係最重要的，就是他會將我們考慮的集合劃分成幾個互斥的等價類，亦即

Property 58. 若 $x \sim y$ 則 $\overline{x} = \overline{y}$；若 $x \not\sim y$ 則 $\overline{x} \cap \overline{y} = \varnothing$。

把這些等價類收集起來稱為商集

Definition 48 (商集；quotient group)**.**
$G/\sim := \{\overline{g} | g \in G\}, \quad G/H := G/\sim_H = \{g \star H | g \in G\}$

Definition 49 (指數；index)**.** $[G : H] := |G/H|$

注意到每個陪集都一樣大，因此我們有

Theorem 11 (拉格朗日定理)**.** 若 G 有限且 $H \trianglelefteq G$，則 $|G| = [G : H]|H|$。特別的，$|H|$ 整除 $|G|$。

Corollary 28. 若 G 有限，則所有元素的階皆整除 $|G|$

我們希望商集可以變成一個（么半）群，因此引入

Definition 50 (同餘關係；congruence relation)**.** 若（么半）群 G 上的等價關係 $\sim$ 滿足若 $a \sim b \wedge c \sim d$ 則 $a \star c \sim b \star d$，則我們稱 $\sim$ 是一個 $(G, \star)$ 的等價關係。

Example 11. $\equiv \pmod p$ 是一個 $(\mathbb{Z}, +)$ 上的同餘關係。

Property 59. $\sim_H$ 是一個同餘關係

Property 60. 若 $\sim$ 是一個同餘關係，則 $(G/\sim, \bar{\star})$ 是一個（么半）群。

Proof. 注意到因為同餘關係的定義，如果 $\overline{a} = \overline{b}, \overline{c} = \overline{d}$，則 $\overline{a \star c} = \overline{b \star d}$。因此我們可以定義 $\bar{\star} : G/\sim \times G/\sim \to G/\sim, (\overline{x}, \overline{y}) \mapsto \overline{xy}$。顯然他還是滿足交換率與分配率；$\overline{e}$ 是單位元；若 x 可逆，則 $\overline{x^{\star -1}}$ 是 $\overline{x}$ 的逆元。 □

此（么半）群稱為商（么半）群。

Definition 51 ((同餘關係的) 核；kernel)**.** $\ker \sim := \overline{e} = \{g \in G | g \sim e\}$

Property 61. $\ker \sim_K = K$, $G/\ker \sim = G/\sim$

你會發現 G 的所有子群跟 G 的所有同餘關係有一一對應。

0.3.4 同態

現在我們要定義兩個 (么半) 群之間的映射。合理的定義就是

Definition 52 (同態；homomorphism)**.** 若 $\varphi : G \to G'$ 滿足 $\varphi(a \star_G b) = \varphi(a) \star_{G'} \varphi(b)$ 且 $\varphi(e_G) = e_{G'}$，則稱 φ 是一個同態。

注意等號左邊的乘法是 G 的乘法；等號右邊則是 G' 的乘法。
而所有 G 到 G' 的同態事實上也構成一個群

Definition 53 (同態群)**.** $\big(\mathrm{Hom}(G, G'), (\varphi \star \phi)(g) = \varphi(g) \star \phi(g)\big)$

Definition 54 (單同態；monomorphism)**.** 單射的同態

Definition 55 (滿同態；epimorphism)**.** 滿射的同態

Definition 56 (雙同態；同構；isomorphism)**.** 雙射的同態

Definition 57 (同構；isomorphic)**.** 若 G 和 G' 之間存在雙同態，則稱兩者同構，記做 $G \simeq G'$

Definition 58 (自同態；endomorphism)**.** 打到自己的同態

Definition 59 (自同構；automorphism)**.** 打到自己的同構

注意到 G 的所有自同構構成一個群

Definition 60 (自同構群；automorphism group)**.** $(\operatorname{Aut} G, \circ)$

Definition 61 ((同態的) 核；kernel)**.** $\ker\varphi = \varphi^{-1}(e_{G'}) = \{g \in G | \varphi(g) = e_{G'}\}$

