Étude et Implémentation des Attaques sur RSA

1A - Cryptanalyse et Sécurité des Systèmes

AHNANI Ali

AVRIL 2025

Préliminaires

Définition 1 (Arithmétique modulaire)**.** *Soit $n \in \mathbb{N}^*$. On dit que deux entiers a et b sont* congruents modulo n *si :*

$$a \equiv b \pmod{n} \quad \textit{c'est-à-dire} \quad n \mid (a-b)$$

Cela définit une relation d'équivalence sur $\mathbb{Z}$, et l'ensemble des classes d'équivalence est noté $\mathbb{Z}/n\mathbb{Z}$.

Définition 2 (Anneau $\mathbb{Z}/n\mathbb{Z}$)**.** *L'ensemble $\mathbb{Z}/n\mathbb{Z}$ est muni d'une structure d'anneau : les opérations d'addition et de multiplication sont définies par :*

$$[a]_n + [b]_n = [a+b]_n, \quad [a]_n \cdot [b]_n = [ab]_n$$

Lorsque n est premier, $\mathbb{Z}/n\mathbb{Z}$ est un corps, et tout élément non nul y est inversible.

Définition 3 (Indicateur d'Euler $\varphi(n)$)**.** *Pour un entier $n \geq 1$, la fonction indicatrice d'Euler, notée $\varphi(n)$, désigne le nombre d'entiers k tels que $1 \leq k \leq n$ et $\gcd(k,n) = 1$, autrement dit :*

$$\varphi(n) = |\{k \in \{1,\dots,n\} \mid \gcd(k,n) = 1\}|$$

Lorsque $n = p \cdot q$ est le produit de deux nombres premiers distincts, on a :

$$\varphi(n) = (p-1)(q-1)$$

Définition 4 (Isomorphisme d'anneaux)**.** *Soient A et B deux anneaux. Une application $f : A \to B$ est un* isomorphisme d'anneaux *si elle est bijective, et si elle conserve les opérations :*

$$f(x+y) = f(x) + f(y) \quad \textit{et} \quad f(xy) = f(x)f(y)$$

Dans le cas du RSA, le théorème des restes chinois permet de montrer que :

$$\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z} \quad \textit{(si } n = pq \textit{ avec } p, q \textit{ premiers)}$$

Théorème 1 (Petit théorème de Fermat)**.** *Soit p un nombre premier, et soit $a \in \mathbb{Z}$ tel que $p \nmid a$. Alors :*

$$a^{p-1} \equiv 1 \pmod{p}$$

Remarque 1. *Ce théorème est un cas particulier du théorème d'Euler, qui affirme que pour tout $a \in \mathbb{Z}$ tel que $\gcd(a,n) = 1$, on a :*

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

Définition 5 (Exponentiation modulaire)**.** *Soient $a \in \mathbb{Z}$, $e \in \mathbb{N}$, et $n \in \mathbb{N}^*$. L'exponentiation modulaire désigne le calcul de $a^e \mod n$, souvent notée :*

$$a^e \bmod n$$

Elle est à la base de nombreux algorithmes cryptographiques, notamment le chiffrement RSA.

Définition 6 (Inversible modulaire)**.** *Un entier $a \in \mathbb{Z}$ est inversible modulo n s'il existe un entier $b \in \mathbb{Z}$ tel que :*

$$ab \equiv 1 \pmod{n}$$

Cela équivaut à dire que $\gcd(a,n) = 1$, et l'inverse b est unique modulo n.

Rappels sur les réseaux euclidiens

Définition 7 (Réseau, dimension, base)**.** *Soit $n \in \mathbb{N}$, et L un sous-ensemble de $\mathbb{R}^n$. On dit que L est un* réseau *s'il existe $m \in \mathbb{N}$ et une famille libre $b_1, \ldots, b_m \in \mathbb{R}^n$ telle que :*

$$L = \sum_{i=1}^{m} \mathbb{Z} b_i = \left\{ \sum_{i=1}^{m} a_i b_i \;\middle|\; a_i \in \mathbb{Z} \right\}$$

On dit alors que m est la dimension *du réseau et que $b_1, \ldots, b_m$ en est une base.*

