uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Camelia Karimianpour

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.Sc. (Mathematics)

GRADE / DEGREE

Department of Mathematics and Statistics

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Lattice-Based Cryptosystems

TITRE DE LA THÈSE / TITLE OF THESIS

Dr. M. Nevins

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Dr. A. Miri

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Dr. I. Déchène

Dr. B. Steinberg

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

LATTICE-BASED CRYPTOSYSTEMS

By
Camelia Karimianpour
December 2007

A Thesis
submitted to the School of Graduate Studies and Research
in partial fulfillment of the requirements
for the degree of
Master of Science in Mathematics[1]

[1]The M.Sc. Program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Your file Votre référence
ISBN: 978-0-494-50893-0
Our file Notre référence
ISBN: 978-0-494-50893-0

Abstract

We describe a generalization of the NTRU cryptosystem over rings other than $\mathbb{Z}$ after giving a survey of some of the most famous lattice-based cryptosystems, namely, Ajtai-Dwork, GGH and NTRU. Our generalization, which follows the idea of CTRU and NTRU over Gaussian integers, implies one may extend the NTRU encryption scheme over integral domains up to some constraints. We give details for realizing such extensions over Euclidean domains and denote by ETRU the corresponding class of cryptosystems.

Acknowledgements

I would like to gratefully acknowledge all of the people who assisted me with this thesis. I am sincerely thankful for the invaluable support offered by my supervisors, Dr. Monica Nevins and Dr. Ali Miri. Their patience in offering advice and guiding me for performing this study is highly appreciated. Thanks to all my friends for their assistance and friendship, in particular, thanks to Younes Nouri, Maryam Haghighi, Aziz Khanchi, Masoud Nasari, Abelkarim El-Basraoui, Ratnadha Kolhatkar, Terme Kousha, Andrea Burgess, Shonda Gosselin and all my friends at the Department of Mathematics and Statistics.

Dedication

Dedicated to my family, as no word would be able to express my gratefulness.

Contents

Chapter 1

Introduction

Lattices have been the subject of studies since the 18th century. They have been mostly used as an algorithmic tool to solve a variety of problems in mathematics and computer science. They were widely used in cryptanalysis (e.g., [26]). Recently the capability of lattices in the construction of cryptographic systems has been demonstrated in several works. These cryptosystems are based on lattice problems which are known to be hard including the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP).

In a short period of time starting in 1996, several public key cryptosystems were presented. Some of them are largely theoretical (that is, impractical) but have provable security while others, which are efficient in terms of implementation and have been proposed as an alternative to other known public key cryptosystems, do not enjoy a rigorous proof of security. In all cases, the resulting cryptosystems are 'imperfect', in the sense that there is a small probability of error in decryption.

In this thesis, our first goal is to analyze and compare three of the main lattice-based cryptosystems: Ajtai-Dwork, GGH and NTRU.

In 1996, Ajtai [1] presented the first lattice-based cryptographic construction by introducing a family of trapdoor functions based on SVP. In 1997, Ajtai and Dwork [2] followed up on this by constructing a public key cryptosystem whose security is based on the worst-case hardness of a special case of SVP, known as unique-SVP. While this cryptosystem is provably secure, it is not practical from the point of view of

implementation due to the size of the public key [40].

At almost the same time, Goldreich, Goldwasser and Halevi (GGH) presented their work on public key cryptosystems based on lattice reduction problems [18]. GGH was also shown to be impractical shortly after it was proposed for the same reason as Ajtai-Dwork [38]. However, it demonstrates a clear method of using lattices to built a one-way trapdoor function.

In 1998, the only efficient lattice-based cryptosystem known so far was introduced by Hoffstein, Pipher and Silverman called NTRU [19]. NTRU, which was originally presented as a cryptosystem over polynomials with coefficients in $\mathbb{Z}$, reveals the connection to the lattices via its particular decryption scheme.

All of the lattice based cryptosystems developed so far are imperfect in the sense that there is a small probability of decryption failure. The possibility of decryption failure in NTRU has been a concern from the security point of view [22]. Since it was proposed, several attacks have been developed against NTRU [11, 23, 13, 33, 43]. Among those, the last three take advantage of the weakness of NTRU due to the decryption failure. On the other hand, the designers of the cryptosystem have been improving it to render NTRU immune from these attacks [42].

The success of NTRU as an efficient and practical cryptosystem has spawned several additional works. In 2002, an analogue of NTRU proposed in [14] called CTRU in which the base ring of NTRU, $\mathbb{Z}$, was replaced by a polynomial ring. Although this cryptosystem was broken soon after it was proposed [25], it suggested the idea of generalizing NTRU over other rings. This idea was followed in [25] where another analogue of NTRU was suggested in which $\mathbb{Z}$ is replaced by Gaussian integers.

The second goal of this thesis is to investigate the question of which rings are suitable for use in NTRU-like cryptosystems. We go on to develop NTRU over several rings and investigate them via their lattice-like properties.

Besides the three mentioned cryptosystems, there have been other works on the subject. Of particular interest, for example, are the papers by Regev [44, 45]. The former's security, like [2], is based on the hardness of unique-SVP and also is not too efficient. The security of [45] is based on the worst-case quantum hardness of the SVP.

The recent interest in lattices and their applications in cryptography is due to several reasons. First of all, there are not many alternatives to traditional cryptosystems, such as RSA and ECC, which are based on the hardness of factoring a composite number or the discrete logarithm problem (DLP). Thus should an efficient algorithm for these problems be discovered, or (given that an efficient quantum algorithm for the mentioned problems already exists) a sufficiently powerful quantum computer be built, then these cryptosystems and all applications based on them would collapse. Secondly, computations in lattice-based cryptosystems are easy, which is promising from the implementation point of view.

In this thesis, we first give a survey of the most famous lattice based cryptosystems mentioned above, that is, Ajtai-Dwork, GGH and NTRU. Then, inspired by [14, 25], we investigate the necessary conditions for a ring to sit in place of the ring of integers in NTRU. We describe CTRU as an example to illustrate and motivate our generalization. Furthermore, we investigate the possibility of replacing $\mathbb{Z}$ in NTRU by $\mathbb{Z}[\sqrt{-2}]$, $\mathbb{Z}[\zeta_3]$ and $\mathbb{Z}[\zeta_5]$.

In the survey part of the thesis, we have omitted several of the proofs of main results, referring the readers instead to the original sources. On the other hand, we have proven some details which were not included in the literature where we thought these would aid in the reader's understanding. However, in the second part of the thesis, we justify all of our steps to make it more useful as a source for the further work in generalization of NTRU.

The organization of the thesis is as follows. In Chapter 2 we set our notation and present the necessary background in complexity theory and public key cryptography. We then go on to present lattices and to describe the hard computational problems in lattices. Chapters 3, 4 and 5 are devoted to describing the Ajtai-Dwork, GGH and NTRU cryptosystems respectively with particular emphasis on drawing out their similarities and differences. In particular, in Chapter 5, we rephrase NTRU in the framework of trapdoor functions to better reveal its similarities with GGH. We also add the proofs of many intermediate results. We go on to interpret their polynomial-based algorithm in terms of lattices to show its lattice-based structure.

In Chapter 6 we begin our generalization of NTRU to other rings. In Section 6.1

we determine some sufficient conditions for a ring to serve as an NTRU base ring. We illustrate this by describing CTRU in Section 6.2, following [14]. In Chapter 7, we apply our analysis from Chapter 6 to develop NTRU over three Euclidean domains: $\mathbb{Z}[\sqrt{-2}]$, $\mathbb{Z}[\zeta_3]$ and $\mathbb{Z}[\zeta_5]$, with a view towards the general case. In the last chapter, we present a comparison of the cryptosystems given in Chapters 3, 4 and 5 as well as a conclusion about the NTRU extensions discussed in Chapter 6 and 7. Moreover, we suggest several avenues of future work on this topic.

Chapter 2

Background and Preliminaries

In this chapter, we briefly explain public key cryptography after setting our notation. Moreover, we provide the background needed on lattices and introduce their basic properties. Finally, we present hard lattice problems that are the basis for several cryptosystems. The material of this chapter is mainly taken from [36, 41]. Except for Section 2.4, the proofs which are included here are my own.

2.1 Notation

In this section, we are setting our notational conventions as well as giving some basic definitions for the concepts used later on. We denote vectors by small bold letters and matrices by capital bold letters. We denote the vector of all 1's by $\mathbf{1}$, identity matrix by $\mathbf{I}$ and the matrix of all 1's by $\mathbf{J}$. By $\mathbf{x}\cdot\mathbf{y}$ we mean the dot product of two vectors $\mathbf{x}$ and $\mathbf{y}$, and $\|\mathbf{x}\| = \sqrt{\mathbf{x}\cdot\mathbf{x}}$ denotes the norm of $\mathbf{x}$. By $\lfloor x \rceil$ we mean the closest integer to the real number x, and $|x|$ denotes the absolute value of x. We denote the set of all $n \times n$ matrices with integer entries by $M(\mathbb{Z}, n)$. By $\mathcal{B}(0, r)$ we mean the set:

$$\{\mathbf{x} \in \mathbb{R}^n, \quad \| \mathbf{x} \| < r\},$$

that is, the n-dimensional open ball of radius r centered at the origin.

Definition 2.1.1. Let $p \geq 1$ be a real number. The ℓ_p norm of the vector $\mathbf{x} = (x_1, \ldots, x_n)$ is defined by

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

The infinity norm ℓ_∞ is defined by

$$\|\mathbf{x}\|_\infty = \max\{|x_1|, \ldots, |x_n|\}.$$

We write $\|\mathbf{x}\| = \|\mathbf{x}\|_2$ unless otherwise stated.

Definition 2.1.2. For any sequence of m-dimensional vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ the corresponding *Gram-Schmidt* orthogonalized vectors $\mathbf{b}_1^\star, \cdots, \mathbf{b}_n^\star$ are defined by

$$\begin{aligned} \mathbf{b}_1^\star &= \mathbf{b}_1 \\ \mathbf{b}_i^\star &= \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^\star \quad \text{for} \quad i > 1 \end{aligned}$$

where

$$\mu_{i,j} = \frac{\mathbf{b}_i \cdot \mathbf{b}_j^\star}{\mathbf{b}_i^\star \cdot \mathbf{b}_j^\star}.$$

Complexity Theory

The following are some basic concepts in complexity theory [32]. All definitions are given under the assumption of having a *well-defined* algorithm, that is, an algorithm that produces an *output* for a given *input*. Analyzing an algorithm means determining the amount of time and storage required to execute that algorithm.

The *size* of the input usually means the number of bits required to code the input of an algorithm. However, depending on the algorithm, one might use other units to express the input size.

It is important to use the same notion for size while comparing two or more algorithms. For example if two algorithm are being compared whose both input is a $n \times n$ matrix, the size of input and be expressed by the number of entries in the matrix.

Definition 2.1.3. Given an input, the number of steps an algorithm takes before producing the output is called the *running time* of the algorithm.

The notion of 'step' may vary from one algorithm to another. But it is crucial to use the same notion while comparing two or more algorithms. Since the exact running time of an algorithm depends on several factors, is not easy to calculate. Therefore, one studies the behavior of the running time as the input size goes to infinity. This is called the *asymptotic* running time analysis.

Definition 2.1.4. Suppose f and g are functions from $\mathbb{N}$ to $\mathbb{R}^+$. We say, $f(n) = O(g(n))$ if there exist a positive constant c and positive integer n_0 such that

$$0 \leq f(n) \leq cg(n) \quad \text{for all} \quad n \geq n_0$$

The notation of O is used to describe the asymptotic running time of an algorithm.

Definition 2.1.5. The *worst-case (average-case)* running time of an algorithm is the maximum (average) number of steps taken on any instance of input of size n. This is expressed as a function of n.

Definition 2.1.6. If the running time of an algorithm A is bounded by a polynomial in its input size n, that is, its worst case running time is $O(n^k)$ for some constant k, we say A is a *polynomial-time algorithm.*

Definition 2.1.7. The problems whose answers are YES or NO are called *decision problems.*

Example 2.1.8 (The Composite Problem). Given a positive integer n, the problem of deciding whether n is composite is a decision problem.

Definition 2.1.9. The complexity class P is the set of all decision problems that are solvable by a polynomial time algorithm.

Definition 2.1.10. The complexity class NP is the set of all decision problems for which, given a piece of information called a *certificate*, a YES answer can be verified by a polynomial time algorithm.

Example 2.1.11 (Hamilton-Cycle Problem). Given a graph G, the problem of deciding whether G contains a Hamilton cycle, that is, a cycle going from one vertex of G, through all the other vertices of G exactly once, is in NP. That is because, a list of vertices that make the cycle can serve as a certificate which allows verification of yes answer in polynomial time.

Obviously, $P \subseteq NP$ while it is an outstanding open problem whether P equals NP [28].

Definition 2.1.12. A decision problem L is said to be *NP-complete* if $L \in NP$ and every other problem in NP can be reduced to L via a polynomial time algorithm.

NP-complete problems are the hardest problems in NP because the existence of a polynomial time algorithm to solve them implies the existence of a polynomial algorithm to solve all problems in NP, which in turn implies $NP = P$.

More generally, a problem L which is not necessarily a decision problem is called *NP-hard* if there exists some NP-complete problem that reduces polynomially to L. Therefore, NP-complete problems are also NP-hard.

Definition 2.1.13. An algorithm that follows the same sequence of steps every time it is run on a fixed input is called a *deterministic* algorithm. On the contrary, an algorithm may take random decisions which lead to having different sequences of steps when run over the same input more than once. Such an algorithm is called a *randomized or probabilistic* algorithm.

We say a problem is NP-complete under deterministic (randomized) reduction, if the polynomial time reduction algorithm in definition 2.1.12 is deterministic (randomized).

Definition 2.1.14. By *computationally easy* problems we mean those problems which lie in P. On the contrary, *computationally infeasible* problems ideally lie in NP. However, there are some problems to which we refer as computationally infeasible without having the proof of NP-hardness.

An example of such problems is the factoring problem. In such cases, the fact that there is no known polynomial time algorithm allows us to use the term computationally infeasible. We may also simply use the term difficult or hard for the same concept.

2.2 Public Key Cryptosystems

A *cryptosystem* consists of an injective transformation f from a finite set $\mathcal{P}$ of plaintext messages to a finite set $\mathcal{C}$ of ciphertext messages. In a cryptosystem, the keys are the pieces of information which allow one to *encrypt* or *decrypt* the message. The cryptosystems are divided into two categories of *symmetrical* or private key cryptosystems and public key cryptosystems. In the former, the transformation f and its inverse are both computable by knowing one piece of information called the *symmetrical key*. That is, the person who is able to encrypt a message to a ciphertext by knowing the key is also able to decrypt the ciphertext. That requires sharing the key between two ends of the communication channel. However, in the latter, the function f is chosen having a special property that allows communication without having to share a common key. We will shortly explain that property.

Definition 2.2.1. A function g from a set A to a set B is called a *one-way function* if $g(x)$ is easy to compute for all $x \in A$ but for almost all $y \in \text{Img}(g)$ it is computationally infeasible to find an element $x \in g^{-1}(y)$.

Definition 2.2.2. A *one-way trapdoor* function is a one-way function $g : A \mapsto B$ such that given some extra information t, called *trapdoor information*, there exist a computationally easy function Φ such that $\Phi(t, g, y) \in g^{-1}(y)$.

In public key cryptosystems, the transformation f is based on a one-way trapdoor function which can be computed in polynomial time. The information needed to compute $f(x)$ is called the *public key* K_E which as it is apparent from its name is public and is used to encrypt the message. That is, knowing K_E, the encryption is done in polynomial time. However, the trapdoor information, which is often called the *private key*, is private and is used to decrypt the message.

Trapdoor functions are based on hard mathematical problems. Note that since we are measuring the hardness asymptotically, the running time of a hard problem which has a known or expected exponential time algorithm might not be that long for small input sizes. That is why for actual implementations the bigger the input size is the more secure the system would be. The *security parameter* is a parameter that determines the input size of the trapdoor function. For explaining a one-way trapdoor function explicitly, one needs to give the four following polynomial time (possibly probabilistic) algorithms: Generate, Sample, Evaluate and Invert [18].

- Generate: This algorithm takes the security parameter as input and outputs (f, t), where f is a one-way trapdoor function with its associated domain D_f and range R_f, and t is the trapdoor information.
- Sample: This algorithm takes f and outputs $x \in D_f$.
- Evaluate: This algorithm takes f and x and returns $f(x)$.
- Invert: This algorithm takes f, t and $y \in R_f$, and returns $x \in D_f$ such that $f(x) = y$.

In addition, Evaluate is a polynomial time algorithm, and for any probabilistic polynomial time algorithm A, the probability that A succeeds in inverting f without knowing the trapdoor information t in negligible. Note that this is why we said f is one-way trapdoor in the Generate step. In some cases, for example NTRU, as we will see further in this thesis, there is a failure probability for the Invert algorithm. The cryptosystems based on such trapdoor functions are regarded as imperfect cryptosystems.

The main advantage of public key cryptosystems to private key cryptosystems is the fact that in public key cryptosystems two parties do not need to share a key, so there is no need to have a secure channel (which is the case for private key cryptosystems). On the other hand, unfortunately, public key cryptosystems are relatively slow in comparison with symmetrical cryptosystems. That is the reason they have been used as a key exchange method for symmetrical cryptosystems. There are more

recent public key cryptosystems based on lattice problems. Among those, NTRU is the one with a promising speed.

2.3 Lattices

In this section, we follow the presentation of lattices given in [36]. Except as is noted, the proofs included here are my own.

Definition 2.3.1. A *lattice* in $\mathbb{R}^m$ is a set

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

of all integer 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 respectively called the *rank* and *dimension* of the lattice. A lattice is called *full rank* if $n = m$. A lattice basis can be represented by a $n \times m$ matrix $\mathbf{B}$ having basis vectors as its rows,

$$\mathbf{B} = \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_n \end{bmatrix}.$$

We refer to $\mathbf{B}$ as a generator matrix of $\mathcal{L}$. So (1) can be written as

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{xB}, \mathbf{x} \in \mathbb{Z}^n\}. \tag{2}$$

Note that the same lattice may have many different bases. The vector space generated by the rows of $\mathbf{B}$ is denoted by

$$span(\mathbf{B}) = \{\mathbf{xB}, \mathbf{x} \in \mathbb{R}^n\}.$$

While any set of n linearly independent vectors in $\mathcal{L}(\mathbf{B})$ is a basis of $span(\mathbf{B})$, it is not necessarily a basis for $\mathcal{L}(\mathbf{B})$.

We say that $\mathbf{B}$ and $\mathbf{B}'$ are equivalent if $\mathcal{L}(\mathbf{B}) = \mathcal{L}(\mathbf{B}')$. The following proposition gives a condition for equivalency of two generator matrices.

Definition 2.3.2. A *unimodular* matrix is a square integer matrix with determinant ± 1.

Proposition 2.3.3. *Two $n \times m$ generator matrices $\boldsymbol{B}$ and $\boldsymbol{B'}$ are equivalent if and only if there exists a unimodular matrix $\boldsymbol{U} \in M(\mathbb{Z}, n)$ such that $\boldsymbol{B'} = \boldsymbol{UB}$.*

Proof. Suppose $\mathbf{B}$ and $\mathbf{B}'$ are equivalent. Both matrices generate the same lattice, so the rows of $\mathbf{B}$ ($\mathbf{B}'$) are integer linear combinations of the rows of $\mathbf{B}'$ ($\mathbf{B}$). Thus, there exist $\mathbf{U}$ and $\mathbf{V}$ in $M(\mathbb{Z}, n)$ such that $\mathbf{B} = \mathbf{UB}'$ and $\mathbf{B}' = \mathbf{VB}$. From this we get

$$\begin{aligned} \mathbf{UV} &= \mathbf{I} \\ \det(\mathbf{UV}) &= \det(\mathbf{I}) \\ \det(\mathbf{U})\det(\mathbf{V}) &= 1 \end{aligned} \tag{3}$$

So, $\mathbf{U} = \mathbf{V}^{-1}$ and since they are both integer matrices so is their determinants. Thus, (3) implies that they are unimodular. Conversely, suppose that a $n \times n$ unimodular matrix $\mathbf{U}$ exists such that $\mathbf{B}' = \mathbf{UB}$. That means rows of $\mathbf{B}'$ are integer linear combinations of rows of $\mathbf{B}$, so $\mathcal{L}(\mathbf{B}') \subseteq \mathcal{L}(\mathbf{B})$. Multiplying the both sides by $\mathbf{V} := \mathbf{U}^{-1}$, which is guaranteed to be integer matrix by unimodularity of $\mathbf{U}$, we get $\mathbf{B} = \mathbf{VB}'$. That means rows of $\mathbf{B}$ are integer linear combinations of rows of $\mathbf{B}'$, so $\mathcal{L}(\mathbf{B}) \subseteq \mathcal{L}(\mathbf{B}')$. Hence $\mathcal{L}(\mathbf{B}) = \mathcal{L}(\mathbf{B}')$. □

Example 2.3.4. Let

$$\mathbf{B}_1 = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix} \quad \text{and} \quad \mathbf{B}_2 = \begin{pmatrix} 3 & 5 \\ 1 & 1 \end{pmatrix}.$$

It is easy to see that the rows of both matrices are linearly independent. So $span(\mathbf{B}_1) = span(\mathbf{B}_2)$. And the matrix $\mathbf{U} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ satisfies $\mathbf{UB}_1 = \mathbf{B}_2$. But since $\mathbf{U}$ is not unimodular, these two matrices generate different lattices. In fact, $\mathcal{L}(\mathbf{B}_2)$ is a *sublattice* of $\mathcal{L}(\mathbf{B}_1)$ (i.e., $\mathcal{L}(\mathbf{B}_2) \subset \mathcal{L}(\mathbf{B}_1)$).

One could define the set $\mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ without the assumption that the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are linearly independent. One might ask if such an arbitrary set of vectors makes a lattice. By basic linear algebra, such a thing happens in the case of vector

spaces. That is, given an arbitrary set of vectors, there is always a (possibly smaller) set of linearly independent vectors which spans the same vector space. However, this is not generally true for lattices, as the following example shows.

Example 2.3.5. Let 1 and $\sqrt{2}$ be two linearly dependent vectors in $\mathbb{R}$. The 'lattice' generated by them is $\mathcal{L} = \{x_1 + x_2\sqrt{2} : x_i \in \mathbb{Z}, i = 1, 2\}$. We claim there is no $b \in \mathbb{R}$ such that $\mathcal{L} = \mathcal{L}(b)$. For any $b \in \mathbb{R}$ we have

$$\mathcal{L}(b) = \{xb, x \in \mathbb{Z}\}.$$

It cannot contain the elements 1 and $\sqrt{2}$ because the quotient of any two non-zero elements in $\mathcal{L}(b)$ is rational but $\frac{\sqrt{2}}{1}$ is not rational. This shows that

$$\{x_1 + x_2\sqrt{2}, x_i \in \mathbb{Z}, i = 1, 2\}$$

is not a lattice in $\mathbb{R}$.

Definition 2.3.6. Suppose $\mathbf{B}$ is a matrix formed by n linearly independent row vectors. The half open parallelepiped is defined to be

$$\mathcal{P}(\mathbf{B}) = \{\mathbf{xB}, \quad 0 \leq x_i < 1 \forall i\} \tag{4}$$

Lemma 2.3.7. *Let $\boldsymbol{B}$ be a matrix formed by n linearly independent row vectors, then $\mathcal{P}(\boldsymbol{B}) \cap \mathcal{L}(\boldsymbol{B}) = \{0\}$.*

Proof. Suppose there exists a nonzero vector $\mathbf{x}$ of $\mathcal{L}(\mathbf{B})$ in $\mathcal{P}(\mathbf{B})$. Since $\mathbf{x} \in \mathcal{L}(\mathbf{B})$, it can be written as

$$\mathbf{x} = \sum_i z_i \mathbf{b}_i,$$

where the z_i's are integers. On the other hand, since $\mathbf{x} \in \mathcal{P}(\mathbf{B})$ it can be written as

$$\mathbf{x} = \sum_i a_i \mathbf{b}_i,$$

where $0 \leq a_i < 1$. But since these linear combinations both represent a vector in the vector space $\mathbb{R}^n$ and the $\mathbf{b}_i$'s form a basis for $\mathbb{R}^n$, it contradicts the unique representation of an element in $\mathbb{R}^n$ with respect to a basis. □

Definition 2.3.8. The *length* l of a lattice basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is defined to be the length of the longest basis vector. That is

$$l = \max_i \|\mathbf{b}_i\|.$$

Definition 2.3.9. The *determinant* of a lattice is defined to be

$$\det(\mathcal{L}(\mathbf{B})) = \sqrt{\det(\mathbf{B}^T\mathbf{B})} \tag{5}$$

If the lattice has full rank, the determinant will be the absolute value of $\det(\mathbf{B})$, since

$$\begin{aligned}
\det(\mathcal{L}(\mathbf{B})) &= \sqrt{\det(\mathbf{B}^T\mathbf{B})} \\
&= \sqrt{\det(\mathbf{B}^T)\det(\mathbf{B})} \\
&= \sqrt{\det(\mathbf{B})^2} \\
&= |\det(\mathbf{B})|.
\end{aligned}$$

The determinant of a full rank lattice generated by $\mathbf{B}$ can be thus defined alternatively as the n-dimensional volume of $\mathcal{P}(\mathbf{B})$. This volume can be computed by constructing the Gram-Schmidt orthogonalized basis and computing the product of their norm. It can be shown that these two definitions are equivalent. See [36, Section 1.1] for more details.

Proposition 2.3.10. *If $\boldsymbol{B}$ and $\boldsymbol{B}'$ are equivalent, then* $\det\mathcal{L}(\boldsymbol{B}) = \det\mathcal{L}(\boldsymbol{B}')$.

Proof. By Proposition 2.3.3 there exist a $\mathbf{U} \in M(\mathbb{Z}, n)$ with $|\det(\mathbf{U})| = 1$ such that $\mathbf{B} = \mathbf{U}\mathbf{B}'$. We have

$$\begin{aligned}
\det(\mathcal{L}(\mathbf{B}')) &= \sqrt{\det(\mathbf{B}'\mathbf{B}'^T)} \\
&= \sqrt{\det(\mathbf{U}\mathbf{B}'\mathbf{B}'^T\mathbf{U}^T)} \\
&= \sqrt{\det(\mathbf{U}\mathbf{B}')(\mathbf{U}\mathbf{B}')^T} \\
&= \sqrt{\det(\mathbf{B}\mathbf{B}^T)} \\
&= \det(\mathcal{L}(\mathbf{B})).
\end{aligned}$$

□

Definition 2.3.11. Let $\mathcal{L}$ be an n-dimensional lattice. The *dual* of $\mathcal{L}$, denoted by $\mathcal{L}^*$, is defined as

$$\mathcal{L}^* = \{\mathbf{x} \in \mathbb{R}^n \quad | \quad \mathbf{x} \cdot \mathbf{y} \in \mathbb{Z} \text{ for all } \mathbf{y} \in \mathcal{L}\}.$$

Lemma 2.3.12. *If $\mathcal{L}$ has the generator matrix $\boldsymbol{B}$, then $\mathcal{L}^* = \mathcal{L}((\boldsymbol{B}^{-1})^T)$.*

Proof. If $\mathcal{L}$ has the generator matrix $\mathbf{B}$, the dual lattice of $\mathcal{L}$ is generated by $\mathbf{B}^*$ where the ith row $\mathbf{b}_i^*$ of $\mathbf{B}^*$ satisfies

$$\mathbf{b}_i^* \cdot \mathbf{b}_j = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j \end{cases}.$$

Since the columns of $\mathbf{B}^{-1}$ satisfy the condition above, $\mathbf{B}^* = (\mathbf{B}^{-1})^T$. □

Definition 2.3.13. Let $\mathcal{L}$ be a n-dimensional lattice. The *successive minima* associated to $\mathcal{L}$ are constants $\lambda_1, \ldots, \lambda_n$ defined by

$$\lambda_i(\mathcal{L}) = \inf\{r : \quad \dim(span(\mathcal{L}(\mathbf{B}) \cap \mathcal{B}(0, r))) \geq i\} \tag{6}$$

In other words, λ_i is the radius of the smallest sphere centered in the origin containing i linearly independent lattice vectors.

Now we will show that any lattice contains a nonzero vector of minimal length. In fact one can prove that in a given lattice of dimension n, there exist n linearly independent lattice vectors achieving the successive minima length [36]. By definition, λ_1 is the radius of the smallest sphere containing a nonzero lattice vector. That means if the lattice contains a vector of length λ_1, that vector is a shortest nonzero vector in the lattice. Since the basis vectors of a lattice are linearly independent, assuming that they are ordered in norm we have

$$\max_{j=1,\ldots,i}(\| \mathbf{b}_j \|) \geq \lambda_i \tag{7}$$

Remark 2.3.14. Note that the definition of successive minima depends on the norm we are using. Different norms give us different values of λ_i's. Throughout the text, we will refer to the successive minima with respect to Euclidian norm unless otherwise is stated.

We will use the following theorem, which is stated without proof, later on to describe the LLL algorithm. The proof can be found in [36, Theorem 1.1].

Theorem 2.3.15. *Let $\boldsymbol{B}$ and $\boldsymbol{B}^\star$ be a basis and its corresponding Gram-Schmidt orthogonalization for the lattice $\mathcal{L}$ respectively. Then the first minimum of the lattice satisfies*

$$\lambda_1 \geq min_j \parallel \boldsymbol{b}_j^\star \parallel > 0.$$

Proposition 2.3.16. *Let $\mathcal{L}(\boldsymbol{B})$ be a n-dimensional lattice. There exists a $\boldsymbol{w} \in \mathcal{L}(\boldsymbol{B})$ such that $\parallel \boldsymbol{w} \parallel = \lambda_1$.*

Proof. By Lemma 2.3.7, $\mathcal{P}(\mathbf{B})$ only contains the zero lattice point. Note that since the matrix of multiplication a row of $\mathbf{B}$ by -1 is unimodular, by replacing each $\mathbf{b}_i$ by its negative we still get a basis for the same lattice whose parallelepiped only contains the lattice point zero. Consider an open ball U centered at the origin in the union of these parallelepipeds. This U intersects the lattice only in origin. Therefore, the lattice is discrete. Hence any closed ball contains only finitely many lattice points. That guarantees the existence of an element with minimum norm which is λ_1.

□

We will shortly see that the product of any number of successive minima, in particular λ_1, can be upper bounded by Minkowski's theorem also known as the convex body theorem. The bound is based on the following so-called Blichfeld theorem. See [36, Theorem 1.3] for the proofs.

Theorem 2.3.17. *Let $\mathcal{L}(\boldsymbol{B})$ be a lattice and S be a set such that $S \subseteq span(\boldsymbol{B})$. If S has volume $vol(S) > \det \mathcal{L}(\boldsymbol{B})$, then there exist two distinct point $\boldsymbol{z}_1, \boldsymbol{z}_2 \in S$ such that $\boldsymbol{z}_1 - \boldsymbol{z}_2 \in \mathcal{L}(\boldsymbol{B})$.*

Definition 2.3.18. A set $S \in \mathbb{R}^n$ is *convex* if all points on the straight line between two points of the set also belong to the set. That is,

$$x, y \in S \Rightarrow tx + (1-t)y \in S$$

for all $0 \leq t \leq 1$.

Theorem 2.3.19. *(Convex Body Theorem) For any lattice $\mathcal{L}$ of rank n and any convex set $S \subset span(\mathcal{L})$ symmetric about the origin, if $vol(S) > 2^n \det(\mathcal{L})$, then S contains a nonzero lattice point $\boldsymbol{v} \in S \cap \mathcal{L} \setminus \{0\}$.*

Now let S be $\mathcal{B}(0, \sqrt{n}\det(\mathcal{L})^{1/n}) \cap span(\mathcal{L})$. Notice that S contains an n-dimensional hypercube with edge of length $2\det(\mathcal{L})^{1/n}$ so the volume of S is strictly greater than $2^n \det(\mathcal{L})$. Applying the Convex Body Theorem, there exists a non-zero lattice vector in S. Since that vector lies in the ball $\mathcal{B}(0, \sqrt{n}\det(\mathcal{L})^{1/n})$, it is strictly smaller than $\sqrt{n}\det(\mathcal{L})^{1/n}$. That gives us an upper bound for λ_1:

$$\lambda_1 < \sqrt{n}\det(\mathcal{L})^{1/n}. \tag{8}$$

The upper bound for the product of any number of successive minima is given by Minkowski's second theorem.

Theorem 2.3.20. *(Minkowski's Second Theorem) For any rank n lattice $\mathcal{L}(\boldsymbol{B})$, the successive minima $\lambda_1, \lambda_2, \ldots, \lambda_n$ satisfy*

$$\left(\prod_{i=1}^{n} \lambda_i\right)^{1/n} < \sqrt{n}\det(\boldsymbol{B})^{1/n}. \tag{9}$$

As we saw, Minkowski's first theorem gives a bound for shortest vector in the lattice. However, in general the shortest vector can be much smaller than $\sqrt{n}\det(\mathcal{L})^{1/n}$. Here is an example of a lattice whose λ_1 is far smaller than Minkowski's bound [36, section 1.2].

Example 2.3.21. Let $\mathcal{L}(\mathbf{B})$ be a 2-dimensional lattice generated by

$$\mathbf{B} = \begin{pmatrix} \epsilon & 0 \\ 0 & (1/\epsilon) \end{pmatrix}$$

Since the determinant of the matrix is 1, by Minkowski's theorem we have

$$\lambda_1 < \sqrt{2};$$

but $\lambda_1 = \epsilon$ which can be arbitrarily small.

Moreover, the proof of the theorem does not give any constructive method to find the vectors smaller than $\sqrt{n}\det(\mathbf{B})^{1/n}$ in the lattice.

2.4 Computational Problems in Lattices

In this section, we consider only sublattices of $\mathbb{Z}^n$ since that is the class of lattices we will use in further chapters.

2.4.1 Shortest Vector Problem

The main computational problem associated with lattices is to find a shortest vector in a given lattice.

Definition 2.4.1. (Shortest Vector Problem, SVP) Given a lattice $\mathcal{L}$, find a nonzero lattice vector $\mathbf{l} \in \mathcal{L}$ such that $\| \mathbf{l} \| \leq \| \mathbf{l}' \|$ for all nonzero $\mathbf{l}' \in \mathcal{L}$.

Since, a lattice is usually represented by its basis, SVP can be stated as: Given a basis $\mathbf{B} \in M_{n\times m}(\mathbb{Z})$, find a non-zero lattice vector $\mathbf{xB}$ where $\mathbf{x} \in \mathbb{Z}^n \setminus \{0\}$, such that $\| \mathbf{xB} \| \leq \| \mathbf{yB} \|$ for any other $\mathbf{y} \in \mathbb{Z}^n \setminus \{0\}$.

Remark 2.4.2. The SVP is well defined due to the Proposition 2.3.16 insuring the existence of a lattice vector of minimal norm.

Remark 2.4.3. In the definition above, $\| \cdot \|$ refers to any norm. Therefore, the solution of the SVP depends on the norm we are using. However, we are mainly interested in the case of the Euclidean norm. Here is an example in which norm dependency of the SVP is demonstrated [36, Section 1.2].

Example 2.4.4. Consider the lattice generated by the matrix

$$\mathbf{B} = \begin{pmatrix} 1 & 1 \\ 0 & 2 \end{pmatrix}.$$

$\mathcal{L}(\mathbf{B})$ can be equivalently defined as

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{v} \in \mathbb{Z}^2, v_1 + v_2 \equiv 0 \mod 2\}.$$

The vector $(0, 2)$ is a shortest vector with respect to ℓ_1 with corresponding $\lambda_1 = 2$, while, the vector $(1, 1)$ is strictly shorter with respect to the ℓ_2 norm. So $\lambda_1 = \sqrt{2}$ with respect to the ℓ_2 norm.

