Théorie algorithmique des nombres, Douala

Andreas Enge

15 au 19 juillet 2024

Table des matières

1 Complexité, arithmétique et cryptographie à clef publique

1.1 Rappels de notions de complexité et d'arithmétique

Définition 1.1. Taille d'un objet (entrée/sortie) : nombre de bits pour l'écrire

Exemple 1.2. — $\mathbb{Z} : L(N) = \lfloor \log_2(N) \rfloor + 1$

— $\mathbb{Z}/NZ : L(N-1)$
— $\mathbb{F}_p[X] : (d+1)L(p-1)$ ou $\#\text{coeff} \cdot L(d)L(p-1)$
— $\mathbb{Z}[X]$: c'est compliqué...

Définition 1.3. Soit $f : \mathbb{N} \to \mathbb{R}^+$.

$$\begin{aligned} O(f) &= \left\{g : \mathbb{N} \to \mathbb{R}^+ | \exists c \in \mathbb{R}^+ \forall n \in \mathbb{N} : g(n) \leqslant cf(n)\right\} \\ \Omega(f) &= \left\{g : \mathbb{N} \to \mathbb{R}^+ | \exists c \in \mathbb{R}^+ \forall n \in \mathbb{N} : g(n) \geqslant cf(n)\right\} \\ \Theta(f) &= \Omega(f) \cap O(f) \\ o(f) &= \{g : g/f \to 0\} \\ f \ll g &:\Leftrightarrow f \in o(g) \\ f \sim g &:\Leftrightarrow g/f \to 1 \\ O^{\tilde{}}(g) &= \bigcup_{k=0}^{\infty} O\left(g \log^k g\right) \end{aligned}$$

Exemple 1.4. $L(N) \in \Theta(\log N)$, $L(N) \sim \log_2 N$.

Littérature pour l'arithmétique : [GG99]

Théorème 1.5.
— $M(n) \in O(n \log n) \subseteq O^{\tilde{}}(n)$ *(Harvey–van der Hoeven 2021)*
— $\mathrm{M_X}(d, n) \in O(dn \log d \log n) \subseteq O^{\tilde{}}(dn)$
— *division avec reste comme multiplication (Newton)*
— pgcd *: comme multiplication avec un* log *en plus*

Exponentiation binaire

Calculer $g^e = \underbrace{g \cdot g \cdots g}_{e \text{ times}}$ pour $g \in \mathbb{Z}/N\mathbb{Z}$

$e = \sum_{i=0}^{n-1} e_i 2^i$, $e_i \in \{0, 1\}$

Précalculer les $g_i = g^{2^i}$ par carrés successifs

Multiplier les g_i avec $e_i = 1$

Complexitié : $\sim n$ carrés, au plus $\sim n$ multiplications (en moyenne, $\sim n/2$) :

$$O(\mathrm{M}(N)n) = O(\mathrm{M}(N) \log e)$$

Exponentiation droite à gauche :

```
r := 1
t := g
for (i := 0 to n-1)
   if (e_i == 1)
      r := r * t
   t := t^2
```

Exponentiation gauche à droite :

```
r := 1
for (i := n-1 downto 0)
   if (e_i == 1)
      r := r^2 * g
   else
      r := r^2
```

Exercise 1.6. *Vérifiez que les deux algorithmes fonctionnent dans l'exemple* $e = 11$. *Programmez-les en PARI/GP ; fonction utile :* `binary`

1.2 Évaluation multipoint et interpolation rapides

Évaluer un polynôme f de degré d "en précision n" (par exemple, sur $\mathbb{F}_p$ avec $n \sim \log_2 p$) en un argument : $O(d\,\mathrm{M}(n))$ (schéma de Horner)

Et en d arguments ? Observation clef : $f(x_i) = f \bmod (X - x_i)$

Arbre des sous-produits

Arbre des restes

Complexité :

$$\begin{aligned} & \mathrm{M_X}(d/2,n) + 2\,\mathrm{M_X}(d/4,n) + 4\,\mathrm{M_X}(d/8,n) + \cdots \\ \leqslant\ & \mathrm{M_X}(d/2,n) + \mathrm{M_X}(d/2,n) + \mathrm{M_X}(d/2,n) + \cdots \\ \sim\ & \log_2 d\ \mathrm{M_X}(d/2,n) \in O(\log d\ \mathrm{M_X}(d,n)) \subseteq O^{\sim}(dn) \end{aligned}$$

Interpolation de Lagrange : Trouver f t.q. $f(x_i) = y_i$ pour $i = 1, \ldots, d$.

Soit $m = \prod_{i=1}^{d}(X - x_i)$.

Pour un i, considérer $\prod_{j\neq i}(X - x_j) = \frac{m(X)}{X - x_i}$; valeur 0 en x_j pour $j \neq i$, valeur $v_i = \prod_{j\neq i}(x_i - x_j) = m'(x_i)$ en x_i

$$f(X) = \sum_{i=0}^{d} (y_i/v_i) \frac{m(X)}{X - x_i}$$

Calculer les v_i comme valeurs de m', et $c_i = y_i/v_i$.

Linear combination for linear moduli :

Arbre des sous-produits pour m.

Complexité : encore la même !

