N° d'ordre NNT : xxx

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'École Normale Supérieure de Lyon

École Doctorale ED512
École Doctorale en Informatique et Mathématiques de Lyon

Spécialité de doctorat : Informatique

Soutenue publiquement le jj/mm/aaaa, par :
Weiqiang Wen

Contributions to the hardness foundations of lattice-based cryptography

Contributions aux fondements de complexité de la cryptographie sur réseaux

---

Devant le jury composé de :

Fouque Pierre-Alain, Professeur, Université Rennes 1	Rapporteur
May Alexander, Professeur, Ruhr-Universität Bochum	Rapporteur
Micciancio Daniele, Professeur, University of California, San Diego	Rapporteur
Fontaine Caroline, Chargée de Recherche, IMT Atlantique	Examinateur
Gaborit Philippe, Professeur, Université de Limoges	Examinateur
Stehlé Damien, Professeur, ENS de Lyon	Directeur de thèse

Résumé

La cryptographie sur les réseaux est l'une des approches les plus compétitives pour protéger la confidentialité, dans les applications actuelles et l'ère post-quantique. Le problème central qui sert de fondement de complexité de la cryptographie sur réseaux est Learning with Errors (LWE). Il consiste à résoudre un système d'équations bruité, linéaire et surdéterminé. Ce problème est au moins aussi difficile que les problèmes standards portant sur les réseaux, tels que le décodage à distance bornée (BDD pour Bounded Distance Decoding) et le problème du vecteur le plus court unique (uSVP pour unique Shortest Vector Problem). Tous ces problèmes sont conjecturés difficiles à résoudre, même avec un ordinateur quantique de grande échelle. En particulier, le meilleur algorithme connu pour résoudre ces problèmes, BKZ, est très coûteux.

Dans cette thèse, nous étudions les relations de difficulté entre BDD et uSVP, la difficulté quantique de LWE et les performances pratiques de l'algorithme BKZ. Tout d'abord, nous donnons une relation de difficulté plus étroite entre BDD et uSVP. Plus précisément, nous améliorons la réduction de BDD à uSVP d'un facteur $\sqrt{2}$, comparément à celle de Lyubashevsky et Micciancio. Ensuite, Nous apportons un nouvel élément à la conjecture que LWE est quantiquement difficile. Concrètement, nous considérons une version relâchée de la version quantique du problème du coset dièdral et montrons une équivalence computationnelle entre LWE et ce problème. Enfin, nous proposons un nouveau simulateur pour BKZ. Dans ce dernier travail, nous proposons le premier simulateur probabiliste pour BKZ, qui permet de prévoir le comportement pratique de BKZ très précisément.

Abstract

Lattice-based cryptography is one of the most competitive candidates for protecting privacy, both in current applications and post quantum period. The central problem that serves as the hardness foundation of lattice-based cryptography is called the Learning with Errors (LWE). It asks to solve a noisy equation system, which is linear and over-determined modulo an integer q. Usually, LWE is called an average-case problem as all the coefficients in the equation system are randomly chosen modulo q. The LWE problem is conjectured to be hard even with a large scale quantum computer. It is at least as hard as standard problems defined in the lattices, such as Bounded Distance Decoding (BDD) and unique Shortest Vector Problem (uSVP). Finally, the best known algorithm for solving these problems is BKZ, which is very expensive.

In this thesis, we study the quantum hardness of LWE, the hardness relations between the underlying problems BDD and uSVP, and the practical performance of the BKZ algorithm. First, we give a new evidence of quantum hardness of LWE. Concretely, we consider a relaxed version of the quantum version of dihedral coset problem and show an computational equivalence between LWE and this problem. Second, we tighten the hardness relation between BDD and uSVP. More precisely, We improve the reduction from BDD to uSVP by a factor $\sqrt{2}$, compared to the one by Lyubashevsky and Micciancio. Third, we propose a new simulator for BKZ. In the last work, we propose the first probabilistic simulator for BKZ, which can predict the practical behavior of BKZ very precisely.

Contents

Notations

$\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}$	.....	sets of natural, integer, rational, real numbers
$\mathbb{Z}_q$	.....	the ring of integers modulo q
$\mathbb{Z}^n$	.....	the set of integer vector of dimension n
$\mathbb{Z}_q^n$	.....	the set of integer vector modulo q of dimension n
$\mathbb{Q}^n, \mathbb{R}^n$	.....	vector-spaces of dimension n
$[n]$	.....	the set $\{1, \cdots, n\}$
$\mathbf{x}$	.....	column vector
$\mathbf{x}^{\mathrm{T}}$	.....	row vector
$\|\mathbf{x}\|$	.....	Euclidean norm of vector $\mathbf{x}$
$\|\mathbf{x}\|_\infty$	.....	ℓ_∞-norm of $\mathbf{x}$: $\max_i \|x_i\|$
$\mathbf{A}$	.....	matrix (composed from vectors, column-wise)
$\mathbf{a}_i$	.....	i^{th} column of matrix $\mathbf{A}$
$\mathbf{0}$	.....	all-zeros vector
$\mathbf{I}_n$	.....	$n \times n$ identity matrix
$\mathrm{Ball}_n(\mathbf{c}, r)$	.....	n-dimensional Euclidean ball of radius r centred at $\mathbf{c}$
Ball_n	.....	n-dimensional Euclidean unit ball centred at $\mathbf{0}$
$\log x$	.....	$\ln x$, the natural logarithm with base e
$\mathcal{D}_{\mathbb{Z},r,c}$	.....	the Gaussian distribution over integer with center c and parameter r
$x \hookleftarrow D$	.....	x is sampled from a probability distribution D
$x \hookleftarrow S$	.....	x is uniformly sampled from set S (assuming S has finite measure)
ω_q	.....	$\exp(2\pi i/q)$
$\mathrm{Span}(\mathbf{W})$	.....	span of column vectors of $\mathbf{W}$
$\mathbf{W}^\perp$	.....	orthogonal complementation of $\mathrm{Span}(\mathbf{W})$
$\textsc{Expo}[\gamma]$	.....	exponential distribution with parameter γ
$\textsc{Beta}(x, y)$	.....	beta function with input x and y
$\#S$	.....	cardinality of set S
$\mathrm{GL}_n(\mathbb{Z})$	.....	unimodular matrices of order n
$\mathrm{Tr}(x)$	.....	trace norm of x
$\|\cdot\rangle$	.....	bra-ket notation for quantum register

We use the Landau notations $\mathcal{O}(\cdot), \Theta(\cdot), \Omega(\cdot), \omega(\cdot), o(\cdot)$.

Chapter 1

Introduction

1.1 Background

Cryptography is used to protect the integrity and confidentiality of messages, as well as to ensure the message source authentication. The hardness foundations of most currently deployed asymmetric cryptographic primitives are number-theoretic problems that are presumed computationally hard. The first two applications of number-theoretic problems in cryptography are the Diffie-Hellman key exchange [DH76] and the RSA cryptosystem by Rivest, Shamir, and Adleman [RSA78]. The number-theoretic problems involved in these cryptosystems are variants of the integer factorization and discrete logarithm problems.

The conjectured hardness of these number-theoretic problems does not hold if we consider the power of quantum computation. Indeed, in 1996, Shor [Sho99] gave an efficient quantum algorithm for solving some typical number-theoretic problems, such as those mentioned above. Taking integer factorization as an example, Shor's algorithm gives a polynomial time solver with respect to n, the bitsize of the integer.

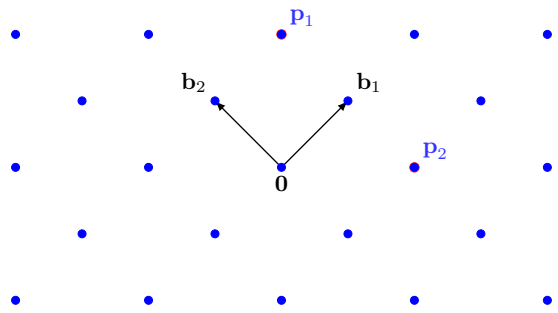

Fig. 1.1: An example of a two-dimensional lattice generated by a basis.

As opposed to these number-theoretic problems, there is no known efficient classical as well as quantum algorithm for solving a set of problems on lattices. A (full-rank) lattice Λ in dimension n is the set of all integer linear relations of a set of linearly independent vectors in a Euclidean space. This set is called a basis of the lattice. As illustrated in Figure 1.1, the lattice is composed of the regularly distributed points (colored in blue) generated by computing all integer combinations of the set $\{\mathbf{b}_1, \mathbf{b}_2\}$. For example, the vectors $\mathbf{p}_1 = \mathbf{b}_1 + \mathbf{b}_2$ and $\mathbf{p}_2 = \mathbf{b}_1 - \mathbf{b}_2$ are in the lattice. A lattice can also be defined as a discrete group of some $\mathbb{R}^m$. The extent of discreteness can be measured by the norm of any shortest non-zero vector (denoted by λ_1) in the lattice.

Next, we give a brief overview of the hardness assumptions underlying lattice-based cryptography. The main problem on lattices is the Shortest Vector Problem (SVP), which asks to find a shortest

non-zero vector in the lattice, given a basis of the lattice as input. Another closely related problem is the Closest Vector Problem (CVP), which asks to find a closest vector to a target vector $\mathbf{t} \in \mathbb{R}^n$. SVP can be seen as a variant of CVP, in which a closest vector to the origin $\mathbf{0}$ other than itself is required. Both of these problems are NP-hard [Emd81, Ajt98], with the caveat that the proof of NP-hardness of SVP relies on a probabilistic polynomial time reduction. In the security analysis of cryptography, we are also interested in the approximation version of SVP (denoted by SVP_γ for some factor $\gamma \geq 1$), in which a relatively (non-zero) short vector with norm $\leq \gamma \cdot \lambda_1$ is required. We know that SVP_γ is NP-hard to for any constant factor γ [Kho05, HR07]. There is also no efficient algorithm for solving SVP_γ, for a γ polynomial in the dimension n of the input lattice. The current best known algorithm for solving SVP_γ with γ polynomial in n, takes time exponential in n [Sch87, SE94].

There is a rich family of variants of SVP and CVP as well as a series of hardness results on them. Among others, the unique SVP (uSVP) problem and Bounded Distance Decoding (BDD) problem are two well-known promise variants of SVP and CVP. uSVP_γ with parameter $\gamma \geq 1$ can be seen as a relaxation of SVP with a restriction that any lattice vector that is independent from the two shortest vectors should be longer by a factor $\geq \gamma$. Thus the shortest vector is "unique" (up to sign) when $\gamma > 1$. In 2001, Kumar and Sivakumar [KS01] proved the NP-hardness of uSVP_γ with $\gamma \leq 1 + 2^{-n^c}$ for some constant c under a randomized reduction. This result was later improved by Aggarwal and Dubey [AD16], who increased the inapproximability factor slightly to $\gamma \leq 1 + 1/\mathrm{poly}(n)$, for which Stephens-Davidowitz [SD16b] further gave a simplified proof. In the BDD_α problem, the target vector in the instance is promised to be within a bounded distance $\alpha \cdot \lambda_1$ to the lattice. The hardness of BDD was initially studied in the context of linear codes by Vardy in [Var97], and later in the context of lattices by Liu *et al.* in [LLM06], where BDD_α with $\alpha = 1/\sqrt{2} + c$ for any constant c is proved to be NP-hard. The relation between these two promise variants is studied by Lyubashevsky and Micciancio [LM09]. They show a reduction from $\mathrm{BDD}_{1/(2\gamma)}$ to uSVP_γ, and also a converse reduction from uSVP_γ to $\mathrm{BDD}_{1/\gamma}$. As we can see, there is a factor 2 gap between these two reductions. There are more variants of SVP and CVP that serve as the security foundation of cryptography, such as SIVP_γ, GapSVP_γ *etc.* Some relations among them are also known [Ste15]. For small γ, these problems are only "related" to lattice-based cryptography. We do not know how to base cryptographic primitives on the conjectured hardness of these problems with very small γ, e.g., γ is constant.

Now, we are ready to give an overview of lattice-based cryptography. In 1996, Ajtai [Ajt96] presented a ground-breaking discovery that a worst-case to average-case reduction exists for lattice problems. Concretely, it is shown [Ajt96] that if certain worst-case lattice problems are computationally hard to solve, then an average-case one-way function (a fundamental cryptographic primitive) exists. Later, Micciancio and Regev [MR07] extended this work and gave a reduction from worst-case lattice problems to a new average-case Short Integer Solution (SIS) problem. The SIS problem, given a under-determined equation system ($\mathbf{A}, \mathbf{x}^{\mathrm{T}}\mathbf{A} \equiv 0 \bmod q$) with uniform $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ and a modulus q, consists in finding a short non-zero solution $\mathbf{x}$. As a follow-up of [Ajt96], the first public-key cryptosystem with security relying on worst-case lattice problems is also proposed [AD97], which is known as Ajtai-Dwork cryptosystem. Concretely, the security of Ajtai-Dwork cryptosystem is based on the hardness of $\mathrm{uSVP}_{\mathcal{O}(n^8)}$. After this, there were several improvements [GGH97, GGH13, Reg03]. In particular, Regev gave a new security proof for the Ajtai–Dwork cryptosystem, in which the underlying problem becomes $\mathrm{uSVP}_{\mathcal{O}(n^{1.5})}$, while maintaining essentially the same bit-size for key and ciphertext. As the uSVP_γ problem is supposed to be harder with a smaller gap on the norm of the shortest vector and the second shortest vector, thus Regev's work gives an improvement on the hardness of the Ajtai–Dwork cryptosystem. Later, in 2005, Regev [Reg05] proposed another average-case problem, learning with error (LWE), and also established reductions from worst-case lattice problems such as GapSVP_γ to it. Here, we only give a brief definition of LWE. The LWE problem, given a under-determined equation system with noise of form ($\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e} [\bmod q]$) with uniform $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, uniform $\mathbf{s} \in \mathbb{Z}_q^n$, short

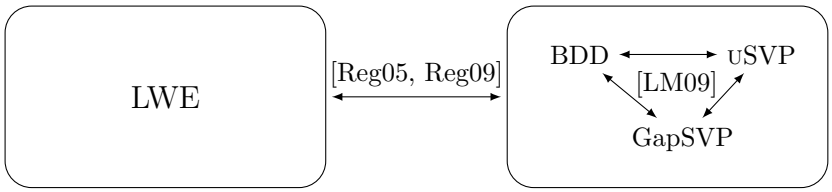

Fig. 1.2: Graph of reductions between known the (average-case) LWE problem and some (worst-case) lattice problems: BDD, uSVP and GapSVP.

$\mathbf{e} \in \mathbb{Z}^m$ and a modulus q, asks to find out the unique solution $\mathbf{s}$. Correspondingly, we can also define its decision version, in which one is asked to distinguish the equation system above from a uniform pair from $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$. Further, LWE may be interpreted as an instance of the Bounded Distance Decoding problem (BDD) (see, e.g., [MR09, Section 5.4]), where both the target vector $\mathbf{t}$ and the lattice Λ are randomly chosen. Note that BDD and GapSVP reduce to one another with limited parameter losses [LM09]. Overall, LWE is equivalent to BDD with same dimension but with limited losses on the noise. Note that Regev's reduction from GapSVP to LWE was partly dequantized in [Pei09, BLP$^+$13], but the corresponding classical reductions involve significant losses in terms of the dimension n or the modulus q. Usually, the cryptographic primitives proved secure under the SIS/LWE assumption are referred to as lattice-based cryptographic primitives as their security is also based on (worst-case) lattice assumptions.

From Regev's elegant public-key cryptosystem [Reg05], we can see the simplicity of applying LWE to design cryptographic schemes. In the scenario of public-key encryption, suppose that Alice (possessing Bob's public key) is going to send a private message to Bob (possessing its own secret key). In the design of Regev's scheme, the equation system $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$ serves as the public key and it is indistinguishable from uniform under the decision LWE assumption, and the unique secret $\mathbf{s}$ serves as the secret key. To send a message, Alice first constructs a random pair

$$(\mathbf{a} = \mathbf{r}^{\mathrm{T}}\mathbf{A}, b = \langle \mathbf{r}, \mathbf{b} \rangle),$$

where $\mathbf{r}$ is selected from a sufficiently large space with small ℓ_2 norm such that the space is exponentially larger than the space of the output pair. Then Alice adds the message bit to the most significant bit of the second component and send $(\mathbf{a}, z = b + \mu \cdot \lfloor q/2 \rceil)$ to Bob. Once Bob receives the ciphertext, with his own secret $\mathbf{s}$, he can recover an approximated value of $z - \mu \cdot \lfloor q/2 \rceil$ by computing

$$\langle \mathbf{a}, \mathbf{s} \rangle = \mathbf{r}^{\mathrm{T}}\mathbf{As} \approx \mathbf{r}^{\mathrm{T}}\mathbf{As} + \mathbf{r}^{\mathrm{T}}\mathbf{e} = \langle \mathbf{r}, \mathbf{b} \rangle = b = z - \mu \cdot \lfloor q/2 \rceil,$$

in which the approximate equality comes from the small ℓ_2 norms of both random vector $\mathbf{r}$ and noise vector $\mathbf{e}$. This also explains why the message bit needs to be stored in the most significant bit. Then Bob can extract the message μ from the most significant bit of $z - \langle \mathbf{a}, \mathbf{s} \rangle$. In fact, the algebraic simplicity of LWE enables the design of primitives with advanced functionalities, such as fully homomorphic encryption [BV11], attribute-based encryption for all circuits [GVW13] and (single key) functional encryption [GKP$^+$13].

Currently, there is no efficient classical algorithm for solving LWE, and LWE is conjectured hard even in the context of quantum computations, making it one of the most appealing candidate security foundations for post-quantum cryptography [BBD08]. However, little is known today on the quantum hardness of LWE, beyond the absence of knowledge of an efficient quantum algorithm. To further back the conjecture that LWE is quantumly hard, one promising idea is to prove that LWE is at least as hard as a hard problem that is well studied in quantum computation.

Last, we briefly introduce the state-of-the-art of cryptanalysis on lattice-based cryptography. Cur-

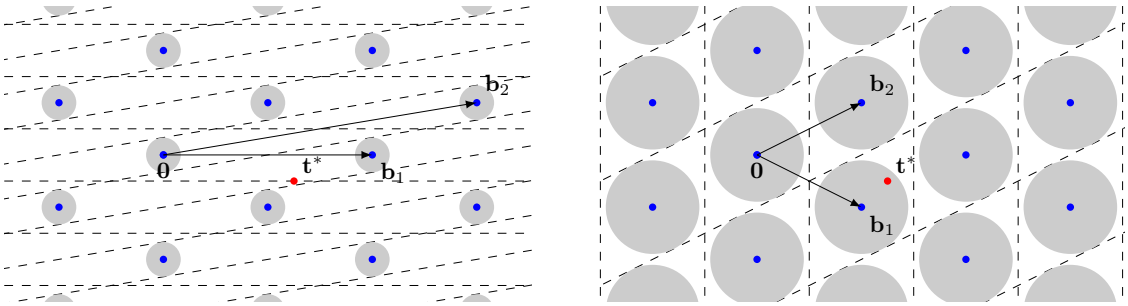

Fig. 1.3: Babai's round-off algorithm for CVP under different bases.

rently, the best known attacks against lattice-based schemes typically consist in finding short vectors/bases of lattices provided by publicly available data [APS15]. To build intuition about this, we first consider the public key (as an LWE instance) of Regev's cryptosystem and view it as a BDD instance $(\mathbf{B}, \mathbf{t})$, then we use an efficient algorithm to find a close vector to $\mathbf{t}$: $\lfloor \mathbf{B}^{-1}\mathbf{t} \rceil$, which is also known as Babai's rounding-off algorithm [Bab86]. Notice that we can write any target vector $\mathbf{t} \in \mathbb{Q}^n$ as $\mathbf{t} = \mathbf{c} + \mathbf{e}$, for any closest lattice vector $\mathbf{c} \in \Lambda(\mathbf{B})$ to $\mathbf{t}$ and some noise vector $\mathbf{e} \in \mathbb{Q}^n$. Given the target vector $\mathbf{t}$, Babai's rounding-off algorithm aims to round off the noise vector $\mathbf{e}$ and recover the closest vector $\mathbf{c}$ to $\mathbf{t}$. However, the ability of Babai's rounding-off algorithm depends on the quality of the input basis and the amount of noise. Concretely, the better quality the input basis has, the larger noise $\mathbf{e}$ this algorithm can manage to round off. Suppose first the basis we have is as on the left-hand side of Figure 1.3: we can only recover the closest vector for the target vectors $\mathbf{t}$ staying in any of the tiny shaded balls. Compared to this, assume we manipulate the basis and obtain another basis made of vectors with relatively smaller norms as on the right-hand side of Figure 1.3: then the upper bound of the noise vectors for which Babai's rounding-off succeeds in rounding off is much larger than before. In particular, from the target vector $\mathbf{t}^*$, we can recover the closest vector with the good basis but not the bad one. Lattice basis reduction algorithms are used to achieve such a task: transform a bad basis to a good basis. Concretely, lattice reduction aims at computing a basis made of relatively short vectors from an arbitrary input basis. Therefore, understanding the practical behavior and limits of lattice basis reduction algorithms is important for setting parameters in lattice-based cryptography.

In 1982, A. Lenstra, H. Lenstra and L. Lovász [LLL82] proposed the first lattice basis reduction algorithm with a provable worst-case bound of the quality of the reduced basis. Later, Schnorr [Sch87] gave a stronger block-wise lattice reduction for applications such as cryptanalysis. It is also made more practical by Schnorr and Euchner [SE94], whose variant is known as the Block Korkine-Zolotarev (BKZ) reduction algorithm.

The BKZ algorithm achieves a stronger reduction on the basis at the expense of an increased runtime. The theoretical behavior of BKZ in the worst case is well studied [HPS11, Neu17]. Unfortunately, these worst-case bounds are quantitatively very far from experimental data. Many efforts have been made to a better understand the behavior of BKZ in practice [Sch03, CN11]. However, as investigated in [CN11, AWHT16, YD17], the analysis or simulations on BKZ still differ significantly with what is observed in practice. Put plainly, BKZ finds shorter vectors than predicted by the analysis or simulations. This inaccuracy may lead to overestimated security evaluations in cryptographic design.

1.2 Our contributions

In this thesis, we focus on the hardness foundation of lattice-based cryptography. First, we will study two worst-case lattice problems, BDD and uSVP, and establish a tighter reduction between them. Second, we present a computational equivalence between LWE and a variant of (the presumably quantum hard) dihedral cost problem. Last, we give a more precise security analysis on the concrete classical hardness of lattice-based cryptography against the BKZ reduction algorithm.

1.2.1 An improved reduction from BDD *to* uSVP

The relationship between BDD and uSVP was initially studied by Lyubashevsky and Micciancio in [LM09], who showed that $\mathrm{BDD}_{1/(2\gamma)}$ reduces to uSVP_γ for any $\gamma \geq 1$ and uSVP_γ to $\mathrm{BDD}_{1/\gamma}$ for $\gamma \geq 1$ polynomially large in the dimension of the lattice.

Thus there is still a gap of a factor 2 between these two reductions. At this stage, we may conjecture that $\mathrm{BDD}_{1/\gamma}$ is computationally equivalent to uSVP_γ and try to describe a reduction from $\mathrm{BDD}_{1/\gamma}$ to uSVP_γ, or that $\mathrm{BDD}_{1/(2\gamma)}$ is computationally equivalent to uSVP_γ and try to describe a reduction from uSVP_γ to $\mathrm{BDD}_{1/(2\gamma)}$. In fact, there are more possibilities, for example, $\mathrm{BDD}_{1/(c\cdot\gamma)}$ could be equivalent to uSVP_γ for some constant $c \in (1, 2)$. It could also be that they are not equivalent for any such c.

Our main contribution in this work is an improvement on the BDD to uSVP reduction by a factor $\sqrt{2}$. All of the prior works rely on Kannan's embedding technique [Kan87, Section 6]. The main ingredient to the improvement is the use of Khot's lattice sparsification [Kho03] before resorting to Kannan's embedding, in order to boost the uSVP parameter. It is worth mentioning that the analysis of the improved reduction mainly benefits from a result on sphere packing [MG02]. This contribution corresponds to the following publication.

[SBW16] *Shi Bai, Damien Stehlé and Weiqiang Wen.* Improved Reduction from the Bounded Distance Decoding Problem to the Unique Shortest Vector Problem in Lattices. ICALP'16.

1.2.2 A computational equivalence between LWE *and extrapolated* DCP

The first connection between lattice problems and quantum computations was given by Regev in [Reg02]. In [Reg02, Reg04b], Regev described a reduction from the unique Shortest Vector Problem (uSVP) to the Dihedral Coset Problem (DCP). As uSVP is also essentially equivalent to BDD [LM09] (up to small parameter losses), this reduction also leads to a reduction from LWE to DCP. Our main contribution in this work is to define an extension of DCP, which we call EDCP, such that we can not only manage to provide a quantum polynomial time reduction from LWE to it, but also in the other direction. Our result implies that a quantum polynomail time solution for LWE might not require the full power of solving DCP, but rather only a solution for its relaxed version, EDCP, which could be easier to solve. This contribution corresponds to the following publication.

[BKSW18] *Zvika Brakerski, Elena Kirshanova, Damien Stehlé and Weiqiang Wen.* Learning With Errors and Extrapolated Dihedral Cosets. PKC'18.

1.2.3 A finer modelling of BKZ: understanding the head concavity

The BKZ algorithm is central in cryptanalysis, in particular for lattice-based cryptography. A precise understanding of its practical behavior in terms of run-time and output quality is necessary for parameter selection in cryptographic design. As discussed above, the provable worst-case bounds poorly reflect

the practical behavior of the BKZ algorithm. The Chen–Nguyen simulator fits better with practical experiments, but still overestimates the norm of the first few vectors in the output basis.

As a first contribution on this topic, we give more insight on this shorter-than-expected phenomenon by reporting many experiments. A second contribution is a presentation of a new BKZ simulator, which makes use of the distribution of shortest lattice vectors rather than the Gaussian heuristic used before. This new BKZ simulator can model the shorter-than-expected phenomenon and predict the practical behavior of the BKZ algorithm very precisely. We also introduce a BKZ variant that exploits the shorter-than-expected phenomenon. For the same cost assigned to the underlying SVP-solver, the new BKZ variant produces bases of better quality. This work is accepted to the Asiacrypt'18 as follows.

[SBW18] *Shi Bai, Damien Stehlé and Weiqiang Wen.* Measuring, simulating and exploiting the head concavity phenomenon in BKZ. To appear in Asiacrypt'18.

1.3 Organization of this thesis

In Chapter 2, we recall all necessary definitions and former results that are used throughout this thesis. These include the basic definitions and tools related to lattices, worst-case problems and average-case problems on lattices, followed by an elementary introduction to quantum computation as well as some (presumably hard) quantum problems that would be connected to lattice problems. Finally, we recall some widely used notions of lattice reduction, in particular, we present the BKZ reduction as well as the BKZ algorithm for achieving this notion of reduction.

In Chapter 3, we first review the Lyubashevsky–Micciancio reduction [LM09] from $\mathrm{BDD}_{1/(2\gamma)}$ to uSVP_γ. Then we describe our improvement: a reduction from $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$ to uSVP_γ. We explain how we obtain our improved result with the sparsification technique.

In Chapter 4, we first review the first known reduction by Regev from the lattice problem uSVP to the quantum hard problem DCP, and then we give our new result: a reduction from LWE to the extension of DCP, called EDCP, and also a converse reduction. We will also show the intrinsic relation between the extension of DCP and the reducedness of noise rate in LWE.

In Chapter 5, we first report experiments that we run on BKZ to measure shorter-than-expected phenomenon. Then we propose a new simulator for BKZ, which gives a very precise estimation on the practical behavior. After the shorter-than-expected phenomenon is well modelled and understood, we give a new variant of BKZ, which can further make use of this phenomenon and allows to compute bases with better quality.

In Chapter 6, we conclude this thesis and present some directions for future works.

Chapter 2

Preliminaries

In this chapter, we first give some basic definitions and results on lattices that will be used throughout the thesis. These include the basic notations and properties of lattices, discrete Gaussian distributions over lattices, and a lattice sparcification technique. Then we recall some worst-case lattice problems such as the BDD and uSVP problems, which will be used in Chapter 3 and Chapter 4. Correspondingly, we introduce the average-case analogue of BDD: the LWE problem. LWE is the security foundation of Regev's encryption scheme. To study the quantum hardness of LWE as well as Regev's encryption scheme, we give a simple introduction on quantum computations and the presumably hard dihedral coset (quantum) problem, which is necessary for Chapter 4. Finally, for presentation in Chapter 5, we recall some basic definitions related to lattice reduction and the BKZ algorithm, as well as prior analyses and modelling of it. In particular, it can be used to analyze the concrete security of Regev's encryption scheme.

Contents

2.1 Lattices

We refer the reader to [MG02] for an introduction to the computational aspects of Euclidean lattices and several lecture notes [Reg, Mic, Pei, Vai] for extensive background on lattices, especially in lattice-based cryptography.

2.1.1 Definitions and properties

We first introduce the definition of lattices.

Definition 2.1 (Lattice)**.** *An n-dimensional lattice $\Lambda \subseteq \mathbb{Q}^m$ $(m \geq n)$ is a discrete additive subgroup of $\mathbb{Q}^m$. The lattice Λ is the set of all integral linear combinations of n linearly independent basis vectors $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\} \subseteq \mathbb{Q}^m$. In other words, we have*

$$\Lambda(\mathbf{B}) = \left\{ \sum_{i \in [n]} u_i \mathbf{b}_i : \ \mathbf{u} \in \mathbb{Z}^n \right\}.$$

We call the matrix $\mathbf{B}$ a basis of the lattice Λ. We can have infinitely many different bases for a lattice. They can be transferred from one to another by multiplying a unimodular matrix $\mathbf{U} \in \mathrm{GL}_n(\mathbb{Z})$, where n is the dimension of the lattice. For example, suppose both $\mathbf{B}_1$ and $\mathbf{B}_2$ are bases of a same lattice Λ, then we can always find a unimodular matrix $\mathbf{U}$ such that $\mathbf{B}_2 = \mathbf{B}_1\mathbf{U}$.

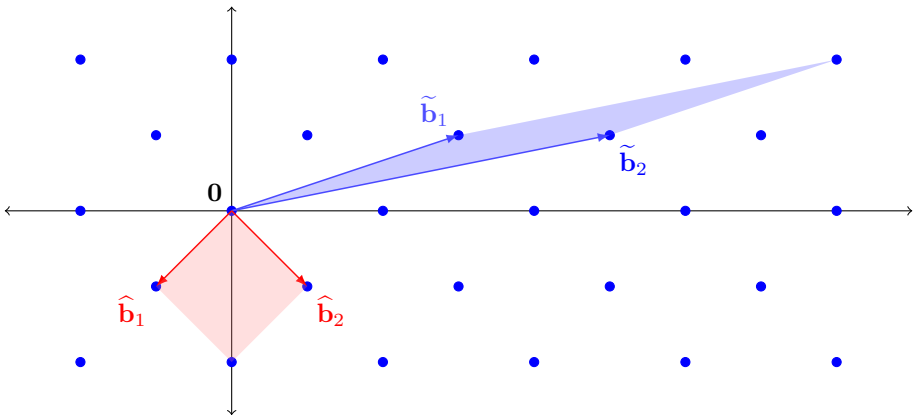

Fig. 2.1: A lattice in $\mathbb{R}^2$ and two of its bases.

For any lattice, we have a unique basis in Hermite normal form, defined as follows.

Definition 2.2 (Hermite normal form)**.** *A matrix $\mathbf{A} \in \mathbb{Z}^{n \times k}$ of full row rank for some integers n, k such that $n \leq k$, is in* Hermite normal form *if it has the form $\mathrm{HNF}(\mathbf{A}) = (\mathbf{H}|\mathbf{0} \cdots \mathbf{0})$, where $\mathbf{H} \in \mathbb{Z}^{n \times n}$ is a square matrix such that*

1. *$h_{i,j} = 0$ for $i < j$;*
2. *$0 \leq h_{i,j} < h_{i,j}$ for $i > j$.*

Here, we recall the uniqueness of the basis of a lattice in Hermite normal form.

Lemma 2.1 ([Sch86, Theorem 4.2])**.** *Let $\mathbf{A}$ and $\widehat{\mathbf{A}}$ for some integers n, k such that $n \leq k$, with Hermite normal forms $(\mathbf{B}|\mathbf{0} \cdots \mathbf{0})$ and $(\widehat{\mathbf{B}}|\mathbf{0} \cdots \mathbf{0})$, respectively. Then $\Lambda(\mathbf{A}) = \Lambda(\widehat{\mathbf{A}})$ if and only if $\mathbf{B} = \widehat{\mathbf{B}}$.*

Further, we also recall the result that any generating set of a lattice can be transformed to a (full-rank) basis of the basis by computing the Hermite normal form.

Lemma 2.2 ([Sch86, Corollary 4.3b]). *For any matrix $\mathbf{A} \in \mathbb{Z}^{n\times k}$ of full row rank for some integers n, k such that $n \leq k$, there is a unimodular matrix $\mathbf{U}$ such that $\mathrm{HNF}(\mathbf{A}) = \mathbf{AU}$ is the Hermite normal form of $\mathbf{A}$.*

According to Lemma 2.2, for any full row rank matrix $\mathbf{A} \in \mathbb{Z}^{n\times k}$ for some integers n, k such that $n \leq k$, we can compute its Hermite normal form $\mathrm{HNF}(\mathbf{A}) = (\mathbf{B}|\mathbf{0}\cdots\mathbf{0})$ with square matrix $\mathbf{B} \in \mathbb{Z}^{n\times n}$, such that $\Lambda(\mathbf{A}) = \Lambda(\mathbf{B})$. Thus, from any generating set $\{\mathbf{a}_i\}_{i\leq k}$, we can obtain a basis $\mathbf{B}$ of the lattice $\Lambda(\mathbf{A})$ by computing its Hermite normal form.

As stated in the next lemma, the Hermite normal form of any matrix of dimension n can be computed in time polynomial in n.

Lemma 2.3 ([Sch86, Corollary 5.3a]). *For any matrix $\mathbf{A} \in \mathbb{Z}^{n\times k}$ of full row rank for some integers n, k such that $n \leq k$, we can find a unimodular matrix $\mathbf{U}$ such that $\mathrm{HNF}(\mathbf{A}) = \mathbf{AU}$, in time $\mathrm{poly}(nk \log \det(\mathbf{A}))$.*

Intuitively, a lattice can be viewed as different grids under different bases, this can be captured by the corresponding parallelepipeds of the bases.

Definition 2.3 (Basis parallelepiped). *For an n-dimensional lattice Λ associated with a basis $\mathbf{B}$, the parallelepiped of lattice $\mathcal{L}$ with respect to $\mathbf{B}$ is $\mathcal{P}(\mathbf{B}) = \{\mathbf{Bx} : 0 \leq x_i < 1\}$.*

Suppose that we shift the parallelepiped by points of the lattice, then each shifted parallelepiped contains exactly one lattice point, *e.g.*, the original parallelepiped contains the origin $\mathbf{0}$. Furthermore, all shifted parallelepipeds (with shift points in the lattice) form a partition of the space $\mathrm{Span}(\mathbf{B})$, where $\mathbf{B}$ is a basis of the lattice.

All the bases of a same lattice share a same determinant. This gives us a first invariant of the lattice, defined as followed.

Definition 2.4 (Determinant). *For an n-dimensional lattice Λ and a basis $\mathbf{B}$, the determinant of Λ is $\det(\Lambda) = \det(\mathbf{B}) = \sqrt{\det(\mathbf{BB}^{\mathrm{T}})}$.*

We can see that the determinant is invariant under different choices of bases as follows. Assume $\mathbf{B}_2 = \mathbf{B}_1\mathbf{U}$ for some unimodular matrix $\mathbf{U}$, we have

$$\sqrt{\det(\mathbf{B}_2^{\mathrm{T}}\mathbf{B}_2)} = \sqrt{\det((\mathbf{B}_1\mathbf{U})^{\mathrm{T}}\mathbf{B}_1\mathbf{U})} = \sqrt{\det(\mathbf{B}_1^{\mathrm{T}}\mathbf{B}_1)\det(\mathbf{U})^2} = \sqrt{\det(\mathbf{B}_1^{\mathrm{T}}\mathbf{B}_1)}.$$

To evaluate the quality of different bases, both the root Hermite factor and the Gram–Schmidt orthogonalization of the bases are important tools.

Definition 2.5 (Root Hermite factor). *Given a basis $\mathbf{B} \in \mathbb{R}^{m\times n}$ of an n-dimensional lattice $\mathcal{L}$, the root Hermite factor of $\mathbf{B}$ is defined as*

$$\delta(\mathbf{B}) = \left(\|\mathbf{b}_1\|/\det(\mathcal{L})^{1/n}\right)^{1/n}.$$

A small root Hermite factor means a high quality for a reduced basis. There is an important quantity called Hermite constant (named after Charles Hermite), which determines how small the first minimum (respectively, the root Hermite factor) can be given the dimension of the lattice.

Definition 2.6 (Hermite Constant)**.** *The n-dimensional Hermite constant γ_n is defined as*

$$\max_{\Lambda \subseteq \mathbb{R}^n} \frac{\lambda_1(\Lambda)^2}{(\det(\Lambda))^{2/n}},$$

where Λ is n-dimensional lattice.

Only a few γ_n's for some small n are known, but we know that $\gamma_n \leq 1 + n/4$ for all n (see [Mar02, Remark 2.7.5]).

For $i \leq n$, we let π_i denote the orthogonal projection onto the linear subspace $(\mathbf{b}_1, \cdots, \mathbf{b}_{i-1})^\perp$. For $i < j \leq n$, we let $\mathbf{B}_{[i,j]}$ denote the local block $(\pi_i(\mathbf{b}_i), \cdots, \pi_i(\mathbf{b}_j))$, and $\Lambda_{[i,j]}$ denote the lattice generated by $\mathbf{B}_{[i,j]}$. It is helpful to consider these projected sub-lattices for reducing a problem in high-dimension to another one in small dimensions. Now we are ready to define the Gram–Schmidt orthogonalization.

Definition 2.7 (Gram–Schmidt orthogonalization, GSO)**.** *Given $\mathbf{B} = (\mathbf{b}_1, \cdots, \mathbf{b}_n)$ a matrix with linearly independent column vectors in $\mathbb{R}^m$, the corresponding GSO is the matrix $\mathbf{B}^* = (\mathbf{b}_1^*, \cdots, \mathbf{b}_n^*)$ where $\mathbf{b}_i^*$ is defined as $\pi_i(\mathbf{b}_i)$.*

To compute the GSO basis vectors, we can first set $\mathbf{b}_1^* = \mathbf{b}_1$, and then compute $\mathbf{b}_i^*$ from $i = 2$ to n as follows:

$$\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^*, \text{ where } \mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}.$$

We can view the GSO as a QR decomposition of the original basis $\mathbf{B}$: $\mathbf{B} = \mathbf{QR}$, where $\mathbf{Q} = (\mathbf{b}_1^*/\|\mathbf{b}_1^*\|, \mathbf{b}_2^*/\|\mathbf{b}_2^*\|, \cdots, \mathbf{b}_n^*/\|\mathbf{b}_n^*\|)$ is an orthogonal matrix, and

$$\mathbf{R} = \begin{pmatrix} \|\mathbf{b}_1^*\| & \mu_{2,1}\|\mathbf{b}_1^*\| & \cdots & \mu_{n,1}\|\mathbf{b}_1^*\| \\ 0 & \|\mathbf{b}_2^*\| & \cdots & \mu_{n,2}\|\mathbf{b}_1^*\| \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & \|\mathbf{b}_n^*\| \end{pmatrix}$$

is an upper triangular matrix with positive diagonal coefficients. Thus the GSO can be seen as a representation of the original basis $\mathbf{B}$ in another form $\mathbf{R}$ with respect to an orthogonal basis $\mathbf{Q}$. The GSO of a basis shares the same determinant, as we can also write $\mathbf{B} = \mathbf{B}^* \widehat{\mathbf{R}}$, where

$$\widehat{\mathbf{R}} = \begin{pmatrix} 1 & \mu_{2,1} & \cdots & \mu_{n,1} \\ 0 & 1 & \cdots & \mu_{n,2} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

with all 1's on the diagonal.

We also introduce a useful lower bound on the first minimum.

Lemma 2.4. *Given any basis $\mathbf{B}$ of an n-dimensional lattice Λ, and its GSO $\mathbf{B}^*$, we have*

$$\lambda_1(\Lambda(\mathbf{B})) \geq \min_{i \in [n]} \|\mathbf{b}_i^*\| > 0.$$

Proof. It suffices to show that for any lattice vector $\mathbf{u} = \mathbf{Bx}$ with some non-zero integer vector $\mathbf{x} \in \mathbb{Z}^n$, we have that $\|\mathbf{Bx}\| \geq \min \|\mathbf{b}_i^*\|$ holds. Let j denotes the largest index with $x_j \neq 0$. The we compute

the projection from $\mathbf{u}$ to the j-th Gram–Schmidt vector, and obtain

$$|\langle \mathbf{u}, \mathbf{b}_j^* \rangle| = |\sum_i^j x_i \langle \mathbf{b}_i, \mathbf{b}_j^* \rangle| = |x_j| \cdot \langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle = |x_j| \cdot \|\mathbf{b}_j^*\|,$$

where we used that $\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle = 0$ for $i < j$ and $\langle \mathbf{b}_j, \mathbf{b}_j^* \rangle = \langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle$. As $|\langle \mathbf{u}, \mathbf{b}_j^* \rangle| \leq \|\mathbf{u}\| \cdot \|\mathbf{b}_j^*\|$, thus we have

$$\|\mathbf{u}\| \geq |x_j| \cdot \|\mathbf{b}_j^*\| \geq \|\mathbf{b}_j^*\| \geq \min_{i \in [n]} \|\mathbf{b}_i^*\|.$$

□

Next, we recall the definition of the dual of a lattice.

Definition 2.8 (Dual lattice)**.** *For an n-dimensional lattice Λ, we let $\Lambda^\dagger = \{\mathbf{y} \in \mathbb{R}^n : \forall \mathbf{x} \in \Lambda, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}\}$ denote the dual of the lattice Λ.*

Further, we let $\mathbf{D} = (\mathbf{d}_1, \cdots, \mathbf{d}_n)$ denote the dual basis of a basis $\mathbf{B} = (\mathbf{b}_1, \cdots, \mathbf{b}_n)$. It is defined by $\mathbf{B}^{\mathrm{T}}\mathbf{D} = \mathbf{I}_n$, where $\mathbf{I}_n$ denotes the identity matrix of dimension n. We have a strong connection between the mimima of the primal lattice and its dual lattice.

Definition 2.9 (Successive minima)**.** *For any lattice Λ, the i-th minimum $\lambda_i(\Lambda)$ is the radius of the smallest ball with center the origin and containing i linearly independent lattice vectors:*

$$\lambda_i(\Lambda) = \inf\{r : \ \dim(\mathrm{Span}(\Lambda \cap \mathcal{B}(\mathbf{0}, r))) \geq i\}.$$

In particular, we let $\lambda_1(\Lambda)$ (respectively, $\lambda_1^\infty(\Lambda)$) denote the ℓ_2-norm (respectively, ℓ_∞-norm) of a shortest non-zero vector of Λ.

If the first minimum of the primal lattice becomes smaller, the last minimum of its dual lattice is likely to become larger. This relation can be quantified as follows.

Lemma 2.5 ([Ban93, Theorem 2.1])**.** *For any n-dimensional lattice Λ, we have $1 \leq \lambda_1(\Lambda) \cdot \lambda_n(\Lambda^\star) \leq n$.*

Next, we present two important results on the first minimum of lattices, one is rigorous, another one is heuristic.

Distribution of the first minimum of a random lattice. If we consider random lattices, there is a rigorous result on the distribution of first minimum. We use $\Gamma_n = \{\Lambda \in \mathbb{R}^n | \mathrm{vol}(\Lambda) = 1\}$ to denote the set of all full-rank lattices of rank n with unit volume. In [Che09], Chen gives the following statement on the distribution of the shortest vector in a random lattice as a direct corollary of [Söd11, Theorem 1].

Corollary 2.1 ([Che09, Corollary 3.1.4])**.** *For a lattice $\Lambda \in \Gamma_n$, the distribution of $v_n \cdot \lambda_1(\Lambda)^n$ converges in distribution to* Expo$[1/2]$ *as $n \to \infty$.*

This is a key ingredient used in Chapter 5 for building a more precise simulator for the BKZ algorithm.

For an n-dimensional random lattice Λ, if we consider $\lambda_1(\Lambda)$ as a random variable $Y = X^{1/n} \cdot \mathrm{GH}(\Lambda)$, where X is a random variable distributed as an Expo$[1/2]$ as $n \to \infty$, then the expected value of λ_1 as variable X is

$$\mathbb{E}(\lambda_1(\Lambda)) = \left(2^{1/n} \cdot \frac{\Gamma(1/n)}{n}\right) \cdot \frac{1}{v_n^{1/n}} \cdot \det(\Lambda)^{1/n}.$$

Gaussian heuristic. Given an n-dimensional lattice Λ with volume $\mathrm{vol}(\Lambda)$, the Gaussian heuristic predicts that the number of lattice points in a measurable subset $\mathcal{S}$ of $\mathbb{R}^n$ of volume $\mathrm{vol}(\mathcal{S})$ is approximately equal to $\mathrm{vol}(\mathcal{S})/\mathrm{vol}(\Lambda)$. Assume that Gaussian heuristic is true when $\mathrm{vol}(\mathcal{S}) \approx \mathrm{vol}(\Lambda)$. Then

we can select $\mathcal{S}$ as an n-ball with volume equal to $\mathrm{vol}(\Lambda)$. In this case, the radius of $\mathcal{S}$ can be expected to be an approximation of $\lambda_1(\Lambda)$, which is denoted by

$$\mathrm{GH}(\Lambda) = v_n^{-1/n} \cdot \mathrm{vol}(\Lambda)^{1//n}.$$

Futher, Minkowski's first theorem states that $\lambda_1(\Lambda) \leq 2 \cdot v_n^{-1/n} \cdot \mathrm{vol}(\Lambda)^{1/n}$, which is larger than $\mathrm{GH}(\Lambda)$ by a factor of 2.

2.1.2 Gaussian distributions over lattices

In the following, we recall some important properties of discrete Gaussian distributions, in particular over lattices. We use them extensively in Chapter 4.

Definition 2.10 (Discrete Gaussian over lattices)**.** *For $\mathbf{c} \in \mathbb{R}^n$, $r > 0$ and lattice Λ, the discrete Gaussian distribution over Λ, with center parameter $\mathbf{c}$ and standard deviation parameter r is defined as:*

$$\mathcal{D}_{\Lambda,r,\mathbf{c}}(\mathbf{x}) = \frac{\rho_{r,\mathbf{c}}(\mathbf{x})}{\rho_{r,\mathbf{c}}(\Lambda)}, \forall \mathbf{x} \in \Lambda,$$

where $\rho_r(\mathbf{x}) = \exp(-\pi\|\mathbf{x}\|^2/r^2)$.

If $\mathbf{c} = \mathbf{0}$, we omit the $\mathbf{0}$ subscript.

For simplicity of analysis in Chapter 4, we need to use the following statement called Poisson Summation Formula. Here, we recall a special form of it, in which the Gaussian function is considered.

Lemma 2.6 (Poisson summation formula)**.** *For any n-dimensional lattice Λ and vector $\mathbf{u} \in \mathbb{R}^n$, it holds that*

$$\rho_r(\Lambda + \mathbf{u}) = \det(\Lambda^\star) \cdot r^n \cdot \sum_{\mathbf{x} \in \Lambda^\star} e^{2\pi i \langle \mathbf{x}, \mathbf{u} \rangle} \rho_{1/r}(\mathbf{x}).$$

In general, it says that the sum of evaluation of an appropriate function on the primal lattice, is equal to the sum of the evaluation of the Fourier transform of the function on the dual lattice, up to a factor related to the determinant of the lattice.

In the following, we introduce a tail bound for the discrete Gaussian distribution over the integer lattice $\mathbb{Z}$.

Lemma 2.7 (Adapted from [Ban93, Lemma 1.5(i)])**.** *For any $r > 0$, we have $\rho_r(\mathbb{Z} \backslash [-\sqrt{\kappa}r, \sqrt{\kappa}r]) < 2^{-\Omega(\kappa)} \cdot \rho_r(\mathbb{Z})$ for parameter κ growing to infinity.*

Proof. On one hand, from the Poisson summation formula (Lemma 2.6) for the lattice $\mathbb{Z}$, we have for $r \geq 1$

$$\rho_{2r}(\mathbb{Z}) = \det(\mathbb{Z}) \cdot 2r \cdot \sum_{y \in \mathbb{Z}} \rho_{1/(2r)}(y) \leq 2r \cdot \sum_{y \in \mathbb{Z}} \rho_{1/r}(y) = 2\rho_r(\mathbb{Z}),$$

where for the inequality, we used the fact that $\rho_{1/(2r)} \leq \rho_{1/(r)}$.

On the other hand, for any $B \geq 0$, we have

$$\begin{aligned} \rho_{2r}(\mathbb{Z}) > \rho_{2r}(\mathbb{Z} \setminus [-B, B]) &= \sum_{y \in \mathbb{Z}, |y| > B} e^{-\frac{\pi y^2}{(2r)^2}} \\ &= \sum_{y \in \mathbb{Z}, |y| > B} e^{\frac{3}{4}\frac{\pi y^2}{(2r)^2}} e^{-\frac{\pi y^2}{r^2}} \geq e^{\frac{3}{4}\frac{\pi B^2}{(2r)^2}} \rho_r(\mathbb{Z} \setminus [-B, B]). \end{aligned}$$

Combining the two inequalities for ρ_{2r} and setting $B = \sqrt{\kappa}r$, we obtain

$$\rho_r(\mathbb{Z} \setminus [-\sqrt{\kappa}r, \sqrt{\kappa}r]) < e^{-\frac{3}{4}\frac{\pi(\sqrt{\kappa}r)^2}{(2r)^2}} \rho_{2r}(\mathbb{Z}) \leq 2e^{-\frac{3\pi\kappa}{16}} \rho_r(\mathbb{Z}) = 2^{-\Omega(\kappa)} \rho_r(\mathbb{Z}).$$

□

From Lemma 2.7, we can see that the tail of the discrete Gaussian distribution over the integer lattice has only negligible proportion compared to the whole sum. More generally, the tail of the Gaussian distribution over any shifted lattice also has only negligible proportion compared to the whole sum.

Lemma 2.8 ([Ban93, Lemma 1.5(ii)]). *For any $r > 0$, n-dimensional lattice Λ and $\mathbf{u} \in \mathbb{R}^n$, it holds that*

$$\rho_r(\Lambda + \mathbf{u} \backslash \mathcal{B}(\mathbf{0}, \sqrt{n}r)) < 2^{-\Omega(n)} \rho_r(\Lambda).$$

2.1.3 Lattice sparsification

In the following, we introduce a tool named lattice sparsification, which was introduced by Khot [Kho03] and later used in hardness proofs of lattice problems [SD16b, ASD18]. The sparsification technique is also the main ingredient that we use to achieve a tighter reduction between BDD and uSVP in Chapter 3.

Lemma 2.9 ([SD16a, Corollary 2.16]). *For any prime p, collection of vectors $\mathbf{v}_1, \cdots, \mathbf{v}_N \in \mathbb{Z}_p^n \setminus \{\mathbf{0}\}$, and $\mathbf{x} \notin \{\mathbf{v}_i\}_{i \le N}$, we have*

$$\frac{1}{p} - \frac{N}{p^2} - \frac{N}{p^{n-1}} \le \Pr_{\mathbf{z},\mathbf{u} \hookleftarrow U(\mathbb{Z}_q^n)} \left[\begin{array}{lll} \forall i, & \langle \mathbf{z}, \mathbf{v}_i + \mathbf{u} \rangle & \neq 0 \bmod p \\ & \langle \mathbf{z}, \mathbf{x} + \mathbf{u} \rangle & = 0 \bmod p \end{array} \right] \le \frac{1}{p} + \frac{1}{p^n}.$$

The upper bound in Lemma 2.9 is not used in Chapter 3, but we keep it to show that the difference between the upper and lower bounds is small, and thus that the lower bound is almost tight. Here, we give a brief idea of how the lower bound is obtained. We can first define the following $N + 1$ sets:

$$A = \{\mathbf{z}' \in \mathbb{Z}_p^N \mid \langle \mathbf{z}', \mathbf{x} + \mathbf{c} \rangle \equiv 0 \bmod p\}$$

and

$$B_i = \{\mathbf{z}' \in \mathbb{Z}_p^N \mid \langle \mathbf{z}', \mathbf{v}_i + \mathbf{c} \rangle \not\equiv 0 \bmod p\}, \ \forall i \in [N].$$

Then the success probability can be roughly computed as

$$\frac{|A \backslash \bigcup_i B_i|}{|\mathbb{Z}_p^n|} \ge \frac{|A| - \sum_{i \in [N]} |A \cap B_i|}{p^n}.$$

This gives us $\frac{1}{p} - \frac{N}{p^2}$, and the last term $\frac{N}{p^{n-1}}$ is due to the possible linear relations between $\mathbf{x} + \mathbf{c}$ and $\{\mathbf{v}_i + \mathbf{c}\}_{i \in [N]}$.

Lemma 2.9 also leads to the definition of the following sub-lattice

$$\Lambda_{p,\mathbf{z}} = \left\{\mathbf{x} \mid \langle \mathbf{B}^{-1}\mathbf{x}, \mathbf{z} \rangle = 0 \mod p\right\},$$

in which only the vectors of $\Lambda(\mathbf{B})$ with coordinates orthogonal (mod p) to $\mathbf{z}$ will be kept. To have an intuition of the sparsification result above, we include Figure 2.2. The lattice $\Lambda_{p,\mathbf{z}}$ (on the right-hand side) is a sublattice of Λ (on the left-hand side), defined by $p = 5$, $\mathbf{z} = (1, 2)$. In particular, the lattice points $(1, 2)$ and $(3, 1)$ are in the sparsified lattice $\Lambda_{p,\mathbf{z}}$ because these lattice points are orthogonal to $\mathbf{z}$ modulus p.

Further, we can take the modulus $p = 2N$, in which case, we can eliminate polynomially many lattice points, with non-negligible probability. In fact, the limitation of the above sparsification to efficiently remove only polynomially many lattice vectors is due to keeping one specific lattice vector. This limitation can be circumvented in other situations. For example, suppose there are two disjoint

sets: $\mathcal{S}_{good}$ and $\mathcal{S}_{bad}$, each with super-polynomially many lattice vectors. If we only require to keep any one from set $\mathcal{S}_{good}$ and remove all in set $\mathcal{S}_{bad}$, and $\#\mathcal{S}_{good}$ is significantly larger than $\#\mathcal{S}_{bad}$, then we can still succeed with non-negligible probability. This situation is considered by Aggarwal and Stephens-Davidowitz [ASD18] in their proof of exponential time lower bound for solving SVP under the gap-exponential time hypothesis. The idea is similar to our brief proof of Lemma 2.9 above. In this case, we can construct the following $\#\mathcal{S}_{good}+\#\mathcal{S}_{bad}$ many sets: $A = \{\mathbf{z}' \in \mathbb{Z}_p^N \mid \langle \mathbf{z}', \mathbf{x}_i + \mathbf{c}\rangle \equiv 0 \bmod p\}$ and $B_j = \{\mathbf{z}' \in \mathbb{Z}_p^N \mid \langle \mathbf{z}', \mathbf{v}_j + \mathbf{c}\rangle \not\equiv 0 \bmod p\}$ for $i \in [\#\mathcal{S}_{good}]$ and $j \in [\#\mathcal{S}_{bad}]$. Then the success probability can be computed as:

$$\frac{|\bigcup_i A_i \setminus \bigcup_j B_j|}{|\mathbb{Z}_p^n|} \geq \frac{|\bigcup_i A_i| - \sum_{i\in[\#\mathcal{S}_{good}], j\in[\#\mathcal{S}_{bad}]} |A_i \cap B_j|}{p^n}.$$

This is non-negligible when $\#\mathcal{S}_{good}$ is asymptotically larger than $\#\mathcal{S}_{bad}$.

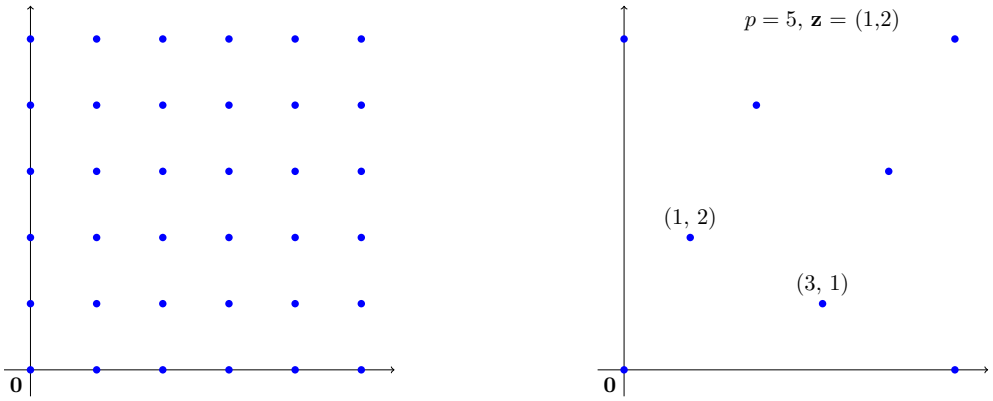

Fig. 2.2: An example of sparsification with $p = 5$, $\mathbf{z} = (1, 2)$.

The lemma below explains that we can efficiently compute a basis of such a sparsified sub-lattice.

Lemma 2.10. *There exists a polynomial-time algorithm which, given as inputs a basis* $\mathbf{B} \in \mathbb{Q}^{n\times n}$ *of an* n*-dimensional lattice* Λ*, an integer* p *and a vector* $\mathbf{z} \in \mathbb{Z}_p^n$*, outputs a basis* $\mathbf{B}_{p,\mathbf{z}}$ *of the lattice* $\Lambda_{p,\mathbf{z}} = \{\mathbf{x} \in \Lambda \mid \langle \mathbf{z}, \mathbf{B}^{-1}\mathbf{x}\rangle = 0 \bmod p\}$.

Proof. According to the definition of the lattice $\Lambda_{p,\mathbf{z}}$, we have

$$\langle \mathbf{z}, \mathbf{y}\rangle = 0 \bmod p,$$

where $\mathbf{y} = \mathbf{B}^{-1}\mathbf{x}$ and $\mathbf{x} \in \Lambda_{p,\mathbf{z}}$. We first compute a basis $\mathbf{S}$ of all solutions $\mathbf{y}$ of equation $\langle \mathbf{z}, \mathbf{y}\rangle = 0$ over $\mathbb{Z}^n$. We compute the column Hermite normal form of $[\mathbf{S}|p\mathbf{I}_n] \in \mathbb{Z}^{n\times 2n}$; and obtain the nonzero columns $\mathbf{S}' \in \mathbb{Z}^{n\times n}$. The columns of $\mathbf{S}'$ generate the lattice orthogonal to $\mathbf{z} \pmod p$. In the end, we compute $\mathbf{B}_{p,\mathbf{z}} = \mathbf{B}\mathbf{S}'$, which is a basis for the lattice $\Lambda_{p,\mathbf{z}}$.

□

As stated in Lemma 2.9, we can construct a sub-lattice such that polynomially many specific lattice vectors can be removed with non-negligible probability. The first question to ask is until how far away, given any center, there are only polynomially many lattice vectors.

The following result in the book by Micciancio and Goldwasser [MG02] considers this question. In fact, the statement in their result is stronger than our statement in Lemma 2.11. In [MG02, Theorem 5.2], it is proved that the maximum number of points at distance at least ℓ from each other that

can be placed in a sphere of radius $\ell/\sqrt{2}$ is $2n$. By taking the lattice minimum as a lower bound on the distance of every two points, we obtain the statement in Lemma 2.11.

Lemma 2.11 ([MG02, Theorem 5.2]). *For any n-dimensional lattice Λ and any vector $\mathbf{t} \in \mathbb{Q}^n$, we have $\#\Lambda \cap \mathcal{B}(\mathbf{t}, \lambda_1(\Lambda)/\sqrt{2}) \leq 2n$.*

For the sake of completeness, we give the full proof of [MG02, Theorem 5.2]. The original proof goes a bit too fast, and some arguments were lacking. We will explained this on the way.

Proof. It proceeds by induction. It is assumed that the statement is true for some dimension n, and then is proved for dimension $n+1$. For dimension $n+1$, we consider N points $\mathbf{x}_1, \cdots, \mathbf{x}_N$ in $\mathbb{Q}^{n+1}$ such that $\|\mathbf{x}_i - \mathbf{x}_j\| \geq \ell$ and $\|\mathbf{x}_i\| \leq \ell/\sqrt{2}$ for all $i, j \in [N]$. Note that for all $i \neq j$, we have

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \frac{1}{2}\left(\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - \|\mathbf{x}_i - \mathbf{x}_j\|^2\right) \leq \frac{1}{2}\left(\frac{\ell^2}{2} + \frac{\ell^2}{2} - \ell^2\right) = 0.$$

Take $\mathbf{x}_N$ as the north pole. The idea is to adjust the N points to the poles or on the equator of an $(n+1)$-dimensional sphere of radius $\ell/\sqrt{2}$. Now we define the set of vectors

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

and let $\mathbf{x}_i'' = \sqrt{2}\mathbf{x}_i'/\|\mathbf{x}_i'\|$. Note that $\|\mathbf{x}_i''\|^2 = \ell$ holds for all i (i.e., the vector $\mathbf{x}_i''$ is on the surface). Further, we have either $\mathbf{x}_i'' = \pm\mathbf{x}_N''$ (i.e., the vector $\mathbf{x}_i''$ is a pole) or $\langle \mathbf{x}_i'', \mathbf{x}_N'' \rangle = 0$ (i.e., $\mathbf{x}_i''$ is on the equator).

