Extendet Abstract

Lower Bounds for the Running Time of the LLL Lattice Reduction Algorithm

Oliver van Sprang

FB 14-Informatik
Universität des Saarlandes
Im Stadtwald
D-66123 Saarbrücken
West Germany

e-mail: `oliver@cs.uni-sb.de`

August 4, 1993

Abstract

It is still an open problem whether the lattice reduction algorithm due to Lenstra, Lenstra and Lovász is effective for the choice of the parameter $y = 1$ and dimension $n > 2$, $n \in \mathbb{N}$, LLL$(n, y = 1)$ for short. The choice of the parameter $y = 1$ yields the best possible reduction results for this algorithm. We present bases $\Omega_l \in \mathbb{Z}^{n \times n}$, $l, n \in \mathbb{N}$, $n \geq 3$ of the lattice $\mathbb{Z}^n$ which depend on a parameter l. We prove that the LLL$(n, y = 1)$ algorithm performs $(2n-3)l + O(1)$ Gaußsteps on input of these bases. For lattices of rank $n = 3$ our parameterized bases cause significant longer running times than bases constructed from worst-case inputs know for the LLL$(n = 2, y = 1)$ algorithm.

Classification of the topic covered: Computational Complexity

Key words: LLL lattice reduction algorithm, parameterized input, recurrency relation, ordinary differential equation.

1 Introduction

A subset L of the real vector space $\mathbb{R}^n$ is called a *lattice* of rank n, if there exists a basis $b_1, b_2, \ldots, b_n$ of $\mathbb{R}^n$ such that $L = \sum_{i=1}^{n} \mathbb{Z} b_i = \{\sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z}, 1 \leq i \leq n\}$.

By $< v, w >= v^t \cdot w$, $v, w \in \mathbb{R}^n$ we denote the Euclidean inner product on $\mathbb{R}^n$, where v^t denotes the transposed column vector v. We denote by $||v||$, $v \in \mathbb{R}^n$ the Euclidean norm of a vector v; so $||v||^2 = < v, v >$. The i-th *sucsessive minimum* $\lambda_i(L)$, $1 \leq i \leq n$ of a lattice $L \subset \mathbb{R}^n$, with respect to the Euclidean norm, is the smallest real number r such that there are i vectors in L of length at most r that are $\mathbb{R}$ -linearly independent. A vector is called a minimum of a lattice if its Euclidean lenght is some successive minimum.

The problem of computing a shortest non zero element of a lattice with respect to the maximum norm $|| \, ||_\infty$ is known to be NP - hard, [1]. The problem of computing the minima of a lattice with respect to the Euclidean norm is know to be NP - hard [3].

Ten years ago Lenstra, Lenstra and Lovász invented a fast algorithm, [2], which computes on input of a basis $\Omega \subseteq \mathbb{Z}^{n \times n}, n \in \mathbb{N}, n \geq 2$ of a lattice L and a parameter y with $\frac{1}{4} < y < 1$ a reduced basis $(b_1, b_2, \ldots, b_n)$ of L such that the lenghts of the vectors of this reduced basis give rough approximations of the minima λ_i, $1 \leq i \leq n$ of L, namely [2]: $\left(\frac{4}{4y-1}\right)^{\frac{1-i}{2}} \lambda_i \leq ||b_i|| \leq \left(\frac{4}{4y-1}\right)^{\frac{n-1}{2}}$ for $i = 1, 2, \ldots, n$. The LLL algorithm generalizes an algorithm due to Gauß which computes a totally reduced basis for lattices of rank two to arbitray dimensions. The algorithm is given in the appendix.
The Gauß lattice reduction algorithm computes on input of a basis of a lattice $L \subseteq \mathbb{Z}^d$, $d \geq 2$ of rank two a basis (b_1, b_2) of the same lattice that satisfies the reduction conditions (1) and (2).

$$||b_1|| \leq ||b_2|| \tag{1}$$

$$| < b_1, b_2 > | \leq 2||b_1||^2. \tag{2}$$

The LLL$(n = 2, y), \frac{1}{4} < y < 1$ algorithm computes a basis which satisfies the condition (2) and condition (3).

