Remarques sur l'algorithme d'Euclide

1. Préliminaires et conventions

Pour $a \in \mathbb{N}^*$, on note

$$\lg(a) := 1 + \log(a) > 0$$

la "taille" de a. Justification : le nombre de chiffres de a, écrit en base B est $\lceil \log_B a \rceil = O_B(\lg(a))$, donc la taille est liée au nombre de chiffres, sans pour autant dépendre de la base de numération, ni de la représentation des entiers. On rajoute 1 pour assurer $\lg(a) > 0$ et donc pouvoir écrire qu'une constante est $O(\lg(a))$. [*Seul* 0 *est* $O(0)$...]. Remarquons aussi que $a \leqslant b$ implique $\lg(a) \leqslant \lg(b)$.

Si $a, b \in \mathbb{N}^*$, alors les algorithmes de l'école primaire (en comptant les opérations élémentaires sur les chiffres) montrent que le calcul de

- $a \pm b$ a un coût $O(\lg(a) + \lg(b))$
- $a \times b$ a un coût $O(\lg(a)\lg(b))$
- (q, r), quotient et reste de la division euclidienne de a par b ($a \geqslant b$), a un coût $O(\lg(a)\lg(q))$ soit $O(\lg(a)(\lg(a) - \lg(b) + 1))$.

Remarque 1.1. Ce sont des majorations. Multiplications et divisions peuvent se faire, asymptotiquement, en temps presque linéaire

$$O(T\log(T)\log\log(T)) \quad \text{si} \quad \lg(a), \lg(b) \leqslant T,$$

cf [1, Chap 4.3]. En utilisant ces primitives à la place des algorithmes quadratiques ci-dessus, on obtient le même coût $O(T\log(T)\log\log(T))$ pour le pgcd étendu (c'est non trivial).

2. Analyse de l'algorithme d'Euclide

Algorithme 2.1
Étant donnés $a, b \in \mathbb{N}$, calcule leur pgcd (a, b).

(1) [*Fini ?*]. Si $b = 0$, renvoyer a.
(2) [*division euclidienne*]. Poser (dans cet ordre) $r := a \bmod b$, $a := b$, $b := r$, et revenir à l'étape 1.

Preuve. Les valeurs successives de b forment une suite décroissante (b_i) dans $\mathbb{N}$. Cette suite est strictement décroissante tant que $b_i \neq 0$. Il existe donc i tel que $b_i = 0$ (sinon on aurait une suite strictement décroissante dans $\mathbb{N}$). Donc l'algorithme termine. On vérifie facilement que la valeur de retour est (a, b). $\square$

Théorème 2.2. *Soit $n \geqslant 1$, et a, b deux entiers $a > b > 0$ tels que l'algorithme d'Euclide appliqué à (a, b) nécessite exactement n divisions, et tels que a soit minimal pour ces conditions. Alors $a = F_{n+1}$, $b = F_n$, où les F_i sont les nombres de Fibonacci définis par la récurrence linéaire $F_{i+2} = F_{i+1} + F_i$ $(i \geqslant 0)$, $F_0 = 0$, $F_1 = 1$.*

Preuve. En appliquant Euclide au couple F_{n+1}, F_n proposé, on obtient successivement

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

soit exactement n divisions. Réciproquement, soit (a, b) satisfaisant les conditions de l'énoncé et notons $x_{n+1} = a$, $x_n = b$, $\ldots$, $x_0 = 0$ la suite des 2 données, suivies des n restes calculés par Euclide. Par récurrence, on a $x_{i+1} > x_i$ pour tout $0 \leqslant i \leqslant n$. Soit $q_i := \lfloor x_{i+1}/x_i \rfloor \geqslant 1$ le quotient de la division euclidienne de x_{i+1} par x_i ; on a

$$x_{i-1} = (x_{i+1} \bmod x_i) = x_{i+1} - q_i x_i \leqslant x_{i+1} - x_i$$

On en déduit que $x_i \geqslant F_i$ pour tout $i \geqslant 0$ par récurrence. Comme a est minimal, on en déduit $a = F_{n+1}$, puis que les quotients successifs sont tous égaux à 1. Ainsi $x_i = F_i$, et en particulier $b = F_n$. □

