Arithmétique modulaire – algos de base

Franck.Quessette@uvsq.fr

8 octobre 2022

1 Euclide étendu

1.1 Problème

Étant donnés a et $b \in \mathbb{N}$, calculer u, $v \in \mathbb{Z}$ et $d \in \mathbb{N}$ tels que :

$$au + bv = d = \text{pgcd}(a, b)$$

1.2 Algo

Calculer trois suites en même temps (r_n, u_n, v_n) avec :

$$\begin{array}{lll} r_0 = a & u_0 = 1 & v_0 = 0 \\ r_1 = b & u_1 = 0 & v_1 = 1 \end{array}$$

et

$$\left\{ \begin{array}{lllll} r_{n+1} & = & r_{n-1} & - & \alpha_n \; r_n \\ u_{n+1} & = & u_{n-1} & - & \alpha_n \; u_n \\ v_{n+1} & = & v_{n-1} & - & \alpha_n \; v_n \end{array} \right. \qquad \text{avec } \alpha_n = \left\lfloor \frac{r_{n-1}}{r_n} \right\rfloor$$

Il existe un N tel que $r_{N+1} = 0$ et on a $d = r_N$, u $= u_N$ et $v = v_N$.

1.3 Propriétés pour preuve de convergence de l'algo

Pour tout $n \leq N$, on a $r_n = au_n + bv_n$. Et si $b < a$ la suite des r_n est décroissante, sinon elle décroit à partir de r_1 et pas de r_0. r_{n+1} peut se définir de façon équivalente par $r_{n+1} = r_{n-1} \bmod r_n$. L'écriture avec α_n rend les trois récurrences plus simillaires.

1.4 Exemple

$a = 123$ et $b = 69$:

n	r_n	α_n	u_n	v_n
0	123		1	0
1	69	1	0	1
2	54	1	1	-1
3	15	3	-1	2
4	9	1	4	-7
5	6	1	-5	9
6	3	2	9	-16
7	0			

On a donc $123 \times 9 + 69 \times (-16) = 3$

1.5 Propriété

Si $au + bv = d$ alors :

$$\forall k \in \mathbb{Z}, \quad a\left(u + k\frac{b}{d}\right) + b\left(v - k\frac{a}{d}\right) = d \tag{1}$$

C'est utile si on veut un $v > 0$ par exemple.

2 Résolution d'équations modulaire (restes chinois)

2.1 Probème

Soient $n_1, n_2, \dots, n_k$ des entiers deux à deux premiers entre eux et $a_1, a_2, \dots, a_k$ des entiers. On veut trouver tous les $x \in \mathbb{Z}$ tels que :

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ x \equiv a_2 \pmod{n_2} \\ \dots \\ x \equiv a_k \pmod{n_k} \end{cases}$$

2.2 Algo

On pose :

$$N = \prod_{i=1}^{k} n_i \qquad \text{et} \qquad \overline{n}_i = \frac{N}{n_i}$$

n_i et $\overline{n}_i$ sont premiers entre eux donc leur pgcd est 1.

Pour tout i, on calcule avec Euclide étendu u_i et v_i tels que $u_i n_i + v_i \overline{n}_i = 1$ et on pose $e_i = v_i \overline{n}_i$.

On a alors :

$$\begin{array}{lll} \forall i & e_i \equiv 1 \pmod{n_i} & \text{car } e_i \text{ est premier avec } n_i \\ \forall i, j \neq i & e_i \equiv 0 \pmod{n_j} & \text{car } e_i \text{ est un multiple de } n_j \end{array}$$

Une solutions est :

$$x = \sum_{i=1}^{k} a_i e_i$$

et l'ensemble des solutions est l'ensemble des x tels que :

$$x = \sum_{i=1}^{k} a_i e_i \quad + \quad k \times N, \qquad \forall k \in \mathbb{Z}$$

2.3 Exemple

Calculer x tel que :

$$\begin{cases} x \equiv 2 \pmod{3} \\ x \equiv 3 \pmod{5} \\ x \equiv 2 \pmod{7} \end{cases}$$

On a $N = 3 \times 5 \times 7 = 105$ et :

$$\begin{cases} n_1 = 3 & \overline{n}_1 = 5 \times 7 = 35 & u_1 = 12 & v_1 = -1 & e_1 = -35 \\ n_2 = 5 & \overline{n}_2 = 3 \times 7 = 21 & u_2 = -4 & v_2 = 1 & e_2 = 21 \\ n_3 = 7 & \overline{n}_3 = 3 \times 5 = 15 & u_2 = -2 & v_2 = 1 & e_2 = 15 \end{cases}$$

et donc $x = 2 \times (-35) + 3 \times 21 + 2 \times 15 = 23$ est une solution. L'ensemble des solutions est $\{x = 23 + 105k, \forall k \in \mathbb{Z}\}$.