Exercise 1.7. *Développez un algorithme rapide pour le théorème des restes chinois effectif.*

1.3 Chiffrement RSA

RSA[RSA78]

- Génération de clefs
  - Clef privée
    - premiers p, q ; $N = pq$
    - $\lambda(N) = \text{ppcm}(p-1, q-1)$
    - e, d t.q. $de \equiv 1 \pmod{\lambda(N)}$
  - Clef publique : N, e
- Chiffrement d'un message $m \in \mathbb{Z}/N\mathbb{Z}$

$$c = m^e$$

- Déchiffrement

$$m = c^d$$

Théorème 1.8 (Sécurité)**.** *Connaître N et $\varphi(N)$ équivaut à connaître p et q.*

Exercise 1.9. *Connaître N et $\lambda(N)$ équivaut à connaître N et $\varphi(N)$. Idée : $g = \text{pgcd}(p-1, q-1) = \varphi(N)/\lambda(N)$. Montrer que $g = \gcd(N-1, \lambda(N))$.*

Théorème 1.10. *Connaître N, e et d équivaut à connaître e, p et q de façon probabiliste.*

Autrement dit, la sécurité élémentaire (trouver la clef privée à partir de la clef publique) de RSA est équivalente à la factorisation.

Problème RSA : Déchiffrer un message donné ; à partir de N, e, m^e, trouver m.

Factoriser casse RSA, sur la réciproque, on ne sait pas.

Boneh–Venkatesan 1998 [BV98] : peut-être non

Aggarwal–Maurer 2008 : peut-être oui

Définition 1.11 (Notions de sécurité)**.** Résistance au but de l'attaquant :
- sécurité élémentaire : trouver clef privée à partir de clef publique
- One-wayness (OW) : trouver m à partir de $c = e(m)$
- Indistinguishability (IND) : distinguer e("oui") de e("non")
- Non-malléabilité (NM) : étant donné $e(m)$ et $e(m')$, trouver $e(m \otimes m')$

selon ses capacités :
- Chosen plaintext attack (CPA) : chiffrer des messages de son choix ; toujours possible avec clef publique
- Chosen ciphertext attack (CCA1) : accès à un oracle de déchiffrement *avant* l'attaque
- Adaptive chosen ciphertext attack (CCA2) : accès à un oracle de déchiffrement avant et *pendant* l'attaque

Exemple 1.12.
- RSA n'est pas IND-CPA : car aucun aléa !
- RSA n'est pas NM : $e(m \cdot m') = (mm')^e = m^e \cdot (m')^e = e(m) \cdot e(m')$

Exemple 1.13. Attaque simple : RSA avec exposant fixé, par exemple $e = 3$ (chiffrement rapide) ; même message à plusieurs destinataires

$$\begin{aligned} c_1 &= m^3 \bmod N_1 \\ c_2 &= m^3 \bmod N_2 \\ c_3 &= m^3 \bmod N_3 \\ c &= m^3 \bmod (N_1 N_2 N_3) \text{ par restes chinois} \end{aligned}$$

Aucune réduction, m est la racine cubique sur $\mathbb{N}$!

Exemple 1.14. Attaques par canaux cachées : Exponentiation binaire fait des choses différentes selon qu'un bit de l'exposant est 1 ou 0. Si on peut observer un déchiffrement (carte à puce, ordinateur partagé), peut-être qu'on peut lire les bits dans la courbe de consommation, le temps pris par des instructions, le rayonnement électromagnétique, etc. Pire, peut-être qu'on peut injecter des fautes.

2 Théorie des nombres élémentaire et analytique

2.1 Résidus quadratiques et racines carrées

Définition 2.1. Ordre $\operatorname{ord} a$ pour $a \in \mathbb{F}_q^\times$ est le plus petit entier $0 < e$ t.q. $a^e = 1$; alors a est une racine primitive e-ème de l'unité.

Générateur de $\mathbb{F}_q$ = élément d'ordre $(q-1)$

Théorème 2.2.

$$a^{q-1} = 1 \forall a \in \mathbb{F}_q^\times$$

Théorème 2.3.

$$\operatorname{ord} a | q-1 \forall a \in \mathbb{F}_q^\times$$

Définition 2.4. Fonction φ d'Euler :

$$\begin{aligned} \varphi(N) &= |(\mathbb{Z}/N\mathbb{Z})^\times| \\ \varphi(p) &= p-1 \\ \varphi(p^k) &= p^k - p^{k-1} \\ \varphi(N_1 N_2) &= \varphi(N_1)\varphi(N_2) \text{ si } \operatorname{pgcd}(N_1, N_2) = 1 \end{aligned}$$

$$\varphi(N) = N \prod_{p|N} \frac{p-1}{p}$$

Théorème 2.5. *Il y a $\varphi(q-1)$ générateurs de $\mathbb{F}_q^\times$; si g en est un, les autres sont les g^e avec* $\operatorname{pgcd}(e, q-1) = 1$.

Définition 2.6. Résidus quadratiques : image de l'homomorpisme de groupes

$$\mathbb{F}_q^\times \to \mathbb{F}_q^\times, \quad x \mapsto x^2$$

de noyau $\{\pm 1\}$; il y en a $(q-1)/2$.

Non-residus quadratiques : les autres $(q-1)/2$.

