COURS 4

Division euclidienne, fractions rationnelles et récurrences linéaires à coefficients constants

Résumé

La multiplication et l'inversion rapide de séries permettent le calcul efficace de la division euclidienne de polynômes et du développement en série des fractions rationnelles. Ils mènent en particulier au calcul rapide d'un ou de plusieurs termes d'une suite récurrente linéaire à coefficients constants.

1. Division de polynômes

Étant donnés deux polynômes F et G de $\mathbb{A}[X]$, avec G unitaire, à coefficients dans un anneau $\mathbb{A}$, il existe d'uniques polynômes Q et R, quotient et reste de la division euclidienne de F par G, tels que :

$$F = QG + R \quad \text{avec} \quad \deg R < \deg G.$$

Calculer rapidement le quotient Q et le reste R est d'importance vitale dans toute la suite du cours. Outre l'application aux suites récurrentes qui sera détaillée plus loin, les algorithmes de division seront utilisés dans le Cours 5 pour l'évaluation multipoint et l'interpolation et dans le Cours 10 pour le calcul de pgcd. À leur tour, ces deux algorithmes sont centraux dans nombre d'applications.

1.1. Méthode naïve. L'algorithme naïf pour calculer Q et R consiste à poser la division. Pour cela, les termes dominants sont d'abord isolés :

$$F = aX^m + T_F, \quad G = X^n + T_G, \qquad \text{avec} \quad \deg T_F < m, \quad \deg T_G < n.$$

Si $m < n$, il n'y a rien à faire. Sinon, la boucle élémentaire de la division de F par G consiste à « tuer » le terme de plus haut degré de F en effectuant la soustraction

$$F - aX^\delta G = (aX^m + T_F) - \left(aX^{\delta+n} + aX^\delta T_G\right) = T_F - aX^\delta T_G,$$

où $\delta = m - n$. Le polynôme obtenu est congru à F modulo G, mais de degré strictement inférieur à m. Il n'y a plus qu'à itérer ce procédé jusqu'à obtenir un polynôme de degré strictement inférieur à n, et à garder trace des quotients successifs.

La complexité de cet algorithme est facile à estimer. La boucle élémentaire présentée ci-dessus s'effectue en $O(n)$ opérations de type $(+, -, \times)$; dans le pire des cas, l'algorithme effectue $(m - n + 1)$ passages dans cette boucle, de sorte que la complexité de la division de F par G est de $O(n(m - n))$ opérations. Le pire des cas est atteint pour $m = 2n$. C'est donc un algorithme *quadratique* (mais un bon algorithme quadratique : la constante dans le $O(\cdot)$ est en fait petite, c'est 2). Le même genre d'estimation s'obtient pour la division euclidienne classique des entiers.

1.2. Algorithme rapide. Un bien meilleur résultat peut être obtenu à base d'itération de Newton.

THÉORÈME 1. *Soient G un polynôme unitaire de $\mathbb{A}[X]$ de degré n et $F \in \mathbb{A}[X]$ de degré $m \geq n$. Alors on peut calculer le quotient Q et le reste R de la division euclidienne de F par G en $5\mathsf{M}(m-n) + \mathsf{M}(n) + O(m)$ opérations $(+, -, \times)$ de $\mathbb{A}$.*

Ainsi, la division euclidienne ne coûte pas plus cher que la multiplication (à une constante près). En particulier, si on utilise une multiplication à base de FFT, le coût de la division est *linéaire* en le degré, à des facteurs logarithmiques près.

DÉMONSTRATION. L'idée principale de l'algorithme part de la réécriture de $F = QG + R$ en

$$\frac{F}{G} = Q + \frac{R}{G}.$$

Si on pense que $\mathbb{A} = \mathbb{R}$, on voit donc que le développement asymptotique de $\frac{F}{G}$ *à l'infini* a ses premiers termes donnés par Q, puisque $\frac{R}{G}$ tend vers 0 à l'infini (à cause des contraintes de degré sur R et G). Ceci suggère que l'on va obtenir Q par calcul du développement de Taylor de $\frac{F}{G}$ au voisinage de l'infini.