Property 62. φ 是單射若且唯若 $\ker\varphi = \{e\}$

Proof. 若 $\varphi(a) = \varphi(b)$，則 $\varphi(a \star b^{\star -1}) = e$，亦即 $a \star b^{\star -1} \in \ker\varphi$。 □

Definition 62 (像；image)**.** $\varphi(G) = \{\varphi(g) | g \in G\}$

Example 12. $\varphi : (\mathbb{Z}, +) \to (\mathbb{Z}, +),\ x \mapsto 2x$ 是一個 (單) 同態

Example 13. $\varphi : G \to G/H,\ x \mapsto \overline{x}$ 是一個 (滿) 同態

Example 14. $(\mathbb{Z}/p\mathbb{Z}, +) \simeq C_p$

Theorem 12 (同態基本定理)**.** 給定同態 $\varphi : G \to G'$，則

$$\ker\varphi \trianglelefteq G,\ \varphi(G) \trianglelefteq G',\ G/\ker\varphi \simeq \varphi(G)$$

Proof. 子群的事情照定義檢查即可。要證明那個同構可以證明 $\varphi : \overline{x} \mapsto \varphi(x)$ 是良好定義的、是同態、是單射、是滿射。 □

Theorem 13 (第一同構基本定理)**.** 給定 $H, N \trianglelefteq G$ 則

$$HN \trianglelefteq G,\ H \cap N \trianglelefteq H,\ (HN)/N \simeq H/(H \cap N)$$

Proof. 子群的事情照定義檢查即可。要證明那個同構可以證明 $H \to HN/N, h \mapsto h + N$ 是同態、是單射，而且核是 $H \cap N$。 □

Definition 63. 給定子群 $H, N \trianglelefteq G$，定義 $H \star N = \{h \star n | h \in H, n \in N\}$。

Theorem 14 (第二同構基本定理)**.** 給定 $N \trianglelefteq G$ 則 G/N 的子群與 $H/N, H \trianglelefteq G$ 有一一對應，而且 $G/H \sim (G/N)/(H/N)$

Proof. 考慮 $\varphi : G \to G/N$，則 φ 會把包含 N 的子群打到 G/N 的子群，而 φ^{-1} 會把 G/N 的子群拉回 G 的子群。因此你有一一對應。要證明那個同構可以證明 $G/H \to G/N, \overline{x} \mapsto \overline{x}$ 是良好定義的、是同態、是單射，而且核是 H/N。 □

0.3.5 有限群結構定理

Definition 64 (直積；direct product)**.** 若 $(G_1, \star_1)$ 與 $(G_2, \star_2)$ 是群，則我們可以在 $G_1 \times G_2 = \{(g_1, g_2) | g_1 \in G_1, g_2 \in G_2\}$ 上定義運算 $(g_1, g_2) \star (h_1, h_2) = (g_1 \star_1 h_1, g_2 \star_2 h_2)$。這會是一個群，稱作兩個群的直積，寫作 $G_1 \times G_2$。

Lemma 23. 若 $H, N \trianglelefteq G$ 滿足 $H \cap N = \{e\}$，$H \star N = G$，則 $H \times N \simeq G$。

Proof. 我們有同態 $H \times N \to G, (h, n) \mapsto hn$，而且他是單射且滿射的。 □

Definition 65 (內直積)**.** 如果兩個子群 $H, N \trianglelefteq G$ 滿足 $H \cap N = \{e\}$，則我們把 $H \star K$ 寫作 $H \circledast K$，稱為他們的內直積。

注意到前一個引理保證 $H \circledast K \simeq H \times K$。
現在我們可以敍述本節最重要的定理了

Theorem 15 (有限（阿貝爾）群結構定理)**.** 任何有限群都是一些循環子群的內直積。

在開始證明之前，我們需要一些引理。

Lemma 24 (柯西定理)**.** 若 G 是一個有限群，且質數 p 整除 $|G|$，則存在一個元素 $x \in G$ 使得 $\operatorname{ord}(x) = p$。