Algorithme 2.7. aléatoire : tirer au hasard, chance est $1/2$

déterministe : tester $2, 3, \ldots$; plus petit résidu : $O(p^\alpha)$ pour un certain α, $O(\log^2 p)$ sous GRH

Théorème 2.8. *Si g générateur de $\mathbb{F}_q^\times$, alors*
- *résidus* = $\{g^{2e}\}$
- *non-résidus* = $\{g^{2e+1}\}$

Définition 2.9 (Symbole de Legendre)**.**

$$\left(\frac{a}{p}\right) = a^{(p-1)/2} \bmod p \in \{0, \pm 1\} = \begin{cases} 1 & \text{si résidu quadratique} \\ -1 & \text{si non-résidu quadratique} \end{cases}$$

Théorème 2.10.

$$\begin{aligned}
\left(\frac{ab}{p}\right) &= \left(\frac{a}{p}\right)\left(\frac{b}{p}\right) \\
\left(\frac{1}{p}\right) &= 1 \\
\left(\frac{-1}{p}\right) &= (-1)^{(p-1)/2} = \begin{cases} 1 & si\ p \equiv 1 \pmod 4 \\ -1 & si\ p \equiv 3 \pmod 4 \end{cases} \\
\left(\frac{2}{p}\right) &= (-1)^{(p^2-1)/8} = \begin{cases} 1 & si\ p \equiv \pm 1 \pmod 8 \\ -1 & si\ p \equiv \pm 3 \pmod 8 \end{cases}
\end{aligned}$$

Théorème 2.11 (Réciprocité quadratique)**.** *Soient p, q deux premiers impairs.*

$$\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}\left(\frac{p}{q}\right) = \begin{cases} -\left(\frac{p}{q}\right) & si\ p \equiv q \equiv 3 \pmod 4 \\ \left(\frac{p}{q}\right) & sinon \end{cases}$$

Définition 2.12 (Symbole de Kronecker)**.**

$$\begin{aligned}
\left(\frac{a}{b}\right) &= 0 \text{ si } \operatorname{pgcd}(a,b) \neq 1 \\
\left(\frac{p}{2}\right) &= \left(\frac{2}{p}\right) \text{ pour } p \text{ impair}
\end{aligned}$$

puis extension multiplicative dans les deux arguments

réciprocité quadratique est encore vraie

Exercise 2.13. *Calculer à la main si* 7411 *est un résidu quadratique modulo* 9283*. Comparer avec la fonction* `kronecker` *de PARI/GP.*

Algorithme 2.14. Si $p \equiv 3 \pmod 4$ et $a = x^2 \in \mathbb{F}_p^\times$, alors $x = \pm a^{(p+1)/4}$.

Exercise 2.15. *Si $p \equiv 5 \pmod 4$, on a $a^{(p-1)/4} = \pm 1$.*

Si c'est $+1$, alors $x = \pm a^{(p+3)/8}$;
sinon, $x = 2a(4a)^{(p-5)/8}$.

Le premier cas se prouve comme précédemment. Dans le deuxième, on utilise $\left(\frac{2}{p}\right) = -1$.

En général : Écrire $p - 1 = 2^e p'$ avec p' impair. Tonelli–Shanks (algorithme 1.5.1 de [Coh93]) de complexité $O(e^2 \log p + \log^2 p)$; rapide en général, mais $O(\log^3 p)$ dans le pire cas.

Cipolla : $O(\log^2 p)$ par exponentiation dans $\mathbb{F}_{p^2}$, donc plus lent en pratique quand e est petit.

2.2 Théorème des nombres premiers

Définition 2.16.

$$\begin{aligned} \pi(x) &:= |\{p \leqslant x : p \text{ premier}\}| = \sum_{p \leqslant x, p \text{ premier}} 1 \\ p_n &:= n\text{-ème nombre premier} \\ \vartheta(x) &:= \sum_{p \leqslant x} \log p = \log(\text{primorial}(x)) \end{aligned}$$

Théorème 2.17 (Hadamard–De la Vallée Poussin)**.**

$$\pi(x) \sim \frac{x}{\log x}$$

Exercise 2.18.

$$\frac{\pi(2x) - \pi(x)}{x} \sim \frac{1}{\log x} \sim \frac{1}{\log(2x)}$$

Théorème 2.19.

$$p_n \sim n \log n$$

Théorème 2.20 (Versions explicites : [RS62])**.** *(3.7)*

$$\frac{x}{\log x} < \pi(x) < (5/4)\,\frac{x}{\log x} \text{ pour } x \geqslant 114$$

(3.10), (3.11)

$$n(\log n + \operatorname{llog} n - 3/2) < p_n < n(\log n + \operatorname{llog} n - 1/2) \text{ pour } n \geqslant 20$$

Estimations dérivées, pour lesquelles il y a aussi des versions effectives :

$$\begin{aligned} \vartheta(x) &= \sum_{p \leqslant x, p \text{ premier}} \log x \approx \sum_{n \leqslant x} \frac{\log n}{\log n} \sim \int_2^x dt \sim x \\ \sum_{p \leqslant x} \frac{\log p}{p} &\approx \int_2^x \frac{dt}{t} \sim \log x \\ \sum_{p \leqslant x} \frac{1}{p} &\approx \int_2^x \frac{dt}{t \log t} \sim \operatorname{llog} x \end{aligned}$$

Exercise 2.21. *Jouer avec PARI/GP pour calculer les quantités $\pi(x)$, $\vartheta(x)$, p_n etc. pour des valeurs grandissantes de x.*