Finally, we prove that $\|\mathbf{x}_i'' - \mathbf{x}_j''\|^2 \geq 4$ for all $i \neq j$. If $\mathbf{x}_i'' = \pm\mathbf{x}_N''$ or $\mathbf{x}_j'' = \pm\mathbf{x}_N''$, it is obviously satisfied. Assume that both $\mathbf{x}_i''$ and $\mathbf{x}_j''$ are not equal to $\pm\mathbf{x}_N''$, we have

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

because $\langle \mathbf{x}_i, \mathbf{x}_j \rangle, \langle \mathbf{x}_i, \mathbf{x}_N \rangle, \langle \mathbf{x}_j, \mathbf{x}_N \rangle \leq 0$ and $\langle \mathbf{x}_N, \mathbf{x}_N \rangle > 0$.

As a result, we have exactly one point in the south pole and the north pole respectively. Then we proceed by induction on the points that are on the n-dimensional equator: there are at most $2n$ of these. Thus, after adjustment, we have at most $2n + 2 = 2(n+1)$ points to be placed in $\mathbb{Q}^{n+1}$ [1]. □

In the following, we supplement the proof of Lemma 2.11 by proving the one-to-one correspondence between points before and after adjustment. In order to present the orthogonalization underlying the adjustment more naturally, we also modify the maps used in the proof of [MG02, Theorem 5.2], as follows:

$$f(\mathbf{x}_i) = \begin{cases} \mathbf{x}_i - \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2}\mathbf{x}_N, & \text{if } \langle \mathbf{x}_i, \mathbf{x}_N \rangle \neq \pm\|\mathbf{x}_i\|\|\mathbf{x}_N\| \\ \mathbf{x}_i, & \text{otherwise} \end{cases}$$

and

$$g(\mathbf{x}_i') = \frac{\sqrt{2}}{\|\mathbf{x}_i'\|}\mathbf{x}_i'$$

[1] This is not sufficient for the statement in Lemma 2.11. We also need to prove that N (the number of starting points $\mathbf{x}_i$'s) is not bigger than $2(n+1)$, or, in other words, that any two distinct points of the N initial points will not be adjusted to the same point of the $(n+1)$-dimensional sphere .

for all $i \in [N]$. We let $\mathbf{x}'_i$ denote $f(\mathbf{x}_i)$ and let $\mathbf{x}''_i$ denote $g(\mathbf{x}'_i)$. The map $g(f(\mathbf{x}_i)) = \mathbf{x}''_i$ is a bijection as all the points $\mathbf{x}_i$ are far away from each other, according to the assumption. We prove this missing statement here.

Proof. First, we recall that $\langle \mathbf{x}_i, \mathbf{x}_j \rangle \leq 0$ for any $i \neq j$ from last proof above. We will use that claim several times in the proof.

We show that for any $i \neq j$, if $\mathbf{x}'_i$ and $\mathbf{x}'_j$ are colinear, then $\langle \mathbf{x}'_i, \mathbf{x}'_j \rangle \leq 0$. Thus, even after applying the second map, the relation between the $\mathbf{x}_i$'s and the $\mathbf{x}''_i$'s is one-to-one. We consider three cases.

Assume first that $\langle \mathbf{x}_i, \mathbf{x}_N \rangle = \pm \|\mathbf{x}_i\| \|\mathbf{x}_N\|$ and $\langle \mathbf{x}_j, \mathbf{x}_N \rangle = \pm \|\mathbf{x}_j\| \|\mathbf{x}_N\|$. In this case, we have $\mathbf{x}'_i = \mathbf{x}_i$ and $\mathbf{x}'_j = \mathbf{x}_j$. As we have $\langle \mathbf{x}_i, \mathbf{x}_j \rangle \leq 0$, thus $\mathbf{x}'_i$ and $\mathbf{x}'_j$ are colinear but with $\langle \mathbf{x}'_i, \mathbf{x}'_j \rangle \leq 0$.

Assume further that $\langle \mathbf{x}_i, \mathbf{x}_N \rangle \neq \pm \|\mathbf{x}_i\| \|\mathbf{x}_N\|$ and $\langle \mathbf{x}_j, \mathbf{x}_N \rangle = \pm \|\mathbf{x}_j\| \|\mathbf{x}_N\|$. In this case, we have $\langle \mathbf{x}'_i, \mathbf{x}_N \rangle = 0$ and $\langle \mathbf{x}'_j, \mathbf{x}_N \rangle = \pm \|\mathbf{x}_j\| \|\mathbf{x}_N\| \neq 0$. Thus, we obtain that $\mathbf{x}'_i$ is not colinear with $\mathbf{x}'_j$.

For the rest of the proof, we assume that $\langle \mathbf{x}_i, \mathbf{x}_N \rangle \neq \pm \|\mathbf{x}_i\| \|\mathbf{x}_N\|$ and $\langle \mathbf{x}_j, \mathbf{x}_N \rangle \neq \pm \|\mathbf{x}_j\| \|\mathbf{x}_N\|$. We assume by contradiction that $\mathbf{x}'_i$ and $\mathbf{x}'_j$ are colinear and $\langle \mathbf{x}'_i, \mathbf{x}'_j \rangle > 0$. It is equivalent to assume that there exists $t > 0$ such that

$$\mathbf{x}'_i = \mathbf{x}_i - \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \mathbf{x}_N = t \cdot \left(\mathbf{x}_j - \frac{\langle \mathbf{x}_j, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \mathbf{x}_N \right) = t \cdot \mathbf{x}'_j. \tag{2.1}$$

Write $\mathbf{x}_j = \mathbf{x}_i + (1/t) \cdot \mathbf{v}$ for some vector $\mathbf{v}$ and use it to substitute $\mathbf{x}_j$ in Equation (2.1). Without loss of generality, we may assume that $\langle \mathbf{v}, \mathbf{x}_N \rangle \leq 0$ (if it is not the case, then we exchange $\mathbf{x}_i$ and $\mathbf{x}_j$). By substitution in Equation (2.1), we obtain

$$\mathbf{v} = (1-t) \cdot \left(\mathbf{x}_i - \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \mathbf{x}_N \right) + \frac{\langle \mathbf{v}, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \mathbf{x}_N.$$

We have

$$\begin{aligned}
\langle \mathbf{x}_i, \mathbf{x}_j \rangle &= \left\langle \mathbf{x}_i, \mathbf{x}_i + \frac{1-t}{t} \cdot \left(\mathbf{x}_i - \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \mathbf{x}_N \right) + \frac{1}{t} \cdot \frac{\langle \mathbf{v}, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \mathbf{x}_N \right\rangle \\
&= \|\mathbf{x}_i\|^2 + \frac{1-t}{t} \cdot \left(\|\mathbf{x}_i\|^2 - \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle^2}{\|\mathbf{x}_N\|^2} \right) + \frac{1}{t} \cdot \frac{\langle \mathbf{v}, \mathbf{x}_N \rangle \langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \\
&= \frac{1}{t} \|\mathbf{x}_i\|^2 - \frac{1-t}{t} \cdot \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle^2}{\|\mathbf{x}_N\|^2} + \frac{1}{t} \frac{\langle \mathbf{v}, \mathbf{x}_N \rangle \langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \\
&\geq \frac{\langle \mathbf{x}_i, \mathbf{x}_N \rangle^2}{\|\mathbf{x}_N\|^2} + \frac{1}{t} \frac{\langle \mathbf{v}, \mathbf{x}_N \rangle \langle \mathbf{x}_i, \mathbf{x}_N \rangle}{\|\mathbf{x}_N\|^2} \\
&\geq 0.
\end{aligned}$$

This leads to a contradiction as we have $\langle \mathbf{x}_i, \mathbf{x}_j \rangle \leq 0$. □

2.2 Worst-case lattice problems

In this section, we introduce some well-known worst-case lattice problems. They serve as the hardness foundations of lattice-based cryptography. In particular, the two central problems SVP and CVP, as well as some of their variants will be recalled. Finally, we also give some results in approximating the first minimum and distance from any target vector to lattice. They are sufficient for presenting our work in Chapter 3. We refer the reader to [Ste15] for more details on known relations between lattice problems, [Emd81, Ajt98, Mic01a, Mic12] for more details on the hardness of SVP and CVP, and [MV10b, MV10c, ADRSD15, AKS01, HS07, GNR10] for more details on the algorithms for solving SVP and CVP.

2.2.1 Definitions and relations

Now we introduce the two most well-known problems in lattices: SVP and CVP, as well as their well-known variants.

Definition 2.11 (Shortest Vector Problem, SVP)**.** *Given as input a lattice basis* $\mathbf{B}$*, the goal is to find a vector* $\mathbf{s} \in \Lambda(\mathbf{B})$ *of norm* $\lambda_1(\Lambda(\mathbf{B}))$*.*

We also introduce its approximation version, which is closely related to the security foundation of lattice-based cryptography.

Definition 2.12 (SVP_γ)**.** *Given as input a lattice basis* $\mathbf{B}$ *and a factor* γ*, the goal is to find a non-zero vector* $\mathbf{s} \in \Lambda(\mathbf{B})$ *such that* $\mathbf{s} \leq \gamma \cdot \lambda_1(\Lambda(\mathbf{B}))$*.*

We also have a decision (gap) version of SVP_γ.

Definition 2.13 ($\mathrm{GapSVP}_{n,\gamma}$)**.** *Given as input an* n*-dimensional lattice associated with basis* $\mathbf{B}$ *and a factor* γ*, for* $d > 0$*, the goal is to distinguish the following two cases:*

1. *Yes instance:* $\lambda_1(\Lambda(\mathbf{B})) < d$*;*
2. *No instance:* $\lambda_1(\Lambda(\mathbf{B})) > \gamma \cdot d$*.*

Here we briefly review the two main practical algorithms for solving SVP (or SVP_γ): the sieve algorithm and the enumeration algorithm. Note that there are algorithms with better cost bounds [MV10b, MV10c, ADRSD15], but they are less practical compared to the two algorithms below.

Sieve algorithm. The sieve algorithm was first introduced by Ajtai *et al.* [AKS01]. The idea is to first sample a lot of lattice vectors in $\Lambda \cap \mathcal{B}_n(r)$ for some initial radius r. We can assume $r = 2^{n/2} \cdot \Lambda$ according to Lemma 2.15. Once we have exponentially many such vectors, we can prove that there is at least a pair of vectors with their addition falling into $\Lambda \cap \mathcal{B}_n(r/c)$ for some constant c. Then by repeating this process polynomially many times, we can successfully find some short vectors with norm close to the first minimum. The polynomial number of iterations contribute to the total solving time by a factor $\mathrm{poly}(n)$. In total, the algorithm runs in time exponential in n.

Enumeration algorithm. First appeared in the 1980s [Kan83, FP83], the enumeration algorithm is the most practical algorithm for solving SVP. The main idea is to search for the optimal solution within a given range. Suppose we aim to find a shortest vector with norm bounded by r in an n-dimensional lattice Λ, the strategy of the enumeration is to find all short vectors with norm $\leq r$ (as potential projections of some shortest vectors) in projected lattice $\pi_i(\Lambda)$ for i from n down to 1. This process can be viewed as a search on a tree: the i-th level is all short vectors in $\Lambda_{[i,n]}$; by going to the $(i-1)$-th level, we add multiples of $\mathbf{b}^*_{i-1}$ to short vectors found in th i-th level, and we keep only short resulting vector within given norm. We continue to search until we reach the first level, where we find a shortest non-zero vector. In practice, we use the depth first strategy when we search through the tree. Thus the enumeration algorithm is space efficient. It was noticed by Hanrot and Stehlé [HS07] that the number of nodes in level i under the Gaussian heuristic, is

$$N_i = \frac{1}{2} \cdot \frac{V_{n-i+1}(r)}{\mathrm{vol}(\Lambda_{[i,n]})}.$$

When the searching radius r is estimated by Gaussian heuristic, and Geometry Series Assumption (refer to Subsection 2.5) is assumed, the number of nodes in level $\lfloor n/2 \rfloor$ is super-exponential in n. In fact, it was shown by Gama *et al* [GNR10] that the number of nodes in level $\lfloor n/2 \rfloor$ is significantly larger than in other levels.

Overall, the sieve algorithm is asymptotically faster than the enumeration algorithm. However, because of the constants hidden in the exponents, there is a crossover point between the curves of complexity of solving time of these two algorithms. For example, the enumeration algorithm seems better than the sieve algorithm for solving SVP with practical dimensions, e.g., less than 200. Furthermore, Alkim *et al* [ADPS16] state that the sieve will become more efficient than enumeration when the dimension is ≥ 250. More recently, Ducas [Duc18] shown that we can save $\Theta(n/\log n)$ dimensions with the sieve algorithm for solving SVP on n-dimensional lattices.

Definition 2.14 (Closest Vector Problem, CVP)**.** *Given as input a lattice basis* $\mathbf{B}$ *and a vector* $\mathbf{t}$*, the goal is to find a vector* $\mathbf{v} \in \Lambda(\mathbf{B})$ *closest to* $\mathbf{t}$*.*

Correspondingly, we also have a decision (gap) version of CVP_γ.

Definition 2.15 ($\text{GapCVP}_{n,\gamma}$)**.** *Given as input an* n*-dimensional lattice associated with basis* $\mathbf{B}$*, a target vector* $\mathbf{t}$ *and a factor* γ*, for* $d > 0$*, the goal is to distinguish the following two cases:*

1. *Yes instance:* $\text{dist}(\mathbf{t}, \Lambda(\mathbf{B})) < d$*;*
2. *No instance:* $\text{dist}(\mathbf{t}, \Lambda(\mathbf{B})) > \gamma \cdot d$*.*

There is an efficient reduction from SVP to CVP [GMSS99]. However, in the converse reduction from CVP to SVP from [Mic01a], the dimension explodes from n to n^c for some constant $c > 1$ and the reduction is probabilistic.

Now, we introduce two promise variants, whose instances are a specific subset of instances of SVP and CVP. They are closely related to the underlying security of lattice-based cryptographic primitives [Reg09, LM09].

Definition 2.16 (unique SVP_γ, uSVP_γ)**.** *Let* $\gamma \geq 1$*. Given as input a lattice basis* $\mathbf{B}$ *such that* $\lambda_2(\mathbf{B}) \geq \gamma \cdot \lambda_1(\mathbf{B})$*, the goal is to find a vector* $\mathbf{s} \in \Lambda(\mathbf{B})$ *of norm* $\lambda_1(\Lambda(\mathbf{B}))$*.* SVP *corresponds to* $\gamma = 1$*.*

uSVP is a promise variant of SVP in the sense that the second minimum is guaranteed to be much larger than the first minimum. Said differently, any vector that is not parallel to the two shortest vectors of norms λ_1, has norm much larger than λ_1. Thus any approximate shortest vector within this gap should be the shortest vector itself or multiple of it. In the literature (in [LM09], for example), uSVP is sometimes be defined with a strict lower bound on $\lambda_2(\mathbf{B})$. We allow equality (as in [LWXZ14]), as it is more convenient for our presentation. Note that Lemma 2.12 below implies that these two variants are computationally equivalent.

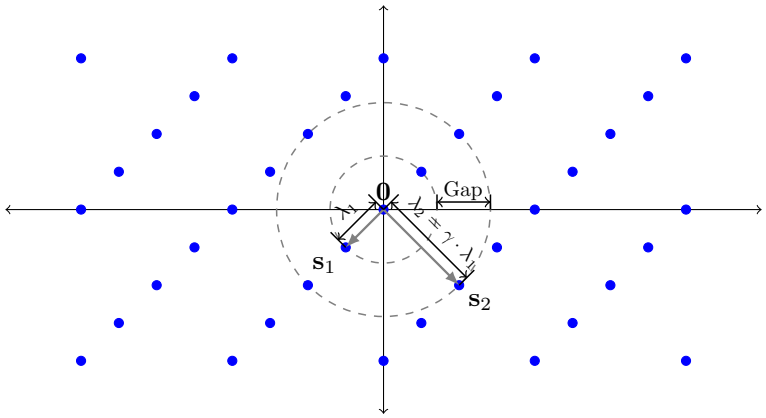

Fig. 2.3: An example of a uSVP instance.

Definition 2.17 (Bounded Distance Decoding, BDD_α)**.** *Let $\alpha > 0$. Given as inputs a lattice basis $\mathbf{B}$ and a vector $\mathbf{t}$ such that $\mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \leq \alpha \cdot \lambda_1(\mathbf{B})$, the goal is to find a lattice vector $\mathbf{v} \in \Lambda(\mathbf{B})$ closest to $\mathbf{t}$.*

As opposed to CVP (in which case the target vector can be as far away from the lattice as possible), the BDD problem promises that the target vector is within a bounded distance from the lattice. Note that in some works, the range of α is restricted to $(0, 1/2)$. This is to guarantee that there is exactly one element of Λ in the ball of radius $\alpha \cdot \lambda_1(\Lambda)$ centered on $\mathbf{t}$. The problem is well-defined even for large α, and in this thesis we actually consider $\alpha \geq 1/2$.

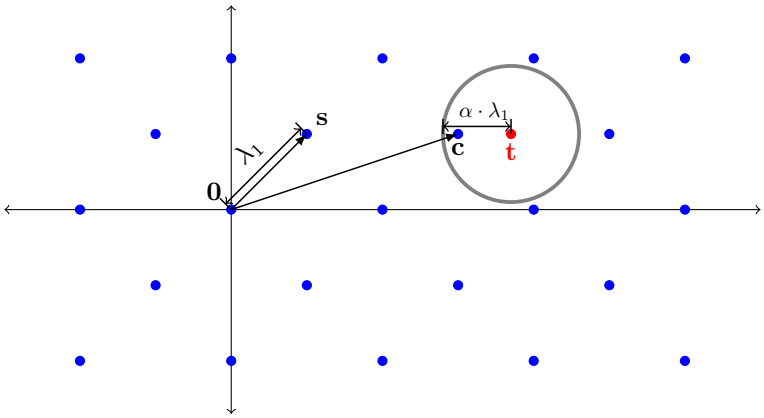

Fig. 2.4: An example of BDD instance.

In the next lemma, it is stated that BDD_α is equivalently hard for any parameter α' that is within a factor $(1 - 1/n)^c$ of α, for any constant c.

Lemma 2.12 ([LM09, Corollary 2])**.** *For any $\alpha > 0$, any constant $c > 0$, there is a polynomial-time reduction from BDD_α to $\mathrm{BDD}_{\alpha(1-1/n)^c}$.*

In [LM09], Lyubashevsky and Micciancio also establish reductions between BDD and uSVP in both directions. We recall them as follows.

Lemma 2.13 ([LM09, Theorem 1])**.** *For any $\gamma \geq 1$, there is a polynomial-time reduction from $\mathrm{BDD}_{1/(2\gamma)}$ to uSVP_γ.*

Lemma 2.14 ([LM09, Theorem 2])**.** *For any polynomially bounded γ, there is a polynomial-time reduction from uSVP_γ to $\mathrm{BDD}_{1/\gamma}$.*

In the forward reduction, we can use a uSVP_γ solver to solve the $\mathrm{BDD}_{1/(2\gamma)}$ problem. However, in the backward reduction, to solve uSVP_γ problem, a $\mathrm{BDD}_{1/\gamma}$ solver is needed instead of $\mathrm{BDD}_{1/(2\gamma)}$ solver. Thus if these two problems are computationally equivalent to each other under some parameters, then there is still a factor 2 to be improved. In Chapter 3, we will discuss in detail how we decrease this non-tightness gap to $\sqrt{2}$. The currently best known algorithms for solving SVP and CVP take exponential time in the dimension n of the lattice. In the following, we introduce an efficient partition algorithm that, given a lattice basis (respectively, a basis and a target vector), outputs a polynomial set, which contains an estimation to the first minimum of the lattice (respectively, distance from the target vector to the lattice) within a small factor. We stress that this efficient partition algorithm does not solve any of GapSVP_γ and GapCVP_γ for any $\gamma \geq 1$ and dimension n. We only know that an precise estimation exists in a polynomially large set, but without knowing which one. This efficient

partition algorithm will be used in the context of some specific reductions between lattice problems. Here, for example, we give a brief description of the partition algorithm for outputting a set containing a precise estimation of first minimum. Given a lattice associated with some basis, we can first efficiently reduce the basis, to some extent (as will be discussed in Section 2.5.2). With the reduced basis, we can find a short vector with norm within an exponential factor of the first minimum. Then we can divide the exponential large range into polynomially many sufficiently small intervals, such that one of these intervals gives a precise estimate within small (inverse polynomial) factor. More details are provided below.

Given as input an n-dimensional lattice basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$, it is possible to find a non-zero vector that has norm at most $2^{n/2} \cdot \lambda_1(\Lambda(\mathbf{B}))$ in time polynomial in n and also the bit-sizes of the entries of $\mathbf{B}$, by using the LLL algorithm [LLL82]. Further, by using the Babai round-off algorithm [Bab86] with inputs an n-dimensional lattice basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$ and a target vector $\mathbf{t} \in \mathbb{Q}^n$, one obtains an estimation of the distance between $\mathbf{t}$ and $\Lambda(\mathbf{B})$ within a factor $2^{n/2}$ in time polynomial in n and also the bit-sizes of the entries of $\mathbf{B}$ and $\mathbf{t}$.

Lemma 2.15 ([LLL82, Proposition 1.6])**.** *There exists a polynomial-time algorithm that, given as input an n-dimensional lattice basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$, outputs $\ell \in \lambda_1(\Lambda) \cdot [1, 2^{n/2})$.*

This is actually discussed further later in this chapter.

Lemma 2.16 ([Bab86, Theorem 3.1])**.** *There exists a polynomial-time algorithm that, given as input an n-dimensional lattice basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$ and a target vector $\mathbf{t} \in \mathbb{Q}^n$, outputs $d \in \mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \cdot [1, 2^{n/2})$.*

According to Lemma 2.15 (resp. Lemma 2.16), we can obtain an approximation to first minimum (resp. distance from any target vector to lattice) within an exponential factor. For example, we can obtain $\ell \in \lambda_1(\Lambda(\mathbf{B})) \cdot [1, 2^{n/2})$ (resp. $d \in \mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \cdot [1, 2^{n/2})$). Then we can partition the interval $[1, 2^{n/2})$ into polynomially many intervals of the form $x \cdot [1, 1+\epsilon)$ for well-chosen rational x's (see Algorithms 1-2).

Algorithm 1 Partition algorithm for exponentially approximated first minimum

Input: An integer $\widetilde{\ell} \in \lambda_1(\Lambda(\mathbf{B})) \cdot [1, 2^{n/2})$ for some basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$.
Output: A set $\mathcal{S}$ containing an estimated value in $\lambda_1(\Lambda(\mathbf{B})) \cdot [1, 1+\epsilon)$, of size $\lceil \log_{1+\epsilon} 2 \cdot n/2 \rceil$.
1. Divide the interval $(\frac{\widetilde{\ell}}{2^{n/2}}, \widetilde{\ell}]$ into polynomially many intervals $(\frac{\widetilde{\ell}}{2^{n/2}(1+\frac{1}{\epsilon})^i}, \frac{\widetilde{\ell}}{2^{n/2}(1+\frac{1}{\epsilon})^{i+1}}]$ for $i \in [k]$, where k is the smallest integer satisfying $2^{n/2} \leq (1+\epsilon)^k$.
2. Output the set $\mathcal{S}$ of the upper bounds of all the k intervals above.

Algorithm 2 Partition algorithm for exponentially approximated distance

Input: An integer $\widetilde{d} \in \mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \cdot [1, 2^{n/2})$ for some target $\mathbf{t} \in \mathbb{Q}^n$ and basis $\mathbf{B} \in \mathbb{Q}^{n\times n}$.
Output: A set $\mathcal{S}$ containing an estimated value in $\mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \cdot [1, 1+\epsilon)$, of size $\lceil \log_{1+\epsilon} 2 \cdot n/2 \rceil$.
1. Divide the interval $(\frac{\widetilde{d}}{2^{n/2}}, \widetilde{d}]$ into polynomially many intervals $(\frac{\widetilde{d}}{2^{n/2}(1+\frac{1}{\epsilon})^{i-1}}, \frac{\widetilde{d}}{2^{n/2}(1+\frac{1}{\epsilon})^i}]$ for $i \in [k]$, where k is the smallest integer satisfying $2^{n/2} \leq (1+\epsilon)^k$.
2. Output the set $\mathcal{S}$ of the upper bounds of all the k intervals above.

In Chapter 3, we would consider the case $\epsilon = 1/(n-1)$ for partitioning the exponentially approximated distance from any target vector to lattice. In this case, the size of the set $\mathcal{S}$ is polynomially large in dimension n. Thus we can run the reduction (from BDD to uSVP) this polynomial time with each value in $\mathcal{S}$. As a result, we have that, in one of the reductions, the estimated distance $d_0 \in d \cdot [1, 1+\epsilon)$, where d is the exact distance from the target vector to the lattice.

2.3 Average-case lattice problems

In this section, we first give the definition of the LWE problem, followed by some important properties of the involved lattices if we view LWE as a BDD variant. These will be used to build the reduction from LWE to some presumably hard quantum problems in Chapter 4. For completeness, we also give a full description of Regev's encryption scheme [Reg09] as well as a brief proof for its correctness and security based on the LWE assumption. We refer the reader to [Reg09, Pei09, BLP+13] for more details on the hardness of LWE, and [ACF+15, APS15] for more details on the algorithm for solving LWE.

2.3.1 Definitions and properties

Now we introduce the LWE problem in two versions: search and decision.

Definition 2.18 (Search Learning with Error, Search LWE)**.** *The input of the search* $\mathrm{LWE}^m_{n,q,\chi}$ *with dimension* $n \geq 1$*, modulus* $q \geq 2$ *and distribution* χ *over* $\mathbb{Z}$*, consists of* m *samples of the form* $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$*, with* $\mathbf{a} \hookleftarrow \mathbb{Z}_q^n$*,* $b = \langle \mathbf{a}, \mathbf{s} \rangle + e$ *and* $e \hookleftarrow \chi$*, where* $\mathbf{s} \hookleftarrow \mathbb{Z}_q^n$ *is fixed.*

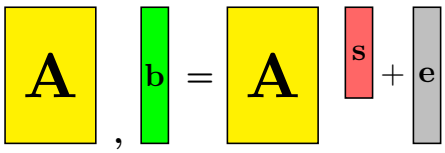

Fig. 2.5: An illustration of the LWE samples.

Definition 2.19 (Decision LWE)**.** *The decision* $\mathrm{LWE}^m_{n,q,\chi}$ *with dimension* $n \geq 1$*, modulus* $q \geq 2$ *and distribution* χ *over* $\mathbb{Z}$*, asks to distinguish between* $m \geq n$ *many* LWE *samples and random samples of the form* $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$*, with* $\mathbf{a} \hookleftarrow \mathbb{Z}_q^n$*,* $b \hookleftarrow \mathbb{Z}_q$*.*

We let $\mathrm{LWE}^m_{n,q,\alpha}$ (resp. $\mathrm{dLWE}^m_{n,q,\alpha}$) denote search (resp. decision) LWE problem with m samples of dimension n, modulus q and error distributed as $\mathcal{D}_{\mathbb{Z},\alpha q}$.

Regev proved that the LWE problem is at least as hard as some lattice problems over arbitrary lattices. In particular, arbitrary lattices contain those lattices for which the problems are hardest to solve.

Theorem 2.1 ([Reg09])**.** *For* $\alpha \in (0,1)$ *and* $q \geq 2$ *such that* $\alpha q > 2\sqrt{n}$*, there is a probabilistic quantum polynomial-time reduction from* $\mathrm{GapSVP}_{n,\mathcal{O}(n/\alpha)}$ *to* $\mathrm{LWE}^m_{n,q,\alpha}$*, in which* m *is polynomially bounded in* n*.*

This reduction is not a classical reduction, some quantum computation is also applied during the reduction. Later in [BLP+13], a dequantized version was proposed, as follows.

Theorem 2.2 (Adapted from [BLP+13, Theorem 2.16,Theorem 4.1,Corollary 3.2])**.** *For* $\alpha \in (0,1)$ *and* $q \geq 2$ *such that* $\alpha q \geq \sqrt{n}$*, there is a probabilistic polynomial-time reduction from* $\mathrm{GapSVP}_{\sqrt{n},\gamma}$ *to* $\mathrm{LWE}^m_{n,q,\alpha}$*, in which* m *is polynomially bounded in* n *and* $\gamma \geq \frac{\sqrt{n}}{\sqrt{10}\alpha \cdot \sqrt{\log n}}$*.*

As we can see, there is $\sqrt{n}$ factor loss in the dimension via the reduction. In particular, to solve the worst case problem GapSVP in dimension $\sqrt{n}$, via this reduction, we need a solver for solving LWE in dimension n. With our new results in Chapter 4, an alternative worst-to-average case reduction can also be established with dimension preserving, but with slightly worse loss compared to Regev's [Reg09] in approximation parameter. We refer the reader to Chapter 4 for more details.

The q-ary lattices is a specific family of lattices, which is of particular importance in lattice-based cryptography. They are defined as follows.

Definition 2.20 (q-ary lattices)**.** *An n-dimensional q-ary lattice is a lattice $\Lambda \subseteq \mathbb{Z}^n$ such that $q\mathbb{Z}^n \subseteq \Lambda \subseteq \mathbb{Z}^m$.*

Thus, in a q-ary lattice, it is sufficient to look at $\Lambda \bmod q$. Because all the shifts of $\Lambda \bmod q$ by $q\mathbb{Z}^n$ form a complete partition of Λ. For $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$, we define the q-ary lattice $\Lambda_q(\mathbf{A}) = \{\mathbf{A}\mathbf{x} \bmod q \colon \mathbf{x} \in \mathbb{Z}_q^n\}$. This specific q-ary lattice is closely related to the LWE problem. In particular, the LWE problem can be viewed as a BDD instance in this q-ary lattice $\Lambda_q(\mathbf{A})$, where $\mathbf{A}$ is the first component of LWE sample, and the second component serves as the target vector of the BDD instance. Once we find the closest vector $\mathbf{c} = \mathbf{A}\mathbf{s}$ to target vector $\mathbf{b}$, the secret vector $\mathbf{s}$ can be simply recovered by Gaussian elimination assuming the matrix $\mathbf{A}$ is full rank. For completeness, we recall that for a matrix $\mathbf{A}$ randomly chosen from $\mathbb{Z}_q^{m\times n}$, the matrix $\mathbf{A}$ is full rank for a sufficiently large m with overwhelming probability [BLP+13, Claim 2.13].

The following lemmata are well-known lower-bounds on the minimum of q-ary lattices.

Lemma 2.17. *Given a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ uniformly chosen for some positive integers m, n and prime q such that $m \geq n$, then we have $\lambda_1^\infty(\Lambda_q(\mathbf{A})) \geq q^{(m-n)/m}/2$ with probability $\geq 1 - 2^{-m}$.*

Proof. Suppose $B = q^{(m-n)/m}/2$, we let $\bar{\mathbf{a}}_i$ denote the i-th row vector of matrix $\mathbf{A}$, then we have

$$\begin{aligned}
\Pr_{\mathbf{A}\in\mathbb{Z}_q^{m\times n}}[\exists \mathbf{x} :\ \mathbf{A}\mathbf{x} = \mathbf{b} \wedge 0 < \|\mathbf{b}\|_\infty < B] &\leq \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \sum_{\mathbf{b}\in[0,B-1]^m} \Pr_{\mathbf{A}\in\mathbb{Z}_q^{m\times n}}[\mathbf{A}\mathbf{x} = \mathbf{b}] \\
&= \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \sum_{\substack{\mathbf{b}\in[0,B-1]^m \\ \mathbf{b}\neq\mathbf{0}}} \prod_{i\in[1,m]} \Pr_{\bar{\mathbf{a}}_i\in\mathbb{Z}_q^n}[\langle \bar{\mathbf{a}}_i, \mathbf{x}\rangle = b_i] \\
&= \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \sum_{\substack{\mathbf{b}\in[0,B-1]^m \\ \mathbf{b}\neq\mathbf{0}}} \frac{1}{q^m} \\
&= \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \left(\frac{B}{q}\right)^m \\
&= 2^{-m}.
\end{aligned}$$

□

Lemma 2.18. *Given a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ uniformly chosen for some positive integers m, n and prime q such that $m \geq n$, then we have $\lambda_1(\Lambda_q(\mathbf{A})) \geq \min\{q, \frac{\sqrt{m}q^{(m-n)/m}}{2\sqrt{2\pi e}}\}$ with probability $\geq 1 - 2^{-m}$.*

Proof. It suffices to prove $\lambda_1(\Lambda_q(\mathbf{A})) \geq B$ when $B = \sqrt{m}q^{(m-n)/m}/(2\sqrt{2\pi e}) < q$ (we use this assumption in Equation (2.2) below), as we have a trivial upper bound q on $\lambda_1(\Lambda_q(\mathbf{A}))$. In this case, we have

$$\begin{aligned}
\Pr_{\mathbf{A}\in\mathbb{Z}_q^{m\times n}}[\exists \mathbf{x} :\ \mathbf{A}\mathbf{x} = \mathbf{b} \wedge 0 < \|\mathbf{b}\| \leq B] &\leq \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \sum_{\substack{\mathbf{b}\in\mathbb{Z}^m \\ 0<\|\mathbf{b}\|\leq B}} \Pr_{\mathbf{A}\in\mathbb{Z}_q^{m\times n}}[\mathbf{A}\mathbf{x} = \mathbf{b}] \\
&= \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \sum_{\substack{\mathbf{b}\in\mathbb{Z}^m \\ 0<\|\mathbf{b}\|\leq B}} \prod_{i\in[1,m]} \Pr_{\mathbf{a}_i\in\mathbb{Z}_q^n}[\langle \mathbf{a}_i, \mathbf{x}\rangle = b_i] \\
&\leq \sum_{\mathbf{x}\in\mathbb{Z}_q^n} \frac{\frac{1}{\sqrt{\pi m}} \cdot \left(\frac{2\pi e}{m}\right)^{\frac{m}{2}} \cdot B^m}{q^m} \qquad (2.2) \\
&\leq 2^{-m}.
\end{aligned}$$

□

2.3.2 Regev's cryptosystem

In the following, we review Regev's public-key cryptosystem. Before this, we should recall the Leftover Hash Lemma (LHL) [HILL99] for a particular case. In this case, for $m, n, q > 1$, the function $f_{\mathbf{A}}(\mathbf{r}) = \mathbf{r}^{\mathrm{T}}\mathbf{A}$ is treated as a universal hash for $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ and $\mathbf{r} \in \{0,1\}^m$.

Definition 2.21 (Universal hash)**.** *A family of function $H = \{h : \mathcal{S}_1 \to \mathcal{S}_2\}$ is called a universal hash family if for any $x, y \in \mathcal{S}_1$ uniformly chosen such that $x \neq y$, we have*

$$\Pr_{h\hookleftarrow H}[h(x) \neq h(y)] \leq \frac{1}{\#\mathcal{S}_2}.$$

We also recall the statistical distance between two probability distributions P and Q over some set $\mathcal{S}$ as follows.

$$\Delta(P, Q) = \frac{1}{2}\sum_{x\in\mathcal{S}} |P(x) - Q(x)|.$$

A small statistical distance between two probability distributions implies that these two distributions are indistinguishable.

Lemma 2.19. *Let $m, n, q \geq 1$ be integers such that $m \geq 4n \log q$ and q prime, and let $\mathbf{A} \hookleftarrow U(\mathbb{Z}_q^{m\times n})$ and $\mathbf{r} \hookleftarrow U(\{0,1\}^m)$. Then the distribution of the pair $(\mathbf{A}, \mathbf{r}^{\mathrm{T}}\mathbf{A})$ is within statistical distance $\leq 2^{-n}$ from the uniform distribution on $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^n$.*

In 2005, Regev [Reg05, Reg09] proposed an elegant public-key cryptosystem, whose security can be directly based on the presumably hardness of the LWE problem. For the sake of completeness, we first recall the definition of public-key encryption (PKE) and the notion of security against chosen-plaintext attacks.

Definition 2.22 (Public Key Encryption, PKE)**.** *Given λ a security parameter, a public key encryption scheme is a tuple of three probabilistic polynomial-time algorithms* (**KeyGen**, **Enc**, **Dec**) *defined as follows.*

- **KeyGen**$(1^\lambda) \mapsto (PK, SK)$. *The key generation algorithm, with input security parameter λ, generates the public key and its associated secret key.*
- **Enc**$(PK, M) \mapsto C$. *The encryption algorithm, with input a public key PK and a message $M \in \{0,1\}$, outputs a ciphertext C.*
- **Dec**$(SK, C) \mapsto \{M, \perp\}$. *The decryption algorithm, with input a secret key SK and a ciphertext C, outputs either the message M or the symbol $\perp$ if the decryption fails.*

$\mathtt{Exp}^{\text{IND-CPA}}(\mathcal{A}, b, \lambda)$:
$(PK, SK) \leftarrow \mathbf{KeyGen}(1^\lambda)$
$(M_0, M_1) \leftarrow \mathcal{A}(PK)$
$C^* \leftarrow \mathbf{Enc}(PK, M_b)$
$b' \leftarrow \mathcal{A}(C^*)$
Return

Fig. 2.6: Experiment for IND-CPA security.

The first requirement on PKE is correctness, which means that the message should be properly recovered from any well-formed ciphertext. Concretely, for any $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathbf{KeyGen}(1^\lambda)$ with λ large

enough, and any message $M \in \{0,1\}$, we have that $\mathbf{Dec}(\mathrm{SK}, \mathbf{Enc}(\mathrm{PK}, M)) = M$ holds with overwhelming probability over the randomness used in the encryption algorithm **Enc**. Next, we introduce the notion of security against chosen-plaintext attacks.

Definition 2.23 (Indistinguishability against Chosen-Plaintext Attack, IND-CPA)**.** *For a PKE scheme (**KeyGen**, **Enc**, **Dec**), the security against chosen-plaintext attack with adversary $\mathcal{A}$ is defined as the experiment in Figure 2.6. The advantage of $\mathcal{A}$ is*

$$\mathrm{Adv}^{\mathrm{IND-CPA}}(\mathcal{A}, \lambda) = \left| \Pr_{b \hookleftarrow \{0,1\}} \left[\mathtt{Exp}^{\mathrm{IND-CPA}}(\mathcal{A}, \lambda, b) = b \right] - \frac{1}{2} \right|.$$

The PKE scheme is secure if this advantage is negligible in the security parameter λ for any probabilistic polynomial-time adversary.

Algorithm 3 Regev's public key encryption scheme

Let $n, m, q \geq 1$ be integers such that $m \geq 4n \log q$ and q prime, let $\alpha < 1/(4m)$ be a positive real number and let λ as the security parameter.

KeyGen(1^λ):

- Generate public parameters: (n, m, q, α)
- Sample $\mathbf{A} \hookleftarrow U(\mathbb{Z}_q^{m \times n})$, $\mathbf{s} \hookleftarrow U(\mathbb{Z}_q^n)$, and $\mathbf{e} \hookleftarrow \mathcal{D}_{\mathbb{Z}^m, \alpha q}$
- Compute $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$
- Set $\mathrm{PK} := (\mathbf{A}, \mathbf{b})$, $\mathrm{SK} := \mathbf{s}$.

Enc$(\mathrm{PK}, M \in \{0,1\})$:

- Sample $\mathbf{r} \hookleftarrow U(\{0,1\}^m)$
- Compute $C = (\mathbf{u}^{\mathrm{T}}, v) = (\mathbf{r}^{\mathrm{T}}\mathbf{A}, \mathbf{r}^{\mathrm{T}}\mathbf{b} + \lfloor q/2 \rfloor \cdot \mu) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

Dec$(\mathrm{SK}, C = (\mathbf{u}^{\mathrm{T}}, v))$:

- Compute $z = v - \mathbf{u}^{\mathrm{T}}\mathbf{s} \bmod q$
- Output 0 if z is closer to 0 than to $\lfloor q/2 \rfloor$, otherwise output 1.

Correctness of the scheme. Here, we show that the honestly encrypted message can be decrypted correctly. In the encryption phase, one first selects a random subset of rows of matrix $(\mathbf{A}|\mathbf{b})$ (in the public key) and computes their summation. The first n components of the resulted vector (as vector $\mathbf{u}$) is kept unchanged, and the message M is hidden by adding to the last component (thus obtaining the component v). In the decryption phase, with input secret key $\mathbf{s}$, one computes $z = v - \mathbf{u}^{\mathrm{T}}\mathbf{s} = \mathbf{r}^{\mathrm{T}}\mathbf{e} + \lfloor q/2 \rfloor \cdot \mu$. To make sure that the message can be perfectly decrypted, it suffices to show that $\|\mathbf{r}^{\mathrm{T}}\mathbf{e}\| \leq q/4$. Thanks to the choices of random vector $\mathbf{r}$ and noise vector $\mathbf{e}$, we have $\|\mathbf{e}\| \leq \sqrt{m}\alpha q$ with probability exponentially close to 1 according to Lemma 2.7, and $\|\mathbf{r}\| \leq \sqrt{m}$. Thus we have $\|\mathbf{r}^{\mathrm{T}}\mathbf{e}\| \leq \sqrt{m}\alpha q \cdot \sqrt{m} \leq q/4$, as required.

Security of the scheme. Regev's scheme is IND-CPA secure under the decision LWE assumption. We refer the reader to [Reg09] for a full proof. The principle idea is to give a reduction from the security of the scheme to the hardness of decision LWE. Concretely, one needs to prove that if there exists an adversary who can distinguish the ciphertext of 0 and 1 in polynomial-time with non-negligible probability, then the adversary can be used to solve the decision LWE problem in polynomial-time with non-negligible probability.

Here, we give a brief idea of the proof. First, we assume that there exists a probabilistic polynomial-time adversary that given public key and ciphertext of message M, has non-negligible advantage in distinguishing the message M between 0 and 1. Now, we change the LWE samples $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e})$ in the public key to be a uniform pair from $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$. Then the pair $(\mathbf{A}|\mathbf{b}, \mathbf{r}^{\mathrm{T}}(\mathbf{A}|\mathbf{b}))$ is statistically

indistinguishable from a uniform pair from $\mathbb{Z}_q^{m\times(n+1)} \times \mathbb{Z}_q^{n+1}$, due to the leftover hash lemma (see Lemma 2.19). In this case, any message M (either 0 or 1) added to the last component $\mathbf{r}^{\mathrm{T}}\mathbf{b}$ will be statistically indistinguishable for any adversary. Recall that the only difference between the starting case and this case is that the public key is LWE sample and uniform sample. As a result, the adversary can be used to distinguish the LWE samples and uniform samples in polynomial-time with non-negligible probability, thus invalidate the decision LWE assumption. In the concrete reduction, given a pair $(\mathbf{A}, \mathbf{b})$ as an instance of the decision LWE problem, we are asked to distinguish it between LWE samples and uniform samples. We use it as public key to encrypt a message $M \hookleftarrow \{0,1\}$, and send it to the adversary $\mathcal{A}$. If the adversary $\mathcal{A}$ guesses correctly what M is, then we say the instance is formed of the LWE samples, otherwise we say it is formed of uniform samples. The larger advantage the adversary has in distinguishing the message, the larger advantage we have to solve the decision LWE problem.

2.4 Quantum computations and the dihedral coset problem

In this section, we recall some necessary notations and concepts of quantum computations, for our study on quantum hardness of lattice-based cryptography. We also introduce the quantum versions of the Gaussian distribution and the Fourier transform, which are core tools used in Chapter 4. Finally, we introduce the dihedral coset problem and recall known results about it. We refer the reader to [NC00] for more details on quantum computations, and [Reg02, Kup05, Reg04c, Kup13] for more details on the hardness of the dihedral coset problem.

2.4.1 Definitions and tools

In quantum computations, data is stored as quantum bits (called *qubits*) in quantum registers. In general, a state of n qubits is written as

$$|\phi\rangle = \sum_{\mathbf{x}\in\{0,1\}^n} a_{\mathbf{x}} |\mathbf{x}\rangle \tag{2.3}$$

where $a_{\mathbf{x}} \in \mathbb{C}$ such that $\sum_{\mathbf{x}\in\{0,1\}^n} |a_{\mathbf{x}}|^2 = 1$. The same state $|\phi\rangle$ can be represented differently, e.g., as follows:

$$|\phi\rangle = \sum_{\mathbf{x}'\in\mathcal{S}} a'_{\mathbf{x}'} |\mathbf{x}'\rangle ,$$

for any finite set $\mathcal{S}$ that can be mapped (by some function g) to a set of independent vectors in the Hermitian space $\mathbb{C}^{\#\mathcal{S}}$. We call the set $\{|g(\mathbf{x}')\rangle\}_{\mathbf{x}'\in\mathcal{S}}$ a basis of the state. For example, the basis of the state $\phi_1 = (1/\sqrt{2})(|0\rangle + |1\rangle)$ can be $(\mathbf{b}_1, \mathbf{b}_2)$, where $\mathbf{b}_1 = (1,0)^{\mathrm{T}}$ and $\mathbf{b}_1 = (0,1)^{\mathrm{T}}$. Typically, we let $|x\rangle |y\rangle$ (or $|x, y\rangle$) denote the tensor product $|x\rangle \otimes |y\rangle$ of the two states $|x\rangle$ and $|y\rangle$. To measure the difference between two quantum states, we recall the trace distance.

Definition 2.24 (Trace distance)**.** *Given two quantum states of n qubits $|\phi_1\rangle$ and $|\phi_2\rangle$, their trace distance is defined as*

$$D(|\phi_1\rangle , |\phi_2\rangle) = \frac{1}{2}\mathrm{Tr}\left(\sqrt{(|\phi_1\rangle - |\phi_2\rangle)^\dagger(|\phi_1\rangle - |\phi_2\rangle)}\right).$$

The trace distance is known to be the maximum distinguishing advantage between two states using quantum measurements. We have

$$D(|\phi_1\rangle , |\phi_2\rangle) = \sqrt{1 - |\langle\phi_1|\phi_2\rangle|^2} \leq \| |\phi_1\rangle - |\phi_2\rangle \|,$$

where the first equality is given in [NC00, Chapter 8] and $\| |\phi_1\rangle - |\phi_2\rangle \|$ is called the ℓ_2-distance, and

is defined as follows.

Definition 2.25 (ℓ_2-distance between two quantum states)**.** *Given two quantum states of n qubits* $|\phi_1\rangle = \sum_{\mathbf{x}\in\mathcal{S}} a_{\mathbf{x}} |\mathbf{x}\rangle$ *and* $|\phi_2\rangle = \sum_{\mathbf{x}\in\mathcal{S}} b_{\mathbf{x}} |\mathbf{x}\rangle$*, their* ℓ_2*-distance is defined as*

$$\| |\phi_1\rangle - |\phi_2\rangle \| = \sqrt{\sum_{\mathbf{x}\in\mathcal{S}} |a_{\mathbf{x}} - b_{\mathbf{x}}|^2}.$$

For simplicity, in this thesis, we always compute ℓ_2-distance for evaluating the difference of two states, as small ℓ_2-distance implies small trace distance.

In quantum computations, the allowed operations consist of two forms: unitary operations and measurements. For the first case, the unitary operations can be represented by unitary operators $\mathbf{U} \in \mathbb{C}^{\#\mathcal{S}\times\#\mathcal{S}}$ satisfying $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}$. In particular, a unitary operator for a single *qubit* is a unitary matrix $\mathbf{U} \in \mathbb{C}^{2\times 2}$. For example, suppose we apply a unitary operation $\mathbf{U} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ on a single *qubit* state $\phi_1 = (1/2)|0\rangle + (\sqrt{3}/2)|1\rangle$, the resulting state will be $\phi_2 = (\sqrt{3}/2)|0\rangle + (1/2)|1\rangle$. If we view the input and output states as vectors $\mathbf{a}_1 = (1/2, \sqrt{3}/2)$ and $\mathbf{a}_2 = (\sqrt{3}/2, 1/2)$ respectively, then we have $\mathbf{a}_2 = \mathbf{U}\mathbf{a}_1$.

Analogously to the classical computations that are done with an electrical circuit containing wires and logic gates, the quantum computations are carried out with a quantum circuit containing wires and elementary quantum gates. In classical computations, we have an important theoretical result that any function on bits can be computed from the composition of (two-bits input) NAND gate: $(x, y) \mapsto \neg(a \wedge b)$, for any two bits $a, b \in \{0, 1\}$. Sometimes, we also call a set of gates that can implement the circuit for computing any functions on bits as a *universal* set of gates. In quantum computations, we have a similar result that any unitary operation can be approximated to arbitrary accuracy using the following four elementary gates (represented with unitary operators, see [NC00, Section 4.5]):

1. (1-*qubit* input) Hadamard gate: $\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$;
2. (1-*qubit* input) phase shift gate: $\begin{pmatrix} 1 & 0 \\ 1 & e^{i\phi} \end{pmatrix}$;
3. (1-*qubit* input) $\pi/8$ gate: $\begin{pmatrix} 1 & 0 \\ 1 & e^{i\pi/4} \end{pmatrix}$;
4. (2-*qubits* input) CNOT gate: $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$.

We assume that all of these gates can be evaluated in constant time. In this thesis, we focus on this specific *universal* set of gates. But we note that there are many other *universal* sets [BDEJ95]. A unitary operator $\mathbf{U}$ is considered to be efficient, if it can be built upon polynomially many (with respect to the size of the inputs) gates from the set of *universal* gates given above.

In classical computations, we are also allowed to apply an irreversible gates to data: $x \mapsto f(x)$ for a possibly non-injective function f. It is also well-known that all the classical circuits can be transformed

into corresponding quantum circuits with similar functionalities with the following map [Ben73] (also refer to Figure 2.7):

$$\sum_{x\in\mathcal{S}} a_x \left|x, y\right\rangle \mapsto \sum_{x\in\mathcal{S}} a_x \left|x, y \oplus f(x)\right\rangle .$$

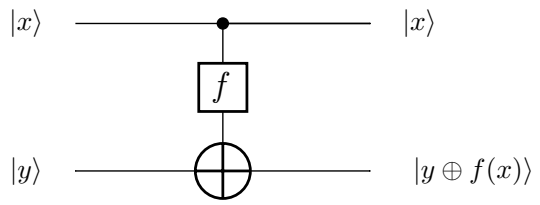

Fig. 2.7: Quantum circuit for evaluating a function f with input $|x\rangle$.

The second operation is measurement. We first define a function $\widehat{g}$ that maps $\mathbf{x} \in \{0,1\}^n$ to basis vector $\widehat{\mathbf{b}}_i \in \{0,1\}^{2^n}$, where $\widehat{\mathbf{b}}_i$ is the i-th column of the identity matrix $\mathbf{I}_{2^n}$ with $i = \sum_{j\leq n} x_j \cdot 2^{n-j}$. Then we can transform the basis of State 2.3 from $\{0,1\}^n$ to $\{\widehat{\mathbf{b}}\}_{i\leq 2^n}$ with $\widehat{b}_i$ defined as above. A measurement M on State 2.3 under the basis $\{\mathbf{b}_i\}_{i\leq 2^n}$ can be represented by the set $\{M_i = |\mathbf{b}_i\rangle \langle \mathbf{b}_i|\}_{i\leq 2^n}$, where $|\mathbf{x}_1\rangle \langle \mathbf{x}_2|$ denotes the *outer* product of $|\mathbf{x}_1\rangle$ and $|\mathbf{x}_2\rangle$. Each of the possibilities $\mathbf{x}$ is observed with probability $\langle \mathbf{b}_i| M_i^\dagger M_i |\mathbf{b}_i\rangle$ with $i = \sum_{j\leq n} x_j 2^{n-j}$, which, in this case, is $|a_{\mathbf{x}}|^2$, where $\langle \mathbf{x}_1|\mathbf{x}_2\rangle$ denotes the *inner* product of $|\mathbf{x}_1\rangle$ and $|\mathbf{x}_2\rangle$. Given a general measurement M under a basis (that is not necessary orthogonal), the set $\{E_i = M_i^\dagger M_i\}$ is known as the positive operator valued measures (POVM) (see [NC00, Chapter 2] for more details). We are also allowed to make a partial measurement. Suppose State 2.3 can be rewritten as $|\phi_1\rangle \otimes |\phi_2\rangle$ (with basis of $|\phi_1\rangle$ orthogonal to the one of $|\phi_2\rangle$), then a partial measurement means a measurement independently on $|\phi_1\rangle$ or $|\phi_2\rangle$ in its corresponding basis.

Now, we can recall the first tool that we need: the quantum Fourier transform. Here we consider a special case of it, which acts on elements in $\mathbb{Z}_q^n$.

Definition 2.26 (Quantum Fourier Transform)**.** *Let $\mathbb{G} = \mathbb{Z}_q^n$ be the group of integer vectors modulo q in dimension n with addition operation. The quantum Fourier transform (QFT) is a unitary operation $\mathcal{F}_q^n$ defined as follows:*

$$\mathcal{F}_q^n : |\mathbf{x}\rangle \mapsto \frac{1}{q^{n/2}} \sum_{\mathbf{y}\in\mathbb{Z}_q^n} \omega_q^{\langle \mathbf{x},\mathbf{y}\rangle} |\mathbf{y}\rangle ,$$

where $\omega_q = e^{\frac{2\pi i}{q}}$.

Note that the evaluation time of quantum Fourier transform is $\mathcal{O}(\log^2(|\mathbb{G}|))$ [NC00, Section 5.1]. In the special case above, the complexity of quantum Fourier transform is $\mathcal{O}(n^2 \log^2 q)$.

We now introduce the second tool that we need: the quantum analogue of the Gaussian distribution. In quantum computations, we can also build a quantum register with amplitude of each possibility distributed as a Gaussian distribution over each possible value. The following lemma is originally due to Grover and Rudolph [GR02] and was adapted to the Gaussian distribution in [Reg09].

Lemma 2.20 (Adapted from [Reg09, Lemma 3.12])**.** *Given a parameter κ and an integer r, there exists an efficient quantum algorithm that outputs a state that is within ℓ_2-distance $2^{-\Omega(\kappa)}$ of the normalized state corresponding to*

$$\sum_{x\in\mathbb{Z}} \rho_r(x) |x\rangle .$$

Proof. Given as input a real number r, we use the technique introduced by Grover and Rudolph [GR02]

to create the normalized state corresponding to:

$$\sum_{x\in\mathbb{Z}\cap[-\sqrt{\kappa}r,\sqrt{\kappa}r]} \rho_r(x)\,|x\rangle\,.$$

According to Lemma 2.7, the above state is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from the normalized state corresponding to:

$$\sum_{x\in\mathbb{Z}} \rho_r(x)\,|x\rangle\,.$$

□

The last tool we need to recall is the quantum rejection sampling due to Ozols *et al.* [ORR13].

Lemma 2.21 ([ORR13, Section 4])**.** *There is a quantum circuit, which given as input*

$$\sum_{k=1}^{n} \pi_k\,|k\rangle\,|\eta_k\rangle\,,$$

for some probability π_k, outputs

$$\frac{1}{\|\mathbf{p}\|}\sum_{k=1}^{n} p_k\,|k\rangle\,|\eta_k\rangle\,.$$

for some $p_k \le \pi_k$, with probability $\|\mathbf{p}\|^2 = \sum_{k=1}^{n} p_k^2$ in time $\mathrm{poly}(\log n)$.

2.4.2 Dihedral coset problem

The first connection between lattice problems and quantum computations was given by Regev [Reg02]. Concretely, Regev gives a reduction from uSVP to DCP, which has been conjectured quantum hard over the last two decades (see, e.g., [GSVV01, Reg02, FIM$^+$03, HRTS00, RB98]).

Definition 2.27 (Dihedral coset problem, DCP)**.** *The input of the* DCP_N^ℓ *with modulus N consists of ℓ states. Each state is of the form (normalization is omitted)*

$$|0\rangle\,|x\rangle \;+\; |1\rangle\,|(x+s) \bmod N\rangle\,,$$

stored on $1+\lceil\log_2 N\rceil$ qubits, where $x\in\mathbb{Z}_N$ is arbitrary and $s\in\mathbb{Z}_N$ is fixed throughout all the states. The task is to output the secret s.

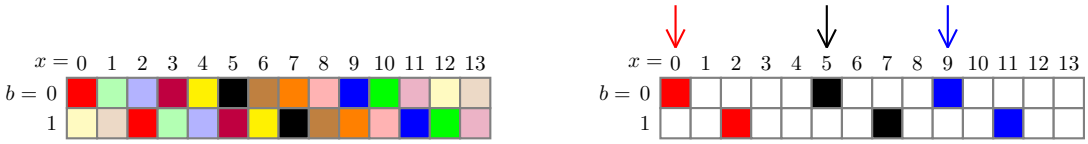

Fig. 2.8: An illustration of DCP samples with modulus $N = 14$.

For an intuitive illustration of the input samples of DCP, we refer the reader to Figure 2.8. Each pair of boxes with same color represents the pair of two possibilities in an input quantum superposition (refer to the left-hand side). In particular, on the right-hand side, the three samples are (up to normalization) $|0\rangle\,|0\rangle + |1\rangle\,|2\rangle$ (colored by red), $|0\rangle\,|5\rangle + |1\rangle\,|7\rangle$ (colored by black) and $|0\rangle\,|9\rangle + |1\rangle\,|11\rangle$ (colored by blue). In this example, one is asked to find out the difference between the values in the second register, which is the secret $s = 2$.

Note that Regev in [Reg02] defines the Dihedral Coset problem slightly differently. Namely, he introduces a failure parameter $f(\kappa)$, and with probability $\leq 1/(\log N(\kappa)^{f(\kappa)})$, we have a state of the form $|b\rangle|x\rangle$ for arbitrary $b \in \{0,1\}^n$ and $x \in \mathbb{Z}_N$. Such a state does not contain any information on s. Our definition takes 0 for the failure parameter. Conversely, Definition 2.27 is Regev's definition is with a bounded number of input states and failure parameter 0. It is also noticed in [EH99] that in an information theoretic sense, it suffices to use $\ell = \mathcal{O}(\log N)$ many samples for uniquely determining the secret s.

There is a closely related classic problem named hidden shift problem (HSP), which can be reduced to DCP. HSP is defined as follows.

Definition 2.28 (hidden shift problem, HSP)**.** *The input of* HSP_N *with modulus* N *consists of oracle access to a function* f *satisfying*

$$f(0,x) = f(1,x+s)$$

for any $x \in \mathbb{Z}_N$ *and a fixed arbitrary secret* $s \in \mathbb{Z}_N$*. The task is to output the secret* s*.*

Here we give a simple algorithm with oracle access to a HSP function f, outputting DCP states. We first prepare a quantum superposition over two registers with same amplitude over all pair (b,x) of $b \in \{0,1\}$ and $x \in \mathbb{Z}_N$. Such a state can be created efficiently using a technique by Grover and Rudolph [GR02], if the underlying distribution is integrable. This is true in this case, as we can integrate uniform distribution over any subset of $\{0,1\} \times \mathbb{Z}_N$ as $\sum_{b=b_1}^{b_2} \sum_{x=x_1}^{x_2} 1/(2N)$ for some $b_1 \leq b_2$ from $\{0,1\}$ and $x_1 \leq x_2$ from $\mathbb{Z}_N$ within good precision. Then we apply the function f on this pair and store the output in the third register. We obtain (normalization is omitted):

$$\sum_{b\in\{0,1\}} \sum_{x\in\mathbb{Z}_N} |b, x, f(b,x)\rangle .$$

Then a measurement on the third register gives us (normalization is omitted):

$$|0\rangle|x\rangle \ + \ |1\rangle|(x+s) \bmod N\rangle$$

for some uniform $x \in \mathbb{Z}_N$. This gives us a valid input state for DCP.

The best algorithm known so far for solving DCP with polynomially many (with respect to $\log N$) input states is full exponential in $\log N$ (e.g., via reducing to subset sum problem [Reg02]). However, there is a trade-off between number of input states and solving time for DCP. In [Kup05], a sub-exponential time algorithm is given to solve DCP with sub-exponentially many input states. This, with the above discussion, also leads to a sub-exponential time solving algorithm for HSP.

For completeness, we briefly review the main idea of Kuperberg's algorithm for solving DCP. In particular, the first step of the algorithm is also used in our work in Chapter 4.

First, a quantum Fourier transform is applied on the second register of the DCP sample, which gives (normalization is omitted):

$$(|0\rangle + \omega_N^{xs}|1\rangle)|x\rangle . \tag{2.4}$$

Then we make a measurement on the second register, and obtain (normalization is omitted):

$$|0\rangle + \omega_N^{\widehat{x}s}|1\rangle ,$$

for some measured $\widehat{x} \in \{0, \cdots, N-1\}$. Our target is $|0\rangle + (-1)^s|1\rangle$. For simplicity, we always let n denote the bit length of the modulus N, and $k = \sqrt{n-1}$. Suppose we are given sub-exponentially (e.g., $\geq 2^k$) many state as in (2.4), then by pigeonhole principle, we have at least one pair of state $|0\rangle + \omega_N^{\widehat{x}_1 s}|1\rangle$ and $|0\rangle + \omega_N^{\widehat{x}_2 s}|1\rangle$ with $\widehat{x}_1$ and $\widehat{x}_2$ that agree on the least k bits. Then a tensor product

between them gives us

$$|0,0\rangle + \omega_N^{\widehat{x}_1 s} |1,0\rangle + \omega_N^{\widehat{x}_2 s} |0,1\rangle + \omega_N^{(\widehat{x}_1+\widehat{x}_2)s} |1,1\rangle .$$

Now we apply a CNOT gate (that can be viewed as a special case of circuit in Figure 2.7 with $f : x \to x$ the identity function) on them and measurement on the second register. With probability 1/2, we measure 'odd' and obtain (normalization is omitted):

$$|0\rangle + \omega^{(\widehat{x}_1-\widehat{x}_2)s} |1\rangle = |0\rangle + \omega^{\widehat{x}' s} |1\rangle .$$

In this case, we obtain a new state which we say in the second level, as $\widehat{x}'$ has 0's for the least k bits. Once we have enough states in the second level, we can continue to move to the next level, until we get the targetted state in the $(k+1)$-th level.

