Hardness of Bounded Distance Decoding on Lattices in ℓ_p Norms

Huck Bennett
University of Michigan, Ann Arbor, MI, USA
hdbco@umich.edu

Chris Peikert
University of Michigan, Ann Arbor, MI, USA
cpeikert@umich.edu

Abstract

Bounded Distance Decoding $\mathrm{BDD}_{p,\alpha}$ is the problem of decoding a lattice when the target point is promised to be within an α factor of the minimum distance of the lattice, in the ℓ_p norm. We prove that $\mathrm{BDD}_{p,\alpha}$ is NP-hard under randomized reductions where $\alpha \to 1/2$ as $p \to \infty$ (and for $\alpha = 1/2$ when $p = \infty$), thereby showing the hardness of decoding for distances approaching the unique-decoding radius for large p. We also show *fine-grained* hardness for $\mathrm{BDD}_{p,\alpha}$. For example, we prove that for all $p \in [1, \infty) \setminus 2\mathbb{Z}$ and constants $C > 1, \varepsilon > 0$, there is no $2^{(1-\varepsilon)n/C}$-time algorithm for $\mathrm{BDD}_{p,\alpha}$ for some constant α (which approaches $1/2$ as $p \to \infty$), assuming the randomized Strong Exponential Time Hypothesis (SETH). Moreover, essentially all of our results also hold (under analogous non-uniform assumptions) for BDD with *preprocessing*, in which unbounded precomputation can be applied to the lattice before the target is available.

Compared to prior work on the hardness of $\mathrm{BDD}_{p,\alpha}$ by Liu, Lyubashevsky, and Micciancio (APPROX-RANDOM 2008), our results improve the values of α for which the problem is known to be NP-hard for all $p > p_1 \approx 4.2773$, and give the very first fine-grained hardness for BDD (in any norm). Our reductions rely on a special family of "locally dense" lattices in ℓ_p norms, which we construct by modifying the integer-lattice sparsification technique of Aggarwal and Stephens-Davidowitz (STOC 2018).

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Lattices, Bounded Distance Decoding, NP-hardness, Fine-Grained Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2020.36

Related Version `https://arxiv.org/abs/2003.07903`

Acknowledgements We thank the Simons Institute for hosting the Spring 2020 program "Lattices: Algorithms, Complexity, and Cryptography," at which some of this work was completed. We also thank Noah Stephens-Davidowitz for sharing his plot-generating code from [5] with us.

1 Introduction

Lattices in $\mathbb{R}^n$ are a rich source of computational problems with applications across computer science, and especially in cryptography and cryptanalysis. (A lattice is a discrete additive subgroup of $\mathbb{R}^n$, or equivalently, the set of integer linear combinations of a set of linearly independent vectors.) Many important lattice problems appear intractable, and there is a wealth of research showing that central problems like the Shortest Vector Problem (SVP) and Closest Vector Problem (CVP) are NP-hard, even to approximate to within various factors and in various ℓ_p norms [31, 8, 7, 22, 23, 17, 16, 14, 25]. (For the sake of concision, throughout this introduction the term "NP-hard" allows for *randomized* reductions, which are needed in some important cases.)

35th Computational Complexity Conference (CCC 2020).
Editor: Shubhangi Saraf; Article No. 36; pp. 36:1–36:21
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Bounded Distance Decoding

In recent years, the emergence of lattices as a powerful foundation for cryptography, including for security against quantum attacks, has increased the importance of other lattice problems. In particular, many modern lattice-based encryption schemes rely on some form of the *Bounded Distance Decoding* (BDD) problem, which is like the Closest Vector Problem with a promise. An instance of BDD_α for *relative distance* $\alpha > 0$ is a lattice $\mathcal{L}$ and a target point $\boldsymbol{t}$ whose distance from the lattice is guaranteed to be within an α factor of the lattice's minimum distance $\lambda_1(\mathcal{L}) = \min_{\boldsymbol{v}\in\mathcal{L}\setminus\{\boldsymbol{0}\}}\|\boldsymbol{v}\|$, and the goal is to find a lattice vector within that distance of $\boldsymbol{t}$; when distances are measured in the ℓ_p norm we denote the problem $\mathrm{BDD}_{p,\alpha}$. Note that when $\alpha < 1/2$ there is a unique solution, but the problem is interesting and well-defined for larger relative distances as well. We also consider *preprocessing* variants of CVP and BDD (respectively denoted CVPP and BDDP), in which unbounded precomputation can be applied to the lattice before the target is available. For example, this can model cryptographic contexts where a fixed long-term lattice may be shared among many users.

