Extraits de Algorithmes Efficaces en Calcul Formel

Marie

16 février 2023

1 Multiplication rapide

1.1 Résultats principaux

Théorème 1. *La multiplication des polynômes de degré au plus n dans* $\mathbb{A}[X]$ *requiert :*
- *O(n) opérations dans* $\mathbb{A}$ *par l'algorithme naïf ;*
- *O($n^{1,59}$) opérations dans* $\mathbb{A}$ *par l'algorithme de Karatsuba ;*
- *O(nlognloglogn) - voire dans certains cas O(nlogn) - opérations dans* $\mathbb{A}$ *par la transformée de Fourier rapide (FFT) ;*

Remarque. Les constantes cachées dans les $O(\cdot)$ sont déterminantes pour l'efficacité pratique de tels algorithmes. Par exemple, lorsque $\mathbb{A}$ est un corps fini de taille « raisonnable » :
- l'algorithme de Karatsuba bat l'algorithme naïf pour des degrés d'environ 20 ;
- les méthodes à base de FFT en $O(n\log n)$ battent l'algorithme de Karatsuba pour des degrés de l'ordre de 100, mais ne peuvent pas être utilisées pour des degrés arbitrairement grands ;
- l'algorithme de type FFT en $O(n\log n\log\log n)$ est utilisé pour des degrés de l'ordre de quelques dizaines ou centaines de milliers.

Ces algorithmes ne sont pas concurrents, mais complémentaires.

Notations. Soient des polynômes F et G à coefficients dans un anneau $\mathbb{A}$, ayant un degré au plus n-1, formellement

$$F = f_0 + ... + f_{n-1}X^{n-1} \text{ et } G = g_0 + ... + g_{n-1}X^{n-1}$$

le problème est alors de calculer les coefficients de

$$H = FG = h_0 + ... + h_{2n-2}X^{2n-2}$$

1.2 Algorithme naïf

L'algorithme naïf consiste à développer le produit, c'est à dire à écrire

$$H = FG = \sum_{i=0}^{2n-2} h_i X^i \text{ avec } h_i = \sum_{j+k=i} f_j g_k$$

Ainsi, calculer tous les h_i demande $O(n^2)$ opérations dans $\mathbb{A}$. C'est un algorithme de complexité *quadratique*.

Pour multiplier deux polynômes de degrés m et n, l'algorithme naïf demande au plus $(m+1)\times(n+1)$ multiplications dans $\mathbb{A}$ et mn additions dans $\mathbb{A}$

1.3 Algorithme de Karatsuba

Il est possible de gagner une multiplication pour le produit des polynômes de degré 1, parmi les 4 du produit par l'algorithme quadratique.
Soient à multiplier les polynômes $F=f_0 + f_1X$ et $G=g_0 + g_1X$. Leur produit

$$H = FG = h_0 + h_1X + h_2X^2$$

peut être obtenu par une forme d'interpolation sur les points 0, 1, ∞ :

— $H(0) = h_0 = f_0 g_0$;
— le coefficient de plus haut degré (qui correspond à l'évaluation en l'infini) $h_2 = f_1 g_1$;
— $H(1) = h_0 + h_1 + h_2 = F(1)G(1) = (f_0 + f_1)(g_0 + g_1)$.

Ainsi on obtient $h_1 = (f_0 + f_1)(g_0 + g_1) - h_0 - h_2$ pour seulement une multiplication supplémentaire, donc l'ensemble des coefficients du produit pour 3 multiplications et 4 additions. Quelques additions sont perdues par rapport à l'algorithme naïf, mais le gain d'une multiplication va se transformer en gain dans l'*exposant* de l'algorithme, par application récursive.

En effet, dans le cas général des degrés quelconques, il suffit de scinder F et G en deux et de procéder de la même manière. Si F et G sont de degré au plus $n-1$, avec k=$\lceil n/2 \rceil$, on pose

$$F = F^{(0)} + F^{(1)}X^k \text{ et } G = G^{(0)} + G^{(1)}X^k$$

pour des polynomes $F^{(0)}, F^{(1)}, G^{(0)}, G^{(1)}$ de degrés au plus k-1.
Le produit H=FG s'écrit

