

University of Technology Darmstadt
Department of Computer Science
Cryptography and Computeralgebra

Bachelor Thesis

August 2007

Rankin Lattice Reduction for Programmers



Zornitsa Borisova

University of Technology Darmstadt
Department Informatics

Supervised by Prof. Dr. Johannes Buchmann,
Richard Lindner

Warranty

I hereby warrant that the content of this thesis is the direct result of my own work and that any use made in it of published or unpublished material is fully and correctly referenced.

Date: Signature:

Contents:

1. Introduction.....	4
2. Mathematical Definitions and Notations.....	6
2.1. Lattice and Basis Properties.....	6
2.2. SVP & CVP.....	8
2.2.1. SVP.....	8
2.2.2. CVP.....	9
2.3. Lattice Reduction.....	9
3. LLL Revisited.....	11
3.1. Informal Description.....	11
3.2. Formal Description.....	11
4. Schnorr's BKZ Algorithm Revisited.....	12
4.1. Informal Description.....	12
4.2. Formal Description.....	13
5. Rankin's Constant and Schnorr's Algorithm.....	14
5.1. Rankin's Constant.....	14
5.2. Better Upper Bound on Schnorr's Constant.....	14
5.3. Lower Bound on Rankin's Constant.....	15
6. The Improvement of Schnorr's Algorithm.....	15
6.1. Smallest Volume Problem.....	15
6.2. The New Method: Block-Rankin-Reduction.....	16
6.3. Higher Blocksizes.....	17
7. Conclusion.....	18
References	

Abstract: Lattice basis reduction is a powerful mathematical tool with many applications. In this paper I am going to review the most classical algorithm for lattice basis reduction, the one of Lenstra, Lenstra, Lovász [1], and also describe the Schnorr's block generalization [2] of it, proposed in 1987. Recently, an improvement of Schnorr's algorithm has been proposed, using the Rankin's constant, introduced by Rankin [3] more than 50 years ago. This improvement implies the introduction of a new lattice problem: the so-called smallest volume problem [4], generalization of shortest vector problem. Because Schnorr's algorithm is based on this problem, it can be obtained a slight improvement over it as well.

1. Introduction

Informally, a lattice is a regular arrangement of points in an n -dimensional space. We can define it as an additive discrete subgroup of \mathbb{R}^n . Graphically, a lattice is the set of vertices of an n -dimensional grid. A basis of a lattice is a set of $m \leq n$ linearly independent vectors (b_1, \dots, b_m) , which generate the lattice: $L(b_1, \dots, b_m) =$

$\{ \sum_{i=1}^m x_i b_i, x_i \in \mathbb{Z} \}$ (Figure 1.1 [5]). A lattice can have infinitely many

bases, but some are more useful than others: those consisting of reasonably short and almost orthogonal vectors (Figure 1.2 [5], 1.3 [6]).

The goal of lattice reduction is to find bases that consists only of short and almost orthogonal vectors. Lately, the study of algorithms for basis reduction has become an important task, because they are used both as a tool for breaking cryptosystems and as a source of hard problems for creating cryptosystems: knapsack cryptosystems [8], RSA in special settings [9, 10]; GGH cryptosystem [7], the cryptographic scheme of Ajtai and Dwork [25], etc. From the mathematical point of view, the history of lattice reduction goes back to Lagrange [11], Hermite [12], Korkine and Zolotarev, among others, and also to Minkovski's geometry of numbers.

But the most important modern advance in the algorithmic theory of lattice reduction is the method discovered by Lenstra, Lenstra, Lovász (LLL). The idea of LLL algorithm is to find a moderately small vector in a lattice L in polynomial time. This algorithm approximates the shortest vector in the lattice to within some factor. The same holds for the improvement of LLL introduced by Schnorr. He constructed a hierarchy of polynomial-time lattice basis reduction algorithms [21]. These algorithms, called blockwise Korkine-Zolotarev reductions or BKZ-reductions, can compute a reduced basis, ranging from LLL-reduced to KZ-reduced (definition comes later), depending on a parameter, called the blocksize.

The problem of finding the smallest vector in a general lattice remains an exponentially hard problem using even the best known algorithms.

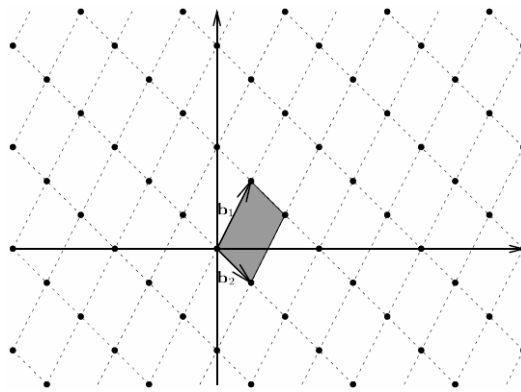


Figure 1.1: 2-dimensional lattice [5]

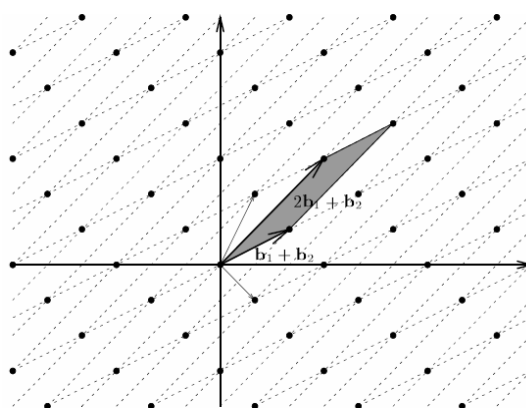


Figure 1.2: Same lattice, different bases [5]