Pour e_1 en utilisant la propriété 1.5 et en posant $k = 1$ on a $u_1 = -23$, $v_1 = 2$, $e_1 = 70$ et $x = 233$, et $233 \equiv 23 \pmod{105}$.

3 Équation diophantienne

3.1 Problème

Calculer les couples d'entiers (x, y) vérifiant l'équation :

$$ax + by = c, \quad a, b, c \in \mathbb{Z}$$

3.2 Algo

1. Calculer $d = \text{pgcd}(a, b)$.
2. Si d ne divise pas c alors $\boxed{\text{pas de solution}}$.
3. Si d divise c, diviser toute l'équation par d.
   Poser $a_0 = \frac{a}{d}$, $b_0 = \frac{b}{d}$, $c_0 = \frac{c}{d}$. L'équation à résoudre est maintenant :
   $$a_0x + b_0y = c_0$$
   avec a_0 et b_0 premiers entre eux.
4. Calculer avec Euclide étendu : u et v tels que $a_0u + b_0v = 1$.
   Alors (cu, cv) est une solution de $a_0x + b_0y = c_0$ et donc de $ax + by = c$.
   L'ensemble des solution est : $\boxed{\{(cu + kb_0, cv - ka_0), \forall k \in \mathbb{Z}\}}$.

3.3 Exemples

$4x + 6y = 2$

pgcd$(4, 6) = 2$, l'équation devient $2x+3y = 1$, $u = -1$, $v = 1$ les solutions sont $(-1+3k, 1-2k)$ pour tout $k \in \mathbb{Z}$.

$4x + 12y = 2$

pgcd$(4, 12) = 4$ et 4 ne divise pas 2 donc pas de solution.

$162x + 207y = 27$

pgcd$(162, 207) = 9$ et 9 divise 27, l'équation devient $18x + 23y = 3$ avec 18 et 23 premiers entre eux. On trouve $u = 9$ et $v = -7$. Les solutions sont $(27 + 23k, -21 - 18k)$ pour tout $k \in \mathbb{Z}$.
Pour $k = 0$, la solution est $(27, -21)$: $162 \times 27 - 207 \times 21 = 4374 - 4347 = 27$.
Pour $k = -1$, la solution est $(4, -3)$: $162 \times 4 - 207 \times 39 = 648 - 621 = 27$.

4 Puissances modulaires

4.1 Problème

Étant donnés b, e et m, calculer x tel que

$$x \equiv b^e \pmod m, \quad 0 \leq x < m$$

sans avoir des nombres "trop grands", en pratique, inférieur à m^2.

4.2 Remarques

On va utiliser
— l'exponentiation rapide qui consiste à décomposer l'exposant en somme de puissance de 2 ;
— la propriété que le modulo du produit est le produit des modulos.
On écrit e comme une somme de puissances de 2 :

$$e = \sum_{i=0}^{n} a_i 2^i \qquad a_i \in \{0, 1\}, \quad a_n = 1$$

on a :

$$b^e = b^{\left(\sum_{i=0}^{n} a_i 2^i\right)} = \prod_{i=0}^{n} \left(b^{2^i}\right)^{a_i}$$

et

$$b^{2^i} = \left(b^{2^{i-1}}\right)^2$$

Comme le modulo du produit est le produit du modulo, il faut calculer le produit modulo m des puissances, au carré, de b qui sont dans sa décomposition en base deux, ce qui donne l'algo :

4.3 Algo

Entrées : b, e, m entiers positifs
Sortie : $x = b^e \pmod m$ entier positif

$x \leftarrow 1$
Tant que $e > 0$ **faire**
  Si (e est impair) **alors** $x \leftarrow (x \times b)$ modulo m **Fin Si**
  $e \leftarrow e$ div 2
  $b \leftarrow (b \times b)$ modulo m
Fin Tant que
Retourner x

Dans l'algo x et b sont toujours inférieurs à m puisqu'ils sont calculés modulo m. Au pire les calculs intermédiaires font apparaître des nombres inférieurs à m^2.

4.4 Code efficace en C

```
// Pas nécessairement cette déclaration, c'est pour illustrer
#typedef unsigned long long int BigNum

puissance_modulaire(BigNum b, BigNum e, BigNum m) {
   BigNum x = 1;
   while (e > 0) {
        if (e & 1) x = (x * b) % m;    // Teste si le bit de poids faible est 1 donc impair
        e >>= 1;       // Décalage d'un bit vers la droite
        b = (b * b) % m;
   }
   return x;
 }
```

4.5 Exemple

Calculer $x = 4^{13} \pmod{497}$. $e = 13$ en base 10 et $e = 1101$ en base 2.

x	e	b
1	1101	4
4	110	16
4	11	256
30	1	429
445	0	

La solution est $x = 445$.