The importance of BDD(P) to cryptography is especially highlighted by the Learning With Errors (LWE) problem of Regev [27], which is an average-case form of BDD that has been used (with inverse-polynomial α) in countless cryptosystems, including several that share a lattice among many users (see, e.g., [13]). Moreover, Regev gave a worst-case to average-case reduction from BDD to LWE, so the security of cryptosystems is intimately related to the worst-case complexity of BDD.

Compared to problems like SVP and CVP, the BDD(P) problem has received much less attention from a complexity-theoretic perspective. We are aware of essentially only one work showing its NP-hardness: Liu, Lyubashevsky, and Micciancio [19] proved that $\mathrm{BDD}_{p,\alpha}$ and even $\mathrm{BDDP}_{p,\alpha}$ are NP-hard for relative distances approaching $\min\{1/\sqrt{2}, 1/\sqrt[p]{2}\}$, which is $1/\sqrt{2}$ for $p \geq 2$. A few other works relate BDD(P) to other lattice problems (in both directions) in regimes where the problems are not believed to be NP-hard, e.g., [24, 11, 9]. (Dadush, Regev, and Stephens-Davidowitz [11] also gave a reduction that implies NP-hardness of $\mathrm{BDD}_{2,\alpha}$ for any $\alpha > 1$, which is larger than the relative distance of $\alpha = 1/\sqrt{2} + \varepsilon$ achieved by [19].)

Fine-grained hardness

An important aspect of hard lattice problems, especially for cryptography, is their *quantitative* hardness. That is, we want not only that a problem cannot be solved in polynomial time, but that it cannot be solved in, say, $2^{o(n)}$ time or even $2^{n/C}$ time for a certain constant C. Statements of this kind can be proven under generic complexity assumptions like the Exponential Time Hypothesis (ETH) of Impagliazzo and Paturi [15] or its variants like Strong ETH (SETH), via *fine-grained* reductions that are particularly efficient in the relevant parameters.

Recently, Bennett, Golovnev, and Stephens-Davidowitz [10] initiated a study of the fine-grained hardness of lattice problems, focusing on CVP; follow-up work extended to SVP and showed more for CVP(P) [5, 2]. The technical goal of these works is a reduction having good *rank efficiency*, i.e., a reduction from k-SAT on n' variables to a lattice problem in rank $n = (C + o(1))n'$ for some constant $C \geq 1$, which we call the reduction's "rank inefficiency." (All of the lattice problems in question can be solved in $2^{n+o(n)}$ time [3, 4, 6], so $C = 1$ corresponds to optimal rank efficiency.) We mention that Regev's BDD-to-LWE reduction [27] has optimal rank efficiency, in that it reduces rank-n BDD to rank-n LWE. However, to date there are no fine-grained NP-hardness results for BDD itself; the prior NP-hardness proof for BDD [19] incurs a large polynomial blowup in rank.

1.1 Our Results

We show improved NP-hardness, and entirely new fine-grained hardness, for Bounded Distance Decoding (and BDD with preprocessing) in arbitrary ℓ_p norms. Our work improves upon the known hardness of BDD in two respects: the relative distance α, and the rank inefficiency C (i.e., fine-grainedness) of the reductions. As p grows, both quantities improve, simultaneously approaching the unique-decoding threshold $\alpha = 1/2$ and optimal rank efficiency of $C = 1$ as $p \to \infty$, and achieving those quantities for $p = \infty$. We emphasize that these are the first fine-grained hardness results of any kind for BDD, for any ℓ_p norm.

