Algèbre et calcul formel

18 octobre 2023

Table des matières

Chapitre 1

Introduction

1.1. Objectifs

Dans ce cours, nous commençons par étudier comment coder différents objets mathématiques sur machine : les entiers, polynômes et éléments d'un quotient. Nous étudions ensuite des algorithmes pour effectuer les opérations élémentaires sur ces objets. En particulier, nous étudions des algorithmes de complexité sous-quadratique pour la multiplication.

Ensuite, nous étudions quelques algorithmes liés à la factorisation sur les anneaux $\mathbb{Z}$, $k[X]$, où k est un corps. Pour cela, nous travaillerons beaucoup dans des quotients $\mathbb{Z}/n\mathbb{Z}$ et $k[X]/Pk[X]$.

Enfin, nous voyons comment travailler dans $k[X_1, \ldots, X_n]$. En particulier, nous nous intéressons aux idéaux et verrons certains systèmes privilégiés de générateurs de ces idéaux : les bases de Gröbner.

1.2. Algorithmes

Définition 1.2.1. *1. Un algorithme est une suite d'instructions bien déterminées exécutées sur des entrées. Le ou les résultats sont appelées sorties.*

2. *Nos entrées seront des entiers identifiés par leur écriture binaire : une suite de* 0 *et de* 1 *appelés bits.*
3. *Les opérations se ramènent à des opérations dites élémentaires sur les bits : affectations, comparaisons, additions, soustractions, multiplications.*
4. *Le temps de calcul d'un algorithme est le nombre $T(s)$ de ces opérations élémentaires exprimé en fonction de la taille s des entrées numériques.*
5. *La taille s d'une entrée est le nombre de bits qui la représentent.*

Par exemple, soit n un entier naturel. La taille $s(n)$ de n est le nombre de chiffres dans l'écriture binaire de n : $s(0) = 1$ et si $n \neq 0$,

$$s(n) = \lfloor \log_2 n \rfloor + 1$$

En effet, si $s(n) = s$, alors $n = \sum_{i=0}^{s-1} n_i 2^i$ où $n_{s-1} = 1$. Donc

$$2^{s-1} \leq n \leq \sum_{i=0}^{s-1} 2^i = 2^s - 1$$

ce qui veut dire que $2^{s-1} \leq n < 2^s$ donc que $s = \lfloor \log_2 n \rfloor + 1$.

Notation. Sauf mention contraire, on écrira $\log = \log_2$.

1.3. Ordres de grandeurs O et O tilde

On s'intéresse à l'ordre de grandeur de $T(s)$. Soient deux fonctions f et g de $\mathbb{R}_+$ dans $\mathbb{R}_+$.

Définition 1.3.1. *On note $f(s) = O(g(s))$ s'il existe $M > 0$ tel que $f(s) \leq Mg(s)$ pour tout s.*

- Si $T(s) = O(s)$, on dit que le temps de calcul est linéaire.
- Si $T(s) = O(s^2)$, on dit que le temps de calcul est quadratique.
- Si $T(s) = O(s^k)$, (où $k \geq 1$) on dit que le temps de calcul est polynomial.
- Dans le cas où $T(s) = O(\exp(cs^k))$, où $c > 0$ et $k > 0$, on dit que le temps de calcul est sous-exponentiel si $k < 1$ et on dit qu'il est exponentiel si $k \geq 1$.

Définition 1.3.2. *On note $f(s) = \widetilde{O}(g(s))$ s'il existe $k > 0$ tel que $f(s) = O\left(g(s)(\log g(s))^k\right)$.*

Par exemple, si $f(s) = O(s^a(\log s)^b(\log(\log s))^c)$ où $a, b, c > 0$, $f(s) = \widetilde{O}(s^a)$.

Chapitre 2

Premiers algorithmes sur les entiers

2.1. Addition

Soient a et b deux entiers strictement positifs. Soit s un entier tel que $s \geq s(a)$ et $s \geq s(b)$. Alors a et b s'écrivent

$$a = \sum_{i=0}^{s-1} a_i 2^i \quad \text{et} \quad b = \sum_{i=0}^{s-1} b_i 2^i$$

On note

$$\{a\}_2 = a_{s-1} \dots a_0 \quad \text{et} \quad \{b\}_2 = b_{s-1} \dots b_0$$

où les a_i et b_i appartiennent à $\{0, 1\}$.

Algorithme 2.1.1. [ADD : ADDITION D'ENTIERS]
Entrées : a, b comme ci-dessus.
Sortie : $c = a + b$
1. (Initialisation de la retenue) $r \leftarrow 0$
2. Pour i *de* 0 *à* $s - 1$ *:*
3. $\quad (c_i, r) \leftarrow$ *reste et quotient de* $a_i + b_i + r$ *par* 2
4. $c_s \leftarrow r$
5. Sortir $c = \sum_{i=0}^{s} c_i 2^i$

On compte l'opération $(c_i, r) \leftarrow$ reste et quotient de $a_i + b_i + r$ comme une opération élémentaire.

Le temps de calcul est ici $T(s) = s + 2$ (en comptant chaque affectation $r \leftarrow 0$ et $c_s \leftarrow r$ comme une opération). Cet algorithme est linéaire : $T(s) = O(s)$.

On ne peut pas faire mieux qu'un temps linéaire car il faut au moins le temps de lecture des données.

Exercice 2.1.2. *Écrire un algorithme qui effectue la soustraction $a-b$ dont le temps de calcul est en $O(s(a))$ (a étant supposé supérieur à b).*

On pourra considérer la soustraction sur un bit avec retenue comme une opération élémentaire pré-programmée ($(c_i, r) \leftarrow (1,1)$ si $(a_i, b_i + r) = (0,1)$, $(c_i, r) \leftarrow (a_i, 1)$ si $b_i = r = 1$ et $(c_i, r) \leftarrow (a_i - b_i - r, 0)$ dans les autres cas).

2.2. Multiplication

On écrit

$$ab = \sum_{i=0}^{s-1} a_i 2^i b$$

où $s = s(a)$. Dans cette somme, si $a_i = 0$ alors $a_i 2^i b = 0$ et si $a_i = 1$, $a_i 2^i b = 2^i b$: cette opération est un décalage dans l'écriture binaire.

Si l'écriture binaire de b est $\{b\}_2 = b_{t-1} b_{t-2} \dots b_0$, celle de $2^i b$ est

$$\{2^i b\}_2 = b_{t-1} b_{t-2} \dots b_0 \underbrace{0 \dots 0}_{i \times 0}$$

Algorithme 2.2.1. [MULT : MULTIPLICATION D'ENTIERS]

Entrées : a, b
Sortie : $c = ab$
1. (initialisation) $b' \leftarrow b$, $c \leftarrow 0$, $s \leftarrow s(a)$
2. Pour i de 0 à $s(a) - 1$:
3. Si $a_i = 1$, $c \leftarrow c + b'$
4. $b' \leftarrow 2b'$
5. Sortir c

Au début, $c < b'$. Soient (c_i) et (b'_i) les suites des valeurs de c et b' dans l'algorithme. Si $c_i < b'_i$, alors

$$c_{i+1} \leq c_i + b'_i < 2b'_i = b'_{i+1}$$

Ainsi, à chaque étape i, la somme $c + b'$ revient à la somme (écrite en binaire) $(c_{i+t-1} \dots c_i) + (b_{t-1} \dots b_0)$, où $t = s(b)$ (on remplace directement les bits concernés dans la liste représentant c et on ne touche pas à la partie $(c_{i-1} \dots c_0)$). C'est en $O(t)$. Comme il y a $s(a)$ passages dans la boucle, on obtient le résultat suivant.

Proposition 2.2.2. *La multiplication de a par b peut se faire en $O(s(a)s(b))$.*

Remarque. On verra qu'on peut faire mieux : jusque $\widetilde{O}(s)$.

2.3. Division euclidienne

On suppose ici que $b \neq 0$. On cherche q et r tels que

$$a = qb + r \quad \text{et} \quad 0 \leq r < b$$

Pour comparer a et b, on compare les écritures binaires pour l'ordre lexicographique, ce qui se fait en temps linéaire. On écrira q sous la forme

$$q = \sum_{i=0}^{k} q_i 2^i$$

où $q_i \in \{0, 1\}$ pour tout $i \in [[0, k]]$ et où $q_k = 1$, c'est-à-dire $k = s(q) - 1$.

Exercice 2.3.1. *Si $a \geq b$, montrer que $k \in \{s(a) - s(b), s(a) - s(b) - 1\}$.*

Algorithme 2.3.2. [DIV : DIVISION EUCLIDIENNE]
Entrées : a, b
Sorties : le quotient et le reste q et r
1. (initialisation) $a' \leftarrow a$, $s \leftarrow s(a)$, $t \leftarrow s(b)$
2. (initialisation) Pour i de 0 *à* $s - t$: $q_i \leftarrow 0$
3. tant que $a' \geq b$:
4. $b' \leftarrow 2^{s-t} b$
5. si $b' \leq a'$:
6. $q_{s-t} \leftarrow 1$
7. $a' \leftarrow a' - b'$
8. sinon :
9. $q_{s-t-1} \leftarrow 1$
10. $b' \leftarrow b'/2$
11. $a' \leftarrow a' - b'$
12. $s \leftarrow s(a')$
13. sortir $q = \sum_{i=0}^{s(a)-s(b)} q_i 2^i$, $r = a'$

Démonstration. Exercice. □

Exercice 2.3.3. *Exécuter l'algorithme pour $(a, b) = (23, 4)$, puis $(23, 7)$.*

Proposition 2.3.4. *Le temps de calcul de l'algorithme 2.3.2 est en*

$$O(s(b)(s(a) - s(b) + 1))$$

Ce temps de calcul est donc quadratique : si $s(a) < s$ et $s(b) < s$, il est en $O(s^2)$.

Démonstration. Le nombre de passages dans la boucle "tant que" (3) est inférieur à $s(q)$, qui est égal à $s(a) - s(b) + 1$ ou à $s(a) - s(b)$. À chaque passage, on calcule $a' - b'$, ce qui peut se faire en modifiant les $s(b')$ ou $s(b') + 1$ premiers bits de a'. Chaque exécution de la boucle est donc en $O(s(b))$. □

Chapitre 3

Représentation

Nous avons vu comment représenter les entiers avec les bits. Passons maintenant en revue la représentation de quelques objets supplémentaires.

3.1. Nombres entiers relatifs

On ajoute un bit de plus à la représentation de la valeur absolue pour spécifier le signe. Ainsi, si $k \in \mathbb{Z}$, la taille de k est

$$s(k) = 1 + s(|k|) = \lfloor \log |k| \rfloor + 2 \sim \log |k|$$

3.2. Nombres rationnels

Tout élément de $\mathbb{Q}$ s'écrit de manière unique sous la forme $q = \dfrac{a}{b}$ où $a, b \in \mathbb{Z}$, $b > 0$ et $\text{pgcd}(a, b) = 1$. On représente alors q par le couple (a, b) et

$$s(q) = s(a) + s(b)$$

3.3. Polynômes à coefficients entiers

Soit $P \in \mathbb{Z}[X]$. On écrit

$$P = \sum_{i=0}^{d} a_i X^i$$

où les a_i appartiennent à $\mathbb{Z}$ et où $a_d \neq 0$ si $P \neq 0$. On représente ce polynôme par $(a_0, \ldots, a_d)$ et donc

$$s(P) = \sum_{i=0}^{d} s(a_i)$$

3.4. L'anneau quotient $\mathbb{Z}/N\mathbb{Z}$

Cet anneau est en bijection avec son système de représentants $\mathcal{R}_N = \{0, 1, \ldots, N-1\}$. Tout élément x de $\mathbb{Z}/N\mathbb{Z}$ peut être représenté de façon unique par un élément de $r \in \mathcal{R}_N$ et

$$s(x) = s(r) < s(N) \sim \log N$$

3.5. Principe général

Soit A un anneau. On suppose que l'on sache représenter chaque élément a de A. On note toujours $s(a)$ la taille d'un tel élément. On en déduit la taille des éléments de $A[X]$

$$s\left(\sum_{i=0}^{d} a_i X^i\right) = \sum_{i=0}^{d} s(a_i)$$

ceux de A^k

$$s((a_1, \ldots, a_k)) = \sum_{i=1}^{k} s(a_i)$$

etc.

Soit I un idéal de A. La taille d'un élément x de A/I dépend de l'élément choisi pour représenter x.

Notation. Soit $a \in A$, on note $[a]_I$ la classe de a modulo I.

Si l'on choisit de représenter x par $a \in A$ (où bien sûr $x = [a]_I$), $s(x) = s(a)$.

Exemple. Soit $A = \mathbb{F}_p[X]/(P)$, où p est un nombre premier, où $P \in \mathbb{F}_p[X]$ et où $d = \deg P$, tout élément de A peut se représenter par un polynôme de degré inférieur ou égal à $d-1$. Soit Q un tel élément.

$$Q = \sum_{i=0}^{d-1} q_i X^i$$

et

$$s(Q) = \sum_{i=0}^{d-1} s(q_i) \leq d\,(\lfloor \log(p-1) \rfloor + 1) \sim d \log p$$

Rappelons au passage que A est un corps si et seulement si P est irréductible dans $\mathbb{F}_p[X]$. On note alors $R = \mathbb{F}_{p^d}$.

3.6. Complexité binaire, complexité algébrique

3.6.1 Définitions

La complexité binaire d'un algorithme est son temps de calcul. C'est donc le nombre d'opérations élémentaires sur les bits exprimé en fonction de la taille des entrées.

Si l'on écrit un algorithme sur des polynômes, par exemple l'addition, on n'a pas besoin de spécifier l'anneau de base A. On compte alors le nombre d'opérations sur les éléments de A. De façon plus générale, on écrit souvent des algorithmes dont les entrées sont des éléments d'un anneau A. La complexité algébrique est alors le nombre d'opérations à effectuer sur ces entrées : affectation, addition, multiplication, division, comparaison.

3.6.2 Exemple 1 : addition dans A^k

Soit A un anneau. La complexité algébrique de l'addition dans A^k est égale à k.

Si $A = \mathbb{F}_p$, la complexité binaire de l'addition dans A^k est en $O(k \log p)$.

Si $A = \mathbb{Z}$, cette complexité binaire dépend de la taille des entiers en jeu. Pour additionner $a = (a_i)$ et $b = (b_i)$ dans $\mathbb{Z}^k$, la complexité binaire est $\sum_i \max(s(a_i), s(b_i))$.

3.6.3 Exemple 2 : Multiplication dans $A[X]$

Soient P et Q deux polynômes de $A[X]$ de degrés respectifs m et n. On peut adapter l'algorithme 2.2.1 pour la multiplication de P par Q. L'algorithme obtenu à une complexité algébrique en $O((m+1)(n+1))$: cette complexité est quadratique puisque si $m, n < d$, elle est en $O(d^2)$.

Exercice 3.6.1. *Écrire cet algorithme et évaluer sa complexité algébrique.*

Par exemple, si $m, n < d$ et si $A = \mathbb{Z}/N\mathbb{Z}$ (où $N > 1$), la complexité binaire de l'algorithme est en $O(d^2 \log N)$.

Chapitre 4

Diviser pour régner

Dans ce chapitre, on applique une stratégie qui consiste à diviser un calcul en plusieurs calculs sur des objets plus petits.

4.1. Exponentiation binaire

Soit $(M, \cdot)$ un monoïde (un ensemble M muni d'une loi interne $\cdot$ associative admettant un élément neutre dans M) et n un élément de $\mathbb{N} \setminus \{0\}$. On veut calculer la puissance n-ème x^n d'un élément x de M.

Si l'on utilise la première méthode qui vient à l'esprit : $x^2 = x \cdot x$, $x^3 = x \cdot x^2$, ... , $x^n = x \cdot x^{n-1}$, on obtient le résultat en $n-1$ multiplications.

Si $n = 2^k$ où $k > 0$, on voit bien que l'on peut faire autrement :

$$x^n = (x^{n/2})^2 = \big(\dots (x^2)^2 \dots\big)^2$$

Ce calcul se fait en $k = \log n$ multiplications.

Si n n'est pas une puissance de 2, on peut utiliser son écriture binaire. Voyons par exemple comment calculer

$$x^{13} = x \cdot x^4 \cdot x^8$$

On fait les opérations suivantes

$$\left.\begin{aligned} x \to x^2 \to x^4 \to x^5 = x \cdot x^4 \\ \to x^8 = (x^4)^2 \end{aligned}\right\} \to x^{13} = x^8 \cdot x^5$$

Ce calcul demande 5 multiplications au lieu de 12.

L'algorithme correspondant s'appelle l'algorithme "d'exponentiation binaire", ou bien l'algorithme "square and multiply".

Algorithme 4.1.1. [Exponentiation binaire]

Entrées : x et $n = 0$ ou $n = \sum_{i=0}^{r} n_i 2^i$ ($n_i \in \{0,1\}$ pour tout i et $n_r = 1$)

Sortie : x^n

1. (initialisation) $y \leftarrow 1$, $z \leftarrow x$

2. Si $n = 0$: sortir 1

3. Pour i de 0 à r :

4. si $n_i = 1$: $y \leftarrow yz$

5. si $i < r$: $z \leftarrow z^2$

6. sortir y

Démonstration. On note y_i et z_i les valeurs de y et z après le passage dans la boucle "pour" correspondant à la valeur i. Soit $s_i = \sum_{j=0}^{i} n_j 2^j$. On note aussi $y_{-1} = 1$, $z_{-1} = x$ et $s_{-1} = 0$. Alors $z_i = x^{2^{i+1}}$ pour tout $i \in [[0, r-1]]$ et $y_{-1} = x^{s_{-1}}$. On montre par récurrence que pour tout $i \in [[0, r]]$, $y_i = x^{s_i}$ (exercice). Cela montre le résultat. □

Proposition 4.1.2. *L'algorithme d'exponentiation binaire calcule x^n en au plus $2s(n) - 1 = 2\lfloor \log n \rfloor + 1$ multiplications dans M.*

Démonstration. La boucle "pour" au pas 3 est de longueur $r + 1 = s(n)$. De plus, chaque étape nécessite au plus 2 multiplications et la dernière en demande exactement une. □

Exemple. Soit N un entier naturel non nul. Soient $x \in \mathbb{Z}/N\mathbb{Z}$ et $n \in \mathbb{N}$. La complexité binaire du calcul de x^n est en $O(\log n M(N))$ où $M(N)$ est la complexité binaire de la multiplication modulo N. Avec les algorithmes classiques de multiplication et de division décrits plus haut, la complexité binaire est donc en $O(\log n (\log N)^2)$. On verra que cela peut aussi se faire avec une complexité binaire de $\widetilde{O}(\log n \log N)$.

Remarque. L'algorithme d'exponentiation binaire est très performant dans l'exemple ci-dessus, et plus généralement quand la taille des éléments du monoïde considéré est borné. Il est beaucoup moins intéressant lorsque ce n'est pas le cas. Par exemple, si l'on calcule 5^n dans $\mathbb{Z}$, cet algorithme conduit à des multiplications entre entiers de plus en plus grand, alors qu'en itérant des multiplications par 5, il y a davantage de multiplications, mais chacune de ces multiplications est moins coûteuse.

4.2. Algorithme de Karatsuba

Soient P et Q deux polynômes de degré inférieur à n. On peut calculer $P+Q$ avec une complexité algébrique en $O(n)$. On sait aussi que l'on peut calculer PQ avec une complexité algébrique en $O(n^2)$.

On décrit ici un algorithme de complexité algébrique inférieur pour effectuer le produit de deux polynômes.

La première étape consiste à couper les polynômes en deux. On suppose que $\deg P < n$, $\deg Q < n$ et que n est une puissance de 2 : $n = 2^k$. On écrit

$$\begin{aligned} P(X) &= P_0(X) + X^{n/2} P_1(X) \\ Q(X) &= Q_0(X) + X^{n/2} Q_1(X) \end{aligned} \tag{4.1}$$

où le degré des polynômes P_0, P_1, Q_0, Q_1 est strictement inférieur à $n/2$. Ce sont les divisions euclidiennes de P et Q par $X^{n/2}$, qui s'obtiennent simplement en coupant en deux les listes qui représentent chacun des deux polynômes. Plus précisément, soit

$$P(X) = \sum_{i=0}^{n-1} a_i X^i$$

Le polynôme P est représenté par la liste

$$[a_0, \ldots, a_{n-1}]$$

Les polynômes P_0 et P_1 sont respectivement représentés par les listes

$$[a_0, \ldots, a_{n/2-1}] \quad \text{et} \quad [a_{n/2}, \ldots, a_{n-1}]$$

Alors

$$PQ(X) = P_0Q_0(X) + X^{n/2}(P_0Q_1 + P_1Q_0)(X) + X^n P_1Q_1(X) \tag{4.2}$$

ce qui donne 4 multiplications entre polynômes de degré inférieur à $n/2$. On peut arranger cette égalité autrement.

$$\begin{aligned} PQ(X) =& P_0Q_0(X) \\ &+ X^{n/2}\left((P_0+P_1)(Q_0+Q_1) - P_0Q_0 - P_1Q_1\right)(X) \\ &+ X^n P_1Q_1(X) \end{aligned} \tag{4.3}$$

Le calcul de PQ en utilisant cette égalité ne demande que 3 multiplications : P_0Q_0, P_1Q_1 et $(P_0+P_1)(Q_0+Q_1)$. Cela mène à l'algorithme suivant.

Algorithme 4.2.1. [KARATSUBA]
Entrées : n (puissance de 2), $P, Q \in A[X]$ tels que $\deg P < n$, $\deg Q < n$
Sortie : PQ
1. Si $n = 1$, sortir PQ
2. $P_0, P_1, Q_0, Q_1 \leftarrow$ les polynômes définis en (4.1)
3. $R_0 \leftarrow$ `Karatsuba`$(n/2, P_0, Q_0)$
4. $R_2 \leftarrow$ `Karatsuba`$(n/2, P_1, Q_1)$
5. $S \leftarrow$ `Karatsuba`$(n/2, P_0 + P_1, Q_0 + Q_1)$
6. $R_1 \leftarrow S - R_0 - R_2$
7. Sortir $R_0 + X^{n/2} R_1 + X^n R_2$

Remarque. Cet algorithme se rappelle lui-même. On dit qu'il est récursif.

Démonstration. Si $n = 1$, `Karatsuba` rend le bon résultat. Supposons que si $\deg P, \deg Q < n/2$, alors `Karatsuba`$(n/2, P, Q)$ se termine et rend PQ. Soient P, Q tels que $\deg P, \deg Q < n$. Soient P_0, P_1, Q_0, Q_1 les polynômes définis en (4.1). L'hypothèse de récurrence montre que les pas 2, 3 et 4 de l'algorithme donnent $R_0 = P_0 Q_0$, $R_2 = P_1 Q_1$ et $S = (P_0 + P_1)(Q_0 + Q_1)$. L'égalité (4.3) montre que le résultat rendu est bien le produit PQ. □

Dans cet algorithme, on commence par diviser en parties les objets considérés, puis on applique l'algorithme à ces objets plus petits obtenus. Il reste ensuite quelques calculs pour déduire le résultat souhaité.

Un tel algorithme est dit de type "diviser pour régner" ou en anglais "divide and conquer". Pour en évaluer la complexité, on utilise le lemme suivant.

