Wessel P.J. van Woerden

The closest vector problem in cyclotomic lattices

Bachelor Thesis

1st supervisor: Dr. Léo Ducas (CWI)
2nd supervisor: Dr. Marcello M. Bonsangue (LIACS)

Date Bachelor exam: 24 June 2016

Mathematical Institute, Leiden University
LIACS, Leiden University

Abstract

In this thesis we are interested in constructing an efficient algorithm for solving the closest vector problem (CVP) in the cyclotomic lattices and their duals. We will show that every cyclotomic lattice can be constructed by direct sums and tensor products from the lattices A_n^* ($n \geq 1$). For the prime power cases this results in a linear CVP algorithm for the cyclotomic lattice and its dual. For the composite case $n = p \cdot q$ with p and q prime we will construct a sub-exponential CVP algorithm and for its dual a polynomial CVP algorithm. Both of these algorithms can efficiently be extended to the $n = p^k q^l$ case.

Table of Contents

1. Introduction

A lattice is a discrete additive subgroup of $\mathbb{F}^n$ generated over $\mathbb{Z}$ by some $\mathbb{F}$-linearly independent (lattice) basis, where $\mathbb{F}$ is the field $\mathbb{Q}$ or $\mathbb{R}$. A central problem in the theory of lattices is the closest vector problem (CVP). Given a lattice and a target point in the $\mathbb{F}$-linear span of that lattice, find a closest lattice point to the target. It is often seen as one of the hardest computational lattice problems as many lattice problems polynomially reduce to it. For example the shortest vector problem (SVP) [1], and more generally finding all successive minima of a lattice [2].

Furthermore it was already proven in 1981 that for general lattices CVP was NP-hard under deterministic reductions [3]. In comparison a weaker result for SVP came almost two decades later when in 1998 SVP was proven to be NP-hard under randomized reductions [4]. A deterministic reduction that SVP is NP-hard hasn't been discovered yet. Although CVP is an NP-hard problem for general lattices, it is interesting to design lattices for which CVP can be solved efficiently while at the same time optimizing other lattice properties like the packing density. Special lattices are for example $A_n (n \geq 1), D_n (n \geq 2), E_n (n = 6, 7, 8)$, their duals and the Leech lattice [5].

Applications of CVP can be found in error correction for transmission over analogue channels [6] and in cryptography [7, 8]. Recent attempts to create lattice-based cryptographic schemes are promising and are mostly based on removing some added error to a lattice vector using a CVP algorithm [9, 10]. At the moment exact CVP algorithms are only used for trivial lattices like $\mathbb{Z}^n$ that have an orthogonal basis. For nontrivial lattices we resort to approximation algorithms that undermine the efficiency of the scheme. To prevent this it would be helpful to find efficient exact CVP algorithms for some nontrivial lattices.

For efficient cryptographic schemes, we are interested in the ring of integers of certain number fields viewed as lattices. In particular the ring of integers of cyclotomic number fields (together with an inner product) and their duals are interesting. Mostly cyclotomic number fields with parameter 2^m are used as the induced lattice has an orthogonal basis which makes CVP trivial. The problem is that this gives us a sparse parameter set and not much variation.

In this thesis we first notice in section 3 that every prime case cyclotomic lattice is in fact equal to some case of A_n^*, the dual of the root lattice A_n. For these lattices efficient CVP algorithms already exist which we will detail in section 4. We will also see that the prime power cases reduce to the prime case in an efficient way using some general CVP techniques in section 5. After this we try to generalize to other cyclotomic lattices and their duals with parameters of the form $n = p \cdot q$ with p and q prime. For these lattices we will find CVP algorithms that work respectively in sub-exponential and polynomial time in sections 6 and 7.

2. Preliminaries

2.1. Lattices and the closest vector problem.

We will start with defining a lattice and some basic properties. In this thesis $\mathbb{F}$ can be the field $\mathbb{Q}$ or $\mathbb{R}$ as long as its use is consistent locally.

Definition 1 *(Lattice).* A *lattice* Λ with $\mathbb{F}$-linearly independent *(lattice) basis* $b_1, \ldots, b_m \in \mathbb{F}^n$ is the discrete additive subgroup

$$\Lambda := \left\{ \sum_{i=1}^{m} z_i b_i : z_i \in \mathbb{Z} \right\}$$

of $\mathbb{F}^n$. Let $B \in \mathbb{F}^{m \times n}$ be the matrix with rows $b_1, \ldots, b_m$. We say that Λ has *rank* m and *generator matrix* B.

Another equivalent way of defining a lattice, which we will use informally, is as a finitely generated free $\mathbb{Z}$-module M with positive-definite symmetric bilinear form $M \times M \to \mathbb{F}$. We can embed M inside $\mathbb{F}^n$ for some $n \in \mathbb{N}$ such that the given positive-definite symmetric bilinear form corresponds to the canonical inner product (dot product) on $\mathbb{F}^n$. In this way we get a lattice by our formal definition.

The matrix $G \in \mathbb{F}^{m \times m}$ consisting of the canonical inner products of basis vectors for a given basis, i.e. $G = BB^\top$, is called the *Gram matrix* of Λ. Let $\mathrm{span}(\Lambda)$ be the linear subspace of $\mathbb{F}^n$ spanned by the elements of Λ over $\mathbb{F}$. The *Voronoi region* of Λ is

$$V(\Lambda) = \{x \in \mathrm{span}(\Lambda) : \|x\| \leq \|x - v\| \text{ for all } v \in \Lambda\}$$

where $\|.\| : \mathbb{F}^n \to \mathbb{R}$ is the canonical norm induced by the canonical inner product on $\mathbb{F}^n$. So the Voronoi region consists of all points of $\mathrm{span}(\Lambda)$ that are at least as close to $0 \in \Lambda$ as to any other point of Λ. We define the *determinant* of Λ, denoted $\det(\Lambda)$, as the m-dimensional volume of $V(\Lambda)$. This can equivalently be defined as $\det(\Lambda) := \sqrt{\det(BB^\top)} = \sqrt{\det(G)}$ which is independent of the chosen basis. The *shortest vectors* of Λ are the nonzero points of Λ with minimal norm. If $v \in \Lambda$ is a shortest vector then $\rho = \frac{\|v\|}{2}$ is the *packing radius* of Λ. The *covering radius* R is the minimal distance such that any point in $\mathrm{span}(\Lambda)$ is at distance at most R to a lattice point. Another lattice $\Lambda' \subset \mathbb{F}^n$ of rank m such that $\Lambda' \subset \Lambda$ is called a sublattice of Λ. [5]

There exist a lot of problems in the theory of lattices and for general lattices these problems are often NP-hard in the lattice rank m. For example we have the Shortest Vector Problem (SVP) where we want to find the shortest vectors of a lattice given a basis. SVP is proven to be NP-hard to solve exactly under randomized reductions [4] and even proven to be NP-hard to approximate within any constant factor under randomized reductions [11].

The lattice problem we will study is the Closest Vector Problem (CVP).

Definition 2 *(Closest Vector Problem).* Let $\Lambda \subset \mathbb{F}^n$ be a lattice. Given an arbitrary point $t \in \mathrm{span}(\Lambda)$, the goal is to find a closest lattice point of Λ to t, i.e., an $x \in \Lambda$ that minimizes the distance $\|t - x\| := \sqrt{\langle t - x, t - x \rangle}$. Such an x is also called a closest vector to t. Note that this is equivalent to finding an $x \in \Lambda$ such that $t - x \in V(\Lambda)$ as $V(\Lambda)$ consists of all points that have 0 as a closest vector. Furthermore the covering radius gives a tight bound on the distance between t and a closest vector to t.

For general lattices CVP is NP-hard (under deterministic reductions) to solve exactly [3]. It is also known to be NP-hard to approximate for factors as large as $m^{1/O(\log\log m)}$ [12]. Even if exponential space and time preprocessing is allowed (CVPP) it is still NP-hard to approximate within a factor of $(\log(m))^{1/(2-\varepsilon)}$ for any $\varepsilon > 0$ [13].

Although this problem is hard for general lattices there exist classes of lattices for which a more efficient algorithm can be found. A trivial example for instance is CVP in the lattice $\mathbb{Z}^n \subset \mathbb{R}^n$ in which case given a $t \in \text{span}(\mathbb{Z}^n) = \mathbb{R}^n$ we can just round each coefficient of t individually to obtain a closest vector to t in $\mathbb{Z}^n$.

2.2. **Composition of lattices and duality.**

In case we want to construct new lattices from other lattices we can use the direct sum, orthogonal sum or tensor product.

Definition 3 *(Direct sum and orthogonal sum).* We will define two different notions of summation of two lattices. First, let $\Lambda_1 \subset \mathbb{F}^{n_1}$ and $\Lambda_2 \subset \mathbb{F}^{n_2}$ be lattices of rank m_1 and m_2 respectively. Then we define the *direct sum* $\Lambda_1 \oplus \Lambda_2 \subset \mathbb{F}^{n_1+n_2}$ between Λ_1 and Λ_2 as

$$\Lambda_1 \oplus \Lambda_2 = \{x_1 \oplus x_2 \in \mathbb{F}^{n_1+n_2} : x_1 \in \Lambda_1, x_2 \in \Lambda_2\}$$

where $x_1 \oplus x_2$ is just the concatenation of the two vectors. Note that the inner product between elements in Λ_1 or Λ_2 (embedded as $x_1 \mapsto x_1 \oplus 0$ and $x_2 \mapsto 0 \oplus x_2$) stays the same and that each two elements $x_1 \in \Lambda_1$ and $x_2 \in \Lambda_2$ are orthogonal in $\Lambda_1 \oplus \Lambda_2$.

For the second notion, let $\Lambda_1, \Lambda_2 \subset \mathbb{F}^n$ be lattices. Suppose Λ_1 has basis $a_1, \ldots, a_{m_1}$ and Λ_2 has basis $b_1, \ldots, b_{m_2}$. In the case that $\langle a_i, b_j \rangle = 0$ for all $i = 1, \ldots, m_1$ and $j = 1, \ldots, m_2$ we call Λ_1 and Λ_2 orthogonal and we define the *orthogonal sum*

$$\Lambda_1 \perp \Lambda_2 \subset \mathbb{F}^n$$

between Λ_1 and Λ_2 as the lattice with basis $a_1, \ldots, a_{m_1}, b_1, \ldots, b_{m_2}$.

The tensor product is known to make hard problems often even harder. For example it is used in [14] to boost the hardness-factor for approximating SVP.

Definition 4 *(Tensor product lattices).* Let $\Lambda_1 \subset \mathbb{F}^{n_1}$ and $\Lambda_2 \subset \mathbb{F}^{n_2}$ with basis $a_1, \ldots, a_{m_1} \in \mathbb{F}^{n_1}$ and $b_1, \ldots, b_{m_2} \in \mathbb{F}^{n_2}$ be lattices of rank m_1 and m_2 respectively. We define $\Lambda_1 \otimes \Lambda_2 \subset \mathbb{F}^{n_1 n_2}$ as the lattice with basis $\{a_i \otimes b_j : i \in \{1, \ldots, m_1\}, j \in \{1, \ldots, m_2\}\}$. Here $c \otimes d = (c_1, \ldots, c_{n_1}) \otimes (d_1, \ldots, d_{n_1})$ with $c \in \mathbb{F}^{n_1}$ and $d \in \mathbb{F}^{n_2}$ is defined as the natural embedding in $\mathbb{F}^{n_1 n_2}$ as follows:

$$c \otimes d := (c_1 d_1, c_1 d_2, \ldots, c_1 d_{n_2}, c_2 d_1, \ldots, c_{n_1} d_{n_2}) \in \mathbb{F}^{n_1 n_2}.$$

Note that for $a, c \in \mathbb{F}^{n_1}$ and $b, d \in \mathbb{F}^{n_2}$ we have for the canonical inner product that:

$$\langle a \otimes b, c \otimes d \rangle = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} a_i b_j \cdot c_i d_j = \sum_{i=1}^{n_1} a_i c_i \sum_{j=1}^{n_2} b_j d_j = \langle a, c \rangle \cdot \langle b, d \rangle.$$

This has as a result that if A and B are the Gram matrices of Λ_1 and Λ_2 respectively, that then $A \otimes B$ (the Kronecker product) is the Gram matrix of $\Lambda_1 \otimes \Lambda_2$.

Then we have that

$$\begin{aligned}\det(\Lambda_1 \otimes \Lambda_2) &= \sqrt{\det(A \otimes B)} = \sqrt{\det((A \otimes I_{m_2}) \cdot (I_{m_1} \otimes B))} \\ &= \sqrt{\det(A)^{m_2} \cdot \det(B)^{m_1}} = \det(\Lambda_1)^{m_2} \cdot \det(\Lambda_2)^{m_1}.\end{aligned}$$

We will now introduce the notion of duality in lattices.

