COURS 3

Calculs rapides sur les séries

Résumé

La multiplication rapide de séries vue au Cours 2 permet des calculs efficaces comme l'inversion, l'exponentiation ou la prise de logarithme d'une série. Ces opérations sont effectuées grâce à une version formelle de la méthode de Newton. Elles entraînent une algorithmique efficace sur les polynômes. La composition de séries est en générale plus coûteuse que le produit, mais peut aussi tirer parti d'une multiplication rapide.

Ce cours porte sur le calcul des premiers termes de développements en séries. Pour fixer les notations, N sera utilisé pour représenter le nombre de termes à calculer, et « série » sera employé pour « série tronquée à l'ordre N ». Ainsi, « calculer une série » voudra toujours dire en calculer les N premiers termes. Comme dans le cours précédent, $\mathsf{M}(N)$ dénote une borne supérieure sur le nombre d'opérations arithmétiques nécessaires pour multiplier deux polynômes de degré au plus N. L'efficacité des algorithmes sera mesurée par leurs complexités arithmétiques.

1. Séries formelles

Si $\mathbb{A}$ est un anneau, on note $\mathbb{A}[[X]]$ l'anneau des séries formelles sur $\mathbb{A}$. Ses éléments sont des suites $(a_i)_{i\in\mathbb{N}}$ de $\mathbb{A}$, notées

$$\sum_{i\geq 0} a_i X^i.$$

Les opérations de $\mathbb{A}[[X]]$ sont l'addition des suites et une multiplication qui généralise la multiplication des polynômes :

$$\sum_{i\geq 0} a_i X^i \times \sum_{i\geq 0} b_i X^i = \sum_{i\geq 0} c_i X^i, \qquad \text{avec} \quad c_i = \sum_{j+k=i} a_j b_k,$$

la dernière somme étant finie.

Exercice 1. Vérifier que ces deux opérations font de $\mathbb{A}[[X]]$ un anneau. Quels en sont les éléments inversibles ?

Lorsque les coefficients sont des nombres complexes, la question de la convergence des séries entières représentées par ces séries formelles ne se posera pas pour les algorithmes considérés dans ce cours. En revanche, il est possible de définir une distance entre deux séries f et g par $d(f,g) = 2^{-\operatorname{val}(f-g)}$, où la *valuation* $\operatorname{val}(f)$ d'une série f est l'indice de son premier terme non-nul (et par convention $\operatorname{val}(0) = \infty$).

Exercice 2. Vérifier que la fonction d définie ci-dessus est bien une distance. Prouver que muni de cette distance, l'anneau des séries formelles forme un espace métrique complet (les suites de Cauchy convergent).

Comme $\mathbb{A}[[X]]$ est un anneau, il peut être utilisé comme anneau de base pour définir des séries formelles en une autre variable y, ce qui définit de la même manière l'anneau des séries formelles bivariées $\mathbb{A}[[X]][[Y]]$, noté aussi $\mathbb{A}[[X, Y]]$.

Algorithmiquement, on ne manipule pas de série à précision « infinie », mais seulement des troncatures, c'est-à-dire un certain nombre des premiers termes. Les séries tronquées deviennent alors de simples polynômes, pour lesquels on dispose en particulier d'algorithmes rapides de multiplication. Étant donnés les N premiers termes d'une série f, l'objectif de ce cours est de donner des algorithmes permettant de calculer efficacement les N premiers termes d'autres séries définies à partir de f.

Pour étendre la notation utilisée avec les polynômes, étant donnée la série

$$S = \sum_{i \geq 0} a_i X^i,$$

on notera $S \bmod X^N$ le polynôme

$$S \bmod X^N := \sum_{0 \leq i < N} a_i X^i.$$

On notera $S = f + O(X^N)$ si les séries ou polynômes S et f coïncident jusqu'au terme de degré N.