Lemma 2.4.5. *If $\boldsymbol{B}$ is a matrix with orthogonal rows and $\mathcal{L} = \mathcal{L}(\boldsymbol{B})$, then the shortest row of $\boldsymbol{B}$, with respect to the ℓ_2 norm, is a shortest non-zero vector of $\mathcal{L}(\boldsymbol{B})$.*

Proof. Suppose that the rows of $\mathbf{B}$ are ordered in increasing order of their norm. A shortest non-zero vector $\mathbf{l}$ can be written as $\sum_i z_i \mathbf{b}_i$ for some integers z_i's. Therefore, we have

$$\begin{aligned} \|\mathbf{l}\|^2 &= \mathbf{l} \cdot \mathbf{l} \\ &= \sum_{i,j} z_i z_j \mathbf{b}_i \cdot \mathbf{b}_j, \end{aligned}$$

Since the $\mathbf{b}_i$s are pairwise orthogonal, we have

$$\begin{aligned} \|\mathbf{l}\|^2 &= \sum_i z_i^2 \mathbf{b}_i \cdot \mathbf{b}_i \\ &= \sum_i z_i^2 \|\mathbf{b}_i\|^2. \end{aligned}$$

Since $\mathbf{l}$ is a shortest vector, in particular we have

$$\|\mathbf{l}\| \leq \|\mathbf{b}_1\|.$$

So,

$$(z_1^2 - 1)\|\mathbf{b}_1\|^2 + \sum_{i=2}^{n} z_i^2 \|\mathbf{b}_i\|^2 \leq 0$$

Thus, it follows that $z_1 = \pm 1$ and the rest of z_i s are zero. □

Thus, for a lattice with a given orthogonal basis, the SVP is trivial. But, lattices typically do not have an orthogonal basis. There is currently no known polynomial time algorithm for finding the shortest vector in a n-dimensional lattice for a given $n > 2$ [36]. In fact, it is NP-hard under randomized reduction [4]. However, such an algorithm exists for 2-dimensional lattices; it is known as the Gauss algorithm.

The Shortest Vector Problem in Two Dimensions

The original version by Gauss (1801), in Euclidean norm, was generalized by Kaib and Schnorr to arbitrary norms [36].

Definition 2.4.6. Let $\{\mathbf{a}, \mathbf{b}\}$ be a basis of a 2-dimensional lattice. This basis is *reduced* if

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

That is, the basis vectors are shorter or equal to both their sum and their difference.

The geometric interpretation of a reduced basis is that the diagonals of the parallelogram determined by the basis are longer than the sides. The proof of the following Theorem is taken from [36, Chapter 2].

Theorem 2.4.7. *Let* $\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}$ *be a generator matrix of a lattice* $\mathcal{L}$*, then* $\{\|\mathbf{a}\|, \|\mathbf{b}\|\} = \{\lambda_1, \lambda_2\}$ *if and only if* $\{\mathbf{a}, \mathbf{b}\}$ *is reduced.*

To prove this theorem, we need the following lemma.

Lemma 2.4.8. *Consider three vectors on a line,* $\mathbf{x}, \mathbf{x}+\mathbf{y}$ *and* $\mathbf{x}+\alpha\mathbf{y}$*, where* $\mathbf{y} \neq 0$ *and* $\alpha > 1$*. For any norm* $\|\cdot\|$*, if* $\|\mathbf{x}\| \leq \|\mathbf{x}+\mathbf{y}\|$*, then* $\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}+\alpha\mathbf{y}\|$.

Proof. Let $\delta = 1/\alpha$, then

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

By the triangle inequality,

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

Also, from the hypothesis $\|\mathbf{x}\| \leq \|\mathbf{x}+\mathbf{y}\|$ we get

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

Hence,

$$\begin{aligned} \|\mathbf{x}+\mathbf{y}\| &\leq (1-\delta)\|\mathbf{x}+\mathbf{y}\| + \delta\|\mathbf{x}+\alpha\mathbf{y}\|, \\ \delta\|\mathbf{x}+\mathbf{y}\| &\leq \delta\|\mathbf{x}+\alpha\mathbf{y}\|, \\ \|\mathbf{x}+\mathbf{y}\| &\leq \|\mathbf{x}+\alpha\mathbf{y}\|. \end{aligned}$$

□

Proof of Theorem 2.4.7. Suppose $\|\mathbf{a}\| = \lambda_1$ and $\|\mathbf{b}\| = \lambda_2$. By definition of λ_1 we know

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

Since $\{\mathbf{a}, \mathbf{b}\}$ is a basis, the sets $\{\mathbf{a}, \mathbf{a}+\mathbf{b}\}$ and $\{\mathbf{a}, \mathbf{a}-\mathbf{b}\}$ are linearly independent. So, by definition of λ_2 we have

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

Therefore

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

Conversely, suppose $\|\mathbf{a}\| \leq \|\mathbf{b}\| \leq \|\mathbf{a}+\mathbf{b}\|, \|\mathbf{a}-\mathbf{b}\|$. Let r, s be two integers. We need to show that $\mathbf{a}$ is a shortest nonzero vector in the lattice and $\mathbf{b}$ is a shortest linearly independent vector from $\mathbf{a}$. Since $\{\mathbf{a}, \mathbf{b}\}$ is a basis for $\mathcal{L}$, this means we need to show that

$$\|\mathbf{a}\| < \|r\mathbf{a}+s\mathbf{b}\| \qquad \text{for} \quad (r,s) \neq (0,0), \quad (r,s) \in \mathbb{Z}^2$$

and

$$\|\mathbf{b}\| \leq \|r\mathbf{a}+s\mathbf{b}\| \qquad \text{for} \quad s \neq 0, \quad (r,s) \in \mathbb{Z}^2$$

Let us consider the three cases below

- $s = 0$ and $r \neq 0$ then $\|r\mathbf{a}+s\mathbf{b}\| = \|r\mathbf{a}\| \geq \|\mathbf{a}\|$.
- $r = 0$ and $s \neq 0$ then

$$\|r\mathbf{a}+s\mathbf{b}\| = \|s\mathbf{b}\| \geq \|\mathbf{b}\| \geq \|\mathbf{a}\|.$$

- $r, s \neq 0$

$\|\mathbf{b}\| \leq \|\mathbf{b} \pm \mathbf{a}\|$ implies by Lemma 2.4.8 that $\|\mathbf{b}\| \leq \|\mathbf{b}+\alpha\mathbf{a}\|$ for $|\alpha| \geq 1$. Thus

$$\|\mathbf{b}\| \leq \|\mathbf{b}+\frac{r}{s}\mathbf{a}\| = \frac{1}{|s|}\|s\mathbf{b}+r\alpha\| \quad \forall r, s \quad \text{such that} \quad |\frac{r}{s}| \geq 1$$

which implies

$$\|\mathbf{b}\| \leq \|s\mathbf{b} + r\mathbf{a}\|.$$

Note that if $n, m > 0$, then

$$\begin{aligned} \|n(\mathbf{b} \pm \mathbf{a}) + m\|\mathbf{b}\| &\geq n\|\mathbf{b} \pm \mathbf{a}\| - m\|\mathbf{b}\| \\ &\geq (n-m)\|\mathbf{b}\| \end{aligned}$$

If $|\frac{r}{s}| < 1$ we have $|r| < |s|$. Since $\|\mathbf{a}\| \leq \|\mathbf{b}\|$ we have

$$\|s\mathbf{b} + r\mathbf{a}\| \geq |s|\|\mathbf{b}\| - |r|\|\mathbf{a}\| \geq (|s| - |r|)\|\mathbf{b}\|.$$

Therefore,

$$\|\mathbf{b}\| \leq \|s\mathbf{b} + r\mathbf{a}\|.$$

If $s < 0$ we can replace both r and s by their negative.

□

Definition 2.4.9. A basis $\{\mathbf{a}, \mathbf{b}\}$ is *well ordered* if

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

Notice that if $\{\mathbf{a}, \mathbf{b}\}$ is well ordered, then since the three points $\mathbf{b} - \mathbf{a}, \mathbf{b}$ and $\mathbf{b} + \mathbf{a}$ are on the same line, it follows from Lemma 2.4.8 that

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

Without loss of generality, we can assume that a given basis, if not reduced, is initially well ordered by swapping $\mathbf{a}$ and $\mathbf{b}$ and possibly replacing the basis by $\{\mathbf{a}, \mathbf{a} - \mathbf{b}\}$ or $\{\mathbf{a}, \mathbf{b} - \mathbf{a}\}$.

The basic idea of the generalized Gauss algorithm is to search through well ordered bases to find a reduced basis. The steps of the algorithm are as follows

1. Begin with a well ordered basis. Find $\mu \in \mathbb{Z}_{>0}$ such that $\|\mathbf{b} - \mu\mathbf{a}\|$ is minimal. Set this minimal value as the new value for $\mathbf{b}$. Note that $\|\mathbf{b} - \mathbf{a}\| < \|\mathbf{b}\|$ and μ must be positive.

2. Change the sign of $\mathbf{b}$ if necessary to get $\|\mathbf{a}-\mathbf{b}\| \leq \|\mathbf{a}+\mathbf{b}\|$.

3. If the obtained basis is not reduced, swap $\mathbf{a}$ and $\mathbf{b}$ and go to the first step.

Remark 2.4.10. Note that the length of the smallest vector in the basis gets smaller. Since there are a finite number of vectors of length k for any integer k, that guarantees the termination of the process.

The following lemma assures that the output of the last step is a good basis to begin with the next cycle.

Lemma 2.4.11. *The result of the last step of the generalized Gauss algorithm is always a well ordered basis.*

Proof. With the same notation and assumptions of the algorithm, the basis computed by the last step is $\mathbf{a}' = \pm(\mathbf{b}-\mu\mathbf{a})$ and $\mathbf{b}' = \mathbf{a}$. So

$$\begin{aligned} \|\mathbf{a}'-\mathbf{b}'\| &= \|\pm(\mathbf{b}-\mu\mathbf{a})-\mathbf{a}\| \\ &= \|\mathbf{b}-(\mu\pm 1)\mathbf{a}\| \\ &\geq \|\mathbf{b}-\mu\mathbf{a}\| \\ &= \|\mathbf{a}'\| \end{aligned}$$

Thus, we have that

$$\|\mathbf{a}'\| \leq \|\mathbf{a}'-\mathbf{b}'\| < \|\mathbf{a}'+\mathbf{b}'\|$$

So if $\|\mathbf{b}'\| \leq \|\mathbf{a}'-\mathbf{b}'\|$, then the basis is reduced. Otherwise, $\|\mathbf{b}'\| > \|\mathbf{a}'-\mathbf{b}'\| \geq \|\mathbf{a}'\|$ and it is well ordered. □

All the algorithms to solve the SVP are exponential time in the rank of the lattice. The SVP in a n-dimensional lattice is proved to be NP-hard under randomized reduction [4]. This lack of efficient algorithm leads to defining an approximation version of the problem.

Definition 2.4.12. (Approximate SVP) Given $\gamma > 1$ and a basis $\mathbf{B} \in M_{n\times m}(\mathbb{Z})$, find a non-zero lattice vector $\mathbf{xB}$ where $\mathbf{x} \in \mathbb{Z}^n \setminus \{0\}$, such that $\| \mathbf{xB} \| \leq \gamma \| \mathbf{yB} \|$ for any other $\mathbf{y} \in \mathbb{Z}^n \setminus \{0\}$.

Thus, the solution of Approximate SVP is a lattice vector whose norm is within a factor of γ of λ_1. The value γ is called the *approximation factor*. The most famous algorithm to approximating the SVP, known as LLL, was presented by Lenstra, Lenstra, Lovasz in 1982 [30]. LLL is a polynomial time algorithm approximates the SVP within the exponential factor $\gamma = 2^{O(n)}$. Here we summarize the LLL algorithm as it came in [36].

The LLL Basis Reduction Algorithm

A part of the hardness of SVP is that a lattice can have many different bases as we saw in Section 2.3. Typically, a given basis for a lattice contains vectors which are much longer than the shortest nonzero vector. The idea of the LLL algorithm, as in the Gauss algorithm, is to find a 'reduced' basis for the lattice whose first vector is within the γ factor of λ_1.

The notion of reduced basis defined before can be reformulated as follows.

Definition 2.4.13. A basis $\{\mathbf{a}, \mathbf{b}\}$ is LLL reduced if $|\mu_{2,1}| \leq \frac{1}{2}$ and $\|\mathbf{a}\| \leq \|\mathbf{b}\|$, where $\mu_{2,1}$ is Gram-Schmidt coefficient $\frac{\mathbf{b}\cdot\mathbf{a}}{\mathbf{a}\cdot\mathbf{a}}$.

Define the projection operation π_i from $\mathbb{R}^m$ onto $span(\mathbf{b}_j^\star)_{j\geq i}$ to be

$$\begin{aligned} \pi_i : \quad \mathbb{R}_m &\rightarrow span_{j\geq i}(\mathbf{b}_j^\star) \\ \mathbf{x} &\rightarrow \sum_{j=i}^{n} \frac{\mathbf{x}\cdot\mathbf{b}_j^\star}{\mathbf{b}_j^\star\cdot\mathbf{b}_j^\star}\mathbf{b}_j^\star. \end{aligned}$$

Definition 2.4.14. A basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ in $\mathbb{R}^n$ is δ-LLL if

- $|\mu_{i,j}| \leq \frac{1}{2}$ for all $i > j$
- For any pair of consecutive vectors $\mathbf{b}_i, \mathbf{b}_{i+1}$,

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

where $\mu_{i,j}$ is the Gram-Schmidt coefficient and δ is a parameter $1/4 < \delta < 1$ introduced for running time purposes which we will not discuss here. 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.

Remark 2.4.15. The second condition of the above definition is equivalent to

$$\delta\|\mathbf{b}_i^\star\|^2 \leq \|\mathbf{b}_{i+1}^\star + \mu_{i+1,i}\mathbf{b}_i^\star\|^2$$

Observe that for a δ-LLL reduced basis we have

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

So

$$(\delta - \frac{1}{4})\|\mathbf{b}_i^\star\|^2 \leq \|\mathbf{b}_{i+1}^\star\|^2. \tag{10}$$

Letting $\alpha = \frac{1}{\delta - \frac{1}{4}}$ and by induction on $i - j$ on equation (10), we get

$$\|\mathbf{b}_j^\star\|^2 \leq \alpha^{i-j}\|\mathbf{b}_i^\star\|^2 \quad \forall j < i.$$

Thus

$$\|\mathbf{b}_i^\star\|^2 \geq \frac{1}{\alpha^{i-1}}\|\mathbf{b}_1^\star\|^2 \geq \frac{1}{\alpha^{n-1}}\|\mathbf{b}_1\|^2$$

using the bound for the first minimum of the lattice, given by Theorem 2.3.15, we get

$$\lambda_1 \geq \min\|\mathbf{b}_i^\star\| \geq \frac{1}{\alpha^{(n-1)/2}}\|\mathbf{b}_1\|.$$

Therefore, $\|\mathbf{b}_1\|$ is at most $\alpha^{(n-1)/2}$ longer than a shortest vector in the lattice. Taking $\delta = 3/4$ in particular, we get an approximation factor $2^{(n-1)/2}$.

The LLL basis reduction algorithm computes the reduced basis from the input basis $\mathbf{B}$ in three steps.

1. Reduction Step

$$\mathbf{b}_i' = \mathbf{b}_i - \sum_{0<j<i} c_{i,j}\mathbf{b}_j \qquad \text{for } i = 1 \text{ to } n,$$

where $c_{i,j} = \lfloor \mu_{i,j} \rceil$. Note that the result of this step is still a basis for the same lattice with the same Gram-Schmidt orthogonalized vectors associated to $\mathbf{B}$. However, all the Gram-Schmidt coefficients are at most $1/2$.

2. Swap Step

   If any two consecutive vectors do not satisfy the second condition of δ-LLL reduced basis, then the algorithm swaps them.

3. Repeat Step

   By the end of step 2, if any swapping happens, the basis does not necessarily satisfy the first condition of δ-LLL reduced basis any more. So if that is the case, the algorithm starts from the first step. Otherwise, it gives out the reduced result.

This algorithm terminates in polynomial time for $\delta < 1$. However, the question of whether it terminates in polynomial time for $\delta = 1$ is an open problem. For details on the running time of the algorithm and the proof of termination, see [36].

Since the invention of the LLL algorithm, many efforts have been made to improve the approximation factor. In 1987, Schnorr presented an algorithm which decreases the approximation factor to $\gamma = 2^{O(n(\ln\ln n)^2/\ln n)}$. In 2001, Ajtai found a probabilistic method to solve the exact problem in $2^{O(n)}$ time [3] which consolidated with Schnorr method improves the approximation factor. The approximate SVP is shown to be NP hard to within factors $2^{(\log n)^{\frac{1}{2}-\epsilon}}$ under randomize reduction [35]. However, approximate SVP to within $\sqrt{d/\log d}$ is not believed to be NP-hard [16]. We should also say that the best known algorithm for solving the exact problem has $2^{O(n)}$ running time [3].

Another problem which is closely related to SVP is to find a basis of a given lattice with minimum size.

Definition 2.4.16. (Shortest Basis Problem, SBP) Given a basis $\mathbf{B}$ for the lattice $\mathcal{L}$, find the basis $\mathbf{B}'$ such that $\prod_i \|\mathbf{b}'_i\|$ is less than any other basis of $\mathcal{L}$.

In fact, solving this problem is the approach that many lattice reduction algorithms take to solve SVP.

2.4.2 Closest Vector Problem

Another related lattice problem is the closest vector problem.

Definition 2.4.17. (Closest Vector Problem, CVP) Given a lattice basis $\mathbf{B} \in M_{n\times m}(\mathbb{Z})$ and a target vector $\mathbf{t} \in \mathbb{R}^m$, find a lattice vector $\mathbf{xB}$ closest to the target $\mathbf{t}$, i.e., find an integer vector $\mathbf{x}\in \mathbb{Z}^n$ such that $\| \mathbf{xB} - \mathbf{t} \| \leq \| \mathbf{yB} - \mathbf{t} \|$ for any other $\mathbf{y} \in \mathbb{Z}^n$.

The Closest Vector Problem is NP-hard [36, Chapter 3]. Therefore, no efficient algorithm to solve CVP exists unless P=NP. So, similar to SVP case, efforts are focused on solving the approximation version of the problem.

Definition 2.4.18. (Approximate CVP) Given a lattice basis $\mathbf{B} \in M_{n\times m}\mathbb{Z}$ and a target vector $\mathbf{t} \in \mathbb{R}^m$, find an integer vector $\mathbf{x}\in \mathbb{Z}^n$ such that $\| \mathbf{xB}-\mathbf{t} \| \leq \gamma \| \mathbf{yB} - \mathbf{t} \|$ for any other $\mathbf{y} \in \mathbb{Z}^n$.

As in approximate SVP, γ in called the approximation factor. Approximating CVP in an n-dimensional lattice is known to be NP-hard under deterministic reduction for any constant approximation factor or even some slowly increasing functions of the dimension ($\gamma(n) = O((\log n)^n)$) [36]). In [8], two algorithms are presented for Approximate CVP which are both based on the LLL algorithm. Here we summarize both algorithms.

Babai Round Off Algorithm

Suppose that the basis $\mathbf{B}'$ and the target point $\mathbf{t}$ are given. The algorithm goes through the following steps:

1. Run the LLL algorithm to get an LLL-reduced basis $\mathbf{B}$.
2. Represent $\mathbf{t}$ as linear combination of $\mathbf{B}$ as follows:

$$\mathbf{t} = \sum_{i=1}^{n} \beta_i \mathbf{b}_i,$$

   where the β_is are real values.
3. Set $\alpha_i = \lfloor \beta_i \rceil$.
4. Set $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \mathbf{b}_i$.

The following theorem gives the approximation factor achieved by the Babai Round-off algorithm. The proof can be found in [8].

Theorem 2.4.19. *For a reduced basis $\boldsymbol{B}$, the Round-off algorithm finds a lattice point $\boldsymbol{w}$, nearest to $\boldsymbol{t}$ within a factor of $\gamma = 1 + 2n(9/2)^{n/2}$.*

Nearest Plane Algorithm

Suppose that the basis $\mathbf{B}'$ and the target point $\mathbf{t}$ are given. The steps of the algorithm are as follows:

1. Run the LLL algorithm to get a LLL-reduced basis $\mathbf{B}$.
2. Set U to be the hyperplane spanned by the first $n-1$ basis vectors and let $\mathcal{L}'$ denote $\mathcal{L}' = \mathcal{L} \cap U$. Then,
$$U = \text{span}_{\mathbb{R}}\{\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}\} \quad \text{and} \quad \mathcal{L}' = \text{span}_{\mathbb{Z}}\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}.$$
3. Write $\mathbf{t}$ as a linear combination of the Gram-Schmidt orthogonal basis $\mathbf{B}^\star$:
$$\mathbf{t} = \sum_{i=1}^{n} y_i \mathbf{b}_i^\star.$$
4. Set $y' = \lfloor y_n \rceil$ and $\mathbf{x}' = \sum_{i=1}^{n-1} y_i \mathbf{b}_i^\star + y' \mathbf{b}_n^\star$ and $\mathbf{v} = y' \mathbf{b}_n$.
5. Recursively, find $\mathbf{w}' \in \mathcal{L}'$ closest to $\mathbf{x}' - \mathbf{v}$ and set $\mathbf{w} = \mathbf{w}' + \mathbf{v}$.

Theorem 2.4.20. *For a reduced basis $\boldsymbol{B}$, the Nearest Plane algorithm finds a lattice point $\boldsymbol{w}$ nearest to $\boldsymbol{t}$ within a factor of $\gamma = 2^{n/2}$. Moreover, $\|\boldsymbol{t} - \boldsymbol{w}\| < 2^{n/2-1}\|\mathbf{b}_n^\star\|$.*

The proof can be found in [8].

It is worth mentioning that there is a gap between the known hardness results for both problems. Although it is easy to establish the NP-hardness of CVP, the question of whether SVP is NP-hard was open for decades and eventually was solved under randomized reduction [4].

Approximating CVP is also NP-hard under deterministic reduction for any constant approximation factors [7]. However, SVP is only known to be NP-hard under

randomized reduction for approximation factors below $\sqrt{2}$ [36]. It can be shown that SVP is not harder than CVP. Indeed, it is possible to reduce the task of finding γ-approximate solution to SVP to the task of finding γ-solution to CVP [15].

In conclusion, the existence of Approximate SVP or CVP time algorithms that achieve approximation factors polynomial in the rank of the lattice is one of the main open problems in the lattice study area. The difficulty of solving these problems has been used to build the cryptosystems that we will study in the rest of the thesis.

Chapter 3

The Ajtai-Dwork Cryptosystem

In 1997, Miklos Ajtai and Cynthia Dwork [2] presented three related probabilistic cryptosystems whose securities are based on a special case of SVP, called the *unique-Shortest Vector Problem* (u-SVP). They introduced the so-called *hidden hyperplane problem* and showed its relation to u-SVP and the fact that solving the former leads to a solution for the latter.

In particular, their third proposed cryptosystem is based on the worst case of u-SVP. That means if the attacker, without knowing the private key, can distinguish between the encryption of 1 and 0 in a random instance of the ciphertext with probability greater than 1/2, then the worst case of u-SVP has probabilistic polynomial time solution. According to [2], this was essentially the first cryptosystem which is as hard to break as the worst case instance of the underlying mathematical problem.

Although it was shown in 1998 by Nguyen and Stern [40] that secure implementation of any of these cryptosystems would require very large keys, and hence be impractical in real life, the Ajtai-Dwork cryptosystems remain one of the major breakthroughs in theoretical cryptography.

In this chapter, we describe the lattice problems upon which the Ajtai-Dwork cryptosystems are based. We then give an overview of the first and third cryptosystems from [2]. The first two cryptosystems only differ in a constraint in the choice of a parameter which results in different key size and security argument. Since we would not be covering the technical security discussion in this chapter, we omit the second

one. The ideas of the proofs included in this chapter are taken from [2]. However, the details are our own.

3.1 The Hidden Hyperplane Problem

In this section, we define the n^c-unique SVP and the hidden Hyperplane problem, as well as the class of lattices for which the lattice problem is defined. "These problems are dual."

Definition 3.1.1. (Unique Shortest Vector Problem, n^c-SVP) Given $n \in \mathbb{N}$ and a constant c, the *n^c-Unique Shortest Vector Problem* (n^c-SVP) is to find the shortest nonzero vector $\mathbf{v}$ in an n-dimensional lattice which has the property that any other lattice vector with length at most $n^c\|\mathbf{v}\|$ is an integer multiple of $\mathbf{v}$. We refer to $\mathbf{v}$ as the *n^c-unique shortest vector.*

Now we consider a special class of lattices called (d, M)-lattices.

Definition 3.1.2. Let $M, d > 0$ be real numbers and $\mathcal{L} \subseteq \mathbb{Z}^n$ be an n dimensional lattice. We say $\mathcal{L}$ is a *(d, M)-lattice* if $\mathcal{L}$ has an $n-1$ dimensional sublattice $\mathcal{L}'$ such that $\mathcal{L}'$ has a basis $\mathbf{B}'$ of length less than or equal to M. Moreover, if $H = \text{span}(\mathbf{B}')$, and $\mathbf{b} \in \mathcal{L}$ and $\mathbf{b}$ in not in H, then the Euclidean distance between H and $\mathbf{b} + H$ is at least d.

Lemma 3.1.3. *Suppose $\mathcal{L}$ is a (d, M)-lattice. If $d > M$, then $\mathcal{L}'$ is unique.*

Proof. Suppose $d > M$ and assume that there are two non-equal sublattices $\mathcal{L}'$ and $\mathcal{L}''$ of $\mathcal{L}$ with basis $\mathbf{B}'$ and $\mathbf{B}''$ respectively, satisfying the conditions of Definition 3.1.2. Let $\mathbf{b}$ be a basis vector of $\mathcal{L}''$ such that $\mathbf{b}$ is not in $H = \text{span}(\mathbf{B}')$. Then the distance between H and $H + \mathbf{b}$ is at most $\|\mathbf{b}\|$. On the other hand, $\|\mathbf{b}\| < M < d$ which contradicts the second condition in Definition 3.1.2. Thus, such $\mathbf{b}$ does not exist and $\mathcal{L}'' \subseteq \mathcal{L}'$. Similarly, $\mathcal{L}' \subseteq \mathcal{L}''$. □

In this case, the unique sublattice $\mathcal{L}'$ is denoted by $\mathcal{L}^{(d,m)}$ and the nonzero distance between H and the closest $\mathbf{b} + H$ is denoted by $d_{\mathcal{L}}$.

Definition 3.1.4. (Hidden Hyperplane Problem, HHP)Given an arbitrary basis $\mathbf{B}$ for an n-dimensional (d, M)-lattice $\mathcal{L}$ with $d > n^c M$ for a constant c, the *hidden hyperplane problem (HHP)* is to find $\mathcal{L}^{(d,M)}$.

Proposition 3.1.5. *Let Λ be a n-dimensional lattice with an n^c unique shortest vector $\mathbf{u}$. Let $\mathcal{L} = \Lambda^*$ be the dual of Λ. The lattice $\mathcal{L}$ is a n-dimensional (d, M)-lattice for some M, in which, using the notations in Definition 3.1.2, the smallest distance between H and $\mathbf{b} + H$ is $\|\mathbf{u}\|^{-1}$.*

Proof. Let Λ, $\mathbf{u}$ and $\mathcal{L}$ be as in the statement of the proposition. We need to show that $\mathcal{L}$ has a $n-1$-dimensional sublattice that satisfies the two conditions of Definition 3.1.2. Observe that since $\mathcal{L}$ is the dual lattice of Λ and $\mathbf{u} \in \Lambda$, by definition of a dual lattice, for all the vectors in $\mathcal{L}$ we have

$$\mathbf{v} \cdot \mathbf{u} \in \mathbb{Z} \quad \forall \mathbf{v} \in \mathcal{L}.$$

For each $i \in \mathbb{Z}$, define H_i as

$$H_i = \{\mathbf{v} \in \mathbb{R}^n | \quad \mathbf{u} \cdot \mathbf{v} = i\}.$$

Thus it is clear that

$$\mathcal{L} \subseteq \bigcup_{i \in \mathbb{Z}} H_i.$$

Define $H = H_0$, which is an $n-1$ dimensional subspace of $\mathbb{R}^n$, and let $\mathcal{L}^{(d,M)}$ be the intersection of H with $\mathcal{L}$. Then $\mathbf{u}$ is orthogonal to H. Note that the cosets of H that intersect $\mathcal{L}$ are exactly the H_is. Let $\mathbf{v}_i \in H_i$, then the distance between H and H_i is

$$\begin{aligned} \|Proj_{\mathbf{u}}(\mathbf{v_i})\| &= \frac{|\mathbf{u} \cdot \mathbf{v}_i|}{\|\mathbf{u}\|} \\ &= \frac{i}{\|\mathbf{u}\|} \end{aligned}$$

So, the minimum distance between H and any H_i is $\frac{1}{\|\mathbf{u}\|}$.

□

In [2], it is further claimed that one may take $M \approx \frac{1}{n^{c-2}\|\mathbf{u}\|}$, by applying an algorithm (see [1]) which produces such a small basis for $\mathcal{L}^{(d,M)}$ in polynomial time.

Theorem 3.1.6. *Given a basis for a random (d, M)-lattice with $d > n^c M$ for some constant c, a solution to the hidden hyperplane problem implies a solution to* u-*SVP in the dual lattice.*

Idea of proof. Suppose $\mathcal{L}$ is a (d, M) lattice. If one finds H or $\mathcal{L}^{(d,M)}$, then by the proof of Proposition 3.1.5, the shortest vector is orthogonal to H, so its direction is determined. To find its length, recall that all the vectors of $\mathcal{L}^*$ are of distance $\frac{i}{\|\mathbf{u}\|}$ from H for an integer i. So as it is claimed in [2], by computing the gcd of the distances of random points in $\mathcal{L}$ from H, d_L can be computed in probabilistic polynomial time. Therefore, upon knowing H, one can find the unique shortest vector in Λ. □

Since the unique shortest vector problem is among the famous hard problems in the lattices, we deduce that the hidden hyperplane problem is also hard.

All three cryptosystems introduced in [2] are based on the hidden hyperplane problem on a (d, M)-lattice. Roughly speaking, the private key is the hidden hyperplane and the public key is a technique to generate a vector close to the lattice. Encryption is done bit by bit: zero is encrypted to a vector close to a lattice point and one is encrypted to a random vector, in $\mathbb{R}^n$. A ciphertext is decrypted by measuring its distance from the closest hyperplane. If the distance is small enough, the message is decrypted as zero. Otherwise, it is decrypted as one. There is a small probability of decryption error, i.e., one might get decrypted as 0, depending on the values of d and M. A technique to eliminate this error is described in [17].

3.2 First and Second Cryptosystems

As it is mentioned in the beginning of the chapter, since these two cryptosystems are quite similar, we will just explain the first one here. The public key is a random basis for a (d, M)-lattice $\mathcal{L}$ with $d \geq n^c M$ for some constant c and some $M > 0$ while the private key is a basis for $\mathcal{L}^{(d,M)}$. Alice, having the public key, encrypts the message 0 to a vector close to $\mathcal{L}$. This is done by choosing a lattice point at random and perturbing it by adding a small random perturbation vector. The message 1 is encrypted to a random vector in the span of $\mathcal{L}$. Bob, having the private key,

has the privilege of distinguishing whether the received vector is close to any of the hyperplanes. The perturbation vector is always much smaller than the distance d between the hyperplanes by suitable choices of the parameters. So, the distinction is done by projecting the received vector onto an orthogonal vector to H and comparing the norm with $d_{\mathcal{L}}$. If it is sufficiently small, Bob deduces that Alice has sent 0.

3.2.1 On The Security of The First Cryptosystem

In this section, we outline how the security of the first cryptosystem is given by the hidden hyperplane problem. We also explain methods for generation of the perturbation vector needed for encryption. Let U^n denote the unit cube in $\mathbb{R}^n$ and $B_n(0,R)$ denote the n-dimensional ball with radius R. In this subsection, we will sample lattice points within a small cube and also we will choose a vector uniformly from $B_n(0,R)$. Methods that show how to perform these tasks are presented in [1] and [2], respectively.

Definition 3.2.1. Let $\mathcal{L}$ be a lattice and $K > 0, R > 0$ be real numbers. The random variable $\zeta_{\mathcal{L},K,R}$ is defined as follows. First take a vector $\mathbf{x}$ chosen uniformly at random from $KU^n \cap \mathcal{L}$. Then choose m vectors $\mathbf{t}_1, \ldots, \mathbf{t}_m$ uniformly at random from $S^n(R)$, where $m = c_0 n$ for some $c_0 > 4$. The value of $\zeta_{\mathcal{L},K,R}$ is

$$\zeta_{\mathcal{L},K,R} = \mathbf{x} + \sum_{i=1}^{m} \mathbf{t}_i.$$

The additional term $\sum_{i=1}^{m} \mathbf{t}_i$ is also referred to as a *perturbation* and is denoted as $\mathrm{pert}(R,m)$.

Definition 3.2.2. Define the random variable η_K whose values are taken with uniform distribution on KU^n.

Definition 3.2.3. Let δ be a random variable taking 0 and 1 as its value each with probability equal to 1/2. Define a new random variable $\nu_{\mathcal{L},K,R}$ by randomizing the three random variables $\delta, \zeta_{\mathcal{L},K,R}$ and η_K independently and setting

$$\nu_{\mathcal{L},K,R} = \begin{cases} \eta_K & \text{if } \delta = 0 \\ \zeta_{\mathcal{L},K,R} & \text{if } \delta = 1. \end{cases}$$

We say that a probabilistic algorithm $\mathcal{B}$ finds $\mathcal{L}^{(d,M)}$ on $\mathcal{L}$ with probability p, if given d, M and a randomly chosen (d, M)-lattice $\mathcal{L}$ as input, $\mathcal{B}$ outputs $\mathcal{L}^{(d,m)}$ with probability p.

Definition 3.2.4. A probabilistic algorithm $\mathcal{A}$ distinguishes $\zeta_{\mathcal{L},K,R}$ and η_K with probability p if given a randomly chosen (d, M)-lattice $\mathcal{L}$ and a random value of $\nu_{\mathcal{L},K,R}$ as an input, $\mathcal{A}$ outputs $w = 0$ or 1 so that $P(w = \delta) = p$.

The following theorem shows that breaking the scheme, that is distinguishing between ciphered zero and one, leads to figuring out the private key, that is the hidden hyperplanes. This is the main theorem which justifies the security of the first cryptosystem. For proof and more details see [2].

Theorem 3.2.5. *There exist $c, c_4, c_5, c_6 > 0$ so that for all $c_1 > 0, c_2 > 0$ there exist $c_3 > 3$ and a probabilistic algorithm $\mathcal{B}$ (using an oracle) so that if n, d, M, K, R are positive integers satisfying the inequalities*

1. $\log d + \log M + \log K + \log R < n_1^c$
2. $d > n^{c_6} M$
3. $R > n^c M$
4. $2^{c_5 n} d > K > 2^{c_4 n} d$

and $\mathcal{D}$ is a distribution on the set of (d, M)-lattices in $\mathbb{Z}^n$ presented by vectors of length at most $n^{c_1} d_{\mathcal{L}}$ and for which $d_{\mathcal{L}} > n^5 M$ and $d \leq d_{\mathcal{L}} \leq 2d$, and $\mathcal{A}$ is a probabilistic algorithm which distinguishes $\zeta_{\mathcal{L},K,R}$ and η_K on $\mathcal{D}$ with probability at least $1/2 + n^{-c_2}$, then $\mathcal{B}$ using $\mathcal{A}$ as an oracle, finds $\mathcal{L}^{(d,M)}$ on $\mathcal{L}$ with a probability at least $1 - 2^{-n}$, in time n^{c_3}.

