Deciding Orthogonality in Construction-A Lattices

Karthekeyan Chandrasekaran [1], Venkata Gandikota[2], and Elena Grigorescu[3]

1 **University of Illinois, Urbana-Champaign, IL, USA**
karthe@illinois.edu
2 **Purdue University, West Lafayette, IN, USA**
vgandiko@purdue.edu
3 **Purdue University, West Lafayette, IN, USA**
elena-g@purdue.edu*

Abstract

Lattices are discrete mathematical objects with widespread applications to integer programs as well as modern cryptography. A fundamental problem in both domains is the Closest Vector Problem (popularly known as CVP). It is well-known that CVP can be easily solved in lattices that have an orthogonal basis *if* the orthogonal basis is specified. This motivates the orthogonality decision problem: verify whether a given lattice has an orthogonal basis. Surprisingly, the orthogonality decision problem is not known to be either NP-complete or in P.

In this paper, we focus on the orthogonality decision problem for a well-known family of lattices, namely Construction-A lattices. These are lattices of the form $C + q\mathbb{Z}^n$, where C is an error-correcting q-ary code, and are studied in communication settings. We provide a complete characterization of lattices obtained from binary and ternary codes using Construction-A that have an orthogonal basis. This characterization leads to an efficient algorithm solving the orthogonality decision problem, which also finds an orthogonal basis if one exists for this family of lattices. We believe that these results could provide a better understanding of the complexity of the orthogonality decision problem in general.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Orthogonal Lattices, Construction-A, Orthogonal Decomposition, Lattice isomorphism

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2015.151

1 Introduction

A lattice is the set of integer linear combinations of a set of basis vectors $B \in \mathbb{R}^{m \times n}$, namely $L = L(B) = \{xB \mid x \in \mathbb{Z}^m\}$. Lattices are well-studied fundamental mathematical objects that have been used to model diverse discrete structures such as in the area of integer programming [7], or in factoring integers [14] and factoring rational polynomials [8]. In a groundbreaking result, Ajtai [1] demonstrated the potential of computational problems on lattices to cryptography, by showing average case/worst case equivalence between lattice problems related to finding short vectors in a lattice. This led to renewed interest in the complexity of two fundamental lattice problems: the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). Concretely, in SVP, given a basis B one is asked to output a shortest non-zero vector in the lattice, and in CVP, given a basis B and a target $t \in \mathbb{R}^n$, one is asked to output a lattice vector closest to t.

* The research of V. G. and of E. G. was partially funded by Purdue Research Foundation grants.

35th IARCS Annual Conf. Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2015).
Editors: Prahladh Harsha and G. Ramalingam; pp. 151–162

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Both SVP and CVP are NP-hard even to approximate up to subpolynomial factors (see [12] for a survey), and a great deal of research in complexity theory has been devoted to finding families of lattices for which SVP/CVP are easy. A simplest lattice for which CVP is easy is $\mathbb{Z}^n$: indeed, finding the closest lattice vector to a target $t \in \mathbb{R}^n$ amounts to rounding the entries of t to the nearest integer. Surprisingly, given an arbitrary basis B, it is not known how to efficiently verify whether the lattice generated by B is isomorphic to $\mathbb{Z}^n$ upto an orthogonal transformation. Further, given an arbitrary basis for a lattice, it is not known how to decide efficiently if the lattice has an orthogonal basis (an orthogonal basis is a basis in which all vectors are pairwise orthogonal). Similar to the case of $\mathbb{Z}^n$, having access to an orthogonal basis leads to an efficient algorithm to solve CVP, but finding an orthogonal basis given an arbitrary basis appears to be non-trivial, with no known efficient algorithms.

Deciding if a lattice is equivalent to $\mathbb{Z}^n$, and deciding if a lattice has an orthogonal basis, are special cases of the more general Lattice Isomorphism Problem (LIP). In LIP, given lattices L_1 and L_2 presented by their bases, one is asked to decide if they are isomorphic, meaning if there exists an orthogonal transformation that takes one to the other. LIP has been studied in [13, 15, 6] and is known to have a $n^{O(n)}$ algorithm [6]. Recent results of [10, 9] show that in certain highly symmetric lattices, isomorphism to $\mathbb{Z}^n$ can be decided efficiently.

The complexity of LIP is not well understood, and is part of the broader study of isomorphism between mathematical objects, of which Graph Isomorphism (GI) is a well-known elusive problem [2]. Interestingly, there is a polynomial time reduction from GI to LIP [15].

Given that LIP, deciding isomorphism to $\mathbb{Z}^n$, and deciding whether a lattice has an orthogonal basis appear to be difficult problems for arbitrary input lattices, it is natural to address families of lattices where these problems are solvable efficiently. In this work, we focus on the problem of deciding orthogonality for a particular family of lattices, commonly known as Construction-A lattices [5]. A Construction-A lattice L is obtained from a linear error-correcting code C over a finite field of q elements[1] (denoted $\mathbb{F}_q$) as $L = C + q\mathbb{Z}^n$. We resolve the problem of deciding orthogonality in Construction-A lattices for $q = 2$ and $q = 3$ showing an efficient algorithm. In addition, the algorithm outputs an orthogonal basis of the lattice if such a basis exists.

Our main technical contribution is a decomposition theorem for Construction-A lattices that admit an orthogonal basis. A natural way to obtain an orthogonal Construction-A lattice is by taking direct products of lower dimensional orthogonal lattices. We show that this is the only possible way and that the lower dimensional orthogonal lattices indeed have constant dimension. We believe that our contributions are a step towards gaining a better understanding of lattice isomorphism problems for more general classes of lattices.