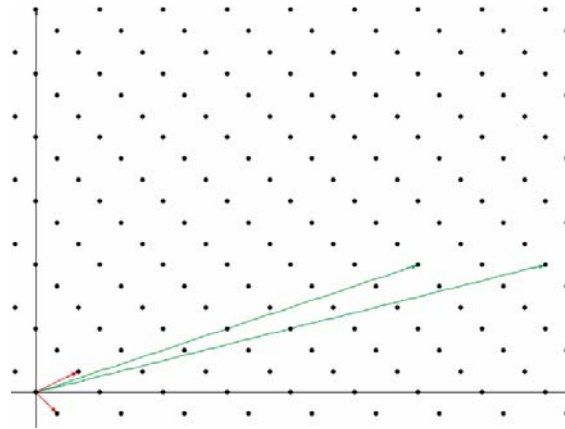


Figure 1.3: Good and bad bases [6]

In this paper I am going to review the newest improvement of LLL and Schnorr's BKZ, stated in paper [4]. In his paper [19], Schnorr introduced a constant, which measures the quality of the output basis. Then Nicolas Gama, Nick Howgrave-Graham, Henrik Koy, Phong Q. Nguyen [4] found a connection between this constant and a constant, introduced by Rankin [3]. It is a generalization of the Hermite constant, used in LLL. Using this constant, an improvement of Schnorr's algorithm has been found. It still follows the LLL framework, but the subroutine where HKZ-reduction (see page 9) of a basis is made, is replaced by the so-called *smallest volume problem*. It is also introduced in [4] and it is a generalization of the smallest vector problem. The new method, using smallest volume problem, is called block-Rankin-reduction.

Road map: Section 2 gives the necessary mathematical background for the subject. Sections 3 and 4 make reviews of LLL and BKZ algorithms. Sections 5 and 6 rely on the paper of Nicolas Gama, Nick Howgrave-Graham, Henrik Koy and Phong Q. Nguyen "Rankin's Constant and Blockwise Lattice Reduction". Section 7 concludes the paper.

2. Mathematical Definitions and Notations

2.1. Lattice and Basis Properties

Let $\|\cdot\|$ denote the *Euclidean norm* and $\langle \cdot, \cdot \rangle$ - the *inner product* in \mathbb{R}^n . Let a matrix be denoted with capital letters and represented using row-vectors. The coefficients of the matrix are written with small letters: m_{ij} .

The *dimension* of a lattice is the number of basis vectors which span L . If the dimension of the space equals the rank, then we say that the lattice is *full-rank* (or equivalently, if the basis matrix is square matrix). A *Gram-matrix* is a $k \times k$ symmetric positive-definite matrix $(\langle v_i, v_j \rangle)_{1 \leq i, j \leq k}$ of all the inner products of the row-vectors in the matrix. A *Gram-matrix* is denoted

with $G(v_1, \dots, v_k)$. On the place of the ij -th entry of the matrix stays the inner product $(\langle v_i, v_j \rangle)_{1 \leq i, j \leq k}$.

Volume. The *volume* of a set of vectors $[v_1, \dots, v_k]$ is $(\det G(v_1, \dots, v_k))^{1/2}$, which equals zero, if the vectors are linearly dependent. The volume (or the *determinant*) of a lattice $vol(L)$ does not depend on the choice of the basis and can be interpreted as the geometric volume of the parallelepiped naturally spanned by the basis vectors.

Direct sum. If L_1 and L_2 are two lattices, for which $span(L_1) \cap span(L_2) = \{0\}$, then we define $L_1 \oplus L_2 = \{u + v, u \in L_1, v \in L_2\}$, whose dimension is $\dim L_1 + \dim L_2$. This is the smallest lattice containing L_1 and L_2 .

Successive Minima. Let $\lambda_r(L)$ denote the smallest radius of a zero-centered ball, containing r linearly independent vectors of L . The first minimum $\lambda_1(L)$ is the norm of the shortest non-zero vector of L . The inequalities $\lambda_1(L) \leq \dots \leq \lambda_n(L)$ hold. The theorem of Minkowski states an upper bound for $\lambda_1(L)$: $\lambda_1(L) \leq \gamma_n \det(B)^{1/n}$ [2], where γ_n is introduced in the following paragraph.

Hermite's constant. The Hermite's invariant of a lattice is defined by $\gamma(L) = (\lambda_1(L) / vol(L)^{1/n})^2$. Hermite's constant γ_n is the maximal value over all n -dimensional lattices. For $1 \leq n \leq 8$ and $n=24$ the exact value is known. The best bounds known for the Hermite's constant are:

$$\frac{n + \log(\pi n)}{2\pi e} \leq \gamma_n \leq \frac{1.744n}{2\pi e} (1 + o(1)) \quad (\text{see [13, 14]}), \text{ where } o(1) \text{ stands for}$$

a number, which even for big n , is smaller than 1. The lower bound follows from the Minkowski-Hlawka theorem (see [15]).