It follows from Theorems 3.2.5 and 3.1.6 that the security of the above cryptosystem relies on the hardness of u-SVP. That is for each instance of the problem, breaking the cryptosystem built on that is as difficult as solving the given instance of the problem.

3.2.2 Key Generation

Let $M, K, R, d > 0$ be positive integers satisfying the conditions in Theorem 3.2.5. The key generating steps are as follows:

1. Generate a random $n-1$ dimensional lattice $\mathcal{L}'$ with a basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}\}$ such that $\|\mathbf{b}_i\| \leq M$ for$1 \leq i \leq n-1$. Let H be the $n-1$ dimensional subspace of $\mathbb{R}^n$ containing $\mathcal{L}'$.

2. Choose a random vector $\mathbf{b}_n$ whose distance $d_{\mathcal{L}}$ from H satisfies $d \leq d_{\mathcal{L}} \leq 2d$. Set $\mathcal{L} = \mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}, \mathbf{b}_n)$

3. The private key is any basis for $\mathcal{L}^{(d,M)} = \mathcal{L}'$.

4. Construct a random basis $\mathbf{B}'$ for $\mathcal{L}$. The public key is $\{\mathbf{B}', M, K, R\}$.

Remark 3.2.6. To generate a random basis one can generate a random unimodular matrix as we do in Chapter 4. Another way to generate the public key $\mathbf{B}'$ is suggested by Micciancio in 2002 using *Hermite normal forms* [34]. This technique can be applied to all the lattice based cryptosystems where the public key is a lattice basis and the ciphertext is a vector close to lattice.

3.2.3 Encryption and Decryption

Encryption

Suppose Alice, having the basis $\mathbf{B}'$ for $\mathcal{L}$ and the parameters M, K and R wants to send a one-bit message to Bob. To encrypt zero, she does the following:

1. She chooses a random lattice point $\mathbf{x}$ in the cube $KU^n \cap \mathcal{L}$.

2. She randomizes a value $\mathbf{w}$ for pert(R, m) for some $m = c_0 n$ where $c_0 > 4$ as is explained in Definition 3.2.1.

3. She sets the ciphertext to be $\mathbf{x} + \mathbf{w}$.

To encrypt one, she chooses a random point in KU^n and sends it as the ciphertext.

Decryption

Bob, having the private basis for H, receives the ciphertext $\mathbf{z}$. He proceeds as follows:

1. He computes $w = (\mathbf{u}_H \cdot \mathbf{z})/d_L$, where $\mathbf{u}_H$ is a unit vector orthogonal to H, and sets $\alpha = w - \lfloor w \rceil$.

2. If $|\alpha| < \frac{nR}{d_L}$, then $\mathbf{z}$ is decrypted as zero, and otherwise as one.

For the parameters satisfying Theorem 3.2.5, the perturbation vector is much shorter than $d_{\mathcal{L}}$ which guarantees the correct decryption of zero. However, sending the message one, decryption error can occur with small probability. That happens when the random point selected in KU^n by the encryption algorithm is close to one of the hyperplanes by chance. A technique to eliminate this error is described in [17].

3.3 The Third Cryptosystem

In this system no lattice is introduced. However, the main idea is still based on the hidden hyperplanes. The difference is that instead of publishing a basis for the lattice $\mathcal{L}$, a collection of m vectors close to hyperplanes is published. The encryption of zero uses the fact that the sum of any subset of these vectors added to a perturbation vector is still close enough to the lattice. The distinction between the encrypted one and zero in the decryption algorithm is done by measuring the distance from the closest hyperplane, having an orthogonal vector to the hyperplanes as a private key. The popularity of this version is due to the worst case/average case equivalency which is shown in the security proof in [2].

We first set some notation. For a fixed integer n, let $\mathcal{K}(n)$ denote the function $2^{n \log n}$ and let Υ be a random variable whose values are taken uniformly from the n-dimensional cube $\mathcal{K}(n)U^n$.

Definition 3.3.1. For a vector $\mathbf{u} \in \mathbb{R}^n$ of norm $0 \leq \|\mathbf{u}\| \leq 1$, $R > 0$ and m a positive integer, the random variable $\Psi_{\mathbf{u},R,m}$ is defined as follows:

- Let X be the set of all $\mathbf{x} \in \mathcal{K}(n)U^n$ so that $\mathbf{x} \cdot \mathbf{u}$ is an integer. Thus X consist of subsets of a finite number of $n-1$ dimensional hyper planes.

- Take a random point $\mathbf{y}$ in X.
- Independently, sample a value $\mathbf{z}$ from the random variable $\text{pert}(R, m)$.

The value $\Psi_{\mathbf{u},R,m}$ is $\mathbf{y} + \mathbf{z}$.

Definition 3.3.2. For vectors $\mathbf{a}_1, \ldots, \mathbf{a}_n$ in $\mathbb{R}^n$, the *width*$(\mathbf{a}_1, \ldots, \mathbf{a}_n)$ is defined by the width of fundemental parallelepiped $\mathcal{P}(\mathbf{a}_1, \ldots, \mathbf{a}_n)$, that is the minimum over i of the distances between the point $\mathbf{a}_i$ and the subspace generated by $\{\mathbf{a}_j | j \neq i\}$.

Definition 3.3.3. Given a vector $\mathbf{x} \in \mathbb{R}^n$ and n vectors $\mathbf{w}_1, \ldots, \mathbf{w}_n$ in $\mathbb{R}^n$, *reducing modulo* $\mathcal{P}(\mathbf{w}_1, \ldots, \mathbf{w}_n)$ means finding a vector $\mathbf{x}'$ inside $\mathcal{P}(\mathbf{w}_1, \ldots, \mathbf{w}_n)$ so that $\mathbf{x} - \mathbf{x}'$ is an integer linear combination of the vectors $\mathbf{w}_1, \ldots, \mathbf{w}_n$.

3.3.1 Key Generation

For a fixed n, the public key and private key are chosen in the following way.

1. Choose a random vector $\mathbf{u} \in \mathbb{R}^n$ uniformly with $\|\mathbf{u}\| \leq 1$.
2. Generate $m = n^3$ independent values $\mathbf{v}_1, \ldots, \mathbf{v}_m$ of the random variable $\Psi_{\mathbf{u},n^{-3},n}$.
3. Find the smallest integer i_0 so that width$(\mathbf{v}_{i_0+1}, \ldots, \mathbf{v}_{i_0+n})$ is at least $n^{-2}\mathcal{K}(n)$. Let $\mathbf{w}_1 = \mathbf{v}_{i_0+1}, \ldots, \mathbf{w}_n = \mathbf{v}_{i_0+n}$.
4. Set the private key to be $\mathbf{u}$.
5. Set the public key to be $\mathbf{v}_1, \ldots, \mathbf{v}_m$ together with $\mathbf{w}_1, \ldots, \mathbf{w}_n$.

3.3.2 Encryption and Decryption

Encryption

Suppose Alice, having Bob's public key, wants to send a one bit message to Bob. She encrypts 0 as follows:

1. Chooses m values $\delta_1, \ldots, \delta_m$ independently from $\{0, 1\}$ with equal probability.

2. Computes the vector $\mathbf{x} = \sum_{j=1}^{m} \delta_i \mathbf{v}_i$.

3. Reduces $\mathbf{x}$ modulo $\mathbf{w}_1, \ldots, \mathbf{w}_n$ to $\mathbf{x}'$.

Then $\mathbf{x}'$ is the encrypted message. To encrypt 1, she chooses a random vector uniformly from the set $\mathcal{P}(\mathbf{w}_1, \ldots, \mathbf{w}_n) \cap 2^{-n}\mathbb{Z}^n$ and sends it as the ciphertext.

Decryption

Suppose Bob, having the private key vector $\mathbf{u}$, wants to decrypt the received ciphertext $\mathbf{z}$.

1. He computes $\mathbf{u} \cdot \mathbf{z} = i + \theta$ where $i \in \mathbb{Z}$ and $|\theta| \leq 1/2$.

2. If $|\theta| < 1/n$ then he decrypts the message as 0.

3. Otherwise, the message is decrypted as 1.

Remark 3.3.4. The bit zero is always decrypted correctly while one is decrypted correctly with a probability of at least $1 - 1/n$.

As claimed in [2], this cryptosystem is the only known system with the worst case/average case equivalent property. In other words, all public key cryptosystems are based on a hard mathematical problems. That is a problem which is either proved to be impossible to solve or is computationally infeasible to solve. But these results are for the worst case of the problem. That is, there might be an instance of the problem which is not that hard to solve. As an example, we know that factoring a large enough number is hard but for an instance to be hard to factor, one should avoid particular distributions such as even numbers. Most cryptosystems are only based on the average case of the underlying problem. In the Ajtai-Dwork third cryptosystem, if one can break a random instance of the system, that is, if one (without having the private key information) can distinguish between ciphered one and ciphered zero in polynomial time with the probability greater than 1/2, then the worst case unique shortest vector problem has a probabilistic polynomial time solution.

3.4 Summary

The Ajtai-Dwork cryptosystem is based on the u-SVP which can be reduced to the hidden hyperplane problem. The private key is produced by constructing a (d, M)-lattice $\mathcal{L}$ for given d and M where $d \geq n^c M$. This is done by picking $\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}$ vectors at random such that width of $\mathbf{b}_i$s is bounded above by M. The last basis vector $\mathbf{b}_n$ is chosen in such a way that its distance from $H = \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$ is between d and $2d$. All the lattice points lie in the hyperplanes $H_k = k\mathbf{b}_n^\star + H$. The public basis is a piece of information that allows one to create a vector close to the lattice. This information is a random basis for $\mathcal{L}$ in first and n^3 vectors close to hyperplanes in third cryptosystem. The private basis is a piece of information which determines $\mathcal{L}^{(d,M)}$. That information could be any basis of $\mathcal{L}^{(d,M)}$ in the first cryptosystem and $\mathbf{b}_n^\star$ in the third one.

The encryption is done bit by bit. The plaintext zero is encrypted to a vector close to $\mathcal{L}$. In first cryptosystem, this is done by choosing a random vector $\mathbf{x}$ in $\mathcal{L}$ and random vectors $\mathbf{t}_1, \ldots, \mathbf{t}_m$ in $B_n(0, R)$. The ciphertext is $\mathbf{x} + \sum_i \mathbf{t}_i$. In the third cryptosystem, summing up the close-to-lattice vectors gives us the encryption of zero. One is encrypted to an arbitrary vector in $\mathbb{R}^n$.

The Ajtai-Dwork cryptosystem does not fit in the general format of one-way trapdoor function based cryptosystems, since the input is an information bit b plus the randomly chosen vectors $\mathbf{t}_1, \ldots, \mathbf{t}_m$, but, the decryption process can only recover b and not the perturbation vector.

Hereafter, when we refer to the Ajtai-Dwork cryptosystem, we will mean the first one presented here, because of its clear lattice-based structure.

Chapter 4

The GGH Cryptosystem

The GGH cryptosystem was proposed by Goldreich, Goldwasser and Halevi in 1997 [18]. The security of their system is based on the difficulty of the closest vector problem (CVP) in lattices. They introduced a trapdoor function, based on which they proposed an encryption scheme and a digital signature scheme. The idea underlying their trapdoor function is that generating a vector close to a lattice vector is easy, given any basis of the lattice. However, given a vector close to a lattice, it is hard to recover this original lattice point. Although the GGH cryptosystem has been shown to be insecure unless over a high dimensional lattice that makes it inefficient [38], it describes a clear method of using lattices in cryptography.

In this chapter, we describe the GGH trapdoor function and the cryptosystem scheme based on it. The material presented in this chapter is mainly taken from [18] as well as the proofs which are included here. The exceptions are Definitions 4.1.1 and 4.1.4, which are added to improve clarity, and the proof of Lemma 4.2.1, which is standard.

4.1 Definitions

The GGH trapdoor function requires the useful notion of the *orthogonality defect* of a lattice basis. This was first defined by Schnorr [47].

Definition 4.1.1. Let $\mathbf{B}$ and $\mathbf{B}'$ denote two bases of the n-dimensional lattice $\mathcal{L}$. We say $\mathbf{B}$ is *smaller* than $\mathbf{B}'$ if $\prod_i \|\mathbf{b}_i\| \leq \prod_i \|\mathbf{b}'_i\|$.

Definition 4.1.2. Let $\mathbf{B}$ be a $n \times n$ non-singular matrix. The *orthogonality defect* of $\mathbf{B}$ is defined as

$$\text{orth-defect}(\mathbf{B}) = \frac{\prod_{i=1}^{n} \|\mathbf{b}_i\|}{\det(\mathbf{B})}. \tag{11}$$

Definition 4.1.3. Let $\mathbf{B}$ be a $n \times n$ non-singular matrix. The *dual orthogonality defect* of $\mathbf{B}$ is defined to be the orthogonality defect of $(\mathbf{B}^{-1})^T$

$$\begin{aligned} \text{orth-defect}^*(\mathbf{B}) &= \frac{\prod_{i=1}^{n} \|\mathbf{b}_i^*\|}{\det(\mathbf{B}^{-1})} \\ &= \det(\mathbf{B}) \prod_{i=1}^{n} \|\mathbf{b}_i^*\|, \end{aligned} \tag{12}$$

Where $\mathbf{b}_i^*$ is the ith row of $(\mathbf{B}^{-1})^T$.

Definition 4.1.4. Let $\mathbf{B}$ and $\mathbf{B}'$ denote two bases of the n-dimensional lattice $\mathcal{L}$. We say $\mathbf{B}$ is *more orthogonal* than $\mathbf{B}'$ if $\text{orth-defect}(\mathbf{B}) < \text{orth-defect}(\mathbf{B}')$.

Note that by Hadamard's inequality we have

$$|\det(\mathbf{B})| \leq \prod_{i=1}^{n} \|\mathbf{b}_i\|,$$

where equality happens if and only if the rows of $\mathbf{B}$ are pairwise orthogonal. So,

$$|\text{orth-defect}(\mathbf{B})| \geq 1$$

and reaches its minimum when the rows of $\mathbf{B}$ are pairwise orthogonal. Let $\mathbf{B}$ and $\mathbf{B}'$ correspond to bases for a lattice $\mathcal{L}$. By Proposition 2.3.10, they have equal determinants so, $\mathbf{B}$ is more orthogonal than $\mathbf{B}'$ implies $\prod_{i=1}^{n} \|\mathbf{b}_i\| < \prod_{i=1}^{n} \|\mathbf{b}'_i\|$, which implies $\mathbf{B}$ is smaller than $\mathbf{B}'$ and viceversa.

Note that with respect to Definition 4.1.1 of a small basis, one can state the Shortest Basis Problem as finding a most possible orthogonal basis for a given lattice.

4.2 GGH One way Trapdoor Function

The trapdoor function proposed in [18] is based on the difficulty of solving the closest vector problem (CVP). The input of the one-way trapdoor function f is an integral vector $\mathbf{v}$ together with a small error vector e. The function f, using a 'bad' basis $\mathbf{B}$ of a lattice $\mathcal{L}$, outputs a vector $f(\mathbf{v}, e)$ in the $\mathbb{R}$-span of the lattice. Recovering $\mathbf{v}$ from $f(\mathbf{v}, e)$ is equivalent to solving the CVP for the lattice $\mathcal{L}$. However, given a 'good' basis $\mathbf{R}$ for $\mathcal{L}$, v can be recovered easily from $f(\mathbf{v}, e)$. So $\mathbf{R}$ is the trapdoor information. We shall see in Section 4.2.3 that a 'good' basis, that allows to inverte f, is one with low dual orthogonality defect while a 'bad' basis has high dual orthogonality. As we saw in Chapter 2, to explain a one-way trapdoor function explicitly, one needs to give four polynomial time algorithms: generate, sample, evaluate and invert.

4.2.1 Generate Algorithm

Given the positive integer n as security parameter, the Generate algorithm produces a lattice basis $\mathbf{B} \in M_{n\times n}(\mathbb{Z})$ with high dual orthogonality defect and a positive real number σ together with the trapdoor information, which is a basis $\mathbf{R}$ for the same lattice with low dual orthogonality defect. We refer to $\mathbf{B}$ as the public basis since, as we will see in Section 4.3, it serves us as a part of public key in encryption scheme. For similar reasons, we refer to $\mathbf{R}$ as the private basis.

Basically, the one-way function $f : (\mathbb{Z}^n \times \mathbb{R}^n) \to \mathbb{R}^n$ is finding a corresponding lattice point to the input integral n-tuple and adding a 'small' perturbation vector to a lattice point. Thus, knowing the lattice and the perturbation vector would completely determine the function. The basis $\mathbf{B}$ determines the lattice and the positive real σ gives the size of the perturbation vector. On one hand, σ should be small enough so that enables a closest vector approximation algorithm using the private key R to invert the one-way trapdoor function. On the other hand, σ should be chosen large enough so that the algorithms having public key or any other basis with not small enough dual orthogonality defect, would not be able to invert the function. The value of σ depends on the private key $\mathbf{R}$. We will discuss the constraints on σ in Section 4.2.3.

Let us denote the function associated to the pair $(\mathbf{B}, \sigma)$ by $f_{\mathbf{B},\sigma}$. The domain of $f_{\mathbf{B},\sigma}$ consists of pairs of vectors $(\mathbf{v}, \mathbf{e}) \in \mathbb{Z}^n \times \mathbb{R}^n$ where $\|\mathbf{e}\| \leq \sigma$. These two vectors are chosen by the Sample algorithm. The task of the Generate algorithm is to construct the bases $\mathbf{B}$ with high dual orthogonality. Since it is supposed to be hard to find $\mathbf{R}$ with low dual orthogonality from $\mathbf{B}$, the way it is done is to construct $\mathbf{R}$ first and then derive $\mathbf{B}$ from it.

Generating a Private Basis

For constructing $\mathbf{R}$ there are two suggested distributions in [18] from which $\mathbf{R}$ can be chosen.

1. Setting $\mathcal{L}(\mathbf{R})$ to be a random lattice. That means to choose the entries of $\mathbf{R}$ independently and uniformly from the set $\{-l, \ldots, +l\}$ for some integer bound l. The experimental results show that the value of l has almost no effect on the value of dual orthogonality defect. Thus, l is chosen to be small for efficiency purposes.

2. Setting $\mathcal{L}(\mathbf{R}')$ to be a rectangular lattice. That means for some integer k, $\mathbf{R} = k\mathbf{I} + \mathbf{R}'$ where $\mathbf{R}'$ is constructed as explained in 1.

This way we get a basis $\mathbf{R}$ with dual orthogonality r.

Generating Public Key

Once the private key in chosen, the public basis should be another basis for the same lattice. Choosing another basis for $\mathcal{L}(\mathbf{R})$ is equivalent, by Proposition 2.3.3, to finding a transformation $T \in SL_n(\mathbb{Z})$. Thus, we are interested in finding a random element in $SL_n(\mathbb{Z})$. Two methods are suggested in [18].

First method: Consider matrices of the form

$$\mathbf{A}_i = \begin{pmatrix} 1 & & & \star & & & \\ & \ddots & & \vdots & & & \\ & & 1 & \star & & & \\ & & & 1 & & & \\ & & & \star & 1 & & \\ & & & \vdots & & \ddots & \\ & & & \star & & & 1 \end{pmatrix}$$

or

$$\mathbf{B}_i = \begin{pmatrix} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ \star & \ldots & \star & 1 & \star & \ldots & \star & \\ & & & & 1 & & & \\ & & & & & \ddots & & \\ & & & & & & 1 & \end{pmatrix}$$

where the blanks represent zeros and $\star$'s are any integer in ith row or column.

Second method: Consider upper (U) and lower (L) triangular matrices of determinant one with ± 1 on the diagonal.

Lemma 4.2.1. *The set of all possible matrices produced by either method is all of* $SL_n(\mathbb{Z})$.

Proof. Observe that $\det(A_i) = \det(B_i) = 1$ which guarantees that they, as well as the product of any number of them are in $SL_n(\mathbb{Z})$. In other words, the multiplicative group generated by such matrices is a subgroup of $SL_n(\mathbb{Z})$. Similarly, the group generated by upper (lower) triangular matrices of determinant one is a subgroup of $SL_n(\mathbb{Z})$. To prove that one can construct any matrix $\mathbf{T} \in SL_n(\mathbb{Z})$ by multiplying such matrices together, note that any matrix $\mathbf{T}$ in $SL_n(\mathbb{Z})$ is reducible to $\mathbf{I}$ and vice versa by elementary column operation of adding a multiple of one column to another. That is because obtaining an entry equal to one in each row and interchanging two columns

(up to a sign) can be achieve by elementary column operations. These elementary column operations correspond to multiplication by elementary matrices. That is, matrices which defer from $\mathbf{I}$ just by one entry [27]. On the other hand, observe that the sets of matrices in both methods contain all elementary matrices. That proves that they generate all $SL_n(\mathbb{Z})$. □

Given a private basis $\mathbf{R}$ with dual orthogonality defect r, one generates i random matrices according to either method. Multiplying them together gives a transformation $\mathbf{T}$. Then, one sets $\mathbf{B} = \mathbf{TR}$. The number i is determined by experiment.

Now the question is whether $\mathbf{R}$ multiplied by any $\mathbf{T}$, constructed using either of the above methods, would serve us a public basis $\mathbf{B}$. Note that we would like our $\mathbf{B}$ to be a basis of $\mathcal{L}$ with high dual orthogonality defect in comparison with $\mathbf{R}$. Moreover, we do not want the adversary to be able to recover our $\mathbf{R}$ from $\mathbf{B}$. As we saw in Chapter 2, given a basis for a lattice, one can use the LLL algorithm to obtain a reduced basis. Since the reduced basis is smaller than the original basis, it has lower orthogonality defect. Similarly, one can get a basis with smaller dual orthogonality defect by reducing the transpose inverse matrix. Therefore, given the public basis $\mathbf{B}$, the attacker can always perform a lattice reduction algorithm to gain a better basis in terms of dual orthogonality.

Starting with $\mathbf{R}$, each time we add a multiple of a row to another row, we expect to get a matrix with bigger entries. That implies that we expect the inverse also have bigger entries and bigger product of the norms of its columns. So, after each step we get a basis for the lattice $\mathcal{L}$ with higher orthogonality defect than r. We need to add enough multiples of the other columns to each columns of $\mathbf{R}$ to 'sufficiently mix' it. By 'sufficiently mixing' the private key, we mean to make sure that even after reduction by lattice reduction algorithms, the result basis still has large dual orthogonality defect in comparison to $\mathbf{R}$ and all the known reduction algorithms fail recovering $\mathbf{R}$.

The number i of required mixing steps suggested in [18] is $2n$ which is based on experimental results over $\mathbf{A}_i$s where the $\star$s are chosen from the set $\{1, 0, -1\}$ to avoid the large size of $\mathbf{B}$ for implementation purposes. Observe that since raising an $\mathbf{A}_i$ ($\mathbf{B}_i$) to the power of k would give us a similar matrix with k times the ith row (column),

Without loss of generality, we can choose the $\star$s from the set $\{1, 0, -1\}$.

Another provably better way to generate the public key $\mathbf{B}$ is suggested by Micciancio in 2002 [34]. He explains a technique of using *Hermite normal forms* to generate the public key $\mathbf{B}$ given the private key. This technique is not limited to the GGH cryptosystem and can be applied to all the lattice-based cryptosystems where the public key is a lattice basis and the ciphertext is a vector close to a lattice.

4.2.2 Sample and Evaluate Algorithms

Suppose that we have the output of the Generate algorithm $(f_{\mathbf{B},\sigma}, \mathbf{R})$. Given $f_{\mathbf{B},\sigma}$, the Sample algorithm outputs an element in the domain of $f_{\mathbf{B},\sigma}$. This element consists of the pair $(\mathbf{v}, \mathbf{e}) \in (\mathbb{Z}^n, \mathbb{R}^n)$. The vector $\mathbf{v}$ is chosen in such a way that the corresponding lattice point $\mathbf{vB}$ looks random. In [18] it is suggested to pick each entry of $\mathbf{v}$, independently, uniformly from the range $\{-n^2, -n^2+1, \ldots, n^2\}$. The vector $\mathbf{e}$ is chosen at random from $\mathbb{R}^n$ such that each of its entries has mean zero and variance σ^2. For example each entry can be picked as $\pm\sigma$ each with probability $1/2$.

Given $\mathbf{B}, \sigma, \mathbf{v}$ and $\mathbf{e}$, the Evaluate algorithm computes the value of $f_{\mathbf{B},\sigma}$ at sample points given by the Sample algorithm.

$$\mathbf{c} = f_{\mathbf{B},\sigma}(\mathbf{v}, \mathbf{e}) = \mathbf{vB} + \mathbf{e} \in \mathbb{R}^n$$

We refer to $\mathbf{c}$ and $\mathbf{v}$ as ciphertext and plaintext respectively.

4.2.3 Invert Algorithm

Given the private key $\mathbf{R}$ and the ciphertext $\mathbf{c}$, the Invert algorithm recovers the pair $(\mathbf{v}, \mathbf{e})$. The suggested method in [18] is using the Babai round-off algorithm (see Section 2.4.2). However, other approximation CVP algorithms such as nearest plane can also be used. The Babai algorithm represents $\mathbf{c}$ as a linear combination of the columns of $\mathbf{R}$. Then, it rounds the coefficient in the linear combination to the nearest integer so the result is a lattice point. Next, it recovers $\mathbf{v}$ by representing that lattice point as linear combination of the basis $\mathbf{B}$. Defining $\mathbf{T}$ to be $\mathbf{RB}^{-1}$, the algorithm steps are as follows:

- Recover $\mathbf{v}$ as $\lfloor \mathbf{c}\mathbf{R}^{-1} \rceil \mathbf{T}$.
- Recover $\mathbf{e}$ as $\mathbf{c} - \mathbf{v}\mathbf{B}$.

If the output of the Invert algorithm is the same as input of f, we say the Invert algorithm performed successfully. Otherwise, we say an *inversion error* has occurred.

Lemma 4.2.2. *Let* $\mathbf{R}$ *be the private basis used in the inversion of* $f_{B,\sigma}(\mathbf{v}, \mathbf{e})$*, then an inversion error occurs if and only if* $\lfloor \mathbf{e}\mathbf{R}^{-1} \rceil \neq 0$.

Proof.

$$\begin{aligned} \lfloor \mathbf{c}\mathbf{R}^{-1} \rceil \mathbf{T} &= \lfloor (\mathbf{v}\mathbf{B} + \mathbf{e})\mathbf{R}^{-1} \rceil \mathbf{T} \\ &= \lfloor \mathbf{v}\mathbf{B}\mathbf{R}^{-1} + \mathbf{e}\mathbf{R}^{-1} \rceil \mathbf{T} \\ &= \lfloor \mathbf{v}\mathbf{T}^{-1} + \mathbf{e}\mathbf{R}^{-1} \rceil \mathbf{T} \end{aligned}$$

Since $\mathbf{T}$ is an integral matrix with determinant 1, $\mathbf{T}^{-1}$ is also integral. Therefore, the result of multiplying the integral vector $\mathbf{v}$ by $\mathbf{T}^{-1}$ is an integral vector. It follows that

$$\begin{aligned} \lfloor \mathbf{v}\mathbf{T}^{-1} + \mathbf{e}\mathbf{R}^{-1} \rceil \mathbf{T} &= \mathbf{v}\mathbf{T}^{-1}\mathbf{T} + \lfloor \mathbf{e}\mathbf{R}^{-1} \rceil \mathbf{T} \\ &= \mathbf{v} + \lfloor \mathbf{e}\mathbf{R}^{-1} \rceil \mathbf{T}. \end{aligned} \tag{13}$$

Hence, the invert algorithm performs successfully if and only if $\lfloor \mathbf{e}\mathbf{R}^{-1} \rceil \mathbf{T} = 0$, which happens if and only if $\lfloor \mathbf{e}\mathbf{R}^{-1} \rceil = 0$. □

The following two theorems give restrictions on σ under which the Invert algorithm is successful. In both of them, it is assumed that the entries of $\mathbf{e}$ are chosen equal to $\pm\sigma$ each with probability $1/2$ for the sake of simplicity.

Theorem 4.2.3. *Let* $\mathbf{R}$ *be the private basis used in the inversion of* $f_{B,\sigma}(\mathbf{v}, \mathbf{e})$*, and denote the maximum* L_1 *norm of the rows in* $(\mathbf{R}^{-1})^T$ *by* ρ*. Then as long as* $\sigma < 1/(2\rho)$*, no inversion error can occur.*

Proof. Let $\mathbf{e} = (\epsilon_1, \ldots, \epsilon_n)$ and set

$$\mathbf{d} = (\delta_1, \ldots, \delta_n) = \mathbf{e}\mathbf{R}^{-1}.$$

By Lemma 4.2.2, we have an inversion error only when $\lfloor \mathbf{e}\mathbf{R}^{-1} \rceil \neq 0$ which means that $|\delta_i| \geq \frac{1}{2}$ for some i. Let ρ_{ij} denote the the (i,j)th entry of $(\mathbf{R}^{-1})^T$. Since we assumed that all the entries of $\mathbf{e}$ are $\pm\sigma$ we have

$$\begin{aligned} |\delta_i| &= |\mathbf{e} \cdot \mathbf{r}_i^*| \\ &= |\sum_j \rho_{ij}\epsilon_j| \\ &\leq \sigma \sum_j |\rho_{ij}| \\ &\leq \sigma\rho \end{aligned}$$

This implies that as long as $\sigma < 1/(2\rho)$, $|\delta_i| < \frac{1}{2}$ for every i. So, no inversion error can occur. □

Remark 4.2.4. By the above theorem, to successfully decrypt the message, one can use any basis of $\mathcal{L}$ in place of the trapdoor $\mathbf{R}$ as long as $\rho < 1/(2\sigma)$ holds.

This theorem gives a sufficient condition to get an error free Invert algorithm. Observe that higher values of σ might cause errors in inversion. On the other hand, make it harder for the closest vector approximation algorithm to recover $\mathbf{v}$ without having the trapdoor information. So, in order to have a reasonable balance between the security and reliability, in practice the value of σ satisfies a looser inequality. For the proof of the following theorem see [18].

Theorem 4.2.5. *Let $\mathbf{R}$ be a private basis used in the inversion of $f_{B,\sigma}(\mathbf{v},\mathbf{e})$, and denote the maximum L_∞ norm of the rows in $(\mathbf{R}^{-1})^T$ by $\frac{\gamma}{\sqrt{n}}$. Then the probability of inversion error is bounded by*

$$Pr[\textit{inversion error using } \mathbf{R}] \leq 2n \exp\left(-\frac{1}{8\sigma^2\gamma^2}\right)$$

In the proof of this theorem, it is assumed that the coefficients of $\mathbf{e}$ are independent and have uniform distribution.

So to have the error probability less than positive real number ϵ, we should have

$$\begin{aligned} 2n\frac{1}{\exp\left(\frac{1}{8\sigma^2\gamma^2}\right)} &< \epsilon \\ \sigma^2 &< \frac{1}{8\gamma^2 \ln\left(2n/\epsilon\right)} \\ \sigma &< \left(\gamma\sqrt{8\ln\left(2n/\epsilon\right)}\right)^{-1} \end{aligned}$$

For $n = 140$ and $\epsilon = 10^{-4}$ for example, the experiments in [18] resulted in $\sigma < 2.7$.

For a given σ, it follows from the proof of Theorem 4.2.3 that for an arbitrary basis $\mathbf{R}$ of $\mathcal{L}$, having L_1 norm of rows of $(\mathbf{R}^{-1})^T$ less than $1/(2\sigma)$ guarantees no inversion error while using $\mathbf{R}$ as trapdoor information. Using the norm inequality $\|\mathbf{x}\| \leq \|\mathbf{x}\|_1$, the dual orthogonality defect of such a basis is

$$\begin{aligned} \text{orth-defect}^*(\mathbf{R}) &= \frac{\prod_{i=1}^n \|\mathbf{r}_i^*\|}{\det(\mathbf{R}^{-1})} \\ &\leq \frac{\prod_{i=1}^n \|\mathbf{r}_i^*\|_1}{\det(\mathbf{R}^{-1})} \\ &\leq \rho^n \det(\mathbf{R}) \\ &\leq \frac{\det(\mathbf{R})}{(2\sigma)^n}. \end{aligned}$$

Moreover, denoting the maximum L_∞ norm of the rows in $(\mathbf{R}^{-1})^T$ by $\frac{\gamma}{\sqrt{n}}$, it follows from Theorem 4.2.5 that having $\gamma < \sqrt{(8\sigma^2 \ln 2n/\epsilon)^{-1}}$ guarantees the error probability to be less than ϵ. The dual orthogonality of such basis using the norm inequality $\|\mathbf{x}\|_2 \leq \sqrt{n}\|\mathbf{x}\|_\infty$ is

$$\begin{aligned} \text{orth-defect}^*(\mathbf{R}) &= \frac{\prod_{i=1}^n \|\mathbf{r}_i^*\|}{\det(\mathbf{R}^{-1})} \\ &\leq \frac{(\sqrt{n})^n \prod_{i=1}^n \|\mathbf{r}_i^*\|_\infty}{\det(\mathbf{R}^{-1})} \\ &\leq \det(\mathbf{R}) \left(\sqrt{n}^n (\frac{\gamma}{\sqrt{n}})^n\right) \\ &\leq \det(\mathbf{R})\gamma^n \\ &\leq \det(\mathbf{R})\sqrt{(8\sigma^2 \ln\left(2n/\epsilon\right))^{-n}}. \end{aligned}$$

Thus, the upper bounds given by these theorems imply upper bounds for the dual orthogonality of the basis. Therefore, one can summarize that a basis $\mathbf{R}$ that has

low enough dual orthogonality can serve us as a good basis for the Invert algorithm. How low that dual orthogonality should be depends on σ. If an attacker can find a basis with low enough dual orthogonality, he/she can use that to Invert the trapdoor function and recover the input. That is the reason why we want our public basis to have high enough dual orthogonality so that deriving a good basis from that using the known techniques is infeasible.

4.2.4 Security of the GGH One-way Trapdoor Function

Suppose that an adversary having $\mathbf{e}$ and the public basis $\mathbf{B}$ wants to recover $\mathbf{v}$. We can divide the attacks into two categories based on the approach of the attacker. The attacker either attempts to invert the $f_{\mathbf{B},\sigma}$ function without having the trapdoor information, or tries to calculate the trapdoor information $\mathbf{R}$ or an alternative for it. The former implies solving the CVP or coming up with a good approximation for it. As it is discussed in Chapter 2, approximation of CVP within a polynomial factor is known to be a hard problem. In the latter case, the attacker has to reduce $(\mathbf{B}^{-1})^T$ using a lattice reduction algorithm in order to find a basis for the dual lattice with smaller norm product and lower orthogonality defect as a result. Thus, the attacker needs to solve SBP in the dual lattice. Similarly to SVP, the approaches to approximate SBP are via lattice reduction algorithm such as LLL. One can argue that in the latter case the attacker needs to solve SVP. Note that given any instance of the GGH one-way trapdoor function, the security of the function is based on the hardness of the problem in that particular lattice lying underneath the function. In fact, as it will be described in Section 4.3.3, the actual instance that the attacker should solve is easier than in seems in the first sight.