Suggestion : `???prime` *ou taper* `prime` *puis deux fois* `<tab>` *pour la complétion automatique.*

2.3 Sommes de carrés

[Cox89]

Théorème 2.22 (Fermat)**.** *Soit p premier impair.*
$p = x^2 + y^2 \Leftrightarrow p \equiv 1 \pmod 4 \Leftrightarrow \left(\frac{-1}{p}\right) = 1$

3 Factorisation

3.1 Algorithmes exponentiels

Algorithme 3.1 (Force brute / trial division)**.** Diviser par 2, puis tous les nombres impairs jusqu'à $\sqrt{N}$; mieux : les $\Theta(\sqrt{N}/\log N)$ nombres premiers

$O(\sqrt{N}/\log N)$ divisions, complexité $O^{\sim}\left(\sqrt{N}\right)$

Algorithme 3.2 (Pollard ρ)**.**
- Choisir une fonction $f : \mathbb{Z}/N\mathbb{Z} \to \mathbb{Z}/N\mathbb{Z}$ "aléatoire" ; par exemple, $x \mapsto x^2 + 1$
- $x_0 \in \mathbb{Z}/N\mathbb{Z}$ au hasard
- Itérer
$$x_{i+1} = f(x_i)$$
- Si $p|N$ et $x_i \equiv x_j \pmod p$, alors $p|\operatorname{pgcd}(x_i - x_j, N)$
- "Théorème" : prépériode λ et période $\mu \in O(\sqrt{p})$
- Si $i \geqslant \max(\lambda, \mu)$, alors $p|\operatorname{pgcd}(x_i - x_{2i}, N)$

Complexité :
- $O(1)$ éléments, donc $O(\log N)$ en mémoire !
- $O(\sqrt{p})$ opérations dans $\mathbb{Z}/N\mathbb{Z}$, donc $O^{\sim}\left(\sqrt{p}\log N\right) \subseteq O^{\sim}\left(\sqrt[4]{N}\right)$

Exercise 3.3. *Programmer cet algorithme.*

Algorithme 3.4 (Pollard $p-1$)**.**
- $a \in \mathbb{Z}/N\mathbb{Z}$ aléatoire, e un exposant.
- Si $p - 1|e$, alors $p|\operatorname{pgcd}(a^e - 1, N)$.
- Supposons $p-1$ B-*friable*, sans facteur premier $> B$. Alors on a une bonne chance que
$$p - 1|B!$$
ou
$$p - 1| \prod_{p^r \leqslant B} \operatorname{ppcm}(1, \ldots, B)$$

— Calculer

$$\mathrm{pgcd}(a^e - 1, N)$$

Complexité :

— $O(\log N)$ en mémoire

— $O(\log e) \subseteq O(B \log B)$ multiplications dans $\mathbb{Z}/N\mathbb{Z}$

Si on veut être sûr du résultat, il faut prendre $B \geqslant (p-1)/2$, donc $B \approx \sqrt{N}$.

Exercise 3.5. *Utiliser la méthode $p-1$ pour factoriser $N = 119$; on peut prendre $B = 3$, $e = B! = \mathrm{ppcm}\{1, \ldots, B\} = 6$.*

3.2 Factorisation avec courbes elliptiques : ECM

[Eng99]

Définition 3.6. K un corps (p.ex. $\mathbb{F}_q$)

Plan affine

$$\mathbb{A}^2(K) : (x, y) \in K^2$$

Plan projectif

$$\mathbb{P}^2(K) : (x : y : z) \in K^3 \backslash \{(0, 0, 0)\} \text{ modulo équivalence}$$

$$(x, y, z) \sim (\lambda x, \lambda y, \lambda z) \text{ pour } \lambda \in K^\times$$

$$\begin{array}{lrcl} \mathbb{A} \to \mathbb{P}, & (x, y) & \mapsto & (x : y : 1) \\ & (x/z, y/z) & \leftarrow & (x : y : z) \text{ si } z \neq 0 \end{array}$$

$z = 0$: droite à l'∞

Définition 3.7 (Courbe elliptique)**.** Solutions dans $\mathbb{A}$ d'une équation de type

$$E : Y^2 = X^3 + aX + b \text{ avec } a, b \in K$$

Solutions dans $\mathbb{P}$ de l'équation

$$Y^2 Z = X^3 + aXZ^2 + bZ^3$$

Point à l'infini : $\mathcal{O} = (0 : 1 : 0)$

Théorème 3.8 (Loi de groupe)**.** — *$\mathcal{O}$ est l'élément neutre*

— *La somme de trois points sur une droite est $\mathcal{O}$.*

Exercise 3.9. *Dériver des formules pour l'addition (générique) et le doublement.*

Remarque 3.10. Pour $P \in E$ et $n \in \mathbb{Z}$, on peut calculer nP par des additions et des doublements sur E avec $O(\log n)$ opérations dans K.

Théorème 3.11 (Hasse)**.**

$$(\sqrt{p}-1)^2 = p+1-2\sqrt{p} \leqslant |E(\mathbb{F}_p)| \leqslant p+1+2\sqrt{p} = (\sqrt{p}+1)^2$$

Théorème 3.12 (Proposition (1.9) de [Len87])**.**
$\exists c_-, c_+ \forall p \forall S$ *ensemble de cardinaux* s *t.q.* $|p+1-s| \leqslant \sqrt{p}$*,*

$$c_-(|S|-2)\frac{\sqrt{p}}{\log p} \leqslant \#E(\mathbb{F}_p) \text{ de cardinal } s \leqslant c_+|S|\sqrt{p}\log p \operatorname{llog}^2 p$$

Borne sup valable dans tout l'intervalle de Hasse.