Définition 8 (Déterminant)**.** *Soit L un réseau de $\mathbb{R}^n$ et $b_1, \ldots, b_m$ une base. On appelle* déterminant *de L la grandeur :*

$$\det(L) = |\det(b_1, \ldots, b_m)|$$

Définition 9 (Orthogonalisation de Gram-Schmidt)**.** *Soit $b_1, \ldots, b_n$ une base de $\mathbb{R}^n$. On pose $b_1^* = b_1$, puis on définit la base orthogonale $b_1^*, \ldots, b_n^*$ par :*

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{où } \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$$

Définition 10 (Base réduite)**.** *Soit $B = (b_1, \ldots, b_n)$ une base d'un réseau L. On dit que B est* réduite *si :*

$$\forall 1 \leq j < i \leq n,\ |\mu_{i,j}| \leq \frac{1}{2} \quad \text{et} \quad \forall 2 \leq i \leq n,\ \|b_i^* + \mu_{i,i-1} b_{i-1}^*\|^2 \geq \frac{3}{4} \|b_{i-1}^*\|^2$$

Propriété 1. *Soit $b_1, \ldots, b_n$ une base réduite de L. Alors :*

$$\forall 1 \leq j \leq i \leq n,\ \|b_j^*\|^2 \leq 2^{i-j} \|b_i^*\|^2$$

$$\det(L) = \prod_{i=1}^{n} \|b_i^*\| \quad \text{et} \quad \|b_1\| \leq 2^{\frac{n-1}{4}} \cdot \det(L)^{1/n}$$

Algorithme LLL

L'algorithme **LLL**, du nom de ses créateurs A. Lenstra, H. Lenstra et L. Lovász, transforme une base d'un réseau en une base réduite.

Il consiste à agir sur la base $b_1, \ldots, b_n$, donnée en entrée, de telle sorte que chaque transformation sur la famille conserve sa propriété de base du réseau. De plus, on voudra qu'à la fin d'une étape $k \in \{1, \ldots, n+1\}$, les invariants suivants soient vérifiés :

$$\forall\ 1 \leq l < k, \quad |\mu_{k,l}| \leq \frac{1}{2} \tag{1}$$

$$\forall\ 1 < i < k, \quad \left\| b_i^* + \mu_{i,i-1} b_{i-1}^* \right\|^2 \geq \frac{3}{4} \|b_{i-1}^*\|^2 \tag{2}$$

On initialise $k = 2$. On peut ensuite résumer l'algorithme de la manière suivante :

— À chaque étape d'indice k, on vérifie la condition (6).

— Si $|\mu_{k,k-1}| > \frac{1}{2}$, on effectue la transformation $b_k \leftarrow b_k - \lfloor \mu_{k,k-1} \rceil b_{k-1}$, puis on met à jour la base et les coefficients de Gram-Schmidt.

— Ensuite, on teste la condition (7). Si elle est satisfaite, on incrémente $k \leftarrow k+1$. Sinon, on échange $b_k \leftrightarrow b_{k-1}$, on recalcule les coefficients et on pose $k \leftarrow \max(k-1, 2)$.

Propriété 2 (Termination de LLL)**.** *L'algorithme LLL se termine.*

On remarque que k est incrémenté uniquement lorsque les invariants sont vérifiés, et que lorsque $k = n+1$, la base fournie est bien réduite.

Théorème 2 (Complexité de LLL)**.** *Soit L un réseau, et $b_1, \ldots, b_n$ une base de ce dernier. On pose*

$$B = \max(2, \|b_1\|, \ldots, \|b_n\|).$$

Alors l'algorithme LLL permet d'obtenir une base réduite en un nombre d'opérations arithmétiques polynomial :

$$O(n^4 \log B)$$

En remarquant que k est incrémenté uniquement lorsque les invariants sont vérifiés, et avec la remarque du cas $k = n+1$, l'algorithme fournit effectivement une base réduite.

Enfin, la force de l'algorithme LLL est sa complexité polynomiale pour trouver une base vecteurs courts, malgré le caractère NP-Difficile de la recherche des vecteurs les plus courts du réseau.