Therefore, any good approximation of one of the above problems might lead to a successful attack. In order to avoid such attacks, the dimension of the lattice should be chosen large enough to make these problems presumably infeasible to approximate using currently known algorithms. The original suggested value for n in [18] was 140 based on experimental observations. However, the GGH instances, up to dimension 400 are broken [38].

Brute Force Attack

In [18] the brute force attack is described in order to give a size for the exhaustive search space and also to show its dependency on the dual orthogonality defect of the basis used in the attack. This attack can be placed under the first category described above. Since having $\mathbf{B}$, it is hard to find another basis with remarkable low dual orthogonality defect, we may assume that the attacker is working with $\mathbf{B}$.

Lemma 4.2.6. *The size of the search space to find the error vector* $\mathbf{v}$ *using the public basis* $\mathbf{B}$ *is*

$$(\pi e)^{n/2} \cdot \sigma^n \cdot \prod_i \|\mathbf{b}_i^*\|$$

Proof. Given $\mathbf{c}$ and $\mathbf{B}$ one computes

$$\mathbf{c}\mathbf{B}^{-1} = \mathbf{v} + \mathbf{e}\mathbf{B}^{-1}.$$

Then one needs to search for $\mathbf{e}\mathbf{B}^{-1}$. Denote $\mathbf{d} = \mathbf{e}\mathbf{B}^{-1} = (\delta_1, \ldots, \delta_n)$ and $\mathbf{e} = (\epsilon_1, \ldots, \epsilon_n)$ and the (i,j)th element of $\mathbf{B}^{-1}$ by β_{ij} . Let us assume that the entries of $\mathbf{d}$ are independent random variables distributed normally. Then the mean of $\delta_i = \sum_j \epsilon_j \beta_{ji}$ is zero and the variance is

$$\begin{aligned} Var(\delta_i) &= Var\left(\sum_j \epsilon_j \beta_{ji}\right) \\ &= \sum_j \beta_{ji}^2 E(\epsilon_j^2) \\ &= (\sigma \|\mathbf{b}_i^*\|)^2 \end{aligned}$$

The size of the search space is 2 to the power of the differential entropy of the vector $\mathbf{d}$ [18]. The *differential entropy* of a normal random variable is

$$h(\mathbf{d}_i) = \frac{1}{2}\log(\pi e \sigma^2).$$

Hence,

$$\begin{aligned} h(\mathbf{d}) &= \frac{1}{2}\sum_i \log(\pi e \sigma^2 \|\mathbf{b}_i^*\|^2) \\ &= \frac{n}{2}\log(\pi e \sigma^2) + \sum_i \log \|\mathbf{b}_i^*\|. \end{aligned}$$

So the size of the search space is

$$2^{h(\mathbf{d})} = (\pi e)^{n/2}\sigma^n \prod_i \|\mathbf{b}_i^*\|.$$

□

This lemma illustrates the relation between the security of the function f and the dual orthogonality of the basis being used for brute force attack.

4.3 GGH Encryption Scheme

Based on the one way trapdoor function described in previous section, an encryption scheme is introduced in [18]. The core of the encryption scheme is the same as the trapdoor function. In the context of encryption, a lattice vector is encrypted to a ciphertext, which is a vector in $\mathbb{R}^n$, by adding a small perturbation vector to it. To encrypt an arbitrary message in a given message space, the message should map to an integral n-tuple using an easily invertible map M [18]. Using the same notation as the trapdoor function, here we describe the encryption scheme.

4.3.1 Key Generation

Bob generates the private and public keys as follows.

Key Generation

1. Take the security parameter n as an input.
2. Run the Generate algorithm to get $(\mathbf{B}, \mathbf{R}, \sigma)$.
3. Set the public key to be $(\mathbf{B}, \sigma)$.
4. Set the private key to be $(\mathbf{R}^{-1}, T)$, where $T = \mathbf{B}^{-1}\mathbf{R}$.

4.3.2 Encryption and Decryption

Encryption

Alice, having Bob's public key $(\mathbf{B}, \sigma)$, wants to send the message $\mathbf{v} \in \mathbb{Z}^n$ to Bob.

1. She picks the perturbation vector $\mathbf{e}$ by running the Sample algorithm.
2. She runs the Evaluate algorithm with inputs $\mathbf{v}$ and $\mathbf{e}$.
3. She sends the output $\mathbf{c}$ to Bob as ciphertext.

Decryption

1. Bob having the private key $\mathbf{R}$ receives $\mathbf{c}$.
2. He runs the Invert algorithm and recovers $\mathbf{v}$.

Remark 4.3.1. Notice that the only additional feature of the encryption scheme is in the case that the message space in not a subset of the domain of f. In that case, an easily invertible public map should be used from the message space to $\mathbb{Z}^n$.

4.3.3 A Successful Attack Against GGH

According to the experimental evidences and as a result of security argument described earlier, arguing that the attacker needs to either solve CVP or SBP, the authors of [18], posted five challenges of ciphertexts together with the corresponding public information on web. These challenges were of dimensions 200, 250, 300 and 400. In 1999 an attack is described in [38] that breaks all the challenges except for dimension 400.

This attack takes advantage of the special form of the error vector in the GGH trapdoor function. More precisely, since the entries of $\mathbf{e}$ are chosen equal to $\pm\sigma$, one can write

$$\mathbf{c} + (\sigma, \ldots, \sigma) \equiv \mathbf{vB} \mod 2\sigma \tag{14}$$

Solving this modular system, one can get the plaintext $\mod 2\sigma$, denoted by $\mathbf{v}_{2\sigma} \in \mathbb{Z}^n$. It is shown in [38] that the system (14) has very few solutions with high probability, and these solutions are easy to compute.

Knowing $\mathbf{v}_{2\sigma}$, one can simplify the problem of finding a closest vector of $\mathcal{L}(\mathbf{B})$ to the target vector $\mathbf{c} = \mathbf{v}\mathbf{B} + \mathbf{e}$ to finding a closest vector in $\mathcal{L}(\mathbf{B})$ to $\frac{\mathbf{c}-\mathbf{v}_{2\sigma}}{2\sigma} = \mathbf{v}'\mathbf{B} + \frac{\mathbf{e}}{2\sigma}$, where $\mathbf{v}' = \frac{\mathbf{v}-\mathbf{v}_{2\sigma}}{2\sigma}$.

Now, since the error vector $\frac{\mathbf{e}}{2\sigma}$ is considerably smaller than $\mathbf{e}$, it is possible to solve this instance of problem using known techniques to solve approximate CVP. However, the running time of these algorithms still increases exponentially in the dimension of the lattice which makes it impossible to solve the problem for very large dimensions. The challenge of dimension 400 remains unsolve in [38].

In the same paper, it is suggested that to repair this weakness, one can choose the entries of $\mathbf{e}$ from the set $\{\pm\sigma, \pm(\sigma - 1)\}$. However, it is noted that the special form of the vector would still be of concern. In conclusion, GGH scheme, as originally described, cannot be reasonably secure without being impractical.

4.4 Summary

The GGH cryptosystem is based on a trapdoor function $f_{\mathbf{B},\sigma} : (\mathbb{Z}^n \times \mathbb{R}^n) \to \mathbb{R}^n$. The public key consists of a basis $\mathbf{B}$ for the lattice $\mathcal{L}$ together with a real parameter σ. The message space is a set together with an additional public, easily invertible map into $\mathbb{Z}^n$. The public key is a basis $\mathbf{B}$ for $\mathcal{L}$. Given $(\mathbf{v}, \mathbf{e}) \in (\mathbb{Z}^n \times \mathbb{R}^n)$, $f_{\mathbf{B},\sigma}(\mathbf{v}, \mathbf{e})$ is $\mathbf{v}\mathbf{B} + \mathbf{e}$.

Computing $f_{\mathbf{B},\sigma}$ is an easy operation. However, computing the inverse is the CVP. While $\mathbf{B}$ is chosen such that the CVP approximation algorithms perform poorly over $\mathbf{B}$, the same algorithms are effective over $\mathbf{R}$. The reason lies in the way these bases are chosen. First, the basis $\mathbf{R}$ is constructed and $\mathcal{L}$ is the lattice generated by $\mathbf{R}$. The value σ, which bounds the entries of $\mathbf{e}$, is chosen such that $\mathbf{R}$ can recover $\mathbf{v}$. Then, $\mathbf{B}$ is obtained by mixing $\mathbf{R}$ thoroughly via multiplication by $2n$ random elements of $SL_n(\mathbb{Z})$ so that one cannot expect to recover $\mathbf{R}$ given $\mathbf{B}$, using any known algorithm.

The main attack against GGH would be either using lattice reduction algorithms

to recover $\mathbf{R}$ from $\mathbf{B}$ or coming up with a better CVP approximation algorithm which allows finding $\mathbf{v}$ having $\mathbf{B}$. A necessary condition to avoid these attacks is to choose the lattice dimension to be chosen large enough. As it is shown in [38], GGH with lattice of dimension less than 400 is not secure. Thus, GGH with reasonable level of security has such large keys that it makes it impractical. So, to improve GGH to a practical alternative for current public key cryptosystems, one should make GGH more efficient and secure.

Chapter 5

The NTRU Cryptosystem

5.1 Introduction

The NTRU system is a public key cryptosystem introduced in 1996 by J. Hoffstein, J. Pipher and J.H. Silverman at a rump session and latter on described in [19]. Encryption and decryption in NTRU involve computation in the ring $\mathbb{Z}[X]/(X^N - 1)$ and reduction modulo two positive integers p and q (not necessarily prime). The encryption process uses random polynomials, so that each message has many possible encryptions. This structure makes NTRU fit into the general category of probabilistic cryptosystems. The decryption is valid in the sense that with high probability, the decryptor would recover the message using the decryption algorithm.

Unlike the other lattice-based cryptosystems, the encryption scheme in NTRU does not demonstrate the notion of a lattice at first sight. However, the security of the NTRU public key cryptosystem ultimately rests on the inability of the LLL algorithm, or any of its variants, to produce particularly good short vectors of a given lattice in a reasonable amount of time. Through this connection, NTRU fits in the category of lattice based cryptosystems.

The main advantages of NTRU, compared to other public key cryptosystems, are its higher speed of encryption and decryption in addition to the speed and simplicity of the key generation algorithm. However, the existence of decryption failure, which has led to some attacks, places NTRU under the category of imperfect cryptosystems.

In this chapter, we describe the NTRU cryptosystem in the general framework of one-way trapdoor function. However, the hardness of the Invert algorithm (without knowing the trapdoor information) is unknown. Also one can argue that the Invert algorithm does not fully recover the input of the one-way trapdoor functions. Materials of this chapter are mainly taken from [19] and [11]. The proofs which are included are our own.

5.2 Notations and Definitions

Let N, p and q be relatively prime positive integers where q is large compared to p. Set $R = \mathbb{Z}[X]/(X^N - 1)$, with multiplication of two elements f and g of R denoted by $f * g$. Since the elements of R are polynomials of degree less than N, they can be represented as vectors. We shall use the following representation of $f \in R$ interchangeably when there is no possibility of confusion.

$$f \doteq \sum_{i=0}^{N-1} f_i X^i \doteq (f_0, f_1, \ldots, f_{N-1}).$$

By reducing an element of R modulo p or q, we mean reducing of its coefficients modulo p or q.

Lemma 5.2.1. *A polynomial f in R is invertible* mod q *if and only if* $\gcd(f, X^N - 1) \equiv 1 \mod q$.

Proof. Suppose that f has an inverse A in $R \mod q$. That means $f * A \in 1 + < X^N - 1 > \mod q$. Thus there exist a $U \in \mathbb{Z}[X]$ such that $f * A \equiv 1 + U(X^N - 1) \mod q$ which implies $f * A - U(X^N - 1) = qk + 1$ for some $k \in \mathbb{Z}[X]$. So by definition, the gcd of f and $X^N - 1$ is $1 \mod q$. Conversely, if the gcd is 1, such U and A exist and therefore A is the inverse of f in $R \mod q$. □

As it will be explained in detail in the next section, the message, private key and the initial polynomials used in public key generation are all polynomials in R, chosen from appropriate sets. The following definitions help us introduce the system parameters and their constraints.

Definition 5.2.2. Let d_1 and d_2 be two positive integers. Define $\mathcal{L}(d_1, d_2)$ to be the set of all $f \in R$ such that f has d_1 coefficients equal 1, d_2 coefficients equal -1 and the rest 0.

Remark 5.2.3. Despite the similar notation, $\mathcal{L}(d_1, d_2)$ is not related to lattices.

Lemma 5.2.4. *The set* $\mathcal{L}(d_1, d_2)$ *has* $\binom{N}{d_1}\binom{N-d_1}{d_2}$ *elements.*

Proof. Elements of the set are polynomials with degree less than N, so they have N coefficients d_1 of which are chosen to be 1 and among the $N - d_1$ remainders d_2 are chosen to be -1. □

Definition 5.2.5. The *width* of an element $f \in R$ is defined to be

$$|f|_\infty = \max_{1 \le i \le N} \{f_i\} - \min_{1 \le i \le N} \{f_i\}.$$

Definition 5.2.6. *The centered* $L_\perp$ *norm* on R is defined by

$$|f|_\perp = \sqrt{\sum_{i=0}^{N-1} {f_i}^2 - \frac{1}{N}\left(\sum_{i=0}^{N-1} f_i\right)^2}.$$

The norm is imposed componentwise on $(f, g) \in R^2$: $|(f, g)|_\perp = |f|_\perp + |g|_\perp$. Note that $|f|_\perp$ is the standard deviation of the entries of f, times $\sqrt{N}$.

Despite the notations $|\ \ |_\infty$ and $|\ \ |_\perp$, the width and the centered norm are not norms. That is because width and centered norm of a nonzero $f = t\mathbf{1}$, where $\mathbf{1}$ is the all one vector, is zero.

Lemma 5.2.7. *If the entries of* g *have zero mean, the centered norm is the same as the* L_2 *norm.*

Proof. Having zero mean implies that $1/N \sum_i f_i = 0$ so

$$\begin{aligned} |f|_\perp &= \sqrt{\sum_{i=0}^{N-1} f_i^2 - 0} \\ &= \|f\|_2. \end{aligned}$$

□

Lemma 5.2.8. $|f|_\perp = (\sum_{i=0}^{N-1}(f_i - \bar{f})^2)^{1/2}$ *where* $\bar{f} = \frac{1}{N}(\sum_{i=0}^{N-1} f_i)$.

Proof.

$$\begin{aligned}
\sum_i (f_i - \bar{f})^2 &= \sum_i \left(f_i - \frac{1}{N}(\sum_i f_i) \right)^2 \\
&= \sum_i \left(f_i^2 - \frac{2}{N} \sum_j f_j f_i + \frac{1}{N^2} \sum_{j,k} f_j f_k \right) \\
&= \sum_i {f_i}^2 - \frac{2}{N} \sum_{i,j} f_i f_j + \frac{N}{N^2} \sum_{j,k} f_j f_k \\
&= \sum_i {f_i}^2 - \frac{1}{N} \sum_{i,j} f_i f_j \\
&= \sum_i {f_i}^2 - \frac{1}{N} \left(\sum_i f_i \right)^2 \\
&= |f|_\perp^2.
\end{aligned}$$

□

Lemma 5.2.9. $|f|_\perp$ *is the* L_2 *norm of* $f - Proj_{\mathbf{1}}(f)$, *where* $Proj_{\mathbf{1}}(f)$ *is the projection of* f *on the all one vector* $\mathbf{1}$.

Proof.

$$\begin{aligned}
\|f - \mathrm{Proj}_{\mathbf{1}}(f)\|_2 &= \left\| f - \frac{f \cdot \mathbf{1}}{\mathbf{1} \cdot \mathbf{1}} \mathbf{1} \right\|_2 \\
&= \left\| f - \frac{\sum_{i=1}^{N-1} f_i}{N} \mathbf{1} \right\|_2 \\
&= \left((f_0 - \frac{1}{N} \sum_{i=1}^{N-1} f_i)^2 + \cdots + (f_{N-1} - \frac{1}{N} \sum_{i=1}^{N-1} f_i)^2 \right)^{1/2} \\
&= \sqrt{\sum_i (f_i - \bar{f})^2} \\
&= |f|_\perp \qquad \text{by Lemma 5.2.8.}
\end{aligned}$$

□

Lemma 5.2.10. *$|f|_\perp$ is invariant under the operation of adding the vector $t\mathbf{1}$.*

Proof. Since $\mathrm{Proj}_1(t\mathbf{1}) = t\mathbf{1}$, we have

$$\begin{aligned} |f + t\mathbf{1}|_\perp &= \|f + t\mathbf{1} - \mathrm{Proj}_1(f + t\mathbf{1})\|_2 \\ &= \|f - \mathrm{Proj}_1(f)\|_2 \\ &= |f|_\perp. \end{aligned}$$

□

The centered norm is quasi-multiplicative [11]. That means

$$|F * G|_\perp = |F|_\perp |G|_\perp + A,$$

where the component A is smaller by a factor bigger than $1/N$ than the first component. We write

$$|F * G|_\perp \approx |F|_\perp |G|_\perp. \tag{15}$$

On the relation between width and centered norm, the following proposition was stated in [19] without proof.

Proposition 5.2.11. *For any $\epsilon > 0$ there are constants $\gamma_1, \gamma_2 > 0$, depending on ϵ and N, such that for randomly chosen polynomials $f, g \in R$, the probability is greater than $1 - \epsilon$ that they satisfy*

$$\gamma_1 |f|_\perp |g|_\perp \leq |f * g|_\infty \leq \gamma_2 |f|_\perp |g|_\perp$$

5.3 NTRU One-way Trapdoor Function

The trapdoor function depends on six positive integer parameters (N, p, q, d_f, d_g, d), where p and q are relatively prime, q is typically a power of 2 which is considerably bigger than p and d, d_g and d_f are positive integers less than $N/2$. Different sets of parameters offer different levels of speed and security. The current version of NTRU uses the standard parameter set [10] $N = 251$, $q = 239$, $p = 2$. $d_f = d_g = d = 72$.

5.3.1 Generate Algorithm

Given a set of parameters, one generates the keys using the following key creation algorithm.

- Choose two polynomials g, f randomly from the sets $\mathcal{L}_g = \mathcal{L}(d_g, d_g)$ and $\mathcal{L}_f = \mathcal{L}(d_f, d_f - 1)$ respectively (see Definition 5.2.2). The polynomial f, to which we refer as private key, should be chosen in such a way that it has inverse modulo both p and q.

- Compute the inverses of f modulo p and q with the use of the extended Euclidean algorithm. Denote them by F_p and F_q respectively.

- Compute the quantity

$$h \equiv F_q * g \mod q \tag{16}$$

  We refer to h as the public key.

- Set the function k to be

$$\begin{aligned} k : (R, R) &\rightarrow R \\ k(m, \phi) &= p\phi * h + m \mod q \end{aligned}$$

- Set the trapdoor information to be f.

- Output the pair (k, f).

By Lemma 5.2.1, for f to have inverses modulo both p and q, it should be relatively prime to $X^N - 1$. In particular, since $X - 1$ is an obvious factor of $X^N - 1$, f should not have 1 as a root. That is the reason why the parameters d_1 and d_2 in the definition of $\mathcal{L}_f$ are not equal as they are for $\mathcal{L}_g$.

So, the polynomials f and g completely determine the one-way trapdoor function and the trapdoor information.

5.3.2 Sample and Evaluate Algorithm

Given the function k of the output of the Generate algorithm together with the parameters, the sample algorithm chooses the input (m, ϕ) in D_k. If p is odd, the message set $\mathcal{L}_m$ is defined as

$$\mathcal{L}_m = \left\{ m \in R : \frac{-1}{2}(p-1) \leq m_i \leq \frac{1}{2}(p-1) \quad \forall i = 0, \ldots, N-1 \right\}; \tag{17}$$

If $p = 2$, then $\mathcal{L}_m$ is instead the set of binary polynomials in R. The sample algorithm chooses random polynomials m and ϕ from $\mathcal{L}_m$ and $\mathcal{L}_\phi = \mathcal{L}(d, d)$ respectively. The Evaluate algorithm computes k at the sample points given by the sample algorithm.

$$e = k(m, \phi) \equiv p\phi * h + m \mod q.$$

We refer to e and m as ciphertext and plaintext respectively.

5.3.3 Invert Algorithm

Given the trapdoor information f and the ciphertext e, the invert algorithm works as follows:

- Compute the quantity

$$\begin{aligned} \bar{a} &\equiv f * e \mod q \\ &\equiv f * p\phi * h + f * m \mod q \\ &= f * p\phi * F_q * g + f * m \mod q \qquad \text{from (16)} \\ &= p\phi * g + f * m \mod q \end{aligned}$$

  Identify $\bar{a}$ with an element of R all of whose coefficients lie in the interval $(-q/2, q/2]$. This is the key step is the invert algorithm in the sense that lack of certain conditions, as we will explain shortly, leads to decryption failure.

- Recover the plaintext as

$$\begin{aligned} F_p * \bar{a} &\equiv pF_p * \phi * g + F_p * f * m \mod p \qquad (18) \\ &= m' \qquad (19) \end{aligned}$$

Note that by reducing the coefficient mod p the first component in (18) vanishes because it is a multiple of p and in the second component F_p and f cancel each other out modulo p.

If the output of the Invert algorithm is the same as the plaintext, we say that the invert algorithm performed successfully. Otherwise, we say an inversion error (decryption failure) has occurred. Let us check what happens in the process of recovering the plaintext.

Remark 5.3.1. Notice that the sample algorithm gives a random polynomial ϕ for each message m. So, for the same message m, there are $\mathcal{L}_\phi$ number of possible ciphertexts. Moreover, the polynomial ϕ is not recovered by the Invert algorithm.

Inversion Criteria

Observe that if, after reduction modulo q, the polynomial $\bar{a}$ in (18) is equal to the (non-modular) polynomial

$$a = p\phi * g + f * m,$$

then the result of multiplication by F_p modulo p is the exact polynomial m. Otherwise, the output m' of the algorithm would not be equal to m. Note that by the choice of m (see equation (17)), it would not be changed under reduction modulo p.

So, $\bar{a} = a$ if and only if all the coefficients of a lie in the interval $(-q/2, q/2]$. So, reduction of $\bar{a}$ modulo q would give us a. Therefore, in order to have $a = \bar{a}$, it is necessary to have

$$|f * m + p\phi * g|_\infty < q. \tag{20}$$

We use this restriction to get a criterion for choosing the parameters. It is claimed in [19] that (20) 'virtually always' holds if we have

$$|f * m|_\infty \leq q/4 \qquad \textit{and} \qquad |p\phi * g|_\infty \leq q/4. \tag{21}$$

There are two possible reasons for the Invert algorithm to fail. Either a does not satisfy (20) or it does satisfy (20) but not all of its coefficients lie in $(-q/2, q/2]$. In the first case, we say that *a gap failure* has occurred. We would like to choose the parameters in such a way that the probability of gap failure is as low as possible. In

the second case, we say that *wrap failure* has occurred. Wrap failures can be fixed by shifting the coefficients or by reducing into an alternative interval $(-q/2+t, q/2+t)$ instead of $(-q/2, q/2)$ for an integer t [50, 51]. The following proposition describes the criterion, which is suggested in [19], too choose the set of parameters.

Proposition 5.3.2. *Let $\epsilon > 0$. The decryption process works successfully with probability greater that $1-\epsilon$ if the system parameters are chosen in such a way that*

$$|f|_\perp |m|_\perp \approx \frac{q}{4\gamma_2} \qquad \textit{and} \qquad |\phi|_\perp |g|_\perp \approx \frac{q}{4p\gamma_2} \tag{22}$$

Where γ_2 is obtain from Proposition 5.2.11.

Proof. We want to have (20). From Proposition 5.2.11 we have

$$|f * m|_\infty \leq \gamma_2 |f|_\perp |m|_\perp \quad \text{and} \quad |pQ * g|_\infty \leq \gamma_2 |pQ|_\perp |g|_\perp$$

Note that

$$|p\phi|_\perp = \left(\sum_{i=1}^{N} (p\phi_i - p\overline{\phi})^2\right)^{1/2} = p\left(\sum_{i=1}^{N} (\phi_i - \overline{\phi})^2\right)^{1/2} = p|\phi|_\perp.$$

So by choosing $|f|_\perp |m|_\perp \approx \frac{q}{4\gamma_2}$ and $|Q|_\perp |g|_\perp \approx \frac{q}{4p\gamma_2}$ for a γ_2 corresponding a small value of ϵ, we get the desired property, by equation (21). □

The problem with Proposition 5.2.11 is that it does not give any constructive way of computing γ_1 and γ_2. In [19], suggested appropriate values of γ_2 for $N = 107, 167$ and 503 are 0.35, 0.27, and 0.17 respectively, based on experimental data.

Here, using the centered $L_\perp$ norm, we describe another approach to get a criteria under which decryption is successful [11]. It follows from Lemma 5.2.9 that

$$\begin{aligned} |a|_\perp^2 &= |p\phi * g + f * m|_\perp^2 \\ &= \|p\phi * g + f * m - \mathrm{Proj}_1(p\phi * g + f * m)\|_2^2 \\ &= \|p\phi * g - \mathrm{Proj}_1(p\phi * g) + f * m - \mathrm{Proj}_1(f * m)\|_2^2. \end{aligned}$$

The author of [11] claim that, according to experimental evidences, we may assume $p\phi * g - \mathrm{Proj}_1(p\phi * g)$ and $f * m - \mathrm{Proj}_1(f * m)$ are nearly orthogonal, so that we may

assume

$$\begin{aligned} |a|_\perp^2 &\approx \|p\phi * g - \mathrm{Proj}_1(p\phi * g)\|_2^2 + \|f * m - \mathrm{Proj}_1(f * m)\|_2^2 \\ &= |p\phi * g|_\perp^2 + |f * m|_\perp^2 \end{aligned}$$

Using the approximation in (15), we have

$$|p\phi * g|_\perp \approx p|\phi|_\perp |g|_\perp.$$

Similarly

$$|f * m|_\perp \approx |f|_\perp |m|_\perp.$$

We consider $|m|_\perp$ and $|\phi|_\perp$ as norms of a typical element in $\mathcal{L}_\phi$ and $\mathcal{L}_m$ respectively. Hence

$$\begin{aligned} |a|_\perp^2 &\approx |p\phi * g|_\perp{}^2 + |f * m|_\perp^2 \\ &\approx p^2|g|_\perp^2|\phi|_\perp^2 + |f|_\perp^2|m|_\perp^2. \end{aligned}$$

We choose to write it as combination of $|g|_\perp^2$ and $|f|_\perp^2$ for later use in the next section:

$$|a|_\perp^2 \approx (p^2|\phi|_\perp^2)|g|_\perp^2 + (|m|_\perp^2)|f|_\perp^2. \tag{23}$$

We may assume that the entries of a are normally distributed with mean $\mu \approx 0$ and standard deviation $\sigma \approx |a|_\perp/\sqrt{N}$. In order to have successful decryption, we want to choose the system parameters in such a way that the coefficients of a lie between $-q/2$ and $q/2$ with high enough probability. That means we want the ratio of $q/2$ to σ to be big enough. The following failure probabilities are computed in [11] by use of density function of normal distribution, letting $q/2$ be $3, 4, 5$ and 6 times of σ.

$(q/2)/\sigma$	Individual failure $\rho = prob\|b_i\| > q/2$	Failure among 167 entries $1-(1-\rho)^N$
3	2.70×10^{-3}	3.63×10^{-1}
4	6.33×10^{-5}	1.05×10^{-2}
5	5.73×10^{-7}	9.57×10^{-5}
6	1.97×10^{-9}	3.30×10^{-7}

According to this result for $((q/2)/\sigma) \geq 5$, decryption process works with reasonable high probability. So we want to have

$$\sigma = |a|_{\perp}/\sqrt{N} < q/10,$$

which gives us a criterion for choosing the parameters q and N. Choosing a smaller value of σ and corresponding $|a|_{\perp}$ would give a lower failure probability.

The table below shows different sets of NTRU parameters. However, we should point out that these are not currently in use set of parameters.

Table 2. NTRU Parameter Sets.

	N	p	q	d_f	d_g	d
NTRU107.3	107	3	64	15	12	5
NTRU167.3	167	3	128	61	20	18
NTRU263.3	263	3	128	50	24	16
NTRU503.3	503	3	256	216	72	55
NTRU167.2	167	2	127	45	35	18
NTRU263.2	263	2	127	35	35	22
NTRU503.2	503	2	253	155	100	65

5.4 Security of NTRU One-way Trapdoor Function

An adversary has all the system parameters (N, q, p, d_f, d_g, d) as well as the public key h. He can eavesdrop through the channel and get the ciphertext e. Having this information, he is looking for either the particular message corresponding to the cipher text e or the polynomials g and f. Notice that f and g can be calculated from one another (see (16)). So it would be enough for an attacker to get one of these polynomials.

Suppose an attacker, having e and all the public information, wants to recover the message. Let us first observe what would happen if the attacker tries an alternative

key, say f' for decrypting the cipher text. He computes

$$\begin{aligned} \bar{a}' &= f' * e \\ &= f' * p\phi * h + f' * m \qquad \mod q. \end{aligned}$$

Then, he chooses a representative of $\bar{a}'$ with coefficients between $-q/2$ and $q/2$.

Note that the attacker would choose an invertible polynomial mod q and p so he can compute both F'_q and F'_p. Moreover, reduction mod p, regardless of using the right or wrong key vanishes the first component of $\bar{a}'$ as it is a multiple of p:

$$F' * \bar{a}' = m \mod q.$$

So the attacker would be able to recover the message if his $\bar{a}'$ is equal to the non-modular expression

$$a' = f' * p\phi * h + f' * m. \tag{24}$$

That happens only if all the coefficients of (24) lie between $-q/2$ and $q/2$.

Let us analyze when this can happen. Assume that the entries of a' are normally distributed with mean near zero and standard deviation $\sigma' = |a'|_\perp/\sqrt{N}$. As we already saw, the probability of having all the coefficients of a in our desired interval is directly related to the ratio $(q/2)/\sigma$.

The attacker uses the equation (16) to compute g':

$$g' = f' * h \tag{25}$$

The approximation (23) gives an estimation for $|a'|_\perp$:

$$|a'|_\perp^2 \approx \left(p^2|\phi|_\perp^2\right)|g'|_\perp^2 + (|m|_\perp^2)|f'|_\perp^2. \tag{26}$$

In which $|\phi|_\perp$ and $|m|_\perp$ may be replaced by their typical values. Setting

$$\lambda = \frac{|m|_\perp}{p|\phi|_\perp} \tag{27}$$

we will have

$$\sigma'^2 = \frac{|a'|_\perp^2}{N} \approx (\frac{p^2|\phi|_\perp^2}{N})(|g'|_\perp^2 + \lambda^2|f'|_\perp^2). \tag{28}$$

Therefore, if the value of $|a'|_\perp$ is smaller or not much larger than $|a|_\perp$, then f' can be used for decryption of the ciphertext and recovers the message with almost the same probability as f does. Note that for an element f' chosen from $\mathcal{L}_f$, having g' in $\mathcal{L}_g$ would guarantee the criterion above. That is because all the elements of $\mathcal{L}_f$ ($\mathcal{L}_g$) have the same centered norm as f (g). Nevertheless, it is not a restriction and any f' and g' which are small enough so that σ'^2 is not much larger than σ^2 would do the decryption.

5.4.1 Brute Force Attack

One approach for the attacker can always be the brute force attack. That is he can try all possible $f \in \mathcal{L}_f$ and see if $h * f$, the potential g, lies in $\mathcal{L}_g$ or has small entries. Conversely, he can try all the possible $g \in \mathcal{L}_g$ and see if $g * h^{-1}$ has small entries. In practice the size of $\mathcal{L}_g$ is always smaller so the key security level, ie., the amount of required work to recover the key, is determined by the cardinality of $\mathcal{L}_g$.

Similarly, the attacker can try all the possible $\phi \in \mathcal{L}_\phi$ and see if $e - \phi * h$ is in $\mathcal{L}_m$. The message security level is thus given by cardinality of $\mathcal{L}_\phi$. This value is reduced in half by a meet-in-the-middle attack described in [23].

5.4.2 Lattice Attack

Now let us see the connection of NTRU with lattices. Another approach for the attacker is the so-called lattice attack. In this method, the attacker considers a lattice in which finding the private key corresponding to the public key h is equivalent to finding a short vector. In order to recover the private key or an alternative key with the same efficiency, the attacker builds a lattice whose elements correspond to the possible private keys f' and g' and with the L_2 norm of its elements being

$$|g'|_\perp^2 + \lambda^2 |f'|_\perp^2.$$

Consider the lattice $\mathcal{L}'$ generated by the $2N \times 2N$ matrix

$$\mathbf{B}' = \begin{pmatrix} \lambda & 0 & \dots & 0 & h_0 & h_1 & \dots & h_{N-1} \\ 0 & \lambda & \dots & 0 & h_{N-1} & h_0 & \dots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda & h_1 & h_2 & \dots & h_0 \\ 0 & 0 & 0 & 0 & q & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q \end{pmatrix},$$

where h is the public key and q is the same as in the system parameters.

Lemma 5.4.1. *The lattice $\mathcal{L}'$ contains the vector $\tau = (\lambda f, g)$.*

Proof. First observe that the upper right block of $\mathbf{B}'$ is the multiplication matrix of the polynomial h:

$$(F_0, \dots, F_{N-1}) \begin{pmatrix} h_0 & h_1 & \dots & h_{N-1} \\ h_{N-1} & h_0 & \dots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ h_1 & h_2 & \dots & h_0 \end{pmatrix} = F(X) * h(X). \tag{29}$$

and the two other non-zero blocks are scalar multiplications by λ and q. The lattice $\mathcal{L}'$ contains a vector if and only if it is a linear combination of $\mathbf{B}'$'s rows with coefficients in $\mathbb{Z}$. Observe that

$$(a_1, ..., a_{2N}) \times \mathbf{B}' = \sum_{i=1}^{2N} a_i \mathbf{r}_i$$

in which $\mathbf{r}_i$s are the rows of $\mathbf{B}'$. So we need to check that there exists a vector in $\mathbb{Z}^{2N}$ whose multiplication by $\mathbf{B}'$ gives us τ. It follows from (29) and the definition of h that

$$(f, 0)B = (\alpha f, f * h) = (\lambda f, g).$$

□

By choice of polynomials f and g, the vector τ is among the short vectors in this lattice. There are several other vectors with the same norm in $\mathcal{L}'$ as it is demonstrated in the following proposition.

Proposition 5.4.2. *the lattice $\mathcal{L}$ contains at least N vectors $(\alpha f', g')$ with the same length as τ satisfying $f' * h = g' \mod q$.*

Proof. We claim that all cyclic shifts of f and g satisfy the desired equation. Observe that

$$\begin{aligned} f * X &= \left(\sum_{i=0}^{N-1} f_i X^i\right)(X) \\ &= f_{N-1} + f_0 X + \cdots + f_{N-2} X^{N-1} \\ &= (f_{N-1}, f_0, \ldots, f_{N-2}) \quad \forall f \in R \end{aligned}$$

Arguing similarly, any cyclic shift of f can be shown as $f * X^i$ for some $0 \leq i \leq N$. Note that all cyclic shifts of a vector have the same size as the original vector. Now observe that if $f * h \equiv g \mod q$, then $f * x^i * h \equiv g * x^i \mod q$. So we have

$$(f * x^i, 0) \begin{pmatrix} \lambda & 0 & \ldots & 0 & h_0 & h_1 & \ldots & h_{N-1} \\ 0 & \lambda & \ldots & 0 & h_{N-1} & h_0 & \ldots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda & h_1 & h_2 & \ldots & h_0 \\ 0 & 0 & 0 & 0 & q & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q \end{pmatrix} = (\lambda f * x^i, g * x^i).$$

Hence, the vector $(\lambda f * x^i, g * x^i)$ is in the lattice, has the same length as τ and satisfies our equation. Since there are N such vectors we have at least N such answers. □

More generally, an arbitrary element of $\mathcal{L}'$ looks like

$$(X, Y) \begin{pmatrix} \lambda I & H \\ 0 & qI \end{pmatrix} = (\lambda X, h * X + qY) \equiv (\lambda X, h * X) \mod q$$

for $X, Y \in \mathbb{Z}^N$. Letting $X = f'$, an element of $\mathcal{L}'$ looks like

$$(\lambda f', h * f') = (\lambda f', g') \mod q.$$

Note that the block H on the upper right of $\mathbf{B}'$ guarantees the satisfaction of equation (16) and the" mod q part" is taken care of by the lower right qI block.