$$H = F^{(0)}G^{(0)} + (F^{(0)}G^{(1)} + F^{(1)}G^{(0)})X^k + F^{(1)}G^{(1)}X^{2k}$$

Entrée F, G de degrés au plus n-1
Sortie H = FG

1. Si $n = 1$, renvoyer FG
2. Décomposer F et G selon
$$F = F^{(0)} + F^{(1)}X^k \text{ et } G = G^{(0)} + G^{(1)}X^k$$
3. Calculer $A_1 = F^{(0)}G^{(0)}$ et $A_2 = F^{(1)}G^{(1)}$ récursivement
4. Calculer $A_3 = F^{(0)} + F^{(1)}$ et $A_4 = G^{(0)} + G^{(1)}$
5. Calculer $A_5 = A_3A_4$ récursivement
6. Calculer $A_6 = A_5 - A_1$ et $A_7 = A_6 - A_2$
7. Renvoyer $A_1 + A_7X^k + A_2X^{2k}$

Théorème 2 1. *si n est une puissance de 2, l'algorithme de Karatsuba calcule le produit de deux polynômes de degrés au plus n-1 en au plus* $9n^{log_2 3}$ *opérations dans* $\mathbb{A}$

Corrolaire. *On peut multiplier deux polynômes de degré n arbitraire en* $O(n^{log_2 3})$=$O(n^{1,59})$ *opérations dans* $\mathbb{A}$

1.4 Transformée de Fourier rapide

Pour simplifier la présentation, on suppose ici que l'on cherche à multiplier des polynômes F et G dans $\mathbb{A}[X]$, de degrés strictement inférieurs à $n/2$ (ou plus généralement tels que deg(FG)<n).

Idée de l'algorithme
En supposant que l'anneau $\mathbb{A}$ le permette, l'idée générale est d'évaluer en des points bien choisis, de multiplier les évaluations, et de reconstruire les coefficients du produit à partir de ces valeurs. Si deux polynômes coïncident sur $1, \omega, ..., \omega^{n-1}$, nous verrons que leur différence est un multiple de $X^n - 1$.

Lorsque l'algorithme est employé avec l'hypothèse deg$H<n$, les coefficients de H mod $X^n - 1$ qui sont renvoyés sont bien ceux de H. Le coût des étapes de précalcul et de produit point à point est linéaire en n, et il reste à voir comment effectuer rapidement les opérations d'évaluation et d'interpolation.

Entrée F et G deux polynômes, n un entier, et ω une racine principale n-ième de l'unité.
Sortie $H = FG \bmod X^n - 1$.

1. *Précalcul :* Calculer les puissances $\omega^2, ..., \omega^{n-1}$.
2. *Evaluation :* Calculer les valeurs :

$$Ev(F) = (F(\omega^0), ..., F(\omega^{n-1}))$$

$$Ev(G) = (G(\omega^0), ..., G(\omega^{n-1}))$$

3. *Produit point à point :*

$$(Ev(F), Ev(G)) \mapsto (FG(\omega^0), ..., FG(\omega^{n-1}))$$

4. *Interpolation :*

$$Ev(FG) \mapsto FG$$

Définition. L'élément ω de $\mathbb{A}$ est une *racine n-ième de l'unité* si $\omega^n = 1$;
- C'est une racine n-ième *primitive* de l'unité si de plus $\omega^t \neq 1$ pour $t \in \{1, ..., n-1\}$
- C'est une racine n-ième *principale* de l'unité si de plus $\omega^t - 1$ est non diviseur de zéro dans $\mathbb{A}$ pour $t \in \{1, ..., n-1\}$ $(\alpha(\omega^t - 1) = 0 \Rightarrow \alpha = 0)$

Lemme 1. *Si ω est racine primitive ou principale n-ième de l'unité, alors*

1. *ω^{-1} aussi ;*
2. *si $n = pq$ alors ω^p est une racine q-ième de l'unité de même nature que ω ;*
3. *pour $\ell \in \{1, ..., n-1\}$ et ω une racine principale de l'unité alors*

$$\sum_{j=0}^{n-1} \omega^{\ell j} = 0$$

Transformée de Fourier rapide
L'opération

$$DFT : F \in \mathbb{A}[X] \mapsto (F(1), F(\omega), ..., F(\omega^{n-1}))$$

où ω est une racine principale n-ième de l'unité, s'appelle la *transformée de Fourier discrète*. Son calcul rapide est effectué par un algorithme de type « diviser pour régner ».