Lemme 4.2.2. *[Lemme maître, ou théorème maître] Soit T une fonction de $\mathbb{R}_+$ dans $\mathbb{R}_+$ qui vérifie les propriétés suivantes.*

1. *Il existe des réels $a > 0$ et $b > 1$ tels que $T(x) = aT\left(\frac{x}{b}\right) + O(x)$.*
2. *Pour tout $x \leq 1$, $T(x) \leq 1$.*

Alors

$$T(x) = \begin{cases} O\left(x^{\log_b(a)}\right) & \text{si } a > b \\ O\left(x \log x\right) & \text{si } a = b \\ O\left(x\right) & \text{si } a < b \end{cases}$$

Remarque. Si on utilise un algorithme de type "diviser pour régner" à un objet de taille n, ce lemme s'applique si l'algorithme s'appelle lui-même a fois sur des objets de taille n/b, et si la complexité des calculs supplémentaires pour exploiter les résultats est linéaire.

Démonstration. Soit $k = \lfloor \log_b x \rfloor + 1 = \left\lfloor \frac{\ln x}{\ln b} \right\rfloor + 1$. Donc k est l'entier tel que

$$b^{k-1} \leq x < b^k \tag{4.4}$$

D'après la condition 1 du lemme, il existe un réel $M > 0$ tel que

$$T(x) \leq aT\left(\frac{x}{b}\right) + Mx$$

donc en appliquant la même inégalité à $T\left(\frac{x}{b}\right)$,

$$\begin{aligned} T(x) &\leq a\left(aT\left(\frac{x}{b^2}\right) + M\frac{x}{b}\right) + Mx \\ &\leq a^2 T\left(\frac{x}{b^2}\right) + Mx\left(1 + \frac{a}{b}\right) \end{aligned}$$

Par récurrence,

$$\begin{aligned} T(x) &\leq a^k T\left(\frac{x}{b^k}\right) + Mx\left(1 + \frac{a}{b} + \cdots + \left(\frac{a}{b}\right)^{k-1}\right) \\ &\leq a^k + Mx\left(1 + \frac{a}{b} + \cdots + \left(\frac{a}{b}\right)^{k-1}\right) \end{aligned}$$

La seconde inégalité provient de la condition 2 du lemme puisque $\frac{x}{b^k} < 1$ d'après (4.4).

Si $a = b$, cela donne

$$T(x) \leq a^k + Mxk$$

or $a^k = b^k = b^{k-1}b \leq xb$ d'après la première inégalité de 4.4. De plus, k est en $O(\log x)$. On obtient bien que $T(x) = O(x \log x)$.

Si $a \neq b$,

$$T(x) \leq a^k + Mx\frac{\left(\frac{a}{b}\right)^k - 1}{\frac{a}{b} - 1}$$

Si $a < b$, $a^k < b^k = O(x)$ donc $T(x) = O(x)$.

Si $a > b$, comme $x < b^k$,

$$\begin{aligned} T(x) &\leq a^k + Mb^k\frac{\left(\frac{a}{b}\right)^k}{\frac{a}{b} - 1} \\ &\leq \left(1 + \frac{M}{\frac{a}{b} - 1}\right) a^k \end{aligned}$$

Or $a^k = e^{k \ln a} = e^{k \ln b \frac{\ln a}{\ln b}} = b^{k \log_b a} = O(x^{\log_b(a)})$. □

Corollaire 4.2.3. *Soient P et Q de degré inférieur strictement à n. L'algorithme de Karatsuba calcule PQ avec une complexité algébrique en $O(n^{\log_2(3)}) = O(n^{1.59})$.*

Démonstration. Soit $C(n)$ cette complexité. $C(n) = 3C(n/2)+O(n)$. D'après le lemme, $C(n) = O(n^{\log_2(3)})$. □

Exercice 4.2.4. *Écrire l'algorithme de type "diviser pour régner" pour calculer un produit de polynômes qui utilise l'égalite (4.2) (au lieu de l'égalité (4.3)) et évaluer sa complexité (on remarquera que cette complexité est quadratique donc pas meilleure que celle de la multiplication classique).*

4.3. Algorithmes de tri

On veut trier une liste $l = [a_0, \ldots, a_{n-1}]$ de n éléments de $\mathbb{N}$, c'est-à-dire construire la liste de ses éléments rangés dans l'ordre croissant. Soit $T(n)$ le nombre de comparaisons nécessaires.

4.3.1 Algorithme naïf

On parcourt toute la liste pour repérer le plus petit élément, que l'on met en tête, puis on recommence avec la liste de taille $n-1$ qui reste une fois cet élément enlevé. Alors $T(n) = n - 1 + T(n-1)$ donc

$$T(n) = (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2} \sim \frac{n^2}{2}$$

C'est donc un algorithme de complexité algébrique quadratique.

4.3.2 Tri fusion (ou merge sort)

On suppose pour simplifier que n est une puissance de 2. On partage la liste en deux listes l_0 et l_1 de taille $n/2$. On rappelle l'algorithme pour trier l_0 et l_1 de façon récursive. On obtient deux nouvelles listes l_0 et l_1 triées puis on interclasse de la façon suivante.

On initialise k, i, j à 0. L'entier i va pointer les éléments de l_0, j ceux de l_1 et k ceux de la liste triée l.

Si $l_0[i] \leq l_1[j]$, alors on pose $l[k] = l_0[i]$ et on incrémente i et k de 1 (sauf si $i = n/2 - 1$, auquel cas l_0 a été parcourue et il suffit de compléter l par les éléments qui restent dans l_1). Sinon, on pose $l[k] = l_1[j]$ et on incrémente k et j de 1 (sauf si $j = n/2 - 1$).

Exercice 4.3.1. *Appliquer cet algorithme "Tri fusion" à la liste*

$$l = [1, 4, 7, 9, 2, 3, 5, 6]$$

Exercice 4.3.2. *Écrire l'algorithme "Tri fusion".*

Le nombre de comparaisons $T(n)$ vérifie l'égalité $T(n) = 2T(n/2)+O(n)$. D'après le lemme 4.2.2, on obtient donc que $T(n) = O(n \log n)$.

4.3.3 Application : recherche par dichotomie

Si une liste est triée, on y retrouve plus facilement ce que l'on cherche.

Si la liste l est triée par ordre croissant et si l'on y cherche un élément x, on compare x avec $l[\lfloor n/2 \rfloor]$. Si $x = l[\lfloor n/2 \rfloor]$, on constate que x est dans la liste à l'emplacement $n/2$. Si $x < l[\lfloor n/2 \rfloor]$, on cherche dans $[l[0], \ldots, l[\lfloor n/2 \rfloor - 1]$ et si $x > l[\lfloor n/2 \rfloor]$, on cherche dans $[l[\lfloor n/2 \rfloor + 1], \ldots, l[n-1]]$.

Soit $T(n)$ le nombre de comparaisons nécessaires pour une liste de taille inférieure ou égale à n. Alors $T(n) \leq 2 + T(\lfloor n/2 \rfloor)$. Comme $s(\lfloor n/2 \rfloor) = s(n) - 1$, quand l'algorithme s'est appelé lui-même $s(n) - 1$ fois, la liste est de taille 1. Finalement, $T(n) \leq 2s(n) - 1$. La complexité de l'algorithme est au plus équivalente à $2 \log n$.

4.4. Transformée de Fourier rapide (FFT)

Ce paragraphe porte sur la transformée de Fourier discrète et sur l'algorithme de transformée de Fourier rapide, appelé FFT (fast Fourier transform). Cet algorithme a été mis au point par James Cooley et John Tukey en 1965.

Soit K un corps de caractéristique différente de 2. Soit n une puissance de 2. On note $n = 2^s$. On note $K[X]_{n-1}$ l'anneau des polynômes de degré inférieur ou égal à $n - 1$. On suppose que K contient une racine primitive n-ème de 1, au sens de la définition suivante.

Définition 4.4.1. *On dit qu'un élément ω de K est une racine primitive n-ème de 1 si $\omega^n = 1$ et si $\omega^k \neq 1$ pour tout $k \in [[0, n-1]]$. Cela signifie que ω est d'ordre n dans $(K^*, \cdot)$ (le groupe multiplicatif de K).*

Comme n est une puissance de 2, un élément ω de K est une racine primitive n-ème de 1 si et seulement si $\omega^n = 1$ et $\omega^{n/2} \neq 1$, ce qui est équivalent à $\omega^{n/2} = -1$.

Exemples.

1. Si $K = \mathbb{C}$, $\omega = e^{2i\pi/n}$ est une racine primitive n-ème de 1. L'ensemble des racines primitives n-èmes de 1 est $\{e^{2ik\pi/n} \ : \ \text{pgcd}(k, n) = 1\}$.

2. Si $K = \mathbb{F}_q$, où $q = p^k$ et où n divise $q-1$, alors il existe une racine primitive n-ème de 1 dans $\mathbb{F}_q$. En effet, le groupe $(\mathbb{F}_q^*, \cdot)$ est cyclique d'ordre $q-1$. Soit g un générateur de $\mathbb{F}_q^*$. L'élément $\omega = g^{(q-1)/n}$ est d'ordre n.

Dans la suite du paragraphe, on va identifier $K[X]_{n-1}$ avec K^n : tout polynôme $P = \sum_{i=0}^{n-1} a_i X^i$ pourra être identifié au n-uplet $(a_0, \ldots, a_{n-1})$, que l'on notera aussi $(a_l)_{l\in[[0,n-1]]}$.

Définition 4.4.2. *Soit ω une racine rimitive n-ème de 1. La transformée de Fourier discrète associée à ω est l'application*

$$\begin{aligned} F_w \ : \ K^n &\longrightarrow K^n \\ a = (a_0, \ldots, a_{n-1}) &\longmapsto (P(1), P(\omega), \ldots, P(\omega^{n-1})) \end{aligned}$$

où P est le polynôme défini par a : $P = \sum_{i=0}^{n-1} a_i X^i$. On notera $F_w(a) = F_w(P) = \big(P(\omega^l)\big)_{l\in[[0,n-1]]}$.

Ce calcul est l'évaluation de P en n éléments de K. Cela peut donc se faire en $O(n^2)$ opérations dans K.

On peut aussi le justifier en remarquant que comme F_w est une application linéaire, le calcul de $F_w(a)$ se ramène à la multiplication d'une matrice de $M_n(K)$ par un vecteur de $M_{n,1}(K)$. Cela demande bien $O(n^2)$ opérations dans K.

La FFT calcule F_w avec une meilleure complexité. C'est un algorithme de type "diviser pour régner".

Soit $P = \sum_{i=0}^{n-1} a_i X^i$. On définit

$$\begin{aligned} P_0 &= a_0 + a_2 X + \cdots + a_{n-2} X^{n/2-1} = \sum_{k=0}^{n/2-1} a_{2k} X^k \\ P_1 &= a_1 + a_3 X + \cdots + a_{n-1} X^{n/2-1} = \sum_{k=0}^{n/2-1} a_{2k+1} X^k \end{aligned} \tag{4.5}$$

En d'autres termes, P_0 et P_1 sont respectivement définis par les $n/2$-uplets $(a_0, a_2, \ldots, a_{n-2})$ et $(a_1, a_3, \ldots, a_{n-1})$. Alors

$$P(X) = P_0(X^2) + X P_1(X^2)$$

Pour tout $l \in [[0, n-1]]$,

$$P(\omega^l) = P_0(\omega^{2l}) + \omega^l P_1(\omega^{2l}) \tag{4.6}$$

Or, ω^2 est une racine primitive $n/2$-ème de 1 et

$$F_{\omega^2}(P_0) = \big(P_0(\omega^{2l})\big)_{l\in[[0,n/2-1]]} \quad \text{et} \quad F_{\omega^2}(P_1) = \big(P_1(\omega^{2l})\big)_{l\in[[0,n/2-1]]}$$

Supposons que l'on ait calculé ces transformées de Fourier $F_{\omega^2}(P_0)$ et $F_{\omega^2}(P_1)$. Alors pour tout $l \in [[0, n/2-1]]$, l'égalité (4.6) permet de calculer $P(\omega^l)$. De plus, comme $\omega^{n/2} = -1$,

$$P(\omega^{l+n/2}) = P_0(\omega^{2l}) - \omega^l P_1(\omega^{2l})$$

En résumé, pour $l \in [[0, n/2-1]]$, on peut calculer les valeurs de $P(\omega^l)$ et $P(\omega^{l+n/2})$ par les formules

$$\begin{cases} P(\omega^l) = P_0(\omega^{2l}) + \omega^l P_1(\omega^{2l}) \\ P(\omega^{l+n/2}) = P_0(\omega^{2l}) - \omega^l P_1(\omega^{2l}) \end{cases}$$

On en déduit l'algorithme FFT.

Algorithme 4.4.3. [FFT]

Entrées : n (puissance de 2), ω (racine primitive n-ème de 1), $P \in K[X]_{n-1}$

Sortie : $F_\omega(P) = (P(1), P(\omega), \ldots, P(\omega^{n-1}))$

1. Si $n = 1$, sortir (P)

2. $(P_0, P_1) \leftarrow$ polynômes définis en (4.5) (donc $P(X) = P_0(X^2) + XP_1(X^2)$)

3. $L_0 \leftarrow$ `FFT`$(n/2, \omega^2, P_0)$

4. $L_1 \leftarrow$ `FFT`$(n/2, \omega^2, P_1)$

5. Pour l de 0 à $n/2$-1 :

6. $\quad R_l \leftarrow L_0[l] + \omega^l L_1[l]$

7. $\quad R_{l+n/2} \leftarrow L_0[l] - \omega^l L_1[l]$

8. Sortir $R = (R_0, \ldots, R_{n-1})$

Proposition 4.4.4. *La complexité algébrique de FFT est en $O(n \log n)$ (pour le calcul de la transformée de Fourier discrète d'un n-uplet).*

Démonstration. Soit $C(n)$ la complexité algébrique de la FFT en taille n. Alors $C(n) = 2C(n/2) + O(n)$ donc le lemme 4.2.2 permet de conclure. □

Exercice 4.4.5. *Dans $\mathbb{C}$, i est une racine primitive 4-ème de 1. Soit $P(X) = X^3 + 2X^2 + 3X + 4 \in \mathbb{C}[X]$. Exécuter "à la main"* `FFT`$(4, i, P)$.

Lemme 4.4.6. *Soit $k \in \mathbb{Z}$.*

$$\sum_{j=0}^{n-1} \omega^{kj} = \begin{cases} 0 \text{ si } k \not\equiv 0 \mod n \\ n \text{ si } k \equiv 0 \mod n \end{cases}$$

Proposition 4.4.7. *F_ω est un isomorphisme d'espaces vectoriels et son inverse est $F_\omega^{-1} = \frac{1}{n} F_{\omega^{-1}}$*

Exercice 4.4.8. *Démontrer le lemme 4.4.6 et la proposition 4.4.7.*

Remarque. L'application F_ω effectue l'évaluation simultanée de P en $1, \omega, \omega^2, \ldots, \omega^{n-1}$.

Soit $b = (b_0, \ldots, b_{n-1}) \in K^n$. $F_\omega^{-1} = \frac{1}{n} F_{\omega^{-1}}$ détermine l'unique polynôme P de $K[X]_{n-1}$ tel que

$$P(1) = b_0 \ , \ P(\omega) = b_1 \ , \ \ldots \ , \ P(\omega^{n-1}) = b_{n-1}$$

On dit que P est le polynôme d'interpolation de Lagrange des couples $(1, b_0), \ldots, (\omega^{n-1}, b_{n-1})$.

4.5. Application au produit de deux polynômes

Soient P et Q deux polynômes de $K[X]$ tels que $\deg P + \deg Q < n$. Le produit PQ est uniquement déterminé par ses valeurs en $1, \omega, \ldots, \omega^{n-1}$. L'idée est de calculer $(P(1), P(\omega), \ldots, P(\omega^{n-1}))$ et $(Q(1), Q(\omega), \ldots, Q(\omega^{n-1}))$ par la FFT, on en déduit $(PQ(1), PQ(\omega), \ldots, PQ(\omega^{n-1}))$ par multiplication terme à terme et enfin, on applique la FFT à ce résultat pour obtenir le polynôme PQ par interpolation.

Algorithme 4.5.1. [PRODUIT PAR FFT]

Entrées : n (puissance de 2), ω (racine primitive n-ème de 1), $P, Q \in K[X]$ tels que $\deg P + \deg Q < n$.

Sortie : PQ

1. $R_P \leftarrow \texttt{FFT}(n, \omega, P)$

2. $R_Q \leftarrow \texttt{FFT}(n, \omega, Q)$

3. Pour l de 0 à $n-1$:

4. $\quad R[l] \leftarrow R_P[l] R_Q[l]$

5. Sortir $\frac{1}{n} \texttt{FFT}(n, \omega^{-1}, R)$

Proposition 4.5.2. *Cet algorithme calcule PQ en au plus $O(n \log n)$ opérations dans K.*

Si $\mathrm{car}(K) = 2$, et si n est une puissance de 2, alors $X^n - 1 = (X-1)^n$ et la seule racine n-ème de 1 est 1. Dans ce cas, on peut couper les polynômes en 3 et appliquer la même stratégie.

On peut encore généraliser cet algorithme à un anneau A (à la place de K).

Théorème 4.5.3. [Cantor-Kaltofen] *Soit A un anneau et soit $n \in \mathbb{N}$. On peut multiplier deux polynômes dont la somme des degrés est strictement inférieur à n en $O(n \log n \log\log n)$ opérations dans A.*

On va se contenter ici de décrire l'idée de la preuve.

Soit A un anneau. On suppose que 2 est inversible dans A.

La définition suivante généralise la notion de racine primitive n-ème de 1 au cas d'un anneau.

Définition 4.5.4. *Soit $\omega \in A$. On dit que ω est une racine primitive n-ème de 1 si les conditions suivantes sont vérifiées.*

1. $\omega^n = 1$
2. *Pour tout diviseur premier t de n, $\omega^{n/t} - 1$ n'est pas un diviseur de 0*

Ici, n est une puissance de 2 donc la condition 2 de la définition signifie que $\omega^{n/2} - 1$ n'est pas un diviseur de 0.

Les résultats que l'on a obtenus sur un corps K qui contient une racine primitive n-ème de 1 se généralisent au cas d'un anneau qui contient une racine primitive de l'unité. En effet, le lemme 4.4.6 reste valable dans ce contexte.

Lemme 4.5.5. *Soit ω une racine primitive n-ème de 1 dans A. Soit $k \in \mathbb{Z}$.*

$$\sum_{j=0}^{n-1} \omega^{kj} = \begin{cases} 0 \text{ si } k \not\equiv 0 \mod n \\ n \text{ si } k \equiv 0 \mod n \end{cases}$$

Démonstration. Exercice. □

Supposons maintenant que A ne contient pas de racine primitive n-ème de 1. L'idée est de plonger A dans un anneau plus grand qui contient des racines n-èmes de 1.

Pour cela, on peut considérer $A' = A[Y]/(Y^{n/2}+1)$. Soit y la classe de Y dans ce quotient. Comme $y^{n/2} = -1$, y est une racine primitive n-ème de 1. Mais les éléments de R' sont eux-même des polynômes en y de degré inférieur à $n/2$ et on va travailler dans $R'[X]$: on a considérablement complexifié le

problème. Cependant, il est possible d'améliorer cette idée de la manière suivante.

Soient P et Q dans $A[X]$ tels que $\deg(PQ) < n$. On pose

$$m = 2^{\lfloor k/2 \rfloor} \quad \text{et} \quad m' = 2^{\lceil k/2 \rceil}$$

Ainsi, $m \leq \sqrt{n}$, $m' \leq 2\sqrt{n}$ et $mm' = n$. On écrit

$$P = \sum_{i=0}^{m'-1} P_i X^{mi} \quad \text{et} \quad Q = \sum_{i=0}^{m'-1} Q_i X^{mi}$$

où P_i et Q_i appartiennent à $A[X]_{m-1}$. On définit les polynômes de $A[X][Y]$ suivants.

$$P^* = \sum_{i=0}^{m'-1} P_i Y^i \quad \text{et} \quad Q^* = \sum_{i=0}^{m'-1} Q_i Y^i$$

Alors $\deg_Y(P^*Q^*) < 2m' \leq 4m$. Les coefficients de P^*Q^* sont des polynômes de $A[X]$ de degré strictement inférieur à $2m$ donc pour connaître P^*Q^*, il suffit de le connaître modulo $X^{2m} + 1$, c'est-à-dire de connaître son image dans $A'[Y]$, où $A' = A[X]/(X^{2m} + 1)$.

Dans A', on note ω la classe de X dans $A[X]/(X^{2m}+1)$. C'est une racine primitive $4m$-ème de 1. Ainsi, FFT permet de calculer P^*Q^*. On en déduit $PQ = P^*Q^*(X, X^m)$.

Le calcul de P^*Q^* dans $A'[Y]$ demande $O(m \log m) = O(\sqrt{n} \log n)$ opérations dans A', les multiplications dans A' se faisant par un appel récursif de l'algorithme.

Théorème 4.5.6. [Schönhage-Strassen, 1971] *Il existe un algorithme de multiplication des entiers de complexité binaire $O(s \log s \log\log s)$ (où s majore la taille des entiers).*

Pour ce théorème, l'idée est de poser $a = A(2)$ et $b = B(2)$ où $A = \sum_{i=0}^{s-1} a_i X^i$ et $B = \sum_{i=0}^{s-1} b_i X^i$, on calcule $C = AB$ dans $\mathbb{Z}[X]$ et on évalue C en 2. Ceci étant, il reste des problèmes : les retenues et surtout la taille des coefficients du polynôme C. Nous n'aborderons pas ces problèmes ici.

Chapitre 5

Algorithme d'Euclide et applications

5.1. Les algorithmes d'Euclide et d'Euclide étendu

Soit (a, b) un élément de $\mathbb{N}^2 \setminus \{(0, 0)\}$. L'algorithme d'Euclide repose sur le fait que pour tout $k \in \mathbb{Z}$,

$$\text{pgcd}(a, b) = \text{pgcd}(a, b + ka)$$

En effet, si d divise a et b, alors d divise a et $b + ka$ et réciproquement, si d divise a et $b + ka$, alors d divise a et $(b + ka) - ka = b$.

Notation. Si $b \neq 0$, on note respectivement $\text{rem}(a, b)$ et $\text{quo}(a, b)$ le reste et le quotient de la division euclidienne de a par b.

Algorithme 5.1.1. [Algorithme d'Euclide]
Entrées : a, b dans $\mathbb{N}$ tels que $(a, b) \neq (0, 0)$
Sortie : $\text{pgcd}(a, b)$
1. $x \leftarrow a$, $y \leftarrow b$ (initialisation)
2. Tant que $y \neq 0$:
3. $r \leftarrow \text{rem}(x, y)$
4. $(x, y) \leftarrow (y, r)$
5. Sortir x

Pour évaluer la complexité de cet algorithme, nous commençons par évaluer le nombre t d'exécutions de la boucle "Tant que". Soit $\phi = \dfrac{1 + \sqrt{5}}{2}$ le nombre d'or.

Lemme 5.1.2. *Si $a > b$, alors $t \leq \dfrac{\ln a}{\ln \Phi}$.*

Démonstration. Le nombre d'or Φ est une racine du polynôme $X^2 - X - 1$. Autrement dit,

$$\Phi^2 = \Phi + 1$$

On pose $r_0 = a$, $r_1 = b$ et pour tout $i \geq 2$, r_i et q_{i-1} sont respectivement les reste et quotient de la division euclidienne de r_{i-2} par r_{i-1}. Ainsi, pour tout $i \geq 2$,

$$r_{i-2} = r_{i-1}q_{i-1} + r_i$$

Comme $a > b$, la suite (r_i) est strictement décroissante. Montrons par récurrence que $r_{t-i} \geq \Phi^i$ pour tout $i \geq 0$.