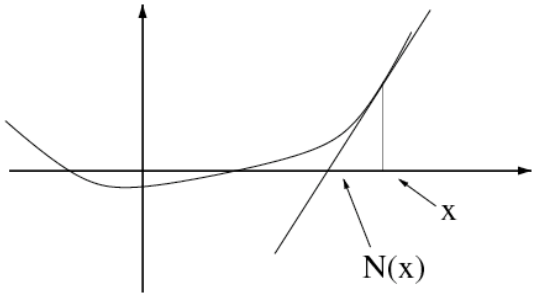
2. La méthode de Newton

L'opérateur de Newton est très largement utilisé en calcul numérique, pour trouver des solutions approchées d'équations. C'est un processus itératif, qui consiste à chaque étape à remplacer le système que l'on veut résoudre par son linéarisé au voisinage de la dernière approximation trouvée. L'adaptation de cette méthode dans un cadre formel est extrêmement fructueuse : elle permet de trouver les solutions séries de grandes variétés d'équations, avec un très bon comportement vis-à-vis de la complexité.

2.1. Version numérique. Soit f une fonction $\mathbb{R} \to \mathbb{R}$. Pour résoudre $f(g) = 0$ dans $\mathbb{R}$, on choisit $g_0 \in \mathbb{R}$ et on itère selon

$$g_{k+1} = N(g_k) = g_k - \frac{f(g_k)}{f'(g_k)}.$$

Graphiquement, l'itération correspond à la figure suivante :



Opérateur de Newton de $\mathbb{R}$ dans $\mathbb{R}$.

La solution serait exacte si f était une fonction linéaire ; dans le cas général, si g_0 est bien choisi (par exemple assez proche d'une racine simple), la suite g_k converge vers une limite g_∞. En outre, à l'itération k la distance à la limite est de l'ordre du carré de cette distance à l'itération $k - 1$. On parle dans ce cas de

convergence *quadratique*. Dans ces conditions, pour k suffisamment grand, le nombre de décimales correctes est approximativement *doublé* à chaque itération.

2.2. Version formelle. L'idée de l'itération de Newton s'adapte au calcul sur les séries, et la convergence reste quadratique, ce qui veut dire dans ce contexte que le nombre de termes corrects double à chaque itération.

THÉORÈME 1 (Itération de Newton sur les séries). *Soit $\Phi(X,Y)$ une série bivariée en X et $Y-\lambda$, où λ est une constante telle que $\Phi(0,\lambda)=0$ et $\frac{\partial\Phi}{\partial Y}(0,\lambda)$ est inversible dans $\mathbb{A}$. Alors, l'équation*

$$\Phi(X,Y)=0$$

admet une série y solution telle que $y(0)=\lambda$. L'itération

$$y_{k+1}=y_k-\frac{\Phi(X,y_k)}{\frac{\partial\Phi}{\partial Y}(X,y_k)} \bmod X^{2^{k+1}}$$

avec $y_0=\lambda$ converge vers cette solution avec $y-y_k=O(X^{2^k})$.

Avant de prouver ce théorème, il faut observer qu'en pratique il est souhaitable de programmer cette itération de manière récursive en partant de la précision N souhaitée et en divisant par 2 la précision à chaque étape. Si N est juste au-dessus d'une puissance de 2, le gain, quoique d'un facteur constant seulement, est appréciable.

La convergence mentionnée dans ce théorème est relative à la métrique des séries formelles.

DÉMONSTRATION. La preuve consiste à montrer par récurrence sur k que d'une part $\Phi(X,y_k)=O(X^{2^k})$ et d'autre part $y_k-y_{k+1}=O(X^{2^k})$ (et donc aussi que $y_k(0)=\lambda$).

Ces deux conditions permettent de conclure. En effet, la seconde condition montre que les y_k forment une suite de Cauchy. La limite de cette suite existe donc. Notons la y. En faisant tendre k vers l'infini dans

$$\Phi(X,y_k)=\Phi(X,y)+(y_k-y)\frac{\partial\Phi}{\partial Y}(X,y)+O((y_k-y)^2),$$

on voit que la première condition entraîne $\Phi(X,y)=0$. Ceci prouve qu'il existe une solution dans l'anneau des séries formelles, et que cette solution est y. Cette même équation donne alors

$$(y_k-y)\frac{\partial\Phi}{\partial Y}(X,y)+O((y_k-y)^2)=O(X^{2^k}),$$

dont se déduit ensuite $y-y_k=O(X^{2^k})$: comme $y_k(0)=\lambda$ pour tout k, $y(0)$ a la même valeur et donc d'une part $\mathrm{val}((y_k-y)^2)>\mathrm{val}(y_k-y)$ et d'autre part $\mathrm{val}(\frac{\partial\Phi}{\partial Y}(X,y))=0$ d'après l'hypothèse $\frac{\partial\Phi}{\partial Y}(0,\lambda)$ inversible.