Our main theorem summarizing the NP- and fine-grained hardness of BDD (with and without preprocessing) appears below in Theorem 1. For $p \in [1, \infty)$ and $C > 1$, the quantities α_p^* and $\alpha_{p,C}^*$ appearing in the theorem statement are certain positive real numbers that are decreasing in p and C, and approaching $1/2$ as $p \to \infty$ (for any C). See Figure 1 for a plot of their behavior, Equations (3.4) and (3.5) for their formal definitions, and Lemma 27 for quite tight closed-form upper bounds.

▶ **Theorem 1.** *The following hold for* $\mathrm{BDD}_{p,\alpha}$ *and* $\mathrm{BDDP}_{p,\alpha}$ *in rank n:*

1. *For every $p \in [1, \infty)$ and constant $\alpha > \alpha_p^*$ (where $\alpha_p^* \le \frac{1}{2} \cdot 4.6723^{1/p}$), and for $(p, \alpha) = (\infty, 1/2)$, there is no polynomial-time algorithm for $\mathrm{BDD}_{p,\alpha}$ (respectively, $\mathrm{BDDP}_{p,\alpha}$) unless $\mathsf{NP} \subseteq \mathsf{RP}$ (resp., $\mathsf{NP} \subseteq \mathsf{P/Poly}$).*
2. *For every $p \in [1, \infty)$ and constant $\alpha > \min\{\alpha_p^*, \alpha_2^*\}$, and for $(p, \alpha) = (\infty, 1/2)$, there is no $2^{o(n)}$-time algorithm for $\mathrm{BDD}_{p,\alpha}$ unless randomized ETH fails.*
3. *For every $p \in [1, \infty) \setminus \{2\}$ and constant $\alpha > \alpha_p^*$, and for $(p, \alpha) = (\infty, 1/2)$, there is no $2^{o(n)}$-time algorithm for $\mathrm{BDDP}_{p,\alpha}$ unless non-uniform ETH fails.*
   Moreover, for every $p \in [1, \infty]$ and $\alpha > \alpha_2^*$ there is no $2^{o(\sqrt{n})}$-time algorithm for $\mathrm{BDDP}_{p,\alpha}$ unless non-uniform ETH fails.
4. *For every $p \in [1, \infty) \setminus 2\mathbb{Z}$ and constants $C > 1$, $\alpha > \alpha_{p,C}^*$, and $\epsilon > 0$, and for $(p, C, \alpha) = (\infty, 1, 1/2)$, there is no $2^{n(1-\epsilon)/C}$-time algorithm for $\mathrm{BDD}_{p,\alpha}$ (respectively, $\mathrm{BDDP}_{p,\alpha}$) unless randomized SETH (resp., non-uniform SETH) fails.*

Although we do not have closed-form expressions for α_p^* and $\alpha_{p,C}^*$, we do get quite tight closed-form upper bounds (see Lemma 27). Moreover, it is easy to numerically compute close approximations to them, and to the values of p at which they cross certain thresholds. For example, $\alpha_p^* < 1/\sqrt{2}$ for all $p > p_1 \approx 4.2773$, so Item 1 of Theorem 1 improves on the prior best relative distance of any $\alpha > 1/\sqrt{2}$ for the NP-hardness of $\mathrm{BDD}_{p,\alpha}$ in such ℓ_p norms [19].

As a few other example values and their consequences under Theorem 1, we have $\alpha_2^* \approx 1.05006$, $\alpha_{3,2}^* \approx 1.1418$, and $\alpha_{3,5}^* \approx 0.917803$. So by Item 2, BDD in the Euclidean norm for any relative distance $\alpha > 1.05006$ requires $2^{\Omega(n)}$ time assuming randomized ETH. And by Item 4, for every $\varepsilon > 0$ there is no $2^{(1-\varepsilon)n/2}$-time algorithm for $\mathrm{BDD}_{3,1.1418}$, and no $2^{(1-\varepsilon)n/5}$-time algorithm for $\mathrm{BDD}_{3,0.917803}$, assuming randomized SETH.