To conclude, with state $|0\rangle + (-1)^s |1\rangle$, we can apply the quantum Fourier transform (acting on group $\mathbb{Z}_2$), which gives us (up to normalization) $(1 + (-1)^s) |0\rangle + \big(1 + (-1)^{s+1}\big) |1\rangle$. Thus a measurement on this state tells us the least significant bit of s: if 0 is measured, then s is even, otherwise s is odd. Then we can iteratively recover all the bits of s from the least significant bit to the most significant bit. Now suppose we already got the least significant bit of s, denoted as s_1. Then we can repeat the above procedure except in the last level, our targetted state becomes

$$|0\rangle + \omega_4^s |1\rangle = |0\rangle + i^s |1\rangle .$$

When we obtain this state, we apply a phase shift gate with the following unitary operator

$$R_{-\frac{2\pi s_1}{4}} = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\frac{2\pi s_1}{4} i} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \omega_4^{s_1} \end{pmatrix},$$

and we obtain

$$|0\rangle + (-1)^{\lfloor s/2 \rfloor} |1\rangle ,$$

with which we can obtain the second least significant bit of s. With this strategy, we can iteratively recover the whole s.

Overall, this algorithm runs in time $2^{\mathcal{O}(\sqrt{\log N})}$. As we can observe, Kuperberg's algorithm also takes $2^{\mathcal{O}(\sqrt{\log N})}$ quantum space and $2^{\mathcal{O}(\sqrt{\log N})}$ samples. In [Reg04c], Regev gives an alternative for solving DCP with polynomial quantum space, with running time $2^{\mathcal{O}(\sqrt{\log N \log\log N})}$ and $2^{\mathcal{O}(\sqrt{\log N \log\log N})}$ samples. Later, Kuperberg [Kup13] shows that, with polynomial quantum space, it is enough to solve DCP in time $2^{\mathcal{O}(\sqrt{\log N})}$, with $2^{\mathcal{O}(\sqrt{\log N})}$ sample. Further, by considering a general number of samples ℓ in Kuperberg's improved algorithm [Kup13], there is an algorithm for solving DCP with ℓ (at least poly$(\log N)$) samples in time $2^{\mathcal{O}(\log \ell + \log N/\log \ell)}$.

For completeness, we give a brief description of the algorithm and a heuristic analysis on the complexity. The heuristic analysis can be made rigorous, we refer the reader to [Kup13] for more details.

The first step is exactly the same as the first step in the first Kuperberg's algorithm [Kup05] discussed above, suppose we have prepared ℓ-many (up to normalization):

$$|0\rangle + \omega_N^{xs} |1\rangle ,$$

for some classically known $x \hookleftarrow \{0, \cdots, N-1\}$. Our target is still $|0\rangle + (-1)^s |1\rangle$. The overall idea is similar to the algorithm above, from each level to the next level, more bits of the multiplicative factor of the secret in the exponent of the phase will become 0's. The difference is that in this algorithm, the number of levels is more flexible to be poly$(\log \ell)$ according to the number of input samples. In the

following, we let $\ell' = \ell/(n-1)$ and $\widehat{k} = (n-1)/k$, where k is the bit length of ℓ'. Then, we tensor $\widehat{k}$ states above $\{|0\rangle + \omega_N^{x_i s}|1\rangle\}_{i\leq 2n}$, tensor them together, and obtain (up to normalization):

$$\sum_{\mathbf{a}\in\{0,1\}^{\widehat{k}}} \omega_N^{\langle \mathbf{a},\mathbf{x}\rangle\cdot s}|\mathbf{a}\rangle = \sum_{\mathbf{a}\in\{0,1\}^{\widehat{k}}} \omega_N^{x_{\mathbf{a}}\cdot s}|\mathbf{a}\rangle,$$

where $x_{\mathbf{a}} = \langle \mathbf{a},\mathbf{x}\rangle$. Then we can apply a function f that maps $\{0,1\}^{\widehat{k}}$ to $\{0,\cdots,2^{\widehat{k}}-1\}$ to the register of state above, and obtain (up to normalization):

$$\sum_{\widehat{a}\in\{0,\cdots,2^{\widehat{k}}-1\}} \omega_N^{x_{\widehat{a}}\cdot s}|\widehat{a}\rangle$$

where we let $x_{\widehat{a}}$ denote $x_{\mathbf{a}}$ correspondingly. This will serve as the state in the first level.

Because each state above consume $\leq n$ input states, we have at least $\ell' = \ell/n$ many such states. Next, we tensor two states: $\sum_{\widehat{a}_1\in\{0,\cdots,2^{\widehat{k}}-1\}} \omega_N^{x_{\widehat{a}_1}\cdot s}|\widehat{a}_1\rangle$ and $\sum_{\widehat{a}_2\in\{0,\cdots,2^{\widehat{k}}-1\}} \omega_N^{x_{\widehat{a}_2}\cdot s}|\widehat{a}_2\rangle$ in the first level together, and obtain (up to normalization):

$$\sum_{\widehat{\mathbf{a}}\in\{0,\cdots,2^{\widehat{k}}-1\}^2} \omega_N^{\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}\rangle\cdot s}|\widehat{\mathbf{a}}\rangle,$$

where $\widehat{\mathbf{a}} = (\widehat{a}_1,\widehat{a}_2)$ and $\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}} = (x_{\widehat{a}_1}, x_{\widehat{a}_2})$.

As we know the $x_{\widehat{a}} \in \{0,\cdots,N-1\}$ for all $\widehat{a}\in\{0,\cdots,2^{\widehat{k}}-1\}$ classically, then we can take value $\widehat{\mathbf{a}}$ from the first register and compute $\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle \bmod 2^{\widehat{k}}$, store it in the second register, and obtain (up to normalization):

$$\sum_{\widehat{\mathbf{a}}\in\{0,\cdots,2^{\widehat{k}}-1\}^2} \omega_N^{\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle\cdot s}|\widehat{\mathbf{a}}\rangle \left|\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle \bmod 2^{\widehat{k}}\right\rangle.$$

And a measurement on the second register gives us (up to normalization):

$$\sum_{\substack{\widehat{\mathbf{a}}\in\{0,\cdots,2^{\widehat{k}}-1\}^2\\ \langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle\equiv b \bmod 2^{\widehat{k}}}} \omega_N^{\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle\cdot s}|\widehat{\mathbf{a}}\rangle|b\rangle.$$

for some measured value b. By omitting the global phase $\omega^{b\cdot s}$, we obtain (up to normalization):

$$\sum_{\substack{\widehat{\mathbf{a}}\in\{0,\cdots,2^{\widehat{k}}-1\}^2\\ \langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle\equiv b \bmod 2^{\widehat{k}}}} \omega_N^{(\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle - b)\cdot s}|\widehat{\mathbf{a}}\rangle|b\rangle = \sum_{\substack{\widehat{\mathbf{a}}\in\{0,\cdots,2^{\widehat{k}}-1\}^2\\ \langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle\equiv b \bmod 2^{\widehat{k}}}} \omega_N^{\bar{x}_{\widehat{\mathbf{a}}}\cdot s}|\widehat{\mathbf{a}}\rangle|b\rangle, \tag{2.5}$$

where $\bar{x}_{\widehat{\mathbf{a}}} = \langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}_{\widehat{\mathbf{a}}}\rangle - b$.

Then we compute all solutions $\widehat{\mathbf{a}}$ to the equation system $\langle \widehat{\mathbf{a}},\widehat{\mathbf{x}}\rangle \equiv b \bmod 2^{\widehat{k}}$ classically with cost $\mathcal{O}(2^{\widehat{k}})$ (that can be done in a brute-force way). We let $\mathcal{S}$ denote the set of all the solutions. Heuristically, we have $\#\mathcal{S} \approx 2^{2\widehat{k}}/2^{\widehat{k}} = 2^{\widehat{k}}$ many solutions for any b (see [Kup13] for a rigorous proof of a slightly modified variant). Without loss of generality, we assume $\#\mathcal{S} = 2^{\widehat{k}}$. As we know all the solution classically, we can apply a bijective map h that maps $\mathcal{S}$ to $\{0,\cdots,2^{\widehat{k}}-1\}$ to the first register of State (2.5) (omitting the second classically known register), and obtain (up to normalization):

$$\sum_{\bar{a}\in\{0,\cdots,2^{\widehat{k}}-1\}} \omega_N^{\bar{x}_{\bar{a}}\cdot s}|\bar{a}\rangle,$$

where $\bar{x}_{\bar{a}} = \bar{x}_{\widehat{\mathbf{a}}}$ for $h(\widehat{\mathbf{a}}) = \bar{a}$. This state will serve as the state in the second level. And we can observe

that $x_{\vec{a}}$ has 0's in the $\widehat{k} = n/k$ least significant bits.

We can repeat this process until we obtain the state $|0\rangle + (-1)^s |1\rangle$ in the $(k+1)$-th level. As for generating one states in the next level, we only need two states in current level. Thus to get a state in the $(k+1)$-th level, we only need 2^k states that is exactly equal to $\ell' = \ell/n$ states that are prepared in the first level. The remaining process for recovering the whole s is the same as the former algorithm. To conclude, the complexity of the algorithm is $2^{\mathcal{O}(\log \ell + \log N/\log \ell)}$, where $2^{\mathcal{O}}(\log \ell)$ is the time for reading in the input samples, and $2^{\mathcal{O}(\log N/\log \ell)}$ is the time for solving the equation systems classically in the algorithm.

There is an extension of HSP, which gives a trade-off for solving time. Instead of giving only queries values from $\{0,1\}$ for the first input of function f, the generalized HSP (GHSP) [CvD07] assumes oracle access to a more structured function f. It is defined as follows.

Definition 2.29 (generalized HSP, GHSP)**.** *The input of the* $\text{GHSP}_{M,N}$ *with parameters an integer* M *and modulus* N *consists of oracle access to a function* f *satisfying*

$$f(b,x) = f(b+1, x+s)$$

for any $b \in \{0, \cdots, M-1\}$ *and* $x \in \mathbb{Z}_N$ *and a fixed arbitrary secret* $s \in \mathbb{Z}_N$*. The task is to output the secret* s*.*

In [CvD07], Childs and van Dam use the "pretty good measurement" to solve $\text{GHSP}_{M,N}$ efficiently in polynomial-time in $\log N$ for any $M = N^\epsilon$ with constant ϵ. In other words, the proposed algorithm can efficiently solve GHSP when the query extension factor $M = 2^{\mathcal{O}(\log N)}$ is exponentially large. In Chapter 4, we show how to decrease the query extension factor to $2^{\mathcal{O}(\sqrt{\log N})}$, while maintaining an efficient solving algorithm.

2.5 Lattice reduction

In this section, we recall some well-known definitions of lattice reductions. Among them, the BKZ reduction is one of the most important tools for security analysis in lattice-based cryptography. This is because the corresponding BKZ algorithm (for achieving the BKZ reduction) by Schnorr and Euchner [SE94], is the best known algorithm for cryptanalysis of lattice-based cryptography in practice. We refer the reader to [NV09, MV10a, HPS11] for more details on lattice reduction and to [CN11, Che09, AWHT16, YD17] for more details on the analysis on the BKZ algorithm. This section is mainly linked to Chapter 5.

2.5.1 Size reduction

First, we define the size-reduction conditions. If we again view the GSO procedure as a QR decomposion, then the size-reduction conditions can be seen as a requirement on the upper triangular matrix.

Definition 2.30 (Size-reduction)**.** *A matrix* $\mathbf{B} \in \mathbb{R}^{m \times n}$ *is called size-reduced, if it satisfies:*

$$|\mu_{i,j}| \leq \frac{1}{2}, \ 1 \leq j < i \leq n,$$

where $\mu_{i,j} = \langle \mathbf{b}_i, \mathbf{b}_j^* \rangle / \langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle$.

If $\mathbf{b}_j^* = \mathbf{0}$, we let $\mu_{i,j} = 0$ for any $i \geq j$.

We also recall an algorithm for achieving the size reduction condition as shown in Algorithm 4.

The size-reduction condition should be understood more clearly if we consider the GSO (or QR-decomposition). Suppose that we start from any basis $\mathbf{B} = \mathbf{QR}$ with an orthogonal matrix $\mathbf{Q}$ and an

Algorithm 4 Size-reduction algorithm

Require: A basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$.
Ensure: A size-reduced basis of $\Lambda(\mathbf{B})$.
1: `Compute the GSO basis` $\mathbf{b}_1^*, \cdots, \mathbf{b}_n^*$
2: **for** $i = 2$ `to` n **do**
3: **for** $j = i - 1$ `down to 1` **do**
4: $\mathbf{b}_i \leftarrow \mathbf{b}_i - \lceil \mu_{i,j} \rfloor \mathbf{b}_j$
5: **for** $k = 1$ `to` j **do**
6: $\mu_{i,k} \leftarrow \mu_{i,k} - \lceil \mu_{i,j} \rfloor \mu_{j,k}$
7: **end for**
8: **end for**
9: **end for**

upper triangular matrix

$$\mathbf{R} = \begin{pmatrix} \ddots & \vdots & \cdots & \vdots & \cdots \\ & \|\mathbf{b}_i^*\| & \cdots & \mu_{ij}\|\mathbf{b}_i^*\| & \cdots \\ & & \ddots & \vdots & \cdots \\ & & & \|\mathbf{b}_j^*\| & \cdots \\ & & & & \ddots \end{pmatrix}.$$

In particular, to achieve the size-reduction condition for a specific μ_{ji} with $i < j$, it suffices to construct a unimodular matrix $\mathbf{U}_{ij}$ and apply it on $\mathbf{R}$ as follows

$$\mathbf{R} \cdot \mathbf{U}_{ij} = \begin{pmatrix} \ddots & \vdots & \cdots & \vdots & \cdots \\ & \|\mathbf{b}_i^*\| & \cdots & \mu_{ji}\|\mathbf{b}_i^*\| & \cdots \\ & & \ddots & \vdots & \cdots \\ & & & \|\mathbf{b}_j^*\| & \cdots \\ & & & & \ddots \end{pmatrix} \cdot \begin{pmatrix} \ddots & & & & \\ & 1 & & -\lfloor \mu_{ji} \rceil & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \ddots \end{pmatrix} = \begin{pmatrix} \ddots & \vdots & \cdots & \vdots & \cdots \\ & \|\mathbf{b}_i^*\| & \cdots & \widehat{\mu}_{ji}\|\mathbf{b}_i^*\| & \cdots \\ & & \ddots & \vdots & \cdots \\ & & & \|\mathbf{b}_i^*\| & \cdots \\ & & & & \ddots \end{pmatrix},$$

where we have $|\widehat{\mu}_{ji}| \leq 1/2$. This step is exactly corresponding to Line 4 of Algorithm 4. Notice that such an operation $\mathbf{R} \cdot \mathbf{U}_{ij}$ for some $i < j$ may also change the values μ_{ik} for $k < i$. Thus we can proceed to apply these specific unimodular matrices $\mathbf{U}_{ij}$ to $\mathbf{R}$ from bottom ($i = j$) to up ($i = 1$) (see Algorithm 4 for a full procedure).

Given a basis $\mathbf{B} = \mathbf{QR}$, the product $\prod_i \|\mathbf{b}_i^*\|$ of the diagonal components of $\mathbf{R}$ is fixed and equal to the determinant of the lattice associated with the basis $\mathbf{B}$. Intuitively, to obtain a good basis, we aim to make the $\|\mathbf{b}_i^*\|$'s have limited decrease.

2.5.2 LLL reduction

We now recall LLL-reduction, which can be reached in polynomial-time (see [LLL82]).

Definition 2.31 (LLL-reduction)**.** *For $\delta \in (1/4, 1)$, a matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$ is called LLL$_\delta$ reduced, if it is size-reduced and satisfies the Lovász condition:*

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

for $1 \leq i < n$.

For an LLL-reduced basis $\mathbf{B}$, the Lovász condition implies

$$\|\mathbf{b}_i^*\| \leq \alpha^{\frac{1}{2}} \|\mathbf{b}_{i+1}^*\|$$

for any $1 \leq i < n$, where $\alpha = (\delta - 1/4)^{-1}$. Thus, an upper bound on $\|\mathbf{b}_1^*\|$ (which is equal to $\|\mathbf{b}_1\|$) can be derived as:

$$\|\mathbf{b}_n^*\|^2 \geq \alpha \cdot \|\mathbf{b}_{n-1}^*\|^2 \geq \cdots \geq \alpha^{n-1} \|\mathbf{b}_1^*\|^2 = \alpha^{n-1} \|\mathbf{b}_1\|^2.$$

Further, we also have

$$\|\mathbf{b}_1^*\| \leq \alpha^{-(i-1)/2} \|\mathbf{b}_i^*\| \leq \alpha^{-(n-1)/2} \|\mathbf{b}_i^*\|, \tag{2.6}$$

for any $i > 1$. Thus, we have

$$\|\mathbf{b}_1\| \leq \alpha^{-(n-1)/2} \min_{i \in [n]} \|\mathbf{b}_i^*\| \leq \alpha^{-(n-1)/2} \cdot \lambda_1(\Lambda(\mathbf{B})).$$

As a result, the LLL-reduction helps to approximate the shortest vector by a factor $\left(\delta - \frac{1}{4}\right)^{-(n-1)/2}$. Take $\delta = 3/4$, we have $\alpha = 2$, thus this factor is $2^{(n-1)/2}$ (as in Lemma 2.15). On the other hand, by taking the product of Equation (2.6) for all $i \in [n]$, we have

$$\|\mathbf{b}_1\|^n \leq \prod_{i \in [n]} \alpha^{\frac{i-1}{2}} \|\mathbf{b}_i^*\| = \alpha^{\frac{n(n-1)}{4}} \det(\mathbf{B}).$$

Equivalently, we have

$$\|\mathbf{b}_1\| \leq \alpha^{(n-1)/4} \det(\mathbf{B})^{1/n}.$$

Thus the root Hermite factor of an LLL-reduced basis can be bounded by $2^{(n-1)/4}$ when we take $\delta = 3/4$.

Further, the LLL-reduction condition can be achieved efficiently in polynomial-time by the LLL-reduction algorithm proposed by Lenstra *et al* [LLL82].

Algorithm 5 LLL-reduction algorithm

Require: A basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$ and parameter δ.
Ensure: A δ-LLL-reduced basis of $\Lambda(\mathbf{B})$.

```
Size-reduce(B).
Swap:
if ∃i such that ‖μ_{i+1,i} b_i^* + b_{i+1}^*‖^2 < δ‖b_i^*‖ then
    swap(b_i, b_{i+1})
    goto Step 1
end if
```

More generally, whenever we want to call the LLL algorithm with input a generating set of lattice (instead of a basis), we always assume that the generating set is pre-processed by computing its Hermite normal form. Then we take the non-zero columns of the resulting matrix in Hermite normal form, as the basis vectors of the lattice, and call the LLL algorithm with this basis. As in Algorithm 5, the LLL-reduction conditions will be satisfied once the algorithm terminates. The following lemma states that Algorithm 5 terminates in time polynomial in $\max\{n, \max_{i \in [n]} \log \|\mathbf{b}_i\|\}$.

Lemma 2.22. *Given a basis $\mathbf{B} \in \mathbb{Z}^{n \times n}$ of an n-dimensional lattice Λ, Algorithm 5 terminates in time polynomial in $K = \max\{n, \max_{i \in [n]} \log \|\mathbf{b}_i\|\}$*

We refer the reader to [LLL82] for a full proof. The main idea is to associate to the basis a quantity (sometimes called 'potential'). It can be shown that this quantity is decreased by at least a constant factor after each `Swap` operation and unchanged through `Size-reduction`. Thus by showing the initial quantity is bounded properly, the number of iterations of `Swap` operation can also be bounded. The

running time of each iteration can be shown to be polynomial in $K = \max\{n, \log \|\mathbf{b}_i\|\}$ (see [LLL82]). In the following, we show that the number of iterations is polynomial in K. First, the 'potential' of a basis $\mathbf{B}$ of an n-dimensional lattice is

$$P_{\mathbf{B}} = \prod_{i=1}^{n} \|\mathbf{b}_i^*\|^{n-i+1} = \prod_{i=1}^{n} \|\mathbf{b}_1^*\| \cdots \|\mathbf{b}_i^*\| = \prod_{i=1}^{n} P_{\mathbf{B},i},$$

where $P_{\mathbf{B},i} = \det(\Lambda_i)$ with Λ_i is the lattice associated with the basis $(\mathbf{b}_1, \cdots, \mathbf{b}_i)$.

According to this definition, we observe that $P_{\mathbf{B}}$ can be bounded by $(\max_{i\in[n]} \|\mathbf{b}_i\|)^{n(n+1)/2}$. Notice that $P_{\mathbf{B}}$ does not change during the `Size-reduction` step, as all $\mathbf{b}_i^*$'s are kept unchanged. During the `Swap` step, suppose that $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$ are swapped for some i. Then the Gram–Schmidt vectors will only be changed at indices i and $i+1$. We let $\widehat{\mathbf{b}}_i^*$ and $\widehat{\mathbf{b}}_{i+1}^*$ denote the updated Gram–Schmidt vectors at these two places. A `Swap` takes place if the Lovász condition is not satisfied, thus we have

$$\|\widehat{\mathbf{b}}_i^*\| = \|\mu_{i+1,i}\mathbf{b}_i^* + \mathbf{b}_{i+1}^*\| < \sqrt{\delta}\|\mathbf{b}_i^*\|,$$

where the first equation holds because $\mathbf{b}_{i+1}^* = \widehat{\mathbf{b}}_i^* - \mu_{i+1,i}\mathbf{b}_i^*$.

As a result, the quantity $P_{\mathbf{B},i}$ decreases by at least a factor $\sqrt{\delta}$ (for a constant $\delta \in (1/4, 1)$). Note that $P_{\mathbf{B},j}$ for $j \neq i$ does not change. Thus, overall, the quantity $P_{\mathbf{B}}$ also decreases by a constant factor $\sqrt{\delta}$. First, since $\mathbf{B}$ is an integer basis, if $P_{\mathbf{B}}$ is positive, then we must have $P_{\mathbf{B}} = \prod_{i=1}^{n} \det(\Lambda(\mathbf{b}_1, \cdots, \mathbf{b}_i)) \geq 1$, where the second inequality follows from the definition of the determinant of the lattice and the fact that the basis $\mathbf{B}$ is integral. Consequently, as we also have $P_{\mathbf{B}} \leq \max_{i\in[n]} \|\mathbf{b}_i\|^{n(n+1)/2}$, the number of `Swap` operations made is at most $\mathcal{O}(n^2 \log \max_{i\in[n]} \|\mathbf{b}_i\|)$, which gives us the result.

Note that in the analysis of the run-time of the LLL algorithm, we require the input basis $\mathbf{B} \in \mathbb{Z}^{n\times n}$. Thus before any application of the LLL algorithm on basis $\mathbf{B}' \in \mathbb{Q}^{n\times n}$, we will first scale it to be an integer basis by multiplying some (properly chosen) integer K. Then we can run the LLL reduction on it and scale the LLL-reduced basis back by dividing K.

2.5.3 HKZ reduction

The Hermite-Korkine-Zolotarev (HKZ) reduction is more promising for approaching the successive minima, even though an HKZ-reduced basis does not necessarily reach them.

Definition 2.32 (HKZ-reduction)**.** *A matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$ is called HKZ-reduced, if it is size-reduced and satisfies:*

$$\|\mathbf{b}_i^*\| = \lambda_1(\Lambda_{[i,n]})$$

for all $i \in \{1, \cdots, n\}$.

Here, we recall a result on the relation between the norms of HKZ-reduced basis vectors and the successive minima of the lattice.

Lemma 2.23 ([LLS90, Theorem 2.1])**.** *Let $\mathbf{B} = (\mathbf{b}_1, \cdots, \mathbf{b}_n)$ be an HKZ-reduced basis of a lattice Λ. We have*

$$\frac{4}{i+3}\lambda_i(\Lambda)^2 \leq \|\mathbf{b}_i\|^2 \leq \frac{i+3}{4}\lambda_i(\Lambda)^2$$

for all $i \in [n]$.

For an n-dimensional HKZ-reduced basis, we can also quantify the relations among its Gram–Schmidt norms $\|\mathbf{b}_i^*\|$'s in the worst case as follows.

Lemma 2.24. *For any HKZ-reduced basis* $\mathbf{B} = \mathbf{QR}$ *of an* n*-dimensional lattice, we have, for all* $i < n$*,*

$$\|\mathbf{b}_i^*\| \leq \sqrt{\gamma_{n-i+1}} \cdot \left(\prod_{j=i}^{n} \|\mathbf{b}_j^*\| \right)^{\frac{1}{n-i+1}}. \tag{2.7}$$

This worst-case result is obtained by considering the Minkowski's first theorem. If we only consider the equalities in the inequalities above, when we fix $\|\mathbf{b}_n^*\|$, all the remaining $\|\mathbf{b}_i^*\|$'s are also fixed. Equation (2.7) can be rewritten as

$$x_i \leq \frac{1}{2} \log \gamma_{n-i+1} + \frac{1}{n-i+1} \sum_{j=i}^{n} x_j$$

for all $i < n$, where $x_i = \log \|\mathbf{b}_i^*\|$. The x_i's are also known as the worst-case HKZ profile.

To obtain an HKZ-reduced basis, one is required to find a shortest vector in each projected lattice, with dimension from 1 to n. As we have already discussed in Section 2.2, the current known best SVP solvers take time exponential in dimension n, which becomes very costly when n grows. In the following, we recall the HKZ-reduction algorithm.

Algorithm 6 HKZ-reduction algorithm

Require: A basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$.
Ensure: A HKZ-reduced basis of $\Lambda(\mathbf{B})$.
1: **for** $k = 1$ `to` $n - 1$ **do**
2: `Find any b such that` $\|\pi_k(\mathbf{b})\| = \lambda_1(\Lambda_{[k,n]})$
3: $\mathbf{B} \leftarrow$ `LLL-reduce`$(\mathbf{b}_1, \cdots, \mathbf{b}_{k-1}, \mathbf{b}, \mathbf{b}_k, \cdots, \mathbf{b}_n)$
4: `Size-reduce`$(\mathbf{B})$
5: **end for**

Note that in Step 3 of Algorithm 6, the LLL-reduction is used to remove the linear dependency between $\mathbf{b}$ and $\{\mathbf{b}_k, \cdots, \mathbf{b}_n\}$ (in this case, the input is a generating set instead of a basis). Further, the inserted vector $\mathbf{b}$ will not be exchanged with any of $\{\mathbf{b}_1, \cdots, \mathbf{b}_{k-1}\}$ because the Lovász condition is always satisfied.

2.5.4 BKZ reduction

Later, Schnorr [Sch87] gave a block-wise lattice reduction for trade-off between the LLL-reduction and HKZ-reduction. It was also made more practical by Schnorr and Euchner [SE91, SE94], whose variant is known as the Block Korkine-Zolotarev (BKZ) reduction algorithm.

Definition 2.33 (BKZ-reduction)**.** *A matrix* $\mathbf{B} \in \mathbb{R}^{m \times n}$ *is called* BKZ_β *reduced for block size* $\beta \geq 2$*, if it is size-reduced and satisfies:*

$$\|\mathbf{b}_i^*\| \leq \lambda_1(\Lambda_{[i,\min(i+\beta-1,n)]})$$

for all $i \in \{1, \cdots, n-1\}$*.*

The BKZ reduction condition can be seen as a run-time/quality trade-off between LLL-reduction and HKZ-reduction. Each $\pi_i(\mathbf{b}_i)$ is promised to be a shortest non-zero vector in a projected lattice $\Lambda_{[i,i+\beta-1]}$ with some fixed dimension β except for the last indices. Intuitively, the larger block-size we choose, the lower the achieved SVP approximation factor, but the higher the run-time. Schnorr and Euchner [SE91, SE94] showed that the BKZ reduction achieves an SVP approximation factor $\leq \gamma_\beta^{(n-1)/(\beta-1)}$ with block-size β, but they can not establish a good bound on the run-time for achieving BKZ-reduction. Since then, a sequence of works focused on studying this strong lattice reduction

(see, e.g., [SE91, SE94, GHGKN06, Sch, GN08b, GN08a, HPS11, Neu17]). Next, we will first review the (practical) BKZ reduction algorithm [SE91, SE94] and discuss some good bounds [HPS11, Neu17] on run-time for (almost) achieving BKZ-reduction later.

The BKZ algorithm. The Schnorr-Euchner BKZ algorithm [SE94] takes as inputs a block-size β and a basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$ of a lattice Λ, and outputs a basis which is 'close' to being BKZ_β-reduced (up to numerical inaccuracies, as the underlying Gram–Schmidt orthogonalization is computed in floating-point arithmetic, and up to the progress parameter $\delta < 1$). BKZ starts by LLL-reducing the input basis, then calls an SVP-solver on consecutive local blocks $\mathbf{B}_{[k,\min(k+\beta-1,n)]}$ for $k = 1, \cdots, n-1$. This is called a *BKZ tour*. After each execution of the SVP-solver, if we have $\lambda_1(\Lambda_{[k,\min(k+\beta-1,n)]}) < \delta \cdot \|\mathbf{b}_k^*\|$, then BKZ updates the block $\mathbf{B}_{[k,\min(k+\beta-1,n)]}$ by inserting the vector found by the SVP-solver between indices $k-1$ and k, and LLL-reducing the block from the first index to the last index of current block (in this case, the input is a generating set instead of a basis). Otherwise, we LLL-reduce the block from first index to the last index of current block directly, without any insertion. The procedure terminates when no change occurs at all during a tour. We refer to Algorithm 7 for a complete description of the BKZ algorithm. For practical reasons, there are diverse BKZ variants.

- Early-abort. The BKZ reduction aborts when a selected number of tours are completed or when a desired output quality has been reached [HPS11].
- SVP-solver. BKZ could be run with any SVP solver; in practice for typical block sizes, the fastest one is lattice enumeration [Kan83, FP83]; the latter can be significantly accelerated with tree pruning [SE94] and even further with extreme pruning [GNR10]. In the case of enumeration with pruning, the SVP solver is not guaranteed to return a shortest non-zero vector in its input lattice. Furthermore, one can set the enumeration radius to either $\|\mathbf{b}_i^*\|$ or the Gaussian heuristic estimate of λ_1 (or whichever is smaller). In the experiments of this thesis, we use the enumeration radius $0.99 \cdot \|\mathbf{b}_i^*\|$, which is the default choice in the implementation of BKZ in `fplll` [dt16] open-source library.
- Pre-processing and post-processing. Pre-processing is performed *before* the call to the SVP solver. In the pre-processing step, some strategies (e.g., BKZ with relatively small block size) are chosen to further improve the local basis before calling the SVP solver. Post-processing is executed *after* the call to the SVP-solver, e.g., running LLL on indices 1 to $\min(k+\beta-1, n)$, in order to propagate the progress made at index k.

Algorithm 7 The Schnorr and Euchner BKZ algorithm

Require: A basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$, a block size $\beta \geq 2$ and a constant $\delta < 1$.
Ensure: A BKZ_β-reduced basis of $\Lambda(\mathbf{B})$.

repeat
  for $k = 1$ `to` $n-1$ **do**
    `Find any b such that` $\|\pi_k(\mathbf{b})\| = \lambda_1\left(\Lambda_{[k,\min(k+\beta-1,n)]}\right)$
    if $\delta \cdot \|\mathbf{b}_k^*\| > \|\mathbf{b}\|$ **then**
      $\mathbf{B} \leftarrow$`LLL-reduce`$\left(\mathbf{b}_1, \cdots, \mathbf{b}_{k-1}, \mathbf{b}, \mathbf{b}_k, \cdots, \mathbf{b}_{\min(k+\beta,n)}\right)$.
    else
      $\mathbf{B} \leftarrow$`LLL-reduce`$\left(\mathbf{b}_1, \cdots, \mathbf{b}_{\min(k+\beta,n)}\right)$.
    end if
  end for
until no change occurs.

It is clear that whenever the BKZ algorithm stops, the reduced basis will satisfy the BKZ-reduction condition. However, there is no good bound on the run-time of the BKZ algorithm. In [HPS11], instead

of analyzing the original BKZ algorithm, Hanrot *et al* consider a slightly modified variant, which is recalled in Algorithm 8.

Algorithm 8 The BKZ′ algorithm

Require: A basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$, a block size $\beta \geq 2$.
Ensure: A BKZ′$_\beta$-reduced basis of $\Lambda(\mathbf{B})$.
1: **repeat**
2: **for** $k = 1$ `to` $n - \beta + 1$ **do**
3: `Modify` $(\mathbf{b}_k, \cdots, \mathbf{b}_{k+\beta-1})$ `so that` $\mathbf{B}_{[k,k+\beta-1]}$ `is HKZ-reduced`
4: `Size-reduce(B)`
5: **end for**
6: **until** no change occurs or termination is requested.

Hanrot *et al* gave a worst-case analysis on the BKZ′ algorithm, and showed that the root Hermite factor $\leq 2\gamma_\beta^{\frac{n-1}{2(\beta-1)}+\frac{3}{2}}$ can be achieved with $\mathrm{poly}(n, \max_{i\in[n]} \log \|\mathbf{b}_i\|)$ calls to the HKZ-reduction algorithm on β-dimensional lattices. Thus the total cost is $\mathrm{poly}(n, \max_{i\in[n]} \log \|\mathbf{b}_i\|) \cdot \mathcal{C}^{HKZ}(\beta)$, where $\mathcal{C}^{HKZ}(\beta)$ is the cost of HKZ-reduction on β-dimensional lattice. Further, by using [Lov86, Page 25], there is an algorithm that can achieve an SVP approximation factor $\leq 4\gamma_\beta^{\frac{n-1}{\beta-1}+3}$ with same cost up to polynomial factor. This is asymptotically similar to the one given by [SE91, SE94] of a BKZ-reduced basis, but only requires a limited number ($\mathrm{poly}(n, \max \log \|\mathbf{b}_i\|)$) of calls to the HKZ-reduction algorithm.

In the original BKZ algorithm, SVP solver together with an LLL-reduction is used for updating the basis. The major modification compared to the original BKZ algorithm is that an HKZ-reduction is used for updating each local block instead, thus we do not need to use an additional LLL-reduction for updating. With this modification, Hanrot *et al* give a (rigorous) analysis of BKZ′, by first analyzing the following heuristic sandpile model.

Assumption 1 (Heuristic sandpile model assumption)**.** *We assume for any HKZ-reduced basis* $\mathbf{B}$*, we have*

$$x_i = \frac{1}{2}\log \gamma_{\beta-i+1} + \frac{1}{\beta-i+1}\sum_{j=i}^{\beta} x_j \tag{2.8}$$

with $x_i = \log \|\mathbf{b}_i^*\|$ *for all* $i \leq \beta$*.*

For completeness, we give a brief idea of the worst-case analysis of the BKZ′ algorithm.

We recall that the BKZ′ algorithm proceeds by running HKZ$_\beta$-reduction on $n - \beta + 1$ successive blocks $\mathbf{B}_{[k,k+\beta-1]}$ for $k \leq n-\beta+1$, and then repeat it until no change occurs. We start by considering the effect of the first HKZ$_\beta$-reduction on block $\mathbf{B}_{[1,\beta]}$. To update the $\{x_i\}_{i\leq\beta}$ such that it satisfies the heuristic sandpile model assumption, we can follow the framework of sandpile [MV10a] as follows.

Note that in the following, we analyze the effect of an HKZ$_\beta$ reduction by the effect of β many SVP on lattices with dimension from n to 1 (see Algorithm 6 for more details). First, corresponding to the effect of the first SVP call (on a β-dimensional lattice), the x_1 is assumed to be replaced by a new value $\widehat{x}_1 = \log \gamma_\beta/2 + \sum_{i=1}^{\beta} x_i$ (no matter it becomes smaller or not). Next, after the first replacement, we also have an updated Gram–Schmidt log-norms $\{x'_i\}_{2\leq i\leq\beta}$ with indices from 2 to β. As the determinant of the block is preserved, we must have $\sum_{i=2}^{\beta} x'_i = \sum_{i=2}^{\beta} x_i + (x_1 - \widehat{x}_1)$. Next, regarding the effect of the second SVP (on a $(\beta-1)$-dimensional lattice), we assume that the Gram–Schmidt log-norms x'_2 will be replaced by $\widehat{x}_2 = \log \gamma_{\beta-1}/2 + \sum_{i=2}^{\beta} x'_i$ relying on the updated Gram–Schmidt log-norms $\{x'_i\}_{2\leq i\leq\beta}$. By repeating this process, we obtain new Gram–Schmidt log-norms $\{\widehat{x}_i\}_{i\leq\beta}$ of the first block that exactly satisfies the heuristic sandpile model assumption.

We can continue to consider the effect of the remaining HKZ$_\beta$-reductions until there is no update needed. The effect of the HKZ-reductions until BKZ′ stops, can be viewed as an evolution of $\mathbf{x}$. We

let one step of evolution of $\mathbf{x}$ correspond to the effect of one tour of BKZ′. It is shown in [HPS11] that the vector $\mathbf{x}$ converges in a polynomial number of steps. In other words, after a polynomial number of steps, the updated $\mathbf{x}$ can be very close to a fixed-point $\bar{\mathbf{x}}$. We refer the reader to [HPS11] for a proof of the existence of such a fixed-point and also the fast convergence of the evolution of $\mathbf{x}$ toward it.

Note that in the analysis above, only the update of the first Gram–Schmidt log-norm of each local block (except the last block) is useful. Because in the analysis, the update of each block, will only depend on the determinant of current block instead of the concrete values of the Gram–Schmidt log-norms. For example, we again let $\mathbf{x}$ denote the starting Gram–Schmidt log-norms, and $\{x'_i\}_{i\leq\beta}$ denote the new Gram–Schmidt log-norms (e.g., satisfying Equations 2.8) after the effect of HKZ_β-reduction on the first block. Then the update of the second block will only be determined by (the logarithm of its determinant) $\sum_{i=1}^{\beta+1} x_i - x'_1$.

Thus we can focus on the update of the first value of each local block (except the last block). In fact, if we update the first value by the Gaussian heuristic, we could instead analyze the practical behavior of the BKZ′ algorithm. This is known as the Chen–Nguyen simulator [CN11], which we will discuss later in this section.

Now, we summarize the bounds on run-times and SVP approximation factors for LLL-reduction, BKZ-reduction and HKZ-reduction, in Figure 2.9.

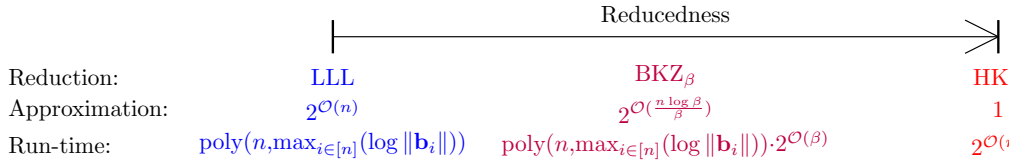

Fig. 2.9: Comparison of run-time and SVP approximation factor between LLL-reduction, BKZ-reduction and HKZ-reduction.

As we have already seen above, there are well-known bounds of the worst-case behavior of BKZ-reduction. However, these worst-case results poorly reflect the practical behavior of the BKZ algorithm [GN08b]. Besides rigorous results, specific heuristic assumptions were also proposed for estimating the behavior of the BKZ algorithm in practice. In [Sch03], Schnorr first introduced the Geometric Series Assumption (GSA), which states that, for typical input bases, the Gram–Schmidt norms $\{\|\mathbf{b}_i^*\|, i\in[n]\}$ of a BKZ-reduced basis behave as a geometric series: there exists a constant $r>1$ such that $\|\mathbf{b}_i^*\|/\|\mathbf{b}_{i-1}^*\| \approx r$ for all $i<n$. Among others, this implies that $\mathrm{HF}(\mathbf{B}) \approx r^{(n-1)/2}$. It was argued in [CN11] (see also [Che09, Chapter 4]) that for β small compared to n, one should have $r \approx (\frac{\beta}{2\pi\mathrm{e}}(\pi\beta)^{\frac{1}{\beta}})^{\frac{1}{\beta-1}}$. The latter value is derived by relying on the Gaussian heuristic to estimate the first minimum of projected sublattices. It was experimentally observed that the GSA is a good first approximation to the practical behavior of BKZ. As shown in Figure 2.10, the plot of Gram–Schmidt log-norms is close to be a line. Further, the larger the block-size is used, the higher quality the basis will be (with more balanced Gram–Schmidt norms), and a line with smaller slope. In this experiment, we consider the knapsack-type lattice bases generated by the Darmstadt lattice challenge generator [2]. We generate the basis of a 100-dimensional lattice, and run the BKZ algorithm with block-sizes $\beta\in\{4,8,16,35\}$. For a comparison with LLL, we also run the LLL algorithm on the generated basis. We conduct the experiment 100 times using input lattices generated with `seed` from the set $\{0,\cdots,99\}$. We record the final scaled Gram–Schmidt log-norms (such that the summation is 0) and plot the averaged values over 100 outputs.

[2] `https://www.latticechallenge.org/svp-challenge/`

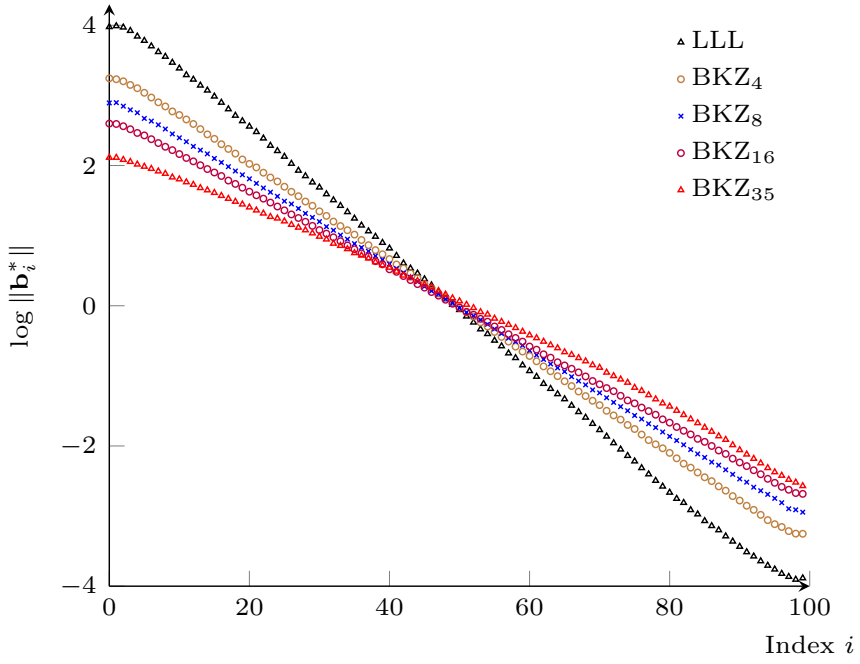

Fig. 2.10: LLL and BKZ outputs for various block-sizes.

The Chen–Nguyen simulator. In [CN11], Chen and Nguyen proposed a simulator to capture the practical behavior of BKZ with relatively large block size (e.g., $\beta \geq 45$). As we have discussed before, the Chen–Nguyen simulator can be seen as revised version of heuristic sandpile model, while the application of the heuristic sandpile model assumption is replaced by the Gaussian heuristic assumption. The goal was to estimate the practical behavior of BKZ for hard-to-solve instances. Overall, the simulation proceeds closely to BKZ. Firstly, at the beginning of each BKZ tour, a boolean flag τ is initialized to `true`. To update each local block, the simulator first (deterministically) compute the Gaussian heuristic value $\mathrm{GH}(\mathbf{B}_{[k,\min(k+\beta-1,n)]})$ as an estimation of the first miminum, by looking at the Gram–Schmidt norms of the current local block (except for small blocks in the end). This is corresponding to `Line 8` of Algorithm 9, we recall that v_d denotes the volume of d-dimensional unit ball. The computed value is then used to update the current local block, if it is smaller than the current $\|\mathbf{b}_k^*\|$ (as written in `Lines 10--11`). Else, the local block is kept unchanged. To update, the first Gram–Schmidt norm is replaced by the selected value, and the boolean flag τ is flipped once such an update occurs (as written in `Line 12`). Then all the remaining Gram–Schmidt norms, of indices $k' \in [k+1, n-45]$ are updated one by one to $\mathrm{GH}(\mathbf{B}_{[k',\min(k'+\beta-1,n)]})$, independently of whether the current $\|\mathbf{b}_k^*\|$ is already small enough or not (as written in `Line 15`). At the end of each tour, there is an additional update of the tail block of length 45 (as written in `Lines 19--21`). The Gram–Schmidt norms in this tail block are simulated with the experimental Gram–Schmidt norms of HKZ-reduced bases of 45-dimensional unit-volume random lattices. The experimental Gram–Schmidt norms are prepared in `Lines 1--2` of Algorithm 9. This length of 45 was chosen because the minimum of blocks of dimension > 45 within BKZ follows the Gaussian heuristic quite well (as observed with the extensive experiments of [CN11]). This special treatment on the tail block also makes the simulator more precise for capturing the practical behavior of BKZ compared to the Gaussian heuristic.

The Aono *et al* random points model. Aono *et al* [AWHT16] propose a BKZ variant, in which they use a relaxation on the SVP step within each local block. Instead of requiring a shortest vector, Aono *et al* only assume to find a short vector within a radius $\alpha \cdot \mathrm{GH}(\mathbf{B}_{[i:j]})$ for some $\alpha \geq 1$. For

Algorithm 9 The Chen–Nguyen BKZ simulator

Require: The Gram–Schmidt log-norms $\{\ell_i = \log \|\mathbf{b}_i^*\|\}_{i \le n}$ and an integer $N \ge 1$.
Ensure: A prediction of the Gram–Schmidt log-norms $\{\widehat{\ell}_i = \log \|\mathbf{b}_i^*\|\}_{i \le n}$ after N tours of BKZ.

for $i = 1$ `to` 45 **do** $r_i \leftarrow \mathbb{E}[\log \|\mathbf{b}_k^*\| : \mathbf{B}$ HKZ-reduced basis of $\Lambda \leftarrow \Gamma_{45}]$
end for
for $j = 1$ `to` N **do**
  $\tau \leftarrow$ `true`
  for $k = 1$ `to` $n - 45$ **do**
    $d \leftarrow \min(\beta, n - k + 1)$; $e \leftarrow k + d - 1$
    $\log \mathrm{vol}(\Lambda_{[k,e]}) \leftarrow \sum_{i=1}^{e} \ell_i - \sum_{i=1}^{k-1} \widehat{\ell}_i$
    $g \leftarrow \left(\log \mathrm{vol}(\Lambda_{[k,e]} - \log v_d\right)/d$
    if τ `= true` **then**
      if $g < \ell_k$ **then**
        $\widehat{\ell}_k \leftarrow g$
        $\tau \leftarrow$ `false`
      end if
    else
      $\widehat{\ell}_k \leftarrow g$
    end if
  end for
  $\log \mathrm{vol}(\Lambda_{[k,e]}) \leftarrow \sum_{i=1}^{n} \ell_i - \sum_{i=1}^{n-45} \widehat{\ell}_i$
  for $k' = n - 44$ `to` n **do**
    $\widehat{\ell}_{k'} \leftarrow \frac{\log \mathrm{vol}(\Lambda_{[k,e]})}{45} + r_{k'+45-n}$
  end for
  $\{\ell_1, \cdots, \ell_n\} \leftarrow \{\widehat{\ell}_1, \cdots, \widehat{\ell}_n\}$
end for

analysis, they assume that the found vector is random in the ball with this radius. This leads to an estimate of $\frac{\beta}{\beta+1} \cdot \alpha \cdot \mathrm{GH}(\mathbf{B}_{[i:j]})$ for the norm of the found vector (corresponding to the case $K = 1$ in Lemma 2.25 below).

Lemma 2.25 ([AWHT16, Lemma 1])**.** *Let $\mathbf{x}_1, \cdots, \mathbf{x}_K$ be K points uniformly sampled from the n-dimensional unit ball. Then, the expected value of the shortest norm of these points is*

$$\mathbb{E}\left[\min_{i \in [K]} \|\mathbf{x}_i\|\right] = K \cdot \textsc{Beta}(K, \frac{n+1}{n}) := K \cdot \int_0^1 t^{1/n}(1-t)^{K-1} dt.$$

Aono *et al* adapt the Chen–Nguyen simulator accordingly, where the Gaussian heuristic model is replaced by the random points model above for estimating the norm of vector found by the (modified) SVP solver.

Chapter 3

An improved reduction from BDD to uSVP

In this chapter, we first review the reduction [LM09] from $\mathrm{BDD}_{1/(2\gamma)}$ to uSVP_γ due to Lyubashevsky and Micciancio, as well as their analysis. To better understand the limitations of this reduction, we also give an improved reduction for all non-integral γ's, which is known as folklore [Mic15, BG15]. Then we discuss the difficulties we meet when using the same uSVP solver to solve a (possibly) harder BDD instance. Finally, we give our improved reduction from $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$ to uSVP_γ for any $\gamma \geq 1$ and its formal analysis.

Contents

3.1 Introduction

In cryptography, BDD is closely related to the Learning With Errors problem (LWE) [Reg09], which serves as a security foundation for numerous cryptographic primitives. When the number of requested LWE samples is bounded (which is most often the case in cryptographic constructions), LWE may be viewed as a variant of BDD in which the offset from $\mathbf{t}$ to Λ is Gaussian (like in the decoding context), and Λ is a randomly sampled lattice. In communications theory, BDD models the task of decoding in the context of continuous channels with white Gaussian noise [Bud75]. The information to be transmitted is stored in a lattice vector, and the receiver should recover this vector from a noisy version thereof. The knowledge of the signal-to-noise ratio implies a bound on the distance from the noisy vector to the lattice. Decoding a white Gaussian noise channel can be seen as a version of BDD in which the distance to the lattice follows a prescribed distribution.

A common approach to solve BDD is via Kannan's embedding technique [Kan87, Se. 6]. The principle is to map the offset between $\mathbf{t}$ and a closest lattice vector to $\mathbf{t}$, to a shortest non-zero vector in an $(n+1)$-dimensional lattice. Lattice reduction [LLL82, Sch87] and short lattice vector enumeration [Kan83, FP83] may then be used to find shortest non-zero vectors in the $(n+1)$-dimensional lattice. Formally, Kannan's embedding technique is a reduction from BDD to a variant of the Shortest Vector Problem (SVP) in which the pair of shortest non-zero vectors in the lattice under scope are known to be much shorter than any other lattice vector not parallel to them. For a lattice Λ, we define the second minimum $\lambda_2(\Lambda)$ as the minimal radius of a zero-centered ball that contains two or more linearly independent vectors from Λ. The unique Shortest Vector Problem (uSVP_γ) of parameter $\gamma \geq 1$ consists in finding a shortest non-zero vector in a lattice Λ described by an input basis $\mathbf{B} = (\mathbf{b}_i)_i$, under the promise that $\lambda_2(\Lambda) \geq \gamma \cdot \lambda_1(\Lambda)$. This reduction was analyzed by Lyubashevsky and Micciancio in [LM09], who showed that $\text{BDD}_{1/(2\gamma)}$ reduces to uSVP_γ for any $\gamma \geq 1$. Later, Liu *et al.* [LWXZ14] refined the analysis of Lyubashevsky and Micciancio and proved that BDD_{1/γ_1} reduces to uSVP_γ with $\gamma_1 = \sqrt{3/(4-\gamma^2)}\gamma + 1$, for any $\gamma \in (1, 1.9318)$. It is folklore [Mic15, BG15] that the analysis can be tightened even more, resulting in a proof that BDD_{1/γ_1} reduces to uSVP_γ with $\gamma_1 = (2\gamma^2 + 2\lfloor\gamma\rfloor\lfloor\gamma+1\rfloor)/(2\lfloor\gamma\rfloor+1)$, for any $\gamma \geq 1$. Note that in the case of $\gamma = 1$ (and in fact all integral γ), all three results are identical: $\text{BDD}_{1/2}$ reduces to uSVP_1.

3.2 Our contributions

We clarify the limitations in the Lyubashevsky-Micciancio reduction and then give a probabilistic polynomial-time improved reduction from $\text{BDD}_{1/(\sqrt{2}\gamma)}$ to uSVP_γ, for any $\gamma \geq 1$. As clearly visible in Figure 3.1, this reduction supersedes all prior results with respect to the BDD problem parameter. In particular, we reduce $\text{BDD}_{1/\sqrt{2}}$ to uSVP_1. Our improvement comes with one weakness: the reduction is probabilistic. Like prior reductions, the dimension of the uSVP instance is only one more than the dimension of the BDD instance. The main ingredient to the improvement is the use of Khot's lattice sparsification [Kho03, Kho05] before resorting to Kannan's embedding, in order to boost the uSVP parameter.

3.2.1 Technical overview

We illustrate our improvement with the case of $\text{BDD}_{1/2}$. Given the $\text{BDD}_{1/2}$ instance $(\mathbf{B}, \mathbf{t})$, Kannan's embedding consists in constructing the following uSVP_1 instance:

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B} & \mathbf{t} \\ \mathbf{0} & d \end{pmatrix} \in \mathbb{Q}^{n+1},$$

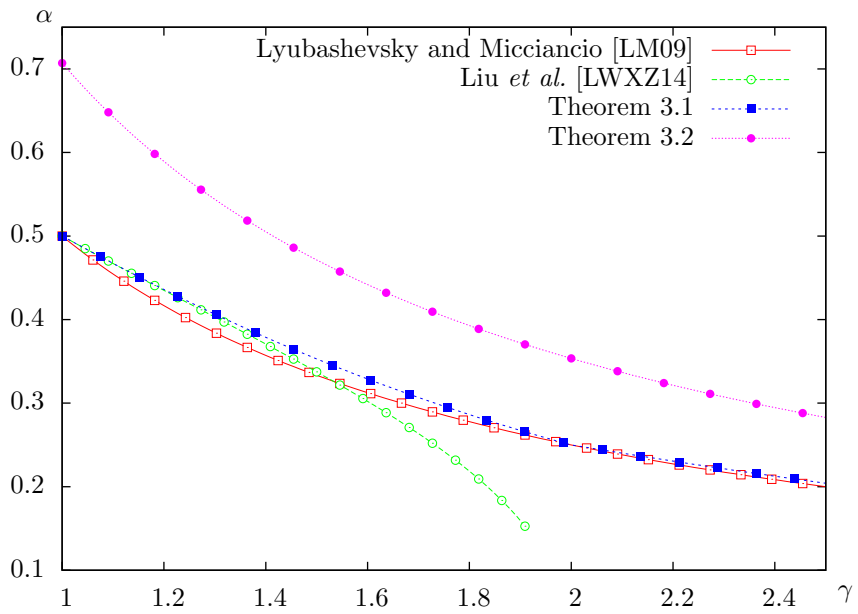

Fig. 3.1: Comparison between prior reductions from BDD_α to uSVP_γ, and ours.

with $d = \mathrm{dist}(\mathbf{t}, \Lambda) \leq \lambda_1(\Lambda)/2$, where Λ is the lattice spanned by $\mathbf{B}$ (in fact, the reduction does not know d, but this is a mere technical problem which can be handled easily, as explained in Section 2.2). If $\mathbf{c}$ denotes a closest vector to $\mathbf{t}$ in Λ then it may be proved that the vector $\mathbf{s}' = ((\mathbf{c} - \mathbf{t})^{\mathrm{T}}, -d)^{\mathrm{T}}$ is a shortest non-zero vector of lattice Λ' of basis $\mathbf{B}'$. Now, let $\mathbf{s}$ denote a shortest non-zero vector in Λ and assume that $\mathbf{t}$ is exactly halfway between $\mathbf{c}$ and $\mathbf{c} + \mathbf{s}$. Then both $\mathbf{s}'$ and $\mathbf{s}' + (\mathbf{s}^{\mathrm{T}}, 0)^{\mathrm{T}}$ in Λ' have norm $\sqrt{2} \cdot d$ but are linearly independent. This shows that we can have $\lambda_2(\mathcal{L}') = \lambda_1(\mathcal{L}')$ and obtain a uSVP_γ instance with $\gamma = 1$. This is a limitation of Kannan's embedding and hence of its analyzes.

We modify the reduction to increase the ratio $\lambda_2(\mathcal{L}')/\lambda_1(\mathcal{L}')$. To achieve this, we use lattice sparsification on Λ. It provides a full-rank sublattice $\Lambda_{\mathrm{sparse}} \subseteq \Lambda$ that still contains a closest vector $\mathbf{c} \in \Lambda$ to $\mathbf{t}$, but no other close-by vector. We consider the vectors of Λ whose coordinates with respect to a basis $\mathbf{B}$ satisfy a linear equation modulo some prime integer p: $\Lambda_{\mathrm{sparse}} = \Lambda_{p,\mathbf{z}} = \{\mathbf{b} \in \Lambda : \langle \mathbf{z}, \mathbf{B}^{-1}\mathbf{b} \rangle = 0 \bmod p\}$, for some vector $\mathbf{z} \in \mathbb{Z}_p^n$. This technique was first introduced by Khot in [Kho03, Kho05]. To guarantee that vectors in Λ remain in the sparsified set with probability close to $1/p$, a uniform coset of $\Lambda_{p,\mathbf{z}}$ (modulo Λ) was considered in [DK13, DRS14]. Technically, we use the formulation from [SD16a] of the latter variant.

The aim of sparsification in our context is to keep a closest vector $\mathbf{c} \in \Lambda$ to $\mathbf{t}$, and remove as many nearby vectors of Λ as possible. After sparsification, vector $\mathbf{c}$ remains in the sparse lattice (with non-negligible probability), and all other remaining vectors are much further away from $\mathbf{t}$. For $\mathrm{BDD}_{1/2}$, a simple example of sparsification is shown in Figure 3.2: there are two points simultaneously closest to the target point; then sparsification is used to remove one of the closest points (either is fine); in the sparse lattice, the closest vector is much closer than any other lattice vector. In Figure 3.2, after sparsification, all the lattice vectors labelled with filled dots are kept, e.g., vector $\mathbf{c} + \mathbf{w}$, and other vectors labelled with hollow dots are removed, e.g., vector $\mathbf{c} + \mathbf{s} + \mathbf{w}$.

The probability of keeping a vector of Λ in the sparsified set is essentially $1/p$. As we want the probability of keeping $\mathbf{c}$ to be non-negligible, we are hence restricted to taking $p \leq \mathrm{poly}(n)$. As a result, we cannot remove more than polynomially many close-by vectors, because each one individually is removed with probability $\approx 1 - 1/p$ (a precise statement is given in Lemma 2.9). To assess the limitation of our reduction, we are hence interested in the largest value of α such that for any lattice Λ and any

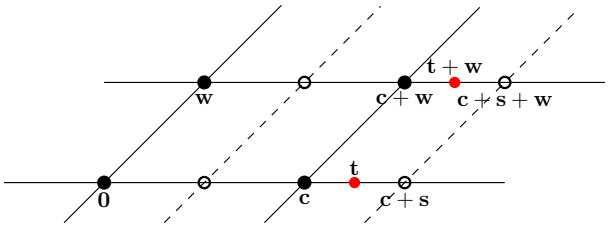

Fig. 3.2: An example of sparsification for $\mathrm{BDD}_{1/2}$ (here $\mathbf{w} \in \Lambda_{p,\mathbf{z}}/\Lambda$).

vector $\mathbf{t}$, there are at most $\mathrm{poly}(n)$ vectors within distance $\alpha \cdot \lambda_1(\Lambda)$ from $\mathbf{t}$. The quantity $\alpha \cdot \lambda_1(\Lambda)$ can be viewed as the worst-case list-decoding radius. Interestingly, this problem was studied by Ajtai [Ajt98] and Micciancio [Mic01b] in the context of proving hardness of SVP. Proofs of the following two statements may be found in [MG02, Chap. 5]:

- For any lattice Λ and vector $\mathbf{t}$, there are $\leq 2n$ vectors of Λ within distance $\lambda_1(\Lambda)/\sqrt{2}$ from $\mathbf{t}$.
- For any $\alpha > 1/\sqrt{2}$, there exists $\varepsilon > 0$ such that for any sufficiently large n we can find an n-dimensional lattice Λ and a vector $\mathbf{t}$ such that there are $\geq 2^{n^\varepsilon}$ vectors of Λ within distance $\alpha \cdot \lambda_1(\Lambda)$ from $\mathbf{t}$.

The overall reduction consists in first sparsifying Λ to $\Lambda_{p,\mathbf{z}}$ and shifting $\mathbf{t}$ (as we use a coset of $\Lambda_{p,\mathbf{z}}$), and then resorting to Kannan's embedding. To increase the ratio $\lambda_2(\mathcal{L}')/\lambda_1(\mathcal{L}')$, we decrease the bottom-right entry in $\mathbf{B}'$ from d to $k \cdot d$ for some $k < 1$. Geometrically, this has the effect of limiting the contribution of the extra dimension. This idea was already used in [LWXZ14], but we decrease k even further, to $1/\mathrm{poly}(n)$. An additional difficulty, related to this decrease of k, is that short vectors in Λ' may be obtained by using multiples of $\mathbf{t}$. Let $m \geq 2$ and $\mathbf{d} \in \Lambda$ closest to $m \cdot \mathbf{t}$. Then, vector $((\mathbf{d} - m \cdot \mathbf{t})^{\mathrm{T}}, mkd)^{\mathrm{T}}$ may be very short (if very unlucky, it has norm mkd). We remove such annoying vectors with sparsification.

