Décomposition en facteurs premiers

L2 Informatique - UFR S.A.T

Pr. Ousmane THIARE

ousmane.thiare@ugb.edu.sn
http://www.ousmanethiare.com

10 mai 2024

Divisions successives

Algorithme de Monte-Carlo (1975)

Algorithme du crible quadratique QS de Pomerance

Algorithme (p-1) de Pollard

Algorithme de Lenstra (courbes elliptiques)

Décomposition en facteurs premiers

Chapitre V : Décomposition en facteurs premiers

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

1 Divisions successives

2 Algorithme de Monte-Carlo (1975)

3 Algorithme du crible quadratique QS de Pomerance

4 Algorithme (p-1) de Pollard

5 Algorithme de Lenstra (courbes elliptiques)

Décomposition en facteurs premiers
Divisions successives

Divisions successives

Algorithme de Monte-Carlo (1975)

Algorithme du crible quadratique QS de Pomerance

Algorithme (p-1) de Pollard

Algorithme de Lenstra (courbes elliptiques)

L'algorithme est très simple : tenter de diviser le nombre par les nombres premiers successifs, dont on dispose dans un tableau.
Cet algorithme n'est pas efficace, mais il est nécessaire d'en disposer : tous les autres algorithmes, conçus pour trouver de « grands » diviseurs, connaissent des cas d'échec, qui sont d'autant plus fréquents que les diviseurs sont petits.
Avant d'envoyer un nombre à un autre algorithme, il est donc indispensable de l'avoir débarrassé de ses « petits » diviseurs. Pour fixer les idées, il s'agit des nombres premiers représentables sur 16 bits, jusqu'à 65 535 ($=2^{16}-1$) (le plus grand est 65 521, et il y en a au total 6 542).

Décomposition en facteurs premiers
Divisions successives

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Cette première phase de la décomposition nécessite en général un temps si faible qu'il n'est pas mesurable.

Remarque : On pourrait alors envisager aussi d'aller plus loin, c'est-à-dire, par exemple, de tenter la division par tous les nombres premiers représentables sur 32 bits (c'est-à-dire inférieurs à 2^{32}=4 294 967 296 : 10 chiffres « seulement » !).

Il faut alors savoir qu'il y en a 203 280 221 et que la manière la plus économique de les stocker nécessite environ 194 Mo...

On n'ose évoquer le temps d'exécution d'un algorithme qui parcourrait un tel tableau. . . pour ne même pas obtenir de résultat, ce qui est le cas dès que le plus petit diviseur du nombre à décomposer possède plus de 10 chiffres 'ce qui est fort peu pour des nombres qui dépassent les 100 chiffres décimaux).

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Présentation

Cet algorithme, dont l'efficacité est tout-à-fait surprenante, utilise un générateur de nombres au hasard (c'est de l'intervention de ce « hasard » que l'algorithme tire son nom). Soit

- f cette fonction (le générateur),
- A la valeur d'initialisation,
- n le nombre à décomposer,
- p un de ses facteurs premiers.

On considère les suites de nombres entiers définies par

$$x_0 = y_0 = A, x_{m+1} = f(x_m)[n], y_{m+1} = f(y_m)[p]$$

Remarque : On ne connaît pas p, bien sûr, mais on sait que $y_m = x_m[p]$, et cela suffit.

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Soit h la plus grande puissance de 2 qui est inférieure ou égale à m (par exemple, pour m=50, h=32). On peut alors montrer qu'il existe un entier m tel que $y_m = y_{h-1}$, c'est-à-dire $x_m - x_{h-1} \equiv 0[p]$.
C'est donc un multiple de p, qu'on pourra obtenir en calculant le PGCD de ce nombre avec n.

L'algorithme :

Initialisations : n au nombre à factoriser
x à 5, x' à 2, k à 1,h à 1, g à 1.

Divisions successives

Algorithme de Monte-Carlo (1975)

Algorithme du crible quadratique QS de Pomerance

Algorithme (p-1) de Pollard

Algorithme de Lenstra (courbes elliptiques)

TANT QUE (n n'est pas premier) ET QUE ($g \neq n$)
FAIRE
REPETER $g \leftarrow (x - x') \wedge n$
SI (g est différent de 1) ALORS
SI (g est différent de n) ALORS
IMPRIMER g
IMPRIMER "est un diviseur de"
IMPRIMER n
$n \leftarrow \frac{n}{g}$
$x \leftarrow x\%n$
$x' \leftarrow x'\%n$
FINSI

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

SINON
$k \leftarrow k - 1$
SI (k=0) ALORS
$x' \leftarrow x$
$h \leftarrow 2h$
$k \leftarrow h$
FINSI
$x \leftarrow (x^2 + 1)\%n$
FINSI
JUSQU'A (g est différent de 1)
FAIT
FIN

On notera que l'algorithme ainsi décrit, si l'on ne se trouve pas dans le cas d'erreur, « tourne » tant qu'il n'a pas terminé la décomposition du nombre, ce qui peut durer très longtemps... Il faut, bien sûr, en plus, prévoir un arrêt au bout d'un certain temps

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

On notera que l'algorithme ainsi décrit, si l'on ne se trouve pas dans le cas d'erreur, "tourne" tant qu'il n'a pas terminé la décomposition du nombre, ce qui peut durer très longtemps... Il faut, bien sûr, en plus, prévoir un arrêt au bout d'un certain temps.