Proof. 我們對 $|G|$ 做數學歸納法。如果 $|G| = 1$ 那顯然成立。隨便選一個 $x \in G$，若 $p | \operatorname{ord}(x)$，則 $x^{\star \operatorname{ord}(x)/p}$ 的階是 p。否則 p 整除 $|G/\langle x \rangle|$。根據歸納假設，存在 $\overline{y} \in G/\langle x \rangle$ 使得 $\operatorname{ord}(\overline{y}) = p$。則 $\overline{y}^{\overline{\star} \operatorname{ord}(y)} = \overline{e}$ 告訴我們 $p | \operatorname{ord}(y)$。因此 $y^{\star \operatorname{ord}(y)/p}$ 的階是 p。 □

Definition 66 (p-群)**.** 一個 p-群指的是元素的階都是 p 的冪次的群。

Corollary 29. 有限 p-群的階都是 p 的冪次。

Proof. 否則如果 q 整除群的階，則必有元素的階是 q。 □

我們想要先做 G 是 p-群的情況。

Lemma 25. 如果一個有限 p-群的大小為 p 的子群只有一個，則他是循環群

Proof. 對 $|G|$ 做數學歸納法。假設 H 是唯一的那個大小為 p 的子群，則 $H = \{g \in G | g^{\star p} = e\}$。考慮 $\varphi : G \to G, g \mapsto g^{\star p}$，則 $\ker \varphi = H$。$\varphi(G) \trianglelefteq G$ 亦滿足「大小為 p 的子群只有一個」的條件（因為 $\varphi(G)$ 的子群都是 G 的），因此根據歸納假設 $\varphi(G)$ 是循環群。假設 $g \star K$ 是 $G/K \simeq \varphi(G)$ 的生成元，則 $G = \langle g \rangle$。 □

Lemma 26. 若 G 是一個有限 p-群，且 $C \trianglelefteq G$ 是 G 的循環子群中，階最大的循環子群之一，則存在 $H \trianglelefteq G$ 使得 $G = C \circledast H$。

Proof. 一樣對 $|G|$ 做數學歸納法。如果 G 是循環群則 $G = C = C \times \{e\}$。否則根據引理，G 一定包含另一個不包含在 C 中，且大小為 p 的群 K。
$(C \star K)/K \simeq C/(C \cap K) \simeq C$，因此 $(C \star K)/K$ 是 G/K 中最大的循環子群（如果 $\langle x \star K \rangle$ 比他還大，那 $\langle x \rangle \trianglelefteq G$ 就比 C 還要大）。根據歸納假設，必存在某個 $H/K \trianglelefteq G/K$ 使得 $G/K = (C \star K)/K \circledast H/K$。則 $C \cap H = \{e\}$，且 $|C||H| = |G|$。因此 $G = C \circledast H$。 □

接著我們只要說明每個有限群都能拆成 p-群的直積即可。

Definition 67. $G(p) = \{g \in G | \exists k\,[o(g) = p^k]\}$

注意到 $G(p)$ 是一個 p-群。回憶 $v_p(n)$ 指的是最大的 k 滿足 $p^k | n$。

Lemma 27. $|G(p)| = p^{v_p(|G|)}$

Proof. 否則 p 整除 $|G/G(p)|$，因此存在 $x \in G \setminus G(p)$ 使得 $\mathrm{ord}(x) = p$。但這代表 $x^{\star p} \in G(p)$，亦即 $\mathrm{ord}(x)$ 也是 p 的冪次，矛盾。 □

Corollary 30. $G = \bigoplus_{p||G|} G(p)$

Proof. 顯然 $\bigoplus_{p \neq p'} G(p)$ 跟 $G(p')$ 的交集只有單位元（因為他們的階互質）。而且根據引理，$|G| = \prod |G(p)| = |\bigoplus G(p)|$。因此 $G = \bigoplus G(p)$。 □

