The Closest Vector Problem on Gadget Lattices: Efficient Algorithms and Voronoi Geometry

Daniele Micciancio*

November 8, 2026

Abstract

Gadget lattices are a prominent family of lattices widely used in lattice-based cryptography, error-correcting codes, and computational complexity. We present efficient algorithms for the Closest Vector Problem (CVP) on power and CRT gadget lattices and their duals, handling arbitrary real-valued target vectors $\mathbf{t} \in \mathbb{R}^n$.

For the parity-check power gadget lattice $\Lambda_q^\perp(\mathbf{g}^t)$, we give a practical, deterministic $O(n)$ linear-time exact CVP solver based on a new, specialized "deferred greedy" graph-cut technique. Through an explicit characterization of the Voronoi cell, we provide a comprehensive analysis of existing decoders: we prove that the standard BDD algorithm of Micciancio and Peikert (2012) unexpectedly solves CVP *exactly* for odd base b and integer targets, but degrades to a worst-case approximation factor of $\gamma = \sqrt{n}$ for binary base $b = 2$ and continuous targets. Furthermore, we prove that standard pruned sphere decoding requires exponential time $2^{\Omega(n)}$ even when using a geometrically optimal basis.

For the dual (generator) power gadget lattice $\Lambda_q(\mathbf{g}^t)$, we construct an obtuse superbasis—proving for the first time that it is a lattice of "Voronoi's first kind"—which yields a polynomial-time $\widetilde{O}(n^3)$ exact CVP algorithm via graph min-cut reductions. Geometrically, we determine all successive minima and enumerate the facets of their Voronoi cells. In particular, we show that while the parity-check lattice has a sparse Voronoi cell with $n^2 + n$ facets, the generator lattice reaches the theoretical upper bound of $2(2^n - 1)$ facets. Finally, we observe that CRT gadget lattices and their duals admit an orthogonal basis, yielding trivial linear-time CVP algorithms.

1 Introduction

The Closest Vector Problem (CVP), given a lattice $L \subset \mathbb{R}^n$ and a target point $\mathbf{t} \in \mathbb{R}^n$, asks to find a lattice point $\mathbf{v} \in L$ closest to the target, i.e., such that the (Euclidean) length $\|\mathbf{v} - \mathbf{t}\|$ is minimized. CVP is a central problem in the study of the computational complexity of lattices. All the most important lattice problems in NP (like the shortest vector problem, SVP, and the shortest independent vectors problem, SIVP) reduce to CVP via efficient polynomial time reductions [GMSS99, Mic08] that preserve both the lattice dimension and approximation factor. Even the covering radius problem (which is in Π_2, at the second level of the Polynomial Hierarchy) reduces to CVP with a small constant loss ($\gamma = 2$) in the approximation factor [GMR05]. On the complexity

*University of California, San Diego. E-mail: daniele@cs.ucsd.edu. Work supported in part by NSF FRG grant DMS-2411704.

front, CVP is a notoriously hard problem. It has long been known to be NP-hard in its exact and approximate versions [vEB81, ABSS97, DKRS03] for factors up to $n^{O(1/\log\log n)}$.[1]

Solving CVP corresponds to finding optimal solutions in many important applications, like maximum likelihood decoding in communication theory, and minimizing distortion in vector quantization (e.g., see [CS99, AEVZ02, Zam14]). In these applications, the goal is not to solve CVP in arbitrary lattices, but rather finding the closest vectors in a specific, fixed lattice (or, more accurately, sequence of lattices L_n in increasing dimension n) with good error correction properties. Interestingly, in a sequence of papers [Mic01, FM04, Reg04, AKKV11], CVP was shown to be NP-hard to approximate within any constant factor even when the lattice is fixed, and only the target point is part of the input.[2] So, fixing the lattice is not, by itself, enough to make the problem efficiently solvable: the complexity of CVP depends on the specific family of lattices under consideration.

Naturally, CVP is easy to solve on some lattices, like the integer lattice $\mathbb{Z}^n$, where closest vectors can be easily computed rounding each coordinate of the target to the closest integer. However, there are surprisingly few lattice families for which polynomial time CVP algorithms are known. These include the "checkerboard" lattices D_n and their duals D_n^* [CS99, Chapter 20], the root lattices A_n and their duals A_n^* [MCQ08], and the more general family of Coxeter lattices $A_{n,m}$ [MSC10], all of which admit a linear time $O(n)$ CVP algorithm. Slower (but still polynomial time) CVP algorithms have been discovered for lattices of "Voronoi first kind" [MGC14] (i.e., lattices that admit an obtuse superbasis, see Section 2), and for the tensor product $A_n \otimes A_m$ of root lattices [DvW18]. Interestingly, these CVP algorithms work by reduction to graph problems, namely max-flow/min-cut, and negative cycle detection. These algorithms have been further generalized (at the cost of an even more complex, but still polynomial time algorithm based on linear programming) to solve CVP in zonotopal lattices [MPSV21].[3] To the best of our knowledge, these are the only (indecomposable[4]) lattices for which CVP is known to admit a polynomial time solution.

Efficient (polynomial time) algorithms are known to decode a few other prominent lattice families, like the Barnes-Wall lattices [MN08, GP17], and certain very dense lattices close to Minkowski's bound [DP19, MP22, KM24]. But these algorithms only solve the Bounded Distance Decoding problem (BDD), a restricted version of CVP where the target $\mathbf{t}$ is promised to be close to the lattice. Solving CVP in these same lattices seems substantially harder, and no polynomial time CVP solution is currently known, or may even exist.[5] This mirrors the situation in coding theory, where

[1] The classical hardness factor of [DKRS03] has been the subject of recent unrefereed preprints exploring polynomial inapproximability bounds.

[2] This is called "CVP with Preprocessing", or CVPP, as one is allowed to arbitrarily preprocess the lattice before seeing the target point. See also [ABGS21] for recent work on the fine-grained complexity of CVPP.

[3] We recall that a zonotope is the projection of a (higher dimensional) hypercube. Zonotopal lattices (lattices whose Voronoi cells are zonotopes) include graphic and cographic lattices defined by totally unimodular matrices, as well as lattices of Voronoi's first kind. Voronoi's conjecture [Vor08] asserts that every zonotopal lattice is of Voronoi's first kind. While [MPSV21] provides a general polynomial-time framework via convex network flows, explicit structured families (such as root lattices and gadget lattices) typically admit much faster specialized combinatorial solvers.

[4] CVP algorithms for lattices L_1, L_2 can be immediately combined to solve CVP in the direct sum lattice $L_1 \oplus L_2$. So, it is enough to consider lattices that cannot be decomposed as the sum of two orthogonal sublattices. Another simple method to extend CVP algorithms is to decompose a lattice L into the union of cosets L/M, modulo some sublattice that admits an efficient CVP algorithm.

[5] It is easy to see that CVP is NP-hard even when restricted to lattices where BDD is polynomially solvable. Moreover, the centrality of CVP in the algorithmic study of lattices, strongly relies on the ability to decode arbitrary target vectors, beyond the BDD decoding radius. E.g., while SVP is easily reducible to CVP [GMSS99], known reductions to BDD [LM09] only solve much weaker variants of SVP, like approximating the length of the shortest vector within $O(\sqrt{n})$ factors, without even finding an approximately shortest vector. In particular, while CVP and

gadget	Lattice	modulus q	determinant	λ_1	λ_k	$\sharp\mathcal{V}$	CVP
power	$\Lambda_q^\perp$	b^n	q	b	$\sqrt{b^2+1}$	n^2+n	$O(n)$
power	Λ_q	b^n	q^{n-1}	q/b	$q\sqrt{\frac{1-\frac{1}{b^{2k}}}{b^2-1}}$	$2(2^n-1)$	$\tilde{O}(n^3)$
crt	$\Lambda_q^\perp$	$\prod_i q_i$	q	q_1	q_k	$2n$	$O(n)$
crt	Λ_q	$\prod_i q_i$	q^{n-1}	q/q_n	q/q_{n-k+1}	$2n$	$O(n)$

Figure 1: Geometric properties of (parity-check and generator) gadget lattices associated to the power gadget vector $\mathbf{g} = (1, b, b^2, \ldots, b^{n-1}) \in \mathbb{Z}_{b^n}^n$ (parameterized by base $b \geq 2$ and dimension n), and crt gadget vector $\mathbf{g} = (q/q_1, \ldots, q/q_n) \in \mathbb{Z}_q^n$ where $q = \prod_i q_i$ is a relatively prime factorization of q with $q_1 < q_2 < \ldots < q_n$. For each lattices, $\lambda_1, \ldots, \lambda_n$ are the successive minima of the lattice, $\sharp\mathcal{V}$ is the number of facets of the Voronoi cell of the lattice, and `CVP` is the running time of our CVP algorithm.

the BDD problem for prominent families of linear codes (like Reed-Solomon codes) is known to be efficiently solvable (even in its list decoding version [GS99]), but no efficient CVP[6] algorithm for the same code families is known or likely to exist [GV05, CW10, GGG18].

Our results In this paper, we establish the geometric properties and present exact CVP algorithms for power and CRT *gadget lattices*. A complete overview of our quantitative results—including determinants, successive minima, Voronoi cell facet counts, and algorithm running times—is summarized in Figure 1.

Gadget lattices are fundamental to lattice-based cryptography, underpinning trapdoor generation [MP12], hierarchical IBE [CHKP12], FHE homomorphic evaluations [GSW13, AP14], and the study of average-case hardness [BLP+13]. (See also [GMP19] for a discussion of how different algorithmic problems on gadget lattices relate to each other.)

Formally, a *gadget* vector $\mathbf{g} \in \mathbb{Z}_q^n$ defines two naturally dual lattices: the *parity-check gadget lattice* $\Lambda_q^\perp(\mathbf{g}^t)$ (the integer kernel of $\mathbf{g} \bmod q$), and the *generator gadget lattice* $\Lambda_q(\mathbf{g}^t)$ (the lattice generated by $\mathbf{g} \bmod q$).[7] Because $\Lambda_q^\perp(\mathbf{g}^t)$ is the lattice most frequently deployed in cryptographic trapdoor constructions, our primary focus is on solving exact CVP for $\Lambda_q^\perp(\mathbf{g}^t)$ in linear time $O(n)$, while also establishing a polynomial-time $\tilde{O}(n^3)$ CVP solver for the generator lattice $\Lambda_q(\mathbf{g}^t)$.

Crucially, defining a lattice via a single vector $\mathbf{g}$ does not automatically make CVP easy: any lattice can be efficiently approximated by a single-vector lattice [PS87], and for a uniform random $\mathbf{g}$, SVP (and thus CVP [GMSS99]) is provably hard on average [Ajt96]. Thus, algorithmic efficiency relies entirely on the specific algebraic structure of $\mathbf{g}$. While CRT gadget lattices easily decompose into an orthogonal sum of 1-dimensional sublattices (yielding trivial $O(n)$ decoding, see Section 6), power gadget lattices $\mathbf{g} = (1, b, \ldots, b^{n-1}) \bmod b^n$ possess non-orthogonal structures where exact CVP decoding is non-trivial.

The standard MP12 algorithm of [MP12] provides an efficient BDD solver designed for discrete integer targets within the BDD radius. However, in many cryptographic settings—such as ciphertext compression via vector quantization [MS23], FHE modulus switching [BGV14], and theoretical

SVP are known to be NP-hard [vEB81, Ajt98, Mic12], no NP-hardness result is know for BDD or the variants of SVP that reduce to it.

[6] In coding theory, this is commonly called the "Nearest Codeword" or "Maximum Likelihood Decoding" problem.

[7] Geometrically, $\Lambda_q(\mathbf{g}^t)$ is a scaled copy of the dual lattice $(\Lambda_q^\perp(\mathbf{g}^t))^* = \frac{1}{q}\Lambda_q(\mathbf{g}^t)$. See Section 2 for details.

reductions [BLP+13]—the target vector $\mathbf{t} \in \mathbb{R}^n$ is naturally continuous (non-integral). Furthermore, while cryptographic protocols keep noise strictly within the BDD radius to prevent decryption failure, broader applications in signal processing, vector quantization, and channel coding under additive white Gaussian noise (AWGN) require solving exact CVP for arbitrary targets without BDD restrictions. Through an explicit characterization of the Voronoi cell (Theorem 4.2), we resolve how prior decoding approaches behave outside the BDD radius and for arbitrary targets:

- **Odd base and integer targets:** We prove that for any odd base b and integral target $\mathbf{t} \in \mathbb{Z}^n$, the MP12 algorithm unexpectedly solves CVP *exactly* (Corollary 4.3), extending past its original BDD analysis.
- **Worst-case approximation gap:** For even bases or continuous targets $\mathbf{t} \in \mathbb{R}^n$, MP12 generally fails once targets move outside the BDD radius (Corollary 4.2). For the standard binary base $b = 2$, its CVP approximation factor degrades to $\gamma = \sqrt{n}$ (Theorem 4.4).
- **Hardness of sphere decoding:** We prove that standard exact decoders, such as pruned sphere decoding [FP85, SE94], require exponential time $2^{\Omega(n)}$ (Theorem 4.5) on gadget lattices, even when given a geometrically optimal basis that simultaneously achieves all successive minima.

Algorithms and Geometric Contributions We overcome these limitations by introducing a deterministic linear-time algorithm that solves exact CVP on the parity-check power gadget lattice $\Lambda_q^\perp(\mathbf{g}^t)$ in $O(n)$ time for arbitrary real-valued targets $\mathbf{t} \in \mathbb{R}^n$ (Theorem 4.3). Our algorithm matches the execution speed of MP12 while guaranteeing exact CVP solutions. Practical experiments (Section 7) confirm sub-millisecond execution times in dimension as high as $n = 1000$ (Figure 4).

For the generator power gadget lattice $\Lambda_q(\mathbf{g}^t)$, we prove the existence of an obtuse superbasis, establishing for the first time that $\Lambda_q(\mathbf{g}^t)$ is a lattice of "Voronoi's first kind". This yields a polynomial-time CVP algorithm running in $\widetilde{O}(n^3)$ time via graph min-cut reductions [MGC14].

Importantly, while all previously known explicit families of Voronoi 1st kind lattices (such as A_n and D_n^*) already possessed faster specialized solvers, $\Lambda_q(\mathbf{g}^t)$ represents the first explicitly named lattice family for which MGC14 provides the primary polynomial-time CVP solution (see Remark 5.1 for further discussion). Geometrically, this result highlights a striking structural contrast: while the parity-check lattice $\Lambda_q^\perp(\mathbf{g}^t)$ has a sparse Voronoi cell with only $n^2 + n$ facets, the generator lattice $\Lambda_q(\mathbf{g}^t)$ features $2(2^n - 1)$ facets—the maximum theoretical upper bound for any n-dimensional lattice. This extreme geometric complexity explains why decoding the generator lattice is more intricate and why optimizing its solver down to $O(n)$ remains an open problem.

Technical Overview The starting point of our work is the observation that the power gadget lattice basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$ introduced in [MP12] to solve the BDD problem on $\Lambda_q^\perp(\mathbf{g}^t)$ gives an "obtuse superbasis", i.e., when extended with a "parity check" vector $\mathbf{b}_0 = -\sum_{i=1}^n \mathbf{b}_i$, it has the property that all scalar products $\langle \mathbf{b}_i, \mathbf{b}_j \rangle \leq 0$ are negative. Without $\mathbf{b}_0$, any lattice has a basis with this property and such a basis can be easily computed. (See Lemma 2.2.) However, with the inclusion of $\mathbf{b}_0$ the property is nontrivial, and most lattices in dimension beyond 3 do not admit such a superbasis.