Projected lattice. If we have a basis $[b_1, \dots, b_n]$ of L , let π_i denote the orthogonal projection over $span(b_1, \dots, b_{i-1})^\perp$. Then $\pi_i(L)$ is an $(n+1-i)$ -dimensional lattice. Compositions of lattices also exist: if $i > j$, then $\pi_i \circ \pi_j = \pi_j \circ \pi_i = \pi_i$. Another projected lattice property is:

$$\pi_i(L) = \pi_i(L(b_1, \dots, b_n)) = L(\pi_i(b_1), \dots, \pi_i(b_n)).$$

Let $i=3$ and $[b_1, b_2, b_3]$ be the canonical basis, then π_3 denotes the orthogonal projection of all lattice points over $span(b_1, b_2)^\perp$, which means a projection over the z -axis. The new lattice has a dimension $(n+1-i)=3+1-3=1$.

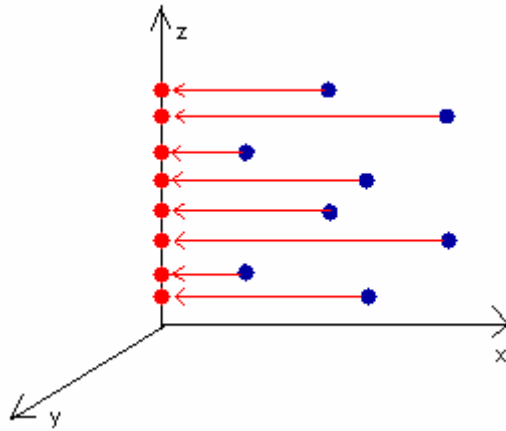


Figure 2.2: $\pi_3(L)$: an example for projected lattice

2.2. Shortest Vector Problem (SVP) and Closest Vector Problem (CVP)

The most famous algorithmic problems concerned with lattices:

2.2.1. SVP

SVP is of prime importance in cryptography since a now quite large family of public-key cryptosystems relies more or less on it. When given a lattice, the SVP consists of finding a shortest non-zero vector in L , i.e. a vector $u \in L \setminus \{0\}$ such that $\|u\| \leq \|v\| \forall v \in L \setminus \{0\}$. In other words, given a lattice, find the smallest nonzero lattice vector closest to the origin. It has been proved that SVP is \mathcal{NP} -hard (see [7, page 3] or [16]). No polynomial algorithm is known for approximating SVP to within a polynomial factor n . Nevertheless, the LLL algorithm achieves to approximate SVP to within $f(n) = 2^{(n-1)/2}$ in polynomial time (Figure 2.2 [17] and [7]).

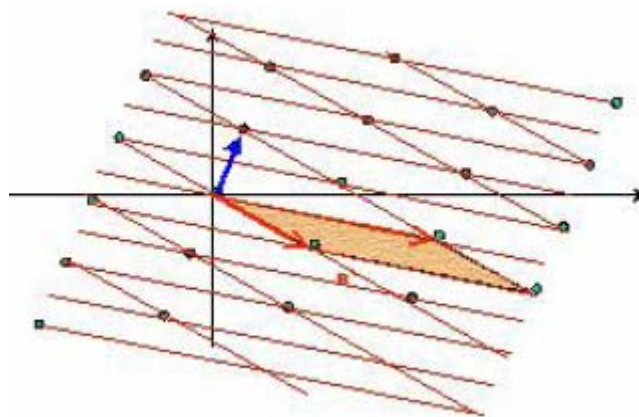


Figure 2.2: SVP [17]

2.2.2. CVP

The CVP is somewhat similar to SVP. In CVP, one considers a vector $x \in \check{Y}^n$, not necessarily in the lattice, and wants to find a point $u \in L$, minimizing the distance between x and u , i.e., minimizing $\|x - u\|$. Finding a lattice vector u such that $\forall v \in L, \|u - x\| \leq f(n) \|v - x\|$ is the approximation problem corresponding to CVP. Another formulation is as follows: given a lattice and a target point, y for example, find the lattice point closest to the target. It has been proven that SVP is not harder than CVP, equivalently this means that given an oracle which approximates CVP to within a factor $f(n)$, there exists a routine which approximates SVP to within the same factor $f(n)$ in polynomial time (Figure 2.3 [17] and [7]).

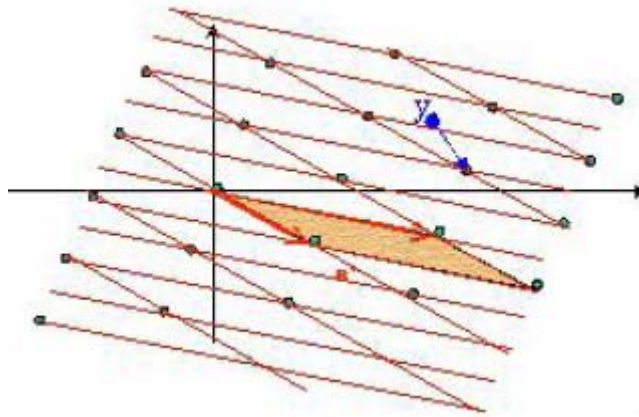


Figure 2.3: CVP [17]

2.3. Lattice Reduction

Gram-Schmidt Orthogonalisation (GSO). It was already mentioned that for the purposes of cryptography we look for a basis of lattice with short and almost orthogonal vectors. The method for making the basis vectors orthogonal is called Gram-Schmidt Orthogonalisation. It runs as follows: given m linearly independent vectors $b_1, \dots, b_m \in \check{Y}^n$, define the vectors $b_1^*, \dots, b_m^* \in \check{Y}^n$ by the recurrence:

$$b_1^* = b_1, \quad b_i^* = b_i - \sum_{1 \leq j < i} \mu_{i,j} b_j^*, \quad \text{for } 2 \leq i \leq n,$$

where $\mu_{i,j}$ are called the Gram-Schmidt coefficients, $\mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$ for

$i \geq j$ and $\mu_{i,j} = 0$ else. With this definition b_i^* is an orthogonal projection of b_i on the space which is orthogonal to b_1, \dots, b_{i-1} . The resulting column-vectors b_j^* form the GSO matrix of the basis.