1.2 Technical Overview

As in prior NP-hardness reductions for SVP and BDD (and fine-grained hardness proofs for the former) [7, 22, 16, 19, 14, 25, 5], the central component of our reductions is a family of rank-n lattices $\mathcal{L} \subset \mathbb{R}^d$ and target points $\boldsymbol{t} \in \mathbb{R}^d$ having a certain "local density" property in a desired ℓ_p norm. Informally, this means that $\mathcal{L}$ has "large" minimum distance $\lambda_1^{(p)}(\mathcal{L}) := \min_{\boldsymbol{v} \in \mathcal{L} \setminus \{\boldsymbol{0}\}} \|\boldsymbol{v}\|_p$, i.e., there are no "short" nonzero vectors, but has many vectors "close" to the target $\boldsymbol{t}$. More precisely, we want $\lambda_1^{(p)}(\mathcal{L}) \ge r$ and $N_p(\mathcal{L}, \alpha r, \boldsymbol{t}) = \exp(n^{\Omega(1)})$ for some relative distance α, where

$$N_p(\mathcal{L}, s, \boldsymbol{t}) := |\{\boldsymbol{v} \in \mathcal{L} : \|\boldsymbol{v} - \boldsymbol{t}\|_p \le s\}|$$

denotes the number of lattice points within distance s of $\boldsymbol{t}$.

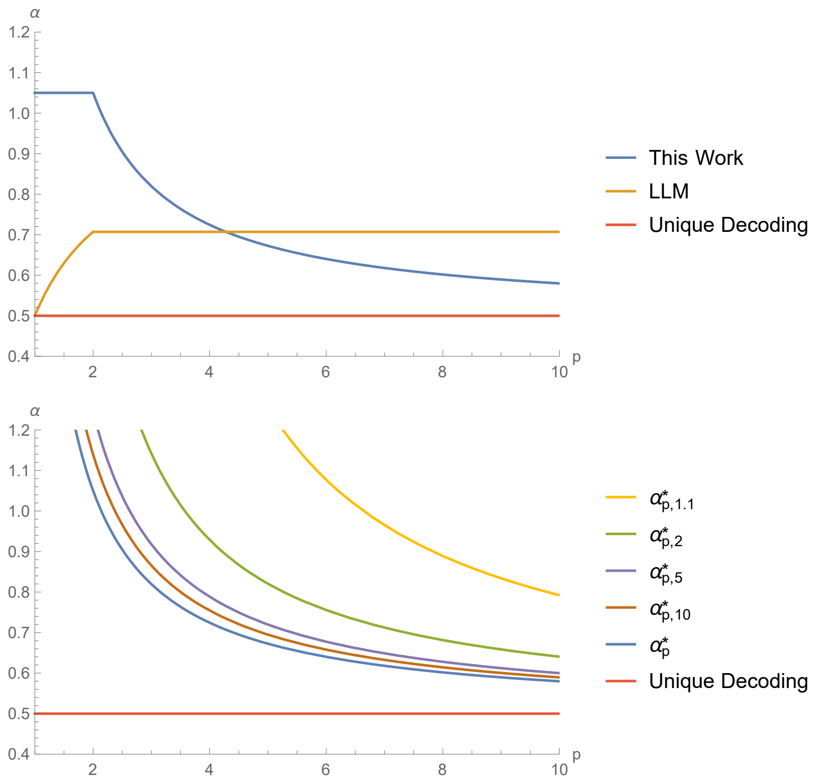

Figure 1 Top: bounds on the relative distances $\alpha = \alpha(p)$ for which $\mathrm{BDD}_{\alpha,p}$ was proved to be NP-hard in the ℓ_p norm, in this work and in [19]; the crossover point is $p_1 \approx 4.2773$. (The plots include results obtained by norm embeddings [28], hence they are maximized at $p = 2$.) Bottom: our bounds $\alpha^*_{p,C}$ on the relative distances $\alpha > \alpha^*_{p,C}$ for which there is no $2^{(1-\varepsilon)n/C}$-time algorithm for $\mathrm{BDD}_{p,\alpha}$ for any $\varepsilon > 0$, assuming randomized SETH.