Concrètement, il suffit de ramener d'abord l'infini en zéro (par le changement de variable $X = 1/T$), pour réduire le calcul à la division de séries formelles. Plus précisément, le point de départ est l'identité

$$\frac{T^m F(1/T)}{T^n G(1/T)} = T^{m-n} Q(1/T) + \frac{T^m R(1/T)}{T^n G(1/T)}.$$

Dans cette identité les numérateurs et dénominateurs des fractions sont des polynômes, et le polynôme $T^n G(1/T)$ a 1 pour terme constant. En outre, la valuation du second sommant du membre droit est supérieure à $m - n$. En découle donc l'algorithme suivant :

1. Calculer $T^m F(1/T)$ et $T^n G(1/T)$ (aucune opération arithmétique n'est nécessaire, il suffit d'inverser l'ordre des coefficients) ;
2. Calculer le quotient $(T^m F(1/T))/(T^n G(1/T)) \bmod T^{m-n+1}$ par une inversion de série formelle suivie d'un produit ;
3. en déduire Q en inversant l'ordre des coefficients ;
4. en déduire R, qui est donné par $R = F - QG \bmod X^n$.

Ces formules mènent au résultat de complexité du Théorème. D'après la Proposition 1 et la remarque qui la suit, l'inverse du dénominateur à l'étape 2 s'obtient en

$$4\mathsf{M}(m-n) + O(m-n)$$

opérations dans $\mathbb{A}$, puis une multiplication supplémentaire donne Q. Pour retrouver R, le dernier produit est effectué modulo X^n, c'est-à-dire pour un coût de $\mathsf{M}(n)$. Toutes les additions sont prises en compte dans le terme $O(m)$. □

REMARQUE. Si Q a un degré beaucoup plus petit que G, on peut accélérer le calcul de QG, en découpant G en « tranches » de taille $m - n$; de la sorte, le dernier produit peut se faire en $\frac{n}{m-n}\mathsf{M}(m-n)$ opérations. Enfin, si de nombreuses réductions sont à faire modulo le même polynôme G, il est évidemment recommandé de stocker l'inverse du réciproque de $T^n G(1/T)$ de G.

1.3. Le cas des entiers. Comme pour la multiplication, il est possible de poser le problème dans $\mathbb{Z}$ comme dans un anneau de polynômes. C'est ce dernier cas qui est le plus simple à étudier, puisqu'il n'y a pas à y gérer de retenue. On obtient cependant un résultat analogue sur les entiers.

THÉORÈME 2. *Soit $\mathsf{M}_{\mathbb{Z}}$ une fonction de multiplication pour $\mathbb{Z}$, et f et g deux entiers positifs avec $g \leq f \leq 2^n$. Soient q et r les quotient et reste de la division euclidienne de f par g :*

$$f = qg + r \quad \textit{avec} \quad 0 \leq r < g.$$

On peut calculer q et r en $O(\mathsf{M}_{\mathbb{Z}}(n))$ opérations binaires.

EXERCICE 1. Estimer la constante dans la complexité ci-dessus.

1.4. Application aux calculs modulaires. Une application importante de la division euclidienne rapide est le calcul efficace dans des structures algébriques de type « quotient », par exemple dans n'importe quel corps fini. Si $\mathbb{A}$ est un anneau et si P est un polynôme unitaire de $\mathbb{A}[X]$, il s'agit de calculer modulo P, c'est-à-dire de rendre effectives les opérations $(+, -, \times)$ dans $\mathbb{A}[X]/(P)$. L'addition dans $\mathbb{A}[X]/(P)$ est facile à opérer : on peut la faire terme à terme, ainsi la complexité est linéaire en le degré n de P. Ensuite, pour multiplier deux éléments $A + (P)$ et $B + (P)$ de $\mathbb{A}[X]/(P)$, on commence par multiplier A et B dans $\mathbb{A}[X]$, puis on réduit le résultat modulo P. D'après ce qui précède, la complexité du produit dans $\mathbb{A}[X]/(P)$ est en $O(\mathsf{M}(n))$.