Voici maintenant la preuve de la récurrence mentionnée ci-dessus. Pour $k=0$ la première condition est une conséquence de $\Phi(0,\lambda)=0$ et la seconde de

$$y_0-y_1=\frac{\Phi(X,\lambda)}{\frac{\partial\Phi}{\partial Y}(X,\lambda)}+O(X^2),$$

le dénominateur ayant un terme constant inversible et le numérateur valuation au moins 1. Pour passer de k à $k+1$, on utilise d'abord un développement de Taylor qui nous donne

$$\Phi(X, y_{k+1}) = \Phi(X, y_k) - (y_k - y_{k+1})\frac{\partial \Phi}{\partial Y}(X, y_k) + O(X^{2^{k+1}}).$$

Ensuite, la définition de y_{k+2} permet de conclure :

$$y_{k+1} - y_{k+2} = \frac{\Phi(X, y_{k+1})}{\frac{\partial \Phi}{\partial Y}(X, y_{k+1})} \bmod X^{2^{k+2}} = O(X^{2^{k+1}}).$$

□

3. Opérations quasi-optimales

Les applications les plus directes de la méthode de Newton sont résumées dans le théorème suivant.

THÉORÈME 2. *Soit f une série de $\mathbb{A}[[X]]$. On peut calculer en $O(\mathsf{M}(N))$ opérations les N premiers termes de*

1. *$1/f$, lorsque $f(0)$ est inversible ;*
2. *$\log f$ et f^α lorsque $f(0) = 1$;*
3. *$\exp(f)$ lorsque $f(0) = 0$.*

Pour les points 2 et 3, nous supposons aussi que $2, 3, \ldots, N-1$ sont inversibles dans $\mathbb{A}$.

3.1. Inversion de séries. L'inversion de séries donne une illustration de l'usage du théorème 1. Si g est une série avec terme constant inversible, il est aisé de vérifier que la série $\Phi(X, Y) = g - 1/Y$ vérifie les conditions du théorème avec $\lambda = 1/g(0)$. Il y a donc convergence quadratique de l'itération

$$y_k = y_{k-1} + y_{k-1}(1 - g y_{k-1}) \bmod X^{2^k} \tag{1}$$

vers $1/g$ avec $y_0 = 1/g(0)$.

Il nous reste maintenant à estimer la complexité de cet algorithme.

PROPOSITION 1. *Soit g dans $\mathbb{A}[[X]]$ avec $g(0)$ inversible et $k \geq 0$. On peut calculer l'inverse de g modulo X^{2^k} en $4\mathsf{M}(2^k) + O(2^k)$ opérations dans $\mathbb{A}$.*

Autrement dit, le coût total du calcul coïncide avec celui de la dernière étape, à une constante près.

DÉMONSTRATION. L'itération (1) demande 2 multiplications modulo X^{2^k}, soit $2\mathsf{M}(2^k)$ opérations, plus $C \cdot 2^k$ opérations supplémentaires (additions, ...), C étant une constante que l'on ne va pas déterminer. Notons $\mathsf{N}(k)$ le coût du calcul modulo 2^k ; on a alors, avec $\mathsf{N}(0) = 0$:

$$\mathsf{N}(k) \leq \mathsf{N}(k-1) + 2\mathsf{M}(2^k) + C2^k.$$

L'application du lemme « Diviser pour régner » du Cours 1 donne donc

$$\mathsf{N}(k) \leq 4\mathsf{M}(2^k) + C2^{k+1},$$

en utilisant l'inégalité $\mathsf{M}(d) \leq \frac{1}{2}\mathsf{M}(2d)$.

EXERCICE 3. Effectuer l'application du lemme.

□

REMARQUE. On peut montrer, en se donnant davantage de peine, que le résultat s'étend pour calculer l'inverse de G modulo n'importe quel X^ℓ (ℓ n'étant pas nécessairement une puissance de 2).

Par ailleurs, le coût ci-dessus est largement surestimé.