於是我們可以把一個群拆成一堆 p-子群的內直積，每個又可以各自拆成循環子群的內直積。因此 G 就是那些循環子群的內直積。

0.3.6 環

群縱然有用，但他能做到的還是有限的。我們希望我們的結構上同時有兩種運算，以便探討他們之間的關係。於是我們定義

Definition 68 (環；ring)**.** 一個環 $(R, +, \cdot)$ 是一個集合 R 和其上的兩個二元運算，使得 $(R, +)$ 是一個群，$(R, \cdot)$ 是一個么半群，且滿足分配律（對於所有 $a, b, c \in R$，$a \cdot (b + c) = a \cdot b + a \cdot c$）。

我們一般把 $(R, +)$ 的單位元寫作 0，$(R, \cdot)$ 的單位元寫做 1，$a \cdot b$ 寫作 ab，a^{+n} 寫作 na，a^{+-1} 寫作 $-a$。

Definition 69 (特徵)**.** $\mathrm{char}\, R$ 指的是 1 在 $(R, +)$ 的階。若不存在，則我們定義 $\mathrm{char}\, R = 0$。

以下定義一些跟群論時頗類似的定義

Definition 70 (子環；subring)**.** 若 $S \subseteq R$ 滿足 $+, \cdot$ 可以被視為 S 上的運算，而且 $(S, +, \cdot)$ 是一個環，則稱 S 是 R 的子環。

Definition 71 (理想；ideal)**.** 若 $I \subseteq R$ 滿足 $(I, +)$ 是 $(R, +)$ 的子群，且 $RI \subseteq I$，則我們稱 I 為 R 的理想。

Example 15. $n\mathbb{Z}$ 是 $\mathbb{Z}$ 的理想。

Definition 72. 注意到對於理想 I, J，$I \cdot J$ 不一定是一個理想。因此我們定義 $IJ = \langle I \cdot J \rangle$。

Property 63. 注意到對於理想 I, J，$I + J$ 是一個理想。

Definition 73 (質理想；prime ideal)**.** 若理想 $p \subseteq R$ 滿足對於所有 $ab \in p$ 都有 $a \in p$ 或 $b \in p$，則我們稱 p 是一個質理想。

Example 16. $n\mathbb{Z}$ 是 $\mathbb{Z}$ 的質理想若且唯若 n 是質數。

Definition 74 (商環)**.** 對於理想 $I \subseteq R$，我們可以定義 $R/I = \{r + I | r \in R\}$ 上的加法和乘法讓他變成一個環，稱為商環

一樣定義可逆、直積、同態、同構為同時滿足 $(R, +)$ 與 $(R, \cdot)$ 中的定義者。一樣定義有限、生成、核等概念。

Property 64. x 可逆若且唯若 $\langle x \rangle = R$。

Definition 75 (主理想；principal ideal)**.** 由一個元素生成的理想。

Theorem 16 (同態基本定理)**.** 給定同態 $\varphi : R \to R'$，則 $\ker \varphi$ 是 R 的理想，$\varphi(R)$ 是 R' 的子環，且 $R/\ker \varphi \simeq \varphi(R)$。

Theorem 17 (第一同構基本定理)**.** 若 S 是子環，I 是理想，則 $S + I$ 也是子環，$S \cap I$ 是 S 的理想，而且 $(S + I)/I \simeq S/(S \cap I)$。