Théorème fondamental du RSA

Soient deux nombres premiers distincts p et q, et posons $n = pq$. Soient $c, d \in \mathbb{Z}$ tels que :

$$cd \equiv 1 \pmod{\varphi(n)}$$

où φ désigne l'indicateur d'Euler. Montrons que pour tout $t \in \mathbb{Z}$, on a :

$$t^{cd} \equiv t \pmod{n}$$

Les nombres p et q étant premiers et distincts, on a :

$$\varphi(n) = (p-1)(q-1)$$

Il existe un entier $k \in \mathbb{Z}$ tel que :

$$cd = 1 + k\varphi(n)$$

Soit $t \in \mathbb{Z}$. Pour prouver que $t^{cd} \equiv t \mod n$, il suffit de montrer que :

$$t^{cd} \equiv t \mod p \quad \text{et} \quad t^{cd} \equiv t \mod q$$

(en vertu de l'isomorphisme $\mathbb{Z}/n\mathbb{Z} \simeq \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ par le théorème des restes chinois).

Prouvons, par exemple, que $t^{cd} \equiv t \mod p$ (le raisonnement modulo q est analogue).

— Si $t \not\equiv 0 \mod p$, alors $t \in (\mathbb{Z}/p\mathbb{Z})^*$. D'après le petit théorème de Fermat, on a :

$$t^{p-1} \equiv 1 \pmod{p}$$

Donc :

$$t^{cd} = t^{1+k(p-1)(q-1)} = t \cdot \left(t^{(p-1)}\right)^{k(q-1)} \equiv t \cdot 1^{k(q-1)} \equiv t \pmod{p}$$

— Si $t \equiv 0 \mod p$, alors :

$$t^{cd} \equiv 0 \equiv t \pmod{p}$$

Ainsi, dans tous les cas, on a bien $t^{cd} \equiv t \mod p$, et de même $t^{cd} \equiv t \mod q$, donc par le théorème des restes chinois :

$$t^{cd} \equiv t \pmod{n}$$

Remarque : L'application $g : \mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}$, définie par $g(t) = t^c$, s'appelle une *fonction de chiffrement*.

L'application $f(t) = t^d$ s'appelle une *fonction de déchiffrement*.

L'exercice affirme que :

$$f \circ g(t) = t$$

On peut donc chiffrer un message (représenté par un élément $t \in \mathbb{Z}/n\mathbb{Z}$) avec g, puis le déchiffrer avec f. Le couple (n, c) est appelé *clé publique*, tandis que l'entier d est la *clé secrète*.

La sécurité de ce système repose sur le fait que, connaissant la clé publique, il est très difficile de déterminer d. En effet, une méthode consiste à factoriser n pour retrouver p et q, ce qui reste, à ce jour, impossible lorsque p et q sont grands (typiquement, en 2020, de l'ordre de 150 à 200 chiffres).

Le record de factorisation d'un entier sans forme particulière est un nombre de 250 chiffres, obtenu en 2020 après 2700 années de calcul distribué.

Ainsi, tout le monde peut chiffrer, mais seuls ceux qui connaissent la clé secrète peuvent déchiffrer.

Ce système de chiffrement est apparu en 1976. Il est appelé **RSA** (du nom des inventeurs :*Rivest, Shamir, Adleman*) et il est encore couramment utilisé aujourd'hui car il est extrêmement robuste.

Son apparition a relancé l'intérêt porté aux algorithmes de factorisation et de primalité.

Algorithme RSA : génération, chiffrement, déchiffrement

L'algorithme RSA se compose de trois étapes principales : la génération de clés, le chiffrement et le déchiffrement. Le tableau ci-dessous résume ces étapes de manière structurée.