Pour $i = 0 : r_t = \text{pgcd}(a, b) \geq 1 = \Phi^0$. Pour $i = 1 : r_{t-1} \geq 2 > \Phi = \Phi^1$.

On suppose maintenant que pour tout $j \leq i$, $r_{t-j} \geq \Phi^j$.

$$\begin{aligned} r_{t-i-1} &= r_{t-i}q_{t-i} + r_{t-i+1} \\ &\geq r_{t-i} + r_{t-i+1} \end{aligned}$$

puisque $q_{t-i} \geq 1$. On en déduit que

$$r_{t-i-1} \geq \Phi^i + \Phi^{i-1} \quad = \Phi^{i-1}(\Phi + 1) = \Phi^{i+1}$$

puisque $\Phi + 1 = \Phi^2$. □

Exercice 5.1.3. *Soit la suite de Fibonacci (F_n) définie comme suit. $F_0 = 0$, $F_1 = 1$ et $F_{n+1} = F_n + F_{n-1}$. Montrer que $a \geq F_{t+2}$.*

Exercice 5.1.4. *Soit $\Phi' = \dfrac{1 - \sqrt{5}}{2}$. Montrer que pour tout $n \geq 0$,*

$$F_n = \frac{\Phi^n - \Phi'^n}{\sqrt{5}}$$

En déduire que la suite (F_n) est exponentielle.

Théorème 5.1.5. *La complexité binaire de l'algorithme d'Euclide ci-dessus est en $O(s(a)s(b))$. C'est un algorithme quadratique.*

Démonstration. Soit T_i le temps de calcul de la division euclidienne de r_{i-1} par r_i. $T_i = O(s(r_i)s(q_i)) = O(s(r_i)(s(r_{i-1}) - s(r_i) + 1))$. Le temps de calcul total est égal à $T = \sum_{i=1}^{t} T_i$. Il existe donc $M \in \mathbb{R}_+^*$ tel que

$$\begin{aligned} T &\leq M \sum_{i=1}^{t} s(r_i)(s(r_{i-1}) - s(r_i) + 1) \\ &\leq Ms(b) \sum_{i=1}^{t} (s(r_{i-1}) - s(r_i) + 1) \\ &\leq Ms(b)(s(r_0) - s(r_t) + t) \\ &\leq Ms(b)(s(a) + t) \end{aligned}$$

Comme $t \leq \dfrac{\ln a}{ln\Phi} = O(s(a))$, on obtient bien que $T = O(s(a)s(b))$. □

Venons en à l'algorithme d'Euclide étendu. Cet algorithme calcule aussi $d = \text{pgcd}(a, b)$, mais aussi une relation de Bézout, c'est-à-dire des entiers u et v tels que $au + bv = d$.

Pour tout i,

$$\begin{aligned}\begin{pmatrix} r_i \\ r_{i+1} \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \begin{pmatrix} r_{i-1} \\ r_i \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -q_{i-1} \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix} \begin{pmatrix} r_0 \\ r_1 \end{pmatrix}\end{aligned}$$

Il existe donc une matrice $\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix}$ de déterminant $(-1)^i$ telle que

$$\begin{pmatrix} r_i \\ r_{i+1} \end{pmatrix} = \begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

En particulier

$$\begin{pmatrix} d \\ 0 \end{pmatrix} = \begin{pmatrix} u_t & v_t \\ u_{t+1} & v_{t+1} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

On obtient bien la relation de Bézout $d = u_t a + v_t b$.

Algorithme 5.1.6. *Entrées : a, $b \in \mathbb{N}$ tels que $(a, b) \neq (0, 0)$*
Sorties : $d = \text{pgcd}(a, b)$ et $u, v \in \mathbb{Z}$ tels que $au + bv = d$.

1. $x \leftarrow a$, $y \leftarrow b$, $U \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ *(Initialisation)*
2. *Tant que $y \neq 0$:*
3. $\quad (r, q) \leftarrow$ *reste et quotient de la division euclidienne de x par y*
4. $\quad (x, y) \leftarrow (y, r)$
5. $\quad U \leftarrow \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix} U$
6. *Sortir* $(x, U[0, 0], U[0, 1])$

Théorème 5.1.7. *La complexité binaire de cet algorithme est en $O(s(a)s(b))$. C'est un algorithme quadratique.*

Démonstration. Voir les TD. □

Si K est un corps, $K[X]$ est un anneau euclidien. Les algorithmes précédents fonctionnent aussi pour des éléments de $K[X]$.

Théorème 5.1.8. *Soient A, $B \in K[X]$, où K est un corps.*

1. *L'algorithme d'Euclide calcule $D = \mathrm{pgcd}(A, B)$ en $O((\deg A+1)(\deg B+1))$ opérations sur K.*
2. *L'algorithme d'Euclide étendu calcule D, ainsi que deux polynômes U et V de $K[X]$ tels que $AU + BV = D$ en $O((\deg A + 1)(\deg B + 1))$ opérations dans K.*

Remarque. On sait qu'il existe un algorithme de complexité binaire quasi-linéaire pour la multiplication des entiers. On peut en déduire un algorithme de complexité binaire quasi-linéaire pour la division euclidienne et pour les algorithmes d'Euclide.

De même, de tels algorithmes de complexité algébrique quasi-linéaires existent pour les polynômes.

5.2. Inversion modulaire

Théorème 5.2.1. $(\mathbb{Z}/n\mathbb{Z})^* = \{[a]_n \ : \ a \in \mathbb{Z} \quad et \quad \mathrm{pgcd}(a, n) = 1\}$. *Si* $\mathrm{pgcd}(a, n) = 1$, *l'inverse de* $[a]_n$ *est* $[u]_n$ *où* $au + nv = 1$.

Pour les polynômes, on a bien sûr le même résultat.

Théorème 5.2.2. *Soient K un corps et P un polynôme non nul de $K[X]$.*

$$(K[X]/(P))^* = \{[R]_P \ : \ R \in K[X] \ , \ \mathrm{pgcd}(R, P) = 1\}.$$

Si $\mathrm{pgcd}(R, P) = 1$, *l'inverse de* $[R]_P$ *est* $[U]_P$ *où* $UR + VP = 1$.

5.3. Théorème des restes chinois

Soient a et b deux entiers non nuls tels que $\mathrm{pgcd}(a, b) = 1$. L'application

$$\begin{aligned} \mathbb{Z} &\to \mathbb{Z}/a\mathbb{Z} \times \mathbb{Z}/b\mathbb{Z} \\ x &\mapsto ([x]_a, [x]_b) \end{aligned}$$

est un homomorphisme d'anneaux de noyau $ab\mathbb{Z}$. Il définit donc un homomorphisme injectif

$$\begin{aligned} \Phi \ : \ \mathbb{Z}/ab\mathbb{Z} &\to \mathbb{Z}/a\mathbb{Z} \times \mathbb{Z}/b\mathbb{Z} \\ [x]_{ab} &\mapsto ([x]_a, [x]_b) \end{aligned}$$

qui est un isomorphisme pour des raisons de cardinaux. Si l'on connaît des entiers u et v tels que $au + bv = 1$, On peut calculer l'inverse de Φ.

$$\Phi^{-1}([x]_a, [y]_b) = [xbv + yau]_{ab}$$

En effet,

$$bv \equiv \begin{cases} 1 & \mod a \\ 0 & \mod b \end{cases} \quad \text{et} \quad au \equiv \begin{cases} 0 & \mod a \\ 1 & \mod b \end{cases}$$

donc

$$xbv + yau \equiv \begin{cases} x & \mod a \\ y & \mod b \end{cases}$$

Plus généralement, soient $a_1, \ldots, a_n$ des entiers non nuls deux à deux premiers entre eux. Soit $A = \prod_{i=1}^{n} a_i$. On définit encore un isomorphisme d'anneaux

$$\begin{aligned} \Phi \ : \ \mathbb{Z}/A\mathbb{Z} &\to \mathbb{Z}/a_1\mathbb{Z} \times \cdots \times \mathbb{Z}/a_n\mathbb{Z} \\ [x]_A &\mapsto ([x]_{a_1}, \ldots, [x]_{a_n}) \end{aligned}$$

On cherche maintenant à écrire l'application inverse de Φ. Pour $b_1, \ldots, b_n \in \mathbb{Z}$, il s'agit de trouver $x \in \mathbb{Z}$ tel que

$$x \equiv b_i \mod a_i \quad \text{pour tout} \quad i \in [[1, n]]$$

Pour tout $i \in [[1, n]]$, on pose $A_i = \prod_{j \neq i} a_i$. L'entier

$$x = \sum_{i=1}^{n} b_i A_i (A_i^{-1} \mod a_i)$$

convient, où $(A_i^{-1} \mod a_i)$ désigne un représentant dans $\mathbb{Z}$ de $[A_i]_{a_i}^{-1}$. Soient pour tout i des entiers u_i et v_i tels que $A_i u_i + a_i v_i = 1$. L'entier $x = \sum_{i=1}^{n} b_i A_i u_i$ convient.

Proposition 5.3.1. *On peut calculer $[x]_A$ avec une complexité binaire en $O(s(A)^2)$.*

Démonstration. Exercice. □

Cela se transpose immédiatement au cas des polynômes. Soient $P_1, \ldots, P_n$ des polynômes de $K[X]$ deux à deux premiers entre eux. Soit $A = \prod_{i=1}^{n} P_i$. On définit l'isomorphisme de K-algèbres

$$\begin{aligned} \Phi \ : \ K[X]/AK[X] &\to \mathbb{K}[X]/P_1K[X] \times \cdots \times \mathbb{K}[X]/P_nK[X] \\ [R]_A &\mapsto ([R]_{P_1}, \ldots, [R]_{P_n}) \end{aligned}$$

Soit pour tout $i \in [[1,n]]$ $A_i = \prod_{j \neq i} P_i$ et soient $U_i, V_i \in K[X]$ tels que $A_i U_i + P_i V_i = 1$. Soient $Q_1, \ldots, Q_n \in K[X]$. Alors si

$$P = \sum_{i=1}^{n} Q_i A_i U_i,$$

alors pour tout i, $P \equiv Q_i \mod P_i$.

Proposition 5.3.2. *On peut calculer $[P]_A$ avec une complexité algébrique en $O((\deg A + 1)^2)$.*

5.4. Interpolation de Lagrange

Soit K un corps et soient $a_1, \ldots, a_n$ n éléments deux à deux distincts de K. Soient $b_1, \ldots, b_n$ n éléments de K. On cherche $P \in K[X]$ tel que

$$P(a_i) = b_i \text{ pour tout } i \in [[1,n]] \tag{5.1}$$

La division euclidienne de P par $X - a_i$ donne $P(X) = (X - a_i)Q(X) + P(a_i)$ donc $P \equiv P(a_i) \mod X - a_i$. On en déduit que (5.1) est équivalent à

$$P \equiv b_i \mod X - a_i \text{ pour tout } i \in [[1,n]] \tag{5.2}$$

Les $X - a_i$ sont deux à deux premiers entre eux : on reconnaît le problème des restes chinois. L'ensemble des solutions est une classe de $K[X]/(A)$ où $A = \prod_{i=1}^{n}(X - a_i)$. En particulier, ce système de congruence a une unique solution de degré strictement inférieur à n.

Pour tout i, on note $A_i = \prod_{j \neq i}(X - a_j)$. Alors

$$[A_i]^{-1}_{X-a_i} = [A_i(a_i)^{-1}]_{X-a_i} = \left[\prod_{j \neq i} \frac{1}{a_i - a_j}\right]_{X-a_i}$$

La solution de degré strictement inférieur à n est

$$P = \sum_{i=1}^{n} b_i L_i$$

où pour tout i,

$$L_i = \frac{A_i(X)}{A_i(a_i)} = \prod_{j \neq i} \frac{X - a_j}{a_i - a_j}$$

On reconnaît les polynômes d'interpolation de Lagrange.

5.5. Interpolation d'Hermite

Soient $a_1, \dots, a_n$ n éléments de K deux à deux distincts, $m_1, \dots, m_n$ des éléments de $\mathbb{N}$ et $b_{i,j}$ pour $(i,j) \in \mathcal{I} = \{(i,j) \in \mathbb{N}^2 \ : \ 1 \leq i \leq n \ , \ 0 \leq j \leq m_i\}$ des éléments de K. On suppose enfin que pour tout i, $m_i < \mathrm{car}(K)$.

On cherche à résoudre

$$P^{(j)}(a_i) = b_{i,j} \quad \text{pour tout} \quad (i,j) \in \mathcal{I} \tag{5.3}$$

Soit $d = \deg P$ En appliquant la formule de Taylor pour les polynômes à P, on obtient

$$\begin{aligned} P(X) &= \sum_{k=0}^{d} \frac{(X-a_i)^k}{k!} P^{(k)}(a_i) \\ &= P(a_i) + (X-a_i)P'(a_i) + \frac{(X-a_i)^2}{2!}P^{(2)}(a_i) + \cdots + \frac{(X-a_i)^d}{d!}P^{(d)}(a_i) \\ &\equiv P(a_i) + (X-a_i)P'(a_i) + \cdots + \frac{(X-a_i)^{m_i}}{m_i!}P^{(m_i)}(a_i) \mod (X-a_i)^{m_i+1} \\ &\equiv \sum_{k=0}^{m_i} \frac{(X-a_i)^k}{k!} P^{(k)}(a_i) \mod (X-a_i)^{m_i+1} \end{aligned}$$

Soit pour tout i

$$B_i = \sum_{k=0}^{m_i} \frac{(X-a_i)^k}{k!} b_{i,k}$$

Le développement de Taylor à un ordre donné m_i est unique puisque $1, (X-a_i), (X-a_i)^2, \dots, (X-a_i)^{m_i}$ est une base de $K[X]_{m_i}$. Donc le système d'équations (5.3) est équivalent au système de congruences

$$P \equiv B_i \mod (X-a_i)^{m_i+1} \quad \text{pour tout} \quad i \in [[1,n]]$$

C'est donc encore un problème de restes chinois. Il existe un unique polynôme P de degré strictement inférieur à $\sum_{i=1}^{n}(m_i+1) = n + \sum_{i=1}^{n} m_i$ qui vérifie ces congruences. Pour tout i, on note $A_i = \prod_{j \neq i}(X-a_j)^{m_j+1}$ Soient U_i et V_i des polynômes tels que $U_iA_i + V_i(X-a_i)^{m_i+1} = 1$, alors cette solution vérifie la congruence

$$P \equiv \sum_{i=1}^{n} B_i U_i A_i$$

L'ensemble des solutions de (5.3) est $P + K[X]A$ où $A = \prod_{i=1}^{n}(X-a_i)^{m_i+1}$.

5.6. Un exemple d'interpolation d'Hermite

On cherche à trouver un polynôme P de $\mathbb{Q}[x]$ de degré inférieur ou égal à 5 tel que $P(1) = 0$, $P'(1) = -3$, $P''(1) = 2$, $P'''(1) = 42$, $P(2) = 11$ et $P'(2) = 45$.

Ces contraintes sont équivalentes au système suivant

$$\begin{cases} P(x) \equiv P(1) + (x-1)P'(1) + \dfrac{(x-1)^2}{2}P''(1) + \dfrac{(x-1)^3}{6}P'''(1) \mod (x-1)^4 \\ P(x) \equiv P(2) + (x-2)P'(2) \mod (x-2)^2 \end{cases}$$

c'est-à-dire :

$$\begin{cases} P(x) \equiv -3(x-1) + 2\dfrac{(x-1)^2}{2} + 42\dfrac{(x-1)^3}{6} \mod (x-1)^4 \\ P(x) \equiv 11 + 45(x-2) \mod (x-2)^2 \end{cases}$$

Bien sûr, $(x-1)^4$ et $(x-2)^2$ sont premiers entre eux. Nous sommes dons ramenés à un problème de restes chinois.

Sur sage, si l'on exécute les commandes

```
Qx.<x>=PolynomialRing(QQ)
crt([-3*(x-1)+(x-1)^2+7*(x-1)^3,11+45*(x-2)],[(x-1)^4,(x-2)^2])
```

on obtient le résultat $P(x) = x^5 - 3x^3 + x + 1$.

Chapitre 6

Primalité et factorisation dans $\mathbb{Z}$

Soit n un entier naturel non nul. Si n est petit, on peut essayer de le diviser par tous les nombres premiers inférieurs à $\sqrt{n}$ pris dans l'ordre croissant.

On se place dans le cas d'un grand entier n. Une telle recherche devient alors trop longue.

On se pose alors les questions suivantes.

1. n est-il composé (c'est-à-dire, n est-il produit de deux facteurs de $\mathbb{N}$ distincts de 1 et n) ?
2. n est-il premier ?
3. Quelle est la factorisation de n ?

Ces questions semblent à peu près les mêmes, surtout les 1 et 2. En réalité, elles sont distinctes et de difficultés distinctes.

En pratique, on pose d'abord la question 1, et on attend une réponse parmi les deux suivantes.

— Réponse A : oui, n est composé.

— Réponse B : n n'est probablement pas composé.

Si la réponse est la réponse B, on peut ensuite vouloir démontrer que n est effectivement premier, et donc on se pose la question 2, qui est plus difficile que la question 1. Si la réponse est la réponse A, on peut chercher à factoriser n. C'est la question 3), qui est la plus difficile des trois.

Nous travaillerons dans l'anneau $\mathbb{Z}/n\mathbb{Z}$.

6.1. Rappels

Soit G un groupe fini de cardinal N et d'élément neutre 1. Le cardinal de tout sous groupe de G divise N (théorème de Lagrange). En particulier,

l'ordre de tout élément g de G divise N. Ainsi, pour tout $g \in G$,

$$g^N = 1$$

Théorème 6.1.1. *L'ensemble $(\mathbb{Z}/n\mathbb{Z})^*$ des éléments inversibles de $\mathbb{Z}/n\mathbb{Z}$ pour la multiplication est égal à $\{[k]_n \ : \ \text{pgcd}(k,n) = 1\}$. C'est aussi l'ensemble des générateurs du groupe $(\mathbb{Z}/nZ, +)$.*

Définition 6.1.2. *La fonction indicatrice d'Euler φ est l'application de $\mathbb{N} \setminus \{0\}$ dans $\mathbb{N} \setminus \{0\}$ définie par*

$$\varphi(n) = |(\mathbb{Z}/n\mathbb{Z})^*|$$

Le théorème suivant est une conséquence du théorème de Lagrange.

Théorème 6.1.3. [EULER] *Pour tout entier a premier à n,*

$$a^{\varphi(n)} \equiv 1 \mod n$$

Le calcul de φ se fait de la manière suivante.

On remarque que $\varphi(1) = 1$ et pour tout nombre premier p et tout entier naturel non nul α,

$$\varphi(p^\alpha) = p^\alpha - p^{\alpha-1}$$

Si $\text{pgcd}(a,b) = 1$, le théorème des restes chinois permet de constater que

$$(\mathbb{Z}/ab\mathbb{Z})^* \simeq (\mathbb{Z}/a\mathbb{Z})^* \times (\mathbb{Z}/b\mathbb{Z})^*$$

donc $\varphi(ab) = \varphi(a)\varphi(b)$: φ est une fonction multiplicative.

On en déduit que si $n = \prod_{i=1}^{r} p_i^{v_i}$ est la décomposition de n en produit de facteurs premiers,

$$\begin{aligned}\varphi(n) &= \prod_{i=1}^{r} \varphi(p_i^{v_i}) \\ &= \prod_{i=1}^{r} p_i^{v_i} - p_i^{v_i-1} \\ &= n \prod_{i=1}^{r} \left(1 - \frac{1}{p_i}\right)\end{aligned}$$

Le théorème suivant permet de décrire la structure de $((\mathbb{Z}/n\mathbb{Z})^*, \times)$ (grâce au théorème des restes chinois).

Théorème 6.1.4. *Soient p un nombre premier impair et $\alpha \in \mathbb{N} \setminus \{0\}$. Le groupe $((\mathbb{Z}/p^\alpha\mathbb{Z})^*, \times)$ est cyclique d'ordre $\varphi(p^\alpha) = p^\alpha - p^{\alpha-1}$. Il est donc isomorphe à $(\mathbb{Z}/\varphi(p^\alpha)\mathbb{Z}, +)$.*

Si $\alpha \geq 2$, $(\mathbb{Z}/2^\alpha\mathbb{Z}, \times) \simeq (\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2^{\alpha-2}\mathbb{Z}, +)$.

6.2. Tests de non primalité

6.2.1 Test de Fermat

Pour démontrer qu'un entier n n'est pas premier, on utilise le théorème de Fermat suivant (c'est un cas particulier du théorème d'Euler).

Théorème 6.2.1. [FERMAT] *Si n est premier et si* $\text{pgcd}(a, n) = 1$*, alors*

$$a^{n-1} \equiv 1 \mod n$$

Corollaire 6.2.2. *Soit a un élément de $[[1, n-1]]$. Si $a^{n-1} \not\equiv 1 \mod n$, alors n est composé.*

Démonstration. Si a est premier à n, c'est le théorème de Fermat. Sinon, $\text{pgcd}(a, n)$ est un diviseur non trivial de n (n ne divise pas a puisque $1 \leq a < n$). □

On en déduit le test de non primalité de Fermat.

Algorithme 6.2.3. [FERMAT]
Entrée : n
Sortie : "n est composée" ou "il se peut que n soit premier"
1. Choisir a au hasard dans $[[1, n-1]]$
2. $d \leftarrow \text{pgcd}(a, n)$
3. Si $d \neq 1$: sortir "n est composé et d est un facteur non trivial"
4. $b \leftarrow \text{rem}(a^{n-1}, n)$
5. Si $b \neq 1$: sortir "n est composé"
6. Si $b = 1$: sortir "il se peut que n soit premier"

Remarques.

— Le calcul de $\text{pgcd}(a, n)$ peut être omis : si n est grand, il y a très peu de chances que ce pgcd soit distinct de 1.

— Ce test fait intervenir un paramètre choisi au hasard : c'est un algorithme probabiliste.

Le calcul de $\text{rem}(a^{n-1}, n)$ est un calcul dans $\mathbb{Z}/n\mathbb{Z}$ en ce sens qu'il ne faut absolument pas calculer a^{n-1} dans $\mathbb{Z}$ puis réduire, mais réduire modulo n après chaque produit effectué. En utilisant l'exponentiation binaire, cele demande $O(\log n)$ multiplications dans $\mathbb{Z}/n\mathbb{Z}$. La complexité binaire est donc en $O((\log n))^3$ avec la multiplication classique, et $\widetilde{O}((\log n)^2)$ avec la FFT.

Malheureusement, si n est composé, cet algorithme ne le détecte pas nécessairement. Voyons avec quelle probabilité ce test échoue.

Soient donc n un entier composé et p_n la probabilité d'échec du test de Fermat, (c'est-à-dire la probabilité de tirer un entier a tel que $a^{n-1} \equiv 1 \mod n$). Soit M_n l'ensemble des *menteurs (ou faux témoins) de Fermat* pour n.

$$M_n = \{x \in (\mathbb{Z}/n\mathbb{Z})^* \ : \ x^{n-1} = 1\}$$

Alors $p_n = \dfrac{|M_n|}{n-1}$. M_n est le noyau de l'homomorphisme de groupes $x \mapsto x^{n-1}$. C'est donc un sous-groupe de $(\mathbb{Z}/n\mathbb{Z})^*$, par conséquent, $|M_n|$ divise $\varphi(n)$. Ainsi, **si** $M_n \neq (\mathbb{Z}/n\mathbb{Z})^*$, $M_n \leq \frac{\varphi(n)}{2}$ et donc

$$p_n = \frac{|M_n|}{n-1} \leq \frac{|M_n|}{\varphi(n)} \leq \frac{1}{2}$$

Dans ce cas, si l'on renouvelle 10 fois le test, la probabilité de ne pas détecter que n est composé est inférieure à $1/2^{10} \sim 0,00098$.