- En regardant de plus près le calcul de y_k, on constate que le deuxième produit est pris entre y_{k-1} et $1-y_{k-1}g$. Ce dernier terme est de la forme $X^{2^{k-1}}R_{k-1}$, alors que y_{k-1} est de degré au plus 2^{k-1}. Ainsi, seul le produit $y_{k-1}R_{k-1}$ est nécessaire, et celui-ci ne demande que $\mathsf{M}(2^{k-1})$ opérations.
- Le produit $y_{k-1}g$ est de la forme $1+X^{2^{k-1}}R_{k-1}$. Nous sommes donc dans la situation on l'on multiplie un polynôme de degré 2^{k-1} (ici, y_{k-1}) par un polynôme de degré 2^k (ici, g), et où l'on connaît les 2^{k-1} premiers coefficients du produit. Dans cette situation, il est possible de descendre le coût du produit de $\mathsf{M}(2^k)$ à $\mathsf{M}(2^{k-1})$, en utilisant des techniques non triviales de « transposition » de code [4] qui seront abordées au Cours 8.

Au total, on peut donc gagner un facteur 2 sur l'estimation du théorème précédent (et cela se répercute sur le coût de la division euclidienne ci-dessous).

3.2. Logarithme. Pour calculer la série $\log f$ avec $f(0)=1$, il suffit d'utiliser l'identité

$$\log f = \int \frac{f'}{f}.$$

Le calcul demande une division de séries, une dérivation et une intégration, mais ces deux dernières opérations sont linéaires. Le bilan est donc une complexité en $O(\mathsf{M}(N))$.

3.3. Exponentielle. Pour calculer la série $\exp(f)$ avec $f(0)=0$, l'idée est d'appliquer une méthode de Newton avec

$$\Phi(Y) = f - \log Y.$$

EXERCICE 4. Montrer que les conditions du Théorème 1 sont vérifiées.

Il s'ensuit une convergence quadratique vers $\exp(f)$ pour l'itération

$$y_{k+1} = y_k + y_k(f - \log y_k).$$

Chaque itération demande donc le calcul d'une multiplication et d'un logarithme, d'où la complexité en $O(\mathsf{M}(N))$.

3.4. Puissance. La série f^α se déduit des précédentes par

$$f^\alpha = \exp(\alpha \log f).$$

Une application remarquable de cette trivialité apparaît dans la section suivante sur l'inverse compositionnel.

EXERCICE 5. Donner une itération de Newton qui calcule directement la racine carrée, sans passer par l'exponentielle et le logarithme. Plus difficile, estimer le gain par rapport à l'algorithme général de calcul de puissance.

EXERCICE 6. Donner des bornes sur les constantes dans les $O(\cdot)$ pour le logarithme, l'exponentielle et la puissance.

3.5. Sommes de Newton. Une conséquence intéressante de l'efficacité du calcul de l'exponentielle est l'utilisation efficace des sommes de Newton comme structure de données.

Si $P \in \mathbb{K}[X]$ est un polynôme de degré d, avec $\mathbb{K}$ est un corps, alors P a d racines $\alpha_1, \dots, \alpha_d$ dans la clôture algébrique $\overline{\mathbb{K}}$ de $\mathbb{K}$. Les *sommes de Newton* de P sont les sommes $p_i = \alpha_1^i + \cdots + \alpha_d^i \in \mathbb{K}$. Leur utilisation efficace repose sur la proposition suivante.

PROPOSITION 2. *Les sommes de Newton $p_1, \dots, p_d$ d'un polynôme $P \in \mathbb{K}[X]$ de degré d peuvent être calculées en $O(\mathsf{M}(d))$ opérations dans $\mathbb{K}$. Lorsque la caractéristique de $\mathbb{K}$ est 0 ou supérieure à d, la conversion inverse est également possible en $O(\mathsf{M}(d))$ opérations.*