This shows that the power gadget lattice $\Lambda_q^\perp(\mathbf{g}^t)$ is a lattice of "Voronoi first kind", and already allows to solve CVP in this lattice in polynomial time using the algorithm of [MGC14]. The high

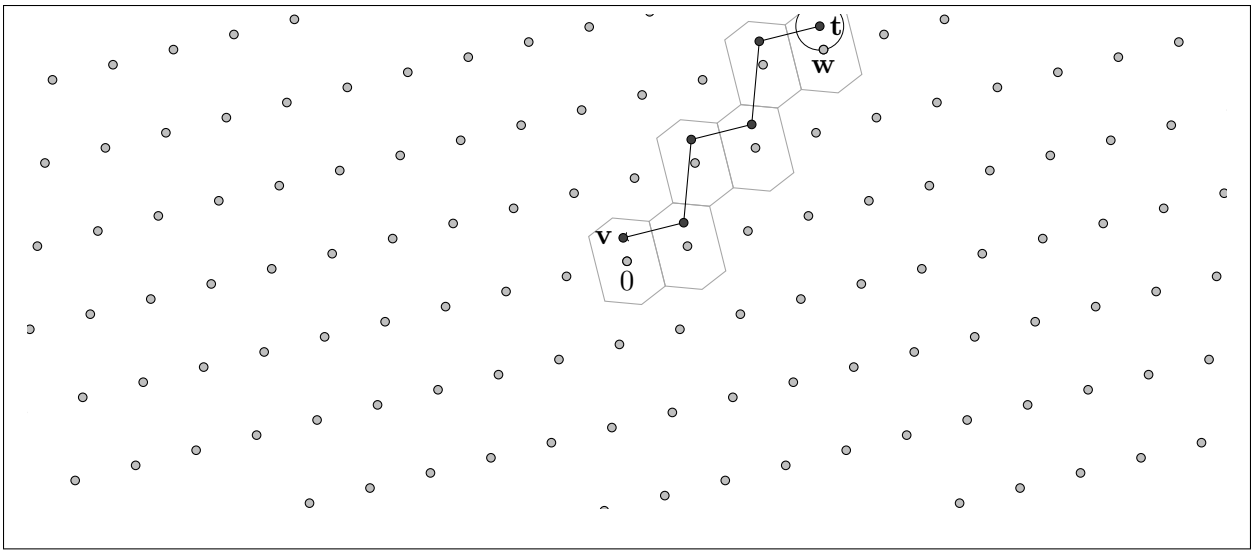

Figure 2: Solving CVP by walking on the Voronoi graph. The length of the target $\mathbf{t}$ is reduced by subtracting a Voronoi relevant vector from it, moving the target to a neighboring Voronoi cell. The process is repeated until the target $\mathbf{v}$ is in the Voronoi cell of the origin, and its length cannot be reduced any further. At this point, $\mathbf{w} = \mathbf{t} - \mathbf{v}$ is a lattice point closest to the original target $\mathbf{t}$.

level structure of the algorithm is illustrated in Figure 2: starting from a target $\mathbf{t}$ in the Voronoi cell of lattice point $\mathbf{w}$, the algorithm performs a walk on the lattice coset $\mathbf{t} + \Lambda_q^\perp(\mathbf{g}^t)$, making $\mathbf{t}$ shorter and shorter. At each step, $\mathbf{t}$ moves between neighboring Voronoi cells, until it reaches $\mathbf{v}$ in the Voronoi cell of the origin. Then, the lattice point closest to the original target is given by $\mathbf{w} = \mathbf{t} - \mathbf{v}$. This structure was already used in previous algorithms [SFS09, MV13] for arbitrary lattices, where selecting a neighboring Voronoi cell can take exponential time, and the length of the walk can also be exponential in the lattice dimension. The novelty in [MGC14] is to show that for lattice of Voronoi first kind, the walk only requires $O(n)$ steps, and each step can be determined in polynomial time by reduction to finding a minimum cut in a graph. While polynomial time, the resulting algorithm is not very efficient: it involves the solution of a sequence of $O(n)$ max-flow/min-cut problems on graphs with $n + 3$ nodes. Using a $O(n^3)$-time algorithm to solve each instance, this results in $O(n^4)$ total running time.[8] We build on the work of [MGC14] to design a much faster algorithm by using the special properties of the graphs associated to gadget lattices. Our improved running time comes from two factors:

1. We introduce a new geometric method (Theorem 3.1) to estimate the number of steps in the Voronoi walk in Figure 2. Then we use this method (together with Varah's bound on the operator norm of the inverse of strictly diagonally dominant matrices [Var75]) to bound the number of min-cut instances required by power gadget lattices $\Lambda_q^\perp(\mathbf{g}^t)$ by a constant $O(1)$, rather than the linear factor $O(n)$ from [MGC14].

2. We exploit the special structure of the graphs associated to gadget lattices to solve Min-Cut in linear time using a specialized "deferred greedy" technique. (See below.)

[8] This is the running time claimed in [MGC14]. Exploiting the sparsity of the graph, and using more recent min-cut algorithms [Kar00, Li21, HLRW24] the running time reduces to $\tilde{O}(n^2)$.

We remark that the graphs associated to power gadget lattices $\Lambda_q^\perp(\mathbf{g}^t)$ are sparse, with only $O(n)$ edges. However, sparsity by itself is not enough to design truly fast algorithms, as the best asymptotic method is not quite practical and would still run in superlinear time $\tilde{O}(n) \approx O(n \log^3 n)$. We propose a variant of the method from [MGC14] where each basic step of the walk in Figure 2 requires the solution of two min-cut problem instances with $n+2$ nodes, rather than a single instance with $n+3$ nodes. Then, we solve this specialized min-cut instances using a variant of the greedy technique that we call the "deferred greedy method" (or "deferred-choice greedy"). Unlike standard greedy approaches that immediately assign nodes to one or the other side of the cut, our algorithm introduces a third possibility: assigning a node to the same side of the cut as the next (yet to be examined) node in the sequence. This "deferred choice" greedy strategy still enjoys the simplicity and linear running time properties of the standard greedy approach, while at the same time producing provably optimal solutions for our min-cut instances.

For the generator lattice, the situation is quite different. The dual of the standard basis from [MP12] achieves the successive minima, but it is not obtuse. In fact, the Voronoi's first kind lattice property is not generally preserved under duality. So, it is not a priori clear that an obtuse superbasis exists at all. Our contribution here is to build an obtuse superbasis, showing that the dual of a gadget lattice is also a lattice of Voronoi's first kind. It follows that CVP can be solved in polynomial time $\tilde{O}(n^3)$ using the algorithms of [MGC14, HLRW24]. We do not optimize the running time of this algorithm, but we conjecture that also for the dual (generator) gadget lattice CVP can be solved in linear time $O(n)$.

Paper organization The rest of the paper is organized as follows. In Section 2 we present preliminary definitions used in this paper. In Section 3 we review the CVP algorithm for lattices of Voronoi first kind of [MGC14] and describe our variant of the algorithm and associated analysis. In Section 4 we present our main results on gadget lattices, including a description of their geometric properties and the linear time CVP algorithm. In Section 5 we describe the geometric properties of the dual (generator) gadget lattices, including a basis of Voronoi first kind that allows to solve CVP in polynomial time using the algorithm from [MGC14]. Section 6 discusses CRT gadget lattices (and their duals), the other class of gadgets most commonly used in lattice-based cryptography, and how they can be decomposed into the orthogonal sum of one-dimensional lattices, giving a trivial algorithm to solve CVP in linear time. Experimental results using a prototype implementation of our CVP algorithm are presented in Section 7. Section 8 concludes with a discussion of other gadget lattices and related open problems.

2 Preliminaries

We write $\mathbf{I}$ for the $n \times n$ identity matrix, and $\mathbf{J}$ for the $n \times n$ Jordan block with 1s on the superdiagonal, and 0s everywhere else. A square matrix $\mathbf{A} = (a_{i,j})_{i,j} \in \mathbb{R}^{n \times n}$ is *strictly diagonally dominant* if the absolute value of each diagonal element is strictly bigger than the sum of the absolute values of all other elements on the same row, i.e., $\sum_{j \neq i} |a_{i,j}| < |a_{i,i}|$ for all $i = 1, \ldots, n$. The ℓ_∞ operator norm of a matrix is $\|\mathbf{A}\|_\infty = \sup_{\mathbf{x}} \frac{\|\mathbf{A}\mathbf{x}\|_\infty}{\|\mathbf{x}\|_\infty}$. We will need the following bound on the operator norm of the inverse of a strictly diagonally dominant matrix.

Theorem 2.1 (Varah's bound, [Var75]) *For any strictly diagonally dominant matrix* $\mathbf{A} \in \mathbb{R}^{n\times n}$,

$$\|\mathbf{A}^{-1}\|_\infty = \sup_{\mathbf{x}} \frac{\|\mathbf{A}\mathbf{x}\|_\infty}{\|\mathbf{x}\|_\infty} \leq \frac{1}{\min_i \left(|a_{i,i}| - \sum_{j\neq i} |a_{i,j}|\right)}.$$

2.1 Lattices

A (full rank) lattice is the set of all integer combinations $\Lambda = \mathbf{B}\mathbb{Z}^n$ of n linearly independent basis vectors $\mathbf{B} = [\mathbf{b}_1, \cdots, \mathbf{b}_n] \in \mathbb{R}^{n\times n}$. The dual lattice of Λ is the set $\Lambda^* = \{\mathbf{z} \in \mathbb{R}^n : \langle \mathbf{z}, \Lambda\rangle \subseteq \mathbb{Z}\}$ of all vectors that have integer scalar product with the elements of Λ. If $\Lambda = \mathbf{B}\mathbb{Z}^n$ is a lattice with basis $\mathbf{B}$, then its dual $\Lambda^* = \mathbf{B}^{-t}\mathbb{Z}^n$ is a lattice with basis $\mathbf{B}^{-t}$.

All the following definitions can be given with respect to any norm, but the Euclidean norm $\|\mathbf{x}\| = \sqrt{\sum_i x_i^2}$ is the most common. For any lattice $\Lambda \subset \mathbb{R}^n$ and target $\mathbf{t} \in \mathbb{R}^n$, let $\mu(\mathbf{t}, \Lambda) = \inf\{\|\mathbf{t} - \mathbf{v}\| \mid \mathbf{v} \in \Lambda\}$ be the distance between $\mathbf{t}$ and the closest lattice point. The closest vector problem, CVP, given a lattice basis $\mathbf{B} \in \mathbb{R}^{n\times n}$ and target $\mathbf{t} \in \mathbb{R}^n$, asks to find a lattice vector closest to the target, i.e., a vector $\mathbf{v} \in \mathbf{B}\mathbb{Z}^n$ such that the distance $\|\mathbf{t} - \mathbf{B}\mathbf{v}\| = \mu(\mathbf{t}, \mathbf{B}\mathbb{Z}^n)$ is minimized.

The covering radius of a (full rank) lattice Λ is the maximum distance $\mu(\Lambda) = \sup\{\mu(\mathbf{t}, \Lambda) \mid \mathbf{t} \in \mathbb{R}^n\}$. Other important parameters associated to a lattice are its successive minima. For any $i = 1, \ldots, n$, the ith successive minimum of Λ, denoted $\lambda_i(\Lambda)$, is the radius of the smallest ball containing i linearly independent lattice vectors from Λ. The Bounded Distance Decoding (BDD) problem is a special version of CVP restricted to instances $(\Lambda, \mathbf{t})$ where the target $\mathbf{t}$ is very close to the lattice, typically $\mu(\mathbf{t}, \Lambda) < \lambda_1(\Lambda)/2$. Note that by triangle inequality the solution to BDD is necessarily unique, i.e., there is at most one lattice point within distance $\lambda_1(\Lambda)/2$ from any given target $\mathbf{t}$. When $\mu(\mathbf{t}, \Lambda) \geq \lambda_1(\Lambda)/2$, BDD algorithms are not required to find a lattice point closest to the target $\mathbf{t}$, or even a lattice point within distance $\mu(\Lambda)$. So, solving CVP in general is substantially harder than BDD.

Given a basis $\mathbf{B} = [\mathbf{b}_1, \cdots, \mathbf{b}_n]$ for a lattice Λ, its Gram-Schmidt orthogonalization is the set of vectors $\widehat{\mathbf{B}} = [\widehat{\mathbf{b}}_1, \cdots, \widehat{\mathbf{b}}_n]$ where $\widehat{\mathbf{b}}_i$ is the component of $\mathbf{b}_i$ orthogonal to $\mathbf{b}_1, \cdots, \mathbf{b}_{i-1}$.

2.2 Voronoi

The (open) Voronoi cell of a lattice $\mathcal{V}(\Lambda)$ is the set of all points $\mathbf{x} \in \mathbb{R}^n$ that are closer to the origin $\mathbf{0}$ than to any other lattice point. For any $\mathbf{v} \in \mathbb{R}^n$, let

$$H_{\mathbf{v}} = \{\mathbf{x} \in \mathbb{R}^n \colon \|\mathbf{x}\| < \|\mathbf{x} - \mathbf{v}\|\} = \{\mathbf{x} \in \mathbb{R}^n \colon 2\langle \mathbf{x}, \mathbf{v}\rangle < \|\mathbf{v}\|^2\} \tag{1}$$

be the set of points that are closer to the origin than to $\mathbf{v}$. By definition, the Voronoi cell $\mathcal{V}(\Lambda)$ is the intersection of halfspaces $\mathcal{V}(\Lambda) = \bigcap_{\mathbf{v}\in V} H_{\mathbf{v}}$ where $V \subseteq \Lambda \setminus \{\mathbf{0}\}$. The minimal such set is called the set of Voronoi relevant vectors, and was already characterized by Voronoi.

Lemma 2.1 (Voronoi [Vor08], e.g., see [CS99, Chapter 21, Theorem 10, page 477]) *For any lattice* Λ*, the vector* $\mathbf{v} \in \Lambda \setminus \{0\}$ *is Voronoi relevant if and only if* $\pm\mathbf{v}$ *are the shortest vectors in* $\mathbf{v} + 2\Lambda$*, and any other vector in* $\mathbf{v} + 2\Lambda$ *is strictly longer than* $\|\mathbf{v}\|$*.*

It follows from Lemma 2.1 that an n-dimensional lattice has at most $2^n - 1$ pairs of Voronoi relevant vectors, a pair $\pm v$ for each congruence class of $\Lambda \pmod{2\Lambda}$ except for the trivial class

2Λ. So, the Voronoi relevant vectors of a lattice $\Lambda = \mathbf{B}\mathbb{Z}^n$ can be enumerated as follows. For any nonzero binary vector $\mathbf{u} \in \{0,1\}^n$, let $V_{\mathbf{u}}$ be the set of all shortest vectors in $2\Lambda + \mathbf{Bu}$. Then, the set of Voronoi relevant vectors is the union of all $V_{\mathbf{u}}$ of size $|V_{\mathbf{u}}| = 2$.

Any lattice basis $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ can be (uniquely) extended with an extra vector $\mathbf{b}_0 = -\sum_{i=1}^n \mathbf{b}_i$ such that $\sum_{i=0}^n \mathbf{b}_i = \mathbf{0}$. The resulting set $\overline{\mathbf{B}} = [\mathbf{b}_0, \ldots, \mathbf{b}_n]$ is called an *obtuse superbasis* if $\langle \mathbf{b}_i, \mathbf{b}_j \rangle \leq 0$ for all $0 \leq i < j \leq n$, i.e., the angle between any two superbasis vectors is obtuse. A lattice is of Voronoi's first kind if it has an obtuse superbasis. We say that $\mathbf{B}$ is a basis of Voronoi's first kind if the corresponding superbasis $\overline{\mathbf{B}}$ is obtuse.

We remark that all lattices Λ have an obtuse basis. In fact, any basis $\mathbf{B}$ can be (efficiently) modified into an equivalent one such that $\langle \mathbf{b}_i, \mathbf{b}_j \rangle \leq 0$ for all $i \neq j$, as shown in the next lemma.

Lemma 2.2 *Any lattice has a (strictly) obtuse basis, i.e., a basis* $\mathbf{B}$ *such that* $\langle \mathbf{b}_i, \mathbf{b}_j \rangle < 0$ *for all* $i \neq j$.

Proof: We show how to modify any lattice basis $\mathbf{B}$ into an obtuse one. We proceed by induction on the rank of the basis. Assume the first $k-1$ basis vectors $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_{k-1}]$ satisfy the obtusity condition $\langle \mathbf{b}_i, \mathbf{b}_j \rangle < 0$ for all $1 \leq i < j < k$, and let $\mathbf{b} = \mathbf{b}_k$ the next basis vector. We claim that there is an integer vector $\mathbf{x} \in \mathbb{Z}^{k-1}$ such that $\langle \mathbf{b}_i, \mathbf{b} - \mathbf{Bx} \rangle < 0$ for all $i < k$, so that $[\mathbf{B}, \mathbf{b} - \mathbf{Bx}]$ is an obtuse basis for $\mathcal{L}([\mathbf{B}, \mathbf{b}])$. In matrix notation, this can be written as $\mathbf{B}^t(\mathbf{b} - \mathbf{Bx}) < 0$, or, equivalently, $\mathbf{Gx} > \mathbf{B}^t\mathbf{b}$, where $\mathbf{G} = (\mathbf{B}^t\mathbf{B})$ is an invertible matrix. Let $\mathbf{x}' = \mathbf{G}^{-1}(\mathbf{B}^t\mathbf{b} + \delta\mathbf{u})$, where $\mathbf{u} = (1, \ldots, 1)$ and δ is a sufficiently large constant to be determined. Write $\mathbf{x}'$ as the sum of an integer vector $\mathbf{x} \in \mathbb{Z}^k$ and $\mathbf{r}$ such that $\|\mathbf{r}\|_\infty \leq 1/2$. Then, we have

$$\mathbf{Gx} = \mathbf{Gx}' - \mathbf{Gr} = \mathbf{B}^t\mathbf{b} + \delta\mathbf{u} - \mathbf{Gr}.$$

It remains to prove that $\delta\mathbf{u} - \mathbf{Gr} > \mathbf{0}$. This is achieved setting δ to any value bigger than

$$\max_{\mathbf{r}} \|\mathbf{Gr}\|_\infty = \max_{i<k} \|\mathbf{b}_i\|^2 + \frac{1}{2}\langle \mathbf{b}_i, \mathbf{b}_0 \rangle$$

where $\mathbf{b}_0 = -(\mathbf{b}_1 + \cdots + \mathbf{b}_{k-1})$ is the superbasis vector of $\mathbf{B}$. □

What makes lattices of Voronoi's first kind special is that also the superbasis vector $\mathbf{b}_0 = -\sum_{i\geq 1} \mathbf{b}_i$ satisfies the obtusity condition, or, equivalently, the sum $\sum_{i\geq 1} b_i$ makes an acute angle with the basis vectors, i.e., $\left\langle \mathbf{b}_j, \sum_{i\geq 1} \mathbf{b}_i \right\rangle \geq 0$ for all $j \geq 1$. All lattices up to dimension 3 have an obtuse superbasis. However, this requirement is nontrivial in higher dimension, and it implies a number of other useful properties.

