Université de Bordeaux
Mathématiques

Préparation à l'agrégation
Année 2013–2014

Algorithme d'Euclide

1. Rappels

Soit A un anneau euclidien. C'est donc que A est commutatif unitaire, intègre, et qu'il existe une application $\varphi\colon A \to \mathbb{N}$ appelée stathme euclidien, telle que

(1) $\varphi(a) = 0$ si et seulement si $a = 0$
(2) Si a et b sont des éléments de A, avec $b \neq 0$, il existe des éléments r et q de A tels que $a = bq + r$ et $\varphi(r) < \varphi(b)$. Les éléments r et q sont respectivement appelés reste et quotient de la division de a par b ; on n'impose pas qu'ils soient uniques.

Par exemple, si $A = \mathbb{Z}$, on peut prendre $\varphi(n) = |n|$. Ici, il n'y a pas unicité de l'écriture $a = bq + r$, où $\varphi(r) < \varphi(b)$; on impose traditionnellement $0 \leqslant r < b$ ce qui garantit l'unicité, mais c'est un choix arbitraire : on pourait tout aussi bien imposer $-b/2 < r \leqslant b/2$ par exemple. Si $A = K[X]$, où K est un corps, on peut choisir $\varphi(P) = \deg P + 1$ si $P \neq 0$ et $\varphi(0) = 0$. Enfin, pour $A = \mathbb{Z}[i]$, on peut prendre comme stathme $\varphi(a + ib) = a^2 + b^2$.

Tout anneau euclidien est principal, et tout anneau principal est factoriel (tout élément non nul se décompose en produit fini d'irréductibles et cette décomposition est unique, à multiplication des facteurs par des inversibles près). Si A est un anneau principal, soient a et b deux éléments de A tels que $(a, b) \neq (0, 0)$, il existe d dans A tel que

(1) $d \mid a$ et $d \mid b$
(2) Si $e \mid a$ et $e \mid b$, alors $e \mid d$.

Cet élément d, unique à multiplication près par un élément inversible de A, est appelé le pgcd de a et b, et noté $\operatorname{pgcd}(a, b)$, ou parfois (a, b) si le contexte ne permet pas d'ambiguïté avec l'élément (a, b) de A^2. On a alors $aA + bA = dA$. En particulier, il existe u et v dans A tels que $au + bv = d$ (identité de Bézout). Ceci donne une définition légèrement plus générale : on appelle pgcd de (a, b) tout générateur de l'idéal $aA + bA$ de A engendré par a et b ; cette définition est équivalente à la précédente quand $(a, b) \neq 0$ et définit $\operatorname{pgcd}(0, 0) = 0$.

Une dernière définition, encore plus générale puisqu'elle est valable si l'anneau A est factoriel. Si A est factoriel, $a \in A \setminus \{0\}$, et p irréductible dans A, on définit $v_p(a)$ la plus grande puissance de p qui divise a : si $v = v_p(a)$, on a $p^v \mid a$ mais $p^{v+1} \nmid a$. On définit $\operatorname{pgcd}(0, b) = b$ puis, pour tout $b \in A$ et pour a, b tous deux non nuls, on pose

$$\operatorname{pgcd}(a, b) = \prod_p p^{\min(v_p(a), v_p(b))}$$

où p parcourt un système de représentants des éléments irréductibles de A modulo les unités. Cette définition généralise les deux précédentes et permet de calculer le pgcd d quand la factorisation dans A est effective.

Si A est de plus euclidien, l'algorithme d'Euclide permet aussi ce calcul, en général à bien moindre coût que celui d'une factorisation. L'algorithme d'Euclide étendu calcule, en même temps que d, des éléments u et v satisfaisant l'identité de Bézout.

Rappelons finalement que dans un anneau principal A, si a et b sont deux éléments de A, il existe un élément c dans A, unique à multiplication près par un inversible de A, tel que

(1) $a \mid c$ et $b \mid c$
(2) Si $a \mid e$ et $b \mid e$, alors $c \mid e$.