Micciancio [22] constructed locally dense lattices with relative distance approaching $2^{-1/p}$ in the ℓ_p norm (for every finite $p \geq 1$), and used them to prove the NP-hardness of γ-approximate SVP in ℓ_p for any $\gamma < 2^{1/p}$. Subsequently, Liu, Lyubashevsky, and Micciancio [19] used these lattices to prove the NP-hardness of BDD in ℓ_p for any relative distance $\alpha > 2^{-1/p}$. However, these works observed that the relative distance depends on p in the opposite way from what one might expect: as p grows, so does α, hence the associated NP-hard SVP approximation factors and BDD relative distances *worsen*. Yet using norm embeddings, it can be shown that ℓ_2 is essentially the "easiest" ℓ_p norm for lattice problems [28], so hardness in ℓ_2 implies hardness in ℓ_p (up to an arbitrarily small loss in approximation factor). Therefore, the locally dense lattices from [22] do not seem to provide any benefits for $p > 2$ over $p = 2$, where the relative distance approaches $1/\sqrt{2}$. In addition, the rank of these lattices is a large polynomial in the relevant parameter, so they are not suitable for proving fine-grained hardness.[1]

[1] We mention that Khot [16] gave a different construction of locally dense lattices with other useful properties, but their relative distance is no smaller than that of Micciancio's construction in any ℓ_p norm, and their rank is also a large polynomial in the relevant parameter.

Local density via sparsification

More recently, Aggarwal and Stephens-Davidowitz [5] (building on [10]) proved fine-grained hardness for *exact* SVP in ℓ_p norms, via locally dense lattices obtained in a different way. Because they target exact SVP, it suffices to have local density for relative distance $\alpha = 1$, but for fine-grained hardness they need $N_p(\mathcal{L}, r, \boldsymbol{t}) = 2^{\Omega(n)}$, preferably with a large hidden constant (which determines the rank efficiency of the reduction). Following [21, 12], they start with the integer lattice $\mathbb{Z}^n$ and all-$\frac{1}{2}$s target vector $\boldsymbol{t} = \frac{1}{2}\mathbf{1} \in \mathbb{R}^n$. Clearly, there are 2^n lattice vectors all at distance $r = \frac{1}{2}n^{1/p}$ from $\boldsymbol{t}$ in the ℓ_p norm, but the minimum distance of the lattice is only 1, so the relative distance of the "close" vectors is $\alpha = r$, which is far too large.

To improve the relative distance, they increase the minimum distance to at least $r = \frac{1}{2}n^{1/p}$ using the elegant technique of *random sparsification*, which is implicit in [12] and was first used for proving NP-hardness of approximate SVP in [17, 16]. The idea is to upper-bound the number $N_p(\mathbb{Z}^n, r, \mathbf{0})$ of "short" lattice points of length at most r, by some Q. Then, by taking a random sublattice $\mathcal{L} \subset \mathbb{Z}^n$ of determinant (index) slightly larger than Q, with noticeable probability none of the "short" nonzero vectors will be included in $\mathcal{L}$, whereas roughly $2^n/Q$ of the vectors "close" to $\boldsymbol{t}$ will be in $\mathcal{L}$. So, as long as $Q = 2^{(1-\Omega(1))n}$, there are sufficiently many lattice vectors at the desired relative distance from $\boldsymbol{t}$.

Bounds for $N_p(\mathbb{Z}^n, r, \mathbf{0})$ were given by Mazo and Odlyzko [21], by a simple but powerful technique using the theta function $\Theta_p(\tau) := \sum_{z\in\mathbb{Z}} \exp(-\tau|z|^p)$. They showed (see Proposition 13) that

$$N_p(\mathbb{Z}^n, r, \mathbf{0}) \le \min_{\tau>0} \exp(\tau \cdot r^p) \cdot \Theta_p(\tau)^n = \Big(\min_{\tau>0} \exp(\tau/2^p) \cdot \Theta_p(\tau)\Big)^n \ , \tag{1.1}$$

where the equality is by $r = \frac{1}{2}n^{1/p}$. So, Aggarwal and Stephens-Davidowitz need $\min_{\tau>0} \exp(\tau/2^p) \cdot \Theta_p(\tau) < 2$, and it turns out that this is the case for every $p > p_0 \approx 2.1397$. (They also deal with smaller p by using a different target point $\boldsymbol{t}$.)

