COMPLEXITÉ DES OPÉRATIONS ARITHMÉTIQUES

RÉSUMÉ ET QUESTIONS

On a vu que l'addition des entiers a une complexité linéaire. La multiplication a une complexité quasi-linéaire en théorie. On s'intéresse ici aux autres opérations arithmétiques classiques : celles sur les polynômes et sur les matrices.

1. Polynômes

Soit A un anneau. La somme de deux polynômes $f(x)$ et $g(x)$ de degrés $\leq d$ se calcule au prix de $d+1$ additions dans A au plus. Le produit $f(x)g(x)$ nécessite $(d+1)(d+1)$ multiplications dans A au plus. Le nombre d'additions est majoré par la même quantité. On voit que dans ce contexte, il est légitime de prendre pour opérations élémentaires les opérations dans A.

À titre d'entraînement, on peut calculer le pgcd de $f(x) = x^4 + x^2 + x + 1$ et $g(x) = x^3 + 1$ dans $\mathbb{F}_2[x]$.

Dans le cadre des polynômes, le principe de la multiplication rapide est plus facile à présenter. Karatsuba a remarqué que le produit de $f(x) = f_0 + f_1 X$ et $g(x) = g_0 + g_1 x$ s'écrit

$$f(x)g(x) = f_0 g_0 + (f_1 g_0 + f_0 g_1)x + f_1 g_1 x^2.$$

Le calcul direct requiert quatre multiplications et une addition. On note que

$$f_1 g_0 + f_0 g_1 = (f_0 + f_1)(g_0 + g_1) - f_0 g_0 - f_1 g_1,$$

ce qui permet de calculer $f(x)g(x)$ au prix de trois multiplications, deux additions, et deux soustractions.

L'utilisation récursive de cette méthode permet de multiplier deux polynômes de degré d au prix de $d^{\log_2 3 + o(1)}$ opérations dans A au lieu de $O(d^2)$. L'algorithme de Karatsuba est la première étape vers des méthodes quasi-linéaires. On pourra l'implémenter à titre d'exercice.

2. Matrices

Soit A un anneau. Soient m, n, p trois entiers positifs. Ajouter deux matrices $m \times n$ à coefficients dans A requiert mn additions dans A. Le produit d'une matrice $m \times n$ par une matrice $n \times p$ avec la méthode standard requiert mnp multiplications dans A et un nombre comparable d'additions.

Strassen a découvert un analogue pour les matrices de la méthode de Karatsuba. Pour multiplier deux matrices 2×2 on peut utiliser les formules ci-dessous. Le coût est alors de 7 multiplications et 18 additions ou soustractions dans A. L'utilisation récursive de cette idée (multiplication par blocs) permet de multiplier deux matrices $d \times d$ au prix de $d^{\log_2^7 + o(1)}$ opérations dans A au lieu de $O(d^3)$. On ne sait pas s'il existe un algorithme en $d^{2+o(1)}$ pour ce problème.

$$a = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}, b = \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix}, c = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}$$

$$\begin{aligned}
m_1 &:= (a_{1,1} + a_{2,2})(b_{1,1} + b_{2,2}) \\
m_2 &:= (a_{2,1} + a_{2,2})b_{1,1} \\
m_3 &:= a_{1,1}(b_{1,2} - b_{2,2}) \\
m_4 &:= a_{2,2}(b_{2,1} - b_{1,1}) \\
m_5 &:= (a_{1,1} + a_{1,2})b_{2,2} \\
m_6 &:= (a_{2,1} - a_{1,1})(b_{1,1} + b_{1,2}) \\
m_7 &:= (a_{1,2} - a_{2,2})(b_{2,1} + b_{2,2}) \\
c_{1,1} &= m_1 + m_4 - m_5 + m_7 \\
c_{1,2} &= m_3 + m_5 \\
c_{2,1} &= m_2 + m_4 \\
c_{2,2} &= m_1 - m_2 + m_3 + m_6
\end{aligned}$$

Le calcul de l'inverse d'une matrice $d \times d$ à coefficients dans un corps se fait en temps $O(d^3)$ avec l'algorithme dit du pivot de Gauss. On peut ainsi résoudre les systèmes linéaires à coefficients dans un corps. On peut aussi calculer un déterminant de cette manière.

Les équations linéaires à coefficients dans un anneau ne sont pas si faciles à résoudre. À titre d'exercice on pourra résoudre les équations et systèmes suivants.

- $3x - 1 = 5$ avec x dans $\mathbb{Z}$.
- $3x + 2y = 1$ avec x et y dans $\mathbb{Z}$.
- $x + 3y + 5z = 0$ et $-7x + 4y + 2z = 0$ avec x, y et z dans $\mathbb{Z}$.

On pourra utiliser les fonctions matkerint et mathnf de pari.

```
gp > M=[1,3,5;-7,4,2]
%1 =
[1 3 5]
[-7 4 2]
gp > mathnf(M~,1)
%2 = [[-37, -22; 14, 9; 0, 1], [-2, -1; 5, 3]]
%[1]~
%3 =
[-37 14 0]
[-22 9 1]
gp > matkerint(M)
%4 =
[-14]
[-37]
```

[25]