Definition 5 *(Dual Lattice).* For a lattice $\Lambda \subset \mathbb{F}^n$ we define its dual lattice $\Lambda^* \subset \mathbb{F}^n$ as

$$\Lambda^* := \{y \in \text{span}(\Lambda) : \forall x \in \Lambda, \langle x, y\rangle \in \mathbb{Z}\}.$$

Furthermore for every basis $b_1, \ldots, b_m$ of Λ there exists a unique dual basis $d_1, \ldots, d_m$ that satisfies $\text{Span}(b_1, \ldots, b_m) = \text{Span}(d_1, \ldots, d_m)$ and

$$\langle b_i, d_j\rangle = \begin{cases} 1 & , \quad \text{if } i = j \\ 0 & , \quad \text{if } i \neq j \end{cases}$$

for all $i, j \in \{1, \ldots, m\}$. Then $d_1, \ldots, d_m$ is a basis for Λ^*. In fact if B and D are the generator matrices corresponding to $b_1, \ldots, b_m$ and $d_1, \ldots, d_m$ respectively, then $D^\top = B^\top (BB^\top)^{-1}$. [15]

This makes it immediately clear that $(\Lambda^*)^* = \Lambda$ as $b_1, \ldots, b_m$ is again the dual basis to $d_1, \ldots, d_m$. Also we have that:

$$\begin{aligned}\det(\Lambda^*) &= \sqrt{\det(DD^\top)} = \sqrt{\det(((BB^\top)^{-1})^\top B \cdot B^\top (BB^\top)^{-1})} \\ &= \sqrt{\det((BB^\top)^{-1})} = \frac{1}{\sqrt{\det(BB^\top)}} = \frac{1}{\det(\Lambda)}\end{aligned}$$

Note that the dual and the direct sum commute as clearly $c \oplus d \in (\Lambda_1 \oplus \Lambda_2)^*$ iff $\langle c \oplus d, a \oplus 0\rangle = \langle c, a\rangle \in \mathbb{Z}$ and $\langle c \oplus d, 0 \oplus b\rangle = \langle d, b\rangle \in \mathbb{Z}$ for all $a \in \Lambda_1$ and $b \in \Lambda_2$. So $(\Lambda_1 \oplus \Lambda_2)^* = \Lambda_1^* \oplus \Lambda_2^*$. The same is also true for the tensor product.

Lemma 6 *(Dual and tensor product commute).* *Let Λ_1 and Λ_2 be lattices with dual Λ_1^* and Λ_2^* respectively. Then the dual of $\Lambda_1 \otimes \Lambda_2$ is given by $(\Lambda_1 \otimes \Lambda_2)^* = \Lambda_1^* \otimes \Lambda_2^*$.*

Proof. Let $a_1, \ldots, a_{m_1} \in \Lambda_1$ and $b_1, \ldots, b_{m_2} \in \Lambda_2$ be a basis of Λ_1 and Λ_2 respectively. Let $a_1^*, \ldots, a_{m_1}^* \in \Lambda_1^*$ and $b_1^*, \ldots, b_{m_2}^* \in \Lambda_2^*$ be their respective dual basis. Then $B^* := \{a_i^* \otimes b_j^* : i \in \{1, \ldots, m_1\}, j \in \{1, \ldots, m_2\}\}$ is a basis of $\Lambda_1^* \otimes \Lambda_2^*$. But we also have that

$$\langle a_i \otimes b_j, a_k^* \otimes b_l^*\rangle = \langle a_i, a_k^*\rangle \cdot \langle b_j, b_l^*\rangle = \begin{cases} 1 & , \quad \text{if } (i,j) = (k,l) \\ 0 & , \quad \text{else} \end{cases}$$

and thus B^* is the dual basis to $\{a_i \otimes b_j : i \in \{1, \ldots, m_1\}, j \in \{1, \ldots, m_2\}\}$ and therefore $\Lambda_1^* \otimes \Lambda_2^*$ must be the dual of $\Lambda_1 \otimes \Lambda_2$. □

3. The cyclotomic lattice

The motivation for this thesis comes from the cyclotomic fields. To be more precise using a (later defined) canonical inner product on cyclotomic fields, the ring of integers of these fields form a lattice by the more abstract definition. Later we will embed these lattices inside of a Euclidean space and see that they can all be constructed from the prime case lattices with the use of the tensor product and orthogonal direct sum.

3.1. Cyclotomic field.

First we recall some facts about the cyclotomic fields.

Definition 7 *(Trace).* Let $K \subset L$ be a finite Galois extension. Then the *Trace* $\mathrm{Tr}_{L/K} : L \to L$ of L over K of $\alpha \in L$ is given by:

$$\mathrm{Tr}_{L/K}(\alpha) = \sum_{\sigma \in \mathrm{Gal}(L/K)} \sigma(\alpha)$$

Because $K \subset L$ is a finite Galois extension we have that $\mathrm{Tr}_{L/K}(\alpha) \in K$ for all $\alpha \in L$ [16].

Definition 8 *(Cyclotomic field).* Let $n > 1$ and let $\zeta_n \in \mathbb{C}$ be an n-th *primitive root of unity*, i.e. $\zeta_n^n = 1$ and $\zeta_n^k \neq 1$ for $0 < k < n$. The n-th *cyclotomic field* $\mathbb{Q}(\zeta_n)$ is obtained by adjoining ζ_n to $\mathbb{Q}$. It is known that $\mathbb{Q} \subset \mathbb{Q}(\zeta_n)$ is a Galois extension. Indeed, $\mathbb{Q}(\zeta_n) \subset \mathbb{C}$ is the splitting field of the n-th cyclotomic polynomial

$$\Phi_n(X) = \prod_{1 \leq k \leq n : \gcd(k,n)=1} \left(X - e^{2i\pi \frac{k}{n}} \right),$$

over $\mathbb{Q}$ which is the unique irreducible monic polynomial of $\mathbb{Q}[X]$ with an n-th primitive root of unity as a root. So $\mathbb{Q}(\zeta_n) \cong \mathbb{Q}[X]/\Phi_n(X) =: C_n$. It is also known that $\mathrm{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q}) \cong \mathrm{Gal}(C_n/\mathbb{Q})$ is isomorphic to the multiplicative group $(\mathbb{Z}/n\mathbb{Z})^*$ by mapping $k \in (\mathbb{Z}/n\mathbb{Z})^*$ to the field automorphism of $\mathbb{Q}(\zeta_n)$ generated by mapping ζ_n to ζ_n^k. [16]

Let $\phi(n) := \deg(\Phi_n(X)) = \#(\mathbb{Z}/n\mathbb{Z})^*$ which is also called Euler's totient function. Note that we can also view $\mathbb{Q}(\zeta_n)$ and C_n as a $\mathbb{Q}$-vector space with basis $1, \zeta_n, \ldots, \zeta_n^{\phi(n)-1}$ and $1, X, \ldots, X^{\phi(n)-1}$ respectively. We define an inner product $\langle \, . \, , \, . \, \rangle : \mathbb{Q}(\zeta_n) \times \mathbb{Q}(\zeta_n) \to \mathbb{Q}$ by $\langle a, b \rangle := \frac{1}{n}\mathrm{Tr}_{\mathbb{Q}(\zeta_n)/\mathbb{Q}}(a\bar{b})$ where $\bar{b}$ is the complex conjugate of b. That this function is indeed an inner product is proved in Lemma 9. Note that for C_n the equivalent inner product is the bilinear extension of $\langle X^i, X^j \rangle = \frac{1}{n}\mathrm{Tr}_{C_n/\mathbb{Q}}(X^{i-j})$ as $\overline{\zeta_n^i} = \zeta_n^{-i}$.

Let $n = \prod_{l=1}^{k} n_l$ be the prime power factorization of n. An important property of the cyclotomic field is that it is isomorphic to the tensor product of prime power cyclotomic fields:

$$C_n \cong \bigotimes_{l=1}^{k} C_{n_l} \cong \mathbb{Q}[X_1, \ldots X_k] / \left(\Phi_{n_l}(X_1), \ldots, \Phi_{n_k}(X_k) \right)$$

via the correspondence $X^{\prod_{l \neq s} n_l} \leftrightarrow X_s$. This correspondence is very natural as $X^{\prod_{l \neq s} n_l}$ is a n_s-th primitive root in C_n.

This decomposition is compatible with the trace and thus the inner product. By the Chinese remainder theorem we have that $(\mathbb{Z}/n\mathbb{Z})^* \cong (\mathbb{Z}/n_1\mathbb{Z})^* \times \cdots \times (\mathbb{Z}/n_k\mathbb{Z})^*$ which in turn implies that

$$\mathrm{Tr}_{C_n/\mathbb{Q}}(a) = \prod_l \mathrm{Tr}_{C_{n_l}/\mathbb{Q}}(a_l)$$

where $a \in C_n$ corresponds to $\otimes_l a_l \in \bigotimes_{l=1}^{k} C_{n_l}$ [10]. As a corollary we get for $\otimes_l c_l, \otimes_l d_l \in \bigotimes_{l=1}^{k} C_{n_l}$ that:

$$\langle \otimes_l c_l, \otimes_l d_l \rangle = \frac{1}{n} \cdot \mathrm{Tr}_{C_n/\mathbb{Q}}(\otimes_l c_l \overline{d_l}) = \prod_l \frac{1}{n_l} \mathrm{Tr}_{C_{n_l}/\mathbb{Q}}(c_l \overline{d_l}) = \prod_l \langle c_l, d_l \rangle$$

which corresponds to the behaviour of the canonical inner product on $\mathbb{F}^n$.

Lemma 9. *The function $\langle \ . \ , \ . \ \rangle : \mathbb{Q}(\zeta_n) \times \mathbb{Q}(\zeta_n) \to \mathbb{Q}$ defined by $\langle a, b \rangle \mapsto \frac{1}{n} Tr_{\mathbb{Q}(\zeta_n)/\mathbb{Q}}(a\bar{b})$ is bilinear, symmetric and positive-definite and thus an inner product on $\mathbb{Q}(\zeta_n)$.*

Proof. Denote $G_n := \mathrm{Gal}(\mathbb{Q}(\zeta_n)/\mathbb{Q})$ and $\mathrm{Tr}_n := \mathrm{Tr}_{\mathbb{Q}(\zeta_n)/\mathbb{Q}}$. The bilinearity follows directly from the fact that Tr_n is a $\mathbb{Q}$-homomorphism.

Note that $-1 \in (\mathbb{Z}/n\mathbb{Z})^*$ and thus the field automorphism $\tau : \mathbb{Q}(\zeta_n) \to \mathbb{Q}(\zeta_n)$ generated by $\zeta_n \mapsto \zeta_n^{-1}$ which acts as the identity on $\mathbb{Q}$ is in G_n. We use the fact that $\overline{\zeta_n} = \zeta_n^{-1} = \tau(\zeta_n)$ which gives us that $\bar{a} = \tau(a)$ for all $a \in \mathbb{Q}(\zeta_n)$. Note that every element of G_n acts transitively on G_n by composition because it is a Galois group. But then for $a, b \in \mathbb{Q}(\zeta_n)$ we have

$$\langle a, b \rangle = \frac{1}{n} \sum_{\sigma \in G_n} \sigma(a\bar{b}) = \frac{1}{n} \sum_{\sigma \in G_n} (\sigma \circ \tau)(\bar{a}b) = \frac{1}{n} \sum_{\sigma \in G_n} \sigma(b\bar{a}) = \langle b, a \rangle$$

and thus $\langle \ . \ , \ . \ \rangle$ is symmetric.

For the positive-definiteness, let $a \in \mathbb{Q}(\zeta_n) \subset \mathbb{C}$. Note that all $\sigma \in G_n$ are field automorphisms and thus $\sigma(a\bar{a}) = \sigma(a)\overline{\sigma(a)} = |\sigma(a)|^2$ where $|.|$ denotes the absolute value on $\mathbb{C}$. This gives us that:

$$\langle a, a \rangle = \frac{1}{n} \mathrm{Tr}_n(a\bar{a}) = \frac{1}{n} \sum_{\sigma \in G_n} \sigma(a\bar{a}) = \frac{1}{n} \sum_{\sigma \in G_n} |\sigma(a)|^2 \geq 0$$

with equality iff $\sigma(a) = 0$ for all $\sigma \in G_n$ and thus iff $a = 0$. □

Before we can know which values this inner product on C_n takes we first need to know what values $\mathrm{Tr}_{C_n/\mathbb{Q}}$ takes on C_n. Let us quickly recall those values for n a power of prime.