Le test est donc très satisfaisant dans le cas où $M_n \neq (\mathbb{Z}/n\mathbb{Z})^*$. Mais il existe des entiers n tels que $M_n = (\mathbb{Z}/n\mathbb{Z})^*$. On les appelle *les nombres de Carmichaël*.

Si n est un nombre de Carmichaël, $p_n = \dfrac{\varphi(n)}{n-1} = \dfrac{n}{n-1} \prod_{p|n} \left(1 - \dfrac{1}{p}\right)$.

Exemple. Le plus petit nombre de Carmichaël est $n = 561 = 3 \cdot 11 \cdot 17$. En effet, $n - 1 = 560 = 2^4 \cdot 5 \cdot 7$ et

$$(\mathbb{Z}/561\mathbb{Z})^* \simeq (\mathbb{Z}/3\mathbb{Z})^* \times (\mathbb{Z}/11\mathbb{Z})^* \times (\mathbb{Z}/17\mathbb{Z})^*$$

Or les ordres des groupes du produit de droite sont respectivement 2, 10 et 16, qui divisent tous 560. Donc pour tout $x \in (\mathbb{Z}/561\mathbb{Z})^*$, $x^{560} = 1$.

Nous admettons le résultat suivant.

Théorème 6.2.4. [Alford, Grandville, Pomerance, 1994] *L'ensemble des nombres de Carmichaël est infini.*

6.2.2 Test de Rabin-Miller

On peut affiner le test de Fermat pour être en mesure de détecter aussi les nombres de Carmichaël en utilisant un lemme bien connu.

Lemme 6.2.5. *Soient A un anneau intègre et $x \in A$.*

$$x^2 = 1 \iff x \in \{\pm 1\}$$

Le test de Rabin-Miller repose sur le théorème du même nom.

Théorème 6.2.6. [RABIN-MILLER] *Soit n un nombre premier impair. On pose $n-1 = 2^e m$ où m est un entier impair. Pour tout entier a premier à n,*

— *soit $a^m \equiv 1 \mod n$*

— *soit il existe $i \in [[0, e-1]]$ tel que $a^{2^i m} \equiv -1 \mod n$.*

Démonstration. On suppose que $a^m \not\equiv 1 \mod n$. On va montrer qu' il existe $i \in [[0, e-1]]$ tel que $a^{2^i m} \equiv -1 \mod n$.

Comme n est impair, l'entier e dans l'égalité $n-1 = 2^e m$ vérifie $e \geq 1$. Si n est premier, $\mathbb{Z}/n\mathbb{Z}$ est un corps. Pour tout $a \in (\mathbb{Z}/n\mathbb{Z})^*$, $a^{2^e m} = a^{n-1} = 1$. L'ensemble $I = \{i \in [[0, e]] \ : \ a^{2^i m} = 1\}$ est non vide et $0 \notin I$. Soit $i_0 = \min I$. Alors $i_0 \geq 1$.

$$\left(a^{2^{i_0-1} m}\right)^2 = a^{2^{i_0} m} = 1$$

D'après le lemme 6.2.5, on en déduit que $a^{2^{i_0-1} m} \in \{\pm 1\}$. Comme i_0 est minimal dans I, $a^{2^{i_0-1} m} = -1$. □

Algorithme 6.2.7. RABIN-MILLER

Entrées : n : entier impair, e, m : $n-1 = 2^e m$, $a \in [[1, n-1]]$.
Sortie : "n est composé" ou "il est possible que n soit premier"
1. $d \leftarrow \text{pgcd}(a, n)$
2. Si $d \neq 1$: sortir "n est composé et d est un facteur"
3. $b \leftarrow \text{rem}(a^m, n)$
4. Si $b \equiv 1 \mod n$ ou s'il existe $i \in [[0, e-1]]$ tel que $b^{2^i} \equiv -1 \mod n$:
5. Sortir "Il se peut que n soit premier"
6. Sortir "n est composé"

La complexité de ce test est la même que celle du test de Fermat. On l'applique à des entiers a pris au hasard dans $[[1, n-1]]$.

Supposons que n soit composé, et étudions la probabilité p_n de tirer un menteur, c'est-à-dire un entier a qui ne permet pas de détecter que n est composé. On note M'_n l'ensemble des menteurs de Rabin-Miller.

$$M'_n = \{x \in (\mathbb{Z}/n\mathbb{Z})^* \ : \text{soit } x^m = 1 \\ \text{soit il existe } i \in [[0, e-1]] \text{ tel que } x^{2^i m} = -1\}$$

Théorème 6.2.8. *On suppose que n est un nombre composé impair. Alors*

$$\text{card}(M'_n) \leq \frac{n-1}{4}$$

La probabilité p_n est donc inférieure à $1/4$.

Si l'on fait 10 essais successifs sur un entier impair composé, la probabilité de ne pas détecter qu'il est composé est inférieure à $4^{-10} \sim 9,5 \cdot 10^{-7}$.

Définition 6.2.9. *Soit n un entier impair. Si le test de Rabin-Miller appliqué à n avec l'entier a indique que n pourrait être premier, on dit que n est pseudo-premier fort de base a.*

Exemple. Soit $n = F_r = 2^{2^r} + 1$. On appelle cet entier le r-ème nombre de Fermat.

$$F_r - 1 = 2^{2^r}$$

Dans le théorème de Rabin-Miller, $e = 2^r$ et $m = 1$. $r \in [[0, e-1]]$ et $2^{2^r} \equiv -1 \mod F_r$, donc F_r est pseudo-premier fort de base 2.

Les entiers F_0, F_1, F_2, F_3 et F_4 sont premiers. Pour $r \in [[5, 32]]$, F_r est composé.

Dans la preuve du théorème 6.2.8, on utilisera le lemme suivant.

Lemme 6.2.10. *Soient G un groupe cyclique d'ordre n et $x \in G$. Soient k un entier et $d = \text{pgcd}(n, k)$.*

$$x^k = 1 \iff x^d = 1$$

Ainsi, $\{x \in G \ : \ x^k = 1\} = \{x \in G \ : \ x^d = 1\}$: c'est l'unique sous-groupe d'ordre d de G.

Démonstration. Exercice (utiliser une relation de Bézout $un + vk = d$). □

Lemme 6.2.11.

- *Soient $\mathcal{P}$ l'ensemble des diviseurs premiers de n et*

$$n = \prod_{p \in \mathcal{P}} p^{\alpha_p}$$

la décomposition de n en produit de facteurs premiers.

- *Soit r le cardinal de $\mathcal{P}$.*

- *Pour tout $p \in \mathcal{P}$, on note $p - 1 = 2^{e_p} m_p$ où m_p est impair.*

- *Soit $E = \min\{e_p \ : \ p \in \mathcal{P}\}$.*

Avec ces notations,

$$\text{card}(M'_n) = \left(1 + \frac{2^{rE} - 1}{2^r - 1}\right) \prod_{p \in \mathcal{P}} \text{pgcd}(p - 1, m)$$

Démonstration. Soit $p \in \mathcal{P}$. Le groupe$(\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^*$ est cyclique d'ordre $p^{\alpha_p-1}(p-1)$. Alors pour tout $i \in \mathbb{N}$,

$$\begin{aligned}\operatorname{card}\left\{x \in (\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^* \ : \ x^{2^i m} = 1\right\} &= \operatorname{pgcd}(p^{\alpha_p-1}(p-1), 2^i m) \\ &= 2^{\min(i,e_p)} \operatorname{pgcd}(p-1, m)\end{aligned}$$

(puisque p est premier à $2^i m$). Comme $(\mathbb{Z}/p^{\alpha_p})^*$ est cyclique,

$$\{y \in (\mathbb{Z}/p^{\alpha_p})^* \ \ y^2 = 1\} = \{1, -1\}$$

Ainsi,

$$\begin{aligned}\operatorname{card}\left\{x \in (\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^* \ : \ x^{2^i m} = -1\right\} &= \operatorname{card}\left\{x \in (\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^* \ : \ x^{2^{i+1} m} = 1\right\} \\ &\quad - \operatorname{card}\left\{x \in (\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^* \ : \ x^{2^i m} = 1\right\} \\ &= \begin{cases} 0 & \text{si } i \geq e_p \\ 2^i \operatorname{pgcd}(p-1, m) & \text{si } i < e_p \end{cases}\end{aligned}$$

puisque si $i \geq e_p$, $\min(i, e_p) = \min(i+1, e_p) = e_p$ et si $i < e_p$, $2^{\min(i+1,e_p)} - 2^{\min(i,e_p)} = 2^{i+1} - 2^i = 2^i$.

Comme $E = \min\{e_p \ : \ p \in \mathcal{P}\}$, on en déduit que dès que $i \geq E$, il existe $p \in \mathcal{P}$ tel que $\operatorname{card}\left\{x \in (\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^* \ : \ x^{2^i m} = -1\right\} = 0$.

On utilise ensuite l'isomorphisme des restes chinois.

$$(\mathbb{Z}/n\mathbb{Z})^* \simeq \prod_{p \in \mathcal{P}} (\mathbb{Z}/p^{\alpha_p})^*$$

Pour tout entier a,

$$a^m \equiv 1 \mod n \iff (a^m \equiv 1 \mod p^{\alpha_p} \ \forall p \in \mathcal{P})$$

et pour tout $i \in [[0, e-1]]$

$$a^{2^i m} \equiv -1 \mod n \iff (a^{2^i m} \equiv -1 \mod p^{\alpha_p} \ \forall p \in \mathcal{P})$$

Ainsi,

$$\begin{aligned}\operatorname{card}\{x \in (\mathbb{Z}/n\mathbb{Z})^* \ : \ x^m = 1\} &= \prod_{p \in \mathcal{P}} \operatorname{card}\{x \in (\mathbb{Z}/p^{\alpha_p}\mathbb{Z})^* \ : \ x^m = 1\} \\ &= \prod_{p \in \mathcal{P}} \operatorname{pgcd}(p-1, m)\end{aligned}$$

et pour tout $i \in [[0, e-1]]$,

$$\operatorname{card}\left\{x \in (\mathbb{Z}/n\mathbb{Z})^* \ : \ x^{2^i m} = -1\right\} = \begin{cases} 0 & \text{si } i \geq E \\ \displaystyle\prod_{p\in\mathcal{P}} 2^i \operatorname{pgcd}(p-1, m) & \text{si } i < E \end{cases}$$

Comme $r = \operatorname{card}\mathcal{P}$, on en déduit que

$$\begin{aligned} \operatorname{card}(M'_n) &= \left(1 + \sum_{i=0}^{E-1} 2^{ri}\right) \prod_{p\in\mathcal{P}} \operatorname{pgcd}(p-1, m) \\ &= \left(1 + \frac{2^{rE}-1}{2^r - 1}\right) \prod_{p\in\mathcal{P}} \operatorname{pgcd}(p-1, m) \end{aligned}$$

□

Démonstration du théorème 6.2.8.

On utilise le résultat du lemme précédent et on veut majorer $Q = \dfrac{M'_n}{n-1}$. Distinguons les cas $r = 1$ et $r > 1$.
Cas 1 : $r = 1$. Alors $n = p^{\alpha_p}$, $p - 1 = 2^{e_p} m_p$ et $E = e_p$. Comme m est impair, $\operatorname{pgcd}(p-1, m) \leq 2^{-e_p}(p-1)$.

$$\begin{aligned} Q &= \frac{2^{e_p} \operatorname{pgcd}(m, p-1)}{p^{\alpha_p} - 1} \leq \frac{p-1}{p^{\alpha_p} - 1} = \frac{p-1}{(p-1)(p^{\alpha_p - 1} + \cdots + 1)} \\ &\leq \frac{1}{p+1} \leq \frac{1}{4} \end{aligned}$$

Cas 2 : $r > 1$. Là encore, comme m est impair,

$$\operatorname{pgcd}(m, p-1) \leq \frac{p-1}{2^{e_p}} \leq \frac{p-1}{2^E} \tag{6.1}$$

De plus,

$$\prod_{p\in\mathcal{P}} (p-1) \leq \prod_{p\in\mathcal{P}} (p^{\alpha_p} - 1) \leq \left(\prod_{p\in\mathcal{P}} p^{\alpha_p}\right) - 1 = n - 1 \tag{6.2}$$

On en déduit que

$$Q \leq \frac{1}{2^{rE}} \left(1 + \frac{2^{rE}-1}{2^r - 1}\right)$$

Le terme de droite décroît quand E croît (pour le voir, il suffit de l'écrire $\frac{1}{2^{rE}}\left(1-\frac{1}{2^r-1}\right)+\frac{1}{2^r-1}$). Sa valeur maximale est donc atteinte pour $E=1$. On obtient $p_r \leq \frac{2}{2^r}$ donc

$$Q \leq \frac{1}{4} \text{ si } r \geq 3 \quad \text{et} \quad Q \leq \frac{1}{2} \text{ si } r = 2$$

Le théorème est donc démontré, sauf dans le cas où $r = 2$. Dans cette situation, on peut en exercice affiner ce résultat en suivant les indications suivantes.

— s'il existe $p \in \mathcal{P}$ tel que $p-1$ ne divise pas $n-1$, on peut améliorer les inégalités dans (6.1).

— si n a au moins un facteur carré (c'est-à-dire s'il existe $p \in \mathcal{P}$ tel que $\alpha_p \geq 2$), on peut améliorer la première inégalité dans (6.2).

— Reste le cas où pour tout $p \in \mathcal{P}$, $p-1$ divise $n-1$ et si n n'a pas de facteur carré, (avec la condition $r = 2$). On peut montrer (en exercice) qu'un tel entier n n'existe pas !

□

6.3. Tests de primalité

Dans ce paragraphe, n est un nombre impair que l'on croit premier : on suppose que plusieurs tests de Rabin-Miller nous ont donné cette conviction. On veut démontrer qu'il est effectivement premier.

6.3.1 Tests basés sur le groupe $(\mathbb{Z}/n\mathbb{Z})^*$

L'entier n est un nombre premier si et seulement si $|(\mathbb{Z}/n\mathbb{Z})^*| = n-1$. Dans ce cas, le groupe $((\mathbb{Z}/n\mathbb{Z})^*, \times)$ est cyclique d'ordre $n-1$.

Pour montrer que n est premier, il suffit de trouver un élément d'ordre $n-1$ dans $(\mathbb{Z}/n\mathbb{Z})^*$.

Lemme 6.3.1. *Soit G un groupe et soit $x \in G$. Alors x est d'ordre d si et seulement si les propriétés suivantes sont vérifiées.*

1. $x^d = 1$
2. *Pour tout diviseur premier l de d, $x^{d/l} \neq 1$.*

Démonstration. L'implication directe est claire. Réciproquement, supposons que x vérifie les conditions 1 et 2. Comme $x^d = 1$, x est d'ordre fini d' tel

que d' divise d. Notons $d'k = d$ et raisonnons par l'absurde : si $k \neq 1$, il est divisible par un nombre premier l, et alors

$$x^{d/l} = x^{d'k/l} = 1$$

ce qui est contraire à la condition 2. □

Pour démontrer que n est premier, on peut utiliser l'algorithme ci-dessous, qui nécessite la connaissance de l'ensemble des diviseurs premiers de $n-1$. Dans la suite de ce paragraphe, la décomposition de $n-1$ en produit de facteurs premiers est

$$n-1 = \prod_{i=1}^{r} p_i^{v_i}$$

Algorithme 6.3.2. [Test de Lucas-Lehmer]

Entrées : n (nombre entier vraisemblablement premier), $p_1, \ldots, p_r$: diviseurs premiers de $n-1$.

Sortie : "n est premier" ou "Échec" ou "n est composé"

1. $a \leftarrow$ entier choisi au hasard dans $[[1, n-1]]$

2. Si $a^{n-1} \not\equiv 1 \mod n$: sortir "n est composé"

3. Pour i de 1 à r :

4. si $a^{(n-1)/p_i} \equiv 1 \mod n$: sortir "Échec"

5. Sortir "n est premier" et a est un certificat.

Remarques.

1. En cas de succès, l'algorithme rend aussi l'entier a. C'est un générateur de $(\mathbb{Z}/n\mathbb{Z})^*$. On le garde comme "certificat de primalité".
2. Cet algorithme présente un inconvénient majeur : il nécessite la connaissance de tous les diviseurs premiers de $n-1$. Comme la factorisation est un problème réputé plus difficile, ce test ne semble pas praticable. Il arrive cependant souvent que les diviseurs premiers de l'entier $n-1$ soient petits (on dit alors que $n-1$ est friable).

Voyons la complexité de l'algorithme.

Pour cela, il faut majorer le nombre r de diviseurs premiers de $n-1$.

$$n-1 = \prod_{i=1}^{r} p_i^{v_i} \geq 2^r$$

donc

$$r \leq \log(n-1)$$

Les calculs de puissances dans l'algorithme se font en $O(\log n)$ multiplications dans $(\mathbb{Z}/n\mathbb{Z})^*$. Si $M(n)$ est la complexité binaire ce chacune de ces multiplications, on voit que la complexité binaire de l'algorithme est en $O((\log n)^2 M(n))$.

Évaluons maintenant la probabilité de succès. Le cardinal de l'ensemble des générateurs de $((\mathbb{Z}/n\mathbb{Z})^*, \times) \simeq (\mathbb{Z}/(n-1)\mathbb{Z}, +)$ est égal à $\varphi(n-1)$. Soit P la probabilité pour que la classe $[a]_n$ soit un générateur.

$$P = \frac{\varphi(n-1)}{n-1} = \prod_{i=1}^{r} \left(1 - \frac{1}{p_i}\right) \geq \left(\frac{1}{2}\right)^r$$

En fait, on peut minorer P de façon beaucoup plus fine et montrer que le nombre de tirages nécessaires est en $O(\ln \ln n)$. C'est déjà bien, mais on peut améliorer cela en cherchant pour chaque i un élément d'ordre $p_i^{v_i}$.

Proposition 6.3.3. *Soit n un entier impair strictement supérieur à 1. Alors n est premier si et seulement si pour tout $i \in [[1, r]]$, il existe $a_i \in \mathbb{Z}$ qui vérifie les propriétés suivantes.*

1. $a_i^{n-1} \equiv 1 \mod n$
2. $a_i^{(n-1)/p_i} \not\equiv 1 \mod n$

Démonstration. Soit $a = \prod_{i=1}^{r} a_i$. Les conditions 1 et 2 montrent que $[a]_n$ est d'ordre $n-1$ dans $(\mathbb{Z}/n\mathbb{Z})^*$. En effet, la condition 2 assure que $p_i^{v_i}$ divise l'ordre de a_i. Le lemme 6.3.4 qui suit permet ensuite de conclure. □

Lemme 6.3.4. *Soit G un groupe abélien. Soient x et y deux éléments de G. L'ordre du produit xy est le* ppcm *des ordres de x et y.*

Démonstration. Exercice. □

Algorithme 6.3.5. [Test de Pocklington-Lehmer]

Entrées : n un entier impair et la décomposition de $n-1$
Sortie : "n est premier" ou "n est composé
1. Pour i de 1 à r :
2. $\beta \leftarrow 1$
3. *Tant que $\beta = 1$:*
4. *Choisir $\alpha \in [[1, n-1]]$ au hasard*
5. $\beta \leftarrow \mathrm{rem}(\alpha^{(n-1)/p_i}, n)$
6. *Si $\beta^{p_i} \not\equiv 1 \mod n$: sortir "n est composé"*
7. $a_i = \mathrm{rem}(\alpha^{(n-1)/p_i^{v_i}}, n)$

8. Sortir "n est premier", $a = rem\left(\prod_{i=1}^{r} a_i, n\right)$ est un certificat de primalité.

Remarque. Les calculs des a_i du pas 7 et du produit a permettent de donner un certificat. Ces calculs ne sont pas nécessaires si l'on ne souhaite pas conserver ce certificat.

Calculons la probabilité de succès. L'ensemble $H = \{x \in (\mathbb{Z}/n\mathbb{Z})^* : x^{(n-1)/p_i} = 1\}$ est le sous-groupe de $(\mathbb{Z}/n\mathbb{Z})^*$ d'ordre $\dfrac{n-1}{p_i}$. Quand on tire α au pas 4, la probabilité d'échec est donc $\dfrac{|H|}{n-1} = \dfrac{1}{p_i}$. Soit X le nombre de tirages nécessaires. L'espérance de X est

$$\begin{aligned} E(X) &= \sum_{k=1}^{+\infty} kP(X=k) \\ &= \sum_{k=1}^{+\infty} k\left(1-\frac{1}{p_i}\right)\left(\frac{1}{p_i}\right)^{k-1} \\ &= \left(1-\frac{1}{p_i}\right)\frac{1}{\left(1-\frac{1}{p_i}\right)^2} \\ &= \frac{1}{1-\frac{1}{p_i}} \\ &\leq 2 \end{aligned}$$

Grâce à ce test, on peut donc montrer que n est premier avec une complexité binaire en $O((\log n)^2 M(n))$ en moyenne, à condition de connaître la décomposition de $n-1$ en produit de facteurs premiers.

Il existe des raffinements de ce test pour lesquels on n'a besoin que d'une factorisation partielle de $n-1$.

Proposition 6.3.6. *On suppose que $n-1 = FU$ où $\mathrm{pgcd}(F,U) = 1$ et où $F > \sqrt{n}$. n est premier si et seulement si pour tout diviseur premier p de F, il existe a_p tel que*

$$a_p^{n-1} \equiv 1 \mod n \quad et \quad \mathrm{pgcd}(a_p^{(n-1)/p} - 1, n) = 1$$

Proposition 6.3.7. *On suppose que $n-1 = FU$ où $\mathrm{pgcd}(F,U) = 1$ et $U > 1$. On suppose aussi que tous les diviseurs premiers de U sont strictement*

supérieurs à B, où $BF \geq \sqrt{n}$. Alors n est premier si et seulement si les conditions suivantes sont vérifiées.

1. *Pour tout diviseur premier p de F, il existe a_p tel que*

$$a_p^{n-1} \equiv 1 \mod n \quad et \quad \text{pgcd}(a_p^{(n-1)/p} - 1, n) = 1$$

2. *Il existe a tel que*

$$a^{n-1} \equiv 1 \mod n \quad et \quad \text{pgcd}(a^F - 1, n) = 1$$

Ce dernier résultat peut être utilisé si l'utilisation de la table de tous les nombres premiers inférieurs à B permet une factorisation suffisante de $n-1$.

6.3.2 Autres tests

Citons brièvement d'autres tests de primalité.

— Il existe des tests basés sur le corps $\mathbb{F}_{n^2}$ qui reposent sur la factorisation de $n+1$: si $n-1$ n'est pas friable, peut-être que $n+1$ l'est.

— Les tests les plus efficaces actuellement utilisent les courbes elliptiques. Toute courbe elliptique E sur $\mathbb{F}_n$ est munie d'une loi de groupe. L'idée est d'utiliser une courbe elliptique dont l'ordre est friable.

— L'algorithme AKS (Agramal-Kazal-Saxena) est un algorithme déterministe de complexité polynomiale.