Étape	**Description**
Entrées	Deux grands premiers p, q Un exposant public e tel que $\gcd(e,(p-1)(q-1))=1$ Un message $t \in \mathbb{Z}/n\mathbb{Z}$
Sorties	Texte chiffré c et message déchiffré t
Initialisation	Calcul de $n \leftarrow p \cdot q$ Calcul de $\varphi(n) \leftarrow (p-1)(q-1)$ Calcul de la clé secrète $d \leftarrow e^{-1} \mod \varphi(n)$ Clé publique $\leftarrow (n,e)$, Clé privée $\leftarrow d$
Chiffrement	À partir du message clair $t \in \mathbb{Z}/n\mathbb{Z}$ Calcul du texte chiffré : $c \leftarrow t^e \bmod n$
Déchiffrement	À partir du texte chiffré c Calcul du message déchiffré : $t \leftarrow c^d \bmod n$ Par construction : $t^{ed} \equiv t \mod n$
Retour	Message original t, texte chiffré c

TABLE 1 – Algorithme RSA — résumé formel des étapes

Attaques sur RSA

Une première attaque

Supposons que l'on trouve (par magie) un nombre qui est multiple de p mais pas de q. Alors en calculant le PGCD de ce nombre avec n, on trouvera... p. C'est embêtant.

Deuxième attaque : exposant privé petit (Wiener)

Cette attaque repose sur deux approches complémentaires : les fractions continues (théorie des approximations) et les réseaux (heuristique par vecteurs courts).

1. Approche par fractions continues (Wiener, 1990)

Soit $\alpha > 0$, avec son développement en fraction continue donné par :

$$\alpha = [q_0; q_1, q_2, q_3, \dots]$$

où :

$$q_0 = \lfloor \alpha \rfloor, \quad \alpha = q_0 + (\alpha - q_0) = q_0 + \frac{1}{\alpha_1}, \quad \alpha_1 = \frac{1}{\alpha - q_0}$$

et on poursuit récursivement :

$$\alpha = q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\dots}}}$$

On note $\left(\frac{a_k}{b_k}\right)$ les *convergents* associés à α.

Théorème de meilleure approximation (Dirichlet) :
Si

$$\left|\alpha - \frac{a}{b}\right| < \frac{1}{2b^2}, \quad \text{alors } \frac{a}{b} \text{ est un des convergents } \frac{a_k}{b_k}$$

Application à RSA :
On suppose que :

$$n = pq, \quad e \ll n, \quad p < q < 2p, \quad d < \frac{1}{3} n^{1/4}$$

et on part de l'équation fondamentale :

$$ed - k\varphi(n) = 1 \Rightarrow \left|\frac{e}{\varphi(n)} - \frac{k}{d}\right| = \frac{1}{d\varphi(n)}$$

En posant $\alpha = \frac{e}{n}$, on a :

$$\left|\frac{e}{n} - \frac{k}{d}\right| < \frac{1}{2d^2}$$

D'après le théorème précédent, $\frac{k}{d}$ est donc un convergent de la fraction continue de α. Il suffit de tester les convergents $\frac{a_k}{b_k}$ jusqu'à retrouver le bon d.

2. Approche par réseaux (Heuristique avec vecteurs courts, SVP)

On suppose ici encore que d est petit, et que $e = O(N)$. Si p et q sont équilibrés :

$$\varphi(N) = N + O(\sqrt{N})$$

On part de :

$$ed = 1 + k\varphi(N) = 1 + k(N + O(\sqrt{N})) \Rightarrow \ell = ed - kN = O(d\sqrt{N})$$

On construit un réseau $L \subset \mathbb{Z}^2$ engendré par :

$$\begin{bmatrix} e & \sqrt{N} \\ N & 0 \end{bmatrix}$$

Alors le vecteur :

$$\vec{\ell} = d \cdot \text{1ère ligne} - k \cdot \text{2e ligne} = (\ell, d\sqrt{N})$$

a pour norme :

$$\|\vec{\ell}\| \approx d\sqrt{N}$$

et le volume du réseau est :

$$\text{vol}(L) = \sqrt{N^3}$$

D'après le théorème de Minkowski et l'heuristique SVP (Shortest Vector Problem), le plus petit vecteur non nul du réseau a une norme $\lesssim N^{3/4}$. Donc si $d \leq N^{1/4}$, alors $\vec{\ell}$ est probablement le plus court vecteur. On peut ainsi retrouver d en résolvant un problème de vecteur court dans un réseau.