Extending our results to values $q > 3$ might require new techniques. For higher q, a decomposition characterization seems to require a complete characterization of *weighing matrices* of weight q which is a known open problem. In particular, a direct product decomposition characterization of weighing matrices for the case of $q = 4$ is known. However, the parts in the direct product decomposition may not be of constant dimension. As a consequence, the lattice decomposition theorem, if true, would only suggest that orthogonal Construction-A lattices necessarily decompose into direct products of lattices, which could be high-dimensional. So designing an efficient algorithm for the orthogonality decision problem exploiting the direct product decomposition characterization appears to be non-trivial.

[1] The term 'Construction-A' strictly refers to the case $q = 2$, but we will not make the distinction in this paper.

1.1 Our results and techniques

As mentioned above, we start by showing a structural decomposition of orthogonal lattices of the form $C + 2\mathbb{Z}^n$ and $C + 3\mathbb{Z}^n$ into constant-size orthogonal lattices. We remark that the decomposition holds up to permutations of the coordinates, and we use the notation $C_1 \cong C_2$ and $L_1 \cong L_2$ to denote the equivalence of codes and lattices under permutation of coordinates. We use the notation $L_1 \otimes L_2$ to denote the direct product of two lattices.

▶ **Theorem 1.** *Let $L_C = C + 2\mathbb{Z}^n$ be a lattice obtained from a binary linear code $C \subseteq \mathbb{F}_2^n$. Then the following statements are equivalent:*

1. *L_C is orthogonal.*
2. *$L_C \cong \otimes_i L_i$, where each L_i is either $\mathbb{Z}$, or $2\mathbb{Z}$, or the 2-dimensional lattice generated by the rows of the matrix $\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$.*
3. *$C \cong \otimes_i C_i$, where each C_i is either a length-1 binary linear code $\subseteq \{0, 1\}$, or the length-2 binary linear code $\{00, 11\}$.*

The decomposition characterization leads to an efficient algorithm to verify if a given lattice obtained from a binary linear code using Construction-A is orthogonal. For the purposes of this algorithmic problem, the input consists of a basis to the lattice. The algorithm finds the component codes given by the characterization thereby computing the orthogonal basis for such a lattice.

▶ **Theorem 2.** *Given a basis for a lattice L obtained from a binary linear code $C \subseteq \mathbb{F}_2^n$ using Construction-A, there exists an algorithm running in time $O(n^6)$ that verifies if L is orthogonal, and if so, it outputs an orthogonal basis.*

We obtain a similar decomposition and algorithm for lattices obtained from ternary codes. For succinctness of presentation we define the following integer matrix:

$$M = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & 0 & -1 & -1 \\ 0 & 1 & -1 & 1 \end{bmatrix}.$$

▶ **Theorem 3.** *Let $L_C = C + 3\mathbb{Z}^n$ be a lattice obtained from a ternary linear code $C \subseteq \mathbb{F}_3^n$. Then the following statements are equivalent:*

1. *L_C is orthogonal.*
2. *$L_C \cong \otimes_i L_i$, where each L_i is either $\mathbb{Z}$, or $3\mathbb{Z}$, or the 4-dimensional lattice generated by the rows of a matrix $\mathcal{T}(M)$ obtained from M by negating some subset of columns.*
3. *$C \cong \otimes_i C_i$, where each C_i is either a linear length-1 ternary code, or the linear length-4 ternary code generated by the rows of $(\mathcal{T}(M) \bmod 3) \in \mathbb{F}_3^{4\times 4}$, where $\mathcal{T}(M)$ is obtained from M by negating some subset of its columns.*

▶ **Theorem 4.** *Given a basis for a lattice L obtained from a ternary linear code using Construction-A, there exists an algorithm running in time $O(n^8)$ that verifies if L is orthogonal, and if so, it outputs an orthogonal basis.*

In the interests of space, we prove Theorems 3 and 4 here and defer the proofs of Theorems 1 and 2 to the full version of this work [4].

2 Preliminaries

We denote by $[n]$ the set of positive integers up to n, the $n \times n$ identity matrix by I_n and its j^{th} row by e_j. For a vector $b \in \mathbb{R}^n$, let b_j denote its j^{th} coordinate, and $\|b\|$ denote its ℓ_2 norm.

A lattice $L \subseteq \mathbb{R}^n$ is said to be of full rank if it is generated by n linearly independent vectors. A lattice L is said to be orthogonal if it has a basis B such that the rows of B are pairwise orthogonal vectors. A lattice L is *integral* if it is contained in $\mathbb{Z}^n$, namely any basis for L only consists of integer vectors.

We will denote by $\mathbb{F}_q$ a finite field with q elements. A *linear* code C of length n over $\mathbb{F}_q$ is a vectorspace $C \subseteq \mathbb{F}_q^n$. A linear code is specified by a generator matrix G that consists of linearly independent vectors in $\mathbb{F}_q^n$. If $C \subseteq \mathbb{F}_2^n$ it is called a *binary* code, and if $C \subseteq \mathbb{F}_3^n$ it is called a *ternary* code.

The Construction-A of a lattice L_C from a linear code $C \subseteq \mathbb{F}_q^n$, where q is a prime, is defined as $L_C := \{c + q \cdot z \mid c \in \phi(C), z \in \mathbb{Z}^n\}$, where ϕ is the the (real embedding) mapping $i \in \mathbb{F}_q \mapsto i \in \mathbb{Z}$. Construction-A is often abbreviated as $L_C = C + q\mathbb{Z}^n$.

