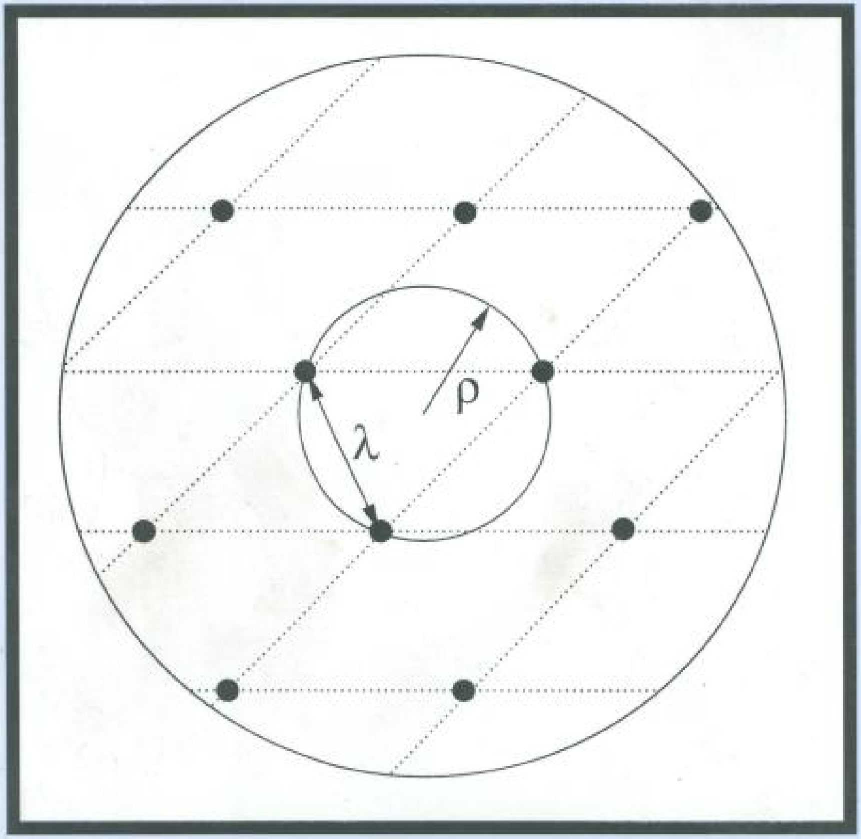
COMPLEXITY OF LATTICE PROBLEMS

A Cryptographic Perspective

Daniele Micciancio

Shafi Goldwasser



COMPLEXITY OF LATTICE PROBLEMS

A Cryptographic Perspective

THE KLUWER INTERNATIONAL SERIES
IN ENGINEERING AND COMPUTER SCIENCE

COMPLEXITY OF LATTICE PROBLEMS
A Cryptographic Perspective

Daniele Micciancio
University of California, San Diego

Shafi Goldwasser
The Massachusetts Institute of Technology,
Cambridge

SPRINGER SCIENCE+BUSINESS MEDIA, LLC

ISBN 978-1-4613-5293-8 ISBN 978-1-4615-0897-7 (eBook)
DOI 10.1007/978-1-4615-0897-7

Library of Congress Cataloging-in-Publication Data

A C.I.P. Catalogue record for this book is available
from the Library of Congress.

Printed on acid-free paper.

Contents

Preface

Lattices are geometric objects that can be pictorially described as the set of intersection points of an infinite, regular n-dimensional grid. Despite their apparent simplicity, lattices hide a rich combinatorial structure, which has attracted the attention of great mathematicians over the last two centuries. Not surprisingly, lattices have found numerous applications in mathematics and computer science, ranging from number theory and Diophantine approximation, to combinatorial optimization and cryptography.

The study of lattices, specifically from a computational point of view, was marked by two major breakthroughs: the development of the LLL lattice reduction algorithm by Lenstra, Lenstra and Lovász in the early 80's, and Ajtai's discovery of a connection between the worst-case and average-case hardness of certain lattice problems in the late 90's.

The LLL algorithm, despite the relatively poor quality of the solution it gives in the worst case, allowed to devise polynomial time solutions to many classical problems in computer science. These include, solving integer programs in a fixed number of variables, factoring polynomials over the rationals, breaking knapsack based cryptosystems, and finding solutions to many other Diophantine and cryptanalysis problems.

Ajtai's discovery suggested a completely different way to use lattices in cryptography. Instead of using algorithmic solutions to computationally tractable lattice approximation problems to *break* cryptosystems, Ajtai's work shows how to use the existence of computationally intractable-to-approximate lattice problems to *build* cryptosystems which are *impossible to break*. Namely, design cryptographic functions that are provably as hard to break as it is to solve a computationally hard lattice problem.

Whereas in complexity theory we say that a problem is hard if it is hard for the worst case instance, in cryptography a problem is deemed hard only if it is hard in the average case (i.e., for all but a negligible fraction of the instances). The novelty in Ajtai's result, is that he shows how to build a cryptographic function which is as hard to break on the average (e.g., over the random choices of the function instance) as it is to solve the worst case instance of a certain lattice problem. This achievement is unique to lattice theory at this time, and points to lattices as an ideal source of hardness for cryptographic purposes.

These new constructive applications of lattices, are deeply rooted in complexity theory, and were followed by a sharp increase in the study of lattices from a computational complexity point of view. This led to the resolution of several long standing open problems in the area. Most notably, the NP-hardness of the shortest vector problem in its exact and approximate versions. We present a self contained exposition of this latter result as well as other results on the computational complexity of lattice problems.

We did not attempt to cover everything known about lattices, as this would have filled several volumes. Rather, we selected a few representative topics, based on our personal taste and research experience. Regrettably, a topic which we neglect is duality and transference theorems. With this notable exception, we believe that most of the current ideas relevant to lattice based cryptography appear within in some form or another.

Many research questions regarding lattices and their cryptographic usage remain open. We hope that this book will help make lattice based cryptography more accessible to a wider audience, and ultimately yield further progress in this exciting research area.

Acknowledgments. Part of the material presented in this book is based on joint work of the authors with Shai Halevi, Oded Goldreich, Muli Safra and Jean-Pierre Seifert. Many other people have indirectly contributed to this book, either through their work, or through many conversations with the authors. Among them, we would like to mention Miklós Ajtai, Ravi Kannan, Amit Sahai, Claus Schnorr, Madhu Sudan and Salil Vadhan. We would like to thank all our coauthors and colleagues that have made this book possible.

The first author would like to thank also the National Science Foundation and Chris and Warren Hellman for partially supporting this work under NSF Career Award CCR-0093029 and a 2001-02 Hellman Fellowship.

DANIELE MICCIANCIO

Chapter 1

BASICS

This book is about algorithmic problems on point lattices, and their computational complexity. In this chapter we give some background about lattices and complexity theory.

1. Lattices

Let $\mathbb{R}^m$ be the m-dimensional Euclidean space. A *lattice* in $\mathbb{R}^m$ is the set

$$\mathcal{L}(\mathbf{b}_1,\ldots,\mathbf{b}_n) = \left\{\sum_{i=1}^{n} x_i\mathbf{b}_i : x_i \in \mathbb{Z}\right\} \tag{1.1}$$

of all integral combinations of n linearly independent vectors $\mathbf{b}_1,\ldots,\mathbf{b}_n$ in $\mathbb{R}^m$ $(m \geq n)$. The integers n and m are called the *rank* and *dimension* of the lattice, respectively. The sequence of vectors $\mathbf{b}_1,\ldots,\mathbf{b}_n$ is called a *lattice basis* and it is conveniently represented as a matrix

$$\mathbf{B} = [\mathbf{b}_1,\ldots,\mathbf{b}_n] \in \mathbb{R}^{m\times n} \tag{1.2}$$

having the basis vectors as columns. Using matrix notation, (1.1) can be rewritten in a more compact form as

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{Bx} : \mathbf{x} \in \mathbb{Z}^n\} \tag{1.3}$$

where $\mathbf{Bx}$ is the usual matrix-vector multiplication.

Graphically, a lattice can be described as the set of intersection points of an infinite, regular (but not necessarily orthogonal) n-dimensional grid. A 2-dimensional example is shown in Figure 1.1. There, the basis vectors are

$$\mathbf{b}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \qquad \mathbf{b}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{1.4}$$

D. Micciancio et al., *Complexity of Lattice Problems*

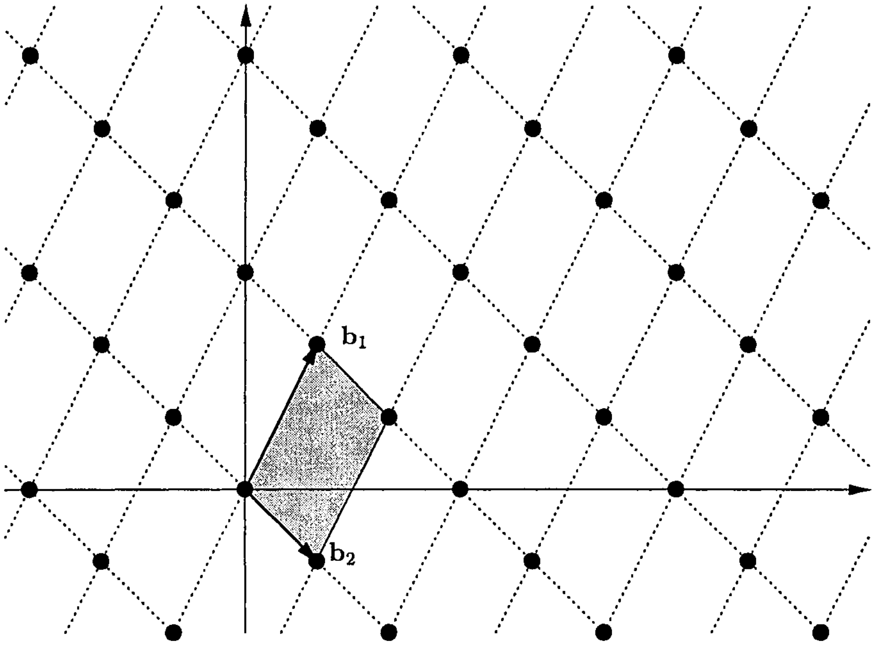


Figure 1.1. A lattice in $\mathbb{R}^2$

and they generate all the intersection points of the grid when combined with integer coefficients. The same lattice has many different bases. For example, vectors

$$\mathbf{b}'_1 = \mathbf{b}_1 + \mathbf{b}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \qquad \mathbf{b}'_2 = 2\mathbf{b}_1 + \mathbf{b}_2 = \begin{bmatrix} 3 \\ 3 \end{bmatrix} \qquad (1.5)$$

are also a basis for lattice $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2)$. The grid generated by $\mathbf{b}'_1, \mathbf{b}'_2$ is shown in Figure 1.2. Notice that although the two grids are different, the set of intersection points is exactly the same, i.e., $\{\mathbf{b}_1, \mathbf{b}_2\}$ and $\{\mathbf{b}'_1, \mathbf{b}'_2\}$ are two different bases for the same lattice $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2) = \mathcal{L}(\mathbf{b}'_1, \mathbf{b}'_2)$.

Throughout the book, we use the convention that lattice points are always represented as *column* vectors. Wherever vectors are more conveniently written as rows, we use transpose notation. For example, the definition of vector $\mathbf{b}_1, \mathbf{b}_2$ in (1.4) can equivalently be rewritten as $\mathbf{b}_1 = [1, 2]^T, \mathbf{b}_2 = [1, -1]^T$, where $\mathbf{A}^T$ denotes the transpose of matrix $\mathbf{A}$.

A simple example of n-dimensional lattice is given by the set $\mathbb{Z}^n$ of all vectors with integral coordinates. A possible basis is given by the

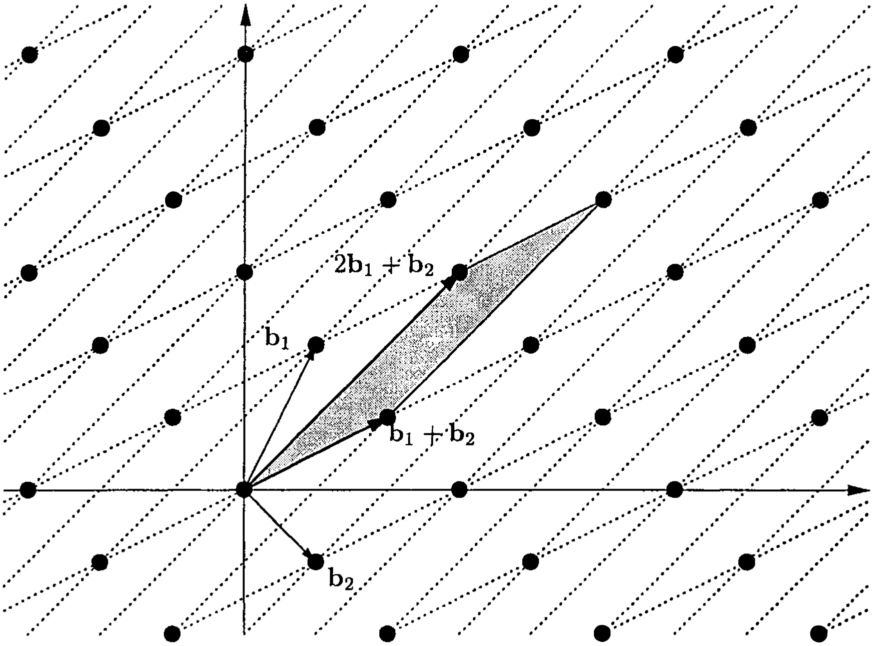


Figure 1.2. A different basis

standard unit vectors

$$\mathbf{e}_i = [\overbrace{\underbrace{0,\ldots,0,1}_{i},0,\ldots,0}^{n}]^T.$$

In matrix notation $\mathbb{Z}^n = \mathcal{L}(\mathbf{I})$ where $\mathbf{I} \in \mathbb{Z}^{n \times n}$ is the n-dimensional identity matrix, i.e., the $n \times n$ square matrix with 1's on the diagonal and 0's everywhere else.

When $n = m$, i.e., the number of basis vectors equals the number of coordinates, we say that $\mathcal{L}(\mathbf{B})$ is *full rank* or *full dimensional.* Equivalently, lattice $\mathcal{L}(\mathbf{B}) \subseteq \mathbb{R}^m$ is full rank if and only if the linear span of the basis vectors

$$\operatorname{span}(\mathbf{B}) = \{\mathbf{Bx}: \mathbf{x} \in \mathbb{R}^n\} \tag{1.6}$$

equals the entire space $\mathbb{R}^m$. The difference between (1.3) and (1.6) is that while in (1.6) one can use arbitrary real coefficients to combine the basis vectors, in (1.3) only integer coefficients are allowed. It is easy to see that span($\mathbf{B}$) does not depend on the particular basis $\mathbf{B}$, i.e., if $\mathbf{B}$ and $\mathbf{B}'$ generate the same lattice then span($\mathbf{B}$) = span($\mathbf{B}'$). So, for any lattice $\Lambda = \mathcal{L}(\mathbf{B})$, we can define the linear span of the lattice $\mathrm{span}(\Lambda)$, without reference to any specific basis. Notice that $\mathbf{B}$ is a basis of $\mathrm{span}(\mathbf{B})$ as a vector space. In particular, the rank of lattice $\mathcal{L}(\mathbf{B})$ equals the dimension of $\mathrm{span}(\mathbf{B})$ as a vector space over $\mathbb{R}$ and it is a lattice invariant, i.e., it does not depend on the choice of the basis.

Clearly, any set of n linearly independent lattice vectors $\mathbf{B}' \in \mathcal{L}(\mathbf{B})$ is a basis for $\mathrm{span}(\mathbf{B})$ as a vector space. However, $\mathbf{B}'$ is not necessarily a lattice basis for $\mathcal{L}(\mathbf{B})$. See Figure 1.3 for a 2-dimensional example. The picture shows the lattice $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2)$ generated by basis vectors (1.4) and the grid associated to lattice vectors

$$\mathbf{b}'_1 = \mathbf{b}_1 + \mathbf{b}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \qquad \mathbf{b}'_2 = \mathbf{b}_1 - \mathbf{b}_2 = \begin{bmatrix} 0 \\ 3 \end{bmatrix}. \qquad (1.7)$$

Vectors $\mathbf{b}'_1$ and $\mathbf{b}'_2$ are linearly independent. Therefore, they are a basis for the plane $\mathbb{R}^2 = \mathrm{span}(\mathbf{b}_1, \mathbf{b}_2)$ as a vector space. However, they are not a basis for $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2)$ because lattice point $\mathbf{b}_1$ cannot be expressed as an *integer* linear combination of $\mathbf{b}'_1$ and $\mathbf{b}'_2$. There is a simple geometric characterization for linearly independent lattice vectors that generate the whole lattice. For any n linearly independent lattice vectors $\mathbf{B}' = [\mathbf{b}'_1, \ldots, \mathbf{b}'_n]$ (with $\mathbf{b}'_i \in \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$ for all $i = 1, \ldots, n$) define the half open parallelepiped

$$\mathcal{P}(\mathbf{B}') = \{\mathbf{B}'\mathbf{x} : 0 \le x_i < 1\}. \qquad (1.8)$$

Then, $\mathbf{B}'$ is a basis for lattice $\mathcal{L}(\mathbf{B})$ if and only if $\mathcal{P}(\mathbf{B}')$ does not contain any lattice vector other than the origin. Figures 1.1, 1.2 and 1.3 illustrate the two cases. The lattice in Figures 1.2 and 1.3 is the same as the one in Figure 1.1. In Figure 1.2, the (half open) parallelepiped $\mathcal{P}(\mathbf{B}')$ does not contain any lattice point other than the origin, and therefore $\mathcal{L}(\mathbf{B}') = \mathcal{L}(\mathbf{B})$. In Figure 1.3, parallelepiped $\mathcal{P}(\mathbf{B}')$ contains lattice point $\mathbf{b}_1$. Therefore $\mathcal{L}(\mathbf{B}') \neq \mathcal{L}(\mathbf{B})$ and $\mathbf{B}'$ is not a basis for $\mathcal{L}(\mathbf{B})$.

Notice that since $\mathbf{B}'$ is a set of linearly independent vectors, $\mathcal{L}(\mathbf{B}')$ is a lattice and $\mathbf{B}'$ is a basis for $\mathcal{L}(\mathbf{B}')$. Clearly, $\mathcal{L}(\mathbf{B}') \subseteq \mathcal{L}(\mathbf{B})$, i.e., any point from lattice $\mathcal{L}(\mathbf{B}')$ belongs also to lattice $\mathcal{L}(\mathbf{B})$. When $\mathcal{L}(\mathbf{B}') \subseteq \mathcal{L}(\mathbf{B})$, we say that $\mathcal{L}(\mathbf{B}')$ is a *sublattice* of $\mathcal{L}(\mathbf{B})$. If $\mathcal{L}(\mathbf{B}') = \mathcal{L}(\mathbf{B})$ we say that bases $\mathbf{B}$ and $\mathbf{B}'$ are *equivalent.* If $\mathcal{L}(\mathbf{B}') \subseteq \mathcal{L}(\mathbf{B})$, but $\mathcal{L}(\mathbf{B}') \neq \mathcal{L}(\mathbf{B})$, then bases $\mathbf{B}$ and $\mathbf{B}'$ are not equivalent, and $\mathcal{L}(\mathbf{B}')$ is a *proper* sublattice of $\mathcal{L}(\mathbf{B})$.

Equivalent bases (i.e., bases that generate the same lattice) can be algebraically characterized as follows. Two bases $\mathbf{B}, \mathbf{B}' \in \mathbb{R}^{m \times n}$ are equivalent if and only if there exists a unimodular matrix $\mathbf{U} \in \mathbb{Z}^{n \times n}$ (i.e., an integral matrix with determinant $\det(\mathbf{U}) = \pm 1$) such that $\mathbf{B}' = \mathbf{B}\mathbf{U}$. The simple proof is left to the reader as an exercise.

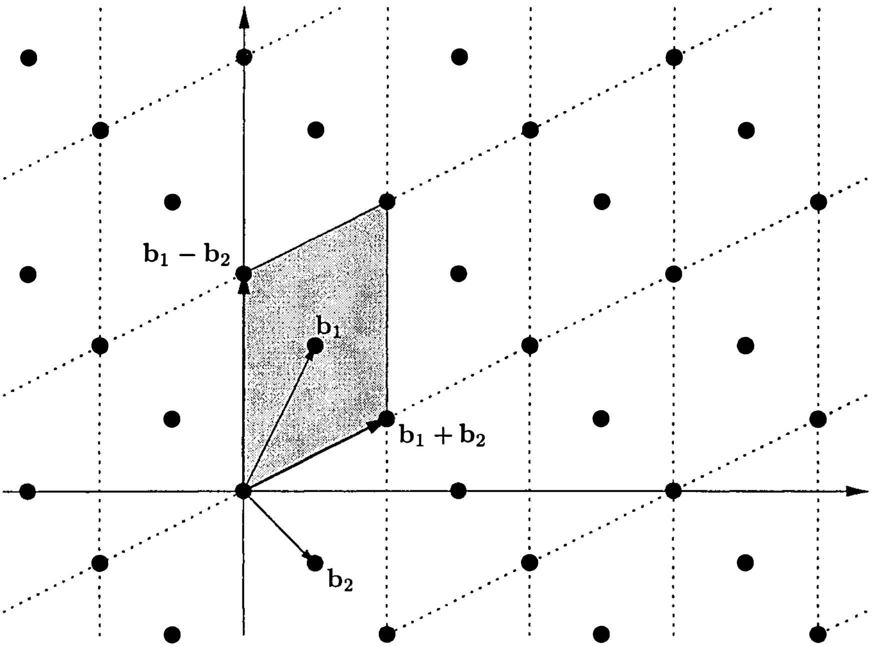


Figure 1.3. The sublattice generated by $\mathbf{b}_1 + \mathbf{b}_2$ and $\mathbf{b}_1 - \mathbf{b}_2$

When studying lattices from a computational point of view, it is customary to assume that the basis vectors (and therefore any lattice vector) have all rational coordinates. It is easy to see that rational lattices can be converted to integer lattices (i.e., sublattices of $\mathbb{Z}^n$) by multiplying all coordinates by an appropriate integer scaling factor. So, without loss of generality, in the rest of this book we concentrate on integer lattices, and, unless explicitly stated otherwise, we always assume that lattices are represented by a basis, i.e., a matrix with integer coordinates such that the columns are linearly independent.

Lattices can also be characterized without reference to any basis. A lattice can be defined as a discrete nonempty subset Λ of $\mathbb{R}^m$ which is closed under subtraction, i.e., if $\mathbf{x} \in \Lambda$ and $\mathbf{y} \in \Lambda$, then also $\mathbf{x} - \mathbf{y} \in \Lambda$. Here "discrete" means that there exists a positive real $\lambda > 0$ such that the distance between any two lattice vectors is at least λ. A typical example is the set $\Lambda = \{\mathbf{x} \in \mathbb{Z}^n \colon \mathbf{A}\mathbf{x} = \mathbf{0}\}$ of integer solutions of a system of homogeneous linear equations. Notice that Λ always contains the origin $\mathbf{0} = \mathbf{x} - \mathbf{x}$, it is closed under negation (i.e., if $\mathbf{x} \in \Lambda$ then $-\mathbf{x} = \mathbf{0} - \mathbf{x} \in \Lambda$), and addition (i.e., if $\mathbf{x}, \mathbf{y} \in \Lambda$ then $\mathbf{x} + \mathbf{y} = \mathbf{x} - (-\mathbf{y}) \in \Lambda$). In other words, Λ is a discrete additive subgroup of $\mathbb{R}^m$.

1.1 Determinant

The *determinant* of a lattice $\Lambda = \mathcal{L}(\mathbf{B})$, denoted $\det(\Lambda)$, is the n-dimensional volume of the fundamental parallelepiped $\mathcal{P}(\mathbf{B})$ spanned by the basis vectors. (See shaded areas in Figures 1.1 and 1.2.) The determinant is a lattice invariant, i.e., it does not depend on the particular basis used to compute it. This immediately follows from the characterization of equivalent bases as matrices $\mathbf{B}' = \mathbf{BU}$ related by a unimodular transformation $\mathbf{U}$. Geometrically, this corresponds to the intuition that the (n-dimensional) volume of the fundamental parallelepiped $\mathcal{P}(\mathbf{B})$ equals the inverse of the density of the lattice points in $\operatorname{span}(\mathbf{B})$. As an example consider the bases in Figures 1.1 and 1.2. The areas of the fundamental regions (i.e., the shaded parallelepipeds in the pictures) are exactly the same because the two bases generate the same lattice. However, the shaded parallelepiped in Figure 1.3 has a different area (namely, twice as much as the original lattice) because vectors (1.7) only generate a sublattice.

A possible way to compute the determinant is given by the usual *Gram-Schmidt orthogonalization* process. For any sequence of vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$, define the corresponding Gram-Schmidt orthogonalized vectors $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$ by

$$\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^* \tag{1.9a}$$

$$\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \tag{1.9b}$$

where $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{m} x_i y_i$ is the inner product in $\mathbb{R}^m$. For every i, $\mathbf{b}_i^*$ is the component of $\mathbf{b}_i$ orthogonal to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$. In particular, $\operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_i) = \operatorname{span}(\mathbf{b}_1^*, \ldots, \mathbf{b}_i^*)$ and vectors $\mathbf{b}_i^*$ are pairwise orthogonal, i.e., $\langle \mathbf{b}_i^*, \mathbf{b}_j^* \rangle = 0$ for all $i \neq j$. The determinant of the lattice equals the product of the lengths of the orthogonalized vectors

$$\det(\mathcal{L}(\mathbf{B})) = \prod_{i=1}^{n} \|\mathbf{b}_i^*\| \tag{1.10}$$

where $\|\mathbf{x}\| = \sqrt{\sum_i x_i^2}$ is the usual Euclidean length. We remark that the definition of the orthogonalized vectors $\mathbf{b}_i^*$ depends on the order of the original basis vectors. Given basis matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$, we denote by $\mathbf{B}^*$ the matrix whose columns are the orthogonalized vectors $[\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*]$. Clearly, $\mathbf{B}^*$ is a basis of $\operatorname{span}(\mathbf{B})$ as a vector space. However, $\mathbf{B}^*$ is not usually a lattice basis for $\mathcal{L}(\mathbf{B})$. In particular, not every lattice has a basis consisting of mutually orthogonal vectors.

Notice that if the $\mathbf{b}_i$'s are rational vectors (i.e., vectors with rational coordinates), then also the orthogonalized vectors $\mathbf{b}_i^*$ are rationals. If lattice $\mathcal{L}(\mathbf{B})$ is full dimensional (i.e. $m = n$), then $\mathbf{B}$ is a nonsingular square matrix and $\det(\mathcal{L}(\mathbf{B}))$ equals the absolute value of the determinant of the basis matrix $\det(\mathbf{B})$. For integer lattices, $\mathbf{B}$ is a square integer matrix, and the lattice determinant $\det(\mathcal{L}(\mathbf{B})) = \det(\mathbf{B})$ is an integer. In general, the reader can easily verify that $\det(\mathcal{L}(\mathbf{B}))$ equals the square root of the determinant of the Gram matrix $\mathbf{B}^T\mathbf{B}$, i.e., the $n \times n$ matrix whose (i,j)th entry is the inner product $\langle \mathbf{b}_i, \mathbf{b}_j \rangle$:

$$\det(\mathcal{L}(\mathbf{B})) = \sqrt{\det(\mathbf{B}^T\mathbf{B})}. \qquad (1.11)$$

This gives an alternative way to compute the determinant of a lattice (other than computing the Gram-Schmidt orthogonalized vectors), and shows that if $\mathbf{B}$ is an integer matrix, then the determinant of $\mathcal{L}(\mathbf{B})$ is always the square root of a positive integer, even if $\det(\mathcal{L}(\mathbf{B}))$ is not necessarily an integer when the lattice is not full rank.

1.2 Successive minima

Let $\mathcal{B}_m(\mathbf{0}, r) = \{\mathbf{x} \in \mathbb{R}^m : \|\mathbf{x}\| < r\}$ be the m-dimensional open ball of radius r centered in $\mathbf{0}$. When the dimension m is clear from the context, we omit the subscript m and simply write $\mathcal{B}(\mathbf{0}, r)$. Fundamental constants associated to any rank n lattice Λ are its successive minima $\lambda_1, \ldots, \lambda_n$. The ith minimum $\lambda_i(\Lambda)$ is the radius of the smallest sphere centered in the origin containing i linearly independent lattice vectors

$$\lambda_i(\Lambda) = \inf\{r \colon \dim(\operatorname{span}(\Lambda \cap \mathcal{B}(\mathbf{0}, r))) \geq i\}. \qquad (1.12)$$

Successive minima can be defined with respect to any norm. A norm is a positive definite, homogeneous function that satisfies the triangle inequality, i.e., a function $\|\cdot\| \colon \mathbb{R}^n \to \mathbb{R}$ such that

- $\|\mathbf{x}\| \geq 0$ with equality only if $\mathbf{x} = \mathbf{0}$
- $\|\alpha\mathbf{x}\| = |\alpha| \cdot \|\mathbf{x}\|$
- $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$. An important family of norm functions is given by the ℓ_p norms. For any $p \geq 1$, the ℓ_p norm of a vector $\mathbf{x} \in \mathbb{R}^n$ is

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} x_i^p\right)^{1/p}. \qquad (1.13a)$$

Important special cases are the l_1-norm

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|, \tag{1.13b}$$

the ℓ_2 norm (or Euclidean norm)

$$\|\mathbf{x}\|_2 = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}, \tag{1.13c}$$

and the ℓ_∞ norm (or max-norm)

$$\|\mathbf{x}\|_\infty = \lim_{p \to \infty} \|\mathbf{x}\|_p = \max_{i=1}^{n} |x_i|. \tag{1.13d}$$

We remark that when $p < 1$, function (1.13) is not a norm because it does not satisfy the triangle inequality. Notice that the value of the successive minima $\lambda_1, \ldots, \lambda_n$, and the lattice vectors achieving them, depend on the norm being used. Consider for example the lattice

$$\Lambda = \{\mathbf{v} \in \mathbb{Z}^2 : v_1 + v_2 = 0 \bmod 2\} \tag{1.14}$$

generated by basis vectors

$$\mathbf{b}_1 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \qquad \mathbf{b}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{1.15}$$

Lattice vector $\mathbf{b}_1$ is a shortest (nonzero) vector in $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2)$ with respect the ℓ_1 norm and $\lambda_1 = \|\mathbf{b}_1\|_1 = 2$ if the ℓ_1 norm is used. However, $\mathbf{b}_1$ is not a shortest vector with respect to the ℓ_2 or ℓ_∞ because in these norms lattice vector $\mathbf{b}_2$ is strictly shorter than $\mathbf{b}_1$ giving first minimum $\lambda_1 = \|\mathbf{b}_2\|_2 = \sqrt{2}$ and $\lambda_1 = \|\mathbf{b}_2\|_\infty = 1$, respectively. In this book we are primarily concerned with the ℓ_2 norm, which corresponds to the familiar Euclidean distance

$$\mathrm{dist}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{1.16}$$

and will consider other norms only when it can be done without substantially complicating the exposition.

In the previous examples, we have seen that lattice (1.14) contains a vector $\mathbf{b}$ such that $\|\mathbf{b}\| = \lambda_1$. It turns out that this is true for every lattice. It easily follows from the characterization of lattices as discrete subgroups of $\mathbb{R}^n$ that there always exist vectors achieving the successive minima, i.e., there are linearly independent vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \Lambda$ such that $\|\mathbf{x}_i\| = \lambda_i$ for all $i = 1, \ldots, n$. So, the infimum in (1.12) is actually a minimum if $\mathcal{B}(\mathbf{0}, r)$ is replaced with the closed ball $\bar{\mathcal{B}}(\mathbf{0}, r) = \{\mathbf{x} \in \mathbb{R}^m : \|\mathbf{x}\| \leq r\}$. In particular, $\lambda_1(\Lambda)$ is the length of the shortest nonzero lattice vector and equals the minimum distance between any two distinct lattice points

$$\lambda_1(\Lambda) = \min_{\mathbf{x} \neq \mathbf{y} \in \Lambda} \|\mathbf{x} - \mathbf{y}\| = \min_{\mathbf{x} \in \Lambda \setminus \{\mathbf{0}\}} \|\mathbf{x}\|. \tag{1.17}$$

In the rest of this section we give a proof that any lattice contains nonzero vectors of minimal length. In doing so, we prove a lower bound for the first minimum that will be useful later on. The result is easily generalized to all successive minima to show that there are n linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ satisfying $\|\mathbf{v}_i\| = \lambda_i$ for all $i = 1, \ldots, n$. Fix some lattice $\mathcal{L}(\mathbf{B})$, and consider the first minimum

$$\lambda_1 = \inf\{\|\mathbf{v}\| : \mathbf{v} \in \mathcal{L}(\mathbf{B})/\{\mathbf{0}\}\}.$$

We want to prove that there exists a lattice vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\|\mathbf{v}\| = \lambda_1$. We first prove that λ_1 is strictly positive.

THEOREM 1.1 *Let* $\mathbf{B}$ *be a lattice basis, and let* $\mathbf{B}^*$ *be the corresponding Gram-Schmidt orthogonalization. Then, the first minimum of the lattice (in the* ℓ_2 *norm) satisfies*

$$\lambda_1 \geq \min_j \|\mathbf{b}_j^*\| > 0.$$

Proof: Consider a generic nonzero lattice vector $\mathbf{Bx}$ (where $\mathbf{x} \in \mathbb{Z}^n$ and $\mathbf{x} \neq \mathbf{0}$) and let i be the biggest index such that $x_i \neq 0$. We show that $\|\mathbf{Bx}\| \geq \|\mathbf{b}_i^*\| \geq \min_j \|\mathbf{b}_j^*\|$. It follows that the infimum $\lambda_1 = \inf \|\mathbf{Bx}\|$ also satisfies $\lambda_1 \geq \min_j \|\mathbf{b}_j^*\|$. From basic linear algebra we know that $|\langle \mathbf{x}, \mathbf{y}\rangle| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\|$ for any two vectors $\mathbf{x}, \mathbf{y}$. We prove that $|\langle \mathbf{Bx}, \mathbf{b}_i^*\rangle| \geq \|\mathbf{b}_i^*\|^2$, and therefore $\|\mathbf{Bx}\| \cdot \|\mathbf{b}_i^*\| \geq \|\mathbf{b}_i^*\|^2$. Since vectors $\mathbf{b}_i$'s are linearly independent, $\|\mathbf{b}_i^*\| \neq 0$ and $\|\mathbf{Bx}\| \geq \|\mathbf{b}_i^*\|$ follows.

So, let us prove that $|\langle \mathbf{Bx}, \mathbf{b}_i^*\rangle| \geq \|\mathbf{b}_i^*\|^2$. From the definition of i, we know that $\mathbf{Bx} = \sum_{j=1}^{i} \mathbf{b}_j x_j$. Using the definition of the orthogonalized vectors (1.9a) we get

$$\begin{aligned} \langle \mathbf{Bx}, \mathbf{b}_i^*\rangle &= \sum_{j=1}^{i} \langle \mathbf{b}_j, \mathbf{b}_i^*\rangle x_j \\ &= \langle \mathbf{b}_i, \mathbf{b}_i^*\rangle x_i \end{aligned}$$

$$\begin{aligned} &= \langle \mathbf{b}_i^* + \sum_{j<i} \mu_{ij}\mathbf{b}_j^*, \mathbf{b}_i^* \rangle x_i \\ &= \langle \mathbf{b}_i^*, \mathbf{b}_i^* \rangle x_i + \sum_{j<i} \mu_{ij} \langle \mathbf{b}_j^*, \mathbf{b}_i^* \rangle x_i \\ &= \|\mathbf{b}_i^*\|^2 x_i. \end{aligned}$$

Since x_i is a nonzero integer,

$$|\langle \mathbf{Bx}, \mathbf{b}_i^* \rangle| = \|\mathbf{b}_i^*\|^2 \cdot |x_i| \geq \|\mathbf{b}_i^*\|^2. \qquad \square$$

In particular, the theorem shows that $\lambda_1 > 0$. We now prove that there exists a nonzero lattice vector of length λ_1. By definition of λ_1, there exists a sequence of lattice vectors $\mathbf{v}_i \in \mathcal{L}(\mathbf{B})$ such that

$$\lim_{i \to \infty} \|\mathbf{v}_i\| = \lambda_1.$$

Since $\lambda_1 > 0$, for all sufficiently large i it must be $\|\mathbf{v}_i\| \leq 2\lambda_1$, i.e., lattice vector $\mathbf{v}_i$ belongs to the closed ball

$$\bar{\mathcal{B}}(\mathbf{0}, 2\lambda_1) = \{\mathbf{z} : \|\mathbf{z}\| \leq 2\lambda_1\}.$$

But set $\bar{\mathcal{B}}(0, 2\lambda)$ is compact, so, we can extract a convergent subsequence $\mathbf{v}_{i_j}$ with limit

$$\mathbf{w} = \lim_{j \to \infty} \mathbf{v}_{i_j}.$$

Clearly, $\|\mathbf{w}\| = \lim_{j\to\infty} \|\mathbf{v}_{i_j}\| = \lambda_1$. We want to prove that $\mathbf{w}$ is a lattice vector. By definition of $\mathbf{w}$ we have $\lim_{j\to\infty} \|\mathbf{v}_{i_j} - \mathbf{w}\| = 0$. Therefore for all sufficiently large j, $\|\mathbf{v}_{i_j} - \mathbf{w}\| < \lambda_1/2$. By triangle inequality, for a sufficiently large j and all $k > j$,

$$\|\mathbf{v}_{i_j} - \mathbf{v}_{i_k}\| \leq \|\mathbf{v}_{i_j} - \mathbf{w}\| + \|\mathbf{w} - \mathbf{v}_{i_k}\| < \lambda_1.$$

But $\mathbf{v}_{i_j} - \mathbf{v}_{i_k}$ is a lattice vector, and no nonzero lattice vector can have length strictly less than λ_1. This proves that $\mathbf{v}_{i_j} - \mathbf{v}_{i_k} = \mathbf{0}$, i.e., $\mathbf{v}_{i_k} = \mathbf{v}_{i_j}$ for all $k > j$. Therefore, $\mathbf{w} = \lim_k \mathbf{v}_{i_k} = \mathbf{v}_{i_j}$, and $\mathbf{w}$ is a lattice vector.

The above argument can be easily generalized to prove the following theorem about all successive minima of a lattice.

THEOREM 1.2 *Let Λ be a lattice of rank n with successive minima λ_1, ..., λ_n. Then there exist linearly independent lattice vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \Lambda$ such that $\|\mathbf{v}_i\| = \lambda_i$ for all $i = 1, \ldots, n$.*

Interestingly, the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ achieving the minima are not necessarily a basis for Λ. Examples of lattices for which all bases must contain at least one vector strictly longer than λ_n are given in Chapter 7.

1.3 Minkowski's theorems

In this subsection we prove an important upper bound on the product of successive minima of any lattice. The bound is based on the following fundamental theorem.

THEOREM 1.3 (BLICHFELDT THEOREM.) *For any lattice* Λ *and for any measurable set* $S \subseteq$ *span*(Λ), *if* S *has volume* $\mathrm{vol}(S) > \det(\Lambda)$, *then there exist two distinct points* $\mathbf{z}_1, \mathbf{z}_2 \in S$ *such that* $\mathbf{z}_1 - \mathbf{z}_2 \in \Lambda$.

Proof: Let $\Lambda = \mathcal{L}(\mathbf{B})$ be a lattice and S be any subset of span(Λ) such that $\mathrm{vol}(S) > \det(\mathbf{B})$. Partition S into a collection of disjoint regions as follows. For any lattice point $\mathbf{x} \in \Lambda$ define

$$S_{\mathbf{x}} = S \cap (\mathcal{P}(\mathbf{B}) + \mathbf{x}) \tag{1.18}$$

where $\mathcal{P}(\mathbf{B})$ is the half open parallelepiped (1.8). Here and below, for any set $A \subset \mathbb{R}^n$ and vector $\mathbf{x} \in \mathbb{R}^n$, expression $A + \mathbf{x}$ denotes the set $\{\mathbf{y} + \mathbf{x} : \mathbf{y} \in A\}$. Notice that sets $\mathcal{P}(\mathbf{B}) + \mathbf{x}$ (with $\mathbf{x} \in \Lambda$) partition span$(\mathbf{B})$. Therefore sets $S_{\mathbf{x}}$ ($\mathbf{x} \in \Lambda$) form a partition of S, i.e., they are pairwise disjoint and

$$S = \bigcup_{\mathbf{x} \in \Lambda} S_{\mathbf{x}}.$$

In particular, since Λ is countable,

$$\mathrm{vol}(S) = \sum_{\mathbf{x} \in \Lambda} \mathrm{vol}(S_{\mathbf{x}}).$$

Define also translated sets

$$S'_{\mathbf{x}} = S_{\mathbf{x}} - \mathbf{x} = (S - \mathbf{x}) \cap \mathcal{P}(\mathbf{B})$$

Notice that for all $\mathbf{x} \in \Lambda$, set $S'_{\mathbf{x}}$ is contained in $\mathcal{P}(\mathbf{B})$ and $\mathrm{vol}(S_{\mathbf{x}}) = \mathrm{vol}(S'_{\mathbf{x}})$. We claim that sets $S'_{\mathbf{x}}$ are not pairwise disjoint. Assume, for contradiction, they are. Then, we have

$$\sum_{\mathbf{x} \in \Lambda} \mathrm{vol}(S'_{\mathbf{x}}) = \mathrm{vol}\left(\bigcup_{\mathbf{x} \in \Lambda} S'_{\mathbf{x}}\right) \leq \mathrm{vol}(\mathcal{P}(\mathbf{B})). \tag{1.19}$$

We also know from the assumption in the theorem that

$$\sum_{\mathbf{x} \in \Lambda} \mathrm{vol}(S'_{\mathbf{x}}) = \sum_{\mathbf{x} \in \Lambda} \mathrm{vol}(S_{\mathbf{x}}) = \mathrm{vol}(S) > \det(\Lambda). \tag{1.20}$$

Combining (1.19) and (1.20) we get $\det(\Lambda) < \mathrm{vol}(\mathcal{P}(\mathbf{B}))$, which is a contradiction because $\det(\Lambda) = \mathrm{vol}(\mathcal{P}(\mathbf{B}))$ by the definition of lattice determinant.

This proves that set $S'_{\mathbf{y}}$ are not pairwise disjoint, i.e., there exist two sets $S'_{\mathbf{x}}$, $S'_{\mathbf{y}}$ (for $\mathbf{x},\mathbf{y} \in \Lambda$) such that $S'_{\mathbf{x}} \cap S'_{\mathbf{y}} \neq \emptyset$. Let $\mathbf{z}$ be any vector in the (nonempty) intersection $S'_{\mathbf{x}} \cap S'_{\mathbf{y}}$ and define

$$\begin{aligned} \mathbf{z}_1 &= \mathbf{z} + \mathbf{x} \\ \mathbf{z}_2 &= \mathbf{z} + \mathbf{y}. \end{aligned}$$

From $\mathbf{z} \in S'_{\mathbf{x}}$ and $\mathbf{z} \in S'_{\mathbf{y}}$ we get $\mathbf{z}_1 \in S_{\mathbf{x}} \subseteq S$ and $\mathbf{z}_2 \in S_{\mathbf{y}} \subseteq S$. Moreover, $\mathbf{z}_1 \neq \mathbf{z}_2$ because $\mathbf{x} \neq \mathbf{y}$. Finally, the difference between $\mathbf{z}_1$ and $\mathbf{z}_2$ satisfies

$$\mathbf{z}_1 - \mathbf{z}_2 = \mathbf{x} - \mathbf{y} \in \Lambda, \tag{1.21}$$

completing the proof of the theorem. □

As a corollary to Blichfeldt theorem we immediately get the following theorem of Minkowski.

THEOREM 1.4 (CONVEX BODY THEOREM) *For any lattice Λ of rank n and any convex set $S \subset span(\Lambda)$ symmetric about the origin, if* $\mathrm{vol}(S) > 2^n \det(\Lambda)$, *then S contains a nonzero lattice point* $\mathbf{v} \in S \cap \Lambda \setminus \{\mathbf{0}\}$.

Proof: Consider the set $S' = \{\mathbf{x}: 2\mathbf{x} \in S\}$. The volume of S' satisfies

$$\mathrm{vol}(S') = 2^{-n}\,\mathrm{vol}(S) > \det(\Lambda). \tag{1.22}$$

Therefore, by Blichfeldt theorem there exist two distinct points $\mathbf{z}_1, \mathbf{z}_2 \in S'$ such that $\mathbf{z}_1 - \mathbf{z}_2 \in \mathcal{L}(\Lambda)$. From the definition of S', we get $2\mathbf{z}_1, 2\mathbf{z}_2 \in S$ and since S is symmetric about the origin, we also have $-2\mathbf{z}_2 \in S$. Finally, by convexity, the midpoint of segment $[2\mathbf{z}_1, -2\mathbf{z}_2]$ also belongs to S, i.e.,

$$\frac{2\mathbf{z}_1 + (-2\mathbf{z}_2)}{2} = \mathbf{z}_1 - \mathbf{z}_2 \in S. \tag{1.23}$$

This proves that $\mathbf{v} = \mathbf{z}_1 - \mathbf{z}_2$ is a nonzero lattice point in S. □

Minkowski's convex body theorem can be used to bound the length of the shortest nonzero vector in an rank n lattice as follows. Let $S = \mathcal{B}(\mathbf{0}, \sqrt{n}\det(\Lambda)^{1/n}) \cap \mathrm{span}(\Lambda)$ be the open ball of radius $\sqrt{n}\det(\Lambda)^{1/n}$ in $\mathrm{span}(\Lambda)$. Notice that S has volume strictly bigger than $2^n \det(\Lambda)$ because it contains an n-dimensional hypercube with edges of length $2\det(\Lambda)^{1/n}$. By Minkowski's theorem there exists a nonzero lattice vector $\mathbf{v} \in \mathcal{L}(\mathbf{B}) \setminus \{\mathbf{0}\}$ such that $\mathbf{v} \in S$, i.e., $\|\mathbf{v}\| < \sqrt{n}\det(\Lambda)^{1/n}$. This proves that for any rank n lattice Λ, the length of the shortest nonzero vector (in the ℓ_2 norm) satisfies

$$\lambda_1 < \sqrt{n}\det(\Lambda)^{1/n}. \tag{1.24}$$

This result (in a slightly stronger form) is the well known *Minkowski's first theorem.* Minkowski also proved a stronger result involving all successive minima, known as the *second theorem* of Minkowski. Namely, $\sqrt{n}\det(\Lambda)^{1/n}$ is an upper bound not only to the first minimum λ_1, but also to the the geometric mean of all successive minima. While Minkowski's first theorem is easily generalized to any norm, the proof of the second theorem for general norms is relatively complex. Here we prove the theorem only for the simple case of the Euclidean norm.

THEOREM 1.5 (MINKOWSKI'S SECOND THEOREM) *For any rank n lattice $\mathcal{L}(\mathbf{B})$, the successive minima (in the ℓ_2 norm) $\lambda_1,\ldots,\lambda_n$ satisfy*

$$\left(\prod_{i=1}^{n}\lambda_i\right)^{1/n} < \sqrt{n}\det(\mathbf{B})^{1/n}. \qquad (1.25)$$

Proof: Let $\mathbf{x}_1,\ldots,\mathbf{x}_n$ be linearly independent lattice vectors achieving the successive minima $\|\mathbf{x}_i\| = \lambda_i$ and assume for contradiction that $\prod_{i=1}^{n}\lambda_i \geq (\sqrt{n})^n\det(\mathbf{B})$. Consider the Gram-Schmidt orthogonalized vectors $\mathbf{x}_i^*$ and define the transformation

$$T\left(\sum c_i\mathbf{x}_i^*\right) = \sum \lambda_i c_i \mathbf{x}_i^* \qquad (1.26)$$

that expands each coordinate $\mathbf{x}_i^*$ by a factor λ_i. Let $S = \mathcal{B}(\mathbf{0},1)\cap \operatorname{span}(\mathbf{B})$ be the n-dimensional open unit ball in $\operatorname{span}(\mathbf{B})$. If we apply T to S we get a symmetric convex body $T(S)$ of volume

$$\begin{aligned}\operatorname{vol}(T(S)) &= \left(\prod_i \lambda_i\right)\operatorname{vol}(S)\\ &\geq (\sqrt{n})^n\det(\mathbf{B})\operatorname{vol}(S)\\ &= \operatorname{vol}(\sqrt{n}S)\det(\mathbf{B})\end{aligned}$$

where $\sqrt{n}S$ is the ball of radius $\sqrt{n}$. The volume of $\sqrt{n}S$ is bigger than 2^n because $\sqrt{n}S$ contains a hypercube with edges of length 2. Therefore, $\operatorname{vol}(T(S)) > 2^n\det(\mathbf{B})$, and by Minkowski's convex body theorem $T(S)$ contains a lattice point $\mathbf{y}$ different from the origin. Since $\mathbf{y}\in T(S)$, it must be $\mathbf{y} = T(\mathbf{x})$ for some $\mathbf{x}\in S$. From the definition of S we get $\|\mathbf{x}\| < 1$. Now express $\mathbf{x}$ and $\mathbf{y}$ in terms of the orthogonalized basis

$$\begin{aligned}\mathbf{x} &= \sum_{i=1}^{n} c_i\mathbf{x}_i^*\\ \mathbf{y} &= \sum \lambda_i c_i\mathbf{x}_i^*.\end{aligned}$$

Since $\mathbf{y}$ is nonzero, some c_i is not zero. Let k be the largest index such that $c_i \neq 0$, and $k' \leq k$ the smallest index such that $\lambda_{k'} = \lambda_k$. Notice that $\mathbf{y}$ is linearly independent from $\mathbf{x}_1, \ldots, \mathbf{x}_{k'-1}$ because $\langle \mathbf{x}_k^*, \mathbf{y} \rangle = \lambda_k c_k \|\mathbf{x}_k^*\|^2 \neq 0$ and $\mathbf{x}_k^*$ is orthogonal to $\mathbf{x}_1, \ldots, \mathbf{x}_{k'-1}$. We now show that $\|\mathbf{y}\| < \lambda_k$.

$$\begin{aligned}
\|\mathbf{y}\|^2 &= \left\| \sum_{i \leq k} \lambda_i c_i \mathbf{x}_i^* \right\|^2 \\
&= \sum_{i \leq k} \lambda_i^2 c_i^2 \|\mathbf{x}_i^*\|^2 \\
&\leq \sum_{i \leq k} \lambda_k^2 c_i^2 \|\mathbf{x}_i^*\|^2 \\
&= \lambda_k^2 \left\| \sum_{i \leq k} c_i \mathbf{x}_i^* \right\|^2 \\
&= \lambda_k^2 \|\mathbf{x}\|^2 < \lambda_k^2.
\end{aligned}$$

This proves that $\mathbf{x}_1, \ldots, \mathbf{x}_{k'-1}, \mathbf{y}$ are k' linearly independent lattice vectors of length strictly less than $\lambda_k = \lambda_{k'}$, contradicting the definition of the k'th successive minimum $\lambda_{k'}$. □

2. Computational problems

Minkowski's first theorem gives a simple way to bound the length λ_1 of the shortest nonzero vector in a lattice $\mathcal{L}(\mathbf{B})$. Although Minkowski's bound is asymptotically tight in the worst case (i.e., there exist lattices such that $\lambda_1 > c\sqrt{n}\det(\mathbf{B})^{1/n}$ for some absolute constant c independent of n), in general λ_1 can be much smaller than $\sqrt{n}\det(\mathbf{B})^{1/n}$. For example, consider the two dimensional lattice generated by orthogonal vectors $\mathbf{b}_1 = \epsilon \mathbf{e}_1$ and $\mathbf{b}_1 = (1/\epsilon)\mathbf{e}_2$. The determinant of the lattice is 1, giving upper bound $\lambda_1 \leq \sqrt{2}$. However $\lambda_1 = \epsilon$ can be arbitrarily small.

Moreover, the proof of Minkowski's theorem is not constructive, in the sense that we know from the theorem that a short nonzero vector exists, but the proof does not give any computational method to efficiently find vectors of length bounded by $\sqrt{n}\det(\Lambda)^{1/n}$, leave alone vectors of length λ_1. The problem of finding a lattice vector of length λ_1 is the well known *Shortest Vector Problem.*

DEFINITION 1.1 (SHORTEST VECTOR PROBLEM, SVP) *Given a basis* $\mathbf{B} \in \mathbb{Z}^{m \times n}$, *find a nonzero lattice vector* $\mathbf{Bx}$ *(with* $\mathbf{x} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$*) such that* $\|\mathbf{Bx}\| \leq \|\mathbf{By}\|$ *for any other* $\mathbf{y} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$.

The lack of efficient algorithms to solve SVP has led computer scientists to consider approximation versions of the problem. In this book we study this and other lattice problems from a computational point of view. Throughout the book, we assume the standard computational model of deterministic Turing machines. The reader is referred to (van Emde Boas, 1990; Johnson, 1990) or any undergraduate level textbook on the subject for an introduction to the basic theory of computability and computational complexity. In the following subsection we simply recall some terminology and basic definitions. Then, in Subsection 2.2 we describe SVP and other lattice problems in their exact and approximation versions, and in Subsection 2.3 we give some background about the computational complexity of approximation problems.

2.1 Complexity Theory

An *alphabet* is a finite set of symbols Σ. A *string* (over Σ) is a finite sequence of symbols from Σ. The *length* of a string y is the number of symbols in y, and it is denoted $|y|$. The set of all strings over Σ is denoted Σ^*, and the set of all strings of length n is denoted Σ^n. A Turing machine M runs in time $t(n)$ if for every input string w of length n (over some fixed input alphabet Σ), $M(n)$ halts after at most $t(n)$ steps. We identify the notion of efficient computation with Turing machines that halt in time polynomial in the size of the input, i.e., Turing machines that run in time $t(n) = a + n^b$ for some constants a, b independent of n. A *decision problem* is the problem of deciding whether the input string satisfies or not some specified property. Formally, a decision problem is specified by a *language*, i.e., a set of strings $L \subseteq \Sigma^*$, and the problem is given an input string $w \in \Sigma^*$ decide whether $w \in L$ or not. The class of decision problems that can be solved by a deterministic Turing machine in polynomial time is called P. The class of decision problem that can be solved by a nondeterministic Turing machine in polynomial time is called NP. Equivalently, NP can be characterized as the set of all languages L for which there exists a relation $R \subseteq \Sigma^* \times \Sigma^*$ such that $(x, y) \in R$ can be checked in time polynomial in $|x|$, and $x \in L$ if and only if there exists a string y with $(x, y) \in R$. Such string y is called NP-witness or NP-certificate of membership of x in L. Clearly, P $\subseteq$ NP, but it is widely believed that P $\neq$ NP, i.e., there are NP problems that cannot be solved in deterministic polynomial time.

Let A and B be two decision problems. A *(Karp) reduction* from A to B is a polynomial time computable function $f : \Sigma^* \to \Sigma^*$ such that $x \in A$ if and only if $f(x) \in B$. Clearly, if A reduces to B and B can be solved in polynomial time, then also A can be solved in polynomial time. A (decision) problem A is NP-*hard* if any other NP problem B

reduces to A. If A is also in NP, then A is NP-*complete.* Clearly, if a problem A is NP-hard, then A cannot be solved in polynomial time unless P = NP. The standard technique to prove that a problem A is NP-hard (and therefore no polynomial time solution for A is likely to exists) is to reduce some other NP-hard problem B to A. Another notion of reduction which will be used in this book is that of *Cook reduction.* A Cook reduction from A to B is a polynomial time Turing machine $\mathcal{M}$ with access to an oracle that takes instances of problem B as input. $\mathcal{M}$ reduces A to B, if, given an oracle that correctly solves problem B, $\mathcal{M}$ correctly solves problem A. A problem A is NP-hard under Cook reductions if for any NP problem B there is a Cook reduction from B to A. If A is in NP, then we say that A is NP-complete under Cook reductions. NP-hardness under Cook reductions also gives evidence of the intractability of a problem, because if A can be solved in polynomial time then P = NP. The reader is referred to (Garey and Johnson, 1979) for an introduction to the theory of NP-completeness and various NP-complete problems that will be used throughout the book.

In the rest of this book algorithms and reductions between lattice problems are described using some informal high level language, and decision problems are described as sets of mathematical objects, like graphs, matrices, etc. In all cases, the translation to strings, languages and Turing machines is straightforward.

Occasionally, we will make use of other complexity classes and different notions of reductions, e.g., randomized complexity classes or nonuniform reductions. When needed, these notions will be briefly recalled, or references will be given.

Throughout the book, we use the standard asymptotic notation to describe the order of growth of functions: for any positive real valued functions $f(n)$ and $g(n)$ we write

- $f = O(g)$ if there exists two constants a, b such that $f(n) \leq a \cdot f(n)$ for all $n \geq b$.

- $f = o(g)$ if $\lim_{n\to\infty} f(n)/g(n) = 0$

- $f = \Omega(g)$ if $g = O(f)$

- $f = \omega(g)$ if $g = o(f)$

- $f = \Theta(g)$ if $f = O(g)$ and $g = O(f)$.

A function f is negligible if $f = o(1/g)$ for any polynomial $g(n) = n^c$.

2.2 Some lattice problems

To date, we do not know any polynomial time algorithm to solve SVP. In fact, we do not even know how to find nonzero lattice vectors of length within the Minkowski's bound $\|\mathbf{Bx}\| < \sqrt{n}\det(\mathbf{B})^{1/n}$. Another related problem for which no polynomial time solution is known is the *Closest Vector Problem* .

DEFINITION 1.2 (CLOSEST VECTOR PROBLEM, CVP) *Given a lattice basis* $\mathbf{B} \in \mathbb{Z}^{m\times n}$ *and a target vector* $\mathbf{t} \in \mathbb{Z}^m$*, find a lattice vector* $\mathbf{Bx}$ *closest to the target* $\mathbf{t}$*, i.e., find an integer vector* $\mathbf{x} \in \mathbb{Z}^n$ *such that* $\|\mathbf{Bx} - \mathbf{t}\| \leq \|\mathbf{By} - \mathbf{t}\|$ *for any other* $\mathbf{y} \in \mathbb{Z}^n$*.*

Studying the computational complexity of these problems is the main subject of this book. Both for CVP and SVP one can consider different algorithmic tasks. These are (in decreasing order of difficulty):

- The *Search Problem*: Find a (nonzero) lattice vector $\mathbf{x} \in \Lambda$ such that $\|\mathbf{x} - \mathbf{t}\|$ (respectively, $\|\mathbf{x}\|$) is minimized.

- The *Optimization Problem*: Find the minimum of $\|\mathbf{x} - \mathbf{t}\|$ (respectively, $\|\mathbf{x}\|$) over $\mathbf{x} \in \Lambda$ (respectively, $\mathbf{x} \in \Lambda \setminus \{\mathbf{0}\}$).

- The *Decision Problem*: Given a rational $r > 0$, decide whether there is a (nonzero) lattice vector $\mathbf{x}$ such that $\|\mathbf{x} - \mathbf{t}\| \leq r$ (respectively,. $\|\mathbf{x}\| \leq r$).

We remark that to date virtually all known (exponential time) algorithms for SVP and CVP solve the search problem (and therefore also the associated optimization and decision problems), while all known hardness results hold for the decision problem (and therefore imply the hardness of the optimization and search problems as well). This suggests that the hardness of solving SVP and CVP is already captured by the decisional task of determining whether or not there exists a solution below some given threshold value. We will see in Chapter 3 that the decision problem associated to CVP is NP-complete, and therefore no algorithm can solve CVP in deterministic polynomial time, unless P = NP. A similar result holds (under randomized reductions) for SVP (see Chapter 4).

The hardness of solving SVP and CVP has led computer scientists to consider approximation versions of these problems. Approximation algorithms return solutions that are only guaranteed to be within some specified factor γ from the optimal. Approximation versions for the SVP and CVP search problems are defined below.

DEFINITION 1.3 (APPROXIMATE SVP) *Given a basis* $\mathbf{B} \in \mathbb{Z}^{m \times n}$*, find a nonzero lattice vector* $\mathbf{Bx}$ *(*$\mathbf{x} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$*) such that* $\|\mathbf{Bx}\| \leq \gamma \cdot \|\mathbf{By}\|$ *for any other* $\mathbf{y} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$.

In the optimization version of approximate SVP, one only needs to find $\|\mathbf{Bx}\|$, i.e., a value d such that $\lambda_1(\mathbf{B}) \leq d < \gamma\lambda_1(\mathbf{B})$.

DEFINITION 1.4 (APPROXIMATE CVP) *Given a basis* $\mathbf{B} \in \mathbb{Z}^{m \times n}$ *and a target vector* $\mathbf{t} \in \mathbb{Z}^m$*, find a lattice vector* $\mathbf{Bx}$ *(*$\mathbf{x} \in \mathbb{Z}^n$*) such that* $\|\mathbf{Bx} - \mathbf{t}\| \leq \gamma\|\mathbf{By} - \mathbf{t}\|$ *for any other* $\mathbf{y} \in \mathbb{Z}^n$.

In the optimization version of approximate CVP, one only need to find $\|\mathbf{Bx} - \mathbf{t}\|$, i.e., a value d such that $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq d < \gamma \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$. Both in the approximate SVP and CVP, the approximation factor γ can be a function of any parameter associated to the lattice, typically its rank n, to capture the fact that the problem gets harder as this parameter increases. To date, the best known polynomial time (possibly randomized) approximation algorithms for SVP and CVP achieve worst case (over the choice of the input) approximation factors $\gamma(n)$ that are essentially exponential in the rank n. Finding algorithms that achieve polynomial approximation factors $\gamma(n) = n^c$ (for some constant c independent of the rank n) is one of the main open problems in this area.

SVP and CVP are the two main problems studied in this book. Chapter 2 describes efficient algorithms to find approximate solutions to these problems (for large approximation factors). The computational complexity of CVP is studied in Chapter 3. The strongest known hardness result for SVP is the subject of Chapters 4, 5 and 6. There are many other lattice problems which are thought to be computationally hard. Some of them, which come up in the construction of lattice based cryptographic functions, are discussed in Chapter 7. There are also many computational problems on lattices that can be efficiently solved (in deterministic polynomial time). Here we recall just a few of them. Finding polynomial time solutions to these problems is left to the reader as an exercise.

1 *Membership:* Given a basis $\mathbf{B}$ and a vector $\mathbf{x}$, decide whether $\mathbf{x}$ belongs to the lattice $\mathcal{L}(\mathbf{B})$. This problem is essentially equivalent to solving a system of linear equations over the integers. This can be done in polynomially many arithmetic operations, but some care is needed to make sure the numbers involved do not get exponentially large.

2 *Kernel:* Given an integral matrix $\mathbf{A} \in \mathbb{Z}^{n \times m}$, compute a basis for the lattice $\{\mathbf{x} \in \mathbb{Z}^m \colon \mathbf{Ax} = \mathbf{0}\}$. A similar problem is, given a modulus M

and a matrix $\mathbf{A} \in \mathbb{Z}_M^{n \times m}$, find a basis for the lattice $\{\mathbf{x} \in \mathbb{Z}^m : \mathbf{Ax} = \mathbf{0} \pmod{M}\}$. Again, this is equivalent to solving a system of (homogeneous) linear equations.

3 *Basis:* Given a set of possibly dependent integer vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$, find a basis of the lattice they generate. This can be done in a variety of ways, for example using the Hermite Normal Form. (See Chapter 8.)

4 *Union:* Given two integer lattices $\mathcal{L}(\mathbf{B}_1)$ and $\mathcal{L}(\mathbf{B}_2)$, compute a basis for the smallest lattice containing both $\mathcal{L}(\mathbf{B}_1)$ and $\mathcal{L}(\mathbf{B}_2)$. This immediately reduces to the problem of computing a basis for the lattice generated by a sequence of possibly dependent vectors.

5 *Dual:* Given a lattice $\mathcal{L}(\mathbf{B})$, compute a basis for the dual of $\mathcal{L}(\mathbf{B})$, i.e., the set of all vectors $\mathbf{y}$ in span($\mathbf{B}$) such that $\langle \mathbf{x}, \mathbf{y} \rangle$ is an integer for every lattice vector $\mathbf{x} \in \mathcal{L}(\mathbf{B})$. It is easy to see that a basis for the dual is given by $\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}$.

6 *Intersection:* Given two integer lattices $\mathcal{L}(\mathbf{B}_1)$ and $\mathcal{L}(\mathbf{B}_2)$, compute a basis for the intersection $\mathcal{L}(\mathbf{B}_1) \cap \mathcal{L}(\mathbf{B}_2)$. It is easy to see that $\mathcal{L}(\mathbf{B}_1) \cap \mathcal{L}(\mathbf{B}_2)$ is always a lattice. This problem is easily solved using dual lattices.

7 *Equivalence:* Given two bases $\mathbf{B}_1$ and $\mathbf{B}_2$, check if they generate the same lattice $\mathcal{L}(\mathbf{B}_1) = \mathcal{L}(\mathbf{B}_2)$. This can be solved by checking if each basis vector belongs to the lattice generated by the other matrix, however, more efficient solutions exist.

8 *Cyclic:* Given a lattice $\mathcal{L}(\mathbf{C})$, check if $\mathcal{L}(\mathbf{C})$ is cyclic, i.e., if for every lattice vector $\mathbf{x} \in \mathcal{L}(\mathbf{C})$, all the vectors obtained by cyclically rotating the coordinates of $\mathbf{x}$ also belong to the lattice. This problem is easily solved by rotating the coordinates of basis matrix $\mathbf{C}$ by one position, and checking if the resulting basis is equivalent to the original one.

2.3 Hardness of approximation

In studying the computational complexity of approximating lattice problems, it is convenient to formulate them as *promise problems.* These are a generalization of decision problems well suited to study the hardness of approximation. A promise problem is a pair $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ of disjoint languages, i.e., $\Pi_{\text{YES}}, \Pi_{\text{NO}} \subseteq \Sigma^*$ and $\Pi_{\text{YES}} \cap \Pi_{\text{NO}} = \emptyset$. An algorithm solves the promise problem $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ if on input an instance $I \in \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$ it correctly decides whether $I \in \Pi_{\text{YES}}$ or $I \in \Pi_{\text{NO}}$. The behavior of the algorithm when $I \not\in \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$ (i.e., when I does not satisfy the promise) is not specified, i.e., on input an instance outside the promise, the algorithm is allowed to return any answer.

Decision problems are a special case of promise problems, where the set $\Pi_{\text{NO}} = \Sigma^* \setminus \Pi_{\text{YES}}$ is implicitly specified and the promise $I \in \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$ is vacuously true. We now define the promise problems associated to the approximate SVP and CVP. These are denoted GAPSVP_γ and GAPCVP_γ.

DEFINITION 1.5 *The promise problem* GAPSVP_γ, *where* γ *(the gap function) is a function of the rank, is defined as follows:*

- YES *instances are pairs* $(\mathbf{B}, r)$ *where* $\mathbf{B} \in \mathbb{Z}^{m\times n}$ *is a lattice basis and* $r \in \mathbb{Q}$ *a rational number such that* $\|\mathbf{Bz}\| \leq r$ *for some* $\mathbf{z} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$.
- NO *instances are pairs* $(\mathbf{B}, r)$ *where* $\mathbf{B} \in \mathbb{Z}^{m\times n}$ *is a lattice basis and* $r \in \mathbb{Q}$ *is a rational such that* $\|\mathbf{Bz}\| > \gamma r$ *for all* $\mathbf{z} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$.

DEFINITION 1.6 *The promise problem* GAPCVP_γ, *where* γ *(the gap function) is a function of the rank, is defined as follows:*

- YES *instances are triples* $(\mathbf{B}, \mathbf{t}, r)$ *where* $\mathbf{B} \in \mathbb{Z}^{m\times n}$ *is a lattice basis,* $\mathbf{t} \in \mathbb{Z}^m$ *is a vector and* $r \in \mathbb{Q}$ *is a rational number such that* $\|\mathbf{Bz} - \mathbf{t}\| \leq r$ *for some* $\mathbf{z} \in \mathbb{Z}^n$.
- NO *instances are triples* $(\mathbf{B}, \mathbf{t}, r)$ *where* $\mathbf{B} \in \mathbb{Z}^{m\times n}$ *is a lattice,* $\mathbf{t} \in \mathbb{Z}^m$ *is a vector and* $r \in \mathbb{Q}$ *is a rational number such that* $\|\mathbf{Bz} - \mathbf{t}\| > \gamma r$ *for all* $\mathbf{z} \in \mathbb{Z}^n$.

Notice that when the approximation factor equals $\gamma = 1$, the promise problems GAPSVP_γ and GAPCVP_γ are equivalent to the decision problems associated to exact SVP and CVP. Occasionally, with slight abuse of notation, we consider instances $(\mathbf{B}, r)$ (or $(\mathbf{B}, \mathbf{t}, r)$) where r is a real number, e.g., $r = \sqrt{2}$. This is seldom a problem in practice, because r can always be replaced by a suitable rational approximation. For example, in the ℓ_2 norm, if $\mathbf{B}$ is an integer lattice then r can be substituted with any rational in the interval $[r, \sqrt{r^2+1})$. Promise problems GAPSVP_γ and GAPCVP_γ capture the computational task of approximating SVP and CVP within a factor γ in the following sense. Assume algorithm $\mathcal{A}$ approximately solves SVP within a factor γ, i.e., on input a lattice Λ, it finds a vector $\mathbf{x} \in \Lambda$ such that $\|\mathbf{x}\| \leq \gamma\lambda_1(\Lambda)$. Then $\mathcal{A}$ can be used to solve GAPSVP_γ as follows. On input $(\mathbf{B}, r)$, run algorithm $\mathcal{A}$ on lattice $\mathcal{L}(\mathbf{B})$ to obtain an estimate $r' = \|\mathbf{x}\| \in [\lambda_1, \gamma\lambda_1]$ of the shortest vector length. If $r' > \gamma r$ then $\lambda_1 > r$, i.e., $(\mathbf{B}, r)$ is not a YES instance. Since $(\mathbf{B}, r) \in \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$, $(\mathbf{B}, r)$ must be a NO instance. Conversely, if $r' < \gamma r$ then $\lambda_1 < \gamma r$ and from the promise $(\mathbf{B}, r) \in \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$ one deduces that $(\mathbf{B}, r)$ is a YES instance. On the other hand, assume one has a decision oracle $\mathcal{A}$ that solves GAPSVP$_\gamma$. (By definition, when the input does not satisfy the promise, the oracle can return any answer.) Let $u \in \mathbb{Z}$ be an upper bound to $\lambda(\mathbf{B})^2$ (for example, let u be the squared length of any of the basis vectors). Notice that $\mathcal{A}(\mathbf{B}, \sqrt{u})$ always returns YES, while $\mathcal{A}(\mathbf{B}, 0)$ always returns NO. Using binary search find an integer $r \in \{0, \ldots, u\}$ such that $\mathcal{A}(\mathbf{B}, \sqrt{r}) =$ YES and $\mathcal{A}(\mathbf{B}, \sqrt{r-1}) =$ NO. Then, $\lambda_1(\mathbf{B})$ must lie in the interval $[\sqrt{r}, \gamma \cdot \sqrt{r})$. A similar argument holds for the closest vector problem.

The class NP is easily extended to include promise problems. We say that a promise problem $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ is in NP if there exists a relation $R \subseteq \Sigma^* \times \Sigma^*$ such that $(x, y) \in R$ can be decided in time polynomial in $|x|$, and for every $x \in \Pi_{\text{YES}}$ there exists a y such that $(x, y) \in R$, while for every $y \in \Pi_{\text{NO}}$ there is no y such that $(x, y) \in R$. If the input x does not satisfies the promise, then R may or may not contain a pair (x, y). The complement of a promise problem $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ is the promise problem $(\Pi_{\text{NO}}, \Pi_{\text{YES}})$. For decision problems, this is the same as taking the set complement of a language in Σ^*. The class of decision problems whose complement is in NP is denoted coNP. Also coNP can be extended to include the complements of all NP promise problems.

Reductions between promise problems are defined in the obvious way. A function $f\colon \Sigma^* \to \Sigma^*$ is a reduction from $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ to $(\Pi'_{\text{YES}}, \Pi'_{\text{NO}})$ if it maps YES instances to YES instances and NO instances to NO instances, i.e., $f(\Pi_{\text{YES}}) \subseteq \Pi'_{\text{YES}}$ and $f(\Pi_{\text{NO}}) \subseteq \Pi'_{\text{NO}}$. Clearly any algorithm $\mathcal{A}$ to solve $(\Pi'_{\text{YES}}, \Pi'_{\text{NO}})$ can be used to solve $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ as follows: on input $I \in \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$, run $\mathcal{A}$ on $f(I)$ and output the result. Notice that $f(I)$ always satisfy the promise $f(I) \in \Pi'_{\text{YES}} \cup \Pi'_{\text{NO}}$, and $f(I)$ is a YES instance if and only if I is a YES instance. A promise problem A is NP-hard if any NP language (or, more generally, any NP promise problem) B can be efficiently reduced to A. As usual, proving that a promise problem is NP-hard shows that no polynomial time solution for the problem exists unless P = NP. In the case of Cook reductions, the oracle Turing machine $\mathcal{A}$ to solve problem $(\Pi_{\text{YES}}, \Pi_{\text{NO}})$ should work given *any* oracle that solves $(\Pi'_{\text{YES}}, \Pi'_{\text{NO}})$. In particular, $\mathcal{A}$ should work no matter how queries outside the promise are answered by the oracle.

3. Notes

For a general introduction to computational models and complexity classes as used in this book, the reader is referred to (van Emde Boas, 1990) and (Johnson, 1990), or any undergraduate level textbook on the subject. Classical references about lattices are (Cassels, 1971) and (Gruber and Lekerkerker, 1987). Another very good reference is (Siegel, 1989). The proof of Minkowski's second theorem presented in Subsection 1.3 is an adaption to the Euclidean norm of the proof given in (Siegel, 1989) for arbitrary norms. None of the above references address algorithmic issues related to lattice problems, and lattices are studied from a purely mathematical point of view. For a brief introduction to the applications of lattices in various areas of mathematics and science the reader is referred to (Lagarias, 1995) and (Gritzmann and Wills, 1993), which also touch some complexity and algorithmic issues. A very good survey of algorithmic application of lattices is (Kannan, 1987a).

Chapter 2

APPROXIMATION ALGORITHMS

In this chapter we describe efficient algorithms to approximately solve SVP and CVP. For both problems, we solve the search version: we give polynomial time algorithms to find approximately shortest nonzero vectors in a lattice, or lattice vectors approximately closest to a given target point. The approximation factor achieved is exponential in the rank of the lattice. In Section 1 we start with an algorithm to solve SVP in dimension 2. For the special case of 2-dimensional lattices, we are able to solve SVP exactly and find a lattice vector of length $\|\mathbf{a}\| = \lambda_1$. In fact, we can find a lattice basis $[\mathbf{a}, \mathbf{b}]$ with $\|\mathbf{a}\| = \lambda_1$ and $\|\mathbf{b}\| = \lambda_2$. So, the algorithm determines all successive minima of the lattice. The algorithm works for any (efficiently computable) norm $\|\cdot\|$, and it is, essentially, the generalization to arbitrary norms of an algorithm of Gauss. Then, in Section 2, we extend Gauss algorithm to n-dimensional lattices. This is the famous Lenstra-Lenstra-Lovász (LLL) lattice reduction algorithm (Lenstra et al., 1982). The extension comes at a price: the LLL algorithm does not find a lattice vector of length λ_1, but only a $\gamma(n) = (2/\sqrt{3})^n$ approximation, i.e., a nonzero lattice vector of length at most $\gamma(n) \cdot \lambda_1$. Finally, in Section 3 we use the LLL algorithm to approximately solve CVP. Also for CVP, the (worst case) approximation factor achieved is $O((2/\sqrt{3})^n)$ where n is the rank of the lattice. Section 4 concludes the chapter with an overview of the latest developments in the design of approximation algorithms for lattice problems, and (exponential time) algorithms to solve lattice problems exactly.

D. Micciancio et al., *Complexity of Lattice Problems*

1. Solving SVP in dimension 2

In this section we describe an algorithm to solve SVP for lattices in dimension 2. The algorithm is generic with respect to the norm, i.e., it correctly computes a shortest vector in the lattice with respect to any norm $\|\cdot\|$, provided $\|\cdot\|$ can be efficiently evaluated. In the rest of this section $\|\cdot\|$ is an arbitrary, but fixed, norm. The input to the algorithm is a pair of linearly independent (integer) vectors $\mathbf{a}, \mathbf{b}$. We want to find a new basis $[\mathbf{a}', \mathbf{b}']$ for $\mathcal{L}([\mathbf{a}, \mathbf{b}])$ such that $\|\mathbf{a}'\| = \lambda_1$ and $\|\mathbf{b}'\| = \lambda_2$, where λ_1 and λ_2 are the minima of the lattice with respect to $\|\cdot\|$. The presentation is structured as follows:

- In Subsection 1.1 we introduce a notion of reduced basis (for two dimensional lattices), and prove that a basis is reduced if and only if the basis vectors have length λ_1 and λ_2.

- In Subsection 1.2 we give an algorithm that on input a 2-dimensional lattice, computes a reduced basis.

- Finally, in Subsection 1.3 we prove that the algorithm terminates in polynomial time.

1.1 Reduced basis

We define a *reduced basis* for 2-dimensional lattices as follows.

DEFINITION 2.1 *Let* $[\mathbf{a}, \mathbf{b}]$ *be a lattice basis. The basis is* reduced *(with respect to norm* $\|\cdot\|$*) if*

$$\|\mathbf{a}\|, \|\mathbf{b}\| \leq \|\mathbf{a} + \mathbf{b}\|, \|\mathbf{a} - \mathbf{b}\|.$$

Geometrically, this definition means that the diagonals of the fundamental parallelepiped associated to the basis of the lattice are at least as long as the edges. (See Figure 2.1.) This definition of reduced basis is motivated by the fact that a basis is reduced if and only if $\mathbf{a}$ and $\mathbf{b}$ have length λ_1 and λ_2. In order to prove this fact we need the following lemma, which, informally, states that if our distance from some point increases as we move in a straight line, then the distance will keep increasing as we keep moving in the same direction.

LEMMA 2.1 *Consider three vectors on a line,* $\mathbf{x}$, $\mathbf{x}+\mathbf{y}$, *and* $\mathbf{x}+\alpha\mathbf{y}$, *where* $\alpha \in (1, \infty)$. *(See Figure 2.2.) For any norm* $\|\cdot\|$, *if* $\|\mathbf{x}\| \leq \|\mathbf{x}+\mathbf{y}\|$ *then* $\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}+\alpha\mathbf{y}\|$. *Moreover, if* $\|\mathbf{x}\| < \|\mathbf{x}+\mathbf{y}\|$ *then* $\|\mathbf{x}+\mathbf{y}\| < \|\mathbf{x}+\alpha\mathbf{y}\|$.

Proof: We prove the lemma for the case in which the inequality is strict. The proof of the other case is easily obtained replacing all "<" signs with

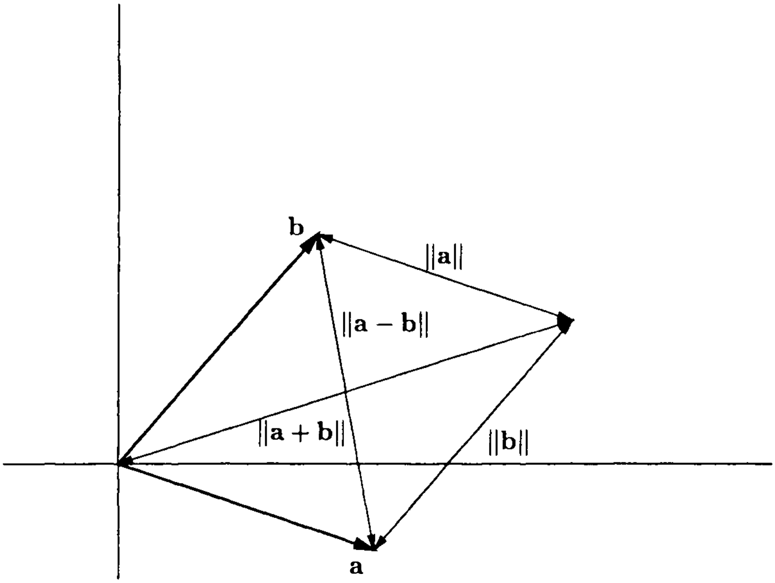


Figure 2.1. A reduced basis in 2 dimensions

"$\leq$". Let $\delta = 1/\alpha$. Then

$$\mathbf{x} + \mathbf{y} = (1 - \delta)\mathbf{x} + \delta(\mathbf{x} + \alpha\mathbf{y})$$

By triangle inequality

$$||\mathbf{x} + \mathbf{y}|| \leq (1 - \delta)||\mathbf{x}|| + \delta||\mathbf{x} + \alpha\mathbf{y}||. \tag{2.1}$$

Also, from $||\mathbf{x}|| < ||\mathbf{x} + \mathbf{y}||$ we get

$$(1 - \delta)||\mathbf{x}|| + \delta||\mathbf{x} + \alpha\mathbf{y}|| < (1 - \delta)||\mathbf{x} + \mathbf{y}|| + \delta||\mathbf{x} + \alpha\mathbf{y}||. \tag{2.2}$$

Combining (2.1) and (2.2) we get

$$||\mathbf{x} + \mathbf{y}|| < (1 - \delta)||\mathbf{x} + \mathbf{y}|| + \delta||\mathbf{x} + \alpha\mathbf{y}||$$

which, after rearranging and simplifying the terms, gives

$$\delta||\mathbf{x} + \mathbf{y}|| < \delta||\mathbf{x} + \alpha\mathbf{y}||. \tag{2.3}$$

Since $\delta > 0$, we can divide (2.3) by δ and get $||\mathbf{x} + \mathbf{y}|| < ||\mathbf{x} + \alpha\mathbf{y}||$ as claimed in the lemma. □

We can now establish a relation between reduced bases and the successive minima of the lattice.

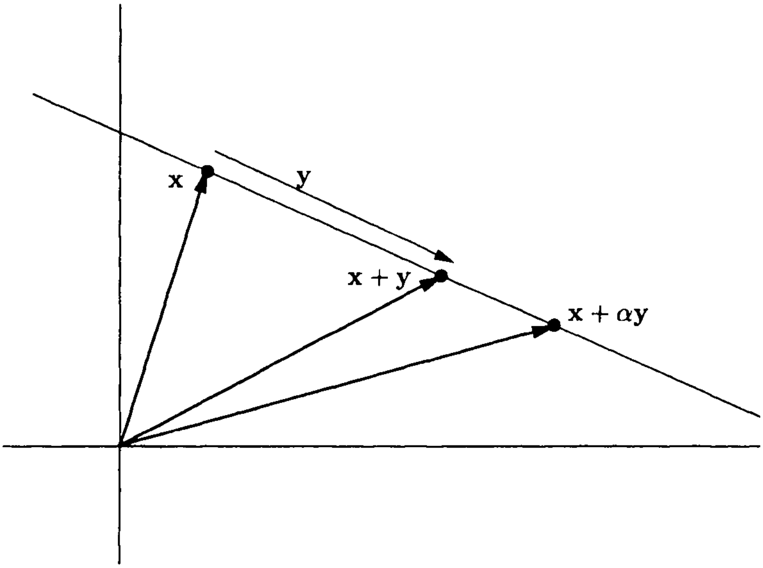


Figure 2.2. Three points on a line

THEOREM 2.2 *Let* $[\mathbf{a}, \mathbf{b}]$ *be a lattice basis, and let* λ_1 *and* λ_2 *be the successive minima of the lattice. Then,* $[\mathbf{a}, \mathbf{b}]$ *is reduced if and only if* $\mathbf{a}$ *and* $\mathbf{b}$ *have norm* λ_1 *and* λ_2.

