COMPLEXITÉ DE L'ALGORITHME LLL POUR LA VALEUR OPTIMALE DU PARAMÈTRE D'APPROXIMATION

ALI AKHAVI

Résumé. Ce rapport étudie la complexité dans le pire des cas de l'algorithme LLL, pour la valeur optimale $t = 1$ du paramètre d'approximation t. L'algorithme trouve alors la meilleure base réduite possible au sens de Lovász, mais le résultat général obtenu par les inventeurs de l'algorithme ne s'applique pas dans ce cas limite. Lagarias et Odlyzko [LO83] citent des expériences qui semblent montrer qu'en moyenne l'algorithme reste polynomial. À ma connaissance, aucune réponse n'a été apportée jusqu'ici à cette question et le problème restait ouvert, même en petite dimension.
J'étudie l'algorithme LLL optimal lorsque la dimension du réseau est fixée, et, grâce à des arguments combinatoires, je montre qu'il fonctionne dans le pire des cas toujours en temps polynomial par rapport à la taille de la base d'entrée. En particulier, en dimension 3, je montre que le nombre d'itérations reste linéaire en la taille de la base d'entrée, et je donne une borne explicite pour le nombre maximum d'itérations. J'obtiens également une caractérisation précise de la base de sortie.

Abstract. This report deals with the open problem of the complexity of the LLL algorithm in the case that the parameter t has its extreme value 1. This case is of interest because it gives the strictest condition on the basis. Statistical results reported by Lagarias and Odlyzko [LO83] seem to show that the optimal algorithm remains polynomial in average. No result is established about the optimal LLL algorithm, even for fixed small dimension (greater than 2). Here we prove that LLL optimal algorithm remains polynomial in the size of the problem for fixed dimension n. Moreover, when the dimension is fixed to 3, we give a linear bound for the number of iterations, and we characterize precisely the output basis.

1. Introduction

Un *réseau euclidien* est l'ensemble des combinaisons linéaires entières d'un système libre de p vecteurs de $\mathbb{R}^n$. Les p vecteurs forment alors une base du réseau et leur nombre est la dimension du réseau. Un même réseau peut ainsi être engendré par de multiples bases. *Réduire un réseau* consiste à trouver une base qui engendre le même réseau et qui soit de bonne qualité vis-à-vis de la structure euclidienne: les vecteurs de la base doivent être assez courts et en même temps assez orthogonaux. De nombreux problèmes issus de la théorie des nombre, de la cryptographie ou du calcul formel se formalisent aisément grâce à des réseaux euclidiens, et la solution du problème posé s'obtient dès que l'on dispose d'une base de bonne qualité. En 1982, pour la première fois, H.W. Lenstra, A.K. Lenstra et L. Lovász ont introduit un algorithme efficace ppour la réduction [LLL82]. Il s'agit du célèbre algorithme LLL.

L'algorithme LLL est un algorithme itératif qui à partir d'une base quelconque d'un réseau euclidien, construit une base réduite au sens de t-Lovász. Le paramètre d'approximation t doit être choisi dans l'intervalle $[1, \sqrt{3}]$. Ce rapport traite de la complexité dans le pire des

cas de l'algorithme LLL lorsque $t = 1$. Plus exactement, on majorera le nombre maximum d'itérations effectuées par cet algorithme.
Les résultats qui vont suivre sont valables pour des réseaux entiers (et peuvent facilement être généralisés à des réseaux rationnels). Un réseau entier $\mathcal{L}$ est donné par une base $(b_1, b_2, ..., b_n)$. Si M désigne la longueur (euclidienne) maximale des vecteurs de cette base

$$M = \max_{i \in \{1,2,..,n\}} |b_i|,$$

la taille de l'entrée de l'algorithme LLL est alors en $(n, \log M)$.
Les inventeurs de l'algorithme LLL ont montré [LLL82] que l'algorithme LLL est polynomial en la taille de l'entrée $(n, \log M)$, pour tout $t \neq 1$. Leur preuve échoue si $t = 1$ et le problème de la complexité de LLL($t = 1$) reste ouvert.
Kannan a presenté en 84 [Kan83] un algorithme qui utilise l'algorithme LLL (avec $t > 1$) et qui construit lorsque la dimension n est fixée, en temps polynomial en $\log M$ une base réduite au sens de Korkine-Zolotareff. La réduction de Korkine-Zolotareff étant plus fine que celle de $t-$Lovász même avec $t = 1$, il semblait naturel que l'algorithme LLL($t = 1$) soit également polynomial en $\log M$ lorsque la dimension n est fixée. C'est l'objet de ce chapitre que de le montrer. Plus précisémment, nous montrons que le nombre maximum d'itérations de LLL($t = 1$) est majoré par un polynôme en $\log M$ (chaque itération ne comporte que des opérations très simples et dont les temps d'exécution sont linéaires en $\log M$).
Plusieurs raisons motivent ce travail:

- C'est lorsque $t = 1$ que l'algorithme LLL peut être considéré comme une généralisation de l'algorithme bien connu de Gauss en dimension 2.
- L'algorithme LLL(t) est un algorithme d'approximation et $t = 1$ permet d'obtenir la meilleure approximation possible. Les critères habituels qui permettent de mesurer la qualité d'une base b sont son défaut d'orthogonalité $\rho(b)$ et ses défauts de longueur $\mu_i(b)$. Pour la base de sortie de l'algorithme LLL(t),

  $$\rho(b) \leq s^{\frac{n(n-1)}{2}} \quad \text{et} \quad \mu_i \leq s^{n-1} \quad \text{avec} \quad s = \sqrt{\frac{4t^2}{4-t^2}}.$$

  Plus t est proche de 1, plus les majorations ci-dessus sont fines et la base de sortie est garantie d'une meilleure qualité.
- Même si la réponse apportée ici n'est que très partielle par rapport à la question ouverte de la complexité de LLL($t = 1$), à notre connaissance il n'existait jusqu'alors aucune preuve montrant que l'algorithme LLL($t = 1$) etait polynomial, ne serait-ce qu'en dimension 3. (En dimension 2, la complexité polynomiale a été établie par Lagarias [Lag82] et la meilleure borne donnée par Vallée [Val91].)

Dans la suite, la première partie donne une formulation de l'algorithme LLL et introduit quelques notations utiles. Nous y définissons en particulier ce qu'est *une situation de $t-$quasi-réduction* durant une exécution de l'algorithme LLL. La troisième partie rappelle les résultats de la dimension 2. Une situation de $t-$quasi-réduction entraîne alors la fin de l'exécution de l'algorithme en 2 itérations (Lemme 1). Dans la quatrième partie, nous établirons par une étude combinatoire qu'une situation de $t-$quasi-réduction entraîne la fin de l'exécution de l'algorithme en au plus 9 itérations (le Théorème de cette partie). Nous en déduirons une borne explicite linéaire en $\log M$ pour le nombre maximum d'itérations de l'algorithme LLL($t = 1$), lorsque la dimension est 3. Quelques preuves techniques de l'étude combinatoire sont détaillées dans la cinquième partie. La dernière partie généralise les raisonnements précédents au cas de la dimension n fixée.

2. Description générale de l'algorithme LLL

Soit $\mathcal{L}$ un réseau donné par une base $(b_1, ..., b_n)$ de $\mathbb{R}^n$. On note $(b_1^*, ..., b_n^*)$, l'orthogonalisée de Schmidt de $(b_1, ..., b_n)$ et $m = (m_{ij})$ la matrice exprimant $(b_1, ..., b_n)$ dans $(b_1^*, ..., b_n^*)$.
En notant $(u)_{\perp H}$ le projeté du vecteur u orthogonalement au sous-espace vectoriel H et $\langle u_1, u_2, ..., u_r \rangle$ l'espace vectoriel engendré par la famille de vecteurs $(u_1, u_2, ..., u_r)$, nous avons:
(1)

$$\left\{ \begin{array}{lcl} b_1^* & = & b_1 \\ b_2^* & = & b_{2\perp\langle b_1 \rangle} \\ \vdots & & \\ b_i^* & = & b_{i\perp\langle b_1, \cdots, b_{i-1}\rangle} \\ \vdots & & \\ b_n^* & = & b_{n\perp\langle b_1, \cdots, b_{n-1}\rangle} \end{array} \right. \quad \text{et} \quad m = \begin{array}{c} \\ b_1 \\ b_2 \\ \vdots \\ b_i \\ b_{i+1} \\ \vdots \\ b_n \end{array} \begin{array}{c} \begin{array}{ccccccc} b_1^* & b_2^* & \cdots & b_i^* & b_{i+1}^* & \cdots & b_n^* \end{array} \\ \left(\begin{array}{ccccccc} 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ m_{21} & 1 & \ddots & & & & \vdots \\ \vdots & \vdots & \ddots & \ddots & & & \vdots \\ \cdots & \cdots & \cdots & 1 & 0 & & \vdots \\ \cdots & \cdots & \cdots & m_{i+1,i} & 1 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & & \ddots & 0 \\ m_{n1} & m_{n2} & \cdots & \cdots & \cdots & \cdots & 1 \end{array} \right) \end{array}.$$

Le procédé même d'orthogonalisation de Gramm-Schmidt montre que la matrice m est triangulaire avec une diagonale formée de 1. Cette orthogonalisation s'effectue en temps polynomial par rapport à la taille $(n, \log M)$ de l'entrée.
La base $(b_1, ..., b_n)$ est dite *propre* si dans la matrice m de passage de la base (b) à sa base orthogonalisée (b^*),

(2) $$-\frac{1}{2} < m_{ij} \leq \frac{1}{2} \quad \text{pour tout } i > j.$$

Il existe un algorithme très simple qui étant donné une base quelconque du réseau la rend propre en un temps polynomial en $(n, \log M)$.[1]
L'algorithme LLL(1) est une généralisation possible à toute dimension de sa version en dimension 2, appelé l'algorithme de Gauss, et dont tous les résultats de complexité sont maintenant connus.