La question du calcul dans $\mathbb{Z}/n\mathbb{Z}$ est légèrement plus délicate que son analogue dans $\mathbb{A}[X]$, à cause des retenues. Malgré tout, les résultats s'étendent.

2. Fractions rationnelles et récurrences à coefficients constants

Si $\mathbb{A}$ est un anneau, on note $\mathbb{A}(X)$ l'anneau des fractions rationnelles sur $\mathbb{A}$. Ses éléments sont de la forme $A(X)/B(X)$, avec $A, B \in \mathbb{A}[X]$ et $B \neq 0$. Les opérations de $\mathbb{A}(X)$ sont l'addition et la multiplication habituelle des fractions

$$\frac{A(X)}{B(X)} + \frac{C(X)}{D(X)} = \frac{A(X)D(X) + B(X)C(X)}{B(X)D(X)}, \quad \frac{A(X)}{B(X)} \times \frac{C(X)}{D(X)} = \frac{A(X)C(X)}{B(X)D(X)}.$$

Le degré d'une fraction rationnelle A/B est défini comme $\max(\deg(A), \deg(B))$. Deux fractions rationnelles de degré strictement inférieur à n peuvent donc être additionnées et multipliées en $O(\mathsf{M}(n))$ opérations dans $\mathbb{A}$.

Si, de plus, $B(0)$ est inversible dans $\mathbb{A}$, alors on peut voir la fraction rationnelle A/B comme une série de $\mathbb{A}[[X]]$, obtenue en divisant la série A par B. Le résultat $\sum_i c_i X^i$ de cette division sera appelé *le développement de Taylor* de $A(X)/B(X)$. On dit alors que $\sum_i c_i X^i$ est une *série rationnelle*. Les séries rationnelles forment un sous-anneau de $\mathbb{A}[[X]]$.

Mathématiquement, les fractions rationnelles et les séries rationnelles sont deux incarnations du même concept. Du point de vue algorithmique, ce dédoublement permet de coder une fraction rationnelle A/B de deux manières différentes, soit par le couple (A, B), soit par le développement en série tronqué de A/B. Les deux structures de données nous seront utiles dans ce cours, ainsi nous nous intéressons au passage de l'une à l'autre en bonne complexité.

EXERCICE 2. Montrer qu'une fraction rationnelle est uniquement déterminée par les $\deg(A)+\deg(B)$ premiers termes de son développement en série de Taylor.

On peut calculer le développement de Taylor à l'ordre N d'une fraction rationnelle de degré $d \leq N$ en $O(\mathsf{M}(N))$ opérations, en utilisant la méthode de Newton, décrite dans le Cours 3. Mais $O(\mathsf{M}(N))$ est en général beaucoup plus gros que les $O(dN)$ opérations requises par la méthode naïve. Une meilleure solution est fournie par la Théorème suivant. Le problème inverse, tout aussi fondamental, consiste à retrouver la forme rationnelle à partir des premiers termes de la série rationnelle, et sera traité efficacement dans le Cours 10.

THÉORÈME 3. *Soit $A(X)/B(X)$ dans $\mathbb{A}(X)$ de degré au plus d, avec $B(0)$ inversible. Le développement de Taylor de $A(X)/B(X)$ à précision $N \geq d$ peut se calculer en $O(N\mathsf{M}(d)/d)$ opérations $(+,-,\times)$ dans $\mathbb{A}$. Le N^e coefficient c_N peut se calculer en $O(\mathsf{M}(d)\log N)$ opérations dans $\mathbb{A}$.*

Si l'anneau $\mathbb{A}$ permet la FFT, ces estimations de complexité deviennent $O(N\log d)$ et $O(d\log d\log N)$; elles sont quasi-optimales, simultanément vis-à-vis de N et de d.

DÉMONSTRATION. Les deux assertions découlent du lemme suivant.