Proof: First assume that the lengths of $\mathbf{a}$ and $\mathbf{b}$ equal the successive minima of the lattice, and assume, without loss of generality that $\|\mathbf{a}\| \leq \|\mathbf{b}\|$, i.e., $\|\mathbf{a}\| = \lambda_1$ and $\|\mathbf{b}\| = \lambda_2$. By definition of λ_1 we know that $\|\mathbf{a} - \mathbf{b}\|$ and $\|\mathbf{a} + \mathbf{b}\|$ are at least as large as $\|\mathbf{a}\|$. Moreover, since $[\mathbf{a}, \mathbf{b}]$ is a basis, $\mathbf{b}$ is linearly independent from $\mathbf{a}$, and therefore each of $\mathbf{a} - \mathbf{b}$ and $\mathbf{a} + \mathbf{b}$ is linearly independent from $\mathbf{a}$. By definition of the second minimum λ_2 we get

$$\lambda_2 \leq \max\{\|\mathbf{a}\|, \|\mathbf{a} - \mathbf{b}\|\} = \|\mathbf{a} - \mathbf{b}\|$$

and

$$\lambda_2 \leq \max\{\|\mathbf{a}\|, \|\mathbf{a} + \mathbf{b}\|\} = \|\mathbf{a} + \mathbf{b}\|.$$

This proves that

$$\|\mathbf{a}\|, \|\mathbf{b}\| \leq \lambda_2 \leq \|\mathbf{a} + \mathbf{b}\|, \|\mathbf{a} - \mathbf{b}\|.$$

Now assume that $\|\mathbf{a}\|, \|\mathbf{b}\| \leq \|\mathbf{a} + \mathbf{b}\|, \|\mathbf{a} - \mathbf{b}\|$. Also assume, without loss of generality, that $\|\mathbf{a}\| \leq \|\mathbf{b}\|$. We want to prove that $\|\mathbf{a}\| = \lambda_1$ and

$\|\mathbf{b}\| = \lambda_2$. Let $r, s \in \mathbb{Z}$, and consider a generic lattice vector $r\mathbf{a} + s\mathbf{b}$. We show that

$$\|\mathbf{a}\| < \|r\mathbf{a} + s\mathbf{b}\| \qquad \text{for all } (r,s) \neq (0,0) \tag{2.4}$$

and

$$\|\mathbf{b}\| \leq \|r\mathbf{a} + s\mathbf{b}\| \qquad \text{for all } s \neq 0. \tag{2.5}$$

Notice that (2.4) says that $\mathbf{a}$ is at least as short as any other nonzero lattice vector, i.e., $\|\mathbf{a}\| = \lambda_1$. Similarly, (2.5) says that $\mathbf{b}$ is at least as short as any lattice vector linearly independent from $\mathbf{a}$. Provided $\mathbf{a}$ is a shortest vector in the lattice, this proves that $\|\mathbf{b}\| = \lambda_2$. In order to prove (2.4) and (2.5) we distinguish three cases:

- If $s = 0$ then $r \neq 0$ and $\|\mathbf{a}\| \leq \|r\mathbf{a}\| = \|r\mathbf{a} + s\mathbf{b}\|$, proving (2.4).

- If $r = 0$ then $s \neq 0$ and $\|\mathbf{a}\| \leq \|\mathbf{b}\| \leq \|s\mathbf{b}\| = \|r\mathbf{a} + s\mathbf{b}\|$, proving both (2.4) and (2.5).

- Finally, if $r, s \neq 0$ are both nonzero, assume $r \geq s \geq 0$ (the other cases are similar and left to the reader as an exercise). Since s is a nonzero integer, we have $s \geq 1$ and therefore

  $$\|(r/s)\mathbf{a} + \mathbf{b}\| = \left\|\frac{r\mathbf{a} + s\mathbf{b}}{s}\right\| \leq \|r\mathbf{a} + s\mathbf{b}\|.$$

  Now consider the three points $\|\mathbf{b}\|$, $\|\mathbf{b}+\mathbf{a}\|$ and $\|\mathbf{b} + (r/s)\mathbf{a}\|$. Notice that $\|\mathbf{b}\| \leq \|\mathbf{b} + \mathbf{a}\|$ and $r/s \geq 1$. Therefore by Lemma 2.1

  $$\|\mathbf{a}\|, \|\mathbf{b}\| \leq \|\mathbf{a} + \mathbf{b}\| \leq \|\mathbf{b} + (r/s)\mathbf{a}\| \leq \|r\mathbf{a} + s\mathbf{b}\|$$

  proving (2.4) and (2.5).

This completes the proof that the vectors of a reduced basis are as short as possible. □

1.2 Gauss' algorithm

In this subsection we describe an algorithm to find a reduced basis for any 2-dimensional lattice. The algorithm, given in Figure 2.3, works by computing a sequence of bases satisfying the following property.

DEFINITION 2.2 *A basis* $[\mathbf{a}, \mathbf{b}]$ *is* well ordered *if* $\|\mathbf{a}\| \leq \|\mathbf{a} - \mathbf{b}\| < \|\mathbf{b}\|$.

Since the input basis $[\mathbf{a}, \mathbf{b}]$ is not necessarily well ordered, the first thing to do is to compute a well ordered (or reduced) basis for $\mathcal{L}([\mathbf{a}, \mathbf{b}])$. This is easily accomplished by a simple case analysis. (See part of the

```
Input: two linearly independent vectors a and b.
Output: a reduced basis for lattice $\mathcal{L}([\mathbf{a},\mathbf{b}])$.

(start): if $\|\mathbf{a}\| > \|\mathbf{b}\|$ then swap(a, b)
         if $\|\mathbf{a}-\mathbf{b}\| > \|\mathbf{a}+\mathbf{b}\|$ then let $\mathbf{b} := -\mathbf{b}$
         if $\|\mathbf{b}\| \leq \|\mathbf{a}-\mathbf{b}\|$ then return $[\mathbf{a},\mathbf{b}]$
         if $\|\mathbf{a}\| \leq \|\mathbf{a}-\mathbf{b}\|$ then go to (loop)
         if $\|\mathbf{a}\| = \|\mathbf{b}\|$ then return $[\mathbf{a},\mathbf{a}-\mathbf{b}]$
         let $[\mathbf{a},\mathbf{b}] := [\mathbf{b}-\mathbf{a},\mathbf{a}]$
(loop):  Find $\mu \in \mathbb{Z}$ such that $\|\mathbf{b}-\mu\mathbf{a}\|$ is minimal
         if $\|\mathbf{a}-\mathbf{b}\| > \|\mathbf{a}+\mathbf{b}\|$ then let $\mathbf{b} := -\mathbf{b}$
         swap(a,b)
         if $[\mathbf{a},\mathbf{b}]$ is reduced
            then return $[\mathbf{a},\mathbf{b}]$
            else go to (loop)
```

Figure 2.3. The generalized Gauss algorithm

code in Figure 2.3 before the (loop) label.) Details follow. Without loss of generality, assume that $\|\mathbf{a}\| \leq \|\mathbf{b}\|$ (which can be achieved by possibly swapping **a** and **b**) and $\|\mathbf{a}-\mathbf{b}\| \leq \|\mathbf{a}+\mathbf{b}\|$ (which can be achieved by possibly changing the sign of **b**). If $\|\mathbf{b}\| \leq \|\mathbf{a}-\mathbf{b}\|$ then $[\mathbf{a},\mathbf{b}]$ is reduced and the algorithm immediately terminates. So, we can assume $\|\mathbf{a}-\mathbf{b}\| < \|\mathbf{b}\|$. If $\|\mathbf{a}\| \leq \|\mathbf{a}-\mathbf{b}\|$ then $[\mathbf{a},\mathbf{b}]$ is well ordered and we can proceed to the loop. So, assume $\|\mathbf{a}-\mathbf{b}\| < \|\mathbf{a}\|$. If $\|\mathbf{a}\| < \|\mathbf{b}\|$, then $[\mathbf{b}-\mathbf{a},-\mathbf{a}]$ is a well ordered basis, and we can proceed to the loop after suitably modifying the basis. The only other case is $\|\mathbf{a}-\mathbf{b}\| < \|\mathbf{a}\| = \|\mathbf{b}\|$, but in this case $[\mathbf{a},\mathbf{a}-\mathbf{b}]$ is reduced because $\|\mathbf{a}-(\mathbf{a}-\mathbf{b})\| = \|\mathbf{b}\| = \|\mathbf{a}\|$ and $\|\mathbf{a}+(\mathbf{a}-\mathbf{b})\| = \|2\mathbf{a}-\mathbf{b}\| \geq 2\|\mathbf{a}\| - \|\mathbf{b}\| = \|\mathbf{a}\|$.

At this point, unless a reduced basis has been found, we have a well ordered basis $[\mathbf{a},\mathbf{b}]$ and the algorithm enters a loop which consists of the following three steps.

1 Find an integer μ such that the value of $\|\mathbf{b} - \mu\mathbf{a}\|$ is minimal. In other words, we make **b** as short as possible by subtracting an integer multiple of **a**. (See below for details.)

2 If $\|\mathbf{a}-\mathbf{b}\| > \|\mathbf{a}+\mathbf{b}\|$, then let $\mathbf{b} = -\mathbf{b}$. Notice that at the end of this step we always have $\|\mathbf{a}-\mathbf{b}\| \leq \|\mathbf{a}+\mathbf{b}\|$.

3 If $[\mathbf{a},\mathbf{b}]$ is not reduced, then swap **a** and **b** and go to the first step.

The problem of finding an integer μ such that $\|\mathbf{b} - \mu\mathbf{a}\|$ is minimized needs further explanations. In the following lemma we show how this can be efficiently done for any efficiently computable norm.

LEMMA 2.3 *Let* $\|\cdot\|$ *an efficiently computable norm, and let* $\mathbf{a}$ *and* $\mathbf{b}$ *be two vectors such that* $\|\mathbf{b}\| > \|\mathbf{b} - \mathbf{a}\|$. *Then, one can efficiently find an integer* μ *such that* $\|\mathbf{b} - \mu\mathbf{a}\|$ *is minimal. Moreover,* μ *satisfies* $\mu \geq 1$ *and* $\mu \leq 2\|\mathbf{b}\|/\|\mathbf{a}\|$.

Proof: Let $c = \lceil 2\|\mathbf{b}\|/\|\mathbf{a}\| \rceil$. By triangle inequality,

$$\|\mathbf{b} - c\mathbf{a}\| \geq c\|\mathbf{a}\| - \|\mathbf{b}\| \geq \|\mathbf{b}\|,$$

and, using Lemma 2.1, we get $\|\mathbf{b} - c\mathbf{a}\| \leq \|\mathbf{b} - (c+1)\mathbf{a}\|$. So, we see that $\|\mathbf{b} - k\mathbf{a}\| \leq \|\mathbf{b} - (k+1)\mathbf{a}\|$ is true for $k = c$, but it is false for $k = 0$. Using binary search we can efficiently find an integer μ between 1 and c such that $\|\mathbf{b} - k\mathbf{a}\| \leq \|\mathbf{b} - (k+1)\mathbf{a}\|$ is true for $k = \mu + 1$ and false for $k = \mu$, i.e.,

$$\|\mathbf{b} - (\mu - 1)\mathbf{a}\| > \|\mathbf{b} - \mu\mathbf{a}\| \leq \|\mathbf{b} - (\mu + 1)\mathbf{a}\|.$$

We claim that this value of μ minimizes the norm $\|\mathbf{b} - k\mathbf{a}\|$ (over all possible integers k). In fact, by Lemma 2.1, for all $k \geq \mu + 1$, we have $\|\mathbf{b} - \mu\mathbf{a}\| \leq \|\mathbf{b} - (\mu + 1)\mathbf{a}\| \leq \|\mathbf{b} - k\mathbf{a}\|$. Similarly, for all $k \leq \mu - 1$, $\|\mathbf{b} - \mu\mathbf{a}\| < \|\mathbf{b} - (\mu - 1)\mathbf{a}\| \leq \|\mathbf{b} - k\mathbf{a}\|$. □

In order to use Lemma 2.3, we need to show that at the beginning of each iteration, $[\mathbf{a}, \mathbf{b}]$ is well ordered, and therefore $\|\mathbf{b}\| > \|\mathbf{a} - \mathbf{b}\|$. This is proved in the next lemma.

LEMMA 2.4 *In any execution of the generalized Gauss algorithm, at the beginning of each iteration basis* $[\mathbf{a}, \mathbf{b}]$ *is well ordered.*

Proof: We have already seen that the basis is well ordered the first time the loop is entered. We need to prove that at the end of each iteration $[\mathbf{a}, \mathbf{b}]$ is either reduced (in which case the program terminates) or well ordered (in which case the loop is repeated). Let $[\mathbf{a}, \mathbf{b}]$ be the (well ordered basis) at the beginning of the loop, and let $[\mathbf{a}', \mathbf{b}']$ be the basis computed by the body of the loop. We have $\mathbf{a}' = \pm(\mathbf{b} - \mu\mathbf{a})$ and $\mathbf{b}' = \mathbf{a}$. From the second step of the loop we know that $\|\mathbf{a}' - \mathbf{b}'\| \leq \|\mathbf{a}' + \mathbf{b}'\|$. Moreover, we know that $\|\mathbf{a}' - \mathbf{b}'\| = \|\pm(\mathbf{b} - \mu\mathbf{a}) - \mathbf{a}\| = \|\mathbf{b} - (\mu \pm 1)\mathbf{a}\|$, which, by the choice of μ, is at least $\|(\mathbf{b} - \mu\mathbf{a})\| = \|\mathbf{a}'\|$. This proves that $\|\mathbf{a}'\| \leq \|\mathbf{a}' - \mathbf{b}'\| \leq \|\mathbf{a}' + \mathbf{b}'\|$. Now there are two possible cases. If $\|\mathbf{b}'\| \leq \|\mathbf{a}' - \mathbf{b}'\|$, then $[\mathbf{a}', \mathbf{b}']$ is reduced. Otherwise, $\|\mathbf{b}'\| > \|\mathbf{a}' - \mathbf{b}'\| \geq \|\mathbf{a}'\|$ and $[\mathbf{a}', \mathbf{b}']$ is well ordered. □

We are now ready to prove the correctness of the generalized Gauss algorithm.

THEOREM 2.5 *On input any two linearly independent vectors* $[\mathbf{a}, \mathbf{b}]$*, the generalized Gauss algorithm shown in Figure 2.3 always terminates and correctly computes a reduced basis for lattice* $\mathcal{L}([\mathbf{a}, \mathbf{b}])$.

Proof: The algorithm performs elementary column operations, therefore $[\mathbf{a}, \mathbf{b}]$ is always a basis of the original lattice. Moreover, if the algorithm terminates then the basis $[\mathbf{a}, \mathbf{b}]$ is clearly reduced. It only remains to be proved that the algorithm actually terminates and does not loop forever. This is easily argued as follows. We know from Lemma 2.4 that at the beginning of each iteration, $[\mathbf{a}, \mathbf{b}]$ is well ordered. In particular $\|\mathbf{b} - \mathbf{a}\|$ is strictly less than $\mathbf{b}$. Therefore, $\mathbf{b}$ is replaced by a new vector $\mathbf{b} - \mu\mathbf{a}$ strictly shorter than $\mathbf{b}$. This proves that at every iteration one of the basis vectors gets strictly shorter. In particular, the values taken by the pairs $[\mathbf{a}, \mathbf{b}]$ never repeat, and they are all shorter than the input basis. Since there are only finitely many lattice vectors of length at most $\|\mathbf{a}\| + \|\mathbf{b}\|$, the algorithm must stop after a finite number of iterations. □

1.3 Running time analysis

In this section we prove that the generalized Gauss algorithm terminates in polynomial time. We have already proved that the algorithm always terminates and that each iteration can be performed in polynomial time. We still have to show that the number of iterations is also polynomial in the size of the input.

Let k be the total number of iterations performed on input $[\mathbf{a}, \mathbf{b}]$. Let $[\mathbf{a}_k, \mathbf{a}_{k+1}]$ be the (well ordered) basis at the beginning of the first iteration. Any subsequent iteration is performed on a well ordered basis $[\mathbf{a}_i, \mathbf{a}_{i+1}]$, until a reduced basis $[\mathbf{a}_1, \mathbf{a}_2]$ is found. (Notice: we are using the indices in reverse order!) Let $(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k, \mathbf{a}_{k+1})$ denote the sequence of values obtained during the execution of the algorithm.

In order to prove that k is polynomial in the size of the input, we show that the length of $\mathbf{a}_i$ decreases at every iteration at least by a factor 2.

LEMMA 2.6 *For every* $i \geq 3$, $\|\mathbf{a}_i\| < 1/2\|\mathbf{a}_{i+1}\|$.

Proof: Consider the subsequence $(\mathbf{a}_{i-1}, \mathbf{a}_i, \mathbf{a}_{i+1})$. In order to keep notation simple, we rename this sequence as $(\mathbf{a}, \mathbf{b}, \mathbf{c})$. We know that $[\mathbf{a}, \mathbf{b}]$ and $[\mathbf{b}, \mathbf{c}]$ are both well ordered, $\|\mathbf{a}\| < \|\mathbf{b}\| < \|\mathbf{c}\|$, and $\mathbf{a} = \epsilon(\mathbf{c} - \mu\mathbf{b})$ for some integer $\mu \geq 1$ and $\epsilon = \pm 1$. Multiplying both sides by ϵ, we get $\mathbf{c} = \epsilon\mathbf{a} + \mu\mathbf{b}$. We prove that $\|\mathbf{c}\| > 2\|\mathbf{b}\|$ by cases:

- Case $\mu = 1$. This case is not possible because $\|\mathbf{c} - \mathbf{b}\| = \|\mathbf{a}\| < \|\mathbf{b}\|$ would contradict the assumption that $[\mathbf{b}, \mathbf{c}]$ is well ordered.

- Case $\epsilon = -1, \mu = 2$. Also not possible because $\|\mathbf{c} - \mathbf{b}\| = \| - \mathbf{a} + \mathbf{b}\|$ would contradict either $\|\mathbf{a} - \mathbf{b}\| < \|\mathbf{b}\|$ or $\|\mathbf{b}\| < \|\mathbf{b} - \mathbf{c}\|$.

- Case $\epsilon = -1, \mu > 2$. In this case,

$$\|\mathbf{c}\| = \| - \mathbf{a} + \mu\mathbf{b}\| \geq \mu\|\mathbf{b}\| - \|\mathbf{a}\|$$

which, by $\|\mathbf{a}\| < \|\mathbf{b}\|$, is strictly bigger than

$$\mu\|\mathbf{b}\| - \|\mathbf{b}\| = (\mu - 1)\|\mathbf{b}\| \geq 2\|\mathbf{b}\|.$$

- Case $\epsilon = 1, \mu \geq 2$. We know that $\|\mathbf{b} - \mathbf{a}\| < \|\mathbf{b}\|$ because $[\mathbf{a}, \mathbf{b}]$ is well ordered. Therefore, by Lemma 2.1, $\|\mathbf{b}\| < \|\mathbf{b} + \mathbf{a}\|$. Using $\|\mathbf{a}\| \leq \|\mathbf{a} - \mathbf{b}\|$ we also get $\|\mathbf{a}\| < \|\mathbf{b} + \mathbf{a}\|$, and by repeated application of Lemma 2.1

$$\|\mathbf{a}\| < \|\mathbf{a} + \mathbf{b}\| < \|\mathbf{a} + 2\mathbf{b}\| \leq \|\mathbf{a} + \mu\mathbf{b}\|.$$

This proves that $\|\mathbf{c}\| = \|\mathbf{a} + \mu\mathbf{b}\| \geq \|2\mathbf{b} + \mathbf{a}\|$. We want to prove that $\|2\mathbf{b} + \mathbf{a}\| > 2\|\mathbf{b}\|$. Consider the point $2\mathbf{b} - \mathbf{a}$. By triangle inequality and using $\|\mathbf{a} - \mathbf{b}\| < \|\mathbf{b}\|$, we get

$$\|2\mathbf{b} - \mathbf{a}\| \leq \|\mathbf{b}\| + \|\mathbf{b} - \mathbf{a}\| < \|\mathbf{b}\| + \|\mathbf{b}\| = \|2\mathbf{b}\|.$$

Applying Lemma 2.1 one last time we get

$$\|2\mathbf{b} - \mathbf{a}\| < \|2\mathbf{b}\| < \|2\mathbf{b} + \mathbf{a}\|,$$

proving that $\|\mathbf{c}\| > 2\|\mathbf{b}\|$.

This proves that $\|\mathbf{c}\| > 2\|\mathbf{b}\|$, i.e., $\|\mathbf{a}_{i+1}\| > 2\|\mathbf{a}_i\|$. □

By induction, we immediately get that for all $i > 1$, $\|\mathbf{a}_i\| \geq 2^{i-3}\|\mathbf{a}_3\|$. In particular, for any input integer vectors $\mathbf{a}, \mathbf{b}$,

$$2^{k-2} \leq 2^{k-2}\|\mathbf{a}_3\| \leq \|\mathbf{a}_{k+1}\| \leq \|\mathbf{a}\| + \|\mathbf{b}\|.$$

This proves that $k \leq 2 + \log_2(\|\mathbf{a}\| + \|\mathbf{b}\|)$, and therefore the running time of the generalized Gauss algorithm is polynomial in the input size.

THEOREM 2.7 *For any efficiently computable norm $\| \cdot \|$, there exists a polynomial time algorithm that on input two linearly independent integer vectors* $\mathbf{a}, \mathbf{b}$, *outputs a basis* $[\mathbf{a}', \mathbf{b}']$ *for* $\mathcal{L}([\mathbf{a}, \mathbf{b}])$ *such that* $\|\mathbf{a}'\| = \lambda_1$ *and* $\|\mathbf{b}'\| = \lambda_2$. *In particular,* SVP *in dimension* 2 *can be solved in polynomial time.*

2. Approximating SVP in dimension n

In the previous section we described a polynomial time algorithm to find the shortest vector in 2-dimensional lattices. The algorithm was developed in three steps.

1 We first defined a notion of reduced basis for 2-dimensional lattices, and showed that the first vector of a reduced basis is a shortest nonzero vector in the lattice

2 Then we gave an algorithm to compute a reduced basis.

3 Finally, we proved that the algorithm terminates in polynomial time.

In this section we do the same for n-dimensional lattices, although this time we can only prove that the first vector in a reduced basis is within an exponential factor $\gamma(n) = (2/\sqrt{3})^n$ from the shortest. So, the algorithm does not necessarily finds the shortest vector in the lattices, but it computes a lattice vector that is guaranteed to be at most $\gamma(n)\lambda_1$ in length. Although the n-dimensional algorithm can also be adapted to a variety of norms (Lovász and Scarf, 1992), for simplicity here we consider only the Euclidean norm ℓ_2.

2.1 Reduced basis

The definition of reduced basis for 2-dimensional lattices in the case of the ℓ_2 norm can be reformulated as follows.

DEFINITION 2.3 *A basis $B = (\mathbf{b}_1, \mathbf{b}_2)$ is reduced if*

- $\mu_{2,1} \leq \frac{1}{2}$
- $\|\mathbf{b}_1\| \leq \|\mathbf{b}_2\|$

where $\mu_{2,1}$ is the Gram-Schmidt coefficient $\frac{\langle \mathbf{b}_2, \mathbf{b}_1 \rangle}{\langle \mathbf{b}_1, \mathbf{b}_1 \rangle}$.

The reader can easily check that Definition 2.1 and Definition 2.3 are equivalent when the ℓ_2 norm is used. We want to generalize this notion to n-dimensional lattices. Remember the Gram-Schmidt orthogonalization process:

$$\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j<i} \mu_{i,j} \mathbf{b}_j^* \qquad \text{where} \qquad \mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}.$$

We define projection operations π_i from $\mathbb{R}^m$ onto $\operatorname{span}(\mathbf{b}_i^*, \mathbf{b}_{i+1}^*, \ldots, \mathbf{b}_n^*)$ by

$$\pi_i(\mathbf{x}) = \sum_{j=i}^{n} \frac{\langle \mathbf{x}, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*. \tag{2.6}$$

For any vector $\mathbf{x} \in \mathrm{span}(\mathbf{B})$, $\pi_i(\mathbf{x})$ is the component of $\mathbf{x}$ orthogonal to $\mathbf{b}_1, \ldots, \mathbf{b}_i - 1$. In particular, the Gram-Schmidt orthogonalized vectors can be expressed as $\mathbf{b}_i^* = \pi_i(\mathbf{b}_i)$.

We can now define reduced bases for n-dimensional lattices. For reasons that will be clarified in the running time analysis of the basis reduction algorithm, we introduce a real parameter $1/4 < \delta < 1$ and parameterize the definition of reduced bases by δ.

DEFINITION 2.4 *A basis $\mathbf{B} = [\mathbf{b}_1 \ldots \mathbf{b}_n] \in \mathbb{R}^{m \times n}$ is LLL-reduced with parameter δ (δLLL-reduced, for short) if*

1 $|\mu_{i,j}| \leq \frac{1}{2}$ for all $i > j$, where $\mu_{i,j}$ are the Gram-Schmidt coefficients,

2 for any pair of consecutive vectors $\mathbf{b}_i, \mathbf{b}_{i+1}$,

$$\delta \|\pi_i(\mathbf{b}_i)\|^2 \leq \|\pi_i(\mathbf{b}_{i+1})\|^2. \tag{2.7}$$

If $\delta = 1$, the above definition says that for any $i = 1, \ldots, n-1$, the 2-dimensional basis $[\pi_i(\mathbf{b}_i), \pi_i(\mathbf{b}_{i+1})]$ is reduced. In the rest of this subsection, we prove that the first vector in a δLLL-reduced basis is not much longer than λ_1.

LEMMA 2.8 *If $\mathbf{B} = [\mathbf{b}_1 \ldots \mathbf{b}_n] \in \mathbb{R}^{m \times n}$ is an δLLL-reduced basis with $\delta \in (1/4, 1)$, then $\|\mathbf{b}_1\| \leq (2/\sqrt{4\delta - 1})^{n-1}\lambda_1$. In particular, if $\delta = (1/4) + (3/4)^{n/(n-1)}$ then $\|\mathbf{b}_1\| \leq (2/\sqrt{3})^n \lambda_1$.*

Proof: Notice that if the basis is LLL reduced, then for all i

$$\begin{aligned}
\delta \|\mathbf{b}_i^*\|^2 &= \delta \|\pi_i(\mathbf{b}_i)\|^2 \leq \|\pi_i(\mathbf{b}_{i+1})\|^2 \\
&= \|\mathbf{b}_{i+1}^* + \mu_{i+1,i}\mathbf{b}_i^*\|^2 \\
&= \|\mathbf{b}_{i+1}^*\|^2 + \mu_{i+1,i}^2 \|\mathbf{b}_i^*\|^2 \\
&\leq \|\mathbf{b}_{i+1}^*\|^2 + \frac{1}{4}\|\mathbf{b}_i^*\|^2
\end{aligned}$$

and rearranging the terms

$$\left(\delta - \frac{1}{4}\right) \|\mathbf{b}_i^*\|^2 \leq \|\mathbf{b}_{i+1}^*\|^2. \tag{2.8}$$

So, the orthogonalized vectors $\|\mathbf{b}_i^*\|$ can get shorter and shorter as i increases, but not too fast. For example, if $\delta = 3/4$, then each $\|\mathbf{b}_{i+1}^*\|$ is at most $\sqrt{2}$ times as short as $\|\mathbf{b}_i^*\|$. By induction on $i - j$, (2.8) implies that for all $i \geq j$

$$\left(\delta - \frac{1}{4}\right)^{i-j} \|\mathbf{b}_j^*\|^2 \leq \|\mathbf{b}_i^*\|^2, \tag{2.9}$$

and, in particular,

$$\|\mathbf{b}_i^*\| \geq \left(\delta - \frac{1}{4}\right)^{\frac{i-1}{2}} \|\mathbf{b}_1^*\| \geq \left(\delta - \frac{1}{4}\right)^{\frac{n-1}{2}} \|\mathbf{b}_1\|.$$

Using the lower bound $\lambda_1 \geq \min_i \|\mathbf{b}_i^*\|$ from Theorem 1.1, we get

$$\lambda_1 \geq \min_i \|\mathbf{b}_i^*\| \geq \left(\delta - \frac{1}{4}\right)^{\frac{n-1}{2}} \|\mathbf{b}_1\|,$$

proving that $\|b_1\|$ is at most $(\delta - 1/4)^{(1-n)/2}$ times longer than λ_1. Setting $\delta = (1/4) + (3/4)^{n/(n-1)}$, we get $\|\mathbf{b}_1\| \leq (2/\sqrt{3})^n \lambda_1$. □

2.2 The LLL basis reduction algorithm

We know that the first vector of any δLLL-reduced basis (with $\delta = (1/4) + (3/4)^{n/(n-1)}$) is within a factor $(2/\sqrt{3})^n$ from the optimal. In this subsection we describe an algorithm to compute LLL reduced bases. This is the LLL basis reduction algorithm of Lenstra, Lenstra and Lovász (Lenstra et al., 1982). The 2-dimensional lattice reduction algorithm of Gauss (specialized to the ℓ_2 norm) is essentially the following:

1 Reduction Step : $\mathbf{b}_2 := \mathbf{b}_2 - c\mathbf{b}_1$, where $c = \left\lceil \frac{\langle \mathbf{b}_2, \mathbf{b}_1 \rangle}{\langle \mathbf{b}_1, \mathbf{b}_1 \rangle} \right\rfloor$

2 Swap Step : if $\|\mathbf{b}_1\| > \|\mathbf{b}_2\|$ then swap $\mathbf{b}_1 \Leftrightarrow \mathbf{b}_2$.

3 If $(\mathbf{b}_1, \mathbf{b}_2)$ is not reduced repeat.

Notice that after the reduction step, we always have $|\mu_{2,1}| \leq 1/2$. The LLL algorithm follows the same outline, alternating a reduction step, after which $|\mu_{i,j}| \leq 1/2$ for all $i > j$, and a swap step, in which pairs of adjacent vectors are exchanged. The algorithm is shown in Figure 2.4. In the rest of this subsection we explain the algorithm and argue that if it ever terminates, the output is correct.

In the reduction step we want to ensure $|\mu_{i,j}| \leq 1/2$ for all $i > j$. This can be achieved using a simple modification of the Gram-Schmidt orthogonalization process. Namely, we iteratively define the sequence of vectors $\mathbf{b}'_1, \ldots, \mathbf{b}'_n$, where $\mathbf{b}'_1 = \mathbf{b}_1$ and each $\mathbf{b}'_i$ is obtained subtracting appropriate integer multiples of $\mathbf{b}'_j$ (with $j < i$) from $\mathbf{b}_i$. Since $\mathbf{B}'$ is obtained from $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ by a sequence of elementary integer column operations, $\mathbf{B}$ and $\mathbf{B}'$ are equivalent bases. The only other operation performed on $\mathbf{B}'$ (after the reduction step) is rearranging the order of the columns in the swap step. Therefore at the end of each iteration $\mathbf{B}$ is a basis for the input lattice.

Input: Lattice basis $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n] \in \mathbb{Z}^{m \times n}$
Output: An LLL reduced basis for $\mathcal{L}(\mathbf{B})$.

```
(loop): for $i = 1, \ldots, n$
            for $j = i - 1, \ldots, 1$
                $\mathbf{b}_i := \mathbf{b}_i - c_{i,j}\mathbf{b}_j$ where $c_{i,j} = \lfloor \langle \mathbf{b}_i, \mathbf{b}_j \rangle / \langle \mathbf{b}_j, \mathbf{b}_j \rangle \rceil$
        if $\delta \|\pi_i(\mathbf{b}_i)\|^2 > \|\pi_i(\mathbf{b}_{i+1})\|^2$ for some $i$
        then swap $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$ and go to (loop)
        else output $\mathbf{B}$.
```

Figure 2.4. The LLL basis reduction algorithm

Notice that the orthogonalized bases $\mathbf{B}^*$ associated to $\mathbf{B}$ before and after the reduction step are the same, i.e., the transformation $\mathbf{B} \to \mathbf{B}'$ defined above does not change the orthogonalized vectors $\mathbf{b}_i^*$. However, one can easily check that after the transformation $\mathbf{B} \to \mathbf{B}'$ all Gram-Schmidt coefficients $\mu_{i,j}$ (with $i > j$) of the new basis $\mathbf{B}'$ satisfy $|\mu_{i,j}| \leq 1/2$.

After the reduction step has been performed (and condition $|\mu_{i,j}| \leq 1/2$ is satisfied), we check that the second property of LLL reduced bases (2.7) holds for all pairs of consecutive vectors $\mathbf{b}_i, \mathbf{b}_{i+1}$. If for some i

$$\delta \|\pi_i(\mathbf{b}_i)\|^2 > \|\pi_i(\mathbf{b}_{i+1})\|^2 \tag{2.10}$$

we swap $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$. Several pairs might violate property (2.7). Which pair is selected to be swapped does not matter. In the original LLL algorithm i was chosen to be the smallest index such that (2.10), but any selection is equally good. Actually, one can even swaps several disjoint pairs at the same time.

If any two vectors are swapped then the basis is not necessarily length reduced anymore. (I.e., the Gram-Schmidt coefficients might be $|\mu_{i,j}| > 1/2$.) So, we go back to the reduction step and repeat the whole process.

It is clear that if at some point, after the reduction step, no pair of consecutive vectors need to be swapped, then $\mathbf{B}$ is an LLL reduced basis. Moreover, the final matrix $\mathbf{B}$ is equivalent to the original input matrix because it has been obtained from it through a sequence of elementary column operations.

Therefore, if the LLL algorithm ever terminates, the output is an LLL reduced basis. The termination of the algorithm is proved in the next section, together with a polynomial bound on the running time.

2.3 Running time analysis

In order to show that the algorithm runs in polynomial time, we have to prove that the number of iterations is polynomial in the input size, and each iteration takes polynomial time. We first bound the number of iterations.

Bounding number of iterations

The number of iterations performed by the algorithm equals the number of times that any two adjacent vectors are exchanged during the swap step. In order to bound this number we associate a positive integer to the basis $\mathbf{B}$ and prove that each time two vectors are swapped this integer decreases by a factor δ. It follows that the number of iterations is logarithmic in the integer associated to the initial basis. For any $k = 1, \ldots, n$, consider the sublattice $\Lambda_k = \mathcal{L}([\mathbf{b}_1, \ldots, \mathbf{b}_k])$ generated by the first k basis vectors. Since Λ_k is an integer lattice, $\det(\Lambda_k)^2$ is a positive integer. The integer associated to basis $\mathbf{B}$ is

$$d = \prod_{i=1}^{n} \det(\Lambda_k)^2. \tag{2.11}$$

Notice that the reduction step does not affect the value of d because the orthogonalized vectors $\mathbf{b}_i^*$ are not modified by the reduction step and each $\det(\Lambda_k)$ can be expressed as a function of $\|\mathbf{b}_1^*\|, \ldots, \|\mathbf{b}_k^*\|$. We want to prove that when two vectors $\mathbf{b}_i, \mathbf{b}_{i+1}$ are swapped, d decreases by a factor δ. Let d and d' be the value of d before and after the swap. Similarly, let Λ_k and Λ'_k be the lattice generated by $[\mathbf{b}_1, \ldots, \mathbf{b}_k]$ before and after the swap. Recall that vectors $\mathbf{b}_i, \mathbf{b}_{i+1}$ are selected in the swap step only if (2.10) holds. We observe that when vectors $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ are exchanged, $\det(\Lambda_k)$ stays the same for all $k \neq i$. This is because for $k < i$ basis $[\mathbf{b}_1, \ldots, \mathbf{b}_k]$ is not modified, while for $k > i$ we are only changing the order of two vectors in $[\mathbf{b}_1, \ldots, \mathbf{b}_k]$. In either case, the lattice $\Lambda_k = \Lambda'_k$ is unchanged, and $\det(\Lambda_k) = \det(\Lambda'_k)$ is the same before and after the swap. Therefore we have

$$\begin{aligned}
\frac{d'}{d} = \frac{\det(\Lambda'_i)}{\det(\Lambda_i)} &= \frac{\det([\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}, \mathbf{b}_{i+1}])^2}{\det([\mathbf{b}_1, \ldots, \mathbf{b}_i])^2} \\
&= \frac{(\prod_{j=1}^{i-1} \|\mathbf{b}_j^*\|^2) \cdot \|\pi_i(\mathbf{b}_{i+1})\|^2}{\prod_{j=1}^{i} \|\mathbf{b}_j^*\|^2} \\
&= \frac{\|\pi_i(\mathbf{b}_{i+1})\|^2}{\|\pi_i(\mathbf{b}_i)\|^2}
\end{aligned}$$

which, by (2.10) is less than δ. This proves that d decreases at least by a factor δ at each iteration. Let d_0 be the integer associated to the input matrix, and let d_k be the integer associated to $\mathbf{B}$ after k iterations. By induction on k,

$$d_k \leq \delta^k d_0.$$

Since d_k is a positive integer, $\delta^k d_0 \geq \delta_k \geq 1$ and for any $\delta < 1$ it must be

$$k \leq \frac{\ln d_0}{\ln(1/\delta)}.$$

Since d_0 is computable in polynomial time from $\mathbf{B}$, $\ln d_0$ is clearly polynomial in the input size. If δ is set to any fixed constant less than 1, then the $(\ln(1/\delta))^{-1}$ factor increases the number of iterations only by a constant factor. The following lemma shows that one can even set δ to some increasing function of n with $\lim_{n\to\infty}\delta = 1$ and still have a polynomial number of iterations.

LEMMA 2.9 *If* $\delta = (1/4) + (3/4)^{n/(n-1)}$, *then for all* $c > 1$ *and all sufficiently large* n, $(\ln(1/\delta))^{-1} \leq n^c$.

Proof: Let $\delta = (1/4) + (3/4)^{n/(n-1)}$. We want to prove that $(\ln(1/\delta))^{-1}$ is at most n^c, or equivalently $1 - e^{(-(1/n)^c)} \leq 1 - \delta$ for all sufficiently large n. Notice that $1 - \delta = (3/4)(1 - (3/4)^{1/(n-1)})$. We show that the limit (for $n \to \infty$) of

$$\frac{1 - e^{(-(1/n)^c)}}{(3/4)(1 - (3/4)^{1/(n-1)})} \tag{2.12}$$

is strictly less than 1. It follows that for all sufficiently large n, (2.12) is less than 1, and $1 - e^{(-(1/n)^c)} \leq (3/4)(1 - (3/4)^{1/(n-1)})$. In fact, we can prove that the limit of (2.12) is 0. Let $x = 1/(n-1)$ and substitute $n = 1 + 1/x$ in (2.12) to get

$$\frac{1 - e^{-(x/(1+x))^c}}{\frac{3}{4}\left(1 - \left(\frac{3}{4}\right)^x\right)} \tag{2.13}$$

We compute the limit of (2.13) for $x \to 0$. Both the numerator and the denominator tend to 0, so we can use L'Hôpital's rule and obtain

$$\lim_{x\to 0} \frac{1 - e^{-(x/(1+x))^c}}{\frac{3}{4}\left(1 - \left(\frac{3}{4}\right)^x\right)} = \lim_{x\to 0} \frac{e^{-(x/(1+x))^c} \frac{c}{(1+x)^2} \left(\frac{x}{1+x}\right)^{c-1}}{\frac{3}{4}\left(\frac{3}{4}\right)^x \ln(4/3)} = 0. \qquad \square$$

This proves that with polynomially many iterations, the LLL algorithm approximates the shortest vector in a lattice within a factor $(2/\sqrt{3})^n$.

Bounding the running time of each iteration

The number of arithmetic operations performed at each iteration is clearly polynomial. So, in order to prove a polynomial bound on the running time we only need to show that the size of the numbers involved in the entire computation also is bounded by a polynomial in the input size. The LLL algorithm uses rational numbers, so we need to bound both the precision required by these numbers and their magnitude.

From the Gram-Schmidt orthogonalization formulas (1.9a), (1.9b) we know that $\mathbf{b}_i - \mathbf{b}_i^*$ belongs to $\operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$, i.e.,

$$\mathbf{b}_i - \mathbf{b}_i^* = \sum_{j=1}^{i-1} v_{i,j}\mathbf{b}_j \tag{2.14}$$

for some reals numbers $v_{i,j}$. (Notice that these real numbers are different from the Gram-Schmidt coefficients $\mu_{i,j}$. In particular $|v_{i,j}|$ can be bigger than 1/2.) Let $t < i$ and take the scalar product of (2.14) with $\mathbf{b}_t$. Since $\mathbf{b}_i^*$ is orthogonal to $\mathbf{b}_t$, we get

$$\langle \mathbf{b}_i, \mathbf{b}_t \rangle = \sum_{j=1}^{i-1} v_{i,j}\langle \mathbf{b}_j, \mathbf{b}_t \rangle. \tag{2.15}$$

Let $\mathbf{B}_t = [\mathbf{b}_1, \ldots, \mathbf{b}_t]$ and $\mathbf{v}_i = [v_{i,1}, \ldots, v_{i,i-1}]^T$. Combining equations (2.15) for all $t = 1, \ldots, i-1$, we get

$$\mathbf{b}_i^T \mathbf{B}_{i-1} = \mathbf{v}_i^T \mathbf{B}_{i-1}^T \mathbf{B}_{i-1}.$$

So, $\mathbf{v}_i$ is the solution to a system of linear equations with coefficient matrix $(\mathbf{B}_{i-1}^T\mathbf{B}_{i-1})$. Let $d_{i-1} = \det(\mathbf{B}_{i-1}^T\mathbf{B}_{i-1}) = \det(\Lambda_{i-1})^2$, where Λ_{i-1} is the same sublattice defined in the analysis of the number of iterations. By Cramer's rule $d_{i-1}\mathbf{v}_i$ is an integer vector. We use this property to bound the denominators that can occur in the coefficients $\mu_{i,j}$ and orthogonalized vectors $\mathbf{b}_i^*$. Notice that

$$d_{i-1} \cdot \mathbf{b}_i^* = d_{i-1} \cdot \mathbf{b}_i + \sum_{j=1}^{i-1}(d_{i-1}v_{i,j})\mathbf{b}_j$$

is an integer combination of integer vectors. So, all denominators that occur in vector $\mathbf{b}_i^*$ are factors of d_{i-1}. Let us now evaluate the Gram-Schmidt coefficients:

$$\mu_{i,j} \;=\; \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}$$

$$\begin{aligned} &= \frac{d_{j-1}\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{d_{j-1}\|\mathbf{b}_j^*\|^2} \\ &= \frac{\langle \mathbf{b}_i, d_{j-1}\mathbf{b}_j^* \rangle}{d_j}, \end{aligned}$$

because $d_j = \prod_{k=1}^{j} \|\mathbf{b}_k^*\|^2$. So, the denominator of $\mu_{i,j}$ divides d_j. This proves that the denominators of all rational numbers that occur during the computation divide $d = \prod_{i=1}^{n} d_i$. (Notice that this is the same integer defined in (2.11).) But we know from the analysis of the number of iterations that $\log d$ is initially bounded by a polynomial in the input size, and it can only decrease during the execution of the algorithm. So, the denominators of all rational numbers occurring during the computation have polynomial size.

It remains to show that also the magnitude of the numbers is polynomial. We already know that $\mu_{i,j}$ are at most $1/2$ in absolute value. We now bound the length of the vectors. Notice that for all $i > 1$, $\|\mathbf{b}_i^*\| \geq 1/d_{i-1}$ because $d_{i-1}\mathbf{b}_i^*$ is a nonzero integer vector, and $\|\mathbf{b}_1^*\| = \|\mathbf{b}_1\| \geq 1$. Moreover, $d_i = \prod_{j=1}^{i} \|\mathbf{b}_i^*\|^2$. Therefore

$$\|\mathbf{b}_i^*\|^2 = \frac{d_i}{\prod_{j=1}^{i-1} \|\mathbf{b}_j^*\|^2} \leq d_i \prod_{j=1}^{i-2} d_j^2 \leq d^2.$$

Finally,

$$\|\mathbf{b}_i\|^2 = \|\mathbf{b}_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\mathbf{b}_j^*\|^2 \leq d^2 + (n/4)d^2 \leq nd^2.$$

This proves that all quantities that occur during the execution of the LLL algorithm can be represented with polynomially many bits. This completes the proof that the LLL algorithm with $\delta = (1/4)+(3/4)^{n/(n-1)}$ runs in polynomial time.

THEOREM 2.10 *There exists a polynomial time algorithm that on input a basis* $\mathbf{B} \in \mathbb{Z}^{m\times n}$, *outputs an LLL reduced basis for* $\mathcal{L}(\mathbf{B})$ *with parameter* $\delta = (1/4) + (3/4)^{n/(n-1)}$.

Together with Lemma 2.8 this immediately gives a polynomial time approximation algorithm for the shortest vector problem.

THEOREM 2.11 *There exists a polynomial time algorithm that on input an integer basis* $\mathbf{B}$ *outputs a nonzero lattice vector* $\mathbf{x} \in \mathcal{L}(\mathbf{B}) \setminus \{\mathbf{0}\}$ *of length* $\|\mathbf{x}\| \leq (2/\sqrt{3})^n \lambda_1$, *where* n *is the dimension of the lattice.*

```
Input: An integer basis $\mathbf{B} \in \mathbb{Z}^{m\times n}$ and a target vector $\mathbf{t} \in \mathbb{Z}^m$.
Output: A lattice vector $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ such that
$\|\mathbf{t} - \mathbf{x}\| \leq 2(2/\sqrt{3})^n \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$.

run the LLL reduction algorithm on B
let $\mathbf{b} := \mathbf{t}$
for $j = n, \ldots, 1$
    $c_j = \lfloor \langle \mathbf{b}, \mathbf{b}_j \rangle / \langle \mathbf{b}_j, \mathbf{b}_j \rangle \rceil$
    $\mathbf{b} := \mathbf{b} - c_j \mathbf{b}_j$
return $\mathbf{t} - \mathbf{b}$
```

Figure 2.5. The nearest plane algorithm

3. Approximating CVP in dimension n

In this section we show how to use LLL reduced bases to approximately solve the closest vector problem within a factor $2(2/\sqrt{3})^n$. In fact, the algorithm to solve CVP is already contained in the LLL reduction procedure. The idea is the following. Given an LLL reduced basis $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ and a target vector $\mathbf{t}$, run the reduction step of the LLL algorithm on input $[\mathbf{B}, \mathbf{t}]$ as if we wanted to add vector $\mathbf{t}$ to the lattice basis. This way we find a lattice vector $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ such that $\mathbf{t} - \mathbf{x}$ can be expressed as $\mathbf{t}^* + \sum_{i=1}^n c_i \mathbf{b}_i^*$ where $\mathbf{t}^*$ is the component of $\mathbf{t}$ orthogonal to $\operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ and $|c_i| \leq 1/2$ for all $i = 1, \ldots, n$. Interestingly, one can show that the distance of $\mathbf{x}$ from $\mathbf{t}$ is within a factor $2(2/\sqrt{3})^n$ from the optimal.

The algorithm to approximately solve the closest vector problem is given in Figure 2.5. The running time of the algorithm is clearly polynomial. In the next lemma we prove that the algorithm achieves approximation factor $2(2/\sqrt{3})^n$. For reasons that will be apparent in the proof of the lemma, this algorithm is called the *nearest plane* algorithm.

LEMMA 2.12 *When* $\delta = (1/4) + (3/4)^{n/(n-1)}$, *the nearest plane algorithm approximately solves* CVP *within a factor* $\gamma(n) = 2(2/\sqrt{3})^n$.

Proof: Assume that basis $\mathbf{B}$ is already LLL reduced. Assume also, without loss of generality, that target point $\mathbf{t}$ belongs to $\operatorname{span}(\mathbf{B})$. (If not, project $\mathbf{t}$ orthogonally to $\operatorname{span}(\mathbf{B})$, and find the lattice point closest to the projection.) The proof is by induction on the dimension n of the lattice and it uses the fact that if $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ is LLL reduced than also $[\mathbf{b}_1, \ldots, \mathbf{b}_k]$ is LLL reduced for all $k = 1, \ldots, n$. Let $\mathbf{B}^* = [\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*]$ be the Gram-Schmidt orthogonalization of $\mathbf{B}$.

Then, the nearest plane algorithm can be equivalently described as follows. (See Figure 2.6.)

- Find an integer c such that hyperplane $c\mathbf{b}_n^* + \operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$ is as close as possible to $\mathbf{t}$
- Recursively find a lattice point $\mathbf{x}' \in \mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$ approximately closest to the projection $\mathbf{t}'$ of $\mathbf{t} - c\mathbf{b}_n$ on $\operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$.
- Output $\mathbf{x} = \mathbf{x}' + c\mathbf{b}_n$.

Let $\mathbf{y} \in \mathcal{L}(\mathbf{B})$ be the lattice point closest to $\mathbf{t}$. We want to prove that $\|\mathbf{t} - \mathbf{x}\|$ is at most $2(2/\sqrt{3})^n\|\mathbf{t} - \mathbf{y}\|$. There are two cases.

Case 1: If $\|\mathbf{t} - \mathbf{y}\| < \|\mathbf{b}_n^*\|/2$, then $\mathbf{y}$ necessarily belongs to the hyperplane $c\mathbf{b}_n^* + \operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$ because the hyperplanes are $\|\mathbf{b}_n^*\|$ apart from each other, and therefore any other hyperplane is more than $\|\mathbf{b}_n^*\|$ away from $\mathbf{t}$. Therefore $\mathbf{y}' = \mathbf{y} - c\mathbf{b}_n$ is the lattice point in $\mathcal{L}([\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}])$ closest to $\mathbf{t}'$ and by induction hypothesis the recursive call finds a lattice point $\mathbf{x}'$ within distance $2(2/\sqrt{3})^{n-1}\|\mathbf{t}' - \mathbf{y}'\| = 2(2/\sqrt{3})^{n-1}\|\mathbf{t} - \mathbf{y}\|$ from $\mathbf{t} - c\mathbf{b}_n$. It follows that $\mathbf{x}' + c\mathbf{b}_n$ is within distance $2(2/\sqrt{3})^{n-1}\|\mathbf{t} - \mathbf{y}\|$ from $\mathbf{t}$.

Case 2: This time assume $\|\mathbf{t} - \mathbf{y}\| \geq \|\mathbf{b}_n^*\|/2$. We use the properties of LLL reduced basis to show that the lattice vector found by the nearest plane algorithm satisfies $\|\mathbf{t} - \mathbf{x}\| \leq (4/3)^n\|\mathbf{b}_n^*\|^2$ and therefore

$$\|\mathbf{t} - \mathbf{x}\| \leq 2(2/\sqrt{3})^n\|\mathbf{t} - \mathbf{y}\|.$$

We know that $\mathbf{t} - \mathbf{x}' = \sum_{i=1}^n \mu_i \mathbf{b}_i^*$ for some real numbers μ_i satisfying $|\mu_i| \leq 1/2$. Remember that, by (2.9), the orthogonalized vectors in an LLL reduced basis satisfy $\|\mathbf{b}_i^*\| \leq \alpha\|\mathbf{b}_{i+1}^*\|$ where $\alpha = 2/\sqrt{4\delta - 1}$, and, by induction on $n - i$, $\|\mathbf{b}_i^*\| \leq \alpha^{n-i}\|\mathbf{b}_n^*\|$. Therefore,

$$\begin{aligned}
\|\mathbf{t} - \mathbf{x}'\|^2 &= \sum_{i=1}^n \mu_i^2\|\mathbf{b}_i^*\|^2 \\
&\leq \frac{1}{4}\sum_{i=1}^n \alpha^{2(n-i)}\|\mathbf{b}_n^*\|^2 \\
&= \frac{1}{4}\frac{\alpha^{2n} - 1}{\alpha^2 - 1}\|\mathbf{b}_n^*\|^2 \\
&= \frac{\alpha^{2(n-1)}}{4}\left(1 + \frac{1 - \alpha^{2(1-n)}}{\alpha^2 - 1}\right)\|\mathbf{b}_n^*\|^2.
\end{aligned}$$

From $\alpha = 2/\sqrt{4\delta - 1}$ and $\delta = (1/4) + (3/4)^{n/(n-1)}$ we get $\alpha^{n-1} = (2/\sqrt{3})^n$, $\alpha^{2(1-n)} = (3/4)^n$ and $\alpha^2 = (4/3)^{1+1/(n-1)}$, which, substituted

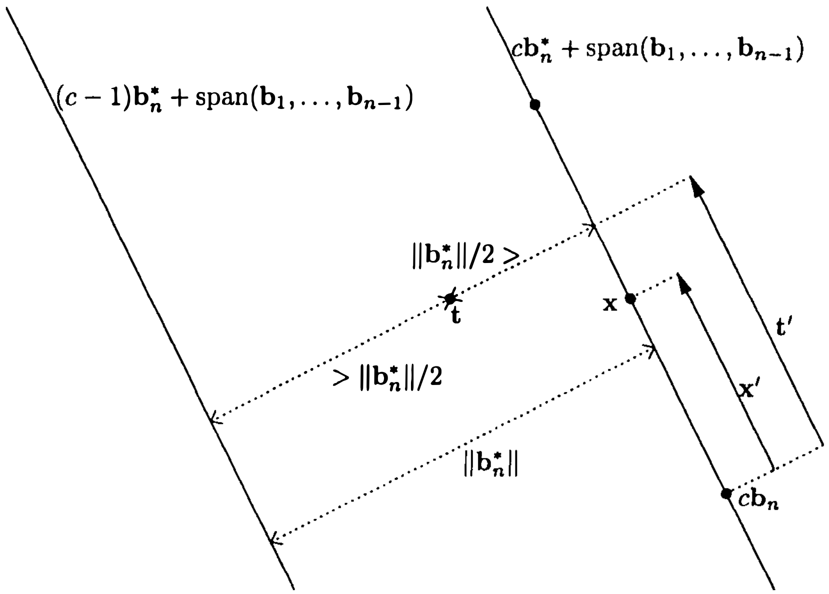

$c\mathbf{b}_n^* + \mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$
$(c-1)\mathbf{b}_n^* + \mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$
$\|\mathbf{b}_n^*\|/2 >$
$\mathbf{t}$
$\mathbf{x}$
$\mathbf{t}'$
$> \|\mathbf{b}_n^*\|/2$
$\mathbf{x}'$
$\|\mathbf{b}_n^*\|$
$c\mathbf{b}_n$

Figure 2.6. The nearest plane algorithm

in the last equation give

$$\|\mathbf{t}-\mathbf{x}\|^2 \leq \frac{1}{4}\left(\frac{4}{3}\right)^n \left(1 + \frac{1-\left(\frac{3}{4}\right)^n}{\left(\frac{4}{3}\right)^{1+\frac{1}{n-1}} - 1}\right) \|\mathbf{b}_n^*\|^2 \leq \left(\frac{4}{3}\right)^n \|\mathbf{b}_n^2\|. \qquad \square$$

4. Notes

A polynomial time algorithm to solve SVP (in the Euclidean norm) for 2-dimensional lattices is already implicit in (Gauss, 1801). A precise worst-case analysis of the algorithm is given in (Vallée, 1991), and the generalization to arbitrary norms presented in Section 1 is due to (Kaib and Schnorr, 1996). Both (Vallée, 1991) and (Kaib and Schnorr, 1996) give almost optimal bounds on the number of iterations, improving the bound of Section 1 by a constant factor.

The algorithm presented in Section 2 is the celebrated LLL basis reduction algorithm (also known as Lovász' reduction algorithm) of (Lenstra et al., 1982). In (Lenstra et al., 1982), the algorithm is formulated and analyzed for the special case of $\delta = \sqrt{3}/2$, and it is almost ubiquitously cited as an algorithm to approximate SVP within a factor $2^{(n-1)/2}$. However, (Lenstra et al., 1982) already observes that δ can be replaced by any constant strictly less than 1, resulting in approximation factors c^{n-1} for any $c < 2/\sqrt{3}$. Proving that the LLL algorithm terminates in polynomial time when $\delta = 1$ is a long standing open problem. Setting $\delta = 1$ in the LLL algorithm would result in approximation factor $(2/\sqrt{3})^{n-1}$. In Section 2 we showed that one can set δ to an increasing function of the rank n, keeping the running time of the algorithm polynomial (in the input size and the rank n of the lattice) and resulting in approximation factor $(2/\sqrt{3})^n$, i.e., essentially the same as setting $\delta = 1$.

Since the invention of the LLL algorithm, the best polynomial time SVP approximation factor has been improved to slightly subexponential functions of the rank. (Schnorr, 1987) presents a hierarchy of reduction algorithms that includes LLL reduction at one end and Korkine-Zolotarev reduction (see Chapter 7) at the other. The algorithm of (Schnorr, 1987), called the *Block Korkine-Zolotarev* (BKZ) reduction algorithm, combines Korkine-Zolotarev reduction with LLL, reducing blocks of consecutive vectors in the sense of Korkine and Zolotarev, and applying an LLL-like algorithm to the blocks. As the size of the blocks increases, the quality of the basis returned improves, but the algorithm also gets slower, going from polynomial time to exponential. Unfortunately, this result of Schnorr is often cited in the literature as an approximation algorithm to within a $2^{\epsilon n}$ factor for any constant $\epsilon > 0$, which corresponds to setting the block size to a large, but fixed, constant. In fact, one can set the block size to a slightly increasing function of the rank, maintaining the running time polynomial, and resulting in a slightly subexponential approximation factor $2^{O(n(\ln\ln n)^2/\ln n)}$ for SVP.

The SVP approximation factor achieved by LLL (and BKZ) reduction is (almost) exponential in the rank of the lattice, still it is quite an achievement because it is a constant for every fixed dimension, independently of the input size. In particular, LLL allowed for the first time to solve SVP exactly in fixed dimension. The dependency of the running time on the dimension is $2^{O(n^2)}$. Better algorithms to solve SVP exactly are given in (Kannan, 1987b), achieving $2^{O(n\log n)}$ running time. Despite the exponential dependency of the running time on the rank, algorithms to solve SVP exactly are of practical relevance because they can be applied to low dimensional sublattices (e.g., the blocks of the BKZ algorithm) to improve the approximation factor of LLL.

Recently, (Ajtai et al., 2001) found a simple and elegant method to *probabilistically* solve SVP exactly in time $2^{O(n)}$. When used in the BKZ algorithm, (Ajtai et al., 2001) allows to reduce the SVP approximation factor from $2^{O(n(\ln\ln n)^2/\ln n)}$ to $2^{O(n\ln\ln n/\ln n)}$, although the output of the algorithm is only guaranteed to be short with high probability.

The nearest plane algorithm of Section 3 is one of two CVP approximation algorithms presented and analyzed in (Babai, 1986). The other algorithm, called the "rounding off" algorithm, simply expresses the target vector $\mathbf{t} = \mathbf{B}\mathbf{x}$ in terms of the LLL reduced lattice basis $\mathbf{B}$, rounds each coordinate of $\mathbf{x}$ to the closest integer $y_i = \lceil x_i \rfloor$, and outputs lattice vector $\mathbf{B}\mathbf{y}$. In (Babai, 1986) it is proved that even this simple rounding procedure result in c^n approximation factors for CVP, although the nearest plane algorithm achieves a better constant c. Using Schnorr's BKZ basis reduction algorithm it is possible to improve the approximation factor for CVP to $2^{O(n(\ln\ln n)^2/\ln n)}$ (Schnorr, 1987; Kannan, 1987a; Schnorr, 1994). Using the probabilistic algorithm of (Ajtai et al., 2001) within Schnorr's BKZ basis reduction, the CVP approximation factor can be further reduced to $2^{O(n\ln\ln n/\ln n)}$. For any fixed dimension, CVP can be solved exactly in polynomial time (Kannan, 1987b), however the dependency of the running time on the rank of the lattice is again $2^{n\ln n}$. For recent refinements and variants of Babai's and Kannan's CVP algorithms the reader is referred to (Blömer, 2000; Klein, 2000).

The existence of (deterministic or probabilistic) SVP (or CVP) approximation algorithms that achieve approximation factors polynomial in the rank n of the lattice is one of the main open problems in the area. In Chapter 8 we will see that the conjectured difficulty of achieving polynomial approximation factors can be used to build provably secure cryptographic functions.

Chapter 3

CLOSEST VECTOR PROBLEM

In Chapter 2 we described algorithms to (approximately) solve SVP and CVP. These algorithms exhibit relatively good performance as far as the running time is concerned. In particular, they terminate within a time bound that is polynomial in the size of the input. However, these algorithms offer very poor guarantees on the quality of the solution returned: the worst-case approximation factor achieved by the best known polynomial time algorithm is essentially exponential in the rank of the lattice. To date no efficient algorithm that provably approximates SVP or CVP within small factors (e.g., factors that are polynomial in the rank of the lattice) is known. In this chapter we start studying lattices from a computational complexity point of view, and, in particular we investigate the hardness of the closest vector problem. We first consider the problem of solving CVP exactly, and prove that this problem is hard for NP. Therefore no efficient algorithm to solve CVP exists, unless P equals NP.

In Chapter 1 we introduced three different formulations of CVP:

- Decisional version: Given integer lattice $\mathbf{B}$, target vector $\mathbf{t}$ and a rational r, determine whether $\text{dist}(\mathbf{t}, \mathbf{B}) \leq r$ or $\text{dist}(\mathbf{t}, \mathbf{B}) > r$.

- Optimization version: Given integer lattice $\mathbf{B}$ and target vector $\mathbf{t}$, compute $\text{dist}(\mathbf{t}, \mathbf{B})$.

- Search version: Given integer lattice $\mathbf{B}$ and target vector $\mathbf{t}$, find a lattice vector $\mathbf{Bx}$ such that $\|\mathbf{Bx} - \mathbf{t}\|$ is minimum.

Each of these problems is easily reduced to the next one as follows. Given a search oracle that finds lattice vectors $\mathbf{Bx}$ closest to $\mathbf{t}$, one can compute the distance of $\mathbf{t}$ from the lattice simply evaluating $\|\mathbf{Bx} -$

D. Micciancio et al., *Complexity of Lattice Problems*

$\mathbf{t}\|$. Similarly, given an optimization oracle to compute $\mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$, one can immediately solve the decisional problem $(\mathbf{B}, \mathbf{t}, r)$ comparing $\mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$ with r.

Interestingly, the search version of CVP is not substantially harder than the optimization or decisional versions, i.e., given an oracle to solve the decision problem associated to CVP, one can solve the search problem in polynomial time. One can try to derive this fact from general principles using the NP-completeness of CVP, but it is interesting to look for a direct reduction. So, before proving the hardness of CVP, in Section 1 we establish the polynomial equivalence of the three versions of this problem. Then, in Section 2 we prove the NP-hardness of (exact) CVP. In Section 3 we study the relationship between SVP and CVP, and prove that in some strong sense the former is not harder than the latter. Finally, we consider variants of CVP, and show that the problem remains NP-hard even if one allows for approximate solutions (in Section 4), or the input lattice can be arbitrarily preprocessed before the target vector is revealed (in Section 5).

1. Decision versus Search

In this section we prove that the search version of CVP can be solved in polynomial time, making a polynomial number of calls to an oracle that solves the decisional CVP problem. In other words, we assume that we have access to a decision oracle $\mathcal{A}$ that on input $(\mathbf{B}, \mathbf{t}, r)$ tells whether $\mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq r$ or not, and show how to use this oracle to efficiently find a lattice point $\mathbf{Bx}$ closest to $\mathbf{t}$, for a given input lattice $\mathbf{B}$ and target vector $\mathbf{t}$.

The idea is to recover the coefficients $x_1, \ldots, x_n$ one bit at a time, but some care is required because the lattice vector closest to $\mathbf{t}$ is not necessarily unique, i.e., there might exist several integer vectors $\mathbf{x}$ such that $\|\mathbf{Bx} - \mathbf{t}\| = \mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$. Therefore, one needs to make sure that the coefficients $x_1, \ldots, x_n$ are all consistent with a single CVP solution $\mathbf{Bx}$. But, let us see first how to recover a single coefficient, say x_1. First we compare the distances of the target $\mathbf{t}$ from the original lattice $\mathcal{L}(\mathbf{B})$ and the sublattice generated by $\mathbf{B}' = [2\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n]$. Clearly, $\mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq \mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}'))$, because $\mathcal{L}(\mathbf{B}')$ is a subset of $\mathcal{L}(\mathbf{B})$. We want to determine if equality holds. The comparison can be easily performed using oracle $\mathcal{A}$ as follows. We start with an upper bound R on the squared distance of $\mathbf{t}$ from the lattice (e.g., one can set $R = \sum_i \|\mathbf{b}\|^2$ to the sum of the squared lengths of all basis vectors) and perform a binary search in $[0, R]$ until we find an integer r such that $r < \mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))^2 \leq r + 1$. Then, we call oracle $\mathcal{A}$ on input

$(\mathbf{B}', \mathbf{t}, \sqrt{r+1})$. If the oracle returns NO, then

$$\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}')) > \sqrt{r+1} \geq \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})).$$

On the other hand, if the oracle returns YES, then we have

$$\sqrt{r} < \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}')) \leq \sqrt{r+1},$$

and therefore $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) = \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}'))$ because both $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))^2$ and $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}'))^2$ are integers. Now notice that if x_1 is even for *some* closest vector $\mathbf{Bx}$, then $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) = \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}'))$, while if x_1 is odd for *all* closest vectors $\mathbf{Bx}$, then $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) < \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}'))$. Therefore, comparing $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$ and $\operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}'))$ allows to determine the parity of x_1 for some closest vector $\mathbf{Bx}$. After the least significant bit of x_1 has been determined, we move to the other bits as follows. We set $\mathbf{t}' = \mathbf{t}$ if x_1 has been determined to be even, and $\mathbf{t}' = \mathbf{t} - \mathbf{b}_1$ otherwise. Then, we repeat the above procedure on lattice $\mathbf{B}'$ and target vector $\mathbf{t}'$ to find the second bit of x_1. Notice that the size of the coefficients x_i can be easily bounded (e.g., using Cramer's rule) and it is polynomial in the size of the input $(\mathbf{B}, \mathbf{t})$. Therefore, after a polynomial number of iterations we will have recovered the first coefficient x_1 entirely. Once we have found coefficient x_1, we move on to the second coefficient, but in order to ensure consistency, we slightly modify the input instance. Instead of using the original lattice $\mathbf{B}$ and target vector $\mathbf{t}$, we consider the sublattice $[\mathbf{b}_2, \ldots, \mathbf{b}_n]$ and target vector $\mathbf{t} - x_1\mathbf{b}_1$. In general, after determining the first k coefficients $x_1, \ldots, x_k$, we consider the sublattice $[\mathbf{b}_{k+1}, \ldots, \mathbf{b}_n]$ and target vector $\mathbf{t} - \sum_{i=1}^{k} x_i\mathbf{b}_i$, and proceed to determine x_{k+1}. Notice that at the end of each iteration, we have a sequence of coefficients $x_1, \ldots, x_k$ such that there exists a solution to the original CVP problem of the form $\sum_{i=1}^{k} x_i\mathbf{b}_i + \mathbf{t}'$ for some vector $\mathbf{t}' \in \mathcal{L}(\mathbf{b}_{k+1}, \ldots, \mathbf{b}_m)$. In particular, after n iterations, lattice vector $\mathbf{Bx} = \sum_{i=1}^{n} x_i\mathbf{b}_i$ is a solution to the CVP problem $(\mathbf{B}, \mathbf{t})$.

This shows that the decisional, optimization and search versions of (exact) CVP are polynomially equivalent, and decisional CVP already captures the hardness of this problem. In the rest of this chapter we concentrate on the decisional version of CVP.

Interestingly, the above reduction does not adapt to the approximation version of CVP, i.e., given an oracle that solves the promise problem GAPCVP$_\gamma$, it is not clear how to efficiently find γ-approximate solutions to the CVP search problem. In Section 4 we will see that GAPCVP$_\gamma$ is NP-hard for any constant γ (or even for certain monotonically increasing functions of the rank). Since CVP (even in its exact version) can be solved in NP (see next section for details), the CVP search problem can be certainly reduced to GAPCVP$_\gamma$ for any constant γ. However, these reductions do not give any insight into the relation between the two versions of the problem. Moreover, they do not work when the approximation factor is sufficiently large (e.g., when γ is polynomial in the rank of the lattice). Giving a simple reduction from the γ-approximate CVP search problem to GAPCVP$_\gamma$ of the kind shown in this section is an interesting open problem.

2. NP-completeness

In this section we show that the decisional version of CVP is NP-complete. We first show that the problem is in NP, i.e., for every instance $(\mathbf{B}, \mathbf{t}, r)$ such that $\mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq r$, there exists a short witness proving that $\mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$ is at most r. The witness is a solution to the search problem, i.e., a lattice point $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ such that $\|\mathbf{x}-\mathbf{t}\| \leq r$. Notice that the size of $\mathbf{x}$ is polynomial because $\mathbf{x}$ is an integer vector and all entries of $\mathbf{x}$ are bounded in absolute value by $\|\mathbf{t}\| + r$. Moreover, the witness can be checked in polynomial time because membership of a vector in a lattice can be decided in polynomial time. We now prove that CVP is hard for NP, i.e., any other problem in NP (or, equivalently, some specific NP-complete problem) can be efficiently reduced to CVP. We give a reduction from the subset sum problem.

DEFINITION 3.1 *The* subset sum *problem* (SS) *is the following. Given $n+1$ integers $(a_1, \ldots, a_n, s)$, find a subset of the a_i's (if one exists) that adds up to s, or equivalently, find coefficients $x_i \in \{0,1\}$ such that $\sum_i a_i x_i = s$. In the decision version of the problem one is given $(a_1, \ldots, a_n, s)$ and must decide if there exist coefficients $x_i \in \{0,1\}$ such that $\sum_i a_i x_i = s$.*

For a proof of the NP-hardness of subset sum see (Garey and Johnson, 1979).

THEOREM 3.1 *For any $p \geq 1$ (including $p = \infty$),* GAPCVP$_1$ *(i.e., the decision problem associated to solving* CVP *exactly) in the ℓ_p norm is* NP*-complete.*

Proof: We already seen that GAPCVP is in NP. We prove that GAPCVP$_1$ is NP-hard by reduction from subset sum. Given a subset sum instance $(a_1, \ldots, a_n, s)$ we define a lattice basis $\mathbf{B}$ with one column $\mathbf{b}_i$ for each subset sum coefficient a_i. Then we associate a target vector $\mathbf{t}$ to the sum s. Vectors $\mathbf{b}_i$ and $\mathbf{t}$ are defined as follows:

$$\mathbf{b}_i = [a_i, \overbrace{0, \ldots, 0}^{i-1}, 2, \overbrace{0, \ldots, 0}^{n-i}]^T \qquad (3.1)$$

$$\mathbf{t} = [\,s, \underbrace{1, \ldots\ldots\ldots\ldots, 1}_{n}]^T \tag{3.2}$$

In matrix notation, the basis $\mathbf{B}$ is easily expressed as

$$\mathbf{B} = \begin{bmatrix} \mathbf{a} \\ 2\mathbf{I}_n \end{bmatrix} \tag{3.3}$$

where $\mathbf{a}$ is the row vector $[a_1, \ldots, a_n]$ and $\mathbf{I}_n$ is the $n \times n$ identity matrix. The output of the reduction is the triple $(\mathbf{B}, \mathbf{t}, \sqrt[p]{n})$. (To be precise, the third element in the output of the reduction should be a rational number. The reader can easily check that $\sqrt[p]{n}$ can be substituted by any rational number t in the interval $[\sqrt[p]{n}, \sqrt[p]{n+1})$, without affecting the correctness of the reduction. For $p = \infty$, this value should be $\lim_{p \to \infty} n^{1/p} = 1$.)

We now prove that the reduction is indeed correct, i.e., if $(\mathbf{a}, s)$ is a YES SS instance, then $(\mathbf{B}, \mathbf{t}, \sqrt[p]{n})$ is a YES CVP instance, while if $(\mathbf{a}, s)$ is a NO SS instance, then $(\mathbf{B}, \mathbf{t}, \sqrt[p]{n})$ is a NO CVP instance. First assume that there exists a solution to the subset sum problem, i.e., there are $x_i \in \{0, 1\}$ such that $\sum_{i=1}^{n} x_i a_i = s$. Then the distance vector is given by

$$\mathbf{Bx} - \mathbf{t} = \begin{bmatrix} \sum_i a_i x_i - s \\ 2x_1 - 1 \\ \vdots \\ 2x_n - 1 \end{bmatrix} \tag{3.4}$$

and the pth power of the ℓ_p distance is

$$\|\mathbf{Bx} - \mathbf{t}\|_p^p = \left|\sum_{i=1}^{n} a_i x_i - s\right|^p + \sum_{i=1}^{n} |2x_i - 1|^p \tag{3.5}$$

which equals n because $\sum_{i=1}^{n} a_i x_i - s = 0$ and $2x_i - 1 = \pm 1$ for all i. This proves that the distance of $\mathbf{t}$ from $\mathcal{L}(\mathbf{B})$ is at most $\sqrt[p]{n}$, and therefore $(\mathbf{B}, \mathbf{t}, \sqrt[p]{n})$ is a YES instance of CVP.

Conversely, assume that $(\mathbf{B}, \mathbf{y}, \sqrt[p]{n})$ is a YES instance, i.e., the distance of $\mathbf{y}$ from the lattice is at most $\sqrt[p]{n}$ and let $\mathbf{x}$ be an integer vector such that $\|\mathbf{Bx} - \mathbf{y}\| \leq \sqrt[p]{n}$. Notice that also in this case (3.5) holds true, and the second summand satisfies $\sum_{i=1}^{n} |2x_i - 1|^p \geq n$ because all $2x_i - 1$ are odd integers. Therefore $\|\mathbf{Bx} - \mathbf{y}\| \leq \sqrt[p]{n}$ is possible only if $\sum_i a_i x_i - s = 0$ and $|2x_i - 1|^p = 1$ for all i. This proves that $\sum_{i=1}^{n} a_i x_i = s$ and $x_i \in \{0, 1\}$ for all i, i.e., $\mathbf{x}$ is a solution to the subset sum problem. □

The reduction from SS to CVP (in the ℓ_2 norm) has obvious connections with the Lagarias-Odlyzko algorithm to solve subset sum (Lagarias and Odlyzko, 1985), or more precisely the improved version of (Coster et al., 1992). The (improved) Lagarias-Odlyzko algorithm works as follows: given a subset sum instance $(\mathbf{a}, s)$, one builds the lattice basis

$$\mathbf{L} = \begin{bmatrix} c \cdot \mathbf{a} & c \cdot s \\ 2\mathbf{I} & \mathbf{1} \end{bmatrix} \tag{3.6}$$

where c is a sufficiently large constant and $\mathbf{1}$ is the all-one column vector. Notice that if $\mathbf{x}$ is a solution to the subset sum problem, then lattice (3.6) has a vector of length $\sqrt{n}$ obtained multiplying the first n columns by $\mathbf{x}$ and the last column by -1. Then, (Lagarias and Odlyzko, 1985) suggests to look for a short(est) nonzero vector in the lattice, e.g., using a lattice basis reduction algorithm. If a short vector $\mathbf{Lx}$ is found, such that $x_{n+1} = -1$ and $x_i \in \{0, 1\}$ for all other $i = 1, \ldots, n$, then $x_1, \ldots, x_n$ is a solution to the subset sum problem.

Notice that this algorithm can be succinctly described as follows:

1 Multiply the subset sum problem by some large constant c to obtain an equivalent subset sum instance $(c \cdot a_1, \ldots, c \cdot a_n, c \cdot s)$

2 Reduce $(c \cdot a_1, \ldots, c \cdot a_n, c \cdot s)$ to a CVP instance $(\mathbf{B}, \mathbf{t})$ using the reduction described in the proof of Theorem 3.1.

3 Solve the closest vector problem $(\mathbf{B}, \mathbf{t}, \sqrt{n})$ using the following heuristics[1]: in order to find the lattice vector closest to $\mathbf{t}$, look for a short vector in the lattice generated by $\mathbf{L} = [\mathbf{B}|\mathbf{t}]$. If this short vector is of the form $\mathbf{Bx} - \mathbf{t}$, then $\mathbf{Bx}$ is a short vector in $\mathcal{L}(\mathbf{B})$ close to $\mathbf{t}$.

The reason the first row of the basis matrix is multiplied by a large constant c is that it is not known how to solve the shortest vector problem exactly, so in practice an approximation algorithm is used (e.g., the LLL algorithm, see Chapter 2). If the first row in the matrix is multiplied by a large constant c, then any moderately short lattice vector must be zero in the first coordinate, and the coefficients $\mathbf{x}$ found by the approximation algorithm must satisfy $\sum a_i x_i = (-x_{n+1})s$. Still, there is no guarantee that the variable x_i are all 0 or 1, and that $x_{n+1} = -1$. Therefore the Lagarias-Odlyzko algorithm does not always find a solution to the subset sum problem. However, if the coefficients $a_1, \ldots, a_n$ are chosen at random among all numbers satisfying certain constraints, proves that the Lagarias-Odlyzko heuristics succeeds with high probability.

The condition on the coefficients can be expressed in terms of a parameter, called density, defined below. Given coefficients $\mathbf{a} = (a_1, \ldots, a_n)$,

[1] In the cryptanalysis literature this heuristics is sometimes referred to as the "embedding technique"

the density $\delta(\mathbf{a})$ of the subset sum problem is defined as the ratio

$$\delta(\mathbf{a}) = \frac{n}{\max_{i=1}^{n} \log a_i}. \tag{3.7}$$

Notice that the density is proportional to the size of the subset sum coefficients a_i, and it equals 1 when $\max_i \lceil \log a_i \rceil$ equals the number n of coefficient. The meaning of the density parameter is better illustrated using the *modular* subset sum problem: given a modulus $M = 2^m$, n coefficients $a_1, \ldots, a_n$ and a target value b, find a 0-1 combination $\sum_i x_i a_i$ which equals b modulo M. Notice that both the reduction to CVP and the Lagarias-Odlyzko heuristics can be easily adapted to the modular subset sum by including one more vector $\mathbf{b}_0 = [c \cdot M, 0, \ldots, 0]^T$ in the lattice basis. In the modular case, the density of the subset sum problem is more conveniently defined as the ratio $\delta = n/m$ between the size of the modulus and the number of coefficients. When $n = m$, the domain and the co-domain of the modular subset sum function $f_{\mathbf{a}}(\mathbf{x}) = \sum_i x_i a_i \pmod{M}$ have the same size 2^n and the density equals $\delta = 1$. When $\delta < 1$, then $f_{\mathbf{a}}$ is injective with high probability (over the choice of coefficients $\mathbf{a}$), while when $\delta > 1$ function $f_{\mathbf{a}}$ is likely to be surjective. In general, if the density is δ, then on the average each point in the co-domain of $f_{\mathbf{a}}$ has 2^{δ} preimages under $f_{\mathbf{a}}$.

Using an oracle that solves SVP exactly, (Coster et al., 1992) shows that it is possible to efficiently solve most subset sum instances with density $\delta < 0.9408$. Given the exponential approximation factor achieved by the LLL algorithm, the Lagarias-Odlyzko algorithm provably solves (with high probability) only subset sum instances with very small density $\delta < 1/O(n)$, i.e., subset sum instances whose coefficients are $O(n^2)$ bits each (Frieze, 1986). With the obvious modifications, the analysis in (Frieze, 1986) also shows that an oracle that approximates SVP within polynomial factors, would result in an efficient algorithm to solve most subset sum instances with density $1/O(\log n)$. Notice that although this density is much higher than $1/O(n)$, it is still an asymptotically vanishing function of the dimension n. Interestingly, even an oracle that solves SVP exactly does not allow to solve most subset sum instances with density arbitrarily close to 1, and (Coster et al., 1992) points out that it seems unlikely that their techniques can be extended to densities higher than 0.9408.

The problem is in the last step of our reformulation of the Lagarias-Odlyzko reduction: while the first two steps correctly reduce *any* subset sum instance $(\mathbf{a}, s)$ to a corresponding CVP instance $(\mathbf{B}, \mathbf{t}, r)$, the last step transforming the CVP instance into SVP instance $([\mathbf{B}|\mathbf{t}], r)$ (in the

ℓ_2 norm) is heuristic, and can only be proved to work in a probabilistic sense when the density δ of the subset sum problem is sufficiently small.

Interestingly, if the max norm $\|\mathbf{x}\|_\infty = \max_i |x_i|$ is used, then any shortest nonzero vector in the lattice (3.6) yields a solution to the original subset sum problem, for any value of the density δ. In fact, this proves that SVP in the ℓ_∞ norm is NP-hard.

THEOREM 3.2 GAPSVP$_1$ *(i.e., the decision problem associated to solving* SVP *exactly) in the ℓ_∞ norm is NP-complete.*

Proof: The problem is clearly in NP because given an instance $(\mathbf{B}, r)$, one can easily guess a short integer vector $\mathbf{y}$ with entries bounded by r in absolute value, and efficiently check that $\mathbf{y}$ belongs to the lattice $\mathcal{L}(\mathbf{B})$. The hardness of GAPSVP$_1$ immediately follows from the reduction from subset sum to CVP given in Theorem 3.1 and the reduction from CVP to SVP in the infinity norm outlined in the discussion of the Lagarias-Odlyzko algorithm. □

As we will see in the following chapters, proving the NP-hardness of SVP in any other norm (and in the Euclidean norm in particular) is a much harder task.

3. SVP is not harder than CVP

In the previous section we proved the NP-hardness of CVP by reduction from subset sum, and we observed how the Lagarias-Odlyzko subset sum algorithm can be described as a reduction from subset sum to CVP followed by a heuristics to solve CVP using an SVP oracle. Reducing CVP to SVP is an interesting problem on its own, as it is widely believed that SVP is not harder than CVP, and many even believe that SVP is strictly easier. Empirical evidence to these beliefs is provided by the gap between known hardness results for both problems. Whereas it is easy to establish the NP-hardness of CVP (see Section 2) and the first proof dates back to (van Emde Boas, 1981), the question of whether SVP (in the ℓ_2 norm) is NP-hard was open for almost two decades, originally conjectured in (van Emde Boas, 1981) and resolved in the affirmative in (Ajtai, 1996), and only for randomized reductions. Furthermore, approximating CVP in n-dimensional lattices is known to be NP-hard (under deterministic reductions) for any constant approximation factor or even some slowly increasing function of the dimension (see Section 4), whereas SVP is only known to be NP-hard under randomized reductions for constant approximation factors below $\sqrt{2}$ (see Chapter 4).

Note that for any ℓ_p norm ($p \geq 1$), SVP can be easily reduced to CVP using the NP-hardness of the latter. However, this general NP-completeness argument produces CVP instances of dimension much bigger than the original SVP problem. An interesting question is whether a direct reduction is possible that preserves the dimension. More importantly, the NP-hardness results do not elucidate on the relation between approximate SVP and approximate CVP when the approximation factor is polynomial (or super-polynomial) in the dimension, or the norm is not an ℓ_p one. We recall that only when the approximation factor is almost exponential ($2^{O(n(\lg\lg n)^2/\lg n)}$) the two problems are known to be solvable in polynomial time. (See Chapter 2.)

In this section we formalize the intuition that SVP is not a harder problem than CVP giving a reduction between the two problems. Notice that the direction of this reduction is opposite to the one implicitly required by the Lagarias-Odlyzko algorithm. Finding an equally simple and general reduction from CVP to SVP (as implicitly required in (Lagarias and Odlyzko, 1985)) is an important open problem.

We show how to reduce the task of finding γ-approximate solutions to SVP to the task of finding γ-approximate solutions to CVP (in the same dimension and rank). The results described in this section hold for any function γ (including finding exact solutions $\gamma = 1$, and polynomial approximations $\gamma(n) = n^c$) for any norm (not necessarily an ℓ_p one), and for the decision, optimization and search versions.

3.1 Deterministic reduction

There are two differences between SVP and CVP. On one hand, SVP asks for a lattice point close to the all-zero vector, while CVP asks for a lattice point close to an arbitrary target vector; on the other hand, SVP disallows the all-zero solution whereas CVP accepts the target vector as an admissible solution (provided it belongs to the lattice). Thus, the two problems are not trivially related. In particular, the obvious "reduction" from SVP to CVP (i.e., $f : \mathbf{B} \mapsto (\mathbf{B}, \mathbf{0})$) does not work since the CVP oracle would always return the all-zero vector. Our aim is to prevent this possibility. The intuitive idea is the following (see Figure 3.1 for a 2-dimensional example). First of all, instead of looking for a lattice point close to the all-zero vector, we look for a lattice point close to some other lattice vector $\mathbf{t} \in \Lambda$ (e.g. $\mathbf{t} = \mathbf{b}_1$). Moreover, to avoid $\mathbf{t}$ being returned as a solution, we run the CVP oracle on a sublattice $\Lambda' \subset \Lambda$ not containing $\mathbf{t}$. The problem is now how to select a sublattice $\Lambda' \subset \Lambda$ without removing all Λ-vectors closest to $\mathbf{t}$. We start with the following observation.