There exist two quantities which help for measuring the quality of a basis. The first is called *the approximation factor*. $\|b_1\|/\lambda_1(L)$. The second

useful quantity is *the Hermite factor*. $\|b_1\|/\text{vol}(L)^{1/n}$. It should be noted that $\text{vol}(B) = \prod_{i=1}^n \|b_i^*\|$, where B is some basis. Another used inequality is [1]: $\lambda_1(L(b_1, \dots, b_n)) \geq \min_{1 \leq i \leq n} \|b_i^*\|$.

Size-reduction. A basis $[b_1, \dots, b_n]$ is *size-reduced* with factor $\eta \geq 1/2$ if its GSO matrix satisfies $|\mu_{i,j}| \leq \eta$ for all $1 \leq j < i \leq n$. Size-reduction is typically achieved by successively size-reducing individual vectors. It was introduced by Hermite.

HKZ-reduced. A basis $[b_1, \dots, b_n]$ is *Hermite-Korkine-Zolotarev-reduced* if it is size-reduced and if b_i^* is a shortest vector of the projected lattice $\pi_i(L)$ for all $1 \leq i \leq n$. In particular, the first basis vector is the shortest in the lattice. HKZ-reduction is very strong, but very expensive. On the contrary, LLL-reduction is fairly cheap, but an LLL-reduced basis is of much lower quality. HKZ-reduction means actually a reduction in a 2-dimensional space, but it has become a common notion for this kind of basis reduction, even in an n -dimensional space.

LLL-reduction. A basis $[b_1, \dots, b_n]$ is *LLL-reduced* [1] with factor (δ, η) ($1/4 < \delta \leq 1, 1/2 \leq \eta < \sqrt{\delta}$) if the basis is size-reduced with factor η and if its GSO matrix satisfies the *(n-1) Lovász conditions*:

$(\delta - \mu_{i+1,i}^2) \|b_i^*\|^2 \leq \|b_{i+1}^*\|^2$ (this means that in $(b_1, \dots, b_{i-2})^\perp$, b_{i-1} is approximately shorter than b_i). LLL-reduction usually refers to the factor $(3/4, 1/2)$, because this was the factor described in the original LLL-paper [1]. Actually, the closer δ and η to 1 and $1/2$ are, the more reduced the basis is. When the reduction factor is close to $(1, 1/2)$, Lovász conditions and size-reduction imply the *Siegel conditions*

[18]: $\|b_i^*\|^2 \leq (4/3 + \varepsilon) \|b_{i+1}^*\|^2$ for some small $\varepsilon > 0$. The LLL-reduction implies that the norms $\|b_1^*\|, \dots, \|b_n^*\|$ of the GSO vectors never drop too fast:

intuitively, the vectors are not too far from being orthogonal. Such bases have useful properties, like providing exponential approximation to SVP and CVP. In particular, their first vector is relatively

short: $\|b_1\|^2 \leq (4/3 + \varepsilon)^{i-1} \|b_i^*\|^2, 1 \leq i \leq n$.

In [19] Schnorr introduced a constant to globally measure the drop of the $\|b_i^*\|$ of $2k$ -dimensional HKZ-bases:

$$\beta_k = \max_{\substack{L: 2k\text{-dim. lattice} \\ H: \text{HKZ-basis of } L}} \left(\frac{\|h_1^*\| \times \dots \times \|h_k^*\|}{\|h_{k+1}^*\| \times \dots \times \|h_{2k}^*\|} \right)^{2/k}, \text{ which can be rewritten as:}$$

$$\beta_k = \max_{\substack{L: 2k\text{-dim. lattice} \\ H: \text{HKZ-basis of } L}} \left(\frac{\text{vol}(h_1, \dots, h_k)}{\text{vol}(\pi_{k+1}(h_{k+1}), \dots, \pi_{k+1}(h_{2k}))} \right)^{2/k}, \text{ where } h_i\text{-s are the}$$

vectors in a HKZ-reduced basis of L . Also the upper and lower bounds of the constant are proven: $k^\varepsilon \leq \beta_k \leq 4k^2$. The value of β_k is very important because it gives us information about the quality of the output basis. In the paper of Nicolas Gama, Nick Howgrave-Graham, Henrik Koy, Phong Q. Nguyen a better upper bound is found and also an explicit lower bound,

which will be reviewed later.

3. LLL Revisited

3.1. Informal Description

The LLL algorithm is an iterative algorithm. It solves the problem of computing a reduced basis of a lattice, given an arbitrary basis. The basis vectors are considered pairwise and each vector b_i is required to be the shortest in the 2-dimensional projected lattice generated by b_i and b_{i+1} . More precisely, at the beginning of each loop iteration, the first i vectors are already LLL-reduced, then the $(i+1)$ -th vector is size-reduced and so on. A check is made: if two consecutive basis vectors b_{i+1} and b_i fulfill the Lovász condition, then i is incremented, if not, the two vectors are swapped and the counter i is decremented. The loop ends when i reaches the value of n . At the end, when all 2-dimensional lattices are already reduced, every ratio $\|b_i^*\|/\|b_{i+1}^*\|$ is roughly less than $\gamma_2 = \sqrt{4/3}$, which is exactly the Siegel's condition, mentioned above.