Pour appliquer cette idée, supposons que n est pair (i.e : $n = 2k$). Alors, $\omega^k = -1$ puisque

$$(\omega^k - 1)(\omega^k + 1) = \omega^n - 1 = 0$$

et le premier facteur n'est pas diviseur de 0. Le polynôme F est décomposé par division euclidienne de deux façons :

$$F = Q_0(X^k - 1) + R_0 \text{ et } F = Q_1(X^k + 1) + R_1$$

avec $\deg R_0 < k$ et $\deg R_1 < k$. Ces décompositions vont nous permettre le calcul de F sur les puissances paires et impaires de ω.

En effet, si ℓ est pair, $\omega^{k\ell} = 1$ et donc $F(\omega^\ell) = R_0(\omega^\ell)$. De même, si ℓ est impair, $F(\omega^\ell) = R_1(\omega^\ell)$. Outre l'application récursive, le point crucial qui est la source de l'efficacité de l'algorithme FFT et qui conduit au choix de racines primitives de l'unité, est que le calcul de R_0 et R_1 est très simple (étape 2). Lors des appels récursifs, les puissances de ω qui sont utilisées sont des ω^{2^i} qui sont bien des racines primitives d'après le Lemme.

Entrée $\mathrm{F} = f_0 + ... + f_{n-1}\mathrm{X}^{n-1}$, les puissances $1, \omega, ..., \omega^{n-1}$ d'une racine n-ième principale de l'unité, n étant une puissance de 2.
Sortie $\mathrm{F}(1), ..., \mathrm{F}(\omega^{n-1})$.

1. Si $n = 1$, renvoyer f_0.
2. Sinon, soit $k = n/2$. Calculer

3. Calculer récursivement

4. Renvoyer

Théorème 3. *L'algorithme de FFT requiert au plus* $\frac{3n}{2}logn$ *opérations dans* $\mathbb{A}$. *Les multiplications font toutes intervenir une puissance de* ω

L'algorithme de FFT requiert $nlogn$ additions dans $\mathbb{A}$, $\frac{1}{2}nlogn$ multiplications d'éléments de $\mathbb{A}$ par des puissances de ω, mais aucune autre multiplication dans $\mathbb{A}$.

Remarque. La transformée de Fourier discrète est un morphisme d'algèbres sur $\mathbb{A}$ de $\mathbb{A}[\mathrm{X}]/(\mathrm{X}^n - 1)$ dans $\mathbb{A}^n$ avec comme multiplication dans $\mathbb{A}^n$ la multiplication coordonée par coordonée. Cette observation permet d'économiser des transformées inverses en effectuant plusieurs calculs directement sur les transformées. Une application typique de cette observation est le produit scalaire, ou plus généralement le produit de matrices.

Interpolation

En termes matriciels, l'opération $\mathrm{F} \mapsto \mathrm{Ev}(\mathrm{F})$ est linéaire et sa matrice (pour des polynômes F de degré au plus n-1, dans la base monomiale 1, X,..., X^n) est la matrice de Vandermonde

$$\mathrm{V}_\omega = \begin{pmatrix} 1 & 1 & ... & 1 \\ 1 & \omega & ... & \omega^{n-1} \\ \vdots & & & \vdots \\ 1 & \omega^{n-1} & ... & \omega^{(n-1)^2} \end{pmatrix}$$

Lemme 2. *Si* $\omega \in \mathbb{A}$ *est une racine n-ième principale de l'unité, alors* $\mathrm{V}_\omega^{-1}\mathrm{V}_\omega = n\mathrm{I}_n$

Autrement dit, l'interpolation sur les puissances de ω est calculée efficacement en la ramenant à une FFT sur les puissances de ω^{-1}, qui est bien principale d'après le Lemme 1.

Conclusion

Théorème 4. *Si 2 est inversible dans* $\mathbb{A}$ *et n une puissance de 2, étant donnée une racine principale n-ième dans* $\mathbb{A}$, *le produit de deux polynômes dont la somme des degrés est inférieure à n peut être calculé en* $\frac{9}{2}nlogn + O(n)$ *opérations dans* $\mathbb{A}$. *Seuls n des produits sont entre deux éléments de* $\mathbb{A}$ *qui ne sont pas des puissances de* ω.

1.5 L'algorithme de Schönage-Strassen