For any vector $v = (v_1, \cdots, v_n) \in \mathbb{Z}^n$ define $v \bmod q = (v_1 \bmod q, \cdots, v_n \bmod q) \in \mathbb{F}_q^n$.

▶ **Claim 5.** *Let q be a prime. If $q\mathbb{Z}^n \subseteq L$ then $C = L \bmod q$ is a linear code over $\mathbb{F}_q$.*

Proof. Let $v \in L$ and $v = (v \bmod q) + qz$ for some $z \in \mathbb{Z}^n$, where here we abuse notation and view $v \bmod q$ as embedded into the integers, instead of a vector in $\mathbb{F}_q^n$. Since $q\mathbb{Z}^n \subseteq L$, it follows that $v - qz = v \bmod q \in L$. To show that $C = L \bmod q$ is a linear code over $\mathbb{F}_q$, let $c_1, c_2 \in C$. Then $c_1 + c_2 \in L$ (where the addition is over $\mathbb{Z}$), and so $(c_1 + c_2) \bmod q \in C$. ◀

We will use the following immediate claim about product of lattices generated from codes.

▶ **Claim 6.** *Let $L = C + q\mathbb{Z}^n$, for some q-ary linear code $C \subseteq \mathbb{F}_q^n$. If $L \cong L_1 \otimes L_2$, and $L_1 \subseteq \mathbb{Z}^k$, then $L_1 \cong C_1 + q\mathbb{Z}^k$ and $L_2 \cong C_2 + q\mathbb{Z}^{n-k}$, for q-ary linear codes C_1 and C_2 that are projections of C on the coordinates corresponding to L_1 and L_2 respectively.*

A matrix U is *unimodular* if $U \in \mathbb{Z}^{n \times n}$ and $\det(U) \in \{\pm 1\}$. Two different bases B_1, B_2 give rise to the same lattice if and only if there exists a unimodular matrix U such that $B_1 = UB_2$.

The *Hermite Normal Form (HNF) basis* for a full rank lattice $L \subseteq \mathbb{R}^n$ is a square, non-singular, upper triangular matrix $B \subseteq \mathbb{R}^{n \times n}$ such that off-diagonal elements satisfy : $0 \leq b_{i,j} < b_{j,j}$ for all $1 \leq i < j \leq n$.

▶ **Fact 7.** *[11] There exists an efficient algorithm which on input a set of rational vectors B, computes a basis for the lattice generated by B: the algorithm simply computes the unique HNF basis of the lattice generated by B.*

We note that $L_C = C + q\mathbb{Z}^n$ contains $q\mathbb{Z}^n$ as a sublattice and hence it is a full rank lattice.

▶ **Fact 8.** *A basis B for the lattice L_C specified by the generator matrix G for the code C can be computed efficiently by taking the HNF of the matrix $\begin{bmatrix} G \\ qI_n \end{bmatrix}$. Conversely, given a basis B of L_C, the generator matrix for C can be computed efficiently by finding a basis for $B \bmod q$ by row reduction over $\mathbb{F}_q$.*

A *weighing matrix* of order n and weight k is a $n \times n$ matrix with entries in $\{0, 1, -1\}$ such that each row and column has exactly k non-zero entries and the row vectors are orthogonal to each other. By definition, a weighing matrix W satisfies $WW^T = kI_n$. For matrices $A \in \mathbb{R}^{n_1 \times n_1}$ and $B \in \mathbb{R}^{n_2 \times n_2}$, we denote the $(n_1 + n_2) \times (n_1 + n_2)$-dimensional block-diagonal matrix obtained using blocks A and B by $A \otimes B$. We will use the following characterization of weighing matrices of weight 2 and 3. Please refer to the full version [4] for the proofs of Theorem 9 and Theorem 10.

▶ **Theorem 9** ([3]). *A matrix W is a weighing matrix of order n and weight 2 if and only if W can be obtained from*

$$\otimes_{i=1}^{n/2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

by negating some rows and columns and by interchanging some rows and columns.

▶ **Theorem 10** ([3]). *A matrix W is a weighing matrix of order n and weight 3 if and only if W can be obtained from $\otimes_{i=1}^{n/4} M$ by negating some rows and columns and by interchanging some rows and columns.*

3 Orthogonal Lattices from Ternary Codes

In this section we focus on lattices obtained from ternary linear codes using Construction-A. In Section 3.1, we show that any orthogonal lattice obtained from a ternary linear codes by Construction-A is equivalent to a product lattice whose components are one-dimensional or four-dimensional. In Section 3.2, we show that given a lattice obtained from a ternary linear code by Construction-A, there exists an efficient algorithm to verify if the lattice is orthogonal.

3.1 Decomposition Characterization

We prove Theorem 3 in this subsection.

Proof of Theorem 3. We show that (1) $\equiv$ (2) and (2) $\equiv$ (3) to complete the equivalence of the three statements.

(1) $\equiv$ (2): We show that $L_C = C + 3\mathbb{Z}^n$ is orthogonal if and only if it decomposes into direct product of lower dimensional orthogonal lattices, $L_C \cong \otimes_i L_i$.
If $L_C \cong \otimes_i L_i$ such that each L_i is orthogonal, then L_C is also orthogonal, since L_C has a block diagonal basis where each block is itself an orthogonal matrix (by definition, a 1×1-dimensional matrix is orthogonal) .