If L is a full-rank integer lattice of dimension n and B is an upper bound on the $\|b_i\|$'s, then the complexity of the LLL algorithm is

$O(n^6 \log^3 B)$. Nevertheless, the best known complexity until now is

$O(n^6 \log^2 B)$ for large entries. In other words, given an arbitrary basis of a lattice L , the LLL algorithm computes a reduced basis of L in polynomial time (for proof: [19]).

3.2. Formal Description

Algorithm 2 The LLL basis reduction algorithm

Require: an arbitrary basis $b_1, \dots, b_n \in \mathbb{R}^m$ of a lattice \mathcal{L} .

Ensure: b_1, \dots, b_n is a reduced basis of \mathcal{L} .

```

1: Compute  $b_1^*, \dots, b_n^*$ , the Gram-Schmidt orthogonal basis of  $b_1, \dots, b_n$ 
2:  $i \leftarrow 2$ 
3: while  $i \leq n$  do
4:   for  $j = i - 1, \dots, 1$  do
5:      $b_i \leftarrow b_i - \lceil \mu_{ij} \rceil b_j^*$ 
6:     Update the  $b_i^*$ 's
7:   if  $i > 1$  and  $\|b_{i-1}^*\| > 2 \|b_i^*\|$  then
8:     Swap  $b_{i-1}$  and  $b_i$  and update the  $b_i^*$ 's
9:      $i \leftarrow i - 1$ 
10:  else
11:     $i \leftarrow i + 1$ 
12:  end if
13: end for
14: end while

```

The pseudocode above is taken from [7] and describes the steps of the algorithm. Steps 7 and 8 can be seen as a local basis reduction in a 2-dimensional lattice, which is not a hard problem. A polynomial algorithm was given by Gauss (the so-called Gaussian algorithm). Step 11 increments i and step 9 decrements it, so, on first sight it seems that the algorithm may never end. It is actually proven that Step 9 is executed only finitely many times, whence follows that LLL terminates and as already said, it terminates in polynomial time. In practice, LLL generally does better than this, but also, if n is large, LLL will not find a vector just a few times longer than $\lambda_1(L)$. It will guarantee a vector whose length is at most $(4/3)^{(n-1)/2} \lambda_1(L)$.

4. Schnorr's BKZ Algorithm Revisited

4.1. Informal Description

Definition 1: A basis $B=[b_1, \dots, b_n]$ of a lattice L is said to be *Korkine-Zolotarev-reduced* (or *KZ-reduced*) if B is size-reduced and b_i^* is a shortest vector in $L[\pi_i(b_i), \pi_i(b_{i+1})]$, where b_i^* is the i -th GS-orthogonalised vector.

KZ-reduction is a generalization of the Hermite reduction, which is for 2-dimensional lattices.

Definition 2: A basis $B=[b_1, \dots, b_n]$ of a lattice L is said to be *block Korkine-Zolotarev-reduced with blocksize k* (*k-BKZ-reduced*) if B is size-reduced and $L([\pi_i(b_i), \dots, \pi_i(b_{i+k-1})])$ is KZ-reduced.

With these definitions, it follows that a basis B of dimension n is LLL-reduced if and only if B is 2-BKZ-reduced, and B is KZ-reduced if and only if B is n -BKZ-reduced [2].

Schnorr defined an LLL-based family of algorithms, whose performances depend on a parameter k , called the blocksize. These algorithms, named block-Korkine-Zolotarev reductions (or BKZ-reductions), can compute a reduced basis ranging from LLL-reduced to KZ-reduced, depending on this parameter, the blocksize. The idea is that instead of reducing pairs of basis vectors, the reduction is over blocks with size k of the basis vectors. The reduction in LLL is actually 2-dimensional KZ-reduction.

The implementation of BKZ differs from LLL only in the step, where the swapping of two vectors is accomplished. It is interchanged with a more complicated procedure, namely the block-KZ-reduction. LLL-reduction of a lattice guarantees a vector whose length is at most $2^{n/2} \lambda_1(L)$ and the block KZ reduction improves the bound to $(1 + \varepsilon)^{n/2} \lambda_1(L)$ at the expense of adding a factor of $1/\varepsilon$ to its running time [20].

The block reduction works well in practice, but it is still not proven to execute in polynomial time. An alternative notion of reduction does have

a polynomial-time algorithm. It is called *semi-k-reduction*. In this reduction, KZ-reduction is performed on disjoint blocks of k vectors. Another condition is added on the last vector of the block and on the first of the next block, so that we can ensure a globally good solution. More formally:

Definition 3: A basis $B=[b_1, \dots, b_n]$ of a lattice L is said to be *semi-k-reduced* if B is size-reduced, $|b_{ik}^*|^2 \leq 2|b_{(i+1)k}^*|^2$, for $0 < i < m$, and $b_{(i+1)k}^*, \dots, b_{(i+1)k}^*$ are KZ-reduced for $0 \leq i < m$.

Schnorr [21] presents an improvement to the semi-k-reduction, called semi-block-k-reduction, that includes a stronger condition on adjacent blocks. This reduction also has a polynomial time algorithm.

Until now the best practical high-dimensional lattice reduction algorithm known is Schnorr's semi-block-2k algorithm.