Discussion :

Même si, quelquefois, cet algorithme permet la factorisation de nombres plus grands, il ne peut pas prétendre arriver à décomposer tous les nombres de 20 chiffres ou moins.

Cette méthode est idéale pour les calculettes programmables.

Divisions successives

Algorithme de Monte-Carlo (1975)

Algorithme du crible quadratique QS de Pomerance

Algorithme (p-1) de Pollard

Algorithme de Lenstra (courbes elliptiques)

L'idée, dans cet algorithme comme dans de nombreux autres, et d'obtenir, si possible, des congruences de la forme $x^2 \equiv y^2[n]$, x n'étant ni congru à y, ni à -y. Dans ce cas, $(x - y) \wedge n$ sera un diviseur non trivial de n.
Ce qui distingue ces méthodes entre elles est la manière d'obtenir ces résidus quadratiques modulo n.
Ici, on prend les valeurs sur les entiers du polynôme $P(x)(x + E(\sqrt{n}))^2 - n$. Ces valeurs fournissent des congruences de la forme $y^2 \equiv r[n]$, où r est le résidu quadratique.
On peut repérer plus facilement ceux qui se factorisent aisément (par la méthode précédente, donc qui sont assez petits) en criblant les valeurs de P(x) pour obtenir la congruence recherchée $x^2 \equiv y^2[n]$ de manière efficace.

Divisions successives

Algorithme de Monte-Carlo (1975)

Algorithme du crible quadratique QS de Pomerance

Algorithme (p-1) de Pollard

Algorithme de Lenstra (courbes elliptiques)

L'intérêt de cette méthode est qu'elle donne ses meilleurs résultats sur les nombres qui font échouer la suivante, mais elle est très difficile à programmer : le criblage n'est pas évident, et il y a énormément de « petites astuces », qui ne peuvent être examinées ici, pour retrouver les congruences recherchées aussi rapidement que possible. C'est la méthode la plus utilisée avec celle, plus récente, des courbes elliptiques. On peut espérer décomposer des nombres jusqu'à 70 chiffres à peu près dans des temps raisonnables (on veut dire : quelques jours. . .).

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Soit p un diviseur premier du nombre n à décomposer.
Si a est premier avec p, $a^p \equiv a[p]$ (théorème de Fermat).
Comme a est premier avec p, il est inversible modulo p, donc $ap - 1 \equiv 1[p]$, soit $(a^{p-1} - 1) \equiv 0[p]$, ou encore $(a^{p-1} - 1) = kp$.
Donc $(a^{p-1} - 1) \wedge n \neq 1$: ce PGCD est donc un diviseur de n.
Le cas d'échec est celui où k=0, on trouve alors que le PGCD est n, et on n'obtient aucun renseignement sur un éventuel diviseur de n.
Par ailleurs, évidemment, on ne peut pas calculer a^{p-1} quand on ne connaît pas p.

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Il suffit en fait d'utiliser un multiple quelconque de p-1, soit h(p-1).
On aura aussi $a^{h(p-1)} \equiv 1[p]$, il faudra donc utiliser comme exposant un nombre qui comporte le plus possible de facteurs premiers distincts, de manière à ce que les diviseurs de p-1 figurent tous dans la liste (c'est-à-dire que cet exposant soit de la forme h(p-1))
Concrètement, on opère étape par étape, en utilisant le PPCM des entiers depuis 1 jusqu'à un maximum fixé.

Décomposition en facteurs premiers
Algorithme (p-1) de Pollard

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Exemple : Soit à décomposer le nombre n=R_7=1111111= 239x4 649.

- On doit choisir a, premier avec p, sans connaître p. C'est facile, il suffit de choisir un nombre premier : s'il n'est pas égal à p, il est premier avec p. Par exemple : 2 (mais il vaut mieux prendre, en général, 3 ; il y a beaucoup plus de cas d'échec avec 2).
- Le PPCM des entiers depuis 1 jusqu'à un maximum fixé dans la recherche est égal à 2x3x2x5x7x2x3x11x13x2x17x19x23x5x3x29x...

Décomposition en facteurs premiers
Algorithme (p-1) de Pollard

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Exemple : Soit à décomposer le nombre n=R_7=1111111= 239x4 649.
(Ces nombres figurent dans une table, déterminée à l'avance, et obtenue par l'algorithme suivant :

- On parcourt les entiers depuis 2 jusqu'au maximum fixé, tout en disposant de la table des nombres premiers inférieurs ou égaux à ce même maximum.
- Chaque fois que l'on rencontre un nombre premier, on rajoute ce nombre premier.
- Chaque fois que l'on rencontre une puissance d'un nombre premier, on rajoute un facteur égal à ce nombre premier, pour compléter le PPCM.