Exercise 3.13. *Pour un p choisi, calculer/afficher les nombres de courbes avec les différents cardinaux.*

Pour cela, lire la doc de PARI/GP, ou taper `ell<tab>`*.*

Astuce : `E = ellinit ([Mod (j, p)]);` *crée la courbe elliptique donnée par l'invariant modulaire j.*

Algorithme 3.14 (ECM)**.** Fixer une borne B et $e = B!$. Prendre une courbe elliptique E "modulo N" et un point $P \in E(\mathbb{Z}/N\mathbb{Z})$ au hasard. Calculer $eP = (x : y : z)$.

Si $p|N$ et $|E(\mathbb{F}_p)||e$, alors $z \equiv 0 \pmod{p}$.

Calculer $\operatorname{pgcd}(z, N)$ et trouver p.

Si pas de succès, répéter ou agrandir B.

Probabilité de succès pour une courbe
$\gtrsim$ probabilité qu'un nombre entre $p+1-\sqrt{p}$ et $p+1+\sqrt{p}$ est B-friable
$\approx$ probabilité qu'un nombre autour de p est B-friable

Définition 3.15.

$$\psi(x,y) = \#\{1 \leqslant n \leqslant x : n \text{ est } y\text{-friable}\}$$

Exercise 3.16. *Fixer $y = 23$, calculer $\psi(x,y)/x$ pour $x = 10^2, 10^3, \ldots$.*

$$u = \frac{\log x}{\log y}$$

Théorème 3.17 (Canfield–Erdős–Pomerance 1983, simplifié)**.** *Pour $x \geqslant 1$ et $u \geqslant 3$,*

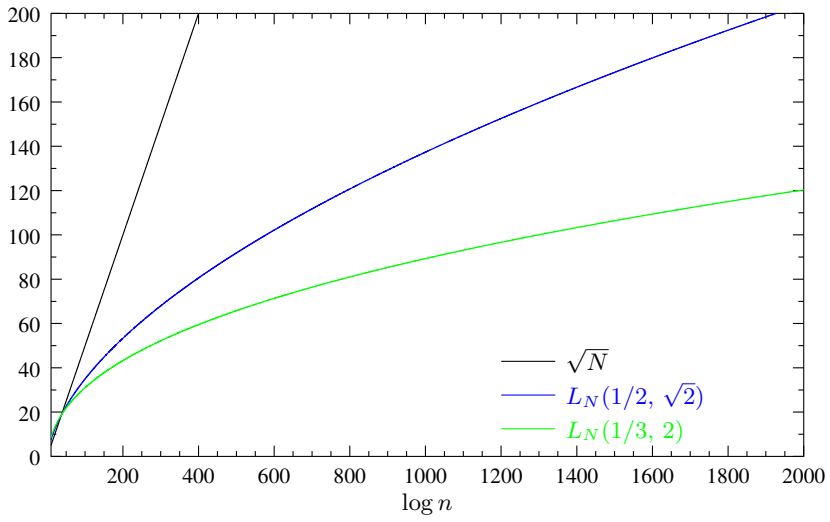
$$\frac{\psi(x,y)}{x} \geqslant \exp\big(-u\log u(1+o(1))\big)$$

Définition 3.18 (fonctions sous-exponentielles)**.**

$$L_N(\alpha,c) = e^{c(\log N)^\alpha(\operatorname{llog} N)^{1-\alpha}},$$

pour $0<\alpha<1$ et $c>0$.

— $\alpha=0$: $\log^c N$, polynomial
— $\alpha=1$: N^α, exponentiel



Théorème 3.19. *Règles de calcul*

— $L_N(\alpha,c_1)\cdot L_N(\alpha,c_2) = L_N(\alpha,c_1+c_2)$
— $L_N(\beta,c) \in L_N(\alpha,o(1))$ *for* $\beta<\alpha$ *; en particulier,* $\log^k(N) \in L_N(\alpha,o(1))$

Théorème 3.20. *Pour $c>0$ et $x\to\infty$,*

$$\frac{\psi\left(x, L_x(1/2,c)\right)}{x} \geqslant L_x\left(1/2, -\frac{1}{2c}+o(1)\right).$$

Théorème 3.21 (heuristique)**.** *Supposons qu'on cherche des facteurs premiers jusqu'à p. Alors en choisissant $B = L_p(1/2,c)$, testant une courbe prend $\tilde{O}(B) \subseteq L_p(1/2, c+o(1))$. Au bout de $1/L_p(1/2,-1/(2c)+o(1)) = L_p(1/2,1/(2c)+o(1))$ essais, on a une bonne chance de succès.*

Temps total :

$$L_p(1/2, c + 1/(2c) + o(1))$$

Valeur optimal $c = \sqrt{2}/2$ *:*

$$L_p(1/2, \sqrt{2} + o(1)) \text{ } \textit{opérations modulo } N$$

Avec $p = \sqrt{N}$ *:*

$$L_N(1/2, 1 + o(1))$$

Record avec `gmp-ecm` : facteur de 73 chiffres en 2024

3.3 Crible quadratique

[Kra26]

Idee de départ comme pour Pollard-ρ : Si $x^2 \equiv y^2 \pmod N$, c'est-à-dire $N | x^2 - y^2$, alors $\text{pgcd}(N, x - y)$ a une chance de trouver un facteur.