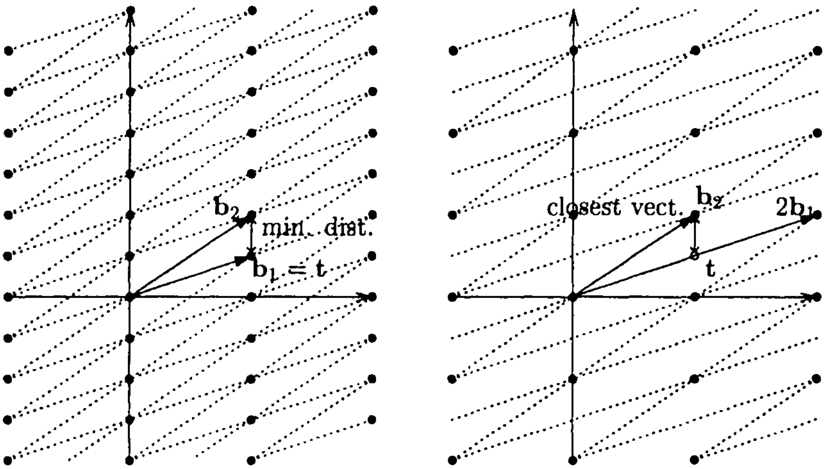


Figure 3.1. Reducing SVP to CVP

PROPOSITION 3.3 *Let* $\mathbf{v} = \sum_{i=1}^{n} c_i \mathbf{b_i}$ *be a shortest nonzero vector in* $\Lambda = \mathcal{L}(\mathbf{B})$. *Then, there exists an* i *such that* c_i *is odd.*

Proof: Let $\mathbf{v} = \sum_{i=1}^{n} c_i \mathbf{b_i}$ be a shortest lattice vector, and assume for contradiction that all c_i's are even. Then $\frac{1}{2} \cdot \mathbf{v} = \sum_{i=1}^{n} \frac{c_i}{2} \mathbf{b_i}$ is also a nonzero lattice vector and it is strictly shorter than $\mathbf{v}$. □

We now show how to reduce the shortest vector problem to the solution of n instances of the closest vector problem.

The reduction. Given a basis $\mathbf{B} = [\mathbf{b_1}, \ldots, \mathbf{b_n}]$, we construct n instances of CVP as follows. The j^{th} instance consists of the basis

$$\mathbf{B^{(j)}} \stackrel{\text{def}}{=} [\mathbf{b_1}, \ldots, \mathbf{b_{j-1}}, 2\mathbf{b_j}, \mathbf{b_{j+1}}, \ldots, \mathbf{b_n}] \tag{3.8}$$

and the target vector $\mathbf{b_j}$. In the search version we use these n instances of CVP in n corresponding queries to the CVP$_\gamma$ oracle, and output the shortest *difference* returned in all these calls (i.e. if $\mathbf{v_j}$ is the vector returned by the j^{th} call on input $(\mathbf{B^{(j)}}, \mathbf{b_j})$, we return the shortest of the vectors $\mathbf{v_1} - \mathbf{b_1}, \ldots, \mathbf{v_n} - \mathbf{b_n}$). In the decision version, we augment these queries by the same parameter r given in the GAPSVP$_\gamma$ instance $(\mathbf{B}, r)$, and return YES if and only if one of the oracle calls was answered by YES.

The validity of the reduction follows from the correspondence between solutions to the input SVP instance and solutions to the CVP instances used in the queries. Specifically:

PROPOSITION 3.4 *Let* $\mathbf{v} = \sum_{i=1}^{n} c_i \mathbf{b_i}$ *be a lattice vector in* $\mathcal{L}(\mathbf{B})$ *such that* c_j *is odd. Then* $\mathbf{u} = \frac{c_j+1}{2}(2\mathbf{b_j}) + \sum_{i \neq j} c_i \mathbf{b_i}$ *is a lattice vector in* $\mathcal{L}(\mathbf{B}^{(\mathbf{j})})$ *and the distance of* $\mathbf{u}$ *from the target* $\mathbf{b_j}$ *equals the length of* $\mathbf{v}$.

Proof: Firstly, note that $\mathbf{u} \in \mathcal{L}(\mathbf{B}^{(\mathbf{j})})$ since $\frac{c_j+1}{2}$ is an integer (as c_j is odd). Secondly, observe that

$$\mathbf{u} - \mathbf{b_j} = \frac{c_j+1}{2} 2\mathbf{b_j} + \sum_{i \neq j} c_i \mathbf{b_i} - \mathbf{b_j} = c_j \mathbf{b_j} + \sum_{i \neq j} c_i \mathbf{b_i} \ = \ \mathbf{v}$$

and the proposition follows. □

PROPOSITION 3.5 *Let* $\mathbf{u} = c_j'(2\mathbf{b_j}) + \sum_{i \neq j} c_i \mathbf{b_i}$ *be a vector in* $\mathcal{L}(\mathbf{B}^{(\mathbf{j})})$. *Then* $\mathbf{v} = (2c_j' - 1)\mathbf{b_j} + \sum_{i \neq j} c_i \mathbf{b_i}$ *is a nonzero lattice vector in* $\mathcal{L}(\mathbf{B})$ *and the length of* $\mathbf{v}$ *equals the distance of* $\mathbf{u}$ *from the target* $\mathbf{b_j}$.

Proof: Firstly, note that $\mathbf{v}$ is nonzero since $2c_j' - 1$ is an odd integer. Secondly, observe that

$$\mathbf{v} = (2c_j' - 1)\mathbf{b_j} + \sum_{i \neq j} c_i \mathbf{b_i} = c_j'(2\mathbf{b_j}) + \sum_{i \neq j} c_i \mathbf{b_i} - \mathbf{b_j} = \mathbf{u} - \mathbf{b_j}. \qquad \square$$

Combining Propositions 3.3 and 3.4, we conclude that one of the CVP-instances has an optimum which is at most the optimum of the given SVP-instance. On the other hand, by Proposition 3.5, the optimum of each of the CVP-instances is bounded from below by the optimum of the given SVP-instance. Details follow.

THEOREM 3.6 *For every function* $\gamma : \mathbb{N} \mapsto \{r \in \mathbb{R} : r \geq 1\}$, SVP_γ *(resp.,* GAPSVP_γ*) is Cook-reducible to* CVP_γ *(resp.,* GAPCVP_γ*). Furthermore, the reduction is non-adaptive*[2], *and all queries maintain the rank of the input instance.*

Proof: We prove the theorem for the decisional (or, more generally, promise) version. The search version is analogous. Let $(\mathbf{B}, r)$ be a GAPSVP_γ instance, and define GAPCVP_γ instances $(\mathbf{B}^{(\mathbf{j})}, \mathbf{b_j}, r)$ for $j = 1, \ldots, n$, where $\mathbf{B}^{(\mathbf{j})}$ is as in (3.8). We want to prove that if $(\mathbf{B}, r)$ is a YES instance, then $(\mathbf{B}^{(\mathbf{j})}, \mathbf{b_j}, r)$ is a YES instance for some $j = 1, \ldots, n$,

[2] A Cook reduction is *non-adaptive* if the queries made to the oracle do not depend on the answers given by the oracle to previous queries, or, equivalently, all the queries are specified in advance before receiving any answer from the oracle.

and if $(\mathbf{B}, r)$ is a NO instance, then $(\mathbf{B}^{(j)}, \mathbf{b}_j, r)$ is a NO instance for all $j = 1, \ldots, n$.

First assume $(\mathbf{B}, r)$ is a YES instance and let $\mathbf{v} = \sum_{i=1}^{n} c_i \mathbf{b}_i$ be the shortest nonzero lattice vector in $\mathcal{L}(\mathbf{B})$. We know that $\|\mathbf{v}\| \leq r$, and (by Proposition 3.3) c_j is odd for some j. Then, the vector $\mathbf{u}$ as defined in Proposition 3.4 belongs to $\mathcal{L}(\mathbf{B}^{(j)})$ and satisfies $\|\mathbf{u} - \mathbf{b}_j\| = \|\mathbf{v}\| \leq r$, proving that $(\mathbf{B}^{(j)}, \mathbf{b}_j, r)$ is a YES instance.

Now assume $(\mathbf{B}^{(j)}, \mathbf{b}_j, r)$ is not a NO instance for some j, i.e., there exists a vector $\mathbf{u}$ in $\mathcal{L}(\mathbf{B}^{(j)})$ such that $\|\mathbf{u} - \mathbf{b}_j\| \leq \gamma(n) \cdot r$. Then, the vector $\mathbf{v}$ defined in Proposition 3.5 is a nonzero lattice vector in $\mathcal{L}(\mathbf{B})$ and satisfies $\|\mathbf{v}\| = \|\mathbf{u} - \mathbf{b}_j\| \leq \gamma(n) \cdot r$, proving that $(\mathbf{B}, r)$ is not a NO instance. □

3.2 Randomized Reduction

In the previous section we showed that any GAPSVP$_\gamma$ instance can be deterministically reduced to solving n instances of GAPCVP$_\gamma$, where n is the rank of the lattices. A natural question is whether it is possible to reduce a GAPSVP problem to a single instance of GAPCVP, i.e., if a Karp reduction exists between the two problems. The proof of Theorem 3.6 suggests that this is possible for randomized reductions. Randomized reductions generalize Karp reductions allowing the mapping function f : GAPSVP$_\gamma \rightarrow$ GAPCVP$_\gamma$ to be computable in polynomial time by a *probabilistic* algorithm. The output of the reduction is only required to be correct with sufficiently high probability. Of special interests are probabilistic reductions in which either YES or NO instances are always mapped correctly. In (Johnson, 1990), these are called

- *Unfaithful random reductions* (UR-reductions for short). These are reductions that always map YES instances to YES instances, and map NO instances to NO instances with probability p. The reduction is called unfaithful because it can produce a YES instance with probability $1 - p$, even if the answer to the original instance was NO. The quantity $1 - p$ (called the soundness error) is required to be at least an inverse polynomial in the input length, i.e., $1 - p \geq 1/n^c$ where n is the input size and c is a constant independent of n.

- *Reverse unfaithful random reductions* (RUR-reductions for short). These are reductions that map YES instances to YES instances with probability p, and always map NO instances to NO instances. The quantity $1 - p$ (called the completeness error) is required to be at least an inverse polynomial in the input length, i.e., $1 - p \geq 1/n^c$ where n is the input size and c is a constant independent of n.

The Cook reduction in the proof of Theorem 3.6 can be transformed into a RUR-reduction as follows. On input $(\mathbf{B}, r)$, choose $j \in \{1, \ldots, n\}$ at random and output $(\mathbf{B}^{(\mathbf{j})}, \mathbf{b_j}, r)$. We notice that YES instances are mapped to YES instances with probability at least $1/n$, and NO instances are always mapped to NO instances. So, this is a RUR-reduction with completeness error $1 - 1/n$. We now show that it is possible to do better than that, and reduce the completeness error to $1/2$.

THEOREM 3.7 *For every function* $\gamma : \mathbb{N} \mapsto \{r \in \mathbb{R} : r \geq 1\}$, *there is a RUR-reduction* SVP_γ *(resp.,* GAPSVP_γ*) to* CVP_γ *(resp.,* GAPCVP_γ*) that has completeness error bounded above by* $1/2$. *Furthermore, the* CVP *instance produced has the same dimension and rank as the original* SVP *problem.*

Proof: Again, we prove the theorem for the decisional version, as the search version is analogous. Let $(\mathbf{B}, r)$ be an SVP instance, where $\mathbf{B} = [\mathbf{b_1}, \ldots, \mathbf{b_n}]$. Output CVP instance $(\mathbf{B}', \mathbf{b_1}, r)$ where $\mathbf{B}' = [\mathbf{b}'_\mathbf{1}, \ldots, \mathbf{b}'_\mathbf{n}]$ is defined as follows. Let $c_1 = 1$ and choose $c_i \in \{0, 1\}$ $(i = 2, \ldots, n)$ uniformly and independently at random. For all i, let $\mathbf{b}'_\mathbf{i} = \mathbf{b_i} + c_i\mathbf{b_1}$. We want to prove that if $(\mathbf{B}, r)$ is a YES instance then $(\mathbf{B}', \mathbf{b_1}, r)$ is a YES instance with probability at least $1/2$, while if $(\mathbf{B}, r)$ is a NO instance then $(\mathbf{B}', \mathbf{b_1}, r)$ is always a NO instance. Notice that $\mathcal{L}(\mathbf{B}')$ is a sublattice of $\mathcal{L}(\mathbf{B})$ and that $\mathbf{b_1}$ is not in $\mathcal{L}(\mathbf{B}')$.

Let us start with the NO case. Assume $(\mathbf{B}', \mathbf{b_1}, r)$ is not a NO instance. By definition, there exists a vector $\mathbf{u}$ in $\mathcal{L}(\mathbf{B}')$ such that $\|\mathbf{u} - \mathbf{b_1}\| \leq \gamma(n) \cdot r$. Since $\mathcal{L}(\mathbf{B}')$ is a sublattice of $\mathcal{L}(\mathbf{B})$ and $\mathbf{b_1}$ is not in $\mathcal{L}(\mathbf{B}')$, $\mathbf{v} = \mathbf{u} - \mathbf{b_1}$ is a nonzero vector in $\mathcal{L}(\mathbf{B})$ of length at most $\gamma(n) \cdot r$, proving that $(\mathbf{B}, r)$ is not a NO instance.

Now assume $(\mathbf{B}, r)$ is a YES instance and let $\mathbf{v} = \sum_{i=1}^{n} x_i\mathbf{b_i}$ be the shortest vector in $\mathcal{L}(\mathbf{B})$. From Proposition 3.4, x_j is odd for some j. Let $\alpha = x_1 + 1 - \sum_{i>1} c_i x_i$. Notice that if x_i is even for all $i > 1$, then x_1 must be odd and α is even. On the other hand, if x_i is odd for some $i > 1$ then α is even with probability $1/2$. In both cases, with probability at least $1/2$, α is even and $\mathbf{u} = \frac{\alpha}{2}\mathbf{b}'_\mathbf{1} + \sum_{i>1} x_i\mathbf{b}'_\mathbf{i}$ is a lattice vector in $\mathcal{L}(\mathbf{B}')$. Finally notice that

$$\begin{aligned}
\mathbf{u} - \mathbf{b_1} &= \left(\alpha\mathbf{b_1} + \sum_{i>1} x_i(\mathbf{b_i} + c_i\mathbf{b_1})\right) - \mathbf{b_1} \\
&= \left(x_1 - \sum_{i>1} c_i x_i\right)\mathbf{b_1} + \sum_{i>1} x_i\mathbf{b_i} + \sum_{i>1} x_i c_i\mathbf{b_1} \quad = \quad \mathbf{v}
\end{aligned}$$

and therefore $\|\mathbf{u} - \mathbf{b_1}\| \leq r$, proving that $(\mathbf{B}', \mathbf{b_1}, r)$ is a YES instance. □

4. Inapproximability of CVP

The NP-hardness of CVP shows that efficient algorithms to solve this problem exactly are unlikely to exist. However, the hardness result presented in Section 2 does not say much about the existence of efficient algorithms to find approximate solutions. In this section we show that even if one allows for solutions which are within a small factor from the optimal, CVP is still hard for NP. As in section 2, the inapproximability results presented in this section hold for any ℓ_p norm. In Subsection 4.1 we prove that for any fixed p, GAPCVP$_\gamma$ is hard for some polylogarithmic function $\gamma(n) = O((\log n)^c)$, where c is a constant independent of n. Then, in Subsection 4.2 we extend the result to *any* polylogarithmic factor. Under the assumption that NP is not contained in QP (quasi polynomial time, i.e., the class of decision problems solvable in time $2^{\log^c n}$), Subsection 4.2 also shows that CVP cannot be approximated within even higher factors $2^{log^{1-\epsilon} n}$ (for any fixed $\epsilon > 0$). These factors, although asymptotically smaller than any polynomial n^c, are bigger than any polylogarithmic function and they are sometime called "quasi-polynomial" approximation factors.

4.1 Polylogarithmic factor

In this section we prove that for any $p \geq 1$, there exists a polylogarithmic function $\gamma = O(\log^{1/p} n)$ such that GAPCVP$_\gamma$ in the ℓ_p norm is NP-hard. The proof is by reduction from the following covering problem.

DEFINITION 3.2 (SET COVER) *For any approximation factor $\gamma \geq 1$, SETCOVER$_\gamma$ is the following promise problem. Instances are pairs $(\mathcal{S}, r)$ where $\mathcal{S} = \{S_1, \ldots, S_n\}$ is a collection of subsets of some set U and r is an integer. (Without loss of generality one can assume that $U = \bigcup_{X \in \mathcal{S}} X$.) Moreover,*

- *$(\mathcal{S}, r)$ is a* YES *instance if $\mathcal{S}$ contains an exact cover of size r, i.e., a sub-collection $\mathcal{S}' \subset \mathcal{S}$ of size $|\mathcal{S}'| = r$ such that $\bigcup_{X \in \mathcal{S}'} X = U$ and the elements of $\mathcal{S}'$ are pairwise disjoint.*
- *$(\mathcal{S}, r)$ is a* NO *instance if $\mathcal{S}$ does not contain any cover of size bounded by γr, i.e., for any sub-collection $\mathcal{S}' \subset \mathcal{S}$ of size $|\mathcal{S}'| \leq \gamma r$, set U is not contained in $\bigcup_{X \in \mathcal{S}'} X$.*

Notice that SETCOVER as defined above is a promise problem, even for $\gamma = 1$. In particular, since YES instances are required to contain an *exact* cover, if $\mathcal{S}$ contains a cover of size r but no exact covers of that size, then $(\mathcal{S}, r)$ is neither a YES nor a NO instance. SETCOVER$_\gamma$ is known to be NP-hard for any constant approximation factor γ (Bellare et al., 1993). In fact, results in (Raz and Safra, 1997) imply that SETCOVER is NP-hard to approximate even within some $O(\log n)$ factor.[3] We reduce SETCOVER$_{O(\log n)}$ to GAPCVP$_{O(\log^{1/p} n)}$ in the ℓ_p norm. It is useful to first reduce SETCOVER$_\gamma$ to a binary variant of GAPCVP defined below.

DEFINITION 3.3 *The promise problem* BINCVP$_\gamma$ *is defined as follows. Instances are triples* $(\mathbf{B}, \mathbf{t}, r)$ *where* $\mathbf{B} \in \mathbb{Z}^{m \times n}$ *is a lattice basis,* $\mathbf{t} \in \mathbb{Z}^m$ *is a vector and* r *is a positive integer such that*

- $(\mathbf{B}, \mathbf{t}, r)$ *is a* YES *instance if there exists a vector* $\mathbf{z} \in \{0, 1\}^n$ *such that* $\mathbf{t} - \mathbf{Bz}$ *is a* 0-1 *vector containing at most* r *ones.*

- $(\mathbf{B}, \mathbf{t}, r)$ *is a* NO *instance if for all* $\mathbf{z} \in \mathbb{Z}^n$ *and all* $w \in \mathbb{Z} \setminus \{0\}$, *vector* $w\mathbf{t} - \mathbf{Bz}$ *has more than* $\gamma(m) \cdot r$ *nonzero entries.*

There are several differences between BINCVP and the standard closest vector problem.

1 First of all, for YES instances the lattice vector close to the target must be one of the vertices of the fundamental parallelepiped associated to the basis. Moreover, the difference between this vector and the target must be a binary vector, so that the distance is uniformly distributed across many different coordinates.

2 For NO instances, we require not only that the target $\mathbf{t}$ be far from the lattice, but also all its nonzero integer multiples $w\mathbf{t}$ should be far away. Moreover, the target (or any of its nonzero multiples) should differ from any lattice points in many coordinates.

3 Finally, for technical reasons, the approximation factor γ is expressed as a function of the dimension of the lattice, instead of its rank. Notice that the lattice in BINCVP is never full rank because otherwise there are integer multiples of $\mathbf{t}$ arbitrarily close to the lattice.

It is clear that BINCVP is just a special case of GAPCVP, and there is a trivial reduction from BINCVP$_\gamma$ to GAPCVP$_{\gamma'}$ in the ℓ_p norm with $\gamma' = \sqrt[p]{\gamma}$. So, proving the hardness of BINCVP immediately implies the hardness of GAPCVP in any ℓ_p norm. We prove the hardness of BINCVP.

[3] Set Cover inapproximability results are usually formulated expressing the approximation factor γ as a function of the size of the underlying set $|U|$. However, the Set Cover instances produced by the reductions always have the property that $|U|$ and $n = |S|$ are polynomially related. Therefore, if Set Cover is NP-hard for some logarithmic function $O(\log |U|)$, then it is also hard for $\gamma = O(\log m)$.

THEOREM 3.8 *The promise problem* BINCVP_γ *is NP-hard for some* $\gamma = O(\log m)$, *where* m *is the dimension of the lattice.*

Proof: The proof is by reduction from SETCOVER_γ. Let $(\mathcal{S}, r)$ be an instance of SETCOVER_γ, and let n and u be the size of $\mathcal{S}$ and $U = \bigcup_{X \in \mathcal{S}} X$ respectively. Without loss of generality, we assume that U is the set $\{1, \ldots, u\}$. Let $S_1, \ldots, S_n$ be the elements of $\mathcal{S}$. Any element $S_i \in \mathcal{S}$ $(i = 1, \ldots, n)$ can be represented as a boolean vector $\mathbf{s}_i \in \{0,1\}^u$ such that the jth coordinate of $\mathbf{s}_i$ equals 1 if and only if $j \in S_i$. We use vectors $\mathbf{s}_i$ to form an $u \times n$ boolean matrix $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_n]$. Let $k = \lceil \gamma r + 1 \rceil$, and define basis $\mathbf{B}$ and target vector $\mathbf{t}$ as follows:

$$\mathbf{B} = \begin{bmatrix} \mathbf{1}_k \otimes \mathbf{S} \\ -\mathbf{I}_n \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} \mathbf{1}_{ku} \\ \mathbf{0}_n \end{bmatrix} \tag{3.9}$$

where $\mathbf{1}_k \otimes \mathbf{S}$ is the $ku \times n$ matrix obtained stacking k copies of $\mathbf{S}$ on top of each other. The output of the reduction is the triple $(\mathbf{B}, \mathbf{t}, r)$. We want to prove that the reduction is correct, i.e., if $(\mathcal{S}, r)$ is a YES instance then $(\mathbf{B}, \mathbf{t}, r)$ is a YES instance, while if $(\mathbf{B}, \mathbf{t}, r)$ is not a NO instance, then $(\mathcal{S}, r)$ is not a NO instance.

First assume $(\mathcal{S}, r)$ is a YES instance, i.e., there exists an exact cover $\mathcal{C} \subset \mathcal{S}$ of size $|\mathcal{C}| = r$. Let $\mathbf{z} \in \{0,1\}^n$ the boolean vector associated to the cover $\mathcal{C}$, i.e., $z_i = 1$ if and only if $S_i \in \mathcal{C}$. Then, since each element of U belongs to one and only one set $S_i \in \mathcal{C}$, we have $\mathbf{Sz} = \mathbf{1}$, and therefore $\mathbf{t} - \mathbf{Bz} = [\mathbf{0}_{uk}^T, \mathbf{z}^T]^T$ is a boolean vector containing exactly r ones. This proves that $(\mathbf{B}, \mathbf{t}, r)$ is a YES instance.

Now assume that $(\mathbf{B}, \mathbf{t}, r)$ is not a NO instance, i.e., there exists a lattice vector $\mathbf{Bz}$ and a nonzero multiple $w\mathbf{t}$ such that $\mathbf{Bz}$ and $w\mathbf{t}$ differ in at most γr coordinates. Let $\mathcal{C}$ be the set of all S_i such that $z_i \neq 0$. We claim that $\mathcal{C}$ is a small cover. First assume for contradiction that $\mathcal{C}$ does not cover U, and let $j \in U$ be an index such that $j \notin \bigcup_{X \in \mathcal{C}} X$. Then the $iu + j$ coordinate of $w\mathbf{t} - \mathbf{Bz}$ equals w for all $i = 0, \ldots, k-1$. This contradicts the assumption that $w\mathbf{t} - \mathbf{Bz}$ has at most $\gamma r < k$ nonzero coordinates, and proves that $\mathcal{C}$ is a cover. Moreover, $\mathcal{C}$ has size less than γr because the last n coordinates of $w\mathbf{t} - \mathbf{Bz}$ equal $\mathbf{z}$, and the size of $\mathcal{C}$ is equal to the number of nonzero entries of vector $\mathbf{z}$. So, $\mathcal{C}$ is a cover of size less than γt, and therefore $(\mathcal{S}, r)$ is not a NO instance.

This proves that BINCVP_γ is NP-hard for some $\gamma = O(\log m)$. Finally, we observe that the dimension of the lattice is $m = ku + n$ is polynomially related to m, therefore $\gamma = O(\log m)$. □

As a corollary, we immediately get the inapproximability of CVP in the ℓ_p norm within some polylogarithmic factor $\gamma = O(\log^{1/p} n)$.

COROLLARY 3.9 *For any $p \geq 1$ there exists a constant c such that the promise problem* GAPCVP$_\gamma$ *is* NP*-hard for $\gamma = c\log^{1/p} n$, where n is the rank of the lattice.*

Proof: The proof is by reduction from BINCVP$_\gamma$. Given instance $(\mathbf{B}, \mathbf{t}, r)$, the reduction outputs $(\mathbf{B}, \mathbf{t}, r')$, where r' is a rational between $\sqrt[p]{r}$ and $\sqrt[p]{r+1}$. □

4.2 Larger factors

For any ℓ_p norm, we proved that CVP is NP-hard to approximate within some polylogarithmic factor $O(\log^{1/p} n)$. In fact, it is possible to prove the hardness of CVP in the ℓ_p norm for *any* polylogarithmic factor $O(\log^c n)$, and beyond. In (Dinur et al., 1998; Dinur et al., 1999), it is proved that CVP is NP-hard to approximate within $2^{O(\log n/\log\log n)}$. Notice that these factors are asymptotically larger than any polylogarithmic function of n, but at the same time they are smaller than any polynomial n^ϵ. Still, hardness results are often interpreted (Arora et al., 1996) as inapproximability within some small polynomial factor n^ϵ. Proofs in (Dinur et al., 1998; Dinur et al., 1999) are rather complex, and they heavily rely on the machinery of probabilistically checkable proofs. In this section we present some general amplification techniques that can be used to achieve almost the same results as in (Dinur et al., 1998), but in a simpler way. In particular, we show that approximating CVP within any polylogarithmic function $\log^c n$ is NP-hard, and approximating CVP within "almost polynomial" functions $2^{O(\log^{1-\epsilon} n)}$ is quasi NP-hard, i.e., no (quasi) polynomial time algorithm exists to approximate CVP within $2^{O(\log^{1-\epsilon} n)}$, unless NP is contained in QP. Here QP is the class of promise problems that can be solved in time $O(2^{\log^c n})$ for some c independent of n.

The idea is to start from a BINCVP$_\gamma$ problem with a certain gap γ between the YES and NO instances, and transform it into another BINCVP instance with a larger gap. We do not know how to perform this amplification operation directly on GAPCVP, and this is one of the reasons we introduced BINCVP as an intermediate problem in the previous subsection.

The amplification technique uses the tensor product operation. Given two vectors $\mathbf{v} \in \mathbb{R}^n$ and $\mathbf{w} \in \mathbb{R}^m$, the tensor product of $\mathbf{v}$ and $\mathbf{t}$ is the $n \cdot m$-dimensional vector $\mathbf{v} \otimes \mathbf{w}$ obtained replacing each entry v_i of $\mathbf{v}$ with a vector $v_i \cdot \mathbf{w}$. More formally, for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$, the $(i-1)m + j$ coordinate of $\mathbf{v} \otimes \mathbf{w}$ is $v_i \cdot w_j$. The tensor product operation is extended to matrices in the obvious way: Given matrix $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_m] \in \mathbb{R}^{m \times n}$ and $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_{m'}] \in \mathbb{R}^{m' \times n'}$, their tensor product is the $(m \cdot m') \times (n \cdot n')$ matrix obtained replacing each entry $v_{i,j}$ in $\mathbf{V}$ with matrix $v_{i,j}\mathbf{W}$. Notice that for any two lattice bases $\mathbf{V}$ and $\mathbf{W}$, the lattice generated by $\mathbf{V} \otimes \mathbf{W}$ depends only on $\mathcal{L}(\mathbf{V})$ and $\mathcal{L}(\mathbf{W})$, and not on the particular choice of basis for the original lattices. So, we can talk of the tensor product of two lattices. It is also important to notice that not every lattice vector in $\mathcal{L}(\mathbf{V} \otimes \mathbf{W})$ can be expressed as $\mathbf{v} \otimes \mathbf{w}$, for $\mathbf{v} \in \mathcal{L}(\mathbf{V})$ and $\mathbf{w} \in \mathcal{L}(\mathbf{W})$.

The amplification technique for BINCVP is described in the following lemma.

LEMMA 3.10 *Let* $(\mathbf{B}, \mathbf{t}, r)$ *be an instance of* BINCVP$_\gamma$, *and define*

$$\mathbf{B}' = [\mathbf{B} \otimes \mathbf{t} | \mathbf{I} \otimes \mathbf{B}], \qquad \mathbf{t}' = \mathbf{t} \otimes \mathbf{t}, \qquad r' = r^2. \tag{3.10}$$

Then, function $f : (\mathbf{B}, \mathbf{t}, r) \mapsto (\mathbf{B}', \mathbf{t}', r')$ *is a reduction from* BINCVP$_\gamma$ *to* BINCVP$_{\gamma^2}$.

Proof: Assume that $(\mathbf{B}, \mathbf{t}, r)$ is a YES instance, i.e., there exists a 0-1 vector $\mathbf{z}$ such that $\mathbf{t} - \mathbf{Bz}$ is a boolean vector with at most r ones. Let

$$\mathbf{z}' = \begin{bmatrix} \mathbf{z} \\ (\mathbf{y} - \mathbf{Bz}) \otimes \mathbf{z}) \end{bmatrix}.$$

Clearly, $\mathbf{z}'$ is a boolean vector too. Moreover,

$$\begin{aligned} \mathbf{t}' - \mathbf{B}'\mathbf{z}' &= \mathbf{t} \otimes \mathbf{t} - (\mathbf{Bz}) \otimes \mathbf{t} - (\mathbf{t} - \mathbf{Bz}) \otimes (\mathbf{Bz}) \\ &= (\mathbf{t} - \mathbf{Bz}) \otimes \mathbf{t} - (\mathbf{t} - \mathbf{Bz}) \otimes (\mathbf{Bz}) \\ &= (\mathbf{t} - \mathbf{Bz}) \otimes (\mathbf{t} - \mathbf{Bz}) \end{aligned}$$

is a 0-1 vector with exactly $r' = r^2$ ones. This proves that $(\mathbf{B}', \mathbf{t}', r')$ is a YES instance.

Now assume that $(\mathbf{B}, \mathbf{t}, r)$ is a NO instance and let $w\mathbf{t}'$ be any nonzero multiple of $\mathbf{t}'$. We want to prove that $w\mathbf{t}'$ differs from any lattice vector in $\mathcal{L}(\mathbf{B}')$ in at least $\gamma^2 r^2$ positions. Let m and n be the dimension and rank of $\mathcal{L}(\mathbf{B})$ and consider a generic lattice vector $\mathbf{B}'\mathbf{x}$ where $\mathbf{x} = [\mathbf{x}_0^T, \mathbf{x}_1^T, \ldots, \mathbf{x}_m^T]^T$ is the concatenation of $m + 1$ n-dimensional integer vectors. Then,

$$w\mathbf{t}' - \mathbf{B}'\mathbf{x} = w\mathbf{t} \otimes \mathbf{t} - (\mathbf{Bx}_0) \otimes \mathbf{t} - \sum_{i=1}^{m} \mathbf{e}_i \otimes \mathbf{Bx}_i \tag{3.11}$$

$$= (w\mathbf{t} - (\mathbf{Bx}_0)) \otimes \mathbf{t} - \begin{bmatrix} \mathbf{Bx}_1 \\ \vdots \\ \mathbf{Bx}_m \end{bmatrix} \tag{3.12}$$

$$= \begin{bmatrix} w_1\mathbf{t} - \mathbf{B}\mathbf{x}_1 \\ \vdots \\ w_m\mathbf{t} - \mathbf{B}\mathbf{x}_m \end{bmatrix} \tag{3.13}$$

$$\tag{3.14}$$

where w_i is the ith coordinate of $(w\mathbf{t} - (\mathbf{B}\mathbf{x}_0))$. Since w is a nonzero integer, then $(w\mathbf{t} - (\mathbf{B}\mathbf{x}_0))$ has more than γr nonzero entries $w_i \neq 0$. Similarly, for each $w_i \neq 0$, vector $w_i\mathbf{t} - \mathbf{B}\mathbf{x}_i$ has more than γr nonzero entries. It follows that the number of nonzero entries in $w\mathbf{t}' - \mathbf{B}'\mathbf{x}$ bigger than $(\gamma r)^2 = \gamma^2 r'$. □

By repeated application of the lemma we obtain the following corollary.

COROLLARY 3.11 *For any constant $c > 0$ and $\gamma(n) = \log^c n$,* BINCVP$_\gamma$ *and* GAPCVP$_\gamma$ *in any ℓ_p norm are NP-hard.*

Proof: We know from Theorem 3.8 that there exists a $c_0 > 0$ such that BINCVP$_{\gamma_0(m)}$ is NP-hard for $\gamma_0(m) = c_0 \log m$. Let c be an arbitrary constant and let $c' > c$. We show that there exists a $c_1 > 0$ such that BINCVP$_{\gamma_0(m)}$ reduces to BINCVP$_{\gamma_1(m')}$ with $\gamma_1(m') = c_1 \log^{c'} m$. Since $\log^c m' < c_1 \log^{c'} m'$ for all sufficiently large m', it follows that BINCVP$_{\gamma'(m')}$ is NP-hard for $\gamma'(m') = \log^c m'$. Finally, we notice that BINCVP$_\gamma$ immediately reduces to GAPCVP$_{\gamma^{1/p}}$ in the ℓ_p norm. So, also GAPCVP$_\gamma$ is NP-hard for any polylogarithmic function $\gamma(m) = \log^c m$.

Let $(\mathbf{B}, \mathbf{t}, r)$ be an instance of BINCVP$_{\gamma_0(m)}$, and let m be the dimension of $\mathcal{L}(\mathbf{B})$. Apply Lemma 3.10 $k = \log_2 c'$ times to $(\mathbf{B}, \mathbf{t}, r)$ to obtain a new instance $(\mathbf{B}', \mathbf{t}', r')$. The dimension of $\mathcal{L}(\mathbf{B}')$ is $m' = m^{2^k} = m^{c'}$, so, the reduction can be computed in polynomial time. Moreover, the gap between YES and NO instances is $(c_0 \log m)^{2^k} = c_0^{c'} \log^{c'} m$. If we express this gap as a function of the dimension of the new lattice we get $\gamma_1(m') = (c_0/c')^{c'} \log^{c'} m' = c_1 \log^{c'} m'$, for $c_1 = (c_0/c')^{c'}$. □

If the reduction from Lemma 3.10 is applied more than a constant number of times, then one obtains quasi NP-hardness results for even larger inapproximability factors, as shown below.

COROLLARY 3.12 *For any constant $\epsilon > 0$, and for any $p \geq 1$,* BINCVP$_\gamma$ *and* GAPCVP$_\gamma$ *in the ℓ_p norm are quasi NP-hard to approximate within $\gamma(m) = 2^{\log^{1-\epsilon} m}$.*

Proof: Similar to the previous corollary, with Lemma 3.10 applied $k = (1/\epsilon) \log\log m$ times. The resulting lattice has dimension $m' = m^{2^k} =$

$2^{(\log m)^{(1+\epsilon)/\epsilon}}$, so it can be computed in quasi polynomial time. Moreover the gap between YES and NO instances is

$$(c_0 \log m)^{(\log m)^{1/\epsilon}} > 2^{(\log m)^{1/\epsilon}}$$

for all sufficiently large m. Substituting $m = 2^{(\log m')^{\epsilon/(1+\epsilon)}}$ we get inapproximability factor $\gamma'(m') = 2^{(\log m')^{1/(1+\epsilon)}}$. □

5. CVP with preprocessing

In this section we consider a different variant of CVP. Instead of allowing for approximate solutions, or considering algorithms that run in quasi-polynomial time, we give unlimited computational power to the CVP solving algorithms, but only in an initial stage of the computation, during which only the lattice is known. In other words, we allow the lattice $\mathcal{L}(\mathbf{B})$ to be arbitrarily preprocessed. Then, we ask for a CVP algorithm that using the preprocessed lattice description, efficiently finds the lattice vector closest to any given target $\mathbf{t}$.

This model is motivated by the applications of lattices in coding theory and cryptography, like vector quantization, communication over band limited channels and encryption. In these applications, the lattice Λ usually represents the code or encryption function, while the target $\mathbf{t}$ is the received message. In this context the closest vector problem corresponds to the decoding or decryption process. Notice that the lattice Λ is usually fixed, and it is known long before transmission occurs. Therefore it makes sense to consider a variant of the closest vector problem in which the lattice is known in advance, and only the target vector $\mathbf{t}$ is specified as input to the problem. Moreover, essentially all known techniques to find (possibly approximate) solutions to the closest vector problem work as follows: (1) first a computationally intensive algorithm is run on the lattice to obtain some information useful for decoding (usually a reduced basis or a trellis); (2) then this information is used to solve CVP using some simple procedure (some form of rounding (Babai, 1986) for methods based on lattice reduction, or the Viterbi algorithm (Forney Jr., 1973) for trellis based decoding). Trellis based decoding is very efficient, provided that a small trellis for the lattice exists. Unfortunately it has been demonstrated that minimal trellis size can grow exponentially with the dimension of the lattice (Forney Jr., 1994; Tarokh and Blake, 1996a; Tarokh and Blake, 1996b). Here we concentrate on methods where the result of preprocessing is always polynomially bounded in the size of the lattice description. Essentially all the preprocessing methods whose output is guaranteed to be small perform some sort of basis reduction, i.e., given any basis for the lattice, they produce a new basis consisting of short vectors. In certain cases the short basis can be computed in polynomial time, resulting in a polynomial time approximation algorithm for the closest vector problem. This is the case for example in the nearest plane algorithm of Chapter 2, LLL reduced bases are used. In other cases it is not known how to efficiently compute the good basis, but once this good basis is found, a much better approximation to the closest vector can be found in polynomial time. For example (Lagarias et al., 1990) shows how to achieve a $O(n^{1.5})$ approximation factor using KZ reduced basis[4] (see also (Kannan, 1987a)). The fastest currently known algorithms to solve the closest vector problem (Banihashemi and Khandani, 1998; Blömer, 2000) also use KZ or dual KZ reduced bases. However, even if the (dual) KZ reduced basis is given, the running time of the algorithm remains exponential in the rank of the lattice.

One natural question is whether it is possible to find optimal solutions to the closest vector problem (with preprocessing) in polynomial time, possibly using a different notion of reduced basis, or more generally using some other form of preprocessing with polynomially bounded output. In other words, we are asking if for every lattice Λ there exists some polynomial amount of information that makes the closest vector problem in Λ easily solvable. Formally, we define the closest vector problem with preprocessing as follows.

DEFINITION 3.4 (CVPP) *The* closest vector problem with preprocessing *asks for a function π (the preprocessing function) and an algorithm $\mathcal{D}$ (the decoding algorithm) with the following properties:*

- *On input a lattice basis $\mathbf{B}$, $\pi(\mathbf{B})$ returns a new description L of the lattice $\mathcal{L}(\mathbf{B})$ whose size is polynomially related to the size of $\mathbf{B}$, i.e. there exists a constant c such that size$(L) <$ size$(\mathbf{B})^c$ for all bases $\mathbf{B}$ and $L = \pi(\mathbf{B})$.*

- *Given L and a target vector $\mathbf{t}$, $\mathcal{D}(L, \mathbf{t})$ computes a lattice point $\mathbf{Bx}$ closest to $\mathbf{t}$. In the decisional version of* CVPP*, $\mathcal{D}$ is also given a distance r, and $\mathcal{D}(L, \mathbf{t}, r)$ decides whether there exists a lattice vector $\mathbf{Bx}$ such that $\|\mathbf{Bx} - \mathbf{t}\| \leq r$.*

As for the standard CVP, the search and decision versions of CVPP are equivalent: any algorithm to solve the search version also solves

[4] The result in (Lagarias et al., 1990) is usually presented as a coNP $O(n^{1.5})$ approximation result for the closest vector problem, meaning that the KZ reduced basis constitutes an NP-proof that the target vector is not too close to the lattice. The $O(n^{1.5})$ approximation factor has been subsequently improved to $O(n)$ in (Banaszczyk, 1993) using techniques from harmonic analysis.

the decision version, and the search version can be reduced to the decision version evaluating the preprocessing function π on all lattices $(2^i \mathbf{b}_j, \mathbf{b}_{j+1}, \ldots, \mathbf{b}_n)$ with i bounded by a polynomial in the size of the input basis, and then using the same reduction as in Section 1.

Notice that no complexity assumption is made on the preprocessing function π (other than the restriction on the size of the output). One may think of π as a preprocessing algorithm with unlimited computational resources. However, only the running of $\mathcal{D}$ is used to measure the complexity of the decoding process, i.e., we say that CVPP is solvable in polynomial time if there exists a function π and a polynomial time algorithm D such that $\mathcal{D}(\pi(\mathbf{B}), \mathbf{t}, r)$ solves the CVP instance $(\mathbf{B}, \mathbf{t}, r)$.

In this section we show that CVPP cannot be solved in polynomial time, under standard complexity assumptions. Namely, we give a reduction from an NP-hard problem H to CVP with the property that any H instance M is mapped to a CVP instance $(\mathbf{B}, \mathbf{t}, r)$ where $\mathbf{B}$ is a lattice basis that depends only on the size of M. It immediately follows that if CVPP has a polynomial time solution, then the NP-hard problem H is solvable in P/poly, and consequently NP $\subseteq$ P/poly.

THEOREM 3.13 CVPP *has no polynomial time solution, unless* NP $\subset$ P/poly. *In particular, there exists a reduction from an* NP*-complete problem* H *to* CVP *such that any* H *instance M is mapped to a* CVP *instance* $(\mathbf{B}, \mathbf{t}, r)$ *where the lattice* $\mathbf{B}$ *depends only on the size of M.*

In fact, Theorem 3.13 immediately follows from the reduction from SS to CVP given in the proof of Theorem 3.1, and the hardness of subset sum with preprocessing from (Lobstein, 1990). Specifically, (Lobstein, 1990) proves that there exists a reduction from an NP-complete problem (3-dimensional matching, 3DM) to subset sum (SS), such that 3DM instance M is mapped to a SS instance $(\mathbf{a}, s)$ where the subset sum coefficients $\mathbf{a}$ depend only on the size of M. Moreover, the NP-hardness proof from Theorem 3.1 reduces a subset sum instance $(\mathbf{a}, s)$ to a CVP instance $(\mathbf{B}, \mathbf{t}, r)$ with the property that the lattice basis $\mathbf{B}$ only depends on the subset sum coefficients $\mathbf{a}$. Therefore, combining the two reductions, we get a reduction from an NP-complete problem to CVP as claimed in Theorem 3.13.

For completeness we now give a direct reduction from an NP-complete problem (X3C, see below) to CVP satisfying the conditions of the theorem.

DEFINITION 3.5 Exact cover by 3-element sets *(*X3C*) is the following problem. Given a finite set M and a collection of three element subsets C, decide if there exists a sub-collection $C' \subseteq C$ such that each element of M is contained in exactly one element of C'.*

Proof: The reduction essentially combines the ideas from (Lobstein, 1990) and the reduction from subset sum presented in Theorem 3.1, but without the complications of using subset sum as an intermediate problem. We give a reduction from X3C to to CVP with the property that the lattice in the CVP instance depends only on the size of the original X3C problem. Let $(M, \mathcal{C})$ be an instance of X3C where M is a set of size $m = |M|$ and $\mathcal{C}$ is a collection of subsets of M, each containing exactly three elements. Assume, without loss of generality, that $M = \{1, \dots, m\}$. We define a lattice that depends only on m, and then show how to encode $\mathcal{C}$ in the target vector. Let $n = \binom{m}{3}$ and consider the $m \times n$ matrix $\mathbf{T} \in \{0,1\}^{m \times n}$ whose columns are all possible m-dimensional binary vectors containing exactly three ones. Notice that the size of $\mathbf{T}$ is polynomial in m. We identify the columns of $\mathbf{T}$ with all the 3-element subsets of M, and X3C instances $\mathcal{C}$ with the corresponding characteristic vectors $\mathbf{c} \in \{0,1\}^n$. The output of the reduction is

$$\mathbf{B} = \begin{bmatrix} \mathbf{T} \\ 2\mathbf{I}_k \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} \mathbf{1}_n \\ \mathbf{c} \end{bmatrix} \qquad r = \|\mathbf{c}\|. \tag{3.15}$$

It is easy to see that X3C instance $\mathbf{c}$ has a solution if and only if CVP instance $(\mathbf{B}, \mathbf{t}, r)$ has a solution. Moreover, the lattice $\mathcal{L}(\mathbf{B})$ depends only on the dimension m of the original 3XC instance. □

6. Notes

The NP-hardness of CVP (in any ℓ_p norm) and SVP (in the ℓ_∞ norm) was originally proved in (van Emde Boas, 1981). The proof presented in Section 2 is from (Micciancio, 2001a). The hardness of approximating CVP within any constant factor, and the amplification technique described in 4 are due to (Arora et al., 1997). Stronger inapproximability results for CVP are given in (Dinur et al., 1998; Dinur et al., 1999), where CVP is proved NP-hard to approximate within $\gamma = 2^{O(\log n / \log\log n)}$. Unfortunately, the proofs in (Dinur et al., 1998; Dinur et al., 1999) heavily rely on the complex machinery of Probabilistically Checkable Proofs (PCP) whose treatment is beyond the scope of this book. Also the results in (Arora et al., 1997) ultimately rely on PCP, but in (Arora et al., 1997) the complexity of PCP techniques is hidden in the proof of inapproximability of Set Cover. Notice that the inapproximability factors (Dinur et al., 1998; Dinur et al., 1999) are asymptotically bigger than any polylogarithmic function $\log^c n$, but at the same time asymptotically smaller than any inverse polynomial $1/n^c$. In (Arora et al., 1996) it is argued that inapproximability within these factors can be interpreted as inapproximability within some small polynomial n^ϵ. Still, proving that CVP is NP-hard to approximate within a factor n^ϵ for some ϵ bounded away from 0 is a major open problem.

The problem of reducing SVP to CVP was explicitly posed by (Babai, 1986). The question asked by Babai is whether one can reduce SVP_γ to $\mathrm{CVP}_{\gamma'}$, for approximation factors such that SVP_γ is not (known to be) solvable in polynomial time (e.g., $\gamma = 2^{\sqrt{n}}$), and $\mathrm{CVP}_{\gamma'}$ is not (known to be) NP-hard (e.g., $\gamma' = n$). This is the question answered, in a very strong sense, in Section 3: there is Cook reduction from SVP to CVP that preserves the approximation factor and the rank of the lattice. The reduction presented in Section 3 is from (Goldreich et al., 1999), which also proves analogous results for coding problems. A problem left open in (Goldreich et al., 1999) is whether it is possible to give a Karp reduction from SVP_γ to CVP_γ. The problem is of interest because it would allow to transform hard SVP distributions (as those built in (Ajtai, 1996) and discussed in Chapter 8) into hard CVP distributions, with potential applications to cryptography. (An explicit construction of a hard CVP distribution has recently been given in (Cai, 2001), but the proof is rather complex. A Karp reduction from SVP to CVP would give a much simpler answer to the same problem, possibly with a different distribution.) The problem of reducing CVP to SVP has also been considered, and it is discussed in Chapter 4.

The hardness proof for the closest vector problem with preprocessing (CVPP) in Section 5 is from (Micciancio, 2001a). Similar results for decoding linear codes and the subset sum problems were already proved in (Bruck and Naor, 1990; Lobstein, 1990). Both CVP and the decoding problem for linear codes are known to be NP-hard not only their exact version, but also when constant (or even $n^{O(1/\log\log n)}$) approximation factors are allowed. A natural question, posed in (Micciancio, 2001a), is if lattice and coding problems with preprocessing are hard to approximate as well. Recently, (Feige and Micciancio, 2001) gave a first answer to this question, showing that CVP in the ℓ_p norm is NP-hard to approximate within any factor $\gamma < \sqrt[p]{5/3}$. Extending this result to arbitrary constants, and possibly almost polynomial factors, is an interesting question.

Chapter 4

SHORTEST VECTOR PROBLEM

In this chapter we study the hardness of approximating the shortest vector problem (SVP). Recall that in SVP one is given a matrix $\mathbf{B} \in \mathbb{Q}^{n\times n}$, and the goal is to find the shortest nonzero vector in the lattice generated by $\mathbf{B}$. In Chapter 3 we have already studied another important algorithmic problem on lattices: the closest vector problem (CVP). In CVP, in addition to the lattice basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$, one is given a target vector $\mathbf{t} \in \mathbb{Q}^n$, and the goal is to find the lattice point in $\mathcal{L}(\mathbf{B})$ closest to $\mathbf{t}$. In Chapter 3 we showed that the NP-hardness of CVP can be easily established by reduction from subset sum (Theorem 3.1), and even approximating CVP within any constant or "almost polynomial" factors is hard for NP. We also observed that the reduction from subset sum to CVP can be easily adapted to prove that SVP in the ℓ_∞ norm is NP-hard (Theorem 3.2). Unfortunately, that simple reduction does not work for any other norm. In this chapter, we investigate the computational complexity of SVP in any ℓ_p norm other than ℓ_∞, with special attention to the Euclidean norm ℓ_2. In the rest of this chapter the ℓ_2 norm is assumed, unless explicitly stated otherwise.

Despite the similarities between SVP and CVP, proving that SVP is NP-hard seems a much harder task, and to date SVP (even in its exact version) is known to be hard for NP only under randomized or non-uniform reductions. Proving such hardness results for SVP is the main goal of this chapter. In particular, we show that for any $p \geq 1$, GAPSVP$_\gamma$ in the ℓ_p norm is NP-hard (under randomized reductions, see Section 3) for any approximation factor $\gamma < \sqrt[p]{2}$. As in Theorem 3.2, the proof is by reduction from (a variant of) CVP. Therefore, the reduction can be considered a "homogenization" process, in which the inhomogeneous problem (CVP) is reduced to its homogeneous counter-

D. Micciancio et al., *Complexity of Lattice Problems*

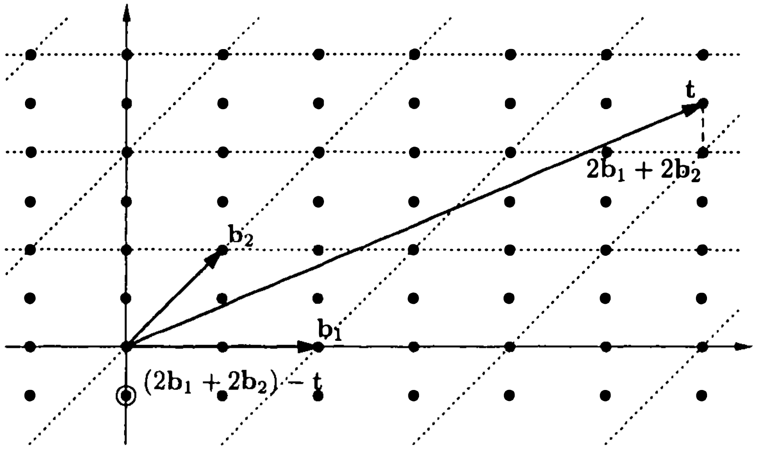


Figure 4.1. The shortest vector $2\mathbf{b}_1 + 2\mathbf{b}_2 - \mathbf{t}$ in the lattice generated by $[\mathbf{b}_1, \mathbf{b}_2, \mathbf{t}]$ corresponds to the lattice vector in $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2)$ closest to $\mathbf{t}$.

part (SVP). This approach is not new in the study of the computational complexity of lattice problems. For example (Kannan, 1987b) reduces approximating CVP within a factor $O(\sqrt{n})$ to computing exact solutions to SVP. In fact, it is not necessary to solve SVP exactly in order to approximate CVP within $O(\sqrt{n})$ factors. In Section 1 we extend Kannan's homogenization technique to show that approximating CVP within some $O(\sqrt{n})$ factor can be reduced to *approximating* SVP within any constant factor $\gamma < 2$. Unfortunately, as we will see in Chapter 9, CVP is not likely to be NP-hard for approximation factors bigger than $O(\sqrt{n}/\log n)$. Therefore, reductions like the one presented in Section 1, are unlikely to be useful to prove the NP-hardness of SVP. In Section 2 we describe a different homogenization technique due to Ajtai and Micciancio and prove that SVP is NP-hard to approximate within any constant factor less than $\sqrt{2}$, by reduction from a variant of CVP.

1. Kannan's homogenization technique

One simple approach to reducing the closest vector problem to the shortest vector problem is as follows. Assume we want to find the point in a lattice $\mathcal{L}(\mathbf{B})$ (approximately) closest to some target vector $\mathbf{t}$. We can look instead for the shortest nonzero vector in the lattice generated by the matrix $[\mathbf{B}|\mathbf{t}]$. If the shortest vector in $\mathcal{L}([\mathbf{B}|\mathbf{t}])$ is of the form $\mathbf{Bx} - \mathbf{t}$ then $\mathbf{Bx}$ is necessarily the lattice vector in $\mathcal{L}(\mathbf{B})$ closest to the target $\mathbf{t}$. See, for example, Figure 4.1. The lattice generated by vectors $\mathbf{b}_1$ and $\mathbf{b}_2$ is the set of intersection points of the grid displayed in the picture.

We want to find the intersection point closest to vector $\mathbf{t}$. Consider the lattice generated by $[\mathbf{b}_1, \mathbf{b}_2, \mathbf{t}]$. This is the set of all black dots in the picture. A shortest nonzero vector in this lattice is given by $2\mathbf{b}_1+2\mathbf{b}_2-\mathbf{t}$. This vector clearly corresponds to the point $2\mathbf{b}_1 + 2\mathbf{b}_2$ in the original lattice $\mathcal{L}(\mathbf{B})$ closest to target $\mathbf{t}$. Unfortunately, if the shortest vector in $\mathcal{L}([\mathbf{B}|\mathbf{t}])$ is not of the form $\mathbf{Bx} \pm \mathbf{t}$ this simple reduction does not work. In the rest of this section we denote by λ the length of the shortest vector in the original lattice $\mathcal{L}(\mathbf{B})$ and by μ the distance of $\mathbf{t}$ from $\mathcal{L}(\mathbf{B})$. There are two different ways in which the above reduction can fail.

- First, if $\lambda \leq \mu$ (i.e., the lattice $\mathcal{L}(\mathbf{B})$ contains vectors as short as the distance of the target $\mathbf{t}$ from the lattice) then the shortest vector in the lattice generated by $[\mathbf{B}|\mathbf{t}]$ can be a vector $\mathbf{Bx}$ from the original lattice. For example, in Figure 4.2, basis vectors $\mathbf{b}_1$ and $\mathbf{b}_2$ generate a lattice with shortest vector $\mathbf{b}_1 - 2\mathbf{b}_2$. Since the distance of target $\mathbf{t}$ from lattice $\mathcal{L}(\mathbf{B})$ is more than $\|\mathbf{b}_1 - 2\mathbf{b}_2\|$, the shortest vector in the lattice generated by $[\mathbf{b}_1, \mathbf{b}_2, \mathbf{t}]$ is still $\mathbf{b}_1 - 2\mathbf{b}_2$.
- Second, even if $\mu < \lambda$, it is still possible that the shortest vector in $\mathcal{L}([\mathbf{B}|\mathbf{t}])$ is of the form $\mathbf{Bx}+w\mathbf{t}$ for some nonzero integer $w \neq \pm 1$. In this case, the shortest vector corresponds to a vector in $\mathcal{L}(\mathbf{B})$ close to a (non-unitary) multiple of $\mathbf{t}$. For example, in Figure 4.3 the lattice point in $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2)$ closest to $\mathbf{t}$ is $\mathbf{b}_1$. However, $3\mathbf{t}$ is strictly closer to the lattice than the original target vector $\mathbf{t}$. So, shortest nonzero vector $2\mathbf{b}_1 + \mathbf{b}_2 - 3\mathbf{t}$ in $\mathcal{L}([\mathbf{b}_1, \mathbf{b}_2, \mathbf{t}])$ yields 'a lattice point $2\mathbf{b}_1 + \mathbf{b}_2 \in \mathcal{L}(\mathbf{B})$ closest to $3\mathbf{t}$.

In both cases a solution vector to the SVP instance $[\mathbf{B}|\mathbf{t}]$ does not seem to help to find lattice vectors in $\mathcal{L}(\mathbf{B})$ close to the target $\mathbf{t}$. A possible way to address these problems is to embed the vectors $[\mathbf{B}|\mathbf{t}]$ in a higher dimensional space and add to $\mathbf{t}$ a component orthogonal to $\mathbf{B}$. In other words, we consider the lattice generated by the matrix

$$\mathbf{B}' = \begin{bmatrix} \mathbf{B} & \mathbf{t} \\ \mathbf{0} & c \end{bmatrix} \tag{4.1}$$

where c is an appropriately chosen rational number. In fact, this method has been used as a heuristics in many cryptanalysis papers and has been reported to be particularly effective when the distance μ is small compared to the minimum distance λ of the lattice. Notice that if $\mathbf{B}$ is a basis for $\mathcal{L}(\mathbf{B})$ then the columns of matrix $\mathbf{B}'$ are linearly independent, i.e., $\mathbf{B}'$ is a basis for $\mathcal{L}(\mathbf{B}')$. (This might not be true of $[\mathbf{B}|\mathbf{t}]$, which generates $\mathcal{L}([\mathbf{B}|\mathbf{y}])$ but is not usually a basis.) If c is sufficiently large, then the shortest vector in the new lattice $\mathcal{L}(\mathbf{B}')$ cannot use the last column

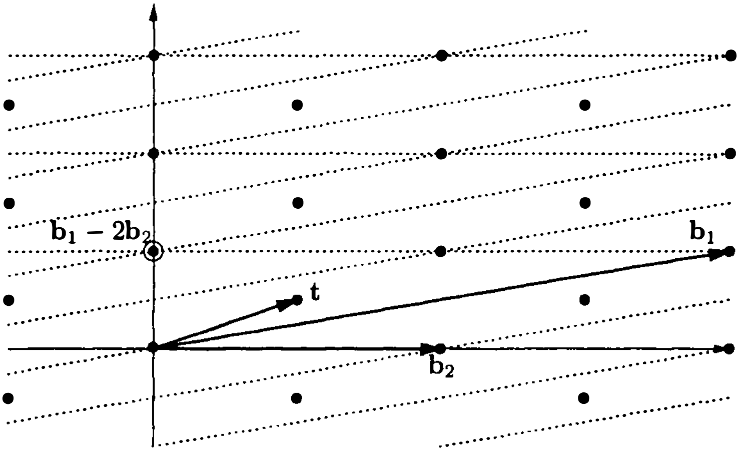


Figure 4.2. The shortest vector $b_1 - 2b_2$ in the lattice generated by $[b_1, b_2, t]$ belongs to the lattice $\mathcal{L}(b_1, b_2)$.

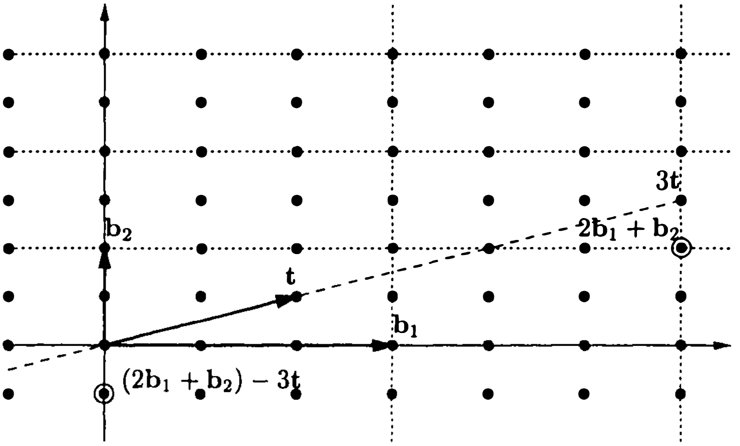


Figure 4.3. The shortest vector $2b_1 + b_2 - 3t$ in the lattice generated by $[b_1, b_2, t]$ correspond to the vector in $\mathcal{L}(b_1, b_2)$ closest to $3t$.

too many times. In particular if $c > \lambda/2$, then the last column can be used at most once. This idea is formalized in (Kannan, 1987b) where c is set to the value 0.51λ. (Notice that λ can be computed applying the SVP oracle to the original lattice.) Still, if $\lambda < \sqrt{\mu^2 + c^2}$ the shortest vector in $\mathcal{L}(\mathbf{B}')$ will be a vector $\mathbf{Bx}$ from the original lattice $\mathcal{L}(\mathbf{B})$, giving no information about the vector in $\mathcal{L}(\mathbf{B})$ closest to $\mathbf{t}$. Kannan then suggests to project $\mathbf{B}$ and $\mathbf{t}$ to the hyperplane orthogonal to the shortest vector of $\mathcal{L}(\mathbf{B})$, recursively solve the lower dimensional CVP problem, and lift the $(n-1)$-dimensional solution to a solution for the original problem. Since λ is not much bigger than μ, the error introduced by the project and lift operations can be bounded as a function of μ. In particular, Kannan shows that CVP can be approximated within a factor $\sqrt{n/2}$ making $O(n)$ calls to an oracle that solves SVP exactly.

In fact, it is not necessary to have an oracle that solves SVP exactly to approximate CVP within $O(\sqrt{n})$ factors, and Kannan's homogenization technique can be extended to approximate CVP within some $O(\sqrt{n})$ factor making $O(n \log n)$ calls to an oracle that approximates SVP within any factor $\gamma < 2$. In the rest of this section we present this improved reduction.

The main idea is to try to set c in (4.1) to some value slightly bigger than $\mu/\sqrt{(2/\gamma)^2-1}$, instead of $\lambda/2$ as in (Kannan, 1987b). This choice for c is motivated by the following lemma.

LEMMA 4.1 *For any $\mu \in [1,2)$, let $\mu > 0$ be the distance of point* $\mathbf{t}$ *from lattice* $\mathcal{L}(\mathbf{B})$ *and let c be a constant strictly bigger than $\mu/\sqrt{(2/\gamma)^2-1}$. If*

$$\mathbf{s} = \begin{bmatrix} \mathbf{B} & \mathbf{t} \\ \mathbf{0}^T & c \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ w \end{bmatrix} = \begin{bmatrix} \mathbf{B}\mathbf{x} + w\mathbf{t} \\ wc \end{bmatrix} \tag{4.2}$$

is a γ-approximate shortest vector in the lattice generated by (4.1), then $|w| \leq 1$.

Proof: First of all, notice that lattice (4.1) contains a vector of length $\sqrt{\mu^2+c^2}$ obtained multiplying the last column of $\mathbf{B}'$ by -1 and the other columns by the coefficients of the lattice vector in $\mathcal{L}(\mathbf{B})$ closest to $\mathbf{t}$. Therefore it must be

$$\|\mathbf{s}\|^2 \leq \gamma^2(\mu^2+c^2). \tag{4.3}$$

We also have

$$\|\mathbf{s}\|^2 = \|\mathbf{B}\mathbf{x}+w\mathbf{t}\|^2 + (wc)^2 \geq (wc)^2. \tag{4.4}$$

Combining (4.3) and (4.4) we get $(wc)^2 \leq \gamma^2(\mu^2+c^2)$. Solving for w and using $c > \mu/\sqrt{(2/\gamma)^2-1}$ we get

$$|w| \leq \gamma\sqrt{\frac{\mu^2}{c^2}+1} < 2. \tag{4.5}$$

Since w is an integer it must be $|w| \leq 1$. □

We use Lemma 4.1 to prove that there is a Cook reduction from approximating CVP within some $O(\sqrt{n})$ factor to approximating SVP within factors less than 2.

THEOREM 4.2 *For any constant approximation factors* $\gamma \in [1,2)$ *and any function*

$$\gamma'(n) > \frac{\sqrt{n}}{\sqrt{(2/\gamma)^2 - 1}},$$

the $\mathrm{CVP}_{\gamma'(n)}$ *search problem (where* n *is the rank of the lattice) is Cook reducible to the* SVP_γ *search problem. Moreover, the number of calls to the* SVP_γ *oracle made by the reduction is* $O(n \log n)$.

Proof: Let γ be any factor in the range $[1,2)$ and let

$$\gamma' = \sqrt{\frac{n(1+\epsilon)}{(2/\gamma)^2 - 1}} \tag{4.6}$$

for some strictly positive, but arbitrarily small, constant $\epsilon \in (0,1]$. We show that given an oracle that approximates SVP within factor γ, one can efficiently approximate CVP within factor γ'. Let $\mathbf{B}$ be a lattice basis of rank n and $\mathbf{t}$ a target vector. We want to find a lattice vector in $\mathcal{L}(\mathbf{B})$ approximately closest to $\mathbf{t}$. To this end, we make calls to the SVP approximation oracle on input basis (4.1), where c is an appropriately chosen constant. Notice that if c is too small then the shortest vector in $\mathcal{L}(\mathbf{B}')$ might use the last column more than once (i.e., with coefficient bigger than 1 in absolute value). On the other hand, if c is too large, then the last column is never used. From Lemma 4.1 we know that it is enough to set c to any value slightly bigger than $\mu/\sqrt{(2/\gamma)^2 - 1}$ in order to make sure that $|w| \leq 1$, e.g.,

$$c \leq \mu\sqrt{\frac{1+\epsilon}{(2/\gamma)^2 - 1}} \tag{4.7}$$

Unfortunately, we don't know the value of μ, so we cannot directly set $c = \mu\sqrt{1+\epsilon}/\sqrt{(2/\gamma)^2 - 1}$. Instead, we first compute a coarse approximation for μ. Using the nearest plane CVP approximation algorithm from Chapter 2, we find in polynomial time an real M such that

$$\mu \leq M \leq 2(2/\sqrt{3})^n \mu < 2^n \mu. \tag{4.8}$$

Then, we consider the monotonically decreasing sequence of constants

$$c_k = \frac{M(\sqrt{1+\epsilon})^{1-k}}{\sqrt{(2/\gamma)^2 - 1}} \tag{4.9}$$

for $k \geq 0$. Notice that if $k = 0$, then c_k equals

$$c_0 = \frac{M\sqrt{1+\epsilon}}{\sqrt{(2/\gamma)^2 - 1}} > \frac{\mu}{\sqrt{(2/\gamma)^2 - 1}} \tag{4.10}$$

and by Lemma 4.1 the short vector $\mathbf{s}$ returned by the SVP_γ oracle on input (4.1) when $c = c_0$ uses the last column $|w| \leq 1$ times. The problem is that c_0 does not necessarily satisfies (4.7). Now consider the value of c_k when k equals

$$K = \lceil 2n/\log_2(1+\epsilon)\rceil \tag{4.11}$$

Using $M \leq 2^n\mu$ we get

$$c_K = \frac{M(\sqrt{1+\epsilon})^{1-K}}{\sqrt{(2/\gamma)^2-1}} \leq \frac{\mu\sqrt{1+\epsilon}}{\sqrt{(2/\gamma)^2-1}}. \tag{4.12}$$

So, c_K satisfies (4.7). The problem this time is that when $c = c_K$ in (4.1) the SVP_γ oracle might return a short vector (4.2) with $|w| > 1$. If this happens, we perform a binary search in $\{0,\ldots,K\}$, and with $\log K = O(\log n)$ calls to the the SVP_γ oracle we find an integer k such that the short vector returned by SVP_γ when $c = c_k$ has $|w| \leq 1$, while the short vector returned by SVP_γ when $c = c_{k+1}$ has $|w| > 1$. We claim that c_k satisfies (4.7). Assume for contradiction that $c_k > \mu\sqrt{1+\epsilon}/\sqrt{(2/\gamma)^2-1}$. Then

$$c_{k+1} = c_k/\sqrt{1+\epsilon} > \mu/\sqrt{(2/\gamma)^2-1}$$

and, by Lemma 4.1, any γ-approximate shortest vector in $\mathcal{L}(\mathbf{B}')$ with $c = c_{k+1}$ must have $|w| \leq 1$. But this is a contradiction because when $c = c_{k+1}$ the SVP_γ oracle returned a short vector with $|w| > 1$.

This shows that $O(\log n)$ calls to the SVP approximation oracle are sufficient to efficiently find a constant c satisfying (4.7), and a short vector $\mathbf{s} \in \mathcal{L}(\mathbf{B}')$ with $|w| \leq 1$. We treat the $|w| = 1$ and $w = 0$ cases separately. First consider the case $w = \pm 1$ and assume, without loss of generality, that $n \geq 3$. (This assumption is justified by the fact that when $n \leq 2$, CVP can be solved exactly in polynomial time.) We claim that $-w\mathbf{Bx}$ is a γ'-approximate solution to CVP instance $(\mathbf{B},\mathbf{t})$. Using (4.3) and (4.4) we get

$$\begin{aligned}
\|\mathbf{t} - (-w\mathbf{Bx})\|^2 &= \|B\mathbf{x} + \mathbf{t}\|^2 \\
&= \|\mathbf{s}\|^2 - (wc)^2 \\
&\leq \gamma^2(\mu^2 + c^2) - c^2 \\
&= \mu^2\gamma^2 + c^2(\gamma^2 - 1) \\
&\leq \mu^2\left(\gamma^2 + \frac{1+\epsilon}{(2/\gamma)^2-1}(\gamma^2-1)\right) \\
&= \mu^2\left(\frac{3+\epsilon(\gamma^2-1)}{(2/\gamma)^2-1}\right)
\end{aligned}$$

$$\begin{aligned}
&\leq \frac{3\mu^2(1+\epsilon)}{(2/\gamma)^2-1} \\
&= (3/n)(\gamma'\mu)^2.
\end{aligned}$$

Therefore, if $n \geq 3$ then $-w\mathbf{Bx}$ is within distance $\gamma'\mu$ from $\mathbf{t}$.

Now consider the case $w = 0$. If $w = 0$, then $\mathbf{s} = \mathbf{Bx}$ is a short nonzero vector in $\mathcal{L}(\mathbf{B})$, and it can be used to find a lattice vector approximately closest to $\mathbf{t}$ as follows. First we bound the length of $\mathbf{s}$. Using (4.3), (4.7) and $\gamma < 2$ we get

$$\begin{aligned}
\|\mathbf{s}\| &\leq \gamma\sqrt{\mu^2+c^2} \\
&\leq \gamma\sqrt{\mu^2+\mu^2\frac{1+\epsilon}{(2/\gamma)^2-1}} \\
&= 2\mu\sqrt{\frac{1+\epsilon\gamma^2/4}{(2/\gamma)^2-1}} \\
&< 2\mu\sqrt{\frac{1+\epsilon}{(2/\gamma)^2-1}} \qquad (4.13)
\end{aligned}$$

Then, we project $\mathbf{B}$ and $\mathbf{t}$ to the orthogonal complement of $\mathbf{s}$ to obtain a matrix $\tilde{\mathbf{B}}$ and vector $\tilde{\mathbf{t}}$ (see Figure 4.4). Let $\mathbf{Bx}$ be a solution to the original CVP problem $(\mathbf{B}, \mathbf{t})$. Notice that the projected vector $\tilde{\mathbf{B}}\mathbf{x}$ is also within distance μ from the target $\tilde{\mathbf{t}}$, and $\mathcal{L}(\tilde{\mathbf{B}})$ has rank $n-1$. So, if we recursively look for an approximate solution to CVP instance $(\widetilde{\mathbf{B}'}, \tilde{\mathbf{t}}')$, we can find a vector $\tilde{\mathbf{u}} = widetilde\mathrm{Bz}$ within distance $\gamma'(n-1)\mu$ from $\tilde{\mathbf{t}}$, i.e.,

$$\|\tilde{\mathbf{u}} - \tilde{\mathbf{t}}\| \leq \mu\sqrt{\frac{(n-1)(1+\epsilon)}{(2/\gamma)^2-1}}. \qquad (4.14)$$

Let $\ell = \{\tilde{\mathbf{u}} + \alpha\mathbf{s} \colon \alpha \in \mathbb{R}\}$ be the set of all points that project to $\mathbf{u}'$, and let $\tilde{\mathbf{u}}$ be the orthogonal projection of $\mathbf{t}$ onto line ℓ. Assume without loss of generality that $\mathbf{Bz}$ is the lattice point on ℓ closest to the projection $\mathbf{u}$. (This can be easily achieved adding an appropriate integer multiple of $\mathbf{s}$ to $\mathbf{Bz}$.) We claim that $\mathbf{Bz}$ is an approximate solution to the original CVP problem. We compute the distance of $\mathbf{Bz}$ from the target $\mathbf{t}$:

$$\|\mathbf{t} - \mathbf{Bz}\|^2 = \|\mathbf{t} - \mathbf{u}\|^2 + \|\mathbf{u} - \mathbf{Bz}\|^2. \qquad (4.15)$$

Using (4.14), we can bound the first term on the right hand side of (4.15) by:

$$\|\mathbf{t} - \mathbf{u}\|^2 = \|\tilde{\mathbf{u}} - \tilde{\mathbf{t}}\|^2 \leq \frac{\mu^2(1+\epsilon)(n-1)}{(2/\gamma)^2-1}. \qquad (4.16)$$

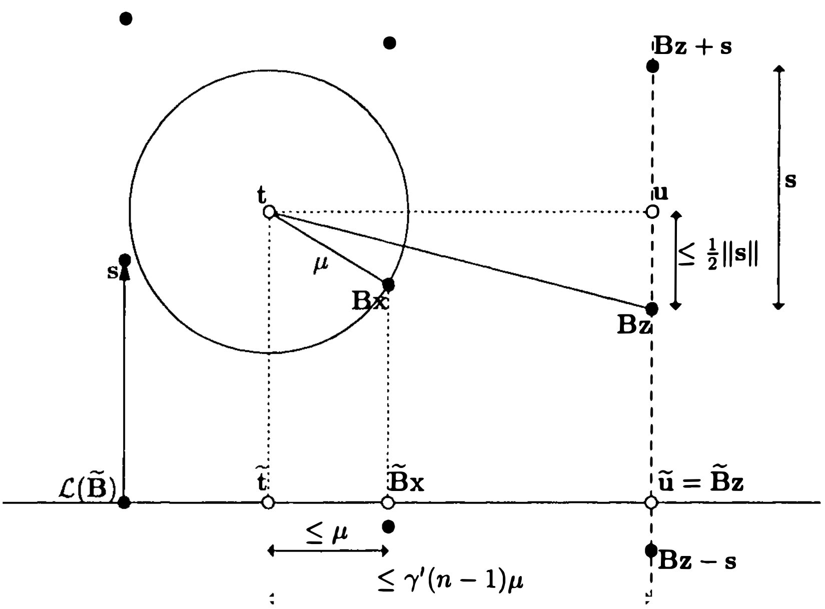


Figure 4.4. Kannan homogenization technique.

For the second term, bound (4.13) on the length of s gives

$$\|\mathbf{u} - \mathbf{Bz}\|^2 \le \left(\frac{1}{2}\|\mathbf{s}\|\right)^2 \le \frac{\mu^2(1+\epsilon)}{(2/\gamma)^2 - 1}. \tag{4.17}$$

Substituting the two bounds (4.16) and (4.17) in (4.15) we get

$$\|\mathbf{t} - \mathbf{Bz}\| \le \sqrt{\frac{(1+\epsilon)n}{(2/\gamma)^2 - 1}}\mu = \gamma'(n)\mu.$$

Therefore, **Bz** is within distance $\gamma'\mu$ from the target **t**. □

If $\gamma = 1$ (i.e., assuming we can solve SVP exactly) the above theorem gives a CVP approximation algorithm with approximation factor arbitrarily close to $\sqrt{n/3}$ (marginally improving Kannan's $\sqrt{n/2}$ factor), but the order of growth is still $O(\sqrt{n})$.

2. The Ajtai-Micciancio embedding

In the previous section we presented a reduction that allows to find $O(\sqrt{n})$-approximate solutions to CVP, given an oracle to find almost exact solutions to SVP. This kind of reductions was considered for a long time a viable way to prove the NP-hardness of SVP. (See for example (Kannan, 1987b) and (Arora et al., 1997).) Unfortunately, as we will see in Chapter 9, CVP is not likely to be NP-hard for approximation factors bigger than $O(\sqrt{n}/\log n)$. Therefore, a more efficient reduction from CVP to SVP is needed in order to obtain NP-hardness results for SVP.

The problem with the reduction presented in Section 1, is that it proceeds by induction on the rank of the lattice. Each time the rank of the lattice is reduced by 1, there is a potential loss in the quality of the final solution. Even if the error at each level of the induction is only a *constant* fraction of the distance of the target from the lattice, adding up n (orthogonal) errors one can get $O(\sqrt{n})$ away from the optimal solution. In this section we present a more efficient reduction technique that allows to embed certain CVP instances in a single instance of SVP. The embedding is more complicated this time, and the dimension of the SVP instance produced is typically much bigger than (but still polynomially related to) the dimension of the original CVP problem. However, the loss in the approximation factor is much smaller. There are a few other important differences between the reduction from Section 1 and the one we are going to present in this section:

- In Section 1 we gave a reduction between the search version of CVP and SVP. In this section we give a reduction between promise problems.

- In Section 1 we reduced SVP from a general CVP instance. In this section, we start from a special version of CVP, namely, the BINCVP problem introduced in Chapter 3.

- The reduction of Section 1 is deterministic. The reduction presented in this section uses a combinatorial gadget for which no efficient deterministic construction is known at the time of this writing. Still, we can give an efficient probabilistic construction, or even a deterministic one if a certain number theoretic conjecture holds true. Therefore, the new reduction implies the NP-hardness of SVP under randomized reductions, or under Karp reductions if the conjecture is correct. Still, the problem of finding an unconditional Karp reduction from an NP-hard problem to SVP remains open.

We now outline the idea underlying the new reduction. Given a lattice basis $\mathbf{B} \in \mathbb{Z}^{m \times n}$ and a target vector $\mathbf{t} \in \mathbb{Z}^m$, we first randomize $\mathbf{B}$ by multiplying it by an integer matrix $\mathbf{T} \in \mathbb{Z}^{n \times k}$ to get a set of vectors $\mathbf{BT} \in \mathbb{Z}^{m \times k}$. The columns of $\mathbf{BT}$ are no longer linearly independent, and each lattice vector has many possible representations as an integer linear combination of them. Then we embed **BT** and **t** in a higher dimensional space using a special lattice **L**, constructed independently from **B**. Lattice **L** has the remarkable property that the distance between any two lattice points (or, equivalently, the length of the shortest nonzero vector in $\mathcal{L}(\mathbf{L})$) is large, but at the same time there is an extremely dense cluster of lattice points all close to a point **s** in span(**L**). In particular, the distance of these lattice points from **s** is smaller (by a constant factor approximately equal to $\sqrt{2}$) than the minimum distance between lattice points in $\mathcal{L}(\mathbf{L})$. The output of the reduction is obtained combining $(\mathbf{L}, \mathbf{s})$ and $(\mathbf{BT}, \mathbf{t})$ in a single matrix

$$\mathbf{B}' = \begin{bmatrix} a\mathbf{BT} & a\mathbf{t} \\ b\mathbf{L} & b\mathbf{s} \end{bmatrix} \tag{4.18}$$

where a and b are appropriate scaling factors. The idea is that if there exists a lattice vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ close to **t**, then we can find a short vector in the new lattice multiplying the last column of $\mathbf{B}'$ by -1 and looking for a lattice point **Lz** close to **s** such that $\mathbf{BTz} = \mathbf{v}$. The lattice vector obtained multiplying matrix (4.18) by $[\mathbf{z}^T, -1]^T$ is short because $\mathbf{BTz} = \mathbf{v}$ is close to **t** and **Lz** is close to **s**. On the other hand, if there are no lattice points in $\mathcal{L}(\mathbf{B})$ close to (any nonzero multiple of) **t**, then the lattice defined by (4.18) has no short vectors because if we use the last column $w \neq 0$ times, then the top part $\mathbf{B}(\mathbf{Tz}) + w\mathbf{t}$ of the lattice vector is long, while if we multiply the last column of $\mathbf{B}'$ by $w = 0$, then the bottom part $\mathbf{Lz} - 0\mathbf{s} = \mathbf{Lz}$ is long.

Notice that the reduction makes crucial use of the special properties of BINCVP. In particular, we use the fact that if the target vector **t** is far from the lattice $\mathcal{L}(\mathbf{B})$, then all (nonzero) multiples of **t** are also far.

As outlined above, the reduction uses three objects $\mathbf{L}, \mathbf{s}$ and **T** satisfying some very special properties. Proving the existence of $\mathbf{L}, \mathbf{s}$ and **T** and giving a polynomial time (possibly randomized) construction for them requires some lattice packing and combinatorial techniques that are developed in Chapter 5 and Chapter 6. Here, we state the properties of $\mathbf{L}, \mathbf{s}$ and **T** and use them to give a reduction from BINCVP to SVP. The properties of $\mathbf{L}, \mathbf{s}$ and **T** are stated with respect to a generic ℓ_p norm, so that we can use the lemma to prove the inapproximability of SVP in the ℓ_p norm for any $p \geq 1$.

LEMMA 4.3 (SPHERE PACKING LEMMA) *For any l_p norm ($p \geq 1$) and constant $\gamma < 2^{1/p}$, there exists a (possibly probabilistic or nonuniform) polynomial time algorithm that on input n outputs (in time polynomial in n) a lattice $\mathbf{L} \in \mathbb{Z}^{k' \times k}$, a vector $\mathbf{s} \in \mathbb{Z}^{k'}$, a matrix $\mathbf{T} \in \mathbb{Z}^{n \times k}$ and a rational number r such that*

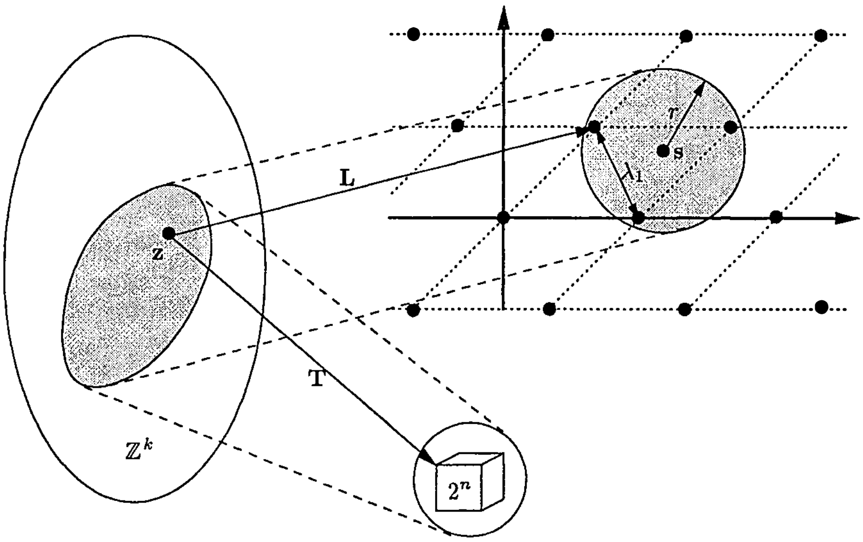


Figure 4.5. The homogenization gadget: lattice **L** has minimum distance γ times the radius of the sphere centered in **s** and all boolean vectors of length n can be expressed as **Tz** for some lattice vector **Lz** inside the sphere.