This work: local density for small relative distance

For the NP- and fine-grained hardness of BDD we use the same basic approach as in [5], but with the different goal of getting local density for as small of a relative distance $\alpha < 1$ as we can manage. That is, we still have 2^n integral vectors all at distance $r = \frac{1}{2}n^{1/p}$ from the target $\boldsymbol{t} = \frac{1}{2}\mathbf{1} \in \mathbb{R}^n$, but we want to "sparsify away" all the nonzero integral vectors of length less than r/α. So, we want the right-hand side of the Mazo-Odlyzko bound (Equation (1.1)) to be at most $2^{(1-\Omega(1))n}$ for as large of a positive hidden constant as we can manage. More specifically, for any $p \ge 1$ and $C > 1$ (which ultimately corresponds to the reduction's rank inefficiency) we can obtain local density of at least $2^{n/C}$ close vectors at any relative distance greater than

$$\alpha^*_{p,C} := \inf\{\alpha^* > 0 : \min_{\tau>0} \exp(\tau/(2\alpha^*)^p) \cdot \Theta_p(\tau) \le 2^{1-1/C}\} \ .$$

The value of $\alpha^*_{p,C}$ is strictly decreasing in both p and C, and for large C and $p > p_1 \approx 4.2773$ it drops below the relative distance of $1/\sqrt{2}$ approached by the local-density construction of [22] for ℓ_2 (and also ℓ_p by norm embeddings.) This is the source of our improved relative distance for the NP-hardness of BDD in high ℓ_p norms.

We also show that obtaining local density by sparsifying the integer lattice happens to yield a very simple reduction to BDD from the *exact* version of CVP, which is how we obtain fine-grained hardness. Given a CVP instance consisting of a lattice and a target point, we essentially just take their direct sum with the integer lattice and the $\frac{1}{2}\mathbf{1}$ target (respectively), then sparsify. (See Lemma 21 and Theorem 19 for details.) Because this results in the (sparsified) locally dense lattice having $2^{\Omega(n)}$ close vectors all *exactly* at the threshold of the BDD promise, concatenating the CVP instance either keeps the target within the (slightly weaker) BDD promise, or puts it just outside. This is in contrast to the prior reduction of [19], where the close vectors in the locally dense lattices of [22] are at various distances from the target, hence a reduction from *approximate*-CVP with a large constant factor is needed to put the target outside the BDD promise. While approximating CVP to within any constant factor is known to be NP-hard [8], no fine-grained hardness is known for approximate CVP, except for factors just slightly larger than one [2].

1.3 Discussion and Future Work

Our work raises a number of interesting issues and directions for future research. First, it highlights that there are now two incomparable approaches for obtaining local density in the ℓ_p norm – Micciancio's construction [22], and sparsifying the integer lattice [12, 5] – with each delivering a better relative distance for certain ranges of p. For $p \in [1, p_1 \approx 4.2773]$, Micciancio's construction (with norm embeddings from ℓ_2, where applicable) delivers the better relative distance, which approaches $\min\{1/\sqrt[p]{2}, 1/\sqrt{2}\}$. Moreover, this is essentially optimal in ℓ_2, where $1/\sqrt{2}$ is unachievable due to the Rankin bound, which says that in $\mathbb{R}^n$ we can have at most $2n$ subunit vectors with pairwise distances of $\sqrt{2}$ or more.