So every vector in the lattice $\mathcal{L}$ corresponds to a pair of private keys f' and g' such that $f' * h = g'$. But not all such vectors can be used to decrypt the message since by equation (28), we need f' such that $|a'|_\perp^2$ be less or not much greater than $|a|_\perp^2$ in equation (23). The attacker wants to construct a lattice in which the L_2 norm of its elements corresponds to the centered norm of the $|a'|_\perp^2$.

Let us compute the L_2 norm of a typical element of $\mathcal{L}'$:

$$\begin{aligned}(\|(\lambda f', g')\|_2)^2 &= \lambda^2 \left(\sum_i f_i'^2 \right) + \sum_j g_j'^2 \\ &= |g'|_2^2 + \lambda^2 |f'|_2{}^2\end{aligned}$$

By Lemma 5.2.7, one can replace the above L_2 norms by centered norms if f' and g' have zero mean. However, they do not necessarily have zero mean. So we alter the lattice by subtracting the mean of each vector in $\mathcal{L}'$ which differ from one block to the other so that we get the desired norm property. The new obtained lattice is

$$\mathbf{B} = \begin{pmatrix} \lambda I - (\lambda/N)J & H - \alpha J \\ 0 & qI - (q/N)J \end{pmatrix}$$

Where α is a suitable scalar. Now an arbitrary vector in the new lattice $\mathcal{L}(\mathbf{B})$ has mean zero and is of the form

$$(f', X) \begin{pmatrix} \lambda I - (\lambda/N)J & H - \alpha J \\ 0 & qI - (q/N)J \end{pmatrix} = (\lambda(f' - (\bar{f}')\mathbf{1}), g' - (\bar{g}')\mathbf{1}) \mod q$$

So the square of its L_2 norm is

$$\begin{aligned}\|(\lambda(f'-(\bar{f}')\mathbf{1}), g'-(\bar{g}')\mathbf{1})\|_2^2 &= \sum_{i=0}^{N-1}\left(\lambda(f_i'-\bar{f}')\right)^2+\sum_{i=0}^{N-1}\left((g_i'-\bar{g}')\right)^2 \\ &= \lambda^2|f'|_\perp^2+|g'|_\perp^2 \\ &= \left(\frac{1}{p^2|\phi|_\perp^2}\right)\left[|m|_\perp^2|f'|_\perp^2+p^2|\phi|_\perp^2|g'|_\perp^2\right] \\ &= \left(\frac{1}{p^2|\phi|_\perp^2}\right)|a'|_\perp^2\end{aligned}$$

Thus, the attacker constructs the lattice $\mathcal{L}$ and if he can find a vector with norm shorter or around $\frac{1}{p^2|\phi|_\perp^2}|a|_\perp^2$, he computes the f' (g') corresponding to that vector which would serve as an alternative decryption key. In other words, the effectiveness of a lattice element which corresponds to a potential decryption key is directly related to its norm. Since the private key f is chosen such that it demonstrates a good decryption reliability, its corresponding lattice element is among the shortest vectors in the lattice.

The cryptanalyst is interested in finding a shortest feasible vector in the lattice $\mathcal{L}$. Therefore, the success of the attack depends on the performance of lattice reduction algorithms. The dimension of the lattice is chosen large enough so that the current reduction algorithm should fail to find a short enough vector in the lattice.

5.5 NTRU Encryption Scheme

The encryption scheme is based on the one-way trapdoor function described in the previous section. In real implementation, this scheme is used together with a padding scheme, that is a technique of embedding a message in to the plaintext space in a secure way, for more security[10]. However, The use of padding does not effect the construction of decryption and decryption explain here. Using the same notations, here we explain the encryption scheme.

5.5.1 Key Generation

Bob generates the private and public keys as follows

Key Generation

1. Take the parameters (N, p, q, d_f, d_g, d).
2. Run the Generate algorithm to get (k, f).
3. Set the public key to be the h computed in the Generate algorithm.
4. Set the private key to be f.

Remark 5.5.1. Since Bob will need the polynomial F_q in the decryption, to save time he stores this polynomial (obtained by the Generate algorithm.)

5.5.2 Encryption and Decryption

Encryption

Alice, having Bob's public key h and the parameters, wants to send the message $m \in \mathcal{L}_m$ to Bob.

1. She runs the Sample algorithm to get a $\phi \in \mathcal{L}_\phi$.
2. She runs the Evaluate algorithm with inputs m and ϕ.
3. She sends the output e to Bob as ciphertext.

Decryption

1. Bob having the private key f receives e.
2. He runs the Invert algorithm and recovers m granted that no decryption failure occur.

5.5.3 Encryption from a Lattice Point of View

Although NTRU was originally presented over a ring of polynomials as described in the previous sections, for the purpose of comparing to other lattice-based cryptosystems, we give a description of the encryption using the lattice. In order to do that

we make a slight alteration in the definition of the public key. That does not effect the principle of the cryptosystem.

Let the public key h be $pF_q * g$ and consider the lattice $\mathcal{L}(\mathbf{B})$ which is determined by the public key h and the parameter q, where

$$\mathbf{B} = \begin{pmatrix} \mathbf{I} & \mathbf{H} \\ 0 & q\mathbf{I} \end{pmatrix}$$

and $\mathbf{H}$ is the matrix of multiplication by h. For the message m and the random ϕ the ciphertext can be computed as

$$\begin{aligned} (\phi, 0)\begin{pmatrix} \mathbf{I} & \mathbf{H} \\ 0 & q\mathbf{I} \end{pmatrix} + (-\phi, m) &= (\phi, p\phi * h) + (-\phi, m) \mod q \\ &= (0, p\phi * h + m) \mod q \end{aligned}$$

So, the message m makes a part of an error vector whose first part is determined by the random polynomial ϕ. This error vector perturbs a lattice vector corresponding to the integral vector $(p\phi, 0) \in \mathbb{Z}^n$ with respect to the public basis $\mathbf{B}$. Therefore, the ciphertext is a vector in $span(\mathbf{B})$ close to $\mathcal{L}(\mathbf{B})$. It is close because the entries of both ϕ and m are small by construction. However, since there is no restriction on the number of nonzero elements in m, as is the case for f, g and ϕ, the perturbation vector $(-\phi, m)$ might be larger than λ_1 for this lattice. Hence, recovering a particular message is not necessarily equivalent to solving the CVP.

5.5.4 Review of Attacks on NTRU

The first attack analysis on the NTRU encryption scheme, except those that are described in [19] by the authors, is the lattice attack proposed in [11]. This attack, as we saw, suggests that the dimension of the NTRU lattice should be taken large enough to make it infeasible for the lattice reduction algorithms to succeed.

As it is described, decryption in the NTRU cryptosystem is imperfect, in the sense that, there is a small probability that the decryption algorithm fails to decipher a validly generated ciphertext. This property has caused intensive research and resulted

in a number of papers dedicated to answering the question of to what extent, if any, the decryption failure effects the security of the cryptosystem, from different approaches. A list of these papers is available through [42].

One can divide the attacks on NTRU in two categories. The first category consists of those in which the NTRU lattice is targeted. Of particular interest, one can mention [11, 21, 9, 48, 49]. Roughly speaking, the attacks in this category discuss the size of the security parameter, that is the lattice dimension, by tackling the instances of the hard mathematical problem underneath the cryptosystem.

The second category consists of the attacks which take advantage of the decryption failure property in NTRU. For instance, a chosen ciphertext attack was presented in [13]. In this attack, the attacker can derive some information about the private key by observing the decryption corresponding to specially constructed invalid ciphertexts. Since this attack, the padding schemes that had been considered for a potential chosen ciphertext attack have been changed [39, 20]. In [39], NTRU with two proposed padding schemes was claimed to be semantically secure *i.e,* secure against a computationally limited attacker. However, as it was claimed in [43], those notions of semantic security are not defined under the assumption of existence of decryption failure. These investigations on decryption failure not only emphasize on the importance of a good choice of parameters so that the decryption failure probability is negligible, but also demonstrate the importance of using suitable padding schemes.

Although there have been changes in the NTRU encryption scheme since it was originally proposed, these changes do not effect the mathematical core of the cryptosystem. That is the reason we chose to present the original version that demonstrates the underlying construction the best.

5.6 Summary

In the NTRU cryptosystem, the message, as well as the public and private keys, are polynomials in R. The private key f is a small polynomial, which is related to the public key by another small polynomial g through the equation $h = F_q * g \mod q$. The message is also a small polynomial which is encrypted into $e = p\phi * h + m \mod q$

where ϕ is a random polynomial. This computation is fast and easy. Reduction modulo q and randomness of ϕ hide the message m in e. The idea of the decryption is the observation that the coefficients of product of small polynomials in R tend to be distributed approximately centered around zero. So, for a good choice of parameters, the expression $f * e = p\phi * g + f * m$ has all its coefficients between $-q/2$ and $q/2$. That implies reliable recovery of m from e by multiplication by $f \mod q$ followed by multiplication by $F_p \mod p$.

The only privilege of the private key f and its corresponding g to other pairs of polynomials in R which satisfy $h = F_q * g \mod q$ is their small size. In fact, the only feature of f and g which leads to having $|f * e| < q$ is their small size. That reduces the problem of breaking the cryptosystem to find a short vector in a lattice whose elements satisfy $h = F_q * g \mod q$. This lattice turns out to be of a special form of:

$$\begin{pmatrix} \lambda I & H \\ 0 & qI \end{pmatrix}.$$

There is not much known about the complexity of finding a short vector in such a lattice. However, the fast encryption and decryption and the small public key size has made NTRU the only practical lattice-based cryptosystem so far. Its performance is compatible with currently common used cryptosystems (see [24]). NTRU is currently being marketed for Java, C, and OMAP platforms.

Chapter 6

NTRU-Like Cryptosystems

The cryptosystem NTRU is defined over ring of polynomials with coefficients in $A = \mathbb{Z}$. In 2002, P. Gaborit, J. Ohler and P. Sole proposed a polynomial analogue of NTRU, called CTRU where the ring of integers is replaced by the ring of polynomials over the binary field $\mathbb{F}_2$ [14]. Although CTRU did not succeed as a reliable analogue of NTRU, it initiated the idea of using the NTRU encryption method over rings other than $\mathbb{Z}$. This idea was followed in [25] where a NTRU-like cryptosystem, with its ring of integers replaced by the Gaussian integers (*i.e,* $\mathbb{Z}[i]$) was proposed. As suggested in [25], it is worthwhile to investigate how the system would work over other rings.

This chapter consists of three sections. In Section 6.1, we briefly explain the prior works that have been done on the subject of generalizing NTRU. In Section 6.2, we investigate the requirements for an integral domain A to be qualified as a NTRU-like ring base and discuss the general steps needed to be taken to define a NTRU-like cryptosystem over A. In Section 6.3, which is summarized from [14], we describe CTRU as an example of our general discussion.

6.1 Prior Works

After CTRU, which will be described in detail in Section 6.3, another generalization of NTRU was described over Gaussian integers by Kouzmenko in [25]. In this thesis, Kouzmenko shows how CTRU can be successfully attacked, and raises the possibility

of extending NTRU to other rings. We follow up on this in Section 6.2. In this section, let us clarify where Kouzmenko's work ends and ours begins. In [25], it is mentioned that "the existence of a division algorithm [in the base ring of NTRU] gives us several useful properties: every such ring possesses a unit element, is a principal ideal ring, we can define prime elements and the notion of greatest common divisor of two elements. An Euclidean ring is also a unique factorization domain". Moreover, it is pointed out that to inverting a polynomial, as is required in the key creation process of NTRU, can be done with use of the extended Euclidean algorithm and Chinese Reminder Theorem if the base ring is a Euclidean domain.

It is further discussed that "to switch from computation mod q to computation mod p in the decryption part" the criterion given in NTRU, that is $|p\phi * g + f * m|_\infty < d(q)$ where d is the Euclidean norm, "does not generally work for all Euclidean rings, since the division algorithm does not produce unique remainder", without discussing the reason why it works on $\mathbb{Z}$ although Euclidean function does not admit unique remainder even on $\mathbb{Z}$. Moreover, it is claimed that "there may exist a constant C such that if $d(x) < \frac{d(q)}{C}$, then x is the unique representative of equivalence class mod q" without including further discission on what that constant might be. Moreover, he does not show why the representatives within such a range should be unique, nor is any connection made in general between Euclidean rings and their lattice construction.

Furthermore, in order to illustrate NTRU over $\mathbb{Z}[i]$, it is shown that $\mathbb{Z}[i]$ is a Euclidean ring and an Euclidean algorithm is given for $\mathbb{Z}[i]$ which gives a unique representative less that $d(q)/4$ modulo q. Moreover, inverting a polynomial in $\mathbb{Z}[i]/(q)[X]$ using extended Euclidean Algorithm and Chinese Reminder Theorem is explained.

Finally it is noted that "there is a connection between the problem of finding the shortest representative modulo a non-zero Gaussian integer x and the CVP". However, it is claimed that using CVP techniques in "an overkill from efficiency point of view" without justification, given that the Euclidean algorithm described for $\mathbb{Z}[i]$ does not differ much from CVP round-off algorithm technique.

6.2 NTRU over an Arbitrary Integral Domain

We wish to extend NTRU to an arbitrary integral domain A. In this section, we deduce some necessary properties of A which allow NTRU to be defined.

6.2.1 Encryption-Decryption System

Let A be an integral domain. For a positive integer N, set R to be the ring $A[X]/(X^N - 1)$ with multiplication denoted by the symbol $*$. For an element $a \in A$, denote by (a) the ideal generated by a and let R_a denote $R/(a) \cong A/(a)[X]/(X^N - 1)$.

Let p and q be two elements in A and let the sets $\mathcal{L}_m$, $\mathcal{L}_f$, $\mathcal{L}_g$ and $\mathcal{L}_\phi$ be subsets of R. In the key creation process, two polynomials f and g are randomly chosen from $\mathcal{L}_f$ and $\mathcal{L}_g$ respectively such that f is invertible in both R_p and R_q. Denote the inverses of f in R_p and R_q by F_p and F_q respectively. The NTRU-like algorithm should be able to follow the steps below:

1. Compute the public key as $h = F_q * g \in R_q$.
2. Choose the message m from the set $\mathcal{L}_m$ and the random polynomial ϕ from $\mathcal{L}_\phi$.
3. Compute the ciphertext as $e = p\phi * h + m \in R_q$.
4. To decrypt, compute $\bar{a} = e * f = p\phi * g + m * f \in R_q$.
5. Identify $\bar{a}$ with an element $a \in R$.
6. If a is equal to $p\phi * g + m * f$ in R, the receiver recovers the message as $m = a * F_p \mod q$.

Observe that once one can calculate F_q and F_p, the first four steps can be executed without difficulty. So, as is pointed out in [25], the ring A should be chosen in such a way that there exist a polynomial time algorithm for computing the inverse of a unit element in R_q and R_p.

Note that in all the computations, we identify the elements of R_p (R_q), by polynomials in R. More precisely, in Steps 1, 3 and 4, we reduce the coefficients modulo

q and identify each coefficient by the representative of its class in $A/(q)$. Similarly, the polynomial in R_p is represented by a polynomial in R in Step 6.

In Step 5, in order to proceed to Step 6, we first have to identify $a \in R_q$ with a coset representative in R. This involves making choice of a set S of coset representatives.

In order to ensure that Step 6 has a high probability of succeeding, we have to derive conditions under which the coefficients of $p\phi * g + m * f$ lie in S. As in the case with NTRU over $\mathbb{Z}$, these conditions help to determine the sets $\mathcal{L}_m$, $\mathcal{L}_f$, $\mathcal{L}_g$ and $\mathcal{L}_\phi$ and also to choose the elements p and q.

Let us pursue the issued raised here in detail in the next two subsections.

6.2.2 Inverting a Unit Element of R_q

As we saw in the previous subsection, one of the steps we need to take while implementing a NTRU-like encryption scheme is inverting an invertible polynomial in R_q and R_p. Here we consider this problem over the ring A in three cases. We start with an integral domain and adopt the additional conditions that we need at each step. We will see that having A to be a Dedekind domain allows us the find the inverse of an invertible polynomial in R_q (R_p). We have included the statements of key theorems from ring theory in the appendix.

Theorem 6.2.1. *Let $f \in R$ and suppose $A/(q)[X]$ is a PID, then f is invertible in R_q if and only if* $\gcd(f, X^N - 1) = 1 \in A/(q)[X]$.

Proof. Assume that f is invertible. There exists a polynomial F_q in R_q such that

$$f * F_q = 1 \quad \in R_q.$$

That implies

$$fF_q = 1 + u(X^N - 1) \in A(q)[X].$$

So, for some $u \in A/(q)[X]$ we have

$$fF_q - u(X^n - 1) = 1.$$

So, 1 is the gcd of f and $X^N - 1$. Conversely, if $\gcd(f, X^N - 1) = 1$, then since $A/(q)[X]$ is a PID, there exist u and v in $A/(q)[X]$ such that

$$uf + v(X^N - 1) = 1.$$

Thus u is the inverse of f in R_q. □

First Case: q is prime

Assume that q is a prime in A and let f be an invertible polynomial in R_q. We need to find an algorithm to find F_q. If $A/(q)[X]$ is a PID, by Theorem 6.2.1, it is enough to find $u, v \in A/(q)[X]$ such that

$$\begin{aligned} fu + (X^N - 1)v &= 1 \in A/(q)[X] \\ f * u &= 1 \in R_q \end{aligned}$$

and set $u = F_q$. If $A/(q)[X]$ is an Euclidean domain, the polynomials u and v are computable by implementing the extended Euclidean algorithm. By Theorem A.0.10, if $A/(q)$ is a field, then $A/(q)[X]$ is an Euclidean domain and by Theorems A.0.9, $A/(q)$ is a field if and only if (q) is a maximal ideal. Since q is a prime element, (q) is a prime ideal by Theorem A.0.1. Therefore, a highly desirable property of A would be: every prime ideal in A is maximal, because it would allow us to use the extended Euclidean algorithm in $A/(q)[X]$.

By Proposition A.0.8, if we choose A to be a PID, (q) would be maximal and the argument above holds. However, a PID is more than what we really want. It follows from Definition A.0.12 that a Dedekind domain, which is a weaker constraint on A by Theorem A.0.13, would also fulfill our desired requirements. Note that in fact, the only property of a Dedekind domain that we are using at this point is the fact that every prime ideal is a maximal ideal.

Second Case: q is a power of a prime

Assume that $q = \alpha^k$ is a power of the prime element α in a Dedekind domain A. So, by first case, we can find $u, v \in A/(\alpha)[X]$ such that

$$fu + (X^N - 1)v = 1 \in A/(\alpha)[X].$$

Note that since $(\alpha^k) \subset (\alpha)$, we can identify the elements of $A/(\alpha)$ in $A/(\alpha^k)$. Under this identification we can write $u, v \in A/(q)$. By re-writing the above equation in $A[X]$ we get

$$fu + (X^N - 1)v = 1 - \alpha l$$

for some $l \in A[X]$. By multiplying both sides by $\prod_{i=0}^{s-1}(1 + (\alpha l)^{2^i})$, where s is the smallest integer such that $2^s > k$ we get

$$\prod_{i=0}^{s-1}(1 + (\alpha l)^{2^i})(fu + (X^N - 1)v) = 1 - (\alpha l)^{2^s}.$$

Thus the right hand side is 1 modulo q which implies that the inverse of f is

$$F_q = \prod_{i=0}^{s-1}(1 + (\alpha l)^{2^i})u.$$

A similar argument is used in [25] for the case $A = \mathbb{Z}[i]$.

Third Case: q is arbitrary

Assume that q is an arbitrary element of the Dedekind domain A and let (q) denote the principal ideal generated by q. By Proposition A.0.15, (q) has a unique factorization into prime ideals. If those prime ideals are principal, we can write $(q) = (p_1)(p_2)\dots(p_m)$. Having $(p)(p) = (p^2)$ and arguing inductively we have

$$(q) = (p_1^{\alpha_1})(p_2^{\alpha_2})\dots(p_n^{\alpha_n}),$$

where the p_i's for $1 \leq i \leq n$ are distinct primes in A and the α_i's are positive integers.

Since the $p_i^{\alpha_i}$'s are relatively prime, by the Chinese Remainder Theorem A.0.17 we have

$$A/(q) \cong A/(p_1^{\alpha_1}) \times \cdots \times A/(p_n^{\alpha_n}).$$

This ring isomorphism in particular gives us an isomorphism between the groups of units of both sides. Hence, it suffices to invert f in each of the rings $A/(p_i^{\alpha_i})$ using the second case and then solve the congruence system

$$x \equiv F_{p_i{}^{\alpha_i}} \mod p_i^{\alpha_i}$$

to determine F_q. The proof of the Chinese Remainder Theorem is constructive and gives us the inverse element. Using the Chinese Remainder Theorem increases the amount of computation in the inversion process in the second case by a factor of the number of distinct prime factors of q.

However, to satisfy the condition of having the ideals in the factorization of (q) principal, every prime ideal in A should be principal. By Proposition A.0.18, that implies A to be a PID. Therefore, in this case, we need to start with an PID.

6.2.3 Decryption Criteria

Let A be a Dedekind domain. In the second last steps of decryption, we need $p\phi * g + m * f$ to be "the same in $A/(q)$ as it is in A". To draw a more clear and precise picture of the last sentence, recall that elements of $A/(q)$ are classes. Define the natural projection map $\pi : A \to A/(q)$. Given an element $x \in A$, denote its image by $\bar{x}$. Thus, the coefficients of $p\phi * g + m * f$ lie in $A/(q)$; denote them $\bar{a}_i$ for $0 \leq i \leq N-1$. Similarly, we denote $\bar{a} = \overline{p\phi * g + m * f} \in R_q$ and $a = p\phi * g + m * f \in R$.

To identify an element of $A/(q)$ with one in A, we need to choose a subset S of A which is a complete set of residue classes modulo q. That means the set S satisfies the following conditions:

1. $\forall \bar{b} \in A/(q), \quad \exists s \in S$ such that $s \in \bar{b}$.

2. If $s_1, s_2 \in S$ and $\bar{s_1} = \bar{s_2}$, then $s_1 = s_2$.

Given S, we can define a map $\Phi : A/(q) \to S$ sending $\bar{s} \to s$. If $\bar{s} = \bar{b}$, then $\Phi(\bar{b}) = s = b + xq$ for some $x \in A$.

Given the Φ, the element a will correspond to $\bar{a}$ exactly when all a_i's lie in S. Otherwise, if a_i is not in S for some $0 \leq i \leq N-1$, then $\Phi(\bar{a_i}) \neq a_i$, so $\Phi(\bar{a}) \neq a$. This implies a constraint according to which we define the sets $\mathcal{L}_m$, $\mathcal{L}_f$, $\mathcal{L}_g$ and $\mathcal{L}_\phi$ and choose the elements p and q.

Note that if our set does not satisfy the first condition of a complete set of residue classes, one can still define a partial function Φ' defined over a subset

$$A' = \{\bar{x} \in A/(q) \quad | \quad \exists s \in S \quad \text{such that} \quad \bar{x} = \bar{s}\},$$

by $\Phi'(\bar{s}) = s$. Again, if all the coefficients of a lie in S, then the a_i's remain invariant under Φ' and a and $\Phi'(\bar{a})$ would be equal.

There might be infinitely many sets S satisfying conditions 1 and 2, and we wish too choose a "good" one. One option to determine a set S is to take a feedback approach. This means that we choose the sets $\mathcal{L}_m$, $\mathcal{L}_f$, $\mathcal{L}_g$ and $\mathcal{L}_\phi$ in a convenient way, and then observe the natural interval in which the resulting a_i's lie. This gives us the idea of finding a condition, namely $\mathfrak{S}$, which the elements of our desired S satisfy. Resuming the last paragraph, we get that S is the set of all $s \in A$ such that they make a complete (or partial) set of residues and they satisfy $\mathfrak{S}$.

The problem of identifying S in general is very broad; let us specialize to the case where A admits a Euclidean norm.

Recall that a *Euclidean domain* A is an integral domain which possesses a function called an *Euclidean function* (*Euclidean norm*) d on $A/\{0\}$ which satisfies

1. $d(ab) \geq d(a), \quad \forall a, b \in A, b \neq 0$
2. For all $a, b \in A$ with $b \neq 0$ there exist $q, r \in A$ such that

$$a = qb + r,$$

where $r = 0$ or $d(r) < d(b)$.

Decryption Criteria Over an Euclidean Domain, Part I

Let us first assume that A possesses a Euclidean norm d such that given two elements of A, the remainder and quotient are *uniquely* determined.

Lemma 6.2.2. *For such an A, the set $S = \{s \in A | \quad d(s) < d(q)\} \cup \{0\}$ is a complete set of residues modulo q.*

Proof. For a given $b \in A$, the Euclidean norm gives us a unique remainder s and quotient k in A such that $b = kq + s$ and $d(s) < d(q)$ or $s = 0$. So $s \in S$ and $s \in \bar{b}$. Therefore the first condition is fulfilled.

To check the second condition, suppose that s_1 and s_2 are two nonequal elements of S such that $\bar{s_1} = \bar{s_2}$. Hence we have $s_1 - s_2 \in (q)$, so $s_1 = kq + s_2$ for some $k \in A$.

Hence, we have two expression for s_1 as

$$\begin{aligned} s_1 &= 0q + s_1 \\ &= kq + s_2. \end{aligned}$$

Since $d(s_1) < d(q)$ and $d(s_2) < d(q)$, this contradicts the assumption that A admits unique remainders. So $s_1 = s_2$ and the second condition is fulfilled. □

Note that here we have defined S by the condition $\mathfrak{S} : d(s) < d(q)$ or $s = 0$.

Having a unique remainder is quite a strong condition which eases finding S remarkably. The CTRU cryptosystem enjoys this property which, as we will see in Lemma 6.3.4, leads to having very simple criteria for successful decryption. However, the following theorem, taken from [46], tells us that the only rings which satisfy this condition are the rings of polynomials over a field.

Theorem 6.2.3. *Let A be an integral domain that is not a field and that possesses a Euclidean function d such that for every a and non-zero q in A, there exist unique r and s such that*

$$a = rq + s \quad d(s) < d(r) \quad or \quad s = 0.$$

Then $A = F[X]$ for some field F.

Proof. First we prove the following lemma.

Lemma 6.2.4. *Having unique quotient and remainder is equivalent to having*

$$d(a + b) \leq \max\{d(a), d(b)\} \quad \forall a, b \in A. \tag{30}$$

Proof. Suppose that we have this condition and for $a, b \neq 0$ there exist two pairs of elements r_1, s_1 and r_2, s_2 such that

$$\begin{aligned} a &= r_1 b + s_1 \quad d(s_1) < d(b) \\ &= r_2 b + s_2 \quad d(s_2) < d(b). \end{aligned}$$

Subtracting the first row from the second we get

$$b(r_2 - r_1) = (s_1 - s_2) \neq 0.$$

Since d is an Euclidean function, this implies $d(b) \leq d(s_1 - s_2)$. Now (30) implies that $d(b) \leq \max\{d(s_1), d(s_2)\}$ which is a contradiction.

Conversely, suppose that we have the uniqueness of the remainder of the Euclidean function and there exist elements a and b in A such that

$$d(a+b) > \max\{d(a), d(b)\}.$$

We can write

$$\begin{aligned} a^2 - b^2 + b &= (a+b)(a-b) + b \qquad d(b) < d(a+b) \\ &= (a+b)(a-b+1) - a \qquad d(a) < d(a+b), \end{aligned}$$

which contradicts the uniqueness of the remainder. □

Now, define the set F as

$$F := A^* = \{a \in A | \quad d(a) = d(1)\}.$$

Recall that every non-zero element $a \in A$ has norm greater or equal than $d(1)$. Moreover, $d(a) = d(1)$ if and only if a is unit. Since A is not a field by assumption, the set $A \setminus F$ is not empty. Let X be an element in $A \setminus F$ such that $d(X)$ is minimum. We can choose this element since $d(a) \geq 1$ for a non-zero element a and $\mathbb{Z}^+$ is well ordered. Thus

$$d(1) < d(X) \leq d(a) \quad \forall a \in A \setminus F.$$

By assumption, for any $a \in A \setminus F$ there exist unique r and s such that

$$a = rX + s \quad d(s) < d(X) \quad \text{or} \quad s = 0$$

If $s = 0$ then $d(a) = d(rX) \geq d(r)$. Observe that s should be in F. Otherwise, it contradicts the choice of X of being the element with smallest norm. Moreover, we have

$$\begin{aligned} d(r) &< d(rX) \\ &= d(a - s) \\ &\leq \max\{d(a), d(s)\} \\ &= d(a). \end{aligned}$$

If r is in F we are done since a is written uniquely as a polynomial in X with coefficients in the field F. If not, there exist unique r_1 and s_1 such that

$$r = r_1 X + s_1 \quad d(s_1) < d(r_1) \quad \text{or} \quad s_1 = 0.$$

For the same reason as above, s_1 is in F and $d(r) > d(r_1)$. This gives a sequence

$$d(a) > d(r) > d(r_1) > d(r_2) > ...$$

of strictly decreasing positive integers. We repeat this process until we get a $r_n \in F$. Then we can write

$$a = r_n X^{n+1} + s_n X^n + \cdots + s_1 X + s$$

where all the coefficients are in the field F. Furthermore, these coefficients are unique. Thus $A = F[X]$.

Conversely, a polynomial ring over a field with standard degree norm satisfies (30) which by the lemma is equivalent to having a unique quotient and remainder. □

Decryption Criteria Over an Euclidean Domain, Part II

Now suppose that A is an Euclidean domain which is not a polynomial ring over a field. Then, A possesses a Euclidean norm for which the quotient and remainder are not unique in all cases. Thus, observe that the set

$$S' = \{s \in A | \quad d(s) < d(q)\} \cup \{0\},$$

while exhausting all residue classes of $A/(q)$, may contain more than one representative of a particular class in $A/(q)$. In this case we need to smartly choose a subset among the classes of congruent elements in S'.

One way to choose S is to consider the given Euclidean Algorithm. For example, if we have an algorithm that always gives us the unique smallest element of a class with respect to the Euclidean norm, our desirable set S would be the set of smallest representatives of all classes in $A/(q)$. We will define an S for $A = \mathbf{Z}[\sqrt{-2}]$ in this manner in next chapter.

Alternatively, one may try to narrow down the permitted range of norms for elements of S. In other words, we are looking for a suitable threshold t, depending on q, such that the set

$$S_t = \{s \in A | \quad d(s) < t\}$$

satisfies the second condition. Unfortunately, with this approach we lose the first condition. That is, there may exist $b \in A$ such that no element of $\bar{b}$ lie in S_t. By decreasing the upper bound t, we exclude all the remainders of the same class with bigger norm but we might exclude the smallest remainder from another class . This leads to the failure of first condition, and hence only a partial function Φ' as discussed in 6.2.3 can be used. In this case, our attempt is to choose the system parameters in such a way that the coefficients of a are not among the unintentionally excluded representatives.

Example 6.2.5. In original NTRU, $A = \mathbf{Z}$ with $d = |\quad|$. The set $\{a \in \mathbb{Z} | \quad |z| < |q|\}$ has two elements from each class modulo q. By choosing the smallest of each pair we get

$$S = (-q/2, q/2) \cap \mathbb{Z}.$$

In the next section, will describe the CTRU cryptosystem as an example of a NTRU-like cryptosystem over a ring which is a polynomial ring over a field.

6.3 CTRU Cryptosystem

6.3.1 Notation

This cryptosystem was presented in [14] together with an analysis of the lattice based attack which we discuss below. Let $A = \mathbb{F}_2[T]$ denote the ring of polynomials over $\mathbb{F}_2$. As above, let N be a positive integer and $R := A[X]/(X^N - 1)$. A typical element of R can be represented as

$$F(X) = F_0(T) + F_1(T)X + F_2(T)X^2 + \cdots + F_{N-1}(T)X^{N-1}$$

where for each i between 0 and $N - 1$ the coefficient of X^i is

$$F_i(T) = F_{i0} + F_{i1}T + F_{i2}T^2 + \cdots + F_{ik_i}T^{k_i}$$

where $0 \leq j \leq k_i$, and F_{ij}'s are binary. Let P and Q be two irreducible elements of A of degree s and t respectively such that $2 \leq s \leq t$ and $\gcd(s,t) = 1$. The sets $\mathcal{L}_f$, $\mathcal{L}_g$, $\mathcal{L}_\phi$ and $\mathcal{L}_m$ are subsets of R. As a brief comparison with NTRU, observe that $\mathbb{Z}$ is replaced by A, the integers p and q are replaced by irreducible polynomials P and Q, and finally, $\mathbb{Z}[X]/(X^N - 1)$ is replaced by $A[X]/(X^N - 1)$.

It is worth observing that the quotient rings $A_P := A/(P)$ and $A_Q := A/(Q)$ are isomorphic to finite fields of order 2^s and 2^t respectively. Thus the quotient rings $R_P := R/(P)$ and $R_Q := R/(Q)$ are isomorphic to $\mathbb{F}_{2^s}[X]/(X^N-1)$ and $\mathbb{F}_{2^t}[X]/(X^N-1)$ respectively.

The parameters t and s are chosen to be relatively prime so that the two extensions of $\mathbb{F}_2$ of degree t and s intersect only in $\mathbb{F}_2$. More precisely, $\mathbb{F}_{2^t}$ is Galois over $\mathbb{F}_2$ with Galois group isomorphic to $\mathbb{Z}/t\mathbb{Z}$. By the fundamental theorem of Galois theory, any subfield of $\mathbb{F}_{2^t}$ corresponds to a subgroup of $\mathbb{Z}/t\mathbb{Z}$. By the Lagrange theorem, the order of any subgroup H of $\mathbb{Z}/t\mathbb{Z}$ divides t and the degree over $\mathbb{F}_2$ of the corresponding subfield is $|\mathbb{Z}/t\mathbb{Z} : H|$. Thus, any subfield of $\mathbb{F}_{2^t}$ is of the form $\mathbb{F}_{2^{t_1}}$ where $t_1|t$. Since $\gcd(t,s) = 1$, the only subfield of $\mathbb{F}_{2^s}$ which is also a subfield of $\mathbb{F}_{2^t}$ is $\mathbb{F}_2$.

Definition 6.3.1. For any polynomial $F \in R$, by $\deg_T(F)$ we mean the maximum degree in T of the coefficients of X. In other words, $\deg_T(F)$ is computed for F as a polynomial in T.

Note that since $\deg(f+g) = \max\{\deg(f), \deg(g)\}$, by Theorem 6.2.3, the function "deg" over A is a Euclidean norm which gives unique quotient and remainder. Hence, CTRU falls under the first category that we discussed in the previous section. We define $\Phi(x)$ to be the remainder of x upon dividing by Q for every $x \in A_Q$.