Définition 1. La base $(b_1, ..., b_n)$ est réduite ou LLL($t = 1$)-réduite si elle vérifie les deux conditions suivantes:
(i) $(b_1, ..., b_n)$ est propre.
(ii) $\forall i \in \{1, 2, \cdots, n-1\} \qquad |b_i^*|^2 \ \leq \ m_{i+1,i}{}^2 |b_i^*|^2 + |b_{i+1}^*|^2$.

Si l'on note, pour tout i appartenant à $\{1, 2, \cdots, n-1\}$:

$$\left\{ \begin{array}{lcl} u_i & = & b_i^* \\ v_i & = & m_{i+1,i} b_i^* + b_{i+1}^* \end{array} \right. \quad \text{et la } \textit{boîte } B_i \text{ le couple } (u_i, v_i),$$

par définition, la base $(b_1, ..., b_n)$ est réduite si elle est propre et les boîtes B_i sont réduites, pour tout i appartenant à $\{1, 2, \cdots, n-1\}$.

1. Pour i variant entre 0 et n, il suffit de translater correctement le i–ème vecteur de la base, parallèlement aux $i - 1$ premiers vecteurs.

Définition 2. Soit t un réel vérifiant $t > 1$. On dira que la base $(b_1, \cdots, b_n)$ est *t-quasi-réduite* lorsqu'elle vérifie les conditions suivantes:

(i) la base $(b_1, \cdots, b_{n-1})$ est propre.

(ii) Pour $1 \leq i \leq n-2$, toutes les boîtes B_i sont réduites.

(iii) La dernière boîte B_{n-1} n'est pas réduite mais

$$|m_{n,n-1}| < \frac{1}{2} \quad \text{et} \quad \frac{1}{t^2}|b_{n-1}^*|^2 \leq m_{n,n-1}{}^2|b_{n-1}^*|^2 + |b_n^*|^2.$$

(La dernière boîte est $t-$Gauss réduite ou simplement $t-$réduite.)

Autement dit, si la base $(b_1, \cdots, b_{n-1})$ est réduite et la base $(b_1, \cdots, b_n)$ est non-réduite alors pour tout $t > 1$ tel que la dernière boîte est $t-$réduite, la base $(b_1, \cdots, b_n)$ est $t-$quasi-réduite.

Voici un énoncé possible de l'algorithme LLL(1):

Algorithme LLL

Donnée : une base b d'un réseau $\mathcal{L}$ de dimension n de $\mathbb{R}^p$.
Résultat : une base b de $\mathcal{L}$ réduite ou $LLL(1)$ réduite

$i := 1$;
Tant que $i < n$ **répéter**
- Calculer la matrice m.
- **Translater** b_{i+1} parallèlement à b_i pour le rendre le plus court possible:
  Calculer r_i, l'entier le plus proche de $m_{i+1,i}$;
  Faire $b_{i+1} := b_{i+1} - r_i b_i$;
- **Tester si** $|b_i^*|^2 \leq m_{i+1,i}{}^2|b_i^*|^2 + |b_{i+1}^*|^2$
  Si OUI :
  Rendre $(b_1, ..., b_{i+1})$ propre.
  $i := i + 1$;
  Si NON :
  Échanger b_i et b_{i+1};
  Si $i \neq 1$ alors $i := i - 1$;

Dans toute la suite, une itération de l'agorithme LLL est une itération de la boucle *tant que* dans l'énoncé précédent. Chaque itération comporte exactement un test de réduction ou non-réduction d'une boîte B_i $(1 \leq i \leq n-1)$. Le nombre d'itérations de l'algorithme est donc exactement le nombre de tests de réductions des boîtes. Les itérations écoulées entre deux passages successifs de i à la valeur $n-1$ forment *une phase* de l'algorithme.

Par ailleurs, pendant une exécution de l'algorithme LLL, l'indice i de l'énoncé ci-dessus est à un instant donné, le plus grand entier k, tel que $(b_1, ..., b_k)$ soit réduite. À l'issue de chaque test, sa valeur évolue d'une des trois façons suivantes.

1. l'indice i peut être incrémenté de 1 (test positif)
2. l'indice i peut être décrémenté de 1 (test négatif)
3. l'indice i peut garder son ancienne valeur, qui est alors nécessairement 1. (i valait déjà 1 avant le test et le test est négatif).

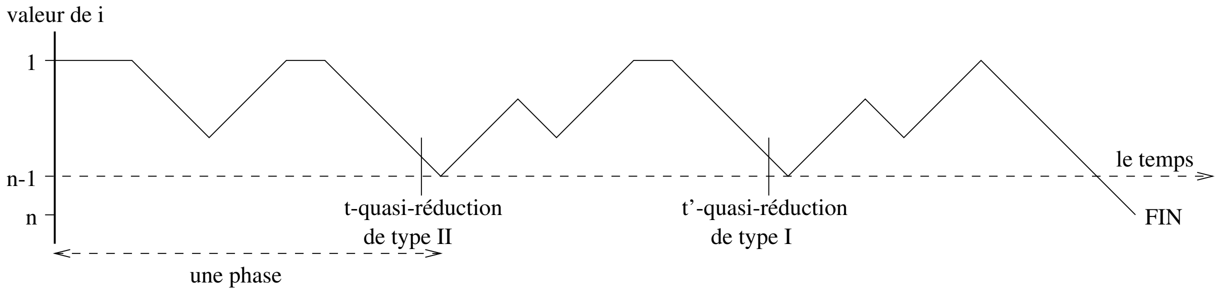

FIG. 1. *Un exemple de la variation de l'indice i représentée sous-forme de marche aléatoire en dimension* 4

Ainsi, la variation de l'indice i pendant une exécution de l'algorithme LLL peut être représentée par une marche aléatoire (figure 1).

Au terme de chaque phase non finale de l'algorithme, l'indice i a la valeur $n-1$ (juste avant le test) et il existe $t > 1$ tel que la base $(b_1, \ldots, b_n)$ est $t-$quasi-réduite. On dit alors qu'on est en *situation de t-quasi-réduction*. Ces situations sont indiquées sur la figure 1.
Par définition, lors d'une situation de quasi-réduction, le test concerne la réduction de la dernière boîte B_{n-1} et le résultat de ce test est négatif.
Nous séparons les situations de t-quasi-réduction en deux types selon qu'entre deux passages successifs à la dernière boîte, on passe plusieurs fois de suite dans la première boîte. Plus précisément, une situation de $t-$quasi-réduction sera dite

de type I, si pendant la phase qui suit, une des 3 conditions équivalentes suivantes est remplie:

(i) l'algorithme ne fait jamais l'échange entre b_1 et b_2.
(ii) il n'existe aucun palier (figure 1).
(iii) il n'y a aucun test négatif, dans la première boîte B_1.

de type II, si pendant la phase qui suit, une des 3 conditions équivalentes suivantes est remplie:

(ii) l'algorithme fait au moins une fois l'échange entre b_1 et b_2.
(ii) il existe au moins un palier (figure 1).
(iii) il y a au moins un test négatif, dans la première boîte.