LEMME 1. *Soit $A(X)/B(X)$ dans $\mathbb{A}(X)$ avec $B(0)$ inversible. Soit d le degré de B et soit $\kappa \geq 0$. Alors les coefficients $c_\kappa, c_{\kappa+1}, \ldots, c_{\kappa+d-1}$ du développement de $A/B = \sum_i c_i X^i$ sont les coefficients de $X^{2d-2}, \ldots, X^d, X^{d-1}$ du produit*

$$\big(X^\kappa \bmod B(1/X)X^d\big)\big(c_{2d-2} + \ldots + c_0 X^{2d-2}\big).$$

En admettant ce lemme, il est aisé de démontrer le Théorème 3. En effet, pour calculer $c_0, \ldots, c_N$, il suffit d'appliquer le lemme $\lceil N/d \rceil$ fois en prenant $\kappa = 0, d, 2d, \ldots$, ce qui ramène le problème au calcul des restes de $X^{d\kappa}$ modulo $\bar{B} = B(1/X)X^d$. Ceci peut se faire en utilisant $O(N/d)$ opérations dans $\mathbb{B} = \mathbb{A}[X]/(\bar{B})$, en posant d'abord $y = x^d$, où x est l'image de X dans l'anneau quotient $\mathbb{B}$, et en calculant ensuite $y^2, y^3, \ldots$ par multiplications successives par y.

De même, pour calculer un seul terme c_N, on applique le lemme à $\kappa = N$ et tout revient à déterminer le reste de X^N modulo $\bar{B}$, donc au calcul de la N^{e} puissance de x dans $\mathbb{B}$. Or, cela peut se faire en $O(\log N)$ multiplications dans $\mathbb{B}$ en exploitant récursivement l'identité

$$x^\kappa = \begin{cases} (x^{\kappa/2})^2, & \text{si } \kappa \text{ est pair,} \\ x \cdot (x^{\frac{\kappa-1}{2}})^2, & \text{sinon.} \end{cases}$$

Comme chaque opération dans $\mathbb{B}$ coûte $O(\mathsf{M}(d))$ opérations dans $\mathbb{A}$, on obtient les estimations de la Proposition. Notons pour conclure que les $2d-1$ premiers coefficients $c_0, \ldots, c_{2d-2}$ de A/B se calculent par itération de Newton en $O(\mathsf{M}(d))$ opérations dans $\mathbb{A}$, ce qui représente dans les deux cas un coût négligeable. □

DÉMONSTRATION. [du Lemme] Partons de l'observation que la matrice de l'application $\mathbb{A}$-linéaire $X\cdot : \mathbb{B} \to \mathbb{B}$ dans la base canonique $\{1, X, \ldots, X^{d-1}\}$ est la matrice compagnon C telle que $[c_{n+1}, \ldots, c_{n+d}] = [c_n, \ldots, c_{n+d-1}] \cdot C$ pour tout $n \geq 0$. De cette dernière égalité il s'ensuit que $c_{\kappa+i}$ est égal au produit du vecteur $[c_i, \ldots, c_{d+i-1}]$ et de la première colonne de la matrice C^κ. Or, puisque C^κ est la matrice de multiplication par X^κ, les entrées de sa première colonne sont précisément les coefficients du polynôme $X^\kappa \bmod \bar{B}$. □

3. Suites récurrentes linéaires à coefficients constants

3.1. Prélude : les nombres de Fibonacci. La suite de Fibonacci, introduite vers 1202 par Leonardo Pisano[1] dans un problème récréatif décrivant la croissance d'une population de lapins, est définie par les conditions initiales $F_0 = 0, F_1 = 1$ et la récurrence

$$F_{n+2} = F_{n+1} + F_n, \quad n \geq 0. \tag{1}$$

La suite de Fibonacci jouit de riches propriétés algébriques, arithmétiques et combinatoires. Par exemple : (a) F_n est le nombre de façons différentes de paver un rectangle $2 \times (n-1)$ au moyen de dominos 2×1 ; (b) $(F_n)_n$ est une *suite de divisibilité, i. e.* F_n divise F_m dès lors que n divise m ; (c) si n est impair, alors

$$F_n = 2^{n-1} \prod_{k=1}^{\frac{n-1}{2}} \left(\frac{1}{4} + \cos^2 \frac{k\pi}{n} \right)$$

Exercice 3. Prouver les assertions (a)–(c).