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Exemple : Soit à décomposer le nombre n=R_7=1111111= 239x4 649.
(On obtient donc 2, puis 3, comme nombres premiers, puis, en passant par 4, on rajoute un facteur 2, puis 5, premier, rien à rajouter pour 6, puis 7, premier, un facteur 2 supplémentaire en passant par 8, un facteur 3 à la rencontre de 9, etc. . .
Cette table contient 6634 éléments pour le PPCM des entiers depuis 2 jusqu'à 65535, tous les nombres représentables sur 16 bits).

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

On effectue donc les calculs suivants

a	q	$a^q[n]$	$a^q - 1$	$(a^q - 1) \wedge n$	comme
2	2	4	3	1	pas de
4	3	64	63	1	pas de
64	2	4096	4095	1	pas de
4096	5	120077	120076	1	pas de
120077	7	1084896	1084895	1	pas de
1084896	2	559627	550626	1	pas de
559627	3	247053	247052	1	pas de
247053	11	339352	339351	1	pas de
339352	13	311394	311393	1	pas de
311394	2	677377	677376	1	pas de
677377	17	569060	569059	239	diviseu

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Explication : on a calculé en fait $2^{2x3x2x5x7x2x3x11x13x2x17} = 2^{24504480}$.

Or 24504480=102960x238, qui est bien de la forme h(p-1) : le calcul de $a^{h(p-1)}$ a permis de trouver p.

La capacité de cet algorithme à trouver un diviseur p d'un nombre n dépend de la taille du plus grand diviseur de p-1, ce qui explique ses résultats très inégaux.

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Introduction aux courbes elliptiques

Définition

Une courbe elliptique sur un corps K a une équation affine de la forme

$$y^2 = x^3 + ax + b$$

(en supposant que le discriminant $\Delta = 4a^3 + 27b^2$ n'est pas nul, pour qu'elle ne soit pas dégénérée).

Remarque. Elle a un point à l'infini dans la direction de Oy.
On considère deux points $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$ d'une pareille courbe. La droite P_1P_2 (la tangente en P_1 à la courbe dans le cas où $P_1 = P_2$) recoupe la cubique en un troisième point de coordonnées $(x_3, -y_3)$...

Introduction aux courbes elliptiques

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Définition

Si pose $P_3 = (x_3, y_3)$ et $P_3 = P_1 + P_2$,

- *Cette addition sur la courbe elliptique est une loi de groupe abélien,*
- *Elle est telle que l'élément neutre est le point à l'infini*
- *... et l'opposé du point P=(x,y) est le point P'=(x,-y).*

Les coordonnées de P_3 sont obtenues comme suit :

$$\begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \text{ si } P_1 \neq P_2 \\ \frac{3x_1^2 + a}{2y_1} \text{ si } P_1 = P_2 \end{cases}$$

alors $x_3 = m^2 - x_1 - x_2$ et $y_3 = m(x_1 - x_3) - y_1$

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Algorithme de Lenstra Ici, il s'agit de calculer dans $\mathbb{Z}/n\mathbb{Z}$, qui n'est pas un corps, donc l'opération risque de ne pas être définie.
Remarque. C'est le cas lorsque $\delta = (x_2 - x_1) \wedge n \neq 1$, auquel cas on ne peut poursuivre le calcul, car l'inverse de $(x_2 - x_1)$ n'est pas défini.
Dans ce cas ($\delta \neq 1$), si $\delta \neq n$, on a trouvé un diviseur de n, et on a gagné. Si $\delta = n$, c'est le cas d' échec.
On applique la méthode de Pollard à la courbe elliptique, en calculant, à partir d'un point P quel- conque P'= kP , en cherchant les coefficients multiplicateurs dans le même tableau (celui des facteurs du PPCM évoqué plus haut).

Divisions successives
Algorithme de Monte-Carlo (1975)
Algorithme du crible quadratique QS de Pomerance
Algorithme (p-1) de Pollard
Algorithme de Lenstra (courbes elliptiques)

Algorithme de Lenstra Si l'on note E($\mathbb{Z}/n\mathbb{Z}$) le groupe additif de la courbe elliptique utilisée, l'intérêt d'opérer sur une courbe elliptique de cette sorte est que le cardinal de E($\mathbb{Z}/n\mathbb{Z}$) n'est pas nécessairement p-1 (comme dans la méthode classique p-1 exposée ci-dessus, où on travaille dans $(\mathbb{Z}/n\mathbb{Z})^*$, de cardinal toujours p-1).
C'est un nombre de la forme p+1-t, où $|t| \leq 2\sqrt{p}$, qui varie selon la courbe utilisée.
Ainsi, en travaillant sur plusieurs courbes simultanément, on augmente les chances que le plus grand diviseur de ce cardinal soit petit, ce qui conditionne, comme on l'a remarqué, le succès de la méthode.