We prove the other side by induction on the dimension, n of the lattice L_C. For the base case consider $n = 1$. Since L is integral, contains $3\mathbb{Z}$ and is of the form $C + 3\mathbb{Z}$ for some ternary code C, it follows that L has to be either $\mathbb{Z}$ or $3\mathbb{Z}$. Let us assume the induction hypothesis for all $n - 1$ or lower dimensional orthogonal lattices obtained from ternary linear codes using construction-A.

Let L_C be an n-dimensional orthogonal lattice and B be its orthogonal basis. Since L_C is an integral lattice, B has only integral entries. The next two claims summarize certain properties of the entries of the basis matrix B.

▶ **Claim 11.** *For every row b of B and for every $j \in [n]$, we have that $3|b_j| \in \{0, \|b\|^2, 3\|b\|^2\}$.*

Proof. Since B is an orthogonal basis, $BB^T = D$, where D is the diagonal matrix with $d_i = \|b^{(i)}\|^2$, where $b^{(i)}$ denotes the i^{th} basis vector.

$$D = \begin{bmatrix} \|b^{(1)}\|^2 & 0 & 0 & \cdots & 0 \\ 0 & \|b^{(2)}\|^2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & 0 & \cdots & \|b^{(n)}\|^2 \end{bmatrix}$$

We know that $3\mathbb{Z}^n \subseteq L_C$ so, $3e_j \in L_C$ for every $j \in [n]$. Therefore, there is an integral matrix $X \in \mathbb{Z}^{n\times n}$ such that $XB = 3I_n$, i.e. $3B^{-1} \in \mathbb{Z}^{n\times n}$. Since we started with an orthogonal basis B,

$$B^{-1} = B^T D^{-1} \in \frac{1}{3}\mathbb{Z}^{n\times n}.$$

Each column of $B^T D^{-1}$ is given by $b/\|b\|^2$, where b is a basis vector. Therefore, for any $j \in [n], 3b_j$ is a multiple of $\|b\|^2$, formally

$$3b_j \equiv 0 \bmod \|b\|^2 \text{ for all } j \in [n], \text{ and rows } b \text{ of } B. \tag{1}$$

Since b_j is integral and $|b_j| \leq \|b\|^2$ for every $j \in [n]$, it follows from the above equation that $3|b_j| \in \{0, \|b\|^2, 2\|b\|^2, 3\|b\|^2\}$. Suppose there exists $j \in [n]$ such that $3|b_j| = 2\|b\|^2$. Since b is a basis vector, it follows that b is not all zeroes. Hence $b_j \neq 0$. We can re-write the condition $3|b_j| = 2\|b\|^2$ as $3|b_j| = 2\sum_{i=1}^n b_i^2$. Rearranging the terms, we have

$$|b_j|\ (3 - 2|b_j|) = 2\sum_{i\neq j} b_i^2.$$

Since the RHS is a sum of squares, it is always non-negative. The LHS is non-zero since $b_j \in \mathbb{Z} \setminus \{0\}$. So the LHS should be strictly positive. Therefore, $|b_j| \in (0, 3/2) \cap \mathbb{Z}$ and hence $|b_j| = 1$. However, this implies that $\sum_{i\neq j} b_i^2 = 1/2$, contradicting the fact that b is integral. Hence, $3||b_j|| = 2\|b\|^2$ is impossible. ◀

▶ **Claim 12.** *Let b be a row of B.*

1. *If there exists $j \in [n]$ such that $3|b_j| = 3\|b\|^2$, then $b_j = \pm 1$ and $b_{j'} = 0$ for every $j' \in [n] \setminus \{j\}$.*
2. *If there exists $j \in [n]$ such that $3|b_j| = \|b\|^2$ and $b_j = \pm 3$, then $b_{j'} = 0$ for every $j' \in [n] \setminus \{j\}$.*
3. *If there exists $j \in [n]$ such that $3|b_j| = \|b\|^2$ and $b_j = \pm 1$, then there exist $j_1, j_2 \in [n] \setminus \{j\}$, such that $|b_{j_1}| = |b_{j_2}| = 1$ and $b_{j'} = 0$ for every $j' \in [n] \setminus \{j, j_1, j_2\}$.*
4. *If there exists $j \in [n]$ such that $3|b_j| = \|b\|^2$, then $b'_j \in \{0, \pm 1, \pm 3\}$ for every $j' \in [n]$.*

Proof.

1. Since, $\|b\|^2 = \sum_{i=1}^n b_i^2$, and each $b_i \in \mathbb{Z}$, we conclude that $|b_j| = 1$ and the remaining coordinates in b have to be 0, i.e $b_{j'} = 0$ for all $j' \in [n] \setminus \{j\}$.
2. Follows from $3|b_j| = \|b\|^2$ and b being integral.
3. We can re-write the condition $3|b_j| = \|b\|^2$ as $3|b_j| = \sum_{i=1}^n b_i^2$. Rearranging the terms, we have

$$|b_j|\ (3 - |b_j|) = \sum_{i\neq j} b_i^2. \tag{2}$$

If $b_j = \pm 1$, then $\sum_{i\neq j} b_i^2 = 2$. Further, b is integral. Hence, b has exactly 2 other non-zero coordinates b_{j_1}, b_{j_2}, $j \neq j_1, j_2$, such that $|b_{j_1}| = |b_{j_2}| = 1$.