Theorem 2.2 ([CS92, Theorem 3]) *Let* $\mathbf{B}$ *be a basis of Voronoi's first kind. Then, all Voronoi relevant vectors of* $\mathcal{L}(\mathbf{B})$ *have the form* $\pm\mathbf{B}\{0,1\}^n \setminus \{0\}^n$. *Moreover, all vectors in* $\pm\mathbf{B}\{0,1\}^n \setminus \{0\}^n$ *are (strictly) Voronoi relevant if and only if* $\langle \mathbf{b}_i, \mathbf{b}_j \rangle < 0$ *is strictly negative for all superbasis vectors* $i, j \in \{0, \ldots, n\}$, $i \neq j$.

This and other useful properties are easily proved using Selling's formula for the norm of a vector $\sum_{i\geq 0} \mathbf{b}_i x_i$ in terms of its superbasis coordinates

$$\left\| \sum_{i\geq 0} \mathbf{b}_i x_i \right\|^2 = \sum_{0\leq i<j} (-\langle \mathbf{b}_i, \mathbf{b}_j \rangle)(x_i - x_j)^2 \tag{2}$$

which can be verified by expanding the products and simplifying. So, for example, since all coefficients $-\langle \mathbf{b}_i, \mathbf{b}_j \rangle$ in (2) are nonnegative, setting $\mathbf{x} = \mathbf{x} \bmod 2 \in \{0,1\}^n$ (and $x_0 = 0$) minimizes the norm $\|\mathbf{Bx}\|$ over the coset $\mathbf{Bx} + 2\mathcal{L}(\mathbf{B})$. It follows from Lemma 2.1 that Voronoi relevant vectors are necessarily of the form $\pm\mathbf{Bx}$ for $\mathbf{x} \in \{0,1\}^n \setminus \{\mathbf{0}\}$ as claimed in Theorem 2.2.

2.3 Gadget lattices

A *q-ary* lattice is a full-rank integer lattice that contains $q \cdot \mathbb{Z}^n$ as a sublattice. Any matrix $\mathbf{A} \in \mathbb{Z}_q^{k \times n}$ defines two q-ary lattices

$$\Lambda_q^\perp(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^n : \mathbf{Az} = \mathbf{0} \mod q\} \tag{3}$$

$$\Lambda_q(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m : \exists\, \mathbf{s} \in \mathbb{Z}^k, \mathbf{v} = \mathbf{A}^t\mathbf{s} \mod q\} \tag{4}$$

which satisfy $\Lambda_q^\perp(\mathbf{A})^* = q \cdot \Lambda_q(\mathbf{A})$, i.e., they are dual up to scaling. Lattice cryptography makes extensive use of two types of q-ary lattices:

- *Random lattices* $\Lambda_q(\mathbf{A}), \Lambda_q^\perp(\mathbf{A})$, obtained picking $\mathbf{A} \in \mathbb{Z}_q^{k \times n}$ uniformly at random, and for which solving the shortest vector problem (and therefore also CVP [GMSS99]) is provably as hard as worst-case lattice problems [Ajt96, MR07, Reg09, BLP+13] even when $k = 1$ (i.e., $\mathbf{A} = \mathbf{a}^t$ is a single row vector), and
- *Gadget lattices* $\Lambda_q(\mathbf{g}^t), \Lambda_q^\perp(\mathbf{g}^t)$ defined by a single, carefully chosen gadget vector[9] $\mathbf{g} \in \mathbb{Z}_q^n$, which makes the lattice problems efficiently solvable.

The computational hardness of random lattices $\Lambda_q(\mathbf{a}^t), \Lambda_q(\mathbf{a}^t)$ shows that one should not expect gadget lattices to admit efficient algorithms, unless the gadget vector $\mathbf{g}$ has some very special properties. There are two types of gadget vectors that have been investigated and used in a wide range of applications:

- The power gadget vector

$$\mathbf{g} = (1, b, b^2, \ldots, b^{n-1}) \in \mathbb{Z}_q^n \qquad \text{with} \quad q = b^n \tag{5}$$

  parameterized by an integer base $b \geq 2$, and dimension n, and

- The CRT gadget vector

$$\mathbf{q} = (\frac{q}{q_1}, \ldots, \frac{q}{q_n}) \in \mathbb{Z}_q^n \qquad q = \prod_i q_i \tag{6}$$

  parameterized by relatively prime integers $q_1, \ldots, q_n$.

As we explain in Section 6, the lattices $\Phi_{\mathbf{q}} = \Lambda_q(\mathbf{q}), \Phi_{\mathbf{q}}^\perp = \Lambda_q^\perp(\mathbf{q})$ defined by the CRT gadget (6) have an orthogonal (in fact, diagonal) basis, which immediately allows to solve CVP in linear time by rounding each coordinate of the target. The focus on this paper are the lattices associated to the power gadget (5), which do not admit an orthogonal basis, and for which solving CVP is much more challenging. These gadgets were originally used in [MP12] to give the first formal treatment and explicit description of gadget lattices, and are still used in a wide range of applications.

[9] More generally, one may consider gadget lattices $\Lambda_q(\mathbf{G}), \Lambda_q^\perp(\mathbf{G})$ defined by a *gadget matrix* $\mathbf{G}$ with more than one row. But in practice, these are typically constructed by combining many copies $\mathbf{G} = \mathbf{I} \otimes \mathbf{g}^t$ of a single gadget vector $\mathbf{g}$. Then, the resulting lattices $\Lambda_q^\perp(\mathbf{G}) = \Lambda_q^\perp(\mathbf{g}^t)^{\oplus k}$ (and their duals $\Lambda_q(\mathbf{G}) = \Lambda_q(\mathbf{g}^t)^{\oplus k}$) are the direct (orthogonal) sum of k copies of a gadget lattice defined by a single row $\mathbf{g}^t$. So, solving CVP in $\Lambda_q^\perp(\mathbf{G})$ and $\Lambda_q(\mathbf{G})$ reduces to k independent CVP instances on the smaller dimensional component lattices $\Lambda_q^\perp(\mathbf{g}^t)$, $\Lambda_q(\mathbf{g}^t)$.

3 The CVP algorithm for lattices of Voronoi first kind

At the top level, the CVP algorithm of [MGC14] follows the "iterative slicing" approach [SFS09, MV13], which, starting from a target $\mathbf{t}$, keeps subtracting Voronoi relevant vectors from $\mathbf{t}$ to make the target shorter and shorter. (See Figure 2.) The novelty in [MGC14] is to use the special properties of obtuse superbases (Theorem 2.2) both to speed up the selection of optimal Voronoi relevant vectors (i.e., the selection of a neighboring Voronoi cell at each step of the walk in Figure 2,) and prove better bounds on the number of required iterations (i.e., the number of steps in the walk from $\mathbf{t}$ to $\mathbf{v}$ in Figure 2,) from exponential in the lattice dimension (as in [SFS09, MV13] for general lattices) to polynomial.[10]

Algorithm 1: Generic CVP Algorithm for lattices of Voronoi first kind

```
1  On input B = [b_1, ..., b_n], t ∈ span(B)
2  v ← t − B ⌊B^{-1} t⌋
3  repeat
4    g ← argmin {‖v − Bg‖ | g ∈ ±{0,1}^n}
5    if ‖v − Bg‖ < ‖v‖
6      then v ← v − Bg
7      else let w = t − v
8           return w(∈ BZ^n)
```

Pseudocode for the top level structure of the CVP algorithm is given in Algorithm 1. On input a lattice basis $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ of Voronoi first kind and a target $\mathbf{t} \in \mathrm{span}(\mathbf{B})$, the algorithm works as follows:

1. Initialization: let $\mathbf{v} = \mathbf{t} - \mathbf{B}\left\lfloor \mathbf{B}^{-1}\mathbf{t} \right\rfloor$ be the (unique) representative of $\mathbf{t} + \mathcal{L}(\mathbf{B})$ in $\mathbf{B}[0,1)^n$. (Usually, this step is not even needed because the application already guarantees that the target belongs to a bounded region of space, like the fundamental parallelepiped, or its orthogonalization.)

2. Find a vector $\mathbf{g} \in \pm\{0,1\}^n$ that minimizes $\|\mathbf{v} - \mathbf{B}\mathbf{g}\|$ by reduction to Min-Cut as described later.

3. If $\|\mathbf{v} - \mathbf{B}\mathbf{g}\| < \|\mathbf{v}\|$, then subtract $\mathbf{B}\mathbf{g}$ from $\mathbf{v}$ and repeat.

4. Otherwise, output $\mathbf{t} - \mathbf{v}$ and terminate.

Notice that throughout the execution of the algorithm, $\mathbf{v}$ is always an element of the lattice coset $\mathbf{t} + \mathcal{L}(\mathbf{B})$, and its length decreases at every iteration. So, since $\mathbf{t} + \mathcal{L}(\mathbf{B})$ is a discrete set, the algorithm necessarily terminates. Moreover, upon exiting the loop, we have

$$\mathrm{dist}(\mathbf{v}, \pm\mathbf{B}\{0,1\}^n) = \min\{\|\mathbf{v} - \mathbf{B}\mathbf{g}\| \colon \mathbf{g} \in \pm\{0,1\}^n\} \geq \|\mathbf{v}\| = \mathrm{dist}(\mathbf{v}, \mathbf{0}),$$

i.e., $\mathbf{v}$ is at least as close to the origin as to any lattice point of the form $\pm\mathbf{B}\{0,1\}^n$. Since $\mathbf{B}$ is a lattice basis of Voronoi first kind, by Theorem 2.2 the set $\pm\mathbf{B}\{0,1\}^n$ includes all Voronoi relevant vectors. So, $\mathbf{v}$ belongs to the Voronoi cell of the origin. Equivalently, $\mathbf{t} = \mathbf{v} + (\mathbf{t} - \mathbf{v})$ belongs to the

[10] See also [DB15] for a probabilistic polynomial bound on the number of iterations. Still, the selection of Voronoi relevant vectors in [DB15] remains exponential time in general.

Voronoi cell of $\mathbf{t} - \mathbf{v} \in \mathcal{L}(\mathbf{B})$. This proves that $\mathbf{t} - \mathbf{v}$ is a lattice point closest to the target $\mathbf{t}$, and the algorithm is correct.

Following [MGC14], the running time is analyzed using the following potential function

$$\Delta_{\mathbf{B}}(\mathbf{v}) = \min\{\max\{0, x_1, \ldots, x_n\} - \min\{0, x_1, \ldots, x_n\} \mid \mathbf{Bx} \text{ is a lattice vector closest to } \mathbf{v} \}$$

which we write as $\Delta(\mathbf{v})$ when $\mathbf{B}$ is clear from the context. Notice that

- $\Delta(\mathbf{v})$ is a nonnegative integer, and
- $\Delta(\mathbf{v}) = 0$ if and only if the origin $\mathbf{0}$ is a closest lattice vector to $\mathbf{v}$. So, when $\Delta(\mathbf{v}) = 0$, the length of $\mathbf{v}$ cannot be reduced any further, and the algorithm terminates.

When $\Delta(\mathbf{v}) > 0$, the next lemma shows that the potential function decreases at every iteration. Since $\Delta(\mathbf{v})$ is a nonnegative integer, the value of $\Delta(\mathbf{v})$ right after initialization is an upper bound on the number of iterations performed by the algorithm.

Lemma 3.1 (Adapted from [MGC14, Lemma 5.8]) *For any lattice basis* $\mathbf{B}$ *of Voronoi first kind and target* $\mathbf{v} \in \mathit{span}(\mathbf{B})$ *such that* $\Delta_{\mathbf{B}}(\mathbf{v}) > 0$*, if* $\mathbf{g} \in \pm\{0,1\}^n$ *minimizes* $\|\mathbf{v} - \mathbf{Bg}\|$*, then* $\Delta_{\mathbf{B}}(\mathbf{v} - \mathbf{Bg}) < \Delta_{\mathbf{B}}(\mathbf{v})$.

For completeness, we give a proof of Lemma 3.1 in Appendix A.

In order to bound the total running time, [MGC14] proves that for any $\mathbf{v} \in \mathbf{B}[0,1)^n$, the potential function satisfies $\Delta_{\mathbf{B}}(\mathbf{v}) \leq n$, and gives a reduction from the minimization problem over $\mathbf{g} \in \pm\{0,1\}^n$ to the Min-Cut problem on a graph with $n+3$ nodes (a source s, a sink t, and one additional node for each superbasis vector). Since Min-Cut can be solved in time $O(n^3)$ using network flow algorithms, and the CVP algorithm performs at most n iterations, the total running time is $O(n^4)$, or $O(n^2 \operatorname{polylog} n)$ when exploiting the sparsity of the graph and using better min-cut algorithms [Kar00, Li21, HLRW24].

Here we give a slightly different reduction to Min-Cut and provide a (geometric) method to bound the potential $\Delta_{\mathbf{B}}(\mathbf{v})$, which will allow us to solve CVP on gadget lattices in linear time. Instead of reducing the minimization problem to a single Min-Cut instance with $n+1$ internal nodes, we search over $\mathbf{g} \in \{0,1\}^n$ and $\mathbf{g} \in -\{0,1\}^n$ separately, and define two Min-Cut instances, each with only n internal nodes corresponding to the regular basis vectors. While this is only a minor change, and may even worsen the running time in general due to the doubling of Min-Cut instances, when applied to gadget lattices it results in much nicer graphs for which Min-Cut admits a simple (deferred-choice greedy) solution that runs in linear time.

We describe the reduction to Min-Cut for the case of $\mathbf{g} \in \{0,1\}^n$. (Minimizing over the other set $\mathbf{g} \in \{0,-1\}^n$ easily reduces to $\mathbf{g} \in \{0,1\}^n$ by flipping the sign of the target $-\mathbf{v}$.) In order to minimize $\|\mathbf{v} - \mathbf{Bg}\|$ over $\mathbf{g} \in \{0,1\}^n$, consider the problem of finding a minimum s-t cut in a graph (see Figure 3) with nodes $\{s, t, 1, \ldots, n\}$ and edge weights

1. $w(s,t) = \|\mathbf{v}\|^2$
2. $w(i,j) = -\langle \mathbf{b}_i, \mathbf{b}_j \rangle$ for all $i, j \in \{1, \ldots, n\}$
3. $w(s,i) = 0$ and $w(i,t) = \langle \mathbf{w}, \mathbf{b}_i \rangle$ for all $i \in \{1, \ldots, n\}$, where $\mathbf{w} = \mathbf{Bu} - 2\mathbf{v}$.