$$y||b_1||^2 \leq ||b_2||^2,\ y \in (\frac{1}{4}, 1). \tag{3}$$

If $y \in [\frac{1}{3}, 1)$ is chosen it follows that a basis satisfying (2) and (3) will satisfy (1) and (2) after at most one Gaußstep. By a *Gaußstep* applied to two vectors $\Omega = (b_1, b_2) \in \mathbb{Z}^{d \times 2}$, $d \geq 2$ of a basis we mean the operation $(b_1', b_2') = (b_1, b_2) \cdot \begin{pmatrix} 1 & -\lfloor \mu_{21} \rceil \\ 0 & 1 \end{pmatrix}, \mu_{21} =< b_1, b_2 > \cdot |||b_1||^{-2}$. Here $\lfloor \mu_{21} \rceil$ is an integer next to μ_{21}. (b_1', b_2') is a basis of the same lattice (b_1, b_2) describe, because the determinant of the matrix describing the transformation is one. We remark that $\min\{||b_2 + x b_1|| : x \in \mathbb{Z}\} = ||b_2 - \lfloor \mu_{21} \rceil b_1||$.

The LLL$(n = 2, y)$ algorithm is fast because each Gaußstep except the first and the last one decreases the lenght of the vector with greater Euclidean length by the significant factor $\sqrt{y} \in [\frac{1}{\sqrt{3}}, 1)$. A sharp upper bound for the number of Gaußsteps for LLL($n = 2, y = 1$) was given by Brigitte Vallée which we want to summarize in the following theorem:

Theorem 1 *Let $(b_1, b_2) \in \mathbb{Z}^{d\times 2}, d \geq 2$ be a lattice basis of inertia $l = ||b_1||^2 + ||b_2||^2$. Then the number T of Gaußsteps the LLL($n = 2, y = 1$) algorihm performs on input of this basis is upper bounded with*

$$T \leq \frac{1}{2}\left[\frac{\log\left(\frac{2\sqrt{2}}{3}l\right)}{\log\left(1+\sqrt{2}\right)}\right] + 2. \tag{4}$$

Let $(u_0, u_1) \in \mathbb{Z}^{d\times 2}$, $d \geq 2$ be a lattice basis satisfying the equations (1) and (2) and let

$$(u_l, u_{l+1}) = (u_0, u_1) \cdot \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}^l. \tag{5}$$

Then the number $T(l)$ of Gaußsteps the LLL($n = 2, y = 1$) algorithm performs on input of the basis $(u_l, u_{l+1}), l \in \mathbb{N} \cup \{0\}$ is upper and lower bounded:

$$l \leq T(l) \leq l + 2. \tag{6}$$

Proof: A collection of the results in [5]. □

The LLL(n, y) algorithm for lattices of arbitrary rank $n \in \mathbb{N}, n > 2$ successively performs a Gaußsteps to the orthogonal projections of the vectors (b_i, b_{i+1}), $i = 1, 2, \ldots, n-1$ in the subspace $(\sum_{j=1}^{i-1} \mathbb{R}b_j)^\perp$. These vectors can be described by means of the Gram-Schmidt orthogonalization process: $b_i^\star = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^\star, i = 1, 2, \ldots, n$, $\mu_{ij} =< b_i, b_j^\star > \cdot ||b_j^\star||^{-2}$, $1 \leq j < i \leq n$. A basis is called LLL(n, y) reduced if equation (1) and (3) hold for each replacement of b_1 by $b_i^\star$ and b_2 by $b_{i+1}^\star + \mu_{i+1,i} b_i^\star$, for all $i \in \{1, 2, \ldots, n-1\}$.
The LLL(n, y) algorithm is know to be fast if y satisfies $\frac{1}{4} < y < 1$. The reason is that each time the condition of instruction (7) of algorithm 5.4 is satsified the quantity $D = \prod_{i=1}^n ||b_i^\star||^{2i} \geq 1$ is decreased by a factor y. But the number of Gaußsteps (i.e. calls of the function star-LLL, algorithm 5.2) between two events of this kind is at most $\sum_{i=1}^{n-1} i = \frac{1}{2}n(n-1)$. For $y = 1$ the argument that D is decreased can no longer be applied. Up to now it is an open question whether the algorithm LLL($n, y = 1$) is fast, that is the number of Gaußsteps is polynomially bounded in the number of bits to decode the input data.

In the next section we show how to construct input bases depending on a parameter $l \in \mathbb{N}$. This parameter controlls both: the number of Gaußsteps the LLL($n, y = 1$) algorithm performs on input of this bases and the number of bits necessary to decode these bases.