C'est le ppcm de a et b, noté $\mathrm{ppcm}(a, b)$, ou encore $[a, b]$. On a

$$aA + bA = (a, b)A, \quad aA \cap bA = [a, b]A \quad \text{et} \quad abA = (a, b)[a, b]A.$$

2. Algorithme d'Euclide

Algorithme 2.1 (Algorithme d'Euclide)
Entrée : a, b dans A
Sortie : $\mathrm{pgcd}(a, b)$

(1) Si $b = 0$, retourner a.
(2) Soit r le reste de de la division euclidienne de a par b. Poser $a \leftarrow b$, $b \leftarrow r$ et revenir à l'étape (1).

Preuve. Le cas $(a, b) = (0, 0)$ est clair puisque l'algorithme retourne directement 0 à l'étape (1). Le cas général vient du fait que $(a, b) = (a + bk, b)$ pour tout $k \in A$ (exercice). L'algorithme se termine car la valeur de $\varphi(b)$ décroit strictement tant que $b \neq 0$. Comme φ prend ses valeur dans $\mathbb{N}$, on finit par arriver à $\varphi(b) = 0$, donc $b = 0$. □

3. Algorithme d'Euclide étendu

Algorithme 3.1 (Algorithme d'Euclide étendu)
Entrée : a et b dans A, $(a, b) \neq (0, 0)$.
Sortie : Le pgcd d de a et b, et un couple (u, v) de A^2 tel que $au + bv = d$.

(1) Si $a = 0$, sortir $d = b$, $u = 0$, $v = 1$.
(2) $x \leftarrow a$, $y \leftarrow b$, $v_x \leftarrow 0$, $v_y \leftarrow 1$, $u_x \leftarrow 1$, $u_y \leftarrow 0$.
(3) Tant que $y \neq 0$
(4) Calculer le quotient q et le reste r de la division euclidienne de x par y.
(5) $(u_x, u_y) \leftarrow (u_y, u_x - qu_y)$.
(6) $(v_x, v_y) \leftarrow (v_y, v_x - qv_y)$.
(7) $(x, y) \leftarrow (y, r)$.
(8) Sortir $d = x$, $u = u_x$, $v = v_x$.

Preuve. Comme dans l'algorithme précédent, l'algorithme se termine, et l'on calcule bien d un pgcd de a et b.

On pose $x_0 = a$, $v_0 = 0$, $u_0 = 1$, $x_1 = b$, $v_1 = 1$ et $u_1 = 0$. Pour $i \geqslant 1$, on écrit la division euclidienne $x_{i-1} = q_i x_i + x_{i+1}$, où $\varphi(x_{i+1}) < \varphi(x_i)$, et on pose $v_{i+1} = v_{i-1} - q_i v_i$ et $u_{i+1} = u_{i-1} - q_i u_i$. C'est donc que

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \begin{pmatrix} u_{i-1} & v_{i-1} \\ u_i & v_i \end{pmatrix}.$$

Donc si

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} x_{i-1} \\ x_i \end{pmatrix},$$

ce qui est vrai pour $i = 1$, alors

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \begin{pmatrix} x_{i-1} \\ x_i \end{pmatrix} = \begin{pmatrix} x_i \\ x_{i+1} \end{pmatrix}.$$

Soit l l'entier tel que $x_{l+1} = 0$. On a bien alors $u_l a + v_l b = x_l = d$. □

4. Complexité dans $K[X]$

Soit K un corps. Alors $K[X]$ est un anneau euclidien. On veut majorer le nombre d'opérations dans K nécessaire à l'exécution des algorithmes d'Euclide en fonction du degré des polynômes concernés. On garde les notations du paragraphe précédent.

4.1. **Algorithme d'Euclide.** Soient a et b deux polynômes de $K[x]$, de degrés respectifs n et m. On suppose que $m \leqslant n$. Le nombre d'itérations l est borné par $\varphi(b)$, donc ici $m + 1$.

Pour tout i, on note $n_i = \deg x_i$. On peut diviser le polynôme x_{i-1} par x_i en utilisant au plus $(2n_i + 1)(n_{i-1} - n_i + 1)$ additions et multiplications, et une inversion dans K. Donc le coût total est inférieur à

$$C = \sum_{i=1}^{l} (2n_i + 1)(n_{i-1} - n_i + 1)$$

additions et multiplications, plus l inversions dans K. On voit alors que

$$C \leqslant (2m+1) \sum_{i=1}^{l} (n_{i-1} - n_i + 1) \leqslant (2m+1)(l + n - n_l) \leqslant (2m+1)(n+m+1).$$

On a donc prouvé le résultat suivant.