6.4. Crible d'Eratosthène

Soit B une borne donnée. On souhaite établir la liste des nombres premiers inférieurs ou égaux à B. On connaît un équivalent en l'infini de la taille d'une telle liste grâce au théorème des nombres premiers, démontré en 1896 par Jacques Hadamard et Charles-Jean de La Vallée Poussin de façon indépendante.

Théorème 6.4.1. [Nombres premiers]

$$\text{Card}\{p \text{ premiers } : \ p \leq x\} \sim \frac{x}{\ln x}$$

Le crible d'Eratosthène fonctionne de la manière suivante. On établit la liste des entiers entre 2 et B. On élimine tous les multiples de 2 distinct de 2. Puis on élimine les multiples de 3. On itère le procédé jusqu'à $\sqrt{B}$.

On s'arrête à $\sqrt{B}$: en effet, si $n \leq B$ n'est pas premier, notons p son plus petit facteur premier. Alors $n = pq$ où $q \geq p$ donc $p^2 \leq pq = n \leq B$.

Algorithme 6.4.2. [ERATOSTHÈNE]
Entrée : B
Sortie : la liste des nombres premiers inférieurs ou égaux à B.

1. $T \leftarrow [1$ *pour* i *entre* 1 *et* $B]$
2. *Pour* n *de* 2 *à* $\lfloor\sqrt{B}\rfloor$ *:*
3. *Si* $T[n] = 1$ *:*
4. *Pour* k *de* 2 *à* $\left\lfloor \dfrac{B}{n} \right\rfloor$ *:*
5. $T[kn] = 0$
6. *Sortir* $[n \ : \ n \in [[2, B]]$ *et* $T[n] = 1]$

Le coût est exponentiel. En effet,

$$B \sum_{n \leq \sqrt{B}} \frac{1}{n} \sim B \ln \sqrt{B} = \frac{1}{2} B \ln B$$

L'algorithme demande donc $O(B \ln B)$ multiplications dans $\mathbb{N}$.

6.5. Factorisation

On ne connaît pas d'algorithme de factorisation dans $\mathbb{N}$ qui soit polynomial. Cela assure une certaine solidité au système RSA.

Les meilleurs algorithmes connus sont sous-exponentiels. Citons en trois.

— Algorithme de Dixon : $O\left(e^{2\sqrt{2}(\ln n)^{1/2}(\ln\ln n)^{1/2}}\right)$

— Crible quadratique : $O\left(e^{(1+o(1))(\ln n)^{1/2}(\ln\ln n)^{1/2}}\right)$

— Crible algébrique : $O\left(e^{c(\ln n)^{1/3}(\ln\ln n)^{2/3}}\right)$ où $c > 1$

Le crible algébrique est actuellement le plus efficace. Les cribles algébrique et quadratique utilisent la même idée de base que l'algorithme de Dixon. C'est cet algorithme de Dixon qui est décrit ici.

À l'origine, l'idée vient de factorisations effectuées par Fermat qui utilisaient le fait que si $n = x^2 - y^2$, alors $n = (x + y)(x - y)$. Cela permet de factoriser des entiers dont les facteurs premiers sont proches de $\sqrt{n}$.

Exemple. Soit $n = 2\,027\,651\,281$. Alors $\lfloor\sqrt{n}\rfloor = 45\,029$. On part de $x = 45\,030$, on regarde si $x^2 - n$ est un carré. Si tel n'est pas le cas, on change x en $x + 1$ et on recommence. Pour $x = 45\,041$, on trouve que $x^2 - n = 1\,020^2$, donc

$$n = 45\,041^2 - 1\,020^2 = 46\,061 \times 44\,029$$

On peut modifier cette idée en remarquant que si

$$x^2 \equiv y^2 \mod n \quad \text{et} \quad x \not\equiv \pm y \mod n$$

alors $\text{pgcd}(x-y, n)$ est un facteur non trivial de n.

Suivant cette idée, on obtient l'algorithme suivant.

On tire $x \in [[\sqrt{n}, n-1]]$ au hasard, puis on calcule $r = \text{rem}(x^2, n)$. S'il existe $s \in \mathbb{N}$ tel que $r = s^2$, on calcule $\text{pgcd}(x-s, n)$. Malheureusement, $\text{card}\{s^2 \in \mathbb{N} \ : \ 0 < s^2 < n\} = \lfloor \sqrt{n-1} \rfloor$: il y a peu de chances que r soit un carré.

La stratégie utilisée dans l'algorithme de Dixon consiste à collecter des relations données par des factorisations d'entiers x_i^2 suivant une table de nombres premiers donnée jusqu'à pouvoir en déduire une relation

$$x^2 \equiv y^2 \mod n$$

Exemple. Factorisons $n = 2\,183$. On tire au hasard un entier, on le met au carré, on réduit modulo n et on factorise le résultat obtenu si c'est facile.
Premier nombre tiré : 453. On vérifie que

$$453^2 \equiv 7 \mod n$$

Second nombre : 1024.

$$1024^2 \equiv 3 \mod n$$

Troisième nombre : 209.

$$209^2 \equiv 21 = 3 \times 7 \mod n$$

On n'a obtenu aucun carré, mais on remarque que

$$(453 \times 1024 \times 209)^2 \equiv (3 \times 7)^2 \mod n$$

c'est-à-dire, en réduisant modulo n

$$687^2 \equiv 21^2 \mod n$$

Suivant l'idée de départ, on calcul $\text{pgcd}(687 - 21, n) = 37$. C'est un facteur de $2\,183$.

$$2\,183 = 37 \times 59$$

Pour mener à bien cette factorisation, nous avons utilisé la table des nombres premiers $\{3, 7\}$. L'algorithme de Dixon utilise une table $\mathcal{B} = \{p_1 < p_2 < \cdots < p_k\}$ de tous les nombres premiers inférieurs à une certaine borne B.

Algorithme 6.5.1. [DIXON]

Entrées : n, B, $\mathcal{B} = \{p_1 < p_2 < \cdots < p_k\}$: ensemble des nombres premiers inférieurs à B.

Sortie : un facteur non trivial de n ou "Échec"

1. *Tirer au hasard $x \in [[\sqrt{n}, n-1]]$. Si $r = \mathrm{rem}(x^2, n)$ se factorise sur $\mathcal{B}$, on garde la relation*
$$x_1^2 \equiv p_1^{e_{1,1}} \dots p_k^{e_{k,1}} \mod n$$
2. *Itérer le pas 1 jusqu'à obtenir $k+1$ relations*
$$\begin{cases} x_1^2 \equiv p_1^{e_{1,1}} \dots p_k^{e_{k,1}} \mod n \\ \vdots \\ x_{k+1}^2 \equiv p_1^{e_{1,k+1}} \dots p_k^{e_{k,k+1}} \mod n \end{cases}$$
3. *Résoudre le système à k équations et $k+1$ inconnues $\varepsilon_i \in \{0,1\}$*
$$\begin{cases} \varepsilon_1 e_{1,1} + \cdots + \varepsilon_{k+1} e_{1,k+1} \equiv 0 \mod 2 \\ \vdots \\ \varepsilon_1 e_{k,1} + \cdots + \varepsilon_{k+1} e_{k,k+1} \equiv 0 \mod 2 \end{cases}$$
Soit $(\varepsilon_1, \dots, \varepsilon_{k+1})$ une solution non nulle
4. *Pour i de 1 à k : $f_i \leftarrow \dfrac{1}{2} \displaystyle\sum_{j=1}^{k+1} \varepsilon_j e_{i,j}$*
5. $g \leftarrow \mathrm{pgcd}\left(x_1^{\varepsilon_1} \dots x_{k+1}^{\varepsilon_{k+1}} - p_1^{f_1} \dots p_k^{f_k}, n\right)$
6. *Si $g \notin \{1, n\}$, sortir g*
7. *Sinon sortir "Échec"*

Remarques

— Au pas 3, le système à résoudre est un système linéaire sans second membre à k équations et $k+1$ inconnues sur $\mathbb{F}_2$. L'ensemble des solutions est un espace vectoriel sur $\mathbb{F}_2$ de dimension supérieure ou égale à 1. Il y a donc des solutions non nulles. On résout ce système en utilisant l'algorithme du pivot de Gauss.

— Au pas 4, les f_i sont bien des entiers puisque d'après le pas 3, les sommes considérées sont des nombres pairs. On obtient une égalité de la forme souhaitée :
$$\left(x_1^{\varepsilon_1} \dots x_{k+1}^{\varepsilon_{k+1}}\right)^2 \equiv \left(p_1^{f_1} \dots p_k^{f_k}\right)^2 \mod n$$

Nous ne faisons pas ici l'analyse de la complexité de cet algorithme (voir [G-G]). Signalons toutefois que pour la borne B, un choix optimal est

$$B \sim e^{\sqrt{\ln n \ln\ln n}}$$

Chapitre 7

Factorisation dans $\mathbb{F}_q[X]$

Soient q une puissance d'un nombre premier et $\mathbb{F}_q$ le corps de cardinal q.

Dans ce chapitre, on cherche à savoir comment reconnaître les polynômes de $\mathbb{F}_q[X]$ qui sont irréductibles, et à factoriser ceux qui ne le sont pas.

7.1. Corps finis

Commençons par quelques rappels sur les corps finis.

Les premiers corps finis que l'on rencontre sont les corps $\mathbb{Z}/p\mathbb{Z}$, où p désigne un nombre premier. On note ce corps $\mathbb{F}_p$.

Soit K un corps commutatif. On note 1_K l'élément neutre de la multiplication de K. Pour tout entier naturel non nul k, on note

$$k \times 1_K = \underbrace{1_K + \cdots + 1_K}_{k\times}$$

pour tout entier $k < 0$, $k \times 1$ est l'opposé de $|k| \times 1$ et $0 \times 1 = 0_K$ (l'élément neutre de l'addition de K=). Soit l'application

$$\begin{aligned} f \ : \ & \mathbb{Z} \to K \\ & k \mapsto k \times 1 \end{aligned}$$

f est un homomorphisme d'anneaux. Il induit donc une injection

$$\mathbb{Z}/\ker f \hookrightarrow K$$

Or K est un corps, donc $\mathbb{Z}/\ker f$ est intègre. Cela signifie que $\ker f$ est un idéal premier de $\mathbb{Z}$.

— Soit $\ker f = \{0\}$. On dit alors que K est de caractéristique nulle.

— Soit il existe un nombre premier p tel que $\ker f = p\mathbb{Z}$. On dit alors que K est de caractéristique p.

On note $\mathrm{car}(K)$ la caractéristique de K.

Si $\mathrm{car}(K) = 0$, f est une injection donc K est infini.

Si K est un corps fini, il existe donc un nombre premier p et une injection

$$\mathbb{F}_p \hookrightarrow K$$

qui munit K d'une structure de $\mathbb{F}_p$-espace vectoriel de dimension finie. Si l'on note d cette dimension, $\mathrm{card}(K) = p^d$. Le cardinal d'un corps fini est donc toujours une puissance d'un nombre premier.

Réciproquement, soit $q = p^d$ une puissance d'un nombre premier. Soit L un corps de décomposition sur $\mathbb{F}_p$ du polynôme

$$Q = X^q - X$$

On considère

$$K = \{x \in L \ : \ x^q - x = 0\}$$

Il est facile de voir que K est un sous-corps de L (cela vient du fait que l'application $\mathrm{Frob}_p \ : \ x \mapsto x^p$ est un morphisme de corps, tout comme l'application $\mathrm{Frob}_q = \mathrm{Frob}_p^d \ : \ x \mapsto x^q$). Comme $Q' = -1$, Q n'a que des racines simples donc $\mathrm{card}(K) = \deg Q = q$. On remarque aussi que $K = L$ puisque K contient toutes les racines de Q.

On sait que le corps de décomposition de Q sur $\mathbb{F}_p$ est unique à isomorphisme près. Donc il existe un et un seul corps de cardinal q à isomorphisme près. On le note $\mathbb{F}_q$.

Pour construire $\mathbb{F}_q$, il suffit de trouver un polynôme irréductible P de degré d $\mathbb{F}_p[X]$. Alors $\mathbb{F}_p[X]/(P)$ est un corps de cardinal $p^d = q$, donc isomorphe à $\mathbb{F}_q$.

Pour montrer que l'on peut toujours trouver un tel polynôme, on peut utiliser le résultat suivant.

Théorème 7.1.1. *Si K est un corps commutatif, tout sous groupe fini de $(K^*, \times)$ est cyclique.*

Soit x un générateur de $(\mathbb{F}_q^*, \times)$. Alors $\mathbb{F}_q = \mathbb{F}_p[x]$. Soit $m \in \mathbb{F}_p[X]$ le polynôme minimal de x sur $\mathbb{F}_p$.

$$\mathbb{F}_q \simeq \mathbb{F}_p[X]/(m)$$

$\mathbb{F}_q$ peut donc toujours s'écrire sous cette forme.

Pour terminer, rappelons aussi les relations d'inclusion entre les corps finis.

Soient $k, l \in \mathbb{N} \setminus \{0\}$. On considère les corps $\mathbb{F}_{q^k}$ et $\mathbb{F}_{q^l}$ inclus dans un corps K. Autrement dit, K est un corps qui contient un corps de décomposition de $(X^{q^k} - X)(X^{q^l} - X)$ sur $\mathbb{F}_q$ et

$$\mathbb{F}_{q^k} = \{x \in K \,:\, x^{q^k} = x\} \quad , \quad \mathbb{F}_{q^l} = \{x \in K \,:\, x^{q^l} = x\}$$

Proposition 7.1.2. *$\mathbb{F}_{q^k} \subset \mathbb{F}_{q^l}$ si et seulement si k divise l.*

Démonstration. Si $\mathbb{F}_{q^k} \subset \mathbb{F}_{q^l}$, alors $\mathbb{F}_{q^l}$ est un $\mathbb{F}_{q^k}$-espace vectoriel. Soit d sa dimension. $q^l = q^{kd}$ donc $l = kd$.

Réciproquement, supposons que $l = kd$. Soit $x \in \mathbb{F}_{q^k}$, $x^{q^k} = x$, donc

$$x^{q^l} = x^{q^{kd}} = \underbrace{(\dots((x^{q^k})^{q^k})\dots)^{q^k}}_{d \text{ exponentiations}} = x$$

ce qui prouve que $x \in \mathbb{F}_{q^l}$. □

Terminons ce paragraphe par un théorème qui nous sera utile.

Théorème 7.1.3. *Pour tout polynôme irréductible P de $\mathbb{F}_q[X]$, les racines de P dans un corps de décomposition donné de P sur $\mathbb{F}_q$ sont deux à deux distinctes. On dit que $\mathbb{F}_q$ est un corps parfait.*

Démonstration. Soient $n = \deg P$ et a une racine de P dans $\mathbb{F}_p^c$. Alors $\mathbb{F}_q[a] \simeq \mathbb{F}_{q^n}$ donc $a^{q^n} = a$. Comme P est le polynôme minimal de a, il divise $Q = x^{q^n} - x$. Ce polynôme Q n'a que des racines simples (car sa dérivée vaut -1), donc P n'a que des racines simples. □

7.2. Irréductibilité dans $\mathbb{F}_q[X]$

Soient p un nombre premier et $q = p^k$ une puissance de p. Pour tout entier $n \in \mathbb{N} \setminus \{0\}$, on note

$$\mathrm{Irr}(q, n) = \{P \in \mathbb{F}_q[X] \,:\, P \text{ est unitaire et irréductible de degré } n\}$$

Lemme 7.2.1. *Pour tout $n \in \mathbb{N} \setminus \{0\}$,*

$$X^{q^n} - X = \prod_{d|n} \prod_{P \in Irr(q,d)} P$$

Démonstration. Notons $G = X^{q^n} - X$ et $D = \prod_{d|n} \prod_{P \in Irr(q,d)} P$. Soit K un corps de décomposition de GD. Les deux polynômes G et D sont unitaires et leurs racines dans K sont deux à deux distinctes. Il suffit donc de montrer que G et D ont les mêmes racines.

Soit a une racine de G. Alors $a^{q^n} = a$, donc $a \in \mathbb{F}_{q^n}$. Soit P le polynôme minimal de a et soit $d = \deg P$. Alors $P \in \mathrm{Irr}(q, d)$. De plus

$$\mathbb{F}_{q^d} \simeq \mathbb{F}_q[X]/(P) \simeq \mathbb{F}_q[a] \subset \mathbb{F}_{q^n}$$

donc $\mathbb{F}_{q^d} \subset \mathbb{F}_{q^n}$. D'après la proposition 7.1.2, on en déduit que d divise n.

Réciproquement, si $D(a) = 0$, il existe un entier d divisant n et $P \in \mathrm{Irr}(q, d)$ tel que $P(a) = 0$.

$$\mathbb{F}[a] \simeq \mathbb{F}[X]/(P) \simeq \mathbb{F}_{q^d}$$

Or $\mathbb{F}_{q^d} \subset \mathbb{F}_{q^n}$ puisque d divise n (toujours par la proposition 7.1.2), donc $a \in \mathbb{F}_{q^n}$ et par conséquent $a^{q^n} = a$. □

Corollaire 7.2.2. *Soit $Q \in \mathbb{F}_q[X]$ de degré n. Alors Q est irréductible si et seulement si les deux conditions suivantes sont réalisées.*

1. *Q divise $X^{q^n} - X$*
2. *Q est premier à $X^{q^{n/l}} - X$ pour tout diviseur premier l de n.*

Démonstration. On suppose Q irréductible. D'après le lemme 7.2.1, Q est l'un des facteurs de $X^{q^n} - X$. Par contre, si l est un diviseur premier de n, n ne divise pas n/l donc Q ne divise pas $X^{q^{n/l}} - X$, toujours d'après le lemme 7.2.1.

Réciproquement, supposons que Q verifie les conditions 1 et 2. Soient P un facteur irréductible de Q et $d = \deg P$. Comme P divise $X^{q^n} - X$ et ne divise $X^{q^{n/l}} - X$ pour aucun diviseur premier l ne n, c'est que d divise n et ne divise n/l pour aucun diviseur premier l ne n (encore par le lemme 7.2.1). On en déduit que $n = d$, donc il existe $\alpha \in \mathbb{F}_q^*$ tel que $Q = \alpha P$. □

Algorithme 7.2.3. [IRRÉDUCTIBILITÉ]

Entrée : $Q \in \mathbb{F}_q[X]$ de degré n, $\mathcal{P}$: ensemble des diviseurs premiers de n.
Sortie : "Q est irréductible" ou "Q est réductible"

1. $h \leftarrow \mathrm{rem}(X^{q^n}, Q)$
2. *Si $h \neq X$, sortir "Q est réductible"*
3. *Pour tout $l \in \mathcal{P}$:*
4. $\quad h \leftarrow \mathrm{rem}(X^{q^{n/l}}, Q)$
5. $\quad d \leftarrow \mathrm{pgcd}(h - X, Q)$
6. $\quad$ *Si $d \neq 1$, sortir "Q est réductible et d est un facteur"*
7. *Sortir "Q est irréductible"*

Voyons la complexité de cet algorithme (si l'on utilise la multiplication classique). Le calcul de $\mathrm{rem}(X^{q^n}, Q)$ demande $O(\log q^n) = O(n \log q)$ multiplications dans $\mathbb{F}_q[X]/(Q)$ (à cette étape, on calcule X^{q^n} dans $\mathbb{F}_q[X]/(Q)$ par l'exponentiation binaire). La complexité algébrique est donc en $O(n^3 \log q)$ multiplications dans $\mathbb{F}_q$. Le calcul de $\mathrm{rem}(X^{q^{n/l}}, Q)$ se fait en $O(n^3 \log q)$, puis le pgcd du pas 5 en $O(n^2)$. Cela reste donc en $O(n^3 \log q)$. Comme n a $O(\log n)$ facteurs premiers, la complexité algébrique est en $O(n^3 \log n \log q)$.
Remarque. On peut aller jusque $\widetilde{O}(n^2 + n \log q)$ (voir [G-G]).

Intéressons nous maintenant au problème suivant.
Problème. Comment construire un polynôme irréductible de $\mathbb{F}_q[X]$ de degré donné n ? Cela peut servir, par exemple pour construire $\mathbb{F}_{q^n}$.

Solution. Tirer au hasard un polynôme unitaire de degré n, puis tester s'il est irréductible. Recommencer jusqu'à obtenir un polynôme irréductible.

La question qui suit immédiatement, c'est le nombre de tirages nécessaires en moyenne avant d'obtenir un polynôme irréductible. Le lemme 7.2.1 fournit une indication pour cela. En effet, en prenant les degrés dans l'égalité de ce lemme 7.2.1, on obtient

$$q^n = \sum_{d|n} d \cdot \mathrm{card}(\mathrm{Irr}(q, d)) \tag{7.1}$$

on en déduit l'inégalité suivante (en ne prenant que le terme correspondant à $d = n$ dans la somme de (7.1))

$$\mathrm{card}(\mathrm{Irr}(q, n)) \leq \frac{q^n}{n}$$

Comme il y a q^n polynômes unitaires de degré n dans $\mathbb{F}_q[x]$, la probabilité p_n d'obtenir un polynôme irréductible est telle que

$$p_n \leq \frac{1}{n}$$

On peut montrer que

$$p_n \sim \frac{1}{n}$$

Ainsi, si X est le nombre de tirages nécessaires, $E(X) \sim n$.

7.3. Algorithme de Cantor-Zassenhaus

On suppose q impair. Nous indiquons plus loin comment on peut traiter le cas où q est une puissance de 2.

L'algorithme qui suit s'applique à des polynômes très particuliers, à savoir les polynômes Q de $\mathbb{F}_q[X]$ **sans facteur carré dont les facteurs irréductibles sont tous de même degré** d. Autrement dit, il s'appliquera aux polynômes Q vérifiant la condition

$$Q = P_1 \dots P_r \quad \text{où} \quad \begin{cases} - \text{ les polynômes } P_i \text{ sont irréductibles} \\ \quad \text{deux à deux premiers entre eux} \\ - \ \forall i \in [[1, r]] \ , \ \deg P_i = d \end{cases} \tag{7.2}$$

où d est un entier naturel non nul donné en paramètre.

