Université de Bordeaux
Mathématiques

Préparation à l'agrégation
2013–2014

Mémento complexité

Rappel. on écrit

- $f = O(g)$ s'il existe $C > 0$ et x_0 tel que $f(x) \leqslant Cg(x)$ pour tout $x > x_0$.
- $f = \widetilde{O}(g)$ s'il existe $C > 0$ tel que $f \leqslant g(\log g)^C$.

1. Exponentiation binaire

Le calcul de x^n s'effectue en utilisant $O(\log n)$ multiplications.

2. Dans $\mathbb{Z}$

La *taille* (binaire) d'un entier a est le nombre de bits[1] utilisé pour l'écrire, i.e $t(a) := \lfloor \log_2(a+1) \rfloor + 1$. On suppose que tous les opérandes sont de taille $\leqslant n$:

Opération	Naïf	Rapide
$a + b$	$O(n)$	$O(n)$
$a \times b$	$O(n^2)$	$\mathsf{M}_{\mathbb{Z}}(n) = \widetilde{O}(n)$
$a = bq + r$	$O(n^2)$	$O(\mathsf{M}_{\mathbb{Z}}(n))$
Relation de Bézout	$O(n^2)$	$O(\mathsf{M}_{\mathbb{Z}}(n) \log n)$
Restes chinois	$O(n^2)$	$O(\mathsf{M}_{\mathbb{Z}}(n) \log n)$

Pour tous ces algorithmes, on compte le nombre d'opérations élémentaires sur les bits, supposé proportionnel au temps d'exécution (complexité en *temps*). La complexité en espace est de $\widetilde{O}(n)$ bits.

- Multiplication : $\mathsf{M}_{\mathbb{Z}}(n)$ est le temps d'une multiplication dans $\mathbb{Z}$ pour deux opérandes de taille $\leqslant n$. L'algorithme rapide (Schönhage-Strassen) utilise la Transformée de Fourier en temps $O(n \log n \log\log n)$.

  Moins rapide mais beaucoup plus simple : l'algorithme de Karatsuba utilise $O(n^{\log_2 3})$ opérations.

  Remarque. Pour des opérantes de tailles différentes, $s(a) \leqslant n$, $s(b) \leqslant m$, l'algorithme naïf (quadratique) utilise $O((n+1)(m+1))$ opérations.
- Division euclidienne : l'entrée est (a, b), $b \neq 0$, et la sortie (q, r) avec $0 \leqslant r < |b|$. L'algorithme rapide résout l'équation $b - a/x = 0$ en utilisant l'équation de Newton

  $$x_{n+1} = x_n - x_n(x_n b - a).$$

  (Calcul effectué en doublant la précision à chaque étape ; soit dans un modèle flottant, soit modulo 2^k.) On pose $q = \lfloor x \rfloor$, puis $r = a - bq$.

[1] Il n'y a pas d'inconvénient à compter les chiffres en base $\beta \geqslant 2$, avec $t(a) := \lfloor \log_\beta(a+1) \rfloor$. Pas vraiment d'intérêt théorique non plus.

L'énoncé de complexité suppose que $\mathsf{M}_{\mathbb{Z}}(n)$ vérifie des propriétés du type $M(n)/n \geqslant M(m)/m$ pour $n \geqslant m$, et $M(mn) \leqslant m^2 M(n)$. Qui sont vraies aussi bien pour l'algorithme naïf que pour Karatsuba ou Schönhage-Strassen.

Remarque. Si $t(a) \leqslant n$, $t(b) \leqslant m$, $n \geqslant m$, l'algorithme naïf (quadratique) utilise $O((n-m+1)(m+1))$ opérations élémentaires.

- Bézout : l'entrée est (a, b) et la sortie consiste en $\operatorname{pgcd}(a, b)$ et deux entiers u, v tels que $au + bv = \gcd(a, b)$.
- Restes chinois : l'entrée consiste en N congruences $x \equiv a_k \pmod{b_k}$ où les b_k sont 2 à 2 premiers entre eux, et la sortie attendue est une solution de ce système de congruences. On suppose $t(a_k) \leqslant t(b_k)$ and $\sum_k t(b_k) \leqslant n$.

3. Dans $\mathbb{Z}/N\mathbb{Z}$

On choisit un représentant canonique dans chaque classe de congruence : les entiers de $[0, N-1]$, pour représenter les éléments de $\mathbb{Z}/N\mathbb{Z}$. La taille des opérandes est $\leqslant n := t(N)$.

L'addition est implantée comme une addition dans $\mathbb{Z}$, suivie d'une (possible) soustraction : temps $O(n)$. La multiplication est une multiplication in $\mathbb{Z}$, suivie par une division euclidienne par N : temps $O(n^2)$ ou $O(\mathsf{M}_{\mathbb{Z}}(n))$. L'inversion utilise une relation de Bézout : temps $O(n^2)$ ou $O(\mathsf{M}_{\mathbb{Z}}(n) \log n)$.