Definition 6.3.2. For an integer number $d \leq t$, where t is the degree of Q, define the set $\mathcal{L}(d)$ as

$$\mathcal{L}(d) = \{F \in R \quad | \quad \deg_T(F) < d\}$$

Lemma 6.3.3. *The set $\mathcal{L}(d)$ has 2^{Nd} elements.*

Proof. A typical element of $\mathcal{L}(d)$ looks like

$$F(X) = F_0(T) + F_1(T)X + F_2(T)X^2 + \cdots + F_{N-1}(T)X^{N-1}$$

where for each i between 0 and $N-1$ the coefficient of X is

$$F_i = F_{i0} + F_{i1}T + F_{i2}T^2 + \cdots + F_{id}T^d$$

where for $0 \leq j \leq d$, the F_{ij}'s are binary. Hence, applying the elementary counting techniques, we get exactly 2^d choices for each F_i. Since the degree in X of F is at most $N-1$, there are 2^{Nd} different possibilities for F. □

6.3.2 CTRU Encryption Algorithm

Similar to NTRU, CTRU depends on six parameters: N, P, Q (as defined in the previous section) and positive integers d_f, d_g and d_ϕ all less than t.

Key Creation

Suppose that Bob wants to create his public and private key. He picks two polynomials f and g from the set $\mathcal{L}_f := \mathcal{L}(d_f+1)$ and $\mathcal{L}_g := \mathcal{L}(d_g+1)$ respectively. The polynomial f should be invertible modulo both P and Q. That means it should be among the unit elements of the rings R_P and R_Q. We denote the inverse of f modulo P (Q) by F_P (F_Q). Having f and g, Bob computes

$$h := g * F_Q \qquad \text{mod } Q. \tag{31}$$

Bob sets his public key as h and his private key as f.

Encryption

Alice chooses her message m from the set $\mathcal{L}_m := \mathcal{L}(s)$. She also picks a polynomial ϕ randomly from the set $\mathcal{L}_\phi := \mathcal{L}(d_\phi + 1)$. She encrypts the message as

$$e := P\phi * h + m \qquad \text{mod } Q.$$

So e is an element of R_Q. She sends e to Bob.

Decryption

Bob having the private key f, receives e. He computes

$$\begin{aligned} \bar{a} &= e * f \\ &= P\phi * g + m * f \qquad \mod Q \end{aligned}$$

If a is equal to the non-modular expression $P\phi g + mf$, Bob can can recover m as

$$m = a * F_P = P\phi g F_P + mfF_P \qquad \mod P$$

Clearly, a and the non-modular $P\phi * g + m * f$ are equal if and only if each coefficient in the latter has degree less than $\deg(Q)$. As in NTRU, we need a condition under which a is equal to the non-modular expression $P\phi g + mf$. Unlike NTRU, this criterion here is convincing and easy to achieve.

Lemma 6.3.4. *For the decryption process to be successful, it is enough to have* $s + d_\phi + d_g \leq t$ *and* $s + d_f \leq t$.

Proof. Clearly, if $\deg_T(P\phi * g + m * f)$ is less that $\deg(Q)$, reduction modulo polynomial Q would not change $P\phi * g + m * f$. Since it has two components, we need each of them to be of degree less or equal than t. Using (30), we conclude

$$\begin{aligned} \deg(p\phi * g + m * f) \leq t &\Leftrightarrow \deg(P\phi * g) \leq t \quad \text{and} \quad \deg(m * f) \leq t \\ &\Leftrightarrow s + d_\phi + d_g \leq t \quad \text{and} \quad s + d_f \leq t \end{aligned} \tag{32}$$

□

Remark 6.3.5. The proof of Lemma 6.3.4 cannot hold without (30) and in NTRU it is replaced by an approximation of the norm of the sum of two elements, which leads to an approximation criteria derived from a probability argument.

Thus by choosing

$$d_f = t - s - 1, \quad d_g = d_q = \left\lfloor \frac{t - s - 1}{2} \right\rfloor,$$

Bob definitely decrypts the message successfully. Note that the assumptions d_f, d_g, d_ϕ and s all less than t is justified here.

6.3.3 Security of CTRU

Similar to NTRU, an adversary has all the system parameters $(N, Q, P, d_f, d_g, d_\phi)$ as well as the public key h. He can eavesdrop through the channel and get the ciphertext e. Having this information, he is looking for either the particular message corresponding to the cipher text e or the polynomials g and f. Notice that f and g can be calculated one from another. So it would be enough for an attacker to get one of these polynomials. Following [14], let us consider some of the analyzed attacks on NTRU adapted to CTRU. The following proposition gives us a condition under which an alternative key f' can decrypt the message.

Proposition 6.3.6. *For a polynomial $f' \in R$ to be a spurious key, it is enough to have*

$$s + d_\phi + deg(h * f') < t \quad and \quad s + deg(f') < t.$$

Proof. Suppose f' is an invertible element of $\mathcal{L}_f$ modulo both P and Q and denote the inverses by F'_P and F'_Q. Given e, we have

$$a' = e * f' = P\phi * h * f' + m * f' \qquad \text{mod } Q$$

As it is discussed in Section 5.4, a condition on f' which makes a' equal to the non-modular expression $P\phi * h * f' + m * f'$ is enough to make f' a spurious key. It follows from Lemma 6.3.4 that it happens when

$$s + d_\phi + deg(h * f') < t \quad \text{and} \quad s + deg(f') < t.$$

□

So by choosing an invertible polynomial $f' \in \mathcal{L}_f$, the attacker computes $g' = h * F'_Q$ mod Q. If

$$s + d_\phi + \deg g' < t, \tag{33}$$

then f' is a successful spurious key. Thus the condition is easy to check.

Brute Force Attack

One approach is the brute force attack. The attacker can go through all the possible $f' \in \mathcal{L}_f$ and compute g' and see if $\deg g' < t - s - d_\phi$. Conversely, if h is invertible, he can try all the possible $g' \in \mathcal{L}_g$ such that $\deg g' < t - s - d_\phi$ and see if $g' * h^{-1} < t - s$. In practice the size of $\mathcal{L}_g$ is always smaller than $\mathcal{L}_f$, the key security level, i.e., the amount of required work to recover the key, is determined by $\#\mathcal{L}_g$.

Similarly, an attacker can try all the possible ϕ and see if $e - P\phi * h$ is in $\mathcal{L}_m$. The message security level is thus given by $\#\mathcal{L}_\phi$.

Semi-Lattice Attack

Another approach to find an alternative key is the semi-lattice attack witch is an analogue to the lattice attack on NTRU. To adopt the lattice attack on NTRU to this case, we replace the notion of lattice with A-module. Thus, L_h is the module

$$L_h := \{(f', g') \in A^{2N} \quad | \quad f' * h = g' \mod Q\}, \tag{34}$$

where we identify A^N with $A[X]/X^N - 1$. Let us define $\deg_T$ of an element $(f', g') \in A^{2N}$ by $\max\{\deg_T(f'), \deg_T(g')\}$. Then L_h is a $\mathbb{F}_2[T]$ module of rank $2N$. The private pair (f, g) is an element of L_h. The module is generated by the rows of the matrix

$$M_h = \begin{pmatrix} I_N & H \\ 0 & QI_N \end{pmatrix}, \tag{35}$$

where I_N is identity matrix, Q is the CTRU parameter and H is a matrix having shifts of h as its rows.

The attacker, having h, constructs L_h. He knows that $\deg_T(g) \leq d_g \leq t - s - d_\phi$ and $\deg_T(f) \leq d_f \leq t - s$. Therefore, he is looking for an element in L_h which satisfies these two conditions. So the attacker is looking for a lattice element (f', g') such that

$$\deg(f')_T \leq d_f \quad \text{and} \quad \deg(g')_T \leq d_g.$$

Below we describe that there exist an algorithm to find a shortest element in L_h with respect to $\deg_T$. Let l be a shortest vector with $deg_T(l) = \mu_h$. Assume that

$d_g < d_f$. If $\mu_h \leq d_f$ and $\mu_h \leq d_g$, l can be used as a spurious key. Otherwise, when $d_g < \mu_h \leq d_f$, the attack is not successful. It is shown in [14] that for $s \approx m/4$ the $\mu_h > d_g$. Therefore, semi-lattice based attacks can be avoided.

On Lattice Reduction for Polynomial Matrices

Let F be a field and $A = F[T]$ be the polynomial ring over F. In this section, we see that for an A-module L, there exists a polynomial time algorithm that gets a basis of L as input and computes the shortest vector in L with respect to $\deg_T$ as output. More precisely, this algorithm computes the *weak Popov* form of a given matrix with entries from $F[T]$ [37]. First we need to introduce some notation.

Let $\mathbf{M} = (m_{ij})$ be a $n \times m$ matrix with $m_{ij} \in F[T]$.

Definition 6.3.7. For $1 \leq i \leq n$, the *ith pivot index* I_i of M is an integer defined as follows: if $m_{ij} = 0$ for $1 \leq j \leq m$, then $I_i = 0$. Otherwise

1. $\deg(m_{ij}) \leq \deg(m_{i,I_i})$ for $1 \leq j < I_i$;
2. $\deg(m_{ij}) < \deg(m_{i,I_i})$ for $I_i < j \leq m$.

If $I_i \neq 0$, the element m_{i,I_i} is called the *ith pivot element* of M and is denoted by P_i. The degree of P_i is called the *ith pivot degree* of M and is denoted by D_i.

In other words, the pivot element is the right most element with maximum degree in its row.

Definition 6.3.8. The carrier set C of M is defined as $C = \{1 \leq i \leq n | I_i \neq 0\}$. In other words, C is the set of indexes of nonzero rows of M.

Definition 6.3.9. M is said to be in *weak Popov* form if the positive pivot indices of M are all different, i.e. if

$$k, l \in C, \quad k \neq l \Rightarrow I_k \neq I_l.$$

In [37] a polynomial time algorithm is proposed. This algorithm transforms a given matrix into weak Popov form by applying unimodular row transformations.

Now suppose that L is a $F[T]$-module and M is the matrix having a basis of L as its rows. Applying this algorithm, we get another basis for L which is in weak Popov form. The lemma below shows how we can get the shortest vector in a module by having its basis matrix in weak Popov form.

Lemma 6.3.10. *If M is in weak Popov form and l is such that* $\deg(P_l) = \min_C(\deg(P_i))$ *then all vectors in the A-module generated by the rows of M have degree at least* $\deg(P_l)$.

Proof. Let $r = \sum_{i=1}^{n} d_i r^i \neq 0$ be an arbitrary element of the associated module L, where r^i denotes the ith row of M and $d_i \in A$. Let k be the row where we have for $i \neq k$ either $\deg(d_i P_i) < \deg(d_k P_k)$ or if $\deg(d_i P_i) = \deg(d_k P_k)$ then $I_i < I_k$. Then for $i \neq k$ we have

1. If $\deg(d_i P_i) < \deg(d_k P_k)$, then $\deg(d_i m_{i,I_k}) \leq \deg(d_i P_i) < \deg(d_k P_k)$;
2. If $\deg(d_i P_i) = \deg(d_k P_k)$ and $I_k < I_k$, then $\deg(d_i m_{i,I_k}) < \deg(d_i P_i) = \deg(d_k P_k)$.

It follows that $\deg(r_{I_k}) = \deg(d_k P_k) \geq \deg(P_l)$.

□

Corollary 6.3.11. *Given an A-module L with associated matrix M, the shortest vector of L with respect to* $\deg_T$ *corresponds to a row of the weak Popov form of M with minimum* $\deg(P_i)$.

Thus, the attacker can find $\mu_h = \deg(P_l)$. As described before, he needs $\mu_h \leq d_f$ and $\mu_h \leq d_g$ to hold. Based on this challenge, the authors of [14] give a so-called *the shortest pair of vectors problem (SPVP)* on which the security of CTRU is based.

Definition 6.3.12. (Shortest Pair of Vectors Problem, SPVP) Let A_1 and A_2 denote two matrices in $M(r \times c_1[\mathbb{F}]$ and $M(r \times c_2[\mathbb{F}]$ respectively, and d_1, d_2 two non-equal integers. Let D be a best upper bound on the degree in T of the entries of both matrices. The *shortest pair of vectors problem* is finding an $u \in \mathbb{F}_2[T]^r$ such that both $|u^t A_1|_s \leq d_1$ and $|u^t A_2|_s \leq d_2$ hold.

Note that if d_1 and d_2 are equal, the shortest vector in the module which is obtained by the Popov normal reduction algorithm in polynomial time is the answer.

Linear System Attack

Although CTRU seems to be immune to lattice attacks, an easily implementable successful attack based on solving a system of linear equations is presented in [25] that. In this attack, equating the coefficients of the equation $f * h = g$ gives a system of equations in $A = F_2[X]$. The fact that the coefficients of these equations are in $\mathbb{F}_2$ together with the constraints given in Proposition 6.3.6 makes it easy to solve the system of linear equations as well as guarantees that every solution can be used as a spurious key. However, this attack can not be implemented on NTRU. That is because, since the coefficients of the obtained linear system are in $\mathbb{Z}$, they may become large and decrease the running time of finding the solution for the system. Moreover, not every solution of the system guarantees successful decryption.

Chapter 7

NTRU Over Euclidean Domains: ETRU

As it was mentioned before, after CTRU, the idea of generalizing NTRU over other rings was followed in [25], where NTRU over Gaussian integers was presented. We expanded upon this idea in the previous chapter by analyzing the requirements of an integral domain to be a NTRU base ring. This chapter consists of three sections, in each of which we develop an example of our generalization of NTRU described in Chapter 6. In our examples, A is $\mathbb{Z}[\sqrt{-2}]$, $\mathbb{Z}[\zeta_3]$ and $\mathbb{Z}[\zeta_5]$. Since our first example $A = \mathbb{Z}[\sqrt{-2}]$ resembles the NTRU over Gaussian integers, we will not describe NTRU over $\mathbb{Z}[i]$. All proofs in this chapter, excepted as noted, are my own.

7.1 ETRU over $\mathbb{Z}[\sqrt{-2}]$

As a result of the discussion in Section 6.2, there are two issues that need to be set for A in order to become NTRU based ring: Decryption criteria and inverting a polynomial. We concluded that A needs to be a Dedekind domain. If A is an Euclidean domain, it follows from Theorem A.0.5 and A.0.13 that it is a Dedekind domain. In the following subsection, we show that the ring $\mathbb{Z}[\sqrt{-2}]$ is an Euclidean domain.

7.1.1 $\mathbb{Z}[\sqrt{-2}]$ is an Euclidean Domain

In this section, we show that $\mathbb{Z}[\sqrt{-2}]$ is norm Euclidean. Moreover, we give a Euclidean algorithm to efficiently compute the quotient and remainder.

The ring $\mathbb{Z}[\sqrt{-2}]$ is defined to be $\{a + b\sqrt{-2} \mid a, b \in \mathbb{Z}\}$. Hence each $\beta \in \mathbb{Z}[\sqrt{-2}]$ can be written uniquely as linear combination of $\{1, \sqrt{-2}\}$.

Observe that $\mathbb{Z}[\sqrt{-2}] \subset \mathbb{Q}[\sqrt{-2}]$. So it inherits the norm of the field $\mathbb{Q}[\sqrt{-2}]$. The minimal polynomial of $\sqrt{-2}$ over $\mathbb{Q}$ is $X^2 + 2$ which has two roots $\pm\sqrt{-2}$ in $\mathbb{Q}[\sqrt{-2}] \subset \mathbb{C}$. Hence, the Galois group $Gal(\mathbb{Q}[\sqrt{-2}]/\mathbb{Q})$ is $\{\sigma_1, \sigma_2\}$ where

$$\sigma_1 = id \quad , \quad \sigma_2(a + b\sqrt{-2}) = a - b\sqrt{-2},$$

for $a, b \in \mathbb{Q}$. Therefore, the norm $d(\beta)$ of an element $\beta = a + b\sqrt{-2}$ in $\mathbb{Q}[\sqrt{-2}]$ is by definition

$$\begin{aligned} \sigma_1(\beta)\sigma_2(\beta) &= (a + b\sqrt{-2})(a - b\sqrt{-2}) \\ &= a^2 + 2b^2. \end{aligned}$$

This is a positive multiplicative norm. Thus, the first condition in Definition A.0.2 is satisfied.

We can also view $\mathbb{Z}[\sqrt{-2}]$ as the lattice $\mathcal{L}(1, \sqrt{-2})$ in $\mathbb{R}^2 \cong \mathbb{C}$. This is an orthogonal lattice since the inner product of $(1, 0)$ and $(0, \sqrt{2})$ is zero. Hence, we can consider the elements of $\mathbb{Z}[\sqrt{-2}]$ as vectors in complex plane. Thus, we have

$$\begin{aligned} |\beta|^2 &= |a + \sqrt{2}bi|^2 \\ &= (a)^2 + (\sqrt{2}b)^2 \\ &= a^2 + 2b^2 \\ &= d(\beta) \end{aligned}$$

where $|\quad|$ denotes the magnitude of a vector in complex plane. Hence, the two norms coincide.

To show the second condition of Definition A.0.2, we need to show that there exist k and r in $\mathbb{Z}[\sqrt{-2}]$ such that

$$\alpha = k\beta + r, \quad d(r) < d(\beta). \tag{36}$$

Let α and β denote two elements in $\mathbb{Z}[\sqrt{-2}]$. The ideal generated by β is

$$\begin{aligned}(\beta) &= \{k\beta| \quad k \in \mathbb{Z}[\sqrt{-2}]\} \\ &= \{(x+y\sqrt{-2})\beta \quad | \quad x,y \in \mathbb{Z}\} \\ &= \{x\beta + y\beta\sqrt{-2} \quad | \quad x,y \in \mathbb{Z}\} \\ &= \mathcal{L}(\beta, \sqrt{-2}\beta)\end{aligned}$$

Note that (β) is also an orthogonal lattice since the inner product of $\beta = \beta_1 + \beta_2\sqrt{-2}$ and $\sqrt{-2}\beta = -2\beta_2 + \beta_1\sqrt{-2}$ is

$$\begin{aligned}(\beta_1, \beta_2\sqrt{2}) \cdot (-2\beta_2, \sqrt{2}\beta_1) &= -2\beta_1\beta_2 + 2\beta_1\beta_2 \\ &= 0.\end{aligned}$$

We can identify $\mathbb{C}$ with $\mathbb{R}^2$ and plot β as a vector in $\mathbb{R}^2$. So, $\mathcal{L}(\beta, \sqrt{-2}\beta)$ is generated by β and another vector perpendicular to the first one of magnitude $\sqrt{2}\beta$. Recall that multiplication by i rotates the vector $\pi/2$ counter-clockwise. This spans the plane and gives us a rectangular lattice whose vertices are $\mathbb{Z}[\sqrt{-2}]$ multiples of β. On the other hand, $\mathbb{Z}[\sqrt{-2}] = \mathcal{L}(1, \sqrt{-2})$ can be plotted the same way. Note that any $\alpha \in \mathbb{Z}[\sqrt{-2}]$ lies in at least one of the rectangles of the lattice $\mathcal{L}(\beta, \sqrt{-2}\beta)$.

Since every vertex in $\mathcal{L}(\beta, \sqrt{-2}\beta)$ corresponds to a multiple of β, having (36) is equivalent to finding a vector r from a vertex $k\beta$ in $\mathcal{L}(\beta, \sqrt{-2}\beta)$ to α with $d(r) < d(\beta)$. The Figure (1) illustrates the α and the vector r. As it is shown, by choosing $k\beta$ to be the closest lattice vector to α and setting $r = \alpha - k\beta$ we have

$$\begin{aligned}|r|^2 &\leq (|\beta|/2)^2 + \left(|\beta|/\sqrt{2}\right)^2 \\ &= d(\beta)/4 + d(\beta)/2 \\ &= \frac{d(\beta) + 2d(\beta)}{4} \\ &= \frac{3}{4}d(\beta)\end{aligned}$$

Thus we showed that for any β and $\alpha \in \mathbb{Z}[\sqrt{-2}]$ we have

$$\alpha = k\beta + r \quad d(r) \leq \frac{3}{4}d(\beta), \tag{37}$$

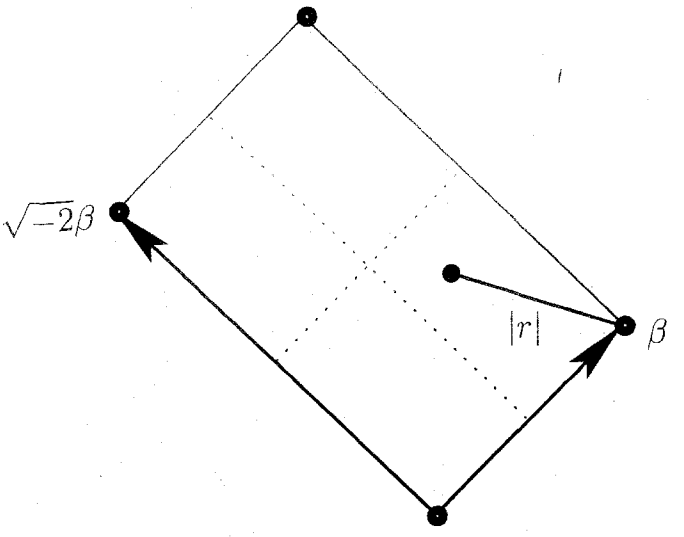

Figure 1: Euclidean Function in $\mathbb{Z}[\sqrt{-2}]$

where $k \in \mathbb{Z}[\sqrt{-2}]$. This is even stronger than what we expect from an Euclidean domain.

Now that we have a geometrical picture of the ring $\mathbb{Z}[\sqrt{-2}]$, let us explain the algebra behind equation (37). Since $\alpha \in span(\beta, \beta\sqrt{-2})$, one can write α as a real linear combination of $\{\beta, \beta\sqrt{-2}\}$:

$$\alpha = \alpha_1\beta + \alpha_2\beta\sqrt{-2} \quad \alpha_1, \alpha_2 \in \mathbb{R}.$$

Set $p = \lfloor \alpha_1 \rceil$ and $q = \lfloor \alpha_2 \rceil$. If α_i is equidistant from two integers, we choose the smaller by convention. Hence, we may write

$$\begin{aligned} \alpha &= (p + p_0)\beta + (q + q_0)\sqrt{-2}\beta & (38) \\ &= (p + q\sqrt{-2})\beta + (p_0 + q_0\sqrt{-2})\beta, & (39) \end{aligned}$$

where $|p_0|, |q_0| \leq 1/2$. Set $k := p + q\sqrt{-2} \in \mathbb{Z}[\sqrt{-2}]$. In fact, by rounding-off the α_i's, we are choosing the closest lattice point $k\beta$ to α. The remainder is $r = \alpha - k\beta =$

$(p_0 + q_0\sqrt{-2})\beta \in \mathbb{Z}[\sqrt{-2}]$ whose norm is

$$\begin{aligned} d(r) &= d(\beta(p_0 + q_0\sqrt{-2})) \\ &= d(\beta)d(p_0 + q_0\sqrt{-2}) \\ &= d(\beta)({p_0}^2 + 2{q_0}^2) \\ &\leq d(\beta)(1/2 + 2/2) \\ &= \frac{3}{4}d(\beta). \end{aligned}$$

To get an efficient algorithm to compute r and k, let us consider the equation (38) in the field $\mathbb{Q}[\sqrt{-2}]$. Using the fact that every nonzero element in $\mathbb{Q}[\sqrt{-2}]$ has a multiplicative inverse, we divide both sides of (38) by β.

$$\begin{aligned} \frac{\alpha}{\beta} &= (p + p_0) + (q + q_0)\sqrt{-2} \\ &= p + q\sqrt{-2} + (p_0 + q_0\sqrt{-2}). \end{aligned}$$

On the other hand, letting $\alpha = a + b\sqrt{-2}$ and $\beta = c + d\sqrt{-2}$ for integers a, b, c and d, we have

$$\begin{aligned} \frac{\alpha}{\beta} &= \frac{a + b\sqrt{-2}}{c + d\sqrt{-2}} \\ &= \frac{ac + 2bd}{d(\beta)} + \frac{bc - ad}{d(\beta)}\sqrt{-2} \end{aligned}$$

Hence,

$$p = \left\lfloor \frac{ac + 2bd}{d(\beta)} \right\rceil \quad and \quad q = \left\lfloor \frac{bc - ad}{d(\beta)} \right\rceil.$$

We summarize our algorithm as follows.

Euclidean Algorithm for $\mathbb{Z}[\sqrt{-2}]$

1. Take $\alpha = a + b\sqrt{-2}$ and $\beta = c + d\sqrt{-2}$ in $\mathbb{Z}[\sqrt{-2}]$ as input.
2. Compute $\alpha_1 = \frac{ac+2bd}{d(\beta)}$ and $\alpha_2 = \frac{bc-ad}{d(\beta)}$.
3. Set $p = \lfloor \frac{ac+2bd}{d(\beta)} \rceil$ *and* $q = \lfloor \frac{bc-ad}{d(\beta)} \rceil$.

4. Set $k := p + q\sqrt{-2}$ and $r := \alpha - k\beta$.

Proposition 7.1.1. *The Euclidean algorithm given for $\mathbb{Z}[\sqrt{-2}]$ is the same as round off algorithm.*

Proof. Denote $\beta \in \mathbb{Z}[\sqrt{-2}]$ by $c + d\sqrt{-2}$ for integers a and b. The ideal (β) is generated by the matrix $R = \begin{pmatrix} \beta \\ \sqrt{-2}\beta \end{pmatrix}$ which can be written as

$$R = \begin{pmatrix} c & d \\ -2d & c \end{pmatrix}$$

with respect to the basis $\{1, \sqrt{-2}\}$ of $\mathbb{Z}[\sqrt{-2}]$. Assume that the element $\alpha = c+d\sqrt{-2}$ in $\mathbb{Z}[\sqrt{-2}]$ is given. According to the round off algorithm, first we represent α as a linear combination of the rows of R i.e., the basis of (β). Computing the inverse of R we get

$$R^{-1} = \frac{1}{d(\beta)} \begin{pmatrix} c & -d \\ 2d & c \end{pmatrix}$$

and multiplying by $\alpha = (a, b)$, we get

$$\left(\frac{ac + 2db}{d(\beta)}, \frac{-ad + bc}{d(\beta)} \right). \tag{40}$$

The components of the (40) are the coefficients in $\mathbb{Q}$ of α written as a linear combination of basis elements. According to the algorithm, linear combination of basis with the rounded off (40) gives us

$$k = \left\lceil \frac{ac + 2db}{d(\beta)} \right\rfloor + \left\lceil \frac{-ad + bc}{d(\beta)} \right\rfloor$$

such that our desired closest vector is $k\beta$ and the shortest representative would be $r = \alpha - k\beta$. A quick comparison with the Euclidean algorithm that we gave in Section 7.1.1 shows that these two methods are the same. □

As we saw in the second chapter, there exist different techniques to estimate the closest vector of a lattice to a given element in the span of the lattice. Among them we saw the rounding off algorithm and the nearest hyper plane algorithm by Babai [8].

Suppose that the integral domain A is an n-dimensional lattice $\mathcal{L}(b_1, \ldots, b_n)$. The ideal generated by an element $\beta \in A$ is an n-dimensional lattice itself $\mathcal{L}(\beta b_1, \ldots, \beta b_n)$. Given an element α in $\mathcal{L}(b_1, \ldots, b_n) \subset span(\mathcal{L})$, the problem of finding the shortest representative of α modulo β can be express as finding a vertex v in (β) such that

$$\alpha = v + r, \qquad r \in span(\mathcal{L}),$$

where the vector r is as short as possible. The proposition above illustrate that over $\mathbb{Z}[\sqrt{-2}]$, it is the same as finding the closest point of the lattice (β) to α.

7.1.2 Inverting a Polynomial in $\mathbb{Z}[\sqrt{-2}]$

In this section, we show how to invert a unit polynomial in

$$\mathbb{Z}[\sqrt{-2}][X]/(X^N - 1).$$

This is a direct use of the method explained in Section 6.2.2. For an arbitrary element $q \in \mathbb{Z}[\sqrt{-2}]$ with prime factorization $p_1^{\alpha_1} \ldots p_n^{\alpha_n}$, suppose that we want to solve the congruence system

$$x \equiv F_{p_i^{\alpha_i}} \mod (p_i^{\alpha_i}).$$

The proof of the Chinese Remainder Theorem gives us a constructive way to find the solution.

Theorem 7.1.2. *Let $m_1, \ldots, m_r$ be relatively prime elements of $\mathbb{Z}[\sqrt{-2}]$ and suppose we have elements $a_1, \ldots, a_r \in \mathbb{Z}[\sqrt{-2}]$. The system of congruences*

$$x \equiv a_i \mod (m_i) \quad 1 \leq i \leq r$$

has a solution $x \in \mathbb{Z}[\sqrt{-2}]$ to all congruences which is unique modulo the product $m_1 \ldots m_r$.

Proof. Let $m = \prod_{i=1}^{r} m_r$. Then $\frac{m}{m_i} \in \mathbb{Z}[\sqrt{-2}]$ and $\gcd(m_i, \frac{m}{m_i}) = 1$. Since $\mathbb{Z}[\sqrt{-2}]$ is a Euclidean domain, by extended Euclidean algorithm there exist u_i and v_i such that

$$\frac{m}{m_i} u_i + m_i v_i = 1. \tag{41}$$

Thus

$$\frac{m}{m_i}u_i \equiv 1 \mod m_i$$

for each i. Set

$$x = \sum_{i=1}^{r} \frac{m}{m_i}u_i a_i.$$

It is easy to check that x satisfies the congruence system and is unique modulo m. □

7.1.3 Decryption Criteria over $\mathbb{Z}[\sqrt{-2}]$

In this section we attempt to choose a suitable set S such that having all coefficients of a in S guarantees a successful decryption. Note that although $\mathbb{Z}[\sqrt{-2}]$ is a Euclidean domain, the quotient and the remainder are not guaranteed to be unique, since as we saw in Theorem 6.2.3, the only ring that has this property is the ring of polynomials over a field. The example below gives two different remainders for the same pair of elements in $\mathbb{Z}[\sqrt{-2}]$.

Example 7.1.3. Suppose $\alpha = \sqrt{-2}$ and $\beta = 1 + \sqrt{-2}$. We have

$$\begin{aligned} \sqrt{-2} &= 0 \cdot (1 + \sqrt{-2}) + \sqrt{-2} \quad d(\sqrt{-2}) = 2 < d(\beta) = 3 \\ &= 1(1 + \sqrt{-2}) - 1 \quad d(-1) = 1 < d(\beta) = 3 \end{aligned}$$

So, the set

$$\{s \in \mathbb{Z}[\sqrt{-2}] | \quad d(s) < d(q)\}$$

is not a good choice. Although by Euclidean algorithm, it contains a representative element of all classes in $\mathbb{Z}[\sqrt{-2}]/(q)$ and thus satisfies the first condition, it fails to fulfill the second condition because it contains more than one element of the same class.

The example below shows that even if we take $t = \frac{3}{4}d(q)$, the bound we found in (37), we would have condition one but not condition two.

Example 7.1.4. Suppose $q = 1 + \sqrt{-2}$ and $\alpha = \sqrt{-2}$. As we saw in Example 7.1.3, $\sqrt{-2} \equiv -1 \mod q$ and they both have norms less that $\frac{3}{4}d(q) = \frac{9}{4}$.

To determine the appropriate set S, observe that any rectangle of the lattice generated by $\{\beta, \sqrt{-2}\beta\}$ in the complex plane makes a complete set of residue modulo β. Figure 7.1.3 depicts four of those squares surrounding zero. We are interested in choosing a set S of complete residues in which the outputs of our algorithm lie. Recall $\beta = c + \sqrt{-2}d$. Define the set rect(β) to be the area in the plane bounded by the lines

$$\begin{aligned} y &= -x + (d+c)/2 \\ y &= -x + (-d+c)/2 \\ y &= x + \sqrt{-2}(c+d)/2 \\ y &= x - \sqrt{-2}(c+d)/2 \end{aligned}$$

So *rect*(β) is a square centered around zero and $\tilde{S} := \text{rect}(\beta) \cap \mathbb{Z}[\sqrt{-2}]$ contains a quarter of each of the surrounding complete residue sets. This makes $\tilde{S}$ a complete set of residues. Moreover, our argument in Section 7.1.1 shows that all the outputs of our Euclidean algorithm lie in $\tilde{S}$.

7.1.4 Probability Argument

The goal of this section is to give a criterion according to which one can determine a good set of parameters that leads to successful decryption with high probability. This is done by finding the probability distributions of the coefficients of the non-modular a. Once this distribution is known, one can compare the probability of having all a_i's in $\tilde{S}$ for different parameters and choose a set of parameters for which this probability is optimum.

Definition 7.1.5. For a positive integer d, let $\mathcal{L}_d$ denote the set of all polynomials in R which have d coefficient equal to 1 $(-1, \sqrt{-2}, -\sqrt{-2})$.