Lemme 2.3. *Soit F_n le n-ème nombre de Fibonacci, défini ci-dessus, et*

$$\varphi = (1 + \sqrt{5})/2, \quad \psi = (1 - \sqrt{5})/2$$

les solutions de l'équation $x^2 = x + 1$. On a

$$F_n = \frac{\varphi^n - \psi^n}{\sqrt{5}}.$$

Preuve. L'équation caractéristique de la récurrence $x^2 = x + 1$ admet pour solution le nombre d'or $\varphi = (1 + \sqrt{5})/2$ et son conjugué. Donc, il existe des constantes universelles a et b telles que $F_n = a\varphi^n + b\psi^n$. En posant $F_0 = 0$, $F_1 = 1$, on trouve $a = -b = 1/(\varphi - \psi) = 1/\sqrt{5}$. □

Corollaire 2.4. *Si $a > b > 0$, le nombre d'étapes d'Euclide appliqué à (a, b) est majoré par*

$$\left\lfloor \log(1 + \sqrt{5}a)/\log\varphi \right\rfloor - 1 = O(\lg(a)).$$

Preuve. Si on a $n - 1$ divisions, on obtient $a \geqslant F_n = (\varphi^n - \psi^n)/\sqrt{5}$, soit $\sqrt{5}a > \varphi^n - 1$ (puisque $|\psi| < 1$) et $n < \log(1 + \sqrt{5}a)/\log\varphi$ □

Corollaire 2.5. *Si $a, b \leqslant N$, le coût du calcul de (a, b) est* $O(\lg(N)^3)$

En fait, on peut facilement faire mieux :

Lemme 2.6. *Si $a, b \leqslant N$, le coût du calcul de (a, b) est* $O(\lg(N)^2)$

Preuve. On peut supposer que $a > b > 0$. Supposons, qu'on ait exactement n divisions ; on sait que $n = O(\lg(N))$. On note $x_{n+1} = a$, $x_n = b$, …, $x_0 = 0$, la suite des restes successifs, puis pour $i = 1, \dots, n$, $q_i := \lfloor x_{i+1}/x_i \rfloor$ la suite des quotients successifs correspondants ; autrement dit, $x_{i+1} = q_i x_i + x_{i-1}$, pour $1 \leqslant i \leqslant n$. Le coût des divisions est dominé par

$$\sum_{i=1}^{n} \lg(x_{i+1}) \lg(q_i) \leqslant \lg(a) \sum_{i=1}^{n} (1 + \log(q_i)) \leqslant \lg(N)(n + \log \prod_{i=1}^{n} q_i)$$

et on remarque ensuite que $\prod q_i \leqslant \prod (x_{i+1}/x_i) = a/(a,b) \leqslant N$. Soit finalement un coût $O(\lg(N)^2)$. □

3. Pgcd étendu

Algorithme 3.1
Étant donnés $a, b \in \mathbb{N}$, calcule $\delta = (a,b)$, et $u, v \in \mathbb{Z}$ tels que $au + bv = \delta$.

(1) Si $a = 0$, renvoyer $\delta = b$, $u = 0$, $v = 1$.
(2) On pose $x = a$, $y = b$, $v_x = 0$, $v_y = 1$.
(3) Tant que $y \neq 0$ [*ici* $v_x b \equiv x \pmod a$ *et* $v_y b \equiv y \pmod a$]
  (a) trouver q, r réalisant la division euclidienne $x = qy + r$.
  (b) remplacer (v_x, v_y) par $(v_y, v_x - qv_y)$.
  (c) remplacer (x, y) par $(y, x - qy)$ [*c'est-à-dire* (y, r)]
(4) Renvoyer $\delta = x$, $u = (x - bv_x)/a$, et $v = v_x$.