A first question, therefore, is whether relative distance less than $1/\sqrt{2}$ can be obtained for all $p > 2$. We conjecture that this is true, but can only manage to prove it via sparsification for all $p > p_1 \approx 4.2773$. More generally, an important open problem is to give a unified local-density construction that subsumes both of the above-mentioned approaches in terms of relative distance, and ideally in rank efficiency as well. In the other direction, another important goal is to give lower bounds on the relative distance in general ℓ_p norms. Apart from the Rankin bound, the only bound we are aware of is the trivial one of $\alpha \geq 1/2$ implied by the triangle inequality, which is essentially tight for ℓ_1 and tight for ℓ_∞ (as shown by [22] and our work, respectively).

More broadly, for the BDD relative distance parameter α there are three regimes of interest: the local-density regime, where we know how to prove NP-hardness; the unique-decoding regime $\alpha < 1/2$; and (at least in some ℓ_p norms, including ℓ_2) the intermediate regime between them. It would be very interesting, and would seem to require new techniques, to show NP-hardness outside the local-density regime. One potential route would be to devise a *gap amplification* technique for BDD, analogous to how SVP has been proved to be NP-hard to approximate to within any constant factor [16, 14, 25]. Gap amplification may also be interesting in the absence of NP-hardness, e.g., for the inverse-polynomial relative distances used in cryptography. Currently, the only efficient gap amplification we are aware of is a modest one that decreases the relative distance by any $(1 - 1/n)^{O(1)}$ factor [20].

A final interesting research direction is related to the *unique* Shortest Vector Problem (uSVP), where the goal is to find a shortest nonzero vector $\boldsymbol{v}$ in a given lattice, under the promise that it is unique (up to sign). More generally, approximate uSVP has the promise that all lattice vectors not parallel to $\boldsymbol{v}$ are a certain factor γ as long. It is known that *exact* uSVP is NP-hard in ℓ_2 [18], and by known reductions it is straightforward to show the NP-hardness of 2-*approximate* uSVP in ℓ_∞. Can recent techniques help to prove NP-hardness of γ-approximate uSVP, for some constant $\gamma > 1$, in ℓ_p for some finite p, or specifically for ℓ_2? Do NP-hard approximation factors for uSVP grow smoothly with p?

2 Preliminaries

For any positive integer q, we identify the quotient group $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ with some set of distinguished representatives, e.g., $\{0, 1, \ldots, q-1\}$. Let $B^+ := (B^t B)^{-1} B^t$ denote the Moore-Penrose pseudoinverse of a real-valued matrix B with full column rank. Observe that $B^+\boldsymbol{v}$ is the unique coefficient vector $\boldsymbol{c}$ with respect to B of any $\boldsymbol{v} = B\boldsymbol{c}$ in the column span of B.

2.1 Problems with Preprocessing

In addition to ordinary computational problems, we are also interested in (promise) problems with *preprocessing*. In such a problem, an instance (x_P, x_Q) is comprised of a "preprocessing" part x_P and a "query" part x_Q, and an algorithm is allowed to perform unbounded computation on the preprocessing part before receiving the query part.

Formally, a preprocessing problem is a relation $\Pi = \{((x_P, x_Q), y)\}$ of instance-solution pairs, where $\Pi_{\text{inst}} := \{(x_P, x_Q) : \exists\, y \text{ s.t. } ((x_P, x_Q), y) \in \Pi\}$ is the set of problem instances, and $\Pi_{(x_P, x_Q)} := \{y : ((x_P, x_Q), y) \in \Pi\}$ is the set of solutions for any particular instance (x_P, x_Q). If every instance $(x_P, x_Q) \in \Pi_{\text{inst}}$ has exactly one solution that is either YES or NO, then Π is called a decision problem.

▶ **Definition 2.** *A* preprocessing algorithm *is a pair (P, Q) where P is a (possibly randomized) function representing potentially unbounded computation, and Q is an algorithm. The execution of (P, Q) on an input (x_P, x_Q) proceeds in two phases:*

- *first, in the preprocessing phase, P takes x_P as input and produces some preprocessed output σ;*
- *then, in the query phase, Q takes both σ and x_Q as input and produces some ultimate output.*

The running time T of the algorithm is defined to be the time used in the query phase alone, and is considered as a function of the total input length $|x_P| + |x_Q|$. The length of the preprocessed output is defined as $A = |\sigma|$, and is also considered as a function of the total input length. Note that without loss of generality, $A \leq T$.