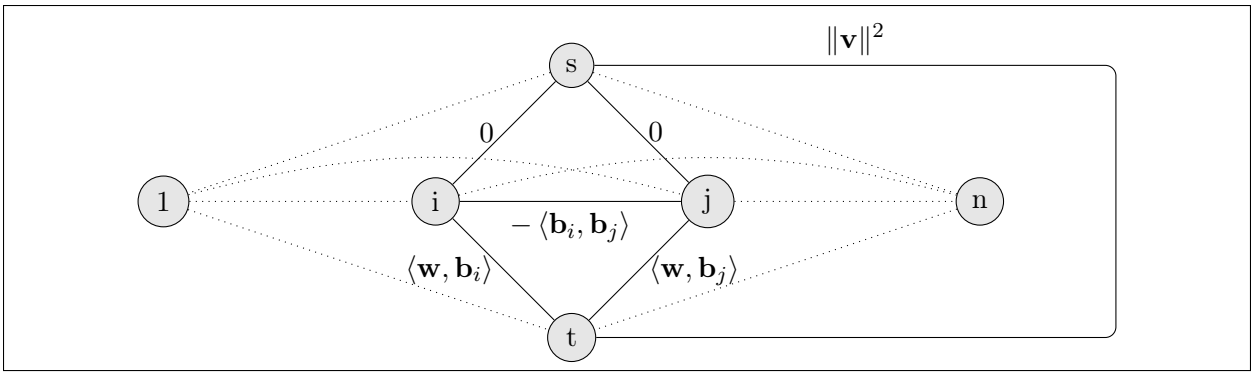

Figure 3: $s-t$ mincut problem for superbasis $\mathbf{B}$ and target $\mathbf{v}$ and $\mathbf{w} = \mathbf{Bu} - 2\mathbf{v}$

Recall that an s-t cut is a set of nodes C such that $s \in C$ and $t \notin C$, and its weight $w(C) = \sum\{w(i,j) \mid i \in C, j \notin C\}$ is the sum of the weights of the edges from C to its complement. In particular, the edge (s,t) is always in the cut, and its only effect is to add the weight $w(s,t) = \|\mathbf{v}\|^2$ to the value of the cut. The following lemma shows that the problem of minimizing $\|\mathbf{v} - \mathbf{Bg}\|$ is equivalent to finding a minimum cut in this graph.

Lemma 3.2 *For any $\mathbf{g} \in \{0,1\}^n$, the weight of the cut $C = \{s\} \cup \{i \mid g_i = 1\}$ in the graph described in Figure 3 equals $w(C) = \|\mathbf{v} - \mathbf{Bg}\|^2$.*

Proof: The cut $(C, \bar{C})$ contains the following edges:

1. the edge (s,t), with weight $w(s,t) = \|\mathbf{v}\|^2$,
2. all edges (i,j) with $g_i \neq g_j$, with total weight $\sum_{g_i=0, g_j=1}(-\langle \mathbf{b}_i, \mathbf{b}_j \rangle) = -\langle \mathbf{B}\bar{\mathbf{g}}, \mathbf{Bg} \rangle$, where $\bar{\mathbf{g}} = \mathbf{u} - \mathbf{g}$,
3. all edges (i,t) such that $g_i = 1$, with total weight $\langle \mathbf{w}, \mathbf{Bg} \rangle = \langle \mathbf{Bu} - 2\mathbf{v}, \mathbf{Bg} \rangle$, and
4. all edges (s,j) such that $g_j = 0$, which have weight 0 and can be ignored.

Adding up the weights of all edges in the cut we get

$$w(C) = \|\mathbf{v}\|^2 + \langle \mathbf{Bu} - 2\mathbf{v}, \mathbf{Bg} \rangle - \langle \mathbf{B}\bar{\mathbf{g}}, \mathbf{Bg} \rangle .$$

Using $\mathbf{u} = \mathbf{g} + \bar{\mathbf{g}}$, the expression for $w(C)$ simplifies to

$$\|\mathbf{v}\|^2 + \langle \mathbf{Bg} - 2\mathbf{v}, \mathbf{Bg} \rangle = \|\mathbf{v}\|^2 - 2\langle \mathbf{v}, \mathbf{Bg} \rangle + \|\mathbf{Bg}\|^2 = \|\mathbf{v} - \mathbf{Bg}\|^2.$$

This proves the lemma. □

For convenience, in Figure 3 we defined a graph where some edges (i,t) may have negative weight. But this is not essential and it is easy to modify the graph into an equivalent one where all weights are nonnegative. To start with, note that the internal edges (i,j) in the graph have nonnegative weight $w(i,j) = -\langle \mathbf{b}_i, \mathbf{b}_j \rangle \geq 0$ because $\mathbf{B}$ is a basis of Voronoi first kind. Other edges (i,t) can be made nonnegative as follows: for any $i = 1, \ldots, n$, if $w(i,t) = \langle \mathbf{w}, \mathbf{b}_i \rangle < 0$, add this weight to $w(s,t)$, and subtract it from $w(s,i)$ and $w(i,t)$, so that $w(s,i) = |\langle \mathbf{w}, \mathbf{b}_i \rangle| > 0$ and

$w(i,t) = 0$. It is immediate to see that this change does not modify the weight of any s-t cut C. This is because for any node $i \in \{1,\ldots,n\}$, the cut C contains either (s,i) or (i,t) (but not both). So, changing $w(s,i)$ and $w(i,t)$ is balanced by the change in $w(s,t)$ which always contributes to the cut. In particular, the minimum cut problem in the modified graph has the same solution and value. Informally, an edge (i,t) with negative weight $w(i,t) < 0$ connecting i to the sink t can be equivalently represented as an edge (s,i) connecting i to the source s. We will make use of this interpretation in some of our proofs. Note that, as a byproduct of these weight changes, the weight of (s,t) may become negative. However, this edge can always be removed from the graph without changing the min-cut solution set, and only modifying the optimal value by an additive constant.

The following theorem gives a geometric method to bound the number of iterations performed by the algorithm.

Theorem 3.1 *For any $c > 0$, if the Voronoi cell of the lattice $\mathcal{V}$ is contained in $\mathbf{B}([\pm c]^n)$, then the CVP algorithm performs at most $\lceil c \rceil + \lfloor c \rfloor \leq 2\lfloor c \rfloor + 1$ iterations.*

Proof: The algorithm maps the target $\mathbf{t}$ to a representative $\mathbf{v} \in \mathbf{t} + \mathcal{L}(\mathbf{B})$ in $\mathbf{B}([0,1)^n)$, i.e., $\mathbf{v} = \mathbf{B}\mathbf{w}$ for some $\mathbf{w} \in [0,1)^n$. Let $\mathbf{B}\mathbf{x}$ be a closest lattice point to $\mathbf{v}$. This vector $\mathbf{B}\mathbf{x}$ belongs to $\mathbf{v} + \mathcal{V}$, where $\mathcal{V}$ is the Voronoi cell of the lattice. It follows that

$$\mathbf{B}\mathbf{x} \in \mathbf{B}\mathbf{w} + \mathcal{V} \subset \mathbf{B}([0,1)^n) + \mathbf{B}([\pm c]^n) = \mathbf{B}([-c, c+1)^n).$$

Since $\mathbf{x}$ is an integer vector, all its coordinates must be in $\{\lceil -c \rceil, \ldots, \lceil c \rceil\}$, and $\Delta_{\mathbf{B}}(\mathbf{v}) \leq \lceil c \rceil - \lceil -c \rceil = \lceil c \rceil + \lfloor c \rfloor$. □

We remark that if the algorithm is modified to map $\mathbf{t}$ to a representative $\mathbf{v} \in \mathbf{B}([\pm 0.5]^n)$ in the centered parallelepiped, then the assumption in the theorem can be relaxed by allowing $\mathcal{V}$ to be contained in the larger region $\mathbf{B}((\pm(c+0.5))^n)$.

4 Solving CVP in parity-check power gadget lattices

Let $\Gamma^{\perp}_{b,n} = \Lambda^{\perp}_q(\mathbf{g}^t)$ be the lattice defined by the power gadget $\mathbf{g} = (1, b, b^2, \ldots, b^{n-1}) \in \mathbb{Z}^n_q$ with base $b \geq 2$ and modulus $q = b^n$. Throughout the section, we fix b, n and $q = b^n$, and omit the subscripts from $\Gamma^{\perp}_{b,n}$ when clear from the context. The main goal of this section is to show that these lattices have a linear time CVP algorithm. Before describing the algorithm, we investigate the geometric properties of the lattices.

Theorem 4.1 *For any $b \geq 2$ and $q = b^n$, the lattice $\Gamma^{\perp}_{b,n}$ has determinant $\det(\Gamma^{\perp}_{b,n}) = q$ and successive minima $\lambda_1 = b$, $\lambda_2 = \ldots = \lambda_n = \sqrt{b^2+1}$ achieved by the basis*

$$\mathbf{B} = b\mathbf{I} - \mathbf{J}^t = \begin{pmatrix} b & & & & \\ -1 & b & & & \\ & -1 & \ddots & & \\ & & \ddots & b & \\ & & & -1 & b \end{pmatrix} \in \mathbb{Z}^{n \times n}. \tag{7}$$

Proof: $\Gamma^\perp = \Lambda_q^\perp(\mathbf{g}^t)$ has determinant q because it is defined by a single equation modulo q and $\mathbf{g}$ is a primitive vector. As observed in [MP12], this lattice has a particularly simple basis (7). It is easy to check that the columns of the matrix $\mathbf{B}$ belong to $\Gamma^\perp$. Moreover, $\mathbf{B}$ is a basis for $\Gamma^\perp$ because $\det(\mathbf{B}) = b^n = \det(\Gamma^\perp)$. These basis vectors show that $\lambda_1 \leq \|\mathbf{b}_n\| = b$ and $\lambda_n \leq \max_{i>0} \|\mathbf{b}_i\| = \sqrt{b^2+1}$. We need to show $\lambda_1 \geq b$ and $\lambda_2 \geq \sqrt{b^2+1}$. Consider an arbitrary, nonzero lattice vector $\mathbf{v} = \mathbf{B}\mathbf{x}$, and let i be the smallest index such that $x_i \neq 0$. Then, the ith coordinate $v_i = b \cdot x_i$ is a nonzero multiple of b, and $\|\mathbf{v}\| \geq b$. This proves $\lambda_1 \geq b$, and the last basis vector achieves $\|\mathbf{b}_n\| = b = \lambda_1$. Now consider any lattice vector $\mathbf{v} = \mathbf{B}\mathbf{x}$ linearly independent from $\mathbf{b}_n$. Then, the smallest index i such that $x_i \neq 0$ necessarily satisfies $i < n$. So, either x_i is a multiple of b and $\|\mathbf{v}\| \geq |v_i| \geq b^2 \geq \sqrt{b^2+1}$, or x_i is not a multiple of b and v_{i+1} is a nonzero integer. In either case $\|\mathbf{v}\| \geq \sqrt{b^2+1}$. So, $\lambda_2 \geq \sqrt{b^2+1}$. □

Note that the basis $\mathbf{B}$ from (7) is obtuse because $\langle \mathbf{b}_i, \mathbf{b}_{i+1} \rangle = -b < 0$ and $\langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0$ for $|i-j| > 1$. Moreover, extending $\mathbf{B}$ with

$$\mathbf{b}_0 = -\sum_{i=1}^{n} \mathbf{b}_i = \begin{pmatrix} -b \\ 1-b \\ 1-b \\ \vdots \\ 1-b \end{pmatrix}$$

gives an obtuse superbasis because

$$\langle \mathbf{b}_i, -\mathbf{b}_0 \rangle = \langle \mathbf{b}_i, \mathbf{b}_{i-1} \rangle + \|\mathbf{b}_i\|^2 + \langle \mathbf{b}_i, \mathbf{b}_{i+1} \rangle \geq b^2 - 2b = b(b-2) \geq 0$$

for all $b \geq 2$. It follows from Theorem 2.2 that the Voronoi cell of $\Gamma^\perp$ can be explicitly described as the intersection of the halfspaces defined by the vectors $\pm\mathbf{B}\{0,1\}^n \setminus \{\mathbf{0}\}$. In fact, a much smaller set of vectors is already sufficient, and the Voronoi cell of $\Gamma^\perp$ has precisely $n(n+1)$ facets, as shown in the next theorem.

Theorem 4.2 *For any $b \geq 2$ and $q = b^n$, the lattice $\Gamma_{b,n}^\perp$ has $n(n+1)$ Voronoi relevant vectors, obtained by adding up consecutive basis vectors $\pm\sum_{i=h}^{k} \mathbf{b}_i$, for $h \leq k$ in $\{1, \ldots, n\}$.*

Proof: We know from Theorem 2.2 that Voronoi relevant vectors are necessarily of the form $\pm\mathbf{B}\mathbf{x}$ where $\mathbf{x} \in \{0,1\}^n$. By Lemma 2.1, the theorem is equivalent to showing that $\pm\mathbf{B}\mathbf{x}$ are the only shortest vectors in $\mathbf{x} + 2\Gamma^\perp$ if and only if $\mathbf{B}\mathbf{x} = \sum_{i=a}^{b} \mathbf{b}_i$, i.e., all the 1s in $\mathbf{x}$ are contiguous. To this end, consider any other vector $\mathbf{B}(\mathbf{x}+2\mathbf{w})$ in the coset, and compute the norm of this vector using Selling's formula (2):

$$\|\mathbf{B}(\mathbf{x}+2\mathbf{w})\|^2 = \sum_{i=1}^{n} (-\langle \mathbf{b}_0, \mathbf{b}_i \rangle)(x_i + 2w_i)^2 + \sum_{i=1}^{n-1} (-\langle \mathbf{b}_i, \mathbf{b}_{i+1} \rangle)(x_i - x_{i+1} + 2w_i - 2w_{i+1})^2$$

where we have omitted the terms with zero coefficient. Using the fact that all remaining coefficients $(-\langle \mathbf{b}_0, \mathbf{b}_i \rangle), (-\langle \mathbf{b}_i, \mathbf{b}_{i+1} \rangle)$ are strictly positive, the norm is minimized by simultaneously minimizing $|x_i + 2w_i|$ and $|x_i - x_{i+1} + 2w_i - 2w_{i+1}|$ for all i. In order to minimize $|x_i + 2w_i|$, all we can do is to change some $x_i = 1$ to $x_i = -1$ by setting $w_i = -1$. Moreover, in order to minimize $|x_i - x_{i+1} + 2w_i - 2w_{i+1}|$, for any two consecutive $x_i = x_{i+1} = 1$, it must be $w_i = w_{i+1}$, i.e., we

can only flip the sign of either both or none of them. Finally, whenever $x_i = 1$ is the first 1 in a sequence (i.e., $x_{i-1} = 0$, and necessarily $w_{i-1} = 0$), we can change $x_i = 1$ to $x_i = -1$ without changing $|x_{i-1} - x_i + 2w_{i-1} - 2w_i| = |x_i + 2w_i|$. (Similarly, also for the last $x_i = 1$ in a sequence.)

In summary, we can flip the sign of any maximal contiguous sequence of 1s in $\mathbf{x}$ while keeping the norm of $\mathbf{Bx}$ minimal. So, $\pm\mathbf{Bx}$ are the unique shortest vectors in $\mathbf{Bx} + 2\Gamma^\perp$ if and only if $\mathbf{x}$ contains a single sequence of contiguous 1s. □

Another consequence of the existence of an obtuse superbasis is that, by [MGC14], CVP in $\Gamma^\perp$ can be solved in polynomial time $O(n^4)$. The main goal of this section is to show that CVP can be solved much faster than that, in $O(n)$ time. This is achieved by showing that the CVP algorithm from Algorithm 1 terminates after only a (very small) constant number of iterations of the main loop, and that each iteration can be implemented in linear time using a specialized deferred-choice greedy technique in place of a general Min-Cut algorithm.

We show that the number of iterations is constant using the geometric method from Theorem 3.1 and Varah's bound (Theorem 2.1).

Lemma 4.1 *The Voronoi region $\mathcal{V}(\Gamma^\perp_{b,n})$ is contained in the parallelepiped $\mathbf{B}[-c_b, +c_b]^n$, for some constant $c_b \le 2.5$. Moreover, $c_3 = 1.25$, and $c_b < 1$ for all $b \ge 4$, with $\lim_{b\to\infty} c_b = 0.5$.*

Proof: Let $\mathbf{v} = \mathbf{Bx}$ be an arbitrary point in the Voronoi region. Then, $\mathbf{v}$ must be in the intersection of the halfspaces $H_{\pm\mathbf{b}_i} = \{\langle \pm 2\mathbf{b}_i, \mathbf{v}\rangle \le \|\mathbf{b}_i\|^2\}$ defined by the basis vectors. (This is a superset of the Voronoi region, but already enough to prove a strong bound on c.) Using matrix notation all these inequalities can be combined into a single equation

$$\pm 2\mathbf{B}^t\mathbf{Bx} \le (\|\mathbf{b}_i\|^2)_i. \tag{8}$$

Let $\mathbf{D}$ be the diagonal matrix with entries $D_{i,i} = \|\mathbf{b}_i\|^2$. If we left-multiply (8) by $\mathbf{D}$ (i.e., divide each row by the right hand side $\|\mathbf{b}_i\|^2$) this equation can be written as a norm inequality

$$\|\mathbf{Ax}\|_\infty \le 1 \qquad \text{where} \quad \mathbf{A} = 2\mathbf{D}^{-1}\mathbf{B}^t\mathbf{B}.$$

Our goal is to show that if $\|\mathbf{Ax}\|_\infty \le 1$ then $\|\mathbf{x}\|_\infty \le c$, i.e., the inverse of matrix $\mathbf{A}$ has operator norm $\|\mathbf{A}^{-1}\|_\infty \le c$. So, let's compute $\mathbf{A}$ explicitly. Substituting the value of $\mathbf{B}$ and performing some calculations gives

$$\mathbf{A} = 2\mathbf{D}^{-1}\mathbf{B}^t\mathbf{B} = \begin{pmatrix} 2 & -2/\beta & & & \\ -2/\beta & 2 & -2/\beta & & \\ & \ddots & \ddots & \ddots & \\ & & -2/\beta & 2 & -2/\beta \\ & & & -2/b & 2 \end{pmatrix}$$

where $\beta = b + \frac{1}{b}$. (Note that $a_{n,n-1} = -2/b$ is different from the other off-diagonal entries $-2/\beta$. The b is not a typo! The entry is different because the last row is divided by $\|\mathbf{b}_n\|^2 = b^2$ rather than $\|\mathbf{b}_{<n}\|^2 = b^2 + 1$.)