On notera en abrégé:

$t-$qr.	pour	$t-$quasi-réduction
$t-$qr.I	pour	$t-$quasi-réduction de type I
$t-$qr.II	pour	$t-$quasi-réduction de type II
FIN	pour	fin de l'algorithme

3. Les résultats connus en dimension 2: l'algorithme de Gauss

En dimension 2, une phase de l'algorithme est tout simplement une itération de l'algorithme (une itération de la boucle *tant que*) et le premier test positif signifie la fin de l'algorithme. Pour compter le nombre maximum d'itérations, il suffit donc de compter le nombre de tests négatifs. Nous allons montrer qu'en un nombre polynomial en $\log_t M$ d'itérations, la base (b_1, b_2) sera au moins $t-$quasi-réduite ($t-$quasi-réduite ou réduite). En outre, pour tout t, $1 < t < \sqrt{3}$, une situation de $t-$quasi-réduction entraîne la fin de l'algorithme en 2 itérations. (la première est l'itération durant laquelle la situation de quasi-réduction se produit)

On pose $$d_1 = |b_1|^2 = |b_1^*|^2.$$

À t fixé, avant d'arriver à une situation de $t-$quasi-réduction, lors de chaque test négatif, nous avons $|b_2|^2 < (1/t^2)|b_1|^2$ et b_1 et b_2 seront échangés, ce qui signifie encore que la quantité d_1 décroît d'au moins $1/t^2$. Après k_- itérations et sachant que le réseau est entier,

$$1 \leq d_1 < \frac{1}{(t^2)^{k_-}} M^2$$

Donc le nombre k_- d'itérations pour arriver à un situation de $t-$quasi-réduction est au plus $\log_t M$.
Le lemme suivant permet de conclure dans le cas de la dimension 2.

Lemme 1. *Soit t un réel vérifiant $1 < t < \sqrt{3}$. En dimension 2, une situation de $t-$quasi-réduction entraîne la fin de l'exécution de l'algorithme en 2 itérations de tests négatif, puis positif.*

$$t - qr. \xrightarrow{\quad 1<t<\sqrt{3} \quad} \textit{FIN, en 2 itérations de tests } (-,+).$$

Preuve. Lors d'une situation de $t-$quasi-réduction, la base (b_1, b_2) vérifie

$$(b_1, b_2) \text{ est propre } (|m_{21}| < \frac{1}{2}) \quad \text{et} \quad \frac{1}{t^2}|b_1|^2 \leq |b_2|^2 < |b_1|^2. \tag{3}$$

Avant l'échange, $m = \begin{array}{c c} & \begin{array}{cc} b_1^* & b_2^* \end{array} \\ \begin{array}{c} b_1 \\ b_2 \end{array} & \begin{pmatrix} 1 & 0 \\ m_{21} & 1 \end{pmatrix} \end{array}$.

Après l'échange, $\hat{m} = \begin{array}{c c} & \begin{array}{cc} \hat{b_1^*} & \hat{b_2^*} \end{array} \\ \begin{array}{c} b_2 \\ b_1 \end{array} & \begin{pmatrix} 1 & 0 \\ \hat{m}_{21} & 1 \end{pmatrix} \end{array}$, avec $\hat{m}_{21} = \dfrac{(b_1, b_2)}{|b_2|^2} = \dfrac{(b_1, b_2)}{|b_1|^2} \dfrac{|b_1|^2}{|b_2|^2} = m_{21} \dfrac{|b_1|^2}{|b_2|^2}$

Les relations (3) donnent $\quad |\hat{m}_{21}| < \dfrac{t^2}{2} < \dfrac{3}{2}, \quad$ pour $1 < t < \sqrt{3}$.

Soit encore, $$\lfloor \hat{m}_{21} \rceil \in \{-1, 0, 1\}.$$

Donc le vecteur b_1 sera transformé en $b_1 + \varepsilon b_2$, avec $\varepsilon \in \{-1, 0, 1\}$.

Or $$\begin{cases} |b_1 + \varepsilon b_2|^2 & = & (1 + \varepsilon m_{21})^2 & |b_1^*|^2 & + & \varepsilon^2 & |b_2^*|^2 \\ |b_2|^2 & = & (m_{21})^2 & |b_1^*|^2 & + & & |b_2^*|^2 \end{cases}$$

Et $\qquad \forall \varepsilon \in \{-1, 0, 1\}, \qquad$ on a bien: $|b_1 + \varepsilon b_2|^2 \geq |b_2^*|^2$.

(Si $\varepsilon = 0$, c'est une conséquence de la $t-$qr. Sinon, $\varepsilon^2 = 1$ et comme $|m_{21}| \leq \frac{1}{2}$, on a bien $(1 + \varepsilon m_{21})^2 \geq m_{21}^2$.)
Le test suivant la $t-$quasi-réduction est donc bien positif. ☐

Le nombre maximum d'itérations de l'algorithme de Gauss est donc majoré par $\log_{\sqrt{3}} M + 2$. Cette borne n'est pas optimale, mais c'est ce raisonnement que nous allons généraliser à la dimension 3, dans le prochain paragraphe.

4. Cas de la dimension 3

On procède comme précédemment:
On montre d'abord qu'en un nombre polynomial d'itérations (en $\log_t M$), la base d'entrée (b_1, b_2, b_3) sera $t-$quasi-réduite. Puis, nous verrons (le théorème de ce paragraphe) que pour $1 < t < \sqrt{6/5}$, une base $t-$quasi-réduite sera réduite en au plus 9 itérations. Les raisonnements que nous utilisons donnent par ailleurs une caractérisation précise de la qualité d'une base réduite en dimension 3.

4.1. **D'une base quelconque à la quasi-réduction.** En partant d'une base quelconque, l'algorithme LLL produit une base quasi-réduite en un nombre polynomial d'itérations. Plus exactement,

Proposition 1. *Soit t un réel vérifiant $1 < t < \sqrt{3}$. Le nombre d'itérations effectuées par l'algorithme LLL, pour arriver à la première $t-$quasi-réduction est majoré par $\log_{\sqrt{3}} M + 6\log_t M + 2$.*

Preuve. On note:
$$\begin{cases} d_1 = |b_1^*|^2 = |b_1|^2 \\ d_2 = |b_1^*|^2 |b_2^*|^2 \end{cases}$$

Il est clair que $d_1 < M^2$ et $d_2 < M^4$.
On va compter le nombre maximum k, de tests pour arriver à la $t-$quasi-réduction (chaque itération comporte 1 unique test).
On note respectivement

- k_1^- , le nombre de tests négatifs avec $i = 1$, avant la $t-$quasi-réduction,
- k_2^- , le nombre de tests négatifs avec $i = 2$, avant la $t-$quasi-réduction.(on ne décomptera pas le dernier test négatif), et
- k^+ , le nombre de tests positifs avant la $t-$quasi-réduction (nécessairement avec $i = 1$).

$$\text{Nous avons alors}\quad k = k_1^- + k_2^- + k^+. \tag{4}$$

En dimension 3, une phase de l'algorithme est la suite d'itérations qui s'exécutent entre deux passages successifs de i à la valeur 2. Chaque phase comporte ici exactement un test positif. Donc, en notant p le nombre de phases avant la $t-$quasi-réduction,

$$k^+ = p = k_2^- + 1. \tag{5}$$

Pendant l'exécution d'une phase tous les tests négatifs avec $i = 1$ entraînent une décroissance de la quantité d_1 de $1/3$ sauf éventuellement la dernière (cf. le lemme 1: un réseau (b_1, b_2), $\sqrt{3}-$qr. sera réduite en deux itérations de test $-, +$).
Si $\hat{k}_1^-$ est le nombre de test négatifs qui font décroître la quantité d_1, comme le réseau est entier et cette quantité ne croît jamais, elle vérifie

$$1 \le d_1 < \frac{1}{(\sqrt{3}^2)^{\hat{k}_1^-}} M^2. \quad \text{D'où:}$$

$$k_1^- \le \log_{\sqrt{3}} M + p. \tag{6}$$

Par ailleurs les tests négatifs ou positifs avec $i = 1$, ne font pas varier la quantité d_2, qui est le déterminant de Gramm du réseau (b_1, b_2).

Les tests négatifs avec $i = 2$ font décroître $|b_2^*|^2$ donc d_2 de $1/t^2$. Donc pour un réseau entier, au terme de k_2^- tests négatifs, la quantité d_2 vérifie

$$1 \;\leq\; d_2 \;<\; \frac{1}{(t^2)^{k_2^-}} M^4. \quad \text{D'où:}$$

$$k_2^- \;\leq\; 2 \log_t M \tag{7}$$

Finalement, d'après les équations (4), (5), (6) et (7), le nombre total k d'itérations avant la $t-$quasi-réduction est majoré:

$$k \leq \log_{\sqrt{3}} M + 6 \log_t M + 2. \qquad \square$$

En particulier pour $1 < t^2 \leq 6/5$, $\quad k \leq \log_{\sqrt{3}} M + 6 \log_{\sqrt{\frac{6}{5}}} M + 2.$

4.2. **De la quasi-réduction à la réduction.** Rappelons que pour tout $t > 1$, (b_1, b_2, b_3) est $t-$quasi-réduite lorsqu'elle vérifie les 2 conditions suivantes.

(i) La boîte B_1 est 1-Lovász-réduite (ou encore Gauss-réduite).