Definition 4: basis $B=[b_1, \dots, b_n]$ of a lattice L is said to be *semi-block-2k-reduced* if B is LLL-reduced, S_i is HKZ-reduced for $1 \leq i \leq n/k$, and $(\text{vol}(S_i)/\text{vol}(S_{i+1}))^2 \leq (1 + \varepsilon)\beta_k^k$ for $1 \leq i \leq n/k$, where $S_i = [\pi_{ik-k+1}(b_{ik-k+1}), \dots, \pi_{ik-k+1}(b_{ik+k})]$ for $1 \leq i \leq n/k$ [4].

4.2 Formal Description

Schnorr's semi block-2k reduction
1: while $i \leq n/k$ do
2a: HKZ-reduce S_i , do the transformations on the basis vectors, not just on the projections
2b: Size-reduce $\mathbf{b}_{ik-k+1}, \dots, \mathbf{b}_{ik}$.
3: $B' \leftarrow$ copy of B
4: Try to decrease $\text{vol}(S_i)$ in B' :
4a: • by swap of $(\mathbf{b}_{ik}, \mathbf{b}_{ik+1})$
4b: • by HKZ-reducing L_i
5: if $\text{vol}(S_i)$ can lose a factor $\frac{1}{(1+\varepsilon)}$
then
6: • perform the changes on B
7: • $i \leftarrow \max(i - 1, 1)$
8: else $i \leftarrow i + 1$
9: endwhile

Schnorr's semi-block-2k-reduction replaces the vectors b_i^* by k -dimensional blocks of vectors S_i , as defined above. Like in LLL, at every iteration every such block is reduced.

5. Rankin's Constant and Schnorr's Algorithm

5.1. Rankin's Constant

In [3] Rankin defined an invariant $\gamma_{n,m}(L)$ for an n -dimensional lattice L and $1 \leq m \leq n$:

$$\gamma_{n,m}(L) = \max_{\substack{x_1, \dots, x_m \in L \\ \text{vol}(x_1, \dots, x_m) \neq 0}} \left(\frac{\text{vol}(x_1, \dots, x_m)}{\text{vol}(L)^{m/n}} \right)^2,$$

which can be rewritten for lattices as:

$$\gamma_{n,m}(L) = \max_{\substack{S \text{ sublattice of } L \\ \dim S = m}} \left(\frac{\text{vol}(S)}{\text{vol}(L)^{m/n}} \right)^2,$$

because the minimum is taken over sets of m linearly independent vectors of L , which form a sublattice S of L .

Then the Rankin's constant is defined as the maximum of $\gamma_{n,m}(L)$ over all n -dimensional lattices: $\gamma_{n,m} = \max \gamma_{n,m}(L)$. It can be seen that

$\gamma_{n,n}(L) = 1$ and $\gamma_{n,1}(L) = \gamma_n(L)$, where $\gamma_n(L)$ is the n -th Hermite's constant.

The conditions, which these constants satisfy are proven also in [3]:

$$\forall n \in \mathbb{I} : \gamma_{n,1} = \gamma_n$$

$$\forall n, m \in \mathbb{I}, m < n : \gamma_{n,m} = \gamma_{n,n-m}$$

$$\forall r \in [m+1, n-1] : \gamma_{n,m} \leq \gamma_{r,m} (\gamma_{n,r})^{m/r}.$$

The value of $\gamma_{4,2}$ is $3/2$ and the values, which correspond to the nine Hermite's constants, are also known.

The following theorem gives us a relation between the Rankin's and Schnorr's constants:

Theorem 1: For $k \geq 1$, $(\gamma_{2k,k})^{2/k} \leq \beta_k$.

Proof: [4].

Now the improvement of the upper bound of Schnorr's constant will be introduced, which gives, respectively, better bounds for Rankin's constant. With a lower bound for the Rankin's constant itself, the interval where it belongs, can be stated. Exactly on these improvements and whence the improvement, following from Rankin's constant used in the smallest volume problem, the improvement of the whole Schnorr's algorithm is based.

5.2. Better Upper Bound on Schnorr's Constant

Theorem 2: For all $k \geq 2$, Schnorr's constant β_k satisfies: $\beta_k \leq (1 + k/2)^{2 \ln 2 + 1/k}$. Asymptotically, it satisfies

$$\beta_k \leq 1/10 k^{2 \ln 2}.$$

Proof: [4].

As said before, Schnorr's constant measures the output quality of the basis. So, without changing Schnorr's algorithm at all, better output quality is proven. The upper bound, which was proven to be the best up to now, was $\beta_k \leq 4k^2$, but the new one reduces the exponent 2 to $2/n^2$, which is approximately 1,386 and the coefficient 1/10 is smaller than the coefficient 4 about 40 times.

5.3. Lower Bound on Rankin's Constant

It was shown by Ajtai in [22] that $\beta_k \geq k^\varepsilon$ for small size of blocks and for $\varepsilon > 0$, but no explicit value of ε was known. It is already known from theorem 1. that $(\gamma_{2k,k})^{2/k} \leq \beta_k$. The following theorem gives us a lower bound for the Rankin's constant, which leads to a lower bound for β_k as well (explicitly, $\beta_k \geq (\gamma_{2k,k})^{2/k} \geq k/12$):

Theorem 3: Rankin's constant satisfies $(\gamma_{2k,k})^{2/k} \geq k/12$ for all $k \geq 1$.

Proof: [4].

6. The Improvement of Schnorr's Algorithm

In Schnorr's algorithm the quality of the output basis depends on the upper bound of $\text{vol}(S_1) / \text{vol}(S_2)$, where $S_1 = L(b_1, \dots, b_k)$ and $S_2 = \pi_{k+1}(L)$. That is why we are trying to minimize this proportion. The newly introduced *smallest volume problem* deals exactly with this.