DÉMONSTRATION. La décomposition en éléments simples

$$S = \frac{XP'}{P} = \sum_{P(\alpha)=0} \frac{X}{X-\alpha} = \sum_{\substack{P(\alpha)=0 \\ i \geq 0}} \alpha^i X^{-i}$$

montre que les coefficients du développement en puissances de X^{-1} de XP'/P sont les p_i. Le calcul des d premiers termes de cette série demande $O(\mathsf{M}(d))$ opérations. L'opération inverse, à savoir le calcul de P à partir des d premiers termes de la série $S = XP'/P$ est effectué en $O(\mathsf{M}(d))$ opérations par la formule

$$P = \exp \int S/X$$

si $P(0) \neq 0$. Sinon, il faut d'abord récupérer la valuation, et le reste se calcule de la façon précédente. La contrainte sur la caractéristique permet de définir l'exponentielle tronquée par son développement en série. □

3.6. ⋆ Application à la somme et au produit composés ⋆. On suppose dans cette section que $\mathbb{K}$ est un corps de caractéristique nulle ou supérieure à d^2.

THÉORÈME 3. *Soient P et Q deux polynômes unitaires de $\mathbb{K}[X]$, de degré d. Alors, les polynômes*

$$P \oplus Q = \prod_{\substack{P(\alpha)=0 \\ Q(\beta)=0}} X - (\alpha + \beta), \qquad P \otimes Q = \prod_{\substack{P(\alpha)=0 \\ Q(\beta)=0}} X - (\alpha\beta)$$

peuvent être calculés en $O(\mathsf{M}(d^2))$ opérations.

Comme ces polynômes sont de degré d^2, il s'agit là encore d'une complexité quasi-optimale. Dans cet énoncé, les polynômes sont définis à l'aide des racines de P et Q dans une clôture algébrique, mais leurs coefficients sont dans $\mathbb{K}$. Il est également possible de les définir sur un anneau quelconque comme des *résultants* bivariés. Les résultants et leur calcul seront vus au cours 10. Dans le cas bivarié général, il n'existe pas encore d'algorithme quasi-optimal et le résultat ci-dessus est l'un des rares cas particuliers qui se traite efficacement.

Ces polynômes $P \oplus Q$ et $P \otimes Q$ sont très utiles en pratique. Les polynômes constituent une structure de données bien commode pour coder leurs racines et calculer avec. Ces polynômes $P \oplus Q$ et $P \otimes Q$ fournissent l'addition et la multiplication avec cette structure de données.

DÉMONSTRATION. Le produit de sommes de Newton est la somme de Newton du produit :

$$\sum_{\alpha} \alpha^i \sum_{\beta} \beta^i = \sum_{\alpha,\beta} (\alpha\beta)^i.$$

Il suffit donc de multiplier *terme à terme* les séries XP'/P et XQ'/Q pour obtenir la série génératrice des sommes de Newton de $P \otimes Q$. Si l'on a calculé d^2 termes de ces séries, le résultant, dont le degré est d^2, est alors reconstruit par une exponentielle en $O(\mathsf{M}(d^2))$ opérations.

Diviser le i^{e} coefficient de XP'/P par $i!$, pour tout $i \geq 0$, produit la série

$$\sum_{\substack{P(\alpha)=0 \\ i \geq 0}} \frac{\alpha^i X^{-i}}{i!} = \sum_{P(\alpha)=0} \exp(\alpha/X).$$

En effectuant la même opération sur Q et en effectuant le produit des séries, on obtient donc

$$\sum_{P(\alpha)=0} \exp(\alpha/X) \sum_{Q(\beta)=0} \exp(\beta/X) = \sum_{\substack{P(\alpha)=0 \\ Q(\beta)=0}} \exp((\alpha+\beta)/X).$$

Il suffit alors de remultiplier le i^{e} coefficient par $i!$ pour obtenir la série génératrice des sommes de Newton de $P \otimes Q$, qui est reconstruit par une exponentielle. L'ensemble des opérations est borné par $O(\mathsf{M}(d^2))$. □

3.7. ⋆ Inverse compositionnel ⋆. Étant donnée une série f avec $f(0) = 0$ et $f'(0)$ inversible, il s'agit ici de calculer une série $f^{(-1)}$ telle que $f(f^{(-1)}(X)) = f^{(-1)}(f(X)) = X$.

3.7.1. *Les N premiers termes.* Les conditions d'application de la méthode de Newton sont vérifiées pour la fonction

$$\Phi(X, Y) = f(Y) - X,$$

avec $\lambda = 1/f'(0)$.

EXERCICE 7. Vérifier les conditions.