3.3 The Lyubashevsky-Micciancio reduction and its limitation

First, we recall the reduction and its original proof. Then we give a slight improvement over it by considering a tighter analysis.

We now explain the Lyubashevsky-Micciancio reduction and its analysis. Thanks to Corollary 2.12, it suffices to reduce $\mathrm{BDD}_{(1-1/n)/(2\gamma)}$ to uSVP_γ. Let $(\mathbf{B}, \mathbf{t})$ be a $\mathrm{BDD}_{(1-1/n)/(2\gamma)}$ instance. Let $\ell = \lambda_1(\Lambda(\mathbf{B}))$ and $d = \mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \leq \ell(1 - 1/n)/(2\gamma)$. We can assume that we have a value $d_0 \in [d, d/(1 - 1/n^2))$ as explained in Section 2.2.

The lattice basis for the SVP instance is constructed as

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B} & \mathbf{t} \\ \mathbf{0} & d_0 \end{pmatrix} \in \mathbb{Q}^{n+1}.$$

Any vector in this lattice can be written as $\mathbf{b}' = ((\mathbf{b} - m \cdot \mathbf{t})^{\mathrm{T}}, -md_0)^{\mathrm{T}}$ with $\mathbf{b} \in \Lambda(\mathbf{B})$ and $m \in \mathbb{Z}$. We claim that any shortest non-zero vector $\mathbf{s}'$ has the form $((\mathbf{c} - m \cdot \mathbf{t})^{\mathrm{T}}, -md_0)^{\mathrm{T}}$ with $\mathbf{c} \in \Lambda(\mathbf{B})$ and $m = \pm 1$, where $\|\mathbf{c} - m \cdot \mathbf{t}\| = d$ (i.e., vector $m \cdot \mathbf{c}$ is a closest vector to $\mathbf{t}$ in $\Lambda(\mathbf{B})$).

We give lower bounds on the norm of any vector $\mathbf{b}' = ((\mathbf{b} - m \cdot \mathbf{t})^{\mathrm{T}}, -md_0)^{\mathrm{T}}$ that is not parallel to $((\mathbf{c} - \mathbf{t})^{\mathrm{T}}, -d_0)^{\mathrm{T}}$. The bounds depend on the value of m.

Case 1: $m = 0$. In this case, we must have $\mathbf{b} \neq \mathbf{0}$. We have

$$\|\mathbf{b}'\|^2 \;=\; \|\mathbf{b}\|^2 \;\geq\; \ell^2 \;\geq\; \frac{4\gamma^2 d^2}{(1-1/n)^2} \;\geq\; 4\gamma^2 d_0^2 \;\geq\; (\sqrt{2}\gamma)^2(d^2+d_0^2).$$

Case 2: $1 \leq m \leq 2\gamma$. We have, by the triangle inequality:

$$\|\mathbf{b} - m\cdot\mathbf{t}\| \geq \Big|\, \|\mathbf{b} - m\cdot\mathbf{c}\| - m\,\|\mathbf{c}-\mathbf{t}\| \,\Big| = \Big|\, \|\mathbf{b} - m\cdot\mathbf{c}\| - md\Big| = \|\mathbf{b} - m\cdot\mathbf{c}\| - md.$$

As $\mathbf{b}'$ is not parallel to $((\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -d_0)^{\mathrm{T}}$, we have $\|\mathbf{b} - m\cdot\mathbf{c}\| \geq \ell \geq (2\gamma d)/(1-1/n) \geq md$, hence justifying the last equality. Therefore:

$$\|\mathbf{b}'\|^2 \geq \left(\frac{2\gamma d}{1-1/n} - md\right)^2 + m^2 d_0^2 \geq 2\left(\frac{\gamma d}{1-1/n} - md\right)^2 + \frac{2\gamma^2 d^2}{(1-1/n)^2} \geq \gamma^2(d^2+d_0^2).$$

We used the fact that $d_0 \in [d, d/(1-1/n^2))$ and the non-negativity of $(\gamma d/(1-1/n) - md)^2$.

Case 3: $m > 2\gamma$. We have

$$\|\mathbf{b}'\|^2 \;\geq\; m^2 d_0^2 \;\geq\; 4\gamma^2 d_0^2 \;\geq\; (\sqrt{2}\gamma)^2(d^2+d_0^2).$$

As a result, we obtain a uSVP instance $\mathbf{B}'$ with gap $\lambda_2(\Lambda(\mathbf{B}'))/\lambda_1(\Lambda(\mathbf{B}')) \geq \min(\sqrt{2}\gamma, \gamma) = \gamma$. Once the uSVP solver outputs $\mathbf{s}' = ((\mathbf{s}'_1)^{\mathrm{T}}, s'_2)^{\mathrm{T}}$, we have a vector $\mathbf{c} = \mathbf{s}'_1 \pm \mathbf{t}$ such that $\|\mathbf{c} \pm \mathbf{t}\| = d$.

In fact, the fixed embedded height (d_0) as the (approximated) distance from target vector t to the lattice makes an artificial restriction on the reduction, such that we can not achieve the optimal result. We show that the same reduction with relaxed embedded height can achieve a tighter result.

Theorem 3.1. *There is a polynomial-time reduction from* $\mathrm{BDD}_{1/\gamma_1}$ *to* uSVP_γ*, where* $\gamma_1 = \frac{2\gamma^2+2\lfloor\gamma\rfloor\lfloor\gamma+1\rfloor}{2\lfloor\gamma\rfloor+1}$ *for* $\gamma \geq 1$.

Proof. We prove here that for $1 \leq \gamma < 2$, there is a polynomial reduction from $\mathrm{BDD}_{1/\gamma_1}$ to uSVP_γ, where $\gamma_1 = 2(\gamma^2+2)/3$. The extension to $\gamma \geq 2$ can be obtained using a similar method.

Given $(\mathbf{B}, \mathbf{t})$ an instance of $\mathrm{BDD}_{1/\gamma_1}$, we have $d = \mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B})) \leq \lambda_1(\Lambda(\mathbf{B}))/\gamma_1$. We assume for the moment that we know the exact value d. We consider the embedded lattice

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B} & \mathbf{t} \\ \mathbf{0} & kd \end{pmatrix}.$$

Notice we have a factor k in front of the embedded height. Introducing such a scaling coefficient k was already proposed in [LWXZ14], to improve the Lyubashevsky-Micciancio reduction and obtain the result stated in [LWXZ14, Thm. 2]. The value used in [LWXZ14] was $k = \gamma/\sqrt{4-\gamma^2} < 1$ (for $1 < \gamma < 1.9318$). Here we have a similar k that will be determined later.

Any vector in this lattice can be written as $\mathbf{b}' = ((\mathbf{b} - m\cdot\mathbf{t})^{\mathrm{T}}, -mkd)^{\mathrm{T}}$ with $\mathbf{b} \in \Lambda(\mathbf{B})$ and $m \in \mathbb{Z}$. Similar to the analysis done for the Lyubashevsky-Micciancio reduction, we can claim that any shortest non-zero vector $\mathbf{s}'$ has the form $((\mathbf{c} - m\cdot\mathbf{t})^{\mathrm{T}}, -mkd)^{\mathrm{T}}$ with $\mathbf{c} \in \Lambda(\mathbf{B})$ and $m = \pm 1$, where $\|\mathbf{c} - m\cdot\mathbf{t}\| = d$ (i.e., vector $m\cdot\mathbf{c}$ is a closest vector to $\mathbf{t}$ in $\Lambda(\mathbf{B})$).

We give lower bounds on the norm of any vector $\mathbf{b}' = ((\mathbf{b} - m\cdot\mathbf{t})^{\mathrm{T}}, -mkd)^{\mathrm{T}}$ that is not parallel to $((\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -kd)^{\mathrm{T}}$. The bounds depend on the value of m.

Case 1: If $m \leq \lfloor \gamma_1 \rfloor$, we have

$$\|\mathbf{b} - m \cdot \mathbf{t}\| \geq \Big| \|\mathbf{b} - m \cdot \mathbf{c}\| - m \|\mathbf{c} - \mathbf{t}\| \Big| = \Big| \|\mathbf{b} - m \cdot \mathbf{c}\| - md \Big| = \|\mathbf{b} - m \cdot \mathbf{c}\| - md.$$

Because $\mathbf{b}'$ is not parallel to $((\mathbf{c} - \mathbf{t})^{\mathrm{T}}, -kd)^{\mathrm{T}}$, we have $\|\mathbf{b} - m \cdot \mathbf{c}\| \geq \lambda_1(\Lambda(\mathbf{B})) \geq \gamma_1 d \geq md$, hence justifying the last equality. In this case, we obtain $\|\mathbf{b} - m \cdot \mathbf{t}\| \geq \gamma_1 d - md$. As $\|\mathbf{b}'\|^2 \geq (\gamma_1 d - md)^2 + m^2 k^2 d^2$, to obtain the result in this case, it suffices that:

$$(\gamma_1 - m)^2 + m^2 k^2 \geq (\gamma \sqrt{1 + k^2})^2.$$

As $m \leq \lfloor \gamma_1 \rfloor$, it is sufficient to consider $m \in \{0, 1, 2, 3\}$ and $\gamma \in (1, 2)$.

- If $m = 0$, the above inequality is equivalent to $k^2 \leq \frac{\gamma_1^2 - \gamma^2}{\gamma^2}$;
- If $m = 1$, the above inequality is equivalent to $k^2 \leq \frac{(\gamma_1 - 1)^2 - \gamma^2}{\gamma^2 - 1}$;
- If $m = 2$, the above inequality is equivalent to $k^2 \geq \frac{\gamma^2 - (\gamma_1 - 2)^2}{2^2 - \gamma^2}$;
- If $m = 3$, the above inequality is equivalent to $k^2 \geq \frac{\gamma^2 - (\gamma_1 - 3)^2}{3^2 - \gamma^2}$.

First, it suffices to derive the smallest $\gamma_1 = 2(\gamma^2 + 2)/3$ with equalities in the middle two cases: $m = 1$ and $m = 2$. Then, we can combine $\gamma_1 = 2(\gamma^2 + 2)/3$ with any one of the middle two cases to determine the value $k = \sqrt{(4\gamma^2 - 1)/9}$. Note that for general γ, it is sufficient to consider $m = \lfloor \gamma \rfloor$ and $m = \lfloor \gamma + 1 \rfloor$ when we want to determine the relation between γ_1 and γ, and also the value of k.

Case 2: If $m > \lfloor \gamma_1 \rfloor$, we have

$$\|\mathbf{b}'\|^2 \geq m^2 k^2 d^2 \geq \gamma_1^2 k^2 d^2.$$

By taking k and γ_1 as above, we obtain that the latter is bounded from below by $(\gamma \sqrt{1 + k^2} d)^2$.

We can extend the above result to larger γ, with γ_1 as in the theorem statement [1]. The exact value of d may be unknown, but we could modify the proof with an approximated value of $d = \mathrm{dist}(\mathbf{t}, \Lambda(\mathbf{B}))$ using a guess $d_0 \in [d, d/(1 - 1/n))$. We use the reduction with the approximated value d_0. The reduction and its analysis above provide a reduction from $\mathrm{BDD}_{(1-1/n)^c/\gamma_1}$ to uSVP_γ for some constant c. Further, combined with Lemma 2.12, we obtain the claimed reduction from $\mathrm{BDD}_{1/\gamma_1}$ to uSVP_γ. □

3.4 Improved reduction from BDD to uSVP

In this section, we modify the Lyubashevsky-Micciancio reduction such that we can circumvent the limitations and obtain the following improvement.

Theorem 3.2. *For $\gamma \geq 1$, there is a probabilistic polynomial-time reduction from $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$ to uSVP_γ.*

To simplify our presentation for one moment, we will assume a polynomially large γ. We will go back to any $\gamma \geq 1$ a bit later. We can see that the main barrier of the Lyubashevsky-Micciancio reduction is in Case 2 of their original proof. in particular, when $m = \gamma$, we may have a vector with norm exactly $\gamma \lambda_1(\Lambda(\mathbf{B}'))$. The vector $m\mathbf{t}$ may get closer and closer to the lattice $\Lambda(\mathbf{B})$ as m increases. For example, when $m = \gamma$, we may have $m \cdot \mathbf{t} \in \Lambda(\mathbf{B})$, which results in $\|\mathbf{b} - m \cdot \mathbf{t}\| = 0$ for some

[1] To obtain the value of k for general γ, one can combine the above formula with $k^2 \leq \frac{(\gamma_1 - 1)^2 - \gamma^2}{\gamma^2 - a^2}$ for $\gamma \in (a, a + 1)$.

$\mathbf{b} \in \Lambda(\mathbf{B})$. To address this issue, we attempt to maintain a large distance between $m \cdot \mathbf{t}$ and $\Lambda(\mathbf{B})$ even when m becomes large. This forces us to remove some superfluous lattice points that are close to $m \cdot \mathbf{t}$.

Notice that to apply the sparsification result stated in Lemma 2.9, we still need that the coordinates of all these lattice vectors within a same ball with radius $\lambda_1/\sqrt{2}$ are different from each other modulo a properly chosen integer p. In fact, we give a stronger result that for any two lattice vectors in Λ with distance smaller than $p \cdot \lambda_1(\Lambda)$ where p is an integer, the coordinate vectors of these two lattice vectors differ modulo p.

Lemma 3.1. *For any basis* $\mathbf{B}$*, any integer* p *and any pair of lattice vectors* $\mathbf{b} \neq \widehat{\mathbf{b}}$ *with* $\|\mathbf{b} - \widehat{\mathbf{b}}\| < p \cdot \lambda_1(\Lambda(\mathbf{B}))$*, we have that* $\mathbf{B}^{-1}\mathbf{b} \neq \mathbf{B}^{-1}\widehat{\mathbf{b}} \bmod p$.

Proof. For an arbitrary lattice vector $\mathbf{a}$, we let $\widetilde{\mathbf{a}}$ denote its coordinate vector under the basis $\mathbf{B}$, i.e., $\widetilde{\mathbf{a}} = \mathbf{B}^{-1}\mathbf{a}$. Assume by contradiction that $\widetilde{\mathbf{x}} = \widetilde{\mathbf{v}} \bmod p$. Equivalently, we have $\mathbf{x} - \mathbf{v} \in p \cdot \Lambda(\mathbf{B})$.

Combined with $\mathbf{x} \neq \mathbf{v}$, we have $\|\mathbf{x}-\mathbf{v}\| \geq p \cdot \lambda_1(\Lambda)$, which is in contradiction with $\|\mathbf{x}-\mathbf{v}\| < p \cdot \lambda_1(\Lambda)$. As a result, we have $\widetilde{\mathbf{x}} \neq \widetilde{\mathbf{v}} \bmod p$. □

We can now construct a random sub-lattice with basis $\mathbf{B}_{p,\mathbf{z}}$ defined by some prime p and vector $\mathbf{z} \in \mathbb{Z}_q^n$ generated by the algorithm of Lemma 2.10, such that with non-negligible probability polynomially many superfluous lattice points that are close to $m \cdot \mathbf{t}$ are removed. Further, thanks to Lemma 2.11, we can make sure that all superfluous lattice points in $\mathcal{B}(m \cdot \mathbf{t}, \lambda_1(\Lambda(\mathbf{B}_{p,\mathbf{z}}))/\sqrt{2})$ except $m \cdot \mathbf{c} \in \Lambda(\mathbf{B}_{p,\mathbf{z}})$ are removed for all $m \leq \gamma$. This gives an improvement of Case 2 in the Lyubashevsky-Micciancio analysis.

However, this does not directly give a $\sqrt{2}$ factor improvement. We also replace the value d_0 by kd_0 with a small k. This is similar to prior improvements (both [LWXZ14] and the folklore [Mic15, BG15] presented above) Here, we reduce k even more, to $k = 1/n$. This relaxation of k already allows us to achieve the limit of the reduction, and decreasing k further does not give us further improvement. As a consequence, it does not suffice to consider only the first γ balls $\mathcal{B}(m \cdot \mathbf{t}, \lambda_1(\Lambda(\mathbf{B}))/\sqrt{2})$. We need to argue that there is no lattice vector except (possibly) $m \cdot \mathbf{c}$ inside $\mathcal{B}(m \cdot \mathbf{t}, \lambda_1(\Lambda(\mathbf{B}))/\sqrt{2})$, even for m's that are polynomially larger than γ.

Now we have a brief idea of how to circumvent the limitations. Let us first describe the improved reduction and then the formal proof. Thanks to Lemma 2.12, it suffices to reduce $\mathrm{BDD}_{(1-1/n)/(\sqrt{2}\gamma)}$ to uSVP_γ.

Algorithm 1. The $\mathrm{BDD}_{(1-1/n)/(\sqrt{2}\gamma)}$ to uSVP_γ reduction.

Input: a basis $\mathbf{B} = \{\mathbf{b}_i\}_{i\in[n]}$ of an n-dimensional lattice $\Lambda \subseteq \mathbb{Q}^n$, and a target point $\mathbf{t} \in \mathbb{Q}^n$.
Output: a lattice vector $\mathbf{c}$ such that $\|\mathbf{c} - \mathbf{t}\| = \mathrm{dist}(\mathbf{t}, \Lambda)$.

0. Guess $d_0 \in [d, d/(1-1/n))$ and $\ell_0 \in [\ell, \ell/(1-1/n))$, where $d = \mathrm{dist}(\mathbf{t}, \Lambda)$ and $\ell = \lambda_1(\Lambda)$ (see Section 2.2).

1. Compute p the smallest prime greater than $16n^4 + 2n$.
Sample $\mathbf{z}, \mathbf{u}$ uniformly and independently in $\mathbb{Z}_p^n$.
Compute $\mathbf{w} = \mathbf{B}\bar{\mathbf{u}} \in \Lambda$, such that $\bar{\mathbf{u}} = \mathbf{u} \bmod p$ and $\|\mathbf{t} + \mathbf{w}\| \geq (n+1)\ell_0/\sqrt{2}$.
Use the algorithm of Lemma 2.10 to compute a basis $\mathbf{B}_{p,\mathbf{z}}$ of $\Lambda_{p,\mathbf{z}} = \{\mathbf{b} \in \Lambda : \langle \mathbf{z}, \mathbf{B}^{-1}\mathbf{b} \rangle = 0 \bmod p\}$.

2. Define

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B}_{p,\mathbf{z}} & \mathbf{t} + \mathbf{w} \\ \mathbf{0} & d_0/n \end{pmatrix}.$$

3. Run the uSVP_γ solver on input $\mathbf{B}'$. Let $\mathbf{s}' = ((\mathbf{s}'_1)^{\mathrm{T}}, s'_2)^{\mathrm{T}}$ be its output. Output $\mathbf{s}'_1 + \mathbf{t}$.

It may be checked that the above algorithm runs in polynomial time. The rest of the section is devoted to proving its correctness.

In this reduction, we are given a $\mathrm{BDD}_{(1-1/n)/(\sqrt{2}\gamma)}$ instance $(\mathbf{B}, \mathbf{t})$. Let $\mathbf{c} \in \Lambda$ be a closest vector to $\mathbf{t}$. In order to construct a uSVP instance, our strategy is to use lattice sparsification to keep only one closest vector $\mathbf{c}+\mathbf{w}$ closest to $\mathbf{t}+\mathbf{w}$ for some lattice shift $\mathbf{w}$ (the shift vector $\mathbf{w}$ comes from Lemma 2.9). As the sparsification results in a lattice, we not only keep $\mathbf{c}+\mathbf{w}$, but also the $m\cdot(\mathbf{c}+\mathbf{w})$'s for all m. To make sure that any vector $\mathbf{b}' = ((\mathbf{b}+m\cdot\mathbf{t})^{\mathrm{T}}, md_0)^{\mathrm{T}}$ that is not parallel to $((\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -d_0)^{\mathrm{T}}$ has norm larger than $\lambda_1(\Lambda)/\sqrt{2}$, we require that all other vectors inside the balls with centers $\{m\cdot(\mathbf{t}+\mathbf{w})\}_m$ and radiuses $\sqrt{\lambda_1^2(\Lambda)/2 - m^2d_0^2/n^2}$ are regarded as superfluous vectors and removed through sparsification. The term $m^2d_0^2/n^2$ is contributed by the embedded dimension. It suffices to consider $2\gamma n$ many balls (i.e., $m \le 2\gamma n$). As when $m > 2\gamma n$, the term contributed by embedded dimension is larger than $\lambda_1(\Lambda)/\sqrt{2}$. For the first balls (i.e., m small), we have $m\cdot(\mathbf{c}+\mathbf{w}) \in \mathcal{B}(m\cdot(\mathbf{t}+\mathbf{w}), \sqrt{\lambda_1^2(\Lambda)/2 - m^2d_0^2/n^2})$. We can keep exactly one vector inside every ball with sparsification over these balls. However, for larger m, all lattice vectors $m\cdot(\mathbf{c}+\mathbf{w})$ may fall out of their corresponding balls, but may end up in another relevant ball: vector $m_1\cdot(\mathbf{c}+\mathbf{w})$ may belong to $\mathcal{B}(m_2\cdot(\mathbf{t}+\mathbf{w}), \sqrt{\lambda_1^2(\Lambda)/2 - m_2^2d_0^2/n^2})$ for some $m_1 \neq m_2$. As a consequence, there can be more than one lattice vector inside a ball, which may result in no gap between the first two minima of the uSVP oracle input lattice. In order to avoid this, we make every two balls far away from each other by choosing $\mathbf{w}$ such that $\mathbf{t}+\mathbf{w}$ is long.

Notice that all the arguments above only go through with a polynomially large γ. So far, we still have not explained how we sparsify all $2\gamma n$ balls when γ grows faster than polynomially in n. Indeed, in Lemma 2.9, it is only promised that polynomially many lattice points can be sparsified with non-negligible probability. To handle this, one can observe that the lattice points in nearby balls are not independent: if $\mathbf{x}$ is a lattice point in a ball, and $\mathbf{x}+\mathbf{c}$ is a lattice point in the next ball, then if $\mathbf{c}$ stays and $\mathbf{x}$ goes, $\mathbf{x}+\mathbf{c}$ goes. To handle this, intuitively, we can simultaneously glue all the superfluous lattice points in many balls with points in one specific ball. By doing this repeatedly, it will be eventually sufficient to consider only polynomiallly many such specific balls. This idea that allows to handle large γ is due to Noah Stephens-Davidowitz. In the next lemma, we present our main result, which allows to efficiently sparsify all superfluous lattice vectors (for any $\gamma \ge 1$), but at the same time keep the closest point to the (shifted) target vector.

Lemma 3.2. *Consider a basis $\mathbf{B}$ of an n-dimensional lattice Λ, a vector $\mathbf{c} \in \Lambda$ and a vector $\mathbf{t} \in \mathbb{R}^n$ such that $\|\mathbf{c}-\mathbf{t}\| = d \le (1-1/n)\cdot\lambda_1(\Lambda)/(\sqrt{2}\gamma)$ for some $\gamma \ge 1$. Let $p \ge (2n+1)/\sqrt{2}$ be a prime. For any $\mathbf{z} \in \mathbb{Z}_p^n$, we have*

$$\Pr_{\mathbf{u},\mathbf{z}\hookleftarrow\mathbb{Z}_p^n}\left[\begin{array}{ccc} \mathbf{c}+\mathbf{w} & \in & \Lambda_{p,\mathbf{z}} \cap \mathcal{B}(\mathbf{t}+\mathbf{w}, \gamma\cdot d) \\ \mathbb{Z}\cdot(\mathbf{c}+\mathbf{w}) & \supseteq & \Lambda_{p,\mathbf{z}} \cap \bigcup\limits_{m\le 2\gamma n} \mathcal{B}(m\cdot(\mathbf{t}+\mathbf{w}), \sqrt{\gamma^2d^2 - m^2d_0^2/n^2}) \end{array}\right] \ge \frac{1}{p} - \frac{N}{p^2} - \frac{N}{p^{n-1}},$$

where $\mathbf{w} \in \Lambda$ is arbitrary such that $\mathbf{B}^{-1}\mathbf{w} = \mathbf{u} \bmod p$, $d_0 \in [d, d/(1-1/n))$ and the number of points $N = \#\Lambda \cap \cup_{m\in\mathcal{S}}\mathcal{B}(m\cdot(\mathbf{t}+\mathbf{w}), \lambda_1(\Lambda)/\sqrt{2})$ for $\mathcal{S} = \left\{1+(m-1)\cdot\max(\lfloor\gamma/(4n^2)\rfloor, 1)\right\}_{m\in[8n^3+1]}$.

Further, if $\mathbf{w}$ is chosen such that $\|\mathbf{t}+\mathbf{w}\| > \gamma(n+1)d$, we have, for all $m \in [2\gamma n]$,

$$m\cdot(\mathbf{c}+\mathbf{w}) \notin \bigcup_{m'\neq m, m'\in[2\gamma n]} \mathcal{B}\left(m'\cdot(\mathbf{t}+\mathbf{w}), \gamma\cdot d\right).$$

Proof. Let $\{\mathbf{x}_{m,i}\}_{i\in[N_m]}$ denote the set of lattice pionts in Λ in the ball $\mathcal{B}(m\cdot(\mathbf{t}+\mathbf{w}), \sqrt{\gamma^2d^2 - m^2d_0^2/n^2})$, where N_m denotes the number of lattice vectors in the ball $\mathcal{B}(m\cdot(\mathbf{t}+\mathbf{w}), \sqrt{\gamma^2d^2 - m^2d_0^2/n^2})$ with any $\mathbf{w} \in \Lambda(\mathbf{B})$.

Intuitively, we first divide all balls (that need to be sparsified) into many disjoint sets (each consists of a sequence of successive balls). We show that once we sparsify each representative ball in all sets, all balls should be sparsifid at the same time. Concretely, we show that for each lattice vector $\mathbf{x}_{m,i}$ close to $m\cdot(\mathbf{t}+\mathbf{w})$, we have that $\mathbf{x}_{m,i} - (m-m')\cdot(\mathbf{c}+\mathbf{w})$ is relatively close to $m'\cdot(\mathbf{t}+\mathbf{w})$. Thus

assuming the closest vector $\mathbf{c}+\mathbf{w}$ to $\mathbf{t}+\mathbf{w}$ is kept in the sparsified lattice $\Lambda_{p,\mathbf{z}}$ for some modulus p and vector $\mathbf{z} \hookleftarrow \mathbb{Z}_q^n$, once relatively close lattice points to $m' \cdot (\mathbf{t}+\mathbf{w})$ are removed, all lattice points close to $m \cdot (\mathbf{t}+\mathbf{w})$ will also be removed for free, for any close $m > m'$.

Formally, for $m \geq m' \geq 2$, $m - m' \leq \gamma/(4n^2)$ and $i \in [N_m]$, we have

$$\begin{aligned}
\|\mathbf{x}_{m,i} - (m-m') \cdot (\mathbf{c}+\mathbf{w}) - m' \cdot (\mathbf{t}+\mathbf{w})\|^2 &= \|\mathbf{x}_{m,i} - m \cdot (\mathbf{t}+\mathbf{w})\|^2 + (m-m')^2 \|\mathbf{c}-\mathbf{t}\|^2 \\
&\quad -2(m-m')\langle \mathbf{x}_{m,i} - m \cdot (\mathbf{t}+\mathbf{w}), \mathbf{c}-\mathbf{t}\rangle \\
&\leq \gamma^2 d^2 - m^2 d_0^2/n^2 + (m-m')^2 d^2 + 2(m-m')\gamma d^2 \\
&\leq \gamma^2 d^2 - m^2 d_0^2/n^2 + \gamma^2 d^2/(16n^2) + \gamma^2 d^2/(2n^2) \\
&< \gamma^2 (d^2 + d_0^2/n^2) - m^2 d_0^2/n^2 \\
&< \lambda_1(\mathcal{L})^2/2.
\end{aligned}$$

Now we can first separate the γn many coefficients m into $8n^3$ sets, each with $\max(\lfloor \gamma/(4n^2) - 1 \rfloor, 1)$ many successive coefficients. Note that we need to assume none of these m's is a multiple of p. Thus once it is not for some index m, we shift all the following indices m's to be $m+1$'s such that all the successive indices within a same set have difference not bigger than $\lfloor \gamma/(4n^2) \rfloor$. Thus, if we take out one ball with index m every other $\max(\lfloor \gamma/(4n^2) \rfloor, 1) - 1$ coefficients, there are $2\gamma n / \max(\lfloor \gamma/(4n^2) \rfloor, 1) \leq 8n^3 + 1$ many balls that needed to be sparsified when n is large enough. This is how $\mathcal{S}$ was designed.

Thus, to sparsify all $2\gamma n$ balls, it turns out to be sufficient to sparsify $\#\mathcal{S}$ many balls with index $i \in \mathcal{S}$. According to Lemma 2.11, there are at most $4\gamma n^2$ many points in $\Lambda(\mathbf{B})$ needed to be removed for the first case, while there are $2n \cdot (8n^3+1) = 16n^4 + 2n$ many points for the second case.

The sparsification of these superfluous points proceeds as follows. For $i \in \mathcal{S}$, we define $\{\mathbf{v}_{m,i}\}_{i \in [\widetilde{N}_m]} = (\Lambda \cap \mathcal{B}(m \cdot \mathbf{t}, \gamma \cdot d)) \setminus \{m \cdot \mathbf{c}\}$, where $\widetilde{N}_m = \#\Lambda \cap \mathcal{B}(m \cdot \mathbf{t}, \gamma \cdot d) \setminus \{m \cdot \mathbf{c}\}$. For any $\mathbf{v} \in \Lambda$, we use $\widetilde{\mathbf{v}}$ to denote the coordinate of $\mathbf{v}$ under the basis $\mathbf{B}$. We claim that, with probability $\geq 1/p - N/p^2 - N/p^{n-1}$, the vector $\mathbf{z}$ is orthogonal (modulo p) to $\widetilde{\mathbf{c}}$, and at the same time not orthogonal to any $\widetilde{\mathbf{v}}_{m,i}$ for $m \in \mathcal{S}$ and $i \in [\widetilde{N}_m]$.

We have, for all $m \in \mathcal{S}$ and $i \in [\widetilde{N}_i]$,

$$\|m \cdot \mathbf{c} - \mathbf{v}_{m,i}\| \leq m \cdot \|\mathbf{c}-\mathbf{t}\| + \|m \cdot \mathbf{t} - \mathbf{v}_{m,i}\| \leq (2n+1)(\gamma d) = \frac{n+1}{\sqrt{2}} \cdot \lambda_1(\Lambda).$$

By choice of p, this is smaller than $p \cdot \lambda_1(\Lambda)$. Thanks to Lemma 3.1, we have $m \cdot \widetilde{\mathbf{c}} \neq \widetilde{\mathbf{v}}_{m,i} \bmod p$. Moreover, as p is prime and $m \not\equiv 0 \mod p$, we have $\widetilde{\mathbf{c}} \neq \frac{1}{m} \cdot \widetilde{\mathbf{v}}_{m,i} \bmod p$.

Now we apply Lemma 2.9 with $\widetilde{\mathbf{c}}$ and $\left\{\frac{1}{m} \cdot \widetilde{\mathbf{v}}_{m,i}\right\}_{m \in \mathcal{S}, i \in [\widetilde{N}_m]}$. We have

$$\Pr_{\mathbf{z},\mathbf{u} \hookleftarrow U(\mathbb{Z}_p^n)} \left[\begin{array}{lll} \forall m,i: & \langle \mathbf{z}, \frac{1}{m} \cdot \widetilde{\mathbf{v}}_{m,i} + \mathbf{u} \rangle & \neq 0 \bmod p \\ & \langle \mathbf{z}, \widetilde{\mathbf{c}} + \mathbf{u} \rangle & = 0 \bmod p \end{array} \right] \geq \frac{1}{p} - \frac{N}{p^2} - \frac{N}{p^{n-1}}.$$

As p is prime and sufficiently large, the inequality $\langle \mathbf{z}, \frac{1}{m} \cdot \widetilde{\mathbf{v}}_{m,i} + \mathbf{u} \rangle \neq 0 \bmod p$ is equivalent to $\langle \mathbf{z}, \widetilde{\mathbf{v}}_{m,i} + m \cdot \mathbf{u} \rangle \neq 0 \bmod p$. Therefore

$$\Pr_{\mathbf{z},\mathbf{u} \hookleftarrow U(\mathbb{Z}_p^n)} \left[\begin{array}{lll} \forall m,i: & \langle \mathbf{z}, \widetilde{\mathbf{v}}_{m,i} + m \cdot \mathbf{u} \rangle & \neq 0 \bmod p \\ & \langle \mathbf{z}, \widetilde{\mathbf{c}} + \mathbf{u} \rangle & = 0 \bmod p \end{array} \right] \geq \frac{1}{p} - \frac{N}{p^2} - \frac{N}{p^{n-1}}.$$

This proves the first claim of the lemma.

Let $i \neq j \leq \gamma n$. Then, by the triangle inequality and the assumption on $\mathbf{w}$, we have:

$$\|i \cdot (\mathbf{c}+\mathbf{w}) - j \cdot (\mathbf{t}+\mathbf{w})\| \geq |j-i| \cdot \|\mathbf{t}+\mathbf{w}\| - i\|\mathbf{c}-\mathbf{t}\| > \gamma(n+1)d - (\gamma n)d = \gamma d.$$

This completes the proof of the lemma. □

As we have $p > 2N$ (thanks to Lemma 2.11), with non-negligible probability, none of the vectors of Λ belonging to the γn balls is in the sparser lattice $\Lambda_{p,\mathbf{z}}$, except possibly those in $\{i \cdot (\mathbf{c}+\mathbf{w})\}_{i\in[\gamma n]}$. In the rest of the reduction analysis, we assume that we are in this situation and do not repeatedly state that this occurs with non-negligible probability.

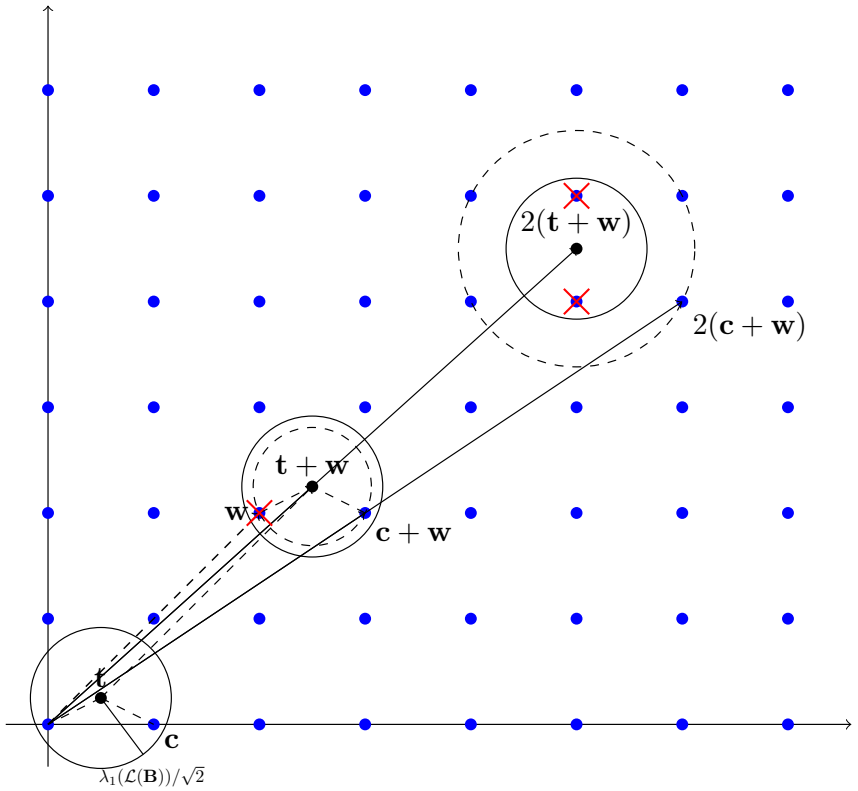

Fig. 3.3: Sparsification process for a $\mathrm{BDD}_{1/(2\sqrt{2})}$ instance.

As an illustration of Lemma 3.2, we include Figure 3.3 and Figure 3.4. First, we use the case of $\gamma = 1$ (Figure 3.3) to illustrate the case of small γ, in which there is only one ball in each set of balls. There are several plain balls with radius $\lambda_1(\Lambda)/\sqrt{2}$, centered in $\mathbf{t}$, $\mathbf{t}+\mathbf{w}$ and $2(\mathbf{t}+\mathbf{w})$. The dashed balls illustrate the distance between $i \cdot (\mathbf{c}+\mathbf{w})$ and $i \cdot (\mathbf{t}+\mathbf{w})$ for all $i \in [\gamma n]$. We can see that $\mathbf{c}+\mathbf{w}$ (within the dashed ball) is inside the plain ball, and $2 \cdot (\mathbf{c}+\mathbf{w})$ (within the dashed ball) is outside of its corresponding plain ball. In this case, we will sparsify all the lattice points except multiples of $\mathbf{c}+\mathbf{w}$ within all $2\gamma n$ balls with radius $\lambda_1(\Lambda)/\sqrt{2}$. This is also sufficient for the reduction.

In contrast with the case of small γ, we are considering a ball chain with decreasing radius in each set of balls for large γ. We will use the case of $\gamma = 2$ (Figure 3.4) to explain this case, in which there are three balls in one set. We have superfluous lattice vectors $\mathbf{p}_2$, $\mathbf{p}_3$ in the second and third balls respectively, which implies a superfluous point $\mathbf{p}_1$ in the first ball by shifting with multiple of $\mathbf{c}+\mathbf{w}$ in the converse direction. Thus by sparsifying the first ball, we simultaneously sparsify the second and the third one. Note in particular that in the case of $\gamma = 2$, the vector $\mathbf{0}$ is not closest to the target vector, but belongs to the plain ball with center $\mathbf{t}$. Thus we have a valid uSVP instance, and vector $\mathbf{0}$ should be removed via sparsification. As it is kept in any sparsified lattice, this is impossible to achieve. This illustrates why center $\mathbf{t}$ is shifted to a new point $\mathbf{t}+\mathbf{w}$ (then the shift $\mathbf{w}$ of $\mathbf{0}$ may be removed via

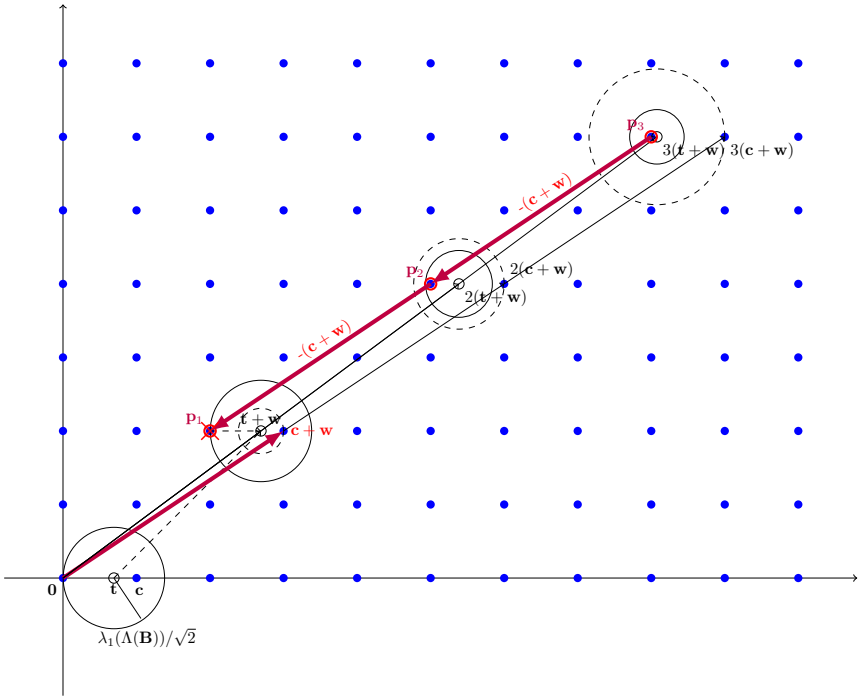

Fig. 3.4: Sparsification process for a $\mathrm{BDD}_{1/\sqrt{2}}$ instance.

sparsification). In both figures, the red crosses denote the points that are removed from the lattice via sparsification.

In Step 2 of the reduction, we construct a basis $\mathbf{B}'$ of an $(n+1)$-dimensional lattice Λ' by using Kannan's embedding technique. In Step 3, we call the uSVP_γ oracle with input basis $\mathbf{B}'$. The correctness of the reduction is provided by Lemmata 3.3, 3.4 and 3.5.

Any vector in lattice Λ' can be written as $\mathbf{b}' = ((\mathbf{b} + m(\mathbf{t} + \mathbf{w}))^{\mathrm{T}}, md_0/n)^{\mathrm{T}}$ with $\mathbf{b} \in \Lambda_{p,\mathbf{z}}$ and $m \in \mathbb{Z}$. We claim that the vector $\mathbf{s}' = (((\mathbf{c} + \mathbf{w}) - (\mathbf{t} + \mathbf{w}))^{\mathrm{T}}, -d_0/n)^{\mathrm{T}} = ((\mathbf{c} - \mathbf{t})^{\mathrm{T}}, -d_0/n)^{\mathrm{T}}$ is a shortest non-zero vector in Λ' and also that $\lambda_2(\Lambda')/\lambda_1(\Lambda') = \gamma(1 + \Omega(1/n))$. Thus $\pm\mathbf{s}'$ will be output by the uSVP_γ oracle. We can then obtain the vector $\mathbf{c} = (\mathbf{c} + \mathbf{w}) - (\mathbf{t} + \mathbf{w}) + \mathbf{t} \in \Lambda$. In the following, we give lower bounds for the norm of $\mathbf{b}' = ((\mathbf{b} + m(\mathbf{t} + \mathbf{w}))^{\mathrm{T}}, md_0/n)^{\mathrm{T}}$ not parallel to $\mathbf{s}'$, which depend on the value of m. Without loss of generality, we restrict ourselves to $m \geq 0$.

The following lemma is analogous to the '$m = 0$ case' of the Lyubashevsky-Micciancio reduction [LM09]. Note that the lower bound in the statement is essentially $2\gamma^2$.

Lemma 3.3. *If* $m = 0$ *and* $\mathbf{b}' \neq \mathbf{0}$, *then* $\|\mathbf{b}'\|^2/\|\mathbf{s}'\|^2 \geq 2\gamma^2/(1 + 1/n^2)$.

Proof. As $m = 0$ and $\mathbf{b}' \neq \mathbf{0}$, we must have $\mathbf{b} \neq \mathbf{0}$. As a result, we have

$$\|\mathbf{b}'\|^2 = \|\mathbf{b}\|^2 \geq \lambda_1^2(\Lambda) \geq \frac{2\gamma^2 d^2}{(1 - \frac{1}{n})^2} \geq 2\gamma^2 d_0^2.$$

Thus, in this case, we have the gap

$$\frac{\|\mathbf{b}'\|^2}{\|\mathbf{s}'\|^2} \geq \frac{2\gamma^2 d_0^2}{d^2 + \frac{d_0^2}{n^2}} \geq \frac{2\gamma^2}{1 + \frac{1}{n^2}}.$$

□

The second lemma bounds the gap for small m's. It is where our improvement over prior reductions stems from. Note that the lower bound in the statement is essentially γ^2.

Lemma 3.4. *If $m \leq 2\gamma n$ and $\mathbf{b}'$ is linearly independent with $\mathbf{s}'$, then $\|\mathbf{b}'\|/\|\mathbf{s}'\| > \gamma$.*

Proof. By Lemma 3.2, we have

$$\mathbf{c}+\mathbf{w} \;\in\; \Lambda_{p,\mathbf{z}} \bigcap \bigcup_{m \leq 2\gamma n} \mathcal{B}\left(i \cdot (\mathbf{t}+\mathbf{w}), \sqrt{\frac{\lambda_1^2(\Lambda)}{2} - \frac{m^2 d_0^2}{n^2}} \right) \;\subseteq\; \mathbb{Z} \cdot (\mathbf{c}+\mathbf{w}).$$

Thus, as $\mathbf{b} \notin \mathbb{Z} \cdot (\mathbf{c}+\mathbf{w})$ (by assumption), we have

$$\|\mathbf{b}'\|^2 = \|\mathbf{b} - m \cdot (\mathbf{t}+\mathbf{w})\|^2 + \frac{m^2 d_0^2}{n^2} > \frac{\lambda_1(\Lambda)^2}{2}.$$

Thus, in this case, we have the gap

$$\frac{\|\mathbf{b}'\|^2}{\|\mathbf{s}'\|^2} > \frac{\frac{\lambda_1(\Lambda)^2}{2}}{\frac{\lambda_1(\Lambda)^2}{2\gamma^2}} = \gamma^2.$$

□

The third lemma bounds the gap for larger m's. This corresponds to the 'large m case' of the Lyubashevsky-Micciancio reduction. As in the previous case, the lower bound in the statement is essentially $4\gamma^2$.

Lemma 3.5. *If $m > 2\gamma n$, then $\|\mathbf{b}'\|^2/\|\mathbf{s}'\|^2 \geq 4\gamma^2/(1+1/(n-1)^2)$.*

Proof. For any $\mathbf{b} \in \Lambda$, we have $\|\mathbf{b}'\|^2 \geq m^2 d_0^2/n^2$. Thus, in this case, we have the gap

$$\frac{\|\mathbf{b}'\|^2}{\|\mathbf{s}'\|^2} \geq \frac{\frac{m^2 d_0^2}{n^2}}{d^2 + d_0^2/n^2} \geq \frac{\frac{m^2 d^2}{n^2}}{d^2 + \left(\frac{d}{n(1-\frac{1}{n})}\right)^2} = \frac{m^2}{n^2 + \frac{n^2}{(n-1)^2}}.$$

The gap is an increasing function in m and hence it suffices to consider the $m = 2\gamma n$. □

Now, we can complete the proof of Theorem 3.2. Thanks to Lemmata 3.3, 3.4 and 3.5, the uSVP gap satisfies, for large enough n

$$\frac{\lambda_2(\Lambda')}{\lambda_1(\Lambda')} > \gamma.$$

Thus the uSVP oracle returns a vectors, from which we can compute the solution to the initial BDD instance. We include Figure 3.5 to geometrically illustrate the overall reduction. For convenience, we take the embedding height as $-d_0/n$ in the figure. We use filled dots to label points of 2-dimensional lattice Λ, and hollow dots to label points of 3-dimensional lattice Λ' that are not in Λ (recall that $\Lambda \subseteq \Lambda'$). With Kannan's embedding technique, the offset between the vectors of Λ (e.g., $\mathbf{c}+\mathbf{w}$) and the shifted target $\mathbf{t}+\mathbf{w}$ are mapped to Λ' (e.g., $((\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -d_0/n)^{\mathrm{T}}$). Thanks to sparsification, all the points of Λ' belonging to the drawn cylinder (of height $|2\gamma n d_0/n|$) are multiples of the shortest non-zero vector $\mathbf{s}' = ((\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -d_0/n)^{\mathrm{T}}$, e.g., $\pm((\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -d_0/n)^{\mathrm{T}}$ and $\pm(2(\mathbf{c}-\mathbf{t})^{\mathrm{T}}, -2d_0/n)^{\mathrm{T}}$. All other points in Λ' that are linearly independent from $\mathbf{s}'$ lie outside of the cylinder, e.g., $((\mathbf{a}_1 - (\mathbf{t}+\mathbf{w}))^{\mathrm{T}}, -d_0/n)^{\mathrm{T}}$ and $((\mathbf{a}_2 - (\mathbf{t}+\mathbf{w}))^{\mathrm{T}}, -2d_0/n)^{\mathrm{T}}$. This cylinder forces the second minimum $\lambda_2(\Lambda')$ to be large, and, more concretely, larger than $\gamma\lambda_1(\Lambda')$. This corresponds to Lemma 3.4 (Lemma 3.3 handles the points of Λ and Lemma 3.5 handles the points of Λ' whose z-component is large).

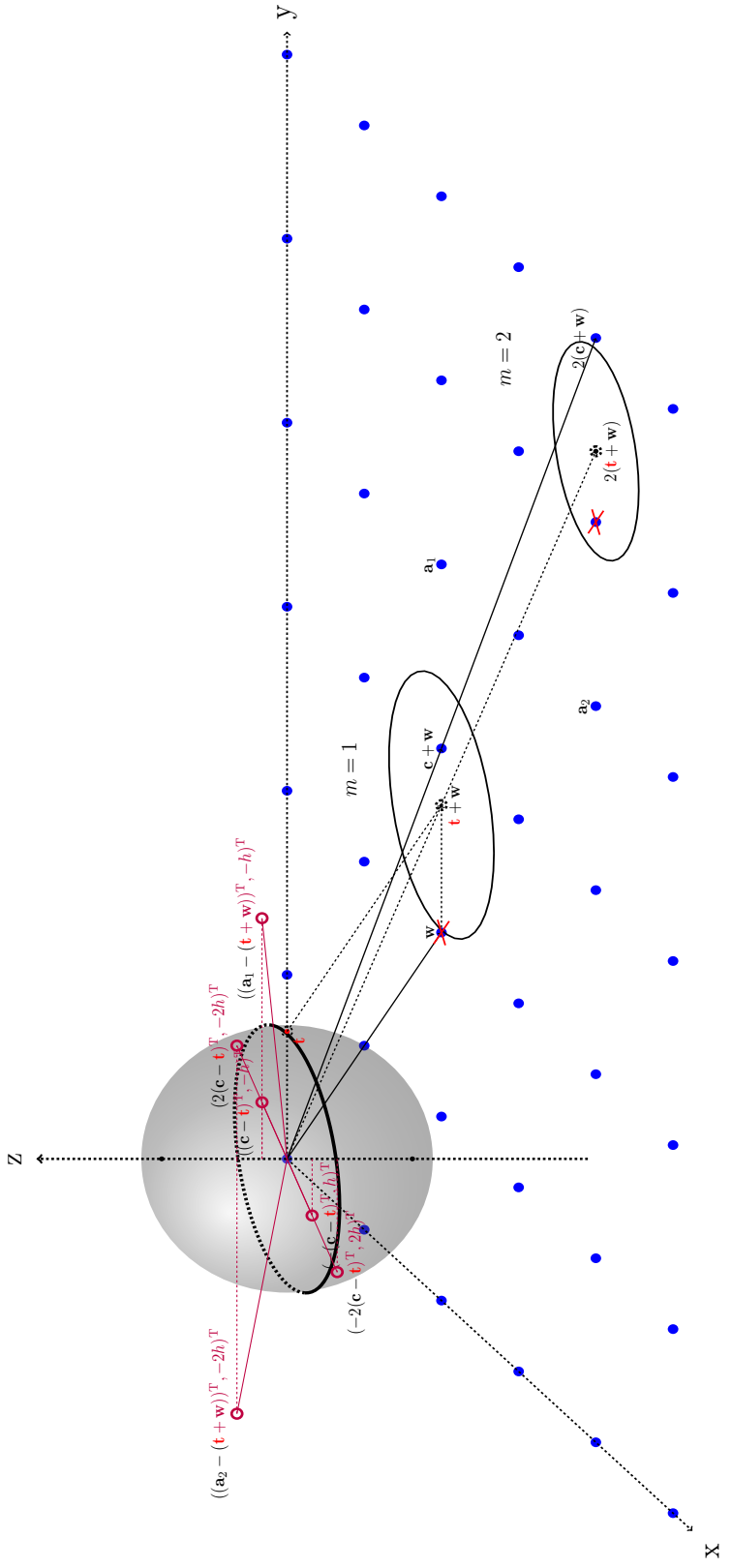

Fig. 3.5: Geometric illustration of the reduction.

Chapter 4

A computational equivalence between LWE and an extrapolated variant of DCP

In this chapter, we first review Regev's reduction [Reg02] from the lattice problem uSVP to the presumed hard quantum problem dihedral coset problem (DCP). Then we extend this work and give a reduction from the learning with errors (LWE) problem to an extrapolated version of the DCP problem (called EDCP). We further show that a converse reduction from EDCP to LWE also exists, which gives the claimed computational equivalence (up to small noise amplification). Correspondingly, we also show that the decision versions of these two problems are equivalent to each other.

Contents

4.1 Introduction

Since its introduction by Regev [Reg05, Reg09], LWE has served as a security foundation for numerous cryptographic primitives (see, e.g., an overview in [Pei16]). The cryptographic attractiveness of LWE stems from two particularly desirable properties. First, its algebraic simplicity enables the design of primitives with advanced functionalities, such as fully homomorphic encryption [BV11], attribute-based encryption for all circuits [GVW13] and (single key) functional encryption [GKP+13]. Second, LWE is conjectured hard even in the context of quantum computations, making it one of the most appealing candidate security foundations for post-quantum cryptography [BBD08]. Current quantum algorithms for LWE do not outperform classical ones, but it is not clear whether this is inherent (for example, it is likely that LWE is computationally *easier* than the Dihedral Coset Problem under polynomial-time reductions, see below). In this work, we characterize the quantum hardness of LWE under polynomial-time reductions and show that it is computationally equivalent (up to small parameter losses) to a quantum problem closely related to the aforementioned Dihedral Coset Problem.

LWE, Lattices and the Dihedral Coset Problem. LWE is tightly connected to worst-case approximation problems over Euclidean lattices. In particular, LWE is an (average-case) instance of the BDD (refer to Section 2.3), but is also known to be quantumly as hard as *worst-case* BDD (with some small losses in parameters) [Reg09]. Regev [Reg02, Reg04b] showed that uSVP, and therefore also BDD and LWE, are no harder to solve than the quantumly-defined DCP. Still, it is quite possible that DCP is in fact much harder to solve than LWE. The best known algorithm for DCP, due to Kuperberg [Kup05], runs in time $2^{\mathcal{O}(\log \ell + \log N/\log \ell)}$, where ℓ is the number of DCP samples (see Section 2.4 for more details). This does not improve upon classical methods for solving LWE [ACF+15, APS15]. Other variants of the problem were explored in [EH99, FIM+14], and of particular relevance to this work is a "vector" variant of the problem where $\mathbb{Z}_N$ is replaced with $\mathbb{Z}_q^n$. These problems have difficulty similar to DCP with $N = q^n$, with respect to current best known solving algorithm. Finally, Regev showed that DCP can be solved given efficient algorithms for the subset-sum problem (which is classically defined), however in a regime of parameters that appears harder to solve than LWE itself.

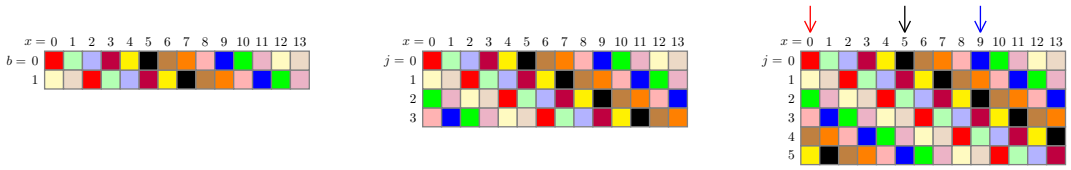

Fig. 4.1: An illustration of one-dimensional U-EDCP with number of possibilities 2, 4 and 6, and modulus 14.

Extrapolated DCP. The focus of this chapter is a generalization of DCP: rather than considering registers containing $|0, x_k\rangle + |1, x_k + s\rangle$, we allow (1) the x_i's and s to be n-dimensional vectors, and (2) non-uniform distribution, for amplitudes. We name this problem Extrapolated DCP (EDCP) as its input registers have more extrapolated states. To be more precise, $\mathrm{EDCP}^{\ell}_{n,N,f}$, with parameters three integers n, N, ℓ and a function $f : \mathbb{Z} \mapsto \mathbb{C}$ with $\sum_{j\in\mathbb{Z}} j \cdot |f(j)|^2 < +\infty$, consists in recovering $\mathbf{s} \in \mathbb{Z}_N^n$ from the following ℓ states over $\mathbb{Z} \times \mathbb{Z}_N^n$:

$$\left\{ \frac{1}{\sqrt{\sum_{j\in\mathbb{Z}} |f(j)|^2}} \cdot \sum_{j\in\mathbb{Z}} f(j)\, |j, \mathbf{x}_k + j \cdot \mathbf{s}\rangle \right\}_{k\leq \ell},$$

where the $\mathbf{x}_k$'s are arbitrary in $\mathbb{Z}_N^n$.[1] Note that DCP is the special case of EDCP for $n = 1$ and f being the indicator function of $\{0, 1\}$.

In [CvD07], Childs and van Dam consider a special case of EDCP where f is the indicator function of $\{0, \ldots, M-1\}$ for some integer M, which we will refer to as uniform EDCP (or, U-EDCP$^{\ell}_{n,N,M}$). As illustrated in Figure 4.1, from left to right, we have U-EDCP defined with number of possibilities 2, 4 and 6 respectively. In the case of $M = 2$, U-EDCP corresponds to DCP. A superposition of all labels of squares with a same color corresponds to a U-EDCP sample. In particular, a randomly selected set of sample registers of the U-EDCP$^{3}_{1,14,6}$ (on the right) can be (up to normalization)

1. $|0\rangle|0\rangle + |1\rangle|2\rangle + |2\rangle|4\rangle + |3\rangle|6\rangle + |4\rangle|8\rangle + |5\rangle|10\rangle$ (in red color);
2. $|0\rangle|5\rangle + |1\rangle|7\rangle + |2\rangle|9\rangle + |3\rangle|11\rangle + |4\rangle|13\rangle + |5\rangle|1\rangle$ (in black color);
3. $|0\rangle|9\rangle + |1\rangle|11\rangle + |2\rangle|13\rangle + |3\rangle|1\rangle + |4\rangle|3\rangle + |5\rangle|5\rangle$ (in blue color).

Once we measure on any of these input states, each possibility (as label of some square) will be observed with equal probability, e.g., $|1\rangle|2\rangle$ will be observed with probability 1/6 if we measure the first input state.

4.2 Our contributions

We show that up to polynomial losses in parameters, U-EDCP is computational equivalent to LWE. We thus provide a formulation of the hardness assumption underlying lattice-based cryptography in terms of the (generalized) Dihedral Coset Problem.

Theorem 4.1 (Informal)**.** *There exists a reduction from* LWE$_{n,q,\alpha}$ *to* U-EDCP$^{\ell}_{n,N,M}$ *under quantum polynomial-time, with* $N = q$, $\ell = \mathrm{poly}(n \log q)$ *and* $M = \frac{\mathrm{poly}(n \log q)}{\alpha}$*. Conversely, there exists a polynomial-time reduction from* U-EDCP *to* LWE *with the same parameter relationships, up to* $\mathrm{poly}(n \log q)$ *factors.*

Our proof crucially relies on a special case of EDCP where f is a Gaussian weight function with standard deviation parameter r. We call this problem Gaussian EDCP (G-EDCP). We show that G-EDCP and U-EDCP are equivalent up to small parameter losses.

EDCP is analogous to LWE in many aspects. The decision version of LWE (dLWE) asks to distinguish between LWE samples and random samples of the form $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where both components are chosen uniformly at random. Similarly, we also consider the decision version of EDCP, denoted by dEDCP. In dEDCP$_{n,N,f}$, we are asked to distinguish between an EDCP state and a state of the form

$$|j\rangle\,|\mathbf{x} \bmod N\rangle\,,$$

where j is distributed according to the function $|f|^2$, and $\mathbf{x} \in \mathbb{Z}_N^n$ is uniformly chosen. EDCP enjoys a reduction between its search and decision variants via LWE. As shown in Figure 4.3, we can first reduce search EDCP to search LWE, the latter one can then be reduced to decision LWE. Finally, we can reduce decision LWE to decision EDCP (as a corollary of the reduction from search LWE to search EDCP).

Related work. In [CvD07], Childs and van Dam show that U-EDCP$^{\ell}_{1,N,M}$ reduces to the problem of finding all the solutions $\mathbf{b} \in \{0, \ldots, M-1\}^k$ to the equation $\langle \mathbf{b}, \mathbf{x} \rangle = w \bmod N$, where $\mathbf{x}$ and w are given and uniformly random modulo N. They interpret this as an integer linear program and use lattice reduction, within Lenstra's algorithm [Len83], to solve it. This leads to a polynomial-time

[1] Note that the assumption on f implies, via Markov's inequality, that one may restrict the sum to a finite index set and obtain a superposition which remains within negligible ℓ_2-distance from the countable superposition above.

algorithm for U-EDCP$^{\ell}_{1,N,M}$ when $M = \lfloor N^{1/k} \rfloor$ and $\ell \geq k$, for any $k \geq 3$. Interestingly, finding small solutions to the equation $\langle \mathbf{b}, \mathbf{x} \rangle = w \bmod N$ is a special case of the Inhomogeneous Small Integer Solution problem [GPV08] (ISIS), which consists in finding a small-norm $\mathbf{x}$ such that $\mathbf{Bx} = \mathbf{w} \bmod q$, with $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{w} \in \mathbb{Z}_q^n$ uniform (where q, n, m are integer parameters). A reduction from the homogeneous SIS (i.e., with $\mathbf{w} = \mathbf{0}$ and $\mathbf{x} \neq \mathbf{0}$) to LWE was provided in [SSTX09]. It does not seem possible to derive from it a reduction from EDCP to LWE via the Childs and van Dam variant of ISIS, most notably because the reduction from [SSTX09] does not provide a way to compute all ISIS solutions within a box $\{0, 1, \ldots, M-1\}^k$.

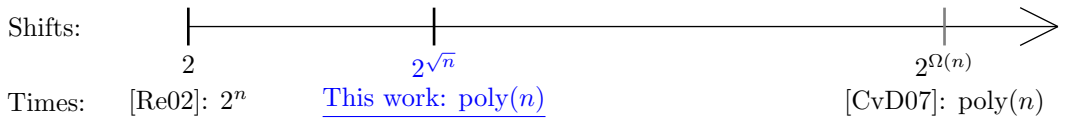

Fig. 4.2: A comparison between our solver (via our LWE to EDCP reduction and solving LWE by the LLL algorithm) and the Childs and van Dam's solver for solving EDCP.