Theorem 18 (第二同構基本定理)**.** R/I 的子環/理想與包含 I 的子環/理想有一一對應。

0.3.7 整環、體

Definition 76 (零因子；zero divisor)**.** 若存在 b 使得 $ab = 0$，則我們稱 a 為一個零因子。

Property 65. 若 $0 \neq 1$，則零因子不可逆。

Definition 77 (整環；integral domain)**.** 沒有零以外的零因子的環稱為整環。

Property 66. R/I 是整環若且唯若 I 是 R 的質理想。

Property 67. 若 R 是整環，則 $\operatorname{char} R$ 是零或質數。

Property 68. 在整環中，如果 $ax = bx$，且 $x \neq 0$，則 $a = b$。

Proof. $(a - b)x = 0$ □

Definition 78 (體；field)**.** $0 \neq 1$ 且所有非零元素都可逆的環稱為體。

Property 69. 體是整環。

Property 70. 有限整環是體。

Proof. 對於所有 x，函數 $f_x : y \mapsto xy$ 是單射的，因此是滿射的，也就是存在 y 使得 $xy = 1$。 □

Corollary 31. $\mathbb{Z}/p\mathbb{Z}$ 是體。

Property 71. 體 F 的理想只有 F 與 $\{0\}$。

Property 72. 若 $\varphi : F \to R$ 是環同態，則他是單射，而且我們可以把 R 看成是 K-向量空間，其中係數積定義為 $x, v \mapsto \varphi(x)v$。

Property 73. 有限體的階是質數冪次的。

Proof. 有限體 F 的特徵值非 0，因此你有 $\varphi : \mathbb{Z}/p\mathbb{Z} \to F, \overline{n} \mapsto n1$。因此 F 是一個 $\mathbb{Z}/p\mathbb{Z}$ 的有限維向量空間。特別的，F 的加法群同構於 $(\mathbb{Z}/p\mathbb{Z})^n$ 的加法群。 □

0.3.8 ED, PID

Definition 79 (歐幾里整環；Euclidean domain (ED))**.** 滿足「存在一個函數 $f : R \setminus \{0\} \to \mathbb{N}$，以及對每對元素 $x \neq 0, y$，都存在元素 q, r，使得 $f(r) < f(x)$ 或 $r = 0$，且 $y = qx + r$」的整環稱為歐幾里整環。

ED 看起來定義很長，但事實上在高中數學內就提到不少 ED 了。

Example 17. $\mathbb{Z}$ 是一個 ED。

Example 18. 若 k 是一個體，則 $k[x]$ 是一個 ED。

以下的定理用這些例子去思考可能會更清楚。

Definition 80 (主理想整環；Principal ideal domain (PID))**.** 滿足每個理想都是主理想的整環稱為主理想整環。

Theorem 19. ED 都是 PID

Proof. 對於一個理想 $I \neq (0)$，令 $x \in I \setminus \{0\}$ 使得 $f(x)$ 是 I 中最小的之一。則我們希望 $I = \langle x \rangle$。$\supseteq$ 是顯然的。對於 $\subseteq$，假設 $y \in I$，則根據 ED 的定義存在 q, r 使得 $r = x - qy \in I$。但根據 x 的選取，若 $r \neq 0$ 則 $f(r) > f(x)$。故 $r \neq 0$ 且 $y = qx \in \langle x \rangle$。 □

接著我們來探討 PID 的一些性質

0.3.8.1 整除與最大公因數

回憶 $a|b$ 的定義是存在 x 使得 $ax = b$。

Definition 81 (關聯；assotiate)**.** 若 $a|b$，$b|a$（亦即 $\langle a \rangle = \langle b \rangle$），則我們稱 $a \sim b$，稱作 a, b 有關聯。

Property 74. 若 R 是一個整環，且 $a \sim b$，則存在 $x \in R^\times$ 使得 $ax = b$

Proof. $ax = b, by = a$，則 $xya = a$，亦即 $xy = 1$（因為 R 是整環） □

也就是在整環中，關聯就只是差一個可逆元素而已。

Definition 82 (gcd)**.** 在 PID 中，定義 $\gcd(a,b)$ 為 $\langle a\rangle + \langle b\rangle$ 的生成元；$\operatorname{lcm}(a,b)$ 為 $\langle a\rangle \cap \langle b\rangle$ 的生成元。若 $\gcd(a,b) \in R^\times$，則我們稱 a, b 互質。

在這裡寫一些基本性質

Property 75.