Théorème 4.1. *Soit $N \geqslant 1$. On peut exécuter l'algorithme d'Euclide pour deux polynômes de degré inférieur ou égal à N en $O(N^2)$ opérations sur le corps de base.*

On peut donner une majoration plus précise de C. Pour cela, on évalue C dans le cas où le degré décroît exactement de 1 à chaque pas, et on montre ensuite que c'est le cas le pire.

Dans ce cas, $l = m+1$, $n_0 = n$, $n_1 = m$ et pour i dans $\{2\dots, m+1\}$, $n_i = m-i+1$. On trouve alors que

$$\begin{aligned} C &= (2m+1)(n-m+1) + 2\sum_{i=2}^{m+1} \big(2(m-i+1)+1\big) \\ &= 2mn + n + m + 1. \end{aligned}$$

Pour montrer que c'est le cas le pire, on pose, pour $n_0 \geqslant n_1 > n_2 > \cdots > n_l \geqslant 0$,

$$\sigma(n_0, \dots, n_l) = \sum_{i=1}^{l} (2n_i + 1)(n_{i-1} - n_i + 1).$$

Montrons que σ croît, si l'on insère un entier k entre n_{j-1} et un n_j, ou bien si l'on ajoute k après n_l. Soit donc k un entier tel que $n_{j-1} > k > n_j$.

$$\begin{aligned} &\sigma(n_0, \dots, n_{j-1}, k, n_j, \dots, n_l) - \sigma(n_0, \dots, n_l) \\ &\quad = (2k+1)(n_{j-1} - k + 1) + (2n_j + 1)(k - n_j + 1) - (2n_j + 1)(n_{j-1} - n_j + 1) \\ &\quad = (2k+1)(n_{j-1} - k + 1) + (2n_j + 1)(k - n_{j-1}) \\ &\quad = 2(n_{j-1} - k)(k - n_j) + 2k + 1 > 0 \end{aligned}$$

On montrerait le méme résultat, en ajoutant un entier k après n_l. On en déduit par récurrence que $\sigma(n_0, \dots, n_l) \leqslant \sigma(n_0, n_1, n_1 - 1, n_1 - 2, \dots, 1)$, et donc que le cas où le degré baisse de 1 à chaque pas est le cas le pire.

Théorème 4.2. *L'algorithme d'Euclide, appliqué à des polynômes a et b de degrés respectifs n et m ($n \geqslant m$), utilise au plus $2mn + m + n + 1$ additions et multiplications, plus $m + 1$ inversions dans K.*

4.2. **Algorithme d'Euclide étendu.** Pour évaluer la complexité de l'algorithme d'Euclide étendu, il nous faut majorer le degré des polynômes u_i et v_i.

Proposition 4.3. *On a*

$$\deg v_i = \sum_{1 \leqslant j < i} \deg q_j = n_0 - n_{i-1}, \quad i \in \{1, \dots, l+1\},$$

$$\deg u_i = \sum_{2 \leqslant j < i} \deg q_j = n_1 - n_{i-1}, \quad i \in \{2, \dots, l+1\}.$$

Preuve. Par récurrence. On a $v_0 = 0$, $v_1 = 1$, $v_2 = v_0 - q_1 v_1 = -q_1$. Ainsi, $\deg v_2 = \deg q_1 = n - m = n_0 - n_1$. Soit $i \geqslant 1$. On suppose que pour tout $j \in \{1, \dots, i\}$, on a $\deg v_j = \sum_{k=1}^{j-1} \deg q_k$. Ainsi, $\deg v_{j-1} < \deg v_j$. On en déduit que

$$\deg v_{i+1} = \deg(v_{i-1} - q_i v_i) = \deg q_i + \deg v_i = \sum_{1 \leqslant k \leqslant i} \deg q_i.$$

