Niveau : 1 **Jaune**

Complexité de l'algorithme d'Euclide pour le calcul du pgcd

1 Introduction

Le calcul du pgcd par l'algorithme d'Euclide, avec éventuellement le calcul des coefficients de Bezout (notamment pour le calcul de l'inverse modulaire), est très souvent utilisé en cryptographie. Cet algorithme aboutit pour des nombres de taille cryptographique. Nous allons étudier plus précisément sa complexité en fonction d'opérations élémentaires. Cette étude s'appuie essentiellement sur la suite de Fibonnacci.

2 La suite de Fibonnacci

Nous allons rappeler succintement les résultats principaux concernant cette suite. Pour une étude plus détaillée, nous renvoyons le lecteur à l'annexe B de l'ouvrage « Cryptographie, Principes et Mises en œuvre » de P. Barthélemy, R. Rolland et P. Véron.

La suite de Fibonnacci est définie par ses deux premiers termes :

$$F_0 = 0, F_1 = 1$$

et par la relation de récurrence linéaire définie pour $n \geq 1$:

$$F_{n+1} = F_n + F_{n-1}.$$

Remarquons que cette suite est positive et croissante strictement, ce qui permet d'interpréter la formule précédente comme la division euclidienne de F_{n+1} par F_n. Cette simple remarque nous montre que quand on prend deux termes consécutifs de la suite de Fibonnacci, et qu'on effectue sur eux l'algorithme d'Euclide, les quotients successifs sont toujours 1 et que l'algorithme ne s'achève que lorsqu'on a atteint F_1. On ne s'étonnera pas, que vis à vis du nombre de pas à exécuter, ce cas soit un mauvais cas, et comme on peut le montrer (théorème de Lamé), le pire des cas.

Notons ϕ le nombre d'or :

$$\phi = \frac{1+\sqrt{5}}{2}.$$

On montre (résolution classique d'une récurrence linéaire) que :

$$F_n = \frac{1}{\sqrt{5}}\left(\phi^n - (-\phi)^{-n}\right),$$

ou encore :

$$F_n = \text{round}\left(\frac{1}{\sqrt{5}}\phi^n\right).$$

3 Complexité de l'algorithme d'Euclide

Le théorème principal qui permet de mesurer la complexité de l'algorithme d'Euclide est le théorème de Lamé :

Théorème 3.1 *Si le calcul du pgcd d des nombres $a \geq b$ par l'algorithme d'Euclide demande n pas, alors :*

$$a \geq dF_{n+2} \text{ et } b \geq dF_{n+1}.$$

En outre si $a = F_{n+2}$ et $b = F_{n+1}$ il y a exactement n pas de calcul.

Autrement dit, le cas d'un couple de termes consécutifs de la suite de Fibonnacci représente le « pire cas » : ce sont les nombres minimaux qui exigent n pas pour l'algorithme d'Euclide.

À partir de la valeur :

$$F_{n+1} = \frac{1}{\sqrt{5}}\left(\phi^{n+1} + (-1)^n \left(\frac{1}{\phi}\right)^{n+1}\right),$$

on obtient successivement :

$$F_{n+1} = \frac{1}{\sqrt{5}}\phi^{n+1}\left(1 + (-1)^n \left(\frac{1}{\phi}\right)^{2(n+1)}\right),$$

$$F_{n+1} \geq \frac{1}{\sqrt{5}}\phi^{n+1}\left(1 - \left(\frac{1}{\phi}\right)^{2(n+1)}\right).$$

Donc :

$$\ln(F_{n+1}) \geq -\ln(\sqrt{5}) + (n+1)\ln(\phi) + \ln\left(1 - \left(\frac{1}{\phi}\right)^{2(n+1)}\right),$$

ce qui donne :

$$n \leq \frac{\ln(F_{n+1})}{\ln(\phi)} + \frac{\ln(\sqrt{5})}{\ln(\phi)} - 1 - \frac{\ln\left(1 - \left(\frac{1}{\phi}\right)^{2(n+1)}\right)}{\ln(\phi)}.$$

En conséquence, appliqué à $a > b > 0$ l'algorithme d'Euclide (ansi que l'algorithme d'Euclide étendu) exécute n boucles, avec :

$$n \leq \frac{\ln(b)}{\ln(\phi)} + \frac{\ln(\sqrt{5})}{\ln(\phi)} - 1 - \frac{\ln\left(1 - \left(\frac{1}{\phi}\right)^{2(n+1)}\right)}{\ln(\phi)}.$$

Comme $n \geq 1$, on peut aussi écrire la majoration :

$$n \leq \frac{\ln(b)}{\ln(\phi)} + \frac{\ln(\sqrt{5})}{\ln(\phi)} - 1 - \frac{\ln\left(1 - \left(\frac{1}{\phi}\right)^{4}\right)}{\ln(\phi)}.$$

Un calcul numérique nous donne donc :

$$n \leq 2.0781\ \ln(b) + 1.$$

Auteur : Ainigmatias Cruptos
Diffusé par l'Association ACrypTA