The matrix $\mathbf{A}$ is strictly diagonally dominant because

$$\max_i \sum_{j\ne i} |a_{i,j}| = \frac{4}{\beta} < 2 = |a_{i,i}|.$$

So, we can apply Varah's bound (Theorem 2.1), and get that the inverse of $\mathbf{A}$ has operator norm at most

$$\|\mathbf{A}^{-1}\|_\infty \leq \frac{1}{\min_i \left(|a_{i,i}| - \sum_{j\neq i} |a_{i,j}|\right)} = \frac{1}{2 - \frac{4}{\beta}} = c_b.$$

This quantity is monotonically decreasing in β, and $\beta = b + 1/b$ is monotonically increasing in b (for $b \geq 1$). So, we can compute $c_2 = 5/2 = 2.5$, $c_3 = 5/4 = 1.25$, $c_b \leq c_4 = 17/18 < 1$ for all $b \geq 4$, and $\lim_{b\to\infty} c_b = 1/2$. □

Corollary 4.1 *The CVP algorithm on $\Gamma^\perp$ terminates after at most 5 iterations. Moreover, the number of iterations is at most 3 for $b = 3$, and just 1 for $b \geq 4$.*

Proof: Immediately follows from Theorem 3.1 and Lemma 4.1. □

By Lemma 4.1, solving CVP in the gadget lattice requires the solution of a constant number of Min-Cut problems. This, in general requires superlinear time. But the Min-Cut instances associated to lattice $\Gamma^\perp_{b,n}$ are not arbitrary graphs, and have a very special structure that can be exploited to solve the Min-Cut problem very efficiently. Specifically, the internal edges (i, j) have fixed weight $w(i, j) = -\langle \mathbf{b}_i, \mathbf{b}_j \rangle = b$ if $|i-j| = 1$ and $w(i, j) = 0$ otherwise. Then, the Min-Cut problem instance is fully specified by the sequence of signed weights $w_i = w(i, t) - w(s, i) = \langle \mathbf{w}, \mathbf{b}_i \rangle$. In other words, the graph consists of a chain of n nodes, with equal edge weights $w(i, i + 1) = b$, and each node i has an additional edge connecting it either to s or to t (depending on the sign of w_i) with positive weight $|w_i|$. (The edge (s, t) is only used to calibrate the cut value, and can be ignored by the Min-Cut finding algorithm.)

The algorithm processes the nodes $1, \ldots, n$ in order, and for each node i it makes one of the following three choices: (1) include i in the cut; (2) exclude i from the cut, or (3) assign i to the same side of the cut as the following (still unassigned) node $i + 1$. The three alternatives are easily implemented by merging i with one of s, t or $i + 1$:

1. If i is being included in the cut, merge i with s. This moves the weight $w(i, i + 1) = b$ to the edge $(s, i + 1)$, which under our signed representation is implemented by subtracting b from $w_{i+1} = w(i + 1, t) - w(s, i + 1)$

2. If i is being excluded from the cut, merge i with t. This moves the weight $w(i, i + 1) = b$ to the edge $(i + 1, b)$, which under our signed representation is implemented by adding b to $w_{i+1} = w(i + 1, t) - w(s, i + 1)$

3. If i is assigned to the same side of the cut as $i + 1$, merge the nodes i and $i + 1$. This moves the weight $w_i = w(i, t) - w(s, i)$ to the edges $(i + 1, t)$, $(s, i + 1)$, which is implemented simply by adding w_i to w_{i+1}.

It remains to specify how to optimally choose between the three alternatives. This is done according to the following rules. Note that the weights w_i are updated in the process of running the algorithm. So, w_i is not necessarily the initial input value.

1. If $w_i \cdot w_{i+1} \geq 0$ (i.e., if both i and $i + 1$ have edges to the same node s or t), then merge i and $i + 1$

2. If $|w_i| < b$ (i.e., if the cost of cutting i from s, t is smaller than the cost of cutting i from $i+1$), also merge i and $i+1$

3. Otherwise merge i with either s or t based on the sign of w_i: if $w_i < 0$ then i is merged with s, so that the edge (i, t) crosses the cut. Otherwise (if $w_i > 0$) i is merged with t, excluding i from the cut, so that the edge (i, t) does not cross the cut.

For reference, detailed pseudocode implementing this greedy algorithm is given in Algorithm 2. The algorithm determines the output vector $\mathbf{g} \in \{0,1\}^n$ by setting its entries in blocks, left to right. Variable i holds the first position in the current block, and W the total (signed) weight associated to it. Initially, $i = 1$ and $W = w_1$ correspond to the first node. The remaining nodes j are visited one at a time, and at each step, either the current block $\{i, \ldots, j-1\}$ is extended with the new index j, or it is assigned to one or the other side of the cut (depending on the sign of W) and the block is reset to contain just the current node j.

Algorithm 2: Gadget MinCut Algorithm

```
On input: (w_1, ..., w_n) ∈ R^n
i ← 1; W ← w_1;
for j = 2...n do
  if (W · w_j ≥ 0 or |W| < b)
    then W ← W + w_j;
    else if (W < 0)
           then for k = i...j − 1 do
                  g_k ← 1;
                i ← j; W ← w_j − b;
           else for k = i...j − 1 do
                  g_k ← 0;
                i ← j; W ← w_j + b;
if W < 0
  then for k = i...n do
         g_k ← 1
  else for k = i...n do
         g_k ← 0
return (g_1, ..., g_n) ∈ {0,1}^n
```

Lemma 4.2 *Algorithm 2 solves the MinCut problem in linear time.*

Proof: The algorithm clearly runs in linear time $O(n)$, as it processes the nodes in sequence, and for each node only consumes a constant amount of work. All we need to do is to show that the algorithm is correct, i.e., show that the greedy strategy produces an minimum cost cut. By induction we may assume that all choices up to step i are correct, and analyze the correctness of the first step of the algorithm on the modified graph. We need to show that in all cases, the choice made by our selection strategy is locally optimal.

Assume first that $w_i w_{i+1} < 0$, i.e., the weights w_i and w_{i+1} are both nonzero and have different signs. We may assume without loss of generality that $w_i < 0$ and $w_{i+1} > 0$, i.e., the graph has edges (s, i), $(i+1, t)$ and $(i, i+1)$ with positive weights $-w_i$, w_{i+1} and b. (The case where $w_i > 0$

and $w_{i+1} < 0$ is symmetric.) We consider two subcases, depending on whether an optimal cut (consistent with all previous choices) puts node $i+1$ on the same side as s or t:

- If $i+1$ is merged with s, then our selection strategy merges i with s as well, either because $|w_i| < b$ or because $w_i < 0$. This is the optimal choice because merging i with t would cut the edges $(i, i+1)$ and (s, i), increasing the cost of the cut by $b + |w_i| > 0$.
- If $i+1$ is merged with t, then merging i with s (resp. t) would cut the edge $(i, i+1)$ with additional cost b (resp. (s, i) with additional cost $|w_i|$). So, the optimal strategy should put i on the same side as $i+1$ when $|w_i| < b$.

This proves that our algorithm is correct when w_i and w_{i+1} have opposing signs.

It remains to consider the case where $w_i \cdot w_{i+1} \geq 0$. Again, we may assume without loss of generality that $w_i, w_{i+1} \geq 0$, i.e., the graph has edges $(i, t), (i+1, t)$ and $(i, i+1)$ with weights w_i, w_{i+1} and b. Our strategy always puts i on the same side of the cut as $i+1$. Assume for contradiction that all optimal solutions put i and $i+1$ on different sides of the cut. There are two possibilities:

- If i is merged with s and $i+1$ is merged with t, then moving i to the same side of $i+1$ reduces the cost of the cut by $w(i, t) + w(i, i+1) = w_i + b > 0$. So, this cannot be an optimal solution.
- If i is merged with t and $i+1$ is merged with s, then moving $i+1$ to the same side as i saves the cost of cutting $w(i, i+1) = b$ and $w(i+1, t) = w_{i+1}$. On the other hand this may raise the cost by at most b by cutting $(i+1, i+2)$ (if $i \leq n-2$ and the optimal solution merges $i+2$ with s). Still the total saving is at least $b + w_{i+1} - b = w_{i+1} \geq 0$. So, putting i and $i+1$ on the same side of the cut is still consistent with some other optimal solution.

This completes the proof that the Greedy algorithm correctly solves MinCut in the graphs defined by the gadget lattice $\Gamma^{\perp}$. □

Combining Lemma 4.2 with Corollary 4.1 gives a linear time solution to CVP on gadget lattice $\Gamma^{\perp}_{b,n}$.

Theorem 4.3 *The closest vector problem in $\Gamma^{\perp}_{b,n}$ can be solved in linear time.*

Comparison with standard algorithms. We conclude this section by comparing our linear time CVP solver to other standard approaches to finding close lattice points (specialized to gadget lattices), which match our algorithm either in terms of running time or output quality:

1. The linear time BDD algorithm of [MP12], which is an optimized variant of Babai's nearest plane algorithm [Bab86] specialized to the sparse triangular structure of (7). This algorithm is just as fast as ours, even when used on arbitrary target points. However, we show that it only achieves a rather poor $\gamma(n) = \sqrt{n}$ worst-case CVP approximation factor.
2. The enumeration (also known as "sphere decoding") algorithm of [FP85, SE94] (in its many variants) for the exact solution of CVP. Similarly to [MP12], also this algorithm can be specialized to the sparse triangular structure of (7), achieving practically significant polynomial speedups. However, the main issue is that the complexity of enumeration algorithms is generally exponential in the lattice dimension n. Still, one may wonder if pruned enumeration techniques reduce the exponential search to a polynomial when the algorithm is applied to a highly structured basis like (7) with excellent geometric properties. We show that even in the very special case of gadget lattices, pruned enumeration has exponential complexity $2^{\Omega(n)}$.

We remark that, as shown in Theorem 4.1, the basis (7) used by these algorithms is optimal, in the sense that it simultaneously achieves all successive minima of the lattice. So, this is precisely the basis one would use to optimize the performance of nearest plane and enumeration algorithms.

Theorem 4.4 *The BDD algorithm [MP12] for the power gadget lattice $\Gamma^{\perp}_{2,n}$ approximates CVP within a factor $\gamma = \sqrt{n}$. More specifically, it always returns a lattice point $\mathbf{v}$ within distance $\|\mathbf{v} - \mathbf{t}\| \leq \sqrt{n} \cdot \mu(\mathbf{t}, \Lambda)$ from any target vector $\mathbf{t}$, and, for any $\gamma < \sqrt{n}$, there is a target $\mathbf{t}$ such that the lattice point $\mathbf{v}$ produced by the BDD algorithm is at distance $\|\mathbf{v} - \mathbf{t}\| \geq \gamma \cdot \mu(\mathbf{t}, \Lambda)$ from the target.*

Proof: We recall that the BDD algorithm of [MP12] is just a special case of Babai's nearest plane algorithm, optimized using the special (sparse, lower triangular) structure of basis (7).

First we observe that the algorithm solves approximate CVP within a factor $\gamma(n) = \sqrt{n}$ because it always finds a lattice vector within distance $\sqrt{n}$ from the target, and when the target is at distance less than 1 from the lattice it outputs the unique closest vector. We need to show that this bound is tight.

Consider the target vector $\mathbf{t} \in \mathbb{R}^n$ with coordinates $t_1 = 1 + \epsilon$ and $t_i = \epsilon$ for $i > 1$. Clearly, the distance to the lattice is (at most) $\mu = \sqrt{(1+\epsilon)^2 + (n-1)\epsilon^2}$. However, the nearest plane algorithm [MP12] will greedily map all the coordinates to $\epsilon - 1$ to produce a point in the orthogonalized parallelepiped $[\pm 1]^n$. So, it produces a lattice point at distance $\sqrt{n}(1-\epsilon)$ from the target. Taking the limit for $\epsilon \to 0$, this distance tends to $\lim_{\epsilon \to 0} \sqrt{n}(1-\epsilon) = \sqrt{n}$, while the true distance tends to $\lim_{\epsilon \to 0} \mu = 1$. So, by taking ϵ small enough, the approximation factor can be made arbitrarily close to $\sqrt{n}$. □

Another way to understand the limitations of the algorithm of [MP12] is to observe that as soon as the distance from the lattice goes beyond the BDD decoding radius, the algorithm may already fail to solve CVP. This easily follows from our characterization of the Voronoi cell in Theorem 4.2.

Corollary 4.2 *For any $\epsilon > 0$ there are points $\mathbf{t}$ at distance $\lambda_1/2 + \epsilon$ from the lattice for which the Algorithm of [MP12] fails to solve CVP on gadget lattice $\Gamma^{\perp}_{2,n}$.*

Proof: Consider the target point $\mathbf{t} = (\epsilon + b/2, 0, \ldots, 0)$ for any sufficiently small $\epsilon < 1/(2b)$. This point is closer to the origin than to any of the Voronoi relevant vectors listed in Theorem 4.2. So, it belongs to the Voronoi cell of the origin. Still the nearest plane algorithm subtracts the first basis vector $\mathbf{b}_1$ from it to make the first coordinate less than $b/2$ is absolute value. So, it returns an incorrect CVP solution. □

Interestingly, from the characterization of the Voronoi cell in Theorem 4.2 one can also see that the algorithm of [MP12] correctly solves CVP in the special setting where the base b is odd and the target $\mathbf{t} \in \mathbb{Z}^n$ has integer coordinates.

Corollary 4.3 *For any odd integer base b, and integer target $\mathbf{t} \in \mathbb{Z}^n$, the algorithm of [MP12] finds the lattice vector in the gadget lattice $\Gamma^{\perp}_{b,n}$ closest to $\mathbf{t}$.*

Proof: Assume without loss of generality that the target belongs to the fundamental region $[\pm b/2]^n$, and note that if $\mathbf{t} \in \mathbb{Z}^n$, then it belongs to the strictly smaller region $[\pm c]^n$ for $c = (b-1)/2$. It can be easily checked that all the vertices $\{\pm c\}^n$ of this region belongs to the Voronoi cell of the origin, simply by comparing their norm with the distance to the Voronoi relevant vectors listed in

Theorem 4.2. By convexity, the entire region $[\pm c]^n$ is contained in the Voronoi cell of the origin. In particular, any integer target $\mathbf{t} \in [\pm b/2]^n \cap \mathbb{Z}^n \subset [\pm c]^n$ is correclty mapped to the origin as its closest lattice point. □

Still, the BDD algorithm of [MP12] fails to solve CVP when b is even or when the target has non integer coordinates. So, in these settings one may consider solving CVP using enumeration algorithms, possibly specialized to the gadget lattice $\Gamma^{\perp}_{2,n}$. However this may take exponential time. The fact that enumeration algorithms can be exponential time even for very simple and highly structured lattices shouldn't be much of a surprise: consider for example the integer lattice $\mathbb{Z}^n$ with target vector $\mathbf{t} = (0.5, \ldots, 0.5)$. Since CVP has 2^n equally good solutions (the vertexes of the hypercube $\{0,1\}^n$), enumeration algorithms will necessarily explore all of them. However, proving an exponential lower bound for our gadget lattices is not as trivial. The next theorem provides a counterexample for which CVP has a unique solution, and still enumeration takes time $2^{\Omega(n)}$.