It is not hard to see that, at least so long as M is polynomial, a solution to DCP implies a solution to EDCP$^{\ell}_{1,N,M}$. Without loss of generality, we consider U-EDCP$_{1,N,M}$, and briefly show that any sample of U-EDCP$_{1,N,M}$ with M is polynomial in $\log N$, can be efficiently transformed to a DCP$_N$ sample. Suppose we are given (up to normalization) $\sum_{j \in [M]} |j\rangle |x + j \cdot s\rangle$. We compute the absolute value of the first register and store it in a new register: $\sum_{j \in [M]} |j\rangle |x + j \cdot s\rangle ||j|\rangle$. Then we measure the third register and we observe $|j| = 1$ with non-negligible probability. In the case of $|j| = 1$, our state becomes: $|-1\rangle |x - s\rangle + |1\rangle |x + s\rangle$. Then we apply a function h that maps -1 to 0 and maps 1 to 1 on the first register (one can check that it is reversible) and replace $x - s$ by a new $\widehat{x}$, and obtain the DCP sample: $|0\rangle |\widehat{x}\rangle + |1\rangle |\widehat{x} + \widehat{s}\rangle$ with $\widehat{s} = 2s$. Once $\widehat{s} = 2s$ is solved, we output any of two possibilities of s with equal probability, which is then the solution to U-EDCP$_{1,N,M}$ with 1/2 probability. Therefore our result implies [Reg02] as a special case.

On the other extreme, our result also subsumes [CvD07] since the LLL algorithm [LLL82] can be used to solve LWE$_{n,q,\alpha}$ in polynomial time when $1/\alpha$ and q are $2^{\Theta(n)}$, which implies a polynomial-time algorithm for EDCP for $M = 2^{\Theta(\sqrt{n \log N})}$, significantly improving Childs and van Dam's $M = 2^{\varepsilon n \log N}$ for any constant $\epsilon \in (0, 1]$.

Finally, we observe that the LWE to U-EDCP reduction (and the uSVP to DCP reduction [Reg04b]) can be adapted to a uSVP to U-EDCP reduction, as explained below. Combining this adaptation with the reduction from U-EDCP to LWE (via G-EDCP) provides a novel quantum reduction from worst-case lattice problems to LWE. However, it does not seem to have advantages compared to [Reg09].

4.2.1 Technical overview

As mentioned above, the hardness of LWE is essentially invariant so long as $n \log q$ is preserved, and therefore we restrict our attention in this overview to the one-dimensional setting. A crucial ingredient in our reduction is a weighted version of EDCP, denoted by G-EDCP and quantified by a Gaussian weight function $f_r(j) = \rho_r(j) = \exp(-\pi j^2/r^2)$, for some standard deviation parameter r. We refer to this problem as Gaussian EDCP (G-EDCP).

Reducing G-EDCP to LWE. Given a G-EDCP state as input, our reduction efficiently transforms it into a classical LWE sample with constant success probability. Thus, making only one query to the LWE oracle, we are able to solve G-EDCP. More precisely, the reduction input consists of a normalized

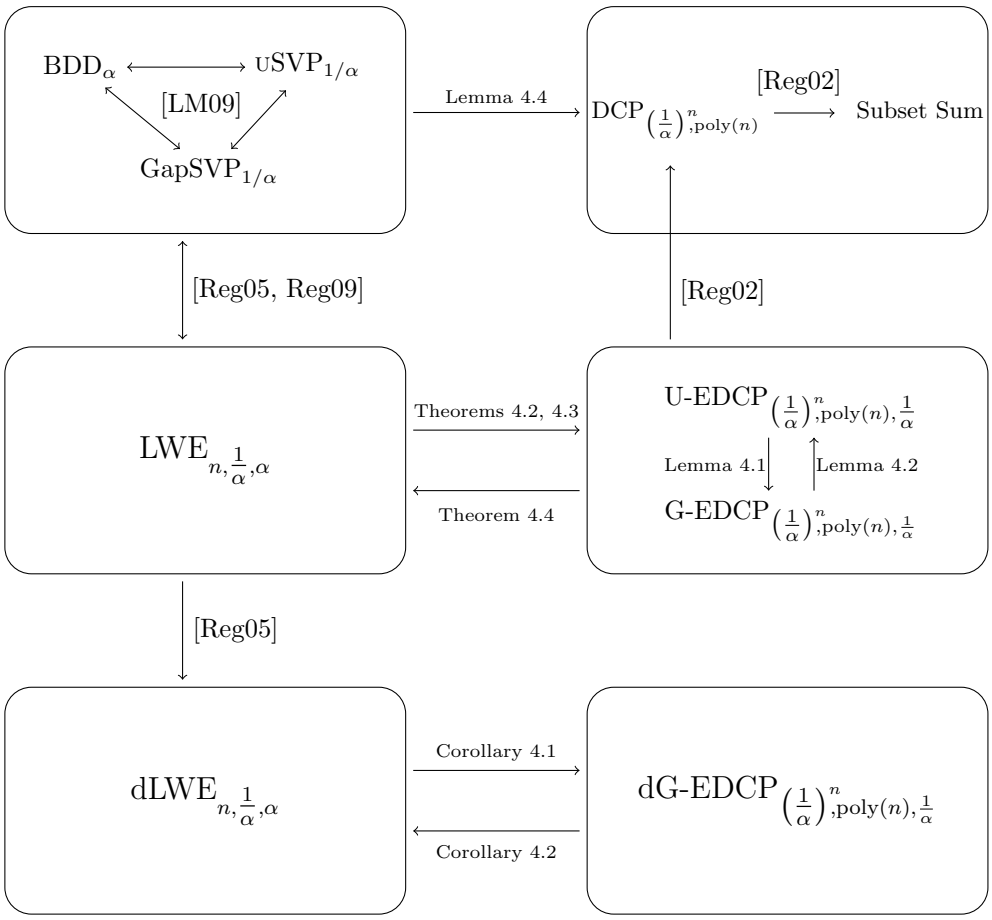

Fig. 4.3: Graph of reductions between the LWE problem (search version on middle-left, decision version on lower-left), the Extrapolated Dihedral Coset problems (search version on middle-left, decision version on lower-left), chosen worst-case lattice problems (upper-left), and DCP (upper-right). Parameters α are given up to $\mathrm{poly}(n)$-factors, where n is the dimension of the LWE problem. The same n stands for the lattice-dimension considered in problems of the upper-left corner. The subset-sum problem stated in the upper-right corner is of density ≈ 1 (in particular, the expected number of solutions is constant).

state corresponding to $\sum_{j\in\mathbb{Z}_N} \rho_r(j)\,|j\rangle\,|x + j\cdot s \bmod N\rangle$, for some integers $r \ll N$. One can think of N as the LWE modulus and of N/r as the standard deviation parameter of the LWE error.

Our first step is to apply a quantum Fourier transform over $\mathbb{Z}_N$ to the second register. This gives us a quantum superposition of the form:

$$\sum_{a\in\mathbb{Z}_N}\sum_{j\in\mathbb{Z}_N} \omega_N^{a\cdot(x+j\cdot s)}\cdot\rho_r(j)\,|j\rangle\,|a\rangle\,.$$

where $\omega_N = \exp(2i\pi/N)$. We then measure the second register and obtain a value $\widehat{a}\in\mathbb{Z}_N$. This leaves us with the state:

$$\sum_{j\in\mathbb{Z}_N} \omega_N^{j\cdot\widehat{a}\cdot s}\cdot\rho_r(j)\,|j\rangle\,|\widehat{a}\rangle\,.$$

Note that $\widehat{a}$ is uniformly random over $\mathbb{Z}_N$, which at the end serves as the first component of LWE sample. The exponent of relative phase in current state has a form similar to the second component of LWE sample but without noise. Now we can benefit from the first register, which stores a superposition corresponding to a Gaussian distribution over $\mathbb{Z}_N$ with standard deviation r. Applying a second quantum Fourier transform over $\mathbb{Z}_N$ to the first register gives us a quantum superposition of the form:

$$\sum_{b\in\mathbb{Z}_N}\sum_{j\in\mathbb{Z}_N} \omega_N^{j\cdot(\widehat{a}\cdot s+b)}\cdot\rho_r(j)\,|b\rangle\,.$$

Now the second component of the LWE sample $\widehat{a}\cdot s + b$ is stored in the phase (up to a factor j). Omitting the exponentially small Gaussian tail, we assume the summation for j is taken over the integers. An application of the Poisson summation formula transfers $\widehat{a}\cdot s + b$ into a shift of the Gaussian distribution defined over $\mathbb{Z}$. In other words, the received state is exponentially close to the superposition:

$$\sum_{e\in\mathbb{Z}_N} \rho_{1/r}\Big(\frac{e}{N}\Big)\,|-\widehat{a}\cdot s + e\rangle\,.$$

Once we measure the state above, we obtain a value $-\widehat{a}\cdot s + e$, where $e \hookleftarrow \mathcal{D}_{\mathbb{Z},N/r}$. Together with already known $\widehat{a}$, this gives us an LWE sample:

$$(-\widehat{a}, -\widehat{a}\cdot s + e)\,.$$

In case the input state is of the form $|j\rangle\,|x \bmod N\rangle$, where j is distributed according to the function ρ_r^2, and $x\in\mathbb{Z}_N$ is uniformly chosen (the decision case), the reduction outlined above outputs a uniform random pair (a, b) from $\mathbb{Z}_N\times\mathbb{Z}_N$. This gives a reduction from decision G-EDCP to decision LWE.

Reducing LWE to G-EDCP. Our reduction from LWE to G-EDCP follows the general design of Regev's reduction from uSVP to DCP [Reg04b], with several twists that enable simplifications and improvements. We note that this reduction is folklore,[2] although we could not find it described explicitly.

First, the use of LWE rather than uSVP allows us to avoid Regev's initial sub-reduction from uSVP to BDD, as LWE is a randomized variant of BDD. Indeed, if we consider m samples $(\mathbf{a}_i, \mathbf{a}_i\cdot\mathbf{s} + e_i)$ from $\mathrm{LWE}_{n,q,\alpha}$, then we have a BDD instance for the lattice $\Lambda = \mathbf{A}\mathbb{Z}_q^n + q\mathbb{Z}^m$ and the target vector $\mathbf{t} = \mathbf{b} + \mathbf{e}\in\mathbb{Z}^m$ with $\mathbf{b}\in\Lambda$ satisfying $\mathbf{b} = \mathbf{As} \bmod q$.

As Regev's, our reduction proceeds by subdividing the ambient space $\mathbb{R}^m$ with a coarse grid, setting the cell width between $\|\mathbf{e}\|$ and $\lambda_1(\Lambda)$. We map each point $\mathbf{y}\in\mathbb{R}^m$ to a cell $\phi(\mathbf{y})$. By choice of the cell width, we have $\phi(\mathbf{c}_1)\neq\phi(\mathbf{c}_2)$ for any $\mathbf{c}_1\neq\mathbf{c}_2$ in Λ. Also for any $\mathbf{c}\in\mathbb{R}^m$, the vectors $\mathbf{c}$ and $\mathbf{c}+\mathbf{e}$ are most likely mapped to the same cell, as $\mathbf{e}$ is short. This intuition fails if a border between two cells falls

[2] `https://groups.google.com/d/msg/cryptanalytic-algorithms/uhr6gGrVkIk/XxEv4uvEBwAJ`

close to $\mathbf{c}$ (see the grid intersects the left-most disk in Figure 4.4). This (rare but non-negligibly so) event is the source of the limitation on the number ℓ of DCP/EDCP states produced by the reduction. The space subdivision by a grid is illustrated in Figure 4.4.

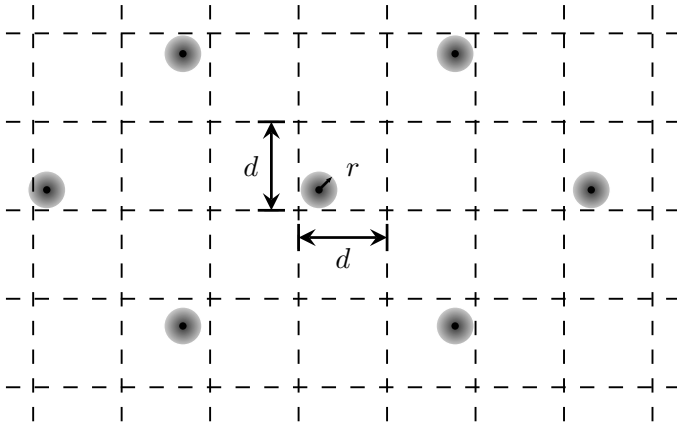

Fig. 4.4: A visualization of the space subdivision. Each radially shaded disk has width r, the upper bound of the error $\|\mathbf{e}\|$. Each cell has width d, chosen to be between $\|\mathbf{e}\|$ and $\lambda_1(L)/\sqrt{m}$. Note that the grid intersects the left-most disk, potentially leading to an error in the reduction.

Regev's reduction and ours differ in the way the grid is used to create the DCP/EDCP states. Let us first briefly recall the core of Regev's reduction. Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)$ be a basis of Λ and subtract an appropriate combination of the $\mathbf{b}_i$'s from $\mathbf{t}$ to get $\mathbf{t}'$ so that the coordinates $\mathbf{x}'$ of the closest vector $\mathbf{b}' \in \Lambda$ to $\mathbf{t}'$with respect to the $\mathbf{b}_i$'s are $\leq 2^m$ (this may be achieved using LLL [LLL82] and Babai's nearest plane algorithm [Bab86]). The first step is the creation of a superposition

$$\sum_{\substack{\mathbf{x}\in\mathbb{Z}^m \\ \|\mathbf{x}\|_\infty \leq 2^{2m}}} \left(|0, \mathbf{x}, \phi(\mathbf{B}\mathbf{x})\rangle + |1, \mathbf{x}, \phi(\mathbf{B}\mathbf{x} - \mathbf{t}')\rangle\right) =$$

$$|0\rangle \sum_{\substack{\mathbf{x}\in\mathbb{Z}^m \\ \|\mathbf{x}\|_\infty \leq 2^{2m}}} |\mathbf{x}, \phi(\mathbf{B}\mathbf{x})\rangle + |1\rangle \sum_{\substack{\mathbf{x}\in\mathbb{Z}^m \\ \|\mathbf{x}+\mathbf{x}'\|_\infty \leq 2^{2m}}} |\mathbf{x} + \mathbf{x}', \phi(\mathbf{B}\mathbf{x} - \mathbf{e})\rangle\,,$$

where the equality holds by a change of variable. By measuring the last register, with overwhelming probability this collapses to $|0\rangle|\mathbf{x}_k\rangle + |1\rangle|\mathbf{x}_k + \mathbf{x}'\rangle$, which corresponds to an m-dimensional DCP input state with modulus $2^{\mathcal{O}(m)}$. The whole process can be repeated multiple times using the same input vector $\mathbf{t}$, and results in different $\mathbf{x}_k$'s but a common $\mathbf{x}'$. Each iteration may fail because of an ill-placed cell delimitation, or if $\mathbf{x}_k + \mathbf{x}'$ has a coordinate whose magnitude is larger than 2^{2m}. This leads to a bounded number of correct DCP input states. Finally, m-dimensional DCP can be reduced to 1-dimensional DCP, with a significant modulus increase: the resulting modulus N is $2^{\mathcal{O}(m^2)}$ (see [Reg02, Section 3.1]).

Instead of using a superposition based on the coordinates with respect to a basis, we exploit the special form of $\Lambda = \mathbf{a}\mathbb{Z}_q + q\mathbb{Z}^m$ (w.l.o.g., we consider 1-dimensional LWE [BLP+13]). We start with the following superposition:

$$\sum_{x\in\mathbb{Z}_q} |0, x, \phi(\mathbf{a}x)\rangle + |1, x, \phi(\mathbf{a}x - \mathbf{t})\rangle = |0\rangle \sum_{x\in\mathbb{Z}_q} |x, \phi(\mathbf{a}x)\rangle + |1\rangle \sum_{x\in\mathbb{Z}_q} |x + s, \phi(\mathbf{a}x - \mathbf{e})\rangle\,.$$

We then measure the last register (classically known and omitted) and hopefully obtain a superposition $|0\rangle|x\rangle + |1\rangle|x + s\rangle$. This approach has several notable advantages. First, by using a grid over the

torus $\mathbb{R}^m/q\mathbb{R}^m$, the only source of failure is the position of the cell delimitation (coordinates cannot spill over, they wrap around). Second, we directly end up with a DCP state, not a vectorial variant thereof. Third, and most importantly, the DCP modulus N is only q and not $2^{\mathcal{O}(m^2)}$. Note that m should be set as $\Omega(\log q)$ for s to be uniquely determined by the LWE samples. This improvement results in a much tighter reduction.

The improvement stems from the use of a small modulus q rather than large integer coordinates. It is possible to obtain such a small DCP modulus while starting from BDD (rather than LWE), by modifying Regev's reduction as follows. One may first reduce BDD to a variant thereof that asks to find the coordinates of the BDD solution modulo a small modulus q rather than over the integers. Such a reduction is presented in [Reg09, Lemma 3.5]. One may then reduce this BDD variant to DCP as we proceed for LWE. Note that this transformation makes the BDD to DCP reduction from [Reg04b] iterative: the DCP oracle is called several times, and the input of an oracle call depends on the output of the previous oracle calls. This is akin to the phenomenon described in the *open questions* paragraph from [BLP+13].

A further difference between our reduction and the one from [Reg04b] is that we consider larger multiples of s in the input superposition to obtain a state of the form $\sum_j \rho_r(j)\,|j\rangle\,|x+js\rangle$, with $r \approx 1/\alpha$ (up to polynomial factors). This does not lead to any extra complication, but leads us to G-EDCP rather than DCP, which we crucially need to allow for a converse reduction. We conjecture that G-EDCP is strictly easier than DCP.

As Regev [Reg04a], we can also improve the resulting deviation parameter r of G-EDCP by a factor of $\sqrt{m}$ using balls rather than cube. We consider intersections of balls drawn around $\mathbf{a}\cdot s$ and its noisy shifts. The radius R of each ball is set to be the largest value such that the balls arising from different s (and their shifts) do not intersect. We are interested in the intersection area the balls drawn around $\pm s, \pm 2s$, etc. Following Regev [Reg04a], this area is large enough to guarantee that once we measure, we hit a point from the intersection of all the balls (see grey areas in Figure 4.5).

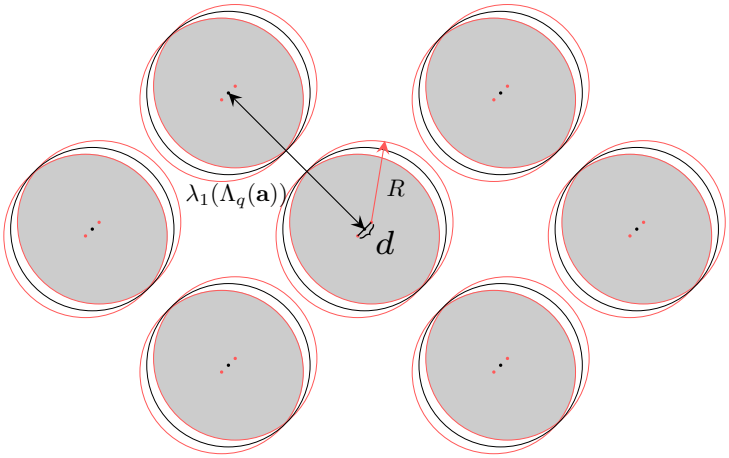

Fig. 4.5: A visualization of the balls' intersections. The lattice points (black dots) are of distance at least the first minimum of the lattice $\Lambda_q(\mathbf{a}) = \mathbf{a}\mathbb{Z}_q + q\mathbb{Z}$ to each other. The distance between the two furthest shifts $\|j\mathbf{e}\|$ (red dots) has an upper bound, denoted by d. Each ball has a radius R chosen to be (approximately) $\lambda_1(\Lambda_q(\mathbf{a}))/2$. Note that once the shaded grey area is measured, the reduction succeeds in outputting a G-EDCP sample. For the reduction to work with a constant success probability, the shaded area has to have a large enough proportion compared to the volume of the balls.

The same algorithm provides a reduction from dLWE to dG-EDCP. Given a random sample $(\mathbf{a}, \mathbf{b}) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^m$, it suffices to show that all the balls centered at $\mathbf{a}s + j\mathbf{b}$ for $s \in \mathbb{Z}_q$ and $j \in \mathbb{Z}$, do not intersect with each other. All the points considered above form the lattice $(\mathbf{a}|\mathbf{b})\mathbb{Z}_q + q\mathbb{Z}$, We argue analogously

using the upper bound on the minima of this lattice. As a result, the superposition collapses exactly to one of the balls, which gives a random sample of dG-EDCP.

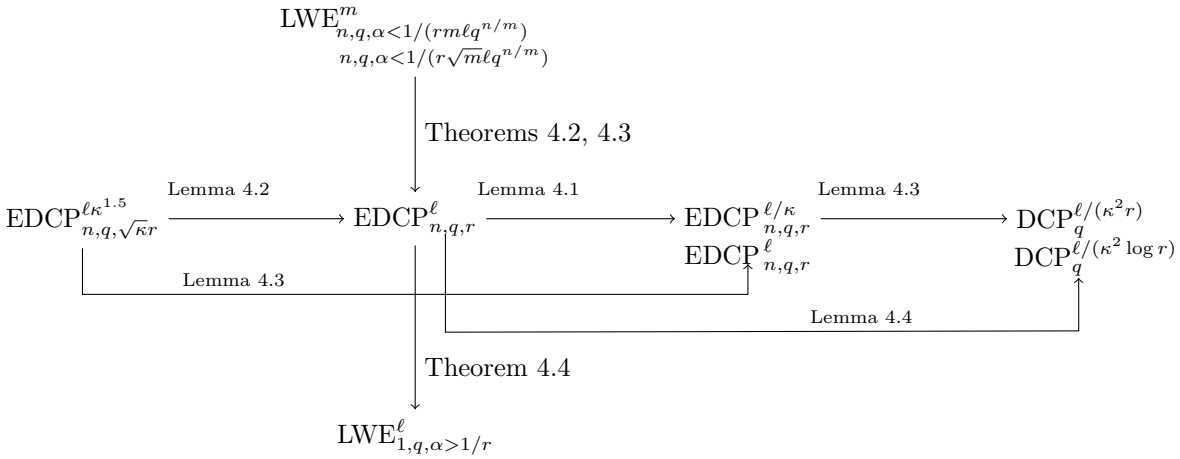

Fig. 4.6: Graph of reductions between the extrapolated Dihedral Coset Problem instantiated with uniform distribution over $\{0, 1, \ldots, r-1\}$ (the first and the third problems from the left) and ℓ-sample Gaussian EDCP with parameter r (the middle problem). We assume all the parameters n (the dimension), q (the modulus) and r are functions of a common parameter κ. The most relevant choice of such a relation one can keep in mind is when n, ℓ, q and r are $\mathrm{poly}(\kappa)$. One can trace the losses in the parameters (with respect to the number of samples ℓ and to r) when we move from one problem to another. Note that some reductions may be performed in several ways. For example, using the self-reducibility property of EDCP (Lemma 4.3), we can bypass Gaussian EDCP and have a more sample-efficient reduction from EDCP with large r to an EDCP with smaller r. Similarly, Gaussian EDCP can be reduced to DCP either directly (Lemma 4.4) or via uniform EDCP.
The two central reductions that show equivalence between the LWE and EDCP problems are on the vertical line. As for EDCP, the LWE parameters n, q, and α are functions of κ. We present two reductions from LWE to EDCP, the stronger one gives a tighter result for the error-parameter by a factor of $\sqrt{m}$.

4.3 Definitions of EDCP and its variants

We first formally define the EDCP problems as an extension of DCP. Then we show the relations between different variants of it. Analogously to LWE, EDCP has two versions: search and decision.

Definition 4.1 (Search Extrapolated Dihedral Coset Problem)**.** *Given a parameter κ, the input to the search Extrapolated Dihedral Coset Problem ($\mathrm{EDCP}^{\ell}_{n,N,D}$) with dimension n, modulus N and a discrete distribution D, consists of ℓ input states of the form (normalization is omitted)*

$$\sum_{j\in\mathrm{supp}(D)} D(j)\,|j\rangle\,|(\mathbf{x}+j\cdot\mathbf{s}) \bmod N\rangle\,, \tag{4.1}$$

where $\mathbf{x}\in\mathbb{Z}_N^n$ is arbitrary, $\mathbf{s}\in\mathbb{Z}_N^n$ is fixed for all ℓ states and all parameters n, N and ℓ are functions of κ. The task is to output the secret $\mathbf{s}$.

Definition 4.2 (Decision Extrapolated Dihedral Coset Problem)**.** *Given a parameter κ, the decision Extrapolated Dihedral Coset Problem ($\mathrm{dEDCP}^{\ell}_{n,N,D}$) with modulus N and a discrete distribution D, asks to distinguish between ℓ many* EDCP *samples and ℓ many random samples of the form*

$$|j_k\rangle\,|\mathbf{x}_k\rangle\,, \tag{4.2}$$

where $j_k \hookleftarrow D^2$, $\mathbf{x}_k \in \mathbb{Z}_N^n$ *is uniformly chosen for* $1 \leq k \leq \ell$ *and all parameters* n, N *and* ℓ *are functions of* κ. *The task is to distinguish the two cases.*

Different choices of D give rise to different instantiations of EDCP. In our case, the two interesting ones are:

1. when D is the uniform distribution over $\{0, \cdots, M-1\}$ for some $M \geq 2$, we let $\text{U-EDCP}^{\ell}_{n,N,M}$ denote this instantiation of EDCP;
2. when D is the discrete Gaussian distribution $\mathcal{D}_{\mathbb{Z},r}$, we let $\text{G-EDCP}^{\ell}_{n,N,r}$ denote this instantiation of EDCP.

The former, named the generalized hidden shift problem, was already considered in [CvD07]. The latter is central in our reductions. Correspondingly, we denote the decision version of G-EDCP by dG-EDCP.

4.4 Reductions between EDCP variants

In this section, we show that the EDCP problem enjoys several interesting properties: (1) Gaussian-EDCP ($\text{G-EDCP}^{\ell'}_{n,N,r}$) and uniform-EDCP ($\text{U-EDCP}^{\ell}_{n,N,M}$) are equivalent, up to small parameter losses; (2) EDCP enjoys a self-reduction property. The main ingredient in both proofs is quantum rejection sampling [ORR13] (see Lemma 2.21).

Lemma 4.1 (G-EDCP $\leq$ U-EDCP)**.** *Let* N, n *and* ℓ *be integers greater than 1,* $r > 0$ *a real number, and let* $M = c \cdot r$ *for some constant* c *such that* M *is an integer. Then there is a probabilistic reduction with run-time polynomial in* κ, *from* $\text{G-EDCP}^{\ell}_{n,N,r}$ *to* $\text{U-EDCP}^{\mathcal{O}(\ell/\kappa)}_{n,N,M}$.

Proof. We are given as input $\text{G-EDCP}^{\ell}_{n,N,r}$ states:

$$\left\{ \sum_{j \in \mathbb{Z}} \rho_r(j) \left| j \right\rangle \left| (\mathbf{x}_k + j \cdot \mathbf{s}) \bmod N \right\rangle \right\}_{k \leq \ell}.$$

Our aim is to find $\mathbf{s}$, given access to a $\text{U-EDCP}^{\mathcal{O}(\ell/\kappa)}_{n,N,cr}$ oracle for some constant c.

For each $\text{G-EDCP}_{n,N,r}$ sample, we proceed as follows. We let $sign(x)$ to denote the sign of x, its output is either 1 (for '$+$') or 0 (for '$-$'). For completeness, we let $sign(0) = 1$. We first compute the sign of the first register and store it in a new register:

$$\sum_{j \in \mathbb{Z}} \rho_r(j) \left| j \right\rangle \left| (\mathbf{x} + j \cdot \mathbf{s}) \bmod N \right\rangle \left| sign(j) \right\rangle.$$

Second, we measure the third register. If the observed value is 0, we discard the state. Note that we observe 1 with probability at least $1/2$, independently over all k's. From states with the observed value 1, we obtain (up to normalization):

$$\sum_{\substack{j \in \mathbb{Z} \\ j \geq 0}} \rho_r(j) \left| j \right\rangle \left| (\mathbf{x} + j \cdot \mathbf{s}) \bmod N \right\rangle.$$

Using quantum rejection sampling (see Lemma 2.21), we can transform a $\text{G-EDCP}_{N,\ell,r}$ state into a $\text{U-EDCP}_{n,N,M}$ state of the form

$$\sum_{j \in [0, M-1]} \left| j \right\rangle \left| (\mathbf{x} + j \cdot \mathbf{s}) \bmod N \right\rangle$$

with probability $\Omega(M\rho_r^2(c\cdot r)/r) = \Omega(1)$.

We repeat the procedure above until we obtain $\mathcal{O}(\ell/\kappa)$ many U-EDCP$_{n,N,M}$ states, which happens with probability $\geq 1-2^{-\Omega(\kappa)}$. We call the U-EDCP$^{\mathcal{O}(\ell/\kappa)}_{n,N,M}$ oracle to recover the secret $\mathbf{s}$ as the solution for the input G-EDCP$^{\ell}_{n,N,r}$ instance. □

Lemma 4.2 (U-EDCP $\leq$ G-EDCP)**.** *Let N, M, n and ℓ be integers greater than 1, $r > 1$ a real number, such that $M = \sqrt{\kappa}\cdot r = \mathrm{poly}(\kappa)$ is an integer. Then there is a probabilistic reduction with run-time polynomial in κ, from* U-EDCP$^{\ell}_{n,N,M}$ *to* G-EDCP$^{\mathcal{O}(\ell/\kappa^{1.5})}_{n,N,r}$*.*

Proof. We are given as input U-EDCP$^{\ell}_{n,N,M}$ states:

$$\left\{\sum_{j\in[0,M-1]} |j\rangle\,|(\mathbf{x}+j\cdot\mathbf{s}) \bmod N\rangle\right\}_{k\leq\ell}.$$

Our aim is to find $\mathbf{s}$, given access to a G-EDCP$^{\mathcal{O}(\ell/\kappa^{1.5})}_{n,N,r}$ oracle where $r = M/\sqrt{\kappa}$.

For each U-EDCP$_{n,N,M}$ state, we proceed as follows. First, we symmetrize the uniform distribution by applying the function $f(x) = x - \lfloor (M-1)/2\rfloor$ to the first register:

$$\sum_{j\in[0,M-1]} |j-\lfloor(M-1)/2\rfloor\rangle\,|(\mathbf{x}+j\cdot\mathbf{s}) \bmod N\rangle = \sum_{j'\in\left[-\lfloor\frac{M-1}{2}\rfloor,\lceil\frac{M-1}{2}\rceil\right]} |j'\rangle\,|(\mathbf{x}'+j'\cdot\mathbf{s}) \bmod N\rangle\,,$$

where $j' = j - \lfloor(M-1)/2\rfloor$, $\mathbf{x}' = \mathbf{x} + \lceil(M-1)/2\rceil\cdot\mathbf{s}$.

Using rejection sampling (Lemma 2.21), we transform each U-EDCP$_{n,N,\lceil\frac{M-1}{2}\rceil}$ state into a G-EDCP$_{n,N,r}$ state with probability $\Omega(r/M) = \Omega(1/\sqrt{\kappa})$:

$$\sum_{j'\in\left[-\lfloor\frac{M-1}{2}\rfloor,\lceil\frac{M-1}{2}\rceil\right]} \rho_r(j')\,|j'\rangle\,|(\mathbf{x}'+j'\cdot\mathbf{s}) \bmod N\rangle\,.$$

According to Lemma 2.7, the latter is within an ℓ_2-distance of $2^{-\Omega(\kappa)}$ away from the state

$$\sum_{j'\in\mathbb{Z}} \rho_r(j')\,|j'\rangle\,|(\mathbf{x}'+j'\cdot\mathbf{s}) \bmod N\rangle\,.$$

We repeat the procedure above until we obtain $\mathcal{O}(\ell/\kappa^{1.5})$ many G-EDCP$_{n,N,r}$ states, which happens with probability $\geq 1-2^{-\Omega(\kappa)}$. Then we can use the G-EDCP$^{\mathcal{O}(\ell/\kappa^{1.5})}_{n,N,r}$ oracle to recover the secret $\mathbf{s}$ as the solution to U-EDCP$^{\ell}_{n,N,M}$. □

Next, we show a self-reducibility property for EDCP.

Lemma 4.3 (EDCP self-reduction)**.** *Let N, n, and ℓ be integers greater than 1, r_1 and r_2 be such that $r_1 > r_2$ and $r_1/r_2 = \mathcal{O}(\kappa^c)$ for any constant c. Then there is a probabilistic reduction with run-time polynomial in κ, from* G-EDCP$^{\ell}_{n,N,r_1}$ *(resp.* U-EDCP$^{\ell}_{n,N,r_1}$*) to* G-EDCP$^{\mathcal{O}(\ell/\kappa^{c+1})}_{n,N,r_2}$ *(resp.* U-EDCP$^{\mathcal{O}(\ell/\kappa^{c+1})}_{n,N,r_2}$*).*

Proof. We are given as input G-EDCP$^{\ell}_{n,N,r_1}$ states:

$$\left\{\sum_{j\in\mathbb{Z}} \rho_{r_1}(j)\,|j\rangle\,|(x_k+j\cdot s) \bmod N\rangle\right\}_{k\leq\ell}.$$

Our aim is to find s, given access to a G-EDCP$^{\mathcal{O}(\ell/\kappa^{c+1})}_{n,N,r_2}$ oracle.

For each G-EDCP$_{n,N,r_1}$ sample, we proceed as follows. Using Lemma 2.21, we can transform a G-EDCP$_{N,\ell,r_1}$ state into a G-EDCP$_{N,\ell,r_2}$ state, with probability $\Omega(1/\kappa^c)$:

$$\sum_{j\in\mathbb{Z}} \rho_{r_2}(j)\,|j\rangle\,|(x_k + j\cdot s) \bmod N\rangle\,.$$

We repeat the procedure above for all given samples. With probability at least $1-2^{-\Omega(\kappa)}$, we obtain $O(\ell/\kappa^{c+1})$ many G-EDCP$_{n,N,r_2}$ states. Then we can use the G-EDCP$^{\mathcal{O}(\ell/\kappa^{c+1})}_{n,N,r_2}$ oracle to recover the secret s as the solution for the G-EDCP$^{\ell}_{n,N,r_1}$ instance.

The same arguments apply to the uniform EDCP: from U-EDCP$_{N,\ell,r_1}$ to U-EDCP$_{N,\mathcal{O}(\ell/\kappa^{c+1}),r_2}$. In this case, via quantum rejection sampling, we transform a wide uniform distribution into a narrower uniform distribution. □

In the following, we give a reduction from Gaussian-EDCP to DCP. Thus uniform-EDCP can also be reduced to DCP in two ways: either via self-reduction, or via Gaussian-EDCP as the next lemma shows. This result is especially interesting when the parameter r (or M for the uniform-EDCP) is super-polynomially large, as in this case, Lemma 4.3 cannot be applied. The lemma below works with 1-dimensional EDCP. This is without loss of generality as we can combine our main result (the computational equivalence of LWE and EDCP up to small parameter losses) with the result of Brakerski et al. [BLP+13] (the computational equivalence of LWE$_{n,q,\alpha}$ and LWE$_{1,q^n,\alpha}$ up to small parameter losses).

Lemma 4.4 (Gaussian-EDCP to DCP)**.** *Let N and ℓ be arbitrary integers. Then there is a probabilistic reduction with run-time polynomial in κ, from* G-EDCP$^{\ell}_{1,N,r}$ *to* DCP$^{\mathcal{O}(\ell/(\log r\cdot\kappa^2))}_{N}$ *if $r \geq 3\log N$, and from* G-EDCP$^{\ell}_{1,N,r}$ *to* DCP$^{\mathcal{O}(\ell/(r\cdot\kappa))}_{N}$ *otherwise.*

Proof. We are given as input G-EDCP$^{\ell}_{1,N,r}$ states:

$$\left\{\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|(x_k + j\cdot s) \bmod N\rangle\right\}_{k\leq\ell}.$$

We show how to find s if we are given access to a DCP$^{\mathcal{O}(\ell/(r\cdot\kappa))}_{N}$ oracle for $r < 3\log N$, and a DCP$^{\mathcal{O}(\ell/(\log r\cdot\kappa^2))}_{N}$ oracle otherwise.

• Case $r \geq 3\log N$.

According to Lemma 4.1, we can transform ℓ many G-EDCP$_{1,N,r}$ states into ℓ/κ many U-EDCP$_{1,N,M'}$ states with $M' = 2c\cdot r + 1$ for some constant c losing a factor of κ samples. Assume we obtain ℓ/κ many U-EDCP$_{1,N,M'}$ samples. For each such state, we symmetrize the interval $[0, M']$ as in the proof of Lemma 4.2. Then we receive a uniform distribution over $[-M, M]$ for $M = (M'-1)/2$. We compute the absolute value of the first register and store it in a new register:

$$\sum_{j\in[-M,M]} |j\rangle\,|(\hat{x}_k + j\cdot s) \bmod N\rangle\,||j|\rangle\,, \tag{4.3}$$

where $\hat{x}_k = x_k - M\cdot s$. We measure the third register and denote the observed value by v_k.

We make use of two well-known facts from number theory. For proofs, the reader may consult [Sho05, Chapter 5]. First, there exist more than $M/\log M$ primes that are smaller than M. Second, the integer N has at most $2\log N/\log\log N$ prime factors. Thus there are at least $M/\log M - 2\log N/\log\log N$ numbers smaller than M that are co-prime with all prime factors of N.

From the above, with probability $\Omega(1/\log M) = \Omega(1/\log r)$, the observed value v_k is non-zero and co-prime with N. If this is not the case, we discard the state. Otherwise, we obtain (up to

normalization):

$$|-v_k\rangle\,|(\hat{x}_k - v_k \cdot s) \bmod N\rangle + |v_k\rangle\,|(\hat{x}_k + v_k \cdot s) \bmod N\rangle\,.$$

We multiply the value in the second register by $v_k^{-1} \bmod N$:

$$|-v_k\rangle\,|(x'_k - s) \bmod N\rangle + |v_k\rangle\,|(x'_k + s) \bmod N\rangle\,,$$

where $x'_k = \hat{x}_k \cdot v_k^{-1}$.

Let $\bar{x}_k = x'_k - s \bmod N$ and $\bar{s} = 2 \cdot s \bmod N$. Rewrite the above state as:

$$|-v_k\rangle\,|\bar{x}_k\rangle + |v_k\rangle\,|(\bar{x}_k + \bar{s}) \bmod N\rangle\,.$$

As we know v_k classically, we uncompute the first register and obtain a DCP state:

$$|0\rangle\,|\bar{x}_k\rangle + |1\rangle\,|(\bar{x}_k + \bar{s}) \bmod N\rangle\,. \tag{4.4}$$

We repeat the above procedure until we obtain $\mathcal{O}(\ell/(\log r \cdot \kappa^2))$ many DCP_N states with probability $\geq 1 - 2^{-\Omega(\kappa)}$.

• Case that $r < 3\log N$.

The first steps are identical to the proof for the case $r \geq 3\log N$: Compute the absolute value of the first register to get a state as in (4.3) and measure the third register. Denote the observed value by v_k. Now we keep only those states, for which $v_k = 1$ was observed. Otherwise, we do not use the state. In case $v_k = 1$, we can easily transform the result to the state given in (4.4) analogously to the proof for $r \geq 3\log N$.

Now we show that $v_k = 1$ occurs with probability $\Omega(1/r)$ independently over all k's. Indeed,

$$\Pr[v_k = 1] = \frac{\rho_r(1)^2 + \rho_r(-1)^2}{\sum_{j\in\mathbb{Z}} \rho_r(j)^2} \geq \frac{2 \cdot \rho_r(1)^2}{\int_{\mathbb{R}} \rho_r(x)^2 \mathrm{d}x + 1} = \frac{2 \cdot \exp(-\frac{2\pi}{r^2})}{\frac{r}{\sqrt{2}} + 1} = \Omega\Big(\frac{1}{r}\Big).$$

We repeat the above procedure until we obtain $\mathcal{O}(\ell/(r \cdot \kappa))$ many DCP_N states, which happens with probability $\geq 1 - 2^{-\Omega(\kappa)}$.

In both cases considered in this lemma, we can use the $\mathrm{DCP}_N^{\mathcal{O}(\ell/(r\cdot\kappa))}$ oracle and get the secret $\bar{s}$. There are at most 2 possible values s such that $\bar{s} = 2s \bmod N$: if there are 2 possibilities, we uniformly choose either, which decreases the success probability by at most a factor of 2. □

4.5 Reduction from LWE to EDCP

In this section, we reduce $\mathrm{LWE}^m_{n,q,\alpha}$ to $\text{G-EDCP}^\ell_{n,q,r}$, where $r \approx 1/\alpha$ up to a factor of $\mathrm{poly}(n \log q)$. Analogously to Regev's reductions from uSVP to DCP, we present two versions of the reduction from LWE to G-EDCP. The second one is tighter with respect to the parameter losses. At the end of the section we show that using the same algorithm, one can reduce the decision version of LWE to the decision version of EDCP (see Definition 4.2).

4.5.1 First reduction: using cubes

The main result of this section is the following theorem.

Theorem 4.2 (LWE ≤ EDCP)**.** *Let (n, q, α) be* LWE *parameters and (n, q, r) be* EDCP *parameters. Assume $m \geq n \log q = \Omega(\kappa)$. There exists a probabilistic quantum reduction, with run-time polynomial in κ, from $\mathrm{LWE}^m_{n,q,\alpha}$ to $\text{G-EDCP}^\ell_{n,q,r}$, where $r < 1/(32m\kappa\alpha\ell q^{n/m})$.*

The main step of our reduction is to partition the ambient space $\mathbb{R}^m$ with an appropriately chosen grid (cubes). This is analogous to Regev's reduction from uSVP to DCP [Reg02]. Lemma 4.5 shows how we choose the width of the cell in our grid. Figure 4.4 gives a 2-dimensional example of such a grid.

Lemma 4.5. *For a constant $c \geq 8$, a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ with integers q, n, $m = n \log q$, and $k \geq m$, we consider a function*

$$g : (x_1, \cdots, x_m) \to (\lfloor x_1/z - w_1 \bmod \bar{q} \rfloor, \cdots, \lfloor x_m/z - w_m \bmod \bar{q} \rfloor),$$

where $z = q/c$ and $z \in [1/c, 1/2] \cdot \lambda_1^\infty(\Lambda_q(\mathbf{A}))$, $w_1, \ldots, w_m$ are uniformly chosen from $[0, 1)$, and $\bar{q} = q/z$. Then for any $\mathbf{x} \in \mathbb{Z}_q^n$, we have the following.

- *For any $\mathbf{u} = \mathbf{Ax} + \mathbf{e}_1, \mathbf{v} = \mathbf{Ax} + \mathbf{e}_2$ where $\|\mathbf{e}_1\|_\infty, \|\mathbf{e}_2\|_\infty \leq \lambda_1^\infty(\Lambda_q(\mathbf{A}))/(2ck)$, with probability $(1 - 1/k)^m$, over the randomness of $w_1, \cdots, w_m$, we have $g(\mathbf{u}) = g(\mathbf{v})$.*
- *For any $\mathbf{u} = \mathbf{Ax} + \mathbf{e}_1, \mathbf{v} = \mathbf{A}\widehat{\mathbf{x}} + \mathbf{e}_2$, where $\|\mathbf{e}_1\|_\infty, \|\mathbf{e}_2\|_\infty \leq \lambda_1^\infty(\Lambda_q(\mathbf{A}))/(2ck)$ and $\mathbf{x} \neq \widehat{\mathbf{x}} \in \mathbb{Z}_q^n$, we have $g(\mathbf{u}) \neq g(\mathbf{v})$.*

Proof. • Proof of the first claim.

Write $\mathbf{u} = \mathbf{Ax} + \mathbf{e}_1 \bmod q$ and $\mathbf{v} = \mathbf{Ax} + \mathbf{e}_2 \bmod q$ for some $\mathbf{x} \in \mathbb{Z}_q^n$ and $\|\mathbf{e}_1\|_\infty, \|\mathbf{e}_2\|_\infty \leq \lambda_1^\infty(\Lambda_q(\mathbf{A}))/(2ck)$.

Let DIFF denote the event that $g(\mathbf{u}) \neq g(\mathbf{v})$, and, for all $i \leq m$, let DIFF_i denote the event that the i^{th} coordinates of $g(\mathbf{u})$ and $g(\mathbf{v})$ differ. Since we choose $w_1, \ldots, w_m$ independently and uniformly from $[0, 1)$, we can consider each of m dimensions separately and view each $e_{1,i}/z + w_i$ and $e_{2,i}/z + w_i$ as random 1-dimensional real points inside an interval of length 1. We have

$$\Pr_{w_i}[\text{DIFF}_i] = \frac{|e_{1,i} - e_{2,i}|}{z} \leq \frac{z/k}{z} = \frac{1}{k},$$

where the inequality follows from the lower-bound on z. This implies

$$\Pr_{\mathbf{w}}[\text{NO DIFF}] = \prod_{i \leq m} \left(1 - \Pr_{w_i}[\text{DIFF}_i]\right) \geq \left(1 - \frac{1}{k}\right)^m.$$

• Proof of the second claim.

Write $\mathbf{u} = \mathbf{Ax} + \mathbf{e}_1 \bmod q$ and $\mathbf{v} = \mathbf{A}\widehat{\mathbf{x}} + \mathbf{e}_2 \bmod q$ for $\mathbf{x} \neq \widehat{\mathbf{x}} \in \mathbb{Z}_q^n$ and $\|\mathbf{e}_1\|_\infty, \|\mathbf{e}_2\|_\infty \leq \lambda_1^\infty(\Lambda_q(\mathbf{A}))/(2ck)$. Then we have

$$\begin{aligned} g(\mathbf{u}) &= \left\lfloor \frac{1}{z} \cdot (\mathbf{Ax}) + \frac{1}{z} \cdot \mathbf{e}_1 + \mathbf{w} \bmod \bar{q} \right\rfloor, \\ g(\mathbf{v}) &= \left\lfloor \frac{1}{z} \cdot (\mathbf{A}\widehat{\mathbf{x}}) + \frac{1}{z} \cdot \mathbf{e}_2 + \mathbf{w} \bmod \bar{q} \right\rfloor. \end{aligned}$$

Now we show that $g(\mathbf{u})$ and $g(\mathbf{v})$ differ in at least 1 coordinate. This is the case if the arguments of the floor function differ by 1 in at least one coordinate, i.e., $\|\frac{1}{z}\mathbf{A} \cdot (\mathbf{x} - \hat{\mathbf{x}}) + \frac{1}{z}(\mathbf{e}_1 - \mathbf{e}_2) \bmod \bar{q}\|_\infty \geq 1$.

Assume that the contrary holds. Note that due to our choice of $\mathbf{e}_i$ and $\bar{q}$, $\|\frac{1}{z}(\mathbf{e}_1 - \mathbf{e}_2) \bmod \bar{q}\|_\infty$ is either at most $1/k$ or at least $\bar{q} - 1/k$. We thus have $\|\frac{1}{z}\mathbf{A}(\mathbf{x} - \hat{\mathbf{x}}) \bmod \bar{q}\|_\infty < 1 + 1/k$ or $\|\frac{1}{z}\mathbf{A}(\mathbf{x} - \hat{\mathbf{x}}) \bmod \bar{q}\|_\infty > \bar{q} - 1 + 1/k$. Due to the bounds on z and c, the former case is equivalent to

$$\|\mathbf{A}(\mathbf{x} - \hat{\mathbf{x}}) \bmod q\|_\infty < z + z/k \leq \lambda_1^\infty(\Lambda_q(\mathbf{A}))\left(\tfrac{1}{2} + \tfrac{1}{2k}\right) \leq \lambda_1^\infty(\Lambda_q(\mathbf{A})).$$

Hence, we have just found a non-zero vector in the lattice $\Lambda_q(\mathbf{A})$ shorter than the minimum of the

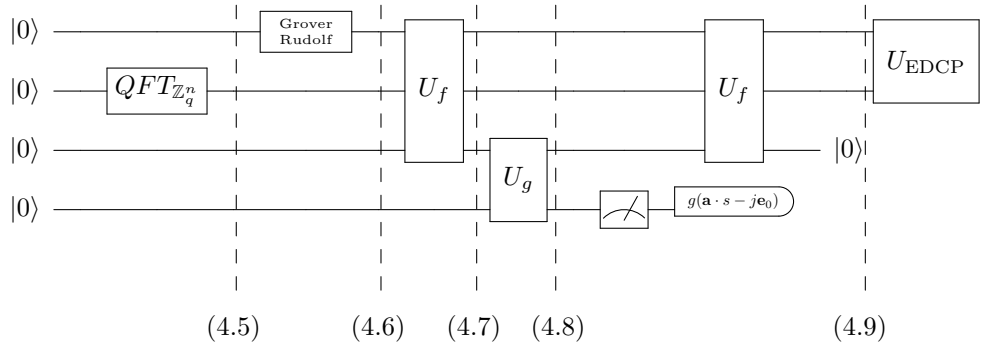

Fig. 4.7: Quantum circuit for our reduction from LWE to EDCP. The input registers are assumed to have the required number of qubits. The gate U_f is defined as $U_f |j\rangle |s\rangle |0\rangle \mapsto |j\rangle |\mathbf{s}\rangle |\mathbf{A}\mathbf{s} - j\mathbf{b} \bmod q\rangle$. The gate U_g is the realization of the function g described in Lemma 4.5, i.e. $U_g |\mathbf{x}\rangle |0\rangle \mapsto |\mathbf{x}\rangle |\lfloor \mathbf{x}/z - \mathbf{w} \bmod \bar{q} \rfloor\rangle$ for appropriately chosen $z, \mathbf{w}, \bar{q}$.

lattice. In the latter case, when $\|\frac{1}{z}\mathbf{A}\cdot(\mathbf{x}-\hat{\mathbf{x}}) \bmod \bar{q}\|_\infty > \bar{q} - 1/k + 1$, we obtain the same contradiction by noticing that $\Lambda_{\bar{q}}$ contains $\bar{q}$-ary vectors. □

According to Lemma 2.17, we have that $\lambda_1^\infty(\Lambda_q(\mathbf{A})) \geq q^{(m-n)/m}/2$ holds with probability $1 - 2^{-m}$, where the matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ is uniformly sampled. Thus in this case, we have that $z \in [1/c, 1/2] \cdot \lambda_1^\infty(\Lambda_q(\mathbf{A}))$ holds with probability $1 - 2^{-m}$, where $z = q/c$ for some constant $c \geq 8$.

Proof of Theorem 4.2. Assume we are given an $\text{LWE}_{n,q,\alpha}^m$ instance $(\mathbf{A}, \mathbf{b}_0)$ with $\mathbf{b}_0 = \mathbf{A}\cdot\mathbf{s}_0 + \mathbf{e}_0 \bmod q$. Our aim is to find $\mathbf{s}_0$ given access to a $\text{G-EDCP}_{n,q,r}^\ell$ oracle.

We first prepare a necessary number of registers in the state $|0\rangle$ and transform them to the state of the form (normalization omitted)

$$\sum_{\mathbf{s}\in\mathbb{Z}_q^n} |0\rangle |\mathbf{s}\rangle |0\rangle . \tag{4.5}$$

We use Lemma 2.20 to obtain a state within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from

$$\sum_{\mathbf{s}\in\mathbb{Z}_q^n} \Big(\sum_{j\in\mathbb{Z}} \rho_r(j) |j\rangle \Big) |\mathbf{s}\rangle |0\rangle . \tag{4.6}$$

According to Lemma 2.7, the state above is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from[3]

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n \\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r, \sqrt{\kappa}\cdot r]}} \rho_r(j) |j\rangle |\mathbf{s}\rangle |0\rangle .$$

We evaluate the function $f(j, \mathbf{s}) \mapsto \mathbf{A}\mathbf{s} - j\cdot\mathbf{b} \bmod q$ and store the result in the third register. The next equality follows from a change of variable on s

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n \\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r, \sqrt{\kappa}\cdot r]}} \rho_r(j) |j\rangle |\mathbf{s}\rangle |\mathbf{A}\mathbf{s} - j\cdot\mathbf{A}\mathbf{s}_0 - j\mathbf{e}_0\rangle = \sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n \\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r, \sqrt{\kappa}\cdot r]}} \rho_r(j) |j\rangle |\mathbf{s} + j\mathbf{s}_0\rangle |\mathbf{A}\mathbf{s} - j\mathbf{e}_0\rangle . \tag{4.7}$$

For $\mathbf{x} \in \mathbb{Z}_q^m$, we define

$$g(\mathbf{x}) = (\lfloor (x_1/z - w_1) \bmod \bar{q} \rfloor, \ldots, \lfloor (x_m/z - w_m) \bmod \bar{q} \rfloor),$$

[3]Here we cut the tail of the Gaussian distribution on the first register. Otherwise, a measurement that follows leads to a state mixed with noisy vectors from different lattice points with large (unbounded) noise. However, it has ℓ_2-distance exponentially close to the state we consider in the current algorithm.

where $w_1, \cdots, w_m$ are chosen uniformly from [0,1), $z = q/c$ for some constant $c \geq 8$ and $\bar{q} = q/z = c$. We evaluate the function g on the third register and store the result on a new register. We obtain

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n \\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}} \rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{e}_0\rangle\,|g(\mathbf{A}\mathbf{s}-j\cdot\mathbf{e}_0)\rangle\,. \tag{4.8}$$

We measure the fourth register and do not consider it further. According to Lemma 2.7, we have that $\|\mathbf{e}_0\|_\infty \leq \sqrt{\kappa}\alpha q$ holds with probability $\geq 1 - 2^{-\Omega(m)} = 1 - 2^{-\Omega(\kappa)}$. Recall that $r < 1/(32m\ell\kappa\alpha q^{n/m}) \leq 1/(4ck\kappa\alpha q^{n/m})$ for $c = 8$ and $k = m\ell$. Therefore, we have $\|\sqrt{\kappa}r\cdot\mathbf{e}_0\|_\infty \leq \lambda_1^\infty(\Lambda_q(\mathbf{A}))/(2ck)$. Note that we also have that $z \in [1/c, 1/2]\cdot\lambda_1^\infty(\Lambda_q(\mathbf{A}))$ holds with probability $2^{-m} = 2^{-\Omega(\kappa)}$. Then by Lemma 4.5, we obtain

$$\sum_{j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]} \rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{e}_0\rangle$$

for some $\mathbf{s}\in\mathbb{Z}_q^n$, with probability $(1-1/k)^m$ over the randomness of $\mathbf{A}$ and $w_1,\cdots,w_m$.

Finally, we evaluate the function $(j,\mathbf{s},\mathbf{b})\mapsto\mathbf{b}-\mathbf{A}\mathbf{s}+j\cdot\mathbf{b}_0$ on the first three registers, which gives $\mathbf{0}$. Discarding this $\mathbf{0}$-register, the state is of the form

$$\sum_{j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]} \rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,.$$

According to Lemma 2.7, the above state is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from

$$\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,. \tag{4.9}$$

We repeat the above procedure ℓ times, and with probability $(1-\frac{1}{k})^{m\ell}$, we obtain ℓ many $\text{G-EDCP}^\ell_{n,q,r}$ states

$$\left\{\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|\mathbf{x}_k+j\cdot\mathbf{s}_0\rangle\right\}_{k\leq\ell},$$

where $\mathbf{x}_k\in\mathbb{Z}_q^n$.

Now we can call the $\text{G-EDCP}^\ell_{n,q,r}$ oracle with the above states as input and obtain $\mathbf{s}_0$ as output of the oracle. □

4.5.2 An improved reduction: using balls

Here we give an improved reduction from LWE to EDCP. Following an idea of Regev (see [Reg02, Section 3.3]), instead of partitioning the ambient space $\mathbb{Z}^m$ by cubes, we consider intersections of balls drawn around the points $\mathbf{A}\mathbf{s}$ and its shifts. Note that, with this reduction, we improve the upper-bound on r, essentially by factor of $\sqrt{m}$.

Theorem 4.3 (LWE $\leq$ EDCP)**.** *Let (n,q,α) be* LWE *parameters and (n,q,r) be* EDCP *parameters. Assume that $m = \Omega(\kappa)$. There exists a quantum reduction, with run-time polynomial in κ, from $\text{LWE}^m_{n,q,\alpha}$ to $\text{G-EDCP}^\ell_{n,q,r}$, where $r < 1/(6\sqrt{2\pi e}\sqrt{m\kappa}\ell\alpha q^{n/m})$.*

We first give an intuitive idea of how the reduction works and then we give the full proof.

Given an LWE instance $(\mathbf{A},\mathbf{b} = \mathbf{A}\mathbf{s}_0+\mathbf{e}_0)\in\mathbb{Z}_q^{m\times n}\times\mathbb{Z}_q^m$, for each $\mathbf{s}\in\mathbb{Z}_q^n$, we consider (in a superposition over all such $\mathbf{s}$) a lattice point $\mathbf{A}\mathbf{s}$ together with its small shifts of $\mathbf{A}\mathbf{s}-j\mathbf{e}_0$, where the j's are drawn from a small interval symmetric around 0. So far, this is exactly what we did in the first

(cube-based) reduction. Note that we receive a configuration of points in $\mathbb{Z}_q^m$ as depicted in Figure 4.5. Also, contrary to Regev's reduction, where there is only one shift (i.e., the DCP case), our extrapolated version considers $\mathrm{poly}(\kappa)$ shifts (as we are interested in EDCP).

Let us fix some $\mathbf{As}$ together with its shifts. Draw a ball around each shift of a maximal radius R such that there is no intersection between the shifts coming from different lattice points, i.e., there is no j, j' such that $\mathcal{B}(\mathbf{As} - j\mathbf{e}_0, R) \cap \mathcal{B}(\mathbf{As}' - j'\mathbf{e}_0, R) \neq \emptyset$ for any two $\mathbf{s}, \mathbf{s}'$ such that $\mathbf{s} \neq \mathbf{s}'$. To satisfy this condition, we can take R almost as large as the first minimum of the lattice $\Lambda_q(\mathbf{A})$ (again, see Figure 4.5). With such an R, due to the fact that the shifts are small, the intersection of the balls drawn around the shifts is sufficiently large (see Lemma 4.6 below). Hence, once we measure the register that 'stores' our balls, the resulting state collapses (with large enough probability) to a superposition of some $\mathbf{As}$ for *one* $\mathbf{s}$ and all its shifts. Informally, the higher this probability, the tighter the parameters achieved by the reduction.

We need two lemmata to establish a proof of Theorem 4.3. The first lemma, due to Regev (see [Reg02, Corollary 3.9]) shows how large the intersection of two shifted balls is. Note that we work with a discretized ball $\frac{1}{L}\mathbb{Z}^n \cap \mathcal{B}_n(\mathbf{0}, R)$ such that we are able to encode it as a quantum superposition. The main difference between Regev's lemma and the one we present below is that we consider intersections of the balls taken mod q.

Lemma 4.6 (Adapted from [Reg02, Corollary 3.9])**.** *Let $L = 2^n$ and consider the scaled integer grid $\frac{1}{L}\mathbb{Z}^n$. For any $R \geq 1$, let $\bar{\mathcal{B}}_n(\mathbf{0}, R) = \mathcal{B}_n(\mathbf{0}, R) + \bar{d}$ for some vector $\bar{d}$ such that $R/\mathrm{poly}(n) \leq \|\bar{\mathbf{d}}\| \leq R$. Then, for an integer $q > 2R + \bar{d}$ we have*

$$\frac{|\frac{1}{L}\mathbb{Z}^n \cap \mathcal{B}_n(\mathbf{0}, R) \cap \bar{\mathcal{B}}_n(\mathbf{0}, R) \bmod q|}{|\frac{1}{L}\mathbb{Z}^n \cap \mathcal{B}_n(\mathbf{0}, R) \bmod q|} \geq 1 - O(\sqrt{n}\|\bar{d}/R\|).$$

According to Lemma 2.18, we have that $\lambda_1(\Lambda_q(\mathbf{A})) \geq \sqrt{m}q^{(m-n)/m}/(2\sqrt{2\pi e})$ holds with probability $1 - 2^{-m}$, where the matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is uniformly sampled and $\sqrt{m}q^{(m-n)/m}/(2\sqrt{2\pi e}) \leq q$. Thus in this case, if the radius $R = \sqrt{m}q^{(m-n)/m}/(6\sqrt{2\pi e}) = \Theta(\sqrt{m}q^{(m-n)/m})$, we have $R \leq \lambda_1(\Lambda_q(\mathbf{A}))/3$. This setup of parameters radius R and first minimum $\lambda_1(\Lambda_q(\mathbf{A}))$ will be considered in the proof of Theorem 4.3.

The following technical lemma is again due to Regev [Reg02]. It shows that while we cannot create a quantum state that perfectly corresponds to a uniform superposition of $x \in \frac{1}{L}\mathbb{Z}^n \cap \mathcal{B}_n(\mathbf{0}, R)$ (i.e., a ball), we can have a good approximation to this state. This fact will only increase the running time of the reduction by a factor of $\mathrm{poly}(\kappa)$.

Lemma 4.7 ([Reg02, Lemma 3.11])**.** *For any $1 \leq R \leq 2^{\mathrm{poly}(n)}$, let*

$$\sum_{x \in \frac{1}{L}\mathbb{Z}^n \cap \mathcal{B}_n(\mathbf{0}, R)} |x\rangle$$

be the uniform superposition (up to normalization) on grid points inside a ball of radius R around the origin, with $L = 2^n$. Then, for any $c > 0$, a state $|\tilde{\eta}\rangle$ whose trace distance from $|\eta\rangle$ is at most $1/n^c$ can be efficiently computed.