1. 若 $a|b, a|c$ 則 $a|\gcd(a,b)$
2. 若 $b|a, c|a$ 則 $\operatorname{lcm}(b,c)|a$
3. $a\gcd(b,c) = \gcd(ab, ac)$
4. $ab \sim \operatorname{lcm}(a,b)\gcd(a,b)$
5. $a\operatorname{lcm}(b,c) = \operatorname{lcm}(ab, ac)$
6. 若 a, b 互質且 $a|bc$，則 $a|c$

Proof.

3. $\langle a\rangle(\langle b\rangle + \langle c\rangle) = \langle a\rangle\langle b\rangle + \langle a\rangle\langle c\rangle = \langle ab\rangle + \langle ac\rangle$
4. 首先我們證明 $\operatorname{lcm}(a,b)\gcd(a,b)|ab$。令 $g = \gcd(a,b), a = xg, b = yg$，則 $a|xyg, b|xyg$，故 $\operatorname{lcm}(a,b)|xyg$，亦即 $\operatorname{lcm}(a,b)\gcd(a,b)$ 整除 $xygg = ab$。

   接著我們證明 $ab|\operatorname{lcm}(a,b)\gcd(a,b)$。但是 $ab|a\operatorname{lcm}(a,b)$ 且 $ab|\operatorname{lcm}(a,b)b$，故 ab 整除 $\gcd(\operatorname{lcm}(a,b)a, \operatorname{lcm}(a,b)b) = \operatorname{lcm}(a,b)\gcd(a,b)$。
6. $a|bc, b|bc$ 故 $ab = \operatorname{lcm}(a,b)$ 整除 bc。故 $a|c$。

□

Lemma 28. $\overline{a} \in (R/\langle b\rangle)^\times$ 若且唯若 a, b 互質。

Proof. 兩者皆若且唯若 $\overline{a} + \overline{b} = \langle\overline{a}\rangle = R$ □

0.3.8.2 不可約元素與質元素

Definition 83 (不可約元素；irreducible element)**.** 給定不是零因子也不可逆的元素 p，若對於所有 $ab = p$，我們都有 $a \in R^\times$ 或 $b \in R^\times$，則我們稱 p 是一個不可約元素。

Definition 84 (質元素；prime element)**.** 若對於所有 $p|ab$，我們都有 $p|a$ 或 $p|b$（也就是 $\langle p\rangle$ 是質理想），則我們稱 p 是一個質元素。

Property 76. 所有質元素都是不可約元素

Proof. 若 p 是質元素，$p = ab$，則 $p|a$ 或 $p|b$，也就是 $b \in R^\times$ 或 $a \in R^\times$。 □

Property 77. 在 PID 中，所有不可約元素都是質元素。

Proof. 若 $p|ab$ 且 p 不整除 a，則 p 與 a 互質。故 $p|b$。 □

注意到不可約元素可能不是質元素。例如在 $\mathbb{Z}[\sqrt{5}]$ 中，$2|4 = (\sqrt{5}-1)(\sqrt{5}+1)$，但是 2 不整除 $\sqrt{5} \pm 1$。

0.3.8.3 諾特環

Definition 85 (諾特環)**.** 若在 R 中任何一串 $I_1 \subseteq I_2 \subseteq I_3 \subseteq \dots$ 理想的序列，都存在一個 n 使得 $I_n = I_{n+1} = \cdots$，則我們稱 R 是一個諾特環。

Lemma 29. R 是諾特環若且唯若 R 的所有理想都是有限生成的。

Proof. ($\Rightarrow$): 任給一個不是有限生成的理想 $I \subseteq R$，定義 $I_0 = \{0\}$，並對每個 i 歸納地選取某個 $x_i \in I \setminus I_{i-1}$，令 $I_i = I_{i-1} + \langle x_i \rangle$（注意到因為 I_i 都是有限生成的，所以 $I \neq I_i$，我們也因此一定選得到 x_i）。則我們就找到一個不會停止的上升理想序列 $I_0 \subseteq I_1 \subseteq I_2 \subseteq \cdots$。
($\Leftarrow$): 假設 $I_1 \subseteq I_2 \subseteq I_3 \subseteq \dots$ 是一串理想的序列，則 $I = \bigcup I_i$ 也是一個理想。因為他是有限生成的，我們假設 $I = \langle x_1, \dots, x_n \rangle$。則每個 x_i 必掉到某個 I_{a_i} 中。取最大的 a_i（因為有限所以取得到），我們知道所有 x_i 都掉在 I_{a_i} 裡面，所以 $I = I_{a_i}$。也就是 $I_{a_i} = I_{a_i+1} = \cdots$。 □