Theorem 4.5 *Solving CVP on gadget lattice $\Gamma^{\perp}_{2,n}$ using the enumeration (sphere decoding) algorithms [FP85, SE94] has complexity $2^{\Omega(n)}$, even when using the optimal gadget lattice basis (7), pruned enumeration with dynamic radius updates, and a target for which the closest lattice point is unique.*

Proof: When specialized to the triangular gadget basis (7), enumeration algorithms can be described as a generalization of [MP12] as follows. In [MP12], a lattice point $\sum_i \mathbf{b}_i x_i$ close to a target $\mathbf{t}$ is found using a greedy strategy that sequentially chooses each integer coefficient x_i to minimize the contribution of coordinate $(t_i + x_{i-1}) - 2x_i$ to the squared distance. Enumeration algorithms, instead of selecting a single value for x_i, consider all possible choices of x_i for which the partial sum $\sum_{j \le i} |(t_j + x_{i-1}) - 2x_j|^2$ is below a known upper bound ρ^2 on the optimal squared distance. In [SE94] this strategy is optimized by initially setting ρ^2 to the solution provided by [Bab86, MP12], and dynamically updating this bound each time a better solution is found. The specific solution proposed in [SE94] adopts a depth first tree traversal and considers the possible values for x_i in order of increasing contribution $|(t_i + x_{i-1}) - 2x_j|^2$, which is very effective in practice. Our exponential lower bound applies to any traversal strategy.

Consider the optimal basis (7) and target $\mathbf{t} = (0.5, \ldots, 0.5)$. The unique lattice point closest to the target is the origin because each coordinate contributes at least 0.5^2 to the squared distance, and any point other than the origin contributes at least 1.5^2 in some coordinate.

Since the target is at distance $\rho = \sqrt{n}/2$ from the lattice, no traversal strategy will ever reduce the search bound below $\rho^2 = n/4$. In order to get a lower bound, consider the subtree containing (at most) two children per node, corresponding to the best values of coefficient x_i. Since t_i is a half integer, the optimal values of $|t_i + x_{i-1}) - 2 \cdot x_i|$ are 0.5 and 1.5. Either choice will contribute at most $(1.5)^2 = 9/4$ to the squared distance. So, before these contributions add up to the pruning bound $\rho^2 = n/4$, we need to reach depth $h = \rho^2/(1.5)^2 = n/9$. This shows that pruned enumeration, for any traversal strategy, will explore at least all $2^{n/9} - 1$ nodes in a full binary tree of height $n/9$, corresponding to the first h levels of the search tree. □

5 Solving CVP in generator power gadget lattices

Let $\Gamma_{b,n} = \Lambda_q(\mathbf{g}^t) \subset \mathbb{Z}^n$ be generator power gadget lattice, i.e., the (scaled) dual of the parity-check gadget lattice $\Gamma^{\perp}_{b,n} = \Lambda^{\perp}_q(\mathbf{g}^t)$ where $q = b^n$ and $\mathbf{g} = (1, b, b^2, \ldots, b^{n-1}) \in \mathbb{Z}^n$. As usual,

we omit the subscripts from $\Gamma_{b,n}$ when clear from the context. By definition, Γ is generated by the vectors $\mathbf{g}, q\mathbf{e}_1, \ldots, q\mathbf{e}_n$. Moreover, the matrix $[\mathbf{g}, q\mathbf{e}_2, \ldots, q\mathbf{e}_n] \in \mathbb{Z}^{n\times n}$ is a basis for Γ because $q\mathbf{e}_1 = q\mathbf{g} - \sum_{i>1} q^{i-1} \cdot (q\mathbf{e}_i)$ can be expressed as a combination of the other vectors. However, this is not a good basis for the lattice, in the sense that its columns are much longer than the successive minima λ_i. A better basis is given by the matrix

$$\mathbf{D} = \begin{pmatrix} b^{n-1} & b^{n-2} & \cdots & b^1 & b^0 \\ & b^{n-1} & \ddots & \ddots & b^1 \\ & & \ddots & & \vdots \\ & & & b^{n-1} & b^{n-2} \\ & & & & b^{n-1} \end{pmatrix} \tag{9}$$

with columns $\mathbf{d}_i = (b^{n-i} \cdot \mathbf{g}) \bmod q \in \{0, \ldots, q-1\}^n$. Note that this basis is precisely the scaled dual $q\mathbf{B}^{-t}$ of the $\Gamma^\perp$ basis $\mathbf{B} = b\mathbf{I} - \mathbf{J}^t$ from (7), i.e., $\mathbf{B}^t\mathbf{D} = q\mathbf{I}$. The next theorem shows that the successive minima of Γ are achieved by this basis.

Theorem 5.1 *For any $b \geq 2$ and $q = b^n$, the generator gadget lattice $\Gamma_{b,n}$ has determinant $\det(\Gamma_{b,n}) = q^{n-1}$, basis $\mathbf{D} = [(b^{n-i} \cdot \mathbf{g}) \bmod q \colon i = 1, \ldots, n]$ and successive minima*

$$\lambda_k = \|\mathbf{d}_k\| = q\sqrt{\frac{1 - \frac{1}{b^{2k}}}{b^2 - 1}} \qquad \textit{for} \quad k = 1, \ldots, n.$$

Proof: Using the duality relation $\Gamma^\perp = q\Gamma^*$, we get $\det(\Gamma^\perp) = q^n \det(\Gamma^*) = q^n / \det(\Gamma)$. So, using $\det(\Gamma^\perp) = q$ from Theorem 4.1, we get $\det(\Gamma) = q^n / \det(\Gamma^\perp) = q^{n-1}$.

The columns of $\mathbf{D}$ belong to the lattice Γ because they are integer multiples of $\mathbf{g} \in \Gamma$, reduced modulo q, and $q\mathbb{Z}^n \subset \Gamma$. The matrix $\mathbf{D}$ is a basis because it has determinant $\det(\mathbf{D}) = (b^{n-1})^n = q^{n-1} = \det(\Gamma)$. So, it generates the entire lattice.

The squared lengths of the basis vectors are easily computed as

$$\|\mathbf{d}_k\|^2 = q^2 \sum_{i=1}^{k} b^{-2i} = q^2 \frac{1 - b^{-2k}}{b^2 - 1}$$

and form a monotonically increasing sequence. So, they provide upper bounds on the successive minima $\lambda_k(\Gamma) \leq \|\mathbf{d}_k\|$. We need to prove that these bounds are tight, i.e., $\lambda_k(\Gamma) \geq \|\mathbf{d}_k\|$ for $k = 1, \ldots, n$.

For any k, any set of k linearly independent lattice vectors must contain at least one vector $\mathbf{v} = \mathbf{D}\mathbf{x}$ with $x_i \neq 0$ for some $i \geq k$. Let $h \geq k$ be the largest index such that $x_h \neq 0$. Then, $v_h = x_h b^{n-1}$ is a nonzero integer multiple of b^{n-1}. If x_h is a multiple of b, then

$$\|\mathbf{v}\| \geq |v_h| \geq b^n = 1 + (b-1)\sum_{i<n} b^i > \sqrt{\sum_{i=0}^{n-1} b^{2i}}.$$

So, assume x_h is not a multiple of b. Then, for all $i \geq h$, $b^{n-h-1+i}$ is the largest power of b that divides v_i. So, $|v_i| \geq b^{n-h-1+i}$ and $\|\mathbf{v}\| \geq \sqrt{\sum_{i=1}^{h} b^{2(n-i)}}$. $\square$

Next we turn our attention to CVP in the generator gadget lattice Γ. Note that neither $[\mathbf{g}, q\mathbf{e}_2, \ldots, q\mathbf{e}_n]$ nor $\mathbf{D}$ give an obtuse superbasis. Simple calculations also show that changing the sign of any subset of basis vectors still does not give an obtuse superbasis. So, it is not immediately clear if Γ is a lattice of Voronoi first kind. The next theorem shows that this is indeed the case, and gives an explicit construction of an obtuse superbasis.

Theorem 5.2 *For any $b \geq 2$ and n, generator gadget lattice $\Gamma_{b,n} = \Lambda_q(\mathbf{g}^t)$ with $q = b^n$ and $\mathbf{g} = (1, b, b^2, \ldots, b^{n-1})$ is a lattice of Voronoi first kind with obtuse superbasis $[-\mathbf{g}, \mathbf{V}]$ where*

$$\mathbf{V} = q\mathbf{I} - (b-1)\mathbf{D} = \begin{pmatrix} b^{n-1} & (1-b)b^{n-2} & \cdots & (1-b)b^1 & (1-b)b^0 \\ & b^{n-1} & \ddots & \ddots & (1-b)b^1 \\ & & \ddots & & \vdots \\ & & & b^{n-1} & (1-b)b^{n-2} \\ & & & & b^{n-1} \end{pmatrix}.$$

Moreover, the set of (strict) Voronoi relevant vectors is $\pm\mathbf{V}\{0,1\}^n \setminus \{0\}^n$. In particular, the Voronoi cell has the maximal number of facets $2(2^n - 1)$.

Proof: First notice that all columns of $\mathbf{V}$ belong to the lattice Γ because they are integer linear combinations of $\mathbf{d}_i, q\mathbf{e}_i \in \Gamma$. This matrix is a basis because $\det(\mathbf{V}) = (b^{n-1})^n = q^{n-1} = \det(\Gamma)$, so it generates the whole lattice.

Next, we observe that the sum $\mathbf{Vu} = \sum_i \mathbf{v}_i$ of all basis vectors equals the gadget vector $\mathbf{g}$. To see this, first notice that the ith coordinate of $(b-1)\mathbf{Du}$ is

$$[(b-1)\mathbf{Du}]_i = \sum_{j=i-1}^{n-1} b^j(b-1) = \sum_{j=i}^{n} b^j - \sum_{j=i-1}^{n-1} b^j = b^n - b^{i-1} = q - g_i.$$

So, $(b-1)\mathbf{Du} = q\mathbf{u} - \mathbf{g}$. It follows that

$$\mathbf{Vu} = (q\mathbf{I} - (b-1)\mathbf{D})\mathbf{u} = q\mathbf{u} - (q\mathbf{u} - \mathbf{g}) = \mathbf{g}$$

and $[\mathbf{g}, \mathbf{V}]$ is a superbasis.

So, in order to prove that $\mathbf{V}$ is a basis of Voronoi first kind, we have to show that $\langle \mathbf{v}_i, \mathbf{v}_j \rangle \leq 0$ for all $i < j$, and $\langle \mathbf{v}_i, \mathbf{g} \rangle \geq 0$. We begin with the second inequality, and compute

$$\begin{aligned} \langle \mathbf{v}_i, \mathbf{g} \rangle &= q \langle \mathbf{e}_i, \mathbf{g} \rangle - (b-1) \langle \mathbf{d}_i, \mathbf{g} \rangle \\ &= qb^{i-1} - (b-1) \sum_{h=0}^{i-1} b^{n-i+2h} \\ &= qb^{i-1} - (b-1)b^{n-i} \frac{b^{2i}-1}{b^2-1} \\ &= q(b^{i-1} - \frac{b^i - b^{-i}}{b+1}) \\ &= q\frac{b^{i-1} + b^{-i}}{b+1} \geq 0. \end{aligned}$$

Finally, for all $i < j$, note that the first i coordinates of $\mathbf{v}_j$ are the same as $(1-b)b^{n-j}\mathbf{g}$, and $\mathbf{v}_i$ is zero in the remaining coordinates. Therefore,

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle = -(b-1)b^{n-j} \langle \mathbf{v}_i, \mathbf{g} \rangle \leq 0.$$

This proves that $[-\mathbf{g}, \mathbf{V}]$ is an obtuse superbasis, and Γ is a lattice of Voronoi first kind.

By Theorem 2.2 Voronoi relevant vectors are necessarily of the form $\pm\mathbf{V}\{0,1\}^n - \{0\}^n$. They are all strictly relevant because their scalar products are strictly negative. □

Corollary 5.1 *For any $b \geq 2$ and n, CVP in the generator gadget lattice $\Gamma_{b,n}$ can be solved in polynomial time $O(n^3 \log^3 n)$.*

Proof: Immediately follows from Theorem 5.2 and the algorithm of [MGC14], using the latest "almost linear time" min-cut algorithm from [Li21, HLRW24]. □

We note that the CVP algorithm of Corollary 5.1—while polynomial time—is not entirely practical, due both to its superlinear running time, and difficulty of implementation, requiring the use of a general Min-Cut algorithm. Still, as a theoretical result, it is a significant instantiation of the MGC14 framework, as explained in the following remark.

Remark 5.1 *We highlight that while the algorithm of [MGC14] established that CVP is solvable in polynomial time for all lattices of Voronoi's first kind, virtually all previously known explicit, named lattice families in this class—such as the root lattices A_n, A_n^* and the dual checkerboard lattice D_n^*—already possessed faster, specialized $O(n)$ or $O(n \log n)$ decoders [CS99, MCQ08]. To the best of our knowledge, the generator power gadget lattice $\Lambda_q(\mathbf{g}^t)$ represents the first explicitly named, cryptographically central family of lattices for which the obtuse superbasis framework of [MGC14] yields the* primary and only known *polynomial-time CVP algorithm.*

6 Linear time CVP for CRT gadgets

By the Chinese Remainder Theorem (CRT), for any sequence of relatively prime integers $q_1, \ldots, q_n$ (with $\gcd(q_i, q_j) = 1$ for $i \neq j$) there exists a (unique) vector $\mathbf{g} \in \mathbb{Z}_q^n$ modulo $q = \prod_i q_i$ such that $g_i \pmod{q_i} = 1$ and $g_i \pmod{q_j} = 0$ for all $i \neq j$. This $\mathbf{g}$ is called the CRT (also known as RNS, for Residue Number System) gadget, and it is often used in practice as an alternative to the power gadget $(b^0, \ldots, b^{n-1})$ in the implementation of lattice cryptography to avoid the use of large integer arithmetics.

Common tasks in cryptography (called gadget decomposition and gadget decoding operations[11]) correspond to the solution of CVP instances on gadget lattices $\Phi_q = \Lambda_q(\mathbf{g}^t)$ and $\Phi_q^\perp = \Lambda_q^\perp(\mathbf{g}^t)$. Here we observe that the existence of efficient algorithms for these common tasks can be given a very natural, geometric explanation: the CRT gadget lattices have an orthogonal (in fact, diagonal) basis.

[11] Gadget decoding is the problem of finding a multiple of $\mathbf{g}$ closest to a given target $\mathbf{t}$ modulo q, and it corresponds to solving CVP on $\Lambda_q(\mathbf{g}^t)$. Gadget decomposition asks for a short vector $\mathbf{x}$ such that $\langle \mathbf{g}, \mathbf{x} \rangle = t$ for a given target value $t \in \mathbb{Z}_q$, and corresponds to the "syndrome decoding" version of CVP on lattice $\Lambda_q^\perp(\mathbf{g}^t)$.

Theorem 6.1 *The generator CRT gadget lattice $\Phi = \Lambda_q(\mathbf{g}^t)$ with modulus $q = \prod_i q_i$ is generated by the basis*

$$\mathbf{B} = [\lambda_1 \mathbf{e}_1, \ldots, \lambda_n \mathbf{e}_n] \qquad \textit{where} \quad \lambda_i = \prod_{j \neq i} q_j = \frac{q}{q_i}.$$

In particular (assuming $q_1 \geq q_2 \geq \cdots \geq q_n$ are sorted in decreasing order) Φ has successive minima $\lambda_1, \ldots, \lambda_n$, determinant $\Delta(\Phi) = q/\prod_i q_i = q^{n-1}$, $2n$ Voronoi relevant vectors $\pm\lambda_i \mathbf{e}_i$, and a simple linear time CVP algorithm that rounds each coordinate of the target to the closest multiple of λ_i.

Proof: Note that by the Chinese Remainder Theorem, the coordinates of the CRT gadget are given by $g_i = \lambda_i \cdot (\lambda_i^{-1} \bmod q_i) \in \mathbb{Z}_q$. We need to show that $\mathbf{B}\mathbb{Z}^n$ and $\Phi = \mathbf{g}\mathbb{Z} + q\mathbb{Z}^n$ are the same lattice. In one direction, any basis vector $\mathbf{b}_i = \lambda_i \mathbf{e}_i$ can be written as $\mathbf{g} \cdot \lambda_i \pmod q$. So it belongs to Φ, and $\mathbf{B}\mathbb{Z}^n \subseteq \Phi$. In the other direction, first notice that $q\mathbf{e}_i$ can be obtained by multiplying $\mathbf{b}_i$ by λ_i. So, $q\mathbb{Z}^n$ is a subset of $\mathbf{B}\mathbb{Z}^n$. Finally, $\mathbf{g}$ is a combination of the vectors in $\mathbf{B}$ with integer coefficients $(\lambda_i^{-1} \bmod q_i)$. So, $\mathbf{g}$ is also in $\mathbf{B}\mathbb{Z}^n$. This proves $\Phi = \mathbf{g}\mathbb{Z} + q\mathbb{Z}^n \subseteq \mathbf{B}\mathbb{Z}^n$. CVP is easily solved in linear time by rounding each coordinate of the target to the closest multiple of λ_i. □

Corollary 6.1 *The CRT parity-check gadget lattice $\Phi^\perp = \Lambda_q^\perp(\mathbf{g}^t)$ with modulus $q = \prod_i q_i$ is generated by the basis*

$$\mathbf{B} = [q_1 \mathbf{e}_1, \ldots, q_n \mathbf{e}_n].$$

In particular (assuming $q_1 \leq q_2 \leq \cdots \leq q_n$ are sorted in increasing order) Φ has successive minima $\lambda_i = q_i$, determinant $\Delta(\Phi^\perp) = q$, $2n$ Voronoi relevant vectors $\pm q_i \mathbf{e}_i$, and a linear time CVP algorithm that rounds each coordinate of the target to the closest multiple of q_i.