Définition 3.22 (Base de facteurs)**.** pour une borne de friabilité B est

$$\mathcal{B} = \{p \leqslant B,\ p \text{ premier}\} = \{p_1, \ldots, p_n\}$$

Algorithme 3.23.

— 1ère phase : Trouver des *relations*, avec des x_i aléatoires :

$$x_i^2 \bmod N = \prod_{j=1}^{n} p_j^{e_{ij}}$$

— 2nde phase : Par algèbre linéare modulo 2, trouver une combinaison I de lignes t.q.

$$\forall j \sum_{i \in I} e_{ij} \equiv 0 \pmod 2$$

— Alors $x^2 \equiv y^2 \pmod N$ avec

$$\begin{aligned} x &= \prod_{i \in I} x_i \\ y &= \prod_{j=1}^{n} p_j^{\left(\sum_{i \in I} e_{ij}\right)/2} \end{aligned}$$

Exemple 3.24. $N = 2041$, $B = 10$

Prendre x à partir de $\sqrt{N}$ pour avoir de petits restes :

$$\begin{array}{lrlllll}
46^2 \bmod 2041 & = & 75 & = & 3 & \cdot 5^2 & \\
47^2 \bmod 2041 & = & 168 & = & 2^3 & \cdot 3 & \cdot 7 \\
49^2 \bmod 2041 & = & 360 & = & 2^3 & \cdot 3^2 & \cdot 5 \\
51^2 \bmod 2041 & = & 560 & = & 2^4 & \cdot 5 & \cdot 7 \\
53^2 \bmod 2041 & = & 768 & = & 2^8 & \cdot 3 &
\end{array}$$

Multiplier toutes les lignes sauf la dernière

$$\begin{aligned}
(46 \cdot 47 \cdot 49 \cdot 51 \bmod 2041)^2 &= 311^2 \\
&= 2^{10} \cdot 3^4 \cdot 5^4 \cdot 7^2 \\
&= (2^5 \cdot 3^2 \cdot 5^2 \cdot 7 \bmod 2041)^2 \\
&= 1416^2 \pmod{2041}
\end{aligned}$$

$$\gcd(1416 - 311, 2041) = \gcd(1105, 2041) = 13$$

Exercise 3.25. *Si $x_i = \lceil \sqrt{N} \rceil +$ petit, alors $x_i^2 - N \in O(\sqrt{N})$.*
Alors on peut obtenir une complexité en

$$L_N(1/2, \sqrt{2} + o(1))$$

Autres algorithmes :

— Crible quadratique

$$L_N(1/2, 1 + o(1))$$

— Crible algébrique général ("general number field sieve")

$$L_N\left(1/3, \sqrt[3]{\frac{64}{9}} + o(1)\right)$$

— Crible algébrique spécial pour des nombres particuliers

$$L_N\left(1/3, \sqrt[3]{\frac{32}{9}} + o(1)\right)$$

4 Primalité

Preuve facile : Tester tous les diviseurs (premiers) possibles jusqu'à $\sqrt{N}$; exponentiel !

4.1 Tests de primalité

Définition 4.1 (Test de primalité)**.** Réponses :
— "non" alors composé ;
— "peut-être" ; probabilité < 1 si nombre composé

Théorème 4.2.

$$(\mathbb{Z}/N\mathbb{Z})^\times = \{a = 1, \dots, N-1 : \operatorname{pgcd}(a, N) = 1\}$$

est cyclique

$$\Leftrightarrow N = 2, 4, p^e \text{ ou } 2p^e \text{ pour } p \text{ premier impair}$$

Exercise 4.3. *Prouver la dernière phrase.*

Corollaire 4.4. $(\mathbb{Z}/N\mathbb{Z})^\times$ *cyclique d'ordre* $N-1 \Leftrightarrow N$ *premier*

Définition 4.5 (Test de Fermat)**.** N *pseudopremier* en base b pour $\operatorname{pgcd}(b, N) = 1$ si $b^{N-1} \equiv 1 \pmod N$.

Nombre de Carmichael : passe tous ces tests.
C'est la suite A002997 dans OEIS. `https://oeis.org/A002997`

Alford–Granville–Pomerance 1994, Annals of Mathematics : Il y en a une infinité.

Exercise 4.6. *Démontrer que* N *est Carmichael* $\Leftrightarrow N$ *sans facteur carré et* $\forall p|N$: $p-1|N-1$.

Faire un test de Fermat avec quelques bases pour 561*. Montrer que c'est un nombre de Carmichael (en fait, c'est le plus petit).*

Théorème 4.7 (Test d'Euler/Solovay–Strassen)**.**

$$b^{(N-1)/2} \equiv \left(\frac{b}{N}\right) \pmod N$$

Si N *composé, alors nombre de* faux témoins $\leqslant \varphi(N)/2$.