Let us set $\mathcal{L}_f := \mathcal{L}_{d_f}$, $\mathcal{L}_g := \mathcal{L}_{d_g}$ and $\mathcal{L}_\phi := \mathcal{L}_{d_\phi}$ for some integers d_f, d_g, d_ϕ to be determined. From the decryption algorithm we know that

$$a_i = p \sum_{k+l \equiv i} \phi_k g_l + \sum_{k+l \equiv i} f_k m_l \quad 0 \leq i \leq N-1. \tag{42}$$

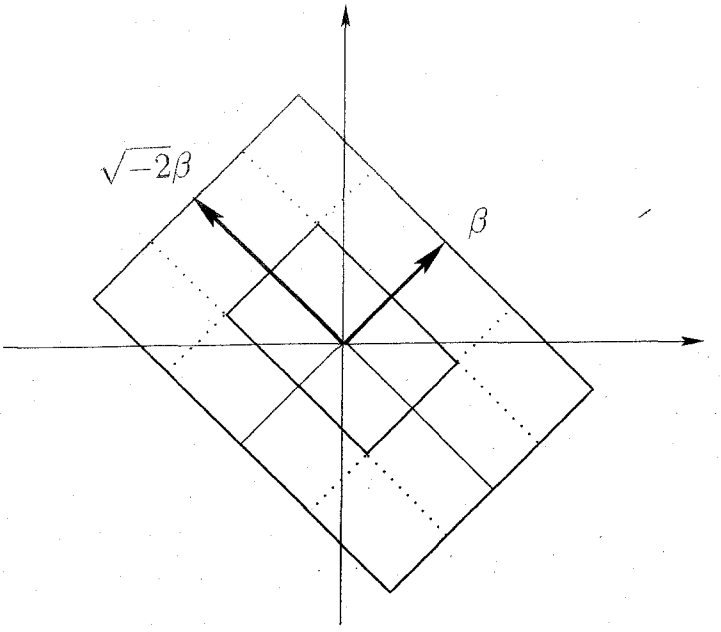

Figure 2: Complete set of residues in $\mathbb{Z}[\sqrt{-2}]$

On the other hand, since a_i is an element of $\mathbb{Z}[\sqrt{-2}]$ it can be written as linear combination of the basis $\{1, \sqrt{-2}\}$ as

$$a_i = R(a_i) + I(a_i)\sqrt{-2}.$$

For sake of simplicity, we look at $R(a_i)$ and $I(a_i)$ as two independent random variables, which is not far from reality. From equation (42) we get

$$\begin{aligned} a_i &= \left(R(p) \sum_{k+l\equiv i} R(\phi_k g_l) - 2I(p) \sum_{k+l\equiv i} I(\phi_k g_l) + \sum_{k+l\equiv i} R(f_k m_l) \right) \\ &+ \left(R(p) \sum_{k+l\equiv i} I(\phi_k g_l) - 2I(p) \sum_{k+l\equiv i} R(\phi_k g_l) + \sum_{k+l\equiv i} I(f_k m_l) \right) \sqrt{-2} \end{aligned} \tag{43}$$

To compute the expected value and variance of $R(a_i)$ we need to have the expected value and variance of each component. In the following pages, we determine the

expected value E and variance var for each component of $R(a_i)$ $(I(a_i)$. By our choice of $\mathcal{L}_g$ and d_ϕ we determine the probabilities:

$$P(R(\phi_k g_l) = 1) = P(R(\phi_k g_l) = -1) = P(R(\phi_k g_l) = -2) = P(R(\phi_k g_l) = 2) = \frac{2d_\phi d_g}{N^2}$$

Thus we have

$$\begin{aligned} E(R(\phi_k g_l)) &= 1(2\frac{d_\phi d_g}{N^2}) - 1(2\frac{d_\phi d_g}{N^2}) + 2(2\frac{d_\phi d_g}{N^2}) - 2(2\frac{d_\phi d_g}{N^2}) \\ &= 0. \end{aligned}$$

The variance of $\phi_k g_l$ is

$$\begin{aligned} var(R(\phi_k g_l)) &= E(R(\phi_k g_l)^2) - (E(R(\phi_k g_l)))^2 \\ &= E((R(\phi_k g_l))^2) - 0 \\ &= 1\left(2\frac{d_\phi d_g}{N^2}\right) + 1\left(2\frac{d_\phi d_g}{N^2}\right) + 4\left(2\frac{d_\phi d_g}{N^2}\right) + 4\left(2\frac{d_\phi d_g}{N^2}\right) \\ &= 20\frac{d_\phi d_g}{N^2}. \end{aligned}$$

Assuming that our random variables are distributed normally, the variance of their sum is the sum of the variances. Since the number of components in the summation over $k + l \equiv i$ for each i is N we have

$$\begin{aligned} var\left(\sum_{k+l\equiv i}(R(\phi_k g_l))\right) &= N\left(\frac{20d_\phi d_g}{N^2}\right) \\ E\left(\sum_{k+l\equiv i}(R(\phi_k g_l))\right) &= 0. \end{aligned} \tag{44}$$

By a similar computation we get

$$P(I(\phi_k g_l) = 1) = P(I(\phi_k g_l) = -1) = \frac{4d_\phi d_g}{N^2}$$

Thus:

$$\begin{aligned} E(I(\phi_k g_l)) &= 1\left(\frac{4d_\phi d_g}{N^2}\right) - 1\left(\frac{4d_\phi d_g}{N^2}\right) = 0 \\ var(I(\phi_k g_l)) &= \frac{8d_\phi d_g}{N^2} \end{aligned}$$

and

$$\begin{aligned} var\left(\sum_{k+l\equiv i} I(\phi_k g_l)\right) &= N\frac{8d_\phi d_g}{N^2} = \frac{8d_\phi d_g}{N} \\ E\left(\sum_{k+l\equiv i} I(\phi_k g_l)\right) &= 0 \end{aligned} \tag{45}$$

Computing the probability distribution of $\sum_{k+l\equiv i} R(f_k m_l)$ is not possible without making some assumption on the choice of coefficients of m. Remember that to have the decrypted m safe and sound at the final reduction modulo p in the decryption process, we need to deal with a similar condition as for the polynomial a. Namely, let $D = \text{rect}(p) \cap \mathbb{Z}[\sqrt{-2}]$ (see Section 7.1.3). Assuming that all the points in the D are equally likely, we can assume that $R(m_l)$ varies from $-\sqrt{2}|p|$ to $+\sqrt{2}|p|$ and $I(m_l)$ varies from $-|p|$ to $|p|$. Thus assuming that $x := R(m_l)$ and $y := I(m_l)$ are distributed uniformly we get

$$F_R(x) = \frac{x+\sqrt{2}|p|}{2\sqrt{2}|p|} \quad \text{and} \quad F_I(y) = \frac{x+|p|}{2|p|},$$

where F is the cumulative distribution function. Thus the density functions are

$$f_R(x) = \frac{1}{2\sqrt{2}|p|} \tag{46}$$

$$f_I(y) = \frac{1}{2|p|} \tag{47}$$

and their expected values and variances are given by

$$\begin{aligned} E(x) &= \int_{-\sqrt{2}|p|}^{\sqrt{2}|p|} x\frac{1}{2\sqrt{2}|p|}dx = 0 \\ var(x) &= \int_{-\sqrt{2}|p|}^{\sqrt{2}|p|} x^2\frac{1}{2\sqrt{2}|p|}dx \\ &= \frac{(2|p|)^2}{3}. \end{aligned}$$

Similarly we get

$$E(y) = 0 \quad \text{and} \quad var(y) = \frac{|p|^2}{3}.$$

Observe that

$$R(f_k m_l) = R(f_k)R(m_l) - 2I(f_k)I(m_l)$$

On the other hand, the set of all possible nonzero values for f_k is

$$\{\pm 1, \pm\sqrt{-2}\}$$

according to the choice of $\mathcal{L}_f$. So the set of all possible nonzero values for $R(f_k m_l)$ is

$$\{\pm R(m_l), \pm 2\sqrt{2} I(m_l)\}$$

each occurring with probability $\frac{d_f}{N}$. This set can be found similarly for $I(f_k m_l)$. From equations (46) and (47) and the set above, we conclude that the probability distribution of $R(f_k m_l)$ is either $f(x)$ or $f(y)$ with probability $\frac{2d_f}{N}$ each.

Finally, from equations (43), (44), (45) we get

$$\sigma_1 = var(R(a_i)) = (R(p))^2 \left(\frac{20 d_\phi d_g}{N}\right) + 4(I(p))^2 \left(\frac{8 d_\phi d_g}{N}\right) + var\left(\sum_{k+l\equiv i} R(f_k m_l)\right)$$

and similarly we get

$$\sigma_2 = var(I(a_i)) = (R(p))^2 \left(\frac{8 d_\phi d_g}{N}\right) + 4(I(p))^2 \left(\frac{20 d_\phi d_g}{N}\right) + var\left(\sum_{k+l\equiv i} I(f_k m_l)\right)$$

To find the probability that $a_i \in S$, we assume that the real and imaginary parts of a_i's are normally distributed independent random variables with mean and variance as determined above. As above, relying on this assumption, for simplicity, we can assume that S is a square parallel to the axes. Then, we have

$$P(a_i \in S) = P(-\sqrt{2}|\beta|/2 < I(a_i) < \sqrt{2}|\beta|/2 \quad \text{and} \quad |\beta|/2 < R(a_i) < |\beta|/2).$$

and we get

$$P(-\frac{\sqrt{2}|\beta|}{2} < I(a_i) < \frac{\sqrt{2}|\beta|}{2}) = \int_{-\frac{\sqrt{2}|\beta|}{2}}^{\frac{\sqrt{2}|\beta|}{2}} \frac{1}{\sigma_1}\phi(\frac{I(a_i)}{\sigma_1})dI(a_i),$$

and

$$P(\frac{|\beta|}{2} < R(a_i) < \frac{|\beta|}{2}) = \int_{-\frac{|\beta|}{2}}^{\frac{|\beta|}{2}} \frac{1}{\sigma_2}\phi(\frac{R(a_i)}{\sigma_2})dR(a_i),$$

where

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}.$$

One can use the above probability distribution to choose the parameters d_g, d_f, d_ϕ, p and q in optimal way.

7.2 ETRU over Eisenstein Integers

A natural next choice of NTRU base ring after $\mathbb{Z}$, $\mathbb{Z}[i]$ and $\mathbb{Z}[\sqrt{-2}]$ would seem to be $\mathbb{Z}[\sqrt{-3}]$. However, as a result of Subsection 6.2.2 and Theory A.0.17 we need to have unique factorization property. The below example shows that this property fails in $\mathbb{Z}[\sqrt{-3}]$.

Example 7.2.1. Consider the element 4 in $\mathbb{Z}[\sqrt{-3}]$. It can be factored in two different ways as follows

$$4 = 2 \times 2 = (1 - \sqrt{-3})(1 + \sqrt{-3})$$

Note that by the norm inherited from $Q(\sqrt{-3})$, we have:

$$d(2) = 4$$

$$d(1 \pm \sqrt{-3}) = 1 + 3 = 4$$

Since 4 is not divisible by any smaller integer of the form $a^2 + 3b^2$ except 1, we deduce that 2 and $1 \pm \sqrt{-3}$ are all primes.

This can be repaired by enlarging $\mathbb{Z}[\sqrt{-3}]$ to $\mathbb{Z}[\zeta_3]$ where $\zeta_3 = \frac{-1+\sqrt{-3}}{2}$. To be more precise, we choose to work with the ring of integers of the field $\mathbb{Q}(\sqrt{-3})$ which is a Dedekind domain by Theorem A.0.14, and for which our inverting argument works as a result. In fact, $\mathbb{Z}[\zeta_3]$ is a Euclidean domain.

7.2.1 $\mathbb{Z}[\zeta_3]$ is a Euclidean Domain

First observe that $\mathbb{Z}[\zeta_3]$ is a lattice in $\mathbb{C} \cong \mathbb{R}^2$ generated by $\{1, \zeta_3\}$. So every element can be written as $a + b\zeta_3$ for some integers a and b. Define the norm d on an element $x = a + b\zeta_3 \in \mathbb{Z}[\zeta_3]$ by

$$d(x) = a^2 + b^2 - ab$$

The lemma below justifies the choice of d.

Lemma 7.2.2. *The norm of x as an element in $\mathbb{Q}[\zeta_3]$ is $a^2 + b^2 - ab$. This is also $|x|^2$, where x is viewed as an element of $\mathbb{C} \cong \mathbb{R}^2$.*

Proof. The minimal polynomial of ζ_3 over $\mathbb{Q}$ is the cyclotomic polynomial $m_{\zeta_3,\mathbb{Q}} = X^2 + X + 1$. Thus, there exist exactly two monomorphisms (isomorphisms in this case) from $\mathbb{Q}$ to $\mathbb{C}$ fixing $\mathbb{Q}$ and permuting the roots of $m_{\zeta_3,\mathbb{Q}}$. Since $m_{\zeta_3,\mathbb{Q}}$ has two roots ζ_3 and ζ_3^2, the isomorphisms are

$$\sigma_1(a + b\zeta_3) = a + b\zeta_3 \quad \text{and} \quad \sigma_2(a + b\zeta_3) = a + b(\zeta_3^2),$$

where $a, b \in \mathbb{Q}$. By definition, the norm of $x = a + b\zeta_3$ is

$$\begin{aligned} N(x) &= \sigma_1(x)\sigma_2(x) \\ &= (a + b\zeta_3)(a + b\zeta_3^2). \end{aligned}$$

Note that $\zeta_3^2 = \bar{\zeta_3}$ and $\zeta_3 + \bar{\zeta_3} = -1$. So we have

$$\begin{aligned} N(x) &= (a + b\zeta_3)(a + b\bar{\zeta_3}) \\ &= a^2 + b^2 + ab(\zeta_3 + \bar{\zeta_3}) \\ &= a^2 + b^2 - ab. \end{aligned} \tag{48}$$

Now we show that $d(x) = N(x) = |x|^2$.

$$\begin{aligned} |x|^2 &= |a + b\zeta_3|^2 \\ &= \left|a + b(\frac{-1 + \sqrt{3}i}{2})\right| \\ &= \left|a - \frac{b}{2} + \frac{b\sqrt{3}i}{2}\right|^2 \\ &= \left(a - \frac{b}{2}\right)^2 + \left(\frac{b\sqrt{3}}{2}\right)^2 \\ &= a^2 + b^2 - ab. \end{aligned}$$

□

Remark 7.2.3. Comparing to the formula we obtained for the norm of an element in $\mathbb{Q}[\sqrt{-2}]$, and knowing that $\mathbb{Q}[\sqrt{-3}] = \mathbb{Q}[\zeta_3]$, one might expect $N(x) = a^2 + 3b^2$ if $x = a + b\sqrt{-3}$. So it is worth working out that these are two representations of the same norm.

Lemma 7.2.4. *Let $x = a' + b'\sqrt{-3} \in \mathbb{Q}[\sqrt{-3}]$, then $N(x) = a'^2 + 3b'^2$*

Proof. Let x be an element of $\mathbb{Q}[\sqrt{-3}] = \mathbb{Q}[\zeta_3]$. Thus it can be written as

$$a + b\zeta_3 = a' + b'\sqrt{-3}$$

for some a, b, a' and b' in $\mathbb{Q}$. Substituting ζ_3 by $\frac{-1+\sqrt{-3}}{2}$, we get $a = a' + b'$ and $b = 2b'$. Using the equation (48) we get that $N(x)$ in terms of a' and b' is

$$\begin{aligned} N(x) &= (a' + b')^2 + (2b')^2 - 2(a' + b')b' \\ &= a'^2 + 3b'^2 \end{aligned}$$

□

For an element $\beta \in \mathbb{Z}[\zeta_3]$ the ideal generated by β is

$$\begin{aligned} (\beta) &= \{k\beta \mid k \in \mathbb{Z}[\zeta_3]\} \\ &= \{(x + y\zeta_3)\beta \mid x, y \in \mathbb{Z}\} \\ &= \{x\beta + y\beta\zeta_3 \mid x, y \in \mathbb{Z}\} \\ &= \mathcal{L}(\beta, \beta\zeta_3). \end{aligned}$$

Thus (β) is a lattice generated by β times a basis of $\mathcal{L}(1, \zeta_3)$ and it has the same shape as $\mathcal{L}(1, \zeta_3)$. Note that $\mathcal{L}(1, \zeta_3)$ is an equilateral triangle shaped lattice with unit side size, as it is shown in the Figure 3. Thus, (β) is a equilateral triangle shaped lattice of side length $|\beta|$.

Each α in $\mathbb{Z}[\zeta_3]$ lies in some equilateral triangle whose vertices lie in the ideal generated by (β). The distance of α from each of the vertices is less than the side length of the triangle. Suppose the closest vertex is $k\beta$, for some $k \in \mathbb{Z}[\zeta_3]$. Then, setting $r = \alpha - k\beta$ we have,

$$\alpha = k\beta + r, \quad k \in \mathbb{Z}[\zeta_3]$$

and by the above argument we have

$$|r| \leq |\beta|. \tag{49}$$

Therefore, the second condition of Definition A.0.2 is satisfied; the first condition is the result of the multiplicativity of the norm. We have thus shown that $\mathbb{Z}[\zeta_3]$ is norm Euclidean.

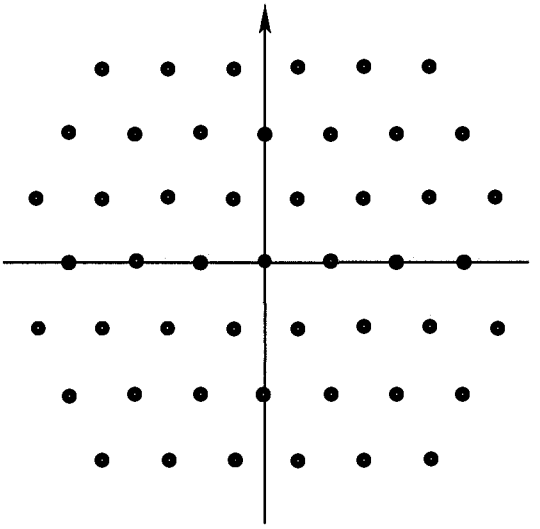

Figure 3: $\mathbb{Z}[\zeta_3]$

7.2.2 A Euclidean Algorithm for $\mathbb{Z}[\zeta_3]$

Suppose that α and β are two elements of $\mathbb{Z}[\zeta_3]$ and

$$\alpha = a + b\zeta_3 \quad \text{and} \quad \beta = c + d\zeta_3,$$

for some $a, b, c, d \in \mathbb{Z}$. Since $\mathbb{Z}[\zeta_3] \subset \mathbb{Q}[\zeta_3]$ we have $\alpha, \beta \in \mathbb{Q}[\zeta_3]$ and

$$\frac{\alpha}{\beta} = \frac{a + b\zeta_3}{c + d\zeta_3}$$

$$\begin{aligned} &= \frac{a+b\zeta_3}{c+d\zeta_3} \times \frac{c+d\bar{\zeta_3}}{c+d\bar{\zeta_3}} \\ &= \frac{(a+b\zeta_3)(c+d\bar{\zeta_3})}{c^2+(\zeta_3+\bar{\zeta_3})cd+(\zeta_3\bar{\zeta_3})d^2} \\ &= \frac{(ac+bd-ad)+(bc-ad)\zeta_3}{a^2+d^2-cd} \\ &= \frac{ac+bd-ad}{d(\beta)} + \frac{bc-ad}{d(\beta)}\zeta_3. \end{aligned}$$

Above, we used the fact that $\bar{\zeta_3}$ is the only other root of X^2+X+1 which implies that $\zeta_3+\bar{\zeta_3}$ and $\zeta_3\bar{\zeta_3}$ are integers. Take

$$p := \left\lfloor \frac{ac+bd-ad}{d(\beta)} \right\rceil \quad \text{and} \quad q := \left\lfloor \frac{bc-ad}{d(\beta)} \right\rceil.$$

where $\lfloor x \rceil$ is the nearest integer to x. If x is equidistant from two integers, we choose the smaller one by convention. Now set

$$k := p + q\zeta_3 \quad and \quad r = \alpha - k\beta$$

Therefore we have

$$\alpha = k\beta + r$$

To show that $d(r) < d(\beta)$, observe that

$$\frac{ac+bd-ad}{d(\beta)} = p + p_0 \quad \text{and} \quad \frac{bc-ad}{d(\beta)} = q + q_0,$$

where $|p_0|, |q_0| \leq 1/2$. Thus we have

$$\begin{aligned} \frac{\alpha}{\beta} &= (p+p_0) + (q+q_0)\zeta_3 \\ &= p + q\zeta_3 + (p_0 + q_0\zeta_3). \end{aligned}$$

Thus by multiplying both sides by β we get

$$\alpha = k\beta + \beta(p_0 + q_0\zeta_3).$$

Therefore $r = \beta(p_0 + q_0\zeta_3)$ and

$$\begin{aligned} d(r) &= d(\beta(p_0 + q_0\zeta_3)) \\ &= d(\beta)d(p_0 + q_0\zeta_3) \\ &= d(\beta)({p_0}^2 + {q_0}^2 - p_0q_0) \\ &\leq d(\beta)(1/4 + 1/4 + 1/4) \\ &= \frac{3}{4}d(\beta). \end{aligned}$$

Proposition 7.2.5. *The Euclidean algorithm given for $\mathbb{Z}[\zeta_3]$ is the same as the round off algorithm.*

Proof. Let α and β be two elements represented with respect to the basis $\{1, \zeta_3\}$ as $a + b\zeta_3$ and $c + d\zeta_3$ respectively. The ideal generated by (β) makes a lattice $\mathcal{L}(\beta, \zeta_3\beta)$. Noting that

$$\begin{aligned} \zeta_3\beta &= \zeta_3(c + d\zeta_3) \\ &= d\zeta_3^2 + c\zeta_3 \\ &= -d + (c - d)\zeta_3, \end{aligned}$$

this basis can be written in the matrix form relative to the basis $\{1, \zeta_3\}$ as

$$R = \begin{pmatrix} c & d \\ -d & c - d \end{pmatrix}.$$

Computing the inverse of R we have

$$R^{-1} = \frac{1}{d(\beta)} \begin{pmatrix} c - d & -d \\ d & c \end{pmatrix}$$

and multiplying by α we get

$$\frac{1}{d(\beta)}(a, b) \begin{pmatrix} c - d & -d \\ d & c \end{pmatrix} = \left(\frac{ac - ad + bd}{d(\beta)}, \frac{bc - ad}{d(\beta)} \right)$$

Thus by the Babai round off algorithm, the closest vector in the lattice is

$$v = \left(\left\lceil \frac{ac - ad + bd}{d(\beta)} \right\rfloor + \left\lceil \frac{bc - ad}{d(\beta)} \right\rfloor \zeta_3 \right) \beta$$

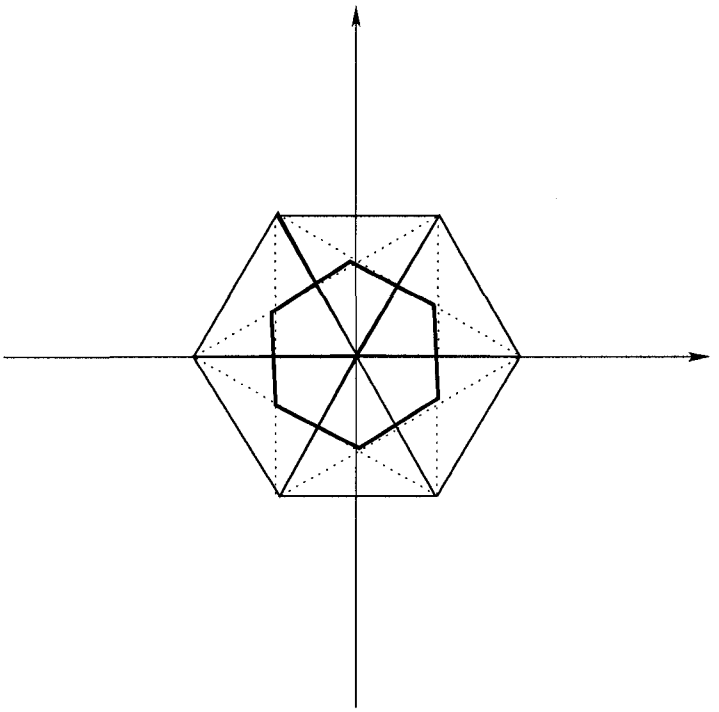

Figure 4: S for $\mathbb{Z}[\zeta_3]$

and the shortest coset representative is

$$\alpha - v,$$

which is the same result we got in the Euclidean algorithm over $\mathbb{Z}[\zeta_3]$. □

The decryption criteria discussion for $\mathbb{Z}[\zeta_3]$ is quite similar to what we gave for $\mathbb{Z}[\sqrt{-2}]$. The set S here is a centered inner hexagon shown in Figure 4. It follows from our Euclidean algorithm (or equivalently from the round off algorithm) that this set is a complete set of residues.

Finally, to choose optimal parameters for NTRU over $\mathbb{Z}[\zeta_3]$, one should calculate the probability distribution of the coefficients of a as in Section 7.1.4. However, let us instead go on to a more complex example.

7.3 Generalization to the Ring of Integers of a Cyclotomic Field

The possibility of using CVP approximation techniques opens the avenue of considering the ring of integers of cyclotomic fields as next natural step for generalization of NTRU. This family of $\phi(n)$ dimensional lattices are Dedekind domains by Theorem A.0.14. However, in general, $\mathbb{Z}[\zeta_n]$ is not a Euclidean domain. But it is known that for n chosen from the set below, $\mathbb{Z}[\zeta_n]$ is norm-Euclidean [29]:

$$\{1, 3, 4, 5, 7, 8, 9, 11, 12, 13, 14, 15, 16, 20, 21, 24\}$$

In fact, it is known that there are precisely 30 values of n, $n \not\equiv 2(\mod 4)$ for which $\mathbb{Z}[\zeta_n]$ is a principal ideal domain [31]. They are:

$$\{1, 3, 4, 5, 7, 8, 9, 11, 12, 13, 15, 16, 17, 19, 20,$$
$$21, 24, 25, 27, 28, 32, 33, 35, 36, 40, 44, 45, 48, 60, 84\}.$$

Thus, the smallest integer for which $\mathbb{Z}[\zeta_n]$ is a Dedekind domain but known not to be a PID is 23. The example we discuss here is the norm-Euclidean integral domain $\mathbb{Z}[\zeta_5]$. Note that $\mathbb{Z}[\zeta_4]$ is the ring of Gaussian integers which has been well discussed in [25].

7.3.1 Ring of Integers of $\mathbb{Q}[\zeta_5]$

In this subsection, we show that $\mathbb{Z}[\zeta_5]$ is norm-Euclidean and we give a Euclidean algorithm for division in this ring. We use the following proposition.

Proposition 7.3.1. *Let O be the ring of integers in a number field K. Then O is norm-Euclidean if and only if for every $y \in K$ there exist a value $x \in O$ such that $N(y - x) < 1$.*

Proof. Suppose that O is norm-Euclidean. Since K is the quotient field of O, we can write a given $y \in K$ as $y = \frac{a}{b}$, $a, b \in O$, $b \neq 0$. Since O is norm-Euclidean, there exist $q, r \in O$ such that

$$a = qb + r \qquad N(r) < N(b)$$

Dividing both sides by b we get

$$\frac{a}{b} - q = \frac{r}{b}$$

Taking the norm on both sides and using the multiplicativity of the norm, we get:

$$\begin{aligned} N\left(\frac{a}{b} - q\right) &= N\left(\frac{r}{b}\right) \\ &= \frac{N(r)}{N(b)} \\ &< 1. \end{aligned}$$

Thus, setting $x = q$ we get the desired property.

Conversely, given $a, b \in O$, $b \neq 0$, the quotient $\frac{a}{b}$ is in K and by assumption we can find $q \in O$ such that $N(\frac{a}{b} - q) < 1$. Setting $r = a - qb$, we have

$$\begin{aligned} N(r) &= N(a - qb) \\ &= N\left(b\left(\frac{a}{b} - q\right)\right) \\ &= N(b)N\left(\frac{a}{b} - q\right) \\ &< N(b). \end{aligned}$$

So, we have

$$a = qb + r \qquad N(r) < N(b),$$

which means O is norm-Euclidean. □

In order to be able to use the above proposition to deduce that $\mathbb{Z}[\zeta_5]$ is norm-Euclidean, we first develop an algorithm to divide two elements in $\mathbb{Q}[\zeta_5]$. We assume that the reader is familiar with basic Galois theory such as covered in [12]. Recall that $\mathbb{Q}[\zeta_5]$ is an algebraic extension of $\mathbb{Q}$ of degree $[\mathbb{Q}[\zeta_5] : \mathbb{Q}] = 4$. Since ζ_5 is a primitive 5th root of unity with minimal polynomial $m_{\mathbb{Q},\zeta_5} = X^4 + X^3 + X^2 + X + 1$, the set $\{1, \zeta_5, \zeta_5^2, \zeta_5^3\}$ forms a basis of $\mathbb{Q}[\zeta_5]$ over $\mathbb{Q}$.

Proposition 7.3.2. *The Galois group of* $\mathbb{Q}[\zeta_5]$ *over* $\mathbb{Q}$ *is isomorphic to the cyclic multiplicative group* $(\mathbb{Z}/5\mathbb{Z})^*$.

Proof. Define a map ϕ by

$$\begin{aligned} \phi : (\mathbb{Z}/5\mathbb{Z})^* = \{1,2,3,4\} &\rightarrow Gal(\mathbb{Q}[\zeta_n]/\mathbb{Q}) \\ i &\rightarrow \sigma_i \end{aligned}$$

where $\sigma_i : \zeta_5 \rightarrow \zeta_5^i$. Since ζ_5 generates $\mathbb{Q}[\zeta_5]$ over $\mathbb{Q}$, σ_i is fully determined by its action on ζ_5. Since

$$\begin{aligned} \sigma_i\sigma_j(\zeta_5) &= \sigma_i(\zeta_5^j) \\ &= \zeta_5^{ij} \\ &= \sigma_{ij}(\zeta_5), \end{aligned}$$

and $\zeta_5^5 = 1$, having $\phi(i)\phi(j) = \sigma_i\sigma_j$ and $\phi(ij) = \sigma_{ij}$, we deduce that ϕ is an homomorphism. This map in injective and the surjectivity follows from the fact that both $(\mathbb{Z}/5\mathbb{Z})^*$ and $Gal(\mathbb{Q}[\zeta_n]/\mathbb{Q})$ have the same cardinality. Thus, $Gal(\mathbb{Q}[\zeta_n]/\mathbb{Q}) \cong (\mathbb{Z}/5\mathbb{Z})^*$. □

Corollary 7.3.3. *$Gal(\mathbb{Q}[\zeta_n]/\mathbb{Q})$ is generated by σ_2.*

Proof. Since 2 is a generator of $(\mathbb{Z}/5\mathbb{Z})^*$, its image under ϕ which is σ_2 is a generator for $Gal(\mathbb{Q}[\zeta_n]/\mathbb{Q})$. □

By the fundamental theorem of Galois theory [12, Theorem14.14], there exists only one nontrivial subfield of $\mathbb{Q}[\zeta_5]$ over $\mathbb{Q}$, which is the fixed field of $< \sigma_4 >= \{1, \sigma_4\}$.

Lemma 7.3.4. *The fixed field of $< \sigma_4 >$ is $\mathbb{Q}[\sqrt{5}]$.*

Proof. Let F denote the fixed field of $< \sigma_4 >$ and write $\mathbb{Z}_n$ for $\mathbb{Z}/n\mathbb{Z}$. By the fundamental theorem of Galois theory, we have the degree of F over Q is equal to the index of the subgroup $< \sigma_4 >$ in $Gal(\mathbb{Q}[\zeta_n]/\mathbb{Q})$ which is $[\mathbb{Z}_4 : \mathbb{Z}_2] = 2$. Moreover, since $\mathbb{Z}_2$ is a normal subgroup, F over $\mathbb{Q}$ is Galois with Galois group $Gal(F/\mathbb{Q}) = \mathbb{Z}_4/\mathbb{Z}_2 \cong \mathbb{Z}_2$.

Consider the element $\alpha := \zeta_5 + \zeta_5^4$. This element is in if F because

$$\begin{aligned} \sigma_4(\alpha) &= \sigma_4(\zeta_5 + \zeta_5^4) \\ &= \sigma_4(\zeta_5) + \sigma_4(\zeta_5^4) \\ &= \zeta_5^4 + \zeta_5 \\ &= \alpha. \end{aligned}$$

So we have $\mathbb{Q}[\alpha] \subseteq F$. On the other hand, using the fact that ζ_5 is the root of $X^4 + X^3 + X^2 + X + 1$ we get

$$\begin{aligned} \alpha^2 &= (\zeta_5 + \zeta_5^4)^2 \\ &= \zeta_5^2 + \zeta_5^3 + 2 \\ &= -\zeta_5 - \zeta_5^4 - 1 + 2 \\ &= -\alpha + 1, \end{aligned} \tag{50}$$

so α is a root of the polynomial $m(X) = X^2 + X - 1$. We see directly that the other root is $\overline{\alpha} = -1 - \alpha$. Therefore, the degree of $\mathbb{Q}[\alpha]$ over $\mathbb{Q}$ is 2, which implies that $\mathbb{Q}[\alpha] = F$ and $\{1, \alpha\}$ is a basis for F over $\mathbb{Q}$. Using the quadratic formula, we get the two roots $\frac{-1 \pm \sqrt{5}}{2}$. The positive of these is α so $\sqrt{5} = 2\alpha + 1$. So, $\mathbb{Q}[\sqrt{5}] \subseteq F$ and since $\sqrt{5}$ is a root of the quadratic polynomial, $X^2 - 5$, with same argument as above we conclude that $F = \mathbb{Q}[\sqrt{5}]$ and $\{1, \sqrt{5}\}$ is a basis for F over $\mathbb{Q}$. □

Since $F \subseteq \mathbb{R}$ is a non-complex field, ζ_5 is not an element of F. Thus, adjoining ζ_5 to F gives us a field extension of degree greater than one over F. To compute the minimal polynomial of ζ_5 over F observe that

$$\begin{aligned} \zeta_5^2 &= \zeta_5^2 + 1 - 1 \\ &= \zeta_5^2 + \zeta_5^5 - 1 \\ &= (\zeta_5 + \zeta_5^4)\zeta_5 - 1 \\ &= \alpha\zeta_5 - 1. \end{aligned} \tag{51}$$

Therefore, ζ_5 is a root of the polynomial $m' = X^2 - \alpha X + 1$ over F. Since ζ_5 is not in F, m' is irreducible over F and the set $\{1, \zeta_5\}$ forms a basis of $\mathbb{Q}[\zeta_5]$ over F. Therefore,

the degree of $F[\zeta_5]$ over F is two. By the tower property of field extensions, we get $F[\zeta_5] = \mathbb{Q}[\zeta_5]$.

From Lemma 7.3.4 and the argument above we conclude that the set $\{1, \alpha, \alpha\zeta_5, \zeta_5\}$ is a basis for $\mathbb{Q}[\zeta_5]$ over $\mathbb{Q}$. We next use this basis derived from the intermediate field F to find a formula for dividing two elements in $\mathbb{Q}[\zeta_5]$.

Division in $\mathbb{Q}[\zeta_5]$

Given two elements r and $s \neq 0$ in $\mathbb{Q}[\zeta_5]$, we want to express $\frac{r}{s}$ as a linear combination of the basis elements of $\mathbb{Q}[\zeta_5]$ with coefficients in $\mathbb{Q}$. We will use the following lemma in this process.

Lemma 7.3.5. *Assume that F is a field and u, v are two distinct roots of a monic irreducible quadratic polynomial p over F, which are not necessarily in F. Then for every a, b in F there exists a $w \in F$ such that*

$$\frac{1}{a+bu} = \frac{a+bv}{w}.$$

Proof. By multiplying the numerator and denominator by $a + bv$ we get

$$\begin{aligned} \frac{1}{a+bu} &= \frac{a+bv}{(a+bu)(a+bv)} \\ &= \frac{a+bv}{a^2+ab(u+v)+b^2(uv)}. \end{aligned}$$

If $u, v \in F$, the result is obvious. Otherwise, $u, v \in F[u]$ and we need the fact that

$$N_F(\chi) = \prod_{\sigma \in Aut(F[u]/F)} \sigma(\chi) \quad \text{and} \quad \text{Trace}_F(\chi) = \sum_{\sigma \in Aut(F[u]/F)} \sigma(\chi)$$

are quantities in F and since p is quadratic with u and v as roots, the two automorphisms are

$$\sigma_1 = id \quad \text{and} \quad \sigma_2(u) = v, \quad \sigma_2(v) = u$$

It follows that $u + v$ and uv are in F. Thus, setting

$$w = a^2 + ab(u+v) + b^2(uv)$$

we get the desired result. □

The elements r and s can be written as linear combinations of 1 and ζ_5 with coefficients in F as follows

$$\frac{r}{s} = \frac{r_1 + r_2\zeta_5}{s_1 + s_2\zeta_5}.$$

In this way, we reduce the problem of division in $\mathbb{Q}[\zeta_5]$ to division in F. Now, ζ_5 is one root of $m' = X^2 - \alpha X + 1$. In order to apply Lemma 7.3.5, we need to compute the other root of the polynomial m'. By the quadratic formula, we get

$$\begin{aligned} x &= \frac{\alpha \pm \sqrt{\alpha^2 - 4}}{2} \\ &= \frac{\zeta_5 + \zeta_5^4 \pm \sqrt{\zeta_5^2 + \zeta_5^3 + 2 - 4}}{2} \\ &= \frac{\zeta_5 + \zeta_5^4 \pm \sqrt{(\zeta_5 - \zeta_5^4)^2}}{2} \\ &= \frac{\zeta_5 + \zeta_5^4 \pm \zeta_5 - \zeta_5^4}{2} \\ &= \begin{cases} \zeta_5 \\ \zeta_5^4 \end{cases}. \end{aligned}$$

Having $N(\zeta_5) = \zeta_5\zeta_5^4 = 1$ and $\text{Trace}(\zeta) = \zeta_5 + \zeta_5^4 = \alpha \in F$. It follows from the proof of Lemma 7.3.5 that $w = s_1^2 + s_2^2 + \alpha s_1 s_2$ and that

$$\begin{aligned} \frac{r}{s} &= \frac{(r_1 + r_2\zeta_5)(s_1 + s_2\zeta_5^4)}{w} \\ &= \frac{r_1' + r_2'\zeta_5}{w} \\ &= \frac{r_1'}{w} + \frac{r_2'}{w}\zeta_5 \end{aligned}$$

Now in each component we have division in F. So we reduced the problem to the issue of dividing in F.