- $\|\mathbf{Lz}\|_p > \gamma r$ *for all* $\mathbf{z} \in \mathbb{Z}^k \setminus \{\mathbf{0}\}$,
- *(with high probability over the internal randomness of the algorithm) for every boolean vector* $\mathbf{x} \in \{0,1\}^n$ *there exists an integer vector* $\mathbf{z} \in \mathbb{Z}^k$ *such that* $\mathbf{Tz} = \mathbf{x}$ *and* $\|\mathbf{Lz} - \mathbf{s}\|_p < r$.

The various kind of algorithms (deterministic, probabilistic or nonuniform) for which we know how to prove the lemma are discussed in Section 3. The homogenization gadget $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ produced by the these algorithms is illustrated in Figure 4.5. Lattice **L** has minimum distance $\lambda_1 > \gamma r$ and every boolean vector of dimension n can be expressed as **Tz** for some lattice vector **Lz** within distance r from **s**. Notice that this implies that the sphere of radius r around **s** contains at least 2^n lattice points. In Chapter 5 we will prove that, at least for the Euclidean norm ℓ_2, Lemma 4.3 is essentially optimal. In particular, if $p = 2$ and $\gamma \geq \sqrt{2}$, then any sphere of radius r contains at most $2k$ lattice points. This is the ultimate reason why the homogenization technique described in this section does not work for approximation factors beyond $\sqrt{2}$. We will go back to the proof of Lemma 4.3 in Section 3. In the rest of this section we use Lemma 4.3 to prove the hardness of GAPSVP$_\gamma$ for factors $\gamma < \sqrt{2}$.

THEOREM 4.4 *For any $p \geq 1$, given an algorithm satisfying the properties described in Lemma 4.3, one can reduce in polynomial time an* NP-*hard problem to* GAPSVP$_\gamma$ *in the ℓ_p for any constant approximation factor $\gamma < \sqrt[p]{2}$.*

Proof: Fix an l_p norm and a constant $\gamma < \sqrt[p]{2}$. Let $\tilde{\gamma}$ be any constant between γ and $\sqrt[p]{2}$, and let

$$\hat{\gamma} = \frac{2^p}{\left(\frac{1}{\gamma}\right)^p - \left(\frac{1}{\tilde{\gamma}}\right)^p}.$$

We reduce BINCVP$_{\hat{\gamma}}$ to GAPSVP$_\gamma$. Notice that $\hat{\gamma}$ is a constant independent of n, so, by Corollary 3.11, BINCVP$_{\hat{\gamma}}$ is NP-hard. We remark that Corollary 3.11 shows that BINCVP$_{\hat{\gamma}}$ is NP-hard not only for constant approximation factors $\hat{\gamma}$, but also for some monotonically increasing function $\hat{\gamma}(n)$ of the rank. Using these stronger inapproximability results one can show that GAPSVP$_\gamma$ is hard to approximate within some factor $\gamma(n) < \sqrt[p]{2}$ such that $\lim_{n\to\infty} \gamma(n) = \sqrt[p]{2}$. This is only marginally better than showing hardness for any constant $\hat{\gamma} < \sqrt[p]{2}$. So, in order to keep the presentation simpler, we consider $\hat{\gamma}$ as fixed.

Let $(\mathbf{B}, \mathbf{t}, d)$ be an instance of BINCVP$_{\hat{\gamma}}$ where $\mathbf{B} \in \mathbb{Z}^{m\times n}$ and $\mathbf{t} \in \mathbb{Z}^m$. Run the algorithm from Lemma 4.3 to obtain a lattice $\mathbf{L} \in \mathbb{Z}^{k'\times k}$, a vector $\mathbf{s} \in \mathbb{Z}^{k'}$, a matrix $\mathbf{T} \in \mathbb{Z}^{n\times k}$ and a rational number r such that

- $\|\mathbf{Lz}\|_p > \tilde{\gamma} r$ for all $\mathbf{z} \in \mathbb{Z}^k \setminus \{\mathbf{0}\}$,
- (with high probability) for all vectors $\mathbf{x} \in \{0,1\}^n$ there exists a $\mathbf{z} \in \mathbb{Z}^k$ such that $\mathbf{Tz} = \mathbf{x}$ and $\|\mathbf{Lz} - \mathbf{s}\|_p < r$.

Let a and b two integers such that

$$\frac{r}{2\sqrt[p]{d}}\sqrt[p]{\left(\frac{\tilde{\gamma}}{\gamma}\right)^p - 1} < \frac{a}{b} < \frac{r}{\sqrt[p]{d}}\sqrt[p]{\left(\frac{\tilde{\gamma}}{\gamma}\right)^p - 1} \qquad (4.19)$$

and define the lattice

$$\mathbf{B}' = \left[\begin{array}{c|c} a\mathbf{BT} & a\mathbf{t} \\ b\mathbf{L} & b\mathbf{s} \end{array}\right].$$

Notice that from the upper bound in (4.19) we get

$$\begin{aligned} a^p d + b^p r^p &< b^p r^p \left(\left(\frac{\tilde{\gamma}}{\gamma}\right)^p - 1\right) + b^p r^p \\ &= b^p r^p \left(\frac{\tilde{\gamma}}{\gamma}\right)^p, \end{aligned}$$

so, we can find a rational number d' in the interval

$$\sqrt[p]{a^p d + b^p r^p} < d' < br\left(\frac{\tilde{\gamma}}{\gamma}\right). \tag{4.20}$$

The output of the reduction is $(\mathbf{B}', d')$.

We want to prove that if $(\mathbf{B}, \mathbf{t}, d)$ is a YES instance of BINCVP$_{\tilde{\gamma}}$ then $(\mathbf{B}', d')$ is a YES instance of GAPSVP$_\gamma$, and if $(\mathbf{B}, \mathbf{t}, d)$ is a NO instance of BINCVP$_{\tilde{\gamma}}$ then $(\mathbf{B}', d')$ is a NO instance of GAPSVP$_\gamma$.

First assume that $(\mathbf{B}, \mathbf{t}, d)$ is a YES instance, i.e. there exists a vector $\mathbf{x} \in \{0,1\}^k$ such that $\mathbf{t} - \mathbf{Bx}$ is a 0-1 vector with at most d 1's. In particular, $\|\mathbf{Bx} - \mathbf{t}\| \leq \sqrt[p]{d}$. By construction, there exists a vector $\mathbf{z} \in \mathbb{Z}^k$ such that $\mathbf{Tz} = \mathbf{x}$ and $\|\mathbf{Lz} - \mathbf{s}\| < r$. Define

$$\mathbf{w} = \begin{bmatrix} \mathbf{z} \\ -1 \end{bmatrix}$$

and compute the norm of the corresponding lattice vector

$$\begin{aligned} \|\mathbf{B}'\mathbf{w}\|_p^p &= a^p\|\mathbf{Bx} - \mathbf{t}\|_p^p + b^p\|\mathbf{Lz} - \mathbf{s}\|_p^p \\ &\leq (a)^p d + (br)^p, \end{aligned}$$

which, by (4.20), is at most $(d')^p$. This proves that $(\mathbf{B}', d')$ is a YES instance.

Now assume $(\mathbf{B}, \mathbf{t}, d)$ is a NO instance and let $\mathbf{w} = [\mathbf{z}^T, w]^T$ be a nonzero integer vector. We want to prove that $\|\mathbf{B}'\mathbf{w}\|^p > (\gamma d')^p$. Notice that

$$\|\mathbf{B}'\mathbf{w}\|^p = a^p\|\mathbf{Bx} + w\mathbf{t}\|^p + b^p\|\mathbf{Lz} + w\mathbf{s}\|^p.$$

We prove that either $a\|\mathbf{Bx} + w\mathbf{t}\| > \gamma d'$ or $b\|\mathbf{Lz} + w\mathbf{s}\| > \gamma d'$. We distinguish two cases

- If $w = 0$ then $\mathbf{z} \neq 0$ and, therefore, $\|\mathbf{Lz}\|_p > \tilde{\gamma} r$. This proves that

  $$b\|\mathbf{Lz} + w\mathbf{s}\| = b\|\mathbf{Lz}\| > b\tilde{\gamma} r,$$

  which, by (4.20), is at least $\gamma d'$.

- If $w \neq 0$ then $\mathbf{Bx} + w\mathbf{t}$ has more than $\hat{\gamma} d$ nonzero entries. In particular

  $$\|\mathbf{Bx} + w\mathbf{t}\|_p > \sqrt[p]{\hat{\gamma} d} = \frac{2\sqrt[p]{d}}{\sqrt[p]{(1/\gamma)^p - (1/\tilde{\gamma})^p}}. \tag{4.21}$$

  Multiplying (4.21) by a, and using the lower bound in (4.19), we get

  $$a\|\mathbf{Bx} + w\mathbf{t}\|_p > br\tilde{\gamma}$$

  which, by (4.20), is at least $\gamma d'$.

This proves that in either case $\|\mathbf{B}'\mathbf{w}\| > \sqrt[p]{\gamma t}$. □

3. NP-hardness of SVP

In Section 2 we have seen that it is possible to efficiently reduce an NP-hard problem to GAPSVP, provided we have an algorithm to compute objects $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ as specified in Lemma 4.3. To date we do not know how to compute $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ in deterministic polynomial time, so Theorem 4.4 does not prove the NP-hardness of GAPSVP under deterministic (Karp or Cook) reductions. However, we know how to efficiently find $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ probabilistically. Moreover, if a certain number theoretic conjecture holds true, then we can give a deterministic algorithm that runs in polynomial time. Using these algorithms it is possible to prove that GAPSVP is NP-hard under different types of reductions. In the following subsections we use lattice packing and combinatorial constructions to be developed in Chapters 5 and 6 to present various solutions to the problem of Lemma 4.3, and obtain corresponding NP-hardness results for GAPSVP.

3.1 Hardness under randomized reductions

In this subsection we give a probabilistic construction for objects $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ satisfying Lemma 4.3. In particular, we give a randomized algorithm that on input an integer n produces a lattice basis $\mathbf{L}$ and a sphere $\mathcal{B}(\mathbf{s}, r)$ such that $\lambda_1(\mathbf{L})$ is guaranteed to be bigger than r by a factor $\gamma < \sqrt[p]{2}$, and with probability arbitrarily close to 1 (over the random choices of the algorithm) the sphere $\mathcal{B}(\mathbf{s}, r)$ contains a lattice point $\mathbf{Lz}$ for every binary string $\mathbf{Tz} \in \{0,1\}^n$.

The construction is based on a sphere packing result to be proved in Chapter 5. (See that chapter for a discussion of the connection between this problem and general sphere packing questions.) The result is the following.

THEOREM 4.5 *For every $p \geq 1$, $\gamma \in [1, \sqrt[p]{2})$ and $\delta > 0$ there exists a probabilistic algorithm that on input an integer h outputs (in time polynomial in h) integers k and r, a matrix $\mathbf{L} \in \mathbb{Z}^{(k+1)\times k}$, and an integer vector $\mathbf{s} \in \mathbb{Z}^{k+1}$ such that*

- *all vectors in $\mathcal{L}(\mathbf{L})$ have ℓ_p norm bigger than γr, and*
- *for all sufficiently large h, with probability at least $1 - 2^{-h}$ the sphere $\mathcal{B}(\mathbf{s}, r)$ contains at least $h^{\delta h}$ lattice points of the form $\mathbf{Lz}$ where $\mathbf{z}$ is a 0-1 vector with exactly h ones.*

Proof: See Chapter 5, Section 4. □

We also need the following probabilistic combinatorial result to be proved in Chapter 6.

THEOREM 4.6 *Let $\mathcal{Z} \subseteq \{0,1\}^k$ be a set of vectors, each containing exactly h ones. If $|\mathcal{Z}| \geq h!k^{4\sqrt{hn}/\epsilon}$, and $\mathbf{T} \in \{0,1\}^{n\times k}$ is chosen setting each entry to 1 independently at random with probability $p = \frac{1}{4hn}$, then the probability that $\mathbf{T}(\mathcal{Z}) = \{\mathbf{Tz} : \mathbf{z} \in \mathcal{Z}\}$ contains all binary vectors $\{0,1\}^n$ is at least $1 - 6\epsilon$.*

Proof: See Chapter 6, Section 3. □

The homogenization gadget of Lemma 4.3 is easily built combining Theorems 4.5 and 4.6.

Proof [of Lemma 4.3 (probabilistic version)]: Fix an ℓ_p norm ($p \geq 1$) and a constant $\gamma \in [1, \sqrt[p]{2})$. Let n be a sufficiently large integer. We want to build (in time polynomial in n) an integer lattice $\mathbf{L} \in \mathbb{Z}^{k'\times k}$, an integer vector $\mathbf{s} \in \mathbb{Z}^{k'}$, an integer transformation matrix $\mathbf{T} \in \mathbb{Z}^{n\times k}$, and a rational number r such that

- all nonzero vectors in $\mathcal{L}(\mathbf{L})$ have ℓ_p norm greater than γr;
- with probability at least $1 - 1/\operatorname{poly}(n)$, for every $\mathbf{x} \in \{0,1\}^n$ there exists a $\mathbf{z} \in \mathbb{Z}^k$ such that $\mathbf{Tz} = \mathbf{x}$ and $\|\mathbf{Lz} - \mathbf{s}\|_p \leq r$.

Run the algorithm of Theorem 4.5 on input $h = n^4$ and $\delta = 2$. Let $\mathbf{L} \in \mathbb{Z}^{(k+1)\times k}$, $\mathbf{s} \in \mathbb{Z}^{k+1}$ and $r \in \mathbb{Z}$ be the output of the algorithm. Notice that since $\mathbf{L}$ and $\mathbf{s}$ are computed in polynomial time, k is polynomial in h, i.e., $k < h^c$ for some constant c independent of h. Let $\mathcal{Z}$ be the set of all vectors $\mathbf{z} \in \{0,1\}^k$ with exactly h ones, such that $\mathbf{Lz} \in \mathcal{B}(\mathbf{s}, r)$. We know from Theorem 4.5 that all nonzero vectors in $\mathcal{L}(\mathbf{L})$ have ℓ_p norm greater than γr, and the size of $\mathcal{Z}$ is bigger than h^{2h} with probability at least $1 - 2^{-h}$. Now, choose matrix $\mathbf{T} \in \{0,1\}^{n\times k}$ by setting each entry to 1 independently with probability $1/(4hn)$. Notice that

$$|\mathcal{Z}| \geq h^{2h} > h!k^{h/c} = h!k^{\frac{4\sqrt{hn}}{\epsilon}},$$

where $\epsilon = 4c/k$. So, by Theorem 4.6, the probability that for each $\mathbf{x}$ there exists a vector $\mathbf{z}$ such that $\mathbf{x} = \mathbf{Tz}$ and $\mathbf{Lz} \in \mathcal{B}(\mathbf{s}, r)$ is at least $1 - 1/O(k)$. □

Using this proof of Lemma 4.3, Theorem 4.4 shows that GAPSVP is hard to approximate under *reverse unfaithful random* reductions (RUR-reductions for short, see (Johnson, 1990)). These are probabilistic reductions that map NO instances to NO instances with probability 1, and YES instances to YES instances with nonnegligible probability. (In fact our proof has success probability $1 - 1/p(n)$ for some polynomial function $p(n)$.) Although not a proper NP-hardness result (i.e., hardness for

NP under Karp reductions, which would imply that SVP is not in P unless P = NP), hardness under RUR-reductions also gives evidence of the intractability of a problem. In particular, it implies that SVP is not in RP unless RP = NP. (Here RP is the class of decision problems with random polynomial time decision algorithms that are always correct on NO instances and "usually" correct on YES instances, say with probability at least 1/2.) So, the NP-hardness result for SVP (under randomized reductions) also gives theoretical evidence that SVP is intractable.

COROLLARY 4.7 *For any fixed* $p \geq 1$ *and constant* $\gamma < \sqrt[p]{2}$*, the promise problem* GAPSVP$_\gamma$ *in the* ℓ_p *norm is* NP*-hard under RUR-reductions. In particular,* GAPSVP$_\gamma$ *cannot be solved in* RP *(random polynomial time), unless* NP = RP.

3.2 Hardness under nonuniform reductions

In this subsection we show that GAPSVP is NP-hard under deterministic nonuniform polynomial time reductions. A *nonuniform* algorithm is an algorithm that, in addition to the problem instance, takes as additional input a hint string that depends only on the size of the problem instance. In other words, the hint string is the same for all problems of a certain size. The complexity class P/poly is defined as the set of decision (or promise) problems that can be solved by a polynomial time algorithm with a hint of size polynomial in the length of the input. Equivalently, P/poly is the set of languages that can be recognized by a (possibly nonuniform) family of circuits C_k (for $k = 1, 2, \ldots$), with each circuit C_k to be used on input strings of length k, and the size of C_k bounded by a polynomial $p(k) = k^c$. These circuit families are called nonuniform because the sequence of circuits is not necessarily computable, i.e., there might be no (efficient) algorithm that on input k outputs the circuit C_k.

It is easy to see that P/poly contains languages that are not in NP. (In fact, P/poly contains undecidable languages.) However, it is widely believed that NP is not contained in P/poly. Clearly, if a problem is NP-hard under nonuniform polynomial time reductions, then the problem cannot be solved in P (or even in P/poly) unless NP $\subset$ P/poly. Therefore, NP-hardness under nonuniform polynomial time reductions gives evidence that a problem is intractable.

Notice that the randomness in the RUR-reductions of Corollary 4.7 comes exclusively from the algorithm of Lemma 4.3. The algorithm of Lemma 4.3 takes as input only the size (or, more precisely, the rank) of the BINCVP instance being reduced. Moreover, whether or not the algorithm of Lemma 4.3 is successful does not depend on the BINCVP instance. In other words, if on input n the algorithm outputs objects

$(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ satisfying the properties in Lemma 4.3, then $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ are good for any BINCVP instance of rank n.

We know that for any rank n there exists a good homogenization gadget $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ and that the size of $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ is polynomial in n because the proof of Lemma 4.3 given in the previous subsection shows that there is a probabilistic polynomial time algorithm to compute $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ with nonzero (in fact, almost 1) probability. So, for any value of n we can pick a good $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ and use it as hint string to a nonuniform algorithm that reduces BINCVP to GAPSVP as described in the proof of Theorem 4.4. Therefore, we have the following result.

COROLLARY 4.8 *For any fixed* $p \geq 1$ *and constant* $\gamma < \sqrt[p]{2}$, *the promise problem* GAPSVP$_\gamma$ *is hard for* NP *under deterministic nonuniform Karp reductions. In particular,* GAPSVP$_\gamma$ *is not in* P/poly *unless* NP $\subset$ P/poly.

Using standard results from nonuniform complexity (Karp and Lipton, 1980), this also implies that for any ℓ_p norm ($p \geq 1$) and any $\gamma \in [1, \sqrt[p]{2})$, the promise problem GAPSVP$_\gamma$ is not in P unless the polynomial hierarchy (Meyer and Stockmeyer, 1972; Stockmeyer, 1977) collapses to the second level.

3.3 Hardness under deterministic reductions

In this subsection we give a deterministic algorithm to build objects $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ as specified in Lemma 4.3. The algorithm is easily obtained from a deterministic variant of Theorem 4.5, but its correctness depends on the validity of a number theoretic conjecture concerning the distribution of square free smooth numbers. For any $b > 0$, an integer n is called b-smooth if all prime factors of n are bounded by b. Moreover, we say that n is square free if all prime factors of n appear with exponent 1. The conjecture is the following.

CONJECTURE 1 *For any* $\epsilon > 0$ *there exists a* d *such that for all large enough* n, *there is an (odd) integer in* $[n, n + n^{\epsilon}]$ *which is square free and* $(\log^d n)$*-smooth, i.e., all of its prime factors have exponent* 1 *and are less than* $\log^d n$.

The conjecture is reasonable because a relatively simple number theoretic analysis shows that the average number of square free $(\ln n)^d$-smooth numbers in $[n, n + n^{\epsilon}]$ exceeds $n^{\epsilon - \frac{1}{d}}$. Therefore, if $d = 2/\epsilon$ one should expect to find $n^{\frac{\epsilon}{2}}$ square free smooth numbers in $[n, n + n^{\epsilon}]$ on the average. If square free smooth numbers are distributed uniformly enough then one can reasonably assume that $[n, n + n^{\epsilon}]$ contains at least one such number for all sufficiently large n.

We remark that although the above conjecture is quite plausible, proving it seems currently beyond known mathematical techniques. For further discussion about the conjecture the reader is referred to the last section of Chapter 5. In Chapter 5 we prove the following deterministic variant of Theorem 4.5, under the assumption that Conjecture 1 is true.

THEOREM 4.9 *If Conjecture 1 is true, then the following holds. For every $p \geq 1$ and $\gamma < \sqrt[p]{2}$ there exists a deterministic algorithm that on input an integer h outputs (in time polynomial in h) integers k and r (with $k > h$), a matrix $\mathbf{L} \in \mathbb{Z}^{(k+1)\times k}$, and an integer vector $\mathbf{s} \in \mathbb{Z}^{k+1}$ such that*

- *all vectors in $\mathcal{L}(\mathbf{L})$ have ℓ_p norm bigger than γr;*
- *for every vector $\mathbf{x} \in \{0,1\}^h$ there exists a vector $\mathbf{y} \in \{0,1\}^{k-h}$ such that the lattice point $\mathbf{L}[\mathbf{y}^T, \mathbf{x}^T]^T$ belongs to the sphere $\mathcal{B}(\mathbf{s}, r)$.*

Proof: See Chapter 5, Section 3. □

In this case the proof of Lemma 4.3 is immediate.

Proof [of Lemma 4.3 (deterministic version)]: Just run the algorithm of Theorem 4.9 on input $h = n$ to obtain $\mathbf{L} \in \mathbb{Z}^{(k+1)\times k}$, $\mathbf{s} \in \mathbb{Z}^{k+1}$ and $r \in \mathbb{Z}$. Also define $\mathbf{T} = [\mathbf{0}|\mathbf{I}]$ where $\mathbf{I}$ is the $n \times n$ identity matrix and $\mathbf{0}$ is the $n \times (k - n)$ zero matrix. Then, the properties of $(\mathbf{L}, \mathbf{T}, \mathbf{s}, r)$ claimed in Lemma 4.3 immediately follow from Theorem 4.9. □

This shows that if the distribution of square free smooth numbers is sufficiently uniform, then GAPSVP is NP-hard under (deterministic) Karp reductions.

COROLLARY 4.10 *In Conjecture 1 holds true, then for any fixed $p \geq 1$ and $\gamma < \sqrt[p]{2}$, the promise problem* GAPSVP$_\gamma$ *is* NP*-hard under Karp reductions. In particular,* GAPSVP$_\gamma$ *is not in* P *unless* P = NP.

4. Notes

The shortest vector problem is probably the most famous and widely studied algorithmic problem on point lattices. The NP-hardness of SVP (in the ℓ_2 norm) was conjectured in (van Emde Boas, 1981), and remained probably the biggest open question in the area for almost two decades. (See for example (Lovász, 1986; Kannan, 1987b; Arora et al., 1997).) In (Kannan, 1987b) it is shown that approximating CVP within $\sqrt{n}/2$ can be reduced to computing SVP exactly. The reduction presented in Section 1 is a simple modification of (Kannan, 1987b), and shows that a relatively good approximation oracle for SVP is already enough to approximate CVP within $O(\sqrt{n})$ factors. Kannan's reduction suggests that a possible way to demonstrate the NP-hardness of SVP might be to prove NP-hardness of approximating CVP within $O(\sqrt{n})$ factors. Progress in the inapproximability of CVP (Arora et al., 1997) seemed encouraging in this perspective, but in 1997 Goldreich and Goldwasser showed that CVP is not likely to be NP-hard to approximate for factors $\gamma = O(\sqrt{n}/\log n)$ (Goldreich and Goldwasser, 2000), eliminating Kannan's reduction as a viable route toward a resolution of van Emde Boas' conjecture. The NP-hardness of SVP (in its exact version) was finally proved (under RUR reductions) by (Ajtai, 1998). Ajtai's proof easily extends to some weak inapproximability result: in (Ajtai, 1998) it is already claimed that SVP is NP-hard to approximate within a factor $1 + 1/2^{cn}$ (for some constant c) and (Cai and Nerurkar, 1999) shows that Ajtai's proof can actually be extended to inapproximability factor $1 + 1/n^{\epsilon}$ for any fixed $\epsilon > 0$. Still, these inapproximability results are rather weak because the approximation factor rapidly approaches 1 when the dimension of the lattice increases, leaving the question of the inapproximability of SVP basically open. The first significant inapproximability result for SVP in which the approximation factor is bounded away from 1 was proved by Micciancio in 1998, who showed that SVP is NP-hard under RUR-reductions for any approximation factor less than $\sqrt{2}$ (Micciancio, 1998; Micciancio, 2001d). This is the result presented in Sections 2 and 3, and at the time of this writing it is still the strongest inapproximability result for SVP known. (Micciancio, 2001d) represents not only a strengthening, but also a substantial simplification of Ajtai's proof, and it allowed to prove analogous results for coding problems (Dumer et al., 1999). Conjecture 1 about the distribution of square free smooth numbers was also put forward in (Micciancio, 2001d), for the purpose of proving the NP-hardness of approximating SVP under (deterministic) Karp reductions. It should be noted that Conjecture 1, although reasonable, seems to be beyond current mathematical techniques, and it is not likely to be proved any time soon. Both proving the NP-hardness of SVP under (deterministic) Karp reduction, and improving the inapproximability factor from $\sqrt{2}$ to any constant (and possibly some some small polynomial function n^{ϵ} of the rank) are major open problems in the area. These and other open problems related to SVP are discussed in the rest of this section.

In Section 3 we proved that approximating the shortest vector problem in any ℓ_p within factors less than $\sqrt[p]{2}$ is not in polynomial time under any of the following assumptions: (1) NP $\neq$ RP, (2) NP $\not\subseteq$ P/poly,or (3) Conjecture 1 is true and NP $\neq$ P. Although all of these results give theoretical evidence that SVP cannot be approximated in polynomial time, the problem whether solving SVP (even exactly) is NP-hard under deterministic (Karp or Cook) reductions is still open. We noticed that the only place where randomness is used in our reduction is the proof of Lemma 4.3. A deterministic polynomial time solution to Lemma 4.3 would immediately give an NP-hardness result for SVP under Karp reductions.

Our NP-hardness proof is by reduction from a variant of CVP. In particular we reduce instances of BINCVP of size n to instances of GAPSVP of size $m = n^c$, where $c > 2$ is a constant independent of n. Although this gives a polynomial relation between n and m it should be noted that m can be much bigger than n. Therefore, in order to assert that an instance of SVP is hard to solve *in practice*, the dimension m must be rather large. Finding a more efficient reduction, where, for example, $m = O(n)$, is an important open problem. Interestingly, a dimension and approximation preserving reduction is possible in the other direction from SVP to CVP. (See Chapter 3.)

The sphere packing lemma used in our reduction is in a certain sense optimal (at least for the ℓ_2 norm): in Chapter 5 we show that any lattice $\mathbf{L}$ satisfying the lemma must have vectors of length less than $r/\sqrt{2}$. Proving that SVP is NP-hard to approximate within factors larger than $\sqrt{2}$ cannot be done simply improving the construction in Lemma 4.3. Extending the NP-hardness result for SVP (even under randomized or nonuniform reductions) to approximation factors beyond $\sqrt{2}$ (and possibly any constant, or polylogarithmic/quasipolynomial functions of the rank) is certainly the most important open question about the complexity of SVP.

Another open problem, related both to the reduction of Section 1 and that of Section 2, is the relationship between the search and optimization versions of approximate SVP. In Chapter 1 we showed that one can compute the approximate length of the shortest vector in a lattice given an oracle to solve the corresponding promise problem, and vice-versa. Now, we consider computing the length (optimization problem) and actually finding the approximately shortest vector (search problem). The homogenization technique presented in Section 1 reduces the search version of CVP to the search version of SVP. In other words, given an oracle for finding approximately shortest vectors in a lattice, one can find lattice points approximately closest to a given target. Notice that it is not enough to compute the (approximate) length of the shortest lattice vector: when $w = 0$ and the SVP oracle is called on lattice $\mathbf{B}$, one actually needs to find an approximately shortest vector $\mathbf{b}$ in order to perform the projection operation and complete the reduction. Interestingly, (Kannan, 1987b) shows that if the length of the shortest vector in a lattice can be computed exactly, then one can also find the shortest vector. The idea is to introduce small errors in the coordinates of the basis vectors. If the errors are sufficiently small, then the shortest vector in the perturbed lattice corresponds to the same linear combination of the basis vectors of the original lattice, and from the exact length of the shortest vector in the two cases one can reconstruct the coordinates of the shortest vector. Thus, Kannan actually gives a reduction from computing $O(\sqrt{n})$ approximate solutions of CVP to the decisional problem GapSVP_1. Clearly, this reduction from search to optimization versions of SVP is very sensitive to errors, and it does not work if one can only compute good (but not perfect) approximations of the shortest vector length. Finding a reduction from the search version of approximate SVP to the corresponding promise or optimization problem is an important open question, also because of the relation of SVP to other lattice problems. (See Chapter 7.)

One last question about the complexity of SVP is related to the specific way we reduced BinCVP to GapSVP in Section 2. It is easy to see that given a short vector in the SVP lattice, one can compute a close vector $\mathbf{BTz}$ for the original problem. However, it is not clear how to map NP-witnesses in the other direction. Given a solved instance of BinCVP, (i.e., given $(\mathbf{B}, \mathbf{t})$ and a lattice point $\mathbf{Bx}$ close to $\mathbf{t}$,) it is not clear how to find a short vector in the new lattice $\mathbf{B}'$. In the reduction of Theorem 4.4 we only proved that a short vector exists, but the proof does not give an efficient way to find it. The problem is that finding a short nonzero vector in $\mathcal{L}(\mathbf{B}')$ involves finding a lattice point $\mathbf{Lz}$ in the ball $\mathcal{B}(\mathbf{s}, r)$. Interestingly, even if Conjecture 1 is true, and we can give a deterministic reduction from BinCVP to GapSVP, finding a lattice point in that ball requires the solution of a number theoretic problem for which no polynomial time algorithm is known. Finding a reduction from an NP-hard problem to GapSVP (possibly different from the one presented here) for which NP-witnesses can be efficiently mapped from the source to the target problem (sometime called a *Levin reduction*) would be desirable, as such kind of reductions are known for virtually any other NP-hard problem.

Chapter 5

SPHERE PACKINGS

In this chapter we study the following question. What is the maximum possible number of lattice points inside an n-dimensional sphere of radius ρ, given that the minimum distance between lattice points (or, equivalently, the length of the shortest non-zero vector in the lattice) is at least λ? Clearly the answer depends on the ratio λ/ρ only, as both the lattice and the sphere can be scaled up or down preserving λ/ρ. If we drop the requirement that the points belong to a lattice, and allow them to be an arbitrary set of points with large minimum distance (say $\lambda = 2$), we get the following sphere packing problem (see Figure 5.1): how many unit balls can be packed inside an n-dimensional sphere of radius $R = 1 + \rho$? Notice that since the unit balls are disjoint, their centers are at distance at least $\lambda = 2$ from each other. Moreover, since the unit balls are contained in a sphere of radius $1 + \rho$, the centers of the balls are inside a sphere of radius ρ. We want to determine for which values of λ/ρ we can pack exponentially (in n) many points. (Here, and in the rest of this chapter, "exponential" means a function of the form 2^{n^c} for some fixed constant c independent of n.) Notice the following (trivial) facts:

- If λ/ρ is sufficiently large, then only a constant number of points can be packed, independently of the dimension. For example, if $\lambda/\rho > 2$ then only one point can be inside the sphere, while if $\lambda/\rho = 2$ one can have at most 2 points.

- If λ/ρ is a vanishing function of the dimension n, say $\lambda/\rho = 2/\sqrt{n}$, then one can pack exponentially many spheres. Consider for example the cubic lattice $2\mathbb{Z}^n$. This lattice has minimum distance $\lambda = 2$. Now take the sphere centered in $\mathbf{s} = [1, \ldots, 1]^T$ of radius $\rho = \sqrt{n}$.

D. Micciancio et al., *Complexity of Lattice Problems*

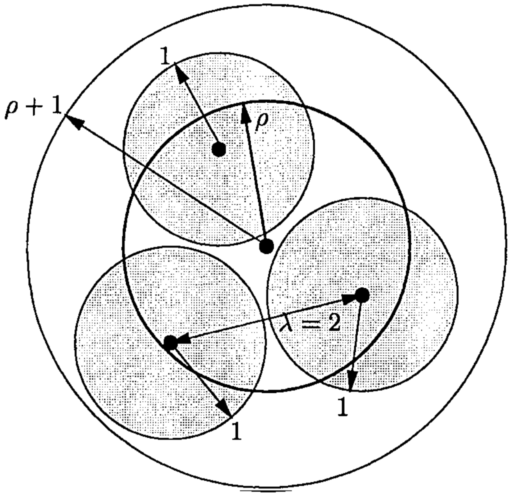


Figure 5.1. Packing unit balls in a bigger sphere

This sphere contains 2^n lattice points, namely all the vertices of the hypercube $[2 \pm 2, \ldots, 2 \pm 2]^T$ (see Figure 5.2).

- For some value of λ/ρ bigger than 1 (i.e., when the distance between lattice points is larger than the radius of the sphere), one can already pack arbitrarily many points, as the dimension n of the lattice grows. For example, consider the set of all integer vectors $\mathbf{x} \in \mathbb{Z}^n$ such that $\sum_{i=1}^n x_i$ is even. This is a lattice generated by basis vectors $\mathbf{b}_i = \mathbf{e}_1 + \mathbf{e}_i$ (for $i = 1, \ldots, n$) with minimum distance $\lambda = \sqrt{2}$. Consider the sphere centered in $\mathbf{e}_1$ of radius $\rho = 1$. The ratio λ/ρ equals $\sqrt{2}$ for every dimension n. Still, the sphere contains $2n$ lattice points $\mathbf{e}_1 \pm \mathbf{e}_i$ (for $i = 1, \ldots, n$). Scaling all coordinates by a factor $\sqrt{2}$, this corresponds to packing $2n$ unit balls in a sphere of radius $1 + \sqrt{2}$ as shown in Figure 5.3.

We are interested in lattices such that $\lambda/\rho > 1$, i.e., the radius of the sphere is smaller than the minimum distance between lattice points. We have just seen that when $\rho = \lambda/\sqrt{2}$, the sphere can contain $2n$ lattice points. A few natural questions arise. Can we do any better when $\lambda/\rho = \sqrt{2}$? What happen when $\lambda/\rho > \sqrt{2}$? Can we pack a superpolynomial (in n) number of points when $\lambda/\rho \in (1, \sqrt{2})$? In the course of this chapter we answer these questions and prove the following facts:

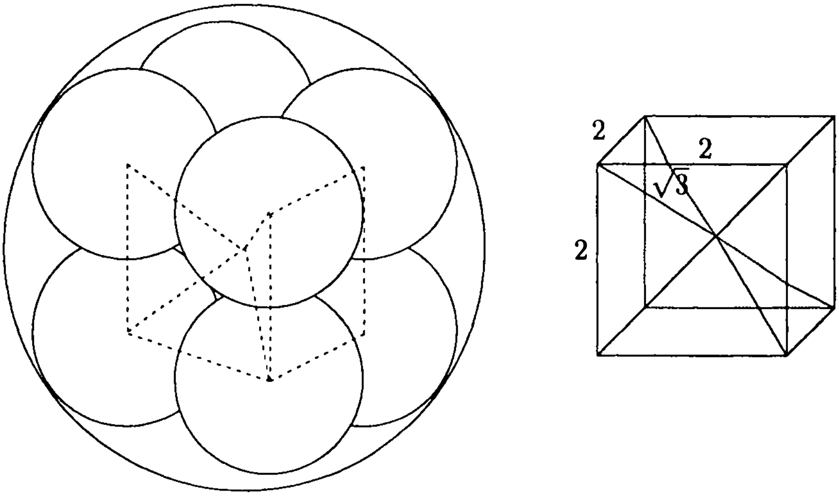


Figure 5.2. The cubic packing

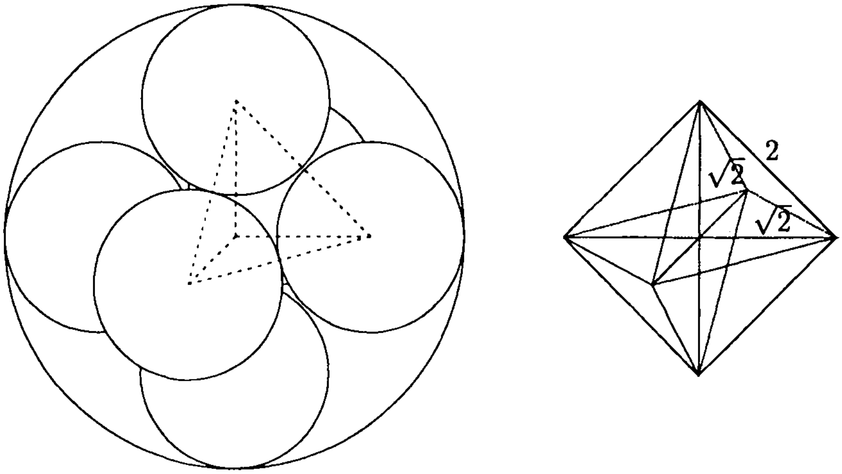


Figure 5.3. The octahedral packing

1 If $\lambda/\rho > \sqrt{2}$, then one can pack only constantly many points (independently of the dimension).

2 If $\lambda/\rho = \sqrt{2}$, then the maximum number of points is precisely $2n$.

3 For any $\lambda/\rho < \sqrt{2}$, one can pack exponentially many points.

Upper bounds 1 and 2 actually hold even if we drop the requirement that the points belong to a lattice, and are proved in Section 1. Lower bound 3 is proved in Section 2. The lattice defined in Section 2 is not rational, i.e., the basis vectors contain arbitrary real entries. This lattice plays a fundamental role in the construction of the homogenization gadget used in the proof of Theorem 4.4. The proof of Theorem 4.4 requires not only the existence of a lattice and a small sphere containing many lattice points, but also an efficient (possibly randomized) algorithm to find such objects. This issue is addressed in Section 3 where we show that the real lattice of Section 2 can be efficiently approximated with a rational one. The construction of Section 2 and 3 is efficient, but probabilistic. In Section 3.3 we use the techniques from Sections 2 and 3 to give a similar, but deterministic, construction that can be proven correct assuming a certain number theoretic conjecture holds true.

1. Packing Points in Small Spheres

In this section we study the cases when $\lambda/\rho \geq \sqrt{2}$ and prove upper bounds on the number of points that can be packed in a sphere of radius ρ while keeping the minimum distance between points at least λ. These upper bounds are not directly relevant to the proof of Theorem 4.4, but they explain why the proof of Theorem 4.4 cannot be easily extended to approximation factors beyond $\sqrt{2}$. We consider arbitrary arrangements of points, not necessarily points of a lattice with large minimum distance. Since we are proving upper bounds on the number of points in a sphere, the results apply to lattice packings as well. Without loss of generality we assume $\lambda = 2$ and bound the maximum number of points that can be placed in a sphere of radius $\rho \leq \sqrt{2}$ while keeping the points at distance at least 2 from each other. Let us start with the simple case $\rho < \sqrt{2}$.

THEOREM 5.1 *For any $\rho < \sqrt{2}$, the maximum number of points at minimum distance 2 from each other that can be packed in a sphere of radius ρ is $\lfloor 2/(2-\rho^2) \rfloor$.*

Proof: Let $\mathbf{x}_1, \ldots, \mathbf{x}_N$ be a set of vectors such that $\|\mathbf{x}_i\| \leq \rho < \sqrt{2}$ and $\|\mathbf{x}_i - \mathbf{x}_j\| \geq 2$ for all $i \neq j$. Notice that

$$\begin{aligned} N(N-1)4 &\leq \sum_{i=1}^{N}\sum_{j=1}^{N} \|\mathbf{x}_i - \mathbf{x}_j\|^2 \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} (\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\langle \mathbf{x}_i, \mathbf{x}_j\rangle) \end{aligned}$$

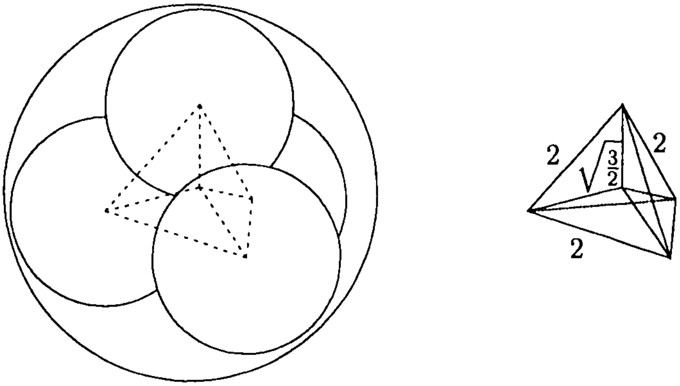


Figure 5.4. The tetrahedral packing

$$
\begin{aligned}
&= 2N\sum_{i=1}^{N}\|\mathbf{x}_i\|^2 - 2\left\|\sum_{i=1}^{N}\mathbf{x}_i\right\|^2 \\
&\leq 2N^2\rho^2
\end{aligned}
$$

and therefore $2(N-1) \leq N\rho^2$. Solving the linear inequality for N one gets $N \leq 2/(2-\rho^2)$ and since N is an integer $N \leq \lfloor 2/(2-\rho^2)\rfloor$. □

Notice that the above bound is sharp: for all $\rho < \sqrt{2}$, one can put $n = \lfloor 2/(2-\rho^2)\rfloor$ unit balls on the vertices of an $(n-1)$-dimensional simplex, and inscribe the simplex inside a sphere of radius $\sqrt{2n/(n+1)} \leq \rho$ (see Figure 5.4). This example also shows that, when $\rho = \sqrt{2}$, for every $n \geq 1$ one can pack $n+1$ balls in the n-dimensional sphere of radius $1+\rho$. In fact, we have already seen that it is possible to do better than that: as soon as λ/ρ reaches $\sqrt{2}$, one can pack $2n$ balls centered at $\pm\sqrt{2}e_i$ for $i = 1,\ldots,n$ inside a sphere of radius $1+\sqrt{2}$. We now show that this packing is optimal. Interestingly, this optimal packing is also a lattice packing: i.e., the distance vectors between the centers of the balls generate a lattice with minimum distance 2.

THEOREM 5.2 *The maximum number of points at distance at least* 2 *from each other that can be placed in a sphere of radius* $\sqrt{2}$ *is* $2n$.

Proof: By induction on n. If $n = 1$, the statement is true. Now assume that the statement holds for some value n, and let us prove it for $n+1$. Let $\mathbf{x}_1,\ldots,\mathbf{x}_N$ vectors in $\mathbb{R}^{n+1}$ such that $\|\mathbf{x}_i\|^2 \leq 2$ and $\|\mathbf{x}_i - \mathbf{x}_j\|^2 \geq 4$. Notice that for all $i \neq j$ one has

$$
\langle \mathbf{x}_i, \mathbf{x}_j\rangle = \frac{1}{2}(\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - \|\mathbf{x}_i - \mathbf{x}_j\|^2)
$$

$$\leq \frac{1}{2}(2+2-4) = 0$$

i.e., the angles between any pair of vectors are at least $\pi/2$. We first explain the geometric idea behind the proof. Assume without loss of generality that $\mathbf{x}_N \neq \mathbf{0}$. Think of $\mathbf{x}_N$ as the north pole. We map all point to the poles and the equator in such a way that all angles between any pair of points remain at least $\pi/2$. Then, we apply induction to the set of points on the equator.

We now give the formal proof. Define the set of vectors

$$\mathbf{x}'_i = \begin{cases} \langle \mathbf{x}_N, \mathbf{x}_N\rangle \mathbf{x}_i - \langle \mathbf{x}_i, \mathbf{x}_N\rangle \mathbf{x}_N & \text{if } \langle \mathbf{x}_N, \mathbf{x}_N\rangle \mathbf{x}_i \neq \langle \mathbf{x}_i, \mathbf{x}_N\rangle \mathbf{x}_N \\ \mathbf{x}_i & \text{otherwise} \end{cases}$$

and let $\mathbf{x}''_i = \sqrt{2}\mathbf{x}'_i/\|\mathbf{x}'_i\|$. Notice that for all i, $\|\mathbf{x}''_i\|^2 = 2$ (i.e., $\mathbf{x}''_i$ is on the surface) and either $\mathbf{x}''_i = \pm\mathbf{x}''_N$ (i.e., $\mathbf{x}''_i$ is a "pole") or $\langle \mathbf{x}''_i, \mathbf{x}''_N\rangle = 0$ (i.e., $\mathbf{x}''_i$ is on the "equator"). We now prove that $\|\mathbf{x}''_i - \mathbf{x}''_j\|^2 \geq 4$ for all $i \neq j$. If $\mathbf{x}''_i = \pm\mathbf{x}''_N$ or $\mathbf{x}''_j = \pm\mathbf{x}''_N$ it is obvious. So, assume $\mathbf{x}''_i \neq \pm\mathbf{x}''_N$ and $\mathbf{x}''_j \neq \pm\mathbf{x}''_N$. Notice that

$$\begin{aligned} \|\mathbf{x}''_i - \mathbf{x}''_j\| &= \|\mathbf{x}''_i\|^2 + \|\mathbf{x}''_j\|^2 - 2\langle \mathbf{x}''_i, \mathbf{x}''_j\rangle \\ &= 2 + 2 - 2\frac{\langle \mathbf{x}_i, \mathbf{x}_j\rangle\langle \mathbf{x}_N, \mathbf{x}_N\rangle^2 - \langle \mathbf{x}_i, \mathbf{x}_N\rangle\langle \mathbf{x}_j, \mathbf{x}_N\rangle\langle \mathbf{x}_N, \mathbf{x}_N\rangle}{\|\mathbf{x}'_i\| \cdot \|\mathbf{x}'_j\|} \\ &\geq 4 \end{aligned}$$

because $\langle \mathbf{x}_i, \mathbf{x}_j\rangle, \langle \mathbf{x}_i, \mathbf{x}_N\rangle, \langle \mathbf{x}_j, \mathbf{x}_N\rangle \leq 0$ and $\langle \mathbf{x}_N, \mathbf{x}_N\rangle > 0$. Therefore all points, except at most two of them, belong to the n-dimensional subspace orthogonal to $\mathbf{x}_N$. By induction hypothesis there are at most $2n$ such points and $N \leq 2(n+1)$. □

2. The Exponential Sphere Packing

In this section we study the case $\lambda/\rho < \sqrt{2}$ and prove that for any radius $\rho \geq 0$ and distance $\lambda > 0$ bounded by $\lambda < \sqrt{2}\rho$, there exist a lattice $\mathcal{L}(\tilde{\mathbf{L}})$ (with minimum distance λ) with exponentially large clusters of lattice points. In particular, we show that there exist spheres $\mathcal{B}(\mathbf{s}, \rho)$ containing 2^{n^δ} lattice points, where $\delta > 0$ is a constant that depends only on the ratio λ/ρ. The construction has several additional properties, as required in the proof of Theorem 4.4. For example, the lattice points in $\mathcal{B}(\mathbf{s}, \rho)$ are vertices of the fundamental parallelepiped defined by the (given) lattice basis $\tilde{\mathbf{L}}$. Since we want to prove Theorem 4.4 for any ℓ_p norm, we give a generic construction of lattice $\tilde{\mathbf{L}}$ with respect to an arbitrary, but fixed, norm ℓ_p. In the rest of this section an arbitrary ℓ_p

norm is assumed, and $\mathcal{B}(\mathbf{s}, r)$ denotes the ℓ_p ball $\{\mathbf{x} : \|\mathbf{x} - \mathbf{s}\|_p \leq r\}$ of radius r centered at $\mathbf{s}$.

2.1 The Schnorr-Adleman prime number lattice

We begin by defining a lattice $\mathcal{L}(\widetilde{\mathbf{L}})$ and prove a lower bound to the length of the shortest non-zero vector in $\mathcal{L}(\widetilde{\mathbf{L}})$. The lattice is a generalization (to a generic ℓ_p norm) of a similar lattice used by (Schnorr, 1993) and (Adleman, 1995) in a different context. For notational convenience we define a rank k lattice in $\mathbb{R}^{k+1}$, i.e., we set $\widetilde{\mathbf{L}}$ to a rectangular (full-rank) matrix in $\mathbb{R}^{(k+1)\times k}$. A full dimensional lattice with the same properties can be easily found by simple linear algebra. The definition of $\mathbf{L}$ is parametric with respect to a real $\alpha > 0$, a sequence of positive integers $\mathbf{a} = a_1, \ldots, a_k$ and an l_p norm ($p \geq 1$). We use the logarithms of the integers a_1 to a_k as entries in the basis vectors, and define a basis vector for each a_i.

The idea is to map the multiplicative structure of integers $a_1, \ldots, a_k$ to the additive structure of lattice $\mathcal{L}(\widetilde{\mathbf{L}})$, defining a basis vector for each a_i and expressing its entries in terms of the logarithm of a_i. This way the problem of finding a sphere containing many lattice points is reduced to the problem of finding a small interval containing many products of the a_i's. At the end we will set α to some large number (exponential in k), and $\mathbf{a}$ to a sequence of small primes. The existence of a sphere containing many lattice points will follow from the density of the primes and a simple averaging argument.

LEMMA 5.3 *Let* $\mathbf{a} = [a_1, \ldots, a_k]$ *be a sequence of relatively prime odd positive integers. Then for any* ℓ_p *norm* ($p \geq 1$)*, and any real* $\alpha > 0$*, all nonzero vectors in the lattice generated by the (columns of the) matrix*

$$\widetilde{\mathbf{L}} = \begin{bmatrix} \sqrt[p]{\ln a_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sqrt[p]{\ln a_k} \\ \alpha \ln a_1 & \cdots & \alpha \ln a_k \end{bmatrix} \in \mathbb{R}^{(k+1)\times k} \qquad (5.1)$$

have ℓ_p *norm bigger than* $\sqrt[p]{2 \ln \alpha}$.

Proof: We want to prove that for all nonzero integer vectors $\mathbf{z} \in \mathbb{Z}^k$,

$$\|\widetilde{\mathbf{L}}\mathbf{z}\|_p^p \geq 2 \ln \alpha.$$

We first introduce some notation. Let $\mathbf{R} \in \mathbb{R}^k$ be the row vector

$$\mathbf{R} = [\ln a_1, \ln a_2, \ldots, \ln a_k] \qquad (5.2)$$

and $\mathbf{D} \in \mathbb{R}^{k\times k}$ be the diagonal matrix

$$\mathbf{D} = \begin{bmatrix} \sqrt[p]{\ln a_1} & 0 & \cdots & 0 \\ 0 & \sqrt[p]{\ln a_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \sqrt[p]{\ln a_k} \end{bmatrix}. \tag{5.3}$$

Notice that

$$\tilde{\mathbf{L}} = \begin{bmatrix} \mathbf{D} \\ \alpha\mathbf{R} \end{bmatrix}$$

and $\|\tilde{\mathbf{L}}\mathbf{z}\|_p^p = \|\mathbf{D}\mathbf{z}\|_p^p + \alpha^p\,|\mathbf{R}\mathbf{z}|^p$. We bound the two terms separately. Define the integers

$$\hat{g} = \prod\{a_i^{z_i} : z_i > 0\}, \qquad \check{g} = \prod\{a_i^{-z_i} : z_i < 0\}, \qquad g = \hat{g}\check{g} = \prod_{i=1}^{k} a_i^{|z_i|}.$$

Using this notation, the first term satisfies

$$\begin{aligned} \|\mathbf{D}\mathbf{z}\|_p^p &= \sum_i |z_i|^p \ln a_i \\ &\geq \sum_i |z_i| \ln a_i \\ &= \ln g \end{aligned}$$

because $p \geq 1$ and the z_i's are integers. Bounding the second term is slightly more complex:

$$\begin{aligned} |\mathbf{R}\mathbf{z}| &= \left|\sum_i z_i \ln a_i\right| \\ &= |\ln\hat{g} - \ln\check{g}| \\ &= \ln\left(1 + \frac{|\hat{g}-\check{g}|}{\min\{\hat{g},\check{g}\}}\right). \end{aligned}$$

Now notice that since $\mathbf{z}$ is nonzero, $\hat{g}$ and $\check{g}$ are distinct odd integers and therefore $|\hat{g}-\check{g}| \geq 2$. Moreover, $\min\{\hat{g},\check{g}\} < \sqrt{\hat{g}\check{g}} = \sqrt{g}$. By monotonicity and concavity of function $\ln(1+x)$ over the interval $[0,2]$, one gets

$$\ln\left(1 + \frac{|\hat{g}-\check{g}|}{\min\{\hat{g},\check{g}\}}\right) > \ln\left(1 + \frac{2}{\sqrt{g}}\right) > \frac{2}{\sqrt{g}}\cdot\frac{\ln 3}{2} > \frac{1}{\sqrt{g}}.$$

Combining the two bounds one gets

$$\|\tilde{\mathbf{L}}\mathbf{z}\|_p^p = \|\mathbf{D}\mathbf{z}\|_p^p + \alpha^p\,(\mathbf{R}\mathbf{z})^p > \ln g + \frac{\alpha^p}{g^{p/2}}$$

which is a continuous function of g with derivative

$$\frac{1}{g}\left(1-\frac{p}{2}\cdot\frac{\alpha^p}{g^{p/2}}\right).$$

The function is minimized (over the reals) when $g = \alpha^2\left(\frac{p}{2}\right)^{2/p}$ with minimum

$$2\ln\alpha+\left(\frac{2}{p}\right)\ln\left(\frac{p}{2}\right)+\left(\frac{2}{p}\right)>2\ln\alpha+\left(\frac{2}{p}\right)\ln p>2\ln\alpha.$$

Therefore, for all nonzero integer vectors $\mathbf{z}$, $\|\tilde{\mathbf{L}}\mathbf{z}\|_p^p > 2\ln\alpha$. □

Notice that a simple (but uninteresting) way to increase the length of the shortest vector in a lattice is to multiply all the coordinates by the same scaling factor α. In lattice $\mathcal{L}(\tilde{\mathbf{L}})$ only the last coordinate is multiplied by the scaling factor. Still, this is enough to make the length of the shortest non-zero vector arbitrarily large. However, while multiplying all coordinated increases the minimum distance of the lattice by the same multiplicative factor α, the minimum distance of $\mathcal{L}(\tilde{\mathbf{L}})$ is only logarithmic in α.

2.2 Finding clusters

In this section we prove that for appropriate choice of the parameters, there exists a sphere (of radius ρ) containing many lattice points. Obviously the center of such a sphere cannot be a point in the lattice if one wants the sphere to contain more than a single lattice point.

We look at spheres with center

$$\tilde{\mathbf{s}}=\begin{bmatrix}0\\ \vdots\\ 0\\ \alpha\ln b\end{bmatrix}\in\mathbb{R}^{k+1}. \tag{5.4}$$

where b is a positive integer, and show that there is a close relationship between finding lattice vectors close to $\mathbf{s}$ and approximating the integer b as a product of the a_i's. In particular, we prove that if b can be approximated by the product of a subset of the $a_i's$, then there are lattice points close to s. (A converse of this lemma is presented in Subsection 2.3.)

LEMMA 5.4 *Let* $\tilde{\mathbf{L}}$ *and* $\tilde{\mathbf{s}}$ *be defined as in* (5.1) *and* (5.4). *For any* ℓ_p *norm* ($p \geq 1$), *reals* $\alpha, b \geq 1$, *positive integers* $a_1,\ldots,a_k$, *and boolean* *vector* $\mathbf{z} \in \{0,1\}^k$, *if the integer* $g = \prod_i a_i^{z_i}$ *belongs to the interval* $[b, b(1+1/\alpha)]$, *then*

$$\|\tilde{\mathbf{L}}\mathbf{z} - \tilde{\mathbf{s}}\|_p \leq \sqrt[p]{\ln b + 2},$$

i.e., lattice point $\tilde{\mathbf{L}}\mathbf{z}$ *is within distance* $\sqrt[p]{\ln b + 2}$ *from* $\tilde{\mathbf{s}}$

Proof: Let $\mathbf{D}$ and $\mathbf{R}$ be as defined in (5.3) and (5.2). Notice that since $\mathbf{z}$ is a 0-1 vector,

$$\|\mathbf{D}\mathbf{z}\|_p^p = \mathbf{R}\mathbf{z} = \ln g,$$

and therefore

$$\begin{aligned}\|\tilde{\mathbf{L}}\mathbf{z} - \tilde{\mathbf{s}}\|_p^p &= \|\mathbf{D}\mathbf{z}\|_p^p + \alpha^p|\mathbf{R}\mathbf{z} - \ln b|^p \\ &= \ln g + \alpha^p|\ln g - \ln b|^p \\ &= \ln b + \ln\frac{g}{b} + \left|\alpha \ln \frac{g}{b}\right|^p.\end{aligned}$$

From the assumption $g \in [b, b(1+1/\alpha)]$ and using the inequality $\ln(1+x) < x$ (valid for all $x \neq 0$) one gets

$$0 \leq \ln\frac{g}{b} \leq \ln\left(1 + \frac{1}{\alpha}\right) < \frac{1}{\alpha}$$

which, substituted in the previous expression, gives

$$\|\tilde{\mathbf{L}}\mathbf{z} - \tilde{\mathbf{s}}\|_p^p < \ln b + \frac{1}{\alpha} + 1 \leq \ln b + 2. \qquad \square$$

Now let ϵ be a small positive real constant and set $\alpha = b^{(1-\epsilon)}$. By Lemmas 5.3 and 5.4, the minimum distance between lattice points is bigger than $\lambda = \sqrt[p]{2(1-\epsilon)\ln b}$, and there are many lattice points within distance $\sqrt[p]{\ln b + 2} \approx \lambda/\sqrt[p]{2}$ from $\tilde{\mathbf{s}}$, provided that the interval $[b, b + b^\epsilon]$ contains many products of the form $\prod_{i \in S} a_i$ (for $S \subseteq \{1, \ldots, k\}$). If $a_1, \ldots, a_k$ are the first k odd prime numbers, this is the same as saying that $[b, b + b^\epsilon]$ contains many square free odd (a_k)-smooth numbers. (An integer x is y-smooth if all prime factors of x are at most y. Moreover, x is square free if all of its prime factors appear with exponent 1.) We now informally estimate for which values of k and b one should expect $[b, b+b^\epsilon]$ to contain a large number of such products. A rigorous probabilistic analysis will follow right after.

Fix some integer $c > 1/\epsilon$, and set $k = h^c$ for a sufficiently large integer h to be determined. Let $a_1, \ldots, a_k$ be the first k odd primes, and consider the set of products of all subsets of size h:

$$M = \left\{\prod_{i \in S} a_i : S \subset \{1, \ldots, k\}, |S| = h\right\}.$$

Notice that

$$|M| = \binom{k}{h} = \prod_{i=0}^{h-1} \frac{k-i}{h-i} \geq \prod_{i=0}^{h-1} \frac{k}{h} = h^{(c-1)h}. \tag{5.5}$$

So, all elements of M belong to the interval $[1, (a_k)^h]$. If we choose b uniformly at random in this interval, the expected size of $[b, b+b^\epsilon]$ is $\Omega((a_k)^{\epsilon h})$ and we can estimate the number of elements of M contained in $[b, b+b^\epsilon]$ to be

$$\Omega((a_k)^{\epsilon h}) \cdot \frac{|M|}{(a_k)^h} \geq \Omega\left(\frac{h^{c-1}}{(a_k)^{1-\epsilon}}\right)^h.$$

By the prime number theorem, $a_k = O(k \ln k) = O(h^c \ln h)$ and therefore our estimate is $\Omega(h^{\epsilon c-1}/\ln h)^h > 2^h$ for all sufficiently large h.

Making the above argument more formal, one can prove that there exists an interval $[b, b+b^\epsilon]$ containing exponentially (in h) many products from M. In fact we can do more than just proving that such a b exists: below we give a natural probability distribution over the integers such that if b is chosen according to this distribution than $[b, b+b^\epsilon]$ is likely to contain many products. Notice that if square free smooth numbers are distributed uniformly enough, then all (or most) choices of b are good. Unfortunately, we do not know enough about the distribution of smooth numbers to prove that most intervals $[b, b+b^\epsilon]$ are good. In particular, choosing b uniformly at random (from all integers smaller than a_k^h) does not necessarily give a good interval $[b, b+b^\epsilon]$ with high probability. In order to overcome this problem, we exploit the smooth number distribution (whatever it is) to bias the choice of the interval toward those containing many smooth numbers. The idea is to set b to the product of a random (size h) subset of the a_i's. This way, the interval $[b, b+b^\epsilon]$ is selected with a probability roughly proportional to the number of square free (a_k)-smooth numbers contained in it. So, for example, intervals containing no smooth numbers are never selected, and intervals containing few smooth numbers are selected with very small probability. The probability of choosing an interval containing few products is bounded in the next lemma. In fact the lemma is quite general and applies to any set M of real numbers bigger than 1.

LEMMA 5.5 *For every positive real numbers $\epsilon \in [0,1)$, $\mu > 1$, integer $H \geq 1$, and any finite subset $M \subset [1, \mu)$, if b is chosen uniformly at random from M, then the probability that $[b, b+b^\epsilon)$ contains less than H elements from M is at most*

$$\Pr_{b \in M}\{|[b, b+b^\epsilon) \cap M| < H\} < \frac{\mu^{1-\epsilon} \cdot H}{\kappa(\epsilon) \cdot |M|},$$

where $\kappa(\epsilon) = 1 - 2^{\epsilon - 1}$ is a constant that depends on ϵ.

Proof: Let B be the set of all $b \in M$ such that $|[b, b+b^{\epsilon}) \cap M| < H$. We show that $|B|$ can be partitioned into at most $K = \mu^{1-\epsilon}/\kappa(\epsilon)$ subsets, each containing less than H elements. It follows that

$$\Pr_{b \in M}\{b \in B\} = \frac{|B|}{|M|} \leq \frac{K(H-1)}{|M|} < \frac{\mu^{1-\epsilon} \cdot H}{\kappa(\epsilon) \cdot |M|}.$$

Divide $[1, \mu)$ into $\lceil \log_2 \mu \rceil$ intervals $[2^m, 2^{m+1})$ for $m = 0, \ldots, \lceil \log_2 \mu \rceil - 1$. Then divide each interval $[2^m, 2^{m+1})$ into $2^m/2^{\epsilon m} = 2^{(1-\epsilon)m}$ subintervals of size $2^{\epsilon m}$. Notice that each subinterval is of the form $[x, x+y)$ for some $y \leq x^{\epsilon}$, therefore it contains at most $H - 1$ points from B. It remains to count the total number of subintervals. Adding up the number of subintervals for each interval $[2^m, 2^{m+1})$ we get

$$\begin{aligned} K &= \sum_{m=0}^{\lceil \log_2 \mu \rceil - 1} 2^{(1-\epsilon)m} \\ &= \frac{2^{(1-\epsilon)\lceil \log_2 \mu \rceil} - 1}{2^{1-\epsilon} - 1} \\ &< \frac{(2\mu)^{1-\epsilon}}{2^{1-\epsilon} - 1} = \frac{\mu^{1-\epsilon}}{\kappa(\epsilon)}. \square \end{aligned}$$

Applying this lemma to the set of square free smooth numbers we get the following corollary.

COROLLARY 5.6 *For all reals $\epsilon, \delta > 0$, there exists a constant c such that for any sufficiently large integer h, the following holds. Let $a_1, \ldots, a_k$ be the first $k = h^c$ odd primes, and M the set of all products $\prod_{i \in S} a_i$, where S is a size h subset of $\{1, \ldots, k\}$. If b is chosen uniformly at random from M then the probability that $[b, b + b^{\epsilon})$ contains less than $h^{\delta h}$ elements of M is at most 2^{-h}.*

Proof: Fix some $\epsilon, \delta > 0$ and let c be an integer bigger than $(1+\delta)/\epsilon$. Let $\mu = a_k^h$. Notice that M is contained in $[1, \mu)$ and, by (5.5), $|M| \geq h^{(c-1)h}$. Applying Lemma 5.5 to set M with $H = h^{\delta h}$, we get

$$\begin{aligned} \Pr\{|[b, b + b^{\epsilon}) \cap M| < H\} &< \frac{h^{\delta h} \cdot \mu^{1-\epsilon}}{\kappa(\epsilon)|M|} \\ &< \frac{h^{\delta h} a_k^{(1-\epsilon)h}}{\kappa(\epsilon) h^{(c-1)h}}. \end{aligned}$$

By the prime number theorem, $a_k = O(k \ln k) = O(h^c \ln h)$, which substituted in the above expression gives

$$\begin{aligned}
\Pr\{|[b, b+b^\epsilon] \cap M| \leq H\} &< \frac{h^{\delta h}\, O(h^c \ln h)^{(1-\epsilon)h}}{\kappa(\epsilon) h^{(c-1)h}} \\
&= \left(\frac{O(\ln h)^{(1-\epsilon)}}{h^{\epsilon c-(1+\delta)}}\right)^h \\
&< \left(\frac{O(\ln h)}{h^{\epsilon c-(1+\delta)}}\right)^h \\
&< 2^{-h}
\end{aligned}$$

for all sufficiently large h because $\epsilon c - (1+\delta) > 0$.

Combining Lemma 5.3, Lemma 5.4, and Corollary 5.6, we immediately get the following theorem.

THEOREM 5.7 *For all reals $\epsilon, \delta > 0$, there exists an integer c such that the following holds. Let h be a sufficiently large positive integer, $k = h^c$, and $a_1, \ldots, a_k$ be the first k odd primes. Let b be the product of a random subset of $\{a_1, \ldots, a_k\}$ of size h and set $\alpha = b^{1-\epsilon}$. Define $\widetilde{\mathbf{L}}$ and $\widetilde{\mathbf{s}}$ as in* (5.1) *and* (5.4), *and let $\widetilde{r} = \sqrt[p]{(1+\epsilon)\ln b} > 1$. Then*

- *all non-zero vectors in $\mathcal{L}(\widetilde{\mathbf{L}})$ have ℓ_p norm greater than*

$$\sqrt[p]{2\,((1-\epsilon)/(1+\epsilon))}\,\widetilde{r},$$

- *with probability at least $1 - 2^{-h}$ (over the choice of b), the ball $\mathcal{B}(\widetilde{\mathbf{s}}, r)$ contains more than $h^{\delta h}$ lattice points of the form $\widetilde{\mathbf{L}}\mathbf{z}$ where $\mathbf{z}$ is a* 0-1 *vector with exactly h ones.*

Theorem 5.7 (or more precisely, its adaption to integer lattices to be described in Section 3) plays a fundamental role in the proof that approximating SVP in the ℓ_p norm is NP-hard (see Theorem 4.4). When specialized to the ℓ_2 norm, Theorem 5.7 also answers the sphere packing question posed at the beginning of this chapter.

COROLLARY 5.8 *For every $\gamma < \sqrt{2}$ there exists a constant $\epsilon > 0$ such that the following holds. For every (sufficiently large) positive integer k, there is a rank k lattice $\widetilde{\mathbf{L}}$ with minimum distance λ and a point $\widetilde{\mathbf{s}}$ such that the ball $\mathcal{B}(\widetilde{\mathbf{s}}, \lambda/\gamma)$ contains 2^{k^ϵ} lattice points.*

2.3 Some additional properties

In this subsection we prove some additional properties about the lattice $\mathcal{L}(\tilde{\mathbf{L}})$. The results in this subsection are presented to improve our understanding of lattice $\tilde{\mathbf{L}}$, but they are not used in any proof in the rest of this book. So, the section can be safely skipped without affecting the reading of the rest of the book.

We first give a closed expression for the determinant of the lattice. Then we prove a converse of Lemma 5.4 for the l_1 norm. Namely, we show that any lattice point sufficiently close to s corresponds to a good approximation of the integer b as a product of the a_i's.

PROPOSITION 5.9 *For sequence of integers $a_1, \ldots, a_k$, the determinant of lattice $\tilde{\mathbf{L}}$ defined in (5.1) is*

$$\sqrt{\left(1 + \alpha^2 \sum \ln a_i\right) \prod_{i=1}^{k} \ln a_i}.$$

Proof: Compute the Gram matrix $B^T \cdot B$ and evaluate its determinant. It can be easily proved by induction on the rank that the value of the determinant equals the formula in the proposition. □

The determinant can be used to bound the length of the shortest vector in the lattice using Minkowski's first theorem. Interestingly, for appropriate choice of the parameters, the upper bound given by Minkowski's theorem is not much bigger than the lower bound proved in Lemma 5.3, and therefore all the successive minima of lattice $\mathcal{L}(\tilde{\mathbf{L}})$ are relatively close to each other.

Finally, we present a converse of Lemma 5.4 for the special case of $p = 1$ (similar results might hold in any l_p norm, but assuming $p = 1$ makes the calculations much simpler). In Lemma 5.4 we showed that if b can be approximated as a product of a subset of the a_i's then there exists a lattice point close to s. We now show that if there are lattice points close to s (in the l_1 norm) then b can be approximated as a product of the a_i's in the following sense.

DEFINITION 5.1 *Let x be an arbitrary (positive) real number and let p/q be a rational. We say that p/q is a Diophantine δ-approximation of x if $|p - qb| < \delta$.*

We prove that if a lattice point is close to $\tilde{\mathbf{s}}$ then the corresponding integer is a good Diophantine approximation of b. The following proposition strengthen a similar result of (Schnorr, 1993).

PROPOSITION 5.10 *Let $\alpha, b > 0$ be two arbitrary positive constant, and let $\widetilde{\mathbf{L}}$ and $\widetilde{\mathbf{s}}$ be as defined in (5.1) and (5.4). For any integer vector $\mathbf{z}$ such that $\|\widetilde{L}\mathbf{z} - \widetilde{\mathbf{s}}\|_1 < \ln b$, $g = \prod a_i^{x_i}$ is a Diophantine (b/α)-approximation of b.*

Proof: Let $g, \hat{g}, \breve{g}$ be defined as in the lemma. We want to find the maximum of the function $|\hat{g} - \breve{g}b|$ subject to the constraint $\|\widetilde{\mathbf{L}}\mathbf{z} - \widetilde{\mathbf{s}}\|_1 < \ln b$. Notice that

$$\|\widetilde{\mathbf{L}}\mathbf{z} - \widetilde{\mathbf{s}}\|_1 = \ln \hat{g} + \ln \breve{g} + \alpha |\ln \hat{g} - \ln \breve{g}b|$$

and $|\hat{g} - \breve{g}b|$ are symmetric with respect to $\hat{g}$ and $\breve{g}b$, i.e., if one replaces $\hat{g}$ by $\breve{g}b$ and $\breve{g}$ by $\hat{g}/b$ the value of the functions is unchanged. Assume without loss of generality that $\hat{g} \geq \breve{g}b$. The problem become to maximize $\hat{g} - \breve{g}b$ subject to the constraint

$$(1 + \alpha)\ln \hat{g} + (1 - \alpha)\ln \breve{g} < (1 + \alpha)\ln b.$$

For every fixed value $\breve{g}$, the function $\hat{g} - \breve{g}b$ is maximized subject to the above constraint when $\hat{g} = b\breve{g}^{\frac{\alpha-1}{\alpha+1}}$. So, let's compute the (unconstrained) maximum of the function

$$b\breve{g}^{\frac{\alpha-1}{\alpha+1}} - \breve{g}b$$

This is a continuous function of $\breve{g}$ with derivative

$$b\left(\frac{\alpha - 1}{\alpha + 1}\right)\breve{g}^{-\frac{2}{\alpha+1}} - b.$$

The maximum is achieved when $\breve{g} = \left(\frac{\alpha-1}{\alpha+1}\right)^{(\alpha+1)/2}$ and equals

$$\left(1 - \frac{2}{\alpha + 1}\right)^{\frac{\alpha-1}{2}} \frac{2b}{\alpha + 1} < \frac{b}{\alpha}$$

for all $\alpha \geq 3$. □

In particular, if $\alpha = b^{1-\epsilon}$ then for every lattice vector within distance $\ln b$ from $\mathbf{s}$, the integer g associated to the vector is a Diophantine b^{ϵ}-approximation of b.

3. Integer Lattices

In the previous section we proved that as far as real entries are allowed one can easily define a basis $\widetilde{\mathbf{L}}$ and probabilistically find a vector $\widetilde{\mathbf{s}}$ with the property that a sphere centered in $\widetilde{\mathbf{s}}$ of radius slightly bigger than $\lambda(\widetilde{\mathbf{L}})/\sqrt[4]{2}$ contains many lattice points. We now prove that the same result can be achieved using a suitable integer approximation of $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{s}}$. The error incurred by approximating (a multiple of) $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{s}}$ with integers is bounded in the following two lemmas.

LEMMA 5.11 *For all $\eta \geq 1$ and all integer vectors $\mathbf{z} \in \mathbb{Z}^k$,*

$$\|\mathbf{L}\mathbf{z}\|_p \geq (\eta - 1)k\|\tilde{\mathbf{L}}\mathbf{z}\|_p,$$

where $\mathbf{L} = \lfloor (k\eta)\tilde{\mathbf{L}} \rceil$ is the matrix obtained multiplying $\tilde{\mathbf{L}}$ by $k\eta$ and rounding each entry to the closest integer.

Proof: By triangular inequality

$$\begin{aligned} \|\mathbf{L}\mathbf{z}\|_p &= \|(k\eta)\tilde{\mathbf{L}}\mathbf{z} + (\mathbf{L} - (k\eta)\tilde{\mathbf{L}})\mathbf{z}\|_p \\ &\geq \|(k\eta)\tilde{\mathbf{L}}\mathbf{z}\|_p - \|(\mathbf{L} - (k\eta)\tilde{\mathbf{L}})\mathbf{z}\|_p \\ &= \eta k\|\tilde{\mathbf{L}}\mathbf{z}\|_p - \|(\mathbf{L} - (k\eta)\tilde{\mathbf{L}})\mathbf{z}\|_p. \end{aligned}$$

It remains to prove that $\|(\mathbf{L} - (k\eta)\tilde{\mathbf{L}})\mathbf{z}\|_p \leq k\|\tilde{\mathbf{L}}\mathbf{z}\|_p$. Notice that all entries in $(\mathbf{L} - (k\eta)\tilde{\mathbf{L}})$ are at most $1/2$ in absolute value. Therefore

$$\begin{aligned} \|(\mathbf{L} - (k\eta)\tilde{\mathbf{L}})\mathbf{z}\|_p &\leq \frac{1}{2}\sqrt[p]{\|\mathbf{z}\|_p^p + \left(\sum |z_i|\right)^p} \\ &\leq \frac{1}{2}\sqrt[p]{\|\mathbf{z}\|_p^p + k^p\|\mathbf{z}\|_p^p} \\ &\leq k\|\mathbf{z}\|_p. \end{aligned}$$

Furthermore,

$$\begin{aligned} \|\tilde{\mathbf{L}}\mathbf{z}\|_p^p &= \|\mathbf{D}\mathbf{z}\|_p^p + \alpha^p|\mathbf{R}\mathbf{z}|^p \\ &\geq \|\mathbf{D}\mathbf{z}\|_p^p \\ &\geq \|\mathbf{z}\|_p^p \end{aligned}$$

because $\mathbf{D}$ is diagonal with all entries greater than 1. This proves that $\|(\mathbf{L} - (k\eta)\tilde{\mathbf{L}})\mathbf{z}\|_p \leq k\|\tilde{\mathbf{L}}\mathbf{z}\|_p$ and therefore $\|\mathbf{L}\mathbf{z}\|_p \geq (\eta - 1)k\|\tilde{\mathbf{L}}\mathbf{z}\|_p$. □

LEMMA 5.12 *For all $\eta > 0$ and all integer vectors $\mathbf{z} \in \mathbb{Z}^k$*

$$\|\mathbf{L}\mathbf{z} - \mathbf{s}\|_p \leq (\eta + 1)k\|\tilde{\mathbf{L}}\mathbf{z} - \tilde{\mathbf{s}}\|_p,$$

where $\mathbf{L} = \lfloor (k\eta)L \rceil$ and $\mathbf{s} = \lfloor (k\eta)\mathbf{s} \rceil$ are the matrices obtained multiplying $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{s}}$ by $k\eta$ and rounding each entry to the closest integer.

Proof: By triangular inequality

$$\begin{aligned}
\|\mathbf{L}\mathbf{z}-\mathbf{s}\|_p &= \|((k\eta)\tilde{\mathbf{L}}\mathbf{z}-(k\eta)\tilde{\mathbf{s}})+(\mathbf{L}-(k\eta)\tilde{\mathbf{L}})\mathbf{z}-(\mathbf{s}-(k\eta)\tilde{\mathbf{s}})\|_p \\
&\le \|((k\eta)\tilde{\mathbf{L}}\mathbf{z}-(k\eta)\tilde{\mathbf{s}})\|_p+\|(\mathbf{L}-(k\eta)\tilde{\mathbf{L}})\mathbf{z}-(\mathbf{s}-(k\eta)\tilde{\mathbf{s}})\|_p \\
&= \eta k\|\tilde{\mathbf{L}}\mathbf{z}-(k\eta)\tilde{\mathbf{s}}\|_p+\|(\mathbf{L}-(k\eta)\tilde{\mathbf{L}})\mathbf{z}-(\mathbf{s}-(k\eta)\tilde{\mathbf{s}})\|_p.
\end{aligned}$$

Notice that all entries in $(\mathbf{L}-(k\eta)\tilde{\mathbf{L}})$ and $(\mathbf{s}-(k\eta)\tilde{\mathbf{s}})$ are at most $1/2$ in absolute value. Therefore

$$\|(\mathbf{L}-(k\eta)\tilde{\mathbf{L}})\mathbf{z}-(\mathbf{s}-(k\eta)\tilde{\mathbf{s}})\|_p^p \le \left(\frac{1}{2}\right)^p \left(\|\mathbf{z}\|_p^p+\left(\sum|z_i|+1\right)^p\right) < k^p\|\mathbf{z}\|_p^p.$$

Furthermore, $\|\tilde{\mathbf{L}}\mathbf{z}-\tilde{\mathbf{s}}\|_p \ge \|\mathbf{D}\mathbf{z}\|_p \ge \|\mathbf{z}\|_p$ because $\mathbf{D}$ is diagonal with all entries greater than 1. This proves that

$$\|(\mathbf{L}-(k\eta)\tilde{\mathbf{L}})\mathbf{z}-(\mathbf{s}-(k\eta)\tilde{\mathbf{s}})\|_p \le k\|\tilde{\mathbf{L}}\mathbf{z}-\tilde{\mathbf{s}}\|_p,$$

and therefore $\|\mathbf{L}\mathbf{z}-\mathbf{s}\|_p \le (\eta+1)\|\mathbf{L}\mathbf{z}-\mathbf{s}\|_p$. □

We can now prove Theorem 4.5. This is essentially a variant of Theorem 5.7 where all the numbers are integers.

Proof [of Theorem 4.5]: We show that for all $p \ge 1, \delta > 0$ and $\epsilon > 0$ the theorem is satisfied with

$$\gamma = \left(\frac{(1-\epsilon)^{1+1/p}}{(1+\epsilon)^{2+1/p}}\right)\cdot\sqrt[p]{2}.$$

Let c be as in Theorem 5.7. On input h, algorithm $\mathcal{A}$ computes $k = h^c$, and the first k odd primes $a_1, a_2, \ldots, a_k$. Let $\tilde{\mathbf{L}}$, $\tilde{\mathbf{s}}$, and $\tilde{r}$ be as defined in Theorem 5.7, and compute the approximations

$$\mathbf{L} = \lfloor (k/\epsilon)\tilde{\mathbf{L}}\rceil, \qquad \mathbf{s} = \lfloor (k/\epsilon)\tilde{\mathbf{s}}\rceil, \qquad r = \lceil (1+1/\epsilon)k\tilde{r}\rceil.$$

Let $\mathbf{z} \in \mathbb{Z}^k$ be a nonzero integer vector. We want to bound $\|\mathbf{L}\mathbf{z}\|_p$. We know from Theorem 5.7 that

$$\|\tilde{\mathbf{L}}\mathbf{z}\|_p > \sqrt[p]{2\frac{1-\epsilon}{1+\epsilon}}\tilde{r}. \tag{5.6}$$

Using Lemma 5.11 (with $\eta = 1/\epsilon$) and (5.6) we get

$$\begin{aligned}
\|\mathbf{L}\mathbf{z}\|_p &\ge \left(\frac{1}{\epsilon}-1\right)k\|\tilde{\mathbf{L}}\mathbf{z}\|_p \\
&> \left(\frac{(1-\epsilon)^{1+1/p}}{\epsilon(1+\epsilon)^{1/p}}\right)k\sqrt[p]{2}\cdot\tilde{r}.
\end{aligned} \tag{5.7}$$

Notice that r satisfies the bounds $r < (1+1/\epsilon)k\tilde{r}+1$ and $r > (1+1/\epsilon)$ because $k \geq 1$ and $\tilde{r} > 1$. Thus, we can bound $\tilde{r}$ as follows:

$$\begin{aligned}\tilde{r} &> \frac{r-1}{(1+1/\epsilon)k} \\ &= \frac{1-1/r}{(1+1/\epsilon)k}\cdot r \\ &> \frac{1-1/(1+1/\epsilon)}{(1+1/\epsilon)k}\cdot r \\ &= \frac{\epsilon}{(\epsilon+1)^2 k}\cdot r.\end{aligned} \tag{5.8}$$

Combining (5.7) and (5.8) we get

$$\|\mathbf{Lz}\|_p > \left(\frac{(1-\epsilon)^{1+1/p}}{\epsilon(1+\epsilon)^{1/p}}\right)\sqrt[p]{2}\frac{\epsilon}{(\epsilon+1)^2}r = \gamma r.$$

Now consider the sphere $\mathcal{B}(\mathbf{s}, r)$. By Theorem 5.7, for all sufficiently large h, with probability at least $1-2^{-h}$, the ball $\mathcal{B}(\tilde{\mathbf{s}}, \tilde{r})$ contains at least $h^{\delta h}$ lattice points of the form $\tilde{\mathbf{L}}\mathbf{z}$ where $\mathbf{z}$ is a 0-1 vector with exactly h ones. For each such point $\tilde{\mathbf{L}}\mathbf{z}$, we can use Lemma 5.12 (with $\eta = 1/\epsilon$) to bound the distance of $\mathbf{Lz}$ from $\mathbf{s}$ as follows:

$$\begin{aligned}\|\mathbf{Lz}-\mathbf{s}\|_p &\leq (1+1/\epsilon)k\|\tilde{\mathbf{L}}\mathbf{z}-\tilde{\mathbf{s}}\|_p \\ &\leq (1+1/\epsilon)k\tilde{r} \leq r.\end{aligned}$$

Therefore $\mathbf{Lz}$ belongs to the sphere $\mathcal{B}(\mathbf{s}, r)$. This proves that $\mathcal{B}(\mathbf{s}, r)$ also contains at least $h^{\delta h}$ lattice points of the desired form. □

4. Deterministic construction

The probabilistic construction of Theorem 4.5 is used in Chapter 4 to prove the NP-hardness of SVP under randomized reductions. Finding a similar deterministic construction would be useful to obtain an NP-hardness result for SVP under Karp reductions. The randomization in the proof of Theorem 4.5 comes from the fact that we do not know which intervals of the form $[b, b+b^\epsilon]$ (for small $\epsilon > 0$) contain square free smooth numbers. The problem is solved in Corollary 5.6 choosing b according to a certain easily samplable distribution. The intuition is that since there are many square free smooth numbers, then some intervals must contain many of them. In fact, if square free smooth numbers are distributed uniformly enough, than any interval $[b, b+b^\epsilon]$ is good. To date we dot not know how to prove that square free smooth numbers are distributed uniformly enough (see Section 5 for further discussion of this issue), however, it seems reasonable to conjecture that for any $\epsilon > 0$ there exists a d such that for all large enough n, the interval $[n, n + n^\epsilon]$ contains an (odd) integer which is square free and $(\log^d n)$-smooth. This is Conjecture 1 from Chapter 4. Using the conjecture, we can prove Theorem 4.9.