(ii) La boîte B_2 est t-Lovász-réduite mais non 1-Lovász-réduite.

Théorème 1. *Soit t un réel vérifiant $1 < t < \sqrt{6/5}$. En dimension 3, une situation de $t-$quasi-réduction entraîne la fin de l'exécution de l'algorithme en au plus 9 itérations.*

$$t - qr. \xrightarrow[\textit{au plus } 9 \textit{ itérations.}]{1<t<\sqrt{6/5}} \textit{FIN.}$$

Preuve. Elle est basée sur le Lemme 2 et la Proposition 2. Si la $t-$quasi-réduction est de type I, le Lemme 2 montre que l'algorithme termine son exécution en 3 itérations. La Proposition 2 prouve par ailleurs qu'une $t-$quasi-réduction de type II entraîne en au plus 6 itérations soit la fin de l'exécution de l'algorithme, soit une nouvelle situation de $t-$quasi-réduction nécessairement de type I et dans ce dernier cas, de nouveau le Lemme 2 permet de conclure.
Toutes les possibilités sont résumées dans le diagramme suivant où au dessus de chaque flèche, sont indiqués les résultats des tests successifs effectués par l'algorithme entre deux situations considérées (le test positif final est omis). À gauche, sont décomptés les nombres totaux d'itérations (ou de tests) après la quasi-réduction, dans chaque cas.

$$\begin{array}{c} \text{non}(t-\text{qr.}) \\ (1 < t^2 < 6/5) \end{array} \Longrightarrow \left\{ \begin{array}{llllllll} \text{FIN} & & & & & & & (1) \\ t-\text{qr.I} & \xrightarrow{-+} & \text{FIN} & & & & & (3) \\ t-\text{qr.II} & \xrightarrow{--+} & \text{FIN} & & & & & (4) \\ t-\text{qr.II} & \xrightarrow{--+} & t'-\text{qr.I} & \xrightarrow{-+} & \text{FIN} & & & (6) \\ t-\text{qr.II} & \xrightarrow{--+} & t'-\text{qr.II} & \xrightarrow{--+} & \text{FIN} & & & (7) \\ t-\text{qr.II} & \xrightarrow{--+} & t'-\text{qr.II} & \xrightarrow{--+} & t''-\text{qr.I} & \xrightarrow{-+} & \text{FIN} & (9) \end{array} \right.$$

$\square$

Remarque: Une étude combinatoire plus fine du cas

$$t-\text{qr.II} \longrightarrow t'-\text{qr.II}$$

permettrait probablement de remplacer dans la Proposition 2, $1 < t^2 < 6/5$ par $1 < t^2 < 12/7$, mais $12/7$ nous paraît le meilleur majorant possible pour cette modélisation.

Lemme 2. *Soit t un réel vérifiant $1 < t < \sqrt{3}$. En dimension 3, une situation de $t-$quasi-réduction de type I entraîne la fin de l'exécution de l'algorithme en 3 itérations de tests respectifs $(-,+,+)$.*

$$t - qr.I \xrightarrow[3 \text{ itérations de tests respectifs } (-,+,+)]{1<t<\sqrt{3}} FIN.$$

Preuve. Comme la base (b_1, b_2, b_3) n'est pas réduite, les vecteurs b_2 et b_3 seront échangés, puis l'algorithme translatera b_3 parallèlement à b_1, afin de le rendre le plus court possible. Soit b_3' le vecteur obtenu. Comme la quasi-réduction est de type I, le vecteur b_1 ne sera pas échangé à ce stade, (ou encore $|b_3'| > |b_1|$) et la base (b_1, b_3') est Gauss-réduite. En résumé,

$$m = \begin{array}{c} \\ b_1 \\ b_2 \\ b_3 \end{array} \!\!\begin{array}{c} \begin{array}{ccc} b_1^* & b_2^* & b_3^* \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ m_{21} & 1 & 0 \\ m_{31} & m_{32} & 1 \end{pmatrix} \end{array} \xrightarrow[\text{de tests: (-,+)}]{\text{en 2 itérations}} \hat{m} = \begin{array}{c} \\ b_1 \\ b_3' \\ b_2 \end{array} \!\!\begin{array}{c} \begin{array}{ccc} \hat{b_1^*} & \hat{b_2^*} & \hat{b_3^*} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ \hat{m}_{21} & 1 & 0 \\ \hat{m}_{31} & \hat{m}_{32} & 1 \end{pmatrix} \end{array}.$$

Puisque (b_1, b_2, b_3) est $t-$qr, il en est de même en particulier pour la base propre $((b_2)_{\perp b_1}, (b_3')_{\perp b_1})$. D'après le Lemme 1 (algorithme de Gauss), la base $((b_3')_{\perp b_1}, (b_2)_{\perp b_1})$ sera réduite en une itération de test positif[2]. □

Proposition 2. *Soit t un réel vérifiant $1 < t < \sqrt{6/5}$. En dimension 3, une situation de $t-$quasi-réduction de type II entraîne la fin de l'exécution de l'algorithme ou une situation de $t'-qr.I$ avec $1 < t' < \sqrt{3}$, en au plus 6 itérations.*

$$t - qr.II \xrightarrow[\text{au plus 6 itérations}]{1<t<\sqrt{6/5}} \left\{ \begin{array}{c} FIN \\ ou \\ t' - qr.I \quad \text{avec } 1 < t' < \sqrt{3} \end{array} \right. .$$

Preuve. Si pendant l'exécution de l'algorithme LLL, se produit une situation de $t-$qr.II pour une base (b_1, b_2, b_3), alors les vecteurs b_2 et b_3 seront échangés, puis l'algorithme translatera b_3 parallèlement à b_1, afin de le rendre le plus court possible. Soit b_3' le vecteur ainsi obtenu. Comme la quasi-réduction est de type II, le vecteur b_3' vérifie $|b_3'| < |b_1|$ et donc b_1 et b_3' seront échangés. En résumé,

$$(8) \quad m = \begin{array}{c} \\ b_1 \\ b_2 \\ b_3 \end{array} \!\!\begin{array}{c} \begin{array}{ccc} b_1^* & b_2^* & b_3^* \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ m_{21} & 1 & 0 \\ m_{31} & m_{32} & 1 \end{pmatrix} \end{array} \xrightarrow[\text{de tests: (-,-)}]{\text{en 2 itérations}} \hat{m} = \begin{array}{c} \\ b_3' \\ b_1 \\ b_2 \end{array} \!\!\begin{array}{c} \begin{array}{ccc} \hat{b_1^*} & \hat{b_2^*} & \hat{b_3^*} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ \hat{m}_{21} & 1 & 0 \\ \hat{m}_{31} & \hat{m}_{32} & 1 \end{pmatrix} \end{array}.$$

La preuve de cette proposition repose alors sur le Lemme 5 et le Corollaire 1. D'après le Lemme 5, une $t-$quasi-réduction de type II peut éventuellement déboucher sur une nouvelle, mais le Corollaire 1 interdit plus de 2 quasi-réductions de type II, ce qui prouve la Proposition.

2. Le Lemme 1 n'exige pas que la base $t-$Gauss réduite soit entière.

Toutes les possibilités sont décrites dans le diagramme suivant où chaque flèche schématise 3 itérations de l'algorithme LLL.

$$
t-\text{qr.II} \xrightarrow[\text{(Lemme 5)}]{1 < t^2 < 6/5} \begin{cases} \text{FIN} \\ t'-\text{qr.I} \\ 1 < t'^2 \leq \frac{12}{7} < 3 \\ \\ t'-\text{qr.II} \qquad \xrightarrow[\left(\begin{array}{c}\text{Lemme 5 et} \\ \text{Corollaire 1}\end{array}\right)]{\qquad\qquad} \qquad t''-\text{qr.I} \\ 1 < t'^2 \leq \frac{12}{7} \qquad\qquad\qquad\qquad 1 < t''^2 \leq 3 \end{cases}
$$

□

Lemme 3. *Soit t un réel vérifiant $1 < t < (3/2)$. Si la base (b_1, b_2, b_3) est $t-$quasi-réduite et propre, alors parmi les vecteurs du réseau se trouvant en dehors du plan $\langle b_1, b_2 \rangle$, il existe au signe près, au plus un vecteur du réseau de norme strictement inférieure à $|b_3|$.*

Ce résultat qui reste vrai a fortiori lorsque la base (b_1, b_2, b_3) est réduite, permet par ailleurs de caractériser précisément la qualité d'une base réduite au sens de 1-Lovász en dimension 3. Une démonstration détaillée est donnée dans la Section 5.