Lemma 10 *(Trace values).* *For $n = p^k$ with p prime and $k > 0$ we have*

$$Tr_{C_n/\mathbb{Q}}(X^i) = \begin{cases} (p-1)p^{k-1} = \phi(n) & , \quad \text{if } i \equiv 0 \bmod p^k \\ -p^{k-1} & , \quad \text{if } i \not\equiv 0 \bmod p^k \text{ and } i \equiv 0 \bmod p^{k-1} \\ 0 & , \quad \text{else.} \end{cases}$$

Proof. We have that $\phi(p^k) = (p-1)p^{k-1}$ as $\gcd(p^k, i) = 1$ iff $p \nmid i$. Then it is clear from the degree that $\Phi_p(Y) = \frac{Y^p - 1}{Y - 1} = 1 + Y + \ldots + Y^{p-1}$. Also because x is a

primitive p^k-th root op unity iff $x^{p^{k-1}}$ is a primitive p-th root of unity we have that $\Phi_{p^k}(Y) = \Phi_p(Y^{p^{k-1}}) = 1 + Y^{p^{k-1}} + \ldots + Y^{(p-1)p^{k-1}}$.

Now first note that $\mathrm{Tr}_{C_n/\mathbb{Q}}(1) = \phi(n) = (p-1)p^{k-1}$ as $\#\mathrm{Gal}(C_n/\mathbb{Q}) = \phi(n)$ and all homomorphisms of $\mathrm{Gal}(C_n/\mathbb{Q})$ act as the identity on 1. Secondly note that $-\mathrm{Tr}_{C_n/\mathbb{Q}}(X)$ is the coefficient before $Y^{\phi(n)-1} = Y^{(p-1)p^{k-1}-1}$ in $\Phi_n(Y) = 1 + Y^{p^{k-1}} + \ldots + Y^{(p-1)p^{k-1}}$ which is clearly 1 if $k = 1$ and 0 if $k > 1$. So $\mathrm{Tr}_{C_n/\mathbb{Q}}(X) = -1$ if $k = 1$ and 0 if $k > 1$. Also $\mathrm{Tr}_{C_n/\mathbb{Q}}(X^i) = \mathrm{Tr}_{C_n/\mathbb{Q}}(X)$ for all $i \in (\mathbb{Z}/p^k\mathbb{Z})^*$ by transitivity of the Galois group on itself.

If $K \subset L$ is a Galois extension and if $K \subset K(x)$ is also a Galois extension for $x \in L$ (only needed for our less general definition of the Trace) we have that [16]:

$$\mathrm{Tr}_{L/K}(x) = [L : K(x)] \cdot \mathrm{Tr}_{K(x)/K}(x).$$

Using this and the fact that $X^{i \cdot p^j}$ is a p^{k-j}th primitive root of unity for all $i \in (\mathbb{Z}/p^k\mathbb{Z})^*$ we get for $i \in (\mathbb{Z}/p^k\mathbb{Z})^*$ and $j = 1, \ldots, k-1$ that:

$$\begin{aligned}\mathrm{Tr}_{C_n/\mathbb{Q}}(X^{i \cdot p^j}) &= [C_n : \mathbb{Q}(X^{i \cdot p^j})] \cdot \mathrm{Tr}_{C_{p^{k-j}}/\mathbb{Q}}(X^{i \cdot p^k}) \\ &= \begin{cases} p^j \cdot -1 & , \quad \text{if } j = k-1 \\ p^j \cdot 0 & , \quad \text{else} \end{cases}\end{aligned}$$

which proves the lemma. □

Note that the values for the trace when n is not a power of a prime follow from $\mathrm{Tr}_{C_n/\mathbb{Q}}(\otimes_l a_l) = \prod_l \mathrm{Tr}_{C_{n_l}/\mathbb{Q}}(a_l)$ where $n = \prod_l n_l$ is the prime power decomposition of n.

3.2. **Embedding the cyclotomic lattice.**

The lattice (by the more abstract definition) we will look at is the ring of integers $\mathbb{Z}[X]/\Phi_n(X)$ of C_n with the canonical inner product on C_n. Note that the decomposition into prime power cases for C_n also holds for $\mathbb{Z}[X]/\Phi_n(X)$. To get a lattice by our formal definition we will define an embedding $L_n \subset \mathbb{Q}^n$ of $\mathbb{Z}[X]/\Phi_n(X)$ such that the canonical inner product on C_n corresponds with the canonical inner product on $\mathbb{Q}^n$. Because of the decomposition into prime powers we only need to define the embedding for the prime power case as the general case follows from this by the tensor product.

Definition 11 *(Embedding in $\mathbb{Q}^n$).* For $n > 1$, the (cyclotomic) lattice $L_n \subset \mathbb{Q}^n$ of rank $\phi(n)$ is recursively defined as:

(1) If $n = p^k$ let L_n be the lattice with basis $b_1, \ldots, b_{\phi(n)} \in \mathbb{Q}^n$ where the coefficients b_{ij}, $1 \leq j \leq n$, of b_i are given by:

$$b_{ij} := \begin{cases} \frac{p-1}{p} & , \quad \text{if } i = j \\ -1/p & , \quad \text{if } i \neq j, i \equiv j \bmod p^{k-1} \\ 0 & , \quad \text{else} \end{cases}$$

We call this basis the powerful basis [10] and it corresponds to the basis $1, X, X^2, \ldots, X^{\phi(n)-1}$ of $\mathbb{Z}[X]/\Phi_n(X)$ by $b_i \leftrightarrow X^{i-1}$ in this case.

(2) If $n = c \cdot d$ with $\gcd(c, d) = 1$. Let $a_1, \ldots, a_{\phi(c)}$ be the powerful basis of L_c and $b_1, \ldots, b_{\phi(d)}$ the powerful basis of L_d. Then $L_n := L_c \otimes L_d \subset \mathbb{Q}^{cd} = \mathbb{Q}^n$ is the lattice with powerful basis $\{a_i \otimes b_j : 1 \leq i \leq \phi(c), 1 \leq$

$j \leq \phi(d)\}$. Note that this powerful basis doesn't correspond to the basis $1, X, \ldots, X^{\phi(n)-1}$ of $\mathbb{Z}[X]/\Phi_n(X)$. Also note that L_n has rank $\phi(c) \cdot \phi(d) = \phi(n)$ because c and d are coprime.

By the associativity of the tensor product this recursive definition of L_n defines L_n uniquely up to the permutation of coordinates. For $n = p^k$ it is easy to check that the Euclidean inner product between the basis vectors $b_1, \ldots, b_{\phi(n)}$ corresponds to the defined inner product between the basis elements $1, X, \ldots, X^{\phi(n)-1}$ of $\mathbb{Z}[X]/\Phi_n(X)$. Namely for $1 \leq i, j \leq \phi(n)$ we have by Lemma 10

$$\begin{aligned}\langle b_i, b_j \rangle &= \begin{cases} \frac{(p-1)^2+(p-1)\cdot(-1)^2}{p^2} = \frac{p-1}{p} & , \quad \text{if } i = 0 \\ \frac{(p-1)\cdot -1-1\cdot(p-1)+(p-2)\cdot(-1)^2}{p^2} = \frac{-1}{p} & , \quad \text{if } i \neq j \text{ and } i \equiv j \bmod p^{k-1} \\ 0 & , \quad \text{else} \end{cases} \\ &= \frac{1}{p^k}\mathrm{Tr}_{C_n/\mathbb{Q}}(X^{i-j}) = \langle X^{i-1}, X^{j-1} \rangle,\end{aligned}$$

so for $n = p^k$ the defined embedding is correct. For general $n > 1$ the correctness follows from the identical behaviour with tensor products of the canonical inner product on C_n and that of the canonical inner product on $\mathbb{F}^n$.

Note that for $n = p^k$ we can group the basis $1, X, \ldots, X^{\phi(n)-1}$ in p^{k-1} groups of the form $X^i, X^{i+p^{k-1}}, \ldots, X^{i+(p-2)p^{k-1}}$ of $p-1$ elements for $i = 0, \ldots, p^{k-1}-1$. By Lemma 10 we have that $\langle X^{i+c_1 p^{k-1}}, X^{j+c_2 p^{k-1}} \rangle = \frac{1}{n} Tr_{C_n/\mathbb{Q}}(X^{i-j+(c_1-c_2)p^{k-1}} = 0$ iff $i \neq j \bmod p^{k-1}$, so all these groups are orthogonal. As each such orthogonal group is in fact the same as L_p when looking at the values of its embedding we get that $L_{p^k} = \bigoplus_{i=1}^{p^{k-1}} L_p$ after reordering some coordinates.

In fact when using this embedding the prime case lattice L_p is identical to the well known lattice A_{p-1}^* which is the dual of the lattice A_{p-1}.

Definition 12 *(Root lattice A_m).* Let $m \geq 1$. The lattice $A_m \subset \mathbb{R}^{m+1}$ of rank m is defined as

$$A_m := \{(x_1, \ldots, x_{m+1}) \in \mathbb{Z}^{m+1} : \sum_{i=1}^{m+1} x_i = 0\},$$

i.e., all integer vectors of $\mathbb{Z}^{m+1}$ that sum up to zero. It has determinant $m+1$ and the shortest vectors are all permutations of $(1, -1, 0, \ldots, 0)$. Its packing radius is $\frac{1}{2}\sqrt{2}$ and its covering radius $\sqrt{\frac{a(m+1-a)}{m+1}}$ where $a = \lfloor (m+1)/2 \rfloor$ [5].

Definition 13 *(Dual lattice A_m^*).* Let $m \geq 1$. The lattice A_m^* dual to A_m has $m \times (m+1)$ generator matrix:

$$M = \frac{1}{m+1}\begin{pmatrix} m & -1 & \ldots & -1 & -1 \\ -1 & m & \ldots & -1 & -1 \\ \vdots & & \ddots & \vdots & \vdots \\ -1 & -1 & \ldots & m & -1 \end{pmatrix}$$

with $\frac{m}{m+1}$ on the diagonal and $\frac{-1}{m+1}$ everywhere else. It has packing radius $\frac{1}{2}\sqrt{\frac{m}{m+1}}$ and covering radius $\sqrt{\frac{m(m+2)}{12(m+1)}}$. Note that when $m = p-1$ for p prime we have that $A_{p-1}^* = L_p$ as they have the same generator matrix. [5]

A small technicality is that L_p is defined in $\mathbb{Q}^p$ and A^*_{p-1} in $\mathbb{R}^p$. For C_n it was important that we worked over $\mathbb{Q}$ instead of $\mathbb{R}$ as the extension $\mathbb{R} \subset \mathbb{R}(\zeta_n)$ wouldn't make much sense. Now we have embedded C_n inside of $\mathbb{Q}^n$ however there arise no problems (certainly no practical problems) when further embedding L_n in $\mathbb{R}^n$. Therefore from now on we will assume that L_n is a lattice in $\mathbb{R}^n$ just like we defined the embedding in $\mathbb{Q}^n$ such that $A^*_{p-1} = L_p$ makes sense.

So for $n = pq$ with p and q prime we have that $L_n = A^*_{p-1} \otimes A^*_{q-1}$ and by Lemma 6 its dual is $A_{p-1} \otimes A_{q-1}$. This encourages us to look at $A_m \otimes A_n$ and $A^*_m \otimes A^*_n$ for general $m, n \geq 1$. For $A_m \otimes A_n$ we will construct a polynomial CVP algorithm and for $A^*_m \otimes A^*_n$ we will construct a sub-exponential CVP algorithm in the rank mn.

4. Solving the closest vector problem in A_m and A_m^*

In this section we will fix $m \geq 1$ and let $m' := m + 1$. In this thesis all (time) complexity is given in the number of basic operations on reals, i.e., arithmetic operation with arbitrary precision count as $O(1)$.

We will show CVP algorithms for A_m and A_m^*, both in $O(m \log(m))$ operations. For A_m^* there exists a linear time algorithm [17], but the general idea lies already in the here presented algorithm.

4.1. The closest vector problem in A_m.

Note that for $t \in \text{span}(A_m) = \{(t_1, \ldots, t_{m'}) \in \mathbb{R}^{m'} : \sum_{i=1}^{m'} x_i = 0\}$ we want to find a closest integer vector x to t such that the coefficients of x sum to zero.

Algorithm 14. Given $t = (t_1, \ldots, t_{m'}) \in \text{span}(A_m)$, this algorithm finds a closest vector x to t in A_m [18].