2 Input bases for the LLL$(n, y = 1)$ algorithm that cause an infinite running time

We call a basis $\hat{\Omega}(l) = (b_1, b_2, \ldots, b_n) \in \mathbb{Z}^{n \times n}$, $l, n \in \mathbb{N}, n \geq 2$ a *parameterized input*, (PI), for the LLL(n, y) algorithm if both, the number of Gaußsteps the LLL(n, y) algorithm performs until termination and the quantity $B = \max\{2, ||b_1||^2, ||b_2||^2, \ldots, ||b_n||^2\}$ is a function of the parameter $l \in \mathbb{N}$.
We first show how to use a worst-case input due to Brigitte Vallée, [5], from theorem 1 in order to obtain a PI for the LLL$(n, y = 1)$, $n > 2, n \in \mathbb{N}$ algorithm.

Lemma 1 *Let $\Omega_2(l) = (u_l, u_{l+1}) \in \mathbb{Z}^{2\times 2}$ denote the basis defined in equation (5) for two vectors $(u_0, u_1) \in \mathbb{Z}^{2\times 2}$ satisfying equations (1) and (2). Let $\hat{\Omega}_2(l)$ be the follwing basis:*

$$\hat{\Omega}_2(l) = \begin{pmatrix} 0 & \boxed{\Omega_2(l)} \\ I_{n-2} & 0 \end{pmatrix}, \; n > 2, \; n \in \mathbb{N}, \tag{7}$$

where I_{n-2} denotes the $(n-2)$ identity matrix. The number $T(l)$ of Gaußsteps the LLL$(n, y = 1)$ algorithm performs until termination on input of $\hat{\Omega}_2(l)$ is upper and lower bounded: $(n-1)l + \frac{(n-2)(n-1)}{2} \leq T(l) \leq (n-1)l + \frac{(n-2)(n-1)}{2} + 2l$.

Proof: The LLL$(n, y = 1)$ algorithm performs $\frac{(n-1)(n-2)}{2}$ Gaußsteps on input of $\hat{\Omega}(l)$ which cause no modification of the basis because $\mu_{ij} = 0$ for the corresponding μ's. After these steps the condition (7) of algorithm 5.4 (see Appendix) is satisfied such that the first call to the update function is made. The masterindex k of the LLL$(n, y = 1)$ algorithm at this time is n. As a consequence of the update call $k \leftarrow k - 1$, $(k = n - 1)$. The combination of the Gaußstep (6) in algorithm 5.4 and the update after this provide a transformation $\Omega' \leftarrow \hat{\Omega}_2(l-1)$ if Ω' denotes the actually computed basis by the LLL$(n, y = 1)$ algorithm. After $n-1$ Gaußsteps the situation described above will repeat with l being replaced by $l-1$. From theorem 1 we know that there are at least l and at most $l+2$ loops of this kind which completes the prove of our lemma. □

We remark that the running time of the LLL$(n, y = 1)$, $n \in \mathbb{N}, n > 2$ algorithm will tend to infinity as l tends to infinity for the input bases from Lemma 1.

We know present new input bases $\hat{\Omega}_3(l) \in \mathbb{Z}^{3\times 3}, l \in \mathbb{N}$ which can be used to construct PIs as shown in Lemma 1 but with significant larger lower bounds for the running time.

Theorem 2 *Let Ω and T be the following matrices:*

$$\Omega = \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -2 \\ 0 & 1 & 0 \end{pmatrix}$$

$$U = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & -2 \\ 0 & 1 & 0 \end{pmatrix}.$$

We define: $\Omega(s) = \Omega U^s = (\Omega_{ij}(s))_{1\leq i,j\leq 3}$, $b_i^{(s)} = (\Omega_{1i}(s), \Omega_{2i}(s), \Omega_{3i}(s))^t$, $P_{ij}^{(s)} = \langle b_i^{(s)}, b_j^{(s)} \rangle, 1 \leq i < j \leq 3, \quad s \geq 0$, *furthermore* $\mu_{ij}^{(s)}$ *as in [2] p. 516 eq. 1.3 with the quantities defined above. Then the following equations hold:*