4. Dans $K[X]$, K un corps

La *taille* (algébrique) d'un polynône $h \in K[X]$ est son nombre de coefficients : $t(h) := \deg h + 1 \leqslant n$.

Pour la complexité, on choisit de compter les opérations dans K (complexité algébrique) : $+$, $-$, $\times$, $/$ dans K. Pour estimer plus réalistement le temps de calcul, on peut multiplier par le coût d'une opération élémentaire dans K quand celui-ci est borné uniformément (K corps fini) ; sinon il faut tenir compte de la taille des coefficients.

Soient f, g deux polynômes dans $K[X]$, de tailles $\leqslant n$.

Opération	Naïf	Rapide
$f + g$	$O(n)$ $[+]$	$O(n)$ $[+]$
$f \times g$	$O(n^2)$ $[+, \times]$	$\mathsf{M}_{K[X]}(n) = \widetilde{O}(n)$
$f = gh + r$	$O(n^2)$	$O(\mathsf{M}_{K[X]}(n))$
Bézout	$O(n^2)$	$O(\mathsf{M}_{K[X]}(n) \log n)$
Restes chinois	$O(n^2)$	$O(\mathsf{M}_{K[X]}(n) \log n)$

5. Dans $K[X]/(T)$:

Deux cas particuliers importants : définitions des corps finis non premiers $\mathbb{F}_q/\mathbb{F}_p$, et des extensions de $\mathbb{Q}$ (nombres algébriques). De façon analogue à $\mathbb{Z}/N\mathbb{Z}$

nous travaillons avec des polynômes de taille $\leqslant t(T)$, avec les coûts ci-dessus. L'inversion utilise toujours une relation de Bézout.

6. Dans $M_{n\times n}(K)$:

On se place de nouveau dans le modèle de complexité algébrique : on compte les opérations K. Pour $A \in M_{n\times n}(K)$, sa *taille* est $t(A) = n^2$; pour $v \in K^n$ sa taille est $t(v) := n$

Opérations	Naïf	Rapide
$A + B$	$O(n^2)$	$O(n^2)$
Av, $v \in K^n$	$O(n^2)$	$O(n^2)$
$A \times B$	$O(n^3)$	$O(n^\omega), \omega = 2.3729$
A^{-1}	$O(n^3)$	$O(n^\omega)$
$PA = LU$	$O(n^3)$	$O(n^\omega)$
Char A	$O(n^3)$	$O(n^\omega)$

La factorisation LU (pivot de Gauss) permet de résoudre la plupart des problèmes d'algèbre linéaire sur K : noyau, image, rang, déterminant, inverse, système linéaire (LU + résolution de systèmes triangulaires en $O(n^2)$ opérations supplémentaires dans K)...

La constante $\omega \leqslant 3$ qui apparaît ci-dessus est appelé *exposant de la multiplication matricielle*. La valeur utilisée dans les implantations (pour des matrices physiquement représentables par les ordinateurs actuels) est $\omega = \log_2 7 \approx 2.8$ (Strassen).

Algèbre linéaire « Boîte Noire ». Dans ce modèle, les coûts sont exprimés en terme du nombre d'évaluations $v \mapsto Av$ pour un "opérateur boîte noire" A. (Un opérateur linéaire dont on ne suppose pas qu'il est donné par une matrice. On sait simplement calculer l'image de tout vecteur $v \in K^n$.) En général, une multiplication matrice-vecteur pour $A \in M_{n\times n}(K)$ et $v \in K^n$ utilise $O(n^2)$ opérations dans K mais la plupart des matrices rencontrées en pratique sont « structurées » et permettent une évaluation plus économique : matrices par bandes, plus généralement matrices creuses (comme dans l'algorithme de Dixon de factorisation dans $\mathbb{Z}$) ; matrice de Sylvester définissant le résultant ; matrice de Berlekamp (pour factoriser les polynômes sur les corps finis) ; matrices de van der Monde (comme les matrices de FFT), etc.

Dans ce modèle il existe des algorithmes *probabilistes* (Wiedemann) qui résolvent efficacement des problèmes d'algèbre linéaire comme le calcul du polynôme caractéristique, de A ou du rang de A (avec faible probabilité d'erreur dans les 2 cas), ou un système linéaire $Ax = b$ (on veut *une* solution). Ils utilisent $O(n)$ evaluations Av, plus $O(n^2)$ opérations dans K en moyenne. On ne gagne rien sur une matrice dense générale (Av utilise $O(n^2)$ opérations dans K) ; mais le gain est significatif si Av se calcule plus rapidement.