(1) Let $x' := (\lceil t_1 \rfloor, \ldots, \lceil t_{m'} \rfloor) \in A_m$ where $\lceil . \rfloor$ means rounding to a nearest integer. It is clear that x' is a closest vector to t in $\mathbb{Z}^{m'}$. Let $\Delta = \sum_{i=1}^{m'} x'_i$ be the deficiency of x'. Note that $x' \in A_m$ iff $\Delta = 0$.

(2) Let $\delta(x_i) = x_i - \lceil x_i \rfloor$. We sort the x'_i on non-decreasing order of $\delta(x_i)$. So we get $i_1, \ldots, i_{m'}$ such that:
$$-\frac{1}{2} \leq \delta(x_{i_1}) \leq \delta(x_{i_2}) \leq \ldots \leq \delta(x_{i_{m'}}) \leq \frac{1}{2}$$

(3) If $\Delta = 0$, $x = x'$ is a closest vector to t.
If $\Delta > 0$, a closest vector x to t is obtained from x' by subtracting 1 from $x'_{i_1}, \ldots, x'_{i_\Delta}$.
If $\Delta < 0$, a closest vector x to t is obtained from x' by adding 1 to $x'_{i_{m'}}, \ldots, x'_{i_{m'+\Delta+1}}$.

This algorithm is correct because it makes the smallest possible changes to the norm of $x' - t$ (which is minimal after step (1)) to make sure x' lies in A_m. Note that every part of the algorithm can be done in time and space $O(m)$ except for the sorting in step (2) which takes time $O(m \log(m))$.

4.2. The closest vector problem in A_m^*.

For A_m^* we first need to narrow our search space. In this section when taking a point $x = (x_1, \ldots, x_m) \in \text{span}(A_m^*)$ we mean the point $\sum_{i=1}^{m} x_i b_i$ where $b_1, \ldots, b_m$ corresponds to the generator matrix given in Definition 13. Note that for this basis $\langle b_i, b_j \rangle = \frac{m}{m+1}$ if $i = j$ and $\frac{-1}{m+1}$ if $i \neq j$. This means that

$$\|x\|^2 = \langle \sum_{i=1}^{m} x_i b_i, \sum_{i=1}^{m} x_i b_i \rangle = \sum_{i=1}^{m} x_i^2 - \frac{1}{m'} \sum_{i=1}^{m} \sum_{j=1}^{m} x_i x_j$$

Lemma 15. *Let $t = (t_1, \ldots, t_m) \in \text{span}(A_m^*))$ be an arbitrary point. Suppose that $x = (x_1, \ldots, x_m) \in A_m^*$ is a closest vector to t, i.e. $\|t - x\| \leq \|t - x'\|$ for all $x' \in A_m^*$. Then $|t_i - x_i| \leq \frac{m}{m+1}$ for all $i = 1, \ldots, m$.*

Proof. Suppose that there exists an $i \in \{1, \ldots, m\}$ such that $|t_i - x_i| > \frac{m}{m+1}$. Because all basis elements are interchangeable with regard to the values of the inner product we can assume that $i = 1$ as the proof is identical for the other cases. We can also assume that $t_1 - x_1 > \frac{m}{m+1}$ as x is a closest vector to t iff $-x$ is a closest vector to $-t$. Let $y := t - x$. We will show that there exists a point x' of the lattice A_m^* that is strictly closer to t than x is. This will we proven in two cases.

First suppose that $y_2 + \ldots + y_m < \frac{m(m-1)}{2(m+1)}$. Let $x' := x + (1, 0, \ldots, 0)$. Then we have

$$\begin{aligned}
\left\|t - x'\right\|^2 - \|t - x\|^2 &= \|(y_1 - 1, y_2, \ldots, y_m)\|^2 - \|(y_1, y_2, \ldots, y_m)\|^2 \\
&= \frac{1}{m'}\left(-2my_1 + m + 2(y_2 + \ldots + y_m)\right) \\
&< \frac{1}{m'}\left(-2m \cdot \frac{m}{m+1} + m + 2 \cdot \frac{m(m-1)}{2(m+1)}\right) \\
&= 0
\end{aligned}$$

and thus $\|t - x'\| < \|t - x\|$ which contradicts the assumption that x is a closest vector to t.

Secondly suppose that $y_2 + \ldots + y_m \geq \frac{m(m-1)}{2(m+1)}$. Then $y_1 + \ldots + y_m > \frac{m(m-1)}{2(m+1)} + \frac{m}{m+1} = \frac{m}{2}$. Let $x' := x + (1, \ldots, 1)$. Then we have

$$\begin{aligned}
&\left\|t - x'\right\|^2 - \|t - x\|^2 = \|(y_1 - 1, y_2 - 1, \ldots, y_m - 1)\|^2 - \|(y_1, y_2, \ldots, y_m)\|^2 \\
&= \frac{1}{m'}\left(-2m(y_1 + \ldots + y_m) + m^2 - 2\left(-(m-1)(y_1 + \ldots + y_m) + \frac{m(m-1)}{2}\right)\right) \\
&= \frac{1}{m'}\left(-2(y_1 + \ldots + y_m) + m\right) \\
&< \frac{1}{m'}\left(-2 \cdot \frac{m}{2} + m\right) = 0
\end{aligned}$$

and thus $\|t - x'\| < \|t - x\|$ which also contradicts the assumption that x is a closest vector to t. So we have that $|t_i - x_i| \leq \frac{m}{m+1}$ for all $i = 1, \ldots, m$. □

Because $\frac{m}{m+1} < 1$ the consequence of Lemma 15 is that a closest vector to a point $t = (t_1, \ldots, t_m) \in \text{span}(A_m^*)$ must be in the following set:

$$S = \{(x_1, \ldots, x_m) \in A_m^* : |t_i - x_i| < 1 \ \ \forall i = 1, \ldots, m\} \subset A_m^*$$

Note that:

$$S \subset S' := \{(\lfloor t_1 \rfloor + s_1, \ldots, \lfloor t_m \rfloor + s_m) : s \in \{0, 1\}^m\}$$

and the closest vector problem is thus reduced to finding the $x \in S \subset S' \subset A_m^*$ that minimizes $\|t - x\|$. Let $y = t - \lfloor t \rfloor = (t_1 - \lfloor t_1 \rfloor, \ldots, t_m - \lfloor t_m \rfloor)$. For each $s \in \{0, 1\}^m$ we get a corresponding $x = \lfloor t \rfloor + s \in S'$ such that:

$$\|t - x\|^2 = \|y - s\|^2 = \sum_{i=1}^{m}\sum_{j=1}^{m} q_{ij}y_iy_j - 2\sum_{i=1}^{m}\sum_{j=1}^{m} q_{ij}s_iy_j + \sum_{i=1}^{m}\sum_{j=1}^{m} q_{ij}s_is_j$$

where $q_{ij} := \langle b_i, b_j \rangle$. Note that the first summation doesn't depend on $s \in \{0, 1\}^m$, so we want to minimize:

$$Q(s) = \sum_{i=1}^{m} c_is_i + \sum_{i=1}^{m}\sum_{j=1}^{m} q_{ij}s_is_j$$

with $c_i = -2 \cdot \sum_{j=1}^{m} q_{ij} y_j = -2y_i + \frac{2}{m'} \sum_{j=1}^{m} y_j$. Also note that with $T := \sum_{i=1}^{m} s_i$, the number of 1's we get that:

$$\sum_{i=1}^{m} \sum_{j=1}^{m} q_{ij} s_i s_j = \frac{m}{m'} \cdot \sum_{i=1}^{m} s_i^2 - \frac{1}{m'} \sum_{i=1}^{m} \sum_{j=1}^{m} s_i s_j = \frac{m}{m'} T - \frac{1}{m'} T(T-1)$$

So the second summation of $Q(s)$ only depends on T.

Now suppose $T = \tau \in \{0, \ldots, m\}$ is fixed, then we want to minimize $\sum_{i=1}^{m} c_i s_i$ under the condition $\sum_{i=1}^{m} s_i = \tau$ for $s \in \{0,1\}^m$. It is clear that we just have to take $s_i = 1$ for the τ smallest c_i's. So let $i_1, \ldots, i_m$ be an ordering of $1, \ldots, m$ such that $c_{i_1} \leq \ldots \leq c_{i_m}$. Then $s_{i_1} = \ldots = s_{i_\tau} = 1$ and $s_{i_j} = 0$ for all $j > \tau$ gives a minimal value of $Q(s)$ for fixed $T = \tau$. As T can only take m' values this gives an efficient way to find a $s \in \{0,1\}^m$ such that $Q(s)$ is minimal. Then a closest vector to t is given by $x = \lfloor t \rfloor + s$ as in that case $\|t - x\|^2$ is minimal by construction. This gives us the following algorithm.

Algorithm 16. Given a target $t = (t_1, \ldots, t_m) \in \operatorname{span}(A_m^*)$ this algorithm finds a closest vector $x \in A_m^*$ to t in A_m^*.

(1) First calculate $y := t - \lfloor t \rfloor$. Also calculate $Y := \sum_{j=1}^{m} y_j$ and $c_i = -2y_i + \frac{2}{m'} Y$ for $i = 1, \ldots, m$.

(2) Find different $i_1, \ldots, i_m$ such that $c_{i_1} \leq c_{i_2} \leq \ldots \leq c_{i_m}$.

(3) Let $Q := 0$, $Q' := 0$, $\text{minT} := 0$. For $\tau = 1, \ldots, m$:

  (a) Let $Q := Q + c_{i_\tau} + \frac{m}{m+1} + \frac{2-2\tau}{m+1}$.

  (b) If $Q < Q'$: Let $Q' := Q$ and $\text{minT} := \tau$.

(4) Let $s \in A_m^*$ be given by $s_{i_1} = \ldots = s_{i_{\text{minT}}} = 1$ and 0 else. Then $x := \lfloor t \rfloor + s$ is a closest vector to y.

Note that $c_{i_\tau} + \frac{m}{m+1} + \frac{2-2\tau}{m+1}$ is just the difference between the minimal $Q(s)$ with $T = \tau - 1$ and the minimal $Q(s)$ with $T = \tau$. Every iteration in step 3 can be calculated in a constant amount of operations. So it is clear that steps 1, 3 and 4 can all be done in $O(m)$ operations. Only in step 2 we need to sort m elements which brings the number of operations of the whole algorithm up to $O(m \log(m))$.

We discovered this algorithm independently for the lattice L_p. Later we discovered that $L_p = A_{p-1}^*$ and that the same algorithm was already presented in 2008 in [19] for general A_m^*. Later that year this was improved to a linear time algorithm in [17]. This algorithm is essentially the same except for the change that the c_i's don't need to be sorted perfectly but only in the buckets $[0, \frac{1}{m'}), [\frac{1}{m'}, \frac{2}{m'}), \ldots, [\frac{m}{m'}, 1]$ which can be done in a linear time.

5. General techniques for solving the closest vector problem

5.1. Composed lattices.

First we will cover two lemmas that relate the closest vector problem in different lattices to each other. To state these lemmas we will first need a definition of the cost of solving CVP in a lattice.

Definition 17. For a lattice Λ, let $C(\Lambda)$ be the maximum number of operations needed to find a closest vector in Λ to any target point in $\text{span}(\Lambda)$.

We will start with a very natural lemma when relating CVP in composed lattices to its components.

Lemma 18 *(Direct sum and orthogonal sum).* *Let Λ be a lattice and let $\Lambda_1, \ldots, \Lambda_k \subset \Lambda$ be orthogonal lattices such that*

$$\Lambda = \Lambda_1 \perp \ldots \perp \Lambda_k$$

Then:

(1) $C(\Lambda) \leq \sum_{i=1}^{k} C(\Lambda_i) + p_i$.

(2) $C(\Lambda_i) \leq C(\Lambda)$ *for all* $i = 1, \ldots, k$.

where p_i is the number of operations needed to project an element $x \in \text{span}(\Lambda)$ to $\text{span}(\Lambda_i)$ and to add a vector x_i to the already computed $x_1 + \ldots + x_{i-1}$ where $x_j \in \Lambda_j$ for all $j = 1, \ldots, i$. If Λ and $\Lambda_1, \ldots, \Lambda_k$ are lattices such that

$$\Lambda = \Lambda_1 \oplus \ldots \oplus \Lambda_k,$$

then we have the same inequalities but with $p_i = 0$ for all $i = 1, \ldots, k$.

Proof. We will first consider the case with the orthogonal sum. For (1), suppose that $t \in \text{span}(\Lambda)$ is the target and $t_1, \ldots, t_k$ are the projections onto $\text{span}(\Lambda_1), \ldots, \text{span}(\Lambda_k)$ of t. For each t_i we can calculate a closest vector $x_i \in \Lambda_i$ in $C(\Lambda_i)$ operations. Then $x = x_1 + \ldots + x_k \in \Lambda$ is clearly a closest vector to t by the orthogonality. The projection and last summation take p_i operations for each $i = 1, \ldots, k$.