L'itération correspondante est donc

$$y_{k+1} = y_k - \frac{f(y_k) - X}{f'(y_k)} \bmod X^{2^{k+1}}.$$

En dérivant $f(Y) - X = 0$, on voit que cette itération se simplifie en

$$y_{k+1} = y_k - y_k'(f(y_k) - X) \bmod X^{2^{k+1}}.$$

Le coût est alors dominé soit par celui du produit, soit plus généralement par celui de la composition par f.

EXERCICE 8. La k^{e} racine positive de l'équation $x = \tan x$ admet un développement asymptotique de la forme

$$r_k = (2k+1)\frac{\pi}{2} + \frac{a_1}{k} + \frac{a_2}{k^2} + \dots.$$

Pour tout i, le coefficient a_i est un polynôme de $1/\pi\mathbb{Q}[1/\pi^2]$ de degré i. Borner le nombre d'opérations dans $\mathbb{Q}$ nécessaires pour calculer $a_1, \dots, a_N$, pour N grand.

3.7.2. *Le N^e terme.* Bien qu'en général le coût de l'inverse fonctionnel soit dominé par celui de la composition et donc en $O(\sqrt{N\log N}\mathsf{M}(N))$ (voir la section suivante), il est possible d'obtenir le N^{e} terme sans calculer les précédents pour $O(\mathsf{M}(N))$ opérations. L'idée repose sur la *formule d'inversion de Lagrange* :

$$[X^N]f^{(-1)}(X) = \frac{1}{N}[X^{N-1}]\frac{1}{(f(X)/X)^N},$$

où la notation $[X^k]f$ représente le coefficient de X^k dans la série f. Ainsi, le calcul par cette formule ne requiert qu'une puissance et donc peut être effectué en $O(\mathsf{M}(N))$ opérations.

3.7.3. *Exemple.* La série génératrice des arbres généraux étiquetés (appelés aussi *arbres de Cayley*) vérifie l'équation

$$y = X\exp(y).$$

Son développement à l'origine est obtenu en $O(\mathsf{M}(N))$ opérations dans $\mathbb{Q}$ en calculant d'abord celui de $y\exp(-y)$ par les méthodes de la section précédente, puis en inversant ce développement par la méthode présentée ci-dessus.

Pour déterminer le comportement asymptotique des coefficients ainsi calculés lorsque N tend vers l'infini (c'est-à-dire l'asymptotique du nombre d'arbres de taille N), une méthode présentée dans le cours d'analyse d'algorithmes passe par le calcul du développement de y au voisinage du point singulier $\exp(-1)$, où y vaut 1. En posant $u = \sqrt{2}\sqrt{1-eX}$, on voit qu'il s'agit alors de calculer le développement de y solution de

$$\sqrt{2-2y\exp(1-y)} = u,$$

au voisinage de $y = 1$ et $u = 0$, équation à laquelle la méthode présentée dans cette section s'applique pour donner le résultat en $O(\mathsf{M}(N))$ opérations.

Remarque. La formule d'inversion de Lagrange donne immédiatement une jolie expression pour les coefficients de y solution de $y = X\exp(y)$ de l'exemple précédent. L'asymptotique mentionné plus haut donne alors la formule de Stirling efficacement.

4. La composition des séries

Si $f(X)$ et $g(X)$ sont des séries, avec $g(0) = 0$, il s'agit de calculer efficacement la série $f(g(X))$. Il s'avère que les algorithmes connus n'atteignent pas une complexité quasi-optimale (c'est-à-dire en $O(N)$ à des facteurs logarithmiques près).

4.1. Méthode naïve. La composition peut être calculée par la méthode de Horner. Si $f(X) = \sum_{i\geq 0} a_i X^i + O(X^N)$, le calcul utilise alors

$$f(g(X)) = a_0 + g(a_1 + g(a_2 + \dots)) + O(X^N).$$

Exercice 9. Montrer que la complexité du calcul par cette formule est $O(N\mathsf{M}(N))$.