Now we are ready to give the proof of Theorem 4.3.

Proof of Theorem 4.3. Assume we are given an $\mathrm{LWE}_{n,q,\alpha}$ instance $(\mathbf{A}, \mathbf{b}_0) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$, with $\mathbf{b}_0 = \mathbf{As}_0 + \mathbf{e}_0 \bmod q$. Our aim is to find $\mathbf{s}_0$ given access to a $\text{G-EDCP}_{n,q,r}^{\ell}$ oracle.

We first prepare a necessary number of registers in the state $|0\rangle$ and transform it to the state of the form (normalization omitted)

$$\sum_{\mathbf{s} \in \mathbb{Z}_q^n} |0\rangle\,|\mathbf{s}\rangle\,|0\rangle\,.$$

We use Lemma 2.20 to obtain a state within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from

$$\sum_{\mathbf{s}\in\mathbb{Z}_q^n}\Big(\sum_{j\in\mathbb{Z}}\rho_r(j)\,|j\rangle\Big)\,|\mathbf{s}\rangle\,|0\rangle\,.$$

According to Lemma 2.7, the state above is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n\\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}}\rho_r(j)\,|j\rangle\,|\mathbf{s}\rangle\,|0\rangle\,.$$

We evaluate the function $f(j,\mathbf{s})\mapsto \mathbf{A}\mathbf{s}-j\cdot\mathbf{b} \bmod q$ and store the result in the third register. A change of variable on $\mathbf{s}$ yields

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n\\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}}\rho_r(j)\,|j\rangle\,|\mathbf{s}\rangle\,|\mathbf{A}\mathbf{s}-j\mathbf{A}\mathbf{s}_0-j\mathbf{e}_0\rangle \;=\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n\\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}}\rho_r(j)\,|j\rangle\,|\mathbf{s}+j\mathbf{s}_0\rangle\,|\mathbf{A}\mathbf{s}-j\mathbf{e}_0\rangle\,.$$

We take another register $|0\rangle$ and set $L=2^m$ as a discretization parameter and $R=\Theta(\sqrt{m}q^{(m-n)/m})$. As the LWE problem is well defined with m samples, it suffices to have $\sqrt{m}q^{(m-n)/m}/(3\sqrt{2\pi e})\le q$. Note that we also have that $R\le\lambda_1(\Lambda_q(\mathbf{A}))/3$ holds with probability $2^{-m}=2^{-\Omega(\kappa)}$. Using Lemma 4.7, we prepare a state which is at most $1/n^c$ away from $|\eta\rangle=\sum_{\mathbf{x}\in\frac{1}{L}\mathbb{Z}^m\cap\mathcal{B}_m(\mathbf{0},R)}|\mathbf{x}\rangle$ for any constant $c>0$. We set c such that it satisfies $(1-1/n^c)>(1-1/\kappa)$ to have the success probability claimed at the end of the reduction. We pretend we work with the state $|\eta\rangle$, which we tensor with our current state, and obtain

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n\\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}}\;\sum_{\mathbf{x}\in\frac{1}{L}\mathbb{Z}^m\cap\mathcal{B}_m(\mathbf{0},R)}\rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,|\mathbf{A}\cdot\mathbf{s}-j\cdot\mathbf{e}_0\rangle\,|\mathbf{x}\rangle\,.$$

We apply the function $g(\mathbf{x},\mathbf{y})=(\mathbf{x},\mathbf{x}+\mathbf{y} \bmod q)$ on the third and fourth registers (note that the mod q operation is done over rational numbers). The state becomes a superposition over the balls of radius R centered at all lattice points $\mathbf{A}\mathbf{s}$ and their shifts defined by j. Formally, we obtain

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n\\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}}\;\sum_{\mathbf{x}\in\frac{1}{L}\mathbb{Z}^m\cap\mathcal{B}_n(\mathbf{0},R)}\rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{e}_0\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{e}_0+\mathbf{x} \bmod q\rangle\,.$$

We measure the fourth register and discard it, as now we know it classically. Using Lemma 2.8 on lattice $\mathbb{Z}^m$ and shift $\mathbf{0}$, we have $\|\mathbf{e}_0\|\le\sqrt{m}\alpha q$ with probability $\ge 1-2^{-\Omega(m)}=1-2^{-\Omega(\kappa)}$. Recall that $r\le 1/(\sqrt{m\kappa}\ell\alpha q^{n/m})$ (up to constants). Therefore, we have $\|\sqrt{\kappa}r\cdot\mathbf{e}_0\|\le q^{(m-n)/m}/\ell$.

We use Lemma 4.6 with $\bar{d}=q^{(m-n)/m}/\ell$ and radius $R=\Theta(\sqrt{m}q^{(m-n)/m})$. Note that due to the large minimal distance of $\Lambda_q(\mathbf{A})$, the balls emerging from two different lattice points (or their shifts) do no intersect. Hence, the measurement yields

$$\sum_{j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}\rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{e}_0\rangle$$

for some $\mathbf{s}\in\mathbb{Z}_q^n$, with probability $\mathcal{O}(1-1/\ell)$ taken over the uniform distribution defined on $\mathcal{B}_m(\mathbf{0},R)$.

Finally, we evaluate the function $(j,\mathbf{s},\mathbf{b})\mapsto\mathbf{b}-\mathbf{A}\mathbf{s}+j\cdot\mathbf{b}_0$ on the first three registers. Discarding the last register, the state is of the form

$$\sum_{j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}\rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,.$$

According to Lemma 2.7, the latter is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from

$$\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|\mathbf{s}+j\cdot\mathbf{s}_0\rangle\,.$$

We repeat the above procedure ℓ times, and with non-negligible probability $(1-1/\ell)^\ell$, we obtain ℓ many G-EDCP$_{n,q,r}$ states

$$\left\{\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|\mathbf{x}_k+j\cdot\mathbf{s}_0\rangle\right\}_{k\leq\ell},$$

where $\mathbf{x}_k\in\mathbb{Z}_q^n$.

Now we can call the G-EDCP$^m_{n,q,r}$ oracle with the above states as input and obtain $\mathbf{s}_0$ as the output of the oracle. □

4.5.3 Reduction from dLWE *to* dEDCP

As a corollary to the above theorem, we show that decision LWE can be reduced to decision EDCP. In fact, to establish the reduction, we use the same algorithm as for Theorem 4.3 (a weaker reduction given in Theorem 4.2 will work as well). Recall that in the proof, starting from an EDCP sample, we obtain an LWE sample with non-negligible probability. Corollary 4.1 below shows that in case we are given a tuple $(\mathbf{A},\mathbf{b})$ drawn uniformly at random from $\mathbb{Z}_q^{m\times n}\times\mathbb{Z}_q^m$, the procedure described in the proof of Theorem 4.3 outputs a state of the form $|j\rangle\,|\mathbf{x}\rangle$, a uniform counterpart to EDCP in the sense of Definition 4.2.

Corollary 4.1 (dLWE $\leq$ dEDCP)**.** *Let (n,q,α) be valid* dLWE *parameters and (n,q,r) be valid* dEDCP *parameters. Assume that $m=\Omega(\kappa)$. There exists a quantum reduction, with run-time polynomial in κ, from* LWE$^m_{n,q,\alpha}$ *to* G-EDCP$^\ell_{n,q,r}$*, where $r<1/(6\sqrt{2\pi e}\sqrt{m\kappa}\ell\alpha q^{(n+1)/m})$.*

Proof of Corollary 4.1. We are given m many samples from $\mathbb{Z}_q^{m\times n}\times\mathbb{Z}_q^m$ either of the form

$$(\mathbf{A},\mathbf{A}\mathbf{s}_0+\mathbf{e}_0),$$

or of the form

$$(\mathbf{A},\mathbf{b}),$$

where $\mathbf{A}\in\mathbb{Z}_q^{m\times n}$ is uniformly chosen, $\mathbf{s}_0\in\mathbb{Z}_q^n$ is fixed, $\mathbf{e}_0\in\mathcal{D}_{\mathbb{Z}^m,\alpha q}$ and $\mathbf{b}\in\mathbb{Z}_q^m$ is uniformly chosen and independent from $\mathbf{A}$. Our aim is to distinguish between the above two cases given access to a dG-EDCP$^\ell_{n,q,r}$ oracle.

We follow the procedure of the reduction algorithm used in the proof of Theorem 4.3 except that now we set $R=\Theta(\sqrt{m}q^{(m-n-1)/m})$. As the LWE problem is well-defined with $m=\Theta(n)$ many samples, we can guarantee (for an appropriate choice of constants) that $\sqrt{m}q^{(m-n)/m}/(3\sqrt{2\pi e})\leq q$. Thus, we also have $R\leq\lambda_1(\Lambda_q(\mathbf{A}|\mathbf{b}))/3$, where the inequality holds with probability $2^{-m}=2^{-\Omega(\kappa)}$ (see Lemma 2.18).

We run the algorithm described in the proof of Theorem 4.3 until we obtain a state of the form

$$\sum_{\substack{\mathbf{s}\in\mathbb{Z}_q^n\\ j\in\mathbb{Z}\cap[-\sqrt{\kappa}\cdot r,\sqrt{\kappa}\cdot r]}}\ \sum_{\mathbf{x}\in\frac{1}{L}\mathbb{Z}^m\cap\mathcal{B}_m(\mathbf{0},R)} \rho_r(j)\,|j\rangle\,|\mathbf{s}\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{b}\rangle\,|\mathbf{A}\mathbf{s}-j\cdot\mathbf{b}+\mathbf{x}\rangle\,.$$

We measure the fourth register and ignore it from now on. Due to the bound on r, we have $q\geq 2\sqrt{\kappa}r$. Thus all the balls created with centers over the lattice points $\mathbf{A}\mathbf{s}$ and their shifts $\mathbf{A}\mathbf{s}-j\mathbf{b}$, do not intersect

with each other. Therefore, with probability $1 - O(1/\ell)$, we obtain

$$|j_k\rangle\,|\mathbf{s}_k\rangle\,|\mathbf{A}\mathbf{s}_k - j_k \cdot \mathbf{b}\rangle$$

for some $j_k \in \mathbb{Z}$ distributed as $\mathcal{D}^2_{\mathbb{Z},r}$ and some uniform $\mathbf{s}_k \in \mathbb{Z}_q^n$.

Finally, we evaluate the function $(j, \mathbf{s}, \mathbf{b}) \mapsto \mathbf{b} - \mathbf{A}\mathbf{s} + j \cdot \mathbf{b}_0$ on the first three registers. This zeroizes the last register, which we can safely discard. Now the state is of the form

$$|j_k\rangle\,|\mathbf{s}_k\rangle\,,$$

where $j_k \hookleftarrow \mathcal{D}^2_{\mathbb{Z},r}$ and $\mathbf{s}_k \in \mathbb{Z}_q^n$ is uniformly chosen.

We repeat the above procedure ℓ times, and with probability $(1 - O(1/\ell))^\ell$, we obtain ℓ many random samples of the form

$$\{|j_k\rangle\,|\mathbf{s}_k\rangle\}_{k \le \ell}\,,$$

where $\mathbf{s}_k$ is chosen uniformly at random from $\mathbb{Z}_q^n$. This is exactly a uniform input to dEDCP with respect to of Definition 4.2. □

4.6 Reduction from EDCP to LWE

In this section, we reduce $\text{G-EDCP}^\ell_{n,N,r}$ to $\text{LWE}^\ell_{n,N,\alpha}$, where $r \approx 1/\alpha$ up to a factor of $\text{poly}(n \log N)$.

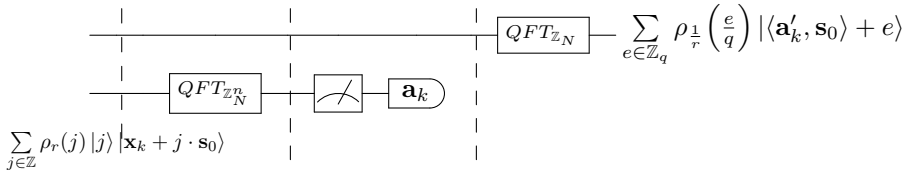

Fig. 4.8: Reduction from G-EDCP to LWE.

Theorem 4.4 (EDCP ≤ LWE)**.** *Let (n, N, r) be valid* EDCP *parameters and (n, N, α) with $r = \Omega(\sqrt{\kappa})$ be valid* LWE *parameters. Assume that $\ell = \Omega(\kappa)$. There exists a quantum reduction, with run-time polynomial in κ, from* $\text{G-EDCP}^\ell_{n,N,r}$ *to* $\text{LWE}^\ell_{n,N,\alpha}$*, where $\alpha = 1/r$.*

Proof. Assume we are given ℓ many $\text{EDCP}_{n,N,r}$ samples

$$\left\{\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|\mathbf{x}_k + j \cdot \mathbf{s}_0 \bmod N\rangle\right\}_{k\in[\ell]}.$$

Our aim is to find $\mathbf{s}_0$, given access to an $\text{LWE}^\ell_{n,N,\alpha}$ oracle.

For each input state, the quantum Fourier transform over $\mathbb{Z}_N^n$ is applied to the second register, which yields (without loss of generality, consider the k^{th} sample)

$$\sum_{\mathbf{a}\in\mathbb{Z}_N^n} \sum_{j\in\mathbb{Z}} \omega_N^{\langle \mathbf{a}, (\mathbf{x}_k + j\cdot\mathbf{s}_0)\rangle} \cdot \rho_r(j)\,|j\rangle\,|\mathbf{a}\rangle\,. \tag{4.10}$$

Then we measure the second register and let $\mathbf{a}_k$ denote the observed value. Note that statistically each element of $\mathbb{Z}_N^n$ is measured with probability $1/N^n$ and that the samples of $\mathbf{a}_k$ for different k's are

independent. Omitting the global phase of each state, we obtain

$$\sum_{j\in\mathbb{Z}} \omega_N^{\langle \mathbf{a}_k,(j\cdot \mathbf{s}_0)\rangle} \cdot \rho_r(j)\,|j\rangle\,|\mathbf{a}_k\rangle\,. \tag{4.11}$$

We omit the second register as we know each $\mathbf{a}_k$ classically. Since $N \geq \sqrt{\kappa} r$, from Lemma 2.7, it follows that the resulting state is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from the state (note the change in the range for j)

$$\sum_{j\in\mathbb{Z}_N} \omega_N^{j\cdot\langle \mathbf{a}_k,\mathbf{s}_0\rangle} \cdot \rho_r(j)\,|j\rangle\,. \tag{4.12}$$

For each such state, the quantum Fourier transform over $\mathbb{Z}_N$ yields

$$\sum_{b\in\mathbb{Z}_N}\sum_{j\in\mathbb{Z}_N} \omega_N^{j\cdot(\langle \mathbf{a}_k,\mathbf{s}_0\rangle+b)} \cdot \rho_r(j)\,|b\rangle\,. \tag{4.13}$$

Once again we use Lemma 2.7 to argue that the state above is within ℓ_2-distance of $2^{-\Omega(\kappa)}$ from the state

$$\sum_{b\in\mathbb{Z}_N}\sum_{j\in\mathbb{Z}} \omega_N^{j\cdot(\langle \mathbf{a}_k,\mathbf{s}_0\rangle+b)} \cdot \rho_r(j)\,|b\rangle\,.$$

Using the Poisson summation formula (Lemma 2.6) and changing the summation variable to $e = N\cdot j + \langle \mathbf{a}_k,\mathbf{s}_0\rangle + b$, the above state can be rewritten as

$$\sum_{b\in\mathbb{Z}_N}\sum_{j\in\mathbb{Z}} \rho_{1/r}\Big(j+\frac{\langle \mathbf{a}_k,\mathbf{s}_0\rangle+b}{N}\Big)\,|b\rangle = \sum_{e\in\mathbb{Z}} \rho_{1/r}\Big(\frac{e}{N}\Big)\,|\langle \mathbf{a}'_k,\mathbf{s}_0\rangle + e \bmod N\rangle$$

where $\mathbf{a}'_k = -\mathbf{a}_k \bmod N$. Since $r = \Omega(\sqrt{\kappa})$, we can apply Lemma 2.7 to the above state (for a scaled $\mathbb{Z}$-lattice), and instead of the above state, consider the state that is within a $2^{-\Omega(\kappa)}$ ℓ_2-distance from it, namely:

$$\sum_{e\in\mathbb{Z}_N} \rho_{1/r}\Big(\frac{e}{N}\Big)\,|\langle \mathbf{a}'_k,\mathbf{s}_0\rangle + e\rangle\,. \tag{4.14}$$

Once we measure the state above, we obtain an LWE sample

$$(\mathbf{a}'_k, \langle \mathbf{a}'_k,\mathbf{s}_0\rangle + e_k)\,,$$

where $e_k \hookleftarrow \mathcal{D}_{\mathbb{Z},N/r}$.

Now we can call the $\mathrm{LWE}_{n,N,\alpha}$ oracle for $\alpha = 1/r$ with the above states as input and obtain $\mathbf{s}_0$ as output of the oracle. □

4.6.1 Reduction from dEDCP *to* dLWE

Similar to the previous section where as a corollary we show that dLWE can be reduced to dEDCP, we complete this section by a reverse reduction, from dEDCP to dLWE. Again, we use exactly the same reduction algorithm as for the search versions (see Figure 4.8). Thus it remains to show that we can obtain a uniform random sample $(\mathbf{a}, b) \in \mathbb{Z}_N^n \times \mathbb{Z}_N$ given as input a state of the form $|j\rangle\,|\mathbf{x} \bmod N\rangle$.

Corollary 4.2 (dEDCP $\leq$ dLWE)**.** *Let (n, N, r) be valid* dG-EDCP *parameters and (n, N, α) be valid* dLWE *parameters. Assume that $\ell = \Omega(\kappa)$. There exists a quantum reduction, with run-time polynomial in κ, from* $\text{dG-EDCP}^{\ell}_{n,N,r}$ *to* $\text{dLWE}^{\ell}_{n,N,\alpha}$*, where $\alpha = 1/r$.*

Proof. Assume we are given ℓ many samples of $\mathrm{EDCP}_{n,N,r}$ either of the form

$$\left\{\sum_{j\in\mathbb{Z}} \rho_r(j)\,|j\rangle\,|\mathbf{x}_k + j\cdot\mathbf{s}_0\rangle \bmod N\right\}_{k\in[\ell]}$$

or of the form

$$\{|j_k\rangle\,|\mathbf{x}_k \bmod N\rangle\}_{k\in[\ell]}\,,$$

where $j_k \hookleftarrow \mathcal{D}^2_{\mathbb{Z},r}$ and $\mathbf{x}_k \in \mathbb{Z}_N^n$ is uniform. Our aim is to distinguish between the above two forms given access to a $\mathrm{dLWE}_{n,N,\alpha}$ oracle.

As explained above, we assume that uniform samples of EDCP are given. For each input state, after the quantum Fourier transform over $\mathbb{Z}_N^n$ on the second register, we obtain

$$\sum_{a\in\mathbb{Z}_N^n} \omega_N^{\langle \mathbf{x}_k,\mathbf{a}\rangle}\,|j_k\rangle\,|\mathbf{a}\rangle\,.$$

Then we measure the second register and let $\mathbf{a}_k$ denote the observed value. Note that each element of $\mathbb{Z}_N^n$ is measured with probability $1/N^n$ and that the samples of $\mathbf{a}_k$ for different k's are independent. Up to a global phase, we have

$$|j_k\rangle\,|\mathbf{a}_k\rangle\,.$$

We omit the second register which is known to us. According to Lemma 2.7, with probability $1-2^{-\Omega(\kappa)}$, the value stored in the first register is in the range $[-\lfloor N/2\rfloor, \lceil N/2\rceil - 1]$. Applying the QFT over $\mathbb{Z}_N$ to the first register, we obtain

$$\sum_{b\in\mathbb{Z}_N} \omega_N^{j_k\cdot b}\,|b\rangle\,.$$

Once we measure the state above and let b_k denote the observed value. Note that each element of $\mathbb{Z}_N$ is measured with probability $1/N$ and that the samples of $\mathbf{a}_k$ for different k's are independent. We obtain a sample

$$(\mathbf{a}_k, b_k)\,,$$

where $(\mathbf{a}_k, b_k)$ are uniformly random over $\mathbb{Z}_N^n \times \mathbb{Z}_N$. □

Chapter 5

A finer modelling of BKZ: understanding the head concavity

In this chapter, we first report experiments that we run on the BKZ algorithm to measure the folklore shorter-than-expected phenomenon in many aspects. Then we propose a new simulator for modelling the behavior of BKZ that is more precise than the prior ones. In this modified simulator, the main new ingredient is the use of randomness to simulate the distribution of the minimum of random lattices. Further, we provide a new BKZ variant that exploits the shorter-than-expected phenomenon, which allows to compute a basis with even better quality.

Contents

5.1 Introduction

The BKZ lattice reduction algorithm, proposed by Schnorr and Euchner [SE94], is the most practical algorithm for solving lattice problems, thus also for cryptanalysis of lattice-based cryptography. However, there is a large performance gap between existing theoretical bounds and experimental results. In [HPS11] (see also [Neu17]), it was shown that in the worst case, BKZ_β (with early termination) achieves a Hermite factor of $\beta^{O(n/\beta)}$ within a polynomial number of calls to the SVP solver, for $\beta = o(n)$ and n growing to infinity. It was also shown in [HS08] that there exist bases with such Hermite factors (up to a constant factor in the exponent) which are left unchanged when given as inputs to BKZ_β. Unfortunately, these worst-case bounds are quantitatively very far from experimental data. The Geometric Series Assumption (GSA), introduced by Schnorr [Sch03], was experimentally observed to be a good first approximation to the practical behavior of BKZ. It states that the ratio between two consecutive Gram–Schmidt norms is almost equal to some constant r. Among others, this implies that a Hermite factor of $r^{(n-1)/2}$. It was argued in [CN11] (see also [Che09, Chapter 4]) that for β small compared to n, one should have $r \approx (\frac{\beta}{2\pi \mathrm{e}}(\pi\beta)^{\frac{1}{\beta}})^{\frac{1}{\beta-1}}$. The latter value is derived by relying on the Gaussian heuristic to estimate the smallest non-zero norm in a β-dimensional lattice.

Nevertheless, the GSA does not provide an exact fit: for $\beta \gtrsim 30$, the typical BKZ output basis has its first few Gram–Schmidt norms and its last $\approx \beta$ Gram–Schmidt norms violate this assumption. These first and last Gram–Schmidt norms are respectively called the *head* and the *tail*, the rest being the *body*. In [CN11], Chen and Nguyen refined the sandpile model from [HPS11] and provided a BKZ simulator based on the Gaussian heuristic (with a modification for the tail, see Section 2.5.4). Their BKZ simulator captures the body and tail behaviors of the Gram–Schmidt norms very precisely [CN11, YD17]. However, as investigated in [CN11, AWHT16, YD17], the Chen–Nguyen simulator fails to capture the head phenomenon: the head almost follows the GSA in the simulations, whereas, in the experiments, the logarithmic Gram–Schmidt norms form a concave curve (instead of a line). Put plainly, BKZ finds shorter vectors than predicted by the Chen–Nguyen simulator. In Figure 5.1 and 5.2 for an example, we run BKZ with block-size 45 on 100-dimensional lattices (generated by the Darmstadt lattice challenge generator) and record the Gram-Schmidt norms of the reduced bases after 2000 tours, each data is averaged over 100 results. As we can see, the first few Gram–Schmidt norms of the BKZ reduced basis are less than the ones estimated by the Chen–Nugyen simulator for BKZ. This inaccuracy may lead to overestimated security evaluations in cryptographic design. Understanding the head concavity phenomenon was put forward as an important open problem in [YD17], for assessing the bit-security of concrete lattice-based cryptosystems.

Contributions. Our first main contribution is the design of a more accurate BKZ simulator, relying on a probabilistic version of the Gaussian heuristic. More precisely, we take into account the fact that the norm of a shortest non-zero vector of a random lattice is not a fixed quantity driven by the Gaussian heuristic, but a random variable. Concretely, we use a distribution derived from the result on the distribution of short vectors in random lattices by Södergren [Söd11]. We compare our probabilistic simulator and experimental BKZ with large block-sizes, and observe that our simulator provides accurate predictions of the head region, while maintaining a good approximation on both body and tail regions. If we focus on the head region, the new simulator is always more precise than the Chen–Nguyen simulator, and similarly accurate for body and tail. Therefore, the Hermite factors estimated by the new simulator are more accurate and fit the experimental results more precisely. This is established through extensive experiments designed to measure the head concavity phenomenon. Such understanding also allows to efficiently assess how it scales for larger block-sizes: when β increases, the head phenomenon decreases, i.e., the GSA is followed more closely.

Our second main contribution is an algorithmic exploitation of the fact that BKZ performs better than the GSA for its first output vectors. We propose a new variant of BKZ, pressed-BKZ, that aims to

exploit the head phenomenon everywhere in the graph of Gram–Schmidt norms. To do so, we proceed iteratively: we run BKZ between indices 1 and n, then we freeze the first basis vector and run BKZ between indices 2 and n (i.e., on the appropriately projected basis), then we freeze the first two basis vectors and run BKZ between 3 and n, etc. The bonus of being at the start of the basis is exploited at every position. The output basis tightly follows the GSA in the head and body regions. The gain is that the Gram–Schmidt slope is better than with the original BKZ. Overall, for the same block-size as in BKZ, pressed-BKZ produces lattice bases of improved quality. We adapt our BKZ simulator to pressed-BKZ, and again, the simulation is quite accurate, giving further confidence that our simulation correctly captures the head phenomenon.

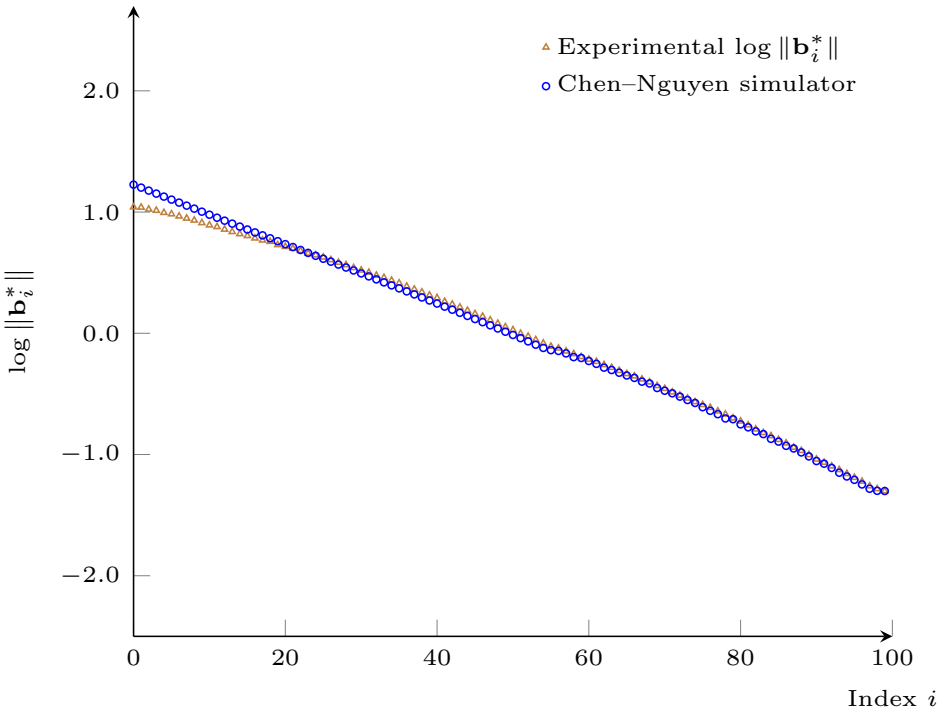

Fig. 5.1: Gram–Schmidt log-norms for BKZ_{45} at tour $2,000$.

Another way to exploit the head phenomenon was suggested in [AWHT16]. As the first Gram–Schmidt norms decrease less fast than the others, this means the first BKZ blocks are more reduced, and hence solving the corresponding SVP instances is easier. In [AWHT16], Aono *et al.* proposed using a larger block-size in the head region than in the rest of the basis. The purpose is to make the head region even better, without increasing the overall cost significantly. We combine this "adaptive block-size" strategy with the pressed-BKZ algorithm. This variant allows us to accelerate the convergence of pressed-BKZ towards its typical output quality.

Finally, we demonstrate the usefulness of the BKZ variant by testing it on the SVP-120 instance of the Darmstadt lattice challenge.[1] We also compare the quality of pressed-BKZ_{60} with that of BKZ_β for various block-sizes β.

Impact. For concrete lattice-based cryptosystems with parameters set using the Chen–Nguyen simulator (or the corresponding GSA ratio), the head phenomenon is a potential security risk: as BKZ performs better than taken into account while setting parameters, the parameters were potentially set too low for the bit-security targets. Our simulator, which accurately predicts the head phenomenon,

[1] `https://www.latticechallenge.org/svp-challenge/`

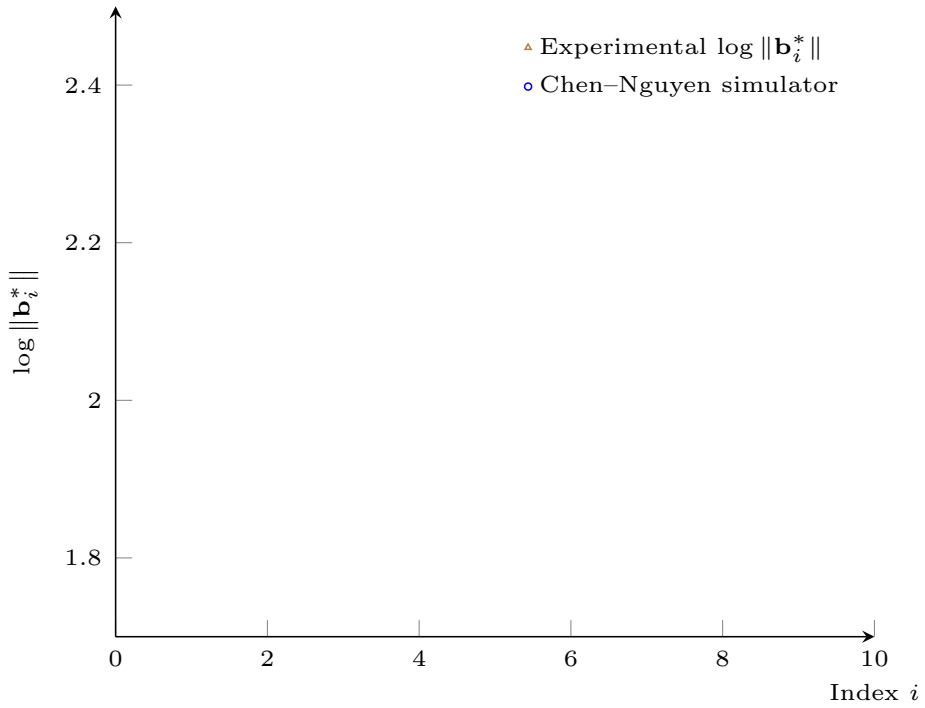

Fig. 5.2: Same as Figure 5.1, but zoomed in.

suggests that the head phenomenon vanishes when the block-size becomes large. We conjecture this is because the distribution of the first minimum in random lattices has a standard deviation that decreases to 0 relatively quickly when the lattice dimension increases (this lattice dimension corresponds to the BKZ block-size β). Quantitatively, the phenomenon has almost fully disappeared for $\beta \approx 200$. It is also less important when n is much larger than β. Concrete figures are provided at the end of Section 5.3.4.

The lattice-based submissions to the NIST post-quantum standardization process[2] use conservative security estimates. In particular, they rely on lower bounds for the cost of solving SVP in dimension β, which are significantly below what can currently be achieved in practice (we refer to [ACD$^+$] for concrete figures). Note that this seems unrelated to the head phenomenon. As a result, the BKZ block-sizes needed to break the scheme are often in the hundreds, a range of block-sizes for which the head phenomenon has already vanished. The NIST candidates most impacted are those that were more aggressive in setting their parameters, though the impact remains limited even for them.

Oppositely, for block-sizes that can be handled in practice (e.g., $\beta \lesssim 100$), the head concavity phenomenon is non-negligible, and can be exploited. Our work can then help make concrete cryptanalysis more accurate. By allowing one to solve SVP in larger dimensions β using pressed-BKZ$_{\beta'}$ with $\beta' < \beta$ as a pre-processing, our work should allow one to perform BKZ in larger block-sizes β. It is well-known that for small block-sizes (say $\beta \lesssim 35$), BKZ$_\beta$ benefits from the fact that the Gaussian heuristic is over-conservative for the BKZ$_\beta$ projected sublattices (see [CN11], for example). This phenomenon vanishes when the block-size becomes higher. But then BKZ$_\beta$ benefits from the head concavity phenomenon. As a result, extrapolating concrete experiments in such block-sizes to draw conclusions in much larger block-sizes seems to amount to wild guessing. On the other hand, we have a better simulation of the head phenomenon for BKZ with practical block-sizes. By exploiting the head phenomenon, we can hope to reach higher block-sizes, for which such small block-size effects do not occur anymore. In this smoother regime, extrapolating experiments should become sounder.

[2] `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography`

Related works. The first simulator for predicting the Gram–Schmidt norms of a BKZ-reduced basis was proposed by Chen and Nguyen in [CN11]. It relies on the assumption that each SVP-solver in the projected local block finds a lattice vector whose norm exactly matches its Gaussian heuristic estimate for that local block, except for a few blocks at the end of the basis. It is a good first approximation, but remains inaccurate in two ways. First, it does not capture the head concavity phenomenon (which is reported nevertheless in the experiments of [CN11]). Second, it does not take into account that in practice it is preferable to use heuristic SVP-solvers which may miss the optimal solutions. The main such heuristic SVP-solver is pruned enumeration, introduced in [SE94] and refined and improved in [GNR10]. It consists in pruning the enumeration tree by keeping only the nodes that are most likely to lead to interesting leaves. As a result, only a subset of lattice points are enumerated within the required radius, and the optimal solution may be missed. Extreme pruning [GNR10] goes even further: it decreases the probability of finding a shortest non-zero vector to lower the time/probability ratio, and runs the process several times to boost the success probability. Each time, the lattice basis is re-randomized and reduced with a lower block-size to prepare for the enumeration.

In [AWHT16], Aono *et al.* described the so-called progressive-BKZ. The main new ingredient is that the latter tries to avoid the re-randomization/pre-processing overheads by using a single enumeration in any SVP call. For this, the authors increase the search radius in the enumeration, aiming to find a short vector but not necessarily a shortest one by pruning the enumeration tree. This search radius is adaptively derived from the current basis quality. Since the authors are not in the regime of finding a shortest non-zero vector, to estimate the success probability, the authors model lattice points of norm below the search radius as random points in the ball of that radius (see [AWHT16, Lemma 1]). This pruned enumeration with increased search radius heuristically produces a non-zero vector in lattice Λ of norm $\frac{\beta}{\beta+1} \cdot \alpha \cdot \mathrm{GH}(\Lambda)$ for some $\alpha \geq 1$, using their random point model (also refer to Section 2.5). For $\alpha = 1$, we obtain an expectation for the first minimum that is lower than the Gaussian heuristic. Aono *et al.* also adapted the Chen–Nguyen simulator by modifying the expected norm found by the SVP-solver using the random points model rather than the Gaussian heuristic value. Note that the updated simulator takes some probabilistic phenomenon into account but remains deterministic. In particular, it does not capture the head concavity phenomenon. Finally, as mentioned earlier, Aono *et al.* also experimentally observed the head concavity phenomenon, and proposed to exploit it by using BKZ with larger block-size on the first few blocks.

Yu and Ducas [YD17] ran extensive experiments to assess the practical behavior of BKZ. They have two main experimental observations. First, the distribution of differences $v_i := \log \|\mathbf{b}_i^*\| - \log \|\mathbf{b}_{i+1}^*\|$ between two consecutive Gram–Schmidt log-norms, varies as a function of the index i when i belongs to the head and tail regions (and it does not in the body region). Second, the covariance between v_i and v_{i+2} is 0 for all i, but v_i and v_{i+1} are negatively correlated: in the head and tail regions, their covariance depends on both i and the block-size β, but in the body region only the block-size β contributes to their covariance. These observations quantify the head concavity phenomenon more precisely.

Software. Our BKZ experiments were run using the `fplll` [dt16] (version 5.2.0) and `fpylll` [dt17] (version 0.4.0dev) open-source libraries. The efficiency of these libraries for large block-sizes $\beta \geq 50$ was essential for obtaining useful statistics. Our simulator, coded in Python, and the BKZ variants, coded in C++ are all freely available online: `https://github.com/BKZsimulator` under the GNU Lesser General Public License (either version 2.1 of the License or any later version).

As mentioned earlier, we report experiments on pressed-BKZ with an adaptive block-size strategy, for the SVP-120 challenge. We expect our BKZ improvements to be useful in larger dimensions as well (e.g., SVP-150), if given sufficient computational resources. We want to stress that our primary goal is to model, predict and exploit the head concavity phenomenon, the SVP-120 experiment being an illustration of its relevance.

Auxiliary supporting material. To make our results reproducible and to report experimental

observations in more details, we provide codes (as mentioned above), experimental raw data and video files online: `perso.ens-lyon.fr/weiqiang.wen/thesis.html`.

5.2 Measuring the head concavity

In this section, we describe in detail the concavity phenomenon in the leading Gram–Schmidt log-norms. In particular, we report experiments on the quality of bases output by BKZ_β and on the evolution of Gram–Schmidt norms during the execution of the algorithm.

In our experiments, we consider the knapsack-type lattice bases generated by the Darmstadt lattice challenge generator. In dimension n, the generator selects a prime p of bitsize $10 \cdot n$ and sets the first basis vector as $(p, 0, \cdots, 0)$. For $i > 1$, the i-th basis vector starts with a uniformly chosen integer modulo p, and all other entries are 0 except the i-th entry which is 1. When using the generator file, the `seed` is from the set $\{0, \cdots, k-1\}$, where k is total number of samples in the experiment, which enables reproducibility. We always run LLL reduction before a BKZ reduction. We use the default LLL in `fplll` of parameter $\delta = 0.99$.

5.2.1 BKZ output quality

In our first set of experiments, we measure the output quality of the BKZ algorithm. We consider the final reduced basis, for increasing block sizes β. We let BKZ run until it fully stops and use a full enumeration as SVP-solver (without pruning), to avoid side-effects. We then measure the Gram–Schmidt log-norms $\{\log \|\mathbf{b}_i^*\|\}_{i \leq n}$ of the reduced basis $\mathbf{B}$. In particular, when BKZ completes, the vector $\mathbf{b}_i^*$ is a shortest non-zero vector of the lattice $\Lambda_{[i, i+\min(i+\beta-1, n)]}$ (up to the 0.99 factor), for every i (see Section 2.5.4).

As we use BKZ until exhaustion with full enumeration, the experiments are quite lengthy. We restricted them to dimension $n = 100$, with selected block sizes β ranging from 4 to 40. For each choice of β, we conduct the experiment 100 times using input lattices generated with different seeds. For each experiment, we normalize the $\log \|\mathbf{b}_i^*\|$'s of the reduced basis by substracting one hundredth of the logarithmic determinants of its input lattice, such that the summation of the new logarithmic Gram–Schmidt norms is normalized to be 0. This step helps eliminate the small differences of determinants of all generated lattices. We then average $\log \|\mathbf{b}_i^*\|$ for each i over the 100 samples.

We plotted the results for the various block sizes in Figures 5.3–5.8. The x-axis corresponds to the basis vector at index i and the y-axis is the Gram–Schmidt log-norm. Each figure contains several graphs: the red dots are the (averaged) experimental $\log \|\mathbf{b}_i^*\|$'s; the brown dots are obtained by applying the upper bound in Minkowski's first theorem to each one of the experimentally obtained local blocks $\mathbf{B}_{[i, i+\min(i+\beta-1, n)]}$ of the output basis; the purple dots are the values obtained by replacing the upper bound in Minkowski's first theorem by the Gaussian heuristic $\mathrm{GH}(\Lambda_{[i, i+\min(i+\beta-1, n)]})$, and the blue dots are the expected $\lambda_1(\Lambda_{[i, i+\min(i+\beta-1, n)]})$ (see Subsection 2.1.1).

The experiments highlight that the Gaussian heuristic and expected value of first minimum are not very accurate for predicting the output of the BKZ algorithm (and neither is Minkowski's first theorem, but that is less surprising). For small block sizes, the experimental Gram–Schmidt log-norms are above the Gaussian heuristic values. Notice this even appears to happen for the tail blocks for large β. When the block size increases, the Gaussian heuristic and first minimum expectation get closer to each other (except in the tail region), but still do not accurately predict the genuine BKZ output. In particular, the experimental Gram–Schmidt log-norms are concave in the head region (the other curves are also somewhat concave, but less so, as they are smoothed versions of the experimental curve). This is the phenomenon we refer to as head concavity. It starts being quite noticeable with $\beta \approx 30$.

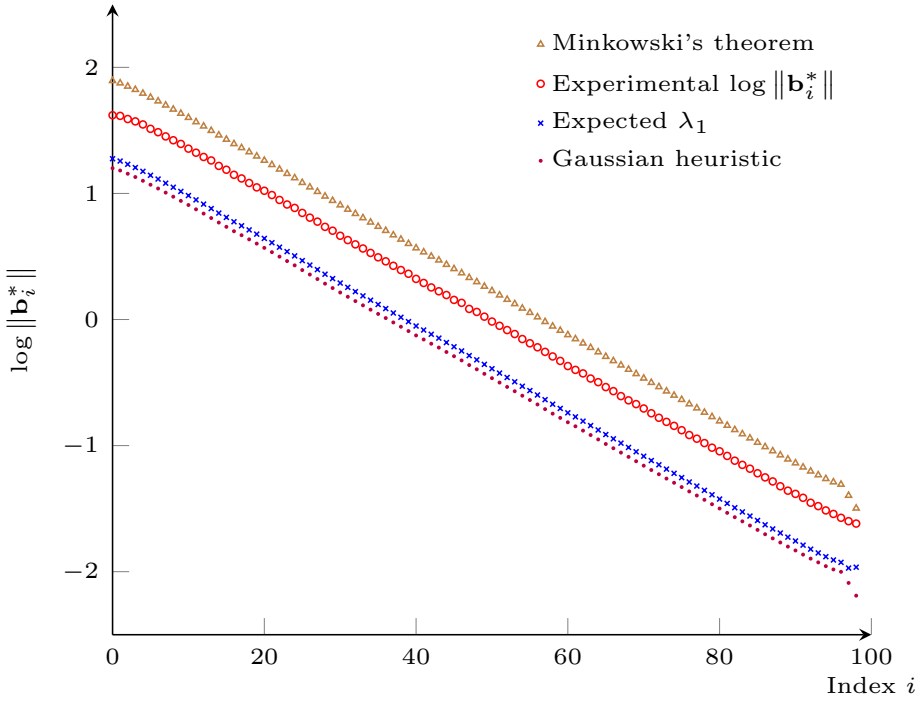

Fig. 5.3: Output of BKZ_4.

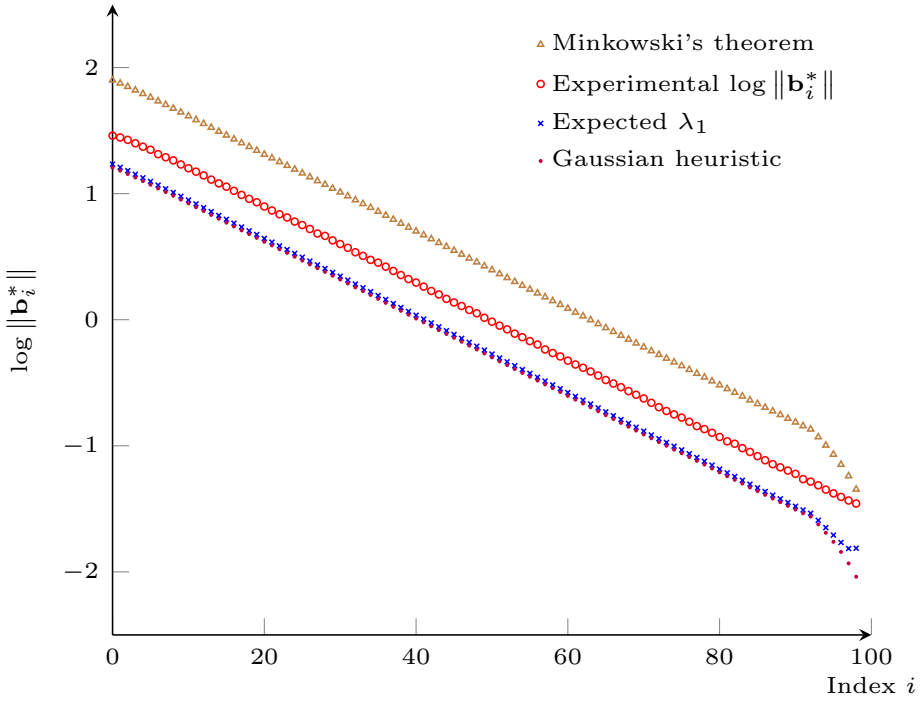

Fig. 5.4: Output of BKZ_8.

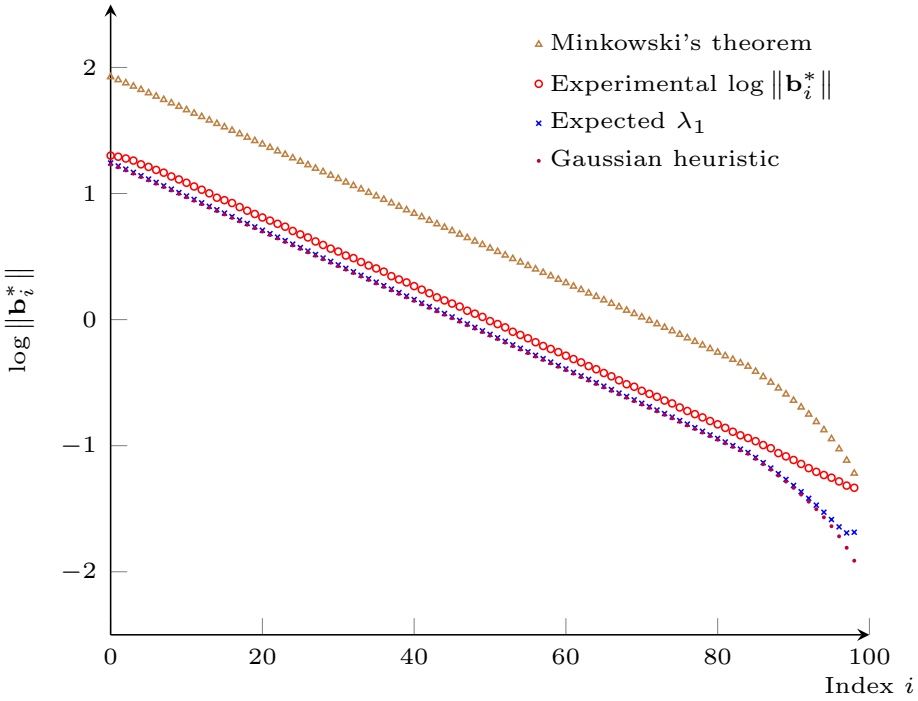

Fig. 5.5: Output of BKZ_{16}.

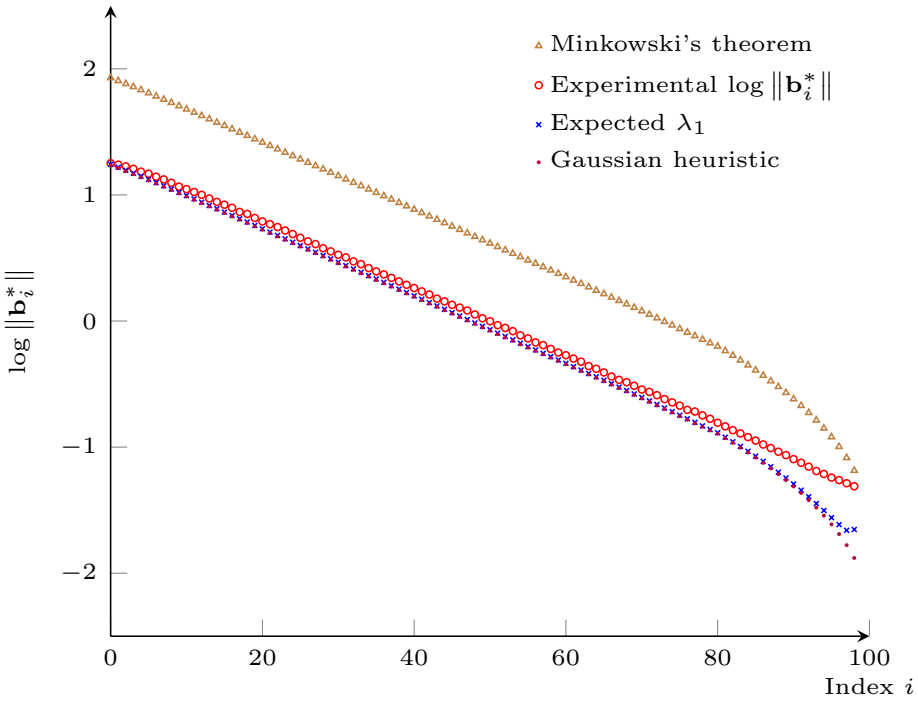

Fig. 5.6: Output of BKZ_{20}.

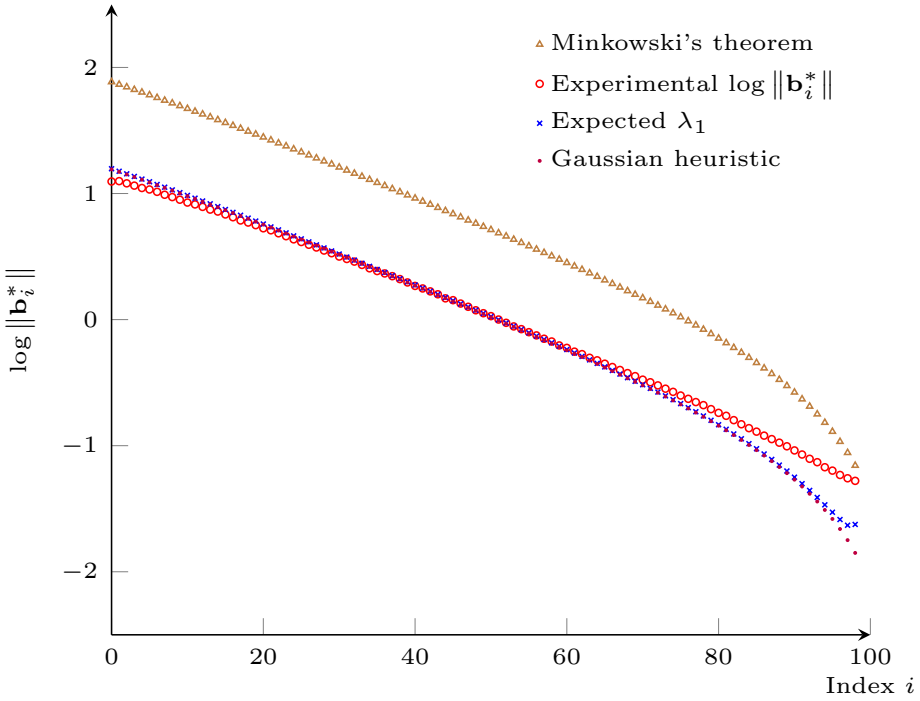

Fig. 5.7: Output of BKZ_{30}.

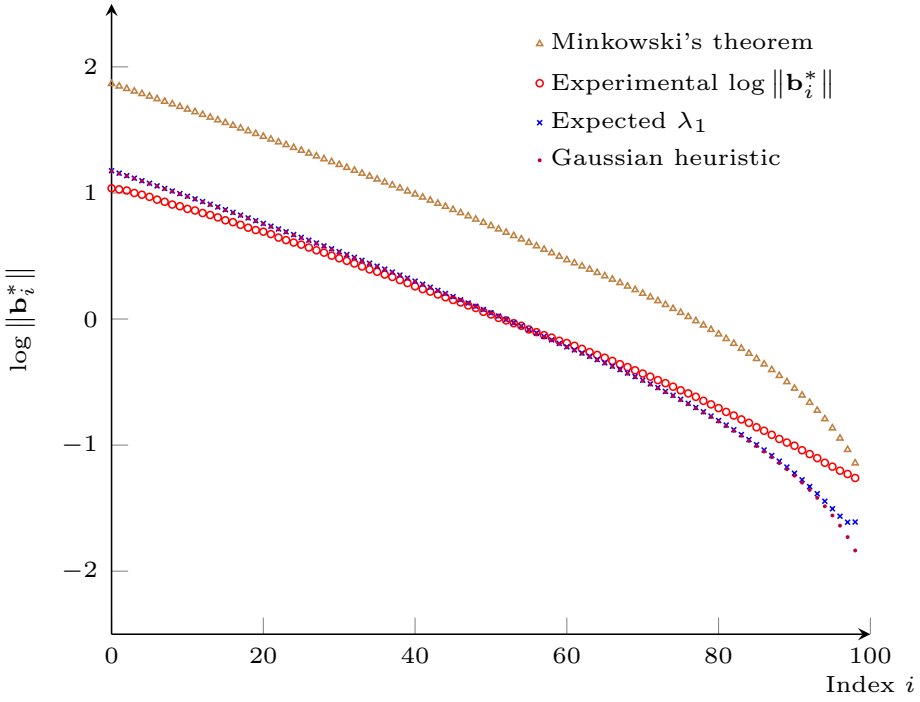

Fig. 5.8: Output of BKZ_{40}.

5.2.2 Enumeration costs in local blocks

The enumeration cost for SVP in each local block is also an interesting quantity for evaluating the extent of the head concavity of a BKZ-reduced basis. It is also the cost for checking that the basis is indeed BKZ_β-reduced. As explained in [HS07], under the Gaussian heuristic, the full enumeration cost (as number of nodes, denoted by "# nodes") of a d-dimensional lattice using enumeration radius $\|\mathbf{b}_1^*\|$ can be estimated by

$$\sum_{k=1}^{d} \frac{1}{2} \cdot \frac{V_k(\|\mathbf{b}_1^*\|)}{\prod_{i=d-k+1}^{d} \|\mathbf{b}_i^*\|}. \tag{5.1}$$

We take the (averaged) BKZ_{40} pre-processed basis from the previous subsection and compute the local SVP costs of SVP_{40}, SVP_{50} and SVP_{60} on the BKZ_{40}-pre-processed basis. In Figure 5.9, we plot the logarithm of the quantity above for each local block for SVP_{40}, SVP_{50} and SVP_{60}. It can be seen that the local SVP costs in the first few blocks are cheaper. The enumeration costs keep increasing until the last $\beta - 1$ blocks. These last blocks are of smaller dimensions, explaining why their enumeration costs become lower. As cheaper enumeration reflects stronger reducedness, the curve in Figure 5.9 reflects a concavity of the $\log \|\mathbf{b}_i^*\|$'s in the head.

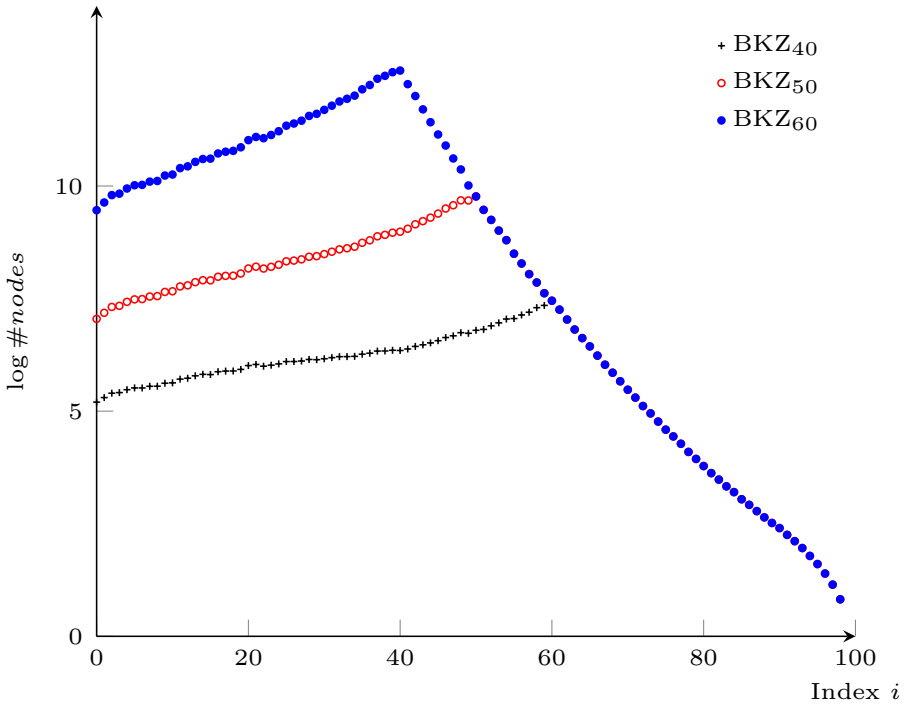

Fig. 5.9: Estimated enumeration costs (of each local block) for BKZ_{40}, BKZ_{50} and BKZ_{60} on a BKZ_{40} reduced basis.

5.2.3 Evolution of the Gram–Schmidt norms during the execution

The previous experiments suggest that the Gaussian heuristic may be inaccurate for a BKZ-reduced basis in the head region. In this subsection, we further investigate the *evolution* of the accuracy of the Gaussian heuristic for each local block $\Lambda_{[i,\min(i+\beta,n)]}$ during the running of the BKZ algorithm. We focus on the evolution of the BKZ_{40} experiments from Subsection 5.2.1. After each BKZ tour, we record $\{\mathbf{b}_i^*\}_{i\in[n]}$ for each experiment. Again we use a full enumeration as SVP-solver (without pruning), to avoid side-effects and wait until BKZ completes.

In Figure 5.10, we plot the Gram–Schmidt log-norms after each tour (for the first 1000 tours, plus those of the initial LLL-reduced input). For each BKZ_{40} experiment, we normalize the log-norms after each tour as in Subsection 5.2.1. Furthermore, we take the average of the log-norms for the 100 experiments (one individual graph would be less smooth). Finally, we plot the log-norms for the first 1000 tours (one BKZ instance completes before 1000 tours; and after this one completes, for a given tour number, we take the average over the BKZ experiments that are running for more than 1000 tours). The dots corresponding to the earlier tours are colored in blue, and those corresponding to the later tours are colored in red (the color changes gradually).

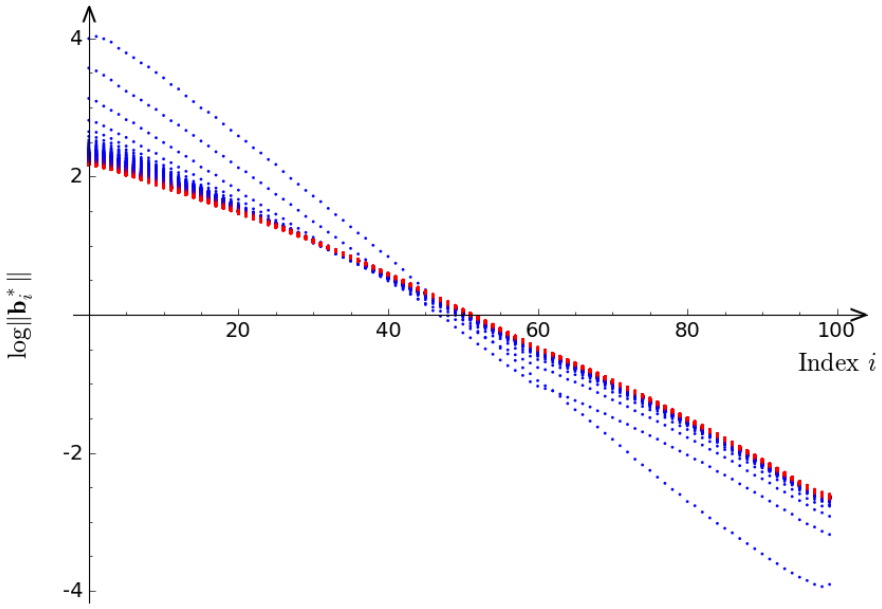

Fig. 5.10: Evolution of the Gram–Schmidt log-norms during BKZ_{40}'s execution.

The plot shows the evolution of the log-norms across tours. However, it does not clearly highlight the evolution of the relation between the $\|\mathbf{b}_i^*\|$'s and the Gaussian heuristic values. Hence we further compute the quantities

$$\frac{\|\mathbf{b}_i^*\|}{\mathrm{GH}(\Lambda_{[i,\min(i+\beta,n)]})} \quad \text{for} \quad i \leq n.$$

Note that this should be expected to be close to 1 for a random lattice, under the Gaussian heuristic. For each tour, for all indices i, we record all the (averaged) quantities at i's across the 100 experiments. We plot a line for each tour and hence Figure 5.11 contains 1001 lines. In Figure 5.11, the x-axis corresponds to the index i; the y-axis corresponds to the quantities above. Note that for each i, there are 1001 dots vertically, corresponding to the number of BKZ tours plus the initial LLL-reduced input.

As observed in prior works, BKZ distorts the distribution of the projected lattices: the first projected sublattices are denser (the minimum is smaller) and the last projected sublattices are sparser. This can be seen from Figure 5.11 since the red points are significantly lower than 1 in the first indices. Further, this distortion occurs often very quickly during the execution of BKZ, sometimes within a few tours.