Corollaire 1. *Soit t un réel vérifiant $1 < t < (3/2)$. Durant une exécution de l'algorithme LLL(1), il se produit au plus 2 situations de $t-$qr.II.*

Preuve. Supposons que durant l'exécution de l'algorithme LLL(1), il se produit une situation $t-$quasi-réduction de type II avec $1 < t < (3/2)$. D'aprés le Lemme précédent, si $(b_1, b_2, b'_3) := \text{Propre}(b_1, b_2, b_3)$, alors au signe près, le réseau contient en dehors du plan $\langle b_1, b_2 \rangle$ au plus un vecteur de norme strictement inférieure à b'_3.
Comme la $t-$quasi-réduction est de type II, $|b'_3| < |b_1|$ et l'algorithme échangera nécessairement $|b_1|$ et $|b'_3|$.
Mais (b_1, b_2) étant Gauss-réduite, b_1 est le plus court vecteur du réseau appartenant au plan $\langle b_1, b_2 \rangle$ et $|b'_3| < |b_1|$ implique qu'il n'existe dans le plan $\langle b_1, b_2 \rangle$ aucun vecteur du réseau plus court que b'_3. Par ailleurs, comme en dehors du plan $\langle b_1, b_2 \rangle$, il existe au plus un vecteur plus court que b'_3, on déduit que dans le réseau tout entier, il existe, au signe près, au plus un vecteur plus court que b'_3. Donc, dans la suite de l'exécution de l'algorithme, b'_3 pourra être échangé au plus une fois. Cela signifie qu'au plus une nouvelle situation de $t-$quasi-réduction de type II pourrait se présenter. □

Le Lemme suivant montre que les $\hat{m}_{i,j}$ (cf. (8)) sont bornés. Plus précisément,

Lemme 4. *Lors d'une situation de $t-$quasi-réduction de type II, après 2 itérations de tests négatifs, les coefficients de la nouvelle matrice $\hat{m}$ de passage vérifient:*

(a)	*Pour* $1 < t < (3/2)\sqrt{3}$,	$\lfloor \hat{m}_{21} \rceil \in \{-1, 0, 1\}$
(b)	*Pour* $1 < t < 2$,	$\lfloor \hat{m}_{32} \rceil \in \{-1, 0, 1\}$
(c)	*Pour* $1 < t^2 < \sqrt{10} - 1$,	$\lfloor \hat{m}_{31} \rceil \in \{-1, 0, 1\}$

La preuve détaillée est donnée dans la Section 5.

Lemme 5. *Soit t un réel vérifiant $1 < t < 2$. En dimension 3, une situation de $t-$quasi-réduction de type II entraîne la fin de l'exécution de l'algorithme ou une situation de $t'-$qr. avec,*

$$\frac{1}{t'^2} > \frac{1}{t^2} - \frac{1}{4},$$

en au plus 3 itérations de tests respectifs $(-,-,+)$.

$$t - qr.II \quad \xrightarrow[\text{au plus 3 itérations de tests respectifs } (-,-,+).]{1<t<2} \quad \begin{cases} \textit{FIN} \\ \textit{ou} \\ t' - qr. \ \textit{avec} \quad \frac{1}{t'^2} > \frac{1}{t^2} - \frac{1}{4} \end{cases}.$$

La preuve de ce dernier fait est basée sur l'assertion (a) du lemme précédent et la définition d'une situation de $t-$qr.II. Elle est donnée dans la Section 5.

5. Les preuves

5.1. **Preuve du Lemme 3.** Soit u un vecteur du réseau. Il s'écrit dans la base (b) sous la forme $u = x\, b_1 + y\, b_2 + z\, b_3$ avec $(x, y, z) \in \mathbb{Z}^3$.
La matrice triangulaire $(m_{i,j})$ de passage de (b) à (b^*) est rappelée en (8). Le vecteur u s'exprime dans la base orthogonale (b^*) et sa norme vérifie

$$|u|^2 = (x + y\, m_{21} + z\, m_{31})^2\, |b_1^*|^2 + (y + z\, m_{32})^2\, |b_2^*|^2 + |b_3^*|^2. \tag{9}$$

Nous allons montrer successivement:

1. Si $|z| > 1$ ou $|y| > 1$ ou $|x| > 1$, alors $|u| > |b_3|$.
2. Si $z = 1$, alors il existe au plus un couple $(x, y) \in \{0, 1, -1\}^2 \backslash \{(0, 0)\}$ tel que $|u| < |b_3|$

1. Lorsque (b_1, b_2, b_3) est $t-$qr. et propre, nous avons:

$$(10) \quad \frac{3}{4} \leq 1 - m_{21}^2 \leq \frac{|b_2^*|^2}{|b_1^*|^2} \quad (10.1) \qquad \text{et} \qquad \frac{4 - t^2}{4t^2} \leq \frac{1}{t^2} - m_{32}^2 \leq \frac{|b_3^*|^2}{|b_2^*|^2} \quad (10.2)$$

Par ailleurs, les vecteurs b_2 et b_3 s'expriment dans la base orthogonale (b^*), comme en (8), et grâce à la propreté leurs normes vérifient

$$|b_2|^2 \ \leq \ |b_2^*|^2 + \frac{1}{4}|b_1^*|^2 \quad \text{et} \quad |b_3|^2 \ \leq \ |b_3^*|^2 + \frac{1}{4}\left(|b_1^*|^2 + |b_2^*|^2\right).$$

Les relations (10) donnent alors

$$(11) \qquad \frac{|b_2^*|}{|b_2|} > \frac{\sqrt{3}}{2} \quad (11.1) \qquad \text{et} \qquad \frac{|b_3^*|}{|b_3|} > \sqrt{\frac{12 - 3t^2}{12 + 4t^2}} \quad (11.2).$$

(a) La relation (9) entraîne $\quad |u| \ \geq \ |z|\, |b_3^*|$.
Donc si $|z| \geq 2$, pour $1 < t \leq \dfrac{3}{2}$ et d'après (11.2), on a bien $|u| > |b_3|$.

(b) Si $|z| = 1$, la relation (9) montre que $|u|^2 \ \geq (y \pm m_{32})^2\, |b_2^*|^2 + |b_3^*|^2$.
Pour $|y| \geq 2$, nous avons:

$$|u|^2 \quad \geq \quad \frac{9}{4}\, |b_2^*|^2 \, + \, |b_3^*|^2 \quad \geq \quad 2\, |b_2^*|^2 \, + \, \frac{1}{4}\, |b_2^*|^2 \, + \, |b_3^*|^2$$

Finalement comme (b_1, b_2) est réduite (10.1) et b_3 est propre,

$$|u|^2 \quad \geq \quad 2\frac{3}{4}\,|b_1^*|^2 \,+\, \frac{1}{4}\,|b_2^*|^2 \,+\, |b_3^*|^2 \quad > \quad |b_3|^2$$

Notons par ailleurs que si $|z| = 1$ et $|y| \geq 2$, alors $|u| > |b_2|$.
En effet, pour $y \in \mathbb{Z}^*$ et grâce à la propreté: $|u| \geq |y \pm m_{32}|\,|b_2^*| \geq (|y| - 1/2)\,|b_2^*|$

$$\text{D'où si } |y| \geq 2, \text{ en utilisant (11.1) } |u| \geq \frac{3}{2}\frac{\sqrt{3}}{2}|b_2| > |b_2|$$

(c) Si $|z| = 1$, $|y| \leq 1$ et $|x| \geq 2$, d'après la relation(9) et grâce encore à la propreté:

$$|u|^2 \;>\; |b_1^*|^2 + (m_{32})^2\,|b_2^*|^2 + \,|b_3^*|^2 \;>\; |b_3|^2$$

Notons également que dans ce cas, $\quad |u| > |b_1|$.