4. We have equation (2). The RHS is a sum of squares and hence the LHS is non-negative. Moreover, b is not all-zeroes vector implies that $b_j \neq 0$. Therefore, $|b_j| \in (0,3] \cap \mathbb{Z}$. If $b_j = \pm 2$, then in order to satisfy $\sum_{i \neq j} b_i^2 = 2$ using integral b_i's, exactly two coordinates b_{j_1}, b_{j_2} should be ± 1, where $j \neq j_1, j_2$. However, in this case, $3|b_{j_1}| = 3|b_{j_2}| = 3 \notin \{0, \|b\|^2 = 6, 3\|b\|^2 = 18\}$, thus contradicting Claim 11. The conclusion follows from parts (2) and (3). ◀

Using the properties of the orthogonal basis B of L_C given in Claims 11 and 12, we show that B is equivalent (up to permutations of its columns) to a block diagonal matrix, i.e

$$B \cong \begin{bmatrix} B_1 & 0 & \cdots & 0 \\ 0 & B_2 & \cdots & 0 \\ \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & B_k \end{bmatrix}$$

where each B_i is either the 1×1 matrix $[1]$ or the 1×1 matrix $[3]$ or the 4×4 matrix obtained from M by negating a subset of its columns, $\mathcal{T}(M)$. It follows that $L_C \cong \otimes_i L_i$ such that B_i is the basis for the lower dimensional lattice L_i.

Let us pick a row b of B with the smallest support. Fix an index $j \in [n]$ to be the index of a non-zero entry with minimum absolute value in b, i.e. $j = \arg\min_k \{|b_k|\}$. As b is a row of a basis matrix, b cannot be the all-zeroes vector and therefore there exists a $j \in [n]$ such that $|b_j| > 0$. Since we are only interested in equivalence (that allows for permutation of coordinates), we may assume without loss of generality that $j = 1$ by permuting the coordinates. By Claim 11, we have that $3|b_1| \in \{\|b\|^2, 3\|b\|^2\}$. We consider each of these cases separately.

1. Suppose $3|b_1| = 3\|b\|^2$. By Claim 12(1), $b = (\pm 1, 0, \ldots, 0)$. Since B is an orthogonal basis, $\langle b, b' \rangle = 0 \Rightarrow b'_1 = 0$ for all $b' \neq b \in B$. The orthogonality of B therefore forces all other basis vectors to take a value of 0 at the 1^{st} coordinate. Thus B is of the form

$$B = \left(\begin{array}{c|c} \pm 1 & 0 \cdots 0 \\ \hline 0 & \\ \vdots & B' \\ 0 & \end{array}\right).$$

   Therefore, we obtain $L_C \cong \mathbb{Z} \otimes L'$, where L' is an orthogonal $(n-1)$-dimensional lattice generated by the basis matrix restricted to the coordinates other than 1, say, B'. From Claim 6, it follows that $L' = C' + 3\mathbb{Z}^{n-1}$ for some ternary linear code $C' \subseteq \mathbb{F}_3^{n-1}$. Thus L' satisfies the induction hypothesis and we have the desired decomposition.
2. Suppose $3|b_1| = \|b\|^2$. We can re-write this condition as $3|b_1| = \sum_{i=1}^n b_i^2$. Rearranging the terms, we have

$$|b_1| \ (3 - |b_1|) = \sum_{i \neq 1} b_i^2.$$

   Since the RHS is a sum of squares, it should be non-negative.

   (i) If RHS is 0, then $b_1 = \pm 3$ and therefore, it follows from Claim 12(2) that $b = (\pm 3, 0, \ldots, 0)$. The orthogonality of B forces all other basis vectors to take a value of

0 at the 1^{st} coordinate.

$$B = \left(\begin{array}{c|c} \pm 3 & 0 \cdots 0 \\ \hline 0 & \\ \vdots & B' \\ 0 & \end{array}\right)$$

Therefore, we obtain $L_C \cong 3\mathbb{Z} \otimes L'$, where L' is an orthogonal $(n-1)$-dimensional lattice generated by the basis matrix restricted to the coordinates other than 1, say B'. From Claim 6, it follows that $L' = C' + 3\mathbb{Z}^{n-1}$ for some ternary linear code $C' \subseteq \mathbb{F}_3^{n-1}$. Thus L' satisfies the induction hypothesis and we have the desired decomposition.

(ii) If RHS is strictly positive, then $|b_1| \in (0,3) \cap \mathbb{Z} = \{1,2\}$. By Claim 12(4), $b_1 \neq \pm 2$. Therefore, $b_1 = \pm 1$. By Claim 12(3), we have that b has exactly three non-zero coordinates and they are ± 1. By permuting the coordinates of B, we can write $b \equiv (\pm 1, \pm 1, \pm 1, 0, \cdots, 0)$.

Since we picked the row b to be the one with the smallest support, it follows that every row has at least 3 non-zero coordinates. By Claims 11 and 12(1), this is possible only if for every other row b', there exists $j' \in [n]$ such that $3|b'_{j'}| = \|b'\|^2$. By Claim 12(4), every other row b' has all its coordinates in $\{0, \pm 1, \pm 3\}$. By Claim 12(2), every other row b' has none of its coordinates in $\{\pm 3\}$. Therefore, every other row b' has all its coordinates in $\{0, \pm 1\}$. By Claim 12(3), every row of the basis matrix has the same form as b: they have exactly three non-zero entries each of which is ± 1.

Since the rows of the basis matrix are orthogonal, it follows that the basis matrix B is a weighing matrix of order n with weight 3. By Theorem 10, B is obtained from $\otimes_{n/4} M$ by either negating some rows or columns and by interchanging rows or columns. We recall that interchanging or negating the rows of the basis matrix of a lattice preserves the basis property while interchanging columns is equivalent to permuting the coordinates. Hence $L_C = L(B) \cong \otimes_{i=1}^{n/4} L(\mathcal{T}_i(M))$, where each $\mathcal{T}_i(M)$ is a 4×4 matrix obtained by negating a subset of columns of M.