Conclusion : les deux approches (fractions continues et réseaux) exploitent le fait que si d est trop petit (typiquement $d < n^{1/4}$), alors la relation $ed \equiv 1 \mod \varphi(n)$ induit des approximations rationnelles ou géométriques exploitables permettant de retrouver la clé privée.

Le cas général : attaque de Coppersmith

Cette partie décrit la méthode, dite de Coppersmith, qui permet de trouver les petites racines modulaires dún polynôme. Comme expliqué en 1.3 cela nous permettra déxploiter des failles de RSA. On introduit donc dans cette partie les objets suivants : $P(x) = \sum_{k=0}^{d} a_k x^k \in \mathbb{Z}[x]$ un polynôme unitaire, $X \in \mathbb{N}^*$ et on suppose quíl existe $x_0 \in \mathbb{Z}$ tel que $x_0 < X$ et $P(x_0) \equiv 0[\mathbb{N}]$. Lóbjectif est de trouver un tel entier x_0.

Plus précisément, on síntéresse aux résultats de N. Howgrave-Graham qui a repris et simplifié la méthode décrite par Coppersmith.

L'idée directrice est de transformer la recherche des solutions de $P(x) \equiv 0[\mathbb{N}]$ en la recherche des racines entières dún bon polynôme $R(x)$. Ce dernier problème est nettement plus simple à résoudre. En effet, il suffit d'appliquer la méthode de Newton et essayer les entiers les plus proches des solutions approchées trouvées.

Théorème de Howgrave-Graham et conséquence

On veut faire le passage de l'équation modulo N à la recherche de racines entières dún polynôme.

Définition 11. *Pour $Q(x) = \sum_{k=0}^{d} q_k x^k$, on note le vecteur ligne $b_Q = (q_0, q_1 X, q_2 X^2, \ldots, q_d X^d)$. De manière réciproque, d'un vecteur quelconque on peut déduire un polynôme associé à ce vecteur de cette manière.*

De manière équivalente, on construit un isomorphisme entre $\mathbb{R}_d[X]$ et $\mathbb{R}^{d+1}$.

Théorème 3 (Howgrave-Graham)**.** *Si $x_0 < X$ est une solution de $P(x) \equiv 0[N]$ avec $\|b_P\| < \frac{N}{\sqrt{d+1}}$ alors $P(x_0) = 0$.*

Propriété 3. *Le réseau engendré par $(b_{G_0}, \ldots, b_{G_{d-1}}, b_P)$ est de dimension $d+1$ et a pour déterminant :*

$$\det(L) = X^{\frac{d(d+1)}{2}} N^d$$

Notons G le polynôme associé à b_1 où $(b_1, \ldots, b_{d+1})$ est la base réduite donnée par l'algorithme LLL à partir de $(b_{G_0}, \ldots, b_{G_{d-1}}, b_P)$.

Théorème 4 (4.2)**.** *Soit G et P les polynômes construits comme précédemment. On suppose que $X < \frac{1}{\sqrt{2(d+1)}} N^{\frac{2}{d(d+1)}}$. Alors si x_0 est une solution de $P(x) \equiv 0[N]$ avec $|x_0| < X$, alors x_0 est une solution de $G(x) = 0$ sur $\mathbb{Z}$.*

Méthode de Coppersmith

La méthode de Coppersmith est fonctionnelle de manière similaire à celle de la partie précédente. La différence est le choix de la base de polynômes pour construire le réseau. La dimension du réseau sera aussi plus importante.

Avant de caractériser la base de polynômes considérée commençons par énoncer le théorème voulu.

Théorème 5 (Coppersmith)**.** *Soit $0 < \varepsilon < 0{,}18(1 - \frac{1}{d})$. On suppose $X < \frac{1}{2} N^{\frac{1}{d} - \varepsilon}$. Si x_0 est une solution de $P(x) \equiv 0[N]$ telle que $|x_0| < X$ alors x_0 peut être retrouvé en un temps polynomial.*

Pour construire la base utilisée dans la méthode de Coppersmith on introduit un entier $h = \left\lceil \frac{d-1}{\varepsilon} + \frac{1}{2} \right\rceil$.