Proof [of Theorem 4.9]: For simplicity, we show how to build real $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{s}}$. Integer $\mathbf{L}$ and $\mathbf{s}$ can be easily obtained using Lemma 5.11 and Lemma 5.12 as explained in Section 3.

Let ϵ be a positive real between 0 and 1. Let d be an integer (whose existence is guaranteed by Conjecture 1) such that for all large enough n there exists a $(\log^d n)$-smooth square free (odd) integer in the interval $[n, n + n^{\epsilon/2}]$. Let $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{s}}$ be as defined in (5.1) and (5.4) with $k = h^{d+1} + h$, $a_1, \ldots, a_k$ the first k (odd) prime numbers, $b = a_k^{2h/\epsilon}$ and $\alpha = b^{1-\epsilon}$. Notice that since k is polynomial in h, the sequence $a_1, \ldots, a_k$ can be generated in deterministic polynomial time.

From Lemma 5.3 we know that for all nonzero vectors $\mathbf{z} \in \mathbb{Z}^h$,

$$\|\tilde{\mathbf{L}}\mathbf{z}\|_p \geq \sqrt[p]{2(1-\epsilon)\ln b}.$$

We now show that for all $\mathbf{x} \in \{0,1\}^h$ there exists a $\mathbf{y} \in \mathbb{Z}^{h^{d+1}}$ such that

$$\left\| \tilde{\mathbf{L}} \begin{bmatrix} \mathbf{y} \\ \mathbf{x} \end{bmatrix} - \tilde{\mathbf{s}} \right\|_p < \sqrt[p]{\ln b + 2}. \tag{5.9}$$

where $\mathbf{z} = [\mathbf{y}^T, \mathbf{x}^T]^T$. Let $g_\mathbf{x} = \prod_{i=1}^h a_{h^{d+1}+i}^{x_i}$. Notice that

$$\frac{b}{g_\mathbf{x}} > \frac{b}{a_k^h} = a_k^{(\frac{2}{\epsilon}-1)h} > 2^h.$$

In particular, as h gets arbitrarily large, also $(b/g_\mathbf{x})$ gets arbitrarily large. So, for all sufficiently large h, there exists a $\log^d(b/g_\mathbf{x})$-smooth square free odd integer in the interval

$$[b/g_\mathbf{x}, (b/g_\mathbf{x}) + (b/g_\mathbf{x})^{\epsilon/2}]. \tag{5.10}$$

Notice that

$$\log^d(b/g_\mathbf{x}) \leq \log^d(b) = O(h \log h)^d < h^{d+1}.$$

So, (5.10) contains an square free h^{d+1}-smooth odd number, i.e., the product of a subset of $a_1, \ldots, a_{h^{d+1}}$. Let $g_\mathbf{y} = \prod_{i=1}^{h^{d+1}} a_i^{y_i}$ be such number, where $\mathbf{y} \in \{0,1\}^{h^{d+1}}$. Multiplying $g_\mathbf{y}$ and (5.10) by $g_\mathbf{x}$ we get

$$g_\mathbf{z} = g_\mathbf{x} g_\mathbf{y} \in [b, b + b^{\epsilon/2} g_\mathbf{x}]$$

where $g_{\mathbf{z}} = \prod_{i=1}^{h^{d+1}+h} = g_{\mathbf{x}}g_{\mathbf{y}}$. Finally, we observe that $g_{\mathbf{x}} \leq a_k^h = b^{\epsilon/2}$ and therefore $g_{\mathbf{z}} = g_{\mathbf{x}}g_{\mathbf{y}}$ belongs to the interval $[b, b+b^{\epsilon}]$. By Lemma 5.4, lattice vector $\widetilde{\mathbf{L}}\mathbf{z}$ satisfies (5.9). The theorem follows setting r to (a rational approximation of) $\sqrt[p]{\ln b + 2}$, and choosing ϵ small enough so that $\sqrt[p]{2(1-\epsilon)\ln b}/r > \gamma$. □

5. Notes

The material presented in this chapter is from (Micciancio, 1998), and part of it also appeared in (Micciancio, 2001d). The upper bounds in Section 1 were first proved in (Rankin, 1955) for the special case of spherical codes (i.e., a sphere packing problem with the additional constraint that all points must be at the same distance from the origin). If we allow arbitrary sets of points with minimum distance λ, an exponential lower bounds for any $\lambda/\rho < \sqrt{2}$ is already implicit in Gilbert bound for binary codes (Gilbert, 1952). Non constructive proofs for spherical codes were given in (Shannon, 1959) and (Wyner, 1965). However, the points generated by these constructions do not form a lattice. We remark that the lower bounds in (Shannon, 1959; Wyner, 1965) show that it is possible to pack $2^{\alpha n}$ points, where α is a constant that depends only on $\rho > \sqrt{2}$, while our construction succeeds in packing only $2^{n^{\alpha}}$ points. An interesting question is whether our construction is asymptotically optimal for lattice packings, i.e., if $2^{n^{\Omega(1)}}$ is the best we can do, or even for lattices one can have $2^{\Omega(n)}$ points inside a small ball.

Variants of the lattice studied in Section 2 have appeared in the computer science literature in various places. A version of the lattice (with $p = 1$) was first used by (Schnorr, 1993) to heuristically factor integers by reduction to SVP. (Adleman, 1995) used a similar lattice (with $p = 2$) to reduce factoring to SVP under some unproven number theoretic assumptions. Finally, an extended version of Adleman's lattice is used in (Ajtai, 1998) to prove the NP-hardness of SVP. The proof of (Micciancio, 2001d) (as presented in Chapter 4), although inspired by (Ajtai, 1996), goes back to the original lattice of Schnorr and Adleman, considerably simplifying Ajtai's proof. The connection between the Schnorr-Adleman lattice and sphere packing problems is explicitly established for the first time in (Micciancio, 1998; Micciancio, 2001d). This more geometric interpretation of the lattice allowed to translate the techniques of (Micciancio, 1998) to other areas, and prove analogous results for coding problems (Dumer et al., 1999).

Chapter 6

LOW-DEGREE HYPERGRAPHS

The goal of this chapter is to prove Theorem 4.6. The theorem states that if $\mathcal{Z} \subset \{0,1\}^k$ is a set of binary vectors, each containing exactly h ones, and $|\mathcal{Z}| \geq h!k^{4\sqrt{hn}/\epsilon}$, then there exists a matrix $\mathbf{T} \in \{0,1\}^{n\times k}$ such that $\{0,1\}^k \subseteq \mathbf{T}(\mathcal{Z})$, where $\mathbf{T}(\mathcal{Z})$ denotes the set $\{\mathbf{Tz} : \mathbf{z} \in \mathcal{Z}\}$. In other words, for every $\mathbf{x} \in \{0,1\}^n$ there exists a $\mathbf{z} \in \mathcal{Z}$ satisfying $\mathbf{x} = \mathbf{Tz}$. Moreover, Theorem 4.6 states that if $\mathbf{T} \in \{0,1\}^{n\times k}$ is chosen at random setting each entry to 1 independently with probability $p = \epsilon/(4hn)$, then $\{0,1\}^k \subseteq \mathbf{T}(\mathcal{Z})$ with high probability (namely, probability at least $1 - 6\epsilon$). In Chapter 4, Theorem 4.6 is used to prove the NP-hardness of approximating the shortest vector problem under RUR-reductions. However, the theorem has a purely combinatorial interpretation and it is better understood if reformulated in terms of hypergraphs, without any reference to integer lattices or matrices. A *hypergraph* is a pair $(V, \mathcal{Z})$, where V is a finite set of *vertices* and $\mathcal{Z}$ is a collection of subsets of V, called *hyperedges*. If all the elements of $\mathcal{Z}$ have the same size, then we say that $(V, \mathcal{Z})$ is *regular*, and the common size of all hyperedges is called the *degree* of the hypergraph.

Theorem 4.6 can be reformulated in terms of regular hypergraphs as follows. Let $(V, \mathcal{Z})$ be an h-regular hypergraph, and let $\mathcal{T} = (T_1, \ldots, T_n)$ be a sequence of subsets of V chosen at random including each element of V in T_i independently with probability $p = \epsilon/(4hn)$. For any subset of vertices $U \subseteq V$, let

$$\mathcal{T}(U) = (|T_1 \cap U|, |T_2 \cap U|, \ldots, |T_n \cap U|)$$

and define $\mathcal{T}(\mathcal{Z}) = \{\mathcal{T}(U) : U \in \mathcal{Z}\}$. We want to prove that if $|\mathcal{Z}| > h!|V|^{4\sqrt{hn}/\epsilon}$, then $\{0,1\}^n \subseteq \mathcal{T}(\mathcal{Z})$ with probability at least $1 - 6\epsilon$.

D. Micciancio et al., *Complexity of Lattice Problems*

The correspondence between the matrix and hypergraph formulation is immediate: identify the hyperedges with the corresponding characteristic vectors in $\{0,1\}^{|V|}$ and the sequence $\mathcal{T} = (T_1, \ldots, T_n)$ with a matrix $\mathbf{T} \in \{0,1\}^{n\times|V|}$ whose rows are the characteristic vectors of the sets T_i. Then $\mathcal{T}(U) = \mathbf{Tu}$ where $\mathbf{u}$ is the characteristic vector of set U. (Notice that for any two vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^{|V|}$, the scalar product $\mathbf{x}\cdot\mathbf{y} = \sum_{i=1}^{|V|} x_i y_i$ equals the size of the intersection of the corresponding sets.) With slight abuse of notation, in the rest of the chapter we will use $\mathbf{T}$ to denote either a boolean matrix or the corresponding sequence of sets.

It can be proved that for any (not necessarily regular) hypergraph $(V, \mathcal{Z})$, if $|\mathcal{Z}| > |V|^n$, then there exists a solution consisting of singleton sets $|T_i| = 1$. This is essentially a combinatorial result proved, independently, by Sauer, Perles and Shelah, and, in a slightly weaker form, by Vapnik and Chervonenkis, which is usually referred to as *Sauer's Lemma.* The proof of this result is relatively simple, but not constructive: it only asserts that $\mathbf{T}$ exists, without giving any effective (even probabilistic) way to find it. Theorem 4.6 can be regarded as a effective probabilistic variant of Sauer's Lemma.

The proof of Theorem 4.6 is divided in two stages. We first prove a weaker result: we show that every vector $\mathbf{x} \in \{0,1\}^n$ belongs to $\mathbf{T}(\mathcal{Z})$ with very high probability. Then, we prove a stronger property as stated in Theorem 4.6. The difference between the weak and strong version of the theorem is in the order of quantification. While the theorem in its strong form asserts that with high probability $\mathbf{T}$ is good for all target vectors $\mathbf{x}$, the weak version only says that for any fixed target vector $\mathbf{x}$, matrix $\mathbf{T}$ is good with high probability.

The weak version of the theorem is proved in Section 2 using a relatively simple argument based on Chebychev inequality. Then, in Section 3 we show that the strong version of the theorem can be easily derived from the weak one using ideas similar to those arising in the proof of (the standard non constructive version of) Sauer's Lemma. So, we begin in Section 1 by presenting a simple proof of Sauer's Lemma. Even if this result is not used in the rest of the book, the proof in Section 1 gives a first exposure to the ideas that will later be used in Section 3 to prove Theorem 4.6.

1. Sauer's Lemma

In this section we present a proof of Sauer's Lemma,. This combinatorial result is usually stated in terms of the Vapnik-Chervonenkis dimension (VC-dimension) of a range space. In order to avoid the introduction of new concepts, we reformulate Sauer's Lemma in terms of the sets $T_1, \ldots, T_k$ and $\mathcal{Z}$. Sauer's result is essentially a solution to our combinatorial problem with the restriction that the T_i's must be singleton sets, i.e., sets containing exactly one element.

When the T_i's are singleton sets, the linear operation associated to $\mathbf{T}$ is more easily described by the projection onto some set $G \subseteq V$ as follows. For any hypergraph $(V, \mathcal{Z})$ and for any subset of nodes $G \subseteq V$, define the restriction of $\mathcal{Z}$ to G by

$$\mathcal{Z}|_G = \{A \cap G : A \in \mathcal{Z}\}.$$

Notice that for every set $G \subseteq V$, the following two conditions are equivalent:

- $\mathcal{Z}|_G = \wp(G)$ is the power set of G,
- $\{0,1\}^G \subseteq \mathbf{T}(\mathcal{Z})$ where $\mathbf{T} = (\{a\})_{a \in G}$ is a sequence of $|G|$ sets, each containing a single element of G.

LEMMA 6.1 (SAUER'S LEMMA) *Let V be a set of size k and $\mathcal{Z}$ be a collection of subsets of V. Let*

$$[k, n] = \sum_{i=0}^{n} \binom{k}{i}$$

be the number of subsets of V of size at most n. For all n, if $|\mathcal{Z}| \geq [k, n]$ then there exists a set G of size n such that $\mathcal{Z}|_G = \wp(G)$.

Proof: The proof is by induction on $k + n$. If $k = n = 0$ the assertion is trivially true. Notice that $[k, n] = [k-1, n] + [k-1, n-1]$. Assume that the lemma holds for $k-1, n$ and $k-1, n-1$, and let's prove it for k, n. Let $|V| = k$ and $|\mathcal{Z}| \geq [k, n]$. Pick an element a from V and define $U = V \setminus \{a\}$ and the following two collections of subsets of U:

$$\mathcal{Z}_0 = \{A \subseteq U : A \in \mathcal{Z}\}$$

$$\mathcal{Z}_1 = \{A \subseteq U : A \cup \{a\} \in \mathcal{Z}\}.$$

Notice that $|U| = k - 1$ and

$$\begin{aligned} |\mathcal{Z}_0 \cup \mathcal{Z}_1| + |\mathcal{Z}_0 \cap \mathcal{Z}_1| &= |\mathcal{Z}_0| + |\mathcal{Z}_1| \\ &= |\mathcal{Z}| \\ &\geq [k, n] \\ &= [k-1, n] + [k-1, n-1]. \end{aligned}$$

Therefore, either $|\mathcal{Z}_0 \cup \mathcal{Z}_1| \geq [k-1,n]$ or $|\mathcal{Z}_0 \cap \mathcal{Z}_1| \geq [k-1,n-1]$. We deal with the two cases separately:

- if $|\mathcal{Z}_0 \cup \mathcal{Z}_1| \geq [k-1,n]$, then by inductive hypothesis there exist a set $G \subseteq U \subset V$ of size $|G| = n$ such that $(\mathcal{Z}_0 \cup \mathcal{Z}_1)|_G = \wp(G)$. Since $a \notin G$, $\mathcal{Z}|_G = (\mathcal{Z}_0 \cup \mathcal{Z}_1)|_G = \wp(G)$.

- if $|\mathcal{Z}_0 \cap \mathcal{Z}_1| \geq [k-1,n-1]$, by inductive hypothesis, there exists a set $G' \subseteq U \subset V$ of size $|G'| = n-1$. such that $(\mathcal{Z}_0 \cap \mathcal{Z}_1)|_{G'} = \wp(G')$. Let $G = G' \cup \{a\}$. We now show that $\mathcal{Z}|_G = \wp(G)$. The inclusion $\mathcal{Z}|_G \subseteq \wp(G)$ is obvious. So, let us prove $\wp(G) \subseteq \mathcal{Z}|_G$. Let $A \in \wp(G)$ be any subset of G. Notice that $A \setminus \{a\}$ belongs to both $\mathcal{Z}_0|_{G'}$ and $\mathcal{Z}_1|_{G'}$. Therefore $A \setminus \{a\} \in \mathcal{Z}_G$ and $A \cup \{a\} \in \mathcal{Z}_G$. Since A equals either $A \setminus \{a\}$ or $A \cup \{a\}$, it follows that $A \in \mathcal{Z}_G$. □

Since $[k,n] < k^n$, one immediately gets the following corollary. We remark that the corollary is already enough to prove the NP-hardness of SVP under nonuniform reductions. (See Corollary 4.8.)

COROLLARY 6.2 *Let $\mathcal{Z} \subset \{0,1\}^k$ be a collection of boolean vectors. If $|\mathcal{Z}| \geq k^n$ then there exists a matrix $\mathbf{T} \in \{0,1\}^{n \times k}$ such that $\{0,1\}^n \subseteq \mathbf{T}(\mathcal{Z})$.*

Observe that the bound in Sauer's Lemma is tight: if $\mathcal{Z}$ is the set of all subsets of V of size n or less, then $|\mathcal{Z}| = [k,n]$ and any set G satisfying the assertion in the lemma has size at most n. The proof of the lemma suggests a possible way to find the set G: select the elements of V one at a time. For each $a \in V$, if there are a lot of subsets A such that both $A \setminus \{a\}$ and $A \cup \{a\}$ belong to $\mathcal{Z}$, then include a in G, otherwise discard it, project $\mathcal{Z}$ onto $V \setminus \{a\}$ and go on to the next element. The problem is that the step of deciding whether a given $a \in V$ is good or bad may not be effective. Notice that a single element might belong to all sets in $\mathcal{Z}$ (or none of them), and still $|\mathcal{Z}|$ be quite large, and selecting such an element would be disastrous. We show in a later section that when $\mathcal{Z}$ is very large ($|\mathcal{Z}| \approx 2^k$), then G can be chosen at random and a probabilistic analogue of Sauer's Lemma holds. But first one has to get rid of the bad elements. This is accomplished in the proof of the weak version of the theorem.

2. Weak probabilistic construction

In this section we prove a weaker version of Theorem 4.6: we show for every vector $\mathbf{x} \in \{0,1\}^n$, $\mathbf{x} \in \mathbf{T}(\mathcal{Z})$ with high probability. (See Theorem 6.8 for the exact statement.) Consider the target vector $\mathbf{x}$

as fixed. We want to bound the probability that $\mathbf{T}(U) \neq \mathbf{x}$ for all $U \in \mathcal{Z}$. Since the set $\mathcal{Z}$ is very big, the expected number of $U \in \mathcal{Z}$ such that $\mathbf{T}(U) = \mathbf{x}$ is also very high. Unfortunately, this is not sufficient to conclude that with high probability there exists a $U \in \mathcal{Z}$ such that $\mathbf{T}(U) = \mathbf{x}$, because the events $\mathbf{T}(U) = \mathbf{x}$ (indexed by the hyperedges $U \in \mathcal{Z}$) might be strongly correlated. Notice that if U and U' are disjoint (i.e., $U \cap U' = \emptyset$), then the corresponding events are independent. In fact the size of the intersection $|U \cap U'|$ is a good measure of the correlation between the events $\mathbf{T}(U) = \mathbf{x}$ and $\mathbf{T}(U') = \mathbf{x}$. Notice that if $|\mathcal{Z}|$ is big, then many hyperedges in $\mathcal{Z}$ intersect because there cannot be more than n/h mutually disjoint hyperedges. However, one can still hope that for most of the pairs $U, U' \in \mathcal{Z}$, the intersection $U \cap U'$ is very small. This is not necessarily true for any hypergraph $\mathcal{Z}$, but one can show that if $\mathcal{Z}$ is sufficiently large, then it must "contain" (in some precise sense to be specified) a large hypergraph with this small intersection property.

The proof of the theorem is divided in three steps:

1 We first show that the probability that $\mathbf{x} \notin \mathbf{T}(\mathcal{Z})$ can be bounded by the expectation
$$\operatorname*{Exp}_{R}[e^{\vartheta R} - 1], \tag{6.1}$$
where ϑ is a small positive real, and $R = |U \cap U'|$ is the random variable defined as the size of the intersection of two randomly chosen hyperedges $U, U' \in \mathcal{Z}$.

2 Then, we show that $\mathcal{Z}$ "contains" a hypergraph such that the intersection of two randomly selected hyperedges is very small with high probability.

3 Finally, we prove the weak version of the theorem applying the bound (6.1) to this hypergraph contained in $\mathcal{Z}$.

Each of the above steps is described in the following subsections.

2.1 The exponential bound

We start by computing the probability that $\mathbf{T}(U) = \mathbf{x}$ for some fixed set U. In the next lemma we prove a more general statement concerning the probability that two events $\mathbf{T}(U) = \mathbf{x}$ and $\mathbf{T}(U') = \mathbf{x}$ are simultaneously satisfied and relate it to the size of the intersection $r = |U \cap U'|$ of the two sets U, U'.

LEMMA 6.3 *Let* $\mathbf{x} \in \{0,1\}^n$ *be any boolean vector,* $U, U' \subset V$ *be two sets of size* d *and let* $\mathbf{T} = (T_1, \ldots, T_n)$ *(where* $T_i \subseteq V$ *for all* $i = 1, \ldots, n$*) be chosen at random including each element of* V *in* T_i *independently* *with probability p. Then, the probability (over the choice of $\mathbf{T}$) that both $\mathbf{T}(U)$ and $\mathbf{T}(U')$ equal $\mathbf{x}$ is*

$$\Phi(r) = (1-p)^{(2d-r)n}\left[\frac{pr}{1-p} + \left(\frac{p(d-r)}{1-p}\right)^2\right]^{\|\mathbf{x}\|_1},$$

where $r = |U \cap U'|$, and $\|\mathbf{x}\|_1$ is the number of 1's in vector $\mathbf{x}$.

Proof: Since the sets $T_1, \ldots, T_n$ are chosen independently,

$$\Pr_{\mathbf{T}}\{\mathbf{T}(U) = \mathbf{T}(U') = \mathbf{x}\} = \prod_{i=1}^{n} \Pr_{T_i}\{|T_i \cap U| = |T_i \cap U'| = x_i\}.$$

We prove that for all $i = 1, \ldots, n$,

$$\Pr_{T_i}\{|T_i \cap U| = |T_i \cap U'| = x_i\} = (1-p)^{(2d-r)}\left[\frac{pr}{1-p} + \left(\frac{p(d-r)}{1-p}\right)^2\right]^{x_i}.$$

First consider the case $x_i = 0$ and compute the probability (over the choice of T_i) that $|T_i \cap U| = |T_i \cap U'| = 0$. This is true if and only if none of the elements of $U \cup U'$ belongs to T_i, so the probability is

$$\Pr_{T_i}\left\{|T_i \cap U| = |T_i \cap U'| = 0\right\} = (1-p)^{|U \cup U'|} = (1-p)^{2d-r}.$$

Now consider the case $x_i = 1$ and compute the probability (over the choice of T_i) that $|T_i \cap U| = |T_i \cap U'| = 1$. This is true if and only if either (1) T_i contains one element of $U \cap U'$ and no other element of $U \cup U'$, or (2) T_i contains one element of $U \setminus U'$, one element of $U' \setminus U$, and no other element of $U \cup U'$. Event (1) has probability

$$|U \cap U'| \cdot p(1-p)^{|U \cup U'|-1} = (1-p)^{2d-r}\left(\frac{pr}{1-p}\right)$$

while event (2) has probability

$$|U \setminus U'| \cdot |U' \setminus U| \cdot p^2(1-p)^{|U \cup U'|-2} = (1-p)^{2d-r}\left(\frac{p(d-r)}{1-p}\right)^2.$$

Adding up the two probabilities, we get

$$\Pr_{T_i}\left\{|T_i \cap U| = |T_i \cap U'| = 1\right\} = (1-p)^{(2d-r)}\left(\frac{pr}{1-p} + \left(\frac{p(d-r)}{1-p}\right)^2\right). \square$$

By choosing $U = U'$ in the previous lemma one gets the following corollary.

COROLLARY 6.4 *Let $\mathbf{x} \in \{0,1\}^n$ be a boolean vector, $U \subseteq V$ a subset of size d, and choose $\mathbf{T} = (T_1, \ldots, T_n)$ at random as in Lemma 6.3. Then,*

$$\Pr_{\mathbf{T}}\{\mathbf{T}(U) = \mathbf{x}\} = \Phi(d) = (1-p)^{dn}\left(\frac{pd}{1-p}\right)^{\|\mathbf{x}\|_1}.$$

Notice that when $U \cap U' = \emptyset$,

$$\begin{aligned}\Pr\{\mathbf{T}(U) = \mathbf{T}(U') = \mathbf{x}\} &= \Phi(0) = \Phi(d)^2 \\ &= \Pr\{\mathbf{T}(U) = \mathbf{x}\}\Pr\{\mathbf{T}(U') = \mathbf{x}\},\end{aligned}$$

i.e., the events $\mathbf{T}(U) = \mathbf{x}$ and $\mathbf{T}(U') = \mathbf{x}$ are independent. We can now prove the following proposition.

PROPOSITION 6.5 *Let $(V, \mathcal{Z})$ be a d-regular hypergraph and let $\mathbf{T}$ be a sequence of subsets of vertices $(T_1, \ldots, T_n)$ chosen at random including each element of V in T_i independently with probability p. Then, for each $\mathbf{x} \in \{0,1\}^n$ the probability (over the choice of $\mathbf{T}$) that $\mathbf{x} \notin \mathbf{T}(\mathcal{Z})$ is at most $Exp_R[e^{\vartheta R}] - 1$, where $\vartheta = \frac{np}{1-p} + \frac{n}{pd^2}$ and $R = |U \cap U'|$ is the random variable defined as the size of the intersection of two randomly chosen hyperedges $U, U' \in \mathcal{Z}$.*

Proof: Fix some vector $\mathbf{x} \in \{0,1\}^n$ and choose $\mathbf{T}$ at random as specified in the proposition. For all $U \in \mathcal{Z}$, let X_U be the indicator random variable

$$X_U = \begin{cases} 1 & \text{if } \mathbf{T}(U) = \mathbf{x}, \\ 0 & \text{otherwise.} \end{cases}$$

Define the random variable $X = \sum_{U \in \mathcal{Z}} X_U$. Notice that $X = 0$ if and only if $\mathbf{x} \notin \mathbf{T}(\mathcal{Z})$. Moreover, if $X = 0$ then $|X - \mathrm{Exp}[X]| \geq \mathrm{Exp}[X]$. Using Chebyshev's inequality we get the following bound:

$$\begin{aligned}\Pr\{\mathbf{x} \notin \mathbf{T}(\mathcal{Z})\} &= \Pr\{X = 0\} \\ &\leq \Pr\{|X - \mathrm{Exp}[X]| \geq \mathrm{Exp}[X]\} \\ &\leq \frac{\mathrm{Var}[X]}{\mathrm{Exp}[X]^2} = \frac{\mathrm{Exp}[X^2]}{\mathrm{Exp}[X]^2} - 1.\end{aligned}$$

So, let us compute the moments $\mathrm{Exp}[X]$ and $\mathrm{Exp}[X^2]$. For the first moment we have

$$\mathop{\mathrm{Exp}}_{\mathbf{T}}[X] = \sum_{U \in \mathcal{Z}} \mathop{\mathrm{Exp}}_{\mathbf{T}}[X_U] = \sum_{U \in \mathcal{Z}} \Pr_{\mathbf{T}}\{\mathbf{T}(U) = \mathbf{x}\} = |\mathcal{Z}| \cdot \Phi(d),$$

and for the second one

$$\begin{aligned}
\operatorname*{Exp}_{\mathbf{T}}[X^2] &= \operatorname*{Exp}_{\mathbf{T}}\left[\left(\sum_{U\in\mathcal{Z}} X_U\right)^2\right]\\
&= \operatorname*{Exp}_{\mathbf{T}}\left[\sum_{U,U'\in\mathcal{Z}} X_U\cdot X_{U'}\right]\\
&= \sum_{U,U'\in\mathcal{Z}} \Pr_{\mathbf{T}}\{\mathbf{T}(U)=\mathbf{T}(U')=\mathbf{x}\}\\
&= |\mathcal{Z}|^2\cdot\operatorname*{Exp}_{R}[\Phi(R)],
\end{aligned}$$

where $R = |U \cap U'|$ is the size of two randomly chosen $U, U' \in \mathcal{Z}$. Therefore,

$$\begin{aligned}
\Pr_{\mathbf{T}}\{\mathbf{x}\notin\mathbf{T}(\mathcal{Z})\} &= \frac{\mathrm{Exp}_R[\Phi(R)]}{\Phi(d)^2}-1\\
&= \operatorname*{Exp}_{R}\left[(1-p)^{-nR}\left(\frac{(1-p)R}{pd^2}+\left(1-\frac{R}{d}\right)^2\right)^{\|\mathbf{x}\|_1}\right]-1\\
&< \operatorname*{Exp}_{R}\left[\left(1+\frac{p}{1-p}\right)^{nR}\left(\frac{R}{pd^2}+1\right)^{n}\right]-1\\
&< \operatorname*{Exp}_{R}\left[e^{\frac{pnR}{1-p}}e^{\frac{nR}{pd^2}}\right]-1\\
&= \operatorname*{Exp}_{R}[e^{\vartheta R}-1],
\end{aligned}$$

where $\vartheta = \frac{np}{1-p}+\frac{n}{pd^2}$. □

2.2 Well spread hypergraphs

In the previous section we showed that the probability that $\mathbf{x} \notin \mathbf{T}(\mathcal{Z})$ is at most $\mathrm{Exp}_R[e^{\vartheta R}]-1$. Obviously, the bound is interesting only when $\mathrm{Exp}_R[e^{\vartheta R}] < 2$. Notice that this can be true only if

$$\Pr_R\{R=r\} < e^{-\vartheta r}$$

for all but a single value of r. Therefore the probability $\Pr_R\{R = r\}$ must decrease exponentially fast in r. This is not necessarily true for any low degree regular hypergraph $\mathcal{Z}$. In this section we show that if $\mathcal{Z}$ is sufficiently large, then $\mathcal{Z}$ must "contain" a hypergraph such that

$$\Pr_R\{R=r\} \leq 1/r!.$$

More precisely we show that $\mathcal{Z}$ contains a hypergraph satisfying the following property.

DEFINITION 6.1 *Let $(V, \mathcal{Z})$ be a d-regular hypergraph. $\mathcal{Z}$ is* well spread *if for all $W \subseteq V$ of size at most d, the fraction of hyperedges containing W is at most*

$$\frac{|\{U \in \mathcal{Z} : W \subseteq U\}|}{|\mathcal{Z}|} \leq \frac{1}{d(d-1)\cdots(d-|W|+1)} = \frac{(d-|W|)!}{d!}.$$

well spread hypergraphs have the important property that the size of the intersection of two randomly selected hyperedges is small with very high probability, as shown in the next lemma.

LEMMA 6.6 *Let $(V, \mathcal{Z})$ a regular well spread hypergraph. Choose U and U' in $\mathcal{Z}$ independently and uniformly at random and let $R = |U \cap U'|$. For all $r > 0$,*

$$\Pr_R\{R \geq r\} < \frac{1}{r!}.$$

Proof: Let d be the degree of the hypergraph. We prove that for any fixed set U of size d, the probability that $|U \cap U'| \geq r$ when U' is chosen at random from $\mathcal{Z}$ is at most $\frac{1}{r!}$. If $|U \cap U'| \geq r$ then U' contains a subset of U of size r. Therefore, by union bound,

$$\begin{aligned}\Pr_{U' \in \mathcal{Z}}\{|U \cap U'| \geq r\} &\leq \sum_{W \in \binom{U}{r}} \Pr_{U' \in \mathcal{Z}}\{W \subseteq U'\} \\ &= \sum_{W \in \binom{U}{r}} \frac{|\{U' \in \mathcal{Z} : W \subseteq U'\}|}{|\mathcal{Z}|},\end{aligned}$$

where $\binom{U}{r}$ denotes the set of all the size r subsets of U. Since $\mathcal{Z}$ is well spread, the fraction $|\{U' \in \mathcal{Z} : W \subseteq U'\}|/|\mathcal{Z}|$ is at most $\frac{(d-r)!}{d!}$, which substituted in the previous expression, gives

$$\Pr_{U' \in \mathcal{Z}}\{|U \cap U'| \geq r\} \leq \binom{d}{r}\frac{(d-r)!}{d!} = \frac{1}{r!}. \qquad \square$$

We now show how to find well spread hypergraphs "inside" any sufficiently big regular hypergraph. For any subset $W \subseteq V$, define the induced hypergraph

$$\mathcal{Z}_W = \{U \subseteq V \setminus W : U \cup W \in \mathcal{Z}\}.$$

In other words, $\mathcal{Z}_W$ is the set of hyperedges containing W, with the nodes in W removed. Notice the following basic facts:

1 hypergraph $\mathcal{Z}$ is well spread if for every set W of size at most d, $|\mathcal{Z}_W| \leq \frac{(d-|W|)!}{d!}|\mathcal{Z}|$.

2 $\mathcal{Z}_W$ is d'-regular with $d' = d - |W|$.

3 If $W = \emptyset$ then $\mathcal{Z}_W = \mathcal{Z}$.

4 $(\mathcal{Z}_W)_U = \mathcal{Z}_{W \cup U}$ if $W \cap U = \emptyset$, and $(\mathcal{Z}_W)_U = \emptyset$ otherwise.

5 If $|W| > d$ then $\mathcal{Z}_W = \emptyset$.

In the following lemma we prove that for any regular hypergraph $\mathcal{Z}$, there exists a set W such that $\mathcal{Z}_W$ is well spread.

LEMMA 6.7 *Let $(V, \mathcal{Z})$ be an h-regular hypergraph. Then there exists a set $W \subset V$ such that $(V, \mathcal{Z}_W)$ is well spread and $|\mathcal{Z}_W| > |\mathcal{Z}|/h!$.*

Proof: If $(V, \mathcal{Z})$ is well spread, let $W = \emptyset$ and the statement is obviously true. Otherwise, there exists some set W of size at most h such that $|\mathcal{Z}_W| > \frac{(h-|W|)!}{h!} \cdot |\mathcal{Z}|$. Let W be maximal (with respect to the set inclusion ordering relation) among these sets. Obviously, $|\mathcal{Z}_W| > |\mathcal{Z}|/h!$. Notice that $\mathcal{Z}_W$ is d-regular, with $d = h - |W|$. We prove that $(V, \mathcal{Z}_W)$ is well spread. Let U be a subset of V of size at most d. There are three cases:

1 If $U \cap W \neq \emptyset$ then $|(\mathcal{Z}_W)_U| = 0 \leq \frac{(d-|U|)!}{d!} \cdot |\mathcal{Z}_W|$.

2 If $U = \emptyset$, then $|(\mathcal{Z}_W)_U| = |\mathcal{Z}_W| = \frac{d!}{d!} \cdot |\mathcal{Z}_W|$.

3 Finally assume $U \neq \emptyset$ and $U \cap W = \emptyset$. By the maximality of W we have

$$\begin{aligned}
|(\mathcal{Z}_W)_U| &= |\mathcal{Z}_{U \cup W}| \\
&\leq \frac{(h - |U \cup W|)!}{h!}|\mathcal{Z}| \\
&= \frac{(d - |U|)!}{d!}\frac{(h - |W|)!}{h!}|\mathcal{Z}| \\
&< \frac{(d - |U|)!}{d!}|\mathcal{Z}_W|. \qquad \square
\end{aligned}$$

2.3 Proof of the weak theorem

We now combine the tools developed in the previous sections to prove the following theorem.

THEOREM 6.8 *For every sufficiently small constant $\epsilon > 0$, positive integer n and h-regular hypergraph $(V, \mathcal{Z})$ of size $|\mathcal{Z}| > h!|V|^{\sqrt{h}n/\epsilon}$ the following holds. Choose $\mathbf{T} = (T_1, \ldots, T_n)$ (where $T_i \subseteq V$ for all $i = 1, \ldots, n$) including each element of V in T_i independently at random with probability $p = \epsilon/(hn)$. Then, for every $\mathbf{x} \in \{0,1\}^n$,*

$$\Pr\{\mathbf{x} \in \mathbf{T}(\mathcal{Z})\} > 1 - 5\epsilon.$$

Proof: From Lemma 6.7, there exists a subset $W \subset V$ such that $(V, \mathcal{Z}_W)$ is well spread and $|\mathcal{Z}_W| \geq |\mathcal{Z}|/h! > |V|^{\sqrt{h}n/\epsilon}$. Choose $\mathbf{T}$ at random as specified in the theorem. Let F be the event that none of the elements of W are included in any set T_i. Notice that $\Pr\{\neg F\} \leq |W|np \leq hnp = \epsilon$. Notice also that

$$\Pr_{\mathbf{T}}\{\mathbf{x} \notin \mathbf{T}(\mathcal{Z}) \mid F\} \leq \Pr_{\mathbf{T}}\{\mathbf{x} \notin \mathbf{T}(\mathcal{Z}_W)\}$$

Let d be the degree of $\mathcal{Z}_W$. Since $|\mathcal{Z}_W| \leq \binom{|V|}{d} < |V|^d$ and $|\mathcal{Z}_W| > |V|^{\sqrt{h}n/\epsilon}$, hypergraph $\mathcal{Z}_W$ has degree at least $d > \sqrt{h}n/\epsilon$.

Applying Proposition 6.5 to d-regular hypergraph $\mathcal{Z}_W$, the probability (over the choice of $\mathbf{T}$) that $\mathbf{x} \notin \mathbf{T}(\mathcal{Z}_W)$ is at most $\mathrm{Exp}_R[e^{\vartheta R}] - 1$, where R is the size of the intersection of two random elements in $\mathcal{Z}_W$ and

$$\begin{aligned}
\vartheta &= \frac{np}{1-p} + \frac{n}{pd^2} \\
&= \frac{\epsilon}{h - \epsilon/n} + \frac{hn^2}{\epsilon d^2} \\
&< \frac{\epsilon}{1-\epsilon} + \epsilon.
\end{aligned}$$

But $\mathcal{Z}_W$ is well spread, so by Lemma 6.6, $\Pr_R\{R \geq r\} < 1/r!$ and the expectation $\mathrm{Exp}_R[e^{\vartheta R}]$ can be bounded as follows:

$$\begin{aligned}
\mathop{\mathrm{Exp}}_R[e^{\vartheta R}] &= \sum_{r \geq 0} e^{\vartheta r} \Pr_R\{R = r\} \\
&= \sum_{r \geq 0} e^{\vartheta r} \left(\Pr_R\{R \geq r\} - \Pr_R\{R \geq r+1\} \right) \\
&= \sum_{r \geq 0} e^{\vartheta r} \Pr_R\{R \geq r\} - \sum_{r \geq 1} e^{\vartheta(r-1)} \Pr_R\{R \geq r\}
\end{aligned}$$

$$\begin{aligned}
&= 1 + (1 - e^{-\vartheta}) \sum_{r \geq 1} e^{\vartheta r} \Pr_R\{R \geq r\} \\
&< 1 + \vartheta \sum_{r \geq 1} \frac{e^{\vartheta r}}{r!} \\
&= 1 + \vartheta(e^{e^{\vartheta}} - 1).
\end{aligned}$$

So, the probability that $\mathbf{x} \not\in \mathbf{T}(\mathcal{Z})$ given F is less than $\vartheta(e^{e^{\vartheta}} - 1)$ and

$$\begin{aligned}
\Pr_{\mathbf{T}}\{\mathbf{x} \not\in \mathbf{T}(\mathcal{Z})\} &\leq \Pr\{\neg F\} + \Pr\{\mathbf{x} \not\in \mathbf{T}(\mathcal{Z}) \mid F\} \\
&\leq \epsilon + \vartheta(e^{e^{\vartheta}} - 1).
\end{aligned}$$

Using the bound $\vartheta < \epsilon(1 + 1/(1 - \epsilon))$, we get that for all sufficiently small ϵ

$$\Pr_{\mathbf{T}}\{\mathbf{x} \not\in \mathbf{T}(\mathcal{Z})\} \leq 5\epsilon. \qquad \square$$

3. Strong probabilistic construction

In the previous section we proved that for every boolean vector $\mathbf{x}$, if $\mathbf{T}$ is chosen as described in Theorem 6.8, then with high probability there exists a $U \in \mathcal{Z}$ such that $\mathbf{T}(U) = \mathbf{x}$. It follows by an averaging argument that with high probability the size of $\mathbf{T}(\mathcal{Z}) \cap \{0,1\}^n$ (the set of all boolean vectors that can be represented as $\mathbf{T}(U)$ for some $U \in \mathcal{Z}$) is almost equal to the size of the whole $\{0,1\}^n$. We now show how to project $\mathbf{T}(\mathcal{Z}) \cap \{0,1\}^n$ onto the set of all binary strings of some shorter length.

Remember the restriction operation $\mathcal{Z}|_G$ defined in Section 1. Here we reformulate the same operation using vector notation. For any vector $\mathbf{x} \in \{0,1\}^n$ and subset of coordinates $G \subseteq \{1,\ldots,n\}$, define the restriction $\mathbf{x}|_G \in \{0,1\}^{|G|}$ as the vector obtained taking the coordinates of $\mathbf{x}$ with index in G. The restriction operation is extended to set of vectors in the obvious way: $\mathcal{W}|_G = \{\mathbf{x}|_G : \mathbf{x} \in \mathcal{W}\}$. The next lemma shows that the probability that a random restriction $\mathcal{W}|_G$ covers the whole set $\{0,1\}^G$ of binary strings is at least equal to the density of $|\mathcal{W}|$ in $\{0,1\}^n$. Our proof closely resemble the proof of Lemma 6.1 and can be considered as a probabilistic variant of Sauer's Lemma.

LEMMA 6.9 *Let $\mathcal{W}$ be a subset of $\{0,1\}^n$. If G is chosen uniformly at random among all subsets of $\{1,\ldots,n\}$, then*

$$\Pr_G\left\{\mathcal{W}|_G = \{0,1\}^G\right\} \geq \frac{|\mathcal{W}|}{2^n}.$$

Proof: By induction on n. The base case $n = 0$ is trivially true. (Notice that $\{0,1\}^G = \{0,1\}^n = \{\varepsilon\}$ and $\mathcal{W}|_G = \mathcal{W} = \{0,1\}^G$ if and only if $|\mathcal{W}| = 1$.) So, assume the statement holds for all $\mathcal{W} \subseteq \{0,1\}^n$ and let us prove it for $\mathcal{W} \subseteq \{0,1\}^{n+1}$. Choose G at random and let $G' = G \backslash \{n+1\}$. Notice that G' is a random subset of $\{1,\ldots,n\}$. Define the following sets:

$$\mathcal{W}_0 = \left\{\mathbf{x} : \begin{bmatrix} \mathbf{x} \\ 0 \end{bmatrix} \in \mathcal{W}\right\}, \qquad \mathcal{W}_1 = \left\{\mathbf{x} : \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \in \mathcal{W}\right\}.$$

Notice that $|\mathcal{W}| = |\mathcal{W}_0| + |\mathcal{W}_1| = |\mathcal{W}_0 \cup \mathcal{W}_1| + |\mathcal{W}_0 \cap \mathcal{W}_1|$. Moreover, if

- either $(n+1) \in G$ and $(\mathcal{W}_0 \cap \mathcal{W}_1)|_{G'} = \{0,1\}^{G'}$
- or $(n+1) \notin G$ and $(\mathcal{W}_0 \cup \mathcal{W}_1)|_{G'} = \{0,1\}^{G'}$,

then $\mathcal{W}|_G = \{0,1\}^G$. Therefore, using the inductive hypothesis, we get

$$\begin{aligned}
\Pr\{\mathcal{W}|_G = \{0,1\}^G\} &\geq \Pr\{(n+1) \in G\} \Pr\{(\mathcal{W}_0 \cap \mathcal{W}_1)|_{G'} = 2^{G'}\} \\
&\quad + \Pr\{(n+1) \notin G\} \Pr\{(\mathcal{W}_0 \cup \mathcal{W}_1)|_{G'} = 2^{G'}\} \\
&\geq \frac{1}{2}\left(\frac{|\mathcal{W}_0 \cup \mathcal{W}_1|}{2^n}\right) + \frac{1}{2}\left(\frac{|\mathcal{W}_0 \cap \mathcal{W}_1|}{2^n}\right) \\
&= \frac{|\mathcal{W}_0 \cup \mathcal{W}_1| + |\mathcal{W}_0 \cap \mathcal{W}_1|}{2^{n+1}} \\
&= \frac{|\mathcal{W}|}{2^{n+1}}. \qquad \square
\end{aligned}$$

Now, we are ready to prove Theorem 4.6.

Proof [of Theorem 4.6]: Instead of choosing the matrix $\mathbf{T} \in \{0,1\}^{n \times k}$ as specified in Theorem 4.6, we do the following mental experiment. First choose a bigger matrix $\mathbf{T}' \in \{0,1\}^{4n \times k}$ at random by setting each entry to 1 independently with probability $p = 4\epsilon/(hn)$. Then choose a random subset $G \subseteq 1,\ldots,4n$ of its rows. If G has size at least n, set $\mathbf{T}$ to the sub-matrix of $\mathbf{T}'$ with rows corresponding to the first n elements of G. If G has less than n elements, the experiment fails and $\mathbf{T}$ can be set to any $n \times k$ matrix.

Let $\mathcal{W} = \mathbf{T}'(\mathcal{Z}) \cap \{0,1\}^{4n}$. Notice that the probability distribution of matrix $\mathbf{T}$ (conditioned on the event $|G| \geq n$) is the same as in Theorem 4.6. Moreover, if $|G| \geq n$ and $\{0,1\}^G \subseteq \mathcal{W}|_G$ then $\{0,1\}^n \subseteq \mathbf{T}(\mathcal{Z})$. So, we can bound the probability that matrix $\mathbf{T}$ does not satisfy Theorem 4.6 as the sum of the probabilities that $|G| < n$ and $\{0,1\}^n \not\subseteq \mathbf{T}(\mathcal{Z})$.

Notice that $\mathrm{Exp}[|G|] = 2n$ and $\mathrm{Var}[|G|] = n$. So, by Chebychev's inequality

$$\begin{aligned}\Pr\{|G| < n\} &< \Pr\{||G| - \mathrm{Exp}[|G|]| < n\} \\ &< \frac{\mathrm{Var}[|G|]}{n^2} \\ &= \frac{1}{n} < \epsilon\end{aligned}$$

for all sufficiently large n. Now, let us bound the probability that $\{0,1\}^G \subseteq \mathcal{W}|_G$ when G and $\mathbf{T}'$ are chosen at random. Using Lemma 6.9 and the independence of G and $\mathbf{T}'$, one gets

$$\begin{aligned}\Pr_{G,\mathbf{T}'}\{\{0,1\}^G \subseteq \mathcal{W}|_G\} &= \mathop{\mathrm{Exp}}_{\mathbf{T}'}[\Pr_G\{\{0,1\}^G \subseteq \mathcal{W}|_G\}] \\ &\geq \mathop{\mathrm{Exp}}_{\mathbf{T}'}\left[\frac{|\mathcal{W}|}{2^{4n}}\right] \\ &= \mathop{\mathrm{Exp}}_{\mathbf{T}'}\left[\Pr_{\mathbf{x}\in\{0,1\}^{4n}}[\mathbf{x} \in \mathcal{W}]\right] \\ &= \mathop{\mathrm{Exp}}_{\mathbf{x}\in\{0,1\}^{4n}}\left[\Pr_{\mathbf{T}'}[\mathbf{x} \in \mathbf{T}'(\mathcal{Z})]\right] \\ &\geq \min_{\mathbf{x}\in\{0,1\}^{4n}} \Pr_{\mathbf{T}'}\{\mathbf{x} \in \mathbf{T}'(\mathcal{Z})\} \\ &\geq 1 - 5\epsilon.\end{aligned}$$

Therefore the probability that $\{0,1\}^n \not\subseteq \mathbf{T}(\mathcal{Z})$ is at most 5ϵ. By union bound, with probability at least $1 - 6\epsilon$ matrix $\mathbf{T}$ satisfies Theorem 4.6. □

4. Notes

A probabilistic variant of Sauer's lemma was first proved by (Ajtai, 1998), and used to establish the first NP-hardness result for SVP. Ajtai's construction and proof is rather involved, with sets T_i chosen according to easily samplable, but not independent, probability distributions. In this chapter we presented an alternative construction from (Micciancio, 1998; Micciancio, 2001d) with a simpler analysis. Moreover, the performance of our construction is arguably better than Ajtai's. Parameters k, h, n in (Ajtai, 1998) (as well in Theorem 4.6) are polynomially related, but the technicality of the proof in (Ajtai, 1998), makes it hard to extract the exact relation, which is currently unknown. So, an accurate comparison between the two results is not possible.

Chapter 7

BASIS REDUCTION PROBLEMS

In the first chapters of this book we studied the shortest vector problem and the closest vector problem both from an algorithmic and computational complexity point of view. In fact, the algorithms presented in Chapter 2 to approximately solve SVP and CVP do somehow more than just finding an approximately shortest lattice vector, or a lattice vector approximately closest to a given target. For example, the LLL algorithm on input a lattice basis $\mathbf{B}$, outputs an equivalent basis $\mathbf{B}'$ such that not only $\mathbf{b}'_1$ is an approximately shortest lattice vector, but also all other basis vectors $\mathbf{b}'_i$ are not too long. Moreover, LLL reduced bases have relatively good geometric properties that make them useful to solve other lattice problems. In particular, we have seen that if an LLL basis is used, then the nearest plane algorithm always finds a lattice vector approximately closest to any input target point. The problem of finding a "good" basis for a given lattice is generically called the *basis reduction* problem. Unfortunately, there is not a unique, clearly defined notion of what makes a basis good, and several different definitions of reduced basis have been suggested. In this chapter we consider the most important notions of basis reduction, define approximation problems naturally associated to such notions, and study the relation between these and other lattice problems.

1. Successive minima and Minkowski's reduction

A possible way to define reduced bases is to identify good bases with bases all of whose vectors are short. In Chapter 1 we have seen that for any lattice Λ of rank n with successive minima $\lambda_1, \ldots, \lambda_n$, there exist linearly independent vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$ of length $\|\mathbf{s}_i\| = \lambda_i$. It immediately follows from the definition of successive minima that these lengths are in-

D. Micciancio et al., *Complexity of Lattice Problems*

deed optimal, i.e., for any linearly independent lattice vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$ (in particular, for any lattice basis) if the vectors are sorted in order of increasing length $\|\mathbf{s}_1\| \leq \|\mathbf{s}_2\| \leq \cdots \leq \|\mathbf{s}_n\|$ then $\|\mathbf{s}_i\| \geq \lambda_i$ for all $i = 1, \ldots, n$. However, the converse is not necessarily true: there are lattices for which no basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$ exists such that $\|\mathbf{b}_i\| \leq \lambda_i$ for all $i = 1, \ldots, n$. Consider for example the lattice of all integer vectors such that all coordinates have the same parity, i.e., either all coordinates are even or they are all odd. (A possible basis for this lattice is given by vectors $\mathbf{b}_i = 2\mathbf{e}_i$ for $i = 1, \ldots, n-1$ and $\mathbf{b}_n = \sum_{i=1}^{n} \mathbf{e}_i$.) For $n \geq 4$, the length of the shortest nonzero lattice vector is clearly $\lambda_1 = 2$. Moreover, the lattice contains n linearly independent vectors $2\mathbf{e}_i$ of length exactly 2. Therefore $\lambda_i = 2$ for all $i = 1, \ldots, n$. However, it is easy to see that any basis $\mathbf{B}'$ generating this lattice must contain a vector of length at least $\sqrt{n}$. This is because if all vectors $\mathbf{b}'_i$ have even coordinates, then $\mathbf{B}'$ does not generate the whole lattice and it is not a basis. On the other hand, all vectors with odd coordinates have length at least $\sqrt{n}$ and for $n > 4$ this is strictly bigger than $\lambda_n = 2$.

The approximation problem associated to finding linearly independent lattice vectors of length as short as possible is formalized below.

DEFINITION 7.1 *The γ-approximate Successive Minima Problem (denoted* SMP$_\gamma$*) is defined as follows. Given a lattice basis* $\mathbf{B}$*, find linearly independent vectors* $\mathbf{S}$ *such that* $\|\mathbf{s}_i\| \leq \gamma\lambda_i$ *for all* $i = 1, \ldots, n$*. The decision (or promise) version of this problem (*GAPSMP$_\gamma$*) is, given a basis* $\mathbf{B}$ *and a sequence of values* $r_1, \ldots, r_n$*, decide if* $\lambda_i \leq r_i$ *for all* $i = 1, \ldots, n$*, or there exists an* i *such that* $\lambda_i > \gamma \cdot r_i$*. If neither of these conditions is satisfied, then* $(\mathbf{B}, r_1, \ldots, r_n)$ *violates the promise, and any answer is allowed.*

It is easy to see that GAPSMP$_\gamma$ can be efficiently reduced to the search problem SMP$_\gamma$: on input $(\mathbf{B}, r_1, \ldots, r_n)$, one calls the SMP$_\gamma$ search oracle on input $\mathbf{B}$. Let $\mathbf{S}$ be the set of linearly independent vectors returned by SMP$_\gamma$. Then, if $\|\mathbf{s}_i\| \leq \gamma \cdot r_i$ for all $i = 1, \ldots, n$ accept, otherwise reject.

Another simple observation is that the shortest vector problem SVP$_\gamma$ (or its promise version GAPSVP$_\gamma$) is (Karp) reducible to SMP$_\gamma$ (resp. GAPSMP$_\gamma$). For the search version, on input $\mathbf{B}$, one calls the SMP$_\gamma$ oracle on input $\mathbf{B}$ to obtain linearly independent vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$. Then $\mathbf{s}_1$ is a solution to SVP$_\gamma$. For the promise version, on input GAPSVP$_\gamma$ instance $(\mathbf{B}, r)$, one outputs GAPSMP$_\gamma$ instance $(\mathbf{B}, r, c, \ldots, c)$, where c is a sufficiently large constant such that $c \geq \lambda_n$. (For example one can set c to the maximum length of the input basis vectors.) It is easy to see that if $(\mathbf{B}, r)$ is a YES instance, then $(\mathbf{B}, r, c, \ldots, c)$ is a YES instance, while if $(\mathbf{B}, r)$ is a NO instance then $(\mathbf{B}, r, c, \ldots, c)$ is a NO instance.

Notice that when $\gamma = 1$, the condition on the lengths of vectors $\mathbf{s}_i$ in the definition of SMP$_\gamma$ can be inductively reformulated as follows:

- $\mathbf{s}_1$ is a shortest nonzero vector in the lattice,
- $\|\mathbf{s}_i\| \leq \|\mathbf{v}\|$ for any lattice vector $\mathbf{v}$ linearly independent from previously determined lattice vectors $\mathbf{s}_1, \ldots, \mathbf{s}_{i-1}$.

However, if an approximation factor γ is allowed, then the inductive definition is somehow misleading: even if $\mathbf{s}_1, \ldots, \mathbf{s}_n$ is a solution to SMP$_\gamma$ then it is not necessarily true that for all $i = 1, \ldots, n$, vector $\mathbf{s}_i$ satisfies $\|\mathbf{s}_i\| \leq \gamma\|\mathbf{v}\|$ for all lattice vectors $\mathbf{v}$ linearly independent from $\mathbf{s}_1, \ldots, \mathbf{s}_{i-1}$. For example, assume $\gamma = 2$ and consider the two dimensional lattice generated by $\mathbf{b}_1 = \mathbf{e}_1$ and $\mathbf{b}_2 = 2\mathbf{e}_2$. Clearly $\lambda_1 = 1$ and $\lambda_2 = 2$. Therefore vectors $\mathbf{s}_1 = \mathbf{b}_2$ and $\mathbf{s}_2 = 3\mathbf{b}_1 + \mathbf{b}_2$ are a solution to SMP$_\gamma$ because $\|\mathbf{s}_1\| = 2 \leq \gamma\lambda_1$ and $\|\mathbf{s}_2\| = \sqrt{13} < \gamma\lambda_2$. However, $\mathbf{s}_1$ and $\mathbf{s}_2$ do not satisfy the inductive definition because there exists a lattice vector $\mathbf{v} = \mathbf{b}_1$ linearly independent from $\mathbf{s}_1$ such that $\|\mathbf{s}_2\| > \gamma\|\mathbf{v}\| = 2$.

A classic notion of reduced basis in which vectors are required to be as short as possible is that of Minkowski.

DEFINITION 7.2 *A basis* $\mathbf{B}$ *is* Minkowski reduced *if for all* $i = 1, \ldots, n$ *vector* $\mathbf{b}_i$ *satisfies* $\|\mathbf{b}_i\| \leq \|\mathbf{b}'_i\|$ *for any lattice vector* $\mathbf{b}'_i$ *such that the sequence* $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}, \mathbf{b}'_i$ *can be completed to a basis, i.e., there exist lattice vectors* $\mathbf{b}'_{i+1}, \ldots, \mathbf{b}'_n$ *such that* $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}, \mathbf{b}'_i, \ldots, \mathbf{b}'_n$ *is a basis for* $\mathcal{L}(\mathbf{B})$.

Generalizing the definition of Minkowski to approximate versions is not an easy task. The problem is that the definition is intrinsically sequential, and the choice of the basis vectors at the beginning of the sequence may affect the possible choices for basis vectors in subsequent positions. Replacing the condition $\|\mathbf{b}_i\| \leq \|\mathbf{b}'_i\|$ in the definition with $\|\mathbf{b}_i\| \leq \gamma\|\mathbf{b}'_i\|$ gives problems similar to those pointed out above when talking about linearly independent sets. Another possibility is to define γ-approximate Minkowski reduced basis as a basis $\mathbf{B}$ such that $\|\mathbf{b}_i\| \leq \gamma\|\mathbf{b}'_i\|$ (for $i = 1, \ldots, n$) for *some* Minkowski reduced basis $\mathbf{B}'$. The problem with this definition is that requiring $\|\mathbf{b}_i\| \leq \gamma\|\mathbf{b}'_i\|$ for *some* Minkowski reduced basis $\mathbf{B}'$ does not necessarily implies that $\|\mathbf{b}_i\| \leq \gamma\|\mathbf{b}'_i\|$ for *any* Minkowski reduced basis. So, there might be Minkowski reduced bases that are better than $\mathbf{B}$ by more than a γ multiplicative factor. An alternative and stronger definition might be to require conditions $\|\mathbf{b}_i\| \leq \gamma\|\mathbf{b}'_i\|$ to hold for *any* Minkowski reduced basis $\mathbf{B}'$ or even requiring $\|\mathbf{b}_i\| \leq \gamma\lambda_i$. (Notice that $\lambda_i \leq \|\mathbf{s}_i\|$ is true for any sorted set of linearly independent lattice vectors, and, in particular, for any Minkowski reduced basis $\mathbf{S} = \mathbf{B}'$.) The problem with these last two definitions is that for values of γ close to 1, there is no guarantee that a solution to the problem exists, which is undesirable. (Later on, in this chapter, we will see that, when $\gamma \geq \sqrt{n}$, for any lattice Λ of rank n there exists a basis $\mathbf{B}$ such that $\|\mathbf{b}_i\| \leq \gamma\lambda_i$.)

Since Minkowski reduced bases do not play a particularly significant role in the study of the computational complexity of lattice problems, we do not define an approximation problem associated to Minkowski's reduction theory, and move on to a different and more easily defined notion of approximately shortest basis. We relax the condition on each basis vector $\mathbf{s}_i$ being individually as short as possible, and replace it with a global condition on all basis vectors $\mathbf{b}_i$. Namely, we use the maximum length

$$\mu(\mathbf{B}) = \max_i \|\mathbf{b}_i\| \tag{7.1}$$

as a measure of the length of a basis, and ask for a basis $\mathbf{B}$ such that $\mu(\mathbf{B})$ is as small as possible.

DEFINITION 7.3 *For any lattice Λ, let $\mu(\Lambda)$ be the minimum value of $\mu(\mathbf{B})$ when $\mathbf{B}$ ranges over all possible bases of Λ. The γ-approximate Shortest Basis Problem (*SBP$_\gamma$*) is, given a basis $\mathbf{B}$ of rank n, find an equivalent basis $\mathbf{B}'$ such that $\mu(\mathbf{B}') \leq \gamma \cdot \mu(\mathcal{L}(\mathbf{B}))$. The corresponding promise problem* GAPSBP$_\gamma$ *is, given a basis $\mathbf{B}$ and a value r, decide if there exists a basis $\mathbf{B}'$ equivalent to $\mathbf{B}$ such that $\mu(\mathbf{B}') \leq r$, or for all equivalent bases $\mu(\mathbf{B}') > \gamma \cdot r$. If neither of these conditions is satisfied, then the promise is violated and any answer is allowed.*

As usual, the promise problem GAPSBP$_\gamma$ immediately reduces to SBP$_\gamma$. (The details are left to the reader as an exercise.) If we do not require $\mathbf{B}$ to be a basis, and look for a set of linearly independent lattice vectors $\mathbf{S}$ such that $\mu(\mathbf{S})$ is minimized, then it is clear that there always exists a set such that $\mu(\mathbf{S}) \leq \lambda_n$ and that this value is optimal. This is the *shortest linearly independent vectors* problem defined below.

DEFINITION 7.4 *The (approximate) Shortest Independent Vectors Problem (denoted* SIVP$_\gamma$*) is, given a basis $\mathbf{B}$ of rank n, find linearly independent lattice vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$ such that $\|\mathbf{s}_i\| \leq \gamma \cdot \lambda_n(\mathcal{L}(\mathbf{B}))$ for all $i = 1, \ldots, n$. The corresponding promise problem* GAPSIVP$_\gamma$ *is, given a basis $\mathbf{B}$ of rank n and a rational number r, decide if $\lambda_n(\mathcal{L}(\mathbf{B})) \leq r$ or $\lambda_n(\mathcal{L}(\mathbf{B})) > \gamma \cdot r$.*

For any approximation factor γ, the promise problem GAPSIVP$_\gamma$ can be efficiently reduced to SIVP$_\gamma$ in the obvious way. The difference between this problem and the successive minima problem is that this time each vector $\mathbf{s}_i$ is not required to have length at most $\gamma\lambda_i$. As long as $\|\mathbf{s}_i\| \leq \gamma\lambda_n$ for all $i = 1, \ldots, n$, set $\mathbf{S}$ is a good solution to SIVP$_\gamma$. Clearly, any solution to SMP$_\gamma$ is also a solution to SIVP$_\gamma$, so there is a trivial reduction from SIVP$_\gamma$ to SMP$_\gamma$. Moreover, essentially the same reduction works for promise problems GAPSIVP$_\gamma$ and GAPSMP$_\gamma$. However, the converse is not necessarily true. In particular, while SVP$_\gamma$ immediately reduces to SMP$_\gamma$, it is not known how to efficiently reduce SVP$_\gamma$ to SBP$_\gamma$ or SIVP$_\gamma$ for any value of the approximation factor. (A reduction is clearly possible for all values of γ such that SVP$_\gamma$ can be solved in polynomial time, or the problems SIVP$_\gamma$ and SBP$_\gamma$ are NP-hard. Moreover, we will see, later in this chapter, that a reduction from SVP to SIVP or SBP is possible at the price of increasing the approximation factor by $\sqrt{n}$.)

We already know that for some lattices there is no basis such that $\mu(\mathbf{B}) \leq \lambda_n$, so the best solution to SBP$_\gamma$ is in general longer than the best solution to SIVP$_\gamma$. A natural question is how long the shortest basis can be. We will see that although for some lattice Λ the length of the shortest basis $\mu(\Lambda)$ can be as much as $\sqrt{n}/2$ times bigger than $\lambda_n(\Lambda)$, this is the worst that can happen, i.e., $\mu(\Lambda) \leq (\sqrt{n}/2)\cdot\lambda_n(\Lambda)$ for any lattice Λ of rank n. Moreover, given any set of linearly independent lattice vectors one can easily compute a basis increasing the length of each vector by at most a factor $\sqrt{n}/2$.

LEMMA 7.1 *There is a polynomial time algorithm that on input a lattice basis* $\mathbf{B}$ *and linearly independent lattice vectors* $\mathbf{S} \subset \mathcal{L}(\mathbf{B})$ *such that* $\|\mathbf{s}_1\| \leq \|\mathbf{s}_2\| \leq \cdots \leq \|\mathbf{s}_n\|$, *outputs a basis* $\mathbf{R}$ *equivalent to* $\mathbf{B}$ *such that* $\|\mathbf{r}_k\| \leq \max\{(\sqrt{k}/2)\|\mathbf{s}_k\|, \|\mathbf{s}_k\|\}$ *for all* $k = 1, \ldots, n$. *Moreover, the new basis satisfies span*$(\mathbf{r}_1, \ldots, \mathbf{r}_k) =$ *span*$(\mathbf{s}_1, \ldots, \mathbf{s}_k)$ *and* $\|\mathbf{r}_k^*\| \leq \|\mathbf{s}_k^*\|$ *for all* $k = 1, \ldots, n$.

Proof: Since the vectors $\mathbf{S}$ belong to the lattice, we can write $\mathbf{S} = \mathbf{BQ}$ for some integer square matrix $\mathbf{Q}$. Matrix $\mathbf{Q}$ is nonsingular, but not necessarily unimodular, i.e., $\det(\mathbf{Q}) \in \mathbb{Z}\setminus\{0\}$. Transform $\mathbf{Q}$ into an upper triangular integer matrix performing a sequence of elementary row operations (An elementary operation is adding an integer multiple of a row to some other row, multiplying a row by -1, or exchanging the order of two rows.), and perform the corresponding sequence of column operations to $\mathbf{B}$. Equivalently, find an unimodular matrix $\mathbf{U}$ such that $\mathbf{T} = \mathbf{UQ}$ is upper triangular, and compute $\mathbf{R} = \mathbf{BU}^{-1}$. Since $\mathbf{U}$ is unimodular, matrix $\mathbf{R}$ is a basis for $\mathcal{L}(\mathbf{B})$. Moreover $\mathbf{S} = \mathbf{BU}^{-1}\mathbf{UQ} = \mathbf{RT}$, so $\mathrm{span}(\mathbf{s}_1,\ldots,\mathbf{s}_k) = \mathrm{span}(\mathbf{r}_1,\ldots,\mathbf{r}_k)$ because $\mathbf{T}$ is upper triangular. Notice that $\mathbf{s}_i^* = \mathbf{r}_i^* t_{i,i}$. Therefore, $\|\mathbf{r}_i^*\| = \|\mathbf{s}_i^*\|/|t_{i,i}| \leq \|\mathbf{s}_i^*\|$ because $t_{i,i}$ is a nonzero integer. We still need to check if $\|\mathbf{r}_i\| \leq \max\{\sqrt{i}/2, 1\}\|\mathbf{s}_i\|$ for all $i = 1,\ldots,n$.

We modify vectors $\mathbf{r}_1,\ldots,\mathbf{r}_n$ sequentially as follows. Assume that $\mathbf{r}_1,\ldots,\mathbf{r}_{n-1}$ have already been modified, and consider the last vector $\mathbf{r}_n$. If $\mathbf{r}_n^* = \pm\mathbf{s}_n^*$, then we can replace $\mathbf{r}_n$ with $\mathbf{s}_n$ and still have a basis. Moreover, the new $\mathbf{r}_n$ clearly satisfies $\|\mathbf{r}_n\| \leq \|\mathbf{s}_n\|$. Conversely, if $\mathbf{r}_n^* \neq \pm\mathbf{s}_n^*$, then $\mathbf{s}_n^* = c\mathbf{r}_n^*$ for some integer $|c| > 1$. Consider the projection $\mathbf{r}_n - \mathbf{r}_n^*$ of $\mathbf{r}_n$ onto $\mathrm{span}(\mathbf{r}_1,\ldots,\mathbf{r}_{n-1})$, and use the nearest plane algorithm of Chapter 2 to find a lattice point $\mathbf{v} \in \mathcal{L}([\mathbf{r}_1,\ldots,\mathbf{r}_n])$ within distance $\sqrt{\sum_{i<n}\|\mathbf{r}_i^*\|^2/2}$ from $\mathbf{r}_n - \mathbf{r}_n^*$. We claim that $\|\mathbf{r}_n - \mathbf{v}\|$ is at most $(\sqrt{n}/2)\cdot\|\mathbf{s}_n\|$. Vector $\mathbf{r}_n - \mathbf{v}$ can be written as the sum of two orthogonal components, $\mathbf{r}_n^*$ and $(\mathbf{r}_n - \mathbf{r}_n^*) - \mathbf{v}$. The first component has length at most $\|\mathbf{r}_n^*\| = \|\mathbf{s}_n^*\|/c \leq \|\mathbf{s}_n^*\|/2$. By the choice of $\mathbf{v}$, the second component has length at most

$$\sqrt{\sum_{i<n}\|\mathbf{r}_i^*\|^2/2} \leq \sqrt{\sum_{i<n}\|\mathbf{s}_i^*\|^2/2} \leq \sqrt{\sum_{i<n}\|\mathbf{s}_i\|^2/2} \leq \sqrt{n-1}\max_i\|\mathbf{s}_i\|/2.$$

It follows that the length of $\mathbf{r}_n - \mathbf{v}$ is bounded by

$$\sqrt{((n-1)/4)\|\mathbf{s}_n\|^2 + (1/4)\|\mathbf{s}_n\|^2} = (\sqrt{n}/2)\|\mathbf{s}_n\|.$$

So, we can replace $\mathbf{r}_n$ with $\mathbf{r}_n - \mathbf{v}$, and obtain a basis satisfying all properties stated in the lemma. □

An immediate consequence of the lemma is that for any lattice Λ of rank n, the length of the shortest basis $\mu(\Lambda)$ is at most $(\sqrt{n}/2)\lambda_n$.

COROLLARY 7.2 *For any lattice Λ of rank n, there exists a basis $\mathbf{B}$ such that $\|\mathbf{b}_k\| \leq \max\{1, \sqrt{k}/2\}\cdot\lambda_k$ for all $k = 1,\ldots,n$. In particular, $\mu(\Lambda) \leq \max\{1, \sqrt{n}/2\}\cdot\lambda_n(\Lambda) \leq \sqrt{n}\cdot\lambda_n(\Lambda)$.*

The lemma can also be used to prove the equivalence (up to polynomial approximation factors) of SIVP_γ and SBP_γ.

THEOREM 7.3 *For any approximation factor γ, there exist Cook reductions from $\mathrm{SBP}_{\gamma\sqrt{n}}$ to SIVP_γ and from $\mathrm{SIVP}_{\gamma\sqrt{n}}$ to SBP_γ, where n is the rank of the lattice. Moreover, there exist Karp reductions from $\mathrm{GAPSBP}_{\gamma\sqrt{n}}$ to $\mathrm{GAPSIVP}_\gamma$ and from $\mathrm{GAPSIVP}_{\gamma\sqrt{n}}$ to GAPSBP_γ.*

Proof: We first reduce $\mathrm{SBP}_{\gamma\sqrt{n}}$ to SIVP_γ. On input $\mathrm{SBP}_{\gamma\sqrt{n}}$ instance $\mathbf{B}$, call the SIVP_γ oracle on input $\mathbf{B}$ to get linearly independent set $\mathbf{S}$ such that $\mu(\mathbf{S}) \leq \gamma\cdot\lambda_n(\mathcal{L}(\mathbf{B}))$. Then, run the algorithm of Lemma 7.1

on input $\mathbf{B}$ and $\mathbf{S}$ to get a basis $\mathbf{R}$ of $\mathcal{L}(\mathbf{B})$ such that $\|\mathbf{r}_i\| \leq \sqrt{n}\|\mathbf{s}_i\| \leq \sqrt{n}\gamma \cdot \lambda_n(\mathcal{L}(\mathbf{B}))$. Since, $\mu(\mathbf{B}) \geq \lambda_n(\mathcal{L}(\mathbf{B}))$, basis $\mathbf{R}$ is a solution to $\mathrm{SBP}_{\sqrt{n}\gamma}$.

In the other direction, given $\mathrm{SIVP}_{\gamma\sqrt{n}}$ instance $\mathbf{B}$, call the SBP_γ oracle on input $\mathbf{B}$, to get an equivalent basis $\mathbf{R}$ such that $\mu(\mathbf{R}) \leq \gamma \cdot \mu(\mathcal{L}(\mathbf{B}))$. Clearly, $\mathbf{R}$ is also a set of linearly independent lattice vectors in $\mathcal{L}(\mathbf{B})$. Moreover, from Corollary 7.2 we know that $\mu(\mathcal{L}(\mathbf{B})) \leq \sqrt{n}\lambda_n(\mathcal{L}(\mathbf{B}))$. Therefore, $\mu(\mathbf{R}) \leq \gamma\sqrt{n} \cdot \lambda_n(\mathcal{L}(\mathbf{B}))$ and $\mathbf{R}$ is a solution to $\mathrm{SIVP}_{\gamma\sqrt{n}}$.

Reductions between promise problems are similar and left to the reader as an exercise. □

2. Orthogonality defect and KZ reduction

In this section we consider a notion of reduced basis where one tries to make the basis vectors as orthogonal as possible. A quantity that has been used to measure how close a basis is to orthogonal is the *orthogonality defect* $\prod_i \|\mathbf{b}_i\|/\det(\mathbf{B})$. The relation between this quantity and almost orthogonal bases is easily explained. Let θ_i is the angle between $\mathbf{b}_i$ and $\mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$. Then $\|\mathbf{b}_i^*\| = \|\mathbf{b}_i\| \cos\theta_i$. Therefore

$$\prod_i \|\mathbf{b}_i\| = \prod_i \frac{\|\mathbf{b}_i^*\|}{\cos\theta_i} = \frac{\det \mathbf{B}}{\prod_i \cos\theta_i} \geq \det(\mathbf{B}). \tag{7.2}$$

So, the orthogonality defect is always at least 1, with equality if and only if $\cos\theta_i = 1$ for all $i = 1, \ldots, n$, i.e., $\theta_i = \pi/2$ and the basis $\mathbf{B}$ is completely orthogonal.

This shows that minimizing the orthogonality defect corresponds to finding a basis with almost orthogonal vectors. Since the orthogonality defect is proportional to the product of the lengths of the basis vectors, it is also clear that there is a close relationship between searching for almost orthogonal bases and bases consisting of short vectors. The definition of orthogonality defect can be extended to linearly independent sets of lattice vectors. Given linearly independent lattice vectors $\mathbf{S}$ in $\mathcal{L}(\mathbf{B})$, we define the orthogonality defect of $\mathbf{S}$ as $\prod_i \|\mathbf{s}_i\|/\det(\mathbf{B})$. It is important that $\prod_i \|\mathbf{s}_i\|$ is divided by the determinant of the original lattice, and not by $\det(\mathbf{S})$, because any full-rank integer lattice contains a set of linearly independent vectors with $\prod_i \|\mathbf{s}_i\|/\det(\mathbf{S}) = 1$. (Consider for example lattice vectors $\mathbf{s}_i = \det(\mathbf{B}) \cdot \mathbf{e}_i \in \mathcal{L}(\mathbf{B})$.) It is convenient to normalize the orthogonality defect, and consider the quantity $(\prod_i \|\mathbf{s}_i\|/\det(\mathbf{B}))^{1/n}$ instead, so that if vectors $\mathbf{S}$ are multiplied by a constant c, then the defect $(\prod_i \|c\mathbf{s}_i\|/\det(\mathbf{B}))^{1/n} = c(\prod_i \|\mathbf{s}_i\|/\det(\mathbf{B}))^{1/n}$ scales up linearly by a factor c.

DEFINITION 7.5 *Let* Λ *be a lattice of rank* n*, and let* $\mathbf{S}$ *be* n *linearly independent lattice vectors in* Λ*. The* normalized orthogonality defect *of* $\mathbf{S}$ *is* $\delta_\Lambda(\mathbf{S}) = (\prod_i \|\mathbf{s}_i\|/\det(\Lambda))^{1/n}$*. When* $\mathcal{L}(\mathbf{S}) = \Lambda$*, i.e.,* $\mathbf{S}$ *is a basis, we omit the subscript and simply write* $\delta(\mathbf{S})$*. The smallest orthogonality defect* $\delta(\mathbf{B})$ *for all possible bases* $\mathbf{B}$ *of a lattice* Λ *is denoted* $\delta(\Lambda)$*. The smallest orthogonality defect* $\delta_\Lambda(\mathbf{S})$ *where* $\mathbf{S}$ *is a maximal set of linearly independent vectors in* Λ*, is denoted* $\delta_\Lambda(\Lambda)$*. Notice that* $\delta_\Lambda(\Lambda) = (\prod_i \lambda_i/\det(\Lambda))^{1/n}$.

The computational problem associated to finding a basis with orthogonality defect approximately as small as possible is defined below.

DEFINITION 7.6 *The* γ-approximate Quasi Orthogonal Basis *problem (denoted* QOB_γ*) is, given a basis* $\mathbf{B}$*, find an equivalent basis* $\mathbf{B}'$ *such that* $\delta(\mathbf{B}') \leq \gamma \cdot \delta(\mathcal{L}(\mathbf{B}))$.

The analogous problem for linearly independent vectors is the following.

DEFINITION 7.7 *The* γ-approximate Quasi Orthogonal Set *problem (denoted* QOS_γ*) is, given a basis* $\mathbf{B}$*, find a set of linearly independent lattice vectors* $\mathbf{S}$ *such that* $\delta_\Lambda(\mathbf{S}) \leq \gamma \cdot \delta_\Lambda(\Lambda)$*, where* $\Lambda = \mathcal{L}(\mathbf{B})$.

We do not define promise problems associated to QOB_γ and QOS_γ because, as we will see soon, for any lattice Λ, $\delta(\Lambda)$ and $\delta_\Lambda(\Lambda)$ always belong to the interval $[1, \sqrt{n}]$. Therefore, for all $\gamma \geq \sqrt{n}$, the optimal value associated to QOS_γ and QOB_γ can be trivially approximated within polynomial factor $\sqrt{n}$. (Still finding a basis or independent set achieving this value seems a computationally hard problem. Compare with the problem of finding a nonzero lattice vector of length at most $\sqrt{n}\det(\mathcal{L}(\mathbf{B}))^{1/n}$: even if such vector is guaranteed to exists by Minkowski's theorem, we do not know any efficient algorithm to find it.) Problems QOB_γ and QOS_γ are equivalent up to $\sqrt{n}$ factors in the approximation parameters.

THEOREM 7.4 $\mathrm{QOB}_{\gamma\sqrt{n}}$ *can be reduced in polynomial time to* QOS_γ*, and* $\mathrm{QOS}_{\gamma\sqrt{n}}$ *can be reduced in polynomial time to* QOB_γ.

Proof: The proof is similar to that of Theorem 7.3, and it is left to the reader as an exercise. □

Minkowski's second theorem shows that for any lattice Λ there exists a set of linearly independent lattice vectors with normalized orthogonality defect

$$\delta_\Lambda(\mathbf{S}) = \sqrt[n]{\prod_i \|\mathbf{s}_i\|/\det(\Lambda)} = \sqrt[n]{\prod_i \lambda_i/\det(\Lambda)} \leq \sqrt{n}. \tag{7.3}$$

Using Lemma 7.1, we also get that for any lattice there exists a basis $\mathbf{B}$ such that $\delta(\mathbf{B}) \leq n$. In fact it is possible to do better than that, and show that any lattice has a *basis* with normalized orthogonality defect bounded by $\sqrt{n}$. A notion of reduced basis that gives smaller orthogonality defect than that guaranteed by Minkowski's theorem is the one studied by Korkine and Zolotarev.

DEFINITION 7.8 *Let* $\mathbf{B}$ *be a lattice basis of rank* n*, and let* $\mathbf{B}^*$ *the corresponding Gram-Schmidt orthogonalized basis. Define the projection functions* $\pi_i(\mathbf{x}) = \sum_{j\geq i}(\langle\mathbf{x}, \mathbf{b}_j^*\rangle/\|\mathbf{b}_j^*\|^2)\mathbf{b}_j^*$ *that map* $\mathbf{x}$ *orthogonally to span*$(\mathbf{b}_i^*, \ldots, \mathbf{b}_n^*)$. *Basis* $\mathbf{B}$ *is Korkine-Zolotarev reduced (KZ reduced, for short) if and only if for all* $i = 1, \ldots, n$,

- $\mathbf{b}_i^*$ *is a shortest nonzero vector in* $\pi_i(\mathcal{L}(\mathbf{B}))$
- *for all* $j < i$*, the Gram-Schmidt coefficients* $\mu_{i,j}$ *of* $\mathbf{B}$ *satisfy* $|\mu_{i,j}| \leq 1/2$.

It is easy to see that if a linearly independent set of lattice vectors $\mathbf{S}$ is KZ reduced, then $\mathbf{S}$ is a basis for the original lattice. So, for this problem there is no difference between lattice bases and linearly independent sets of vectors. This definition of Korkine-Zolotarev reduced basis is intrinsically sequential, i.e., the length of $\|\mathbf{b}_i^*\|$ depends on the choice of the previous basis vectors $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$. Below we give a slightly weaker, but conceptually simpler, definition of reduced basis that naturally generalizes to approximation versions of the same problem.

DEFINITION 7.9 *A basis* $\mathbf{B}$ *is* γ*-approximate Korkine-Zolotarev reduced (KZ*γ *reduced, for short) if for all* $i = 1, \ldots, n$,

- $\|\mathbf{b}_i^*\| \leq \gamma \cdot \lambda_i$.
- *for all* $j < i$*, the Gram-Schmidt coefficients* $\mu_{i,j}$ *of* $\mathbf{B}$ *satisfy* $|\mu_{i,j}| \leq 1/2$.

The γ*-approximate Korkine-Zolotarev problem (*KZP_γ*) is, given a basis* $\mathbf{B}$*, output a basis equivalent to* $\mathbf{B}$ *which is KZ*γ *reduced.*

Notice the similarity between this definition and our previous attempts to define approximate Minkowski reduced basis. In Section 1, we tried to define a γ-approximate Minkowski reduced basis as a basis $\mathbf{B}$ such that $\|\mathbf{b}_i\| \leq \gamma \cdot \lambda_i$. Unfortunately, for values of γ close to 1 no such basis is guaranteed to exist. Interestingly, if the orthogonalized vectors are used, there is always a basis such that $\|\mathbf{b}_i^*\| \leq \gamma \cdot \lambda_i$, even for $\gamma = 1$. (See Proposition 7.5 below.) Notice that for $\gamma = 1$, Definition 7.9

does not implies that $\mathbf{B}$ is reduced in the sense originally defined by Korkine and Zolotarev. However, the definition is justified by the following proposition.

PROPOSITION 7.5 *If* $\mathbf{B}$ *is a KZ reduced basis, then* $\mathbf{B}$ *is also KZ*γ *reduced for any* $\gamma \geq 1$.

In particular, the proposition shows that KZP_γ has a solution for all $\gamma \geq 1$. (Clearly, no solution exists for $\gamma < 1$ because $\|\mathbf{b}_1^*\| = \|\mathbf{b}_1\| \geq \lambda_1$.) We do not prove the proposition here, as this result will follow from a more general theorem to be proved below. The theorem states that the problems KZP_γ and SVP_γ are equivalent under Cook reductions. In particular, for any $\gamma \geq 1$, and for any lattice, there exists a basis that solves KZP_γ, and this basis can be efficiently found given access to an SVP_γ oracle.

THEOREM 7.6 *For any approximation factor* γ, SVP_γ *and* KZP_γ *are equivalent under Cook reductions, i.e., there exist Cook reductions from* SVP_γ *to* KZP_γ, *and from* KZP_γ *to* SVP_γ.

Proof: One direction is obvious: if $\mathbf{B}$ is a KZγ reduced basis, then $\|\mathbf{b}_1\| = \|\mathbf{b}_1^*\| \leq \gamma \cdot \lambda_1$ and $\mathbf{b}_1$ is a solution to SVP_γ. Now, assume we have access to an SVP_γ oracle. We use this oracle to compute a KZγ reduced basis. Let Λ be the input lattice. We compute the KZγ basis vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ sequentially, making appropriate calls to SVP_γ. We assume, without loss of generality, that the approximate solutions $\mathbf{v}$ returned by SVP_γ are primitive vectors, i.e., $\mathbf{v} \neq c\mathbf{w}$ for any integer $c \neq \pm 1$ and any lattice vector $\mathbf{w}$. (If this is not the case, $\mathbf{w}$ is a better solution to SVP_γ, and we can replace $\mathbf{v}$ with $\mathbf{w}$.)