6.1. The Smallest Volume Problem

Description of the problem: given an n -dimensional lattice L and the integer m , $1 \leq m \leq n$, find an m -dimensional sublattice S of L such that $\text{vol}(S)$ is minimal. That is: $\text{vol}(S) / \text{vol}(L)^{m/n} = \sqrt{\gamma_{n,m}(L)}$. We get as a result square root of the Rankin's constant. For $m = 1$ we have simply the shortest vector problem. When $(n, m) = (2k, k)$, the problem is called *the half-volume problem*.

A basis B of an n -dimensional lattice L is *m-Rankin-reduced* if its first m vectors solve the smallest volume problem.

If a basis is already m -Rankin-reduced and we apply the LLL algorithm, then a swap to the pair $(m, m+1)$ never occurs, because the volume of the first m vectors never increases. So, an m -Rankin-reduced basis is also an LLL-reduced one.

6.2. The New Method: Block-Rankin-Reduction

2k-Block-Rankin-reduced. A basis B is *2k-Block-Rankin-reduced* with factor $\delta \in [1/2, 1)$ if it is LLL-reduced with factor $(1/2, \delta)$ and all the blocks S_i and L_i satisfy: $\text{vol}(S_i) / \text{vol}(S_{i+1}) = 1/\delta \gamma_{2k,k}(L_i)$. From the definition (Definition 4) of semi-block-2k-reduction is known that such a basis should fulfill $(\text{vol}(S_i) / \text{vol}(S_{i+1}))^2 \leq (1 + \varepsilon) \beta_k^k$. With the above definition for 2k-Block-Rankin-reduced we got that $\text{vol}(S_i) / \text{vol}(S_{i+1}) = 1/\delta \gamma_{2k,k}(L_i)$, which is proved to be a better result, because of the relation between $\gamma_{2k,k}$ and β_k^k (chapter 5.3). Hence, with 2k-Block-Rankin-reduction a better quality of the output basis is obtained (because the right side is a smaller number, the ratio on left side needs to be made smaller as well). One may think that the better quality of the basis would be at the expense of more running time, but this is not the case, as explained below.

The smallest possible block-Rankin-reduction is when $k=2$. Then we have 4-dimensional-Rankin-reduction. A pseudocode for it runs as follows:

4-dimensional Rankin-reduction

Input: An HKZ-reduced basis $[b_1, \dots, b_4]$ of a 4-dim lattice

Output: A Rankin-reduced basis $[c_1, c_2, c_3, c_4]$ minimizing $\text{vol}(c_1, c_2)$

```

1: if  $\|b_2^*\| > \sqrt{\frac{4}{3}} \|b_1\|$  then
2:   return  $(b_1, b_2, b_3, b_4)$ 
3: end if
4:  $(u, v) \leftarrow (b_1, b_2)$ 
5: for each lattice vector  $c_1$  shorter than  $\sqrt{\frac{4}{3}} \|b_1\|$  do
6:   find the shortest vector  $c_2$  in the lattice projected over  $c_1^\perp$  (We can limit the
   enumeration to  $\|c_2\|$  lower than  $\frac{\text{vol}(b_1, b_2)}{\|c_1\|}$ ).
7:   if  $\text{vol}(c_1, c_2) < \text{vol}(u, v)$  then  $(u, v) \leftarrow (c_1, c_2)$ 
8: end for
9: compute  $c_3$  and  $c_4$  a reduced basis of the lattice projected over  $(u, v)^\perp$ 
10: return  $(u, v, c_3, c_4)$ 

```

The input required from this algorithm is HKZ-reduced basis. This means that there are only finitely many vectors c_1 , which we look for in Step 5. It follows that the cost of the algorithm is at most constant times more expensive than a normal HKZ-reduction of a 4-dimensional lattice.

Theorem 4: Let (b_1, \dots, b_4) be an HKZ-reduced basis of a lattice L . Let λ_1 and λ_2 denote respectively $\lambda_1(L)$ and $\lambda_2(L)$. For all c_1 and c_2 , such that $\text{vol}(c_1, c_2) \leq \text{vol}(b_1, b_2)$ and (c_1, c_2) is reduced: $\|c_1\| \leq \|c_2\|$ and

$$\|c_1^*\|^2 \leq 4/3 \|c_2^*\|^2.$$

1. Then c_1 satisfies: $\lambda_1^2 \leq \|c_1\|^2 \leq 4/3 \lambda_1^2$.

2. If $\lambda_2 > \sqrt{4/3} \lambda_1$, then $\text{vol}(c_1, c_2) = \text{vol}(b_1, b_2)$, given by HKZ-reduction.

3. Otherwise c_2 satisfies $\|c_2\|^2 \leq (4/3 \lambda_1)^2$.

Proof: [4].

This theorem shows that the 4-dimensional Rankin-reduction can efficiently solve the half-volume problem in dimension 4, given as input a HKZ-basis.

Now, what the method block-Rankin-reduction looks like. If we plug the algorithm for 4-dimensional-Rankin-reduction in the algorithm for semi 4-block-reduction, a new polynomial-time algorithm is obtained. The output of this algorithm, called the block-Rankin-reduction, is of a bit better quality than the one of Schnorr's semi-4-block-reduction. This is so, because the 2k-Block-Rankin-reduction is used.