For (2) suppose $t_i \in \text{span}(\Lambda_i) \subset \text{span}(\Lambda)$ is our target. Suppose $x \in \Lambda$ is a closest vector to t_i which can be obtained in $C(\Lambda)$ operations. Then $x \in \Lambda_i$ by the orthogonality because $t_i \in \text{span}(\Lambda_i)$ and thus x is a closest vector to t_i in Λ_i.

For the direct sum the proof is identical by using the embedding $\Lambda_i' = 0 \oplus \ldots \oplus \Lambda_i \oplus \ldots \oplus 0 \subset \Lambda$ such that $\Lambda = \Lambda_1' \perp \ldots \perp \Lambda_k'$. In this case the projections are along the coordinates and the summation is just concatenation and thus we can assume that $p_i = 0$ for all $i = 1, \ldots, k$ as no arithmetic operations are needed. □

Because there exists an algorithm for the prime case lattice $L_p = A_{p-1}^*$ that solves CVP in $O(p)$ operations [17], we get by $L_{p^k} = \bigoplus_{i=1}^{p^{k-1}} L_p$ and (1) a CVP algorithm for L_{p^k} in $p^{k-1} \cdot O(p) = O(p^k)$ operations. So we can also solve the prime power case $n = p^k$ of the cyclotomic lattice in linear time in n. Using the same technique for the dual of L_{p^k} we get a practically linear algorithm in $n = p^k$ as we can solve

it in $p^{k-1} \cdot O(p\log(p)) = O(p^k \log(p))$ operations with the algorithm for A_{p-1} showed in section 4.

Because of (2) an idea would be to add some orthogonal components to the lattice for which we are trying to solve CVP to obtain a much nicer lattice for solving CVP. This is exactly what we are going to do for $A_m^* \otimes A_n^*$.

If a lattice consists of multiple translated copies of another lattice we get the following lemma.

Lemma 19 *(Gluing Lemma).* *Let $\Lambda \subset \mathbb{F}^n$ be a lattice and let $\Lambda' \subset \Lambda$ be a sublattice. Note that Λ consists of multiple translated copies of Λ'. To be more precise, we can see Λ' as a subgroup of Λ, and then let $G = \Lambda/\Lambda'$ be the so called glue group consisting of cosets. Let $[\Lambda : \Lambda'] =: |G|$ be the index of Λ' in Λ and let $\mathcal{G} \subset \Lambda$ be a set consisting of a single representative for each coset in G, so called glue vectors. Then*

$$\Lambda = \bigcup_{g \in \mathcal{G}} (g + \Lambda')$$

and we have that

$$C(\Lambda) \leq |G|(O(n) + C(\Lambda')).$$

Proof. We make use of the fact that if $x \in \Lambda$ is a closest vector to $t \in \text{span}(\Lambda)$ that then $x \in g + \Lambda'$ for some $g \in \mathcal{G}$. This is equivalent to the fact that $x - g$ is a closest vector to $t - g$ in Λ'. So for all $g \in \mathcal{G}$ we find the closest vector x_g to $t - g$ in Λ' in $C(\Lambda')$ operations and we remember the $h = g$ for which which x_g has the minimal distance to their respective $t - g$. Then $x_h + h$ is a closest vector to t in Λ. Because we are calculating a distance and adding and subtracting vectors of length n for each $g \in \mathcal{G}$ we get the extra $O(n)$ operations on top of $C(\Lambda')$. □

We will use the two lemmas to later find a sub-exponential time CVP algorithm for the lattice $A_m^* \otimes A_n^*$. Now we will consider a method to solve CVP for general lattices which we will later use to find a polynomial CVP algorithm for the lattice $A_m \otimes A_n$.

5.2. Using the Voronoi region.

Although in 2015 there was found a general algorithm for solving CVP in $2^{n+O(n)}$ time and space with another technique [20], promising attempts to achieve a single time exponential complexity of $2^{O(n)}$ before that were driven by the use of a description of the Voronoi region of the lattice [21, 22]. We will quickly repeat the definition of the Voronoi region of a lattice.

Definition 20 *(Voronoi region).* The Voronoi region (around 0) of a lattice Λ is defined by:

$$\begin{aligned} V(\Lambda) &:= \{x \in \text{span}(\Lambda) : \|x\| \leq \|x - v\| \quad \forall v \in \Lambda\} \\ &= \{x \in \text{span}(\Lambda) : 2\langle x, v\rangle \leq \langle v, v\rangle \quad \forall v \in \Lambda\} \end{aligned}$$

consisting of all points in $\text{span}(\Lambda)$ that have 0 as a closest vector. It is known that the Voronoi region is a convex polytope which is symmetric by reflection in 0 [5].

The Voronoi region is just the intersection of half spaces $H_v := \{x \in \text{span}(\Lambda) : 2\langle x, v\rangle \leq \langle v, v\rangle\}$ for all $v \in \Lambda \setminus \{0\}$. Note that the only half spaces H_v in this intersection that matter are those corresponding to a facet ($\text{rank}(\Lambda) - 1$ dimensional face of $V(\Lambda)$) $\{x \in \text{span}(\Lambda) : \|x\| = \|x - v\|\} \cap V(\Lambda)$ of the Voronoi region. Such $v \in \Lambda$ are called Voronoi relevant vectors.

Definition 21 *(Voronoi Relevant vectors).* Let Λ be a lattice. The *Voronoi relevant vectors* are the minimal set $RV(\Lambda) \subset \Lambda$ of vectors such that

$$V(\Lambda) = \bigcap_{v \in RV(\Lambda)} H_v.$$

Voronoi showed that for $v \in \Lambda \setminus \{0\}$ we have that v is a Voronoi relevant vector iff 0 and v are the only closest vectors to $\frac{1}{2}v$ in Λ [23].

It was proved by Minkowski in 1897 that a lattice of rank m can only have at most $2(2^m - 1)$ Voronoi relevant vectors [24]. Voronoi showed that almost all general lattices have this number of Voronoi relevant vectors [23]. We will use a slightly different but equivalent definition for the Voronoi relevant vectors of a lattice.

Lemma 22. *Let Λ be a lattice. $v \in \Lambda \setminus \{0\}$ is a Voronoi relevant vector iff*

$$\langle v, x\rangle < \langle x, x\rangle$$

for all $x \in \Lambda \setminus \{0, v\}$.

Proof. Note that $\left\|\frac{1}{2}v - x\right\|^2 - \left\|\frac{1}{2}v\right\|^2 = \langle x, x\rangle - \langle v, x\rangle$ and thus for a $v \in \Lambda \setminus \{0\}$ and all $x \in \Lambda \setminus \{0, v\}$ we have that $\left\|\frac{1}{2}v - x\right\|^2 > \left\|\frac{1}{2}v\right\|^2$ iff $\langle v, x\rangle < \langle x, x\rangle$. Note that both 0 and v have exactly distance $\left\|\frac{1}{2}v\right\|$ to $\frac{1}{2}v$ and therefore the first statement is that of the definition, while the latter statement is that of the lemma. □

What makes the Voronoi relevant vectors relevant for CVP algorithms is the following lemma.

Lemma 23. *Let $t \in \text{span}(\Lambda)$ and $x \in \Lambda$. There exists a vector $y \in \Lambda$ such that $\|(x + y) - t\| < \|x - t\|$ iff there exists a Voronoi relevant vector $v \in RV(\Lambda)$ such that $\|(x + v) - t\| < \|x - t\|$.*

Proof. The implication from right to left is trivial by taking $y = v$. Now suppose there exists a vector $y \in \Lambda$ such that $\|t - x - y\| = \|(x + y) - t\| < \|x - t\| = \|t - x\|$. Then by definition $t - x \notin V(\Lambda)$. So there exists a $v \in RV(\Lambda)$ such that $\|t - x\| > \|(t - x) - v\|$, i.e., such that $\|x + v - t\| < \|x - t\|$. □

Because of Lemma 23 a basic iterative CVP algorithm can be constructed if the Voronoi relevant vectors are known. Given a target $t \in \text{span}(\Lambda)$ we can start the iterative algorithm with an arbitrary lattice point x. In each iteration if the current approximation x isn't yet a closest vector to t then by Lemma 23 there exists a Voronoi relevant vector v (which we can all check) such that $x \leftarrow x + v$ is strictly closer to t. We can repeat this until such a Voronoi relevant vector doesn't exist any more and Lemma 23 says that x is then a closest vector to t. This algorithm always concludes in a finite number of iterations because there are only a finite number of lattice points in the sphere around t with radius $\|t - x\|$ and in the

worst case the algorithm can only visit all those points a single time because of the strict improvement.

The just described algorithm is known as the Iterative Slicer (2007, [25]) and the main problem is that there isn't a nice bound on the number of iterations for general lattices except for the number of points with distance at most $\|t - x\|$ from t. But the number of such points can even lie above $n^{O(n)}$ for general lattices of rank n. In 2010 Micciancio and Voulgaris were able to compute the relevant vectors of a lattice in deterministic $2^{O(n)}$ time and space. Using these relevant vectors they were able to construct an algorithm that finds a closest point in deterministic $O(4^n)$ time by reducing (in polynomial time) the problem to a CVPP instance where the target is guaranteed to belong to $2V(\Lambda)$ [21]. In 2014 this was improved to a Las Vegas $O(2^n)$ expected time and space algorithm to find a closest vector by Dadush and Bonifas [22].

Although all these algorithms take exponential time and space to solve CVPP and thus also CVP for general lattice it may be possible that this isn't the case for special classes of lattices. In 2014 it was for example shown that a variant of the Iterative Slicer can be implemented in polynomial time for lattices of Voronoi's first kind (lattices which admit a set of $r + 1$ generators whose Gram matrix is the Laplacian of a non-negatively weighted graph) [26]. We will try to achieve a similar result for the lattice $A_m \otimes A_n$ for general $m, n \geq 1$.

6. Solving the closest vector problem in $A_m^* \otimes A_n^*$

Fix $m, n \geq 1$ and let $m' = m+1$ and $n' = n+1$. Before using Lemmas 18 and 19 for the lattice $A_m^* \otimes A_n^*$ we will first demonstrate their use for the lattice A_n^* which will also give the inspiration for solving CVP in $A_m^* \otimes A_n^*$ in sub-exponential time. Note that A_n^* has a generator matrix of the following form:

$$M = \frac{1}{n'} \begin{pmatrix} n & -1 & \dots & -1 & -1 \\ -1 & n & \dots & -1 & -1 \\ \vdots & & \ddots & \vdots & \vdots \\ -1 & -1 & \dots & n & -1 \end{pmatrix}.$$

Let $\mathbb{1}_{n'} = (1, \dots, 1) \in \mathbb{Z}^{n'}$. Let $I_{n'}$ be the lattice $\frac{1}{n'}\mathbb{1} \cdot \mathbb{Z} \subset \frac{1}{n'}\mathbb{Z}^{n'}$ with basis $\frac{1}{n'}\mathbb{1}$. Note that $I_{n'}$ is orthogonal to A_n^*. Therefore let $\overline{A_n^*} := A_n^* \perp I_{n'}$. Note that $\overline{A_n^*}$ has generator matrix

$$\overline{M} = \frac{1}{n'} \begin{pmatrix} n & -1 & \dots & -1 & -1 \\ -1 & n & \dots & -1 & -1 \\ \vdots & & \ddots & \vdots & \vdots \\ -1 & -1 & \dots & n & -1 \\ 1 & 1 & \dots & 1 & 1 \end{pmatrix}.$$

But we can just add the last row to all previous rows to get the following generator matrix of $\overline{A_n^*}$:

$$\overline{M} = \frac{1}{n'} \begin{pmatrix} n' & 0 & \dots & 0 & 0 \\ 0 & n' & \dots & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & n' & 0 \\ 1 & 1 & \dots & 1 & 1 \end{pmatrix}.$$

It is clear that then $\overline{A_n^*} = \mathbb{Z}^{n'} + I_{n'}$ (all possible $a + b$ such that $a \in \mathbb{Z}^{n'}$ and $b \in I_{n'}$). Because $n' \cdot I_{n'} \subset \mathbb{Z}^{n'}$ we even have that

$$\overline{A_n^*} = \mathbb{Z}^{n'} + I_{n'} = \bigcup_{i=0}^{n} \frac{i}{n'}\mathbb{1} + \mathbb{Z}^{n'}$$

where the glue group G of $\mathbb{Z}^{n'} + I_{n'}$ in $\mathbb{Z}^{n'}$ has glue vectors $\mathcal{G} = \{\frac{i}{n'}\mathbb{1} : i = 0, \dots, n\}$. Note that by Lemmas 18 and 19 we have that

$$\begin{aligned} C(A_n^*) \leq C(\overline{A_n^*}) = C(\mathbb{Z}^{n'} + I_{n'}) &= C(\bigcup_{i=0}^{n} \frac{i}{n'}\mathbb{1} + \mathbb{Z}^{n'}) \\ &\leq n'(O(n') + C(\mathbb{Z}^{n'})) = O((n')^2). \end{aligned}$$

So these two lemmas already give a quadratic algorithm for A_n^*. There already exists a linear time algorithm for A_n^*, so this seems useless, but it gives inspiration for a CVP algorithm in $A_m^* \otimes A_n^*$.