L'égalité pour le degré de u_i se prouve de la même façon. □

Le calcul de $v_{i+1} = v_{i-1} - q_i v_i$ demande au plus $2 \deg q_i \deg v_i + \deg q_i + \deg v_i + 1$ opérations dans K pour le produit et au plus $\deg v_{i+1}+1$ opérations pour la soustraction, c'est-à-dire en tout $2(n_{i-1} - n_i)(n_0 - n_{i-1} + n_0 - n_i + 1)$, sauf pour $i = 1$: le calcul de $v_2 = -q_1$ demande $n - m + 1$ changements de signe. On obtient que le nombre d'opérations dans K pour calculer les v_i est majoré par

$$\begin{aligned} C_v &= n - m + 1 + 2\sum_{i=2}^{l}((n_{i-1} - n_i)(n_0 - n_{i-1}) + n_0 - n_i + 1) \\ &\leqslant n - m + 1 + 2n\sum_{i=2}^{l}(n_{i-1} - n_i) + 2(l-1)(n+1) \\ &\leqslant n - m + 1 + 2n(n_1 - n_l) + 2(l-1)(n+1) \\ &\leqslant n - m + 1 + 2mn + 2m(n+1) \\ &\leqslant 4mn + n + m + 1. \end{aligned}$$

On montrerait de même que le nombre d'opérations dans K nécessaires au calcul des u_i est inférieur ou égal à $4m^2 + m$.

On peut améliorer ces majorations, en calculant le nombre d'opérations nécessaires au calcul des u_i et des v_i dans le cas où le degré du reste baisse de 1 à chaque étape, puis en montrant que c'est le cas le pire. On obtient alors

$$\begin{aligned} C_v &= n - m + 1 + 2\sum_{i=2}^{m+1}((n - (m - i + 2)) + n - (m - i + 1) + 1)) \\ &= 4mn - 2m^2 + n + m + 1 \\ C_u &= 2 + 2\sum_{i=3}^{m+1}((m - (m - i + 2)) + m - (m - i + 1) + 1) \\ &= 2(m^2 + m). \end{aligned}$$

On peut aussi noter que l'algorithme d'Euclide étendu calcule inutilement la suite des v_i : on peut entièrement supprimer ces calculs, sans affecter le calcul de u et d. Si $b \neq 0$, on peut alors poser $v = (d - au)/b$.

Théorème 4.4. *L'algorithme d'Euclide étendu, appliqué à des polynômes a et b de degrés respectifs n et m ($n \geqslant m$), utilise $6mn + O(n)$ additions et multiplications, plus $m + 1$ inversions dans K.*

5. Complexité dans $\mathbb{Z}$

5.1. **Algorithme d'Euclide.** On applique l'algorithme d'Euclide à deux entiers a et b positifs. Le nombre d'itérations est inférieur ou égal à b, mais on peut trouver une bien meilleure majoration :

Théorème 5.1. *Soit n un entier naturel non nul, et soient a et b deux entiers tels que $a > b$ et tels que l'algorithme d'Euclide appliqué à a et b nécessite exactement n*

divisions, et tels que a soit minimal pour cette propriété. Alors $a = F_{n+1}$ et $b = F_n$, où (F_n) est la suite de Fibonacci définie par $F_0 = 0$, $F_1 = 1$, $F_{i+2} = F_{i+1} + F_i$ pour $i \leqslant 0$.

Preuve. Si on applique l'algorithme d'Euclide à F_{n+1} et F_n, on obtient

$$(F_{n+1}, F_n) = (F_n, F_{n-1}) = \cdots = (F_1, F_0) = 1.$$

On a donc exactement n divisions.

Réciproquement, on suppose que a et b satisfont les conditions de l'énoncé. Soient $x_{n+1} = a$, $x_n = b$, et pour $i < n$, soit x_i le reste de la division de x_{i+2} par x_{i+1}, jusqu'à $x_0 = 0$. Alors pour tout i, $x_{i+1} > x_i$. On note q_i la partie entière de x_{i+1}/x_i. Comme $x_{i+1} = x_{i-1} + q_i x_i \geqslant x_{i-1} + x_i$ pour tout i et comme $x_0 = F_0 = 0$ et $x_1 \geqslant F_1 = 1$, on obtient par récurrence que $x_i \geqslant F_i$ pour tout i. Donc $a \geqslant F_{n+1}$. La minimalité de a montre alors que $a = F_{n+1}$. De plus, $x_{n-1} \geqslant F_{n-1} = F_{n+1} - F_n \geqslant a - b$, puisque $F_n \leqslant b$. Comme en outre $x_{n-1} \leqslant x_{n+1} - x_n = a - b$, les inégalités ci-dessus sont des égalités et $F_n = b$. □