2k-block-Rankin-reduction

Input: A basis $B = [b_1, \dots, b_n]$ of a lattice and $\delta \in [\frac{1}{2}; 1[$

Output: A semi block 2k-reduced basis.

```

1:  $i \leftarrow 1$ ;
2: while  $i \leq n/k$  do
3:   LLL-reduce  $S_i$  with factor  $\delta$ , do the transformations on the basis vectors, not
   just on their projections
4:   return  $B$  if  $i = n/k$ .
5:    $B_{\text{tmp}} \leftarrow B$ ; k-Rankin-reduce  $L_i$  in  $B_{\text{tmp}}$ 
6:   if  $\text{vol}(S_i)$  in  $B_{\text{tmp}} \leq \delta \text{vol}(S_i)$  in  $B$  then
7:      $B \leftarrow B_{\text{tmp}}$ ;  $i \leftarrow i - 1$ 
8:   else
9:      $i \leftarrow i + 1$ 
10:  end if
11: end while

```

6.3. Higher Blocksizes

In dimensions higher than 4, the situation looks more difficult because up to now an algorithm, which does not use a gigantic exhaustive search and which solves the half-volume problem, is not known. There exists only an approximation algorithm, called the transference reduction. The problem is that it achieves a volume ratio lower than $1/95 k^2$ and hence, for $k \geq 10$ this kind of reduction does not work any more, because the volume ratio is already greater than 1.

7. Conclusion

The algorithm of block- $2k$ -Rankin-reduction gives out better quality of the output basis than the $2k$ -BKZ-reduction, known up to now as the best practical reduction algorithm. The reason is that the Rankin-reduction demands more on the basis in order to reduce it. The $2k$ -BKZ-reduction algorithm outputs a basis for which $(\text{vol}(S_i)/\text{vol}(S_{i+1}))^2 \leq (1 + \varepsilon)\beta_k^k$ and the basis vectors in the $2k$ -Block-Rankin-reduction must fulfill

$\text{vol}(S_i) / \text{vol}(S_{i+1}) = 1/\delta \gamma_{2k,k}(L_i)$, which is a better result due to the new relation between β_k and $\gamma_{2k,k}$. This improvement doesn't affect the running time: it takes at most constant times more time to do the Rankin-reduction.

However, there is a serious drawback: the algorithm is better only for $k < 10$.

References:

1. Lenstra A., Lenstra, H., Lovasz, L., *Factoring polynomials with rational coefficients*, Mathematische Ann. 261 (1982), 513-534. [The famous LLL algorithm.]
2. C. P. Schnorr and M. Euchner. *Lattice basis reduction: Improved practical algorithms and solving subset sum problems*, Math. Programming, 66:181-199, 1994.
3. R. A. Rankin. *On positive definite quadratic forms*. J. London Math. Soc., 28:309-314, 1953.
4. Nicolas Gama, Nick Howgrave-Graham, Henrik Koy, Phong Q. Nguyen. *Rankin's Constant and Blockwise Lattice Reduction*
5. Daniele Micciancio. *On the hardness of the shortest vector problem*
6. Joseph H. Silverman. *An Introduction to the Theory of Lattices and Applications to Cryptography*
7. Thomas Baignères. *Cryptosystems and LLL*
8. A. M. Odlyzko. *The rise and fall of knapsack cryptosystems*. In Proc. of Cryptology and Computational Number Theory, volume 42 of Proc. of Symposia in Applied Mathematics, pages 75-88. AMA, 1989.
9. D. Boneh. *Twenty years of attacks on the RSA cryptosystem*. Notices of the AMS, 46(2):203-213, 1999.
10. D. Coppersmith. *Small solutions to polynomial equations, and low exponent RSA vulnerabilities*. J. of Cryptology, 10(4):233-260, 1997. Revised version of two articles from Eurocrypt '96.

11. J. L. Lagrange. *Recherches d'arithmétique*. *Nouveaux Mémoires de l'Académie de Berlin*, 1773
12. C. Hermite. *Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres, deuxième lettre*. *J. Reine Angew. Math.*, 40:279–290
13. J. Milnor and D. Husemoller. *Symmetric bilinear forms*. *Math. Z.*, 1973.
14. J. Conway and N. Sloane. *Sphere Packings, Lattices and Groups*. Springer-Verlag, 1998. Third edition
15. Peter M. Gruber. *Convex and discrete geometry: ideas, problems and results*
16. \mathcal{NP} -hard problems-definition: www.en.wikipedia.org/wiki/NP-hard
17. Daniele Micciancio. *From Ajtai-Dwork to NTRU: The design of practical lattice based cryptosystems*
18. J. W. S. Cassels. *Rational quadratic norms*, vol.13 of London Mathematical Society Monographs
19. J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2nd edition, 2003
20. Matthew C. Cary. *Lattice Basis Reduction. Algorithms and Applications*.
21. C. P. Schnorr. *A hierarchy of polynomial-time lattice basis reduction algorithms*. *Theoretical computer science*
22. M. Ajtai. *The worst-case behaviour of Schnorr's algorithm approximating the shortest vector in a lattice*. In *Proceedings of the 35th annual ACM symposium of theory of computing*
23. M. I. Boguslavsky. *Radon transforms and packings*. *Discrete Appl. Math*
24. J. L. Thunder. *Higher-dimensional analogs of Hermite's constant*.
25. P. Q. Nguen and J. Stern. *The two faces of lattices in cryptography*. In *Proc. Of CALC'01*, volume 2146 of LNCS, Springer-Verlag