Définition 12. *Soit $0 \leq i \leq d$ et $0 \leq j < h$. On note :*

$$G_{i,j}(x) = N^{h-1-j} P^j(x) x^i$$

Si x_0 est tel que $P(x_0) \equiv 0[N]$ alors $G_{i,j}(x_0) \equiv 0[N^h]$. En cherchant des racines modulo N^h, on augmente la borne de droite dans le théorème de Howgrave-Graham, ce qui rend plus facile le passage à une équation sur les entiers.

Le bon choix de ces polynômes relève aussi des majorations astucieuses obtenues lors des calculs de bornes dans la démonstration du théorème énoncé.

Pseudo-code de l'algorithme LLL

Algorithm 1 Algorithme 1 LLL

Input : Une base $b_1, \ldots, b_n$ d'un réseau L
Output : La base $b_1, \ldots, b_n$ transformée en une base réduite

for $i = 1$ **to** n **do**
    $b_i^* = b_i$
    for $j = 1$ **to** $i - 1$ **do**
        $\mu_{i,j} = \dfrac{\langle b_i, b_j^* \rangle}{B_j}$
        $b_i^* = b_i^* - \mu_{i,j} b_j^*$
    end for
    $B_i = \|b_i^*\|^2$
end for
$k \leftarrow 2$
while true **do**
    Appliquer **taille**$(k, k-1)$ ▷ (*)
    if $\frac{3}{4} B_{k-1} > B_k + \mu_{k,k-1}^2 B_{k-1}$ **then**
        Appliquer **lovasz**(k)
        Aller à (*)
    end if
    for $l = k - 2$ **down to** 1 **do**
        Appliquer **taille**(k, l)
    end for
    if $k = n$ **then**
        Arrêter l'algorithme
    end if
    $k \leftarrow k + 1$
    Aller à (*)
end while

Algorithm 2 Algorithme 2 LOVASZ

Input : Un entier k

Output : Fait l'échange de b_k et b_{k-1} et calcule les nouveaux coefficients de Gram–Schmidt

$\mu \leftarrow \mu_{k,k-1}$
$B \leftarrow B_k + \mu^2 B_{k-1}$
$\mu_{k,k-1} \leftarrow \dfrac{\mu B_{k-1}}{B}$
$B_k \leftarrow \dfrac{B_k B_{k-1}}{B}$
$B_{k-1} \leftarrow B$
Échanger b_k et b_{k-1}
for $j = 1$ **to** $k-2$ **do**
    Échanger $\mu_{k-1,j}$ et $\mu_{k,j}$
end for
for $i = k+1$ **to** n **do**
    $\mu' \leftarrow \mu_{i,k-1}$
    $\mu_{i,k-1} \leftarrow \mu_{i,k} - \mu_{k,k-1}\mu' + (1 - \mu_{k,k-1})\mu_{i,k}$
    $\mu_{i,k} \leftarrow \mu' - \mu_{k,k-1}\mu_{i,k}$
end for
if $k > 2$ **then**
    $k \leftarrow k - 1$
end if

Algorithm 3 Algorithme 3 TAILLE

Input : Un couple (k, l)

Output : Un vecteur b_k et des valeurs $\mu_{k,j}$ avec $j = 1, \ldots, l-1$ et $|\mu_{k,l}| \leq \dfrac{1}{2}$

if $|\mu_{k,l}| > \dfrac{1}{2}$ **then**
    $b_k \leftarrow b_k - \lfloor \mu_{k,l} \rceil b_l$
    for $j = 1$ **to** $l-1$ **do**
        $\mu_{k,j} \leftarrow \mu_{k,j} - \lfloor \mu_{k,l} \rceil \mu_{l,j}$
    end for
    $\mu_{k,l} \leftarrow \mu_{k,l} - \lfloor \mu_{k,l} \rceil$
end if

C Preuves

Preuve :