4.2. ⋆ Pas de bébés—pas de géants ⋆. Une technique dite *pas de bébés, pas de géants* réduit ce coût à $O(\sqrt{N}(\mathsf{M}(N) + N^{\omega/2}))$, où ω est l'exposant de la complexité du produit de matrices, sur lequel reviendra le Cours 9. La série f est d'abord écrite

$$f(X) = f_0(X) + X^k f_1(X) + \cdots + X^{\lceil N/k\rceil} f_{\lceil N/k\rceil}(X) + O(X^N),$$

où les polynômes f_i en nombre $1+\lceil N/k\rceil$ comportent chacun k termes. Ensuite, on calcule les puissances $g^2,\dots,g^k$ de g en $k\mathsf{M}(N)$ opérations. Soient $f_{i,j}$ les coefficients des f_i et $g_{i,j}$ ceux des g^i. Le développement

$$f_i(g)=\sum_{j=0}^{N-1}\left(\sum_{\ell=0}^{k-1}f_{i,\ell}g_{\ell,j}\right)x^j+O(x^N)$$

montre que les coefficients des $f_i(g)$ sont les coefficients du produit de la matrice $(f_{i,j})$ de taille $(\lceil N/k\rceil+1)\times k$ par la matrice $(g_{i,j})$ de taille $k\times N$. En découpant chacune de ces matrices en blocs de taille $k\times k$, ce produit est effectué en au plus $O(N^2k^{\omega-3})$ opérations. Ensuite, il ne reste plus qu'à effectuer $\lceil N/k\rceil$ produits à précision N suivis d'autant d'additions pour un coût total de $O(N\mathsf{M}(N)/k)$. Le choix $k=\lfloor\sqrt{N}\rfloor$ mène à la complexité annoncée.

4.3. L'algorithme de Brent & Kung. Bien qu'il n'y ait pas d'algorithme permettant d'abaisser la complexité au même ordre que $\mathsf{M}(N)$, il est possible de faire sensiblement mieux que la méthode naïve grâce à un algorithme sophistiqué dû à Brent & Kung, détaillé dans cette section.

THÉORÈME 4. *Si $2,3,\dots,\lceil\sqrt{N\log N}\rceil$ sont inversibles dans l'anneau $\mathbb{A}$, la série $f(g(X))$ peut être calculée en $O(\sqrt{N\log N}\mathsf{M}(N))$ opérations arithmétiques, ou en $O(\sqrt{N}\mathsf{M}(N))$ opérations arithmétiques si $\log\mathsf{M}(N)/\log N\to\gamma>1$.*

L'idée de départ de l'algorithme permettant d'aboutir à cette complexité est d'utiliser la formule de développement de Taylor, sous la forme

$$f(g_1+X^mg_2)=f(g_1)+f'(g_1)X^mg_2+f''(g_1)\frac{X^{2m}g_2^2}{2!}+\cdots, \tag{2}$$

où $g=g_1+X^mg_2$ et g_1 est un polynôme de degré inférieur à m.

Le premier gain de complexité par rapport à la méthode naïve provient de l'observation suivante.

LEMME 1. *Étant données les séries g et $f\circ g$, avec $g'(0)$ inversible, la série $f'\circ g$ peut être calculée en $O(\mathsf{M}(N))$ opérations arithmétiques.*

DÉMONSTRATION. La dérivation de séries a une complexité linéaire, et la division est en $O(\mathsf{M}(N))$. Le résultat provient alors de la formule de dérivation d'une composée :

$$f'\circ g=\frac{(f\circ g)'}{g'}.$$

L'hypothèse sur $g'(0)$ permet de garantir l'inversion de la série du dénominateur. Elle est un peu plus forte que nécessaire : il suffit que le premier coefficient non nul de la série g' soit inversible. □

L'utilisation répétée de cette idée dans la formule (2) mène à une complexité

$$C(f(g_1))+\frac{N}{m}O(\mathsf{M}(N))\text{ opérations}$$

pour l'ensemble de la composition, où $C(f(g_1))$ représente la complexité du calcul de la première composition, donnée par le lemme suivant.