$$b_1^{(s)} = b_2^{(s-1)} = b_3^{(s-2)}, \quad s \geq 2 \tag{8}$$

$$b_1^{(s+3)} = b_1^{(s)} - 2b_1^{(s+1)}, \quad s \geq 0 \tag{9}$$

$$|\mu_{21}^{(s)}| \leq \frac{1}{2}, \quad s \geq 0 \tag{10}$$

$$-\frac{5}{2} \leq \mu_{31}^{(s)} < -\frac{3}{2}, \quad s \geq 0 \tag{11}$$

$$|b_1^{(s)}| < |b_1^{(s+1)}|, \quad s \geq 0 \tag{12}$$

$$-\frac{1}{2} < \mu_{32}^{(s)} < 0, \quad s \geq 0 \tag{13}$$

$$|b_1^{(s)}||b_3^{(s)}| - {P_{13}^{(s)}}^2 < |b_1^{(s)}||b_2^{(s)}| - {P_{12}^{(s)}}^2, \; s > 0. \tag{14}$$

The number $T(s)$ of Gaußsteps the LLL($n = 3, y = 1$) algorithm performs until termination on input of the bases $\Omega(s)$ is exactly $T(s) = 5+3s$. The Euclidean lenght of the input vectors of these bases can be upper and lower bounded by the following set of inequalities: $0.781 \cdot (\lambda+2)^s - |\epsilon(s)| < |b_1^{(s)}| < 1.122 \cdot (\lambda+2)^s + |\epsilon(s)|, s \in \mathbb{N}, s \geq 5$, *where* $|\epsilon(s)| < 3 \cdot 0.674^{-s}$, $s \in \mathbb{N}, s \geq 5$ *and λ is the only real root of the polynomial* $p(x) = (x^3 + 4x^2 + 4x - 1)$, $\lambda + 2 \approx 2.205$.

Proof: The proof is very lengthy and technical and will be given in a full version of the paper. An outline of the proof will be given after the next theorem below. □

From the basis of theorem 2 we can construct PIs similar to these from lemma 1.

Theorem 3 *The LLL($n, y = 1$) algorithm performs at least $T(l) = (2n-3)l$ Gaußsteps on input of the bases*

$$\Omega_l = \begin{pmatrix} 0 & \boxed{\Omega_3(l)} \\ I_{n-3} & 0 \end{pmatrix} \in \mathbb{Z}^{n\times n}, n \geq 3, l, n \in \mathbb{N}$$

$$\Omega_3(l) = \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -2 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & -2 \\ 0 & 1 & 0 \end{pmatrix}^l.$$

Proof: The proof is very similar to that given to prove lemma 1 and will be given in a full version of the paper. □

3 Outline of the proof of theorem 2

Equations (8) and (9) are very easy to show. Equation (8) follows immedeatly from the shape of the matrix U. To prove (9) we consider the linear vectorial recurrency $\Omega U^s = (b_s, b_{s+1}, b_{s+2})$. The characteristical polynomial of the matrix U is $c(x) = x^3 + 2x - 1$. So it follows from the theorem of Cayley Hamilton that $c(U) = 0$, where 1 is to be replaced by the 3×3 identity matrix. From this we have $U^{3+l} + 2U^{1+l} = U^l, l \in \mathbb{Z}$ which proves the correctnes of equation (9) because (8) holds. The very difficult part of the prove consists in proving lemma 2.

Lemma 2 *The $LLL(n = 3, y = 1)$ algorithm performs on input of $\Omega(s) = \Omega U^s, s \in \mathbb{N} \cup \{0\}$ exactly three Gauß- and two update steps. These steps transform the basis $\Omega(s)$ into the basis $\Omega(s-1)$. After these operations the algorithm is before the execution of instruction (5) of algorithm 5.4. We express this suggestively by the notation: Input(Input(ΩU^s)+3 Gaußsteps) = Input(ΩU^{s-1}), $s \geq 1, s \in \mathbb{N}$. Ω and U are the matrices defined in theorem 2.*

Proof: (We sketch the proof. A complete proof is given in a full version of the paper.) In order to prove this lemma we derive from equation (9) the recurrency for the quantities given below:

$$\begin{pmatrix} |b_1^{(k)}| \\ |b_2^{(k)}| \\ |b_3^{(k)}| \\ \mathrm{P}_{12}^{(k)} \\ \mathrm{P}_{13}^{(k)} \\ \mathrm{P}_{23}^{(k)} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 4 & 0 & -4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -2 \end{pmatrix}^k \begin{pmatrix} |b_1^{(0)}| \\ |b_2^{(0)}| \\ |b_3^{(0)}| \\ \mathrm{P}_{12}^{(0)} \\ \mathrm{P}_{13}^{(0)} \\ \mathrm{P}_{23}^{(0)} \end{pmatrix}. \tag{15}$$