5.2.4 Evolution of root Hermite factor of the basis

We now consider the asymptote of the root Hermite factor of the basis being BKZ-reduced, as the number of BKZ tours increases. A similar experiment was done in [HPS11]. We compare the experimental behavior to the Chen–Nguyen simulator. We fix the block size at $\beta = 45$ and run BKZ_β on 100

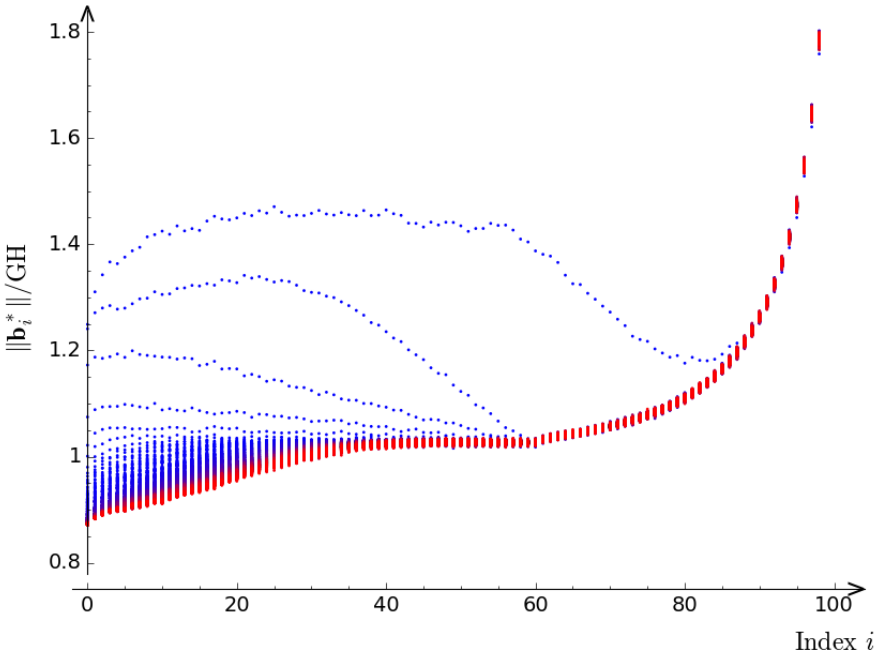

Fig. 5.11: Evolution of the $\|\mathbf{b}_i^*\|$/GH's during BKZ_{40}'s execution.

random instances. After each tour, we record the average root Hermite factor. In Figure 5.21 (we also duplicate it here for convenience), we plot the averaged root Hermite factors over all experiments. The root Hermite factor δ is computed as:

$$\delta = (\|\mathbf{b}_1^*\| / (\det \Lambda)^{1/n})^{1/n}.$$

It can be seen that the evolution of the root Hermite factor does not match with its prediction by the Chen–Nguyen simulator. Indeed, the root Hermite factor obtained with the Chen–Nguyen simulator does not further improve after the first few tours; while it keeps improving in the actual experiments. It should also be noted that in the first few tours, the root Hermite factors in the actual experiments are worse than those of the Chen–Nguyen simulation. This might be due to the fact that the SVP solver is not exact in practical experiments (also the local SVP computation is aborted after a number of trials). In particular, the SVP solver on lattice Λ may stop when it finds a short vector, e.g., with norm within $1.05 \cdot \text{GH}(\Lambda)$. As the number of tours increases, the root Hermite factors in experiments are smaller than those obtained by the Chen–Nguyen simulator.

5.2.5 BKZ with pruning

All of the previous experiments used BKZ without pruning to avoid side-effects on the quality. The purpose of this subsection is to show that using extreme pruning within the enumeration indeed affects the behavior of BKZ to some extent (this was also observed in [YD17]). Nevertheless, the head concavity phenomenon remains visible even in pruned-enumeration BKZ. Again we run the BKZ experiments until they fully complete (i.e., no early-abort).

We consider two experiments. First, we run BKZ_{40} with pruned enumeration and compare it with standard BKZ_{40}.We used the default pruning strategy of `fplll`. We note that there is no known canonically best way to prune, and the experimental results may vary a little across different pruning strategies.We see by comparing Figure 5.13 (with pruned enumeration) with Figure 5.8 (without) that the extent of the head concavity phenomenon is less than without pruned enumeration. In the second

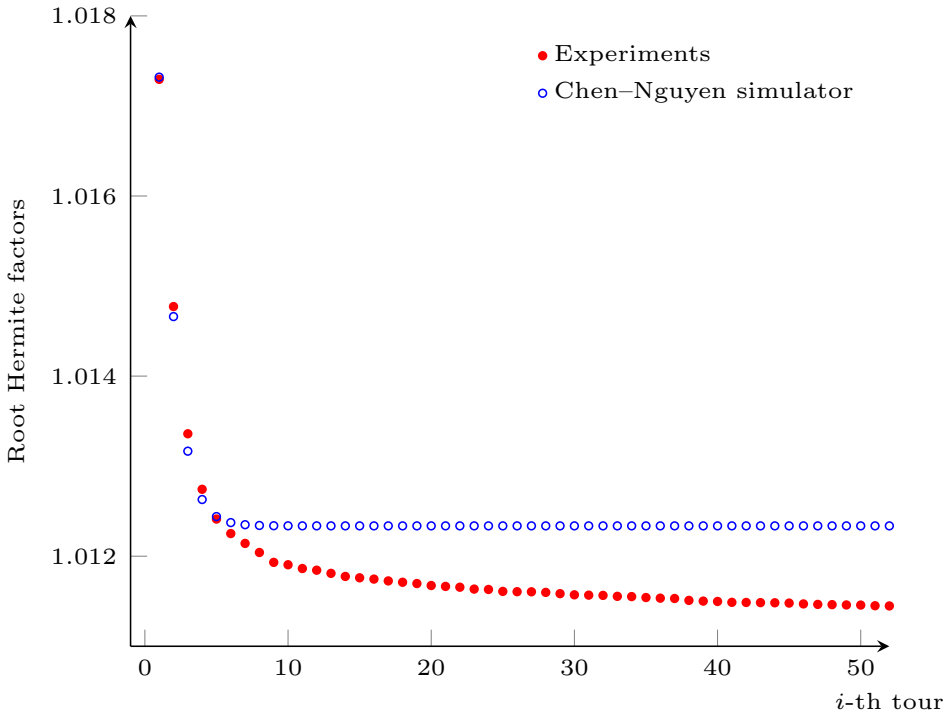

Fig. 5.12: Evolution of Root Hermite factors during the execution of BKZ_{45}.

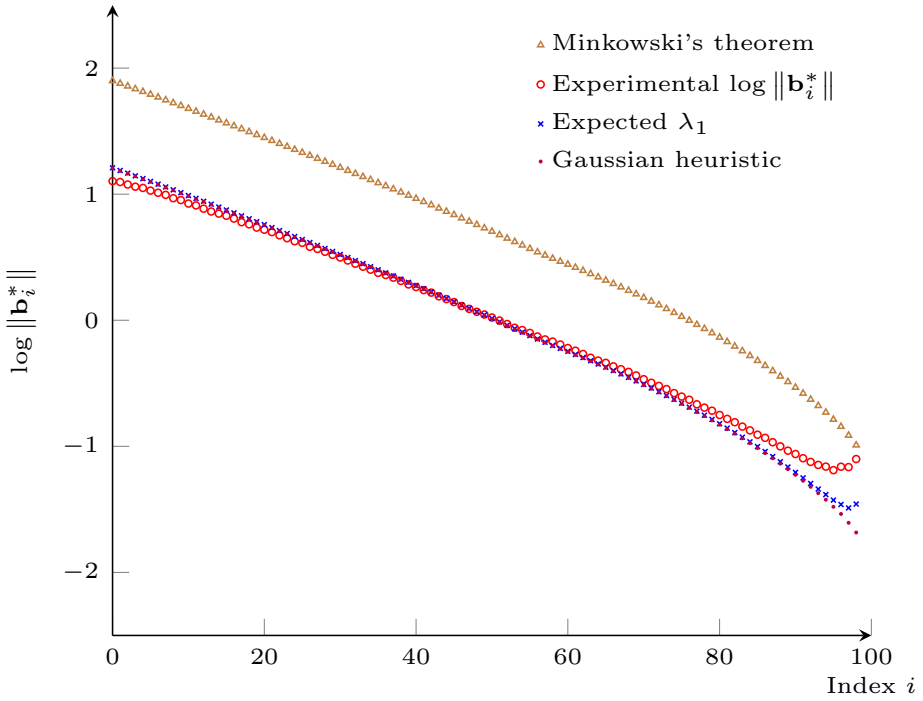

Fig. 5.13: Output Gram–Schmidt log-norms for BKZ_{40} with pruning.

experiment, we run BKZ with pruned enumeration with larger block size, which is also more relevant for cryptanalysis. In particular, we run BKZ_{60} with pruned enumeration and then plot the evolution of the corresponding $\|\mathbf{b}_i^*\|/\mathrm{GH}$'s (for the first 500 tours). This is to be compared with Figure 5.11. We can conclude that the head concavity phenomenon still exists for practical versions of BKZ with larger block sizes.

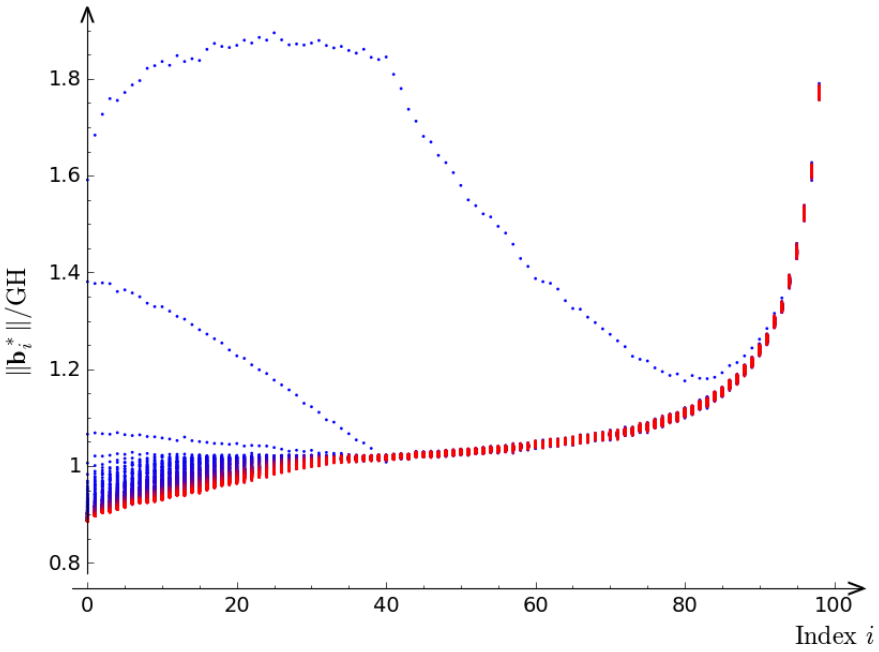

Fig. 5.14: Evolution of the $\|\mathbf{b}_i^*\|/\mathrm{GH}$'s during the execution of BKZ_{60} with pruning.

5.3 A refined BKZ simulator

In this section, we describe a refined BKZ simulator, and report on experiments indicating that the simulation is quite accurate, in particular in capturing the head concavity phenomenon.

5.3.1 The refined simulator

Our probabilistic BKZ simulator aims to provide a more accurate simulation of the experimental behavior of the BKZ algorithm. Our simulator has similar structure as the Chen–Nguyen simulator (refer to Subsection 2.5.4), in particular, we also consider the Gram–Schmidt log-norms. We assume the reader is familiar with the Chen-Nguyen simulator, and, below, we comment on the differences.

The main difference is the emulation of the probabilistic nature of the minimum in random lattices. Theorem 2.1 provides the distribution of the minimum of a uniform unit-volume lattice. The new BKZ simulator, given as Algorithm 10, is probabilistic. It takes this distribution into consideration when updating each local block. In more detail, suppose that we are updating the local block $\Lambda_{[k,e]} = \Lambda(\mathbf{B}_{[k,e]})$ for some $k \leq n - 45$ and $e = \min(k + \beta - 1, n)$ with dimension $d = \min(\beta, n - k + 1)$. Let us assume this is a random lattice. By Theorem 2.1, we have that $\lambda_1(\Lambda_{[k,e]})$ is distributed as

$$\lambda_1(\Lambda_{[k,e]}) = \left(\frac{X \cdot \mathrm{vol}(\Lambda_{[k,e]})}{v_d}\right)^{1/d},$$

where X is sampled with distribution Expo[1/2]. Recall that v_d is the volume of n-dimensional unit ball. Now we can take the logarithm to obtain

$$\log \lambda_1(\Lambda_{[k,e]}) = \frac{\log X + \log \mathrm{vol}(\Lambda_{[k,e]}) - \log v_d}{d}.$$

This explains `Line 14` in Algorithm 10. At `Line 15`, the simulator checks whether a value sampled as above, e.g., $\log \|\mathbf{b}_k^*\|$, is smaller than the current $\log \|\mathbf{b}_k^*\|$. If it is indeed smaller, then the value of $\log \|\mathbf{b}_k^*\|$ is updated. Else the former one is kept. This corresponds to the main step in the BKZ algorithm (`Line 4--7`): once the found vector (in the SVP call) is shorter than current $\mathbf{b}_i^*$ in current local block, then the found vector will be used to replace $\mathbf{b}_i$, thus $\mathbf{b}_i^*$ is updated. Otherwise the found vector will be discarded, and the simulation assumes that $\mathbf{b}_i^*$ is not changed during the LLL-reduction.

A further difference with the Chen–Nguyen simulator is the way we handle the remaining Gram–Schmidt log-norms in the current local block in case $\log \|\mathbf{b}_k^*\|$ has been updated. In the Chen–Nguyen simulator, after updating the first Gram–Schmidt log-norms $\log \|\mathbf{b}_k^*\|$ in the current local block, all the remaining log-norms $\log \|\mathbf{b}_i^*\|$ for $i > k$ will be updated by using the Gaussian heuristic directly without further checking whether the estimated value gives an improvement or not (refer to Algorithm 9). In our simulator, we consider a refined update of the remaining $\log \|\mathbf{b}_i^*\|$'s of the block. Concretely, we update all the remaining indices in the current local block by increasing them by a common amount chosen so that the volume of current block is preserved (it compensates for the decrease of $\log \|\mathbf{b}_k^*\|$). There is one further subtlety in the actual update applied by the simulator. For the second Gram–Schmidt log-norm of the block, it sets $\widehat{\ell}_{k+1} \leftarrow \ell_k + \log(\sqrt{1-1/d})$ rather than increasing it by the same amount as for the $d-2$ remaining log-norms. Here the quantity $\sqrt{1-1/d}$ is used to simulate the updated norm of $\mathbf{b}_k^*$ after removing its projection on the new inserted vector (the found shortest vector of current local block). This twist also follows in experiments: the updated second Gram–Schmidt norm is almost always a bit smaller than the old first Gram–Schmidt norm of the block. Such a strategy also makes the simulator more flexible. In the new simulator, it is not necessary to update all the remaining blocks with the value estimated by the Gaussian heuristic once we have an update: the simulator makes an update only when needed, i.e., when an improving Gram–Schmidt norm is sampled.

We also use two sets of boolean values $\{t_0^{(i)}\}_{i\le n}$ and $\{t_1^{(i)}\}_{i\le n}$, which are used to record if there is a change of $\log \|\mathbf{b}_i^*\|$ in the last and the current tours, respectively. If we know there was no change at all in current local block during the last tour, we simply skip the current block and go to the next one. Correspondingly, in the real procedure of BKZ, it means that the found shortest vector in current block in this tour will be the same as the one in current block in the previous tour. As it was not used to make an update during the previous tour, so will it not be used in the current tour.

As the Chen–Nguyen simulator is deterministic, it terminates relatively fast, within a few hundreds of tours typically. Oppositely, our probabilistic simulator may perform far more tours and continue making further (though smaller and smaller) Gram–Schmidt progress. Another difference between the behaviors of the simulators comes from the fact that the expectation of a given sample (corresponding to g in `line 14` of Algorithm 10) for updating each local block in our simulator is slightly larger than the one used in the Chen–Nguyen simulator that uses the Gaussian heuristic (but they become closer to each other as the block size increases). As a result, the sampled value for the first minimum of a local block can be slightly larger than in the Chen–Nguyen simulator. However (and more importantly), the chosen value can also be smaller than the Gaussian heuristic, and in fact smaller than the current value even if that one is already quite small. This is exactly what makes the Gram–Schmidt log-norms progress further in the simulations and make the simulations closer to the practical behavior of BKZ.

Algorithm 10 The probabilistic BKZ simulator

Require: The Gram–Schmidt log-norms $\{\ell_i = \log \|\mathbf{b}_i^*\|\}_{i \le n}$ and an integer $N \ge 1$.
Ensure: A prediction of the Gram–Schmidt log-norms after N tours of BKZ.

for $i = 1$ `to` 45 **do** $r_i \leftarrow \mathbb{E}[\log \|\mathbf{b}_k^*\| : \mathbf{B}$ HKZ-reduced basis of $\Lambda \leftarrow \Gamma_{45}]$
end for
$t_0^{(i)} \leftarrow$ `true`, $\forall i \le n$
for $j = 1$ `to` N **do**
  $t_1^{(i)} \leftarrow$ `false`, $\forall i \le n$
  for $k = 1$ `to` $n - 45$ **do**
    $d \leftarrow \min(\beta, n - k + 1)$; $e \leftarrow k + d$
    $\tau \leftarrow$ `false`
    for $k' = k$ `to` e **do** $\tau \leftarrow \tau \| t_0^{(k')}$
    end for
    $\log \mathrm{vol}(\Lambda_{[k,e]}) \leftarrow \sum_{i=1}^{e-1} \ell_i - \sum_{i=1}^{k-1} \widehat{\ell}_i$
    if $\tau =$ `true` **then**
      $X \hookleftarrow$ `Expo`$[1/2]$
      $g \leftarrow (\log X + \log \mathrm{vol}(\Lambda_{[k,e]}) - \log v_d)/d$
      if $g < \ell_k$ **then**
        $\widehat{\ell}_k = g$
        $\widehat{\ell}_{k+1} \leftarrow \ell_k + \log(\sqrt{1 - 1/d})$
        $\gamma \leftarrow (\ell_k + \ell_{k+1}) - (\widehat{\ell}_k + \widehat{\ell}_{k+1})$
        for $k' = k + 2$ `to` e **do**
          $\widehat{\ell}_{k'} \leftarrow \ell_{k'} + \gamma/(d-2)$
          $t_1^{(k')} \leftarrow$ `true`
        end for
        $\tau \leftarrow$ `false`
      end if
    end if
    $\{\ell_k, \cdots, \ell_{e-1}\} \leftarrow \{\widehat{\ell}_k, \cdots, \widehat{\ell}_{e-1}\}$
  end for
  $\log \mathrm{vol}(\Lambda_{[k,e]}) \leftarrow \sum_{i=1}^{n} \ell_i - \sum_{i=1}^{n-45} \widehat{\ell}_i$
  for $k' = n - 44$ `to` n **do**
    $\widehat{\ell}_{k'} \leftarrow \frac{\log \mathrm{vol}(\Lambda_{[k,e]})}{45} + r_{k'+45-n}$
    $t_1^{(k')} \leftarrow$ `true`
  end for
  $\{\ell_1, \cdots, \ell_n\} \leftarrow \{\widehat{\ell}_1, \cdots, \widehat{\ell}_n\}$
  $\{t_0^{(1)}, \cdots, t_0^{(n)}\} \leftarrow \{t_1^{(1)}, \cdots, t_1^{(n)}\}$
end for

5.3.2 Heuristic justification

We now give a heuristic explanation as to why the probabilistic simulator (and BKZ) keeps making progress even after some significant amount of time. Every time it considers a block, it keeps trying to find a shorter vector than the current first vector of the block, thanks to fresh random sampling. Let $X \hookleftarrow \mathrm{Expo}(1/2)$ and $Y = X^{1/n}$. Assume for simplicity that there is only one block (i.e., $n = \beta$) and that the lattice Λ has been scaled so that the volume of the lattice is 1, which implies that $\lambda_1(\Lambda)$ has the same distribution as Y. The CDF of Y is

$$F(y) = 1 - \mathrm{e}^{-y^n/2}.$$

Let $Y_{\min,K}$ be the minimum among K independent Y_i's. Its CDF and PDF are

$$F_{\min,K}(y) = 1 - \mathrm{e}^{-Ky^n/2} \quad \text{and} \quad f_{\min,K}(y) = Kny^{n-1}\mathrm{e}^{-Ky^n/2}/2,$$

respectively. We can hence compute the expected value

$$\mathbb{E}(Y_{K,min}) = (2/K)^{1/n} \cdot \Gamma(1 + 1/n) = \mathbb{E}(\lambda_1(\Lambda))/K^{1/n}.$$

One sees that the expectancy keeps decreasing, although much more slowly as K increases. Notice that K here can be regarded as proportional to the number of blocks treated in our probabilistic simulator. We conjecture that BKZ is enjoying a similar phenomenon.

This simple model does not work for explaining BKZ with a single block (because for a single block, once the SVP instance has been solved, it cannot be improved further). In the more interesting case where $\beta < n$, the fact there are many intertwined blocks helps as an improvement for one block 'refreshes' the neighbouring blocks, which then have a chance to be improved. In this case, however, this simple model does not capture the impact of one block on the neighbouring blocks, nor the fact that the SVP instances across blocks are not statistically independent (in particular, the blocks overlap).

5.3.3 Quality of the new simulator

In this subsection, we describe experiments aiming at assessing the accuracy of our probabilistic BKZ simulator. We measure the quality of our simulator by comparing with the practical BKZ and the Chen-Nguyen simulator using two quantities: the Gram–Schmidt log-norms after certain tours and the root Hermite factors. We then describe some limitations of our simulator.

(a) Graph of Gram–Schmidt log-norms. In practice, the full sequence of Gram–Schmidt log-norms is important for evaluating the quality of a basis. For example, if we extrapolate the log-norms by a straight line, the slope of the line can be used to indicate whether the basis is of good quality or not. For this reason, we are interested in how accurately the simulator predicts the evolution of the full sequence of Gram–Schmidt log-norms during the BKZ execution. We consider the following two experiments:

(1) The input lattices are SVP-100 instances and we use BKZ_{45} without pruned enumeration during up to $2,000$ tours. Note that for this experiment, the setup is the same as the one used in Subsection 5.2.4.

(2) The input lattices are SVP-150 instances and we use BKZ_{60} with pruned enumeration up to $20,000$ tours.

We record the full log-norm sequences at the end of selected tours. As shown in Figures 5.15–5.20, after a few tours, both the new BKZ simulator and the Chen–Nguyen simulator approach the experimental

behavior of BKZ. As the number of BKZ tours increases, both the experimental BKZ and the probabilistic BKZ simulator evolve, and the corresponding log-norms eventually become concave in the head region. However, the Chen–Nguyen simulator stops making progress after a few tours. By comparison, the new simulator fits the experimental results quite accurately in both situations.

(b) Root Hermite factor. In Subsection 5.2.4, we have seen that the asymptotes (for a large number of tours) of the root Hermite factors obtained with the genuine BKZ algorithm and the Chen–Nguyen simulator diverge. Here in Figures 5.21 and 5.22, we investigate the behavior of the root Hermite factor obtained with the new probabilistic simulator, when the number of tours increases. One can observe that, after a few tours, the probabilistic simulator predicts the experimental data more closely. One can also observe that, in the very first several tours, neither the Chen–Nguyen simulator nor the probabilistic simulator is very accurate. In the case of pruned enumeration, the experimental root Hermite factors drop faster than the simulators; while with non-pruned enumeration, the root Hermite factors in experiments evolve more slowly. This may be due to the algorithmic and implementation complications brought by the pre-processing, the SVP solver, pruning and post-processing.

(c) Limitations of the new simulator. The probabilistic simulator does not fully match with the experimental behavior of BKZ in the first few tours. In particular, the progress of the real Gram–Schmidt log-norms with BKZ with non-pruned enumeration is slower than the simulator's (refer to Figure 5.21). One potential reason is the local SVP solver only attempts to find a vector smaller than $\|\mathbf{b}_i^*\|$. On the other hand, it may be observed that the progress of the real Gram–Schmidt log-norms is a bit faster than the simulated log-norms in the very first BKZ tours, for BKZ with pruned enumeration (refer to Figure 5.22). One potential reason could be that the pruned enumeration uses extensive pre-processing in a local block (which is not captured by the simulator), and this helps to lower the root Hermite factor in the beginning of the execution. However, as soon as the number of tours increases, the probabilistic phenomenon seems to weigh more and the new simulator becomes accurate.

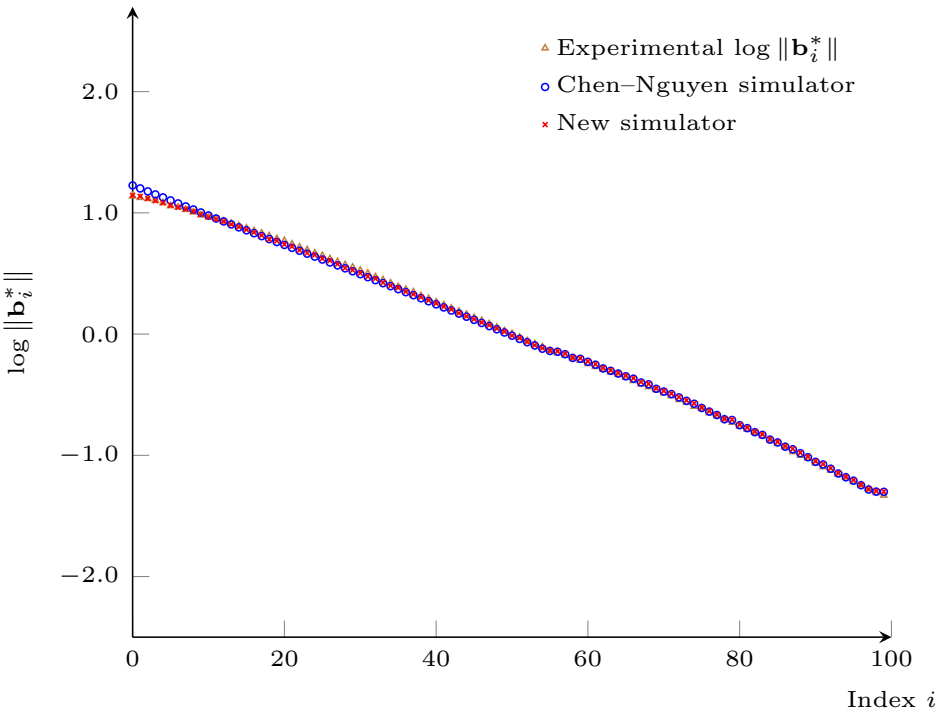

Fig. 5.15: Gram–Schmidt log-norms for BKZ_{45} at tour 50.

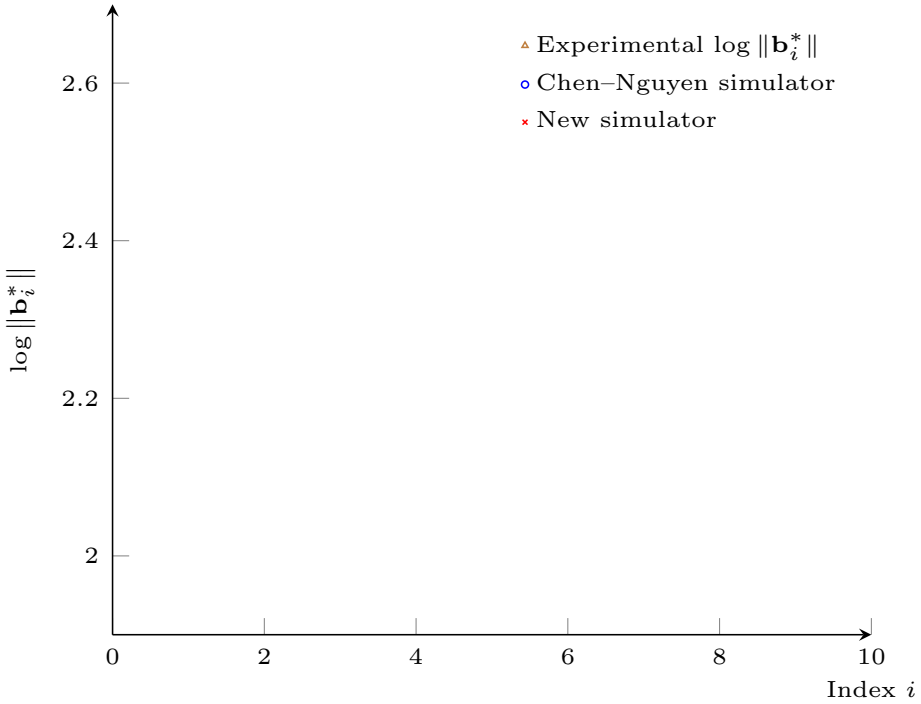

Fig. 5.16: Same as Figuer 5.15, but zoomed in.

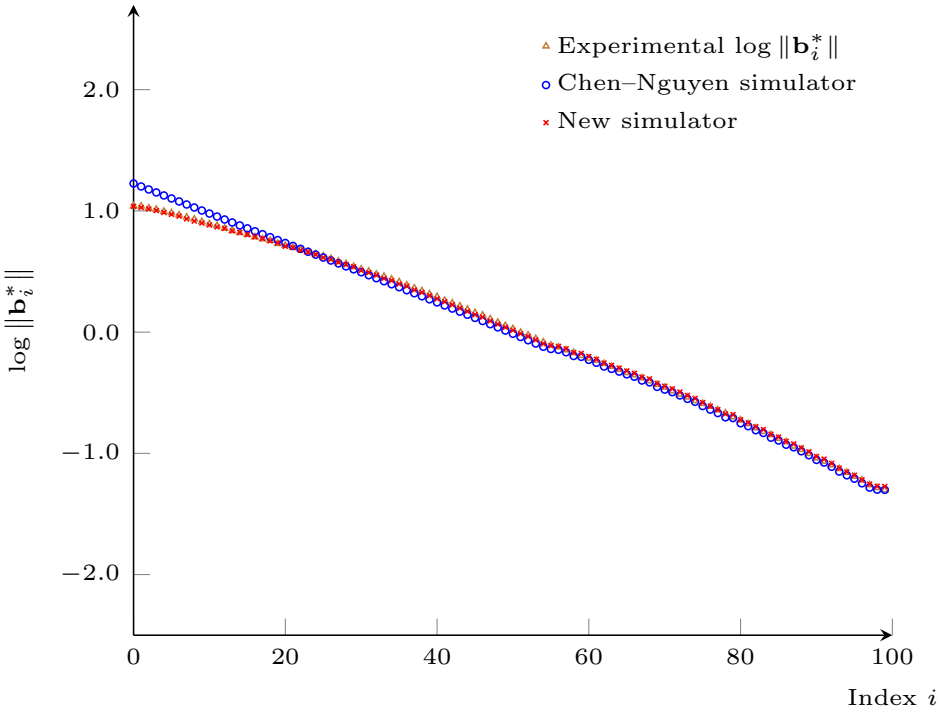

Fig. 5.17: Gram–Schmidt log-norms for BKZ_{45} at tour $2,000$.

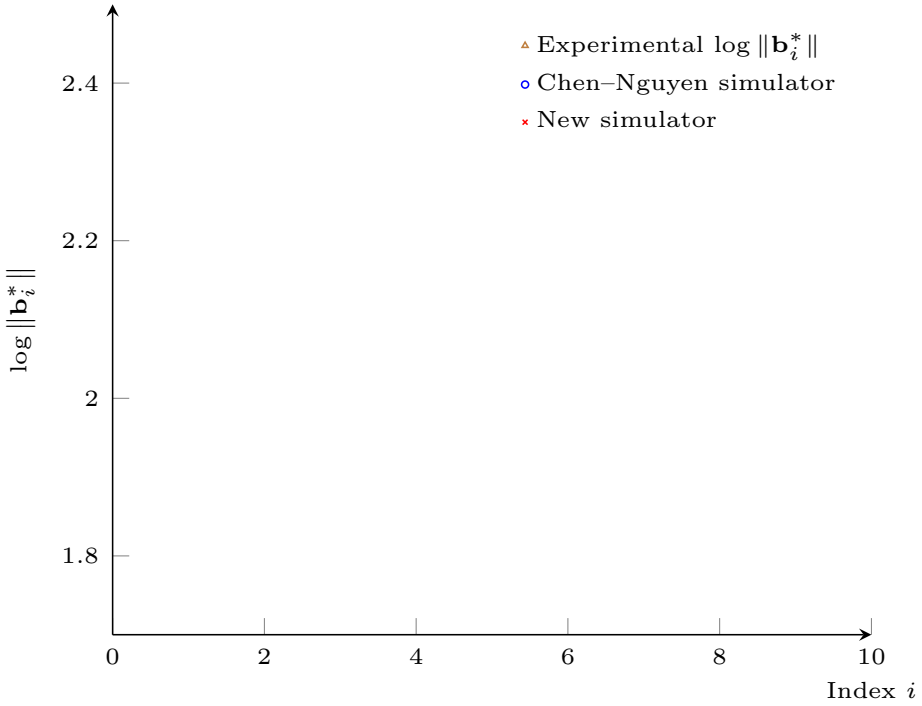

Fig. 5.18: Same as Figure 5.17, but zoomed in.

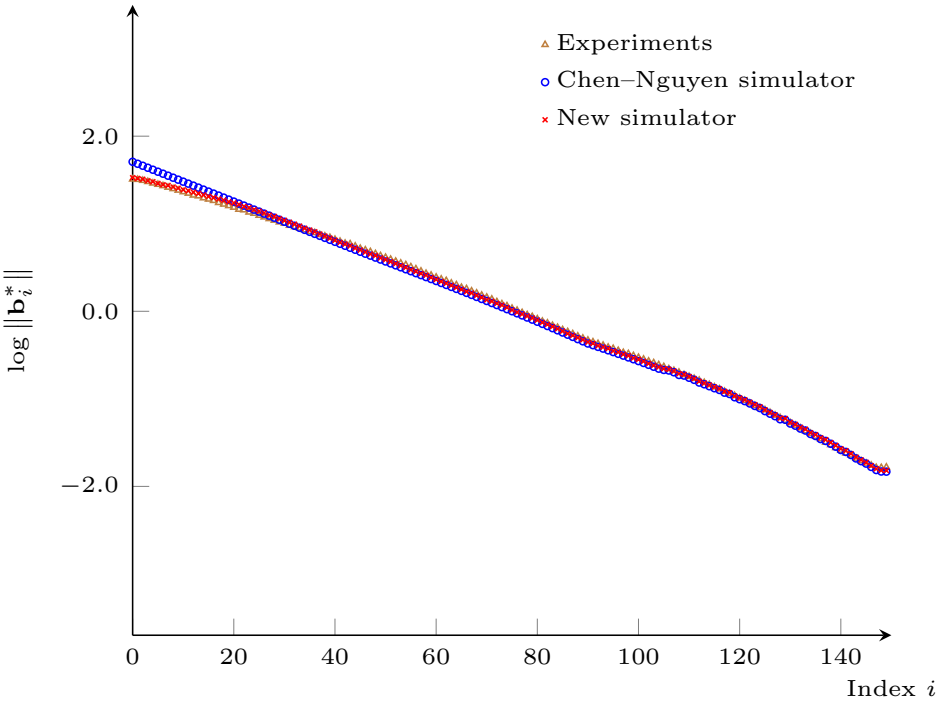

Fig. 5.19: Gram–Schmidt log-norms for BKZ_{60} at tour $20,000$.

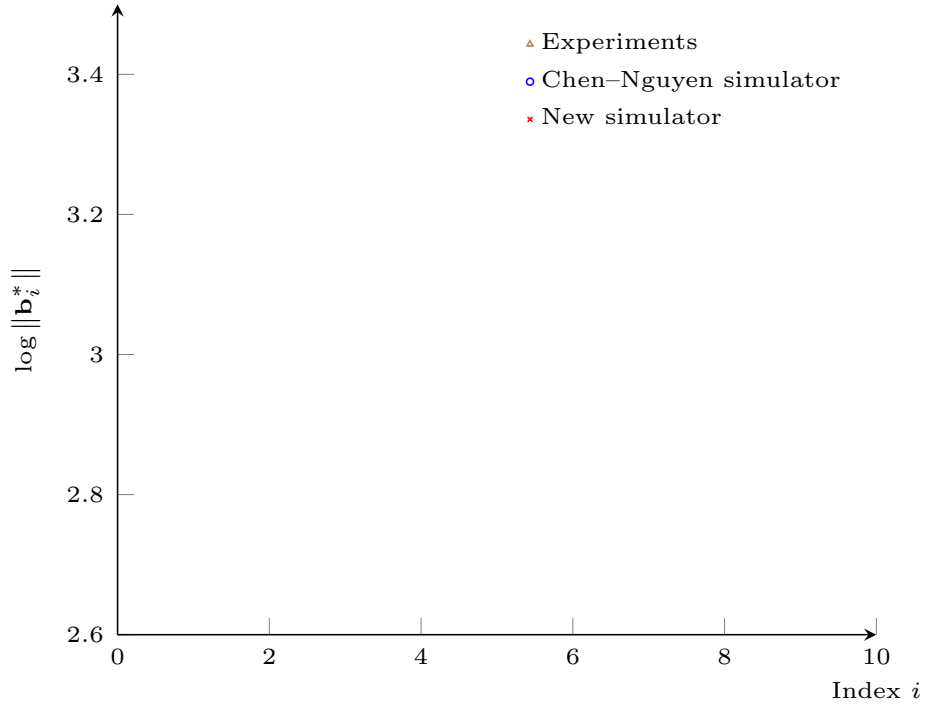

Fig. 5.20: Same as Figure 5.19, but zoomed in.

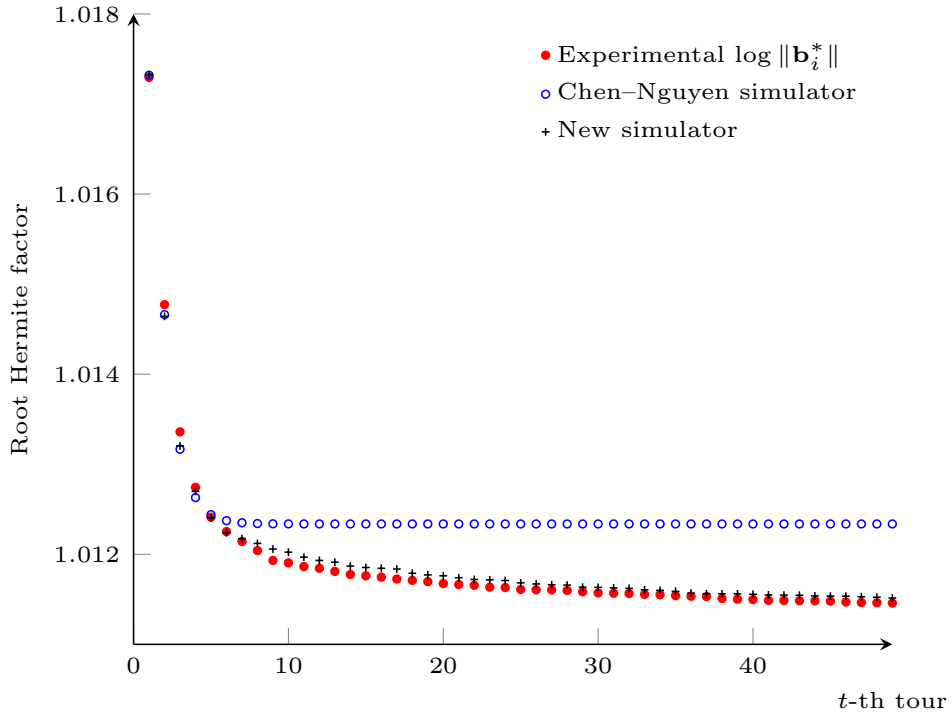

Fig. 5.21: Evolution of the root Hermite factor during the execution of BKZ_{45} (no pruned enumeration) on SVP-100.

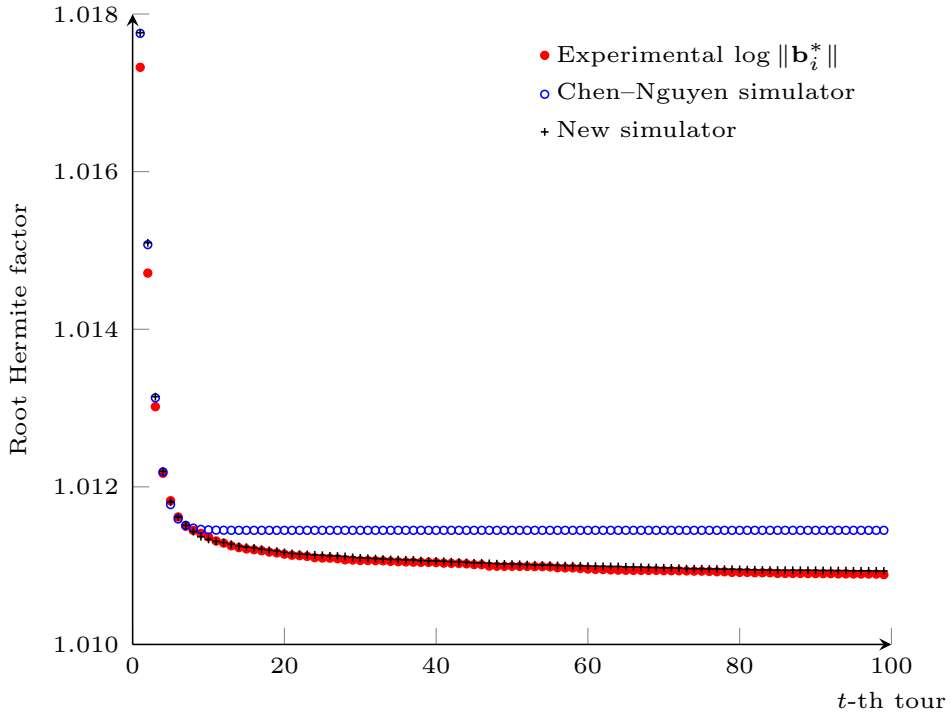

Fig. 5.22: Evolution of the root Hermite factor during the execution of BKZ_{60} (with pruned enumeration) on SVP-150.

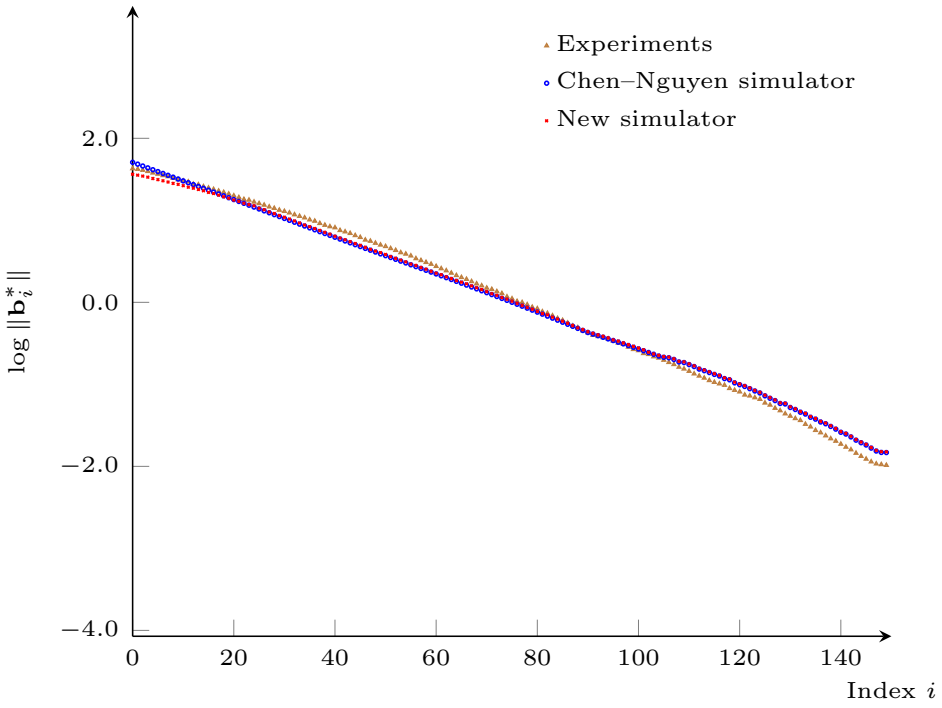

Fig. 5.23: Comparison of Gram–Schmidt log-norms obtained by the simulators and BKZ_{60} (no preprocessing) on SVP-150, after 4,000 tours.

Next, we verify if the pre-processing indeed helps make the experimental root Hermite factor decrease faster than the corresponding quality in the simulator. To verify the impact of pre-processing in pruned enumeration, we run BKZ_{60} (with pruned enumeration) without pre-processing on SVP_{150} instances. We plot the root Hermite factor of of the basis after each tours (up to 100 tours). Each data is taken averaged over 100 results. As shown in Figure 5.23, the BKZ variant without pre-processing makes progress no faster than the simulator. This suggests that pre-processing indeed accelerates the progress made by BKZ. On the other hand, we can also observe that without pre-processing, the root Hermite factor decrease more slowly than the corresponding quality in the simulator. One possible reason for this could be that the vector found by extreme pruning in each local block is slightly longer than the minimum of the local lattice.

In conclusion, it seems quite difficult to use the simulator to estimate the precise evolution of Gram–Schmidt log-norms for the first few tours due to the following two reasons: (1) it is unclear how much improvement is provided by the pre-processing; (2) we do not have a precise understanding on the distribution of norms of vectors output by the enumeration (ideally, with pruning). On the other hand, after the first few tours, the simulator seems to be more accurate when estimating the Hermite factors, which is important for cryptographic applications.

5.3.4 Predicting the root Hermite factor for large block sizes

As our proposed simulator predicts real BKZ quite well for the range of block sizes for which such experiments can be run, we expect that our simulator keeps this accuracy for larger block sizes. This is in particular relevant in cryptanalysis and for security analyses of concrete lattice-based cryptosystems. Indeed, many of the existing security analyses rely on the root Hermite factor predicted by the Chen–Nguyen simulator (see [ACD+] and the references therein), which, as we have seen, is an over-estimate. We thus run the simulators for large block sizes and large dimensions, to assess how the discrepancy scales.

In this experiment, we consider two cases:

(1) the dimension n is much larger than the block-size β.

(2) the dimension n is a small constant times larger than the block-size β.

Case (1) is a scenario often considered to assess the quality of BKZ-type algorithms (see, e.g., [CN11]). On the other hand, in practice, we are also interested in Case (2) where the dimension/block-size ratio is small. This is a typical situation for the lattice-based NIST candidates (see [ACD+]). Concretely, in the first case, we run our simulator of BKZ with block-size $\beta \in [50, 250]$ on 1000-dimensional lattices and our simulator on BKZ with block-size $\beta \in [260, 300]$ in 2000-dimensional lattices, both with 20,000 tours. In the second case, we run the same experiment as above except with a fixed ratio of 3 between dimension and block-size. Each data point is averaged over 10 samples. We plot the root Hermite factor computed as $\delta = e^{(\|\mathbf{b}_1^*\| - (\sum_{i\in[n]} \|\mathbf{b}_i^*\|)/n\|)/n}$, where $\|\mathbf{b}_i^*\|$'s are the final simulated Gram–Schmidt norms output by our simulator.

As can be seen in Figures 5.24–5.25, for large block sizes, the discrepancy vanishes: both simulators converge to the same root Hermite factors. This may be explained by considering the distribution of the minimum of a uniform unit-volume lattice, used in the probabilistic simulator. The expectation is $2^{1/\beta} \cdot \Gamma(1 + 1/\beta)$, which converges to 1, the Gaussian heuristic value (when β grows to infinity). Further, as we have seen in Subsection 2.1.1, the variance of the selected value is $2^{2/\beta} \cdot \left(\Gamma\left(1 + 2/\beta\right) - \left(\Gamma\left(1 + 1/\beta\right)\right)^2\right) \cdot v_\beta^{-2/\beta}$, which decreases to 0 as $O(1/\beta^2)$, making the distribution "more concentrated" and lowering the chance of being "lucky" in finding unexpectedly short vectors in local lattices.

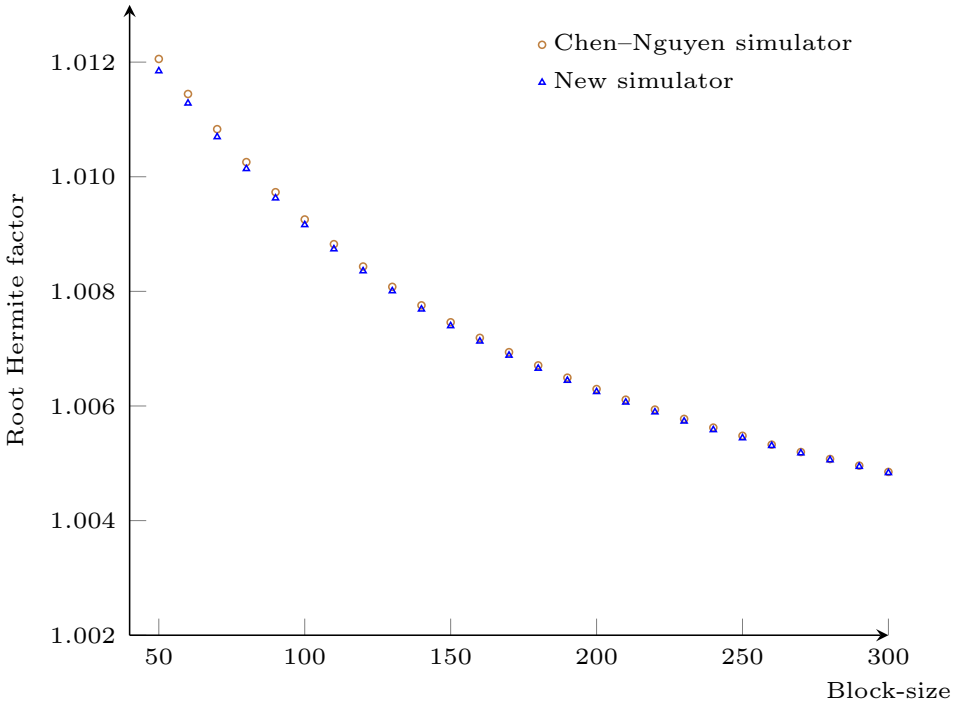

Fig. 5.24: Root Hermite factor for selected $\beta \in \{50, 60, \cdots, 300\}$. Here the dimension is 1000 for $\beta \in [50, 250]$ and 2000 for $\beta \in [260, 300]$.

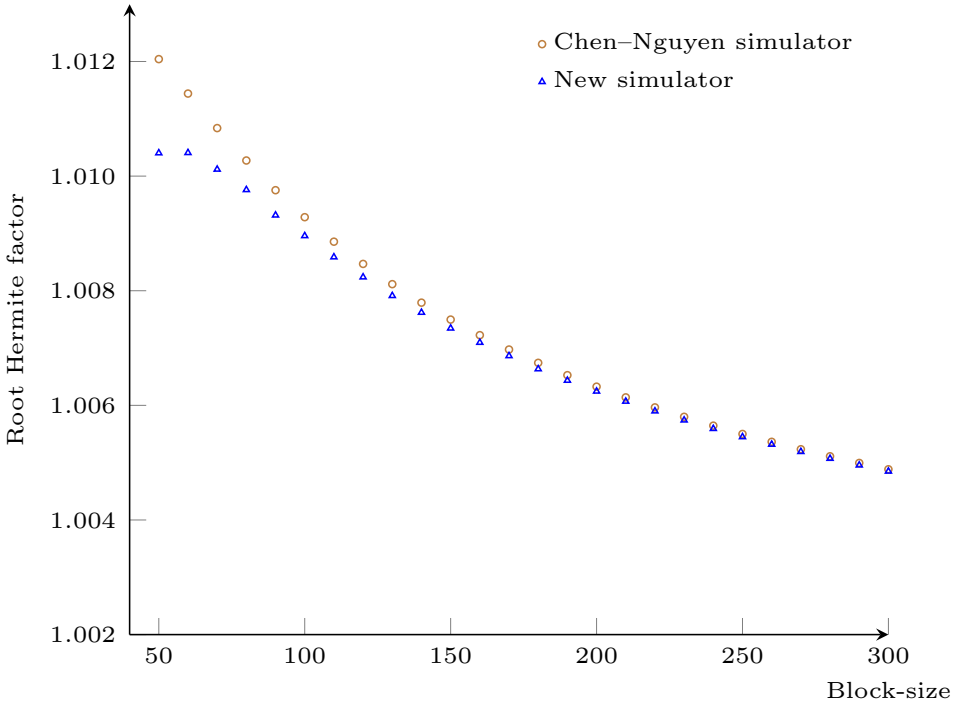

Fig. 5.25: Root Hermite factor for selected $\beta \in \{50, 60, \cdots, 300\}$. Here the dimension is $3 \cdot \beta$.

5.4 Pressing the concavity

In this section, we propose a new BKZ variant. For practical purposes, we further twist this new algorithm with several different strategies. We also quantify the quality of the obtained lattice bases.

5.4.1 Pressed-BKZ

Below, we first describe the new BKZ variant, pressed-BKZ, and then explain why it provides an improvement. Pressed-BKZ is described as Algorithm 11.

Algorithm 11 The pressed-BKZ algorithm

Require: A basis $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$, a block-size $\beta \geq 2$ and a constant $\delta < 1$.
Ensure: A basis of $\Lambda(\mathbf{B})$.

```
1: for s = 1 to n − β + 1 do                    // progressive starting point
2:     Re-randomize the projected lattice Λ_[s,n].
3:     repeat
4:         for k = s to n − 1 do
5:             Find b such that ‖π_k(b)‖ = λ_1(Λ_[k,min(k+β−1,n)])
6:             if δ · ‖b*_k‖ > ‖b‖ then
7:                 LLL-reduce(b_1, ··· , b_{k−1}, b, b_k, ··· , b_{min(k+β,n)}).
8:             else
9:                 LLL-reduce(b_1, ··· , b_{min(k+β,n)}).
10:            end if
11:        end for
12:    until no change occurs (or other condition).
13: end for
```

The pressed-BKZ algorithm runs standard BKZ on block $\Lambda_{[s,n]}$ with an incrementally increased starting index $s \in [1, n-\beta+1]$. In particular, in the case of $s = 1$, pressed-BKZ executes standard BKZ. Note that in `line 12`, "no change occurs" means that no local block was updated during the last tour (from $k = s$ to $k = n-1$). The difference between pressed-BKZ and standard BKZ starts with $s > 1$. At that stage, it does not run BKZ on the full lattice basis anymore. Instead, it freezes the first $s-1$ lattice vectors $\{\mathbf{b}_i\}_{i\in[1,s-1]}$ and re-randomizes the projected lattice $\Lambda_{[s,n]}$, then runs standard BKZ on the projected lattice. Note that the re-randomization is necessary, otherwise, after the BKZ reduction on $\Lambda_{[1,n]}$, no improvement will happen in BKZ reduction on $\Lambda_{[2,n]}$ in the second iteration and neither will it in the following iterations. In particular, in the second iteration, the re-randomization helps to re-randomize the basis vectors of the projected lattice $\Lambda_{[2,n]}$, thus gives a chance of generating a denser leading block of $\Lambda_{[2,n]}$ via BKZ reduction. The re-randomization on the projected lattice is done via transforming the basis of the projected lattice with a unimodular matrix. Here, we use the unimodular matrix generated in `fplll` library (within the `rerandomize_block` function of the `BKZReduction` class).

The design rationale is as follows. Suppose BKZ creates a head concavity for the Gram–Schmidt log-norms. Then the first iteration with $s = 1$ will help to lower $\log \|\mathbf{b}_1^*\|$. The iteration with $s = 2$ will preserve $\log \|\mathbf{b}_1^*\|$ and help to lower $\log \|\mathbf{b}_2^*\|$, etc. This explains the name of the algorithm.

5.4.2 On the behavior of pressed-BKZ

The goal of pressed-BKZ is to further improve the quality of bases obtained by the original BKZ algorithm without a block-size increase. In order to illustrate the idea, we run standard BKZ_{60} on 120-dimensional random lattices (generated in the same way as mentioned at the start of Section 4) with 500 tours (i.e., with early-abort) first, and then run pressed-BKZ with the same number of tours

in each iteration with start index $s = 2$. Each data point is averaged over 100 samples. As shown in Figure 5.26, pressed-BKZ successfully presses the "head concavity" that was produced by the standard BKZ algorithm.

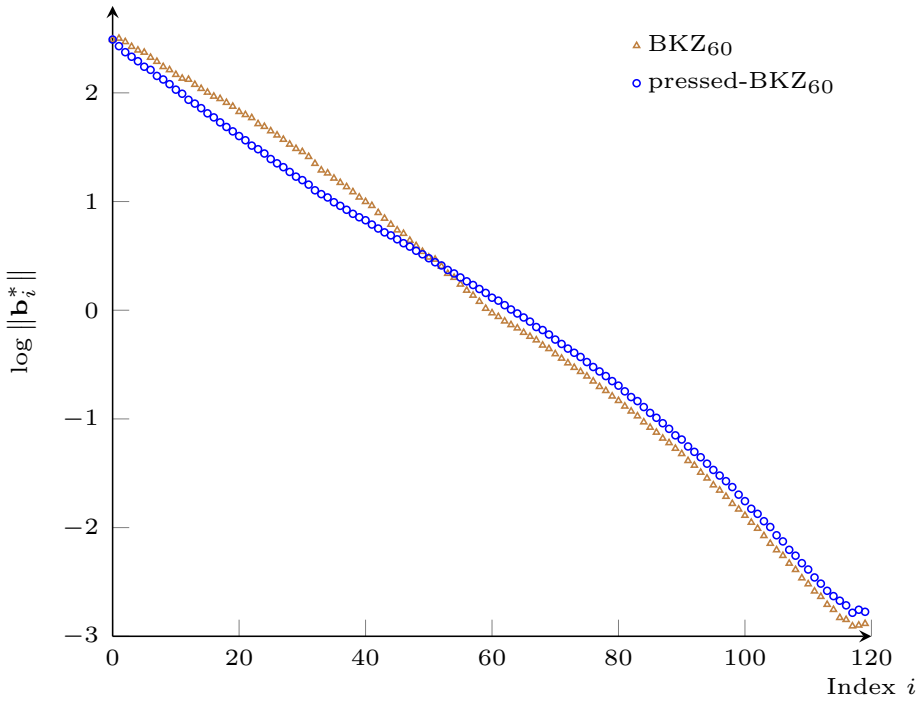

Fig. 5.26: Full sequences of Gram–Schmidt log-norms of bases returned by BKZ_{60} and pressed-BKZ_{60}.

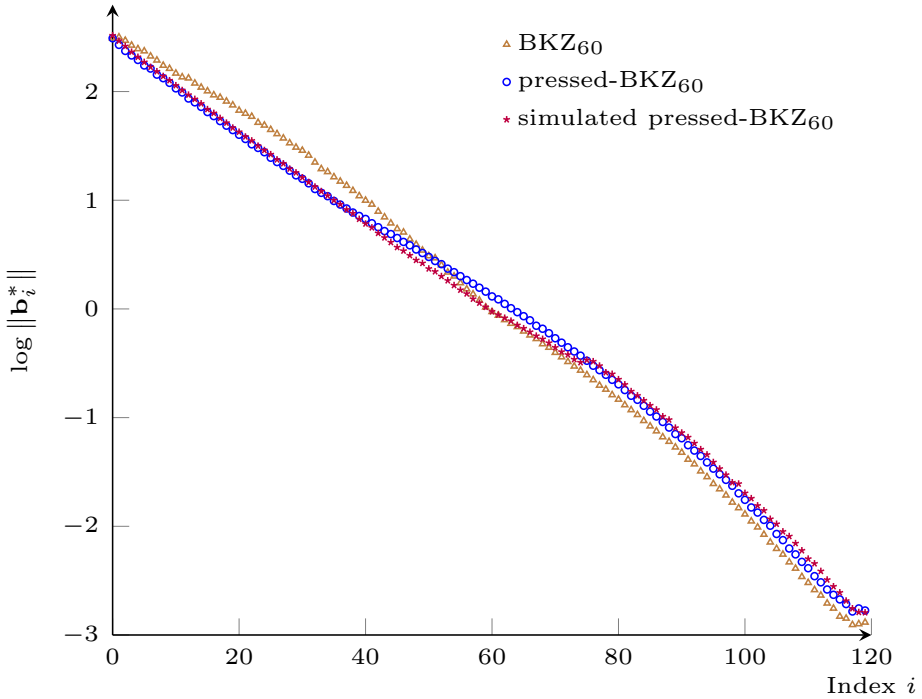

Fig. 5.27: Full sequences of Gram–Schmidt log-norms of bases returned by BKZ_{60}, pressed-BKZ_{60} and simulated pressed-BKZ_{60}.

From the experiment above, we can already see that pressed-BKZ produces a basis with better quality, as its corresponding Gram–Schmidt log-norms achieve a smaller slope. Next, we try to assess by how much pressed-BKZ improves standard BKZ in this respect. We first adapt the simulator for BKZ from Section 5.3 in the direct way to simulate pressed-BKZ. Before we go further, we check the accuracy of our simulator when simulating the behavior of pressed-BKZ by running the same experiment above, but with the simulator. As shown in Figure 5.27, our simulator produces a result that is close to the one experimentally obtained with pressed-BKZ.

Now that we have an accurate simulator for pressed-BKZ, we can proceed to check the behavior of pressed-BKZ further. For this, we run our simulator of pressed-BKZ for block-sizes between 50 and 300, with many tours. We again consider two cases:

(1) the dimension n is much larger than the block-size β.

(2) the dimension n is a small constant times larger than the block-size β.