Lemma 24. *Given $t \in \operatorname{span}(A_m^* \otimes A_n^*)$ we can find a closest vector $x \in A_m^* \otimes A_n^*$ to t in $O(n(m+1)^{n+1})$ operations.*

Proof. For $A_m^* \otimes A_n^*$ we consider the lattice $\overline{A_m^*} \otimes A_n^* \supset A_m^* \otimes A_n^*$. We get that:

$$\overline{A_m^*} \otimes A_n^* = (A_m^* \otimes A_n^*) \perp (I_{m'} \otimes A_n^*)$$

such that $C(A_m^* \otimes A_n^*) \leq C(\overline{A_m^*} \otimes A_n^*)$ by Lemma 18. Now instead of considering the sublattice $\mathbb{Z}^{m'} \subset \overline{A_m^*}$ as before we consider the sublattice $\mathbb{Z}^{m'} \otimes A_n^* \subset \overline{A_m^*} \otimes A_n^*$. Note that $\mathbb{Z}^{m'} \otimes A_n^* = \bigoplus_{i=1}^{m'} A_n^*$ and thus $C(\mathbb{Z}^{m'} \otimes A_n^*) = m' \cdot C(A_n^*) \in O(mn)$ by Lemma 18 and the linear time algorithm for A_n^* [17].

The glue group $G := (\overline{A_m^*} \otimes A_n^*)/(\mathbb{Z}^{m'} \otimes A_n^*)$ consists of $(m')^n$ cosets represented by glue vectors $\sum_{i=1}^{n} (b_i \otimes \frac{a_j}{m'} \mathbb{1})$ for all $a = (a_1, \ldots, a_n) \in \{0, \ldots, m\}^n$ where the basis $b_1, \ldots, b_n$ is the basis corresponding to the generator matrix M of A_n^*. Summarizing we get a time complexity of:

$$\begin{aligned} C(A_m^* \otimes A_n^*) &\leq C(\overline{A_m^*} \otimes A_n^*) = C(\bigcup_{g \in \mathcal{G}} g + (\mathbb{Z}^{m'} \otimes A_n^*)) \\ &\leq (m')^n \cdot (O(m'n') + C(\mathbb{Z}^{m'} \otimes A_n^*)) \in O(m'n'(m')^n) = O(n(m')^{n'}) \end{aligned}$$

□

Assuming $m \geq n$, which we can do as $A_m^* \otimes A_n^*$ is the same as $A_n^* \otimes A_m^*$ after permuting some coordinates, we then get in the rank $r = mn$ of $A_m^* \otimes A_n^*$ a sub-exponential time complexity of $O(r \cdot r^{\sqrt{r}}) = O(e^{(\sqrt{r}+1)\log(r)})$.

We tried several different techniques to construct a CVP algorithm for $A_m^* \otimes A_n^*$ but they all delivered at best the same complexity as the stated algorithm.

Note that this gives a time $O(q \cdot p^q)$ complexity to solve CVP for the lattice $L_n = L_p \otimes L_q = A_{p-1}^* \otimes A_{q-1}^*$ with $n = p \cdot q$ and p and q prime. For the case $n = p^k \cdot q^l$ we then get that L_n is the direct sum of $p^{k-1} \cdot q^{l-1}$ times the lattice $L_p \otimes L_q$ and thus by Lemma 18 we get a time complexity of $p^{k-1}q^{l-1} \cdot O(qp^q) = O(np^{q-1})$.

7. Solving the closest vector problem in $A_m \otimes A_n$

Again let $m, n \geq 1$, $m' = m+1$ and $n' = n+1$. We consider the lattice $A_m \otimes A_n \subset \mathbb{Z}^{m' \cdot n'}$ of rank mn. Note that this lattice consists of all elements $x = (x_{11}, \ldots, x_{1n'}, x_{21} \ldots, x_{m'n'}) \in \mathbb{Z}^{m' \cdot n'}$ which satisfy the following conditions:

(1) $\sum_{i=1}^{m'} x_{ij} = 0$ for all $j = 1, \ldots, n'$

(2) $\sum_{j=1}^{n'} x_{ij} = 0$ for all $i = 1, \ldots, m'$.

These indices will be used throughout this section.

7.1. Characterizing the Voronoi relevant vectors.

As announced we will try to construct a polynomial CVP algorithm for the lattice $A_m \otimes A_n$ inspired by the Iterative Slicer algorithm. For this we will try to characterize the Voronoi relevant vectors of $A_m \otimes A_n$. First we will limit our search space.

Lemma 25. *For all Voronoi relevant vectors $v \in A_m \otimes A_n$ we have that $|v_{ij}| < 2$ for all $i = 1, \ldots, m'$ and $j = 1, \ldots, n'$.*

Proof. Let $v \in A_m \otimes A_n$ be a Voronoi relevant vector. Because of symmetry we can assume without loss of generality that $|v_{11}| \geq 2$. Because v is a Voronoi relevant vector if and only if $-v$ is a Voronoi relevant vector we can also assume that $v_{11} \geq 2$. Let $x^{ij} \in A_m \otimes A_n$ for all $i = 2, \ldots, m'$ and $j = 2, \ldots, n'$ be given by $x_{11} = 1$, $x_{i1} = -1$, $x_{1j} = -1$, $x_{ij} = 1$ and 0 otherwise. Note that this is indeed a lattice point of $A_m \otimes A_n$ and that it is not the same as 0 or v. Also note that $\langle x^{ij}, x^{ij} \rangle = 4$ for all i, j. Then by Lemma 22 we get

$$v_{11} - v_{1j} - v_{i1} + v_{ij} = \langle v, x^{ij} \rangle < \langle x^{ij}, x^{ij} \rangle = 4$$

for all $i = 2, \ldots, m'$ and $j = 2, \ldots, n'$. Also note that because these are all integers we even have that $v_{11} - v_{1j} - v_{i1} + v_{ij} \leq 3$. Summing multiple of these relations for a fixed $i = 2, \ldots, m'$ gives

$$n \cdot v_{11} - n \cdot v_{i1} - \sum_{j=2}^{n'} v_{1j} + \sum_{j=2}^{n'} v_{ij} = \sum_{j=2}^{n'} \left(v_{11} - v_{1j} - v_{i1} + v_{ij} \right) \leq 3(n'-1)$$

but we have that $-\sum_{j=2}^{n'} v_{1j} = v_{11}$ and $\sum_{j=2}^{n'} v_{ij} = v_{i1}$ and thus this gives us

$$n' \cdot v_{11} - n' \cdot v_{i1} \leq 3(n'-1).$$

As a result of $v_{11} \geq 2$ we now get that $n' \cdot v_{i1} \geq -n' + 3$ and thus $v_{i1} \geq -1 + \frac{3}{n'} > -1$, which again means that $v_{i1} \geq 0$ because it is an integer. So $v_{i1} \geq 0$ for all $i = 2, \ldots, m'$ and $v_{11} \geq 2$. But in that case:

$$0 = \sum_{i=1}^{m'} v_{i1} \geq 2 + 0 + \ldots + 0 = 2$$

which gives a contradiction. So $|v_{11}| < 2$. □

As a result all Voronoi relevant vectors of $A_m \otimes A_n$ must lie in $X := \{-1, 0, 1\}^{m' \cdot n'} \cap (A_m \otimes A_n)$. To be able to describe the Voronoi relevant vectors in a nice way and to later construct the CVP algorithm we will show that there is a correspondence between the elements of X and certain subgraphs of the complete directed bipartite labelled graph $K_{m',n'} = (V, E)$. We label the m' nodes $V_1 := \{v_1, \ldots, v_{m'}\}$ and the n' nodes $V_2 := \{w_1, \ldots, w_{n'}\}$. Let $V := V_1 \cup V_2$. Next we let a coefficient $t_{ij} \in X$ correspond to the pair (v_i, w_j) of nodes of $K_{m',n'}$. We can even go further and let the value of t_{ij} correspond with an edge from v_i to w_i, no edge, or an edge from w_i to v_i. We will now make this more formal.

Definition 26. Let $t \in \{-1, 0, 1\}^{m' \cdot n'}$ be given. We will define the subgraph $G_t = (V_t, E_t) \subset K_{m',n'} = (V, E)$ corresponding to t. Let E_t consist of the following directed edges:

- The edge (v_i, w_j) for each t_{ij} that has value -1.
- The edge (w_j, v_i) for each t_{ij} that has value 1.

and let V_t consist of all nodes with nonzero in- or outdegree.

It is clear that all G_t are different for $t \in \{-1, 0, 1\}^{m' \cdot n'}$. Also note that the conditions for $t \in \{-1, 0, 1\}^{m' \cdot n'}$ to be part of $A_m \otimes A_n$ correspond exactly to the fact that for every node of G_t the indegree must equal the outdegree. I.e., every node of G_t has exactly as much incoming edges as outgoing edges if and only if $t \in X$. An example is shown in Figure 7.1.

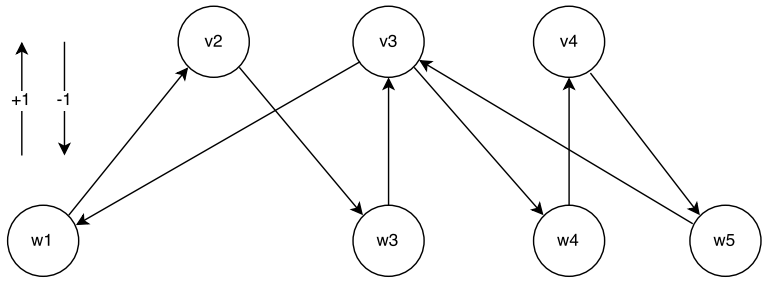

FIGURE 1. Example graph G_t corresponding to $t = (0, 0, 0, 0, 0, 1, 0, -1, 0, 0, -1, 0, 1, -1, 1, 0, 0, 0, 1, -1) \in A_3 \otimes A_4$

It is also clear that for every subgraph $H \subset K_{m',n'}$ that has at most one edge between any pair of nodes, no nodes without incoming or outgoing edges, and such that every node has indegree equal to its outdegree, we can construct an $x \in X$ such that $H = G_x$.

Proposition 27 *(Voronoi relevant vectors of $A_m \otimes A_n$).* *The Voronoi relevant vectors of $A_m \otimes A_n$ are precisely all $v \in X \setminus \{0\}$ such that G_v is connected and the indegree and outdegree of every node is exactly* 1.

Proof. Let $v \in X \setminus \{0\}$ be given. Note that we already have

$$\langle v, x \rangle \leq \sum_{i,j} |x_i| \leq \sum_{i,j} |x_i|^2 = \langle x, x \rangle$$

for all $x \in A_m \otimes A_n$ because $v \in X \subset \{-1, 0, 1\}^{m'n'}$. The second inequality can only be an equality if also $x \in X$. The first inequality then becomes an equality iff $v_{ij} x_{ij} = |x_{ij}|$ for all $i = 1, \ldots, m'$ and $j = 1, \ldots, n'$. So $x_{ij} = 0$ or $x_{ij} = v_{ij}$. This

makes it clear that the only candidates such that $\langle v, x\rangle = \langle x, x\rangle$ are those $x \in X$ such that $G_x \subset G_v$. By Lemma 22 we get that $v \in RV(A_m \otimes A_n)$ iff G_0 and G_v are the only subgraphs of that form of G_v.

In fact note that each G_x with $x \in X$ consists of a union of disconnected Eulerian graphs and thus a union of disconnected cycles. Furthermore note that every cycle in G_x corresponds to a subgraph $H \subset G_x$ for which there exists an $x' \in X$ such that $H = G_{x'}$. But that means that G_v is a Voronoi relevant vector iff G_v contains only the trivial cycles G_0 and G_v and no other cycles. We will show that this is only the case when G_v is a simple cycle.