2. Soit u un vecteur du plan $z = 1$ de la forme $u = b_3 + x\,b_1 + y\,b_2$ avec $(x, y) \in \{-1, 0, 1\}^2 \backslash \{(0, 0)\}$.
Nous allons montrer qu'il existe au plus un vecteur u, vérifiant $|u| < |b_3|$. Ce vecteur dépend des signes de m_{21}, m_{31} et m_{32}. Le tableau de la figure 1 résume la situation. Par la suite, on va étudier le rapport $|u|^2/|b_3|^2$.

m_{21}	m_{31}	m_{32}	u
$+$	$+$	$+$	$b_3 - b_2$
$+$	$+$	$-$	$b_3 - b_1 + b_2$
$+$	$-$	$+$	$b_3 + b_1 - b_2$
$+$	$-$	$-$	$b_3 + b_2$
$-$	$+$	$+$	$b_3 - b_2 - b_1$
$-$	$+$	$-$	$b_3 + b_2$
$-$	$-$	$+$	$b_3 - b_2$
$-$	$-$	$-$	$b_3 + b_2 + b_1$

TAB. 1. *L'unique vecteur u suscesptible d'être plus court que le vecteur b_3 en fonction des signes de m_{21}, m_{31} et m_{32}.*

$$\frac{|u|^2}{|b_3|^2} = 1 + x^2\,\frac{|b_1|^2}{|b_3|^2} + y^2\,\frac{|b_2|^2}{|b_1|^2}\,\frac{|b_1|^2}{|b_3|^2} + 2xy\,\frac{(b_1, b_2)}{|b_1|^2}\,\frac{|b_1|^2}{|b_3|^2} + 2x\,\frac{(b_1, b_3)}{|b_1|^2}\,\frac{|b_1|^2}{|b_3|^2} + 2y\,\frac{(b_2, b_3)}{|b_1|^2}\,\frac{|b_1|^2}{|b_3|^2}$$

Or les vecteurs b_2 et b_3 s'expriment dans la base orthogonale (b^*) comme en (8) et

$$\frac{(b_2, b_3)}{|b_1|^2} \;=\; m_{21}m_{31} + m_{32}\,\frac{|b_2^*|^2}{|b_1|^2}.$$

On pose alors: $$\frac{|u|^2}{|b_3|^2} = 1 + A(m_{21}, m_{31}, m_{32}, x, y)\,\frac{|b_1|^2}{|b_3|^2},$$

avec $A(m_{21}, m_{31}, m_{32}, x, y) := x^2 + 2m_{21}xy + 2m_{31}x + 2m_{21}m_{31}y + y^2\,\frac{|b_2|^2}{|b_1|^2} + 2m_{32}\,\frac{|b_2^*|^2}{|b_1|^2}\,y.$

Il ne reste plus qu'à montrer que dans l'ensemble $\{-1, 0, 1\}^2 \backslash \{(0, 0)\}$, il existe au plus un couple (x, y) tel que $A(m_{21}, m_{31}, m_{32}, x, y) < 0$.
On remarque d'abord que:

$$A(m_{21}, m_{31}, m_{32}, -x, y) = A(-m_{21}, -m_{31}, m_{32}, x, y) \tag{12}$$

$$A(m_{21}, m_{31}, m_{32}, x, -y) = A(-m_{21}, m_{31}, -m_{32}, x, y) \tag{13}$$

Lorsqu'il n'y a pas d'ambiguïté, $A(m_{21}, m_{31}, m_{32}, x, y)$ est noté simplement $A(x, y)$. Tous les couples (x, y) de $\{-1, 0, 1\}^2 \backslash \{(0, 0)\}$ sont envisagés ci-dessous.

Si $y = 0$, aucun vecteur u du plan $z = 1$ n'est plus court que b_3.
En effet, $A(x, 0) = x^2 + 2m_{31}x$.
Etant donné que $|x| = 1$ et que la base est propre: $A(x, 0) \geq 0$.

Le vecteur $b_3 + b_1 + b_2$ est plus court que b_3 si $A(1, 1) < 0$.

Or, $A(1,1) = \overbrace{\frac{|b_2|^2}{|b_1|^2} + 2m_{32}\,\frac{|b_2^*|^2}{|b_1|^2}}^{\geq 0} + 1 + 2m_{21} + 2m_{31} + 2m_{21}m_{31}$

1. La fonction f définie par $f(m_{21}, m_{31}) := 1 + 2m_{21} + 2m_{31} + 2m_{21}m_{31}$,
$$\text{a une dérivée} \quad \frac{\partial f}{\partial m_{31}} = 2(1 + m_{21}) > 0.$$
$$\text{Donc} \quad f(m_{21}, -\frac{1}{2}) = m_{21} < f(m_{21}, m_{31}).$$
En particulier, l'inégalité $m_{21} \geq 0$ entraîne $A(1, 1) > 0$. et la fonction f étant symétrique, il en est de même si $m_{31} \geq 0$.
2. Supposons maintenant que $m_{21} < 0$ et $m_{31} < 0$.
$$\text{Si} \quad m_{32} \geq 0, \quad \text{alors } \frac{|b_2|^2}{|b_1|^2} + 2m_{32}\,\frac{|b_2^*|^2}{|b_1|^2} \geq 1,$$
$$\text{et} \quad A(1,1) \geq \overbrace{2 + 2m_{21} + 2m_{31}}^{\geq 0} + 2m_{31}m_{21} > 0.$$

On déduit de ce qui précède:

$$A(1,1) < 0 \longrightarrow m_{21} < 0 \text{ et } m_{31} < 0 \text{ et } m_{32} < 0. \tag{14}$$

La condition nécessaire (14) et les relations (13) et (13) montrent alors

$$A(1,-1) < 0 \longrightarrow m_{21} > 0 \text{ et } m_{31} < 0 \text{ et } m_{32} > 0. \tag{15}$$

$$A(-1,1) < 0 \longrightarrow m_{21} > 0 \text{ et } m_{31} > 0 \text{ et } m_{32} < 0. \tag{16}$$

$$A(-1,-1) < 0 \longrightarrow m_{21} < 0 \text{ et } m_{31} > 0 \text{ et } m_{32} > 0. \tag{17}$$

Le vecteur $b_3 + b_2$ est plus court que b_3 si $A(0,1) < 0$.
Or, $A(0,1) = \frac{|b_2|^2}{|b_1|^2} + 2m_{32}\frac{|b_2^*|^2}{|b_1|^2} + 2m_{21}m_{31}$.

$$\text{Mais si} \quad m_{32} \geq 0, \quad \text{alors } \frac{|b_2|^2}{|b_1|^2} + 2m_{32}\frac{|b_2^*|^2}{|b_1|^2} \geq 1.$$

Et donc $\quad \forall (m_{21}, m_{31}) \in [-\frac{1}{2}, \frac{1}{2}]^2, \quad A(0,1) \geq 0$.

$$\text{Par ailleurs,} \quad \forall m_{32} \in [-\frac{1}{2}, \frac{1}{2}], \quad \frac{|b_2|^2}{|b_1|^2} + 2m_{32}\frac{|b_2^*|^2}{|b_1|^2} \geq 0.$$

Finalement l'inégalité $A(0,1) < 0$ implique $m_{21}m_{31} < 0$. On en déduit:

$$A(0,1) < 0 \longrightarrow \left\{ \begin{array}{ll} m_{21} > 0 \,,\, m_{31} < 0 \,,\, m_{32} < 0 & (18.1) \\ \text{ou} & \\ m_{21} < 0 \,,\, m_{31} > 0 \,,\, m_{32} < 0 & (18.2) \end{array} \right. \tag{18}$$

La condition nécessaire (18) et la relation (13) montrent

$$A(0,-1) < 0 \longrightarrow \left\{ \begin{array}{ll} m_{21} > 0 \,,\, m_{31} > 0 \,,\, m_{32} > 0 & (19.1) \\ \text{ou} & \\ m_{21} < 0 \,,\, m_{31} < 0 \,,\, m_{32} > 0 & (19.2) \end{array} \right. \tag{19}$$

Pour terminer la preuve, on remarque que les conditions (14), (15), (16), (17), (18.1), (18.2), (19.1), (19.2) s'excluent deux à deux. Par exemple, (b_1, b_2, b_3) est $t-$quasi-réduite et propre et les m_{ij} sont tous strictement positifs, seul le vecteur $b_3 - b_2$ est susceptible d'être plus court que b_3.

□

5.2. **Preuve du Lemme 4 (Encadrements des m_{ij}). Rappel**: Si une base (b_1, b_2, b_3) est $t-$qr.II, b_2 et b_3 seront échangés, puis l'algorithme translatera b_3 parallèlement à b_1, afin de le rendre le plus court possible. Plus précisément, avec $m'_{31} := m_{31} - \lfloor \hat{m}_{31} \rceil$, le vecteur b'_3 ainsi obtenu vérifie $b'_3 := m'_{31}b_1^* + m_{32}b_2^* + b_3^*$. Comme la quasi-réduction est de type II, $|b'_3| < |b_1|$ et donc les vecteurs b_1 et b'_3 seront échangés. En résumé,

$$m = \begin{array}{c} \\ b_1 \\ b_2 \\ b_3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} b_1^* & b_2^* & b_3^* \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ m_{21} & 1 & 0 \\ m_{31} & m_{32} & 1 \end{pmatrix} \end{array} \xrightarrow[\text{de tests: (-,-)}]{\text{en 2 itérations}} \hat{m} = \begin{array}{c} \\ b'_3 \\ b_1 \\ b_2 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} \hat{b_1^*} & \hat{b_2^*} & \hat{b_3^*} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ \hat{m}_{21} & 1 & 0 \\ \hat{m}_{31} & \hat{m}_{32} & 1 \end{pmatrix} \end{array}$$

5.2.1. *Encadrement de $\hat{m}_{21}$.* Par définition,

$$\hat{m}_{21} = \frac{(b_1, b'_3)}{|b'_3|^2} = \frac{m'_{31}|b_1^*|^2}{|b'_3|^2}. \tag{20}$$

Or, le vecteur b'_3 s'exprime dans la base (b^*) et sa norme vérifie

$$|b'_3|^2 = {m'_{31}}^2|b_1^*|^2 + {m_{32}}^2|b_2^*|^2 + |b_3^*|^2.$$

Et comme (b_1, b_2, b_3) est $t-$qr., les relations (10.2) et (10.1) montrent l'inégalité

$$|b'_3|^2 > \frac{1}{t^2}|b_2^*|^2 > \frac{3}{4}\frac{1}{t^2}|b_1^*|^2. \tag{21}$$

La propreté et les relations (20) et (21) permettent alors de conclure:

$$|\hat{m}_{21}| \leq \frac{2}{3}\, t^2. \tag{22}$$

Donc pour $1 < t < \dfrac{3}{2}$, on a bien $|\hat{m}_{21}| < \dfrac{3}{2}$ et $\lfloor \hat{m}_{21} \rceil \in \{-1, 0, 1\}$.