(2) $\equiv$ (3): We now show that L_C decomposes into direct product of lower dimensional lattices, $L_C \cong \otimes_i L_i$ if and only if the code C also decomposes, $C \cong \otimes_i C_i$.

Let $L_C \cong \otimes_i L_i$. Without loss of generality, we can consider $L_C = \otimes_i L_i$. We have $C = L_C \bmod 3 = \otimes_i L_i \bmod 3$. We observe that if L_i has dimension n_i, then $L_i \supseteq 3\mathbb{Z}^{n_i}$. Therefore, $C_i = L_i \bmod 3$ is a ternary code. Let $C_i := L_i \bmod 3$ for every i. Then $C = \otimes_i C_i$. (If $c \in C$, then $c \in L$ and hence the projection of c to the subset of coordinates corresponding to L_i is in C_i. Let $c_i \in C_i$ for every i. The concatenated vector $\otimes_i c_i$ is in $\otimes_i L_i \mod 3$ and hence is in C.)

To show the other side, let $C \cong \otimes_i C_i$, where each $C_i \subseteq \mathbb{F}_3^{n_i}$ and $n = \sum_i n_i$. Therefore $L_C = C + 3\mathbb{Z}^n \cong \otimes_i C_i + 3\mathbb{Z}^n \cong \otimes_i (C_i + 3\mathbb{Z}^{n_i})$, since $\mathbb{Z}^n \cong \otimes_i \mathbb{Z}^{n_i}$. ◀

3.2 The algorithm

Theorem 3 shows that a lattice of the form $C + 3\mathbb{Z}^n$ is orthogonal if and only if the underlying code decomposes into direct product of ternary linear codes isomorphic to $\{0,1,2\}$ or $\{0\}$ or the four dimensional code generated by $\mathcal{T}(M) \bmod 3$, where $\mathcal{T}(M)$ is obtained from matrix M by negating a subset of its columns. We now give a polynomial time algorithm which finds the decomposition of the code C into the component codes, C_i, if there exists one.

Algorithm 1 decompose – length – 1(G):

Input: $G = \{g_1, \ldots, g_n\} \in \mathbb{F}_3^n$ (A generator for the code C)

1: **for** $j \in \{1, \cdots, n\}$ **do**
2: Let $G' \leftarrow$ projection of vectors in G on coordinates $[n] \setminus \{j\}$
3: For $g \in G'$, define $g^0, g^1, g^2 \in \mathbb{F}_3^n$ as the n-dimensional vectors obtained by extending g using 0, 1 and 2 along the j'th coordinate respectively.
4: **if** $g^0,\ g^1,\ g^2 \in C$ for all $g \in G'$ **then**
5: **return** j
6: **return** FAIL

Therefore, if the lattice L_C is orthogonal, the algorithm decides in polynomial time if it is orthogonal and also gives the orthogonal basis for the lattice.

The algorithm recursively finds the component codes. If it is unable to decompose the code at any stage, then it declares that L_C is not orthogonal. At every step we check if $C \cong \{0,1,2\} \times C'$ or $\{0\} \times C'$ or $C_{\mathcal{T}(M)} \times C'$ where $C_{\mathcal{T}(M)}$ is the code generated by $\mathcal{T}(M)$ mod 3 and then recurse on C'.

Proof of Theorem 4. Given a basis for L_C as input, we first compute the generator for C. From Theorem 3, we know that if L_C is orthogonal, then $C \cong \otimes_i C_i$ where each C_i is either the length-1 code $\{0,1,2\}$ or the length-1 code $\{0\}$ or a 4-dimensional code generated by the rows of $\mathcal{T}(M)$ mod 3 where $\mathcal{T}(M)$ obtained from matrix M by negating a subset of its columns.

The algorithm therefore in each step decides if $C \cong \{0,1,2\} \otimes C'$ or $C \cong \{0\} \otimes C'$ or $C \cong C_{\mathcal{T}(M)} \otimes C'$, where $C_{\mathcal{T}(M)}$ denotes the code generated by $\mathcal{T}(M)$ mod 3. Theorem 13 shows that using Algorithm 1 we can check in $O(n^4)$ time, if $C \cong \{0,1,2\} \otimes C'$. The same algorithm can be modified to check in $O(n^4)$ time, if $C \cong \{0\} \otimes C'$. Theorem 14 shows that Algorithm 2 can verify if $C \cong C_{\mathcal{T}(M)} \otimes C'$ in $O(n^7)$ time. If any one of the algorithms finds a decomposition, then we recurse in the lower dimensional code C' to find further decomposition. We recurse at most n times. If all the algorithms fail to find a decomposition, then L_C is not orthogonal. Therefore, it takes $O(n^8)$ time to decide if L_C is orthogonal. ◀

We now describe the individual algorithms to verify if $C \cong \{0,1,2\} \otimes C'$ or $C \cong \{0\} \otimes C'$ or $C \cong C_{\mathcal{T}(M)} \otimes C'$.