Exercise 4.8. *Prouver le théorème, selon les lignes suivantes :*
— *Si oui pour* b_1*, non pour* b_2*, alors non pour* $b_1 b_2$*.*
$\Rightarrow$ *S'il y a un non, alors au moins la moitié est non.*
— *Si* $p^2|N$*, on peut trouver un* b *avec* $\not\equiv \pm 1$*.*
— *Sinon et* N *composé, soit* $p|N$*. On peut trouver deux* b *avec* $\left(\frac{b}{p}\right) = \pm 1$ *et* $b \equiv 1 \pmod{(N/p)}$

Définition 4.9 (Test de Miller–Rabin)**.** N *pseudopremier* fort en base b pour N impair, $N - 1 = 2^e N'$ avec N' impair, si

$$b^{N'} \equiv 1 \pmod N \text{ ou } b^{2^r N'} \equiv -1 \pmod N \text{ pour un } 0 \leqslant r < e$$

Théorème 4.10. *pseudopremier fort ⇒ pseudopremier d'Euler*

nombre de faux témoins $\leqslant \frac{\varphi(N)}{4}$

Théorème 4.11 (Burthe 1996)**.** *Si on nombre passe k tests, la probabilité qu'il soit composé est* $\leqslant \frac{1}{4^k}$.

4.2 Preuves de primalité

Théorème 4.12 (Certificats de [Pra75])**.** *(N impair). Si (et seulement si)*

$$N - 1 = \prod p_i^{e_i};$$

les p_i sont premiers ; il y a un élément $g \in \mathbb{Z}/N\mathbb{Z}$ t.q.

$$\begin{aligned} g^{N-1} &\equiv 1 \pmod{N} \\ g^{(N-1)/p_i} &\not\equiv 1 \pmod{N}\, \forall i \end{aligned}$$

alors N est premier.

Corollaire 4.13. *"Every prime has a succinct certificate" (récursion sur les p_i), Primalité $\in$ co-NP $\cap$ NP*

Théorème 4.14 ([AKS04])**.** *"PRIMES is in P"*

Voir aussi [Mor04].

Théorème 4.15. *N est premier ssi $P(X) = (X+1)^N - X - 1$ est identiquement nul modulo N.*

Théorème 4.16. *Soit N impair, pas une puissance parfaite. s entier, r premier, q plus grand premier divisant $r-1$.*

- *Condition arithmétique*

$$N^{(r-1)/q} \pmod{r} \notin \{0, 1\}$$

- *Condition combinatoire*

$$\binom{q-1+s}{s} > N^{2\lfloor\sqrt{r}\rfloor}$$

- *Divisibilité élémentaire : N n'a pas de facteur premier $\leqslant s$*
- *Tests de pseudoprimalité*

$$(X-a)^N \equiv X^N - a \pmod{X^r - 1, N} \quad \forall 1 \leqslant a \leqslant s$$

Alors N est premier.

$L = \log_2 N$

Complexité : $O^{\sim}(srL^2)$; bon choix de s, r donne $O^{\sim}(L^{12})$; améliorations pour $O^{\sim}(L^6)$.

Théorème 4.17 ([GK86])**.** *E courbe elliptique modulo N*
$m = \#$ points sur E, comptés comme si N était premier
$m = cN'$
P point sur E avec

$$cP = (x : y : z) \text{ t.q. } \operatorname{pgcd}(z, N) = 1; \quad mP = \mathcal{O}$$

$$N' > \left(\sqrt[4]{N} + 1\right)^2.$$

Si N' est premier, alors N est premier.

Complexité (heuristique)
- $O(L)$ étapes
- comptage de points avec SEA : $O^{\sim}(L^4)$
- probabilité de succès heuristique pour $m = 2N' \approx 1/\log N' \approx 1/L$, en moyenne $O^{\sim}(L)$ courbes à tester

$\Rightarrow O^{\sim}(L^6)$

Théorème 4.18 (ECPP, [AM93])**.** *Utiliser CM (multiplication complexe) pour trouver des courbes.*

$D < 0$ discriminant fondamental ($D \neq -3, -4$)

Si N premier et

$$4N = t^2 - v^2 D$$

(autrement dit, N décomposé dans les entiers de $\mathbb{Q}(\sqrt{D})$ en deux idéaux principaux engendrés par $\frac{t \pm v\sqrt{D}}{2}$), alors

$$\exists E/\mathbb{F}_N \text{ avec } N + 1 - t \text{ points.}$$

Pour la trouver,
- *calculer un* polynôme de classes *sur $\mathbb{Z}[X]$, de degré $h(D) \in O^{\sim}\left(\sqrt{|D|}\right)$ et avec des coefficients de taille $O^{\sim}\left(\sqrt{|D|}\right)$, en temps $O^{\sim}(|D|)$ [Eng09];*
- *réduire modulo N,*
- *trouver une racine,*