Malgré sa simplicité, la suite de Fibonacci fait l'objet de nombreux problèmes ouverts. Par exemple, on ignore s'il existe une infinité de nombres de Fibonacci premiers. Les nombres de Fibonacci sont omniprésents en mathématiques et en informatique[2] : ils interviennent aussi bien dans l'analyse de l'algorithme d'Euclide pour le calcul du plus grand commun diviseur de deux entiers, que dans la solution négative de Matiyasevich du dixième problème de Hilbert[3].

Du point de vue algorithmique, on s'intéresse typiquement au calcul efficace d'un terme F_N de la suite de Fibonacci (pour N « grand »)[4] ou de ses N premiers termes. Nous traitons ces questions dans un cadre plus général.

3.2. Calcul rapide des termes d'une suite. Une suite $(a_n)_{n\geq 0}$ d'éléments de l'anneau $\mathbb{A}$ est appelée suite récurrente linéaire à coefficients constants (srlcc) d'ordre d si elle satisfait une récurrence de la forme

$$a_{n+d} = p_{d-1}a_{n+d-1} + \cdots + p_0 a_n, \qquad n \geq 0,$$

où les p_i sont des éléments de $\mathbb{A}$. Le polynôme $P = X^d - p_{d-1}X^{d-1} - \cdots - p_0$ de $\mathbb{A}[X]$ est appelé *polynôme caractéristique* de la suite $(a_n)_{n\geq 0}$.

Exercice 4. Soit M une matrice de taille $d \times d$ à coefficients dans $\mathbb{A}$, de polynôme caractéristique $\chi_M(X) = \det(XI_d - M)$, soit u une matrice ligne et v une matrice colonne. Montrer que la suite $(uM^n v)_{n\geq 0}$ est une suite récurrente linéaire à coefficients constants admettant χ_M comme polynôme caractéristique.

Exercice 5. Soit $a_1, \ldots, a_r$ des entiers strictement positifs. Soit A_n et B_n le nombre de r-uplets non-ordonnés, resp. ordonnés, $(x_1, \ldots, x_r) \in \mathbb{N}^r$, solutions de l'équation $a_1x_1 + \cdots + a_rx_r = n$. Montrer que les suites (A_n) et (B_n) sont récurrentes linéaires à coefficients constants, admettant $(X^{a_1} - 1) \cdots (X^{a_r} - 1)$ et $X^{a_1} + \cdots + X^{a_r} - 1$ comme polynômes caractéristiques.

---

[1] Mieux connu sous le pseudonyme de Fibonacci.

[2] Le journal *The Fibonacci Quarterly* est entièrement dédié à l'étude de ses propriétés.

[3] Ce problème proposait de trouver un algorithme pour décider si un système d'équations diophantiennes (polynômes à coefficients entiers) admet une solution en nombres entiers.

[4] On pourra admirer $F_{1\,000\,000}$ à l'url http://www.upl.cs.wisc.edu/~bethenco/fibo/

EXERCICE 6. Si $P \in \mathbb{A}[X]$ est de degré d, alors la suite $(P(n))_{n\geq 0}$ est une srlcc de polynôme caractéristique $(X-1)^{d+1}$.

3.2.1. *Exponentiation binaire.* L'approche naïve pour le calcul du N^{e} terme d'une srlcc consiste tout simplement à dérouler la récurrence et requiert $O(dN)$ opérations dans $\mathbb{A}$. Une alternative à cette méthode naïve est basée sur l'exponentiation binaire de la matrice compagnon associée à la suite. Par exemple, la récurrence (1) se récrit matriciellement

$$\begin{pmatrix} F_N \\ F_{N-1} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}}_{C} \begin{pmatrix} F_{N-1} \\ F_{N-2} \end{pmatrix} = C^{N-1} \begin{pmatrix} F_1 \\ F_0 \end{pmatrix}, \qquad N \geq 1.$$

La puissance de la matrice constante C se calcule alors récursivement par

$$C^N = \begin{cases} (C^{N/2})^2, & \text{si } N \text{ est pair,} \\ C \cdot (C^{\frac{N-1}{2}})^2, & \text{sinon.} \end{cases}$$

On en déduit que le calcul de F_N peut être effectué sans calculer les précédents en $O(\log N)$ produits de matrices 2×2, soit $O(\log N)$ opérations dans $\mathbb{A}$.