Algorithme 7.3.1. [Cantor-Zassenhaus]
Entrée : $d \in \mathbb{N} \setminus \{0\}$, un polynôme Q vérifiant (7.2)
Sortie : un facteur non trivial de Q ou "Échec" ou Q est irréductible

1. $n \leftarrow \deg Q$
2. *si $n = d$, sortir "Q est irréductible"*
3. $A \leftarrow$ *un polynôme choisi au hasard dans* $\mathbb{F}_q[X]_{n-1} \setminus \{0\}$
4. $D \leftarrow \text{pgcd}(A, Q)$
5. *Si $D \neq 1$, sortir D*
6. $B \leftarrow \text{rem}(A^{\frac{q^d-1}{2}}, Q)$
7. $D \leftarrow \text{pgcd}(B - 1, Q)$
8. *Si $D = 1$ ou Q, sortir "Échec"*
9. *Sinon, sortir D*

Théorème 7.3.2. *Si Q est réductible, l'algorithme de Cantor-Zassenhaus rend un facteur non trivial avec une probabilité supérieure ou égale à $1/2$. Ainsi, si X est le nombre de tirages nécessaires pour trouver un facteur, $E(X) \leq 2$.*

Démonstration. On utilise l'isomorphisme des restes chinois

$$\begin{array}{rccc} f \ : & \mathbb{F}_q[X]/(Q) & \simeq & \mathbb{F}_q[X]/(P_1) \times \cdots \times \mathbb{F}_q[X]/(P_r) \\ & [A]_Q & \mapsto & (\ [A]_{P_1} \ , \ \dots \ , \ [A]_{P_r} \ ) \end{array}$$

Si au pas 5, $\text{pgcd}(A, Q) \neq 1$, ce qui a très peu de chance d'arriver, l'algorithme donne un facteur non trivial. Sinon, A est premier à Q, donc

$$[A]_Q \in (\mathbb{F}_q[X]/(Q))^*$$

L'isomorphisme d'anneaux f induit un isomorphisme de groupes

$$(F_q[X]/(Q))^* \simeq (\mathbb{F}_q[X]/(P_1))^* \times \cdots \times (\mathbb{F}_q[X]/(P_r))^*$$

Comme pour tout i, le polynôme P_i est irréductible de degré d, $\mathbb{F}_q[X]/(P_i) \simeq \mathbb{F}_{q^d}$ donc

$$(\mathbb{F}_q[X]/(P_i))^* \simeq \mathbb{F}_{q^d}^*$$

Pour tout $x \in \mathbb{F}_{q^d}^*$, $x^{q^d-1} = 1$ donc $x^{\frac{q^d-1}{2}} = \pm 1$. Soit

$$\begin{aligned} g \ : \ \mathbb{F}_{q^d}^* &\to \{-1, 1\} \\ x \ &\mapsto \ x^{\frac{q^d-1}{2}} \end{aligned}$$

De plus, $\mathrm{card}(\ker g) = \frac{q^d-1}{2}$ (et bien sûr, $\ker g = g^{-1}(1)$). Ainsi,

$$\mathrm{card}(g^{-1}(1)) = \mathrm{card}(g^{-1}(-1)) = \frac{q^d - 1}{2}$$

Autrement dit, si on choisit x au hasard, on a autant de chance d'obtenir $g(x) = 1$ que $g(x) = -1$.

Considérons maintenant

$$\begin{aligned} h \ : \ (\mathbb{F}_q[X]/(Q))^* \to \quad & \{\pm 1\} \times \cdots \times \{\pm 1\} \\ [A]_Q \quad \mapsto & \left([A]_{P_1}^{\frac{q^d-1}{2}}, \ldots, [A]_{P_r}^{\frac{q^d-1}{2}} \right) = ([B]_{P_1}, \ldots, [B]_{P_r}) \end{aligned}$$

D'après ce qui précède, pour tout i, les événements

$$([B]_{P_i} = 1) \quad \text{et} \quad ([B]_{P_i} = -1)$$

sont équiprobables. Or

$$\mathrm{pgcd}(B - 1, Q) = \prod_{[B]_{P_i}=1} P_i$$

Donc $\mathrm{pgcd}(B-1, Q)$ est un facteur non trivial de Q, sauf dans les deux cas suivants.

— si $[B]_{P_i} = 1$ pour tout i, auquel cas $\mathrm{pgcd}(B-1, Q) = Q$

— si $[B]_{P_i} = -1$ pour tout i, auquel cas $\mathrm{pgcd}(B-1, Q) = 1$

Finalement, si A est premier à Q, la probabilité d'échec est

$$P = \frac{2}{\mathrm{card}(\{\pm 1\}^r)} = \frac{1}{2^{r-1}}$$

Si Q est réductible, $r \geq 2$ donc $P \leq \frac{1}{2}$. □

Exercice 7.3.3.
1. *Appliquer l'algorithme 7.3.1 à $X^2 - 1 \in \mathbb{F}_5[X]$, où le hasard aura désigné $A = X$.*
2. *Recommencer avec $A = X + 2$.*

Proposition 7.3.4. *la complexité algébrique de l'algorithme 7.3.1 est en $O(n^2 d \log q)$ si l'on utilise la multiplication classique. On peut aller jusque $\widetilde{O}(nd \log q)$ en utilisant la FFT.*

Démonstration. Exercice. □

7.4. Un algorithme de factorisation complète

On reprend les notations du paragraphe précédent et q reste impair.

7.4.1 Produits d'irréductibles de même degré

Tout d'abord, restons dans le cas où Q vérifie la condition (7.2) pour un entier naturel non nul d donné (c'est-à-dire : Q est sans facteur carré et est produit de polynômes irréductibles de degré d). On suppose aussi Q unitaire.

Appelons `CZ` l'algorithme 7.3.1. Pour factoriser Q complètement, on applique l'algorithme `DegresEgaux` suivant.

Cet algorithme prendra comme entrées $d \in \mathbb{N} \setminus \{0\}$ et le polynôme Q vérifiant (7.2) et il donnera en sortie la liste $[P_1, \ldots, P_r]$ des facteurs irréductibles unitaires de Q.

Si $\deg Q = d$, alors l'algorithme s'arrête et rend $l = [Q]$.

Sinon, il calcule `CZ`(d, Q) jusqu'à obtenir un facteur non trivial D de Q puis il s'appelle lui-même pour calculer les deux factorisations données par `DegresEgaux`(d, D) et `DegresEgaux`$(d, Q/D)$ pour ensuite concaténer les résultats.

7.4.2 Cas général

Soit maintenant Q quelconque dans $\mathbb{F}_q[X] \setminus \{0\}$. Alors Q peut s'écrire

$$Q = \lambda \prod_{i=1}^{r} P_i^{\alpha_i}$$

où les P_i sont irréductibles unitaires deux à deux premiers entre eux et où $\lambda \in \mathbb{F}_q^*$. Il s'agit de calculer les P_i et les α_i.

On commence par calculer

$$D_1 = \mathrm{pgcd}(X^q - X, Q)$$

D'après le lemme 7.2.1

$$D_1 = \prod_{P \in \mathcal{I}_1(Q)} P$$

où

$$\mathcal{I}_1(Q) = \{P \in I(q,1) \ : \ P|Q\}$$

D_1 est le produit des polynômes irréductibles unitaires de degré 1 qui divisent Q (D_1 est unitaire car on calcule le pgcd unitaire). Alors, la fonction `DegresEgaux`$(1, D_1)$ donne

$$\mathcal{I}_1(Q) = \{P_{1,1}, \ldots, P_{1,r_1}\}$$

On pose

$$Q_1 \leftarrow Q$$

et on effectue une double boucle

1. Pour $j \in [[1, r_1]]$:
2. $\quad \alpha_{i,j} \leftarrow 0$
3. $\quad$ Tant que $P_{1,j}$ divise Q_1 :
4. $\quad\quad Q_1 \leftarrow Q_1/P_{1,j}$
5. $\quad\quad \alpha_{i,j} \leftarrow \alpha_{i,j} + 1$

À la sortie, les facteurs irréductibles unitaires de degré 1 de Q sont les $P_{1,j}$ avec multiplicité $\alpha_{1,j}$ et Q_1 n'a pas de facteurs irréductibles de degré 1.

On calcule ensuite

$$D_2 \leftarrow \text{pgcd}(X^{q^2} - X, Q_1)$$

D'après le lemme 7.2.1 et comme les degrés des facteurs irréductibles de Q_1 sont supérieurs ou égaux à 2,

$$D_2 = \prod_{P \in \mathcal{I}_2(Q)} P$$

où

$$\mathcal{I}_2(Q) = \{P \in I(q,2) \ : \ P|Q\}$$

D_2 est le produit des polynômes irréductibles unitaires de degré 2 qui divisent Q.

On continue ainsi jusqu'à obtenir un polynôme Q_s de degré 0.

Proposition 7.4.1. *Cet algorithme permet de factoriser Q en $O(n^3 \log(q))$ en moyenne avec la multiplication classique. Avec la FFT, on peut aller jusque $\widetilde{O}(n^2 \log q)$.*

7.5. Algorithme de Berlekamp

On suppose toujours que q est impair.

L'algorithme de Berlekamp s'applique aux polynômes $Q \in \mathbb{F}_q[X]$ sans facteurs carrés. Ses facteurs irréductibles peuvent être de degrés quelconques.

Dans le cas général, il peut être utilisé comme une alternative à l'algorithme de Cantor-Zassenhaus dans la stratégie expliquée ci-dessus. Si $\deg Q = n$, cela donne un algorithme de factorisation complète de Q de complexité algébrique $O(n^3 + M(n)\log q)$ en moyenne (où $M(n)$ désigne la complexité de la multiplication de polynômes de degrés $< n$). On peut donc aller jusqu'à $\widetilde{O}(n^3 + n\log q)$.

Il est aussi possible de calculer la partie sans facteur carrée de Q au sens suivant. Soit $Q = \lambda \prod_{i=1}^{r} P_i^{\alpha_i}$ la décomposition de Q en produit de facteurs irréductibles unitaires. On appelle partie sans facteur carrée de Q le polynôme $\lambda \prod_{i=1}^{r} P_i$.

Nous supposons donc ici que Q est déjà sous cette forme.

$$Q = \lambda \prod_{i=1}^{r} P_i$$

où les P_i sont irréductibles unitaires deux à deux distincts. On utilise à nouveau l'isomorphisme des restes chinois

$$\begin{array}{rcl} f \ : \ \mathbb{F}_q[X]/(Q) \simeq \mathbb{F}_q[X]/(P_1) \times \cdots \times \mathbb{F}_q[X]/(P_r) \\ [A]_Q \quad \mapsto \quad (\ [A]_{P_1} \quad , \ \ldots \ , \quad [A]_{P_r} \ ) \end{array}$$

Pour tout i, on note $d_i = \deg P_i$. Alors $F_q[X]/(P_i) \simeq \mathbb{F}_{q^{d_i}}$. Par composition, on obtient donc un isomorphisme d'algèbres

$$g \ : \ \mathbb{F}_q[X]/(Q) \simeq \mathbb{F}_{q^{d_1}} \times \cdots \times \mathbb{F}_{q^{d_r}}$$

Or

$$\mathbb{F}_q^r = \mathbb{F}_q \times \cdots \times \mathbb{F}_q \subset \mathbb{F}_{q^{d_1}} \times \cdots \times \mathbb{F}_{q^{d_r}}$$

On note

$$\mathcal{A} = g^{-1}(\mathbb{F}_q^r)$$

C'est un sous-espace vectoriel de dimension r du $\mathbb{F}_q$-espace vectoriel $\mathbb{F}_q[X]/(Q)$.

On sait reconnaître les éléments de $\mathcal{A}$. En effet, soit $x \in \mathbb{F}_{q^{d_1}} \times \cdots \times \mathbb{F}_{q^{d_r}}$. Alors $x \in \mathbb{F}_q^r$ si et seulement si $x^q = x$. Comme g est un isomorphisme d'algèbres, pour tout $a \in \mathbb{F}_q[X]/(Q)$,

$$a \in \mathcal{A} = g^{-1}(\mathbb{F}_q^r) \iff a^q = a$$

Autrement dit , $\mathcal{A} = \ker(F - \mathrm{Id})$ où

$$\begin{array}{rccc} F & : & \mathbb{F}_q[X]/(Q) & \to & \mathbb{F}_q[X]/(Q) \\ & & a & \mapsto & a^q \end{array} \tag{7.3}$$

On peut donc calculer une base de $\mathcal{A}$: on écrit la matrice M de F dans la base $([1]_Q, [X]_Q, \ldots, [X^{n-1}]_Q)$ de $\mathbb{F}_q[X]/(Q)$ où n est le degré de Q, puis on calcule $\ker(M - I_n)$ par la méthode du pivot de Gauss.

On choisit alors $a \in \mathcal{A} \setminus \{0\}$ au hasard. Cet élément est la classe modulo Q d'un polynôme A de degré $< n$. Si $\mathrm{pgcd}(A, Q) \neq 1$, alors D est un facteur non trivial de Q. Si $\mathrm{pgcd}(A, Q) = 1$, on note $C = \mathrm{rem}(A^{\frac{q-1}{2}}, Q)$. Pour tout i, $[A]_{P_i}^{q-1} = 1$, donc $[C]_{P_i} = [A]_{P_i}^{\frac{q-1}{2}} \in \{\pm 1\}$, les événements

$$([C]_{P_i} = 1) \quad \text{et} \quad ([C]_{P_i} = -1)$$

étant équiprobables.

$$\mathrm{pgcd}(C - 1, Q) = \prod_{[C]_{P_i}=1} P_i$$

est un facteur non trivial de Q, sauf dans les cas suivants :

— si $[C]_{P_i} = 1$ pour tout i, auquel cas $\mathrm{pgcd}(C - 1, Q) = Q$

— si $[C]_{P_i} = -1$, auquel cas $\mathrm{pgcd}(C - 1, Q) = 1$

Finalement, si A est premier à Q, la probabilité d'échec est

$$P = \frac{2}{\mathrm{card}(\{\pm 1\}^r)} = \frac{1}{2^{r-1}}$$

Si Q est réductible, $r \geq 2$ donc $P \leq \dfrac{1}{2}$.

Pour automatiser tout cela, il vaut mieux calculer d'abord une base de $\mathcal{A}$.

Algorithme 7.5.1. NOYAU

Entrée : $Q \in \mathbb{F}_q$ sans facteurs carrés. de degré n

Sortie : Base du noyau de $F - Id$ (F étant définie en (7.3))

1. *Pour* $j \in [[0, n-1]]$ *:*
2. *Calculer* $m_{0,j}, \ldots, m_{n-1,j}$ *tels que*

$$X^{jq} \equiv m_{0,j} + m_{1,j}X + \cdots + m_{n-1,j}X^{n-1} \mod Q$$

3. $M \leftarrow (m_{i,j}) \in M_n(\mathbb{F}_q)$
4. $\mathcal{B} \leftarrow$ *base de* $\ker(M - I_n)$ *(calculée par le pivot de Gauss)*
5. *sortir* $\mathcal{B}$

Une fois le noyau calculé, on peut appliquer l'algorithme de Berlekamp suivant jusqu'à obtenir un facteur non trivial de Q.

Algorithme 7.5.2. [Berlekamp]
Entrées : Q, une base $\mathcal{B} = (b_0, \ldots, b_{r-1})$ *de* $\ker(F - Id)$ *calculée par* `Noyau`
Sortie : "Échec" ou un facteur non trivial de Q ou "Q est irréductible"

1. *si* $r = 1$ *: sortir "Q est irréductible"*
2. $v = (v_i) \leftarrow$ *un élément au hasard dans* $\mathbb{F}_q^r \setminus \{0\}$
3. $A \leftarrow \sum_{i=0}^{r-1} v_i B_i$ *(où* $B_i \in \mathbb{F}_q[X]_{n-1}$ *et* $[B_i]_Q = b_i$ *pour tout i)*
4. $D \leftarrow \mathrm{pgcd}(A, Q)$
5. *si* $D \neq 1$*, sortir* D
6. $C \leftarrow \mathrm{rem}(A^{\frac{q-1}{2}}, Q)$ *(à calculer par exponentiation rapide dans* $\mathbb{F}_q[X]/(Q)$*)*
7. $D \leftarrow \mathrm{pgcd}(C - 1, Q)$
8. *si* $D = 1$ *ou* $\deg D = n$ *: sortir "Échec"*
9. *sinon : sortir* D

7.6. Factorisation dans $\mathbb{Z}[X]$

Ce paragraphe décrit les idées qui permettent d'utiliser les résultats trécédent pour factoriser un polynôme de $\mathbb{Z}[X]$. Soit $Q \in \mathbb{Z}[X]$ un polynôme non nul. On note

$$Q = \sum_{i=0}^{n} a_i X^i$$

Soit $\mathrm{cont}(Q) = \mathrm{pgcd}(a_0, \ldots, a_n)$ le contenu de Q. Il existe $Q_1 \in \mathbb{Z}[X]$ de contenu 1 tel que $Q = \mathrm{cont}(Q)Q_1$.

On considère un polynôme Q de contenu 1. L'idée est d'exploiter la factorisation de $Q \mod p$, ou p est un nombre premier bien choisi.

Si

$$Q = \prod_{i=1}^{r} P_i^{e_i}$$

où les P_i sont des polynômes irréductibles deux à deux distincts, alors

$$\frac{Q}{\text{pgcd}(Q, Q')} = \prod_{i=1}^{r} P_i$$

On peut donc supposer que $Q = \prod_{i=1}^{r} P_i$. On note $[Q]_p$ la classe de Q modulo p.

$$\begin{aligned}[Q]_p &= [P_1]_p \dots [P_r]_p \\ &= R_1 \dots R_s\end{aligned}$$

où les R_i sont des polynômes irréductibles de $\mathbb{F}_p[X]$.

- On sait factoriser dans $\mathbb{F}_p[X]$. On sait donc calculer les R_i.
- Pour p assez grand, $[Q]_p$ est sans facteurs carrés, donc les R_i sont deux à deux distincts.
- Pour tout i, $[P_i]_p$ est le produit des certains R_j, mais on ne sait pas lesquels.
- Les bornes de Mignotte donne une majoration des coefficients des P_i en fonction de ceux de Q. Cela permet de trouver une borne B telle que si $p > B$, pour tout produit des R_j, il existe un unique relèvement dans $\mathbb{Z}[X]$ satisfaisant aux bornes de Mignotte.

On peut tester toutes les combinaisons possibles, mais la complexité est exponentielle.

Au lieu d'utiliser une factorisation modulo p où p est assez grand, on peut factoriser modulo p^k où p^k est assez grand.

Pour la phase de relèvement, on peut faire mieux en utilisant l'algorithme LLL (de A.-K. Lenstra, H.-W. Lenstra, L. Lovász, 1982) de recherche de petits vecteurs dans un réseau. Cela donne un algorithme de complexité polynomiale.

Chapitre 8

Polynômes multivariés

8.1. Position du problème

Soit K un corps. On considère l'anneau $R = K[X_1, \ldots, X_n]$.

On rappelle la définition d'un idéal.

Définition 8.1.1. *Soit $(A, +, \times)$ un anneau commutatif unitaire. Un idéal I de A est un sous-groupe de $(A, +)$ tel que pour tout $a \in I$ et $f \in A$, l'élément af appartient à I.*

Proposition 8.1.2. *Soient A un anneau et $\mathcal{P}$ une partie de A. L'ensemble*

$$< \mathcal{P} >= \left\{ \sum_{p \in J} q_p p \ : \ J \subset \mathcal{P} \text{ est fini et } q_p \in A \ \forall p \in J \right\}$$

est le plus petit idéal contenant $\mathcal{P}$. Cet idéal est appelé l'idéal engendré par $\mathcal{P}$.

Soient $f_1, \ldots, f_s$ des éléments de R. On va s'intéresser à l'idéal $I =< f_1, \ldots, f_s >$ engendré par $f_1, \ldots, f_s$ (donc $I = \{\sum_{i=1}^{s} q_i f_i \ : \ (q_1, \ldots, q_s) \in R^s\}$). Comment travailler avec un tel idéal ? Plus précisément, on peut se poser les questions suivantes.

— Soit f un élément de R. Comment savoir si f appartient ou non à I ?

— Soit J un autre idéal de R. Comment savoir si J est inclus dans I ?

— Comment trouver un système de représentants de R/I ?

— Soit

$$\begin{aligned} V(I) &= \{x = (x_1, \ldots, x_n) \in (K^c)^n \ f(x) = 0 \ \forall f \in I\} \\ &= \{x = (x_1, \ldots, x_n) \in (K^c)^n \ f_i(x) = 0 \ \forall i \in [[1, s]]\}. \end{aligned}$$

Comment savoir si $V(I)$ est vide ? Comment savoir s'il est fini ? Et dans ce cas, comment calculer ses éléments ?

Voyons ce qu'on sait faire dans le cas où $n = 1$. Dans ce cas, tout idéal I est principal. Donc I s'écrit $I = gR$ où $g \in R$ (c'est le pgcd des f_i). Pour savoir si f appartient à I, on effectue la division euclidienne $f = gq + r$. Alors $f \in I$ si et seulement si $r = 0$.

R/I est représenté par l'ensemble des polynômes de degré $< d$, où $d = \deg g$. C'est un K-espace vectoriel de base $([1]_g, [X]_g, \ldots, [X^{d-1}]_g)$.

Enfin, $V(I)$ est l'ensemble des racines de g dans K. La réponse est plus ou moins difficile suivant le corps de base K. Nous avons vu au chapitre précédent des algorithmes pour cela dans le cas où K est fini.

Pour répondre à ces question dans le cas où $n \geq 2$, nous allons définir une *division multivariée dans R*. Cela nous mènera à définir certains systèmes de générateurs de I pour lesquels cette division possèdera la propriété d'unicité du reste. Ces systèmes sont appelés *bases de Gröbner*.

8.2. Division multivariée avec reste

Un monôme est un élément de R de la forme

$$X^\alpha = X_1^{\alpha_1} \ldots X_n^{\alpha_n} \text{ où } \alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$$

Nous parlerons aussi de termes. Un terme est un élément de la forme λX^α où $\alpha \in \mathbb{N}^n$ et $\lambda \in K^*$.

Pour définir la division, nous aurons besoin d'une relation d'ordre total sur les monômes (donc sur $\mathbb{N}^n$), qui vérifieront certaines propriétés supplémentaires. Rappelons d'abord ce qu'est une relation d'ordre.

Définition 8.2.1. *Soit A un ensemble. Une relation $\preceq$ de A est une relation d'ordre si les conditions suivantes sont réalisées.*

1. *Pour tout x dans A, $x \preceq x$ ($\preceq$ est réflexive).*
2. *Si $x \preceq y$ et $y \preceq x$, alors $x = y$ ($\preceq$ est anti-symétrique).*
3. *Si $x \preceq y$ et $y \preceq z$, alors $x \preceq z$ ($\preceq$ est transitive).*

De plus, cette relation est dite d'ordre total si pour tout $(x, y) \in A^2$, soit $x \preceq y$ soit $y \preceq x$.

Les ordres que nous utiliseront sont appelés *ordres monomiaux*. Ils sont définis de la manière suivante.

Définition 8.2.2. *Un ordre monomial $\preceq$ sur $\mathbb{N}^n$ est une relation d'ordre total sur $\mathbb{N}^n$ qui vérifie les deux propriétés suivantes.*

1. *Si $\alpha \preceq \beta$, alors $\alpha + \gamma \preceq \beta + \gamma$.*
2. *Tout ensemble non vide de $\mathbb{N}^n$ admet un plus petit élément.*

On note alors $\alpha \prec \beta$ si $\alpha \preceq \beta$ et si $\alpha \neq \beta$. Par abus de langage, on parlera d'ordre monomial indifféremment pour $\prec$ ou pour $\preceq$. Enfin, on dit que $X^\alpha \preceq X^\beta$ (resp. $X^\alpha \prec X^\beta$) si $\alpha \preceq \beta$ (resp. $\alpha \prec \beta$).

Exemples.