— *construire la courbe,*
— *prendre un point dessus au hasard.*

Algorithme 4.19 ([FKMW04, Mor07])**.**

1. phase : "Downrun"
   — $\mathcal{Q} = \{q^*\}$ "premiers signés" : $q^* \equiv 1 \pmod 4$, ou $-4, \pm 8$;
   et $\left(\frac{q^*}{N}\right) = 1$
   Tonelli–Shanks/Cipolla : Calculer les $\sqrt{q^*} \bmod N$.
   $\#\mathcal{Q} \in \Theta^{\sim}(L)$
   Complexité : $O^{\sim}(L \cdot L^2) \subseteq O^{\sim}(L^3)$
   — Calculer un ensemble $\mathcal{D} = \{D\}$, $D < 0$, de produits de q^* (discriminants fondamentaux),
   et $\sqrt{D} \bmod N$ par produits.
   $\#\mathcal{D} \in \Theta^{\sim}(L^2)$
   $|D| \in O^{\sim}(L^2)$
   — Cornacchia : Essayer de résoudre

   $$4N = t^2 - v^2 D \text{ pour les } D$$

   Cela donne un ensemble $\mathcal{M} = \{m\}$ de cardinaux de courbes elliptiques, $m = p + 1 \pm t$.
   $\#\mathcal{M} \in \Theta^{\sim}(L)$
   Complexité : $O^{\sim}(L^2 \cdot L) \subseteq O^{\sim}(L^3)$
   — Faire de la trial division pour écrire les $m = cN'$,
   c friable pour une borne B.

   Complexité : $O^{\sim}(L \cdot L)$ négligeable
   — Tester la primalité des N'.
   Si un premier trouvé, récursion.
   Sinon, prendre plus de q^*
   en moyenne $\Theta^{\sim}(1)$ valeurs restent
   Complexité : $O^{\sim}(L \cdot L^2) \subseteq O^{\sim}(L^3)$
2. phase : CM ; pour chaque D, $m = cN'$ retenu en Phase 1 :
   — Construire polynôme de classes de degré $O^{\sim}\left(\sqrt{|D|}\right) \subseteq O^{\sim}(L)$ en temps $O^{\sim}(L^2)$.
   — Trouver une racine en temps $O^{\sim}(L^3)$.
   — Écrire la courbe en temps $O^{\sim}(L)$.
   — Trouver P (racine carrée) en temps $O^{\sim}(L^2)$.
   — cP et mP en temps $O^{\sim}(L^2)$.

Complexité totale : $O^{\sim}(L^4)$

Implantation en MPI : Parallélisation sur $\Omega(L)$ cœurs triviale, temps réel $O^{\sim}(L^3)$.

Binaire `ecpp` du logiciel `cm` : `https://www.multiprecision.org/cm/`

Nouveau record [Eng24] : $(10^{86453}-1)/9$

1. phase
   - 2980 étapes
   - 760 à 2640 cœurs
   - 383 années CPU
   - 103 jours temps réel
   - répartition
     - 13% racines carrées
     - 47% Cornacchia
     - 10% factorisation
       $B = w'\,2^{29}$ pour $43 \leqslant w' \leqslant 172$
     - 30% tests de primalité
2. phase
   - machine avec 96 cœurs et 1TB de mémoire
   - 25 années CPU (6% du temps total)
   - répartition
     - 5% polynômes de classes
     - 95% racines modulo N
3. phase : Vérification !
   - machine avec 96 cœurs
   - 7 mois CPU
   - 48 h temps réel

Références

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2) :781–793, 2004.

[AM93] A. O. L. Atkin and F. Morain. Elliptic curves and primality proving. *Mathematics of Computation*, 61(203) :29–68, 1993.

[BV98] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may be easier than factoring (extended abstract). In Kaisa Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 59–71, Berlin, 1998. Springer-Verlag.

[Coh93] Henri Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1993.

[Cox89] David A. Cox. *Primes of the Form $x^2 + ny^2$ — Fermat, Class Field Theory, and Complex Multiplication*. John Wiley & Sons, New York, 1989.

[Eng99] Andreas Enge. *Elliptic Curves and Their Applications to Cryptography — An Introduction*. Kluwer Academic Publishers, 1999.

[Eng09] Andreas Enge. The complexity of class polynomial computation via floating point approximations. *Mathematics of Computation*, 78(266) :1089–1107, 2009.

[Eng24] Andreas Enge. FastECPP over MPI. Technical Report 4522492, HAL, 2024. To appear in International Congress on Mathematical Software — ICMS 2024.

[FKMW04] J. Franke, T. Kleinjung, F. Morain, and T. Wirth. Proving the primality of very large numbers with fastECPP. In Duncan Buell, editor, *Algorithmic Number Theory — ANTS-VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 194–207, Berlin, 2004. Springer-Verlag.

[GG99] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

[GK86] Shafi Goldwasser and Joe Kilian. Almost all primes can be quickly certified. In *Proc. 18th Annual ACM Symp. on Theory of Computing*, pages 316–329, 1986.

[Kra26] M. Kraitchik. *Théorie des nombres*, volume 2 : *Analyse indéterminée du second degré et factorisation*. Gauthier-Villars, Paris, 1926.

[Len87] H. W. Lenstra Jr. Factoring integers with elliptic curves. *Annals of Mathematics*, 126(3) :649–673, 1987.

[Mor04] François Morain. La primalité en temps polynomial [d'après Adleman, Huang ; Agrawal, Kayal, Saxena]. *Astérisque*, 294(917) :205–230, 2004.

[Mor07] F. Morain. Implementing the asymptotically fast version of the elliptic curve primality proving algorithm. *Mathematics of Computation*, 76(257) :493–505, 2007.

[Pra75] Vaughan R. Pratt. Every prime has a succinct certificate. *SIAM Journal on Computing*, 4(3) :214–220, 1975.

[RS62] J. Barkley Rosser and Lowell Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6 :64–94, 1962.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public–key cryptosystems. *Communications of the ACM*, 21(2) :120–126, February 1978.