EXERCICE 7. Montrer que le N^{e} terme de toute récurrence linéaire d'ordre d à coefficients constants peut se calculer en $O(d^3 \log N)$ opérations dans $\mathbb{A}$.

En utilisant des algorithmes plus évolués pour la multiplication matricielle, on peut abaisser cette complexité à $O(d^\omega \log N)$ opérations dans $\mathbb{A}$, où $2 \leq \omega < 2,38$ (voir Cours 7). La dépendance en d reste cependant plus que quadratique. La section suivante présente une meilleure méthode, de complexité quasi-linéaire en d.

3.2.2. *Exponentiation modulaire.* La complexité arithmétique de l'algorithme de la section précédente (Exercice 7) est quasi-optimale par rapport à l'indice N, mais pas par rapport à l'ordre d de la récurrence. Une méthode plus efficace, quasi-optimale à la fois en d et N, exploite le lien très étroit entre fractions rationnelles et suites récurrentes à coefficients linéaires. Ce lien est précisé dans le résultat suivant.

LEMME 2. *La série génératrice d'une srlcc est rationnelle. Plus exactement,*

$$\sum_{n\geq 0} a_n X^n = \frac{N(X)}{\bar{P}(X)},$$

où N_0 est un polynôme de degré $< d$ et $\bar{P}(X) = P(1/X)X^{\deg(P)}$ est le polynôme réciproque du polynôme caractéristique de la suite.

EXERCICE 8. Prouver ce lemme.

EXEMPLE 1. La série génératrice de la suite de Fibonacci est $\frac{X}{1-X-X^2}$.

Le lemme 3.2.2 montre donc qu'il y a équivalence entre le calcul des premiers termes de suites récurrentes linéaires et le développement en série de Taylor de fractions rationnelles. En vue du Théorème 3, cela entraîne le résultat suivant.

THÉORÈME 4. *Soit (a_n) une suite récurrente linéaire, donnée par une récurrence d'ordre d et des conditions initiales $a_0, \ldots, a_{d-1}$. Soit $N \geq d$. Alors, les N premiers termes $a_0, \ldots, a_N$ peuvent se calculer en $O(N\mathsf{M}(d)/d)$ opérations dans $\mathbb{A}$; le calcul du N^e terme peut se faire en seulement $O(\mathsf{M}(d) \log N)$ opérations dans $\mathbb{A}$.*

EXERCICE 9. Un polynôme $P \in \mathbb{A}[X]$ de degré d peut être évalué aux $N \gg d$ points $1, 2, \ldots, N$ en $O(N\,\mathsf{M}(d)/d)$ opérations dans $\mathbb{A}$.

3.3. Propriétés de clôture. La classe des srlcc admet de nombreuses propriétés de clôture : si $\mathbf{a} = (a_n)_n$ et $\mathbf{b} = (b_n)_n$ sont deux srlcc de polynômes caractéristiques P et Q, alors

1. la somme $\mathbf{a} + \mathbf{b} = (a_n + b_n)_n$ et le produit de Cauchy $\mathbf{a} \star_{\mathbf{C}} \mathbf{b}$, de terme général $\sum_{i=0}^{n} a_i b_{n-i}$, sont deux srlcc de polynôme caractéristique PQ ;
2. le produit de Hadamard $\mathbf{a} \star_{\mathbf{H}} \mathbf{b} = (a_n b_n)_n$ est une srlcc de polynôme caractéristique le produit composée $P \otimes Q$ définie au Cours 3 ;
3. la suite $\left(\sum_{i=0}^{n} \binom{n}{i} a_i b_{n-i}\right)_n$ est une srlcc de polynôme caractéristique la somme composée $P \oplus Q$.