1. L'ordre lexicographique. On le note $\prec_{lex}$. On dit que $\alpha \prec_{lex} \beta$ si $\alpha \neq \beta$ et si le premier coefficient non nul de $\alpha - \beta$ est strictement négatif. Ainsi, $(1,1,2,3) \prec_{lex} (1,3,2,1)$ et donc $X_1X_2X_3^2X_4^3 \prec_{lex} X_1X_2^3X_3^2X_4$.
2. L'ordre lexicographique gradué. On le note $\prec_{deglex}$.
   $\alpha \prec_{deglex} \beta$ si l'une des deux conditions suivantes est vérifiées.
   — $\sum \alpha_i < \sum \beta_i$
   — ou ($\sum \alpha_i = \sum \beta_i$ et $\alpha \prec_{lex} \beta$)
   Ainsi, $(1,1,2,3) \prec_{deglex} (1,3,2,1)$, et $(2,0,0,0) \prec_{deglex} (1,1,1,1)$ (alors que $(1,1,1,1) \prec_{lex} (2,0,0,0)$).
3. L'ordre lexicographique gradué inverse. On le note $\prec_{degrevlex}$.
   $\alpha \prec_{degrevlex} \beta$ si $\alpha \neq \beta$ et si l'une des deux conditions suivantes est vérifiées.
   — soit $\sum \alpha_i < \sum \beta_i$
   — soit $\sum \alpha_i = \sum \beta_i$ et le premier terme non nul de $\alpha - \beta$ en partant de la droite est > 0.
   Comme pour l'ordre lexicographique gradué,
   $$(1,1,2,3) \prec_{degrevlex} (1,3,2,1) \text{ et } (2,0,0,0) \prec_{degrevlex} (1,1,1,1)$$
   Par contre, $(2,1,3,4) \prec_{degrevlex} (1,2,4,3)$ alors que si l'on prend l'ordre lexicographique gradué, $(2,1,3,4) \succ_{deglex} (1,2,4,3)$.

Définition 8.2.3. *Soit $\prec$ un ordre monomial fixé. Soit $f \in R \setminus \{0\}$. Cet élément s'écrit*
$$f = \sum_{\alpha \in \mathbb{N}^n} c_\alpha X^\alpha$$
où $\{\alpha \ : \ c_\alpha \neq 0\}$ est fini. On utilisera le vocabulaire suivant.

— *Terme de f : tout $c_\alpha X^\alpha$ tel que $c_\alpha \neq 0$.*
— *$T(f)$ est l'ensemble des termes de f :*

$$T(f) = \{c_\alpha X^\alpha \ : \ c_\alpha \neq 0\}$$

— *Multidegré : $mdeg(f) = \max\{\alpha \in \mathbb{N}^n \ : \ c_\alpha \neq 0\}$.*
— *Coefficient dominant : $lc(f) = c_{mdeg(f)}$.*
— *Monôme dominant : $lm(f) = X^{mdeg(f)}$.*
— *Terme dominant : $lt(f) = c_{mdeg(f)} X^{mdeg(f)} = lc(f)lm(f)$.*

On note $lt(0) = lm(0) = lc(0) = 0$ et $mdeg(0) = -\infty$.

Venons en maintenant à la division multivariée. On veut diviser f par $f_1, \ldots, f_s$. Au debut, le reste r est égal à 0. On pose aussi $p = f$. Si $\mathrm{lt}(p)$ est divisible par l'un des $\mathrm{lt}(f_i)$, on effectue la division. Sinon, on remplace p par $p - \mathrm{lt}(p)$ et r par $r + \mathrm{lt}(p)$. On continue ainsi jusqu'à arriver à $p = 0$.

Mais voyons plutôt un exemple.

Exemple. Dans $K[x, y]$, divisons $f = x^2y + xy^2 + y^2$ par $f_1 = xy - 1$ et $f_2 = y^2 - 1$, en utilisant l'ordre lexicographique $\preceq$ tel que $x \succeq y$. Partons de $p \leftarrow f$, $r \leftarrow 0$, $q_1 \leftarrow 0$ et $q_1 \leftarrow 0$. Alors $f = p + q_1 f_1 + q_2 f_2 + r$, égalité qui restera vraie tout au long de la division. On voit que $(2, 1) \succeq (1, 2) \succeq (0, 2)$, donc $\mathrm{lt}(p) = x^2y$. Ce terme est divisible par $\mathrm{lt}(f_1) = xy$. On fait donc la division.

$$p \leftarrow p - xf_1 = xy^2 + x + y^2$$
$$q_1 \leftarrow q_1 + x = x$$

Après ces opérations, $\mathrm{lt}(p) = xy^2$ est encore divisible par $\mathrm{lt}(f_1)$.

$$p \leftarrow p - yf_1 = x + y^2 + y$$
$$q_1 \leftarrow q_1 + y = x + y$$

Maintenant, $\mathrm{lt}(p) = x$ n'est plus divisible ni par $\mathrm{lt}(f_1)$ ni par $\mathrm{lt}(f_2)$. On fait alors

$$r \leftarrow r + x = x$$
$$p \leftarrow p - x = y^2 + y$$

Après cela, $\mathrm{lt}(p) = y^2$ est divisible par $\mathrm{lt}(f_2) = y^2$.

$$p \leftarrow p - f_2 = x + y + 1$$
$$q_2 \leftarrow q_2 + 1 = 1$$

À présent, $\mathrm{lt}(p) = x$ n'est plus divisible ni par $\mathrm{lt}(f_1)$ ni par $\mathrm{lt}(f_2)$.

$$\begin{aligned} r &\leftarrow r + x = x \\ p &\leftarrow p - x = y + 1 \end{aligned}$$

Maintenant encore, $\mathrm{lt}(p) = y$ n'est divisible ni par $\mathrm{lt}(f_1)$ ni par $\mathrm{lt}(f_2)$.

$$\begin{aligned} r &\leftarrow r + y = x + y \\ p &\leftarrow p - y = 1 \end{aligned}$$

Et enfin

$$\begin{aligned} r &\leftarrow r + 1 = x + y + 1 \\ p &\leftarrow p - 1 = 0 \end{aligned}$$

On obtient finalement

$$\begin{aligned} f &= q_1 f_1 + q_2 f_2 + r \\ &= (x + y) f_1 + f_2 + x + y + 1 \end{aligned}$$

On peut présenter ces calculs dans un tableau.

$x^2y + xy^2 + y^2$	$xy - 1$	$y^2 - 1$
$xy^2 + x + y^2$	x	
$x + y^2 + y$	y	
$x + y + 1$		1

Remarquons que nous aurions pu faire ces opérations dans en ordre différent, comme par exemple celui décrit dans le tableau suivant.

$x^2y + xy^2 + y^2$	$xy - 1$	$y^2 - 1$
$xy^2 + x + y^2$	x	
$2x + y^2$		x
$2x + 1$		1

On obtient alors

$$f = x f_1 + (x + 1) f_2 + 2x + 1.$$

Il n'y a donc pas unicité dans l'écriture

$$f = q_1 f_1 + q_2 f_2 + r.$$

Le plus ennuyeux, c'est qu'il n'y a pas unicité du reste r. Ainsi, si la question est l'appartenance ou non de f à un idéal I, et si cette division donne $r = 0$, on peut répondre « oui, f appartient à I ». Par contre, si le reste est non nul, on ne peux rien dire à priori.

Il existe cependant des systèmes de générateurs pour lesquels il y a unicité du reste dans cette division multivariée. C'est l'objet de la section suivante.

Avant d'aborder cette prochaine section, on termine celle ci en écrivant l'algorithme de la division multivariée.

Algorithme 8.2.4. [DIVISION MULTIVARIÉE AVEC RESTE]

Entrées : $f, f_1, \ldots, f_s \in K[X_1, \ldots X_n]$ *et un ordre monomial* $\preceq$

Sorties : $q_1, \ldots, q_s, r \in K[X_1, \ldots X_n]$ *tels que* $f = \sum q_i f_i + r$ *et tels qu'aucun terme de* r *n'est divisible par aucun* $lt(f_i)$

1. $r \leftarrow 0$
2. $q_i \leftarrow 0 \ \forall i \in [[1, n]]$
3. $p \leftarrow f$
4. *Tant que* $p \neq 0$ *faire*
5. *Si* $lt(p)$ *est divisible par l'un des* $lt(f_i)$ *:*
6. $m \leftarrow \frac{lt(p)}{lt(f_i)}$
7. $p \leftarrow p - m f_i$
8. $q_i \leftarrow q_i + m$
9. *Sinon :*
10. $r \leftarrow r + lt(p)$
11. $p \leftarrow p - lt(p)$
12. *Sortir* $q_1, \ldots, q_s, r$

La preuve de la proposition suivante est laissée en exercice. Mis à part le point 3, elle permet de montrer l'algorithme. Le point 3 montre que dans la somme $\sum_{i=1}^{s} q_i f_i$, il n'y a pas d'annulation de termes de multidegré strictement supérieur à $\mathrm{mdeg}(f)$. Nous verrons par la suite l'importance de ce fait.

Proposition 8.2.5. *La suite des* $mdeg(p)$ *est strictement décroissante. De plus, après chaque exécution de la boucle « Tant que » de cet algorithme, les égalités suivantes sont vérifiées.*

1. $f = p + \sum_{i=1}^{s} q_i f_i + r$
2. *Pour tout* i*, aucun terme de* r *n'est divisible par* $lt(f_i)$
3. *Si* $q_i \neq 0$*, alors* $mdeg(q_i f_i) \preceq mdeg(f)$ *pour* $1 \leq i \leq s$

Notation. Si r est le reste obtenu par la division de f par un ensemble $F = \{f_1, \ldots, f_s\}$ de polynômes, on note (même si r n'est pas uniquement défini en général)

$$f \xrightarrow{F} r$$

8.3. Bases de Gröbner

Si $\mathcal{P}$ est une partie de R, on note

$$< \mathrm{lt}(\mathcal{P}) >=< \mathrm{lt}(f) \ : \ f \in \mathcal{P} >$$

c'est-à-dire l'idéal engendré par les $\mathrm{lt}(f)$, où f parcourt $\mathcal{P}$.

Soit I un idéal de R. Il est clair que si

$$I =< f_1, \ldots, f_s >$$

alors

$$< \mathrm{lt}(f_1) \ldots, \mathrm{lt}(f_s) > \subset < \mathrm{lt}(I) > .$$

Mais en général, ces deux idéaux ne sont pas forcément égaux : il arrive fréquemment que

$$< \mathrm{lt}(f_1) \ldots, \mathrm{lt}(f_s) > \neq < \mathrm{lt}(I) > .$$

Exemple. Dans $R = \mathbb{Q}[x, y]$, nous utilisons l'ordre lexicographique gradué $\prec_{deglex}$ tel que $x \succ_{deglex} y$. Soient $f_1 = x^3 - 2xy$ et $f_2 = x^2y - 2y^2 + x$. Alors $\mathrm{lt}(f_1) = x^3$ et $\mathrm{lt}(f_2) = x^2y$.

$$< \mathrm{lt}(f_1), \mathrm{lt}(f_2) >=< x^3, x^2y > .$$

Or, $-yf_1 + xf_2 = x^2 \in I$, donc $x^2 \in < \mathrm{lt}(I) >$, mais $x^2 \notin < \mathrm{lt}(f_1), \mathrm{lt}(f_2) >$.

Dans cet exemple, on constate qu'un terme x^2 n'appartient pas à un idéal engendré par des monômes $< x^3, x^2y >$ parce-que si tel était le cas, x^2 serait divisible par x^3 ou x^2y.

C'est un cas particulier du lemme élémentaire suivant dont la preuve est laissée en exercice. Nous utiliserons fréquemment ce résultat.

Lemme 8.3.1. *Soient $\alpha, \alpha_1, \ldots, \alpha_s$ des éléments de $\mathbb{N}^n$. le monôme X^α appartient à $< X^{\alpha_1}, \ldots, X^{\alpha_s} >$ si et seulement s'il existe $i \in [[1, s]]$ tel que X^{α_i} divise X^α.*

Soient $f_1, \ldots, f_s$ des éléments de R. Soit t un terme de R. Alors $t \in < lt(f_1), \ldots, lt(f_s) >$ si et seulement s'il existe $i \in [[1, s]]$ tel que $lt(f_i)$ divise t.

Définition 8.3.2. *Soit G une partie finie de I. On dit que G est une base de Gröbner de I si $< lt(I) >=< lt(G) >$.*

Remarque. Si pour toute partie $\mathcal{P}$ de R, on note

$$< \mathrm{lm}(\mathcal{P}) >=< \mathrm{lm}(f) \ : \ f \in \mathcal{P} >$$

Une partie finie G de I est une base de Gröbner de I si et seulement si $< \mathrm{lm}(I) >=< \mathrm{lm}(G) >$.

Remarque. Il n'est pas imposé dans la définition 8.3.2 que G engendre I. Ce n'est pas nécessaire : les choses se passent ici joliment, comme l'indique le résultat suivant.

Théorème 8.3.3. *Si G est une base de Gröbner de I, alors $I =< G >$.*

Preuve du théorème 8.3.3. Par l'absurde, on suppose qu'il existe un élément f dans $I \setminus < G >$, et on choisit cet élément de telle sorte que $\mathrm{mdeg}(f)$ soit minimal. Comme $\mathrm{lt}(f) \in < \mathrm{lt}(G) >$, et en vertu du lemme 8.3.1, on en déduit qu'il existe $g \in G$ tel que $\mathrm{lt}(g)$ divise $\mathrm{lt}(f)$. Soit alors

$$f_1 = f - \frac{\mathrm{lt}(f)}{\mathrm{lt}(g)} g.$$

Alors $f_1 \in I$ et $\mathrm{mdeg}(f_1) \prec \mathrm{mdeg}(f)$. Par minimalité de $\mathrm{mdeg}(f)$, on conclut que $f_1 \in < G >$, donc que $f \in < G >$, ce qui est contraire à l'hypothèse. □

Venons en au résultat que nous souhaitions, c'est-à-dire l'unicité du reste dans la division multivariée.

Proposition 8.3.4. *Soit $G = \{f_1, \ldots, f_s\}$ une base de Gröbner. S'il existe $q_1, \ldots, q_s, q'_1 \ldots, q'_s, r, r' \in R$ tels que*

1. $\sum q_i f_i + r = \sum q'_i f_i + r'$
2. $\forall i \in [[1, s]] \ , \forall t \in T(r) \cup T(r') \ , \ lt(f_i) \not| t$

(c'est-à-dire que pour tout i, aucun des termes de r ni de r' n'est divisible par $lt(f_i)$). Alors $r = r'$.

Preuve. Si $\sum q_i f_i + r = \sum q'_i f_i + r'$, alors $r - r' \in I =< G >$. Si $r - r' \neq 0$, on considère $\mathrm{lt}(r - r')$. Cet élément appartient à $< \mathrm{lt}(I) >=< \mathrm{lt}(G) >$. Le lemme 8.3.1 permet de déduire qu'il existe i tel que $\mathrm{lt}(f_i)$ divise $\mathrm{lt}(r - r')$, et donc que $\mathrm{lt}(f_i)$ divise l'un des termes de r ou de r', ce qui contredit l'hypothèse. □

On déduit facilement le résultat suivant.

Corollaire 8.3.5. *Si G est une base de Gröbner de I, alors pour tout élément f de R, $f \in I$ si et seulement si le reste de la division multivariée de f par G est nul.*

Ainsi, si l'on connaît une base de Gröbner de I, il est plus facile de travailler sur cet idéal. Reste à savoir si tout idéal I possède une base de Gröbner, et aussi comment calculer une telle base. Pour répondre à ces questions, nous utiliserons une nouvelle relation d'ordre sur $\mathbb{N}^n$.

Définition 8.3.6. *Soient $\alpha = (\alpha_1, \ldots, \alpha_n)$ et $\beta = (\beta_1, \ldots, \beta_n)$ deux éléments de $\mathbb{N}^n$. On écrit :*

$$\alpha \leq \beta \text{ si } \forall i,\ \alpha_i \leq \beta_i.$$

$$\alpha < \beta \text{ si } \alpha \leq \beta \text{ et } \alpha \neq \beta.$$

La relation $\leq$ est une relation d'ordre. Ce n'est pas un ordre total si $n > 1$ Par exemple, dans $\mathbb{N}^2$, les couples $(1, 0)$ et $(0, 1)$ ne sont pas comparables. Il ne s'agit donc pas d'un ordre monomial.

Définition 8.3.7. *Soit A une partie de $\mathbb{N}^n$. Soit $\alpha \in A$ On dit que α est minimal dans A si*

$$(\beta \in A \text{ et } \beta \leq \alpha) \Rightarrow \beta = \alpha$$

Lemme 8.3.8. *De toute suite de $\mathbb{N}^n$ on peut extraire une sous-suite croissante.*

Démonstration. Montrons ce lemme par récurrence sur n.

Cas où $n = 1$: soit $(u_k)_{k\in\mathbb{N}}$ une suite de $\mathbb{N}$. Soit $m_0 = \min\{u_k \ :\ k \in \mathbb{N}\}$. Cet élément existe puisque toute partie de $\mathbb{N}$ a un plus petit élément. Il existe au moins un entier k tel que $u_k = m_0$. Soit $\varphi(0)$ un tel entier. On suppose avoir défini

$$\varphi(0) < \varphi(1) < \cdots < \varphi(i-1)$$

tels que

$$u_{\varphi(0)} \leq u_{\varphi(1)} \leq \cdots \leq u_{\varphi(i-1)}$$

Soient $m_i = \min\{u_k \ :\ k \in \mathbb{N}\ ,\ k > \varphi(i-1)\}$ et $\varphi(i) > \varphi(i-1)$ tel que $u_{\varphi(i)} = m_i$. On construit ainsi par récurrence une fonction φ strictement croissante de $\mathbb{N}$ dans $\mathbb{N}$ telle que la suite $(u_{\varphi(k)})$ est croissante.

On suppose maintenant le résultat vrai pour $A = \mathbb{N}^{n-1}$. Soit (u_k) une suite de $\mathbb{N}^n = A \times \mathbb{N}$. on note $u_k = (a_k, v_k)$ où $a_k \in A$ et $v_k \in \mathbb{N}$. D'après l'hypothèse de récurrence, il existe une fonction strictement croissante φ de $\mathbb{N}$ dans $\mathbb{N}$ telle que $(a_{\varphi(k)})$ est croissante. On considère la suite $(v_{\varphi(k)})$. Il existe une fonction strictement croissante ψ de $\mathbb{N}$ dans $\mathbb{N}$ telle que $(v_{\varphi(\psi(k))})$ est croissante. Alors la suite $(u_{\varphi(\psi(k))})$ est croissante. □

Lemme 8.3.9. [DICKSON] *Pour toute partie A de $\mathbb{N}^n$ l'ensemble des éléments minimaux de A est fini.*

Démonstration. Soit B l'ensemble des éléments minimaux de A. Si B est infini, il existe une suite (u_k) d'éléments de B deux à deux distincts. On peut donc en extraire une sous-suite croissante. C'est absurde puisque les éléments de B sont minimaux. □

Exemple. Soit $A = \{\alpha \in \mathbb{N}^2 \ : \ \alpha_1 + \alpha_2 \geq 2\}$, alors $B = \{(2,0),(1,1),(0,2)\}$ est bien fini.

Théorème 8.3.10. *Tout idéal de R possède une base de Gröbner.*

Démonstration. Pour toute partie $\mathcal{P}$ de $\mathbb{N}^n$, on note

$$X^{\mathcal{P}} = \{X^\alpha \ : \ \alpha \in \mathcal{P}\}.$$

Soit I un idéal de R. Soit $A = \{\alpha \in \mathbb{N}^n \ : \ X^\alpha \in \mathrm{lm}(I)\}$ Alors

$$< \mathrm{lt}(I) >=< \mathrm{lm}(I) >=< X^A >$$

Soit B l'ensemble des éléments minimaux de A, Montrons que $< X^A >=< X^B >$. Comme $B \subset A$, $< X^B > \subset < X^A >$. Soit $\alpha \in A$. Il existe $\beta \in B$ tel que $\beta \leq \alpha$ (si tel n'était pas le cas, on pourrait construire une suite strictement décroissante d'éléments de $\mathbb{N}^n$ inférieurs à α, ce qui est absurde puisque l'ensemble des éléments de $\mathbb{N}^n$ inférieurs à α est fini). Alors $X^\alpha = X^\beta X^{\alpha-\beta} \in < X^B >$.

Pour toute partie finie G de I telle que $\mathrm{lm}(G) = X^B$, G est une base de Gröbner de I. □

Ce résultat a pour conséquence immédiate le célèbre théorème de la base de Hilbert.

Corollaire 8.3.11. [Théorème de la base de Hilbert] *Tout idéal de R est de type fini. Autrement dit, R est Noethérien.*

8.4. Algorithme de Buchberger

Maintenant, étant donné un idéal I de R, nous voudrions pouvoir calculer une base de Gröbner de I. L'algorithme de Buchberger résout ce problème. Cet algorithme est basé sur le calcul de *S-polynômes* que nous définissons ici.

Définition 8.4.1. *Soient f et g deux éléments de R. On note $\alpha = (\alpha_1, \ldots, \alpha_n) = mdeg(f)$ et $\beta = (\beta_1, \ldots, \beta_n) = mdeg(g)$. Soit*

$$\gamma = (\max(\alpha_i, \beta_i))_{i \in [[1,n]]} \in \mathbb{N}^n$$

(en d'autres termes, γ est tel que $X^\gamma = ppcm(lm(f), lm(g))$). On appelle S-polynôme associé à f et g le polynôme

$$S(f,g) = \frac{X^\gamma}{lt(f)} f - \frac{X^\gamma}{lt(g)} g.$$

Exemple. Dans $\mathbb{Q}[x,y]$, on utilise l'ordre monomial $\prec_{\text{lex}}$ tel que $y \prec_{\text{lex}} x$. Soient $f = 2x^2y - 3xy$ et $g = x^3 + y^5$. Alors $\text{lt}(f) = 2x^2y$ et $\text{lt}(g) = x^3$. Ainsi, $\alpha = (2,1)$, $\beta = (3,0)$) et $\gamma = (3,1)$. Enfin :

$$\begin{aligned} S(f,g) &= \frac{x^3y}{2x^2y}(2x^2y - 3xy) - \frac{x^3y}{x^3}(x^3 + y^5) \\ &= -\frac{3}{2}x^2y - y^6. \end{aligned}$$

Dans la différence définissant $S(f,g)$, les termes dominants s'annulent. C'est ce qu'indique la proposition suivante.

Proposition 8.4.2. *Avec les notations de la définition 8.4.1,*

$$mdegS(f,g) \prec \gamma.$$

Le lemme ci-dessous exprime le fait que si dans une somme on constate l'élimination des termes de plus haut degré, cela "provient" de S-polynômes.

Lemme 8.4.3. *Soient $g_1, \ldots, g_s \in R$, $\alpha_1, \ldots, \alpha_s \in \mathbb{N}^n$, $c_1, \ldots, c_s \in K^*$,*

$$f = \sum_{i=1}^{s} c_i X^{\alpha_i} g_i \in R$$

et $\delta \in \mathbb{N}^n$ tel que $\alpha_i + mdeg(g_i) = \delta$ pour tout $i \in [[1,s]]$ et tel que $mdeg(f) \prec \delta$ (c'est-à-dire que les termes dominants s'éliminent). Pour $i < j$, on définit $\gamma_{ij} \in \mathbb{N}^n$ tel que

$$X^{\gamma_{ij}} = ppcm(lm(g_i), lm(g_j))$$

(c'est le γ de la définition 8.4.1 correspondant à $S(g_i, g_j)$). Alors les propriétés suivantes sont vérifiées.

1. *$X^{\gamma_{ij}}$ divise X^δ pour tout (i,j) tel que $1 \leq i < j \leq s$.*
2. *$mdeg(X^{\delta - \gamma_{ij}} S(g_i, g_j)) \prec \delta$ pour tout (i,j) tel que $1 \leq i < j \leq s$.*
3. *Il existe des éléments c_{ij} dans K tels que*

$$f = \sum_{1 \leq i < j \leq s} c_{ij} X^{\delta - \gamma_{ij}} S(g_i, g_j).$$

Démonstration. Le 1 vient du fait que $\mathrm{lm}(g_i)$ divise X^δ pour tout i et que $X^{\gamma_{ij}} = \mathrm{ppcm}(\mathrm{lm}(g_i), \mathrm{lm}(g_j))$.