Because G_v is a union of disconnected cycles we must have that G_v is connected as otherwise taking one of those disconnected cycles would give a nontrivial subgraph $G_x \subsetneq G_v$. So G_v must be connected and thus consist of a single cycle. In the case G_v contains a node w that has degree at least 2 we know that the single cycle that G_v consists of, must contain a nontrivial cycle, the one when starting in w and returning to w for the first time. So G_v must be connected and the indegree and outdegree of every node must be 1. But in that case G_v is a simple cycle and it is clear that G_v only has the trivial cycles corresponding to G_0 and G_v. So v is a Voronoi relevant vector in that case. □

The notion that G_v is connected and that the indegree and outdegree of each node is equal to 1 means just that the whole graph consists of a single directed simple cycle. Furthermore note that every simple cycle of more than 2 nodes (only a simple cycle with 2 nodes has two edges between a pair of nodes which isn't allowed) corresponds to a Voronoi relevant vector of $A_m \otimes A_n$.

Lemma 28. *The number of Voronoi relevant vectors of $A_m \otimes A_n$ is equal to*

$$\sum_{i=2}^{min\{m',n'\}} \binom{m'}{i}\binom{n'}{i} \cdot i! \cdot (i-1)!.$$

Proof. The number of Voronoi relevant vectors of $A_m \otimes A_n$ is by proposition 27 equal to the number of simple cycles of the directed labelled complete bipartite graph $K_{m',n'}$ of more than 2 nodes. Note that a simple cycle of $2 < 2i \leq 2(m'+n')$ nodes of this bipartite graph must consist of i nodes from V_1 and i nodes from V_2. Which nodes will be used for fixed i can be chosen in $\binom{m'}{i}\binom{n'}{i}$ ways as $|V_1| = m'$ and $|V_2| = n'$.

After these nodes are fixed the problem is reduced to the number of Hamiltonian cycles in the directed complete bipartite graph $K_{i,i}$. Let W_1 and W_2 be the two sets of nodes of $K_{i,i}$. Because every Hamiltonian cycle must visit every node once we can just fix our starting node in W_1. From this node we can visit i nodes of W_2. After this choice is made we can visit $i-1$ nodes of W_1. After this $i-1$ nodes of W_2, etc... So in the end we have $i \cdot (i-1) \cdots 1$ ways to go from W_1 to W_2 and $(i-1)(i-2)\cdots 1 = (i-1)!$ ways to go from W_2 to W_1. So in total there are $i! \cdot (i-1)!$ Hamiltonian cycles in $K_{i,i}$. So there are $\binom{m'}{i}\binom{n'}{i} \cdot i! \cdot (i-1)!$ simple cycles in $K_{m',n'}$ of $2i$ nodes. Summing over $i = 2, \ldots, \min\{m', n'\}$ gives the result. □

So the Voronoi relevant vectors of $A_m \otimes A_n$ correspond with directed simple cycles in $K_{m',n'}$ of at least 4 nodes.

7.2. Finding the closest vector in $A_m \otimes A_n$.

For constructing an efficient iterative CVP algorithm we need to find, if one exists, a Voronoi relevant vector that improves our current approximation in an efficient way. By Lemma 28 it is clear that we can't just check all Voronoi relevant vectors as there are too many of them. To find such a Voronoi relevant vector efficiently we will use the correspondence found in the previous subsection.

For the algorithm we will need to detect negative simple cycles in a directed graph and we can use the Bellman-Ford algorithm to do so [27]. The main goal of this algorithm is to find the shortest path from a single source node to all other nodes of a weighted graph. The algorithm does this by remembering for each node i the shortest distance $d(i)$ known to the source node (initialized on ∞ for all nodes except the source node itself) and to check if any edge between nodes i and j can improve this distance to j. This is the case iff $d(i) + c_{ij} < d(j)$ where c_{ij} is the (possibly negative) cost for travelling by the edge (i, j). In each iteration we do this check for every edge (i, j) and we update $d(j)$ to $d(i) + c_{ij}$ if needed.

If there are no negative weight cycles all shortest paths have at most length k for k the number of nodes and thus after k iterations all $d(i)$ are minimal. If there does exist a negative cycle and assuming the graph is connected there will still be improvements made to some $d(j)$ in the $(k+1)$-th iteration. So in this way we can detect negative weight cycles in the graph. If we also remember for each node j the last node i that improved $d(j)$ by the edge (i, j) we are also able to find such a simple negative weight cycle. Note however that there can't be made any guarantees about how negative the found simple negative weight cycle is compared to others that may exist in the graph. Finding the most negative cycle is in fact NP-hard for most types of graphs [28].

Lemma 29. *Let $x \in A_m \otimes A_n$ and let $t \in Span(A_m \otimes A_n)$ be our target. If there exists a Voronoi relevant vector $v \in RV(A_m \otimes A_n)$ such that $\|(x+v) - t\| < \|x - t\|$ we can find such a Voronoi relevant vector in $O((m+n)mn)$ operations. If it doesn't exist this will also be detected by the algorithm.*

Proof. Let $u := x - t$ be the difference vector of t and x. We construct the weighted directed complete bipartite graph $K_{m',n'}$ with weight function W defined as follows for $i = 1, \dots, m'$ and $j = 1, \dots, n'$:

$$W(v_i, w_j) = (u_{ij} - 1)^2 - u_{ij}^2 = 1 - 2u_{ij}$$
$$W(w_j, v_i) = (u_{ij} + 1)^2 - u_{ij}^2 = 1 + 2u_{ij}.$$

Now consider a $G_v \subset K_{m',n'}$ with the same weights for an arbitrary $v \in RV(A_m \otimes A_n)$. Then by construction

$$W(G_v) = \sum_{i,j:v_{ij} \neq 0} 1 + 2v_{ij} \cdot u_{ij} = \langle v, v \rangle + 2\langle v, u \rangle = \|u + v\|^2 - \|u\|^2 .$$

So $\|(x+v) - t\| < \|x - t\|$ for a $v \in RV(A_m \otimes A_n)$ iff $G_v \subset K_{m',n'}$ has negative weight. By Lemma 27 every simple cycle of length at least 4 in $K_{m',n'}$ corresponds to a Voronoi relevant vector. So the problem of finding a $v \in RV(A_m \otimes A_n)$ such that $\|(x+v) - t\| < \|x - t\|$ is equivalent to finding a simple cycle of length at least 4 with negative weight in $K_{m',n'}$. Note that because $W(v_i, w_j) + W(w_j, v_i) = 2 \geq 0$ for all $i = 1, \dots, m'$ and $j = 1, \dots, n'$ there exist no simple cycles of length 2. So we just need to find a simple cycle of negative weight. This can be done by the Bellman-Ford algorithm in $O(|V| \cdot |E|) = O((m' + n')m'n') = O((m+n)mn)$ operations. The construction of the graph itself can easily be done in $O((m +$

$n)mn)$ operations and thus adds nothing to the complexity. The Bellman-Ford algorithm also detects if simple negative weight cycles exist or not. □

Before we move on we will introduce a basis of $A_m \otimes A_n$ that has some nice properties. First let $b^{ij} \in A_m \otimes A_n$ be given by $b^{ij}_{ij} = 1, b^{ij}_{i+1,j} = -1, b^{ij}_{i,j+1} = -1, b^{ij}_{i+1,j+1} = 1$ and 0 otherwise for all $i = 1, \dots, m$ and $j = 1, \dots, n$. Note that $B := \{b^{ij} : i = 1, \dots, m \text{ and } j = 1, \dots, n\}$ is a basis of $A_m \otimes A_n$. Because the basis B is so sparse we can efficiently encode and decode elements in this basis.

Lemma 30. *For any $t \in \text{span}(A_m \otimes A_n)$ we can find an $x \in A_m \otimes A_n$ such that $\|x - t\| \leq 2\sqrt{m'n'}$ in $O(mn)$ operations.*

Proof. Suppose that $t' := t = \sum_{i,j} a_{ij} b^{ij}$. Then we have that $a_{11} = t'_{11}$ as all other basis elements have coefficient 0 there. Then let $t' \leftarrow t' - a_{11} \cdot b^{11}$ and consider a_{12}. We again have that $a_{12} = t'_{12}$ and after this we set $t' \leftarrow t' - a_{12} \cdot b^{12}$. This equality will be the case for all basis elements if we continue $b^{13}, \dots, b^{1n}, b^{2m}, \dots, b^{mn}$. Note that calculating $t' \leftarrow t' - a_{ij} b^{ij}$ can be done in a constant amount of operations as b^{ij} always has only 4 nonzero coefficients. In total calculating all a_{ij} can thus be done in $O(mn)$ operations. So we now have $a_{ij} \in \mathbb{R}$ such that $t = \sum_{i,j} a_{ij} b^{ij}$.

Let $x := \sum_{i,j} \lfloor a_{ij} \rceil b^{ij} \in A_m \otimes A_n$. Again it is clear that x can be calculated in $O(mn)$ operations as every b^{ij} has only 4 nonzero coefficients. Now note that

$$\|x - t\| = \left\| \sum_{i,j} (\lfloor a_{ij} \rceil - a_{ij}) b^{ij} \right\| \leq \sqrt{m'n' \cdot (4 \cdot \frac{1}{2})^2} = 2\sqrt{m'n'}$$

which is the case because the (kl)-th coefficient is nonzero in at most 4 basis vectors b^{ij} and combining this with the fact that $|\lfloor a_{ij} \rceil - a_{ij}| \leq \frac{1}{2}$ gives us that the (kl)-th coefficient of $x - t$ is bounded in absolute value by $4 \cdot \frac{1}{2} = 2$ for all $k = 1, \dots, m'$ and $l = 1, \dots, n'$. □

Now we have enough to construct a polynomial time CVP algorithm for the lattice $A_m \otimes A_n$. Given a target $t = \sum_{i,j} a_{ij} b^{ij} \in \text{span}\,(A_m \otimes A_n) \cap \left(2^{-d}\mathbb{Z}^{m'n'}\right)$ we will find a closest vector to t in $O(d \cdot (mn)^2(m+n))$ operations. From the transformation in the proof of Lemma 30 it is clear that then also all $a_{ij} \in 2^{-d}\mathbb{Z}$. Note that we only need to find a way to bound the number of iterations of the iterative slicer. We use the fact that $A_m \otimes A_n$ has only integer vectors and thus if $t \in 2^{-i}\mathbb{Z}^{m'n'}$ the squared distance to the target will in each iteration improve with at least 2^{-i+1} which is exactly what we need to bound the number of iterations.

Algorithm 31. *A polynomial CVP algorithm for the lattice $A_m \otimes A_n$.*

Input : $m, n, d \geq 1$ *and* $t = \sum_{i,j} a_{ij} b^{ij} \in \operatorname{span}(A_m \otimes A_n)$ *with* $a_{ij} \in 2^{-d}\mathbb{Z}$

Output: *a closest vector to* t *in* $A_m \otimes A_n$

```
(a_kl)_{k,l} = EmbedtoBasis( t );
a = Σ_{k,l} ⌊a_kl⌉ b^kl;
for i = 0, ..., d do
    // Outer loop
    t_i = Σ_{k,l} 2^{-i} ⌊2^i · a_kl⌉ b^kl;
    while true do
        // Inner loop
        Construct weighted K_{m',n'}; (as in Lemma 29 with u = a − t_i )
        if K_{m',n'} has a negative cycle G_v then
            a = a + v;
        else
            break;
    x_i = a;
return x_d;
```

Theorem 32. *Given a target* $t = \sum_{i,j} a_{ij} b^{ij} \in \operatorname{span}(A_m \otimes A_n)$ *with all* $a_{ij} \in 2^{-d}\mathbb{Z}$ *and with* $d \geq 1$ *we can find a closest vector to* t *in* $A_m \otimes A_n$ *in* $O(d \cdot (mn)^2(m+n))$ *operations using Algorithm 31.*

Proof. First note that by Lemmas 23 and 29 it is clear that after each outer loop x_i is a closest vector to t_i. Therefore we will focus on the complexity. First let $a_{kl} \in 2^{-d}\mathbb{Z}$ such that $t = \sum_{k,l} a_{kl} b^{kl} \in 2^{-d}\mathbb{Z}^{m'n'}$. Recall that this can be done in time $O(mn)$. Let $t_i := \sum_{k,l} 2^{-i} \lceil 2^i \cdot a_{kl} \rfloor b^{kl}$ for $i = 0, \ldots, d$, so $t_d = t$. Recall that these can also be calculated in time $O(mn)$ each as each b^{kl} has only 4 nonzero coefficients. Let x_i be the closest vector to t_i as obtained by the algorithm for $i = 0, \ldots, d$. Let $e_i = \sum_{k,l} a'_{kl} b^{kl} := t_i - t_{i-1}$ and note that $\|t_i - t_{i-1}\| = \|e_i\| \leq 4 \cdot 2^{-i}\sqrt{m'n'}$ as every $|a'_{kl}| \leq 2^{-i}$ and for every coefficient there are at most 4 basis elements that are nonzero there.