Soit $2 \leq i \leq n$. Par inégalité triangulaire dans (2), on a

$$\|b_i^*\|^2 \geq \left(\tfrac{3}{4} - \mu_{i,i-1}^2\right) \|b_{i-1}^*\|^2$$

et d'après (1), $\|b_i^*\|^2 \geq \frac{1}{2}\|b_{i-1}^*\|^2$. On déduit alors d'une récurrence que pour tout $1 \leq j \leq i \leq n$,

$$2^{1-i}\|b_j^*\|^2 \leq \|b_i^*\|^2 \quad (*)$$

Ainsi, pour $1 \leq i \leq n$,

$$\begin{aligned}
\|b_i\|^2 &= \|b_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2 \|b_j^*\|^2 \\
&= \|b_i^*\|^2 + \sum_{j=1}^{i-1} |\mu_{i,j}|^2 \|b_j^*\|^2 \quad \text{(par orthogonalité)} \\
&\leq \left(1 + \frac{1}{4}\sum_{j=1}^{i-1} 2^{j-i}\right) \|b_i^*\|^2 \quad \text{(d'après (*) et (1))} \\
&= \left(1 + \frac{1}{4}(2^{i-1} - 1)\right) \|b_i^*\|^2 \\
&\leq 2^{i-1}\|b_i^*\|^2 \qquad (**)
\end{aligned}$$

Donc, d'après (*) et (**), pour $1 \leq j \leq i \leq n$,

$$\|b_j^*\|^2 \leq 2^{j-i}\|b_i^*\|^2 \leq 2^{j-1}\|b_i\|^2 \quad \text{; d'où (3).}$$

Puisque le déterminant est multilinéaire et alterné, on a

$$\det(L) = \det(b_1, \ldots, b_n).$$

Puis, puisque $b_1, \ldots, b_n$ est orthogonale $\Rightarrow \det(L) = \prod_{i=1}^n \|b_i^*\|$. De plus, puisque la projection orthogonale contracte les normes : pour tout $1 \leq i \leq n$, avec (3) on a

$$\det(L) \leq \prod_{i=1}^n \|b_i\| \leq \prod_{i=1}^n 2^{\frac{n-i}{2}} \|b_i^*\| = 2^{\frac{n(n-1)}{4}} \det(L)$$

D'où (4). Enfin, d'après (3) avec $j = 1$, et en passant au produit pour $i = 1, \ldots, n$, on retrouve (5).

Preuve : Terminaison de LLL

Pour démontrer la terminaison de LLL nous introduisons le déterminant de Gram : $d_i = \det((b_j, b_j)_{1\leq j\leq i}) = \text{Gram}(b_1, \ldots, b_i)$. Notons de plus que $b_i = b_i^* + a$, avec a le projeté orthogonal de b_i sur $\text{Vect}(b_1, \ldots, b_{i-1})$.
On montre alors par bilinéarité du produit scalaire et non-linéarité du déterminant que $\text{Gram}(b_1, \ldots, b_i) = d_i = \prod_{j=1}^{i} \|b_j^*\|^2$. Or par orthogonalité, $\text{Gram}(b_1, \ldots, b_i)$ est diagonale par blocs ce qui permet de trouver que $d_i = \det(b_i)^2$.
On montre alors par récurrence que $d_i = \prod_{j=1}^{i} \|b_j^*\|^2$.

Posons ensuite $D = \prod_{i=1}^{n} d_i$. D'après l'expression de d_i, la valeur de D est modifiée si celle d'un $\|b_i^*\|^2$ l'est. On remarque dans le pseudo-code que ces valeurs ne sont modifiées que dans l'appel de la fonction **Lovasz** : il existe $a < \frac{3}{4}$ tel que en appelant $\|b_k^*\|^2 = a\|b_k^*\|^2$ et $\|b_{k-1}^*\|^2 = \frac{1}{a}\|b_{k-1}^*\|^2$, les autres $\|b_i^*\|^2$ sont inchangés. Il en découle que $d_k - 1$ est réduit d'un facteur $< \frac{3}{4}$ et que les autres d_i sont inchangés.
Et donc que D est réduit d'un facteur $< \frac{3}{4}$.

Nous admettons que D est minoré par un réel strictement positif. Cela implique que l'on peut appliquer **Lovasz** qu'un nombre fini de fois. Or, puisque k est décrémenté dans **Lovasz** mais incrémenté sinon, cela implique que l'algorithme se termine.

a. Cela découle d'une minoration de la taille du plus court vecteur non nul d'un réseau.