Assume that two elements of F are given. We can write them as linear combination of the basis $\{1, \alpha\}$ over $\mathbb{Q}$ as $w + v\alpha$ and $m + n\alpha$ with $w, v, m, n \in \mathbb{Q}$. Then we wish to calculate

$$\frac{w + v\alpha}{m + n\alpha}.$$

Similar to what we did above, we want to eliminate α from the denominator to reduce to division over $\mathbb{Q}$. Knowing that the other root of $m_{\alpha,\mathbb{Q}} = X^2 + X - 1$ is

$\overline{\alpha} = -1 - \alpha$, and $\alpha\overline{\alpha} = -\alpha - \alpha^2 = -1$, we get

$$\begin{aligned} \frac{w + v\alpha}{m + n\alpha} &= \frac{w + v\alpha}{m + n\alpha} \times \frac{m + n\bar{\alpha}}{m + n\bar{\alpha}} \\ &= \frac{wm - vn + wn\bar{\alpha} + vm\alpha}{m^2 - n^2 - mn}. \end{aligned}$$

Since $\overline{\alpha} = -1 - \alpha$, this simplifies to

$$\begin{aligned} \frac{w + v\alpha}{m + n\alpha} &= \frac{wm - vn - wn + (vm - wn)\alpha}{m^2 - n^2 - mn} \\ &= \frac{wm - vn - wn}{m^2 - n^2 - mn} + \frac{vm - wn}{m^2 - n^2 - mn}\alpha. \end{aligned}$$

The resulting coefficients are in $\mathbb{Q}$.

We have thus developed an algorithm for division in $\mathbb{Q}[\zeta_5]$.

7.3.2 $\mathbb{Z}[\zeta_5]$ is norm-Euclidean

The proof of the following theorem closely follows the one given in [5].

Proposition 7.3.6. *For every $y \in \mathbb{Q}[\zeta_5]$, there exist an element $x \in \mathbb{Z}[\zeta_5]$ such that $N(y - x) < 1$.*

Proof. Suppose that $y \in \mathbb{Q}[\zeta_5]$ is given. We can write y as linear combination of the basis $\{1, \zeta_5, \zeta_5^2, \zeta_5^3\}$ plus the extra element ζ_5^4.

$$y = y_0 + y_1\zeta_5 + y_2\zeta_5^2 + y_3\zeta_5^3 + y_4\zeta_5^4.$$

Each y_i can be written as

$$y_i = \lfloor y_i \rceil + b_i,$$

where $|b_i| \leq \frac{1}{2}$. Let β and C be

$$\begin{aligned} \beta &= b_0 + b_1\zeta_5 + b_2\zeta_5^2 + b_3\zeta_5^3 + b_4\zeta_5^4 \\ C &= \lfloor y_0 \rceil + \lfloor y_1 \rceil\zeta_5 + \lfloor y_2 \rceil\zeta_5^2 + \lfloor y_3 \rceil\zeta_5^3 + \lfloor y_4 \rceil\zeta_5^4. \end{aligned} \tag{52}$$

Thus,

$$y = C + \beta, \quad \text{where} \quad C \in \mathbb{Z}[\zeta_5] \quad \text{and} \quad \beta \in \mathbb{Q}[\zeta_5].$$

All we need is to show that β has norm less or equal to 1.

At least three of the coefficients of the linear combination (52) have the same sign. Without loss of generality, we assume that the three first coefficients are positive numbers. Since ζ_5 is a root of the polynomial $1 + X + X^2 + X^3 + X^4$ we have $1 + \zeta_5 + \zeta_5^2 + \zeta_5^3 + \zeta_5^4 = 0$. So we can write

$$\begin{aligned}
\beta &= (b_0 - 1/4) + (b_1 - 1/4)\zeta_5 + (b_2 - 1/4)\zeta_5^2 + (b_3 - 1/4)\zeta_5^3 + (b_4 - 1/4)\zeta_5^4 \\
&+ 1/4(1 + \zeta_5 + \zeta_5^2 + \zeta_5^3 + \zeta_5^4) \\
&= (b_0 - 1/4) + (b_1 - 1/4)\zeta_5 + (b_2 - 1/4)\zeta_5^2 + (b_3 - 1/4)\zeta_5^3 + (b_4 - 1/4)\zeta_5^4 \\
&= a_0 + a_1\zeta_5 + a_2\zeta_5^2 + a_3\zeta_5^3 + a_4\zeta_5^4
\end{aligned}$$

Now, for the first three coefficients we have

$$|a_i| \leq 1/4 \quad i = 0, 1, 2,$$

but for $i = 3$ and 4, a_i might be less than $\frac{-1}{2}$. To those coefficients for which it is the case, we add 1 to make it less than $1/2$ in absolute value and subtract 1 from the corresponding $\lfloor y_i \rceil$ to keep the y_i fixed. Thus, now without loss of generality, we can assume that

$$|a_i| \leq 1/2 \quad i = 3, 4.$$

Let F be the fixed field of $< \sigma_4 >$. By the fundamental theorem of Galois theory [12, Theorem 14.14], $< \sigma_4 >$ is the Galois group of $\mathbb{Q}(\zeta_5)$ over F. Thus

$$N_{\mathbb{Q}(\zeta_5)/F}(\beta) = \sigma_1(\beta)\sigma_4(\beta).$$

Note that this is an element in F. Since F is an extension of degree two over $\mathbb{Q}$, $< \sigma_4 >$ is a normal subgroup of $< \sigma_2 >$ which implies that F is Galois over $\mathbb{Q}$ with Galois group $< \sigma_2 > / < \sigma_4 >$. We show that σ_2 is an element of $Gal(F/\mathbb{Q})$. Since σ_2 fixes $\mathbb{Q}$, it is enough to show that it maps α to another root of its minimal polynomial over $\mathbb{Q}$ (see Lemma 7.3.4).

$$\begin{aligned}
\sigma_2(\alpha) &= \sigma_2(\zeta_5 + \zeta_5^4) \\
&= \zeta_5^2 + \zeta_5^3 \\
&= -1 - \alpha \\
&= \overline{\alpha}.
\end{aligned}$$

Since $Gal(F/\mathbb{Q})$ is of order two, the only other element is $id = \sigma_1$. Now, we have

$$\begin{aligned} N_{F/\mathbb{Q}}(\sigma_1(\beta)\sigma_4(\beta)) &= \sigma_1(\sigma_1(\beta)\sigma_4(\beta))\sigma_2(\sigma_1(\beta)\sigma_4(\beta)) \\ &= (\beta\sigma_4(\beta))\,(\sigma_2(\beta)\sigma_3(\beta)) \\ &= N_{\mathbb{Q}(\zeta_5)/\mathbb{Q}}(\beta) \end{aligned}$$

An easy computation can show that

$$\begin{aligned} (\beta\sigma_4(\beta)) + (\sigma_2(\beta)\sigma_3(\beta)) &= \frac{1}{2}\sum_{0\le i<j\le 4}(a_i - a_j)^2 \\ &= \frac{1}{2}\left(5\sum_{i=0}^{4} a_i^2 - \left(\sum_{i=0}^{4} a_i\right)^2\right) \\ &\le \frac{5}{2}\sum_{i=0}^{4} a_i^2 \\ &\le \frac{5}{2}\left[3\left(\frac{1}{4}\right)^2 + 2\left(\frac{1}{2}\right)^2\right] \\ &= \frac{5}{2}\times\frac{11}{16}. \end{aligned}$$

Observe that both $(\beta\sigma_4(\beta))$ and $(\sigma_2(\beta)\sigma_3(\beta))$ are real valued since they lie in F which is equal to $\mathbb{Q}(\sqrt{5})$ by Lemma 7.3.4. Now, by applying arithmetic-geometric mean inequality we get

$$\begin{aligned} N(\beta) &= \beta\sigma_2(\beta)\sigma_3(\beta)\sigma_4(\beta) \\ &\le \left(\frac{\beta\sigma_4(\beta) + \sigma_2(\beta)\sigma_3(\beta)}{2}\right)^2 \\ &= \left(\frac{55}{64}\right)^2 \\ &\le 1. \end{aligned}$$

□

Combining this result with the Proposition 7.3.3 gives us the desired result. The Proposition above guarantees the existence of the Euclidean function and our division algorithm in 7.3.1 gives us an constructive method to find the quotient and remainder.

Corollary 7.3.7. $\mathbb{Z}[\zeta_5]$ *is norm-Euclidean.*

7.3.3 Euclidean Algorithm for $\mathbb{Z}[\zeta_5]$

Suppose that two elements m and n in $\mathbb{Z}[\zeta_5]$ are given. First we compute $\frac{m}{n}$ as we saw in Section 7.3.1. The result is an element in $\mathbb{Q}[\zeta_5]$ expressed as linear combination of the basis $B' = \{1, \alpha, \zeta_5, \alpha\zeta_5\}$. To use Proposition 7.3.6, we need to transform our result to the basis $B = \{1, \zeta_5, \zeta_5^2, \zeta_5^3\}$. From equation (51) we have

$$\zeta_5^2 = \alpha\zeta_5 - 1,$$

and also we have

$$\alpha = -1 - \zeta_5^2 - \alpha_5^3.$$

So the basis transformation matrix from B' to B is

$$\begin{pmatrix} 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & 0 \end{pmatrix}$$

Suppose that after dividing, we get $\frac{m}{n} = a_0 + a_1\alpha + a_2\zeta_5 + a_3\alpha\zeta_5$, with a_i's in $\mathbb{Q}$. Transforming $\frac{m}{n}$ into the second basis using the above matrix we get:

$$\begin{pmatrix} 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} a_0 - a_1 + a_3 \\ a_2 \\ -a_1 + a_3 \\ -a_1 \end{pmatrix} =: \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}.$$

Thus, we have

$$\frac{m}{n} = y_0 + y_1\zeta_5 + y_2\zeta_5^2 + y_3\zeta_5^3. \tag{53}$$

Then, following from Proposition 7.3.6, by rounding off the coefficients, we write

$$\frac{m}{n} = C + \beta, \quad \text{where} \quad C \in \mathbb{Z}[\zeta_5] \quad \text{and} \quad \beta \in \mathbb{Q}[\zeta_5],$$

where

$$\beta = b_0 + b_1\zeta_5 + b_2\zeta_5^2 + b_3\zeta_5^3 \tag{54}$$

and $|b_i| \leq 1/2$ for $0 \leq i \leq 3$. Let us now adapt the proof of Proposition 7.3.6 to make our algorithm. Observe that equation $1 + \zeta_5 + \zeta_5^2 + \zeta_5^3 + \zeta_5^4 = 0$ implies that for any $t \in \mathbb{Q}$ we have

$$\beta = (a_0 + t) + (a_1 + t)\zeta_5 + (a_2 + t)\zeta_5^2 + (a_3 + t)\zeta_5^3 + t\zeta_5^4.$$

If at least two of the coefficients in (54) are non-negative, we set $t = -1/4$ and write

$$\beta = (a_0 - 1/4) + (a_1 - 1/4)\zeta_5 + (a_2 - 1/4)\zeta_5^2 + (a_3 - 1/4)\zeta_5^3 - 1/4\zeta_5^4.$$

So at least three of these coefficients are at most $1/4$ in absolute value. If for all other coefficients we have $|a_i - 1/4| < 1/2$, then $N(\beta) \leq 1$ and we are done. Otherwise, if only the ith coefficient has $a_i - 1/4 < -1/2$, then replace C with $C - \zeta_5^i$ and replace β with $\beta + \zeta_5^i$; then again we have $\frac{m}{n} = (C - \zeta_5^i) + (\beta + \zeta_5^i)$ and $N(\beta) \leq 1$.

Finally, if there are two coefficients $a_i - 1/4$ and $a_j - 1/4$ which are less than $-1/2$, then replace C by $C - \zeta_5^i - \zeta_5^j$ and replace β by $\beta + \zeta_5^i + \zeta_5^j$. Then, $\frac{m}{n} = (C - \zeta_5^i - \zeta_5^j) + (\beta + \zeta_5^i + \zeta_5^j)$ and $N(\beta) \leq 1$. When at least one coefficient is positive, we repeat the process with $t = 1/4$, replacing C with $C + \zeta_5^i$ and β with $\beta - \zeta_5^i$.

According to Proposition 7.3.1, we have

$$m = Cn + \beta n \quad N(\beta n) \leq N(n).$$

Therefore, our quotient is C and our remainder is βn.

Thus, we gave an algorithm to calculate the quotient and remainder in the Euclidean domain $\mathbb{Z}[\zeta_5]$. The set S of possible values of the remainder upon division by q using this algorithm is a complete set of residues of $A/(q)$ in A. Thus, this algorithm can be used to define a map from $A/(q)$ to A. Namely, given any preimage a' of $\overline{a}$ in A, find its remainder mod q; the result is an element $s \in S$ such that $\overline{s} = \overline{a}$. From our discussion in Chapter 6 it now follows that ETRU can be implemented over $\mathbb{Z}[\zeta_5]$, as required.

Chapter 8

Comparison and Conclusions

This chapter is composed of three sections: Section 8.1 is a comparison of the three cryptosystems explained in Chapters 3, 4 and 5. Section 8.2 outlines the conclusions of the results obtained in Chapters 6 and 7. Finally, recommendations for future work are presented in Section 8.3.

8.1 Comparison of Lattice-based Cryptosystems

The three cryptosystems Ajtai-Dwork, GGH and NTRU, were all discovered independently at almost the same time, in the late 1990's. All three of them demonstrate methods to hide a message in a lattice with their security based on cases of famous hard lattice problems, like SVP, CVP and SBP. In this section, we point out the similarities and differences of these cryptosystems, with the intent of helping to reveal the common pattern of these lattice-based cryptosystems. When referring to a cryptosystem, we use the notations introduced in the corresponding chapter.

8.1.1 Ciphertext and Plaintext

Plaintext

The way in which the plaintext is hidden in the cipher vector differs from one cryptosystem to another:

- In GGH, the message corresponds to a particular lattice point which is perturbed by a random error vector of a given size.
- In NTRU, we saw in Section 5.5.3 that the message is half of an error vector whose other half is chosen at random. This error vector perturbs a random lattice vector.
- In Ajtai-Dwork, all the lattice points correspond to the message zero and all the far-from-lattice points correspond to one.

Thus, assuming that the underlying lattice has dimension n, the length of the message block is n, $n/2$ and one for GGH, NTRU and Ajtai-Dwork respectively.

Ciphertext

In all three cryptosystems, the ciphertext c is a vector in the span of the underlying lattice:

- In GGH and NTRU, c is always a close-to-lattice vector.
- In Ajtai-Dwork, depending on the ciphertext bit, c might be close or far from the lattice.

8.1.2 Encryption-Decryption

Probabilistic Encryption

In all three cryptosystem, there exists a random element in the encryption process that makes the encryption probabilistic. That is, for the same message, the encryption algorithm would most likely output different ciphertexts each time it is run.

- In GGH, this is because of the randomness of the perturbation vector e.
- In NTRU, the randomness arises from the choice of the polynomial ϕ.
- In Ajtai-Dwork, it follows from the fact that any lattice point can be picked to encrypt zero and the perturbation string is also chosen at random.

Decryption

The output of the decryption algorithm also varies from one cryptosystem to another.

- In GGH, one can compute the original lattice point from the output of the decryption algorithm. So, the error vector can also be computed by subtracting the ciphertext from that lattice point.
- In NTRU, only half of the error vector, the message m, is recovered by the decryption algorithm. The other half, as well as the lattice point, remains unknown. It should be emphasized that the decryption process in NTRU may fail with a small probability.
- In Ajtai-Dwork, the decryption algorithm only makes a decision about the closeness of the ciphertext to the lattice and outputs yes or no. That means neither the lattice point nor the error vector is recovered.

As a result, GGH fits nicely in the general definition of cryptosystems defined on one-way trapdoor functions, since the input is entirely recovered in the decryption process. NTRU does not completely fit, since ϕ is not recovered. In spite of that, the description of NTRU via one-way trapdoor functions which we gave in Chapter 5 helps make its similarities with GGH more clear. Finally, it is not straightforward to fit Ajtai-Dwork in this category.

Alternative Keys

An interesting similarity between GGH and NTRU is the fact that in both of them, the private key is not the only key which allows decryption. In both GGH and NTRU, the feature which distinguishes these alternative keys from an arbitrary element in the key space is their "smallness" in some particular sense.

- In GGH the transpose-inverse of the alternative key $\mathbf{R}'$ should have small rows.
- In NTRU, the alternative key f' and its corresponding g' should have small enough coefficients.

Unlike GGH and NTRU, in Ajtai-Dwork, breaking the system implies recovering the private key itself.

Efficiency and Key size

- In GGH we need to publish the entire basis matrix of the lattice as public key. The public key size is $O(n^3 \log n)$ [36]. Although this is polynomial in the rank of the lattice, since the rank of the lattice should be chosen large enough in order to have a reasonable security [38] [40], the key size is too big to allow efficient implementation.

- Similar to GGH, the entire basis matrix should be published which leads to the key size $O(n^4)$. This prevents efficient implementation.

- On the contrary, in NTRU, a single polynomial h of $O(n \log n)$ bit size, determines the entire public basis. Moreover, the generation of h is fast and easy compared to numerous steps of public key generation in GGH and Ajtai-Dwork. Therefore, these features make NTRU much more efficient compared to the other two cryptosystems.

Security

- The GGH cryptosystem does not have a security proof as Ajtai-Dwork. Moreover, it is shown that because of the special form of its error vector, the instance of the hard underlying problem can be reduced to an easier instance. That makes the system vulnerable unless for high dimensional lattices [38].

- The Ajtai-Dwork cryptosystems are superior to the others in that they only enjoy security proofs but also their last cryptosystem is based on the worst case of the underlying problem, which implies a stronger security.

- In NTRU, the underlying lattice has a special format. Although breaking the system using the lattice attack requires solving approximate-SVP, it is not known if approximate-SVP in this lattice is hard as for a general lattice.

Based on the comparisons made between the GGH, Ajtai-Dwork and NTRU, we chose NTRU for our further investigation due to the following reasons.

1. NTRU was the only efficient lattice-based cryptosystem.

2. The existence of the decryption failure inspired us to conduct our further studies.

8.2 Conclusions About Extensions of NTRU

In this thesis, we described possible generalizations of the NTRU cryptosystem. Our work is inspired by CTRU and NTRU over Gaussian integers.

What makes it worthwhile to investigate NTRU over base rings other that $\mathbb{Z}$ is its interesting decryption algorithm. In NTRU, as we saw, the decryption process does not use the properties of the lattice used to attack the system, which is the case for other lattice based cryptosystems that we discussed. On the contrary, the decryption process depends on the specific structure of the base ring.

One drawback of NTRU is the existence of decryption failure. This motivated the development of CTRU which has no decryption failure. This shows that the decryption process of NTRU depends on some unknown properties of the base ring. Hence, an investigation into the performance and adaptability of NTRU over other rings is warranted.

To be more precise, as we saw in Chapter 6, the suitability of a ring A for use as a base ring for NTRU hinges on the existence of a good set S of complete residues modulo q. In original NTRU, complete sets of residues modulo q are easy to find. Namely, any set of q consecutive integers is a set of complete residues (Figure 5). Moreover, given a sufficiently small set of integers Σ, one can hope to easily identify a complete set of residues containing Σ. In particular, consider the set Σ of coefficients of $a = p\phi * g + m * f$. Having a successful decryption depends on being able to identify a complete set of residues containing Σ.

This property inspires us to consider $\mathbb{Z}$ as a one dimensional lattice and attempt to generalize NTRU over rings which are lattices of higher dimension. In $\mathbb{Z}[\sqrt{-2}]$

($\mathbb{Z}[i]$), these complete sets of residues are rectangles in the lattice with sides equal to β and $\sqrt{-2}\beta$ ($i\beta$, respectively).

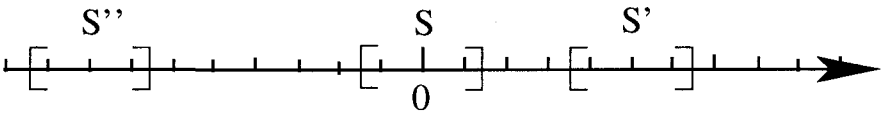

Figure 5: Complete sets of residues in $\mathbb{Z}$

The difficulty arises when we want to find an algorithm to reduce the coefficients modulo q to the desired set S. When our ring admits a constructive Euclidean function, that function, with some restrictions which assure the output lies in S, might be used to reduce the coefficients. For example, these restrictions on $\mathbb{Z}$ could include choosing the positive remainder.

Interestingly, we noticed that in the cases of $\mathbb{Z}[\sqrt{-2}]$ and $\mathbb{Z}[\zeta_3]$, our Euclidean algorithm is following the same flowchart as the Babai round off algorithm to find the closest vector in the lattice to a given vector. The similarity between round off algorithm technique and finding the smallest representative of a class in these two dimensional lattices suggests the possibility of using CVP techniques instead of Euclidean algorithm. This indicates the possibility of working over non-Euclidean lattices as well as the Euclidean lattices on which the Euclidean algorithm has not either explicitly determined or is too complicated to implement. Inspired by this idea, we started working on $\mathbb{Z}[\zeta_5]$. We obtained an implementable Euclidean algorithm which can be used to reduce the elements of $\mathbb{Z}[\zeta_5]$ modulo q into a set of complete residues.

One main question that arises here is the effect of the higher dimensional base rings on the cryptographic properties (i.e,efficiency and security) of the NTRU-like cryptosystems. Obviously, as the dimension of the base lattice ring increases, the number of computations in the process of key generation, encryption and decryption increases and this reduces the speed and efficiency. However, this also increases the dimension of the lattice where the key pair lies. For instance, the public basis for $\mathbb{Z}[\sqrt{-2}]$ has dimension $2N$ since each entry is a 2-tuple itself. That makes the system

more secure against lattice reduction algorithms.

8.3 Future Work

Based on the results of our study, the following research avenues are suggested for future work and further investigation:

- Coding and implementation of our suggested algorithms for $\mathbb{Z}[\sqrt{-2}]$, $\mathbb{Z}[\zeta_3]$ and $\mathbb{Z}[\zeta_5]$.
- Comparison the experimental results of decryption failure with the probability of failure based on our theoretical results.
- Investigate the applicability to ETRU of various attacks, eg,. [13] proposed against NTRU.
- Applying the CVP algorithms, other than round off algorithm, instead of the Euclidean algorithm.
- Investigating the possibility of developing our theoretical result of using a non-PID Dedekind domain as the NTRU base ring.

Appendix A

Ring Theory

In this appendix, we overview the ring theory material that used in this document. Most of the theorems come without proof. The proofs can be found in any abstract Algebra book such as [12] or Algebraic number theory book such as [6]. We assume that the reader is familiar with the basic notions of rings and fields.

An *integral domain* is a commutative ring with multiplicative identity 1 which has no zero devisors. Every integral domain A lies in a field that contains all quotients of A known as *quotient field.* An integral domain A is said to be integrally closed if the only elements of its quotient field that are root of a monic polynomial in $A[X]$, i.e. *integral* elements over A, are those of A itself.

In particular, a complex number which is integral over $\mathbb{Z}$ is called an *Algebraic integer.* The set of all algebraic integers lying in an *Algebraic number field* K, i.e. a finite extension of $\mathbb{Q}$, is called *ring of integers* and denoted by O_k. Ring of integers of an algebraic number field K is an integrally closed integral domain whose quotient filed is K.

Now, let K be an algebraic number field. Denote the set of all monomorphism from K to $\mathbb{C}$ who fixes $\mathbb{Q}$ by $Mon(K, \mathbb{C})$. There are exactly n such monomorphisms where n is the degree of K over $\mathbb{Q}$ as a vector space. For each element $x \in K$, there

are two very important quantities associated with x namely norm and Trace.

$$N(x) = \prod_{\sigma \in Mon(K,\mathbb{C})} \sigma(x)$$
$$Trac(x) = \sum_{\sigma \in Mon(K,\mathbb{C})} \sigma(x)$$

Let us mentioning that for an *Galois* extension K of $\mathbb{Q}$ we have

$$Gal(K/\mathbb{Q}) = Mon(K, \mathbb{C})$$

Let A be an integral domain . An element of A is called a *unit* if it divides 1. An element $a \in A$ is said to be *irreducible* if $a = bc$ implies that either b or c is a unite. A non-zero non-unit element p of A is called a *prime* if $p|ab$ for $a, b \in A$ implies either $p|a$ or $p|b$. A proper ideal P of an integral domain A is called a *prime ideal* if $a, b \in A$ and $ab \in P$ implies either $a \in P$ *or* $b \in P$.

Theorem A.0.1. *Let A be an integral domain, Let $a \in A$ be a non-unit, non-zero element. Then (a) is a prime ideal if and only if a is a prime element.*

An ideal I of an integral domain A is called a *principal ideal* if there exist an element $a \in I$ such that $I = (a)$. The element a is called the generator of the ideal I. An integral domain A is called a *principal ideal domain* if every ideal in A is principal. If every non-zero non-unit element of A can be expressed uniquely as finite number of irreducible elements of A, it is called a *unique factorization domain.*

Definition A.0.2. Let A be an integral domain. A mapping $\phi : A \to \mathbb{Z}$ is called a *Euclidean function* on A if it has the following two properties

1. $\phi(ab) \geq \phi(a), \quad \forall a, b \in A, b \neq 0$
2. For all $a, b \in A$ with $b \neq 0$ there exist $q, r \in A$ such that

$$a = qb + r \quad \phi(r) < \phi(b)$$

Theorem A.0.3. *Let A be an integral domain that possesses a Euclidean function ϕ. Let $a, b \in A$. Then*

1. *An element a is a unit if and only if $\phi(a) = \phi(1)$.*

2. *$\phi(a) > \phi(0)$, if $a \neq 0$.*

3. *If $a|b$ and $b|a$ then $\phi(a) = \phi(b)$.*

Definition A.0.4. An integral domain A possessing a Euclidean function ϕ is called a *Euclidean domain* with respect to ϕ. In particular, if A is *ring of integers* of a Algebraic Number field K, and the function *norm* over K is Euclidean, we say A is *norm-Euclidean.*

Theorem A.0.5. *Every Euclidean domain is a principal ideal domain.*

Proposition A.0.6. *Any principal ideal domain is a unique factorization domain.*

Definition A.0.7. A proper ideal M of an integral domain A is called a maximal ideal if whenever I in an ideal of A such that $M \subseteq I \subseteq A$, then $I = M$ or $I = A$.

Proposition A.0.8. *Every non-zero prime ideal in a principal ideal domain is a maximal ideal.*

Proposition A.0.9. *In a commutative ring, the ideal M is maximal if and only if R/M is a field.*

Theorem A.0.10. *Let F be a field. The ring of polynomials in one variable over F is a Euclidean domain.*

Definition A.0.11. An integral domain A in which every ascending chain of ideals terminates is called a *Noetherian* domain.

Definition A.0.12. An integral domain D that satisfies the following three properties is called a *Dedekind* domain.

1. A is a Notherian domain.

2. A as integrally closed.

3. Each prime idea of A is a maximal ideal.

Theorem A.0.13. *Let A be a principal ideal domain, then A is a Dedekind domain.*

Theorem A.0.14. *Let K be an algebraic number field. Let O_k be the ring of integers of K. Then O_K is a Dedekind domain.*

Theorem A.0.15. *Let A be a Dedekind domain. Every proper ideal of A is a product of prime ideals and this factorization is unique.*

Theorem A.0.16. ***Chinese remainder Theorem over a general ring.*** *Let A be a commutative ring with identity and $I_1, \ldots, I_n$ be pairwise comaximal ideals of A. Then*

$$A/(I_1 \ldots I_n) \cong A/I_1 \times A/I_2 \times \cdots \times R/I_n$$

Theorem A.0.17. ***Chinese Remainder Theorem.*** *Let A be a Dedekind domain. Let $P_1, \ldots, P_n$ be distinct prime ideals in D. Let $a_1, \ldots, a_n$ be positive integers. Let $\alpha_1, \ldots, \alpha_n$ be elements of D. Then there exists $\alpha \in D$ such that*

$$\alpha \equiv \alpha_i \mod P_i^{a_i}, \quad i = 1, \ldots, n$$

Proposition A.0.18. *Let A be a commutative ring. If every prime ideal in A is principal, then A is a PID.*

Bibliography

[1] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, New York, NY, USA, 1996. ACM Press.

[2] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293, New York, NY, USA, 1997. ACM Press.

[3] Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 601–610, New York, NY, USA, 2001. ACM Press.

[4] Mikls Ajtai. The shortest vector problem in l_2 is NP-hard for randomized reductions. pages 10 – 19, New York, 1998. Proceedings of the thirtieth annual ACM symposium on Theory of computing, ACM Press.

[5] R. Akhtar. Cyclotomoc Euclidean number fields. Senior thesis, Harvard University, 1995.

[6] Saban Alaca and Kenneth S. Williams. *Introductory algebraic number theory.* Cambridge University Press, NewYork, USA, 2004.

[7] Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.*, 54(2):317–331, 1997.

[8] László Babai. On Lovsz' lattice reduction and the nearest lattice point problem (shortened version). In *STACS '85: Proceedings of the 2nd Symposium of Theoretical Aspects of Computer Science*, pages 13–20, London, UK, 1985. Springer-Verlag.

[9] Johannes Buchmann and Christoph Ludwig. Practical lattice basis sampling reduction. Cryptology ePrint Archive, Report 2005/072, 2005.

[10] Consortium for Efficient Embedded Security. *Efficient Embedded Security standard (EESS), Implementation Aspects of NTRUEncrypt and NTRUSign*, June 2003.

[11] Don Coppersmith and Adi Shamir. Lattice attacks on NTRU. In *Lecture Notes in Computer Science*, volume 1233, pages 52–61, Berlin / Heidelberg, 1997. Springer.

[12] David S. Dummit and Richard M. Foote. *Abstract Algebra.* John Wiley and Sons, Inc., USA, 2004.

[13] Éliane Jaulmes and Antoine Joux. A chosen-ciphertext attack against NTRU. In *CRYPTO '00: Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, pages 20–35, London, UK, 2000. Springer-Verlag.

[14] P. Gaborit, J. Ohler, and P. Sole. CTRU, a polynomial analogue of NTRU. Technical report, INRIA, 2002.

[15] O. Goldreich, D. Micciancio, S. Safra, and J.-P. Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Information Processing Letters*, 71(2):55–61, 1999.

[16] Oded Goldreich and Shafi Goldwasser. On the limits of non-approximability of lattice problems. volume 60, pages 540–563, Orlando, FL, USA, 2000. Academic Press, Inc.

[17] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Eliminating decryption errors in the Ajtai-Dwork cryptosystem. In *CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, pages 105–111, London, UK, 1997. Springer-Verlag.

[18] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In *CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, pages 112–131, London, UK, 1997. Springer-Verlag.

[19] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU, a ring-based public-key cryptosystem. In *Algorithmic Number Theory*, volume 1423, pages 267–288. Springer, 1998.

[20] Jeff Hoffstein and Joseph H. Silverman. Protecting NTRU from chosen ciphertext and reaction attacks. Technical Report 016, NTRU cryptosystem.

[21] Nick Howgrave-Graham, Jeff Hoffstein, Jill Pipher, and William Whyte. On estimating the lattice security of NTRU. availabel from http://www.ntru.com/cryptolab/articles.htm2005$_2$.

[22] Nick Howgrave-Graham, Phong Q. Nguyen, David Pointcheval, John Proos, Joseph H. Silverman, Ari Singer, and William Whyte. The impact of decryption failures on the security of NTRU encryption. *Advances in Cryptology-CRYPTO*, 2003.

[23] Nick Howgrave-Graham, Joseph H. Silverman, and William Whyte. A meet-in-the-middle attack on an NTRU private key. Technical Report 004, NTRU cryptosystem.

[24] Priit Karu and Jonne Loikkanen. Practical comparison of fast public-key cryptosystems. In *Seminar on Network Security*, Helsinki University of Technology, 2001.

[25] R. Kouzmenko. Generalizations of the NTRU cryptosystem. Diploma project, Ecole Polytechnique Federale de Lausanne, 2005-2006.

[26] J. C. Lagarias and A. M. Odlyzko. Solving low-density subset sum problems. *J. ACM*, 32(1):229–246, 1985.

[27] Serge Lang. *Algebra.* Addison-Wesley, third edition, 1993.

[28] J. Van Leeuwen. *Hand book of theoretical computer science*, volume A Algorithms and Complexity. The MIT Press, 1990.

[29] Franz Lemmermeyer. The Euclidean algorithm in algebraic number fields. *Expos ition. Math.*, 13(5):385–416, 1995.

[30] A. K. Lenstra, H. W. Lenstra Jr, and L. Lovsz. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[31] J. Myron Masley and Hugh L. Montgomery. Cyclotomic fields with unique factorization. *J. Reine Angew. Math.*, 286/287:248–256, 1976.

[32] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography.* CRC press, fifth edition, 1996.

[33] Tommi Meskanen and Ari Renvall. A wrap error attack against NTRUEncrypt. *Discrete Appl. Math.*, 154(2):382–391, 2006.

[34] Daniele Micciancio. Improving lattice based cryptosystems using the hermite normal form. In *CaLC '01: Revised Papers from the International Conference on Cryptography and Lattices*, pages 126–145, London, UK, 2001. Springer-Verlag.

[35] Daniele Micciancio. The shortest vector problem is NP-hard to approximate to within some constant. *SIAM Journal on Computing*, 30(6):2008–2035, March 2001.

[36] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, 2002.

[37] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. *J. Symb. Comput.*, 35(4):377–401, 2003.

[38] Phong Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem, Crypto '97. *Lecture Notes in Computer Science*, 1666:288–304, 1999.

[39] Phong Nguyen and David Pointcheval. Analysis and improovement of NTRU encryption paddings. In *Proceedings of CRYPTO '02 In Advances in Cryptology*, pages 210–225, Santa Barbara, CA, USA, 2002. Springer-Verlag.

[40] Phong Q. Nguyen and Jacques Stern. Cryptanalysis of the ajtai-dwork cryptosystem. In *CRYPTO '98: Proceedings of the 18th Annual International Cryptology Conference on Advances in Cryptology*, pages 223–242, London, UK, 1998. Springer-Verlag.

[41] Phong Q. Nguyen and Jacques Stern. The two faces of lattices in cryptology. In *CaLC '01: Revised Papers from the International Conference on Cryptography and Lattices*, pages 146–180, London, UK, 2001. Springer-Verlag.

[42] NTRU Cryptosystems, Inc. www.ntru.com. CryptoLab Oct 2007.

[43] J. Proos. Imperfect decryption and an attack on the NTRU encryption scheme. *Cryptology ePrint Archive*, 2003.

[44] Oded Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004.

[45] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 84–93, New York, NY, USA, 2005. ACM Press.

[46] Tong-Shieng Rhai. A characterization of polynomial domains over a field. *The American Mathematical Monthly*, 69(10):984–986, 1962.

[47] C. P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53(2-3):201–224, 1987.

[48] Jeff Hoffstein Joseph H. Silverman and William Whyte. Estimated breaking times for NTRU lattice. Technical Report 012, NTRU cryptosystem.

[49] Joseph H. Silverman. Dimension reduced lattices, zero-forced lattices, and the NTRU public key cryptosystem. Technical Report 013, NTRU cryptosystem.

[50] Joseph H. Silverman. Wraps, gaps and lattice constants. Technical Report 001, NTRU cryptosystem, 2001.

[51] Weichi Yu and Dake He. Study on NTRU decryption failures. In *ICITA '05: Proceedings of the Third International Conference on Information Technology and Applications (ICITA'05) Volume 2*, pages 454–459, Washington, DC, USA, 2005. IEEE Computer Society.