Une étude plus précise montrerait que le résultat reste valable pour $1 < t < \frac{3}{2}\sqrt{3}$.

5.2.2. *Encadrement de $\hat{m}_{31}$.* Par définition, $\quad \hat{m}_{31} = \dfrac{(b_2, b'_3)}{|b'_3|^2} = \dfrac{m_{21}m'_{31}|b_1^*|^2 + m_{32}|b_2^*|^2}{|b'_3|^2}.$

D'après la définition de $\hat{m}_{21}$ (20) et la relation (21),

$$|\hat{m}_{31}| < |m_{21}||\hat{m}_{21}| + |m_{32}|t^2.$$

Et grâce à la propreté et la majoration de $|\hat{m}_{21}|$ en (22), $\quad |\hat{m}_{31}| < \dfrac{5}{6}\, t^2.$

Finalement lorsque $1 < t^2 < \dfrac{9}{5}$, on a $|\hat{m}_{31}| < \dfrac{3}{2}$ et $\lfloor \hat{m}_{31} \rceil \in \{-1, 0, 1\}$.

Une étude plus précise montre que le résultat reste valable pour $1 < t^2 < \sqrt{10} - 1$.

5.2.3. *Encadrement de $\hat{m}_{32}$.* Par définition, $\quad \hat{m}_{32} = \dfrac{(b_2, (b_1)_{\perp b'_3})}{\left|(b_1)_{\perp b'_3}\right|^2} = \dfrac{(b_2, b_1 - \hat{m}_{21} b'_3)}{\left|(b_1)_{\perp b'_3}\right|^2}.$

En exprimant les vecteurs b_2 et b'_3 dans la base orthogonale (b^*) comme dans le rappel du début de la preuve, on obtient

$$\hat{m}_{32} = \frac{m_{21}(1 - \hat{m}_{21}m'_{31})|b_1|^2 - \hat{m}_{21}m_{32}|b_2^*|^2}{|(b_1)_{\perp b'_3}|^2}. \tag{23}$$

Or, $\quad |(b_1)_{\perp b'_3}|\,|b'_3| = |(b'_3)_{\perp b_1}|\,|b_1| \quad$ et $\quad (b'_3)_{\perp b_1} = (b_3)_{\perp b_1},$. D'où,

$$|(b_1)_{\perp b'_3}|^2 = \frac{\left(m_{32}^2|b_2^*|^2 + |b_3^*|^2\right)|b_1|^2}{|b'_3|^2}. \tag{24}$$

Les relations (20), (23) et (24) montrent après simplification

$$\hat{m}_{32} = m_{21} - m'_{31}m_{32}\frac{|b_2^*|^2}{m_{32}^2|b_2^*|^2 + |b_3^*|^2}.$$

Pour finir, comme (b_1, b_2, b_3) est $t-$quasi-réduite, $\quad \hat{m}_{32} \leq \dfrac{1}{2} + \dfrac{t^2}{4}.$

Pour $1 < t < 2$, on a $|\hat{m}_{32}| \leq \dfrac{3}{2}$ et $\lfloor \hat{m}_{32} \rceil \in \{-1, 0, 1\}$. $\square$

5.3. **Preuve du Lemme 5.** Nous partons ici de la base (b'_3, b_1, b_2) (obtenue nécessairement en 2 itérations de tests négatifs à partir de la base (b_1, b_2, b_3), supposée $t-$qr.II) et nous nous intéresserons à l'action de l'algorithme sur cette base.
Le vecteur b_1 sera échangé par $b'_1 := b_1 - \lfloor \hat{m}_{21} \rceil b'_3$. Le test qui s'en suit est positif. (En effet (b'_3, b_1) était propre et la propreté de (b_1, b'_3) n'exige au pire qu'une translation entière de b_1 selon $\pm b'_3$, car d'après le Lemme 4, $\quad \lfloor \hat{m}_{21} \rceil \in \{-1, 0, 1\}$.)

En conséquence, le deuxième orthogonalisé $\hat{b_2^*} = (b_1)_{\perp b'_3}$ ne sera pas modifié et sa norme est donnée par la relation (24).
Comme (b_1, b_2, b_3) est $t-$qr.II, en particulier la deuxième boîte $(b_2^*, m_{32}b_2^* + b_3^*)$ est non réduite. D'où:

$$|(b'_1)_{\perp b'_3}|^2 \leq \frac{|b_2^*|^2 |b_1|^2}{|b'_3|^2}. \tag{25}$$

A ce stade, le vecteur b_2 sera translaté parallèlement à b'_1 (afin de rendre propre la base $\left((b'_1)_{\perp b'_3}, (b_2)_{\perp b'_3}\right)$). Appelons b'_2 le vecteur obtenu après la translation.

Le test qui suivra concerne le rapport $\dfrac{|(b'_2)_{\perp b'_3}|^2}{|(b'_1)_{\perp b'_3}|^2}$, que nous allons minorer dans la suite.

Nous avons: $b'_2 = b_2 - \lfloor \hat{m}_{32} \rceil b'_1 = b_2 - \lfloor \hat{m}_{32} \rceil b_1 + \lfloor \hat{m}_{32} \rceil \lfloor \hat{m}_{21} \rceil b'_3$
Le vecteur b'_2 est donc de la forme $b_2 - nb_1 + mb'_3$.

$$\text{D'où:} \quad (b'_2)_{\perp b'_3} = (b_2 - nb_1)_{\perp b'_3} = (b_2 - nb_1) - \frac{(b_2 - nb_1, b'_3)}{|b'_3|^2} b'_3 \qquad (n \text{ désigne } \lfloor \hat{m}_{32} \rceil)$$

$$\text{et:} \quad \left|(b'_2)_{\perp b'_3}\right|^2 = |b_2 - nb_1|^2 - \frac{(b_2 - nb_1, b'_3)^2}{|b'_3|^2}$$

Les vecteurs b_1, b_2 et b'_3 s'expriment dans la base orthogonale (b^*), et après simplification,

$$\left|(b_2 - nb_1)_{\perp b'_3}\right|^2 = \frac{|b_2^*|^2 |b_1^*|^2}{|b'_3|^2} \left(\frac{|b_3^*|^2}{|b_1|^2} + (m_{21} - n)^2 \frac{m_{32}^2 |b_2^*|^2 + |b_3^*|^2}{|b_2^*|^2} + {m'}_{31}^2 - 2(m_{21} - n){m'}_{31} m_{32} \right).$$

Etant donnée la majoration obtenue pour $|(b'_1)_{\perp b'_3}|^2$ en (25),

$$\frac{\left|(b_2 - nb_1)_{\perp b'_3}\right|^2}{|(b'_1)_{\perp b'_3}|^2} \geq \frac{|b_3^*|^2}{|b_1|^2} + (m_{21} - n)^2 \frac{m_{32}^2 |b_2^*|^2 + |b_3^*|^2}{|b_2^*|^2} + {m'}_{31}^2 - 2(m_{21} - n){m'}_{31} m_{32}.$$

Mais la base (b_1, b_2, b_3) est $t-$quasi-réduite. D'après les relation(10),

$$\frac{\left|(b_2 - nb_1)_{\perp b'_3}\right|^2}{|(b'_1)_{\perp b'_3}|^2} \geq (1 - m_{21}^2) \left(\frac{1}{t^2} - m_{32}^2 \right) + \frac{(m_{21} - n)^2}{t^2} + {m'}_{31}^2 - 2(m_{21} - n){m'}_{31} m_{32}$$

Après simplification et comme la base (b_1, b_2, b_3) est propre,

$$\frac{\left|(b_2 - nb_1)_{\perp b'_3}\right|^2}{|(b'_1)_{\perp b'_3}|^2} \geq (1 + n^2 - 2n\, m_{21}) \left(\frac{1}{t^2} - m_{32}^2 \right) + \left(m_{32}(m_{21} - n) - {m'}_{31} \right)^2 \geq (1 + n^2 - |n|) \left(\frac{1}{t^2} - \frac{1}{4} \right)$$