Corollary 32. PID 都是諾特環。

當然，諾特環還有許多其他性質與用途，但因為本文不會用到，因此在這裡不多提。

0.3.8.4 中國剩餘定理

Theorem 20 (中國剩餘定理；CRT)**.** 若 $a_1, \dots a_n$ 兩兩互質，則 $f : R/\langle a_1 a_2 \dots a_n \rangle \to R/\langle a_1 \rangle \times R/\langle a_2 \rangle \times \cdots \times R/\langle a_n \rangle$ 是一個同構。

Proof. 根據歸納法，我們只要考慮 $n = 2$ 的情形。注意到 $\langle a_1 \rangle + \langle a_2 \rangle = R$，且 $\langle a_1 \rangle \cap \langle a_2 \rangle = \langle a_1 a_2 \rangle$。因此 $R/\langle a_1 a_2 \rangle \simeq \langle a_1 \rangle / \langle a_1 a_2 \rangle \times \langle a_2 \rangle / \langle a_1 a_2 \rangle$（Lemma 23）。根據第二同構定理，$\langle a_1 \rangle / \langle a_1 a_2 \rangle \simeq (\langle a_1 \rangle + \langle a_2 \rangle)/\langle a_2 \rangle = R/\langle a_2 \rangle$。同樣的 $\langle a_2 \rangle / \langle a_1 a_2 \rangle \simeq R/\langle a_2 \rangle$。 □

0.3.9 UFD

Definition 86 (唯一分解整環；Unique Factorization domain (UFD))**.** 滿足「每個不可逆的非零元素都可以被拆成不可約元素的積，而且這個分解是唯一的（至多只差一個排列後的關聯）」的整環稱為唯一分解整環。

Theorem 21. PID 都是 UFD

這個證明比較煩躁一些，因此我們分成以下幾個引理。

Lemma 30. 若 R 是 PID，$x \notin R^\times$，則存在不可約元素 p 使得 $p|x$。

Example 19. 假設不存在那種 p。根據不可約的定義，存在 $a_1b_1 = x$ 使得 $a_1, b_1 \notin R^\times$。因為 $a_1|p$，所以 a_1 可約。我們可以找到 $a_2b_2 = a_1$。利用歸納法，我們可以得到一串 $a_i, b_i \notin R^\times$ 使得 $a_ib_i = a_{i-1}$。因此我們得到一個 $\langle a_1\rangle \subsetneq \langle a_2\rangle \subsetneq \cdots$，跟 R 是諾特環矛盾。

Lemma 31. 若 R 是 PID，$x \notin R^\times$，則 x 可以被寫成一些不可約元素的乘積。

Proof. 根據前一題，我們可以遞迴的定義 $x = x_0 = p_1x_1, x_1 = p_2x_2, x_2 = p_3x_3, \cdots$。如果我們可以無限制的取下去，則我們得到 $\langle x_1\rangle \subsetneq \langle x_2\rangle \subseteq \cdots$，跟 R 是諾特環矛盾。因此有某個 $x_n \in R^\times$，亦即 $x = p_1p_2 \dots p_{n-1}(p_nx_n)$（因為 $x \notin R^\times$ 所以至少有一項。 □

因此我們現在可以把 PID 的元素分解成不可約元素的乘積了。現在我們必須證明他們相同

Proof of Theorem 21. 假設 $p_1 \dots p_n \sim q_1 \dots q_m$。因為 p_1 是質元素，p_1 整除某個 q_i。因為 q_i 不可約，$q_i \sim p_1$。根據歸納法得證。 □