Le cas le pire est donc atteint pour deux éléments consécutifs de la suite de Fibonacci. On peut écrire F_n de façon explicite. Soient

$$\varphi = \frac{1+\sqrt{5}}{2} \quad \text{et} \quad \psi = \frac{1-\sqrt{5}}{2}$$

les racines de $x^2 - x - 1$, alors $F_n\sqrt{5} = \varphi^n - \psi^n$. Ainsi, si l'algorithme d'Euclide appliqué à a et b demande $n-1$ divisions, alors $a\sqrt{5} \geqslant \varphi^n - \psi^n$. Donc $a\sqrt{5} > \varphi^n - 1$ et

$$n < \frac{\log(1 + a\sqrt{5})}{\log \varphi}.$$

On en déduit que le nombre de divisions est majoré par

$$\left\lceil \log_\varphi(1 + a\sqrt{5}) \right\rceil - 2.$$

Donc, si a et b sont inférieurs à N, alors le nombre d'opérations élémentaires dans l'algorithme d'Euclide est en $O((\log N)^3 + 1)$. Mais on peut faire mieux :

Théorème 5.2. *Si a et b sont inférieurs à N, le coût du calcul de (a, b) par l'algorithme d'Euclide est en $O((\log N)^2 + 1)$.*

Preuve. On suppose que $a > b > 0$, et que l'algorithme d'Euclide pour a et b demande exactement n divisions. On note $x_0 = a, x_1 = b, x_2, \ldots x_{n+1} = 0$ la suite des restes successifs, et pour i dans $\{1, \ldots, n\}$, on note q_i le quotient de x_{i-1} par x_i. Donc $x_{i-1} = q_i x_i + x_{i+1}$. Le coût des divisions est inférieur à une constante multipliée par

$$\begin{aligned}\sum_{i=1}^{n} (\log x_i + 1)(\log q_i + 1) &\leqslant (\log a + 1) \sum_{i=1}^{n} (\log q_i + 1) \\ &\leqslant (\log N + 1)(n + \log \prod_{i=1}^{n} q_i).\end{aligned}$$

Comme

$$\prod_{i=1}^{n} q_i \leqslant \prod_{i=1}^{n} \frac{x_{i-1}}{x_i} = \frac{a}{(a,b)} \leqslant N,$$

le coût est bien en $O((\log N)^2 + 1)$. □

5.2. **Coût de l'algorithme d'Euclide étendu.** On reprend les notations du §3. Donc $x_0 = a$, $v_0 = 0$, $u_0 = 1$, $x_1 = b$, $v_1 = 1$ et $u_1 = 0$. Pour $i \geqslant 1$, $x_{i-1} = q_i x_i + x_{i+1}$, où $\varphi(x_{i+1}) < \varphi(x_i)$, et on pose

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \begin{pmatrix} u_{i-1} & v_{i-1} \\ u_i & v_i \end{pmatrix}.$$

Proposition 5.3. *Pour tout i, on a les inégalités $|u_i| \leqslant b/x_{i-1}$ et $|v_i| \leqslant a/x_{i-1}$.*

Preuve. On a

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix},$$

ainsi

$$\det \begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} = (-1)^i$$

et

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix}^{-1} = (-1)^i \begin{pmatrix} v_{i+1} & -v_i \\ -u_{i+1} & u_i \end{pmatrix}.$$

Or tous les

$$\begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix}^{-1} = \begin{pmatrix} q_i & 1 \\ 1 & 0 \end{pmatrix}$$

sont à coefficients positifs, donc leur produit aussi ! Donc

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix}^{-1} = \begin{pmatrix} |v_{i+1}| & |v_i| \\ |u_{i+1}| & |u_i| \end{pmatrix}.$$

Or

$$\begin{pmatrix} u_i & v_i \\ u_{i+1} & v_{i+1} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} x_i \\ x_{i+1} \end{pmatrix},$$

donc

$$\begin{pmatrix} |v_{i+1}| & |v_i| \\ |u_{i+1}| & |u_i| \end{pmatrix} \begin{pmatrix} x_i \\ x_{i+1} \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}.$$

D'où l'on tire les inégalités annoncées. □

Un calcul similaire à celui du paragraphe précédent permet alors de montrer le résultat suivant.

Théorème 5.4. *Si a et b sont inférieurs à N, l'algorithme d'Euclide étendu appliqué à a et b utilise $O((\log N)^2 + 1)$ opérations élémentaires.*