We prove that this recurrency corresponds to the linear ordinary differential equation

$$\frac{d^6F(x)}{dx^6} = -2\frac{d^5F(x)}{d^5} + 4\frac{d^4F(x)}{d^4} + 10\frac{d^3F(x)}{d^3} + 2\frac{d^2F(x)}{d^2} + 4\frac{dF(x)}{dx} - F(x), \tag{16}$$

where $F(x) = \sum_{l=0}^{\infty} \frac{x^l}{l!} a_k$ is a formal power series. The ordinary differential equation (16) has a fundamental system of solutions $\varphi_i(x) = \exp(\lambda_i x), 1 \leq i \leq 6$, where λ_i are the six distinct roots of the polynomial

$$x^6 + 2x^5 - 4x^4 - 10x^3 - 2x^2 - 4x + 1 \tag{17}$$

The closed form, i. e. the coefficients a_k of the formal power series, of any of the quantities from equation (15) is obtained by solving the corresponding initial value problem, for instance:

$$\frac{d^l\,F}{dx^l}(x = 0) = ||b_1^{(l)}||^2,\ 0 \leq l \leq 5. \tag{18}$$

The initial value problem could only be solved by means of a compteralgebra system (xmaple in our case) because be have proven certain connections between the roots λ_i of the polynomial given by equation (17) which can be factored in two irreducible polynomials, namely: $p(x) = x^3 + 4x^2 + 4x - 1$ and $q(x) = x^3 - 2x^2 - 1$. We have $p(t) = q(t+2)$ and if $p(\lambda_i) = 0$, $1 \leq i \leq 3$: $\sum_{i=1}^{3} \lambda_i = -4$, $\prod_{i=1}^{3} \lambda_i = 1$, $\sum_{i=1}^{3} \lambda_i^2 = 8$, $|\lambda_2| = |\lambda_1 + 2|$. So we can solve the initial value problem by performing our computations in the algebraic number field with minimal polynomial $p(x) = x^3 + 4x^2 + 4x - 1$, which is one irreducible factor of the polynomial given by equation (17). We proved that all quantities from (15) can be expressed by $||b_1^{(k)}||^2 =: B_k$ for instance:

$$P_{12}^{(k)} = \frac{1}{4}(B_k, B_{k+1}, B_{k+2}, B_{k+3}) \cdot (1, 4, 0, -1)^t,\ k \geq 0 \tag{19}$$

$$P_{13}^{(k)} = \frac{1}{4}(B_{k-1}, B_k, B_{k+1}, B_{k+2}) \cdot (1, -4, 0, -1)^t,\ k \geq 1. \tag{20}$$

Furthermore we have $\mu_{31}^{(k+1)} = -2 + \frac{\mathrm{P}_{12}^{(k)}}{||b_2^{(k)}||^2}$, $k \geq 0$ and $P_{12}^{(k)} = P_{13}^{(k+1)}$, $k \geq 0$. So the only initial value problem we need to solve is given by equation (18). We obtain the closed form form of $||b_1^{(s)}||^2$ as given in theorem 2 with some uncertainty which results from inequalities $-|z| \leq \Re(z) \leq |z|$, where $z = \lambda_2^s(-2\lambda_2^2 - \lambda_2 + 5)$ and $\Re(z)$ denotes the real part of the complex number z. These parts of the closed form of $||b_1^{(s)}||^2$ cannot be avoided though they only represent a small perturbation of the exponential growth. This are the necessary tools to prove the lemma 2 and so theorem 2. □

4 Conclusion

We have given bases such that the LLL$(n, y = 1)$ algorithm performs on input of these bases $l \cdot n$ Gaußsteps where l can be any natural number. As a result we have show that there exist inputs for the LLL$(n, y = 1)$ algorithm which cause infinite running time.