Preuve. (x, y) prennent les mêmes valeurs que dans l'algorithme d'Euclide, donc l'algorithme étendu termine après le même nombre de boucles. La démonstration de l'assertion du (3) est immédiate par récurrence. On en déduit qu'en (4), $x \equiv bv_x \pmod a$, donc $u \in \mathbb{Z}$. Le résultat suit. □

Théorème 3.2. *Soient a, b deux entiers non tout deux nuls. Tout au long de l'algorithme d'Euclide étendu, on a $|v_x|, |v_y| \leqslant a/(a,b)$. En particulier les coefficients de Bezout $ua + vb = \delta$ obtenus satisfont $|u| \leqslant |b|/\delta$ et $|v| \leqslant |a|/\delta$.*

Preuve. On peut supposer $a \geqslant b > 0$. On définit $(x_i)_{0 \leqslant i \leqslant n+1}$ et $(q_i)_{1 \leqslant i \leqslant n}$ comme précédemment, ainsi que les suites définies par récurrence descendante :

- $v_{n+1} = 0$, $v_n = 1$, $v_{i-1} = v_{i+1} - q_i v_i$, $1 \leqslant i \leqslant n$. [*correspond à* v_x, v_y].
- $u_{n+1} = 1$, $u_n = 1$, $u_{i-1} = u_{i+1} - q_i v_i$, $1 \leqslant i \leqslant n$. [*On pourrait calculer* u_x, u_y *dans l'algorithme, et remplacer* $u = (x - bv_x)/a$ *par* $u = u_v$.]

Alors pour tout $0 \leqslant i \leqslant n$

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

et on peut voir l'algorithme d'Euclide étendu comme une suite d'opérations élémentaires sur les lignes de ce système ; donc, pour tout $0 \leqslant i \leqslant n$, on a

$$\begin{pmatrix} u_{i+1} & v_{i+1} \\ u_i & v_i \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_{i+1} \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -q_n \end{pmatrix} \begin{pmatrix} u_{n+1} & v_{n+1} \\ u_n & v_n \end{pmatrix},$$

où la dernière matrice est l'identité. On en déduit que $\det\left(\begin{smallmatrix} u_{i+1} & v_{i+1} \\ u_i & v_i \end{smallmatrix}\right) = (-1)^{n-i}$ et donc

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

Mais cet inverse est aussi le produit des

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

dont tous les coefficients sont positifs, donc tous ses coefficients sont positifs ! On en déduit

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

Donc $|v_i|\, x_{i+1} + |v_{i+1}|\, x_i = a$, et en particulier $|v_i| \leqslant a/x_{i+1} \leqslant a/(a, b) \leqslant a$. (Rapelons que, lorsque i décroit de $n + 1$ vers 0, x_i est strictement décroissante avec $x_1 = (a, b)$ et $x_0 = 0$.)

□

Corollaire 3.3. *Si $a, b \leqslant N$, le coût d'obtention de (a, b) et des coefficients de Bezout est $O(\lg(N)^2)$.*

Preuve. Le coût est dominé par

$$\lg(N)^2 \;\; [\textit{divisions} + \textit{soustractions}] + \sum_{i=1}^{n} \lg(v_i) \lg(q_i) \;\; [\textit{multiplications}] = O(\lg(N)^2)$$

en majorant $\lg(v_i) \leqslant \lg(a) \leqslant \lg(N)$, et $\sum \lg(q_i)$ comme dans la section précédente. On a majoré le coût des soustractions par $n \cdot O(\lg(N)) = O(\lg(N)^2)$. □

Références

[1] D. E. Knuth, *The art of computer programming. Vol. 2*, second ed., Addison-Wesley Publishing Co., Reading, Mass., 1981, Seminumerical algorithms, Addison-Wesley Series in Computer Science and Information Processing.