First of all we call the SVP_γ oracle on input Λ to get a nonzero lattice vector $\mathbf{b}_1$ such that $\|\mathbf{b}_1^*\| = \|\mathbf{b}_1\| \leq \gamma\lambda_1(\Lambda)$. After vectors $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$ have been determined, we compute $\mathbf{b}_i$ as follows. Let $\Lambda_i = \pi_i(\Lambda)$ be the projection of Λ to the orthogonal complement of $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$. We call the SVP_γ oracle on input Λ_i, and find a lattice vector $\mathbf{b}_i \in \Lambda$ such that $\mathbf{b}_i^* = \pi_i(\mathbf{b}_i)$ is an approximately shortest vector in Λ_i. Notice that Λ contains i linearly independent vectors of length at most $\lambda_i(\Lambda)$. At least one of these vectors has a non zero component orthogonal to $\mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$. Therefore, Λ_i contains a nonzero vector of length at most $\lambda_i(\Lambda)$. This proves that $\lambda_1(\Lambda_i) \geq \lambda_i(\Lambda)$ and therefore $\|\mathbf{b}_i^*\| = \|\pi_i(\mathbf{b}_i)\| \leq \gamma\lambda_i(\Lambda)$.

This gives a sequence of lattice vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ such that $\|\mathbf{b}_i^*\| \leq \gamma\lambda_i$ for all $i = 1, \ldots, n$. Since each $\mathbf{b}_i^*$ is a primitive vector in Λ_i, then $\mathbf{B}$ is a basis for the original lattice. Finally, the condition $|\mu_{i,j}| \leq 1/2$

on the Gram-Schmidt coefficients can be easily achieved as follows. For every $\mathbf{b}_i$ (starting from $\mathbf{b}_2$, up to $\mathbf{b}_n$), run the nearest plane algorithm on lattice $[\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}]$ and target vector $\mathbf{b}_i$. Let $\mathbf{v}$ be the lattice point returned by the nearest plane algorithm, and replace $\mathbf{b}_i$ with $\mathbf{b}_i - \mathbf{v}$. It is easy to see that the Gram-Schmidt coefficients of the new $\mathbf{b}_i$ satisfy $|\mu_{i,j}| \leq 1/2$. □

So, an SVP_γ oracle can be used to efficiently compute KZγ reduced bases. Moreover, if an exact SVP_1 oracle is available, then the above reduction returns a basis which is reduced in the sense originally defined by Korkine and Zolotarev. The next theorem shows that KZγ reduced bases approximately solve the the basis (or independent set) reduction problems studied in Section 1.

THEOREM 7.7 *For any approximation factor γ, any solution to* KZP_γ *is also a solution to* $\mathrm{SIVP}_{\gamma\sqrt{n}}$, $\mathrm{SMP}_{\gamma\sqrt{n}}$, *and* $\mathrm{SBP}_{\gamma\sqrt{n}}$. *In particular,* $\mathrm{SIVP}_{\gamma\sqrt{n}}$, $\mathrm{SMP}_{\gamma\sqrt{n}}$ *and* $\mathrm{SBP}_{\gamma\sqrt{n}}$ *are Cook reducible to* KZP_γ.

Proof: We have already observed that any KZγ reduced set is also a basis. We prove that if $\mathbf{B}$ is KZγ reduced, then $\|\mathbf{b}_i\| \leq \sqrt{n}\gamma\lambda_i$. It follows that $\mathbf{B}$ is a solution to $\mathrm{SIVP}_{\gamma\sqrt{n}}$, $\mathrm{SMP}_{\gamma\sqrt{n}}$, and $\mathrm{SBP}_{\gamma\sqrt{n}}$. Let $\mu_{i,j}$ be the Gram-Schmidt coefficients associated to $\mathbf{B}$. We know, from Definition 7.9, that $|\mu_{i,j}| \leq 1/2$. Using the Gram-Schmidt orthogonalized vectors we get

$$\begin{aligned}
\|\mathbf{b}_i\|^2 &= \|\mathbf{b}_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\mathbf{b}_j^*\|^2 \\
&\leq \lambda_i^2 + \frac{1}{4}\sum_{j=1}^{i-1} \gamma^2\lambda_j^2 \\
&\leq \left(\frac{i+3}{4}\right) \cdot \gamma^2\lambda_i^2.
\end{aligned}$$

This proves that $\|\mathbf{b}_i\| \leq \sqrt{i}\gamma\lambda_i \leq \sqrt{n}\gamma\lambda_i$. □

Now that we have established the equivalence between KZP_γ and SVP_γ, we prove that any solution to KZP_γ (and therefore SVP_γ) can be used to approximately solve QOB_γ and QOS_γ.

THEOREM 7.8 *For any approximation factor γ, any solution to* SMP_γ *is also a solution to* QOS_γ. *In particular, any KZγ reduced basis is also a solution to* $\mathrm{QOB}_{\gamma\sqrt{n}}$ *and* $\mathrm{QOS}_{\gamma\sqrt{n}}$.

Proof: Let $\Lambda = \mathcal{L}(\mathbf{B})$ be a lattice of rank n. The first statement is obvious: if $\mathbf{S}$ is a linearly independent set of lattice vectors such that

$\|\mathbf{s}_i\| \leq \gamma\lambda_i$, then

$$\delta_\Lambda(\mathbf{S}) = \frac{\sqrt[n]{\prod_i \|\mathbf{s}_i\|}}{\det(\Lambda)} \leq \frac{\sqrt[n]{\prod_i \gamma\lambda_i}}{\det(\Lambda)} = \gamma\delta_\Lambda(\Lambda). \tag{7.4}$$

Now assume we have a KZγ reduced basis $\mathbf{B}$. We know from Theorem 7.7 that $\mathbf{B}$ is a solution to SMP$_{\gamma\sqrt{n}}$. From the first part, $\mathbf{B}$ is also a solution to QOS$_{\gamma\sqrt{n}}$. Finally, we observe that $\delta(\mathbf{B}) \leq \gamma\sqrt{n} \cdot \delta_\Lambda(\Lambda) \leq \gamma\sqrt{n} \cdot \delta(\Lambda)$ and $\mathbf{B}$ is a basis. Therefore, $\mathbf{B}$ is also a solution to QOB$_{\gamma\sqrt{n}}$. □

3. Small rectangles and the covering radius

In the last two sections we considered bases and linearly independent sets such that the maximum length $\max_i \|\mathbf{s}_i\|$ or the geometric mean $\sqrt[n]{\prod_i \|\mathbf{s}_i\|}$ is minimized. In this section we consider still another quantity that can be used to measure the quality of an independent set. Two fundamental constant associated to any lattice are the *packing radius* and the *covering radius.*

DEFINITION 7.10 *The* packing radius *of a lattice Λ is the largest radius r such that any two (open) spheres of radius r centered at two distinct lattice points do not intersect. The* covering radius *of Λ, denoted $\rho(\Lambda)$, is defined as the smallest radius ρ such that the (closed) spheres of radius ρ centered at all lattice points cover the entire space, i.e., any point in span($\mathbf{B}$) is within distance ρ from the lattice.*

It is easy to see that for any lattice Λ, the packing radius equals exactly $\lambda_1(\Lambda)/2$. So, determining the packing radius is equivalent to solving (the optimization version of) SVP. The covering radius ρ is also related to a familiar lattice problem (CVP), but this time the connection is weaker. The covering radius of $\mathcal{L}(\mathbf{B})$ is the smallest ρ such that CVP instance $(\mathbf{B}, \mathbf{t}, \rho)$ has solution for any $\mathbf{t} \in$ span($\mathbf{B}$). So, the covering radius corresponds to the worst case solution to CVP when the target point $\mathbf{t}$ ranges over span(Λ). (Notice that if lattice is not full dimensional and $\mathbf{t}$ is allowed to be any point in space, the distance of $\mathbf{t}$ from the lattice can be arbitrarily large.)

We do not introduce a new problem associated to the packing radius, as the problem is equivalent to SVP. Below we formalize the promise problem associated to computing the covering radius.

DEFINITION 7.11 *For any approximation factor $\gamma \geq 1$, the (approximate) Covering Radius Problem (denoted* GAPCRP$_\gamma$*) is the following promise problem. Instances are pairs $(\mathbf{B}, r)$. Moreover,*

- $(\mathbf{B}, r)$ *is a* YES *instance if* $\rho(\mathcal{L}(\mathbf{B})) \leq r$
- $(\mathbf{B}, r)$ *is a* NO *instance if* $\rho(\mathcal{L}(\mathbf{B})) > \gamma \cdot r$

It is clear that solving the GAPCRP$_\gamma$ promise problem, is equivalent to approximately computing the value of the covering radius, i.e., finding, on input a lattice Λ, a value r that belongs to the interval $[\rho(\Lambda), \gamma \cdot \rho(\Lambda)]$. We do not define any search problem for the covering radius. The reason is that there is no known natural search problem associated to computing the covering radius exactly, whose solution can be checked in polynomial time. One possibility might be to ask for a point in span(Λ) at distance ρ from the lattice, a so called *deep hole.* (A deep hole is a point in span(Λ) as far as possible from Λ.) However, given a point $\mathbf{t} \in \text{span}(\mathbf{B})$, it is not clear how to check in polynomial time that $\mathbf{t}$ is indeed a deep hole. In fact, the covering radius problem is not known to be solvable in nondeterministic polynomial time. The straightforward solution to the problem requires first to guess the position of a point $\mathbf{t}$ as far as possible from the lattice (i.e., a deep hole), and then check that there are no lattice points within distance ρ from $\mathbf{t}$. This alternation of quantifiers puts the (exact) covering radius problem in Π_2 at the second level of the polynomial hierarchy, a presumably strictly bigger class than NP.

In order to study the covering radius problem, we introduce one last basis reduction problem. As mention at the beginning of this section, we introduce one more quantity to measure the quality of a basis, which will be used in the next chapter to prove the security of lattice based cryptographic functions. Given a basis $\mathbf{B}$ with corresponding orthogonalized vectors $\mathbf{B}^*$, we consider the length of the diagonal of the orthogonal parallelepiped defined by $\mathbf{B}^*$:

$$\sigma(\mathbf{B}) = \sqrt{\sum_i \|\mathbf{b}_i^*\|^2}. \tag{7.5}$$

We want to find a set of linearly independent vectors $\mathbf{S}$ such that $\sigma(\mathbf{S})$ is as small as possible. Notice that by Lemma 7.1, any set of linearly independent lattice vectors $\mathbf{S}$ can be converted into a basis $\mathbf{B}$ preserving (or even reducing) $\sigma(\mathbf{S}) \geq \sigma(\mathbf{B})$. Therefore, without loss of generality we can search for a basis $\mathbf{B}$ such that $\sigma(\mathbf{B})$ is minimized.

DEFINITION 7.12 *For any lattice* Λ, *let* $\sigma(\Lambda)$ *be the smallest value of* $\sigma(\mathbf{B})$ *when* $\mathbf{B}$ *ranges over all possible bases. The* γ-approximate Shortest Diagonal Problem *(*SDP$_\gamma$*) is, given a basis* $\mathbf{B}$, *find an equivalent basis* $\mathbf{B}'$ *such that* $\sigma(\mathbf{B}') \leq \gamma \cdot \sigma(\mathcal{L}(\mathbf{B}))$. *The promise version of this problem*

*(*GAPSDP$_\gamma$*) is, given a lattice* $\mathbf{B}$ *and a value* r*, decide if* $\sigma(\mathcal{L}(\mathbf{B})) \leq r$ *or* $\sigma(\mathcal{L}(\mathbf{B})) > \gamma \cdot r$*. If neither of these conditions is satisfied, then* $(\mathbf{B}, r)$ *violates the promise and any answer is allowed.*

The nearest plane algorithm of Chapter 2, on input linearly independent vectors $\mathbf{S}$ and a target point $\mathbf{t} \in \operatorname{span}(\mathbf{S})$, always returns a lattice point $\mathbf{v} \in \mathcal{L}(\mathbf{S})$ within distance $\sigma(\mathbf{S})/2$ from $\mathbf{t}$. This proves that for any lattice Λ, the covering radius is at most $\rho(\Lambda) \leq \sigma(\Lambda)/2$. We want to prove that $\sigma(\Lambda)$ is not much bigger than the covering radius, in particular, for any lattice Λ, $\sigma(\Lambda)/2$ is within a factor $\sqrt{n}$ from $\rho(\Lambda)$. The following theorem establish relations between ρ, σ and λ_n.

THEOREM 7.9 *Let* Λ *be any lattice of rank* n*. Then*

$$\lambda_n(\Lambda) \leq 2\rho(\Lambda) \leq \sigma(\Lambda) \leq \sqrt{n} \cdot \lambda_n(\Lambda). \qquad (7.6)$$

Proof: We start from the last inequality. Fix a lattice Λ of rank n, and let $\mathbf{B}$ be a KZ reduced basis for Λ. We know from Proposition 7.5 that $\|\mathbf{b}_i^*\| \leq \lambda_i(\Lambda)$ for all $i = 1, \ldots, n$. Therefore,

$$\sigma(\Lambda) \leq \sigma(\mathbf{B}) = \sqrt{\sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2} \leq \sqrt{n} \cdot \lambda_n(\Lambda). \qquad (7.7)$$

Now, consider the second inequality $2\rho \leq \sigma$. Let $\mathbf{B}$ be such that $\sigma(\mathbf{B}) = \sigma(\Lambda)$. Notice that given a point $\mathbf{t}$ in $\operatorname{span}(\mathbf{B})$, one can always find a lattice point within distance $\frac{1}{2}\sqrt{\sum_i \|\mathbf{b}_i^*\|^2}$ from $\mathbf{t}$, for example using the nearest plane algorithm from Chapter 2. Therefore, $\rho(\Lambda) \leq \sigma(\Lambda)/2$.

It remains to prove the first inequality $\lambda_n \leq 2\rho$. Assume for contradiction $\lambda_n > 2\rho$ and let ϵ be a real number such that $\epsilon < \lambda_n - 2\rho$. We iteratively build a set of linearly independent lattice vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$ as follows. For any $i = 1, \ldots, n$, let $\mathbf{t}_i$ be any vector of length $\rho + \epsilon$ orthogonal to $\mathbf{s}_1, \ldots, \mathbf{s}_{i-1}$, and let $\mathbf{s}_i$ be a lattice point within distance ρ from $\mathbf{t}_i$. Then $\mathbf{s}_i$ is linearly independent from $\mathbf{s}_1, \ldots, \mathbf{s}_{i-1}$, because the distance of $\mathbf{s}_i$ from $\operatorname{span}(\mathbf{s}_1, \ldots, \mathbf{s}_{i-1})$ is at least $\|\mathbf{t}_i\| - \|\mathbf{s}_i - \mathbf{t}_i\| \geq \epsilon$. Moreover, by triangle inequality, $\|\mathbf{s}_i\| \leq \|\mathbf{t}_i\| + \|\mathbf{s}_i - \mathbf{t}_i\| \leq 2\rho + \epsilon < \lambda_n$. By induction on i, we obtain a set $\mathbf{s}_1, \ldots, \mathbf{s}_n$ of linearly independent lattice vectors of length $\|\mathbf{s}_i\| < \lambda_n$, contradicting the definition of λ_n. □

The relation between ρ, σ and λ_n established in the previous theorem immediately gives an approximate reduction between problems GAPCRP, GAPSDP and GAPSIVP.

THEOREM 7.10 *For any approximation factor* γ*, there is a Karp reduction between any of the following pairs of problems:*

- *From* GAPCRP$_{\gamma\sqrt{n}}$ *to* GAPSDP$_\gamma$,
- *From* GAPCRP$_{\gamma\sqrt{n}}$ *to* GAPSIVP$_\gamma$,
- *From* GAPSDP$_{\gamma\sqrt{n}}$ *to* GAPCRP$_\gamma$,
- *From* GAPSDP$_{\gamma\sqrt{n}}$ *to* GAPSIVP$_\gamma$,
- *From* GAPSIVP$_{\gamma\sqrt{n}}$ *to* GAPCRP$_\gamma$,
- *From* GAPSIVP$_{\gamma\sqrt{n}}$ *to* GAPSDP$_\gamma$, *and*
- *From* GAPSBP$_{\gamma\sqrt{n}}$ *to* GAPSDP$_\gamma$.

In particular, since GAPSDP$_1$ *is in* NP, *then also* GAPCRP$_{\sqrt{n}}$ *is in* NP.

Proof: On input GAPCRP$_{\gamma\sqrt{n}}$ instance $(\mathbf{B}, r)$, output GAPSDP$_\gamma$ instance $(\mathbf{B}, \sqrt{n}r)$. It is easy to see that the reduction maps YES instances to YES instances, and NO instances to NO instances. All other reductions are similar. □

We remark that for approximation factors $\gamma = o(\sqrt{n})$, GAPCRP$_\gamma$ is not known to be in NP. The following theorem gives additional reductions between the search versions of some of the above problems.

THEOREM 7.11 *For any approximation factor γ, there is a Cook reduction between any of the following pairs of problems:*

- *From* SDP$_{\gamma\sqrt{n}}$ *to* SIVP$_\gamma$,
- *From* SIVP$_{\gamma\sqrt{n}}$ *to* SDP$_\gamma$,
- *From* SBP$_{\gamma\sqrt{n}}$ *to* SDP$_\gamma$, *and*
- *From* SDP$_{\gamma\sqrt{n}}$ *to* KZP$_\gamma$.

Proof: The simple proofs are left to the reader as an exercise. □

All the relations between lattice approximation problem proved in this chapter are summarized in Figure 7.1. Each node correspond to a lattice approximation problem. An arrow from problem A to problem B indicate that there is a Cook reduction from A to B. In all cases, if both A and B are decision (or promise) problems, then there is also a Karp reduction between the two. If the arrow has no label, then the reduction preserves the approximation factor, i.e., approximating problem A within a factor γ can be reduced to approximating B within a factor γ for any $\gamma \geq 1$. Labeled arrows indicate that the reduction increases the

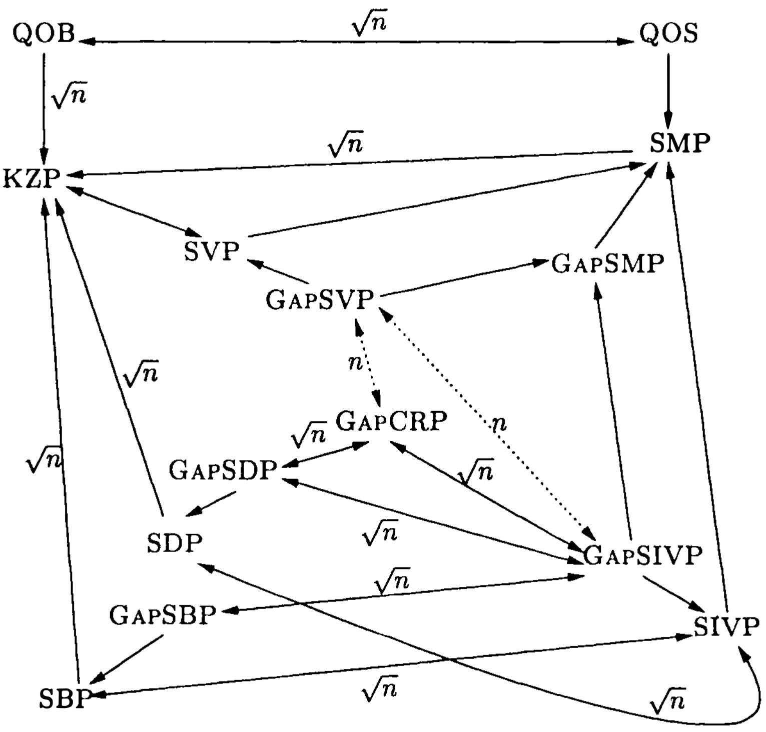


Figure 7.1. Relations among lattice approximation problems.

approximation factor. For example an arrow from A to B with label $\sqrt{n}$ means that approximating problem A within $\gamma \cdot \sqrt{n}$ can be reduced in polynomial time to approximating problem B within a factor γ. Reductions can be combined in the obvious way. For example, since SIVP$_\gamma$ reduces to SMP$_\gamma$ and SMP$_{\gamma\sqrt{n}}$ reduces to KZP$_\gamma$, then SIVP$_{\sqrt{n}\gamma}$ reduces to KZP$_\gamma$. Notice that a solution to SVP$_\gamma$ or KZP$_\gamma$, would allow to solve all other lattice reduction problems within a factor $\gamma\sqrt{n}$. Dotted lines from GapSVP to GapCRP and GapSIVP represent reductions that have not been described yet, and rely on harmonic analysis techniques that are beyond the scope of this book. In (Banaszczyk, 1993) it is proved that for any lattice $\Lambda = \mathcal{L}(\mathbf{B})$ of sufficiently high rank n, if $\Lambda' = \mathcal{L}(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1})$ is the so called *dual lattice* of Λ, then the covering radius and the successive minima of Λ and Λ' are related by the following bounds:

$$1 \leq \lambda_1(\Lambda)\lambda_n(\Lambda') < n \tag{7.8}$$

$$1 \leq \lambda_1(\Lambda)\rho(\Lambda') < n. \tag{7.9}$$

This kind of bounds are called *transference theorems*, and allow to infer information about a lattice, studying the properties of its dual. Notice that the dual of the dual equals the original lattice.

Using the transference theorems one can give simple reductions between the corresponding lattice problems. Interestingly, reductions obtained using transference theorems swap YES and NO instances.

THEOREM 7.12 *For any approximation factor γ, there are Karp reductions from the complement of* $\mathrm{GAPSVP}_{n\gamma}$ *to* GAPCRP_{γ} *or* $\mathrm{GAPSIVP}_{\gamma}$, *and from* $\mathrm{GAPCRP}_{\gamma n}$ *or* $\mathrm{GAPSIVP}_{\gamma n}$ *to the complement of* GAPSVP_{γ}.

Proof: We show how to reduce $\mathrm{GAPSVP}_{n\gamma}$ to the complement of GAPCRP_{γ}. The other reductions are analogous. Let $(\mathbf{B}, r)$ be an instance of $\mathrm{GAPSVP}_{n\gamma}$. The output of the reduction is GAPCRP_{γ} instance

$$\left(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}, \frac{1}{\gamma r}\right).$$

We want to prove that if $(\mathbf{B}, r)$ is a YES instance then $(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}, \frac{1}{\gamma r})$ is a NO instance, while if $(\mathbf{B}, r)$ is a NO instance then $(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}, \frac{1}{r\gamma})$ is a YES instance. Let $\Lambda = \mathcal{L}(\mathbf{B})$ be the lattice generated by the input basis, and let $\Lambda' = \mathcal{L}(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1})$ be its dual. Assume $(\mathbf{B}, r)$ is a YES instance. Then $\lambda_1(\Lambda) \leq r$, and using the transference theorems we get $\rho(\Lambda') \geq 1/\lambda_1(\Lambda) \geq 1/r$. This proves that $(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}, 1/(r\gamma))$ is a NO instance. (To be precise, we should have shown that inequality $\rho(\Lambda') > 1/r$ is strict. This is just a technicality, and can be easily fixed, either increasing the inapproximability factor γ by an arbitrarily small $\epsilon > 0$, or using the fact that the second inequality in (7.8) and (7.9) are strict.) Conversely, assume that $(\mathbf{B}, r)$ is a NO instance. Then $\lambda_1(\Lambda) > \gamma n r$ and using the transference theorems we get $\rho(\Lambda') \leq n/\lambda_1(\Lambda) < 1/(\gamma r)$, proving that $(\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}, 1/(r\gamma))$ is a YES instance. □

4. Notes

For all computational problems considered in this chapter, no polynomial time algorithm is known. Approximate solutions can be found in polynomial time using the LLL reduction algorithm, or any of its improved variants discussed in Chapter 2. The proof of Lemma 2.8 can be easily adapted to show that the length of the kth vector of an LLL reduced basis are within an exponential factor $\gamma = 2^{O(n)}$ from the kth successive minimum λ_k. Therefore, the LLL algorithm gives a polynomial time solution to SMP_{γ} for exponential approximation factor $\gamma = 2^{O(n)}$. Similarly, the improved algorithms of (Schnorr, 1987) give

$2^{O(n(\log\log n)^2/\log n)}$ approximation factors, which can be further reduced to $2^{O(n\log n\log n/\log n)}$ using randomization (Ajtai et al., 2001). Polynomial time algorithms to approximate all other problems considered in this chapter within similar approximation factors are readily obtained using the reductions depicted in Figure 7.1.

From the complexity point of view, we know from Theorem 4.4 that GAPSVP$_\gamma$ and SVP$_\gamma$ are NP-hard (under RUR reductions) for all $\gamma < \sqrt{2}$. It immediately follows that KZP$_\gamma$ and GAPSMP$_\gamma$ and SMP$_\gamma$ are also NP-hard. In fact, basis reduction problems seem much closer to CVP than SVP from a computational point of view, and better inapproximability results can be proven for most of them. For example, (Blömer and Seifert, 1999) proved that GAPSBP$_\gamma$ and GAPSIVP$_\gamma$ (and therefore also GAPSMP$_\gamma$, SBP$_\gamma$, SIVP$_\gamma$ and SMP$_\gamma$) are NP-hard for $\gamma = n^{1/\log\log n}$ by reduction from a variant of CVP.

Problems QOB and QOS have been considered before in the computer science literature (see for example (Kannan, 1987a) or (Goldreich et al., 1997b)), but they have not received much attention so far specifically from a computational complexity point of view. The SDP (and its decisional version GAPSDP) were introduced here just as an intermediate problem to study the relation between the covering radius problem and other more standard basis reduction problems. In the rest of this section we discuss the covering radius problem (GAPCRP).

Computing the covering radius of a lattice is presumably a very hard problem. This is a classic problem in the geometry of numbers, but it has received so far almost no attention from an algorithmic point of view. We do not know any polynomial time algorithm achieving approximation factors that are substantially less than exponential in the rank n of the lattice, and (the decisional problem associated to) computing the covering radius exactly is not even known to be solvable in NP (nondeterministic polynomial time). The obvious upper bound to the complexity of the exact covering radius problem (i.e., when $\gamma = 1$) is in Π_2^p, at the second level of the polynomial hierarchy. Interestingly, the analogous problem for linear codes is hard for Π_2^p (McLoughlin, 1984), so it is unlikely to be solvable in NP. We believe that GAPCRP is also hard for Π_2^p, but no proof is known at the time of this writing.

Maybe, the reason GAPCRP$_\gamma$ has attracted so little attention so far, is its perceived difficulty. In Chapter 8, we will see that the hardness of GAPCRP$_\gamma$ can be used to build provably secure cryptographic functions. This calls for a deeper investigation of the computational complexity of GAPCRP$_\gamma$: Is the problem NP-hard when $\gamma = 1$? Is it hard for Π_2^p? What is the highest value of γ for which the problem is hard for NP?

Chapter 8

CRYPTOGRAPHIC FUNCTIONS

Generally speaking, the goal of cryptography is the design of systems that withstand any malicious attempt to subvert them. The archetypical problem in cryptography is that of secret communication: two parties want to communicate with each other, and keep the conversation private, i.e., no one, other than the two legitimate parties, should be able to get any information about the messages being exchanged. This secrecy goal can be achieved if the two parties share a common random key that is known only to them. Then, in order to privately send a message, one can encode it using the key, and send the enciphered message to the other party over a public communication network. The receiver uses the shared key to invert the encoding procedure, and recover the original message. The original message, the enciphered message and the encoding and decoding processes are usually called *cleartext*, *ciphertext*, *encryption* and *decryption*. An encryption scheme is secure if recovering (any partial information about) the cleartext from the ciphertext without knowing the secret key is a computationally infeasible task. So, an adversary intercepting the ciphertext won't learn anything about the message, other than the fact that a message was sent, and possibly the length of the message. (For technical reasons, it is not possible to hide the length of the message being sent without making the communication scheme extremely inefficient, so leaking the message length is usually considered an acceptable compromise between efficiency and security.)

It has long been realized that the relevant notion of hardness in cryptography is *average-case* hardness: if the key is chosen at random, then no probabilistic polynomial time algorithm can break the scheme with nonnegligible probability. This is different from the more common *worst-case* notion of hardness used in computational complexity, e.g., in the

D. Micciancio et al., *Complexity of Lattice Problems*

theory of NP-completeness. Proving that a problem is NP-hard shows that (unless P = NP) there is no polynomial time algorithm that correctly solves *all* the instances of that problem. In other words, for any polynomial time program, (and for infinitely many input sizes,) there is *some* instance of the problem for which the program gives the wrong answer. This is clearly not enough for cryptography. It is not sufficient to know that there exists *some* key which is hard to break: the user wants some reasonable guarantee that, if her key is chosen at random according to the prescribed key generation procedure, then (with high probability) *her* key is hard to break. Typically, even if a small, but nonnegligible fraction of the keys, can be broken then the scheme is not considered sufficiently secure for cryptographic purposes. So, the notion of average-case hardness customarily used in cryptography is that of problems for which any probabilistic polynomial time algorithm has only a negligible chance of success at solving them. Formally, a function $f(n)$ is called *negligible* if it is less than any inverse polynomial $1/n^c$ for all sufficiently large n. This definition naturally corresponds to the identification of efficient computation with (probabilistic) computations that take time polynomial in the input size. A cryptographic construction is *asymptotically* secure if for any inverse polynomial function $1/n^c$ and any probabilistic polynomial time adversary, there exists an n_0 such that for all n bigger than n_0 the success probability of the adversary subverting the construction is less than $1/n^c$, where n is the security parameter of the system.

The ultimate goal of modern cryptography, is the construction of cryptographic functions that are provably hard to break (on the average). Unfortunately, based on our current understanding of computational complexity, no such construction is likely to come any time soon: if P = NP then most cryptographic problems would be unsolvable because the adversary can nondeterministically guess the secret key. So, an unconditionally secure cryptographic function would yield a proof that P $\neq$ NP, a major open problem in computational complexity. The second most desirable goal is the construction of cryptographic functions that are provably hard to break (on the average), under some standard (worst-case) computational complexity assumption. For example, assuming that there is no polynomial time algorithm that on input an integer n outputs the prime factorization of n, build a secure encryption scheme, i.e., an encryption scheme such that any polynomial time adversary has only a negligible chance of breaking it. To date, we do not know any such construction, and all cryptographic constructions based on the factoring problem typically require the assumption that factoring is hard not only in the worst case, but also on the average, for a suitable distribution of n. The same is true for almost any other hard mathematical problem that has been used for cryptographic applications. Recently, lattices have attracted considerable interest for their potential cryptographic applications because of a remarkable connection between their worst-case and average-case complexity (Ajtai, 1996). In this breakthrough paper, Ajtai showed that if there is no algorithm that approximately solves the (decisional) shortest vector approximation problem for any lattice within some polynomial factor $\gamma(n) = n^c$, then the shortest vector (search) problem is hard to solve exactly when the lattice is chosen at random according to a certain easily samplable distribution. Building on this result, Ajtai suggested a lattice-based one way function. *One way functions* are the simplest primitive in cryptography: a function f that is easy to compute in the forward direction, but hard to invert on randomly chosen input. Despite their simplicity, one way functions are known to be sufficient to solve many important problems in cryptography, like the construction of digital signatures, pseudo-random generators, private key encryption schemes and commitment protocols, among others.

Right after Ajtai's discovery of the connection between the worst-case and average-case hardness of lattice problems, many researchers suggested to use lattices for the solution of other, more complex, cryptographic problems beside one way functions. Most notably collision resistant hashing and public key encryption schemes. The construction of collision resistant hash functions closely resemble Ajtai's one way function, and it well illustrates how lattices can be used to construct cryptographic functions that are as hard to break as the worst-case instance of approximating certain lattice problems. It should be remarked that building cryptographic functions that are as hard to break as the worst case instance of the underlying mathematical problem is especially important in the case of lattices because lattice approximation algorithms (like the LLL algorithm studied in Chapter 2) are believed to perform much better on the average than the worst-case theoretical upper bounds. So, while it is reasonable to conjecture that there is no polynomial time algorithm that approximates lattice problems within small polynomial factors in the worst case, assuming that no such algorithm exists that succeeds with nonnegligible probability when the input lattice is chosen at random might not be a reasonable conjecture at all, depending on the particular input distribution. For example, we do not know any algorithm to approximate the length of the shortest vector in a lattice within a factor $\sqrt{n}$ in the worst case, and, based on our current knowledge (i.e., no known polynomial time algorithm achieves approximation factors that are substantially better than exponential in

n), assuming that no such algorithm exists is a legitimate mathematical conjecture. However, if we consider the same problem for randomly chosen lattices, and the input is given by n basis vectors selected independently and uniformly at random (from a sufficiently large ball), then with high probability the basis vectors are within an $O(\sqrt{n})$ factor from the shortest lattice vector, so approximating the length of the shortest lattice vector on the average can be easily accomplished for this input distribution. What's so remarkable about Ajtai's connection is that it provides an explicit probability distribution on lattices such that randomly selected lattices from this distribution are provably hard, under the sole assumption that there is no efficient algorithm that solves some (other) lattice problem in the worst case.

In the case of public key encryption, several different methods have been suggested. Some of them have provable security guarantees with worst-case/average-case connection similar to Ajtai's one-way functions, others are heuristics (with no known proof of security) that have been suggested as practical alternatives to commonly used public key encryption functions. In this chapter we introduce the ideas behind the design of lattice based cryptographic function. We start in Section 1 with some general techniques that are useful in many constructions. Then, is Section 2 we present a full, self contained description of a new collision resistant hash function with worst-case/average-case connection that generalizes and improves Ajtai's construction. Finally, we conclude with an overview of the principal lattice based public key encryption schemes in Section 3, presenting all schemes in a unified framework that illustrates the similarities and differences among all the schemes. Additional bibliographical and historical notes, and information about the latest developments in the area are given in Section 4.

1. General techniques

Most lattice based constructions are better understood and analyzed if formulated in group theoretic terms. In this section we explore the relation between lattices and finite commutative groups, we prove some discrepancy results that play an important role in the probabilistic analysis of lattice constructions with worst-case/average-case connection, and we briefly recall the definition and basic properties of the statistical distance, a useful tool for the analysis of randomized algorithms and reductions.

1.1 Lattices, sublattices and groups

Let $\mathbf{L}$ be a rank n lattice and let $\mathbf{M}$ be a full rank sublattice of $\mathbf{L}$. Equivalently, let $\mathbf{M} = \mathbf{LA}$ for some nonsingular integer matrix $\mathbf{A} \in \mathbb{Z}^{n \times n}$. The sublattice $\mathcal{L}(\mathbf{M})$ defines a natural equivalence relation on $\mathcal{L}(\mathbf{L})$ as follows: two lattice points $\mathbf{x}, \mathbf{y} \in \mathcal{L}(\mathbf{L})$ are equivalent (written $\mathbf{x} \equiv \mathbf{y}$) if and only if $\mathbf{x} - \mathbf{y} \in \mathcal{L}(\mathbf{M})$. (The equivalence relation $\equiv$ depends on the lattice $\mathcal{L}(\mathbf{M})$, so formally it should be written $\equiv_{\mathcal{L}(\mathbf{M})}$ or $\equiv_{\mathbf{M}}$. To simplify the notation, we omit the subscript and simply write $\equiv$ whenever the lattice $\mathbf{M}$ is obvious from the context.) The reader can easily check that $\equiv$ is an equivalence relation, i.e., it satisfies the reflexive, symmetric and transitive properties:

- $\mathbf{x} \equiv \mathbf{x}$ for all $\mathbf{x} \in \mathcal{L}(\mathbf{L})$.
- $\mathbf{x} \equiv \mathbf{y}$ if and only if $\mathbf{y} \equiv \mathbf{x}$
- if $\mathbf{x} \equiv \mathbf{y}$ and $\mathbf{y} \equiv \mathbf{z}$, then $\mathbf{x} \equiv \mathbf{z}$.

DEFINITION 8.1 *Let $\mathcal{L}(\mathbf{L})$ be a lattice and $\mathcal{L}(\mathbf{M})$ a full rank sublattice of $\mathcal{L}(\mathbf{L})$. The $\mathcal{L}(\mathbf{M})$-equivalence class of $\mathbf{x} \in \mathcal{L}(\mathbf{L})$ (denoted $[\mathbf{x}]_{\mathbf{M}}$) is the set of all $\mathbf{y} \in \mathcal{L}(\mathbf{L})$ such that $\mathbf{x} \equiv_{\mathbf{M}} \mathbf{y}$. The quotient $\mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ is the set of all $\mathcal{L}(\mathbf{M})$-equivalence classes of $\mathcal{L}(\mathbf{L})$.*

(Also for equivalence classes $[\mathbf{x}]_{\mathbf{M}}$ we often omit the subscript $\mathbf{M}$ wherever $\mathbf{M}$ is clear from the context, and write $[\mathbf{x}]$ instead of $[\mathbf{x}]_{\mathbf{M}}$.) The equivalence relation $\equiv$ is a congruence relation with respect to the addition operation, i.e., if $\mathbf{x} \equiv \mathbf{x}'$ and $\mathbf{y} \equiv \mathbf{y}'$, then $(\mathbf{x} + \mathbf{y}) \equiv (\mathbf{x}' + \mathbf{y}')$. It follows that for any two equivalence classes $[\mathbf{x}]$ and $[\mathbf{y}]$, the sum $[\mathbf{x} + \mathbf{y}]$ is well defined, i.e., it does not depend on the choice of representatives $\mathbf{x}$, $\mathbf{y}$, and the quotient $\mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ is an additive group with the sum operation just described.

PROPOSITION 8.1 *Let $\mathcal{L}(\mathbf{L})$ be a lattice and $\mathcal{L}(\mathbf{M})$ a full rank sublattice of $\mathcal{L}(\mathbf{L})$. The quotient $G = \mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ is an additive group with respect to operation*

$$[\mathbf{x}] + [\mathbf{y}] = [\mathbf{x} + \mathbf{y}].$$

Moreover, the function $\psi(\mathbf{x}) = [\mathbf{x}]$ is a group homomorphism from $(\mathcal{L}(\mathbf{L}), +, \mathbf{0})$ to $(G, +, 0)$ with kernel $\mathcal{L}(\mathbf{M})$, i.e., for every $\mathbf{x}, \mathbf{y} \in \mathcal{L}(\mathbf{L})$, function ψ satisfies

$$\begin{aligned} \psi(\mathbf{x} + \mathbf{y}) &= \psi(\mathbf{x}) + \psi(\mathbf{y}) \\ \psi(-\mathbf{x}) &= -\psi(\mathbf{x}) \\ \psi(\mathbf{x}) = 0 &\Leftrightarrow \mathbf{x} \in \mathcal{L}(\mathbf{M}). \end{aligned}$$

Equivalence classes in $\mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ can be associated with unique representative elements from $\mathcal{L}(\mathbf{L})$ in various ways. For example, one can use the set of lattice points $\mathcal{L}(\mathbf{L}) \cap \mathcal{P}(\mathbf{M})$ in the half open parallelepiped

$$\mathcal{P}(\mathbf{M}) = \{\mathbf{Mz} : \forall i.0 \le z_i < 1\}.$$

It is easy to see that for every equivalence class $[\mathbf{x}]$ there exists a unique element $\mathbf{x}' \in \mathcal{L}(\mathbf{L}) \cap \mathcal{P}(\mathbf{M})$ such that $\mathbf{x} \equiv \mathbf{x}'$, and such representative can be efficiently computed as follows: write $\mathbf{x}$ as $\mathbf{Mz}$, define $z_i' = \lfloor z_i \rfloor$ for all $i = 1, \ldots, n$, and set $\mathbf{x}' = \mathbf{Mz}'$. In particular, this proves that the group G is finite, and it has cardinality

$$|\mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})| = \det(\mathcal{L}(\mathbf{M}))/\det(\mathcal{L}(\mathbf{L})) = \det(\mathbf{A})$$

where $\mathbf{A}$ is the unique (square) integer matrix such that $\mathbf{M} = \mathbf{LA}$.

An alternative way to uniquely represent equivalence classes is to use integer points inside the orthogonalized parallelepiped $\mathcal{P}(\mathbf{A}^*)$. The lattice point represented by $\mathbf{z} \in \mathcal{P}(\mathbf{A}^*) \cap \mathbb{Z}^n$ is $\mathbf{Lz}$. The reader can easily check that for every equivalence class $[\mathbf{x}]$ there exists a unique $\mathbf{z} \in \mathcal{P}(\mathbf{A}^*) \cap \mathbb{Z}^n$ such that $\mathbf{Lz} \equiv \mathbf{x}$.

This time computing the representative $\mathbf{v} \in \mathbb{Z}^n \cap \mathcal{P}(\mathbf{A}^*)$ of an equivalence class $[\mathbf{x}]$ is more complicated, but, depending on the choice of the bases $\mathbf{M}, \mathbf{L}$, this representation can be much more efficient. A possible way to compute the representative $\mathbf{z}'$ for $[\mathbf{Lz}]$ is to use a variant of the nearest plane algorithm described in Chapter 2. Namely, if we apply that algorithm to lattice $\mathcal{L}(\mathbf{A})$ and target $\mathbf{z}$, we find a vector $\mathbf{a} \in \mathcal{L}(\mathbf{A})$ such that $\mathbf{z} - \mathbf{a}$ belongs to the *centered* orthogonal parallelepiped

$$\mathcal{P}'(\mathbf{A}^*) = \mathcal{P}(\mathbf{A}^*) - \frac{1}{2}\sum_i \mathbf{a}_i^* = \left\{\mathbf{A}^*\mathbf{z} : \forall i. - \frac{1}{2} \le z_i < +\frac{1}{2}\right\}.$$

Clearly, $\mathcal{P}'(\mathbf{A}^*) \cap \mathbb{Z}^n$ could also be used as set of representatives instead of $\mathcal{P}(\mathbf{A}^*) \cap \mathbb{Z}^n$. Alternatively, if the nearest plane algorithm is modified, replacing the line $c_j = \lfloor \langle \mathbf{b}, \mathbf{b}_j \rangle / \langle \mathbf{b}_j, \mathbf{b}_j \rangle \rceil$ with $c_j = \lfloor \langle \mathbf{b}, \mathbf{b}_j \rangle / \langle \mathbf{b}_j, \mathbf{b}_j \rangle \rfloor$ (See Figure 2.5 in Chapter 2.) then the vector $\mathbf{a} \in \mathcal{L}(\mathbf{A})$ returned by the modified nearest plane algorithm satisfies $\mathbf{z} - \mathbf{a} \in \mathcal{P}(\mathbf{A}^*)$.

The Hermite Normal Form

The set $\mathcal{P}(\mathbf{M}) \cap \mathcal{L}(\mathbf{L})$ and $\mathcal{P}(\mathbf{A}^*) \cap \mathbb{Z}^n$ can be used to represent the elements of the quotient group $G = \mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ with strings of length polynomial in the size of the bases $\mathbf{L}$ and $\mathbf{M}$. Although polynomial, this representation is not particularly efficient. In particular, this size can be much bigger than $\log |G|$, i.e., the minimal size required to represent all elements of the group. We show that if the basis **M** (or **L**) is appropriately chosen, then elements of $\mathcal{P}(\mathbf{A}^*) \cap \mathbb{Z}^n$ can be stored using $\log|G|$ bits, giving a space optimal representation for the elements of the group.

Let **L** be a rank n lattice basis and let $\mathbf{M} = \mathbf{LA}$ be a basis of a full rank sublattice of $\mathcal{L}(\mathbf{L})$. We know that $G = \mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ is a finite commutative groups of size $|G| = \det(\mathbf{A})$. Group G depends only on the lattices generated by **L** and **M**, so we can apply unimodular transformations to either basis without changing the group G. We consider bases **M** such that **A** has a special form.

DEFINITION 8.2 *A square nonsingular integer matrix* $\mathbf{A} \in \mathbb{Z}^{n \times n}$ *is in* Hermite Normal Form *(HNF) if*

- **A** *is upper triangular,. i.e.,* $a_{i,j} = 0$ *for all* $i > j$.
- *All diagonal elements of* **A** *are strictly positive, i.e.,* $a_{i,i} > 0$ *for all* $i = 1, \ldots, n$.
- *All non diagonal elements are reduced modulo the corresponding diagonal element on the same row, i.e.,* $0 \leq a_{i,j} < a_{i,i}$ *for all* $i < j$.

It is a classical result of Hermite that any matrix **A** is (column) equivalent to a (unique) matrix **H** in Hermite normal form. Equivalently, every lattice $\mathcal{L}(\mathbf{A})$ has a basis $\mathbf{H} = \mathbf{AU}$ (where **U** is a unimodular matrix) such that **H** is in Hermite normal form. The Hermite normal form of an integer matrix and the corresponding unimodular transformation can be computed in polynomial time. (See for example (Cohen, 1996).) In order to efficiently represent the elements of group G, we compute the Hermite normal form of **A**, and apply the corresponding unimodular transformation to the basis **M**. Equivalently, we can assume that $\mathbf{M} = \mathbf{LA}$ is the (unique) basis of $\mathcal{L}(\mathbf{M})$ such that **A** is in Hermite normal form. Notice that if **A** is in Hermite normal form, then the orthogonalized vectors are simply given by $\mathbf{a}_i^* = a_{i,i}\mathbf{e}_i$ and $\mathbb{Z}^n \cap \mathcal{P}(\mathbf{A}^*)$ is the set of all vectors $\mathbf{v} \in \mathbb{Z}^n$ such that

$$0 \leq v_i < a_{i,i}$$

In particular, each coordinate can be represented using $\log_2 a_{i,i}$ bits, and the size of the group element representation is

$$\sum_{i=1}^{n} \log a_{i,i} = \log_2 \prod_{i=1}^{n} a_{i,i} = \log_2 \det(\mathbf{A}) = \log_2 |G|.$$

When **A** is in Hermite normal form, the modified nearest plane algorithm to compute the representative for a group element becomes particularly simple. Using the triangular structure of **A**, the element of

```
Input: An integer basis $\mathbf{H} \in \mathbb{Z}^{n \times n}$ in Hermite normal form
and a target vector $\mathbf{t} \in \mathbb{Z}^n$.
Output: The unique integer vector $\mathbf{b} \in \mathcal{P}(\mathbf{H}^*)$ such that $\mathbf{b} - \mathbf{t} \in \mathcal{L}(\mathbf{H})$

let $\mathbf{b} := \mathbf{t}$
for $j = n, \ldots, 1$
    $c_j = \lfloor b_j / h_{j,j} \rfloor$
    $\mathbf{b} := \mathbf{b} - c_j \mathbf{b}_j$
return $\mathbf{b}$
```

Figure 8.1. Reducing a vector modulo an HNF basis

$\mathbb{Z}^n \cap \mathcal{P}(\mathbf{A}^*)$ associated to group element $[\mathbf{Lv}]$ can be easily computed using the algorithm shown in Figure 8.1. This algorithm gives also a way to efficiently implement the group operation in G. Given group element representations $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^n \cap \mathcal{P}(\mathbf{A}^*)$, the group element associated to their sum is easily computed adding up the two vectors $\mathbf{x} + \mathbf{y}$, and applying the algorithm of Figure 8.1 to the result.

A special case is when $\mathcal{L}(\mathbf{L}) = \mathbb{Z}^n$ is the lattice of all integer vectors, and $\Lambda = \mathcal{L}(\mathbf{M})$ is any integer lattice. Then, we use notation $\mathbf{v} \bmod \Lambda$ for the unique representative of $[\mathbf{v}]_\Lambda$ which is reduced modulo the (unique) HNF basis of Λ and use this element as the standard representative for $[\mathbf{v}]_\Lambda$.

The Smith Normal Form

We have seen that the elements of group G can be efficiently represented using $\log_2 |G|$ bits, and the group operation computed in *polynomial* time. We now present still another way to represent group elements that, in addition to providing a space efficient representation, allows to perform the group operation in *linear* time. The idea is the following. We know that $G = \mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ is a finite commutative group $\det(\mathbf{M})/\det(\mathbf{L})$, and therefore it can be decomposed into the direct sum of cyclic groups. The cycle structure of G can be recovered computing another normal form for matrix A.

DEFINITION 8.3 *A matrix $\mathbf{D} \in \mathbb{Z}^{n \times n}$ is in Smith Normal Form (SNF) if $\mathbf{D}$ is diagonal with nonnegative coefficients such that $d_{i+1,i+1}$ divides $d_{i,i}$ for all $i = 1, \ldots, n$.*

It is a well known fact that for every square nonsingular matrix $\mathbf{A}$ there exist unimodular matrices $\mathbf{U}, \mathbf{V}$ such that $\mathbf{UAV}$ is in Smith Normal Form. Moreover, $\mathbf{U}, \mathbf{V}$ and $\mathbf{D}$ can be computed from $\mathbf{A}$ in polynomial time. (See for example (Cohen, 1996).) It should be remarked that matrix $\mathbf{A}$ and its Smith normal form $\mathbf{D}$ do not generate the same lattice. However, the two matrices are equivalent in the sense that groups $\mathbb{Z}^n/\mathcal{L}(\mathbf{A})$ and $\mathbb{Z}^n/\mathcal{L}(\mathbf{D})$ are isomorphic. So, if $\mathbf{D} = \mathbf{UAV}$ is the Smith Normal Form of $\mathbf{A}$, then the group $G = \mathcal{L}(\mathbf{L})/\mathcal{L}(\mathbf{M})$ is isomorphic to the (additive) group

$$S = \mathbb{Z}_{d_{1,1}} \times \cdots \mathbb{Z}_{d_{n,n}}.$$

As for the HNF representation, elements of this group can be represented as integer vector $\mathbf{s} \in \mathbb{Z}^n$ such that

$$0 \leq s_i < d_{i,i}$$

for all $i = 1, \dots, n$. Therefore this representation has size

$$\sum_{i=1}^{n} \log_2 d_{i,i} = \log_2 \det(\mathbf{D}) = \log_2 \det(\mathbf{A}) = \log_2 |G|.$$

Moreover, the group operations are modular componentwise addition and negation, so they can be performed in

$$O(\sum_{i=1}^{n} \log_2 d_{i,i}) = O(\log_2 |G|)$$

time. It remains to show how to compute the SNF representation of a group element $[\mathbf{x}]$. The reader can easily verify that the function

$$\psi : [\mathbf{x}] \mapsto \mathbf{DM}^{-1}\mathbf{x} \bmod \mathbf{D}$$

is a group isomorphism from G to S. In particular, for any $\mathbf{x} \in \mathcal{L}(\mathbf{L})$, $\mathbf{DM}^{-1}\mathbf{x}$ is an integer vector, and $\psi([\mathbf{x}]) = \mathbf{0}$ if and only if $\mathbf{x} \in \mathcal{L}(\mathbf{M})$.

Sampling elements from finite groups

In this section we study two problems related to sampling elements from a finite group (almost) uniformly at random. In the first problem, the group is given as a quotient $\mathcal{L}(\mathbf{B})/\mathcal{L}(\mathbf{C})$ of two lattices $\mathcal{L}(\mathbf{C}) \subset \mathcal{L}(\mathbf{B})$ and we want to select a representative from $\mathcal{P}(\mathbf{C}) \cap \mathcal{L}(\mathbf{B})$ with perfectly uniform distribution. In the second problem, we consider a generic group G and a distribution X on G which is not too far from uniform, and show how to obtain almost uniform samples adding up a small number elements drawn according to X.

We consider the problem of selecting a lattice point uniformly at random from $\mathcal{L}(\mathbf{B}) \cap \mathcal{P}(\mathbf{C})$. Since $\mathcal{L}(\mathbf{B})$ is an infinite set, we cannot choose an element of $\mathcal{L}(\mathbf{B})$ uniformly at random and reduce it modulo $\mathbf{C}$. However, a simple procedure to sample $\mathcal{L}(\mathbf{B}) \cap \mathcal{P}(\mathbf{C})$ uniformly at random can be devised using the group structure of the quotient $\mathcal{L}(\mathbf{B})/\mathcal{L}(\mathbf{C})$.

PROPOSITION 8.2 *There is a probabilistic polynomial time algorithm that on input a lattice basis* $\mathbf{B}$ *and a full rank sublattice* $\mathcal{L}(\mathbf{C})$, *outputs a lattice point* $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ *with uniform distribution over* $\mathcal{L}(\mathbf{B}) \cap \mathcal{P}(\mathbf{C})$.

Proof: Although $\mathcal{L}(\mathbf{B})$ and $\mathcal{L}(\mathbf{C})$ are *infinite* commutative groups, their quotient $G = \mathcal{L}(\mathbf{B})/\mathcal{L}(\mathbf{C})$ is always *finite* and the order of group G is easily computed as $l = \det(\mathbf{C})/\det(\mathbf{B})$. Let $\mathbf{x} = \sum_i r_i \mathbf{b}_i$ a random combination of the generators of $\mathcal{L}(\mathbf{B})$ with coefficients r_i chosen uniformly and independently at random from $\{0, \ldots, l-1\}$. It is easy to see that $[\mathbf{x}]_\mathbf{C}$ is distributed uniformly at random in G. In order to compute the representative of $\mathbf{x}$ modulo $\mathbf{C}$, we write $\mathbf{x} = \mathbf{C}\mathbf{y}$ for some real vector $\mathbf{y}$. Then for all $i = 1, \ldots, n$, we set $y_i' = y_i - \lfloor y_i \rfloor$ to the fractional part of y_i. The final output is $\mathbf{C}\mathbf{y}'$. It is immediate to see that $\mathbf{C}\mathbf{y} \equiv \mathbf{C}\mathbf{y}'$ (mod $\mathbf{C}$), and $\mathbf{C}\mathbf{y}' \in \mathcal{P}(\mathbf{C})$. □

We now turn to the second problem. Let G be a finite group, and let A be a random variable over G with distribution not too far from uniform. Here "not too far" means that for any group element $g \in G$, the probability that A equals g is at least $0.5/|G|$ and no more than $1.5/|G|$. We show that if we combine (using the group binary operation) a relatively small number of independent copies of A, the resulting distribution rapidly approaches the uniform one.

PROPOSITION 8.3 *Let* $(G, +)$ *be a finite group and let* $A_1, \ldots, A_k$ *be* k *independent random variables over* G *such that for any* $g \in G$

$$\left|\Pr(A_i = g) - \frac{1}{|G|}\right| \leq \frac{1}{2|G|} \qquad (8.1)$$

for all $i = 1, \ldots, k$. *Then, the random variable defined by the sum* $A = \sum_{i=1}^k A_i$ *satisfies*

$$\left|\Pr(A = g) - \frac{1}{|G|}\right| \leq \frac{1}{2^k |G|}. \qquad (8.2)$$

Proof: By induction on k. If $k = 1$ then the statement in the proposition is trivially true. So, assume that the proposition holds for some k, and let us prove it for $k+1$. Let $A' = \sum_{i=1}^k A_k$ be the sum of the first k variables. By induction hypothesis, for all $g' \in G$,

$$|\Pr(A' = g') - 1/|G|| \leq 1/(2^k |G|).$$

Consider the sum $A = \sum_{i=1}^{k+1} A_i = A' + A_{k+1}$. Then, we have

$$\begin{aligned} \Pr\{A = g\} &= \Pr\{A' + A_{k+1} = g\} && (8.3) \\ &= \sum_{g' \in G} \Pr\{A' = g'\} \Pr\{A_{k+1} = g - g'\} && (8.4) \end{aligned}$$

It is easy to verify that the last expression equals

$$\sum_{g'\in G}\left(\Pr\{A'=g'\}-\frac{1}{|G|}\right)\left(\Pr\{A_{k+1}=g-g'\}-\frac{1}{|G|}\right)+\frac{1}{|G|} \tag{8.5}$$

Therefore $|\Pr\{A=g\}-1/|G||$ is at most

$$\sum_{g'\in G}\left|\Pr\{A'=g'\}-\frac{1}{|G|}\right|\cdot\left|\Pr\{A_{k+1}=g-g'\}-\frac{1}{|G|}\right| \tag{8.6}$$

and using the induction hypothesis and the hypothesis on A_{k+1} we get

$$\left|\Pr\{A=g\}-\frac{1}{|G|}\right|\leq\sum_{g'\in G}\frac{1}{2^k|G|}\cdot\frac{1}{2|G|}=\frac{1}{2^{k+1}|G|}. \qquad \square$$

1.2 Discrepancy

The determinant of a lattice $\det(\Lambda)$ can be informally defined as the inverse of the density of lattice points Λ in $\text{span}(\Lambda)$, meaning that if Q is a sufficiently large and regular region in $\text{span}(\Lambda)$, then the number of lattice points in Q is roughly proportional to $\text{vol}(Q)/\det(\Lambda)$. The exact number of points in Q depends on the shape and position of Q. Discrepancy theory studies the maximum possible deviations of this number from $\text{vol}(Q)/\det(\Lambda)$, and it is an interesting area of mathematics with many applications in discrete geometry and number theory, from volume estimations to exponential sums. A general treatment of the theory is beyond the scope of this book. In this section we give elementary proofs of some simple results that will be used later in this chapter. We prove upper and lower bounds on the number of lattice points contained inside a convex body Q. The bound is given in terms of the covering radius ρ of the lattice and the radius r of the biggest sphere completely contained in Q. The body Q is required neither to be centered around the origin, nor symmetric with respect to its center of gravity. The only assumptions about Q are convexity and the fact that Q contains a sphere of radius r. For example, these properties are satisfied by the Voronoi cells of a lattice.

DEFINITION 8.4 *Let Λ be a lattice and $\mathbf{x}\in\Lambda$ an arbitrary lattice point. The (open)* Voronoi *cell of $\mathbf{x}$ is the set $\mathcal{V}(\mathbf{x},\Lambda)$ of all points $\mathbf{z}\in span(\Lambda)$ that are closer to $\mathbf{x}$ than to any other lattice point:*

$$\mathcal{V}(\mathbf{x},\Lambda)=\{\mathbf{z}\in span(\Lambda)\mid\forall\mathbf{y}\in\mathcal{L}(\mathbf{B}).\|\mathbf{z}-\mathbf{x}\|<\|\mathbf{z}-\mathbf{y}\|\}. \tag{8.7}$$

The closed cell $\bar{\mathcal{V}}(\mathbf{x},\Lambda)$ *is the topological closure of* $\mathcal{V}(\mathbf{x},\Lambda)$

$$\bar{\mathcal{V}}(\mathbf{x},\Lambda) = \{\mathbf{z} \in span(\Lambda) \mid \forall \mathbf{y} \in \mathcal{L}(\mathbf{B}).\|\mathbf{z}-\mathbf{x}\| \leq \|\mathbf{z}-\mathbf{y}\|\}. \qquad (8.8)$$

We need some simple properties about Voronoi cells, as listed below. All properties are easily verified and their proofs are left to the reader.

PROPOSITION 8.4 *Let* Λ *be a lattice with covering radius* ρ *and minimum distance* λ_1. *Then the Voronoi cells of* Λ *satisfy the following properties:*

- *All Voronoi cells* $\mathcal{V}(\mathbf{x},\Lambda)$ *(with* $\mathbf{x} \in \Lambda$*) are shifted copies*

  $$\mathcal{V}(\mathbf{x},\Lambda) = \mathcal{V}(\mathbf{0},\Lambda) + \mathbf{x}$$

  of the fundamental cell associated to the origin.

- $\mathcal{V}(\mathbf{x},\Lambda)$ *is a bounded, open, convex set, symmetric about* $\mathbf{x}$.

- *Each cell* $\mathcal{V}(\mathbf{x},\Lambda)$ *contains a sphere of radius* $\lambda_1/2$, *and it is completely contained in a sphere of radius* ρ:

  $$\mathcal{B}(\mathbf{x},\lambda_1/2) \subset \mathcal{V}(\mathbf{x},\Lambda) \subset \mathcal{B}(\mathbf{x},\rho).$$

- *The volume of* $\mathcal{V}(\mathbf{x},\Lambda)$ *(or, equivalently,* $\bar{\mathcal{V}}(\mathbf{x},\Lambda)$*) equals*

  $$\mathrm{vol}(\mathcal{V}(\mathbf{x},\Lambda)) = \mathrm{vol}(\bar{\mathcal{V}}(\mathbf{x},\Lambda)) = \det(\Lambda).$$

- *For any two distinct lattice points* $\mathbf{x} \neq \mathbf{y} \in \Lambda$, *the corresponding Voronoi cells are disjoint, i.e.,*

  $$\mathcal{V}(\mathbf{x},\Lambda) \cap \mathcal{V}(\mathbf{y},\Lambda) = \emptyset \qquad (8.9)$$

- *The union of all closed Voronoi cells covers the entire space, i.e.,*

  $$\bigcup_{\mathbf{x}\in\Lambda} \bar{\mathcal{V}}(\mathbf{x},\Lambda) = span(\Lambda). \qquad (8.10)$$

The bounds on the number of lattice points inside a convex body are based on the following two simple lemmas.

LEMMA 8.5 *Let* Λ *be a lattice with covering radius* ρ *and* $\mathcal{Q}$ *an arbitrary (closed) convex body in* $span(\Lambda)$ *containing a sphere of radius* r. *If* $\mathbf{x} \in \Lambda \cap \mathcal{Q}$ *is a lattice point inside* $\mathcal{Q}$, *then the entire cell* $\mathcal{V}(\mathbf{x},\Lambda)$ *is contained in the body* $\mathcal{Q}'$ *obtained expanding* $\mathcal{Q}$ *by a factor* $(1+\rho/r)$. *(Expansion performed using the center of the sphere as the origin.)*

Proof: Let $\mathbf{x}$ be any point of $\Lambda \cap Q$, and $\mathbf{y}$ any point of $\mathcal{V}(\mathbf{x}, \Lambda)$. We want to prove that $\mathbf{y}$ belongs to Q'. If $\mathbf{y}$ belongs to Q then the statement is trivially true because Q is contained in Q'. So, assume $\mathbf{y} \notin Q$ and let $\mathbf{x}'$ be a point of (the closure of) Q closest to $\mathbf{y}$. (See Figure 8.2) Clearly, $\mathbf{x}'$ cannot be an internal point of Q, and it must belong to the frontier of Q. We also have $\|\mathbf{y} - \mathbf{x}'\| \leq \|\mathbf{y} - \mathbf{x}\| \leq \rho$ because $\mathbf{x}$ belongs to Q and $\mathbf{y}$ belongs to the Voronoi cell of $\mathbf{x}$. Now consider the segment connecting $\mathbf{y}$ to the center $\mathbf{c}$ of a sphere of radius r contained in Q. Since $\mathbf{c}$ is internal to Q and $\mathbf{y}$ does not belong to Q, this segment intersect the boundary ∂Q in a unique point $\mathbf{y}'$. Let ℓ' be the line connecting $\mathbf{x}'$ and $\mathbf{y}'$, and let ℓ be the unique line parallel to ℓ' containing $\mathbf{y}$. Notice that $\mathbf{c}$, ℓ and ℓ' all belong to a common plane. Define the projection $\mathbf{z}$ and $\mathbf{z}'$ of center $\mathbf{c}$ on lines ℓ and ℓ'. Notice that since $\mathbf{x}'$ and $\mathbf{y}'$ are boundary points of Q and Q is convex, then $\mathbf{z}'$ cannot be an internal point of Q. Since all points within distance r from $\mathbf{c}$ belong to Q, it must be $\|\mathbf{z}' - \mathbf{c}\| \geq r$. Also, the distance between ℓ and ℓ' equals $\|\mathbf{z} - \mathbf{z}'\|$, and therefore $\|\mathbf{z} - \mathbf{z}'\| \leq \|\mathbf{x}' - \mathbf{y}\| \leq \rho$. Then we have

$$\frac{\|\mathbf{y} - \mathbf{c}\|}{\|\mathbf{y}' - \mathbf{c}\|} = \frac{\|\mathbf{z} - \mathbf{c}\|}{\|\mathbf{z}' - \mathbf{c}\|} = 1 + \frac{\|\mathbf{z} - \mathbf{z}'\|}{\|\mathbf{z}' - \mathbf{c}\|} \leq 1 + \frac{\rho}{r}.$$

Since point $\mathbf{y}'$ belongs to (the closure of) Q and $\|\mathbf{y} - \mathbf{c}\| \leq (1 + \rho/r)\|\mathbf{y}' - \mathbf{c}\|$, point $\mathbf{y}$ belongs to (the closure of) Q'. □

LEMMA 8.6 *Let Λ be a lattice with covering radius ρ and Q an arbitrary (open) convex body containing a sphere of radius r. If $\mathbf{x} \in \Lambda \setminus Q$ is a lattice point outside Q, then the entire cell $\bar{\mathcal{V}}(\mathbf{x}, \Lambda)$ is disjoint from the body Q'' obtained contracting Q by a factor $(1 - \rho/r)$. (Contraction performed using the center of the sphere as the origin.)*

Proof: The proof, similar to the one of Lemma 8.5, is left as an exercise to the reader. □

We use Lemmas 8.5 and 8.6 to bound the number of lattice points inside Q. Notice that on average Q contains $\mathrm{vol}(Q)/\det(\Lambda)$ lattice points. The following proposition shows that if r is large with respect to ρ, then the number of lattice points inside Q is approximately $\mathrm{vol}(Q)/\det(\Lambda)$.

PROPOSITION 8.7 *Let Λ be a lattice and Q an arbitrary open convex body in span(Λ). If Q contains an (open) sphere of radius $r \geq \rho(\Lambda)n$, then the number of lattice points inside Q (or its closure $\bar{Q}$) satisfies*

$$\frac{\mathrm{vol}(Q)}{\det(\Lambda)}\left(1 - \frac{\rho(\Lambda)n}{r}\right) < \quad |\Lambda \cap Q|$$

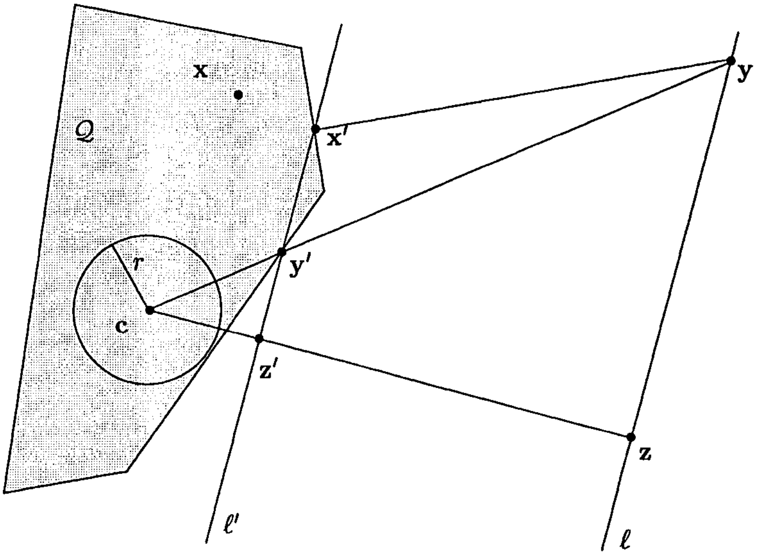


Figure 8.2. Lattice points inside a convex body

$$\leq \quad |\Lambda \cap \bar{Q}| \quad < \frac{\mathrm{vol}(Q)}{\det(\Lambda)}\left(1 + \frac{2\rho(\Lambda)n}{r}\right).$$

Proof: We start with the lower bound. Let $\rho = \rho(\Lambda)$ be the covering radius of the lattice and $\mathbf{c}$ be the center of a sphere of radius r contained in Q. Let Q'' be the convex body obtained contracting Q around $\mathbf{c}$ by a factor $(1 - \rho/r)$. Clearly the volume of Q'' satisfies

$$\mathrm{vol}(Q'') = \left(1 - \frac{\rho}{r}\right)^n \mathrm{vol}(Q) > \left(1 - \frac{n\rho}{r}\right)\mathrm{vol}(Q) \qquad (8.11)$$

Let X be the set of all lattice points $\mathbf{x} \in \Lambda$ such that $\bar{\mathcal{V}}(\mathbf{x}, \Lambda)$ intersects Q''. By Lemma 8.6, all points in X belong to Q, so it is enough to bound the size of X (from below). Since sets $\bar{\mathcal{V}}(\mathbf{x}, \Lambda)$ cover the entire space, Q'' is completely contained in $\bigcup_{\mathbf{x}\in X} \bar{\mathcal{V}}(\mathbf{x}, \Lambda)$, and

$$\mathrm{vol}(Q'') \leq \sum_{\mathbf{x}\in X} \mathrm{vol}(\bar{\mathcal{V}}(\mathbf{x}, \Lambda)) = |X| \cdot \det(\Lambda) \qquad (8.12)$$

Combining (8.11) and (8.12) we get

$$|X| > \frac{\mathrm{vol}(Q)}{\det(\Lambda)}\left(1 - \frac{\rho n}{r}\right), \qquad (8.13)$$

proving the lower bound on $|\Lambda \cap Q|$.

The proof of the upper bound is similar. Let $\mathcal{Q}'$ be the convex body obtained expanding $\bar{\mathcal{Q}}$ around $\mathbf{c}$ by a factor $(1+\rho/r)$. The volume of $\mathcal{Q}'$ satisfies

$$\mathrm{vol}(\mathcal{Q}') = \left(1+\frac{\rho}{r}\right)^n \mathrm{vol}(\mathcal{Q}) \leq e^{n\rho/r}\,\mathrm{vol}(\mathcal{Q}). \tag{8.14}$$

Using the convexity of the exponential function, and condition $(\rho n/r) \leq 1$ we get

$$e^{n\rho/r} \leq e^0 + (e^1 - e^0)\left(\frac{n\rho}{r}\right) < 1 + 2\left(\frac{n\rho}{r}\right). \tag{8.15}$$

Combining (8.14) and (8.15) we get

$$\mathrm{vol}(\mathcal{Q}') < \left(1 + \frac{2n\rho}{r}\right)\mathrm{vol}(\mathcal{Q}). \tag{8.16}$$

Let X be the set of all lattice points $\mathbf{x} \in \Lambda$ such that $\mathcal{V}(\mathbf{x},\Lambda)$ is contained in $\mathcal{Q}'$. By Lemma 8.5, all lattice points in $\bar{\mathcal{Q}}$ belong to set X, so it is enough to bound the size of X (from above). Since cells $\mathcal{V}(\mathbf{x},\Lambda)$ are disjoint

$$\mathrm{vol}(\mathcal{Q}') \geq \sum_{\mathbf{x}\in X}\mathrm{vol}(\mathcal{V}(\mathbf{x},\Lambda)) = |X|\cdot\det(\Lambda) \tag{8.17}$$

Combining (8.16) and (8.17) we get

$$|X| < \frac{\mathrm{vol}(\mathcal{Q})}{\det(\Lambda)}\left(1+\frac{2\rho n}{r}\right), \tag{8.18}$$

proving the upper bound on $|\Lambda \cap \bar{\mathcal{Q}}|$. □

1.3 Statistical distance

The statistical distance is a measure of how two probability distributions are far apart from each other, and it is a convenient tool in the analysis of randomized algorithms and reductions. In this section we define the statistical distance and prove some simple facts that will be used in the analysis of cryptographic functions.

DEFINITION 8.5 *Let X and Y be two discrete random variables over a (countable) set A. The statistical distance between X and Y is the quantity*

$$\Delta(X,Y) = \frac{1}{2}\sum_{a\in A}|\Pr\{X=a\} - \Pr\{Y=a\}|.$$

We say that two random variables X,Y are identically distributed (written $X \equiv Y$) if and only if $\Pr\{X = a\} = \Pr\{Y = a\}$ for every

$a \in A$. The reader can easily check that the statistical distance satisfies the usual properties of distance functions, i.e., it is a positive definite binary symmetric function that satisfies the triangle inequality:

$$\begin{aligned} \Delta(X,Y) &\geq 0 \quad \text{with equality if and only if } X \equiv Y && (8.19)\\ \Delta(X,Y) &= \Delta(Y,X) && (8.20)\\ \Delta(X,Z) &\leq \Delta(X,Y) + \Delta(Y,Z). && (8.21) \end{aligned}$$

The following property of the statistical distance is useful when analyzing a probabilistic algorithm that is part of a larger randomized process.

PROPOSITION 8.8 *Let X, Y be random variables over a set A, and let Z be a third random variable over a (possibly different) set B. If Z is statistically independent from X and Y. Then*

$$\Delta((X,Z),(Y,Z)) = \Delta(X,Y).$$

Proof: From the definition of statistical distance and the independence of Z from X and Y we immediately get

$$\begin{aligned} &\Delta((X,Z),(Y,Z))\\ &= \frac{1}{2}\sum_{a,b} |\Pr\{X = a, Z = b\} - \Pr\{Y = a, Z = b\}|\\ &= \frac{1}{2}\sum_{a,b} |\Pr\{X = a\}\Pr\{Z = b\} - \Pr\{Y = a\}\Pr\{Z = b\}|\\ &= \frac{1}{2}\sum_{a} |\Pr\{X = a\} - \Pr\{Y = a\}| \sum_{b} \Pr\{Z = b\}\\ &= \frac{1}{2}\sum_{a} |\Pr\{X = a\} - \Pr\{Y = a\}|\\ &= \Delta(X,Y). \quad \square \end{aligned}$$

Notice that if Z is not independent from X or Y, then the proposition is not necessarily true. Consider for example two identically distributed, but independent, random variables X, Y, and let $Z = Y$. Then $\Delta((X,Z),(Y,Z)) = \Delta((X,Y),(Y,Y))$ is nonzero (unless X is trivial), while $\Delta(X,Y) = 0$ because X and Y are identically distributed. Using Proposition 8.8 and the triangle inequality we get the following useful bound.

PROPOSITION 8.9 *Let $X_1, \ldots, X_k$ and $Y_1, \ldots, Y_k$ be two lists of totally independent random variables. Then*

$$\Delta((X_1, \ldots, X_k), (Y_1, \ldots, Y_k)) \leq \sum_{i=1}^{k} \Delta(X_i, Y_i). \tag{8.22}$$

Proof: We prove the inequality for lists of length 2. The general case follows by induction on k. By triangle inequality, the statistical distance $\Delta((X_1, X_2), (Y_1, Y_2))$ is at most

$$\Delta((X_1, X_2), (X_1, Y_2)) + \Delta((X_1, Y_2), (Y_1, Y_2)).$$

By Proposition 8.8, and using the independence of X_1 and Y_2 from the other variables, these two terms are at most $\Delta(X_2, Y_2) + \Delta(X_1, Y_1)$. □

The following proposition shows that applying a (possibly randomized) function to two distributions does not increase the statistical distance.

PROPOSITION 8.10 *Let X, Y be two random variables over a common set A. For any (possibly randomized) function f with domain A, the statistical distance between $f(X)$ and $f(Y)$ is at most*

$$\Delta(f(X), f(Y)) \leq \Delta(X, Y) \tag{8.23}$$

Proof: We first consider the case of (deterministic) functions. Let f be a function from set A to some other set B. Then,

$$\begin{aligned}
\Delta(f(X), f(Y)) &= \frac{1}{2} \sum_{b \in B} |\Pr\{f(X) = b\} - \Pr\{f(Y) = b\}| \\
&= \frac{1}{2} \sum_{b \in B} \left| \sum_{a \in f^{-1}(b)} (\Pr\{X = a\} - \Pr\{Y = a\}) \right| \\
&\leq \frac{1}{2} \sum_{b \in B} \sum_{a \in f^{-1}(b)} |\Pr\{X = a\} - \Pr\{Y = a\}| \\
&= \frac{1}{2} \sum_{a \in A} |\Pr\{X = a\} - \Pr\{Y = a\}| \\
&= \Delta(X, Y).
\end{aligned}$$

So, (8.23) holds true for every function f. Combining Proposition 8.8 with (8.23), we immediately get that (8.23) holds also for randomized functions. □

Notice that $\Delta(f(X), f(Y))$ can be strictly less than $\Delta(X, Y)$. For example, if $f(x) = c$ is a constant function, then $\Delta(f(X), f(Y)) = 0$ regardless of $\Delta(X, Y)$. However, if f is injective, then equality holds and $\Delta(f(X), f(Y)) = \Delta(X, Y)$.

PROPOSITION 8.11 *If X and Y are random variables over set A and $f\colon A \to [a, b]$ is a real valued function, then*

$$|\,Exp[f(X)] - Exp[f(Y)]| \leq |b - a| \cdot \Delta(X, Y) \qquad (8.24)$$

Proof: Define function $g(x) = f(x) - \frac{a+b}{2}$. Notice that $|g(x)| \leq \frac{b-a}{2}$ for all x, and $g(X) - g(Y) = f(X) - f(Y)$. Therefore

$$\begin{aligned}
|\,\mathrm{Exp}[f(X)] - \mathrm{Exp}[f(Y)]| &= |\,\mathrm{Exp}[g(X)] - \mathrm{Exp}[g(Y)]| \\
&= \left|\sum_a g(a)\Pr\{X = a\} - \sum_a g(a)\Pr\{Y = a\}\right| \\
&\leq \sum_a |g(a)| \cdot |\Pr\{X = a\} - \Pr\{Y = a\}| \\
&\leq |b - a| \cdot \Delta(X, Y). \qquad \square
\end{aligned}$$

Using the statistical distance, we can reformulate Proposition 8.3 as follows.

COROLLARY 8.12 *Let $(G, +)$ be a finite group and let $A_1, \ldots, A_k$ be k independent (but possibly not identically distributed) random variables over G such that for any $g \in G$*

$$\left|\Pr\{A_i = g\} - \frac{1}{|G|}\right| \leq \frac{1}{2|G|} \qquad (8.25)$$

for all $i = 1, \ldots, k$. Then, the statistical distance between their sum $A = \sum_{i=1}^{k} A_i$ and the uniform distribution U over G is at most

$$\Delta\left(\sum_{i=1}^{k} A_i, U\right) \leq 2^{-(k+1)} \qquad (8.26)$$

Proof: We know from Proposition 8.3 that the sum $\sum_i A_i$ satisfies (8.2). Therefore, the statistical distance from uniform is

$$\Delta\left(\sum_i A_i, U\right) = \frac{1}{2}\sum_{g \in G}\left|\Pr\left\{\sum_i A_i = g\right\} - \frac{1}{|G|}\right| \leq \frac{1}{2} \cdot \frac{1}{2^k}. \qquad \square$$

2. Collision resistant hash functions

Let q be any positive integer, $\mathbb{Z}_q$ the set of all integers modulo q, and $\mathbf{A}$ be an $n \times m$ matrix with entries in $\mathbb{Z}_q$. Matrix $\mathbf{A}$ is naturally associated with a linear function

$$f_{\mathbf{A}}\colon \mathbf{x} \mapsto \mathbf{A}\mathbf{x} \bmod q$$

from $\mathbb{Z}_q^m$ to $\mathbb{Z}_q^n$. Clearly, function $f_{\mathbf{A}}$ is easily computable in both directions: computing $f_{\mathbf{A}}$ in the forward direction is just a matrix-vector multiplication, while inverting $f_{\mathbf{A}}$ is essentially the problem of solving a system of linear equations modulo q. In (Ajtai, 1996), Ajtai proved that, when $\mathbf{A}$ is chosen uniformly at random, a suitable restriction of function $f_{\mathbf{A}}$ is at least as hard to invert on the average as the worst case complexity of approximating certain lattice problems within a polynomial factor. Subsequently, Goldreich, Goldwasser and Halevi observed that under essentially the same complexity assumption as Ajtai's, it is possible to prove that a similarly restricted function $h_{\mathbf{A}}$ is collision resistant (Goldreich et al., 1997b), i.e., given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, it is computationally hard to find two distinct input vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^m$ such that $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{y}$. The Ajtai-GGH hash function

$$h_{\mathbf{A}}(\mathbf{x}) = \sum_{i:x_i=1} \mathbf{a}_i \pmod q$$

is defined as the restriction of function $f_{\mathbf{A}}$ to the set of binary vectors $\mathbf{x} \in \{0,1\}^m$, and the problem of finding collision $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{y}$ is clearly equivalent to finding integer vectors $\mathbf{z} = \mathbf{x} - \mathbf{y}$ such that $\|\mathbf{z}\|_\infty = 1$ and $\mathbf{A}\mathbf{z} = 0 \pmod q$. Notice that if $m > n \log q$, then function h_A is indeed a hash function, i.e., it compresses the size of the input, and collisions are guaranteed to exist. The goal is to prove that collisions are computationally hard to find. In this section we describe a hash function family that generalizes and improves the Ajtai-GGH hash functions, and such that finding collisions for randomly chosen functions is at least as hard as approximating the covering radius of any lattice in the worst case within some polynomial factor $\gamma(n) < O(n^{2.5} \log n)$. Using the transference theorems from Chapter 7, we get that finding collisions is at least as hard as approximating the length of the shortest vector problem in any lattice within factors $\gamma(n) < O(n^{3.5} \log n)$. Both factors can be further reduced by up to $\sqrt{n}$ if the closest vector problem for a certain sequence of "almost perfect" lattices can be efficiently solved. (See Subsection 2.4.) Notice that collisions corresponds to short vectors $\|\mathbf{z}\|_2 \leq \sqrt{n}\|\mathbf{s}\|_\infty = \sqrt{n}$ in the lattice $\Lambda_{\mathbf{A}} = \{\mathbf{z} : \mathbf{A}\mathbf{z} = 0 \bmod q\}$. So, the security of the hash function can be reformulated as a connection between finding short nonzero vectors in a lattice on the average, and approximating the length of the shortest nonzero vector for any lattice in the worst case.

2.1 The construction

Let Λ be a full-rank n-dimensional lattice such that the closest vector problem in Λ can be efficiently solved. (Formally, we consider a sequence of full-rank lattices Λ_n, one for every dimension n, such that there exists a polynomial time algorithm CVP_Λ that on input n and $\mathbf{t} \in \mathbb{Q}^n$, finds a lattice vector in Λ_n as close as possible to $\mathbf{t}$.)

For example, if $\Lambda = \mathbb{Z}^n$, then a lattice vector $\mathbf{x} \in \Lambda$ closest to a given target $\mathbf{t} \in \mathbb{Q}^n$ can be easily found rounding each coordinate of $\mathbf{t}$ to the closest integer $x_i = \lceil t_i \rfloor$. We are interested in lattices Λ that are "almost perfect", as follows. Remember the definition of the packing radius and the covering radius: the packing radius is the largest radius such that (open) spheres centered at distinct lattice points do not intersect, and the covering radius is the smallest radius such that (closed) spheres centered at lattice points cover the entire space. Clearly, the covering radius is always at least as big as the packing radius.

DEFINITION 8.6 *The* packing-covering ratio *of a lattice Λ is the ratio τ between the covering radius and the packing radius of the lattice and it equals $2\rho(\Lambda)/\lambda_1(\Lambda)$. For any $\tau > 1$, a lattice Λ is called τ-perfect if its packing-covering ratio is at most τ. We say that a sequence of lattices Λ_n is* almost prefect *if all lattices Λ_n are τ-perfect for a constant τ independent of the rank n.*

This is analogous to the definition of perfect codes. Codes are sets of strings (called *codewords*) of some fixed length n over a finite alphabet Σ, with the (Hamming) distance between strings measured as the number of positions in which the two strings differ. Then, the packing radius and covering radius of a code are defined as the largest and smallest radii such that the Hamming spheres centered at codewords are disjoint or cover the entire space Σ^n, respectively. A code is called *perfect* if the packing radius equals the covering radius. In other words, the code is perfect if it is possible to partition the entire space Σ^n with equal (Hamming) spheres centered at the codewords. Interestingly perfect codes do exist, but the same is not true for lattices: it is not possible to partition the Euclidean space $\mathbb{R}^n$ with spheres of radius bounded away from 0.

We would like the packing-covering ratio τ of lattice Λ to be as small as possible. Remember that the Voronoi cell of a lattice Λ contains a sphere of radius $\lambda_1(\Lambda)/2$ and it is contained in a sphere of radius $\rho(\Lambda)$. When τ is close to one, then these two radii are almost the same, and the Voronoi cells $\mathcal{V}(\mathbf{x}, \Lambda)$ are almost spherical. So, even if $\mathbb{R}^n$ cannot be partitioned with equal spheres, τ-perfect lattices partition $\mathbb{R}^n$ into equal almost spherical regions. The questions is: how close to a sphere can these regions possibly be? Setting $\Lambda = \mathbb{Z}^n$ to the integer lattice gives $\tau = \sqrt{n}$. However, as we will see in Subsection 2.4, it is possible to do much better than that. For now we assume Λ is a τ-perfect (easily decodable) lattice for some τ between 1 and $\sqrt{n}$.