Note that if our current target is t_i and our current best approximation is $a \in A_m \otimes A_n$ we will improve in every iteration with at least 2^{-i+1} between squared distances if we improve at all as for a relevant vector $v \in RV(A_m \otimes A_n)$ we have

$$\|a + v - t_i\|^2 - \|a - t_i\|^2 = 2\langle a - t_i, v\rangle + \langle v, v\rangle \in 2^{-i+1}\mathbb{Z}^{m'n'}$$

because a and v are integer vectors and $t_i \in 2^{-i}\mathbb{Z}^{m'n'}$.

When searching a closest vector to t_i we start with the approximation x_{i-1}. To bound the number of iterations of the inner loop to get to x_i we need the following

bound for $i \geq 1$:

$$\begin{aligned}
&\|t_i - x_{i-1}\|^2 - \|t_i - x_i\|^2 \\
&= (\|t_i - x_{i-1}\| + \|t_i - x_i\|)(\|t_i - x_{i-1}\| - \|t_i - x_i\|) \\
&\leq (\|t_{i-1} - x_{i-1}\| + \|e_i\| + \|t_i - x_i\|)\,(\|t_{i-1} - x_{i-1}\| + \|e_i\| - \|t_i - x_i\|)
\end{aligned}$$

Note that by Lemma 30 be have that $\|t_i - x_i\| \leq 2\sqrt{m'n'}$ for all $i \geq 0$. Therefore:

$$\begin{aligned}
&\leq \left(4 + 2^{-i+2}\right)\sqrt{m'n'}\left(2^{-i+2}\sqrt{m'n'} + \operatorname{dist}(t_{i-1}, A_m \otimes A_n) - \operatorname{dist}(t_i, A_m \otimes A_n)\right) \\
&\leq \left(4 + 2^{-i+2}\right)\sqrt{m'n'}\left(2^{-i+2}\sqrt{m'n'} + \|t_{i-1} - t_i\|\right) \\
&\leq \left(4 + 2^{-i+2}\right)\sqrt{m'n'}\left(2^{-i+2}\sqrt{m'n'} + 2^{-i}\sqrt{m'n'}\right) = 10 \cdot 2^{-i+1}\left(1 + 2^{-i}\right)m'n'
\end{aligned}$$

So for fixed i the inner loop starts with $a = x_{i-1}$ and improves this approximation until $\|t_i - a_l\| = \|t_i - x_i\|$. So we get the following

$$\|t_i - x_{i-1}\|^2 = \|t_i - a\|^2 < \|t_i - a_1\|^2 < \ldots < \|t_i - a_l\|^2 = \|t_i - x_i\|^2$$

and because $\|t_i - x_{i-1}\|^2 - \|t_i - x_i\|^2 \leq 10 \cdot 2^{-i+1}\left(1 + 2^{-i}\right)m'n'$ and in every iteration this decreases with at least 2^{-i+1} there can be at most $10 \cdot \left(1 + 2^{-i}\right)m'n' + 1$ iterations ($+1$ for the final check) for every $i \geq 1$. So given a closest vector x_{i-1} to t_{i-1} we can find a closest vector x_i to t_i in $O(mn)$ iterations. By Lemma 29 each iteration takes $O(mn(m+n))$ operations. So in total we need $O((mn)^2(m+n))$ operations to go from x_{i-1} to x_i for $i \geq 1$. So given x_0 we can find x_d in $O(d \cdot (mn)^2(m+n))$ operations. By Lemma 30 we can find an $a \in A_m \otimes A_n$ such that $\|t_0 - a\|^2 \leq 4m'n'$ and thus

$$\|t_0 - a\|^2 - \|t_0 - x_0\|^2 \leq 4m'n'$$

and as this difference decreases with at least $2^{-0+1} = 2$ every iteration the number of iterations to obtain x_0 from the first approximation is also in $O(mn)$ and thus the total number of operations to find x_0 is in $O((mn)^2(m+n))$. This changes nothing to the total complexity and thus we can find a closest vector to $t_d = t$ in $A_m \otimes A_n$ in $O(d \cdot (mn)^2(m+n))$ operations. □

Note that the used technique of turning a polynomial time algorithm for finding, if it exists, a Voronoi relevant vector that improves the current approximation into a polynomial CVP algorithm can be used for any lattice type with only integer vectors for which the covering radius is polynomially bounded in the rank. Such lattices with basis in $\mathbb{Q}^n$ that can be scaled by a polynomial factor to be in $\mathbb{Z}^n$ of course also qualify.

For all practical purposes this algorithm gives a polynomial algorithm for solving CVP in $A_m \otimes A_n$. For theoretic purposes we can find a really good approximation in polynomial time.

Corollary 33. *Given a target $t = \sum_{i,j} a_{ij} b^{ij} \in \operatorname{span}(A_m \otimes A_n)$ and a closest vector $c = A_m \otimes A_n$ to t and $d \geq 1$ we can find an $x \in A_m \otimes A_n$ such that $\|t - x\| - \|t - c\| \leq 2^{-d+2}\sqrt{m'n'}$ in $O(d \cdot (mn)^2(m+n))$ operations.*

Proof. Use Algorithm 31 with input $t' = \sum_{i,j} 2^{-d} \cdot \lfloor 2^d \cdot a_{ij} \rceil b^{ij}$ and let x be a closest vector to t' as returned by the algorithm. Let $e = t - t'$, then $\|e\| \leq 2^{-d+1}\sqrt{m'n'}$.

We now have that

$$\|t - x\| - \|t - c\| \leq \left\|t' - x\right\| + \|e\| - \|t - c\| + \|e\| \leq 2\|e\| \leq 2^{-d+2}\sqrt{m'n'}$$

and the result follows from Theorem 32. □

It isn't clear if with some alterations it is possible to bound the number of iterations when going directly to the given target instead of this successive rounding technique. There exist ways to obtain simple negative weight cycles in a graph which weight is guaranteed to be in some factor of the most negative weighted cycle (there is a polynomial algorithm that finds the minimal mean weight cycle, where the weight of a cycle is divided by its length). Maybe this could help to give some lower bound on the improvement made in each iteration such that we can bound the number of iterations when going directly to the target.

So for $n = pq$ with p and q prime we have found a polynomial algorithm for the dual lattice $L_n^* = L_p^* \otimes L_q^* = A_{p-1} \otimes A_{q-1}$ of L_n. Note that, just as for L_n itself, with the use of Lemma 18 this polynomial algorithm extends trivially to a polynomial algorithm for the case $n = p^k q^l$.

8. Conclusions and further work

We have shown that every cyclotomic lattice can be constructed by direct sums and tensor products from the lattices A_n^* ($n \geq 1$). For the prime power cases this resulted in a linear CVP algorithm for the cyclotomic lattice and an almost linear algorithm for its dual. For the composite case $n = p \cdot q$ with p and q prime the cyclotomic lattice became a lot more complex and we were able to construct a sub-exponential CVP algorithm. For its dual we were able to construct a polynomial CVP algorithm. Furthermore these algorithms extend trivially to the case $n = p^k q^l$.

Unfortunately the polynomial CVP algorithm for $A_m \otimes A_n$ doesn't seem to extend trivially to a polynomial CVP algorithm for general $\bigotimes_{i=1}^{k} A_{n_i}$ when $k > 2$. It seems however that it wouldn't be too hard to characterize the Voronoi relevant vectors of this more general lattice and an open problem is if this could again result in a polynomial CVP algorithm.

It isn't hard to see that the technique used to construct a sub-exponential CVP algorithm for the lattice $A_m^* \otimes A_n^*$ can be extended inductively to the more general lattice $\bigotimes_{i=1}^{k} A_{n_i}^*$. Further research could explore the resulting complexity of this. It is still an open problem if there exists a polynomial CVP algorithm for the lattice $A_m^* \otimes A_n^*$ and thus for the non power of prime cases of the cyclotomic lattice. An interesting start and result on its own would be to characterize the Voronoi relevant vectors of $A_m^* \otimes A_n^*$.

References

[1] 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.*, vol. 71, pp. 55–61, July 1999.

[2] D. Micciancio, "Efficient reductions among lattice problems," in *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, (Philadelphia, PA, USA), pp. 84–93, Society for Industrial and Applied Mathematics, 2008.

[3] P. van Emde Boas, "Another NP-complete problem and the complexity of computing short vectors in a lattice," *Technical Report 81-04*, 1981.

[4] M. Ajtai, "The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract)," in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, (New York, NY, USA), pp. 10–19, ACM, 1998.

[5] J. Conway and N. Sloane, *Sphere Packings, Lattices and Groups*. Grundlehren der mathematischen Wissenschaften, Springer New York, 1998.

[6] M. O. Damen, H. E. Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Transactions on Information Theory*, vol. 49, pp. 2389–2402, Oct 2003.

[7] O. Goldreich, S. Goldwasser, and S. Halevi, "Public-key cryptosystems from lattice reduction problems," in *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '97, (London, UK, UK), pp. 112–131, Springer-Verlag, 1997.

[8] O. Goldreich, S. Goldwasser, and S. Halevi, *Public-key cryptosystems from lattice reduction problems*, pp. 112–131. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997.

[9] V. Lyubashevsky, C. Peikert, and O. Regev, *On Ideal Lattices and Learning with Errors over Rings*, pp. 1–23. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[10] V. Lyubashevsky, C. Peikert, and O. Regev, *A Toolkit for Ring-LWE Cryptography*, pp. 35–54. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[11] D. Micciancio, "Inapproximability of the shortest vector problem: Toward a deterministic reduction," *Theory of Computing*, 2012.

[12] I. Dinur, G. Kindler, R. Raz, and S. Safra, "Approximating cvp to within almost-polynomial factors is NP-hard," *Combinatorica*, vol. 23, no. 2, pp. 205–243, 2003.

[13] D. Micciancio, "The hardness of the closest vector problem with preprocessing," *IEEE Trans. Inf. Theor.*, vol. 47, pp. 1212–1215, Sept. 2006.

[14] I. Haviv and O. Regev, "Tensor-based hardness of the shortest vector problem to within almost polynomial factors," *Theory of Computing*, vol. 8, no. 23, pp. 513–531, 2012.

[15] O. Regev, "Lattices in computer science, lecture 8, dual lattices." University Lecture, 2004.

[16] P. Stevenhagen, "Algebra III." University Lecture Notes, 2010.

[17] R. G. McKilliam, I. V. L. Clarkson, W. D. Smith, and B. G. Quinn, "A linear-time nearest point algorithm for the lattice A_n^*," in *Information Theory and Its Applications, 2008. ISITA 2008. International Symposium on*, pp. 1–5, Dec 2008.

[18] J. Conway and N. Sloane, "Fast quantizing and decoding and algorithms for lattice quantizers and codes," *IEEE Transactions on Information Theory*, vol. 28, pp. 227–232, Mar 1982.

[19] R. G. McKilliam, I. V. L. Clarkson, and B. G. Quinn, "An algorithm to compute the nearest point in the lattice A_n^*," *CoRR*, 2008.

[20] D. Aggarwal, D. Dadush, and N. Stephens-Davidowitz, "Solving the closest vector problem in 2^n time - the discrete gaussian strikes again!," *CoRR*, 2015.

[21] D. Micciancio and P. Voulgaris, "A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations," in *Proceedings of the Forty-second ACM Symposium on Theory of Computing*, STOC '10, (New York, NY, USA), pp. 351–358, ACM, 2010.

[22] N. Bonifas and D. Dadush, "Short paths on the voronoi graph and the closest vector problem with preprocessing," *CoRR*, 2014.

[23] G. Voronoi, "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les parallélloèdres primitifs.," *Journal für die reine und angewandte Mathematik*, vol. 134, pp. 198–287, 1908.

[24] H. Minkowski, *Gesammelte Abhandlungen*, vol. 2. 1911.

[25] N. Sommer, M. Feder, and O. Shalvi, "Finding the closest lattice point by iterative slicing," in *2007 IEEE International Symposium on Information Theory*, pp. 206–210, June 2007.

[26] R. G. McKilliam, A. J. Grant, and I. V. L. Clarkson, "Finding a closest point in a lattice of voronoi's first kind," *CoRR*, 2014.

[27] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd ed., 2009.

[28] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.