EXERCICE 10. Prouver les assertions précédentes.

EXERCICE 11. Lorsque les coefficients de la récurrence sont des entiers, étudier la complexité binaire de tous les algorithmes traités dans ce cours.

3.4. Application : tests de primalité. Une application du calcul rapide d'un terme d'une récurrence est une famille de tests probabilites de primalité, de complexité polynomiale. L'idée est de construire une suite récurrente (a_n) d'entiers telle que la primalité de n soit équivalente (ou presque équivalente) à $a_n = 0 \bmod n$.

Un cas particulier important en est *le test de Fermat* (pour lequel $a_n = a^{n-1} - 1$) implanté dans la plupart des systèmes de calcul formel. Bien qu'il soit probabiliste (si n ne passe pas le test, n est composé, mais si n passe le test, alors il est premier seulement avec une grande probabilité), sa grande simplicité le rend souvent préférable à d'autres algorithmes sophistiqués.

EXERCICE 12. Soit (a_n) une srlcc d'ordre d. Montrer qu'il existe des constantes entières $c_0, c_1, \ldots, c_d$ telles que p divise $c_0 + c_1 a_{p-1} + \cdots + c_d a_{p-d}$ dès lors que p est un nombre premier. De plus, pour tout premier p, les constantes $c_i \bmod p$ peuvent être trouvées en $O(\mathsf{M}(d))$ opérations arithmétiques dans $\mathbb{A} = \mathbb{Z}/p\mathbb{Z}$.

Par exemple, si (F_n) est la suite de Fibonacci, alors p divise $F_{p-2} + 3F_{p-1} - 1$ dès lors que p est premier et la réciproque est vraie avec une bonne probabilité. L'exercice précédent fournit un test de primalité similaire au test de Fermat. D'après le Théorème 4 son coût est de $O(\mathsf{M}(d) \log p)$ opérations arithmétiques dans $\mathbb{Z}/p\mathbb{Z}$, soit $O(\mathsf{M}(d)\mathsf{M}_{\mathbb{Z}}(\log p) \log p)$ opérations binaires.

EXERCICE 13. Soient a et N deux entiers premiers entre eux. Montrer que N est premier *si et seulement si* $X^N + a = (X + a)^N \bmod N$ dans $\mathbb{Z}[X]$.

EXERCICE 14. Montrer que si N est premier, $0 \leq a < N$ et $P(X) \in \mathbb{Z}[X]$, alors $X^N + a = (X + a)^N \bmod P(X)$ dans $\mathbb{Z}/N\mathbb{Z}[X]$; si de plus, $P(X)$ est de degré $r = O(\log^c N)$, pour un $c > 0$, alors cette égalité peut être testée « en temps polynomial », c'est-à-dire en un nombre d'opérations binaires polynomial en $\log N$.

Notes

Historiquement, le premier algorithme rapide pour la division des polynômes est dû à Moenck et Borodin [7]. Son point clef est que *le quotient de la division euclidienne de deux polynômes ne dépend que de leurs coefficients de poids fort.* Cette remarque sera également à la base du calcul rapide de pgcd, étudié au Cours 10.

L'algorithme de la Section 1.2 est dû à Strassen [10]. Une alternative, de même complexité asymptotique, à l'algorithme esquissé en Section 1.4 pour les calculs modulaires a été proposée par Montgomery [8].

Calculer les N premiers termes d'une srlcc de polynôme caractéristique fixé est une opération linéaire en les conditions initiales ; c'est le *dual* de l'opération de division d'un polynôme de degré N par un polynôme fixé. Les conséquences algorithmiques de ce fait seront décrites dans le Cours 32.