0.3.10 多項式環

Corollary 33. 一個 $f \in k[x]$ 至多只有 $\deg f$ 個根

Proof. 首先注意到 $k[x]$ 是 ED，且 $x - a$ 和 $x - b$ 互質。因此如果 $x_1, \dots, x_n$ 是根，則 $\prod(x - x_n)$ 整除 f，故 $\deg f \geq n$。 □

Corollary 34. 若 K 是有限體，則 $K^\times$ 是一個循環群。

Proof. 根據有限群基本定理，他是一些循環群的直積。如果存在某個 q 使得 $K^\times(q)$ 不是循環群，則 $x^p = 1$ 在 K 有多於 p 個解，矛盾。又根據 CRT，一些互質的循環群的直積也是循環群。 □

Bibliography

[1] Leonard Adleman, Kenneth Manders, and Gary Miller. "On Taking Roots in Finite Fields". In: *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*. SFCS '77. USA: IEEE Computer Society, 1977, pp. 175–178. DOI: 10.1109/SFCS.1977.18. URL: https://www.cs.cmu.edu/~glmiller/Publications/b2hd-AMM77.html.

[2] N. C. Ankeny. "The Least Quadratic Non Residue". In: *Annals of Mathematics* 55.1 (1952), pp. 65–72. ISSN: 0003486X. URL: http://www.jstor.org/stable/1969420.

[3] Tom M. Apostol. *Introduction to Analytic Number Theory*. 1976. ISBN: 978-0-387-90163-3. DOI: 10.1007/978-1-4757-5579-4.

[4] D. A. Burgess. "The distribution of quadratic residues and non-residues". In: *Mathematika* 4.2 (1957), pp. 106–112. DOI: 10.1112/S0025579300001157.

[5] Henri Cohen. *A Course in Computational Algebraic Number Theory*. 1993. ISBN: 978-3-540-55640-4. DOI: 10.1007/978-3-662-02945-9.

[6] M. Deléglise and J. Rivat. "Computing pi(x): the Meissel, Lehmer, Lagarias, Miller, Odlyzko method". In: *Math. Comput.* 65 (1996), pp. 235–245. URL: https://www.ams.org/journals/mcom/1996-65-213/S0025-5718-96-00674-6/S0025-5718-96-00674-6.pdf.

[7] G. Hardy and E. Wright. "An Introduction to the Theory of Numbers". In: (1938).

[8] Swastik Kopparty. "The Weil bounds". In: (2013). URL: https://sites.math.rutgers.edu/~sk1233/courses/finitefields-F13/weil.pdf.

[9] D. H. Lehmer. "On the exact number of primes less than a given limit". In: *Illinois Journal of Mathematics* 3 (1959), pp. 381–388.

[10] Gary L. Miller. "Riemann's Hypothesis and Tests for Primality". In: *Proceedings of the Seventh Annual ACM Symposium on Theory of Computing*. STOC '75. Albuquerque, New Mexico, USA: Association for Computing Machinery, 1975, pp. 234–239. ISBN: 9781450374194. DOI: 10.1145/800116.803773.

[11] min25. *Sum of Multiplicative Function*. 2018. URL: https://min-25.hatenablog.com/entry/2018/11/11/172216.

[12] Paul Pritchard. "A Sublinear Additive Sieve for Finding Prime Number". In: *Commun. ACM* 24.1 (Jan. 1981), pp. 18–23. ISSN: 0001-0782. DOI: 10.1145/358527.358540.

[13] Richard Sladkey. *A Successive Approximation Algorithm for Computing the Divisor Summatory Function*. 2012. arXiv: 1206.3369 [math.NT]. URL: https://arxiv.org/abs/1206.3369.

[14] whzzt. 积性函数求和问题的一种筛法. 2020. URL: https://blog.csdn.net/whzzt/article/details/104105025.