5 Appendix

5.1 Algorithms

For completeness of the paper we present the LLL(n, y) algorithm. We represent the alorithm with his subalgorithms by means of flowcharts due to Nassi and Shneiderman, [4]. The algorithm is devided into the following parts:

Flowchart	Description
5.1	Initialization
5.2	this part is called $(\star)$ in [2]
5.3	this part is called (2) in [2]
5.4	the complete LLL(n, y) algorithm

Algorithm 5.1 (Init-LLL)

Input: $(b_1, b_2, \ldots, b_n) \in \mathbb{Z}^{n \times n}$			(1)
FOR i=1 to n			(2)
	$b_i^\star = b_i$		(3)
	FOR j=1 to i-1		(4)
		$\mu_{ij} = \frac{\langle b_i, b_j^\star \rangle}{c_j}$	(5)
		$b_i^\star = b_i^\star - \mu_{ij} b_j^\star$	(6)
	$c_i = \lvert b_i^\star \rvert$		(7)
Output: μ_{ij}, c_i $1 \leq i < j \leq n$			(8)

Algorithm 5.2 (star-LLL(k, l))

Input: $(k, l, \{\mu_{kj} : 1 \leq j \leq l\}$			(1)
IF $\lvert \mu_{kl} \rvert > \frac{1}{2}$			(2)
THEN	$r = \lfloor \mu_{kl} \rceil$		(3)
	$b_k = b_k - r b_l$		(4)
	FOR $j = 1$ to $l - 1$		(5)
		$\mu_{kj} = \mu_{kj} - r \mu_{lj}$	(6)
	$\mu_{kl} = \mu_{kl} - r$		(7)
ELSE	$r = 0$		(8)
Output: $\{\mu_{k1}, \ldots, \mu_{kl}\}$, r			(9)

Algorithm 5.3 (update-LLL(k))

Input: $(k, \{\mu_{kl} : 1 \leq l < k\})$	(1)
$\mu = \mu_{k,k-1}$	(2)
$c = c_k + \mu^2 c_{k-1}$	(3)
$\mu_{k,k-1} = \frac{\mu\, c_{k-1}}{c}$	(4)
$c_k = \frac{c_{k-1}\, c_k}{c}$	(5)
$c_{k-1} = c$	(6)
Swap(b_{k-1}, b_k)	(7)
FOR $j = 1$ to $k-2$	(8)
Swap$(\mu_{k-1,j},\ \mu_{kj})$	(9)
FOR $i = k+1$ to 3	(10)
$z_1 = \mu_{k,\,k-1}\,(\mu_{i,\,k-1} - \mu_{ik}\,\mu) + \mu_{ik}$	(11)
$z_2 = \mu_{i,\,k-1} - \mu\,\mu_{ik}$	(12)
$\mu_{i,\,k-1} = z_1$	(13)
$\mu_{ik} = z_2$	(14)
IF $k > 2$	(15)
THEN $k = k-1$	(16)
Output: $(\{\mu_{kl}, b_j, c_j : 1 \leq l \leq k\},\ k)$ adjusted with respect to the swap of b_{k-1} and b_k	(17)

Algorithm 5.4 (LLL(n, y))

Input: $(b_1, b_2, \ldots, b_n) \in \mathbb{Z}^{n \times n}$	(1)
Init-LLL	(2)
$k = 2$	(3)
WHILE $k \neq n + 1$	(4)
$l = k - 1$	(5)
star-LLL(k, l)	(6)
IF $c_k < (y - \mu^2_{k,k-1}) c_{k-1}$	(7)
THEN update-LLL(k)	(8)
ELSE FOR $l = k - 2$ downto 1	(9)
star-LLL(k, l)	(10)
$k = k + 1$	(11)
Output: $(b_1, b_2, \ldots, b_n)$ y - LLL reduced	(12)

References

[1] P. van Emde Boas, Another NP-complete problem and the complexity of computing short vectors in a lattice, Technical Report, University of Amsterdam, No. 81-04 (1981).

[2] A.K. Lenstra, H.W. Lenstra, Jr. und L. Lovász, Factoring polynomials with rational coefficients, Math. Annalen 261 (1982), pp. 515–534.

[3] L. Lovász unpublished.

[4] I. Nassi, B. Shneiderman, Flowchart techniques for structured programming, SIGPLAN Notices **8** (1973), No. 8, 12 - 26.

[5] B. Vallée, Gauß algorithm revisited. Journal of Algorithms, 123 - 131, (1991).