Dans la relation précédente, le minorant est clairement une fonction croissante de n. Aussi,

$$\frac{\left|(b_2 - nb_1)_{\perp b'_3}\right|^2}{|(b'_1)_{\perp b'_3}|^2} \geq \frac{1}{t^2} - \frac{1}{4} \qquad \text{valable pour tout } n \in \mathbb{Z}.$$

Finalement, pour $1 < t < 2$, l'algorithme débouche bien au pire sur une situation de $t'-$quasi-réduction avec:

$$\frac{1}{t'^2} = \frac{1}{t^2} - \frac{1}{4}$$

□

6. Cas de la dimension n fixée

Les raisonnements des dimensions 2 et 3 se généralisent à une dimension n fixée. Ils permettent de montrer que là encore le nombre d'itérations de l'algorithme LLL(1) est un polynôme en $\log M$. Mais les bornes obtenues par cette modélisation se dégradent de façon exponentielle avec la dimension.
La difficulté principale est la suivante. Il existe à priori de nombreuses bases réduites au sens de 1-Lovász dans un réseau donné. En particulier, une estimation grossière du nombre de vecteurs du réseau "candidats" à être le premier vecteur d'une base réduite (Lemme 6) laisse supposer que ce nombre croît exponentiellement avec la dimension et le raisonnement qui suit est basé sur ce nombre.
Dans la suite, le Lemme 6 joue le même rôle que le Lemme 3 pour la dimension 3. Ce Lemme est une version plus courte d'un Lemme utilisé par Kannan [Kan83] pour mettre en évidence un algorithme qui construit une base réduite au sens de Korkine-Zolotarev avec une complexité en $\mathrm{O}(n^n)$. La borne polynomiale en $\log M$ pour le nombre d'itérations de l'algorithme LLL(1) à dimension n fixée s'en déduit par réccurence sur n.

Lemme 6. *Soient t un réel vérifiant $t > 1$ et $(b) := (b_1, \ldots, b_n)$ une base $t-$quasi-réduite et propre d'un réseau $\mathcal{L}$. Le nombre de vecteurs du réseau $\mathcal{L}$, de normes strictement inférieures à $|b_1|$, est majoré par une constante qui ne dépend que de la dimension n et du paramètre t. Elle est notée $\alpha(n,t)$.*

Preuve. Tout vecteur $v = \sum_{i=1}^n v_i b_i$ du réseau s'exprime dans la base orthogonale (b^*) comme en (1) et si il est plus court que b_1, il vérifie:

$$|v|^2 = \sum_{i=1}^n \left(\sum_{j=i}^n v_j m_{j,i} \right)^2 |b_i^*|^2 < |b_1|^2.$$

La recherche d'un vecteur plus court que $|b_1|$ revient donc à chercher un vecteur entier $(v_1, \ldots, v_n)$ vérifiant

$$0 \leq |v_n| < \frac{|b_1|}{|b_n^*|}$$

et

$$\left(\sum_{j=i}^n v_j m_{j,i} \right)^2 |b_i^*|^2 < |b_1|^2 - \sum_{k=i+1}^n \left(\sum_{j=k}^n v_j m_{j,i} \right)^2 |b_k^*|^2 \quad \text{pour } i = 1, \ldots, n-1.$$

Comme $|\sum_{j=i}^n v_j m_{j,i}| < |b_1|/|b_i^*|$, le nombre de valeurs possibles pour v_i lorsque $v_{i+1}, \ldots, v_n$ sont fixés est au plus $\lfloor 2|b_1|/|b_i^*| \rfloor + 1$. Donc, au plus

$$\prod_{i=2}^n \left(\lfloor 2 \frac{|b_1|}{|b_i^*|} \rfloor + 1 \right)$$

vecteurs $(v_1, \ldots, v_n)$ sont susceptibles d'être plus courts que b_1. Mais la base $(b_1, \ldots, b_n)$ est $t-$quasi-réduite. La Définition (2) entraîne alors

$$\frac{|b_i^*|^2}{|b_{i+1}^*|^2} \leq \frac{4}{3} \quad \text{pour } i = 1, \ldots, n-2$$

et

$$\frac{|b^*_{n-1}|^2}{|b^*_n|^2} \leq \frac{4t^2}{4-t^2}.$$

Ces dernières relations entraînent après simplification, que le nombre de vecteurs $(v_1, \ldots, v_n)$ susceptibles d'être plus courts que b_1 est au plus

$$\left(\frac{6t}{\sqrt{4-t^2}}\right)^n \left(\frac{2}{\sqrt{3}}\right)^{\frac{(n-1)(n-2)}{2}}.$$

□

Remarque. Il est légitime de penser que la majoration donnée par le Lemme précédent sur $\alpha(n,t)$ pourrait être considérablement amélioriée. Une approche (inspirée du cas de la dimension 3) serait de montrer que les signes des $m_{i,j}$ déterminent le(s) vecteur(s) du réseau de norme strictement inférieure à $|b_1|$, lorsque b_1 est le premier vecteur d'une base t-quasi–réduite et propre du réseau.

Théorème 2. *Lorsque la dimension n est fixée, le nombre maximum d'itérations de l'algorithme LLL(1) est polynomial en la taille* $\log M$, *de la base d'entrée.*

Preuve. Elle se fait par récurrence sur n. La propriété est vraie en dimension 2 et 3 comme le montrent les parties précédentes.
Supposons que la propriété est vraie au rang $n-1$. Soit t un réel vérifiant $t > 1$. Les phases non finales qui se produisent pendant une exécution de l'algorithme peuvent être classées en trois groupes:

1. Les phases non finales qui produisent une base $(b_1, \ldots, b_n)$ non $t-$quasi-réduite. Si d désigne le produit $\prod_1^{n-1} |b^*_i|^2$, une telle phase se termine par un test négatif dans la dernière boîte entraînant une décroissance de la quantité d d'un facteur $1/t^2$. La quantité d et le nombre maximum k de ces phases vérifient alors respectivement
$$1 \leq d \leq \frac{1}{(t^2)^k} M^{2(n-1)} \quad \text{et} \quad k \leq (n-1)\log_t M.$$
2. Les phases qui finissent sur des situations $t-$qr.II. Mais d'après le Lemme 6, le nombre total de situation $t-$qr.II, se produisant pendant une exécution de l'algorithme est une constante qui ne dépend que de la dimension n et du paramètre t. $(\alpha(n,t)+1)$.
3. Les phases qui finissent sur des situations $t-$qr.I. Au plus, un nombre polynomial de phases de ce type peuvent se succéder, car tant que les phases débouchent sur des $t-$qr.I successives, l'algorithme travaille sur la base
$$\left((b_2)_{\perp\langle b_1\rangle}, \ldots, (b_n)_{\perp\langle b_1\rangle}\right)$$
qui a $n-1$ vecteurs (et qui est d'ailleurs également $t-$qr.I) et l'hypothèse de récurrence entraîne que le nombre de phases pour réduire une base en dimension $n-1$ est majoré par un polynôme en $\log M$. On le note $\mathcal{P}_{n-1}(\log M)$.

Donc le nombre maximum de phases durant une exécution de l'algorithme LLL(1) en dimension n est bien majoré par un polynôme $\mathcal{P}_n$ en $\log M$ avec

$$\mathcal{P}_n(\log M) = ((n-1)\log_t M + \alpha(n,t) + 1)\, \mathcal{P}_{n-1}(\log M).$$

Les deux remarques suivantes permettent alors de conclure.

1. Toutes les bases intermédiaires obtenues par l'algorithme sont de tailles polynomiales par rapport à la taille de la base initiale.
2. Une phase de l'algorithme consiste en la réduction de la base $(b_1, \ldots, b_{n-1})$, qui se fait d'après l'hypothèse de récurrence en un nombre ploynomial d'itérations, suivie d'une translation du vecteur b_n parallèlement au vecteur b_{n-1}. Le nombre d'itération d'une phase est donc polynomial en $\log M$.

$\square$

Références

[Kan83] R. Kannan. Improved algorithm for integer programming and related lattice problems. In *15th Ann. ACM Symp. on Theory of Computing*, pages 193–206, 1983.

[Lag82] J. C. Lagarias. The computationnal complexity of simultaneous diphantine approximation problems. In *23rd IEEE Symposium On Foudations of Computer Science*, pages 32–39, 1982.

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:513–534, 1982.

[LO83] J. C. Lagarias and A. M. Odlyzko. Solving low–density subset sum peoblems. In *24th IEEE Symposium FOCS*, pages 1–10, 1983.

[Val91] B. Vallée. Gauss' algorithm revisited. *Journal of Algorithms*, 12:556–572, 1991.

GREYC - Université de Caen