Proof: Immediately follows from the duality of $\frac{1}{q}\Phi$ and $\Phi^\perp$. □

As a side remark, we note that since Φ and $\Phi^\perp$ have orthogonal bases, rounding each coordinate of the target solves CVP simultaneously in the ℓ_2 and ℓ_∞ (in fact, any ℓ_p) norm.

7 Implementation and Experiments

In order to assess the practicality of our new algorithm, we wrote a quick prototype implementation in the high level functional programming language Haskell. More than demonstrating raw performance, our goal was to illustrate the easiness of implementation, and the linear scaling of the running time with the lattice dimension. The Haskell programming language proved to be more than adequate for both tasks. We implemented the algorithm from scratch, without using any external library, and with a simplistic implementation of vectors as linked lists. The full implementation, including the deferred greedy min-cut solver (Algorithm 2), and main loop (Algorithm 1) is less than 30 lines of code, essentially the same as the pseudocode listings included in this paper. The code was compiled with `ghc` compiler version 9.6.6, and standard optimization flag `-O2`, and then used to perform the following experiments. All experiments were performed on a desktop computer with a (2014) Intel Core i7-4790 processor running at 3.60GHz.

For a given base b and dimension n, each test generates 1000 random target vectors in $[0, b)^n$, and then runs our algorithm on them. This is a fundamental region for lattice $\Lambda_q^\perp(\mathbf{g})$, so that the targets are distributed uniformly modulo the lattice. We ran the algorithm for base $b = 2$ (powers-of-two gadget) and $b = 5$, and dimension $n = 10, \ldots, 1000$, in increments of 10. The running time

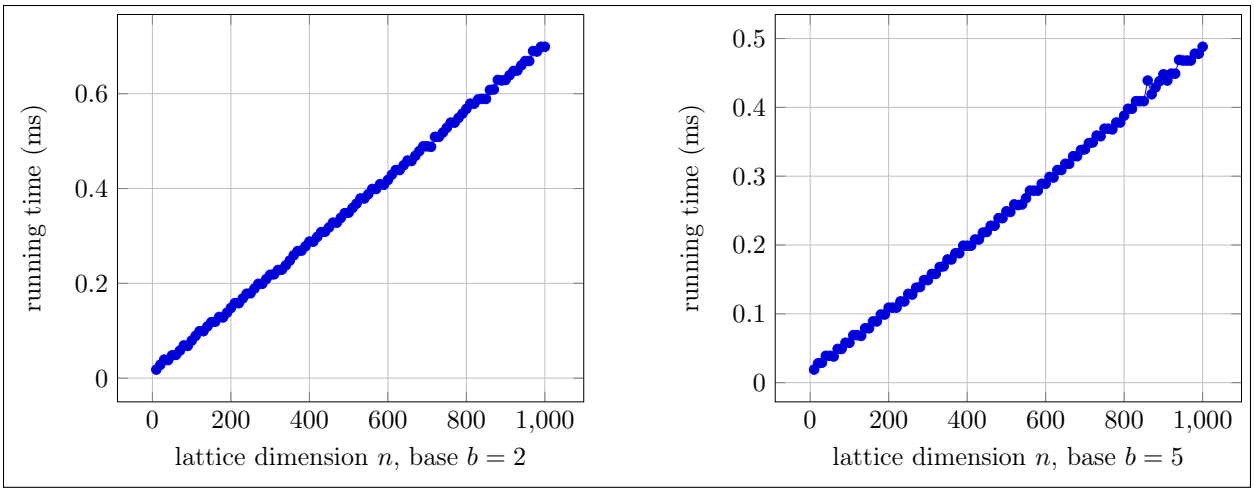

Figure 4: Running time of our CVP algorithm for the power gadget lattice $\Lambda_q^{\perp}(\mathbf{g})$ in dimension $n = 10, \ldots, 1000$ for base $b = 2$ (left) and $b = 5$ (right). Each point is the total running time in seconds for 1000 target vectors randomly selected in a fundamental region of the lattice, or, equivalently, the average running time in milliseconds (ms).

of the algorithm, as a function of the dimension, is shown in Figure 4, showing an almost perfect linear scaling of the running time as a function of n, and essentially independent of the base b, with minor fluctuations in running time probably due to garbage collection. The slightly better running time for $b = 5$ is consistent with our theoretical analysis (Corollary 4.1) showing that the maximum number of iterations is a monotonically decreasing function of b. We also ran experiments with larger values of b, producing running times essentially equal to those for $b = 5$.

In another experiment we also ran the BDD algorithm of [MP12] (also implemented from scratch in less than 10 lines of Haskell code), and compared its output quality with that of our CVP algorithm on two types of input, this time just for the powers-of-two gadget ($b = 2$.) Figure 5 (left) shows the approximation factor achieved by the BDD algorithm on the counterexample vectors of Theorem 4.4, providing an empirical confirmation that the worst-case approximation factor is $\sqrt{n}$. Figure 5 (right) shows the fraction of (1000 in each dimension) target vectors for which the BDD algorithm of [MP12] finds the closest vector, matching the performance of the CVP algorithm. The experiments show that the probability with which the BDD algorithm produces correct CVP solutions on random target quickly drops to 0 as the dimension increases.

8 Conclusion and Open Problems

We have presented efficient CVP algorithms for prominent families of gadget lattices used in cryptography and discrete optimization: the parity-check gadget lattices $\Lambda_q^{\perp}(\mathbf{g}^t)$ and generator gadget lattices $\Lambda_q(\mathbf{g}^t)$ associated with both power gadgets and CRT gadgets. Except for the generator power gadget lattice, all our CVP algorithms run in optimal linear time $O(n)$. For the generator power gadget lattice, we established that it is a lattice of Voronoi's first kind, yielding a polynomial-time algorithm running in $\tilde{O}(n^3)$ time by applying the min-cut framework of [MGC14] as a black box. We conjecture that exact CVP on generator power gadget lattices also admits an $O(n)$ linear-time solver, and leave explicit linear-time optimization for this case as an open problem.

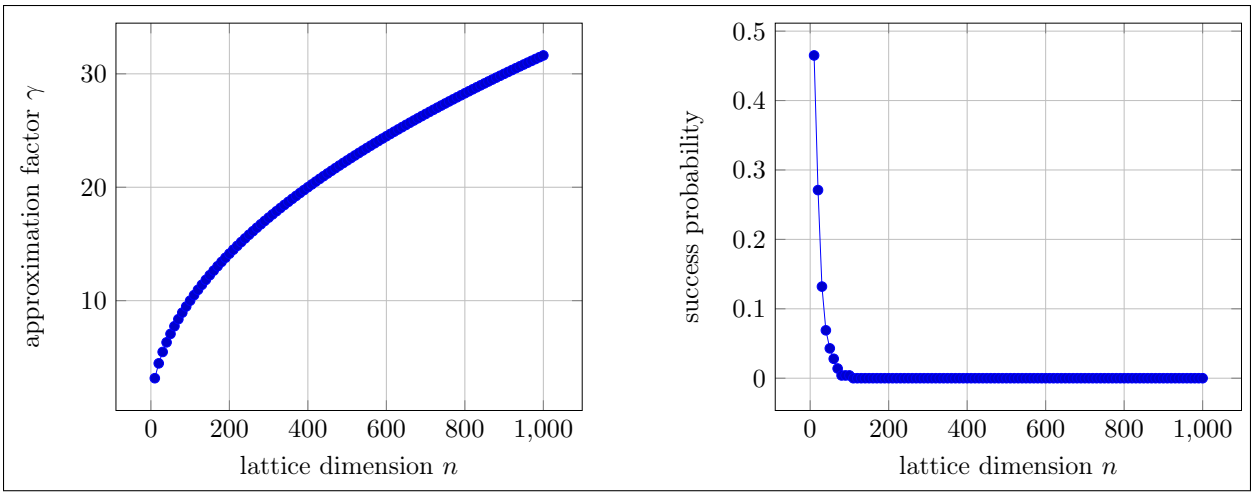

Figure 5: Approximation factor (left) and exact solution success probability (right) of the BDD algorithm for lattice $\Lambda_q^\perp(\mathbf{g})$ with powers-of-two gadget $\mathbf{g} = (1, 2, 4, \ldots)$, compared to our (exact) CVP solver. *(Left:)* the algorithms are run on the worst-case target vectors from Theorem 4.4, and the graph shows the ratio between the distance achieved by the BDD algorithm and that achieved by our (optimal) CVP algorithm, confirming that the approximation factor is $\sqrt{n}$. *(Right:)* Both algorithms are run on 1000 random targets per dimension, measuring the fraction of inputs for which the BDD algorithm matches the (optimal) distance achieved by our CVP exact solver.

The algorithmic tools and geometric insights developed in this work can be modularly combined with existing lattice decoders to yield efficient CVP algorithms for broader classes of lattices:

- **Hybrid Power-CRT Gadgets:** The diagonal (orthogonal) structure of CRT gadget lattices allows one to seamlessly combine CRT and power gadgets. Specifically, for a composite modulus $q = \prod_i q_i$ with prime-power factors $q_i = b_i^{e_i}$, replacing each scalar component g_i with a power gadget $(b_i^0, \ldots, b_i^{e_i - 1})$ yields a "hybrid" gadget lattice in dimension $n = \sum_i e_i$. Because the CRT components decouple orthogonally, applying our CVP algorithms independently on each block yields linear or polynomial-time decoders for the combined high-dimensional lattice.

- **Algebraic and Ring Gadget Lattices:** In lattice-based cryptography, gadgets are frequently instantiated over ring of integers of algebraic number fields. Due to the orthogonal structure of CRT gadgets, the resulting ring gadget lattice decomposes as the tensor product $\Lambda \otimes R$ of the CRT lattice Λ and the ring of integers R. Consequently, whenever the underlying algebraic lattice R admits an efficient CVP algorithm, CVP on the corresponding ring gadget lattice is efficiently solvable. This applies to:

  1. prime-power cyclotomics C_{p^e} and $C_{2^f p^e}$ (and their duals), which map to root lattices A_n, A_n^* and admit $O(n)$ linear-time decoding [MSC10];
  2. composite cyclotomics $C_{2^f p_1^{e_1} p_2^{e_2}}$ with at most two distinct odd prime factors, which map to tensor products of root lattices $A_n \otimes A_m$ and admit polynomial-time decoding [DvW18].

  Finding polynomial-time CVP algorithms for arbitrary cyclotomic rings (or general algebraic number fields) and their duals, or accelerating existing solvers to $O(n)$ linear time, remains

an open problem.

- **Tensor Products of Non-Orthogonal Gadgets:** Combining power gadget lattices with non-power-of-two cyclotomic lattices is significantly more challenging because neither constituent lattice possesses an orthogonal basis. In general, efficiently solving CVP on a tensor product $\Lambda_1 \otimes \Lambda_2$ given decoders for Λ_1 and Λ_2 is currently only known when at least one factor has an orthogonal basis. Resolving exact CVP for general tensor products $\Lambda_1 \otimes \Lambda_2$ (or specifically for power gadgets over general cyclotomic rings) is an important open direction.
- **Extension to Arbitrary Moduli:** Finally, an important practical direction is extending our $O(n)$ power gadget solver from moduli of the form $q = b^n$ to arbitrary integer moduli q. While the linear-time BDD algorithm of [MP12] was extended to arbitrary moduli in [GMP19], that result remains restricted to the BDD radius. We conjecture that the Voronoi analysis of [GMP19] can be generalized to achieve exact CVP decoding for power gadgets with arbitrary modulus q in polynomial or linear time.

References

[ABGS21] Divesh Aggarwal, Huck Bennett, Alexander Golovnev, and Noah Stephens-Davidowitz. Fine-grained hardness of CVP(P) - everything that we can prove (and nothing else). In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1816–1835. SIAM, 2021.

[ABSS97] Sanjeev Arora, László Babai, Jacques Stern, and Elizabeth (Z) Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.*, 54(2):317–331, 1997.

[AEVZ02] Erik Agrell, Thomas Eriksson, Alexander Vardy, and Kenneth Zeger. Closest point search in lattices. *IEEE Trans. Inf. Theory*, 48(8):2201–2214, 2002.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Symposium on Theory of Computing - STOC 1996*, pages 99–108. ACM, 1996.

[Ajt98] Miklós Ajtai. The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract). In *Proceedings of STOC*, pages 10–19. ACM, May 1998.

[AKKV11] Mikhail Alekhnovich, Subhash Khot, Guy Kindler, and Nisheeth Kumar Vishnoi. Hardness of approximating the closest vector problem with pre-processing. *Comput. Complex.*, 20(4):741–753, 2011.

[AP14] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 297–314. Springer, 2014.

[Bab86] László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Comb.*, 6(1):1–13, 1986.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory*, 6(3):13:1–13:36, 2014. Preliminary version in ITCS 2012.

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 575–584. ACM, 2013.

[CHKP12] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. *J. Cryptol.*, 25(4):601–639, 2012.

[CS92] John Horton Conway and Neil James Alexander Sloane. Low-dimensional lattices. vi. Voronoi reduction of three-dimensional lattices. *Proc. R. Soc. London*, A(436):55–68, 1992.

[CS99] John Horton Conway and Neil James Alexander Sloane. *Sphere Packings, Lattices and Groups*, volume 290 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, New York, NY, third edition edition, 1999.

[CW10] Qi Cheng and Daqing Wan. Complexity of decoding positive-rate primitive reed-solomon codes. *IEEE Trans. Inf. Theory*, 56(10):5217–5222, 2010.

[DB15] Daniel Dadush and Nicolas Bonifas. Short paths on the Voronoi graph and closest vector problem with preprocessing. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 295–314. SIAM, 2015.

[DKRS03] Irit Dinur, Guy Kindler, Ran Raz, and Shmuel Safra. Approximating CVP to within almost-polynomial factors is NP-hard. *Combinatorica*, 23(2):205–243, 2003. Preliminary version in FOCS 1998.

[DP19] Léo Ducas and Cécile Pierrot. Polynomial time bounded distance decoding near minkowski's bound in discrete logarithm lattices. *Des. Codes Cryptogr.*, 87(8):1737–1748, 2019.

[DvW18] Léo Ducas and Wessel P. J. van Woerden. The closest vector problem in tensored root lattices of type A and in their duals. *Des. Codes Cryptogr.*, 86(1):137–150, 2018.

[FM04] Uriel Feige and Daniele Micciancio. The inapproximability of lattice and coding problems with preprocessing. *J. Comput. Syst. Sci.*, 69(1):45–67, 2004.

[FP85] Ulrich Fincke and Michael Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Mathematics of Computation*, 44(170):463–471, 1985.

[GGG18] Venkata Gandikota, Badih Ghazi, and Elena Grigorescu. Np-hardness of reed-solomon decoding, and the prouhet-tarry-escott problem. *SIAM J. Comput.*, 47(4):1547–1584, 2018.

[GMP19] Nicholas Genise, Daniele Micciancio, and Yuriy Polyakov. Building an efficient lattice gadget toolkit: Subgaussian sampling and more. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 655–684. Springer, 2019.

[GMR05] Venkatesan Guruswami, Daniele Micciancio, and Oded Regev. The complexity of the covering radius problem. *Comput. Complex.*, 14(2):90–121, 2005.

[GMSS99] Oded Goldreich, Daniele Micciancio, Shmuel Safra, and Jean-Pierre Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Inf. Process. Lett.*, 71(2):55–61, 1999.

[GP17] Elena Grigorescu and Chris Peikert. List-decoding Barnes-Wall lattices. *Comput. Complex.*, 26(2):365–392, 2017.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometry codes. *IEEE Trans. Inf. Theory*, 45(6):1757–1767, 1999.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, 2013.