We use lattice Λ and the corresponding decoding algorithm CVP_Λ to define a hash function as follows. First, we build an almost orthogonal sublattice $\mathcal{L}(\mathbf{M}) \subset \Lambda$. Let α be a scaling factor to be specified, and for all $i = 1, \dots, n$, let $\mathbf{m}_i = \mathrm{CVP}_\Lambda(\alpha\rho(\Lambda)\mathbf{e}_i)$ be a lattice point within distance $\rho(\Lambda)$ from $\alpha\rho(\Lambda)\mathbf{e}_i$. In matrix notation,

$$\mathbf{M} = \alpha\rho(\Lambda)\mathbf{I} + \mathbf{R} \tag{8.27}$$

where $\mathbf{R}$ is a matrix with columns of length bounded by

$$\|\mathbf{r}_i\| \leq \rho(\Lambda). \tag{8.28}$$

Lattices Λ and $\mathcal{L}(\mathbf{M})$ define a finite Abelian group

$$G = \Lambda/\mathcal{L}(\mathbf{M}). \tag{8.29}$$

The elements of group G can be represented using any of the techniques described in Subsection 1.1. The only important properties here are that elements of G can be represented using $\log|G|$ bits, and the group operation can be computed in polynomial time. Moreover, there is an easily computable homomorphism $\psi : \Lambda \to G$ that maps each lattice vector to the corresponding group element. Notice that $\psi(\mathbf{x}) = 0$ in G if and only if vector $\mathbf{x}$ belongs to sublattice $\mathcal{L}(\mathbf{M}) \subset \Lambda$.

We define a family of G-valued hash functions. Let m be an integer, and fix a sequence of m group elements $a_1, \dots, a_m \in G$. The vector $\mathbf{a} = [a_1, \dots, a_m]^T \in G^m$ defines a function $h_\mathbf{a} : \{0,1\}^m \to G$ that maps binary vector $\mathbf{x} \in \{0,1\}^m$ to group element

$$h_\mathbf{a}(\mathbf{x}) = \sum_{i=1}^{m} x_i a_i = \sum\{a_i : x_i = 1\}. \tag{8.30}$$

If $m > \log_2|G|$, then $h_\mathbf{a}$ is a hash function (i.e., a function that compresses the size of its input) and collisions $h_\mathbf{a}(\mathbf{x}) = h_\mathbf{a}(\mathbf{y})$ (with $\mathbf{x} \neq \mathbf{y}$) are guaranteed to exist. We want to prove that if vector $\mathbf{a} \in G^m$ is chosen uniformly at random, then collisions are hard to find. As for the Ajtai-GGH hash function, collisions can be represented as vectors $\mathbf{z} \in \{-1, 0, +1\}^m \backslash \{0\}^m$ such that $\sum_i z_i a_i = 0$. In the rest of this chapter we will refer to such vectors as $h_{\mathbf{a}}$-collisions.

Before proving that $h_{\mathbf{a}}$ is collision resistant, we bound the size of group G from above, so to get an estimate for the length m of the key $\mathbf{a}$.

LEMMA 8.13 *For any constant $\alpha \geq 1$ and any lattice Λ with packing-covering ratio $\tau \leq \sqrt{n}$, let $\mathbf{M}$ be a set of vectors of Λ as defined in (8.27) and (8.28). Then, the elements of group $G = \Lambda/\mathcal{L}(\mathbf{M})$ can be represented with binary strings of length*

$$\log_2 |G| < n\left(\log_2 n + \log_2 \alpha\right).$$

Proof: We need to bound the group size $|G| = \det(\mathcal{L}(\mathbf{M}))/\det(\Lambda)$. We bound the two determinants separately. The columns of $\mathbf{M}$ have length at most

$$\|\alpha\rho(\Lambda)\mathbf{e}_i + \mathbf{r}_i\| \leq \|\alpha\rho(\Lambda)\mathbf{e}_i\| + \|\mathbf{r}_i\| \leq (\alpha + 1)\rho(\Lambda).$$

Therefore, by Hadamard's inequality

$$|\det(\mathbf{M})| \leq (\alpha + 1)^n \rho(\Lambda)^n \leq (2\alpha)^n \rho(\Lambda)^n.$$

To bound the determinant of Λ, we use Minkowski's theorem. By Theorem 1.5, the length of the shortest nonzero vector in Λ satisfies $\lambda_1(\Lambda) < \sqrt{n}\det(\Lambda)^{1/n}$. Therefore, $\det(\Lambda)$ is greater than $(\lambda_1(\Lambda)/\sqrt{n})^n$. Combining the two bounds, we get that group G has cardinality

$$|G| = \frac{\det(\mathbf{M})}{\det(\Lambda)} < \left(\frac{2\alpha\rho(\Lambda)\sqrt{n}}{\lambda_1(\Lambda)}\right)^n = \left(\alpha\tau\sqrt{n}\right)^n. \qquad (8.31)$$

Taking the logarithm of both sides, and using $\tau \leq \sqrt{n}$, we get the bound in the lemma. □

In particular, if α is bounded by a polynomial in n, then group elements can be represented using $O(n \log n)$ bits.

Let $\mathcal{F}$ be a hypothetical collision finder algorithm that on input a randomly chosen vector $\mathbf{a} \in G^m$, outputs (with nonnegligible probability δ) an $h_{\mathbf{a}}$-collision $\mathbf{z}$. We show that, given oracle access to $\mathcal{F}$, one can approximate the length of the covering radius of any lattice $\mathcal{L}(\mathbf{B})$ of rank n within some small factor $\gamma(n)$.

2.2 Collision resistance

Let $\gamma(n)$ be any function slightly bigger than $\tau n^2 \log n$:

$$\omega(\tau n^2 \log n) \leq \gamma(n) \leq \tau n^2 \log^2 n. \qquad (8.32)$$

For example one can set $\gamma(n) = \tau n^2 \log^2 n$, but the smaller the better. Set the scaling factor α to

$$\alpha = \frac{\gamma(n)}{8\sqrt{n}\tau}. \tag{8.33}$$

Notice that this choice of α satisfies

$$n^{1.5} \log n \leq \alpha \leq n^{1.5} \log^2 n.$$

Let the length of the key $\mathbf{a}$ be $m = 2\log_2 |G|$ so that $h_{\mathbf{a}}$ is a family of hash functions that compress the size of their input by a factor 2. By Lemma 8.13, this value satisfies

$$m \leq 6n \log_2 n. \tag{8.34}$$

We want to prove that finding $h_{\mathbf{a}}$-collisions when $\mathbf{a}$ is chosen uniformly at random is at least as hard as approximating the covering radius of *any* lattice within a factor $\gamma(n)$. Formally, we give a polynomial time probabilistic reduction from the promise problem GAPCRP$_\gamma$ to the problem of finding $h_{\mathbf{a}}$-collisions when $\mathbf{a}$ is chosen at random. More specifically, given access to a collision finder algorithm $\mathcal{F}$ that on input a random $\mathbf{a} \in G^m$ outputs (with nonnegligible probability over the choice of $\mathbf{a}$) an $h_{\mathbf{a}}$-collision $\mathcal{F}(\mathbf{a})$, we show how to efficiently solve GAPCRP$_\gamma$ for any approximation factor $\gamma(n) = \omega(\tau n^2 \log n)$, where τ is the packing-covering ratio of Λ. For example, if $\Lambda = \mathbb{Z}^n$, then the corresponding hash function is as hard to break as approximating the covering radius of any lattice within any factor $\gamma(n) = \omega(n^{2.5} \log n)$. We remark that while the collision finder algorithm $\mathcal{F}$ is required to work only for a nonnegligible fraction of the keys $\mathbf{a}$, the reduction should correctly solve (with high probability) any GAPCRP$_\gamma$ instance $(\mathbf{B}, r)$. We will give a randomized reduction that rejects all NO instances (with probability 1), and accepts all YES instances with probability exponentially close to 1. In particular, when the reduction accepts an instance $(\mathbf{B}, r)$, the randomness used by the reduction constitutes an NP proof that $\rho(\mathbf{B}) \leq \gamma r$, and therefore $(\mathbf{B}, r)$ is not a NO instance. (From the promise that $(\mathbf{B}, r)$ is either a YES or a NO instance, we can also deduce that $(\mathbf{B}, r)$ is a YES instance, i.e., $\rho(\mathbf{B}) \leq r$. However, if $(\mathbf{B}, r)$ does not satisfies the promise, then the covering radius can be as large as γr.) In fact, the reduction produces more compact and informative NP proofs than the whole sequence of coin tosses used in the reduction process. The short proof produced by the reduction consists of a sequence of n linearly independent vectors $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_n]$ in $\mathcal{L}(\mathbf{B})$ such that the length of the diagonal of the orthogonalized parallelepiped $\sigma(\mathbf{S}) = \sqrt{\sum_i \|\mathbf{s}_i^*\|^2}$ is at most $2\gamma \cdot r$. Since, by Theorem 7.9, all linearly independent sets $\mathbf{S}$ satisfy $\sigma(\mathbf{S}) \geq 2\rho(\mathbf{B})$, this proves that

$$\rho(\mathbf{B}) \leq \frac{\sigma(\mathbf{S})}{2} \leq \gamma r$$

and $\mathbf{S}$ is an NP-witness for GAPCRP$_\gamma$ instance $(\mathbf{B}, r)$. Notice that vectors $\mathbf{S}$ not only prove that for every target point $\mathbf{t} \in \operatorname{span}(\mathbf{B})$ there exists a lattice vector $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ within distance γr from $\mathbf{t}$, but also allow to algorithmically find such lattice vector in polynomial time, for example using the nearest plane algorithm from Chapter 2.

The idea of the reduction is the following. Given a basis $\mathbf{B}$ we want to find linearly independent vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$ in $\mathcal{L}(\mathbf{B})$ such that $\sigma(\mathbf{S})$ is at most $2\gamma(n) \cdot \rho$. We proceed iteratively as follows. We start from $\mathbf{S} = \mathbf{B}$, and assume for simplicity that vectors are sorted according to their lengths $\|\mathbf{s}_1\| \leq \|\mathbf{s}_2\| \leq \cdots \leq \|\mathbf{s}_n\|$. These vectors are clearly linearly independent, but they are potentially much longer than $\rho(\mathbf{B})$. We show that if $\sigma(\mathbf{S}) \geq 2\gamma\rho(\mathbf{B})$ then we can efficiently find (with nonnegligible probability) a new lattice vector $\mathbf{s} \in \mathcal{L}(\mathbf{B})$ linearly independent from $\mathbf{s}_1, \ldots, \mathbf{s}_{n-1}$ such that $\|\mathbf{s}\| \leq \frac{1}{2}\|\mathbf{s}_n\|$. So, we can replace $\mathbf{s}_n$ with $\mathbf{s}$, possibly sort the vectors again according to their lengths, and proceed with another iteration. Since lattices are discrete objects, the length of the vectors $\mathbf{s}_i$ cannot be reduced indefinitely, and at some point the iterative step must fail. If the iterative step repeatedly fails to find a short vector $\mathbf{s}$, then it must be the case (with very high probability) that the assumption $\sigma(\mathbf{S}) \geq 2\gamma\rho(\mathbf{B})$ is false, i.e., the set of vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$ satisfies $\sigma(\mathbf{S}) < 2\gamma\rho(\mathbf{B})$. Details follow.

As outlined above, the main component of the reduction is a solution to the problem described in the following proposition.

PROPOSITION 8.14 *Let Λ be a full rank n-dimensional τ-perfect lattice such that the closest vector problem in Λ can be solved in polynomial time, and let $\mathbf{M}$, G, γ, α and m be as defined in (8.27), (8.29), (8.32), (8.33) and (8.34). Let $\mathcal{F} : G^m \to \{-1, 0, +1\}^m \backslash \{0\}^m$ be a function such that $\mathcal{F}(\mathbf{u})$ is an $h_{\mathbf{u}}$-collision for a nonnegligible fraction δ of the inputs $\mathbf{u} \in G^m$. Let $\mathbf{B} \in \mathbb{Z}^{n \times n}$ be a basis and $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_n]$ a sequence of linearly independent lattice vectors in $\mathcal{L}(\mathbf{B})$ such that $\|\mathbf{s}_n\| = \max_i \|\mathbf{s}_i\|$ and $\sigma(\mathbf{S}) \geq 2\gamma(n)\rho(\mathbf{B})$. Given $\mathbf{B}, \mathbf{S}$ and oracle access to $\mathcal{F}$ one can efficiently find (with probability $\Omega(\delta)$) a lattice vector $\mathbf{s} \in \mathcal{L}(\mathbf{B})$ linearly independent from $\mathbf{s}_1, \ldots, \mathbf{s}_{n-1}$ such that $\|\mathbf{s}\| \leq \frac{1}{2}\|\mathbf{s}_n\|$. Moreover, the reduction makes only one call to $\mathcal{F}$.*

In the formulation of the problem above, we made the simplifying assumption that the collision finder $\mathcal{F}$ is deterministic. We will see, in the proof of Theorem 8.15 below, that this assumption is not restrictive.

We emphasize that while the success probability of collision finder $\mathcal{F}$ is computed with respect to the random choice of vector $\mathbf{a} \in G^m$, a solution to Proposition 8.14 is required to work for *any* lattice $\mathbf{B}$, and its success probability is computed only with respect to the internal randomness of the reduction. (Notice that the sequence $\mathbf{a}$ passed as input to oracle $\mathcal{F}$ possibly depends on this internal randomness. That is why the probability of finding $\mathbf{s}$ depends on the average-case success probability δ of the collision finder function. However, no randomization on input lattice $\mathbf{B}$ is performed.)

The proof of Proposition 8.14 will be given in Section 2.3. In the rest of this section we use Proposition 8.14 to prove that function $h_{\mathbf{a}}$ is collision resistant.

THEOREM 8.15 *Let Λ_n be a sequence of τ-perfect lattices (with $\tau(n)$ possibly a function of the rank n) such that the closest vector problem in Λ_n can be solved in polynomial time. Let also G, $h_{\mathbf{a}}$ and γ be as defined in (8.29), (8.30) and (8.32). If there exists a probabilistic polynomial time algorithm $\mathcal{F}$ that finds collisions $h_{\mathbf{a}}(\mathbf{z}) = 0$ with nonnegligible probability δ when $\mathbf{a}$ is chosen uniformly at random, then* GAPCRP$_\gamma$ *can be solved in* RP *(random polynomial time). Moreover, on input a* YES *instance $(\mathbf{B}, r)$, the* GAPCRP$_\gamma$ *algorithm produces an equivalent basis* $\mathbf{S}$ *such that $\sigma(\mathbf{S}) \leq 2\gamma r$ with probability exponentially close to* 1.

Proof: Let δ be the success probability of $\mathcal{F}$. First of all we need to transform this randomized collision finder $\mathcal{F}$ into a (deterministic) function that (almost certainly) finds collisions for a nonnegligible fraction of the inputs. Let r be the randomness used by $\mathcal{F}$. By Markov inequality, if $\mathcal{F}$ succeed with probability δ when $\mathbf{u}$ and r are chosen at random, then there is at least a $\delta/2$ fraction of the inputs for which $\mathcal{F}(\mathbf{u})$ succeeds with probability $\delta/2$. (Probability computed for a fixed $\mathbf{u}$ only with respect to the choice of the randomness r.) We build a $\mathcal{F}'$ as follows: on input $\mathbf{u}$, run $\mathcal{F}(\mathbf{u})$ $O(n \log(1/\delta))$ times using independent random strings r each time. If any of these runs find an $h_{\mathbf{u}}$-collision, then output it. If not, return any element of $\{-1, 0, +1\}^m \setminus \{0\}^m$. All queries are stored, so that if a query $\mathbf{u}$ is asked twice, then the same $\mathcal{F}'(\mathbf{u})$ is returned both times, and $\mathcal{F}'$ behaves like a function. It is easy to see that with probability exponentially close to 1, $\mathcal{F}'$ correctly answers at least a $\delta/2$ fraction of the queries. The rest of the proof follows the outline given at the beginning of this subsection and it is left to the reader as an exercise. □

2.3 The iterative step

In this section we show how to use a collision finder function $\mathcal{F}$ to find short vectors in any lattice $\mathcal{L}(\mathbf{B})$.

Proof [of Proposition 8.14]: Let $\Lambda, \mathbf{M}, G, \tau$ and $\mathcal{F}$ be as defined in the proposition. Let $\mathcal{G}$ be the set of all $\mathbf{g} \in G^m$ such that $\mathcal{F}(\mathbf{g})$ is an $h_{\mathbf{g}}$-collision. We know that if $\mathbf{u} \in G^m$ is chosen uniformly at random, then

$$\Pr\{\mathbf{u} \in \mathcal{G}\} = \delta$$

for some nonnegligible function $\delta(n)$.

Let $\mathbf{B}$ be a full-rank n-dimensional lattice basis, and $\mathbf{S}$ a set of linearly independent lattice vectors such that $\sigma(\mathbf{S}) \geq 2\gamma\rho(\mathbf{B})$. We want to use $\mathcal{F}$ to find a lattice vector $\mathbf{s} \in \mathcal{L}(\mathbf{B})$ such that $\mathbf{s} \notin \text{span}(\mathbf{s}_1, \ldots, \mathbf{s}_{n-1})$ and $\|\mathbf{s}\| \leq \|\mathbf{s}_n\|/2$ with probability $\Omega(\delta)$. Define the scaling factor

$$\beta = \frac{\sqrt{n}\sigma(\mathbf{S})}{\alpha\rho(\Lambda)} \tag{8.35}$$

and consider the almost orthogonal matrix $\beta\mathbf{M}$. We use $\mathbf{S}$ to approximate each vector $\beta\mathbf{m}_i$ with a lattice point $\mathbf{c}_i \in \mathcal{L}(\mathbf{B})$. In particular, using the nearest plane algorithm of Chapter 2, we find, for $i = 1, \ldots, n$, a lattice point $\mathbf{c}_i \in \mathcal{L}(\mathbf{S}) \subseteq \mathcal{L}(\mathbf{B})$ within distance $\sigma(\mathbf{S})/2$ from $\beta\mathbf{m}_i$. Let $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_n]$. Using matrix notation,

$$\mathbf{C} = \beta\mathbf{M} + \mathbf{Q}, \tag{8.36}$$

where $\mathbf{Q}$ is a matrix with columns of length at most

$$\|\mathbf{q}_i\| \leq \sigma(\mathbf{S})/2. \tag{8.37}$$

Define the integer

$$k = 3\log_2 n + \log(1/\delta). \tag{8.38}$$

Notice that since δ is nonnegligible, $\delta \geq 1/n^c$ for some constant c independent of n, and $\log(1/\delta) = O(\log n)$. In particular, k is also $O(\log n)$. Sample mk group elements $[\mathbf{x}_{i,j}]_{\mathbf{C}}$ ($i \leq m$ and $j \leq k$) in $\mathcal{L}(\mathbf{B})/\mathcal{L}(\mathbf{C})$, and for every $\mathbf{x}_{i,j}$, let $\mathbf{x}'_{i,j} = \mathbf{M}\mathbf{C}^{-1}\mathbf{x}_{i,j}$. (See Figure 8.3.) Equivalently, one can choose $[\mathbf{x}'_{i,j}]_{\mathbf{M}}$ uniformly at random in $\mathcal{L}(\mathbf{B}')/\mathcal{L}(\mathbf{M})$, where $\mathbf{B}' = \mathbf{M}\mathbf{C}^{-1}\mathbf{B}$, and set $\mathbf{x}_{i,j} = \mathbf{C}\mathbf{M}^{-1}\mathbf{x}'_{i,j}$. (We do not specify at this point the choice of representatives $\mathbf{x}_{i,j}$ and $\mathbf{x}'_{i,j}$.) Then, use the decoding algorithm CVP_Λ to find a lattice point $\mathbf{w}'_{i,j} \in \Lambda$ within distance $\rho(\Lambda)$ from $\mathbf{x}'_{i,j}$. Let also $a_{i,j} = \psi(\mathbf{w}'_{i,j})$ be the group element corresponding to

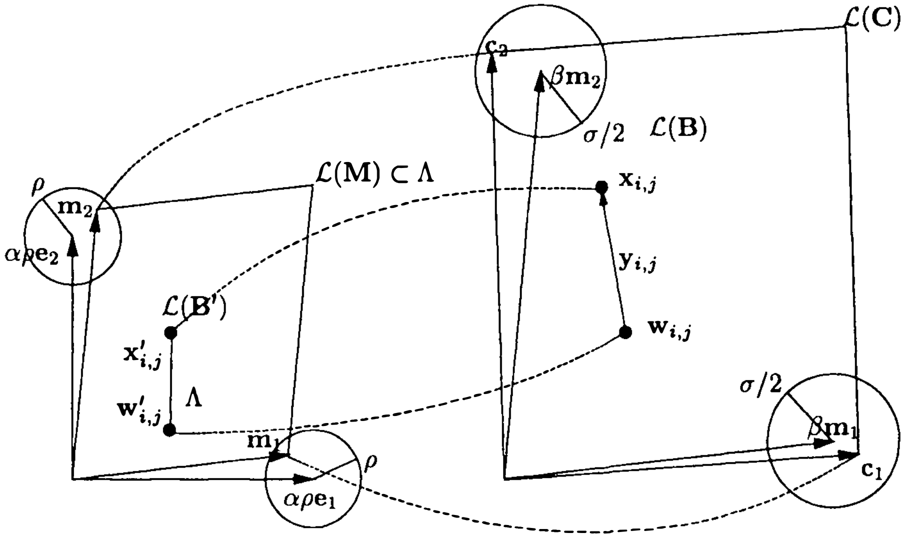


Figure 8.3. Connecting lattices

lattice point $\mathbf{w}'_{i,j}$ and for every $i = 1, \ldots, m$, define $a_i = \sum_{j=1}^{k} a_{i,j}$. Pass $\mathbf{a} = [a_1, \ldots, a_m]^T$ as input to the collision finder to get a vector $\mathbf{z} = \mathcal{F}(\mathbf{a})$. For every i, j, let $\mathbf{w}_{i,j} = \mathbf{C}\mathbf{M}^{-1}\mathbf{w}'_{i,j}$ and define $\mathbf{y}_{i,j} = \mathbf{x}_{i,j} - \mathbf{w}_{i,j}$. The output of the reduction is the vector

$$\mathbf{s} = \sum_{i=1}^{m} z_i \sum_{j=1}^{k} \mathbf{y}_{i,j}. \tag{8.39}$$

Before analyzing the success probability of the reduction, we introduce some useful conventions about the lattice decoding algorithm CVP_Λ and the choice of representatives of the sampling procedure. We assume the following:

- For any vector $\mathbf{t} \in \mathbb{R}^m$ and $\mathbf{v} \in \Lambda$,

$$\text{CVP}_\Lambda(\mathbf{x}' + \mathbf{v}) = \text{CVP}_\Lambda(\mathbf{x}') + \mathbf{v}. \tag{8.40}$$

(If CVP_Λ is randomized, then (8.40) should be interpreted as equality between probability distributions.) This property can be easily achieved modifying CVP_Λ as follows: on input $\mathbf{x}'$, compute $\mathbf{x}'' = \mathbf{x}' \bmod \Lambda$ and output $\mathbf{w}' = \text{CVP}_\Lambda(\mathbf{x}'') - \mathbf{x}'' + \mathbf{x}'$. The reader can easily verify that this modified decoding procedure also solves the closest vector problem in Λ, and it satisfies (8.40). Notice that this property implies that vector $\mathbf{y}_{i,j} = \mathbf{CM}^{-1}(\mathbf{x}'_{i,j} - \mathrm{CVP}_\Lambda(\mathbf{x}'_{i,j}))$ does not depend on the representatives used for $\mathbf{x}'_{i,j} \in \mathcal{L}(\mathbf{B}')/\mathcal{L}(\mathbf{M})$. In particular, the final output $\mathbf{s}$ (8.39) does not depend on the choice of representatives.

- If $\mathbf{x}'$ is chosen uniformly at random from $\mathcal{L}(\mathbf{B}')/\mathcal{L}(\mathbf{M})$, then

$$\operatorname*{Exp}_{\mathbf{x}'}[\mathbf{x}' - \mathrm{CVP}_\Lambda(\mathbf{x}')] = \mathbf{0}.$$

In particular, the distribution of the difference vectors $\mathbf{y}_{i,j} = \mathbf{x}_{i,j} - \mathbf{w}_{i,j}$ satisfies

$$\mathrm{Exp}[\mathbf{y}_{i,j}] = \mathbf{CM}^{-1}\,\mathrm{Exp}[\mathbf{x}'_{i,j} - \mathrm{CVP}_\Lambda(\mathbf{x}'_{i,j})] = \mathbf{0}. \qquad (8.41)$$

This property is easily achieved modifying CVP_Λ as follows: on input $\mathbf{x}'$, choose $b \in \{0,1\}$ uniformly at random and output vector $(-1)^b\mathrm{CVP}_\Lambda((-1)^b\mathbf{x}')$.

- Let $\Lambda' \subset \Lambda$ be a set of representatives for $\Lambda/\mathcal{L}(\mathbf{M})$. For example, let $\Lambda' = \Lambda \cap \mathcal{P}(\mathbf{M})$. We assume that, for all $i = 1,\ldots,m$ and $j = 1,\ldots,k$, vector $\mathbf{w}'_{i,j}$ belongs to Λ'. This property can be achieved by appropriately choosing the set of representatives used for $\mathbf{x}'_{i,j}$ and modifying the sampling procedure accordingly. For example, we first choose an auxiliary vector $\mathbf{x}''_{i,j}$ uniformly at random from $\mathcal{L}(\mathbf{B}')/\mathcal{L}(\mathbf{M})$. (At this point, which representative is used does not matter.) Then we compute $\mathbf{w}''_{i,j} = \mathrm{CVP}_\Lambda(\mathbf{x}''_{i,j})$ and find the unique element $\mathbf{w}'_{i,j} \in \Lambda'$ congruent to $\mathbf{w}''_{i,j}$ modulo $\mathbf{M}$. Finally we set $\mathbf{x}'_{i,j} = \mathbf{x}''_{i,j} - \mathbf{w}''_{i,j} + \mathbf{w}'_{i,j}$.

The conventions just described are not strictly necessary, but they help simplifying the reduction. So, we assume that $\mathbf{a}$ and $\mathbf{s}$ are chosen according to the procedure described above. We want to prove that $\mathbf{s} \in \mathcal{L}(\mathbf{B})$, $\mathbf{s} \notin \mathrm{span}(\mathbf{s}_1,\ldots,\mathbf{s}_{n-1})$ and $\|\mathbf{s}\| \leq \|\mathbf{s}_n\|/2$ with probability $\Omega(\delta)$. We will prove the following:

1 If $\mathbf{a} \in \mathcal{G}$ (i.e., if $\mathcal{F}(\mathbf{a})$ is an $h_\mathbf{a}$-collision) then $\mathbf{s} \in \mathcal{L}(\mathbf{B})$

2 The distribution of vector $\mathbf{a} = [a_1,\ldots,a_m]^T$ is statistically close to uniform over G^m, and, in particular, $\Pr\{\mathbf{a} \in \mathcal{G}\} = \delta \cdot (1 - o(1))$.

3 The conditional probability that $\mathbf{s} \notin \mathrm{span}(\mathbf{s}_1,\ldots,\mathbf{s}_{n-1})$ (or, equivalently, $\mathbf{s}$ is not orthogonal to $\mathbf{s}^*_n$, written $\mathbf{s} \not\perp \mathbf{s}^*_n$), given that $\mathbf{a} \in \mathcal{G}$, is at least $1/6$.

4 The probability that $\mathbf{a} \in \mathcal{G}$, but $\|\mathbf{s}\| > \|\mathbf{s}_n\|/2$, is at most $\delta \cdot o(1)$.

It follows that the success probability of the reduction is

$$\begin{aligned}
&\Pr\{\mathbf{s} \in \mathcal{L}(\mathbf{B}) \wedge \mathbf{s} \not\sim \mathbf{s}_n^* \wedge \|\mathbf{s}\| \leq \|\mathbf{s}_n\|/2\} \\
&\geq \Pr\{\mathbf{a} \in \mathcal{G} \wedge \mathbf{s} \not\sim \mathbf{s}_n^* \wedge 2\|\mathbf{s}\| \leq \|\mathbf{s}_n\|\} \\
&\geq \Pr\{\mathbf{a} \in \mathcal{G} \wedge \mathbf{s} \not\sim \mathbf{s}_n^*\} - \Pr\{\mathbf{a} \in \mathcal{G} \wedge 2\|\mathbf{s}\| > \|\mathbf{s}_n\|\} \\
&= \Pr\{\mathbf{a} \in \mathcal{G}\} \Pr\{\mathbf{s} \not\sim \mathbf{s}_n^* \mid \mathbf{a} \in \mathcal{G}\} - \Pr\{\mathbf{a} \in \mathcal{G} \wedge 2\|\mathbf{s}\| > \|\mathbf{s}_n\|\} \\
&\geq \delta(1 - o(1)) \cdot \frac{1}{6} - \delta \cdot o(1) \\
&= \frac{\delta \cdot (1 - o(1))}{6} = \Omega(\delta).
\end{aligned}$$

We first prove that if $\mathbf{a} \in \mathcal{G}$, then $\mathbf{s}$ is a lattice point. Remember that $\mathbf{a} \in \mathcal{G}$ if and only if $\mathbf{z} = \mathcal{F}(\mathbf{a})$ is an $h_{\mathbf{a}}$-collision, i.e., $\sum_i z_i a_i = 0$. Consider the vector $\mathbf{w}' = \sum_{i=1}^m z_i \sum_{j=1}^k \mathbf{w}'_{i,j}$. Since all $\mathbf{w}'_{i,j}$ belong to Λ, also $\mathbf{w}'$ is a lattice point of Λ and we can apply the homomorphism $\psi : \Lambda \rightarrow G$ to $\mathbf{w}'$. The group element corresponding to lattice vector $\mathbf{w}'$ is

$$\psi(\mathbf{w}') = \sum_{i=1}^m z_i \sum_{j=1}^k \psi(\mathbf{w}_{i,j}) = \sum_{i=1}^m z_i \sum_{j=1}^k a_{i,j} = \sum_{i=1}^m z_i a_i = 0. \qquad (8.42)$$

Since G is the quotient of Λ modulo $\mathcal{L}(\mathbf{M})$, this proves that $\mathbf{w}' \in \mathcal{L}(\mathbf{M})$, i.e., $\mathbf{w}' = \mathbf{M}\mathbf{v}$ for some integer vector $\mathbf{v}$. Now, substituting $\mathbf{y}_{i,j} = \mathbf{x}_{i,j} - \mathbf{w}_{i,j}$ in the definition of $\mathbf{s}$ we get

$$\begin{aligned}
\mathbf{s} &= \sum_{i,j} z_i(\mathbf{x}_{i,j} - \mathbf{w}_{i,j}) \\
&= \sum_{i,j} z_i \mathbf{x}_{i,j} - \sum_{i,j} z_i \mathbf{w}_{i,j} \\
&= \sum_{i,j} z_i \mathbf{x}_{i,j} - \sum_{i,j} z_i \mathbf{C}\mathbf{M}^{-1}\mathbf{w}'_{i,j} \\
&= \sum_{i,j} z_i \mathbf{x}_{i,j} - \mathbf{C}\mathbf{M}^{-1}\mathbf{w}' \\
&= \sum_{i,j} z_i \mathbf{x}_{i,j} - \sum_l v_l \mathbf{c}_l
\end{aligned}$$

which is a lattice vector because $\mathbf{x}_{i,j}, \mathbf{c}_l \in \mathcal{L}(\mathbf{B})$ and v_l, z_i are integers. This proves that if $\mathbf{a} \in \mathcal{G}$ then $\mathbf{s} \in \mathcal{L}(\mathbf{B})$.

We now show that the distribution of $(a_1, \ldots, a_m)$ is very close to uniform. We first show that the probability distribution of each $a_{i,j}$ is not too far from uniform. Then, we use Corollary 8.12 to prove that the statistical distance of $a_i = \sum_j a_{i,j}$ from uniform is very small. In order to prove that $a_{i,j}$ is not too far from uniform, we need an upper bound on the covering radius of $\mathcal{L}(\mathbf{B}')$.

LEMMA 8.16 *Let α be as defined in (8.33). Then, the covering radius of $\mathbf{B}'$ is at most*

$$\rho(\mathbf{B}') \leq \lambda_1(\Lambda)/(8n).$$

Proof: Let $\mathbf{t}'$ be a deep hole in $\mathcal{L}(\mathbf{B}')$, i.e., a point in $\mathbb{R}^n$ such that $\|\mathbf{t}' - \mathbf{B}'\mathbf{v}\| \geq \rho(\mathbf{B}')$ for every lattice point $\mathbf{B}'\mathbf{v}$. Let $\mathbf{Bv}$ be the lattice point in $\mathcal{L}(\mathbf{B})$ closest to $\mathbf{t} = \mathbf{CM}^{-1}\mathbf{t}'$. Then,

$$\begin{aligned}
\rho(\mathbf{B}) &\geq \operatorname{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \\
&= \|\mathbf{t} - \mathbf{Bv}\| \\
&= \|\mathbf{CM}^{-1}(\mathbf{t}' - \mathbf{B}'\mathbf{v})\| \\
&\geq \|\mathbf{t}' - \mathbf{B}'\mathbf{v}\| \cdot \min_{\mathbf{d}\in\mathbb{R}^n} \frac{\|\mathbf{CM}^{-1}\mathbf{d}\|}{\|\mathbf{d}\|} \\
&\geq \rho(\mathbf{B}') \cdot \min_{\mathbf{d}\in\mathbb{R}^n} \frac{\|\mathbf{Cd}\|}{\|\mathbf{Md}\|}.
\end{aligned}$$

Moreover, for any vector $\mathbf{d} \in \mathbb{R}^n$, we have

$$\frac{\|\mathbf{Cd}\|}{\|\mathbf{Md}\|} = \frac{\|\beta\mathbf{Md} + \mathbf{Qd}\|}{\|\mathbf{Md}\|} \geq \beta - \frac{\|\mathbf{Qd}\|}{\|\mathbf{Md}\|}. \tag{8.43}$$

We compute an upper bound on $\|\mathbf{Qd}\|/\|\mathbf{Md}\|$ that will also be useful later on.

$$\begin{aligned}
\frac{\|\mathbf{Qd}\|}{\|\mathbf{Md}\|} &\leq \frac{\|\mathbf{Qd}\|}{\|\alpha\rho(\Lambda)\mathbf{d}\| - \|\mathbf{Rd}\|} \\
&\leq \frac{\sum_i |d_i| \cdot \|\mathbf{q}_i\|}{\alpha\rho(\Lambda)\|\mathbf{d}\| - \sum_i |d_i| \cdot \|\mathbf{r}_i\|} \\
&\leq \frac{\sum_i |d_i| \cdot \sigma(\mathbf{S})/2}{\alpha\rho(\Lambda)\|\mathbf{d}\| - \sum_i |d_i| \cdot \rho(\Lambda)}.
\end{aligned}$$

But the sum $\sum_i |d_i|$ is at most

$$\sum_{i=1}^{n} |d_i| = \|\mathbf{d}\|_1 \leq \sqrt{n}\|\mathbf{d}\|_2.$$

Substituting this value in the previous expression we get

$$\frac{\|\mathbf{Qd}\|}{\|\mathbf{Md}\|} \leq \frac{\sqrt{n}\sigma(\mathbf{S})}{2\rho(\Lambda)(\alpha - \sqrt{n})} = \left(\frac{\beta}{2}\right)\left(\frac{\alpha}{\alpha - \sqrt{n}}\right) \leq \frac{3}{4}\beta, \tag{8.44}$$

where in the last inequality we used $\alpha \geq 3\sqrt{n}$. This proves that the covering radius of $\mathcal{L}(\mathbf{B})$ satisfies

$$\rho(\mathbf{B}) \geq \rho(\mathbf{B}') \cdot \min_{\mathbf{d}} \frac{\|\mathbf{Cd}\|}{\|\mathbf{Md}\|} \geq \rho(\mathbf{B}') \left(\beta - \max_{\mathbf{d}} \frac{\|\mathbf{Qd}\|}{\|\mathbf{Md}\|}\right) \geq \frac{\rho(\mathbf{B}')\beta}{4}.$$

Finally, solving for $\rho(\mathbf{B}')$ and using (8.35), $\sigma(\mathbf{S}) \geq 2\gamma\rho(\mathbf{B})$, (8.33) and $\tau = 2\rho(\Lambda)/\lambda_1(\Lambda)$, we get

$$\rho(\mathbf{B}') \leq \frac{4\rho(\mathbf{B})}{\beta} = \frac{4\rho(\mathbf{B})\alpha\rho(\Lambda)}{\sqrt{n}\sigma(\mathbf{S})} \leq \frac{2\alpha\rho(\Lambda)}{\gamma\sqrt{n}} = \frac{\lambda_1(\Lambda)}{8n}. \qquad \square$$

We use the bound on the covering radius to estimate the probability that $a_{i,j}$ equals any fixed group element $a \in G$. Remember that $a_{i,j}$ is chosen selecting a lattice point $\mathbf{x}'_{i,j} \in \mathcal{L}(\mathbf{B}')/\mathcal{L}(\mathbf{M})$ uniformly at random, and setting $a_{i,j} = \psi(\text{CVP}_\Lambda(\mathbf{x}'_{i,j}))$. Let $\mathbf{w}'$ be the (unique) lattice point in $\Lambda' \subset \Lambda$ such that $\psi(\mathbf{w}') = a$, and let $\mathcal{Q} = \mathcal{V}(\mathbf{w}', \Lambda)$ and $\bar{\mathcal{Q}} = \bar{\mathcal{V}}(\mathbf{w}', \Lambda)$ be the open and closed Voronoi cells of $\mathbf{w}'$, respectively. In particular, the volumes of $\mathcal{Q}$ and $\bar{\mathcal{Q}}$ equal

$$\text{vol}(\mathcal{Q}) = \text{vol}(\bar{\mathcal{Q}}) = \det(\Lambda).$$

The probability that group element a is chosen is roughly proportional to the number of lattice points of $\mathcal{L}(\mathbf{B}')$ contained in $\bar{\mathcal{Q}}$. (The probability is not exactly proportional to this number because lattice points on the boundary of $\bar{\mathcal{Q}}$ are not necessarily mapped to $\mathbf{w}'$.) Since the number of equivalence classes of $\mathbf{x}'_{i,j} \in \mathcal{L}(\mathbf{B}')$ modulo $\mathcal{L}(\mathbf{M})$ is $\det(\mathbf{M})/\det(\mathbf{B}')$, the probability that $a_{i,j} = a$ satisfies

$$\frac{|\mathcal{L}(\mathbf{B}') \cap \mathcal{Q}| \cdot \det(\mathbf{B}')}{\det(\mathbf{M})} \leq \Pr\{a_{i,j} = a\} \leq \frac{|\mathcal{L}(\mathbf{B}') \cap \bar{\mathcal{Q}}| \cdot \det(\mathbf{B}')}{\det(\mathbf{M})}.$$

Notice that both $\mathcal{Q}$ and $\bar{\mathcal{Q}}$ contain an open sphere of radius $r = \lambda_1(\Lambda)/2$. Therefore, by Proposition 8.7 and using the bound $\rho(\mathbf{B}') \leq \lambda_1(\Lambda)/(8n)$ from Lemma 8.16, the number of lattice points in $\mathcal{Q}$ satisfies

$$\begin{aligned} |\mathcal{L}(\mathbf{B}') \cap \mathcal{Q}| &\geq \frac{\det(\Lambda)}{\det(\mathbf{B}')}\left(1 - \frac{2\rho(\mathbf{B}')n}{\lambda_1(\Lambda)}\right) \\ &\geq \frac{\det(\Lambda)}{\det(\mathbf{B}')}\left(1 - \frac{1}{4}\right) \end{aligned}$$

and the probability that $a_{i,j} = a$ is

$$\Pr\{a_{i,j} = a\} \geq \frac{3}{4}\frac{\det(\Lambda)}{\det(\mathbf{B}')}\frac{\det(\mathbf{B}')}{\det(\mathbf{M})} = \frac{3}{4|G|}.$$

Similarly, the number of lattice points in $\bar{Q}$ satisfies

$$\begin{aligned} |\mathcal{L}(\mathbf{B}') \cap \bar{Q}| &\leq \frac{\det(\Lambda)}{\det(\mathbf{B}')}\left(1 + \frac{2\rho(\mathbf{B}')n}{\lambda_1(\Lambda)}\right) \\ &\leq \frac{\det(\Lambda)}{\det(\mathbf{B}')}\left(1 + \frac{1}{2}\right) \end{aligned}$$

and

$$\Pr\{a_{i,j} = a\} \leq \frac{3}{2}\frac{\det(\Lambda)}{\det(\mathbf{B}')}\frac{\det(\mathbf{B}')}{\det(\mathbf{M})} = \frac{3}{2|G|}.$$

This proves that, for all $a \in G$,

$$\left|\Pr\{a_{i,j} = a\} - \frac{1}{|G|}\right| \leq \frac{1}{2|G|}.$$

So, the probability distribution of each $a_{i,j}$ is not too far from uniform. Adding up a relatively small number of $a_{i,j}$ we get a group element $a_i = \sum_j a_{i,j}$ which is almost uniformly distributed. In particular, by Corollary 8.12, the statistical distance between a_i and a uniformly distributed $u_i \in G$ is at most

$$\Delta(a_i, u_i) \leq \frac{1}{2^{k+1}}. \tag{8.45}$$

Since random variables a_i are independent, by Proposition 8.9 the statistical distance between vector $\mathbf{a} = [a_1, \ldots, a_m]^T$ and a uniformly distributed $\mathbf{u} \in G^m$ is at most

$$\Delta(\mathbf{a}, \mathbf{u}) < \sum_{i=1}^{m} \Delta(a_i, u_i) \leq \frac{m}{2^{k+1}} \tag{8.46}$$

and, using (8.38) and (8.34), we get

$$\Delta(\mathbf{a}, \mathbf{u}) \leq \frac{m\delta}{n^3} \leq \frac{\delta 6n \log_2 n}{n^3} \leq \delta \cdot o(1). \tag{8.47}$$

We use the bound on the statistical distance between **a** and a uniformly distributed **u** to compute the probability that $\mathbf{a} \in \mathcal{G}$. Let φ the characteristic function of set $\mathcal{G}$:

$$\varphi(\mathbf{g}) = \begin{cases} 1 & \text{if } \mathbf{g} \in \mathcal{G} \\ 0 & \text{otherwise} \end{cases}.$$

We want to bound the value of $\Pr\{\mathbf{a} \in \mathcal{G}\} = \operatorname{Exp}[\varphi(\mathbf{a})]$. Notice that function φ takes values in $[0, 1]$. Therefore, by Proposition 8.10,

$$|\operatorname{Exp}[\varphi(\mathbf{a})] - \operatorname{Exp}[\varphi(\mathbf{u})]| \leq \Delta(\mathbf{a}, \mathbf{u}) \leq \delta \cdot o(1). \tag{8.48}$$

This proves that the probability of querying function $\mathcal{F}$ on a point of $\mathcal{G}$ is at least

$$\begin{aligned}\Pr\{\mathbf{a} \in \mathcal{G}\} &= \operatorname{Exp}[\varphi(\mathbf{a})] \\ &\geq \operatorname{Exp}[\varphi(\mathbf{u})] - |\operatorname{Exp}[\varphi(\mathbf{a})] - \operatorname{Exp}[\varphi(\mathbf{u})]| \\ &\geq \delta - \delta \cdot o(1) = \delta \cdot (1 - o(1)).\end{aligned}$$

The same technique used to show that $\mathbf{a}$ is almost uniformly distributed, can be employed to prove that the conditional probability of $\langle \mathbf{s}, \mathbf{s}_n^* \rangle \neq 0$ (given $\mathbf{a} \in \mathcal{G}$) is at least 1/6. In fact, we can prove something stronger: for any fixed $\mathbf{g} \in G^m$, the conditional probability, given $\mathbf{a} = \mathbf{g}$, is at least 1/6. Define auxiliary vectors

$$\begin{aligned}\mathbf{s}' &= \mathbf{M}\mathbf{C}^{-1}\mathbf{s} \\ \hat{\mathbf{s}} &= \mathbf{M}^{-T}\mathbf{C}^{T}\mathbf{s}_n^*.\end{aligned}$$

Notice that $\langle \mathbf{s}', \hat{\mathbf{s}} \rangle = \langle \mathbf{s}, \mathbf{s}_n^* \rangle$. Therefore, $\mathbf{s}$ is orthogonal to $\mathbf{s}_n^*$ if and only if $\mathbf{s}'$ is orthogonal to $\hat{\mathbf{s}}$. We want to bound the (conditional) probability that $\langle \mathbf{s}', \hat{\mathbf{s}} \rangle = 0$. We show that for any fixed $\mathbf{g} \in G^m$

$$\Pr\{\langle \mathbf{s}', \hat{\mathbf{s}} \rangle \neq 0 \mid \mathbf{a} = \mathbf{g}\} \geq \frac{1}{6}.$$

Fix the value of $\mathbf{w}'_{i,j} \in \Lambda'$ for all $i = 1, \ldots, m$ and $j = 1, \ldots, k$. Notice that this uniquely determines also $a_{i,j} = \psi(\mathbf{w}'_{i,j})$, $a_i = \sum_j a_{i,j}$ and $\mathbf{z} = \mathcal{F}(a_1, \ldots, a_n)$. For any i, j, let $\mathcal{Q}_{i,j} = \mathcal{V}(\mathbf{w}'_{i,j}, \Lambda)$ be the (open) Voronoi cell of $\mathbf{w}'_{i,j}$, and let $\bar{\mathcal{Q}}_{i,j}$ be its closure. It is easy to see that (given $\mathbf{a} = \mathbf{g}$) vectors $\mathbf{y}'_{i,j}$ are totally independent, although not in general identically distributed. Moreover, each $\mathbf{w}'_{i,j}$ is distributed almost uniformly in $\bar{\mathcal{Q}}_{i,j} \cap \mathcal{L}(\mathbf{B}')$. (As before, the distribution is not perfectly uniform because points on the boundary of $\bar{\mathcal{Q}}_{i,j}$ are not necessarily mapped to $\mathbf{w}'_{i,j}$, so the points on the boundary might have conditional probability smaller than those in the interior $\mathcal{Q}$.)

Since $\mathbf{z} \neq \mathbf{0}$, there exists a coordinate i such that $z_i = \pm 1$. Assume without loss of generality that $z_1 = \pm 1$. Let Y be the list of all vectors $\mathbf{y}'_{i,j}$ except $\mathbf{y}'_{1,1}$. Variable Y takes values in $\bar{\mathcal{V}}(\mathbf{0}, \Lambda)^{mk-1}$. Fix the value of all vectors in Y and define the constant

$$\alpha = z_1 \langle \mathbf{w}'_{i,j}, \hat{\mathbf{s}} \rangle + \sum_{\mathbf{y} \in Y} z_i \langle \mathbf{y}, \hat{\mathbf{s}} \rangle.$$

The scalar product of $\mathbf{s}'$ and $\hat{\mathbf{s}}$ is

$$\langle \mathbf{s}', \hat{\mathbf{s}} \rangle = \sum_{i,j} z_i \langle \mathbf{y}'_{i,j}, \hat{\mathbf{s}} \rangle = z_1 \langle \mathbf{x}'_{1,1}, \hat{\mathbf{s}} \rangle + \alpha.$$

So, $\langle \mathbf{s}', \hat{\mathbf{s}} \rangle = 0$ if and only if $\mathbf{x}'_{1,1}$ belongs to the hyperplane

$$\mathcal{H}_\alpha = \{\mathbf{x} : z_1 \langle \mathbf{x}, \hat{\mathbf{s}} \rangle + \alpha = 0\}.$$

We want to compute the probability that $\mathbf{x}'_{1,1} \notin \mathcal{H}_\alpha$. Let

$$\mathcal{H}_\beta = \{\mathbf{x} : z_1 \langle \mathbf{x}, \hat{\mathbf{s}} \rangle + \beta = 0\}$$

be the hyperplane parallel to $\mathcal{H}_\alpha$ that passes though $\mathbf{w}'_{1,1}$. (See Figure 8.4.) Hyperplane $\mathcal{H}_\beta$ defines two open half spaces

$$\begin{aligned} \mathcal{H}_\beta^+ &= \{\mathbf{x} \mid z_1 \mathbf{x}^T \hat{\mathbf{s}} + \beta > 0\} \\ \mathcal{H}_\beta^- &= \{\mathbf{x} \mid z_1 \mathbf{x}^T \hat{\mathbf{s}} + \beta < 0\} \end{aligned}$$

and at least one of them does not intersect $\mathcal{H}_\alpha$. Let $\mathcal{Q}'$ be the intersection of the interior of $\mathcal{Q}_{1,1}$ with this open half space, e.g.,

$$\mathcal{Q}' = \begin{cases} \mathcal{Q}_{1,1} \cap \mathcal{H}_\beta^- & \text{if } \alpha \leq \beta \\ \mathcal{Q}_{1,1} \cap \mathcal{H}_\beta^+ & \text{if } \alpha > \beta. \end{cases}$$

The probability that $\mathbf{x}'_{1,1} \notin \mathcal{H}_\alpha$ is at least as large as the probability that $\mathbf{x}'_{1,1} \in \mathcal{Q}'$. This last probability is not smaller than the ratio between the number of lattice points in $\mathcal{Q}'$ and those in $\bar{\mathcal{Q}}_{1,1}$:

$$\Pr\{\mathbf{x}'_{1,1} \notin \mathcal{Q}' \mid \mathbf{a}, Y\} \geq \frac{|\mathcal{L}(\mathbf{B}') \cap \mathcal{Q}'|}{|\mathcal{L}(\mathbf{B}') \cap \bar{\mathcal{Q}}_{1,1}|}. \qquad (8.49)$$

(The reason equality does not necessarily hold is that points on the boundary of $\mathcal{Q}'$ may have probability smaller than points in the interior.) Notice that $\mathcal{Q}'$ is convex because it is the intersection of two convex sets. So, we can use Proposition 8.7 to bound the number of lattice points in $\bar{\mathcal{Q}}_{1,1}$ and $\mathcal{Q}'$.

Since $\mathcal{Q}_{1,1}$ is symmetric with respect to $\mathbf{w}'_{1,1}$, the volume of $\mathcal{Q}'$ is exactly half that of $\mathcal{Q}_{1,1}$. Moreover, $\mathcal{Q}_{1,1}$ contains a sphere of radius $\lambda_1(\Lambda)/2$ centered at $\mathbf{w}'_{1,1}$, and $\mathcal{Q}'$ contains an (open) sphere of radius $\lambda_1(\Lambda)/4$ centered at $w'_{1,1} \pm \lambda_1(\Lambda)/(4\|\hat{\mathbf{s}}\|)\hat{\mathbf{s}}$. (See Figure 8.4.) Therefore, by Proposition 8.7, the number of lattice points in $\mathcal{Q}'$ is at least

$$|\mathcal{L}(\mathbf{B}') \cap \mathcal{Q}'| \geq \frac{\operatorname{vol}(\mathcal{Q}_{1,1})}{2\det(\mathbf{B})}\left(1 - \frac{\rho(\mathbf{B}')n}{\lambda_1(\Lambda)/4}\right). \qquad (8.50)$$

Also, by Proposition 8.7, the number of lattice points in $\bar{\mathcal{Q}}_{1,1}$ is at most

$$|\mathcal{L}(\mathbf{B}') \cap \bar{\mathcal{Q}}_{1,1}| \leq \frac{\operatorname{vol}(\mathcal{Q}_{1,1})}{\det(\mathbf{B}')}\left(1 + \frac{2\rho(\mathbf{B}')n}{\lambda_1(\Lambda)/2}\right). \qquad (8.51)$$

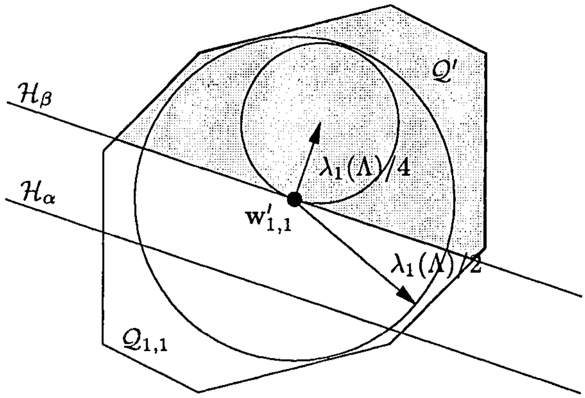


Figure 8.4. Lattice points that avoid a hyperplane

Substituting (8.50) and (8.51) in (8.49) we get

$$\begin{aligned}\Pr\{\mathbf{x}'_{1,1} \not\in \mathcal{H}_\alpha \mid \mathbf{a}, Y\} &\geq \frac{1}{2}\frac{\lambda_1(\Lambda) - 4\rho(\mathbf{B}')n}{\lambda_1(\Lambda) + 4\rho(\mathbf{B}')n} \\ &= \frac{1}{2}\left(1 - \frac{1}{\frac{\lambda_1(\Lambda)}{8\rho(\mathbf{B}')n} + \frac{1}{2}}\right)\end{aligned}$$

and using the bound on the covering radius from Lemma 8.16

$$\Pr\{\mathbf{x}'_{1,1} \not\in \mathcal{H}_\alpha \mid \mathbf{a}, Y\} \geq \frac{1}{2}\left(1 - \frac{1}{1+\frac{1}{2}}\right) = \frac{1}{6}.$$

Averaging over all possible values for $\mathbf{a} \in \mathcal{G}$ and $Y \in \bar{\mathcal{V}}(\mathbf{0}, \Lambda)^{mk-1}$, we get

$$\Pr\{\mathbf{s} \not\perp \mathbf{s}^*_n \mid \mathbf{a} \in \mathcal{G}\} \geq \frac{1}{6}.$$

It only remains to be proved that vector $\mathbf{s}$ is short (with high probability). In the next lemma we bound the length of each $\mathbf{y}_{i,j}$.

LEMMA 8.17 *For any $i = 1, \ldots, m$ and $j = 1, \ldots, k$, the length of $\mathbf{y}_{i,j}$ is at most*

$$\|\mathbf{y}_{i,j}\| \leq \frac{\|\mathbf{s}_n\|}{\omega(\sqrt{n}\log n)}.$$

Proof: The proof is similar to that of Lemma 8.16. From the definition of $\mathbf{y}_{i,j}$ and $\mathbf{w}'_{i,j} = \mathrm{CVP}_\Lambda(\mathbf{x}'_{i,j})$ we get

$$\|\mathbf{y}_{i,j}\| = \|\mathbf{x}_{i,j} - \mathbf{w}_{i,j}\|$$

$$\begin{aligned}
&= \|\mathbf{C}\mathbf{M}^{-1}(\mathbf{x}'_{i,j} - \mathbf{w}'_{i,j})\| \\
&\leq \|\mathbf{x}'_{i,j} - \mathbf{w}'_{i,j}\| \cdot \max_{\mathbf{d}} \frac{\|\mathbf{C}\mathbf{M}^{-1}\mathbf{d}\|}{\|\mathbf{d}\|} \\
&\leq \rho(\Lambda) \cdot \max_{\mathbf{d}} \frac{\|\mathbf{C}\mathbf{d}\|}{\|\mathbf{M}\mathbf{d}\|}.
\end{aligned}$$

Then, we notice that for every vector $\mathbf{d} \in \mathbb{R}^n$,

$$\frac{\|\mathbf{C}\mathbf{d}\|}{\|\mathbf{M}\mathbf{d}\|} = \frac{\|\beta\mathbf{M}\mathbf{d} + \mathbf{Q}\mathbf{d}\|}{\|\mathbf{M}\mathbf{d}\|} \leq \beta + \frac{\|\mathbf{Q}\mathbf{d}\|}{\|\mathbf{M}\mathbf{d}\|}.$$

Finally, we use (8.44) to get

$$\|\mathbf{y}_{i,j}\| \leq \rho(\Lambda)\left(\beta + \max_{\mathbf{d}} \frac{\|\mathbf{Q}\mathbf{d}\|}{\|\mathbf{M}\mathbf{d}\|}\right) < 2\beta\rho(\Lambda).$$

Substituting (8.35) for β, (8.33) for α, (8.38) for k and using $\sigma(\mathbf{S}) \leq \sqrt{n}\|\mathbf{s}_n\|$, we get the bound in the lemma. □

At this point, by triangle inequality we immediately get

$$\|\mathbf{s}\| \leq mk \max_{i,j} \|\mathbf{y}_{i,j}\| \leq \sqrt{n}\log n\|\mathbf{s}_n\|.$$

This bound is not good enough because we want $\|\mathbf{s}\|$ to be smaller than $\|\mathbf{s}_n\|$. In the rest of the proof we give a better probabilistic bound on the length of $\mathbf{s}$. The intuition is that in a random walk the typical distance from the origin grows with the square root of the number of steps, instead of being linear. In our case, the steps are given by vectors $z_i\mathbf{y}_{i,j}$, and the final point is vector $\mathbf{s}$. So, even if $\|\mathbf{s}\|$ can be mk times as long as a single step $z_i\mathbf{y}_{i,j}$, on the average it is much shorter than that. The problem is that since coefficients $\mathbf{z} = \mathcal{F}(\mathbf{a})$ depend on $\mathbf{a}$, and vectors $\mathbf{a}_{i,j}$ and $\mathbf{y}_{i,j}$ are correlated, the steps $z_i\mathbf{y}_{i,j}$ are not independent and general bounds on the average length of random walks cannot be directly applied. Still we can prove that $\mathbf{s}$ is usually not much longer than $\sqrt{mk}$ times $\max_{i,j}\|\mathbf{y}_{i,j}\|$, even under the assumption that $\mathbf{a} \in \mathcal{G}$. Namely, we show that the probability that $\mathbf{a} \in \mathcal{G}$ and $\|\mathbf{s}\| > \|\mathbf{s}_n\|/2$ is at most $\delta \cdot o(1)$. Notice that since $\|\mathbf{s}_n\| > 0$, events $\mathbf{a} \in \mathcal{G}$ (or, equivalently, $\varphi(\mathbf{a}) = 1$) and $\|\mathbf{s}\| > \|\mathbf{s}_n\|/2$ are simultaneously satisfied if and only if $4\varphi(\mathbf{a})\|\mathbf{s}\|^2 > \|\mathbf{s}_n\|^2$. By Markov inequality

$$\begin{aligned}
\Pr\{\mathbf{a} \in \mathcal{G} \wedge \|\mathbf{s}\| > \|\mathbf{s}_n\|/2\} &= \Pr\{\varphi(\mathbf{a})\|\mathbf{s}\|^2 > \|\mathbf{s}_n\|^2/4\} \\
&\leq \frac{4\,\mathrm{Exp}[\varphi(\mathbf{a})\|\mathbf{s}\|^2]}{\|\mathbf{s}_n\|^2}
\end{aligned}$$

We want to prove that $\mathrm{Exp}[\varphi(\mathbf{a})\|\mathbf{s}\|^2]$ is at most $\delta o(1)\|\mathbf{s}_n\|^2$, and therefore $\Pr\{\mathbf{a} \in \mathcal{G} \wedge \|\mathbf{s}\| > \|\mathbf{s}_n\|/2\} \leq \delta \cdot o(1)$. For $i, j \in \{1, \ldots, m\}$ and $\mathbf{g} \in G^m$, define the functions

$$\varphi_{i,j}(\mathbf{g}) = w_i w_j \varphi(\mathbf{g}) \qquad \text{where } \mathbf{w} = \mathcal{F}(\mathbf{g}) \tag{8.52}$$

and let

$$\Gamma(\mathbf{g}) = \sum_{i,j=1}^{m} \sum_{h,l=1}^{k} \langle \mathbf{y}_{i,l}, \mathbf{y}_{j,h} \rangle \varphi_{i,j}(\mathbf{g}).$$

Functions $\varphi_{i,j}$ satisfy $\varphi_{i,j}(\mathbf{a}) = z_i z_j \varphi(\mathbf{a})$, where $\mathbf{z} = \mathcal{F}(\mathbf{a})$ is the output of the collision finder algorithm on input $\mathbf{a}$. Therefore

$$\begin{aligned}
\Gamma(\mathbf{a}) &= \sum_{i,j=1}^{m} \sum_{h,l=1}^{k} \langle \mathbf{y}_{i,l}, \mathbf{y}_{j,h} \rangle \varphi_{i,j}(\mathbf{a}) \\
&= \sum_{i,j=1}^{m} \left\langle \sum_{l=1}^{k} \mathbf{y}_{i,l}, \sum_{h=1}^{k} \mathbf{y}_{j,h} \right\rangle z_i z_j \varphi(\mathbf{a}) \\
&= \left\langle \sum_{i,l} z_i \mathbf{y}_{i,l}, \sum_{j,h} z_j \mathbf{y}_{j,h} \right\rangle \varphi(\mathbf{a}) \\
&= \|\mathbf{s}\|^2 \varphi(\mathbf{a}).
\end{aligned}$$

We want to bound the expectation

$$\mathrm{Exp}[\Gamma(\mathbf{a})] \leq \mathrm{Exp}[\Gamma(\mathbf{u})] + |\,\mathrm{Exp}[\Gamma(\mathbf{a}) - \Gamma(\mathbf{u})]|.$$

We prove that, if $\mathbf{u}$ is chosen uniformly at random, then both $\mathrm{Exp}[\Gamma(\mathbf{u})]$ and $|\,\mathrm{Exp}[\Gamma(\mathbf{a}) - \Gamma(\mathbf{u})]|$ are at most $\delta\|\mathbf{s}_n\|^2 \cdot o(1)$.

We start with $\mathrm{Exp}[\Gamma(\mathbf{u})]$. The key observation is that vector $\mathbf{u}$ is statistically independent from $\mathbf{y}_{i,j}, \mathbf{y}_{h,l}$. Therefore,

$$\begin{aligned}
\mathrm{Exp}[\Gamma(\mathbf{u})] &= \sum_{i,j} \sum_{h,l} \mathrm{Exp}[\langle \mathbf{y}_{i,l}, \mathbf{y}_{j,h} \rangle \varphi_{i,j}(\mathbf{u})] \\
&= \sum_{i,j} \sum_{h,l} \mathrm{Exp}[\langle \mathbf{y}_{i,l}, \mathbf{y}_{j,h} \rangle] \cdot \mathrm{Exp}[\varphi_{i,j}(\mathbf{u})].
\end{aligned}$$

Moreover, unless $(i, l) = (j, h)$, random variables $\mathbf{y}_{i,l}$ and $\mathbf{y}_{j,h}$ are independent and

$$\mathrm{Exp}[\langle \mathbf{y}_{i,l}, \mathbf{y}_{j,h} \rangle] = \langle \mathrm{Exp}[\mathbf{y}_{i,l}], \mathrm{Exp}[\mathbf{y}_{j,h}] \rangle = 0. \tag{8.53}$$

So, by Lemma 8.17, the expectation of $\Gamma(\mathbf{u})$ is

$$\begin{aligned}\mathrm{Exp}[\Gamma(\mathbf{u})] &= \sum_{i=1}^{m}\sum_{l=1}^{k}\mathrm{Exp}[\|\mathbf{y}_{i,l}\|^2]\cdot\mathrm{Exp}[\varphi_{i,i}(\mathbf{u})] && (8.54)\\ &\le \frac{\|\mathbf{s}_n\|^2 mk\delta}{\omega(n\log^2 n)} \le \delta\|\mathbf{s}_n\|^2\cdot o(1). && (8.55)\end{aligned}$$

Now consider the expectation of the difference $\mathrm{Exp}[\Gamma(\mathbf{a})-\Gamma(\mathbf{u})]$. The statistical distance between $(\mathbf{y}_{1,1},\ldots,\mathbf{y}_{m,k},\mathbf{a})$ and $(\mathbf{y}_{1,1},\ldots,\mathbf{y}_{m,k},\mathbf{u})$ is not necessarily small. So, in order to bound the expectation of $\Gamma(\mathbf{a}) - \Gamma(\mathbf{u})$, we first break this expression into smaller components as follows. Let $f_{i,j}:\mathbb{R}^n\times\mathbb{R}^n\times G^m\to\mathbb{R}$ be the functions

$$f_{i,j}(\mathbf{v}_1,\mathbf{v}_2,\mathbf{g}) = \langle\mathbf{v}_1,\mathbf{v}_2\rangle\cdot\varphi_{i,j}(\mathbf{g})$$

Then, we have

$$\begin{aligned}|\mathrm{Exp}[\Gamma(\mathbf{a})-\Gamma(\mathbf{u})]| &= \left|\sum_{i,j,h,l}\mathrm{Exp}[f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{a}) - f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{u})]\right|\\ &\le \sum_{i,j,h,l}|\mathrm{Exp}[f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{a}) - f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{u})]|\end{aligned}$$

where, in the summations, i,j range over $\{1,\ldots,m\}$, and l,h range over $\{1,\ldots,k\}$. We bound each term

$$|\mathrm{Exp}[f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{a}) - f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{u})]| \qquad (8.56)$$

separately. We first bound the absolute value of $f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{g})$, and then the statistical distance between the two distributions $(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{a})$ and $(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{u})$. By Lemma 8.17, for every $\mathbf{g}\in G^m$, the absolute value of $f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{g})$ is at most

$$|f_{i,j}(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{g})| \le \|\mathbf{y}_{i,l}\|\cdot\|\mathbf{y}_{j,h}\|\cdot\|\varphi_{i,j}(\mathbf{g})\| \le \frac{\|\mathbf{s}_n\|^2}{\omega(n\log^2 n)}. \qquad (8.57)$$

Now consider the statistical distance

$$\Delta((\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{a}),(\mathbf{y}_{i,l},\mathbf{y}_{j,h},\mathbf{u})). \qquad (8.58)$$

By Proposition 8.10, (8.58) is at most

$$\Delta((\mathbf{y}_{i,l},\mathbf{y}_{j,h},a_{i,l},a_{j,h},\mathbf{a}),(\mathbf{y}_{i,l},\mathbf{y}_{j,h},a_{i,l},a_{j,h},\mathbf{u})). \qquad (8.59)$$

We distinguish three cases:

- If $(i, l) = (j, h)$, then by Proposition 8.10, (8.59) is at most

$$\Delta((\mathbf{y}_{i,l}, a_{i,l}, \mathbf{a} - a_{i,l}\mathbf{e}_i), (\mathbf{y}_{i,l}, a_{i,l}, \mathbf{u} - a_{i,l}\mathbf{e}_i)). \tag{8.60}$$

  Notice that $\mathbf{u}' = \mathbf{u} - a_{i,l}\mathbf{e}_i$ is uniformly distributed over G^m independently from $\mathbf{y}_{i,l}, a_{i,l}$. Also $\mathbf{a}' = \mathbf{a} - a_{i,l}\mathbf{e}_i = \sum_{(i',l')\neq(i,l)} a_{i',l'}\mathbf{e}_{i'}$ is independent from $\mathbf{y}_{i,l}, a_{i,l}$. Therefore, by Proposition 8.8, (8.60) is equal to $\Delta(\mathbf{a}', \mathbf{u}')$. The components of $\mathbf{a}'$ and $\mathbf{u}'$ are totally independent. Therefore,

$$\Delta(\mathbf{a}', \mathbf{u}') = \sum_{t=1}^{m} \Delta(a'_t, u'_t)$$

  where each u'_t is distributed uniformly at random over G, while $a'_i = \sum_{l'\neq l} a_{i,l'}$ (resp. $a'_t = \sum_{l'} a_{i,l'}$ for $t \neq i$) is the sum of $k - 1$ (resp. k) independent random variables, each satisfying the hypothesis of Corollary 8.12. So, the statistical distance (8.60) is at most

$$\frac{m-1}{2^{k+1}} + \frac{1}{2^k} = \frac{m+1}{2^{k+1}}.$$

- If $i = j$ and $l \neq h$, then (8.59) is at most

$$\Delta((\mathbf{y}_{i,l}, \mathbf{y}_{i,h}, a_{i,l}, a_{i,h}, \mathbf{a}'), (\mathbf{y}_{i,l}, \mathbf{y}_{i,h}, a_{i,l}, a_{i,h}, \mathbf{u}')) \tag{8.61}$$

  where $\mathbf{u}' = \mathbf{u} - (a_{i,l} + a_{i,h})\mathbf{e}_i$ and $\mathbf{a}' = \mathbf{a} - (a_{i,l} + a_{i,h})\mathbf{e}_i$ are both independent from $\mathbf{y}_{i,l}, \mathbf{y}_{i,h}, a_{i,l}, a_{i,h}$. So, (8.61) equals $\Delta(\mathbf{a}', \mathbf{u}')$. Again, the components of $\mathbf{a}'$ and $\mathbf{u}'$ are totally independent, with u'_t distributed uniformly at random over G and a'_t equal to the sum of independent random variables, each satisfying the hypothesis of Corollary 8.12. This time a'_i is the sum of $k - 2$ such variables. So, the statistical distance (8.61) is at most

$$\frac{m-1}{2^{k+1}} + \frac{1}{2^{k-1}} = \frac{m+3}{2^{k+1}}.$$

- The last case $i \neq j$ is also analogous. This time (8.59) is at most $\Delta(\mathbf{a}', \mathbf{u}')$ where $\mathbf{a}' = \mathbf{a} - a_{i,l}\mathbf{e}_i - a_{j,k}\mathbf{e}_j$ and $\mathbf{u}' = \mathbf{u} - a_{i,l}\mathbf{e}_i - a_{j,k}\mathbf{e}_j$. As usual, u'_t are distributed independently and uniformly at random over G, while a'_t is equal to the sum of $k-1$ (if $t \in \{i, j\}$) or k (if $t \notin \{i, j\}$) independent random variables each satisfying the hypothesis of Corollary 8.12. Therefore, the statistical distance is at most

$$\frac{m-2}{2^{k+1}} + \frac{2}{2^k} = \frac{m+2}{2^{k+1}}.$$

In all three cases, the statistical distance is at most

$$\Delta((\mathbf{y}_{i,l}, \mathbf{y}_{j,h}, \mathbf{a}), (\mathbf{y}_{i,l}, \mathbf{y}_{j,h}, \mathbf{u})) \leq \frac{m+3}{2^{k+1}} \leq O\left(\frac{\delta \log n}{n^2}\right). \tag{8.62}$$

So, by (8.57), (8.62) and Proposition 8.10, the expectation (8.56) is at most

$$\frac{\delta \|\mathbf{s}_n\|^2}{n^3 \log n}.$$

Adding up for all $i, j = 1, \ldots, m$ and $h, l = 1, \ldots, k$ we get

$$|\operatorname{Exp}[\Gamma(\mathbf{a}) - \Gamma(\mathbf{u})]| \leq \frac{\delta m^2 k^2 \|\mathbf{s}_n\|^2}{n^3 \log n} \leq \frac{\delta \log^3 n \|\mathbf{s}_n\|^2}{n} = \delta \|\mathbf{s}_n\|^2 \cdot o(1). \tag{8.63}$$

This concludes the proof that $\operatorname{Exp}[\Gamma(\mathbf{u})] \leq \delta \|\mathbf{s}_n\|^2 \cdot o(1)$. □

2.4 Almost perfect lattices

In the previous subsections we showed that any τ-perfect easily decodable lattice can be used to construct a collision resistant family of hash functions which is at least as hard to break (on the average) as approximating the covering radius of any lattice within a factor $\gamma(n) = \omega(\tau n^2 \log n)$. So, in order to make these hash functions as hard to break as possible one needs lattices with packing-covering ratio τ as close as possible to 1. We observed that $\tau = \sqrt{n}$ can be easily achieved setting Λ to the lattice of all integer points $\mathbb{Z}^n$. In this subsection we show that it is possible to do much better than that.

THEOREM 8.18 *For every n, there exist a lattice with packing-covering ratio $\tau < 4$.*

Proof: We give an iterative procedure that starting from the lattice $\mathbb{Z}^n$ of all integer points, builds denser and denser lattices such that the length of the shortest vector in the lattice does not decrease. At every iteration, either $\tau < 4$ and we stop, or we find a new point (not already in the lattice) that can be added to the current lattice without increasing the length of the shortest nonzero lattice vector. So, all lattices in the sequence have first minimum $\lambda_1 = 1$. Notice that each time a lattice gets bigger, its determinant decreases (at least) by a factor 2. The determinant of the first lattice in the sequence is $D_0 = \det(\mathbb{Z}^n) = 1$. So, after k iteration the determinant is (at most) $D_k = 2^{-k}$. By Minkowski's theorem, the determinant D_k of any of these lattices satisfies $1 = \lambda_1 \leq \sqrt{n} D_k^{1/n}$, i.e., $D_k \geq 1/n^{(n/2)}$. Using $D_k \leq 2^{-k}$, we get that the number of iteration is at most $k \leq (1/2)n \log n$, i.e., after at most $O(n \log n)$ iteration the packing-covering ratio of the lattice must satisfy $\tau < 4$.

Now we describe the iterative step. Let $\mathbf{B}$ be a lattice with packing-covering ratio at least 4, i.e., $\rho(\mathbf{B}) \geq 2\lambda_1(\mathbf{B})$. We claim that there exists a lattice point $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that the distance of $\mathbf{b} = (1/2)\mathbf{v}$ from the lattice is at least $\lambda_1(\Lambda)$. Since $2\mathbf{b}$ belongs to the lattice, the length of the shortest vector in the lattice generated by $[\mathbf{B}|\mathbf{b}]$ is

$$\lambda_1([\mathbf{B}|\mathbf{b}]) = \min\{\lambda_1(\mathbf{B}), \operatorname{dist}(\mathbf{b}, \mathcal{L}(\mathbf{B}))\} = \lambda_1(\mathbf{B}).$$

So, let us prove the existence of such a $\mathbf{v}$. Let $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ be a lattice point such that $\mathbf{b} = (1/2)\mathbf{v}$ is as far from $\mathcal{L}(\mathbf{B})$ as possible:

$$\operatorname{dist}(\mathbf{b}, \mathcal{L}(\mathbf{B})) = \max_{\mathbf{v} \in \mathcal{L}(\mathbf{B})} \operatorname{dist}(\mathbf{v}/2, \mathcal{L}(\mathbf{B}))$$

and let d be the distance of $\mathbf{b}$ from $\mathcal{L}(\mathbf{B})$. It follows that all points in $(1/2)\mathcal{L}(\mathbf{B})$ are within distance d from $\mathcal{L}(\mathbf{B})$, i.e.,

$$\frac{1}{2}\mathcal{L}(\mathbf{B}) \subset \bigcup_{\mathbf{v} \in \mathcal{L}(\mathbf{B})} \mathcal{B}(\mathbf{v}, d) = \mathcal{L}(\mathbf{B}) + \mathcal{B}(\mathbf{0}, d).$$

By induction, it follows that

$$\begin{aligned}
\left(\frac{1}{2}\right)^{k+1} \mathcal{L}(\mathbf{B}) &\subset \frac{1}{2}\left(\mathcal{L}(\mathbf{B}) + \mathcal{B}(\mathbf{0}, 2(1-2^{-k})d)\right) \\
&\subset \frac{1}{2}\mathcal{L}(\mathbf{B}) + \mathcal{B}(\mathbf{0}, (1-2^{-k})d) \\
&\subset \mathcal{L}(\mathbf{B}) + \mathcal{B}(\mathbf{0}, 2(1-2^{-(k+1)})d) \\
&= \bigcup_{\mathbf{v} \in \mathcal{L}(\mathbf{B})} \mathcal{B}(\mathbf{v}, 2(1-2^{-(k+1)})d).
\end{aligned}$$

This proves that for every k, $2^{-k}\mathcal{L}(\mathbf{B})$ is contained in $\bigcup_{\mathbf{v} \in \mathcal{L}(\mathbf{B})} \mathcal{B}(\mathbf{v}, 2d)$. Since $\cup_k 2^{-k}\mathcal{L}(\mathbf{B})$ is dense in $\operatorname{span}(\mathbf{B})$, this shows that $\operatorname{span}(\mathbf{B})$ is contained in $\bigcup_{\mathbf{v} \in \mathcal{L}(\mathbf{B})} \mathcal{B}(\mathbf{v}, 2d)$, i.e., the covering radius of $\mathcal{L}(\mathbf{B})$ is at most $\rho(\mathbf{B}) \leq 2d$, and $\operatorname{dist}(\mathbf{b}, \mathcal{L}(\mathbf{B})) = d \geq \rho(\mathbf{B})/2 \geq \lambda_1(\mathbf{B})$. □

The proof of the theorem also gives an algorithmic procedure to find such almost perfect lattices Λ_n with constant τ. However, the simplest way to build these lattices and solve the corresponding closest vector problem requires time exponential in the dimension n. Still, these lattices can be used to improve the approximation factor in the construction of cryptographic hash functions as follows: divide all the coordinates into $n/\log n$ equal blocks, and set Λ_n to the direct sum of $(n/\log n)$ 4-perfect

$(\log n)$-dimensional lattices. These lattices can be constructed in time exponential in $\log n$, and therefore polynomial in n. Moreover, the closest vector problem in Λ_n can also be solved in polynomial time, decoding each block of coordinates independently. This shows that lattices with packing-covering radius asymptotically less than $\sqrt{n}$ can be efficiently constructed and decoded in polynomial time, improving the connection factor in the construction of lattice based hash functions by a factor $\sqrt{\log n}$. Finding efficient decoding algorithms for almost perfect lattices would allow to reduce this factor even further, resulting in hash functions that are as hard to break as approximating the covering radius of any lattice within a factor $\gamma = \omega(n^2 \log n)$. (See Section 4 for further discussion.)

3. Encryption Functions

In this section we present a brief overview of various public key encryption schemes based on lattices. Public key encryption is one of the most important cryptographic primitives. The difference between symmetric encryption, as described in the introduction of this chapter, and public key encryption is that in the latter the key used to encrypt and the one used to decrypt are different. Moreover, the encryption key can be published without compromising the security of the scheme. This allows to securely communicate over public networks without agreeing beforehand on a common shared key for every pair of communicating parties: each user can generate a pair of matching encryption and decryption keys. The decryption key is kept secret and used by the recipient to decode messages. The encryption key is published on a directory, so that anybody can access it. When one wants to send a message to some other party, one retrieves the recipient public key from the directory, and use it to encrypt the message. The recipient uses the matching secret key to decrypt the ciphertext. Clearly, recovering the secret key from the corresponding public key must be computationally hard, as well as recovering (any partial information about) the cleartext given the ciphertext and the public key without knowing the matching secret key.

Several public key encryption schemes whose security is based on lattices were proposed in the last few years. Some of them are mostly theoretical, with rigorous proofs of security based on worst case computational assumptions similar to the one used in the construction of collision resistant hash functions. Others are concrete proposals, without a proof of security, that have been suggested as practical alternatives to commonly used cryptosystems. The main lattice based public key encryption schemes were all suggested independently and essentially at the same time. So, there is not a clear chronological order to follow in the presentation. Here, we have chosen to present the various schemes in an order that helps getting a better understanding of the similarities and differences among them. A concrete analysis of the practical security offered by these schemes is beyond the scope of this book. Some of the schemes presented here have been subject to more or less serious cryptanalytic attacks, but no asymptotic attack that runs in polynomial time is known for any of them. So, in the following subsections we concentrate on the ideas underlying the *design* of the schemes, instead of attempting a careful analysis of the practical value of any of them, occasionally giving some pointers to relevant cryptanalytic literature.

The standard notion of security for encryption schemes is that not only it is hard to recover the plaintext from the ciphertext, but also gaining partial information about the plaintext is computationally infeasible. Such encryption schemes can be constructed using standard techniques starting from schemes meeting a weaker notion of security (Yao, 1982; Goldwasser and Micali, 1984; Goldreich and Levin, 1989; Bellare and Rogaway, 1993). These weaker schemes are called *trapdoor functions*: functions that are easy to compute, but hard to invert unless some trapdoor information is known. For simplicity, in the following subsections, we concentrate on the trapdoor functions underlying the schemes, whenever possible.

3.1 The GGH scheme

The GGH cryptosystem (Goldreich et al., 1997b) was proposed by Goldreich, Goldwasser and Halevi, and it is probably the most intuitive method of using lattices to devise a public key encryption scheme. The idea underlying the construction is that, given *any* basis for a lattice, it is easy to generate a vector which is close to a lattice point (i.e., by taking a lattice point and adding a small perturbation vector to it). However, it seems hard to return from this "close-to-lattice" vector to the original lattice point (given an arbitrary lattice basis.) Thus, the operation of adding a small perturbation vector to a lattice point can be thought of as a one-way computation.

To introduce a trapdoor mechanism into this one-way computation and allow efficient decryption when the trapdoor is known, (Goldreich et al., 1997b) uses the fact that different bases of the same lattice seem to yield a difference in the ability to find close lattice points to arbitrary vectors in $\mathbb{Q}^n$. Therefore, the trapdoor information may be a basis of a lattice which allows very good approximation of the closest vector problem (CVP). Then the public basis is derived from it using a randomized unimodular transformation. Details follow.

Private key. Two methods for generating the private key are suggested in (Goldreich et al., 1997b). The first method is to set $\mathbf{R}$ to an $n \times n$ matrix with entries chosen independently and uniformly at random from an interval $\{-l, \ldots, +l\}$. The second method is to set $\mathbf{R} = k\mathbf{I} + \mathbf{R}'$ to an orthogonal matrix $k\mathbf{I}$ (where k is a parameter, e.g., $k = \sqrt{n}l$), and then add a perturbation matrix $\mathbf{R}'$ with entries chosen independently and uniformly at random in $\{-l, \ldots, +l\}$. The second method has the advantage of giving private bases $\mathbf{R}$ that can recover from longer perturbation vectors, but it might also help an adversary to recover the private basis from the public basis.

Public key. Once the private basis $\mathbf{R}$ is chosen, the public basis should be selected according to some probability distribution over all possible bases for $\mathcal{L}(\mathbf{B})$. Two methods for generating the public basis $\mathbf{B}$ from $\mathbf{R}$ are considered. In the first method $\mathbf{R}$ is transformed into $\mathbf{B}$ applying a sequence of elementary column operations, i.e., at every step one add to a column a random integer combination of the other columns. The coefficients in the integer combination are chosen at random in $\{-1, 0, +1\}$. In the second method, $\mathbf{B}$ is obtained multiplying $\mathbf{R}$ by a small number of random unimodular matrices. The unimodular matrices are generated multiplying together a lower triangular matrix $\mathbf{L}$ and an upper triangular matrix $\mathbf{U}$, with ± 1 on the diagonal, and the other elements chosen at random in $\{-1, 0, +1\}$. We do not go into more details here as we will see in Subsection 3.2 that there are provably better ways of generating $\mathbf{B}$.

Encryption. As outlined before, the trapdoor function takes as input an integer vector $\mathbf{x}$, and a small perturbation vector $\mathbf{r}$, and outputs $\mathbf{t} = \mathbf{B}\mathbf{x} + \mathbf{r}$. Vector $\mathbf{r}$ should be short enough to allow the recovery of $\mathbf{B}\mathbf{x}$ from $\mathbf{t}$ using the private basis $\mathbf{R}$. The maximum length allowed for the perturbation vector δ is included in the public basis. (See next paragraph for details about the value of δ.) Vector $\mathbf{x}$ is chosen in a sufficiently large region of space so that $\mathbf{B}\mathbf{x}$ looks like a "random" lattice point. As for the public basis, we will see in Subsection 3.2 that there are provably better ways of choosing $\mathbf{x}$, so we do not elaborate about this choice any further.

Decryption. Also for the decryption process, two different methods are considered. Both methods are based on the CVP approximation algorithms of (Babai, 1986). One method is to use the nearest plane algorithm (see Chapter 2) with basis $\mathbf{R}$. Using this algorithm one can efficiently find the (unique) lattice point $\mathbf{B}\mathbf{x}$ within distance $\delta =$

$(1/2)\min_i \|\mathbf{r}_i^*\|$ from $\mathbf{t}$, and recover the input $(\mathbf{x}, \mathbf{t} - \mathbf{Bx})$. The other method is simply to compute $\mathbf{R}^{-1}\mathbf{t}$, round each coordinate of this vector to the closest integer, and multiply the result by $\mathbf{R}$. Bounds on the maximum and average length of the errors that can be corrected using this method are given in (Goldreich et al., 1997b).