Preuve : Théorème Howgrave-Graham

Il suffit de remarquer, en vertu de l'inégalité de Cauchy–Schwarz, que

$$P(x_0) \leq \sum_{k=0}^{d} |a_k| X^k \leq \sqrt{d+1} \sqrt{\sum_{k=0}^{d} (a_k X^k)^2} = \sqrt{d+1} \|b_{P(x)}\| < N$$

Preuve : Propriété 3.1

La famille $(G_0, \ldots, G_{d-1}, P)$ est libre par théorème des degrés étagés, donc $b_{G_0}, \ldots, b_{G_{d-1}}, b_P$ qui est l'image par l'isomorphisme qui fait la correspondance avec les vecteurs l'est aussi.
De plus, la famille de vecteurs est représentée par la matrice triangulaire M suivante :

$$M = \begin{pmatrix} X^d & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & X^d & 0 \\ a_0 & \cdots & a_{d-1} & X^t \end{pmatrix}$$

dont le déterminant est $\det(M) = \det(L) = X^{d\ell+t}$.

Preuve :

D'après la majoration (5) et la propriété 2.1, on a

$$\|b_0\| \leq 2^{\frac{\ell}{4}} \det(L)^{1/\ell} = 2^{\frac{\ell}{4}} X^{d+t/\ell}$$

D'après le théorème de Howgrave-Graham, une condition suffisante pour avoir le résultat voulu est que

$$2^{\ell/4} X^{d+t/\ell} < \sqrt{d+1} \cdot N^{1/(d+1)}$$

ce qui est équivalent à

$$X < \frac{1}{2^{\ell/4}} \cdot \frac{N^{1/(d+1)}}{\sqrt{d+1}}$$

Preuve : Théorème Coppersmith

En ordonnant les G_j par degrés croissants, la matrice associée aux b_{G_j} est triangulaire avec les coefficients diagonaux $X^d, \ldots, X^d, X^t$. Donc :

$$\det(L) = X^{d\ell+t} \quad \Rightarrow \quad \|b_0\| \leq 2^{\ell/4} \cdot X^{d+t/\ell}$$

Or, toujours en notant b_0 le vecteur de la base réduite associée à $(h_0, \ldots, h_\ell)$, on a la majoration (5) qui donne :

$$\|h_0\| \leq \|b_0\| < N^{1/d}$$

En appliquant le théorème de Howgrave-Graham, une condition suffisante pour passer à l'équation sur x, telle que $f(x) \equiv 0 \mod N$, est donc :

$$2^{\ell/4} X^{d+t/\ell} < N^{1/d}$$

En notant $\alpha(x) = \sqrt{d(d+1)}$, cette inégalité devient :

$$X < \frac{1}{2^{\ell/4}} \cdot \frac{N^{1/d}}{\alpha(x)}$$

Pour avoir une borne théorique, il suffit que $\ell = d = \lceil \frac{1}{\varepsilon} \rceil \Rightarrow \alpha(x) \leq 2$. Une étude de $2^{\ell/4}(1 + \frac{t}{d\ell})$ montre que le coefficient constant est au plus $2^{1/4}(1+\varepsilon)$, ce qui donne :

C'est le résultat classique de l'algorithme LLL (polynomial en ℓ et en $\log N$)

Preuve : Théorème RSA (autre)

Soit $m \in \mathbb{N}$. Par hypothèse il existe $k \in \mathbb{N}$ tel que $ed = 1 + k\phi(N)$. Montrons que $m^{ed} - m = 0[p]$. Si $p \mid m$, c'est immédiat.
Sinon, d'après le petit théorème de Fermat, on a

$$m^{ed} = m^{1+k\phi(N)} = m(m^{p-1})^{k(q-1)} = m[N].$$

Alors de même $m^{ed} - m = 0[q]$, et puisque $p \wedge q = 1$, d'après le théorème des restes chinois,

$$m^{ed} - m = 0[pq].$$

Soit $m^{ed} = m[N]$.