▶ **Theorem 13.** *Let C be a ternary linear code and $G = \{g_1, \ldots, g_n\} \in \mathbb{F}_3^{n \times n}$ be its generator. Then Algorithm 1 decides if $C \cong \{0,1,2\} \otimes C'$ for some linear code $C' \subseteq \mathbb{F}_3^{n-1}$ and if so outputs the coordinate corresponding to the direct product decomposition. Moreover the algorithm runs in time $O(n^4)$.*

Proof. For $j \in [n]$, let $C'_{\overline{j}} \subseteq \mathbb{F}_3^{n-1}$ be the projection of C on the indices $[n] \setminus \{j\}$ and for a vector $c \in C'_{\overline{j}}$, let $c^0, c^1, c^2 \in \mathbb{F}_3^n$ be extensions of c using $0, 1, 2$ respectively along the j'th coordinate. We note that $C \cong \{0,1,2\} \otimes C'$ for some ternary linear code C' if and only if there exists an index $j \in [n]$, such that

$$C = \left\{ c^0, c^1, c^2 \mid \ \forall\ c \in C'_{\overline{j}} \right\}. \tag{3}$$

From the definition of $C'_{\overline{j}}$, it follows that $C \subseteq \{c^0, c^1, c^2 \mid \ \forall\ c \in C'_{\overline{j}}\}$ up to a permutation of coordinates. So, the algorithm just needs to verify if the other side of the containment holds for some j.

Algorithm 2 decompose – length – 4(G):

Input: $G \in \mathbb{F}_3^{n \times n}$ (Generator for C)

for $j_1, j_2, j_3, j_4 \in \{1, 2, \cdots, n\}$ **do**
  Let $G' \leftarrow$ projection of vectors in G on coordinates $[n] \setminus \{j_1, j_2, j_3, j_4\}$
  Let $G'' \leftarrow$ projection of vectors in G on coordinates $\{j_1, j_2, j_3, j_4\}$
  for $S \subseteq [4]$ **do**
    Let $\mathcal{T}(M) \leftarrow M$ with columns in S negated
    if $C_{\mathcal{T}(M)} \equiv$ Code generated by G'' **then**
      For $g \in G'$ define $g^{p_1}, g^{p_2}, g^{p_3}, g^{p_4} \in \mathbb{F}_3^n$ be n-dimensional vectors obtained by extending g using the rows of $\mathcal{T}(M)$ along the j_1, j_2, j_3, j_4 coordinates.
      if $g^{p_1}, g^{p_2}, g^{p_3}, g^{p_4} \in C$ for all $g \in G'$ **then**
        return j_1, j_2, j_3, j_4 and $\mathcal{T}(M)$
return FAIL

Let G' be the set of vectors of G projected on the coordinates $[n] \setminus \{j\}$. Algorithm 1 verifies if g^0, g^1 and g^2 are codewords in C, for every vector $g \in G'$. We now show that this is sufficient. Since C is a code, if g^0, g^1, $g^2 \in C$ for every $g \in G'$, then all linear combinations of these vectors are also in C. Therefore, $\{c^0, c^1, c^2 \mid \forall\ c \in C'_{\bar{j}}\} \subseteq C$.

It takes $O(n^2)$ time to compute a parity check matrix from the generator G and $O(n^2)$ time to verify if an input vector is a codeword using the parity check matrix. For every possible choice of the index j, Algorithm 1 checks if each of the $3n$ vectors of the form g^0, g^1, g^2 are C. Therefore, Algorithm 1 takes $O(n^4)$ time to decide if $C \cong \{0, 1, 2\} \otimes C'$. ◀

▶ **Theorem 14.** *Let C be a ternary linear code and $G = \{g_1, \ldots, g_n\} \in \mathbb{F}_3^{n \times n}$ be its generator. For a matrix $\mathcal{T}(M)$ obtained by negating a subset of columns of M, let $C_{\mathcal{T}(M)}$ be the length-4 code whose generators are the rows of $\mathcal{T}(M)$. Then Algorithm 2 decides if $C \cong C_{\mathcal{T}(M)} \otimes C'$ for some linear codes $C' \subseteq \mathbb{F}_3^{n-4}$ and $C_{\mathcal{T}(M)} \subseteq \mathbb{F}_3^4$ and if so outputs the coordinates corresponding to the direct product decomposition as well as the matrix $\mathcal{T}(M)$. Moreover the algorithm runs in time $O(n^7)$.*

Proof. For $1 \leq j_1 < j_2 < j_3 < j_4 \leq n$, let C''_{j_1,j_2,j_3,j_4} be the projection of C on the indices $\{j_1, j_2, j_3, j_4\}$. We first verify if C''_{j_1,j_2,j_3,j_4} is the code generated by the rows of $\mathcal{T}(M)$ (denoted as $C_{\mathcal{T}(M)}$) for some $\mathcal{T}(M)$ which is obtained by negating a subset of columns of M. We would like to check if every $c \in C''_{j_1,j_2,j_3,j_4}$ is in $C_{\mathcal{T}(M)}$ and vice versa. For this purpose, it is sufficient to check if the generator vectors of C''_{j_1,j_2,j_3,j_4} are codewords in $C_{\mathcal{T}(M)}$ and each row of $\mathcal{T}(M)$ is a codeword in C''_{j_1,j_2,j_3,j_4}. We know that the generators of C''_{j_1,j_2,j_3,j_4} are contained in G'' where G'' is the set of vectors in G projected on the indices $\{j_1, j_2, j_3, j_4\}$.