Analysis. Notice that in order to avoid attacks based on exhaustive search, the sequence of operations applied to $\mathbf{R}$ to obtain the public basis, and the region of space from which the lattice vector $\mathbf{Bx}$ is chosen must be sufficiently large. Since the lattice repeats identically if translated by $\det(\mathbf{R})$ along any of the main axes, we can always assume that the entries of $\mathbf{B}$ and $\mathbf{x}$ are reduced modulo $\det(\mathbf{R})$ without decreasing the security of the scheme. We can use this observation to estimate the proper size of the public key $\mathbf{B}$ and the ciphertext $\mathbf{c} = \mathbf{Bx} + \mathbf{r}$ as $O(n^2 \cdot \lg(\det(\mathbf{R})))$ and $O(n \cdot \lg(\det(\mathbf{R})))$. Applying Hadamard's bound to the private basis, and assuming $l = \mathrm{poly}(n)$, one can estimate the determinant $\det(\mathbf{R}) \approx 2^{O(n \lg n)}$. This results in public keys and ciphertexts of size $O(n^3 \lg n)$ and $O(n^2 \lg n)$. Although polynomial in the security parameters, these sizes grow pretty fast, and in (Nguyen, 1999) it is shown that in order to provide a reasonable level of security the size of the keys in the GGH cryptosystem has to be so large that the system would be impractical. However, it is important to realize that the attacks described in (Nguyen, 1999) are not asymptotic: they only prove that the system can be efficiently broken for specific values of the security parameter and increasing the security parameter avoids the attacks. The problem pointed out by (Nguyen, 1999) is mainly an efficiency issue: in order to provide a concrete alternative to commonly used cryptosystems, it is necessary to make the GGH cryptosystem more efficient, so that larger values of the security parameters can be used.

3.2 The HNF technique

This scheme, proposed in (Micciancio, 2001c), is more a general technique than a full fledged encryption scheme, and it can be used to improve the efficiency and better understand the security of most lattice based encryption functions, including the GGH cryptosystem, one of the cryptosystems proposed by Ajtai and Dwork, and NTRU. The private key $\mathbf{R}$ and the corresponding decryption algorithm are not specified, and the technique can be applied to all schemes where the public key is a lattice basis $\mathbf{B}$, the ciphertext is a vector $\mathbf{t}$ close to the lattice, and the decryption process involves finding a lattice vector in $\mathcal{L}(\mathbf{B})$ close to $\mathbf{t}$ (or, in some cases, determining if $\mathbf{t}$ is close to the lattice or not). The questions addressed in (Micciancio, 2001c) are the following:

- What is the best way to select the public basis $\mathbf{B}$?
- Given a lattice basis $\mathbf{B}$ and a short perturbation vector $\mathbf{r}$, what is the best way to generate a hard CVP instance $(\mathbf{B}, \mathbf{t})$ whose solution is $\mathbf{t} - \mathbf{r}$?

In both cases, we are interested in schemes that are best from a security point of view: solving CVP instance $(\mathbf{B}, \mathbf{t})$ should be as hard as possible, unless some trapdoor information $\mathbf{R}$ is known. Interestingly, (Micciancio, 2001c) shows that there is an optimal way to choose $\mathbf{B}$ and $\mathbf{t}$ that also leads to considerable efficiency improvements with respect to GGH and similar schemes.

The technique is the following. Assume that a private basis $\mathbf{R}$ has been chosen that allows to solve the CVP problem whenever the target $\mathbf{t}$ is within distance δ from $\mathcal{L}(\mathbf{R})$. Let $\mathbf{r}$ be a perturbation vector of length at most δ. We want to define a basis $\mathbf{B}$ for $\mathcal{L}(\mathbf{R})$ and a target vector $\mathbf{t} = \mathbf{B}\mathbf{x} + \mathbf{r}$ such that CVP instance $(\mathbf{B}, \mathbf{t})$ is as hard as possible, in some technical sense. Micciancio proposes to use the Hermite normal form (HNF) of $\mathbf{R}$ as the public basis. This public basis is the worst possible one from a computational point of view, because it can be efficiently computed from any other basis. Technically, any attack that works given a public basis in HNF can be transformed into an equally efficient attacks that receives as input an arbitrary basis (e.g., a random basis) for the lattice: given a "random basis" $\mathbf{B}$, one first computes the HNF of $\mathbf{B}$, and then applies the HNF based attack. Similarly, the encryption function takes as input an error vector $\mathbf{r}$, and outputs vector $\mathbf{r}$ reduced modulo the public basis, i.e., $\mathbf{t} = \mathbf{r} \bmod \mathbf{B}$. Again, this is at least as secure as adding $\mathbf{r}$ to a "random" lattice vector $\mathbf{B}\mathbf{x}$, because given $\mathbf{B}\mathbf{x} + \mathbf{r}$ one can efficiently compute $\mathbf{t} = (\mathbf{B}\mathbf{x} + \mathbf{r}) \bmod \mathbf{B} = \mathbf{r} \bmod \mathbf{B}$.

One interesting way to look at HNF cryptosystems is to consider the encryption function as a function from a set of short vectors to the group $G = \mathbb{Z}^n/\mathcal{L}(\mathbf{R})$ studied in Subsection 1.1. If perturbation vectors are short enough, then any two perturbations correspond to different group elements, and the function can be inverted using the private basis $\mathbf{R}$. The public key and the encryption procedure correspond to choosing some standard way to represent group G, and its elements. As described in Subsection 1.1, using the HNF technique has the advantage that both the group and the group elements have space efficient representations: the HNF public basis has size $O(n^2 \log n)$ instead of $O(n^3 \log n)$ as in the original GGH cryptosystem. Similarly, the size of the ciphertext is $O(n \log n)$ instead of $O(n^2 \log n)$. For typical values of the security parameters (e.g., $n = 400$) this is a huge improvement with respect to using random public bases and random lattice vectors in the encryption function. The resulting cryptosystem can outperform commonly used encryption functions based on number theory, as far as the encryption time is concerned. Still, in order to be really competitive in terms of public key size, further efficiency improvements are needed. In (Micciancio, 2001c), it is pointed out that the Hermite normal form representation is essentially optimal: the bit-size of a matrix in HNF is roughly equal to the logarithm of the number of lattices with the same determinant. So, in order to reduce the size of the public key below $O(n^2 \log n)$, one has to restrict the choice of $\mathcal{L}(\mathbf{R})$ to special classes of lattices. One such class that results in public keys of size $O(n \log n)$ is the one used by NTRU.

3.3 The Ajtai-Dwork cryptosystem

In (Ajtai and Dwork, 1997) two related cryptosystems based on lattices are proposed. The first one fits our general framework, and can be improved using the HNF technique described in Subsection 3.2. The second cryptosystem is interesting because it exhibits a worst-case/average-case connection similar to the one studied for hash function in Section 2. This is essentially the only known cryptosystem which is as hard to break as the worst case instance of the underlying mathematical problem. Unfortunately, the system (as described in (Ajtai and Dwork, 1997)) does not seem efficient enough to represent a practical alternative to commonly used encryption schemes. (Both the key size and running time grow as $O(n^4)$, where n is the rank of the lattice.) The fact that the Ajtai-Dwork cryptosystem is not efficient enough to be practical and secure at the same time was experimentally confirmed by (Nguyen and Stern, 1998). Still this cryptosystem remains one of the major breakthroughs in theoretical cryptography and is an important step in the design of provably secure cryptosystems. Below, we describe the ideas underlying the two cryptosystems.

Private key. The main difference between the Ajtai-Dwork and GGH cryptosystem is in the choice of the secret key. Instead of choosing a lattice basis that allows to solve CVP for all target points sufficiently close to the lattice, (Ajtai and Dwork, 1997) suggests to pick the private basis as follows. Let M and d be two parameters, with $d \geq n^c M$ for some sufficiently large polynomial function of the lattice rank. Then, pick $n-1$ (linearly independent) random vectors $\mathbf{r}_1, \ldots, \mathbf{r}_{n-1}$, and let $\mathcal{H}$ be the hyperplane spanned by them. The last basis vector $\mathbf{r}_n$ is chosen as a random vector whose distance from $\mathcal{H}$ is approximately equal to d. (Say between d and $2d$.) Notice that all lattice points belong to a collection of hyperplanes $\mathcal{H}_k = k\mathbf{r}_n^* + \mathcal{H}$. Only the orthogonalized vector $\mathbf{r}_n^*$ needs to be stored as part of the secret key, as the other basis vectors are not used by the decryption algorithm.

Public key. In (Ajtai and Dwork, 1997) it is suggested to set the public key to a random basis $\mathbf{B}$ of $\mathcal{L}(\mathbf{R})$. As discussed in Subsection 3.2, a better choice (both from the efficiency and security point of view) is to set $\mathbf{B}$ to the HNF of $\mathbf{R}$.

Encryption. The Ajtai-Dwork cryptosystem is not based on a trapdoor function, meaning that even using the decryption key it is not clear how to fully recover the input to the encryption function. The encryption algorithm takes as input a single message bit b and a random string r. Only the bit b is recovered by the decryption algorithm. The idea is to encode 0 as points that are close to the lattice and 1 as points that are far from the lattice. So, if $b = 0$ one selects a random lattice point and adds a small random perturbation to it. (Alternatively, using the HNF improvement, one only chooses the small perturbation at random and reduces it modulo the public basis.) The perturbation vector is chosen as the sum of $O(n)$ vectors independently and uniformly distributed in the sphere of radius n^3M. If $b = 1$, one simply selects a random point in space (possibly reduced modulo the public basis), which will be far away from the lattice with high probability.

Decryption. Parameters are chosen in such a way that the perturbation vector is always much shorter than the distance d between hyperplanes $\mathcal{H}_k$. Therefore, given a target vector $\mathbf{t}$, one can simply compute the distance from the closest hyperplane (e.g., evaluating the $\langle \mathbf{r}_n^*, \mathbf{t}\rangle$ and comparing it to the closest multiple of $\|\mathbf{r}_n^*\|$) and use this distance to decide whether $\mathbf{t}$ is close to the lattice or not. (Remember, all lattice points belong to the hyperplanes.) It should be noted that decryption errors can occur with small, but nonnegligible, probability: when sending message $b = 1$, the random point $\mathbf{t}$ selected by the encryption algorithm might be close to one of the hyperplanes just by chance. If this happens, the ciphertext would be decrypted as 0. A technique to eliminate these decryption errors is described in (Goldreich et al., 1997a).

Analysis. The Ajtai-Dwork cryptosystem, as described above, has the property that breaking the scheme is equivalent to recovering the secret key. The idea is the following. Assume that we can tell the difference between encryptions of 0 (i.e., points close to the hyperplanes $\mathcal{H}_k$) and encryptions of 1 (i.e., points far from the same hyperplanes). We can use this ability to find $n-1$ linearly independent long vectors very close to $\mathcal{H} = \mathcal{H}_0$ as follows: we start from the origin and we keep moving at random, using the decryption oracle to check that we are staying close to the plane. After $n-1$ long linearly independent vectors close to $\mathcal{H}$ are found, one can compute (a multiple of) $\mathbf{r}_n^*$ using standard lattice approximation algorithms. (The actual procedure involves the dual lattice and it is not described here.) Once we have found a multiple of $\mathbf{r}_n^*$, we can also find the exact length of $\mathbf{r}_n^*$ by projecting the lattice orthogonally to the line $\mathbb{R} \cdot \mathbf{r}_n^*$, and compute the length of the shortest nonzero vector in this one-dimensional lattice.

Ajtai and Dwork also propose a variant of this cryptosystem which is provably as hard to break as the worst case instance of a certain lattice problem. The problem is the *hidden hyperplane problem*: given a random basis $\mathbf{B}$ for $\mathcal{L}(\mathbf{R})$, find the hyperplane $\mathcal{H} = \text{span}(\mathbf{r}_1, \ldots, \mathbf{r}_{n-1})$, i.e., find a long orthogonalized vector $\mathbf{r}_n^*$. (This problem can be equivalently formulated as finding a short vector in the dual lattice, for lattices in which this short vector is unique in some technical sense. This is the *unique shortest vector problem* also studied in (Ajtai, 1996) in connection with the construction of one-way functions with worst-case/average-case equivalence.) The idea is that instead of publishing a basis for lattice $\mathcal{L}(\mathbf{R})$, one can simply publish a collection of polynomially many (e.g., n^3) points close to the lattice. Then, in order to send $b = 0$, one selects a random subset of these points and add a small perturbation vector, while to send $b = 1$ one sends a random point in space as usual. Here we are omitting several important technical details, but the basic idea is the following: since the sublattice generated by $\mathbf{r}_1, \ldots, \mathbf{r}_{n-1}$ is very dense in $\mathcal{H}$, the perturbed lattice points can be selected as random points close to the collection of hyperplanes $\mathcal{H}_k$, independently from the particular basis $\mathbf{r}_1, \ldots, \mathbf{r}_{n-1}$. So, a decryption algorithm for this cryptosystem can be used to find the hidden hyperplane in any underlying lattice $\mathbf{R}$. The reader is referred to (Ajtai and Dwork, 1997) for further details.

3.4 NTRU

NTRU is a public key encryption scheme based on arithmetic in polynomial rings, but it is closely related to lattice problems for a certain class of lattices. Let p and q be two small, relatively prime integers, e.g., $p = 3$ and $q = 128$. (In general, we want p very small, and q polynomial in a security parameter n.) Let $R = \mathbb{Z}[X]/(X^n - 1)$ be the ring of all polynomial with integer coefficients modulo $X^n - 1$. Polynomials in R can be naturally represented as integer vectors in $\mathbb{Z}^n$. The private key of the system is a pair of polynomials $\mathbf{f}, \mathbf{g} \in R$ with small coefficients (e.g., $\{0, 1, -1\}$ coefficients) such that f is invertible modulo p and modulo q.

The public key is given by the polynomial $\mathbf{h} = p\mathbf{f}^{-1} \cdot \mathbf{g} \bmod q$, where the inverse and the product are computed in $\mathbb{Z}_q^n[X]/(X^n - 1)$. (In (Hoffstein et al., 1998), $\mathbf{h}$ is defined as $\mathbf{f}^{-1} \cdot \mathbf{g} \bmod q$, but the two definitions are clearly equivalent. Here, we modified the definition slightly, in order to make the comparison with other lattice based cryptosystems easier.) The encryption function takes as input two polynomials $\mathbf{m}$ and $\mathbf{r}$ with small coefficients (e.g., coefficients in $\{-1, 0, +1\}$) and outputs the polynomial $\mathbf{t} = \mathbf{m} + \mathbf{hr} \bmod q$. The decryption algorithm, takes $\mathbf{t}$ as input, and computes $\mathbf{a} = \mathbf{ft} \bmod q$ (where the coefficients of $\mathbf{a}$ are chosen in the interval $\{-q/2, \ldots, +q/2\}$) and $\mathbf{m}' = \mathbf{f}^{-1}\mathbf{a} \bmod p$. In (Hoffstein et al., 1998) it is shown that for an appropriate choice of the parameters, the decryption procedure recovers the original message $\mathbf{m}' = \mathbf{m}$ with high probability. The idea is roughly the following. From the definition of $\mathbf{t}$ and $\mathbf{h}$, we get

$$\mathbf{a} = \mathbf{ft} \bmod q = \mathbf{f}(\mathbf{m} + \mathbf{hr}) \bmod q = \mathbf{fm} + p\mathbf{gr} \bmod q.$$

Since $\mathbf{f}, \mathbf{m}, \mathbf{g}, \mathbf{r}$ are all polynomials with small coefficients and p is also small, the coefficients of $\mathbf{fm}+p\mathbf{gr}$ belongs to the interval $\{-q/2, \ldots, q/2\}$ with high probability. So, vector $\mathbf{a}$ equals $\mathbf{fm} + p\mathbf{gr}$ over the integers, and $\mathbf{m}' = \mathbf{f}^{-1}(\mathbf{fm} + p\mathbf{gr}) = \mathbf{m} \pmod p$. It is clear that once $\mathbf{m}$ is recovered, one can also recover $\mathbf{r} = (\mathbf{t} - \mathbf{m})(p\mathbf{h})^{-1} \bmod q$.

This cryptosystem can be described in terms of lattices as follows. We consider the class of q-modular, bi-cyclic lattices in dimension $2n$. Here, q-modular means that all lattice vectors $q\mathbf{e}_i$ belong to the lattice, so the coordinates of the lattice vectors can be defined modulo q. We say that a lattice is bi-cyclic if the following holds. For any vector $\mathbf{x} = [x_1, \ldots, x_n]^T$, define the rotation function $\mathrm{rot}(\mathbf{x}) = [x_n, x_1, \ldots, x_{n-1}]^T$. Define also the circulant matrix of a vector $\mathbf{x}$ as the matrix $\mathbf{M_x} = [\mathbf{x}, \mathrm{rot}(\mathbf{x}), \ldots, \mathrm{rot}^{n-1}(\mathbf{x})]$ with all possible rotations of $\mathbf{x}$ as columns. The relation between circulant matrices and polynomials is that for any two polynomials $\mathbf{x}, \mathbf{y}$ in R, $\mathbf{M_x M_y} = \mathbf{M_{xy}}$, i.e., the product in the quotient ring R correspond to the standard matrix product. For any $2n$-dimensional vector $\mathbf{z} = [\mathbf{x}^T, \mathbf{y}^T]^T$ (with $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^n$), define also the double rotation $\mathrm{rot}_2(\mathbf{z}) = [\mathrm{rot}(\mathbf{x})^T, \mathrm{rot}(\mathbf{y})^T]^T$. A $2n$-dimensional lattice is bi-cyclic if it is close under double rotations, i.e., if for any vector $\mathbf{z}$ in the lattice, also $\mathrm{rot}_2(\mathbf{z})$ belongs to the lattice. It is easy to see that the intersection of q-modular bi-cyclic lattices is also q-modular and bi-cyclic. So, for any set of vectors S, we can define the smallest q-modular bi-cyclic lattice containing S. We now give a lattice based description of NTRU.

Private key. The private key is given by a short vector $\mathbf{v}$. The lattice associated to this vector is the smallest bi-cyclic q-modular lattice containing $\mathbf{v}$. A generating set for this lattice is easily obtained taking all double rotations of $\mathrm{rot}_2^k(\mathbf{v})$ (for $k = 0, \ldots, n-1$), and all vectors of the form $q\mathbf{e}_k$ (for $k = 0, \ldots, 2n-1$).

Public key. The public key is set to the HNF basis of the q-modular bi-cyclic lattice generated by $\mathbf{v}$. Interestingly, if $\mathbf{v} = [p\mathbf{g}^T, \mathbf{f}^T]^T$, then the public basis is given by

$$\mathbf{H} = \begin{bmatrix} q\mathbf{I} & \mathbf{M_h} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}.$$

In other words, the lattice can be described as the smallest q-modular bi-cyclic lattice containing $[\mathbf{h}^T, \mathbf{e}_1^T]^T$.

Encryption. Interestingly, the encryption can also be described as a special instance of the general HNF framework. Consider the short perturbation vector $[\mathbf{m}^T, -\mathbf{r}^T]^T$. If we reduce this vector modulo the HNF basis $\mathbf{H}$, we obtain the ciphertext vector $[\mathbf{t}^T, \mathbf{0}^T]^T$, where $\mathbf{t}$ is the polynomial defined in the NTRU polynomial ring description.

Decryption. The decryption algorithm seems to depend on the specific polynomial ring trapdoor, and it is not clear how to interpret it from a geometric point of view.

Analysis. The special structure of q-modular bi-cyclic lattices allows to represent the secret key, the public key and the ciphertext with only $O(n \log n)$ bits. The culprit is that only one vector needs to be stored to implicitly represent the entire secret or public basis. This allows to reduce the storage required by general HNF cryptosystems by a factor n, although using a special class of lattices. From the efficiency point of view, NTRU offers clearly lot of advantages: extremely fast encryption, decryption and key generation, with public key size comparable to widely used number theory based cryptosystems. The main questions regarding NTRU are about security: are lattice problems for the special class of lattices used by NTRU as hard as the general case? Are these problems NP-hard to solve exactly? NP-hard to approximate? Is it possible to prove an worst-case/average-case connection for these lattices similar to the one proved by Ajtai for general lattices? There is still very little known about NTRU from a theoretical point of view, but the practicality of the system definitely makes further investigations worthwhile.

4. Notes

Most of the techniques described in Section 1 are folklore. Algorithms to compute the HNF and SNF of integer matrices can be found in (Cohen, 1996). A new, space efficient algorithm to compute the HNF has recently been proposed in (Micciancio and Warinschi, 2001).

One-way functions which are as hard to break as the worst case instance of some lattice approximation problem were first discovered by (Ajtai, 1996). The approximation factor in (Ajtai, 1996) is a rather large polynomial: n^c for $c > 8$. The factor was subsequently improved by (Cai and Nerurkar, 1997), who showed that inverting the one-way function is at least as hard as solving GAPSVP within $n^{4+\epsilon}$, or GAPSIVP within $n^{3+\epsilon}$. In Section 2 we presented an improved construction recently discovered by (Micciancio, 2001b). Following (Goldreich et al., 1996), Micciancio shows that the the function is not simply one-way: it is collision resistant. Moreover, breaking the hash functions is at least as hard as approximating GAPSVP within $n^{3.5} \log n$ or GAPCRP within $n^{2.5} \log n$, improving (Ajtai, 1996) and (Cai and Nerurkar, 1997). Factors can be further reduced by $\sqrt{n}$ if CVP can be efficiently solved for certain almost perfect lattices. (See (Micciancio, 2001b) for the description of weaker requirement on the decoding algorithm that still allow to build improved hash functions.)

The GGH, Ajtai-Dwork and NTRU cryptosystems were all discovered independently at about the same time around 1996. The cryptosystems are fully described in (Goldreich et al., 1997b; Ajtai and Dwork, 1997; Hoffstein et al., 1998). The HNF technique was suggested later by (Micciancio, 2001d) as a method to improve or better understand lattice based cryptosystems. Various other cryptosystems based on lattices have been proposed, usually variants of those described in Section 3. For example, (Fischlin and Seifert, 1999) suggests a variant of the GGH cryptosystem where the trapdoor is based on the tensor product of lattices. The HNF technique of (Micciancio, 2001d) applies to these cryptosystems as well. The construction of cryptosystems based on lattices is still subject to investigations. (Micciancio, 2001d) points out that basing cryptosystems on restricted class of lattices seems crucial to obtain encryption functions with public keys of subquadratic size. The NTRU cryptosystem of (Hoffstein et al., 1998) seems an interesting proposal from this point of view. Still, very little is known about the computational complexity of specific classes of lattices, as those used by NTRU.

Chapter 9

INTERACTIVE PROOF SYSTEMS

A natural question associated with the SVP and the CVP search problems is whether one can recognize the optimality of solutions once they are found.

SVP. In the case of SVP, this may correspond in its most ambitious form, to given a lattice and a length d (presumably the length of the shortest vector in a lattice) to be able to efficiently verify that (1) there exists a short vector of length d and (2) no other vector in the lattice is shorter than d.

A more modest goal, is to ask whether there even exists a "short and easy to verify" proof of properties (1) and (2). Clearly, a vector $\mathbf{v}$ in the lattice of length d, is in itself a short and easy to verify proof for (1). Whether there exist short and easy to verify proofs that $\mathbf{v}$ is shortest (namely property (2)) is a more challenging question. In this chapter we formulate and address it for approximation versions of SVP and CVP.

Recall the promise problem GAPSVP$_\gamma$. YES instances of GAPSVP$_\gamma$ are pairs $(\mathbf{B}, d)$ where $\mathbf{B}$ is a basis for a lattice in $\mathbb{Q}^n$, and $d \in \mathbb{Q}$ such that there exist vectors in the lattice of length d. NO instances are pairs $(\mathbf{B}, d)$ where $\mathbf{B}$ and d are as above, but the shortest vector in the lattice is of size greater than $\gamma(n) \cdot d$. Pairs $(\mathbf{B}, d)$ where $\mathbf{B}$ is a basis for a lattice whose shortest vector is between d and $\gamma(n)d$ are not in the promise, and thus are invalid instances. GAPSVP$_1$ is the straight forward decision problem for SVP.

For any $\gamma(n) \geq 1$, GAPSVP$_{\gamma(n)}$ is clearly in NP (or, more precisely, the the extension of NP to promise problems). The NP-witness for $(\mathbf{B}, d)$ being a YES instance is merely a vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ of length less than d. But, what about the complement of GAPSVP$_\gamma$? For which γ is there a

D. Micciancio et al., *Complexity of Lattice Problems*

short and easily verifiable proof that all vectors in $(\mathbf{B}, d)$ are "long", and what would such proof look like? Intuitively, the bigger the gap γ is, the easier this problem becomes (i.e., it should be easier to distinguish between lattices with "really long" shortest vectors and lattices with "short" shortest vectors).

Partial answers exist. In a sequence of results, (Lagarias et al., 1990), (Håstad, 1988) and (Banaszczyk, 1993) showed that the complement of GAPSVP$_n$ is in NP. Namely, there exists an NP witness for those $(\mathbf{B}, d)$ for which the shortest vector is long enough, at least of length nd. (Goldreich and Goldwasser, 2000) improved on this factor, and showed that the complement of GAPSVP$_{\gamma(n)}$ is in AM for $\gamma(n) = \sqrt{n}/O(\log n) = o(\sqrt{n})$. (AM is the class of languages recognized by a constant round interactive proof system.) For brevity, in the rest of this chapter we will write $o(\sqrt{n})$ to denote approximation factor $\sqrt{n}/O(\log n)$. They do this, by exhibiting a constant-round interactive proof system for the complement of GAPSVP$_{o(\sqrt{n})}$. Namely, instances $(\mathbf{B}, d)$ for which the shortest vector is "long" (greater than $d \cdot o(\sqrt{n})$) are always accepted, and the instances for which the shortest vector is of length d (or less) are rejected with all but negligible probability. This result places a potentially harder problem (referring to smaller gaps) in a larger class (as coNP $\subset$ coAM). Unlike the proofs of (Lagarias et al., 1990; Håstad, 1988; Banaszczyk, 1993) which rely on deep duality results regarding lattices, the interactive proof is elementary and we shall present it fully in this chapter.

CVP. In the case of CVP, the analogous question is given a lattice, a length d , and a target vector $\mathbf{v}$, whether there exists a short and easy to verify proof that (1) there exists a vector $\mathbf{u}$ in the lattice at distance d from $\mathbf{v}$ and (2) no other vector in the lattice is closer to $\mathbf{v}$.

In complexity theoretic terms, recall the definition of GAPCVP$_\gamma$. Let dist$(\mathbf{v}, \mathbf{u})$ denote the Euclidean distance between the vectors $\mathbf{v}, \mathbf{u} \in \mathbb{R}^n$, and dist$(\mathbf{v}, \mathcal{L}(\mathbf{B}))$ denote the distance of $\mathbf{v}$ from the lattice, $\mathcal{L}(\mathbf{B})$ generated by the basis $\mathbf{B}$. Then, YES instances of GAPCVP$_{\gamma(n)}$ are triples $(\mathbf{B}, \mathbf{v}, d)$ where $\mathbf{B}$ is a basis for a lattice in $\mathbb{Q}^n$, $\mathbf{v} \in \mathbb{Q}^n$ is a vector, and $d \in \mathbb{Q}$ is a length where dist$(\mathbf{v}, \mathcal{L}(\mathbf{B})) < d$. NO instances (i.e., instances that strongly violate the closeness property) are triples $(\mathbf{B}, \mathbf{v}, d)$ where $\mathbf{B}$ is a basis for a lattice in $\mathbb{Q}^n$, $\mathbf{v} \in \mathbb{Q}^n$ is a vector, and $d \in \mathbb{Q}$ a length where dist$(\mathbf{v}, \mathcal{L}(\mathbf{B}) > \gamma(n) \cdot d$.

For any $\gamma \geq 1$, the promise problem GAPCVP$_\gamma$ is in NP. The NP-witness for $(\mathbf{B}, \mathbf{v}, d)$ being a YES instance is merely a vector $\mathbf{u} \in \mathcal{L}(\mathbf{B})$ satisfying dist$(\mathbf{v}, \mathbf{u}) \leq d$. About the complement of GAPCVP$_\gamma$ less is known.

(Lagarias et al., 1990; Håstad, 1988; Banaszczyk, 1993) showed that GAPCVP$_n$ is in NP$\cap$coNP. (Goldreich and Goldwasser, 2000) improved on this factor, and showed that GAPSVP$_{o(\sqrt{n})}$ is in NP $\cap$ coAM. They present a constant-round interactive proof system for the complement of the above promise problem with $\gamma(n) = \sqrt{n}/O(\log n) = o(\sqrt{n})$. That is, they show a proof system such that very-far instances (NO-instances) are always accepted, whereas close instances (YES-instances) are accepted with negligible probability. We shall present this interactive proof fully in this chapter.

Comment on Zero-Knowledge. The (constant-round) interactive proofs (for the complement of GAPCVP$_{o(\sqrt{n})}$ and GAPSVP$_{o(\sqrt{n})}$) are Perfect Zero-Knowledge with respect to the Honest Verifier (HVPZK). Thus, the complement of GAPCVP$_{o(\sqrt{n})}$ (resp., GAPSVP$_{o(\sqrt{n})}$) is in the class HVPZK. The existence of honest verifier statistical zero knowledge (HVSZK) proofs for the "NP direction" follows by Okamoto's result by which the class HVSZK is closed under complementation (Okamoto, 1996). (His result does extend to promise problems; cf., (Sahai and Vadhan, 1997)). Thus, GAPCVP$_{o(\sqrt{n})}$ (resp., GAPSVP$_{o(\sqrt{n})}$) is in the class HVSZK.

Comment on other norms. The proof systems can be adapted to any ℓ_p norm (and in particular to ℓ_1 and ℓ_∞). Specifically, we obtain constant-round (HVPZK) interactive proof systems for gap $n/O(\log n)$ (rather than gap $\sqrt{n/O(\log n)}$ as in l_2 norm). The result extends to any *computationally tractable* norm as defined in Section 3. (Except for Section 3, the rest of the chapter refers to CVP and SVP in ℓ_2 norm.)

Implication on proving non-approximability of CVP and SVP. Chapters 3 and 4 contain results on the hardness of approximating CVP and SVP. In particular we have seen that CVP is NP-hard to approximate within $n^{1/O(\log logn)}$ and SVP problem was shown NP-hard, under RUR-reductions, to approximate that for any constant factor less than $\sqrt{2}$. A natural question is what happens to the difficulty of SVP and CVP for larger factors of approximation. Can these results be improved or has the limit of inapproximability been reached? For which factor, do SVP and CVP become tractable?

Resolving this question is of interest also from a cryptographic stand point. As we have shown in Chapter 8 the conjectured difficulty of versions of both GAPCVP and GAPSVP have been suggested as basis for cryptographic primitives and schemes (Ajtai, 1996; Goldreich et al., 1997b; Ajtai and Dwork, 1997). In particular, Ajtai's one-way function assumes that GAPSVP$_{n^c}$ is hard (in worst case), where $c > 8$. (The constant c has been reduced to $c > 4$ by (Cai and Nerurkar, 1997), and in Chapter 8 we have seen that it can be further reduced to $c > 3.5$, or even $c > 3$ if certain "almost perfect" lattices can be efficiently decoded.) The security of the Ajtai-Dwork public-key encryption scheme is reduced to a special case of (a search version of) GAPSVP$_{n^c}$ (with some big c). And the trapdoor permutation suggested in (Goldreich et al., 1997b) relies on the conjectured difficulty of the CVP problem. A possible end goal toward which one could hope to carry this direction of research, is to base the existence of a one-way functions (and other cryptographic primitives) on the hardness of GAPSVP$_{\gamma(n)}$ for γ such that we can prove that GAPSVP$_\gamma$ and GAPCVP$_\gamma$ are NP-hard (or quasi-NP hard).

Placing the complement of promise problem GAPSVP$_{o(\sqrt{n})}$ in AM, and thus GAPSVP$_{o(\sqrt{n})} \in$ NP $\cap$ coAM sheds light on this question as follows ((Goldreich and Goldwasser, 2000; Cai and Nerurkar, 2000)). Two possibilities exist,

1 Either, GAPSVP$_{o(\sqrt{n})}$ is not NP-hard

2 Or, GAPSVP$_{o(\sqrt{n})}$ is NP-hard, which implies coNP $\subset$ AM and the Polynomial-Time Hierarchy collapses (by a result of (Boppana et al., 1987)).

Similarly, placing GAPCVP$_{o(\sqrt{n})}$ in NP $\cap$ coAM, implies that either

1 GAPCVP$_{o(\sqrt{n})}$ is not NP-hard, or

2 GAPCVP$_{o(\sqrt{n})}$ is NP-hard, and then the Polynomial-Time Hierarchy collapses.

Assuming the polynomial-time hierarchy does not collapse, this can be viewed as establishing limits on the NP-hardness of approximating CVP and SVP: Approximations to within a factor of $o(\sqrt{n})$ are not NP-hard. In terms of the cryptographic perspective, this seems to mean that if one attempts to base the security of a cryptosystem on an NP-hard version of approximate SVP or CVP, one should at minimum aim for approximation factors of $o(\sqrt{n}/\log n)$.

1. Closest vector problem

We consider the promise problem GAPCVP$_\gamma$ defined in the introduction, and present a constant-round interactive proof system for the complement of the above problem for gap $\gamma(n) = \sqrt{n/O(\log n)}$. Recall that the input is a triple $(\mathbf{B}, \mathbf{v}, d)$, where $\mathbf{B}$ is a basis for a lattice, $\mathbf{v}$ is a vector and $d \in \mathbb{Q}$ a length. We give an interactive proof system such that NO-instances (in which $\mathbf{v}$ is at distance greater than $\gamma(n) \cdot d$ from the lattice) are always accepted, whereas YES-instances (in which $\mathbf{v}$ is within distance d from $\mathcal{L}(\mathbf{B})$) are accepted with probability bounded away from 1.

More precisely, the theorem we prove is,

THEOREM 9.1 GAPCVP$_{\sqrt{n/O(\log n)}}$ *is in* coAM.

The proof system. Consider a "huge" sphere, denoted H. Specifically, we consider a sphere of radius $2^n \cdot \|(\mathbf{B}, \mathbf{v})\|$ centered at the origin, where $\|(\mathbf{B}, \mathbf{v})\|$ denotes the length of the largest vector in $\mathbf{B} \cup \{\mathbf{v}\}$. Let $\gamma = \gamma(n)$.

1 The verifier uniformly selects $\sigma \in \{0, 1\}$, a random lattice point in H, denoted $\mathbf{r}$, and an error vector, $\mathbf{t}$, uniformly distributed in a sphere of radius $\gamma d/2$. The verifier sends $\mathbf{x} \stackrel{\text{def}}{=} \mathbf{r} + \sigma\mathbf{v} + \mathbf{t}$ to the prover.

2 The prover responses with $\tau = 0$ if $\text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B})) < \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + \mathbf{v})$ and $\tau = 1$ otherwise.

3 The verifier accepts if and only if $\tau = \sigma$.

Implementation details. Several obvious implementation questions, arising from the above description, are

- *How to uniformly select a lattice point in H?* We uniformly select a point in H, represent this point as a linear combination of the basis vectors, and obtain a lattice point by rounding. This procedure partitions H into cells, most of them are parallelepipeds which are isomorphic to the basic cell/parallelepiped defined by the lattice. The exceptions are the partial parallelepipeds which are divided by the boundary of the sphere H. All the latter parallelepipeds are contained between two co-centered spheres, the larger being of radius $(2^n + n) \cdot L$ and the smaller being of radius $(2^n - n) \cdot L$, where $L \stackrel{\text{def}}{=} \|(\mathbf{B}, \mathbf{v})\| \geq \|\mathbf{B}\|$ is the radius of H. Thus, the fraction of these ("partial") parallelepipeds in the total number of parallelepipeds is bounded above by the volume encompassed between the above two spheres divided by the volume of the smaller sphere. This relative volume is at most

$$\frac{(2^n + n)^n - (2^n - n)^n}{(2^n - n)^n} = \left(1 + \frac{2n}{2^n - n}\right)^n - 1$$

$$< \frac{3n^2}{2^n}$$

It follows, that the above procedure generates random lattice points in a distribution which is at most poly$(n)\cdot 2^{-n}$ away from the uniform distribution over $\mathcal{L}(\mathbf{B}) \cap H$.

- *How to uniformly select a point in the unit sphere?* One may just invoke the general algorithm of (Dyer et al., 1991). Using this algorithm, it is possible to select almost uniformly a point in any convex body (given by a membership oracle). Alternatively, one may select the point by generating n samples from the standard normal distribution, and normalize the result so that a vector of length r appears with probability proportional to r^{-n} (see, e.g., (Knuth, 1981, Sec. 3.4.1)).

- *How to deal with finite precision?* In the above description, we assume all operations to be done with infinite precision. This is neither possible nor needed. We assume, instead, that the input entries (in the vectors), are given in rational representation and let m denote the number of bits in the largest of the corresponding integers. Then making all calculations with $n^3 \cdot m$ bits of precision, introduces an additional stochastic deviation of less than 2^{-n} in our bounds.

Analysis of the protocol. By the above, it should be clear that the verifier's actions in the protocol can be implemented in probabilistic polynomial-time. We will show that, for $\gamma(n) = \sqrt{n/O(\log n)}$, the above protocol constitutes a (honest verifier perfect zero-knowledge) proof system for the promise problem GAPCVP$_\gamma$, with perfect completeness and soundness error bounded away from 1.

CLAIM 9.2 (COMPLETENESS) *If* $\text{dist}(\mathbf{v}, \mathcal{L}(\mathbf{B})) > \gamma(n)\cdot d$ *then the verifier always accepts* (when interacting with the prover specified above).

Proof: Under the above hypothesis, for every point $\mathbf{x}$ (and in particular the messages sent by verifier in step 1), we have $\text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B})) + \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + \mathbf{v}) > \gamma d$ (or else $\text{dist}(\mathbf{v}, \mathcal{L}(\mathbf{B})) = \text{dist}(\mathcal{L}(\mathbf{B}) + \mathbf{v}, \mathcal{L}(\mathbf{B})) \leq \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + \mathbf{v}) + \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B})) \leq d\gamma$). Thus, for every message, $\mathbf{x} = \mathbf{r} + \sigma\mathbf{v} + \mathbf{t}$, sent by the verifier we have

$$\begin{aligned} \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + \sigma\mathbf{v}) &= \text{dist}(\mathbf{r} + \mathbf{t}, \mathcal{L}(\mathbf{B})) \leq \|\mathbf{t}\| \leq \frac{d\gamma}{2} \\ \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + (1-\sigma)\mathbf{v}) &> \gamma d - \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + \sigma\mathbf{v}) \geq \frac{d\gamma}{2} \end{aligned}$$

Thus, it is always the case that $\text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + \sigma\mathbf{v}) < \text{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}) + (1-\sigma)\mathbf{v})$ and the prover responses with $\tau = \sigma$. □

CLAIM 9.3 (ZERO-KNOWLEDGE) *The above protocol is a honest-verifier perfect zero-knowledge interactive proof system for triples* $(\mathbf{B}, \mathbf{v}, d)$ *satisfying* $\text{dist}(\mathbf{v}, \mathcal{L}(\mathbf{B})) > \gamma(n) \cdot d$.

Proof: The simulator just reads the verifier's choice and returns it as the prover's message. Thus, the simulator's output will consist of coins for the verifier and the prover's response. By the above proof, this distribution is identical the verifier's view in the real protocol. □

CLAIM 9.4 (SOUNDNESS) *Let* $c > 0$ *be a constant independent of* n, *and* $\gamma(n) \geq \sqrt{n/(c \ln n)}$. *If* $\text{dist}(\mathbf{v}, \mathcal{L}(\mathbf{B})) \leq d$ *then, no matter what the prover does, the verifier accepts with probability at most* $1 - n^{-2c}$.

The above is slightly inaccurate as the statement holds only for sufficiently large n's (depending on the constant c). For smaller (fixed) dimension, one may replace the protocol by an immediate computation using Lenstra's algorithm (Lenstra, 1983). The same holds for Claim 9.12 below.

1.1 Proof of the soundness claim

Let ξ_0 (resp., ξ_1) a random variable representing the message sent by the verifier condition on $\sigma = 0$ (resp., $\sigma = 1$). We first bound the statistical distance between the two random variables by $(1 - 2n^{-2c})$. Given this bound, we have for any prover strategy P'

$$\begin{aligned}
\Pr(P'(\xi_\sigma) = \sigma) &= \frac{1}{2} \cdot \Pr(P'(\xi_0) = 0) + \frac{1}{2} \cdot \Pr(P'(\xi_1) = 1) \\
&= \frac{1}{2} \cdot \big(\Pr(P'(\xi_0) = 0) + 1 - \Pr(P'(\xi_1) = 0)\big) \\
&= \frac{1}{2} + \frac{1}{2} \cdot \big(\Pr(P'(\xi_0) = 0) - \Pr(P'(\xi_1) = 0)\big) \\
&\leq \frac{1}{2} + \frac{1}{2} \cdot (1 - 2n^{-2c}) \\
&= 1 - n^{-2c}
\end{aligned}$$

Thus, all that remains is to prove the above bound on the statistical distance between ξ_0 and ξ_1. The statistical distance between the two random variables is due to two sources:

1 In case $\sigma = 1$ the point $\mathbf{r} + \mathbf{v}$ may be out of the sphere H (whereas, by choice, $\mathbf{r}$ is alway in H). However, since H is much bigger than $\mathbf{v}$ this

happens rarely (i.e., with probability at most $3n^2 \cdot 2^{-n}$; see above). Furthermore, the statistical difference between uniform distribution on the lattice point in the sphere H and the same distribution shifted by adding the vector v is negligible. Specifically, we may bound it by $n^{-2c} > 3n^2 \cdot 2^{-n}$.

2 Let $\mathbf{v}'$ be $\mathbf{v}$ reduced modulo the basis. For each lattice point, $\mathbf{p}$, we consider the statistical distance between $\mathbf{p}+\mathbf{t}$ and $\mathbf{p}+\mathbf{v}'+\mathbf{t}$, where $\mathbf{t}$ is as above. The main thing is to bound this statistical distance. The rest of the proof is devoted to this.

Thus, it suffices to consider the statistical distance between $\mathbf{t}$ and $\mathbf{v}'+\mathbf{t}$, where $\mathbf{t}$ is as above. In the first case the probability mass is uniformly distributed in a sphere of radius $\gamma d/2$ centered at $\mathbf{0}$ whereas in the second case the probability mass is uniformly distributed in a sphere of radius $\gamma d/2$ centered at $\mathbf{v}'$, where $\|\mathbf{v}'\| \leq d$. Without loss of generality, we consider $\mathbf{v}' = [d, 0, ..., 0]^T$. Normalizing things (by division with $\gamma d/2$), it suffices to consider the statistical distance between the following two distributions:

(D1) Uniform distribution in a unit sphere centered at the origin.

(D2) Uniform distribution in a unit sphere centered at $[\epsilon, 0, .., 0]^T$ where $\epsilon = d/(\gamma d/2) = 2/\gamma$.

Observe that the statistical distance between the two distributions equals *half* the volume of the symmetric difference of the two spheres divided by the volume of a sphere. Thus, we are interested in the relative symmetric difference of the two spheres. Recall two basic facts –

FACT 9.5 (e.g., (Apostol, 1969, Vol. 2, Sec. 11.33, Ex. 4)) *The volume of an n-dimensional sphere of radius r is* $v_n(r) \stackrel{\text{def}}{=} \frac{\pi^{n/2}}{\Gamma((n/2)+1)} \cdot r^n$, *where* $\Gamma(x) = (x-1)\cdot\Gamma(x-1)$, $\Gamma(1) = 1$, $\Gamma(0.5) = \sqrt{\pi}$.

FACT 9.6 (e.g., (Knuth, 1973, Sec. 1.2.11.2, Exer. 6)) *For every real* $x \geq 2$, $\Gamma(x+1) \approx \sqrt{2\pi x}\cdot(x/e)^x$. *Thus, for every integer* $m \geq 2$,

$$\frac{\Gamma(m+0.5)}{\Gamma(m)} \approx \sqrt{m} \approx \frac{\Gamma(m+1)}{\Gamma(m+0.5)}$$

LEMMA 9.7 (SYMMETRIC DIFFERENCE OF SPHERES) *Let* S_0 *(resp.* S_ϵ*) be a unit sphere at the origin (resp. at distance* ϵ *from the origin). Then* *relative symmetric difference between the spheres (i.e., the symmetric difference divided by the volume) is at most*

$$2 - \epsilon \cdot \frac{(1-\epsilon^2)^{(n-1)/2}}{3} \cdot \sqrt{n}$$

Our bound is not tight. Still, we note that the bound cannot be decreased below $2 - (1 - (\epsilon/2)^2)^{(n-1)/2} \cdot \sqrt{n}$, and that both expressions are equivalent as far as our application goes.

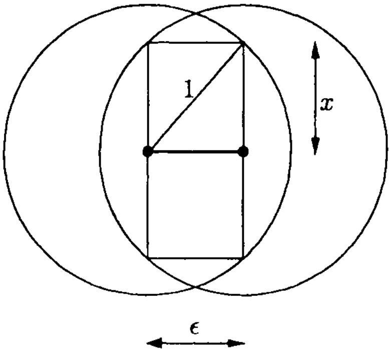


Figure 9.1. The cylinder encompassed by S_0 and S_ϵ. The axis is marked in bold and its radius $x = (1-\epsilon^2)^{0.5}$ is computed from the center of the left sphere.

Proof: We bound the volume of the intersection between S_0 and S_ϵ from below. Specifically, we look at the $(n-1)$-dimensional cylinder of height ϵ, which is centered at the axis connecting the centers of S_0 and S_ϵ and is encompassed by $S_0 \cap S_\epsilon$. See Figure 9.1. The radius of this cylinder is $\sqrt{1-\epsilon^2}$. Thus its volume is $\epsilon \cdot v_{n-1}(\sqrt{1-\epsilon^2})$. Using Facts 9.5 and 9.6 we have

$$\begin{aligned}
\frac{\mathrm{vol}(S_0 \cap S_\epsilon)}{\mathrm{vol}(S_0)} &> \frac{\epsilon \cdot v_{n-1}(\sqrt{1-\epsilon^2})}{v_n(1)} \\
&= \frac{\epsilon \cdot (1-\epsilon^2)^{(n-1)/2} \cdot v_{n-1}(1)}{v_n(1)} \\
&= \epsilon \cdot (1-\epsilon^2)^{(n-1)/2} \cdot \frac{\pi^{(n-1)/2}/\Gamma(((n-1)/2)+1)}{\pi^{n/2}/\Gamma((n/2)+1)} \\
&= \epsilon \cdot (1-\epsilon^2)^{(n-1)/2} \cdot \frac{\Gamma((n/2)+1)}{\sqrt{\pi} \cdot \Gamma((n/2)+0.5)} \\
&\approx \epsilon \cdot (1-\epsilon^2)^{(n-1)/2} \cdot \frac{\sqrt{n/2}}{\sqrt{\pi}}
\end{aligned}$$

The lemma follows.□

Using Lemma 9.7, with $\epsilon = 2/g(n) \leq \sqrt{4c\ln n/n}$, we bound form above the statistical distance between distributions (D1) and (D2) by

$$\begin{aligned}\frac{1}{2}\cdot\left(2-\epsilon\sqrt{n}\cdot\frac{(1-\epsilon^2)^{\frac{n-1}{2}}}{3}\right) &= 1-\frac{\sqrt{4c\ln n}}{6}\cdot\left(1-\frac{4c\ln n}{n}\right)^{\frac{n-1}{2}} \\ &< 1-\frac{\sqrt{c\ln n}}{3}\cdot\left(1-\frac{2c\ln n}{\frac{n}{2}}\right)^{n/2} \\ &< 1-3\cdot n^{-2c}\end{aligned}$$

where the last inequality uses $\sqrt{c\ln n} > 9$. Thus, the statistical distance between ξ_0 and ξ_1 is bounded by $n^{-2c}+1-3\cdot n^{-2c}$ (where the extra n^{-2c} term comes from Item 1 above). The soundness claim follows. □

1.2 Conclusion

Combining the above protocol with known transformations (Goldwasser and Sipser, 1986; Babai, 1985) we get

THEOREM 9.8 *For any approximation factor $\gamma(n) = \sqrt{n/O(\log n)}$, the promise problem* GAPCVP$_{\gamma(n)}$ *is in* NP∩coAM. *Furthermore, the complement of* GAPCVP$_{\gamma(n)}$ *has a* HVPZK *constant-round proof system.*

The interesting part is the membership of GAPCVP$_{\sqrt{n}}$ in coAM. This reduces the gap factor for which "efficient proof systems" exists: (Lagarias et al., 1990), (Håstad, 1988) and (Banaszczyk, 1993) have previously shown that GAPCVP$_n$ is in coNP.

2. Shortest vector problem

Let us slightly modify the definition of GAPSVP$_\gamma$ given in the introduction to the following (equivalent) definition. The YES instances (i.e., having short vectors) of GAPSVP$_{\gamma(n)}$ are pairs $(\mathbf{B}, d)$ where $\mathbf{B}$ is a basis for a lattice $\mathcal{L}(\mathbf{B})$ in $\mathbb{Q}^n$, $d \in \mathbb{Q}$ and $\text{dist}(\mathbf{v}_1, \mathbf{v}_2) \leq d$ for some $\mathbf{v}_1 \neq \mathbf{v}_2$ in $\mathcal{L}(\mathbf{B})$. The NO instances (i.e., "strongly violating" short vectors) are pairs $(\mathbf{B}, d)$ where $\mathbf{B}$ and d are as above but $\text{dist}(\mathbf{v}_1, \mathbf{v}_2) > \gamma(n)\cdot d$ for all $\mathbf{v}_1 \neq \mathbf{v}_2$ in $\mathcal{L}(\mathbf{B})$.

We present a constant-round interactive proof system for the complement of the above problem for gap $\gamma(n) = \sqrt{n/O(\log n)}$. Recall that the input is a pair $(\mathbf{B}, d)$, where B is a basis for a lattice and $d \in \mathbb{Q}$. That is, we'll show that NO-instances (in which the shortest vector in $\mathcal{L}(\mathbf{B})$ has length greater than $\gamma(n)\cdot d$) are always accepted, whereas YES-instances (in which $\mathcal{L}(\mathbf{B})$ has a nonzero vector of length at most d) are accepted with probability bounded away from 1.

The exact theorem to be proven is

THEOREM 9.9 GAPSVP$_{\sqrt{n/O(\log n)}}$ *is in* coAM.

The proof system. Consider a huge sphere, denoted H (as in Section 1). Specifically, we consider a sphere of radius $2^n \cdot ||B||$ centered at the origin. Let $\gamma = \gamma(n)$.

1 The verifier uniformly selects a random lattice point, $\mathbf{p}$, in H, and an error vector, $\mathbf{t}$, uniformly distributed in a sphere of radius $\gamma d/2$. The verifier sends $\widetilde{\mathbf{p}} \stackrel{\text{def}}{=} \mathbf{p} + \mathbf{t}$ to the prover.

2 The prover sends back the closest lattice point to $\widetilde{\mathbf{p}}$.

3 The verifier accepts iff the prover has answered with $\mathbf{p}$.

CLAIM 9.10 (COMPLETENESS) *If any two distinct lattice points are at distance greater than γd, then the verifier always accepts.*

Proof: Under the above hypothesis, for every point $\mathbf{x}$ (and in particular the message sent by verifier in step 1), we have at most one lattice $\mathbf{p}$ so that $\text{dist}(\mathbf{x}, \mathbf{p}) \leq \gamma d/2$ (or else $\text{dist}(\mathbf{v}_1, \mathbf{v}_2) \leq \text{dist}(\mathbf{x}, \mathbf{v}_1) + \text{dist}(\mathbf{x}, \mathbf{v}_2) \leq \gamma d$). Since we have $\text{dist}(\widetilde{\mathbf{p}}, \mathbf{p}) \leq \gamma d/2$, the prover always returns $\mathbf{v}$. □

CLAIM 9.11 (ZERO-KNOWLEDGE) *The above protocol is honest-verifier perfect zero-knowledge for pairs $(\mathbf{B}, d)$ such that every two distinct points in $\mathcal{L}(\mathbf{B})$ are at distance greater than γd.*

Proof: The simulator just reads the verifier's choice and returns it as the prover's message. Thus, the simulator's output will consist of coins for the verifier and the prover's response. By the above proof, this distribution is identical to the verifier's view in the real protocol. □

CLAIM 9.12 (SOUNDNESS) *Let $c > 0$ and $\gamma(n) \geq \sqrt{n/(c \ln n)}$, if for some $\mathbf{v}_1 \neq \mathbf{v}_2$ in $\mathcal{L}(\mathbf{B})$, $\text{dist}(\mathbf{v}_1, \mathbf{v}_2) \leq d$ then, no matter what the prover does, the verifier accepts with probability at most $1 - n^{-2c}$.*

Proof: Let $\mathbf{p}' \stackrel{\text{def}}{=} \mathbf{p} + (\mathbf{v}_1 - \mathbf{v}_2)$, where $\mathbf{p}$ is the lattice point chosen by the verifier in Step 1. Clearly, $\text{dist}(\mathbf{p}, \mathbf{p}') \leq d$. Let ξ be a random variable representing the message actually sent by the verifier, and let $\xi' = \xi + (\mathbf{v}_1 - \mathbf{v}_2)$. Using the analysis in the proof of Claim 9.4, we bound the statistical distance between these two random variables by $(1 - 3n^{-2n})$. (Note that ξ corresponds to ξ_0 and ξ' corresponds to ξ_1

with $\mathbf{v} = \mathbf{v}_1 - \mathbf{v}_2$.) Given this bound, we have for any prover strategy P'

$$\begin{aligned}\Pr(P'(\xi) = \mathbf{p}) &\leq (1 - 3n^{-2n}) + \Pr(P'(\xi') = \mathbf{p}) \\ &\leq 2 - 3n^{-2n} - \Pr(P'(\xi') = \mathbf{p}')\end{aligned}$$

However, the event $P'(\xi') = \mathbf{p}'$ is almost as probable as $P'(\xi) = \mathbf{p}$ (with the only difference in probability due to the case where $\mathbf{p}'$ is outside the sphere which happens with probability at most n^{-2n}). Thus, we have

$$\begin{aligned}2 \cdot \Pr(P'(\xi) = \mathbf{p}) &< \Pr(P'(\xi) = \mathbf{p}) + \Pr(P'(\xi') = \mathbf{p}') + n^{-2n} \\ &\leq 2 - 2n^{-2n}\end{aligned}$$

and the claim follows. □

Conclusion. Combining the above protocol with known transformations (i.e., (Goldwasser and Sipser, 1986) and (Babai, 1985)), we get

THEOREM 9.13 *For any approximation factor* $\gamma(n) = \sqrt{n/O(\log n)}$, *the promise problem* GAPSVP$_{\gamma(n)}$ *is in* NP ∩ coAM. *Furthermore, the complement of* GAPSVP$_{\gamma(n)}$ *has a* HVPZK *constant-round proof system.*

Again, the interesting part is the membership of GAPSVP$_{o(\sqrt{n})}$ in coAM. This reduces the gap factor for which "efficient proof systems" exists: Lagarias *et. al.* (Lagarias et al., 1990) had previously shown that GAPSVP$_n$ is in coNP.

3. Treating other norms

The underlying ideas of Theorems 9.8 and 9.13 can be applied to provide (HVPZK) constant-round proof systems for corresponding gap problems regarding any "computationally tractable" norm and in particular for all ℓ_p-norms (e.g., the ℓ_1 and ℓ_∞ norms). The gap factor is however larger: $n/O(\log n)$ rather than $\sqrt{n/O(\log n)}$.

Tractable norms. Recall the norm axioms (for a generic norm $\|\cdot\|$) –

(N1) For every $\mathbf{v} \in \mathbb{R}^n$, $\|\mathbf{v}\| \geq 0$, with equality holding if and only if $\mathbf{v}$ is the zero vector.

(N2) For every $\mathbf{v} \in \mathbb{R}^n$ and any $\alpha \in \mathbb{R}$, $\|\alpha\mathbf{v}\| = |\alpha| \cdot \|\mathbf{v}\|$.

(N3) For every $\mathbf{v}, \mathbf{u} \in \mathbb{R}^n$, $\|\mathbf{v} + \mathbf{u}\| = \|\mathbf{v}\| + \|\mathbf{u}\|$. (Triangle Inequality).

To allow the verifier to conduct is actions in polynomial-time, we make the additional two requirements

(N4) The norm function is polynomial-time computable. That is, there exist a polynomial-time algorithm that, given a vector $\mathbf{v}$ and an accuracy parameter δ, outputs a number in the interval $[\|\mathbf{v}\| \pm \delta]$. We stress that the algorithm is uniform over all dimensions.

(N5) The unit sphere defined by the norm contains a ball of radius $2^{-\mathrm{poly}(n)}$ centered at the origin, and is contained in a ball of radius $2^{\mathrm{poly}(n)}$ centered at the origin. That is, there exists a polynomial $\mathbf{p}$ so that for all n's

$$\begin{aligned} \{\mathbf{v} \in \mathbb{R}^n : \|\mathbf{v}\|_2 \leq 2^{-p(n)}\} &\subseteq \{\mathbf{v} \in \mathbb{R}^n : \|\mathbf{v}\| \leq 1\} \\ &\subseteq \{\mathbf{v} \in \mathbb{R}^n : \|\mathbf{v}\|_2 \leq 2^{p(n)}\} \end{aligned}$$

where $\|\mathbf{v}\|_2$ is the Euclidean (ℓ_2) norm of $\mathbf{v}$.

Note that axioms (N4) and (N5) are satisfied by all (the standard) ℓ_p-norms.[1] On the other hand, by (Dyer et al., 1991), axioms (N4) and (N5) suffice for constructing a probabilistic algorithm which given n, generates in time poly(n) a vector which is almost uniformly distributed in the n-dimensional unit sphere w.r.t the norm. Specifically, by axioms (N2) and (N3), the unit sphere is a convex body, and axioms (N4) and (N5) imply the existence of a so-called "well-guaranteed weak membership oracle" (cf., (Grotschel et al., 1993)) as required by the convex body algorithm of Dyer *et. al.* (Dyer et al., 1991) (and its improvements – e.g., (Kannan et al., 1997)).

Our protocols can be adapted to any norm satisfying the additional axioms (N4) and (N5). We modify the protocols of the previous sections so that the error vector, $\mathbf{t}$, is chosen uniformly among the vectors of norm less than $\gamma(n)d/2$ (rather than being chosen uniformly in a sphere of radius $\gamma(n)d/2$). Here we use $\gamma(n) \stackrel{\mathrm{def}}{=} n/O(\log n)$. Clearly the completeness and zero-knowledge claims continue to hold as they merely relied on the triangle inequality (i.e., Norm axiom (N3)). In the proof of the soundness claim, we replace Lemma 9.7 by the following lemma in which distance refers to the above norm (rather than to Euclidean norm):

LEMMA 9.14 *For every $c > 0$, let p be a point at distance $\epsilon < 1$ from the origin. Then the relative symmetric difference between the set of points*

[1] Actually, for any ℓ_p-norm, there is a simple algorithm for uniformly selecting a point, $(x_1, ..., x_n)$, in the corresponding unit sphere: Generate n independent samples, $x_1, ..., x_n$, each with density function e^{-x^p}, and normalize the result so that a vector of norm r appears with probability proportional to r^{-n}.

of distance 1 from the origin and the set of points of distance 1 from p *is at most* $2 \cdot (1 - (1 - \epsilon)^n)$.

We comment that the bound is quite tight for both the ℓ_1 and the ℓ_∞ norm. That is, in both cases the relative symmetric difference is at least $2 - (1 - (\epsilon/2))^n$.[2]

Proof: Let $\mathcal{B}(\mathbf{0}, r)$ (resp., $\mathcal{B}(\mathbf{p}, r)$) denote the set of points within distance r from the origin (resp., from $\mathbf{p}$). The symmetric difference between $\mathcal{B}(\mathbf{0}, 1)$ and $\mathcal{B}(\mathbf{p}, 1)$ equals twice the volume of $\mathcal{B}(\mathbf{p}, 1) \setminus \mathcal{B}(\mathbf{0}, 1)$. This volume is clearly bounded above by $\mathcal{B}(\mathbf{p}, 1) \setminus \mathcal{B}(\mathbf{p}, 1 - \epsilon)$. By the norm axioms (N1) and (N2), we have

$$\frac{\mathrm{vol}(\mathcal{B}(\mathbf{p}, 1) \setminus \mathcal{B}(\mathbf{p}, 1 - \epsilon))}{\mathrm{vol}(\mathcal{B}(\mathbf{p}, 1))} = 1 - (1 - \epsilon)^n,$$

and the lemma follows. □

Using $\epsilon = 2/\gamma(n)$ and $\gamma(n) = n/O(\log n)$, we conclude that the proof system has soundness error bounded above by

$$1 - \left(1 - \frac{O(\log n)}{n}\right)^n = 1 - \frac{1}{\mathrm{poly}(n)}.$$

Repeating it polynomially many times in parallel we get

THEOREM 9.15 *Both* GAPCVP *and* GAPSVP *defined for any norm and gap factor* $\gamma(n) = n/O(\log n)$ *are in* NP ∩ coAM. *Furthermore, the complement promise problems have* HVPZK *constant-round proof systems.*

4. What does it mean?

Let $\Pi = (\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$ be a promise problem for which Π_{MAYBE} denotes the instances which are neither YES nor NO instances of Π. Thus, the entire set of instances is $\Pi_{\mathrm{YES}} \cup \Pi_{\mathrm{NO}} \cup \Pi_{\mathrm{MAYBE}}$. The complement of a promise problem Π is simply $(\Pi_{\mathrm{NO}}, \Pi_{\mathrm{YES}})$. To say that a language L is reducible to a promise problem Π means that:
There exists a polynomial time procedure that on input $x \in L$,

- computes instances $x_1, ..., x_k$ (possibly adaptively), such that

[2] To verify the above claim for ℓ_∞, consider the point $p = (\epsilon, \epsilon, ..., \epsilon)$. Clearly, the intersection of the unit sphere centered at the origin and the unit sphere centered at p is $(2 - \epsilon)^n$, whereas each sphere has volume 2^n. For ℓ_1, consider the point $p = (\epsilon, 0, ..., 0)$. Again, the intersection is a sphere of radius $1 - (\epsilon/2)$ (according to the norm in consideration).

- given b_i such that $b_i = \text{YES}$ for all $x_i \in \Pi_{yes}$ and $b_i = \text{NO}$ for all $x_i \in \Pi_{\text{NO}}$, cann compute whether $x \in L$ or not.

Note that in the above definition, we do not care about the value of b_i when x_i does not satisfy the promise. Recall that Π is NP-hard if for all languages L in NP, L reduces to Π. We are now ready to show the following theorem (which will be used to interpret the significance of the interactive proofs we showed for CVP and SVP in the previous sections).

THEOREM 9.16 *Let* $\Pi = (\Pi_{\text{YES}}, \Pi_{\text{NO}})$ *be a promise problem. Suppose there exists a polynomial-time recognizable relation* R *so that*

- *For every* $x \in \Pi_{\text{YES}} \cup \Pi_{\text{MAYBE}}$, *there exists a* $y \in \{0,1\}^*$ *such that* $(x, y) \in R$ *(and* $|y| = \text{poly}(|x|)$*) and*
- *For every* $x \in \Pi_{\text{NO}}$, *for all* $y \in \{0,1\}^*$, (x, y) *is not in* R.
- *The complement of* Π *is in* AM.

Then: Π *is* NP*-hard implies* coNP $\subseteq$ AM.

Proof: Let $L \in coNP$, and Π be NP-hard. By the NP-hardness of Π, and thus corresponding coNP-hardness of Π complement, L is reducible to Π complement (itself a promise problem). We shall use this latter reduction to construct an AM-proof system for L and conclude our proof.

Let us denote from here on Π complement as $(\Pi'_{\text{YES}}, \Pi'_{\text{NO}})$. On input x, the prover first sends to the verifier a transcript of the reduction (from L to Π complement) applied to x. This transcript consists of instances $x_1, ..., x_k$, and corresponding $b_1, ..., b_k$ such that $b_i = \text{YES}$ for all $x_i \in \Pi'_{\text{YES}}$ and $b_i = \text{NO}$ for all $x_i \in \Pi'_{\text{NO}}$. For all those x_i which are outside of the promise, and are in Π_{MAYBE} – the prover sends $b_i = \text{NO}$.

Next, the prover proves each of the answers it gave as follows: for $x \in \Pi'_{\text{NO}} = \Pi_{\text{YES}}$, the prover sends the verifier y such that (x, y) in R, for $x \in \Pi'_{\text{YES}} = \Pi_{\text{NO}}$, the prover and verifier run the AM-proof system for Π complement. And, for all those queries x in Π_{MAYBE}, the prover shows again a y such that $(x', y) \in R$. The main observation is that a yes instance can not turn into a no instance and vice versa, which is true by the soundness of the AM procedure for Π complement and condition (2) in the theorem statement. The fact that prover claimed that maybe instances are no instances are of no concern by the definition of a reduction to a promise problem. All these AM-proofs are run in parallel, and so the result is an MAM-proof system (which can be converted into an AM-proof system (Babai, 1985)). In case of a randomized (smart) reduction, we let the verifier select the random input (to the reduction) and continue as above. □

We can finally get our implications.

COROLLARY 9.17 *For any* $\gamma(n) = O(\sqrt{n}/\log n)$, *if either* GAPCVP$_\gamma$ *or* GAPSVP$_\gamma$ *is* NP-*hard then* coNP $\subseteq$ AM.

An older result of (Boppana et al., 1987) showed that if coNP $\subset$ AM then the Polynomial-Time Hierarchy collapses. Thus (if one does not believe the collapse of the polynomial time hierarchy) we can take the corollary as evidence of the impossibility of proving NP-Hardness result for approximation factor below $\sqrt{n}$ for CVP or SVP.

5. Notes

The techniques described in Subsection 1.1 of Chapter 8 can be used to somehow simplify the proof systems presented in this chapter. (See (Goldreich and Goldwasser, 2000, Section 8) for details.) The proof systems presented in this chapter can be easily adapted to other lattice problems. For example, a proof system for GAPCRP$_{o(\sqrt{n})}$ is the following: the prover guesses a deep hole in the lattice, and then uses the proof system of (Goldreich and Goldwasser, 2000) to prove that this point is far from the lattice. Together with Theorem 7.10, this puts GAPCRP$_{\sqrt{n}}$ in NP $\cap$ coAM, showing that the covering radius problem is not likely to be NP-hard to approximate within factors $\gamma = \sqrt{n}$. Interestingly, when the factor γ is less than $\sqrt{n}$, GAPCRP$_\gamma$ is not even known to be in NP.

Proof systems for SIVP$_\gamma$ and SBP$_\gamma$ were given in (Blömer and Seifert, 1999), but only for approximation factors $\gamma(n) = n/\log n$. It is not clear if those results can be improved to $\gamma(n) = \sqrt{n}/\log n$ as for the other lattice problems.

-4Ckj6UjgE2iN1+kY-

References

Adleman, L. M. (1995). Factoring and lattice reduction. Manuscript.

Ajtai, M. (1996). Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 99–108, Philadelphia, Pennsylvania. ACM.

Ajtai, M. (1998). The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 10–19, Dallas, Texas.

Ajtai, M. and Dwork, C. (1997). A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 284–293, El Paso, Texas.

Ajtai, M., Kumar, R., and Sivakumar, D. (2001). A sieve algorithm for the shortest lattice vector problem. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, STOC 2001*, pages 266–275, Heraklion, Crete, Greece. ACM.

Apostol, T. (1969). *Calculus*, volume 2. John Wiley & Sons, Inc., second edition.

Arora, S., Babai, L., Stern, J., and Sweedyk, E. Z. (1996). *Approximation algorithms for NP-hard problems*, chapter Hardness of Approximation. PWS Publishing, Boston.

Arora, S., Babai, L., Stern, J., and Sweedyk, E. Z. (1997). The hardness of approximate optima in lattices, codes, and systems of linear equations. *Journal of Computer and System Sciences*, 54(2):317–331. Preliminary version in FOCS'93.

Babai, L. (1985). Trading group theory for randomness. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, STOC 85*, pages 421–429, Providence, Rhode Island. ACM, ACM.

Babai, L. (1986). On Lovasz' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13.

Banaszczyk, W. (1993). New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296:625–635.

Banihashemi, A. H. and Khandani, A. K. (1998). On the complexity of decoding lattices using the Korkin-Zolotarev reduced basis. *IEEE Transactions on Information Theory*, 44(1):162–171.

Bellare, M., Goldwasser, S., Lund, C., and Russell, A. (1993). Efficient probabilistically checkable proofs and applications to approximation. In *Proc. 25th Ann. ACM Symp. on Theory of Computing*, pages 294–304, San Diego, CA.

Bellare, M. and Rogaway, P. (1993). Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the first ACM Conference on Computer and Communications Security*. ACM.

Blömer, J. (2000). Closest vectors, successive minima and dual hkz-bases of lattices. In Montanari, U., Rolim, J. D. P., and Welzl, E., editors, *Proc. 27th Ann. International Coll. on Automata, Languages and Programming (ICALP)*, volume 1853 of *Lecture Notes in Computer Science*, pages 248–259, Geneva, Switzerland. Springer Verlag.

Blömer, J. and Seifert, J.-P. (1999). On the complexity of computing short linearly independent vectors and short bases in a lattice. In *Proceedings of the 31th Annual ACM Symposium on Theory of Computing*, pages 711–720, Atlanta, Georgia.

Boppana, R., Håstad, J., and Zachos, S. (1987). Does co-np have short interactive proofs? *Information Processing Letters*, 25:127–132.

Bruck, J. and Naor, M. (1990). The hardness of decoding linear codes with preprocessing. *IEEE Transactions on Information Theory*, 36(2):381–385.

Cai, J.-Y. (2001). On the average-case hardness of CVP. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 308–317, Las Vegas, Nevada. IEEE, IEEE.

Cai, J.-Y. and Nerurkar, A. (2000). A note on the non-NP-hardness of approximate lattice problems under general Cook reductions. *Information Processing Letters*, 76(1–2):61–66.

Cai, J.-Y. and Nerurkar, A. P. (1997). An improved worst-case to average-case connection for lattice problems (extended abstract). In *38th Annual Symposium on Foundations of Computer Science*, pages 468–477, Miami Beach, Florida. IEEE.

Cai, J.-Y. and Nerurkar, A. P. (1999). Approximating the SVP to within a factor $(1 + 1/dim^{\epsilon})$ is NP-hard under randomized reductions. *Journal of Computer and System Sciences*, 59(2):221–239.

Cassels, J. W. S. (1971). *An introduction to the geometry of numbers*. Springer-Verlag, New York.

Cohen, H. (1996). *A course in computational algebraic number theory*, volume 138 of *Graduate texts in mathematics*. Springer.

Coster, M. J., Joux, A., LaMacchia, B. A., Odlyzko, A. M., Schnorr, C.-P., and Stern, J. (1992). Improved low-density subset sum algorithms. *Computational Complexity*, 2(2):111–128.

Dinur, I., Kindler, G., Raz, R., and Safra, S. (1999). An improved lower bound for approximating cvp. Manuscript.

Dinur, I., Kindler, G., and Safra, S. (1998). Approximating CVP to within almost-polynomial factors is NP-hard. In *39th Annual Symposium on Foundations of Computer Science*, Palo Alto, California. IEEE.

Dumer, I., Micciancio, D., and Sudan, M. (1999). Hardness of approximating the minimum distance of a linear code. In *40th Annual Symposium on Foundations of Computer Science*, pages 475–484, New York, New York. IEEE.

Dyer, M., Frieze, A., and Kannan, R. (1991). A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM*, 38:1–17.

Feige, U. and Micciancio, D. (2001). The inapproximability of lattice and coding problems with preprocessing. Manuscript.

Fischlin, R. and Seifert, J.-P. (1999). Tensor-based trapdoors for CVP and their application to public key cryptography. In *7th IMA International Conference "Cryptography and Coding"*, volume 1746 of *Lecture Notes in Computer Science*, pages 244–257. Springer-Verlag.

Forney Jr., G. D. (1973). The Viterbi algorithm. In *Proc. IEEE*, volume 61, pages 268-278. IEEE.

Forney Jr., G. D. (1994). Density/length profiles and trellis complexity of lattices. *IEEE Transactions on Information Theory*, 40(6):1753–1772.

Frieze, A. M. (1986). On the Lagarias-Odlyzko algorithm for the subset sum problem. *SIAM Journal on Computing*, 15(2):536–539.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability, a guide to the theory of NP-completeness*. A Series of books in the mathematical sciences. W. H. Freeman, San Francisco.

Gauss, C. F. (1801). Disquisitiones arithmeticae, (leipzig 1801), art. 171. Yale University Press, 1966. English translation by A.A. Clarke.

Gilbert, E. N. (1952). A comparison of signaling alphabets. *AT&T Technical Journal*, 31:504–522.

Goldreich, O. and Goldwasser, S. (2000). On the limits of nonapproximability of lattice problems. *Journal of Computer and System Sciences*, 60(3):540–563. Preliminary version in STOC'98.

Goldreich, O., Goldwasser, S., and Halevi, S. (1996). Collision-free hashing from lattice problems. Technical Report TR96-056, Electronic Colloquium on Computational Complexity (ECCC).

Goldreich, O., Goldwasser, S., and Halevi, S. (1997a). Eliminating decryption errors in the Ajtai-Dwork cryptosystem. In (Kaliski Jr., 1997), pages 105–111.

Goldreich, O., Goldwasser, S., and Halevi, S. (1997b). Public-key cryptosystems from lattice reduction problems. In (Kaliski Jr., 1997), pages 112–131.

Goldreich, O. and Levin, L. (1989). A hard predicate for all one-way functions. In *Proceedings of the 21st Annual Symposium on Theory of Computing (STOC)*. ACM.

Goldreich, O., Micciancio, D., Safra, S., and Seifert, J.-P. (1999). Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Information Processing Letters*, 71(2):55–61.

Goldwasser, S. and Micali, S. (1984). Probabilistic encryption. *Journal of Computer and System Sience*, 28(2):270–299. Preliminary version in STOC'82.

Goldwasser, S. and Sipser, M. (1986). Private coins versus public coins in interactive proof systems. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing, STOC 86*, pages 59–68. ACM, ACM.

Gritzmann, P. and Wills, J. M. (1993). *Handbook of convex geometry*, chapter 3.2, Lattice Points, pages 765–797. Elsevier.

Grotschel, M., Lovász, L., and Schrijver, A. (1993). *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, second edition.

Gruber, P. M. and Lekerkerker, C. G. (1987). *Geometry of Numbers*. North Holland. Previous edition by Lekkerkerker.

Håstad, J. (1988). Dual vectors and lower bounds for the nearest lattice point problem. *Combinatorica*, 8:75–81.

Hoffstein, J., Pipher, J., and Silverman, J. H. (1998). NTRU: A ring based public key cryptosystem. In Buhler, J., editor, *Algorithmic Number Theory (ANTS III)*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288, Portland, OR. Springer.

Johnson, D. S. (1990). *Handbook of Theoretical Computer Science*, volume A (Algorithms and Complexity), chapter 2, A catalog of complexity classes, pages 67–161. Elsevier.

Kaib, M. and Schnorr, C.-P. (1996). The generalized Gauss reduction algorithm. *Journal of Algorithms*, 21(3):565–578.

Kaliski Jr., B. S., editor (1997). *Advances in Cryptology—CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*. Springer-Verlag.

Kannan, R. (1987a). *Annual Reviews of Computer Science*, volume 2, chapter Algorithmic Geometry of numbers, pages 231–267. Annual Review Inc., Palo Alto, California.

Kannan, R. (1987b). Minkowski's convex body theorem and integer programming. *Mathematics of operation research*, 12(3):415–440.

Kannan, R., Lovász, L., and Simonovits, M. (1997). Random walks and $o^*(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*.

Karp, R. M. and Lipton, R. J. (1980). Some connections between nonuniform and uniform complexity classes. In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, pages 28–30, Los Angeles, California. Appeared in journal form as: R.M. Karp and R.J. Lipton, Turing machines that take advice, Enseign. Math. **28** (1982) 191–209.

Klein, P. (2000). Finding the closest lattice vector when it's unusually close. In *Proceedings of the 11th Symposium on Discrete Algorithms*, San Francisco, California. SIAM.

Knuth, D. (1973). *The Art of Computer Programming*, volume 1. Addison–Wesley Publishing Company, Inc., second edition.

Knuth, D. (1981). *The Art of Computer Programming*, volume 2. Addison–Wesley Publishing Company, Inc., second edition.

Lagarias, J. C. (1995). *Handbook of combinatorics*, chapter 19, Point Lattices, pages 919–966. Elsevier.

Lagarias, J. C., Lenstra, Jr., H. W., and Schnorr, C.-P. (1990). Korkine-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4):333–348.

Lagarias, J. C. and Odlyzko, A. M. (1985). Solving low-density subset sum problems. *Journal of the ACM*, 32(1):229–246.

Lenstra, A. K., Lenstra, Jr., H. W., and Lovász, L. (1982). Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:513–534.

Lenstra, H. W. (1983). Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548.

Lobstein, A. (1990). The hardness of solving subset sum with preprocessing. *IEEE Transactions on Information Theory*, 36(4):943–946.

Lovász, L. (1986). *An algorithmic theory of numbers, graphs and convexity*, volume 50 of *CBMS*. SIAM.

Lovász, L. and Scarf, H. (1992). The generalized basis reduction algorithm. *Mathematics of operations research*, 17(3):751–764.

McLoughlin, A. M. (1984). The complexity of computing the covering radius of a code. *IEEE Transactions on Information Theory*, 30:800–804.

Meyer, A. R. and Stockmeyer, L. J. (1972). The equivalence problem for regular expression with squaring requires exponential space. In *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 25–29. IEEE.

Micciancio, D. (1998). *On the hardness of the Shortest Vector Problem*. PhD thesis, Massachusetts Institute of Technology.

Micciancio, D. (2001a). The hardness of the closest vector problem with preprocessing. *IEEE Transactions on Information Theory*, 47(3):1212–1215.

Micciancio, D. (2001b). Improved cryptographic hash functions with worst case / average case connection. Manuscript.

Micciancio, D. (2001c). Improving lattice based cryptosystems using the Hermite Normal Form. In Silverman, J., editor, *Cryptography and Lattices Conference — CaLC'2001*, volume 2146 of *Lecture Notes in Computer Science*, pages 126–145, Providence, Rhode Island. Springer-Verlag.

Micciancio, D. (2001d). The shortest vector problem is NP-hard to approximate to within some constant. *SIAM Journal on Computing*, 30(6):2008–2035. Preliminary version in FOCS 1998.

Micciancio, D. and Warinschi, B. (2001). A linear space algorithm for computing the Hermite Normal Form. In Mourrain, B., editor, *International Symposium on Symbolic and Algebraic Computation*. ACM, ACM. To Appear.

Nguyen, P. (1999). Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In Wiener, M., editor, *Advances in Cryptology—CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*. Springer-Verlag.

Nguyen, P. and Stern, J. (1998). Cryptanalysis of the ajtai-dwork cryptosystem. In Krawczyk, H., editor, *In Advances in Cryptology - Proceedings of CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 223–242, Santa Barbara, California. IACR, Springer Verlag.

Okamoto, T. (1996). On relationships between statistical zero-knowledge proofs. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 649–658, Philadelphia, Pennsylvania. ACM.

Rankin, R. A. (1955). The closest packing of spherical caps in n dimensions. In *Proceedings of the Glasgow Mathematical Association*, volume 2, pages 139–144. Oliver and Boyd.

Raz, R. and Safra, S. (1997). A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 475–484, El Paso, Texas. ACM.

Sahai, A. and Vadhan, S. (1997). A complete promise problem for statistical zero-knowledge. In *38th Annual Symposium on Foundations of Computer Science*, Miami Beach, Florida. IEEE.

Schnorr, C.-P. (1987). A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2-3):201–224.

Schnorr, C.-P. (1993). Factoring integers and computing discrete logarithms via Diophantine approximation. In *Advances in Computational Complexity*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 171–182. AMS. Preliminary version in Eurocrypt'91, Springer-Verlag LNCS 547.

Schnorr, C.-P. (1994). Block reduced lattice bases and successive minima. *Combinatorics, Probability and Computing*, 3:507–522.

Shannon, C. E. (1959). Probability of error for optimal codes in a Gaussian channel. *AT&T Technical Journal*, 38:611–656.

Siegel, C. L. (1989). *Geometry of Numbers*. Springer-Verlag. Notes by B. Friedman, rewritten by Komaravolu Chandrasekharan with the assistance of Rudolf Suter.

Stockmeyer, L. J. (1977). The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22.

Tarokh, V. and Blake, I. F. (1996a). Trellis complexity versus coding gain of lattices I. *IEEE Transactions on Information Theory*, 42(6):1796–1807.

Tarokh, V. and Blake, I. F. (1996b). Trellis complexity versus coding gain of lattices II. *IEEE Transactions on Information Theory*, 42(6):1808–1816.

Vallée, B. (1991). Gauss' algorithm revisited. *Journal of Algorithms*, 12(4):556–572.

van Emde Boas, P. (1981). Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical Report 81-04, Mathematische Instituut, Universiry of Amsterdam.

van Emde Boas, P. (1990). *Handbook of Theoretical Computer Science*, volume A (Algorithms and Complexity), chapter 1, Machine Models and Simulations, pages 1–66. Elsevier.

Wyner, A. D. (1965). Capabilities of bounded discrepancy decoding. *AT&T Technical Journal*, 44:1061–1122.

Yao, A. (1982). Theory and applications of trapdoor functions. In *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, Chicago, IL. IEEE.

About the Authors

Daniele Micciancio got his PhD in computer science from the Massachusetts Institute of Technology in 1998 and he is currently assistant professor in the Computer Science and Engineering department at the University of California, San Diego, where he teaches and performs research in cryptography and computational complexity. His work on the complexity of the shortest vector problem, received a Machtey Award as best student paper at the 39th IEEE Annual Symposium on Foundation of Computer Science (FOCS 98), and a Sprowls Award for best PhD thesis at MIT EECS department in 1999. Micciancio is a 2001 Hellman Fellow, the recipient of an NSF Career Award, and he is regarded as one of the leading experts about the complexity of lattice problems.

Shafi Goldwasser, is the RSA professor of computer science at the Massachusetts Institute of Technology since 1983, and a professor of computer science and applied mathematics at Weizmann Insitute in Israel. She received her PhD in computer science from the university of California at Berkeley in 1983. Her research interests include foundations of cryptography, complexity theory, computational number theory, and fault tolerance of distributed systems. She was awarded the 1993 Gödel prize for her work introducing zero knowledge interactive proofs, and the 2001 Gödel prize for her work on probabilistically checkable proofs and their connections to hardness of approximation. She is the recipient of the 1996 ACM Grace Murray Hopper Award, and the 1998 RSA prize for Outstanding Mathematical Contributions to Cryptography. She is a member of the American Academy of Arts and Sciences.

Index

COMPLEXITY OF LATTICE PROBLEMS: A Cryptographic Perspective is an essential reference for those researching ways in which lattice problems can be used to build cryptographic systems. It will also be of interest to those working in computational complexity, combinatorics, and foundations of cryptography.

The book presents a self-contained overview of the state of the art in the complexity of lattice problems, with particular emphasis on problems that are related to the construction of cryptographic functions. Specific topics covered are: the strongest known inapproximability result for the shortest vector problem; the relations between this and other computational lattice problems; an exposition of how cryptographic functions can be built and proven secure based on worst-case hardness assumptions about lattice problems; and a study of the limits of non-approximability of lattice problems. Some background in complexity theory, but no prior knowledge about lattices, is assumed.

The aim of the authors is to make lattice-based cryptography accessible to a wide audience, ultimately yielding further research and applications. *COMPLEXITY OF LATTICE PROBLEMS: A Cryptographic Perspective* will be valuable to anyone working in this fast-moving field. It serves as an excellent reference, providing insight into some of the most challenging issues being examined today.

ISBN 0-7923-7688-9

9 780792 376880

KLUWER ACADEMIC PUBLISHERS
SECS671
0–7923–7688–9