[GV05] Venkatesan Guruswami and Alexander Vardy. Maximum-likelihood decoding of reed-solomon codes is np-hard. *IEEE Trans. Inf. Theory*, 51(7):2249–2256, 2005.

[HLRW24] Monika Henzinger, Jason Li, Satish Rao, and Di Wang. Deterministic near-linear time minimum cut in weighted graphs. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 3089–3139. SIAM, 2024.

[Kar00] David R. Karger. Minimum cuts in near-linear time. *J. ACM*, 47(1):46–76, 2000.

[KM24] Elena Kirshanova and Ekaterina Malygina. Construction-d lattice from garcia-stichtenoth tower code. *Des. Codes Cryptogr.*, 92(5):1127–1142, 2024.

[Li21] Jason Li. Deterministic mincut in almost-linear time. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 384–395. ACM, 2021.

[LM09] Vadim Lyubashevsky and Daniele Micciancio. On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 577–594. Springer, 2009.

[MCQ08] Robert Gordon McKilliam, Ian Vaughan Leigh Clarkson, and Barry Gerard Quinn. An algorithm to compute the nearest point in the lattice a_n^*. *IEEE Trans. Inf. Theory*, 54(9):4378–4381, 2008.

[MGC14] Robert Gordon McKilliam, Alexander James Grant, and Ian Vaughan Leigh Clarkson. Finding a closest point in a lattice of Voronoi's first kind. *SIAM J. Discret. Math.*, 28(3):1405–1422, 2014.

[Mic01] Daniele Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Trans. Inf. Theory*, 47(3):1212–1215, 2001.

[Mic08] Daniele Micciancio. Efficient reductions among lattice problems. In Shang-Hua Teng, editor, *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 84–93. SIAM, 2008.

[Mic12] Daniele Micciancio. Inapproximability of the shortest vector problem: Toward a deterministic reduction. *Theory Comput.*, 8(1):487–512, 2012.

[MN08] Daniele Micciancio and Antonio Nicolosi. Efficient bounded distance decoders for Barnes-Wall lattices. In Frank R. Kschischang and En-Hui Yang, editors, *2008 IEEE International Symposium on Information Theory, ISIT 2008, Toronto, ON, Canada, July 6-11, 2008*, pages 2484–2488. IEEE, 2008.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, 2012.

[MP22] Ethan Mook and Chris Peikert. Lattice (list) decoding near minkowski's inequality. *IEEE Trans. Inf. Theory*, 68(2):863–870, 2022.

[MPSV21] Seth Thomas McCormick, Britta Peis, Robert Scheidweiler, and Frank Vallentin. A polynomial time algorithm for solving the closest vector problem in zonotopal lattices. *SIAM J. Discret. Math.*, 35(4):2345–2356, 2021.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

[MS23] Daniele Micciancio and Mark Schultz. Error correction and ciphertext quantization in lattice cryptography. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V*, volume 14085 of *Lecture Notes in Computer Science*, pages 648–681. Springer, 2023.

[MSC10] Robert Gordon McKilliam, Warren Douglas Smith, and Ian Vaughan Leigh Clarkson. Linear-time nearest point algorithms for Coxeter lattices. *IEEE Trans. Inf. Theory*, 56(3):1015–1022, 2010.

[MV13] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *SIAM J. Comput.*, 42(3):1364–1391, 2013.

[PS87] Azaria Paz and Claus-Peter Schnorr. Approximating integer lattices by lattices with cyclic factor groups. In Thomas Ottmann, editor, *Automata, Languages and Programming, 14th International Colloquium, ICALP87, Karlsruhe, Germany, July 13-17, 1987, Proceedings*, volume 267 of *Lecture Notes in Computer Science*, pages 386–393. Springer, 1987.

[Reg04] Oded Regev. Improved inapproximability of lattice and coding problems with preprocessing. *IEEE Trans. Inf. Theory*, 50(9):2031–2037, 2004.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009.

[SE94] Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.*, 66:181–199, 1994.

[SFS09] Naftali Sommer, Meir Feder, and Ofir Shalvi. Finding the closest lattice point by iterative slicing. *SIAM J. Discret. Math.*, 23(2):715–731, 2009.

[Var75] James Milton Varah. A lower bound for the smallest singular value of a matrix. *Linear Algebra and its Applications*, 11(1):3–5, 1975.

[vEB81] Peter van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical Report 81-04, Mathematische Instituut, Universiry of Amsterdam, 1981.

[Vor08] Georgy Feodosevich Voronoi. Recherches sur les parallélloèdres primitifs. *Journal für die Reine und Angewandte Mathematik*, 134:198–287, 1908.

[Zam14] Ram Zamir. *Lattice Coding for Signals and Networks: A Structured Coding Approach to Quantization, Modulation, and Compression.* Cambridge University Press, Cambridge, 2014.

A Proof of Lemma 3.1

This is similar to the proof given in [MGC14], but adapted to the notation (and potential function) used in this paper.

We need to prove that for any lattice basis $\mathbf{B}$ of Voronoi first kind and target $\mathbf{v} \in \mathrm{span}(\mathbf{B})$ such that $\Delta_{\mathbf{B}}(\mathbf{v}) > 0$, if $\mathbf{g} \in \pm\{0,1\}^n$ minimizes $\|\mathbf{v} - \mathbf{B}\mathbf{g}\|$, then $\Delta_{\mathbf{B}}(\mathbf{v} - \mathbf{B}\mathbf{g}) < \Delta_{\mathbf{B}}(\mathbf{v})$.

The lemma is most conveniently proved using the (obtuse) superbasis $\overline{\mathbf{B}} = [\mathbf{b}_0, \mathbf{B}]$. Using $\overline{\mathbf{B}}$, any lattice point $\mathbf{v} = \mathbf{B}\mathbf{x} \in \mathcal{L}(\mathbf{B})$ admits infinitely many representations as $\overline{\mathbf{B}}\mathbf{x}'$, for $\mathbf{x}' \in (0, \mathbf{x}) + c \cdot \mathbf{u} \in \mathbb{Z}^{n+1}$, where $\mathbf{u} = (1, \ldots, 1)$ and $c \in \mathbb{Z}$ is an arbitrary integer. The representation can be made unique by requiring $\min\{x_i \mid i = 0, \ldots, n\} = 0$. We call this vector $\overline{\mathbf{x}} = (0, \mathbf{x}) - \min(0, \mathbf{x}) \cdot \mathbf{u}$ the standard representation of $\mathbf{v}$ with respect to $\mathbf{B}$.

As a special case, the standard representation for the lattice points $\mathbf{B}\mathbf{g}$ with $\mathbf{g} \in \pm\{0,1\}^n$ used by the CVP algorithm is given by the indicator vectors $\mathbf{u}_G \in \{0,1\}^{n+1}$ of sets $G \subset N = \{0, \ldots, n\}$:

- If $\mathbf{g} \in \{0,+1\}^n$, then $G = \{i\colon g_i = 1\}$, and $\mathbf{u}_G = (0, \mathbf{g})$.
- If $\mathbf{g} \in \{0,-1\}^n$, then $G = \{0\} \cup \{i\colon g_i = 0\}$, and $\mathbf{u}_G = (1, \mathbf{u} + \mathbf{g})$.

Using this notation, the main loop of the algorithm can be rewritten as:

1. Find $G \subset N$ such that $\|\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G\|$ is minimized
2. If $\|\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G\| < \|\mathbf{v}\|$, then subtract $\overline{\mathbf{B}}\mathbf{u}_G$ from $\mathbf{v}$ and repeat.

Expressing vectors with respect to the superbasis $\overline{\mathbf{B}}$ also allows to write the potential function as

$$\Delta_{\mathbf{B}}(\mathbf{v}) = \max(\overline{\mathbf{x}}) - \min(\overline{\mathbf{x}}) = \max(\overline{\mathbf{x}})$$

where $\overline{\mathbf{x}}$ is the standard representative of the lattice vector $\mathbf{Bx}$ closest to $\mathbf{v}$ achieving $\Delta_{\mathbf{B}}(\mathbf{v})$.

We can now prove the lemma. Let $\mathbf{v}$ be any target in $\mathrm{span}(\mathbf{B})$, and $\overline{\mathbf{B}}\overline{\mathbf{x}}$ a closest lattice point at distance $\mu = \mathrm{dist}(\mathbf{v}, \mathcal{L}(\mathbf{B}))$ from $\mathbf{v}$, with standard representative $\overline{\mathbf{x}} \in \mathbb{Z}^{n+1}$ such that $\min \overline{\mathbf{x}} = 0$ and $\max \overline{\mathbf{x}} = \Delta(\mathbf{v})$. We need to prove that $\Delta(\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G) < \Delta(\mathbf{v})$, i.e., there is an integer vector $\overline{\mathbf{x}}' \in \mathbb{Z}^{n+1}$ such that $\min \overline{\mathbf{x}}' = 0$, $\max \overline{\mathbf{x}}' < \max \overline{\mathbf{x}}$ and $\|(\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G) - \overline{\mathbf{B}}\overline{\mathbf{x}}'\| \leq \mu$. Let $\overline{\mathbf{x}}'$ be the vector with coordinates

$$\overline{x}'_i = \begin{cases} \Delta(\mathbf{v}) - 1 & \text{if } \overline{x}_i = \Delta(\mathbf{v}) \\ 0 & \text{if } \overline{x}_i = 0 \\ \overline{x}_i - (\mathbf{u}_G)_i & \text{otherwise} \end{cases}$$

If we let $\mathbf{w} = \overline{\mathbf{x}} - \overline{\mathbf{x}}' - \mathbf{u}_G$, then we have

$$\begin{aligned} \|(\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G) - \overline{\mathbf{B}}\overline{\mathbf{x}}'\|^2 &= \|(\mathbf{v} - \overline{\mathbf{B}}\overline{\mathbf{x}}) + \overline{\mathbf{B}}\mathbf{w}\|^2 \\ &= \|\mathbf{v} - \overline{\mathbf{B}}\overline{\mathbf{x}}\|^2 + \|\overline{\mathbf{B}}\mathbf{w}\|^2 + 2\left\langle \mathbf{v} - \overline{\mathbf{B}}\overline{\mathbf{x}}, \overline{\mathbf{B}}\mathbf{w}\right\rangle. \end{aligned}$$

The first term in this expression equals $\|\mathbf{v} - \overline{\mathbf{B}}\overline{\mathbf{x}}\|^2 = \mu^2$. In order to evaluate the second term, we compute

$$\begin{aligned} \|\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G\|^2 &= \|\mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}') + \overline{\mathbf{B}}\mathbf{w}\|^2 \\ &= \|\mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}')\|^2 + \|\overline{\mathbf{B}}\mathbf{w}\|^2 + 2\left\langle \mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}'), \overline{\mathbf{B}}\mathbf{w}\right\rangle. \end{aligned}$$

Rearranging the terms we get

$$\begin{aligned} \|\overline{\mathbf{B}}\mathbf{w}\|^2 &= \|\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G\|^2 - \|\mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}')\|^2 - 2\left\langle \mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}'), \overline{\mathbf{B}}\mathbf{w}\right\rangle \\ &\leq -2\left\langle \mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}'), \overline{\mathbf{B}}\mathbf{w}\right\rangle \end{aligned}$$

where the inequality follows from the fact that $\overline{\mathbf{x}} - \overline{\mathbf{x}}' \in \{0,1\}^{n+1}$ is a binary vector, and the choice of $\mathbf{u}_G$ as the binary vector minimizing $\|\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G\|^2$. Substituting in the previous expression, we obtain

$$\begin{aligned} \|(\mathbf{v} - \overline{\mathbf{B}}\mathbf{u}_G) - \overline{\mathbf{B}}\overline{\mathbf{x}}'\|^2 &\leq \mu^2 - 2\left\langle \mathbf{v} - \overline{\mathbf{B}}(\overline{\mathbf{x}} - \overline{\mathbf{x}}'), \overline{\mathbf{B}}\mathbf{w}\right\rangle + 2\left\langle \mathbf{v} - \overline{\mathbf{B}}\overline{\mathbf{x}}, \overline{\mathbf{B}}\mathbf{w}\right\rangle \\ &= \mu^2 - 2\left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}', \overline{\mathbf{B}}\mathbf{w}\right\rangle. \end{aligned}$$

It remains to show that $\left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}', \overline{\mathbf{B}}\mathbf{w}\right\rangle \geq 0$. To this end, it is useful to introduce some additional notation. For any vector $\mathbf{z} \in \mathbb{Z}^{n+1}$ and set $S \subseteq N = \{0, \ldots, n\}$, let $\mathbf{z}_S \in \mathbb{Z}^{n+1}$ be the vector with coordinates $(\mathbf{z}_S)_i = z_i$ if $i \in S$, and $(\mathbf{z}_S)_i = 0$ otherwise.

Notice that the vector $\mathbf{w}$ has coordinates

$$w_i = \overline{x}_i - (\mathbf{u}_G)_i - \overline{x}'_i = \begin{cases} +1 & \text{if } \overline{x}_i = \Delta(\mathbf{v}) \text{ and } i \notin G \\ -1 & \text{if } \overline{x}_i = 0 \text{ and } i \in G \\ 0 & \text{otherwise} \end{cases}$$

So, we can write $\mathbf{w} = \mathbf{u}_S - \mathbf{u}_T$ for appropriate sets $S, T \subseteq N = \{0, \ldots, n\}$ such that $\overline{x}_i = \Delta(\mathbf{v})$ for all $i \in S$ and $\overline{x}_i = 0$ for all $i \in T$. For any vector $\overline{\mathbf{x}}'$ and set S, we also have $\overline{\mathbf{x}}' = \overline{\mathbf{x}}'_S + \overline{\mathbf{x}}'_{\bar{S}}$, where $\bar{S} = N \setminus S$ is the complement of S in $N = \{0, \ldots, n\}$. It follows by linearity that

$$\overline{\mathbf{B}}\mathbf{u}_{\bar{S}} = \overline{\mathbf{B}}(\mathbf{u} - \mathbf{u}_S) = \overline{\mathbf{B}}\mathbf{u} - \overline{\mathbf{B}}\mathbf{u}_S = -\overline{\mathbf{B}}\mathbf{u}_S.$$

Using these properties, we have

$$\begin{aligned}
\left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}', \overline{\mathbf{B}}\mathbf{w} \right\rangle &= \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}', \overline{\mathbf{B}}\mathbf{u}_S \right\rangle - \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}', \overline{\mathbf{B}}\mathbf{u}_T \right\rangle \\
&= \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_S + \overline{\mathbf{x}}'_{\bar{S}}, \overline{\mathbf{B}}\mathbf{u}_S \right\rangle - \left\langle \overline{\mathbf{B}}(\overline{\mathbf{x}}'_T + \overline{\mathbf{x}}'_{\bar{T}}), \overline{\mathbf{B}}\mathbf{u}_T \right\rangle \\
&= \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_S, \overline{\mathbf{B}}\mathbf{u}_S \right\rangle + \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_{\bar{S}}, \overline{\mathbf{B}}\mathbf{u}_S \right\rangle - \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_T, \overline{\mathbf{B}}\mathbf{u}_T \right\rangle - \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_{\bar{T}}, \overline{\mathbf{B}}\mathbf{u}_T \right\rangle \\
&= -\left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_S, \overline{\mathbf{B}}\mathbf{u}_{\bar{S}} \right\rangle + \left\langle \overline{\mathbf{B}}\mathbf{u}_S, \overline{\mathbf{B}}\overline{\mathbf{x}}'_{\bar{S}} \right\rangle + \left\langle \overline{\mathbf{B}}\mathbf{u}_{\bar{T}}, \overline{\mathbf{B}}\overline{\mathbf{x}}'_T \right\rangle - \left\langle \overline{\mathbf{B}}\overline{\mathbf{x}}'_{\bar{T}}, \overline{\mathbf{B}}\mathbf{u}_T \right\rangle \\
&= \sum_{i \in S, j \notin S} \langle \mathbf{b}_i, \mathbf{b}_j \rangle (\overline{x}'_j - \overline{x}'_i) + \sum_{i \notin T, j \in T} \langle \mathbf{b}_i, \mathbf{b}_j \rangle (\overline{x}'_j - \overline{x}'_i).
\end{aligned}$$

All the terms in the summations are positive because $\langle \mathbf{b}_i, \mathbf{b}_j \rangle \leq 0$ for all $i \neq j$, and $\overline{x}'_j \leq \overline{x}'_i$ whenever $i \in S$ (and $\overline{x}'_i = \Delta(\mathbf{v}) - 1 = \max(\overline{\mathbf{x}}')$) or $j \in T$ (and $\overline{x}'_j = 0 = \min(\overline{\mathbf{x}}')$.)