Once we fix $\mathcal{T}(M)$ such that $C''_{j_1,j_2,j_3,j_4} = C_{\mathcal{T}(M)}$, to see if $C \cong C_{\mathcal{T}(M)} \otimes C'$ for some ternary linear code $C' \subseteq \mathbb{F}_3^{n-4}$. Define $C'_{\bar{j_1},\bar{j_2},\bar{j_3},\bar{j_4}}$ to be the projection of C on the indices $[n] \setminus \{j_1, j_2, j_3, j_4\}$. For a vector $c \in C'_{\bar{j_1},\bar{j_2},\bar{j_3},\bar{j_4}}$, let $c^p \in \mathbb{F}_3^n$ be the extensions of c using a codeword $p \in C_{\mathcal{T}(M)}$ along the j_1, j_2, j_3, j_4 coordinates. We note that $C \cong C_{\mathcal{T}(M)} \otimes C'$ for some ternary linear code C' if and only if there exist indices $j_1, j_2, j_3, j_4 \in [n]$, such that

$$C = \left\{ c^p \mid c \in C'_{\bar{j_1},\bar{j_2},\bar{j_3},\bar{j_4}}, p \in C_{\mathcal{T}(M)} \right\}. \tag{4}$$

From the definition of $C'_{\bar{j_1},\bar{j_2},\bar{j_3},\bar{j_4}}$ and C''_{j_1,j_2,j_3,j_4} $(= C_{\mathcal{T}(M)})$, it follows that $C \subseteq \{c^p \mid c \in C'_{\bar{j_1},\bar{j_2},\bar{j_3},\bar{j_4}}, p \in C_{\mathcal{T}(M)}\}$. So, the algorithm just needs to verify if the other side of the containment holds for some indices j_1, j_2, j_3, j_4.

Let G' be the set of vectors of G projected on the coordinates $[n] \setminus \{j_1, j_2, j_3, j_4\}$. Algorithm 2 verifies if g^{p_0}, g^{p_1}, g^{p_3} and g^{p_4} are codewords in C, for every vector $g \in G'$. We now show that this is sufficient. Since C is a code, if $g^{p_0}, g^{p_1}, g^{p_3}, g^{p_4} \in C$ for every $g \in G'$ and $p_i \in \mathcal{T}(M)$, then all linear combinations of these vectors are also in C. Therefore, $\{c^p \mid c \in C'_{\bar{j_1},\bar{j_2},\bar{j_3},\bar{j_4}}, p \in C_{\mathcal{T}(M)}\} \subseteq C$.

There are $2^4 4^4$ possible choices of $\mathcal{T}(M)$ including permutations. For each matrix $\mathcal{T}(M)$, it takes $O(n)$ time to verify if $C_{\mathcal{T}(M)} = C''_{j_1,j_2,j_3,j_4}$. As we had seen that it takes $O(n^2)$ time to verify if an input vector is a codeword using the parity check matrix. We perform this check for $4n$ vectors of the form $\{g^{p_0}, g^{p_1}, g^{p_3}, g^{p_4} \mid g \in G'\}$. So, for a given $\mathcal{T}(M)$ such that $C_{\mathcal{T}(M)} = C''_{j_1,j_2,j_3,j_4}$, It takes $O(n^3)$ time to verify $C \cong C_{\mathcal{T}(M)} \otimes C'$.

Therefore, for every possible choice of $\{j_1, j_2, j_3, j_4\}$, Algorithm 2 takes $O(n^3)$ time to verify if $C \cong C_{\mathcal{T}(M)} \otimes C'$. Since there are at most $\binom{n}{4}$ possible choices of indices, it takes $O(n^7)$ time in total to decide if $C \cong C_{\mathcal{T}(M)} \otimes C'$. ◀

Acknowledgments. We thank Daniel Dadush for helpful suggestions and pointers.

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108, 1996.
2. Laszlo Babai. Automorphism groups, isomorphism, reconstruction. In *Handbook of Combinatorics*, volume chapter 27, pages 1447–1540. North-Holland, 1996.
3. H.C. Chan, C.A. Rodger, and J. Seberry. On inequivalent weighing matrices. *Ars Combinatoria*, 21(A):229–333, 1986.
4. Karthekeyan Chandrasekaran, Venkata Gandikota, and Elena Grigorescu. Deciding Orthogonality in Construction-A Lattices. Under Preparation, 2015.
5. John H. Conway and Neil J. A. Sloane. *Sphere Packings, Lattices and Groups*. Springer-Verlag, New York, 1998.
6. Ishay Haviv and Oded Regev. On the lattice isomorphism problem. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 391–404, 2014.
7. Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 193–206, 1983.
8. Arjen K. Lenstra, Hendrik W. Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.
9. Hendrik W. Lenstra and Alice Silverberg. Lattices with symmetries. Manuscript, 2014.
10. Hendrik W. Lenstra and Alice Silverberg. Revisiting the gentry-szydlo algorithm. In *Advances in Cryptology – CRYPTO 2014*, volume 8616 of *Lecture Notes in Computer Science*, pages 280–296. Springer Berlin Heidelberg, 2014.
11. Daniele Micciancio. Lecture notes on lattice algorithms and applications, Winter 2010. Lecture 2.
12. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-Quantum Cryptography*, pages 147–191. Springer Berlin Heidelberg, 2009.
13. Wilhelm Plesken and Bernd Souvignier. Computing isometries of lattices. *J. Symb. Comput.*, 24(3/4):327–334, 1997.

14 Claus-Peter Schnorr. Factoring integers by CVP algorithms. In *Number Theory and Cryptography – Papers in Honor of Johannes Buchmann on the Occasion of His 60th Birthday*, pages 73–93, 2013.

15 Mathieu Dutour Sikiric, Achill Schürmann, and Frank Vallentin. Complexity and algorithms for computing voronoi cells of lattices. *Math. Comput.*, 78(267):1713–1731, 2009.