Le 2 provient de la proposition 8.4.2.

Pour le 3, on procède par récurrence comme suit. En multipliant chaque c_i par $\mathrm{lc}(g_i)$ et en divisant chaque g_i par ce même coefficient $\mathrm{lc}(g_i)$, on se ramène au cas où $\mathrm{lc}(g_i) = 1$ pour tout i.

Si $s = 1$, il ne peut y avoir simplification dans la somme donc le lemme est vide. Il n'y a rien à démontrer.

Si $s > 1$, on définit

$$g = f - c_1 X^{\delta - \gamma_{12}} S(g_1, g_2)$$

Alors

$$\mathrm{mdeg}(g) \preceq \max\left(\mathrm{mdeg}(f), \mathrm{mdeg}(X^{\delta-\gamma_{12}} S(g_1, g_2))\right) \prec \delta$$

puis on calcule

$$\begin{aligned}
g &= f - c_1 X^{\delta-\gamma_{1,2}} S(g_1, g_2) \\
&= c_1 X^{\alpha_1} g_1 + c_2 X^{\alpha_2} g_2 + \sum_{i\geq 3} c_i X^{\alpha_i} g_i - c_1 X^\delta \left(\frac{g_1}{\mathrm{lt}(g_1)} - \frac{g_2}{\mathrm{lt}(g_2)} \right) \\
&= c_1 X^{\alpha_1} g_1 + c_2 X^{\alpha_2} g_2 + \sum_{i\geq 3} c_i X^{\alpha_i} g_i - c_1 g_1 X^{\delta - \mathrm{mdeg}(g_1)} + c_1 g_2 X^{\delta - \mathrm{mdeg}(g_2)} \\
&= (c_1 + c_2) X^{\alpha_2} g_2 + \sum_{i=3}^{s} c_i X^{\alpha_i} g_i
\end{aligned}$$

en regroupant les termes, et en tenant compte du fait que pour tout i,

$$\alpha_i + \mathrm{mdeg}(g_i) = \delta$$

Le polynôme g est soit nul (si $s = 2$), soit de la même forme que f, mais avec $s-1$ ou $s-2$ termes. On peut donc appliquer l'hypothèse de récurrence. □

Théorème 8.4.4. *Soit $G = \{g_1, \ldots, g_s\} \subset R$. Alors G est une base de Gröbner de I si et seulement si les deux conditions suivantes sont vérifiées.*

1. *$I =< g_1, \ldots, g_s >$.*
2. *Pour tout $(i, j) \in [[1, s]]^2$ tel que $1 \leq i < j \leq s$, le reste de la division de $S(g_i, g_j)$ par G est égale à 0.*

Démonstration. Le sens direct est clair. Réciproquement, supposons les conditions (1) et (2) vérifiées. Montrons que $< \mathrm{lt}(I) >=< \mathrm{lt}(G) >$, c'est-à-dire

que $< \mathrm{lt}(I) > \subset < \mathrm{lt}(G) >$, l'autre inclusion étant évidente. Soit donc $f \in I$. Comme G engendre I, ce polynôme f s'écrit

$$f = \sum_{g \in G} q_g g \tag{8.1}$$

où $q_g \in R$ pour tout $g \in G$. On veut montrer que $\mathrm{lt}(f) \in < \mathrm{lt}(G) >$. On note $\alpha = \mathrm{mdeg}(f)$. Si $\mathrm{mdeg}(q_g g) \preceq \alpha$ pour tout $g \in G$, alors il n'y a pas de simplification des termes de plus haut degré des $q_g g$ dans l'expression de f donnée par l'égalité (8.1). Ainsi :

$$\begin{aligned} \mathrm{lt}(f) &= \sum_{g \,:\, \mathrm{mdeg}(q_g g) = \alpha} \mathrm{lt}(q_g g) \\ &= \sum_{\mathrm{g} \,:\, \mathrm{mdeg}(q_g g) = \alpha} \mathrm{lt}(q_g)\mathrm{lt}(g) \in < \mathrm{lt}(G) > \end{aligned}$$

Cela règle la question dans ce cas là. Mais il se peut qu'il y ait des simplifications des termes de plus haut degré. Alors

$$\alpha \prec \max_{g \in G} \mathrm{mdeg}(\mathrm{lt}(q_g g)).$$

Soit β minimal tel qu'il existe une écriture $f = \sum_{g \in G} q_g g$ de f telle que

$$\beta = \max_{g \in G} \mathrm{mdeg}(\mathrm{lt}(q_g g))$$

On suppose par l'absurde que $\beta \succ \alpha$. Posons

$$f^* = \sum_{\mathrm{g} \,:\, \mathrm{mdeg}(q_g g) = \beta} \mathrm{lt}(q_g) g.$$

Alors le lemme 8.4.3 montre l'existence de $\lambda_{h,g} \in K$ et $\alpha_{h,g} \in \mathbb{N}^n$ (pour $g, h \in G$) tels que

$$f^* = \sum_{g,h} \lambda_{g,h} X^{\alpha_{g,h}} S(g,h)$$

de telle sorte que pour tout $(g,h) \in G^2$, $\mathrm{mdeg}(X^{\alpha_{g,h}} S(g,h)) \prec \beta$. Si l'on fait la division de f^* par G (en considérant les S-polynômes les uns après les autres), on obtient un reste nul, donc

$$f^* = \sum_{g \in G} q_g^* g$$

où d'après le point (3) de la proposition 8.2.5

$$\mathrm{mdeg}(q_g^* g) \preceq \mathrm{mdeg}(f^*) \prec \beta$$

pour tout g. Finalement, $f = (f - f^*) + f^*$ où $f - f^*$ et f^* s'écrivent tous deux sous la forme d'une somme $\sum_{g \in G} p_g g$ où les p_g sont des éléments de R tels que $\mathrm{lt}(p_g g) \prec \beta$ pour tout $g \in G$. □

L'algorithme de Buchberger prend en entrée une famille $F = (f_1, \ldots, f_s)$ d'éléments de R et rend en sortie une base de Gröbner de l'idéal $I =< f_1, \ldots, f_s >$. L'idée est de calculer le reste r de chaque $S(f_i, f_j)$ divisé par F, et d'ajouter ce reste à la famille s'il est non nul.

Algorithme 8.4.5. [Algorithme de Buchberger]
Entrées : $f_1, \ldots, f_s \in K[X_1, \ldots X_n]$ et un ordre monomial $\prec$
Sortie : Une base de Gröbner G pour $\prec$ de $I =< f_1, \ldots, f_s >$

1. $G \leftarrow \{f_1, \ldots, f_s\}$
2. $\mathcal{S} \leftarrow G$
3. Tant que $\mathcal{S} \neq \emptyset$ *:*
4. $\quad \mathcal{S} \leftarrow \emptyset$
5. $\quad \{g_1, \ldots, g_t\} \leftarrow G$ *(on numérote les éléments de G de 1 à t)*
6. $\quad$ *Pour i de 1 à $t-1$:*
7. $\quad\quad$ *Pour j de $i+1$ à t :*
8. $\quad\quad\quad r \leftarrow$ *reste de la division de $S(g_i, g_j)$ par G*
9. $\quad\quad\quad$ *Si $r \neq 0$:*
10. $\quad\quad\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{r\}$
11. $\quad G \leftarrow G \cup \mathcal{S}$
12. Sortir G

Démonstration. L'algorithme se termine dès que $\mathcal{S} = \emptyset$. Alors le théorème 8.4.4 montre que l'ensemble G obtenu est une base de Gröbner.

Montrons que l'algorithme se termine. Soient $G_1, \ldots, G_k, \ldots$ la suite des ensembles G successifs de l'algorithme. Alors $G_i \subset G_{i+1}$ pour tout i donc $< \mathrm{lt}(G_i) > \subset < \mathrm{lt}(G_{i+1}) >$ pour tout i. Comme l'anneau R est noethérien, il existe i tel que $< \mathrm{lt}(G_i) > = < \mathrm{lt}(G_{i+1}) >$. Montrons qu'alors $G_i = G_{i+1}$: cela prouvera qu'à cette étape, $\mathcal{S} = \emptyset$ et donc que l'algorithme se termine. Pour plus de commodité, posons $G = G_i$ et $G' = G_{i+1}$. Alors $G' = G \cup \mathcal{S}$. On pose aussi

$$G = \{g_i \ : \ i \in \{1, \ldots, t\}\}.$$

Pour tout (i,j) tel que $i<j$, soit $r_{i,j}$ le reste de la division de $S(g_i,g_j)$ par G.

Supposons par l'absurde qu'il existe i,j tel que $r_{i,j}\neq 0$. Alors $r_{i,j}\in G'$. Comme $<\mathrm{lt}(G)>=<\mathrm{lt}(G'>$, c'est que $\mathrm{lt}(r_{i,j})\in<\mathrm{lt}(G)>$. Donc il existe $g\in G$ tel que $\mathrm{lt}(g)$ divise $\mathrm{lt}(r_{i,j})$. C'est absurde car $r_{i,j}$ est un reste de la division par G.

Donc pour tout (i,j), $r_{i,j}=0$, ce qui montre que $\mathcal{S}=\emptyset$. □

Remarque. Il est possible d'améliorer cet algorithme en évitant le calcul inutile de certains S-polynômes. Par exemple, on peut montrer que si $\mathrm{pgcd}(\mathrm{lt}(g_i),\mathrm{lt}(g_i))=1$, alors $S(g_i,g_j)\xrightarrow{g_i,g_j}0$. Il n'est donc pas nécessaire de calculer ce S-polynôme.

Remarques sur la complexité (voir [G-G] et [F-G-L-M]).

C'est un problème difficile.

Soit $X^\alpha=x_1^{\alpha_1}\dots x_n^{\alpha_n}$ un monôme de R. On appelle degré de X^α l'entier $\sum_{i=1}^{n}\alpha_i$. Le degré d'un polynôme est le maximum des degrés de ses monômes. Soit $I=\langle f_1,\dots,f_s\rangle$. On note $d=\max\{\deg f_i\ i\in[[1,s]]\}$ le degré maximum de tous les f_i.

1. Les degrés des éléments de la base de Gröbner réduite de I sont inférieurs ou égaux à
$$2\left(\frac{d^2}{2}+d\right)^{2^{n-1}}$$
2. Il existe des idéaux I pour lesquels toute base de Gröbner contient au moins $2^{2^{cn}}$ éléments et des éléments de degré aumoins $2^{2^{cn}}$ pour une constante strictement positive c.
3. Soit K' un corps algébriquement clos contenant K. Si l'ensemble des solutions $\mathcal{S}$ dans K'^n du système $f_i(x)=0$ pour tout $i\in[[1,s]]$ est fini, alors $\mathrm{Card}(\mathcal{S})\leq d^n$. Pour l'ordre lexicographique gradué inverse, on peut dans ce cas calculer une base de Gröbner avec une complexité algébrique polynomiale en d^{n^2}.
4. Généralement, le calcul d'une base de Gröbner est plus rapide si l'ordre choisi est l'ordre lexicographique gradué inverse.
5. Si l'on connaît une base de Gröbner pour un ordre donné, il existe des algorithmes efficaces pour en déduire une base de Gröbner pour un autre ordre.

8.5. Bases de Gröbner réduites

Lemme 8.5.1. *Soit G une base de Gröbner de I. Soit $g \in G$ tel que*

$$lt(g) \in < lt(G \setminus \{g\}) >$$

ce qui veut dire qu'il existe $g' \in (G \setminus \{g\})$ tel que $lt(g')$ divise $lt(g)$. Alors $(G \setminus \{g\})$ est une base de Gröbner de I

Démonstration. Si $\mathrm{lt}(g) \in < \mathrm{lt}\,(G \setminus \{g\}) >$, alors $< \mathrm{lt}(G) > = < \mathrm{lt}\,(G \setminus \{g\}) >$. On en déduit que $< \mathrm{lt}(I) > = < \mathrm{lt}\,(G \setminus \{g\}) >$ puisque $< \mathrm{lt}(I) > = < \mathrm{lt}\,(G) >$. □

Définition 8.5.2. *Une base de Gröbner G est dite minimale si pour tout $g \in G$, les deux propriétés suivantes sont réalisées.*

1. $lc(g) = 1$.
2. $lt(g) \notin < lt(G \setminus \{g\}) >$.

Définition 8.5.3. *Un élément g d'une base de Gröbner G est réduit pour G si aucun terme de g n'appartient à $< lt(G \setminus \{g\}) >$.*

Définition 8.5.4. *Une base de Gröbner minimale G est réduite si tous ses éléments sont réduits pour G.*

Théorème 8.5.5. *Tout idéal de R admet une unique base de Gröbner réduite.*

Démonstration. La preuve de l'existence est laissée en exercice.

Montrons l'unicité. Soient donc G et G' deux bases de Gröbner réduites d'un idéal I. Donc

$$< \mathrm{lt}(I) > = < \mathrm{lt}(G) > = < \mathrm{lt}(G') > .$$

Montrons d'abord que $\mathrm{lt}(G) = \mathrm{lt}(G')$. Soit $g \in G$. Montrons que $\mathrm{lt}(g) \in \mathrm{lt}(G')$. Comme $\mathrm{lt}(g) \in < \mathrm{lt}(G) > = < \mathrm{lt}(G') >$, il existe $g' \in G'$ tel que $\mathrm{lt}(g')$ divise $\mathrm{lt}(g)$. De même, il existe $g'' \in G$ tel que $\mathrm{lt}(g'')$ divise $\mathrm{lt}(g')$. Ainsi, $\mathrm{lt}(g'')$ divise $\mathrm{lt}(g)$. Si $g'' \neq g$, alors $\mathrm{lt}(g)$ est divisible par un élément de $\mathrm{lt}(G \setminus \{g\})$. C'est absurde puisque G est une base de Gröbner réduite. On en déduit donc que $g = g''$. Comme $\mathrm{lt}(g)$ divise $\mathrm{lt}(g')$ et $\mathrm{lt}(g')$ divise $\mathrm{lt}(g)$, et comme $\mathrm{lc}(g) = \mathrm{lc}(g') = 1$ c'est donc que $\mathrm{lt}(g) = \mathrm{lt}(g')$, d'où l'inclusion $\mathrm{lt}(G) \subset \mathrm{lt}(G')$, puis l'égalité $\mathrm{lt}(G) = \mathrm{lt}(G')$.

On peut maintenant montrer que $G = G'$. Soit $g \in G$. Comme $\mathrm{lt}(G) = \mathrm{lt}(G')$, il existe $g' \in G'$ tel que $\mathrm{lt}(g) = \mathrm{lt}(g')$. Alors, les termes dominants de g et de g' s'annulent dans $g - g'$. Comme G et G' sont des bases réduites, et

comme $\mathrm{lt}(G) = \mathrm{lt}(G')$, aucun des termes de g ni de g' n'est divisible par un élément de $< \mathrm{lt}(G) \setminus \{g\} >$, donc aucun des termes de $g - g'$ n'est divisible par un élément de $< \mathrm{lt}(G) >$, ce qui veut dire que le reste de la division de $g - g'$ par G est égal à $g - g'$. Or, comme $g - g' \in I$, ce reste est égal à 0. On en déduit que $g = g'$. Ainsi, $G \subset G'$, et par symétrie $G = G'$. □

8.6. Applications

8.6.1 Monômes standards

On considère toujours $R = K[X_1, \ldots, X_n]$ (qu'on note aussi $K[X]$), muni d'un ordre monomial $\prec$. Soit I un idéal de R et B une base de Gröbner de I pour $\prec$.

Définition 8.6.1. *Soit $\alpha \in \mathbb{N}^n$. Le monôme X^α est un monôme standard de R relativement à G si pour tout $g \in G$, le terme $lt(g)$ ne divise pas X^α.*

Théorème 8.6.2. *L'ensemble des monômes standards relativement à G est une base du K-espace vectoriel $K[X]/I$.*

Démonstration. Tout polynôme f de R s'écrit

$$f = \sum_{g \in G} c_g g + r \tag{8.2}$$

où aucun des termes de r n'est divisible par un élément de $< \mathrm{lt}(G) >$, c'est à dire que les monômes apparaissant dans r sont des monômes standards. L'ensemble des monômes standards est donc une famille génératrice du K-espace vectoriel $K[X]/I$. L'unicité de r dans (8.2) montre que la famille est libre. □

Exemple. Soit $R = \mathbb{Q}[x, y]$, muni de l'ordre lexicographique gradué $\prec$ tel que $x \succ y$. Soient $f_1 = x^3 - 2xy$, $f_2 = x^2y - 2y^2 + x$ et $I =< f_1, f_2 >$. Alors la base de Gröbner réduite de I est égale à $G = (x^2, xy, y^2 - x/2)$. Alors, l'ensemble des monômes standards pour G est égal à

$$\mathcal{M} = \{1, x, y\}.$$

On peut visualiser ces monômes sur un graphique.

L'axe des abscisses représentent les puissances de x, les ordonnées les puissances de y. Comme $x^2 \in \mathrm{lt}(G)$, les points (a, b) tels que $a \geq 2$ sont à exclure (on enlève un quart de plan).

Le $\mathbb{Q}$-espace vectoriel $\mathbb{Q}[x, y]/I$ est donc de dimension 3. $\mathbb{Q}[x, y]/I = \{a_0 + a_1 x + a_2 y \ : \ (a_0, a_1, a_2) \in \mathbb{Q}^3\}$ Pour additionner deux éléments de $\mathbb{Q}[x, y]/I$,

il suffit d'ajouter leurs composantes. Pour la multiplication, on établit la table de multiplication des monômes standards.

$\times$	1	x	y
1	1	x	y
x	x	0	0
y	y	0	$x/2$

Ainsi, les bases de Gröbner permettent de calculer dans de tels quotients. La multiplication par un élément du quotient est une application linéaire. En dimension finie, il peut être utile de calculer la matrice de la multiplication par chacun des éléments d'une base dans ette base.

Dans l'exemple ci-dessus, la multiplication par 1 a pour matrice l'identité, les multiplications par $[x]_I$ et par $[y]_I$ ont respectivement pour matrice dans la base $([1]_I, [x]_I, [y]_I)$

$$M_x = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \text{et} \quad M_y = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{pmatrix}$$

8.6.2 Résolution de systèmes algébriques

Il arrive fréquemment qu'un problème se ramène à la résolution d'un système d'équations polynomiales. Soit à résoudre

$$\begin{cases} f_1(x_1, \ldots, x_n) = 0 \\ \vdots \\ f_s(x_1, \ldots, x_n) = 0 \end{cases} \tag{8.3}$$

Soient $I =< f_1, \ldots, f_s$ et $G = (g_1, \ldots, g_t)$ une base de Gröbner de I. Alors le système (8.3) est équivalent au système

$$\begin{cases} g_1(x_1, \ldots, x_n) = 0 \\ \vdots \\ g_t(x_1, \ldots, x_n) = 0 \end{cases} \tag{8.4}$$

Souvent, ce nouveau système est plus facile à résoudre, car certaines variables peuvent avoir été éliminées dans certaines équations.

Reprenons l'exemple où $f_1 = x^3 - 2xy$, $f_2 = x^2y - 2y^2 + x$ dans $\mathbb{Q}[x, y]$. Alors

$$\begin{cases} f_1(x,y) = 0 \\ f_2(x,y) = 0 \end{cases} \iff \begin{cases} x^2 = 0 \\ xy = 0 \\ y^2 - x/2 = 0 \end{cases} \iff \begin{cases} x = 0 \\ y = 0 \end{cases}$$

Dans cet exemple, l'ordre monomial choisi est l'ordre lexicographique gradué. Ce n'est pas forcément le meilleur choix en général.

L'ordre lexicographique se prête particulièrement bien à l'élimination des variables. Pour tout idéal I de R, on note

$$I_l = I \cap K[X_{l+1}, \ldots, X_n].$$

C'est un idéal de $K[X_{l+1}, \ldots, X_n]$, appelé l-ème idéal d'élimination de I.

Théorème 8.6.3. *Soit $\prec$ l'ordre lexicographique tel que $X_1 \succ \cdots \succ X_n$. Soit G une base de Gröbner de I. Alors*

$$G_l = G \cap K[X_{l+1}, \ldots, X_n]$$

est une base de Gröbner de I_l.

Démonstration. Il s'agit de montrer que $< \mathrm{lt}(I_l) >=< \mathrm{lt}(G_l) >$. L'inclusion $< \mathrm{lt}(G_l) >\subset< \mathrm{lt}(I_l) >$ est claire.

Montrons que $< \mathrm{lt}(I_l) >\subset< \mathrm{lt}(G_l) >$. Soit $f \in I_l$. Alors $f \in I$ et donc comme $< \mathrm{lt}(I) >=< \mathrm{lt}(G) >$, il existe $g \in G$ tel que $\mathrm{lt}(g)$ divise $\mathrm{lt}(f)$. Comme $f \in I_l$, $\mathrm{lt}(f) \in K[X_{l+1}, \ldots, X_n]$ et donc $\mathrm{lt}(g) \in K[X_{l+1}, \ldots, X_n]$. Comme il s'agit de l'ordre lexicographique, tout monôme inférieur à $\mathrm{lt}(g)$ appartient à $K[X_{l+1}, \ldots, X_n]$. Donc tous les termes de g appartiennent à $K[X_{l+1}, \ldots, X_n]$ ce qui veut dire que $g \in K[X_{l+1}, \ldots, X_n]$, donc $g \in G_l$. □

Exemple. Soit à résoudre dans $\mathbb{R}$ le système suivant.

$$\begin{cases} f_1(x,y,z) = 0 \\ f_2(x,y,z) = 0 \\ f_3(x,y,z) = 0 \end{cases} \text{ où } \begin{array}{l} f_1(x,y,z) = x^2 + y + z - 1 \\ f_2(x,y,z) = x + y^2 + z - 1 \\ f_3(x,y,z) = x + y + z^2 - 1 \end{array}$$

Soit I l'idéal $< f_1, f_2, f_3 >$ de $\mathbb{Q}[x, y, z]$. On munit $\mathbb{Q}[x, y, z]$ de l'ordre lexicographique $\prec$ tel que $x \succ y \succ z$. Alors la base de Gröbner réduite de I est $G = (g_1, g_2, g_3, g_4)$, où

$$\begin{aligned} g_1 &= x + y + z^2 - 1 \\ g_2 &= y^2 - y - z^2 + z \\ g_3 &= z^2(y + z^2/2 - 1/2) \\ g_4 &= z^2(z-1)^2(z^2 + 2z - 1) \end{aligned}$$

On remarque que g_4 ne dépend que de z et que g_2 et g_3 ne dépendent que de y et z. Il est alors facile de résoudre le système et on trouve comme ensemble des solutions

$$\mathcal{S} = \big\{(1,0,0),(0,1,0),(0,0,1),(-1+\sqrt{2},-1+\sqrt{2},-1+\sqrt{2}),\\ (-1-\sqrt{2},-1-\sqrt{2},-1-\sqrt{2})\big\}.$$

Bibliographie

[B-W] T. Becker, V. Weispfenning *Gröbner Bases*, Springer.

[F-G-L-M] J.-C. Faugère, P. Gianni, D. Lazard, T. Mora *Efficient computation of zero-dimensional Gröbner Bases by change of ordering*, J. Symbolic Computation (1994).

[G-G] J. von zur Gathen, J. Gerhard *Modern Computer Algebra, second edition* Cambridge University Press (2003).