In the first case, we simulate pressed-BKZ with block-size $\beta \in [50, 250]$ on 1,000-dimensional lattices with 5,000 tours for each iteration, and pressed-BKZ with block-size $\beta \in [260, 300]$ on 2,000-dimensional lattices with 10,000 tours. Each data is averaged over 10 samples. In the second case, we run the same experiment as above except with the dimension/block-size ratio set to 3. Further, we recall the data produced by the Chen–Nguyen simulator, for the sake of a comparison. Note that we can also adapt the Chen–Nguyen simulator for pressed-BKZ, which however, gives the same result as the simulation for standard BKZ. Here, we use the slope to evaluate the quality of a reduced basis. To compute a slope, we extrapolate the Gram–Schmidt log-norms with a line and take its slope. Concretely, we call the tracer in fpylll. As we can see in Figure 5.28, there is a significant difference between our simulator for pressed-BKZ and the Chen–Nguyen simulator (for standard BKZ).

As can be seen in Figure 5.29, when the dimension is relatively close to the block-size, our simulator for pressed-BKZ outputs the Gram–Schmidt log-norms with slope more significantly slower than the one output by the Chen–Nguyen simulator. In particular, for small block-size $\beta = 50$, our simulator for pressed-BKZ can produce Gram–Schmidt log-norms with slope almost equal to the one produced by the Chen–Nguyen simulator for standard BKZ with block-size 85. Thus we earn almost 35 dimensions while only relying on an SVP solver in dimension 50. The difference becomes very small when the block-size considered is larger than 200.

5.4.3 Pressed-BKZ with variable block-size

Pressed-BKZ helps improving the quality of the basis such that its Gram–Schmidt log-norms approximate a line. However, the concavity phenomenon still exists within each iteration in pressed-BKZ. In particular, in the BKZ_β reduction on each projected sub-lattice $\Lambda_{[s,n]}$ for $s \in [1, \cdots, n-\beta+1]$ of n-dimensional lattice, we can still observe the head concavity phenomenon after a few tours of the BKZ algorithm. As a result, the costs of solving the SVP instances corresponding to the leading blocks become less than those corresponding to the blocks in the middle. We refer to Subsection 5.2.2 for the correspondence between quality of basis and enumeration cost for SVP. Here, we use the method from [AWHT16], in which (adaptively) larger block-sizes are used for the leading blocks, so that the enumeration cost matches the costs in the middle blocks as mentioned above. Concretely, for the case of BKZ_β on the projected $(n-s+1)$-dimensional sub-lattice, we always take the specific block $\Lambda_{[k,e]}$ in the middle with $k = \lfloor n/2 \rfloor - \lfloor \beta/2 \rfloor + \lfloor s/2 \rfloor + 1$ and $e = \lfloor n/2 \rfloor + \lfloor \beta/2 \rfloor + \lfloor s/2 \rfloor$ as the standard of SVP cost for comparison. Then, when the estimated SVP cost in any leading block is smaller than the cost in this middle block, we incrementally increase the block-size of current leading block until the SVP cost on it matches the cost in the middle block. Correspondingly, we only consider the variable strategy for those leading blocks, with starting index not exceeding k (the starting index of the selected middle block).

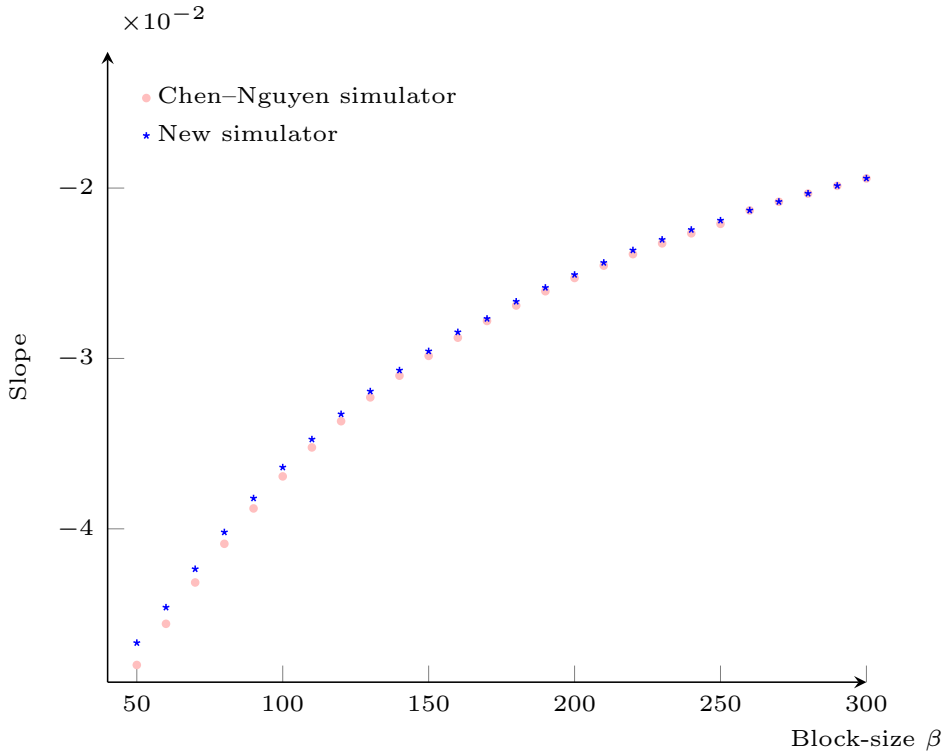

Fig. 5.28: Comparison between our simulator for pressed-BKZ and the Chen–Nguyen simulator for standard BKZ for selected $\beta \in \{50, 60, \cdots, 300\}$. Here the dimension is 1000 for $\beta \in [50, 250]$ and 2000 for $\beta \in [260, 300]$.

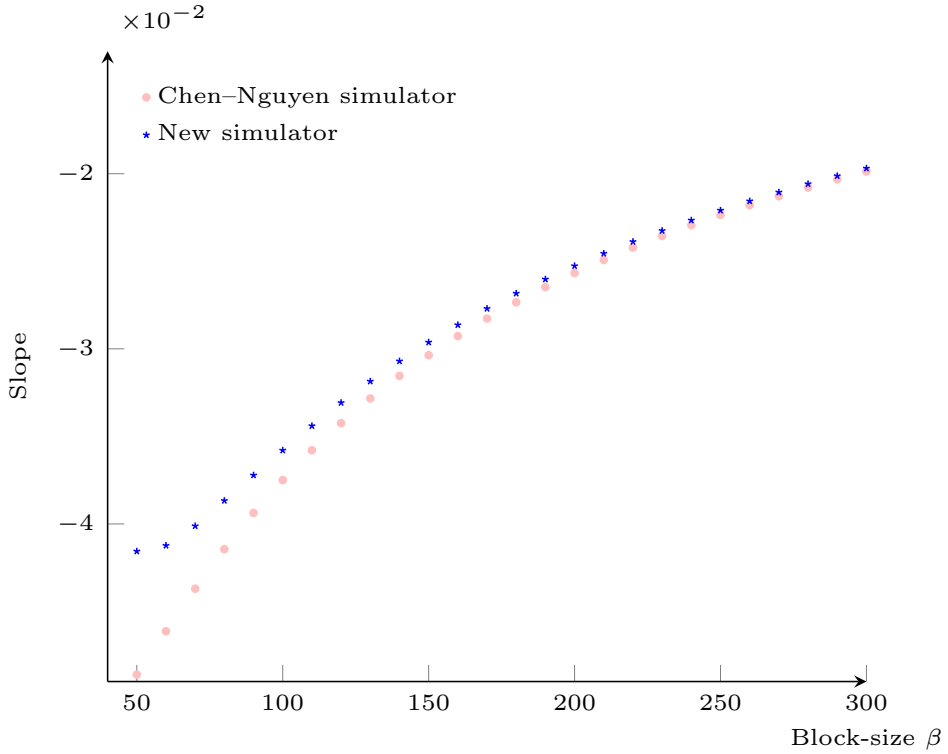

Fig. 5.29: Comparison between our simulator for pressed-BKZ and the Chen–Nguyen simulator for standard BKZ for selected $\beta \in \{50, 60, \cdots, 300\}$. Here the dimension is $3 \cdot \beta$.

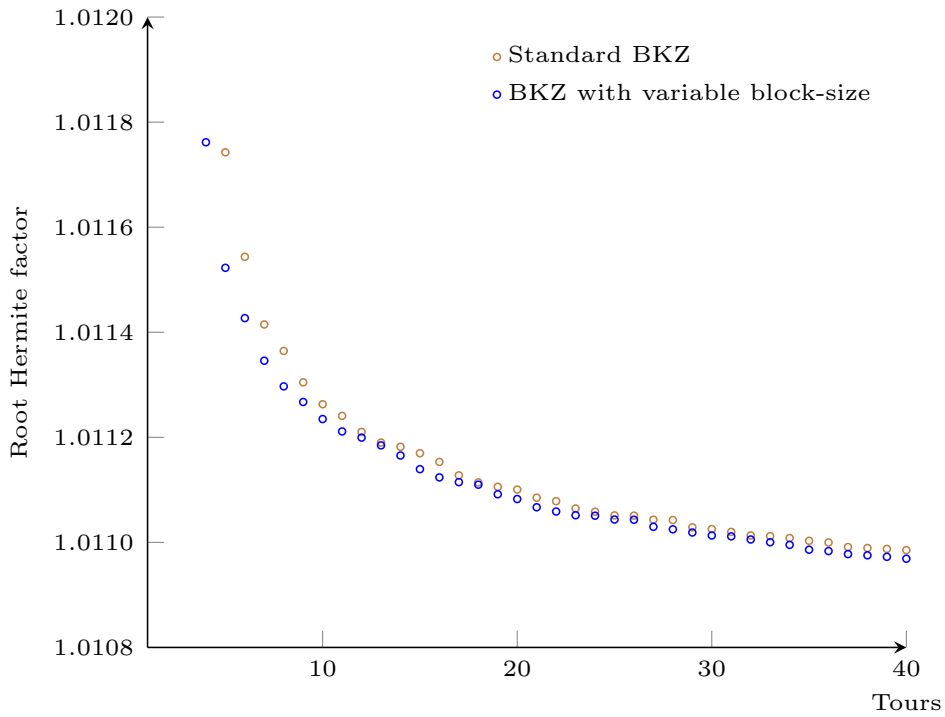

Fig. 5.30: Comparison of root Hermite factors of simulated BKZ with and without variable block-size. The simulation is performed with our new simulator up to 40 tours.

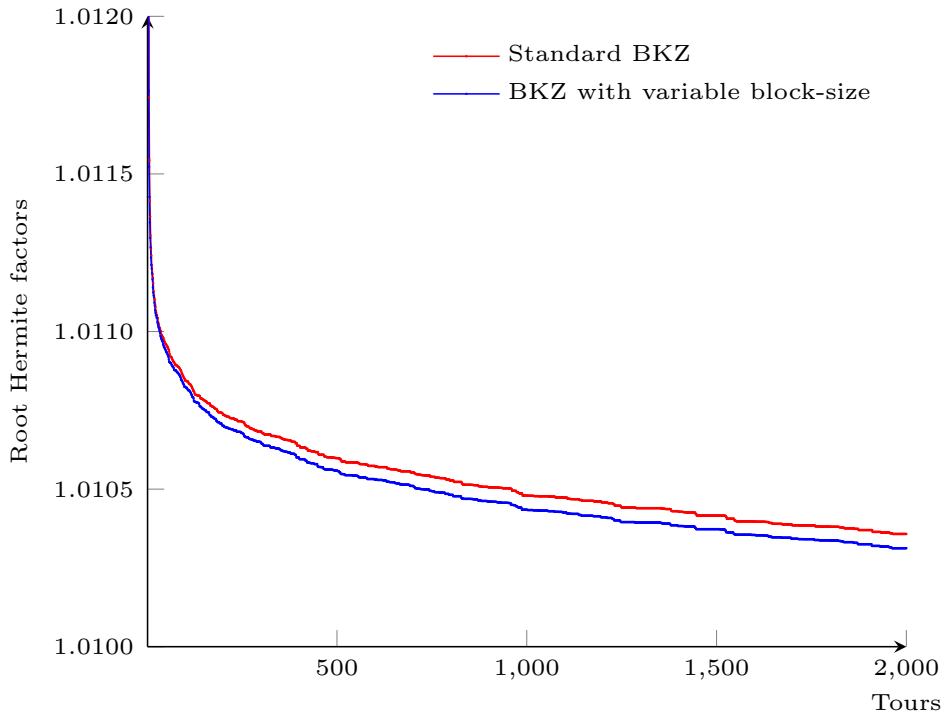

Fig. 5.31: Comparison of root Hermite factors of simulated BKZ with and without variable block-size. The simulation is performed with our new simulator up to 2,000 tours.

We observe that with the same number of tours, pressed-BKZ with variable block-size can produce a basis with better quality. In each iteration, with variable block-size strategy, if the standard BKZ reduction can decrease the first Gram–Schmidt norm of the projected lattice $\Lambda_{[k,n]}$ a bit more for k from 1 to $n-\beta+1$, compared to standard BKZ without such strategy. Then overall, the improvement in each iteration will eventually contribute to the final pressed-BKZ reduced basis (with variable block-size strategy). Thus it suffices to try the standard BKZ with variable block-size and observe whether this gives a better root Hermite factor compared to the standard BKZ. As our simulator gives a precise estimation on the performance of BKZ, to verify this, we run our simulator of standard BKZ with and without this variable block-size strategy, with a fixed block-size $\beta = 60$ (in the case of variable block-size, this is the block-size of the middle and trailing blocks except the 60 last blocks). We plot the average root Hermite factor (over 100 samples) after each tour. As shown in Figure 5.31, BKZ with variable block-size improves the root Hermite factor slightly faster. Our simulator can be readily adapted to this BKZ variant with variable block-size.

However, we notice that after sufficiently many tours, this difference vanishes and the simulated log-norms become closer (as can be seen in Figure 5.31, the difference is ≈ 0.0001). We may conclude that the variable block-size strategy helps improving the root Hermite factor faster. However, it does not give a significant improvement on the root Hermite factor for higher number of tours.

Next, we run experiments to check whether the variable block-size strategy indeed helps decreasing the root Hermite factor faster. We run standard BKZ_{60} with and without variable block-size on SVP-120 challenge. We plot the average root Hermite factor (over 100 samples) after each tour. As can be seen in Figure 5.33, the convergence of root Hermite factor of basis output by the standard BKZ_{60} with variable block-size is slightly faster than the one output by BKZ_{60} without such strategy.

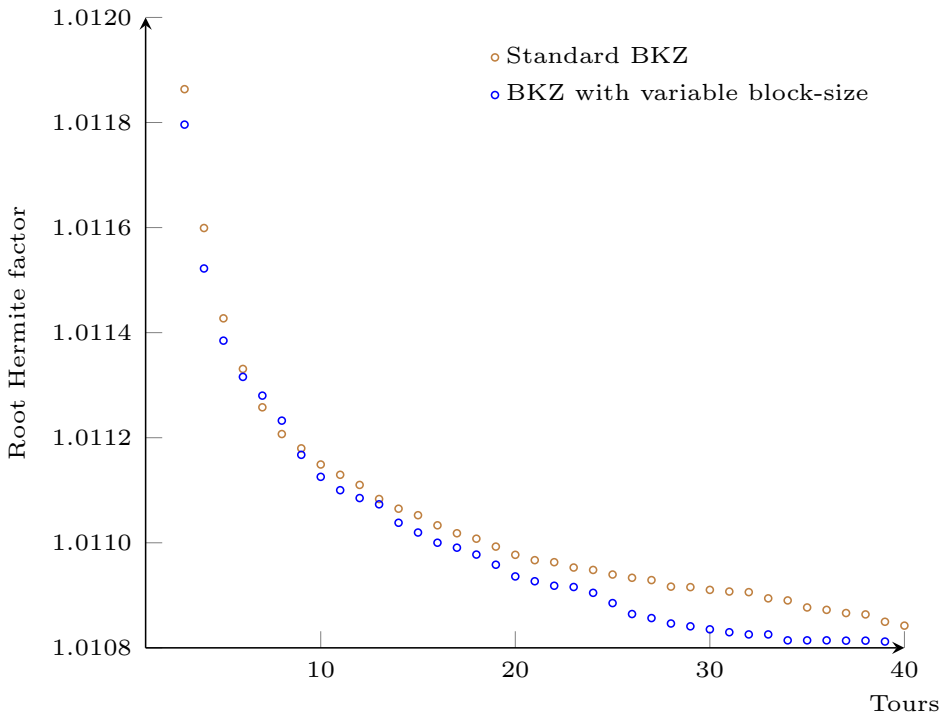

Fig. 5.32: Comparison of root Hermite factors of standard BKZ_{60} with and without variable block-size within 40 tours.

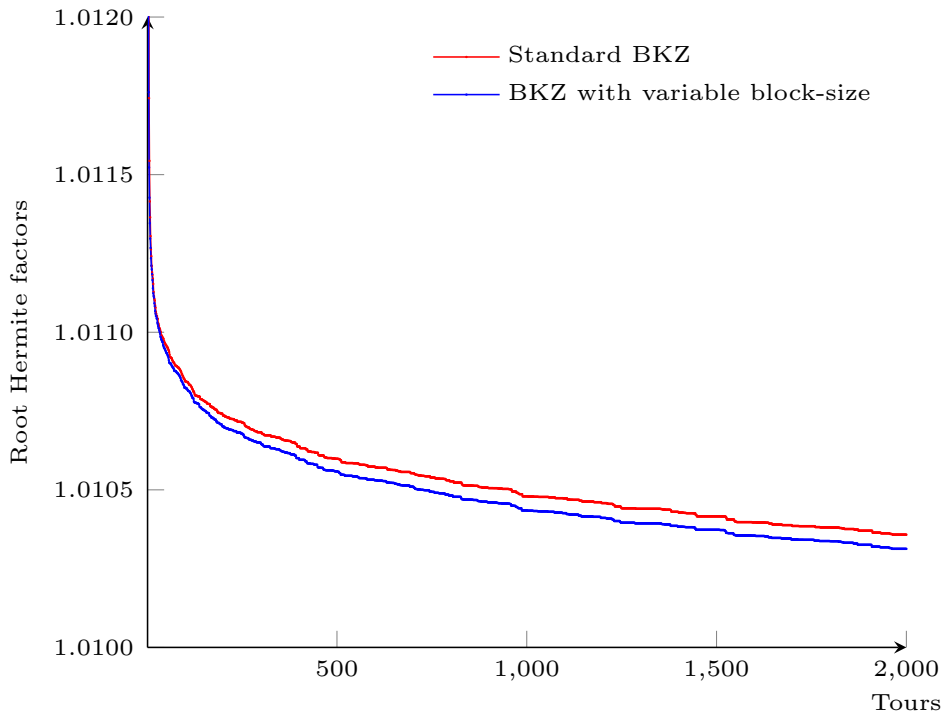

Fig. 5.33: Comparison of root Hermite factors of standard BKZ_{60} with and without variable block-size within 2,000 tours.

5.4.4 Solving SVP-120 with pressed-BKZ

In this subsection, we use pressed-BKZ for the pre-processing to solve an SVP-120 challenge, to demonstrate its practicability. We are interested in the quality of pressed-BKZ reduced bases which can be reflected by the total enumeration cost, i.e., the sum the pre-processing and enumeration costs, divided by the success probability. In the experiment, we consider an SVP challenge of dimension 120 (generated using the Darmstadt lattice challenge generator) and pre-process it using pressed-BKZ_{60} with the adaptive block-size strategy described in the previous subsection. The pre-processing took a total 5×10^5 seconds on an Intel Xeon processor of 2.67GHz (the enumeration speed is 2×10^7 nodes per second and hence the run-time corresponds to an enumeration tree of 1×10^{13} nodes).

Comparison with standard BKZ.

We first investigate the quality of the pressed-BKZ_{60}-reduced basis in terms of BKZ-reducedness. The aim is to find the block-sizes β (and the number of tours) for which the output of standard BKZ_β would be of similar quality. This suggests that the bases produced by pressed-BKZ_{60} and standard BKZ_β have similar full enumeration cost.

We have to determine some criterion for the quality of bases. Essentially, one wants to compare the *pruned* enumeration cost of the pressed-BKZ_{60}-reduced basis with the pruned enumeration cost of the standard BKZ_β-reduced basis. As a first approximation, we compute the *full* enumeration cost of BKZ_β-reduced basis (right after each BKZ tour) and stop as soon as it is close to the *full* enumeration cost for the pressed-BKZ-reduced basis. This also gives us roughly the number of tours needed for BKZ with the corresponding block-size to achieve a similar quality as our basis reduced by pressed-BKZ_{60}. A better approach, which we did not implement, would be to invoke the pruning optimizer right after each local SVP to estimate the enumeration cost and stop as soon as it is close to the pruned enumeration cost for the pressed-BKZ reduced basis.

Instead of doing the actual BKZ_β experiment for all candidate blocksizes, we first use simulation to find the most competitive blocksizes. As investigated in Section 5.3, the probabilistic simulator seems quite precise after the first few tours: if the number of tours involved in the simulation is tiny, we conduct true BKZ experiments for confirmation. After we have determined the most appropriate blocksizes β, we conduct true BKZ experiments for these blocksizes to double-check their quality and running-time (as opposed to simulation).

To start with, we have to determine some suitable searching range for the block-size β. As we already saw, in the case where the dimension is not much larger than the block-size, the quality of a basis reduced by pressed-BKZ_{60} can be quite superior to that obtained by using BKZ_β for $\beta > 60$. Thus we try several larger blocksizes starting for $\beta = 70, 75, 80, 85, 90$. For each blocksize β, the Gram–Schmidt norms of standard BKZ_β are simulated by the probabilistic simulator. We start with the LLL-reduced basis of the SVP-120 input and average over the 100 simulations for each β. We set a maximum of $50,000$ tours in the simulator but break as soon as (if possible) the full enumeration cost is smaller than the full enumeration cost of our reduced pressed-BKZ_{60}.

For blocksizes 70 and 75, the full enumeration cost cannot beat the full enumeration cost of pressed-BKZ_{60} reduced basis within the limit of $50,000$ tours and therefore terminates. For other blocksizes, after the simulator terminates, we compute the (average) minimum number of tours needed. They are listed in the legend of Figure 5.34. In Figure 5.34, we also plot the Gram–Schmidt log-norms of the pressed-BKZ_{60}-reduced basis along with the average simulated Gram–Schmidt log-norms of the output bases of standard BKZ_β (for the relevant numbers of tours). One can observe that, at the point of termination, the simulated Gram–Schmidt log-norms have comparable shape as the pressed-BKZ_{60} reduced basis. We confirm this by examining their full enumeration cost in Table 5.1. The full enumeration cost is tabulated for each (averaged) simulated basis. As a conclusion, it can be seen that the most competitive blocksizes are 85 and 90: the number of tours involved for blocksizes $70, 75, 80$ are too large.

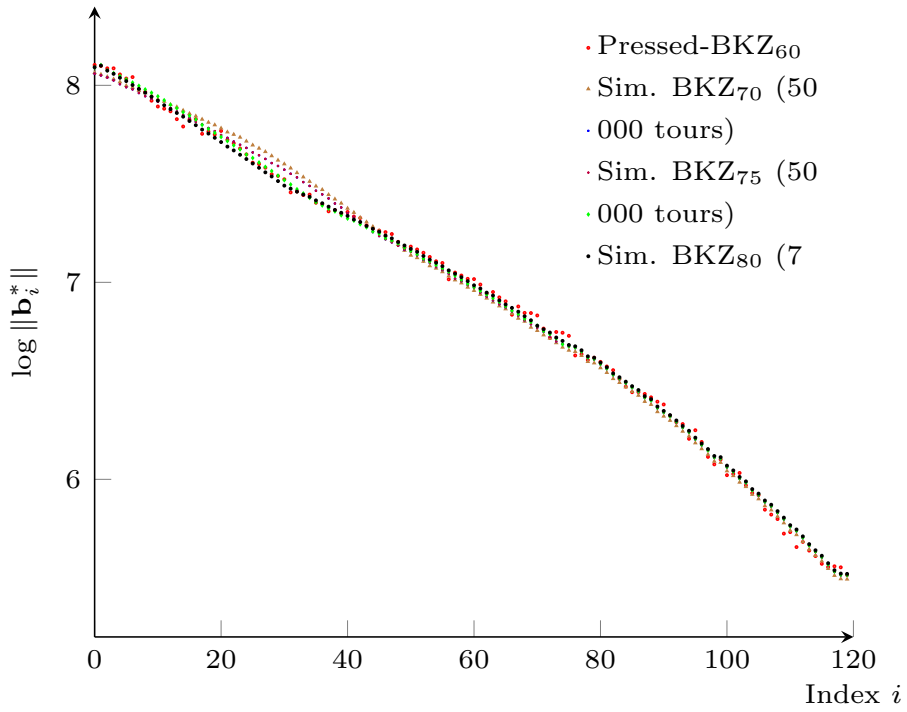

Fig. 5.34: Gram–Schmidt log-norms of simulated pressed-BKZ_{60} and simulated BKZ_β.

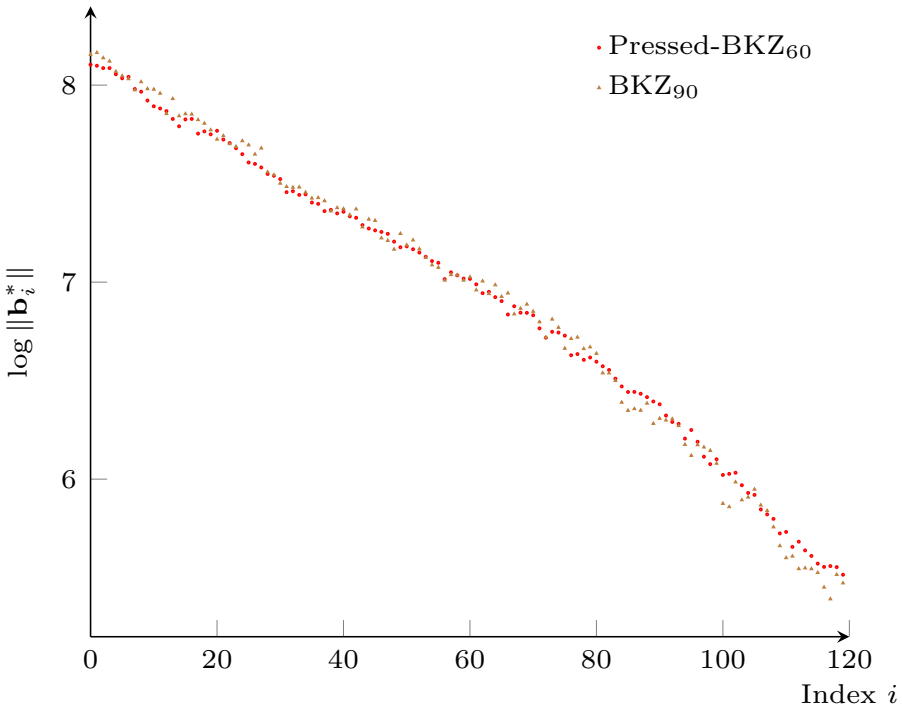

Fig. 5.35: Gram–Schmidt log-norms of experimental pressed-BKZ$_{60}$ and BKZ$_{90}$ (28 tours).

So far, we have only judged the quality of the preprocessed basis using simulation. It should be noted that the probabilistic simulator may not to be accurate when the number of tours involved are tiny (see Subsection 5.3.3). This is the case for blocksizes 85 and 90. Therefore, instead of simulation, we conduct actual BKZ$_{85}$ and BKZ$_{90}$ experiments with the LLL-reduced basis as input: we used a parallel implementation of the BKZ algorithm implemented in `fpylll` and ran BKZ$_{90}$ on the LLL-reduced basis with 280 cores. The local SVP solver attempts to find a vector smaller than 1.05 times the Gaussian heuristic of the local block. It is also "greedy" in the sense that if too many trials have been attempted without a success it moves to the next block. Therefore, the number of tours in experiments could be larger than simulation (but each local SVP takes less time).

Tab. 5.1: Estimated enumeration cost to solve the SVP-120 instance. The row "Full of Sim." records the full enumeration cost (in terms of number of nodes) based on the simulated preprocessed basis. The row "Full of Exp." records the full enumeration cost on BKZ$_{85}$ and BKZ$_{90}$-reduced bases (from experiments) after 100 tours and 28 tours respectively. The row "Prune of Exp." records the cost for pruned enumeration for Pressed-BKZ$_{60}$ and BKZ$_{90}$ reduced basis (from experiments): it includes the costs of all trial enumerations and the cost of preprocessing before each trial enumeration (but excludes the initial preprocessing cost).

	Pressed-BKZ$_{60}$	BKZ$_{70}$	BKZ$_{75}$	BKZ$_{80}$	BKZ$_{85}$	BKZ$_{90}$
Full of Sim.	n/a	6.83×10^{27}	3.21×10^{27}	1.17×10^{27}	1.15×10^{27}	0.83×10^{27}
Full of Exp.	1.21×10^{27}	n/a	n/a	n/a	2.64×10^{27}	1.35×10^{27}
Prune of Exp.	5.9×10^{13}	n/a	n/a	n/a	n/a	6.3×10^{13}

In the experiments, we aborted the BKZ$_{90}$ execution right after the full enumeration cost of the

current basis is similar to (if possible) that of the pressed-BKZ_{60}-reduced basis. Then we used the previous BKZ tour (where the full enumeration cost was slightly larger than that of the pressed-BKZ_{60}-reduced basis). BKZ_{90} took 28 tours to reach a similar full enumeration cost and the overall running-time was 5×10^6 seconds (the number of cores is taken into account). In Figure 5.35, we plot the Gram–Schmidt log-norms of this BKZ_{90}-reduced basis and compare it with pressed-BKZ_{60}. This confirms that their qualities are analogous. For BKZ_{85}, we aborted the computation after 100 tours as the the overall running-time was already 8×10^6 seconds. Note that both are already much larger than the cost we spent on the pressed-BKZ_{60} preprocessing, of 5×10^5 seconds. The full enumeration cost of BKZ_{90} and BKZ_{85}-reduced bases is computed in Table 5.1. Note that the experiments for BKZ_{90} (and BKZ_{85}) took much more tours to achieve the same quality (if possible) compared to simulation. This might be due to the facts that our implementation is greedy as mentioned above and does not always solve the local SVP problem.

So far, we have only used the full enumeration cost to measure the quality. We confirm this using a pruner to estimate the enumeration cost (for Pressed-BKZ_{60} and BKZ_{90}-reduced bases). A pruner optimizes the pruning coefficients to minimize the overall running-time of preprocessing plus enumeration divided by the success probability. The general strategy in extreme pruning [CN11] is to preprocess the basis using BKZ and then run the enumeration with a certain success probability p. If the enumeration fails, it rerandomizes the basis and then conducts the preprocessing and enumeration again. The expected number of repetitions to succeed in the enumeration is $\approx 1/p$. It remains to determine the preprocessing time before each enumeration. It should be noted if the first enumeration fails, one usually runs a mild re-randomization before the next preprocessing, thus the next preprocessing time will be smaller than the first preprocessing time, since it still benefits from the BKZ reduction in last preprocessing.

We determine the preprocessing time with the following experiment. For the BKZ_{90}-reduced basis, after re-randomization, the full enumeration cost increases from 1.35×10^{27} to 1.56×10^{27}. We re-preprocess the randomized basis using BKZ_{80} until the full enumeration cost decreases to around 1.35×10^{27}. Here we just used BKZ_{80} for simplicity (there could be other strategies). The re-preprocessing took 1.7×10^5 seconds (i.e., 3.4×10^{12} nodes). We use this preprocessing cost (the preprocessing before each trial enumeration except the initial preprocessing) as input to the pruner (for both Pressed-BKZ_{60} and BKZ_{90}-reduced bases). The total pruned enumeration cost estimate in `fpylll`, tabulated in Table 5.1, confirms that Pressed-BKZ_{60} and BKZ_{90}-reduced bases indeed have similar quality as they all admit similar total pruned enumeration costs. In general, the pruner seems to be quite precise in practice (hence so are the estimates in Table 5.1). Thus it suffices to compare the initial preprocessing cost between Pressed-BKZ_{60} and BKZ_{90}: pressed-BKZ_{60} (5×10^5 seconds) took less time compared to BKZ_{90} (28 tours in 5×10^6 seconds).

In this subsection, we have only considered a straightforward strategy, BKZ plus enumeration, for solving the SVP-120 instance. In the following we will further compare with progressive-BKZ[AWHT16].

Comparison with progressive-BKZ.

The main idea of progressive-BKZ is to proprocess the basis using BKZ with progressively increased block-sizes and use local enumeration with high success probability to avoid the overheads brought in by the pre-processing. Furthermore, a progressive block-size strategy (optimized based on their adaptation of the Chen-Nguyen simulator for their progressive-BKZ algorithm) was used for pre-processing before a final enumeration. In particular, [AWHT16, Table 4] gives the cost of solving SVP challenges using their block-size strategy: this table is to be understood as the cost of an idealized algorithm and is hence optimistic compared to current algorithms. We re-investigate the estimates in that table by combining the progressive-BKZ method with our pressed-BKZ algorithm.

Given the pressed-BKZ$_{60}$ reduced basis, we use the progressive-BKZ method in the `bkz2_sweet_spot`[3] branch in `fplll`. Note that it implements a variant of progressive-BKZ: the progressive strategy differs from that of [AWHT16] but it suffices for our comparison. It should be noted that our pressed-BKZ$_{60}$ reduced basis is already quite reduced so we start progressive-BKZ with block-size ≈ 75 to avoid a superfluous re-computation. We used 80 cores for the computation on the pressed-BKZ$_{60}$ reduced basis. We spent 5.78 core days (5×10^5 seconds) on the initial pressed-BKZ$_{60}$ and 1.21 core days for the progressive-BKZ to complete the SVP instance. In total, we completed the computation in a total of 6.99 core days (with enumeration speed of $\approx 2 \times 10^7$ nodes per second), faster than the 14.94 lower bound (with enumeration speed of 6×10^7 nodes per second) in [AWHT16, Table 4].

For further comparison, we also ran the same experiment using an LLL-reduced basis instead of pressed-BKZ$_{60}$ reduced basis in the beginning. The overall running-time was 8.75 core days. This implies that `bkz2_sweet_spot` is faster than the estimates in [AWHT16]. Compared to this LLL-based experiment, the pressed-BKZ$_{60}$-reduced basis helps to reduce the overall running-time by about 20%.

It should be noted that we only provide one such strategy that lowers the estimates in [AWHT16, Table 4] and demonstrate the usefulness of the pressed-BKZ algorithm. It is quite possible that this is far from an optimal strategy, which could combine variants of progressive preprocessing, extreme pruning and adaptive choices based on simulation. For instance, it may be better to also use pressed-BKZ inside progressive-BKZ (recursively for any preprocessing) to better maintain the shape of the pressed-BKZ preprocessed basis. Also, we only conducted two SVP-120 experiments, which is not statistically significant. We leave the question of how to optimize the strategy based on the existing approaches open for future work.

[3] `https://github.com/fplll/fpylll/tree/bkz2_sweet_spot`

Chapter 6

Conclusion and open problems

In this chapter, we will give a brief conclusion of this thesis. Further, some open problems that are related to the works in this thesis will also be presented.

6.1 Conclusion

In this thesis, we have studied some aspects of the security foundations of lattice-based cryptography. In particular, we studied two well-known worst-case lattice problems, BDD and uSVP and put forward an improved reduction from one to another. We also showed an equivalence between one of the most important hardness assumptions in lattice-based cryptography, LWE, and a presumed hard quantum problem, DCP. Last, we studied the practical behavior of the BKZ algorithm.

First, we gave a reduction from $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$ to uSVP_γ. This improved the state of the art by a factor $\sqrt{2}$. As a result, to solve the same BDD problems, we can use a (possibly) easier uSVP oracle. The main ingredient used in this improved reduction is lattice sparsification. With this technique, we can efficiently sparsify polynomially many lattice points around the target vector before we resort to Kannan's embedding. In particular for $\gamma = 1$, this allows us to improve the prior reduction from BDD_1 to uSVP_1 to a reduction from $\mathrm{BDD}_{1/(\sqrt{2})}$ to uSVP_1. We further found that such an improvement can be adapted for any $\gamma > 1$. Together with the backward reduction from uSVP_γ to $\mathrm{BDD}_{1/\gamma}$ [LM09], the gap between the reductions between these two problems becomes $\sqrt{2}$.

Second, we showed a computational equivalence between LWE and EDCP up to small parameters losses. The latter one, as an extrapolated version of DCP, is presumed hard under quantum computations. The extent of extrapolation varies with the LWE noise rate. By considering different extents of extrapolation, our result generalizes Regev's proof [Reg02] that if DCP is in quantum polynomial time, then so is LWE. Our result can also be seen as a new evidence of the quantum hardness of LWE, as a probabilistic polynomial time solver for LWE can be transferred into a quanum polynomial time solver for EDCP, which has been studied under quantum computation [Kup05, Reg04c, Kup13]. Last, we also give a computational equivalence between dLWE and dEDCP. Together with the search to decision reduction for LWE [MM11], there is also a search to decision reduction for EDCP. As a result, to solve LWE under quantum computations, one might not need to solve DCP, but its relaxed version, EDCP, which could be easier.

Finally, we proposed a better simulator of the BKZ algorithm, which can precisely predict the behavior of BKZ for practical block-sizes. As the provable worst-case bounds poorly reflect the practical behavior of BKZ, cryptanalysts rely instead on the heuristic Chen–Nguyen simulator. It fits better with practical experiments, but not entirely. In particular, the first few Gram–Schmidt norms in the experiments are smaller than the ones predicted by the Chen–Nguyen simulator, which is of strong significance in cryptanalysis. Our simulator improves the Chen–Nguyen simulator in the aspect that we can also predict the leading Gram–Schmidt norms accurately. As we have a better understanding of the head phenomenon with our simulation for practical block-sizes, we can hope to accurately predict BKZ

for larger block-sizes. This strengthens the security estimates of lattice-based cryptography [ACD+]. We also gave a BKZ variant, named pressed-BKZ, which can exploit the head phenomenon and reduce the basis further. Our (non-optimized) experiments show that our pressed-BKZ is better than prior works for solving SVP challenge in dimension 120 (generated by the Darmstadt lattice challenge generator[1]). We could expect that it also helps improving the SVP solvers for lattices in larger dimensions.

6.2 Open problems

In this section, we present some open problems for future works.

Open problem 1. Can we further improve the relation between BDD and uSVP?

The decoding radius $\lambda_1(\Lambda)/\sqrt{2}$ seems to be a limitation of our reduction based on the sparsification technique. Thus we would consider an improvement from the following two aspects.

First, we conjecture that computational equivalence holds between $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$ and uSVP_γ. We already know that $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$ reduces to uSVP_γ. For the opposite direction, there is a reduction from uSVP_γ to $\mathrm{BDD}_{1/\gamma}$ given by Lyubashevsky and Micciancio [LM09]. To obtain the conjectured equivalence, we would need a reduction from uSVP_γ to $\mathrm{BDD}_{1/(\sqrt{2}\gamma)}$.

Second, it is interesting to look after a reduction from BDD with larger decoding radius $(1/\sqrt{2}+\epsilon)\cdot\lambda_1(\Lambda)$ for some constant $\epsilon > 0$. For instance, if we allow to sparsify more than polynomially many lattice vectors, can we reduce it to $\mathrm{uSVP}_{>1}$? Such a reduction would need a superpolynomial number of repetitions to succeed with non-negligible probability. As the $\mathrm{BDD}_{1/\sqrt{2}+\epsilon}$ is known to be NP-hard [LM09], for which the current known best algorithm takes full exponential time to solve, we would get insight from it, if only the number of points that need to be removed grows asymptotically slower than exponentially. However, we are not aware of a tight upper bound ($2^{\mathcal{O}(n)}$ is an upper bound, but we do not know if it can be reached) on the maximum number of lattice points within distance $(1/\sqrt{2}+\epsilon)\cdot\lambda_1(\Lambda)$ of a given target vector.

Open problem 2. Is there an alternative reduction from EDCP to LWE?

In [CvD07], Childs and van Dam obtain a state of the form

$$\sum_{a\in\mathbb{Z}_N}\sum_{\substack{\mathbf{j}\in\{0,\ldots,M-1\}^\ell\\ \langle\mathbf{j},\mathbf{y}\rangle = a \bmod N}} \omega_N^{a\cdot s}\,|\mathbf{j}\rangle\,.$$

for some uniform $\mathbf{y}\in\mathbb{Z}_N^\ell$. Note the uniform distribution of weights for $\mathbf{j}$. To recover s, the authors use the Pretty Good Measurement technique from [HW94] as was done in [BCvD05, BCvD06] for similar problems. Implementing this general technique to this particular setup requires the construction of a POVM with operators corresponding to superpositions of *all* the $\mathbf{j}$'s in $\{0,\ldots,M-1\}^\ell$ such that $\langle\mathbf{j},\mathbf{y}\rangle = a \bmod N$. As we already discussed in Chapter 4, a unitary operator that realizes such a POVM in [CvD07], uses a lattice-reduction technique as its main subroutine and, hence, works efficiently only for large values of M.

The question we did not address is the interpretation of the POVM technique (and, possibly, a different reduction to LWE) for Gaussian-weighted superpositions. It might be simpler to obtain Gaussian $\mathbf{j}$'s rather than uniform $\mathbf{j}$'s from a cube, and hence it is possible that such a technique may lead to an improved reduction to LWE.

Open problem 3. Can we trade samples number ℓ for extrapolation bound M for $\mathrm{EDCP}_{N,\ell,M}$?

We showed that LWE and U-EDCP are computationally equivalent up to small parameter losses, when the number of U-EDCP states ℓ is polynomial. In these reductions, the U-EDCP bound M is

[1] `https://www.latticechallenge.org/svp-challenge/`

within a polynomial factor of the LWE noise rate $1/\alpha$. When more states are available, U-EDCP is likely to become easier. For instance, with $M = 2$, the best known algorithms when ℓ is polynomially bounded are exponential. Oppositely, Kuperberg's algorithm [Kup05] runs in time $2^{\widetilde{O}(\sqrt{\log N})}$ when $\ell = 2^{\widetilde{O}(\sqrt{\log N})}$. This suggests that there may be a U-EDCP self-reduction allowing to trade ℓ for M: Is it possible to reduce $\mathrm{EDCP}_{N,\ell,M}$ to $\mathrm{EDCP}_{N,\ell',M'}$ with $\ell' \leq \ell$, while allowing for $M' \geq M$? In particular, can we reduce DCP to EDCP, which will eventually give a reduction from DCP to LWE together with our reduction from EDCP to LWE?

Open problem 4. Can we exploit pressed-BKZ to improve concrete solvers for SVP challenges in larger dimensions?

In this thesis, by using pressed-BKZ_{60} for pre-processing, we can solve the SVP challenges in dimension 120 faster. However, the dimension of lattices we consider, 120, is significantly less than the current record, which is 150-dimensional. Can we use pressed-BKZ to improve current SVP solvers for current records or even larger dimensions? Note that we only consider the SVP solvers based on enumeration. In such cases, pre-processing with pressed-BKZ_{60} seems insufficient. We would need to run pressed-BKZ_{β} with larger block-size β for pre-processing instead. However, it would also consume much more time. One idea to decrease the pre-processing cost would be, for each SVP_{β} call on some block, to pre-process this block with pressed-$\mathrm{BKZ}_{\beta'}$ with smaller β', e.g., 30. For the ease of analysis, it would be interesting to have precise simulations on the practical behavior of (pressed-)BKZ with small block-sizes (< 45).

List of publications

[SBW16] Damien Stehlé, Shi Bai and Weiqiang Wen. Improved reduction from the bounded distance decoding problem to the unique shortest vector problem in lattices. In *Proc. of ICALP*, pages 76:1–76:12, 2016. Citations: § 5.

[BKSW18] Zvika Brakerski, Elena Kirshanova, Damien Stehlé and Weiqiang Wen. Learning with errors and extrapolated dihedral cosets. In *Proc. of PKC, part II*, pages 702–727, 2018. Citations: § 5.

[SBW18] Damien Stehlé, Shi Bai and Weiqiang Wen. Measuring, simulating and exploiting the head concavity phenomenon in BKZ. To appear in the Proc. of Asiacrypt 2018. Citations: § 6.

Bibliography

[ACD+] M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, and T. Wunderer. Estimate all the {LWE, NTRU} schemes! To appear in the Proc. of SCN 2018. Citations: § 80, 99, and 113.

[ACF+15] M. R. Albrecht, C. Cid, J.-C. Faugère, R. Fitzpatrick, and L. Perret. On the complexity of the BKW algorithm on LWE. *Designs, Codes and Cryptography*, 74(2):325–354, 2015. Citations: § 21 and 56.

[AD97] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proc. of STOC*, pages 284–293, 1997. Citations: § 2.

[AD16] D. Aggarwal and C. Dubey. Improved hardness results for unique shortest vector problem. *Inf. Process. Lett.*, 116(10):631–637, 2016. Citations: § 2.

[ADPS16] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange—a new hope. In *Proc. of USENIX Security Symposium*, pages 327–343, 2016. Citations: § 18.

[ADRSD15] D. Aggarwal, D. Dadush, O. Regev, and N. Stephens-Davidowitz. Solving the shortest vector problem in 2^n time using discrete Gaussian sampling: Extended abstract. In *Proc. of STOC*, pages 733–742, 2015. Citations: § 16 and 17.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proc. of STOC*, pages 99–108, 1996. Citations: § 2.

[Ajt98] M. Ajtai. The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract). In *Proc. of STOC*, pages 284–293, 1998. Citations: § 2, 16, and 45.

[AKS01] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. of STOC*, pages 601–610, 2001. Citations: § 16 and 17.

[APS15] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Mathematical Cryptology*, 9(3):169–203, 2015. Citations: § 4, 21, and 56.

[ASD18] D. Aggarwal and N. Stephens-Davidowitz. (Gap/S)ETH hardness of SVP. In *Proc. of STOC*, pages 228–238, 2018. Citations: § 13 and 14.

[AWHT16] Y. Aono, Y. Wang, T. Hayashi, and T. Takagi. Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In *Proc. of EUROCRYPT*, pages 789–819, 2016. Citations: § 4, 32, 40, 41, 78, 79, 81, 103, 110, and 111.

[Bab86] L. Babai. On Lovász lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986. Citations: § 4, 20, and 61.

[Ban93] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(4):625–635, 1993. Citations: § 11, 12, and 13.

[BBD08] D. J. Bernstein, J. Buchmann, and E. Dahmen. *Post Quantum Cryptography*. Springer, 1st edition, 2008. Citations: § 3 and 56.

[BCvD05] D. Bacon, A. M. Childs, and W. van Dam. From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups. In *Proc. of FOCS*, pages 469–478. IEEE Computer Society Press, 2005. Citations: § 113.

[BCvD06] D. Bacon, A. M. Childs, and W. van Dam. Optimal measurements for the dihedral hidden subgroup problem. *Chicago J. Theor. Comput. Sci.*, 2006. Citations: § 113.

[BDEJ95] A. Barenco, D. Deutsch, A. Ekert, and R. Jozsa. Conditional quantum dynamics and logic gates. *Phys. Rev. Lett.*, 74:4083–4086, 1995. Citations: § 26.

[Ben73] C. H. Bennett. Logical reversibility of computation. *IBM J. Res. Dev.*, 17(6):525–532, November 1973. Citations: § 27.

[BG15] S. Bai and S. Galbraith. Private communication, 2015. Citations: § 42, 43, and 48.

[BLP+13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *Proc. of STOC*, pages 575–584, 2013. Citations: § 3, 21, 22, 61, 62, and 66.

[Bud75] R. de Buda. The upper error bound of a new near-optimal code. *IEEE Trans. on Information Theory*, 21(4):441–445, 1975. Citations: § 43.

[BV11] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *Proc. of FOCS*, pages 97–106, 2011. Citations: § 3 and 56.

[Che09] Y. Chen. *Réduction de réseau et sécurité concrète du chiffrement complètement homomorphe*. PhD thesis, Université Paris Diderot, 2009. Citations: § 11, 32, 39, and 78.

[CN11] Y. Chen and P. Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *Proc. of ASIACRYPT*, pages 1–20, 2011. Citations: § 4, 32, 39, 40, 78, 80, 81, 99, and 110.

[CvD07] A. M. Childs and W. van Dam. Quantum algorithm for a generalized hidden shift problem. In *Proc. of SODA*, pages 1225–1232, 2007. Citations: § 32, 57, 58, 64, and 113.

[DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22:644–654, November 1976. Citations: § 1.

[DK13] D. Dadush and G. Kun. Lattice sparsification and the approximate closest vector problem. In *Proc. of SODA*, pages 1088–1102. SIAM, 2013. Citations: § 44.

[DRS14] D. Dadush, O. Regev, and N. Stephens-Davidowitz. On the closest vector problem with a distance guarantee. In *Proc. of CCC*, pages 98–109, 2014. Citations: § 44.

[dt16] The FPLLL development team. fplll, a lattice reduction library. `https://github.com/fplll/fplll`, 2016. Citations: § 37 and 81.

[dt17] The FPYLLL development team. fpylll, a Python interface to fplll. `https://github.com/fplll/fpylll`, 2017. Citations: § 81.

[Duc18] L. Ducas. Shortest vector from lattice sieving: A few dimensions for free. In *Proc. of EUROCRYPT*, pages 125–145, 2018. Citations: § 18.

[EH99] M. Ettinger and P. Høyer. On quantum algorithms for noncommutative hidden subgroups. In *Proc. of STACS*, pages 478–487, 1999. Citations: § 29 and 56.

[Emd81] P. van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical report, Mathematische Instituut, University of Amsterdam, 1981. Report 81-04. Citations: § 2 and 16.

[FIM+03] K. Friedl, G. Ivanyos, F. Magniez, M. Santha, and P. Sen. Hidden translation and orbit coset in quantum computing. In *Proc. of STOC*, pages 1–9, 2003. Citations: § 28.

[FIM+14] K. Friedl, G. Ivanyos, F. Magniez, M. Santha, and P. Sen. Hidden translation and translating coset in quantum computing. *SIAM J. Comput*, 43(1):1–24, 2014. Citations: § 56.

[FP83] U. Fincke and M. Pohst. A procedure for determining algebraic integers of given norm. In *Proc. of EUROCAL*, pages 194–202, 1983. Citations: § 17, 37, and 43.

[GGH97] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Proc. of CRYPTO*, pages 112–131, 1997. Citations: § 2.

[GGH13] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *Proc. of EUROCRYPT*, pages 1–17, 2013. Citations: § 2.

[GHGKN06] N. Gama, N. Howgrave-Graham, H. Koy, and P. Q. Nguyen. Rankin's constant and blockwise lattice reduction. In *Proc. of CRYPTO*, pages 112–130, 2006. Citations: § 37.

[GKP+13] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Proc. of STOC*, pages 555–564, 2013. Citations: § 3 and 56.

[GMSS99] O. Goldreich, D. Micciancio, S. Safra, and J.-P. Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Inf. Process. Lett.*, 71(2):55–61, 1999. Citations: § 18.

[GN08a] N. Gama and P. Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *Proc. of STOC*, pages 207–216, 2008. Citations: § 37.

[GN08b] N. Gama and P. Q. Nguyen. Predicting lattice reduction. In *Proc. of EUROCRYPT*, pages 31–51, 2008. Citations: § 37 and 39.

[GNR10] N. Gama, P. Q. Nguyen, and O. Regev. Lattice enumeration using extreme pruning. In *Proc. of EUROCRYPT*, pages 257–278, 2010. Citations: § 16, 17, 37, and 81.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proc. of STOC*, pages 197–206, 2008. Citations: § 58.

[GR02] L. Grover and T. Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions, 2002. Draft. Available at `https://arxiv.org/pdf/quant-ph/0208112v1`. Citations: § 27 and 29.

[GSVV01] M. Grigni, L. Schulman, M. Vazirani, and U. Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. In *Proc. of STOC*, pages 68–74, 2001. Citations: § 28.

[GVW13] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In *Proc. of STOC*, pages 545–554, 2013. Citations: § 3 and 56.

[HILL99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. Citations: § 23.

[HPS11] G. Hanrot, X. Pujol, and D. Stehlé. Analyzing blockwise lattice algorithms using dynamical systems. In *Proc. of CRYPTO*, pages 447–464, 2011. Citations: § 4, 32, 37, 39, 78, and 87.

[HR07] I. Haviv and O. Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proc. of STOC*, pages 469–477, 2007. Citations: § 2.

[HRTS00] S. Hallgren, A. Russell, and A. Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *Proc. of STOC*, pages 627–635, 2000. Citations: § 28.

[HS07] G. Hanrot and D. Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm. In *Proc. of CRYPTO*, pages 170–186, 2007. Citations: § 16, 17, and 86.

[HS08] G. Hanrot and D. Stehlé. Worst-case Hermite-Korkine-Zolotarev reduced lattice bases. *CoRR*, abs/0801.3331, 2008. Citations: § 78.

[HW94] P. Hausladen and W. K. Wootters. A 'pretty good' measurement for distinguishing quantum states. *Journal of Modern Optics*, 41(12):2385–2390, 1994. Citations: § 113.

[Kan83] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. of STOC*, pages 99–108, 1983. Citations: § 17, 37, and 43.

[Kan87] R. Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, 1987. Citations: § 5 and 43.

[Kho03] S. Khot. Hardness of approximating the shortest vector problem in high L_p norms. In *Proc. of FOCS*, pages 290–297, 2003. Citations: § 5, 13, 43, and 44.

[Kho05] S. Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, 2005. Citations: § 2, 43, and 44.

[KS01] R. Kumar and D. Sivakumar. On the unique shortest lattice vector problem. *Theoretical Computer Science*, 255(1-2):641–648, 2001. Citations: § 2.

[Kup05] G. Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM J. Comput*, 35(1):170–188, 2005. Citations: § 25, 29, 30, 56, 112, and 114.

[Kup13] G. Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. In *Proc. of TQC*, pages 20–34, 2013. Citations: § 25, 30, 31, and 112.

[Len83] H. W. Lenstra, Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. Citations: § 57.

[LLL82] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann*, 261:515–534, 1982. Citations: § 4, 20, 33, 34, 35, 43, 58, and 61.

[LLM06] Y.K. Liu, V. Lyubashevsky, and D. Micciancio. On bounded distance decoding for general lattices. In *Proc. of RANDOM*, pages 450–461, 2006. Citations: § 2.

[LLS90] J. C. Lagarias, W. H. Lenstra, and C. P. Schnorr. Korkine-Zolotarev bases and successive minimal of a lattice and its reciprocal lattice. *Combinatorica*, 10:333–348, 1990. Citations: § 35.

[LM09] V. Lyubashevsky and D. Micciancio. On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In *Proc. of CRYPTO*, pages 577–594, 2009. Citations: § 2, 3, 5, 6, 18, 19, 42, 43, 44, 52, 59, 112, and 113.

[Lov86] L. Lovász. *An Algorithmic Theory of Numbers, Graphs and Convexity.* SIAM, 1986. CBMS-NSF Regional Conference Series in Applied Mathematics. Citations: § 38.

[LWXZ14] M. Liu, X. Wang, G. Xu, and X. Zheng. A note on BDD problems with λ_2-gap. *Inf. Process. Lett.*, 114(1-2):9–12, January 2014. Citations: § 18, 43, 44, 45, 46, and 48.

[Mar02] J. Martinet. *Perfect Lattices in Euclidean Spaces.* Springer, 2002. Citations: § 10.

[MG02] D. Micciancio and S. Goldwasser. *Complexity of Lattice problem: A Cryptography Perspective.* Kluwer, 2002. Citations: § 5, 8, 14, 15, and 45.

[Mic] D. Micciancio. Lecture notes of *lattices algorithms and applications*, taught at the Computer Science & Engineering Department, University of California, San Diego. Available at `http://cseweb.ucsd.edu/classes/wi10/cse206a`. Citations: § 8.

[Mic01a] D. Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM J. Comput.*, 30(6):2008–2035, 2001. Citations: § 16 and 18.

[Mic01b] D. Micciancio. The shortest vector problem is NP-hard to approximate to within some constant. *SIAM J. Comput*, 30(6):2008–2035, 2001. Citations: § 45.

[Mic12] D. Micciancio. Inapproximability of the shortest vector problem: Toward a deterministic reduction. *Theory of Computing*, 8(22):487–512, 2012. Citations: § 16.

[Mic15] D. Micciancio. Private communication, 2015. Citations: § 42, 43, and 48.

[MM11] D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *Proc. of CRYPTO*, pages 465–484, 2011. Citations: § 112.

[MR07] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput*, 37(1):267–302, 2007. Citations: § 2.

[MR09] D. Micciancio and O. Regev. Lattice-based cryptography. In *Proc. of PQCRYPTO*, pages 147–191, 2009. Citations: § 3.

[MV10a] M. Madritsch and B. Vallée. Modelling the lll algorithm by sandpiles. In *Proc. of LATIN*, pages 267–281, 2010. Citations: § 32 and 38.

[MV10b] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In *Proc. of STOC*, pages 351–358, 2010. Citations: § 16 and 17.

[MV10c] D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proc. of SODA*, 2010. Citations: § 16 and 17.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000. Citations: § 25, 26, and 27.

[Neu17] A. Neumaier. Bounding basis reduction properties. *Des. Codes Cryptography*, 84(1-2):237–259, 2017. Citations: § 4, 37, and 78.

[NV09] P. Q. Nguyen and B. Vallée. *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer, 2009. Citations: § 32 and 121.

[ORR13] M. Ozols, M. Roetteler, and J. Roland. Quantum rejection sampling. *ACM Trans. Comput. Theory*, 5(3):11:1–11:33, August 2013. Citations: § 28 and 64.

[Pei] C. Peikert. Lecture notes of *lattices in cryptography*, taught at the Computer Science and Engineering, University of Michigan. Available at `http://web.eecs.umich.edu/~cpeikert`. Citations: § 8.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proc. of STOC*, pages 333–342, 2009. Citations: § 3 and 21.

[Pei16] C. Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016. Citations: § 56.

[RB98] M. Rötteler and T. Beth. Polynomial-time solution to the hidden subgroup problem for a class of non-abelian groups, 1998. Draft. Available at `https://arxiv.org/abs/quant-ph/9812070`. Citations: § 28.

[Reg] O. Regev. Lecture notes of *lattices in computer science*, taught at the Computer Science, Tel Aviv University. Available at `http://www.cs.tau.il/~odedr`. Citations: § 8.

[Reg02] O. Regev. Quantum computation and lattice problems. In *Proc. of FOCS*, pages 520–529, 2002. Citations: § 5, 25, 28, 29, 55, 56, 58, 59, 61, 68, 70, 71, and 112.

[Reg03] O. Regev. New lattice based cryptographic constructions. In *Proc. of STOC*, pages 407–416, 2003. Citations: § 2.

[Reg04a] O. Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004. Citations: § 62.

[Reg04b] O. Regev. Quantum computation and lattice problems. *SIAM J. Comput*, 33(3):738–760, 2004. Citations: § 5, 56, 58, 60, and 62.

[Reg04c] O. Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space, 2004. Draft. Available at `https://arxiv.org/pdf/quant-ph/0406151`. Citations: § 25, 30, and 112.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proc. of STOC*, pages 84–93, 2005. Citations: § 2, 3, 23, 56, and 59.

[Reg09] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009. Citations: § 3, 18, 21, 23, 24, 27, 43, 56, 58, 59, and 62.

[RSA78] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978. Citations: § 1.

[Sch] C. P. Schnorr. Progress on LLL and lattice reduction. Chapter of [NV09]. Citations: § 37.

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Inc., 1986. Citations: § 8 and 9.

[Sch87] C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theor. Comput. Science*, 53:201–224, 1987. Citations: § 2, 4, 36, and 43.

[Sch03] C. P. Schnorr. Lattice reduction by random sampling and birthday methods. In *STACS*, pages 145–156, 2003. Citations: § 4, 39, and 78.

[SD16a] N. Stephens-Davidowitz. Discrete Gaussian sampling reduces to CVP and SVP. In *Proc. of SODA*, 2016. Citations: § 13 and 44.

[SD16b] N. Stephens-Davidowitz. Search-to-Decision Reductions for Lattice Problems with Approximation Factors (Slightly) Greater Than One. In *Proc. of APPROX/RANDOM*, pages 19:1–19:18, 2016. Citations: § 2 and 13.

[SE91] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In *Proc. of FCT*, pages 68–85, 1991. Citations: § 36, 37, and 38.

[SE94] C. P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematics of Programming*, 66:181–199, 1994. Citations: § 2, 4, 32, 36, 37, 38, 78, and 81.

[Sho99] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332, 1999. Citations: § 1.

[Sho05] V. Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, New York, NY, USA, 2005. Citations: § 66.

[Söd11] A. Södergren. On the Poisson distribution of lengths of lattice vectors in a random lattice. *Mathematische Zeitschrift*, 269(3):945–954, 2011. Citations: § 11 and 78.

[SSTX09] D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. Efficient public key encryption based on ideal lattices. In *Proc. of ASIACRYPT*, pages 617–635, 2009. Citations: § 58.

[Ste15] N. Stephens-Davidowitz. Dimension-preserving reductions between lattice problems. Available at `http://noahsd.com/latticeproblems.pdf`, 2015. Citations: § 2 and 16.

[Vai] V. Vaikuntanathan. Lecture notes of *advanced topics in cryptography: lattices*, taught at the Electrical Engineering & Computer Science, Massachusetts Institute of Technology. Available at `http://people.csail.mit.edu/vinodv`. Citations: § 8.

[Var97] A. Vardy. Algorithmic complexity in coding theory and the minimum distance problem. In *Proc. of STOC*, pages 92–109, 1997. Citations: § 2.

[YD17] Y. Yu and L. Ducas. Second order statistical behavior of LLL and BKZ. In *Proc. of SAC*, pages 3–22, 2017. Citations: § 4, 32, 78, 81, and 88.

List of Figures

List of Tables

List of Algorithms