Le théorème de Skolem-Mahler affirme que pour toute srlcc (a_n), l'ensemble de ses zéros (les indices i pour lesquels $a_i = 0$) est la réunion d'un ensemble fini et d'un nombre fini de suites arithmétiques. Son étude est une question subtile. Par exemple, déterminer si l'ensemble des zéros est vide est un problème NP-dur [2]. Les srlcc sont étudiées dans [3, 11]. Le livre [4] constitue une référence très complete.

L'exercice 4 est le point clef de la fameuse méthode de Wiedemann pour la résolution de systèmes linéaires creux, qui sera traitée dans le Cours 16.

Le calcul rapide d'un terme de la suite de Fibonacci par exponentiation binaire de la matrice compagnon associée relève du folklore mathématique. Sa généralisation (Exercice 7) est décrite dans [6], mais était probablement connue bien avant.

Les Théorèmes 3 et 4 sont tirés de [5] et [9]. Leur récente généralisation au cas des matrices polynomiales est à la base du meilleur algorithme pour la résolution de systèmes linéaires à coefficients polynomiaux, qui sera exposé dans le Cours 9.

Il n'existe pas de tests *déterministes* de primalité basés sur le test modulaire d'un terme d'une récurrence à coefficients constants. Par contre, on peut caractériser un nombre premier N à l'aide de suites qui vérifient des récurrences à coefficients polynomiaux (comme la factorielle, via le test de Wilson $(N-1)! = -1 \bmod N$). Malheureusement, cela ne fournit pas d'algorithme efficace. Le premier algorithme déterministe qui prouve la primalité en temps polynomial est très récent [1]. Cet article part de la caractérisation de type Fermat donnée en Exercice 14, et exhibe une constante c et un polynôme P tels que la primalité de N est impliquée par la vérification de l'identité de l'Exercice 14 pour seulement $r^{1/2}\log(N)$ valeurs de a.

Bibliographie

[1] Agrawal (Manindra), Kayal (Neeraj), and Saxena (Nitin). – PRIMES is in P. *Annals of Mathematics. Second Series*, vol. 160, n° 2, 2004, pp. 781–793.

[2] Blondel (Vincent D.) and Portier (Natacha). – The presence of a zero in an integer linear recurrent sequence is NP-hard to decide. *Linear Algebra and its Applications*, vol. 351/352, 2002, pp. 91–98. – Fourth special issue on linear systems and control.

[3] Cerlienco (L.), Mignotte (M.), and Piras (F.). – Suites récurrentes linéaires. Propriétés algébriques et arithmétiques. *L'Enseignement Mathématique*, vol. XXXIII, 1987, pp. 67–108. – Fascicule 1-2.

[4] Everest (Graham), van der Poorten (Alf), Shparlinski (Igor), and Ward (Thomas). – *Recurrence sequences*. – American Mathematical Society, Providence, RI, 2003, *Mathematical Surveys and Monographs*, vol. 104, xiv+318p.

[5] Fiduccia (C. M.). – An efficient formula for linear recurrences. *SIAM Journal on Computing*, vol. 14, n° 1, 1985, pp. 106–112.

[6] Miller (J. C. P.) and Brown (D. J. Spencer). – An algorithm for evaluation of remote terms in a linear recurrence sequence. *Computer Journal*, vol. 9, 1966, pp. 188–190.

[7] Moenck (R. T.) and Borodin (A.). – Fast modular transforms via division. *Thirteenth Annual IEEE Symposium on Switching and Automata Theory (Univ. Maryland, College Park, Md., 1972)*, 1972, pp. 90–96.

[8] Montgomery (Peter L.). – Modular multiplication without trial division. *Mathematics of Computation*, vol. 44, n° 170, 1985, pp. 519–521.

[9] Shoup (V.). – A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic. In *Proceedings of ISSAC'91*. pp. 14–21. – ACM Press, 1991.

[10] Strassen (V.). – Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten. *Numerische Mathematik*, vol. 20, 1972/73, pp. 238–251.

[11] van der Poorten (A. J.). – Some facts that should be better known, especially about rational functions. In *Number theory and applications*, pp. 497–528. – Kluwer, Dordrecht, 1989. Proceedings of a Conference held at Banff, AB, 1988.