LEMME 2. *Soient f et g deux* polynômes *de degrés k et m, avec $g(0) = 0$. Le calcul des N premiers coefficients de la* série *$f \circ g$ peut être effectué en $O(km \log N \mathsf{M}(N)/N)$ opérations arithmétiques, ou en $O(km\mathsf{M}(N)/N)$ opérations si $\log \mathsf{M}(N)/\log N \to \gamma > 1$.*

DÉMONSTRATION. [du théorème à l'aide de ce lemme] Ce lemme donne $C(f(g_1)) = O(m\mathsf{M}(N)\log N)$. La complexité totale est donc bornée par

$$O(m\mathsf{M}(N)\log N) + \frac{N}{m}O(\mathsf{M}(N)) = O\left(\mathsf{M}(N)(m\log N + \frac{N}{m})\right).$$

Cette dernière somme est minimisée par le choix de $m = \sqrt{N/\log N}$ qui donne le résultat du théorème.

Le cas sans le facteur $\log N$ est laissé en exercice. □

DÉMONSTRATION. [du lemme] Sans perte de généralité, on peut supposer que le degré k de f est une puissance de 2. Il suffit de considérer sinon que les coefficients de f pour les degrés allant de $k+1$ jusqu'à la puissance de 2 immédiatement supérieure sont nuls, et la perte de complexité est d'au plus un facteur 2, absorbé dans la constante du $O(\cdot)$.

L'idée est d'effectuer ce calcul par « diviser pour régner » en récrivant le polynôme f sous la forme

$$f = f_1 + X^{k/2} f_2,$$

où f_1 et f_2 sont deux polynômes de degré au plus $k/2$. Dans la composition

$$f(g) = f_1(g) + g^{k/2} f_2(g),$$

les deux polynômes du second sommant ont degré au plus $km/2$. La complexité C_k^m découlant de l'utilisation de cette formule vérifie donc

$$C_k^m \leq 2C_{k/2}^m + 2\mathsf{M}(\min(N, km/2)) + O(N).$$

C'est dans cette prise de minimum que réside toute la subtilité : tant que k est grand, les calculs sont tronqués à l'ordre N, et dès que k devient suffisamment petit, on exploite l'arithmétique polynomiale. Il vient donc

$$\begin{aligned} C_k^m \leq & 2\mathsf{M}(N) + 4\mathsf{M}(N) + \cdots + 2^{\ell-1}\mathsf{M}(N) \\ & + 2^\ell \left(\mathsf{M}\left(\frac{km}{2^\ell}\right) + 2\mathsf{M}\left(\frac{km}{2^{\ell+1}}\right) + \cdots\right), \end{aligned}$$

où ℓ est déterminé par

$$\frac{km}{2^\ell} < N \leq \frac{km}{2^{\ell-1}}.$$

Cette valeur de ℓ, combinée à l'utilisation du lemme « Diviser pour régner » conclut la preuve du lemme. □

Notes

L'essentiel de ce cours provient de l'article exceptionnel [3] où ces techniques ont été étudiées du point de vue de la complexité pour la première fois. L'application aux sommes et produits composées est beaucoup plus récente et se trouve dans [2]. En ce qui concerne la composition, il est possible de descendre à une complexité quasi-optimale dans le cas où la caractéristique est finie [1]. La méthode de Newton permet également la résolution d'équations ou de systèmes différentiels. Ceci fera l'objet du cours 7. En pratique, les constantes cachées dans les $O(\cdot)$ jouent un grand rôle, et

nécessitent généralement l'implantation à la fois des algorithmes asymptotiquement meilleurs et des algorithmes plus classiques, souvent plus efficaces en petite taille.

Bibliographie

[1] Bernstein (Daniel J.). – Composing power series over a finite ring in essentially linear time. *Journal of Symbolic Computation*, vol. 26, n° 3, 1998, pp. 339–341.

[2] Bostan (Alin), Flajolet (Philippe), Salvy (Bruno), and Schost (Éric). – Fast computation of special resultants. *Journal of Symbolic Computation*, vol. 41, n° 1, January 2006, pp. 1–29.

[3] Brent (R. P.) and Kung (H. T.). – Fast algorithms for manipulating formal power series. *Journal of the ACM*, vol. 25, n° 4, 1978, pp. 581–595.

[4] Hanrot (Guillaume), Quercia (Michel), and Zimmermann (Paul). – The middle product algorithm I. *Applicable Algebra in Engineering, Communication and Computing*, vol. 14, n° 6, March 2004, pp. 415–438.

[5] Karp (A. H.) and Markstein (P.). – High-precision division and square root. *ACM Transactions on Mathematical Software*, vol. 23, n° 4, 1997, pp. 561–589.