If (P, Q) is deterministic, we say that it solves preprocessing problem Π if $Q(P(x_P), x_Q) \in \Pi_{(x_P, x_Q)}$ for all $(x_P, x_Q) \in \Pi_{inst}$. If (P, Q) is potentially randomized, we say that it solves Π if

$$\Pr[Q(P(x_P), x_Q) \in \Pi_{(x_P, x_Q)}] \geq \frac{2}{3}$$

for all $(x_P, x_Q) \in \Pi_{inst}$, where the probability is taken over the random coins of both P and Q.[2]

As shown below using a routine quantifier-swapping argument (as in Adleman's Theorem [1]), it turns out that for NP relations and decision problems, any randomized preprocessing algorithm can be derandomized if the length of the query input x_Q is polynomial in the length of the preprocessing input x_P. So for convenience, in this work we allow for randomized algorithms, only switching to deterministic ones for our ultimate hardness theorems.

[2] Note that it could be the case that some preprocessed outputs fail to make the query algorithm output a correct answer on some, or even all, query inputs.

▶ **Lemma 3.** *Let preprocessing problem* Π *be an* NP *relation or a decision problem for which* $|x_Q| = \mathrm{poly}(|x_P|)$ *for all* $(x_P, x_Q) \in \Pi_{inst}$*. If* Π *has a randomized* T*-time algorithm, then it has a deterministic* $T \cdot \mathrm{poly}(|x_P| + |x_Q|)$*-time algorithm with* $T \cdot \mathrm{poly}(|x_P| + |x_Q|)$*-length preprocessed output.*

Proof. Let $q(\cdot)$ be a polynomial for which $|x_Q| \le q(|x_P|)$ for all $(x_P, x_Q) \in \Pi_{\text{inst}}$. Let (P, Q) be a randomized T-time algorithm for Π, which by standard repetition techniques we can assume has probability strictly less than $\exp(-q(|x_P|))$ of being incorrect on any $(x_P, x_Q) \in \Pi_{\text{inst}}$, with only a $\mathrm{poly}(|x_P| + |x_Q|)$-factor overhead in the running time and preprocessed output length. Fix some arbitrary x_P. Then by the union bound over all $(x_P, x_Q) \in \Pi_{\text{inst}}$ and the hypothesis, we have

$$\Pr[\exists\ (x_P, x_Q) \in \Pi_{\text{inst}} : Q(P(x_P), x_Q) \notin \Pi_{(x_P,x_Q)}] < 1.$$

So, there exist coins for P and Q for which $Q(P(x_P), x_Q) \in \Pi_{(x_P,x_Q)}$ for all $(x_P, x_Q) \in \Pi_{\text{inst}}$. By fixing these coins we make P a deterministic function of x_P, and we include the coins for Q along with the preprocessed output $P(x_P)$, thus making Q deterministic as well. The resulting deterministic algorithm solves Π with the claimed resources, as needed. ◀

Reductions for preprocessing problems

We need the following notions of reductions for preprocessing problems. The following generalizes Turing reductions and Cook reductions (i.e., polynomial-time Turing reductions).

▶ **Definition 4.** *A* Turing reduction *from one preprocessing problem* X *to another one* Y *is a pair of algorithms* (R_P, R_Q) *satisfying the following properties:* R_P *is a (potentially randomized) function with access to an oracle* P*, whose output length is polynomial in its input length;* R_Q *is an algorithm with access to an oracle* Q*; and if* (P, Q) *solves problem* Y*, then* (R_P^P, R_Q^Q) *solves problem* X*. Additionally, it is a* Cook reduction *if* R_Q *runs in time polynomial in the total input length of* R_P *and* R_Q*.*

Similarly, the following generalizes mapping reductions and Karp reductions (i.e., polynomial-time mapping reductions) for decision problems.

▶ **Definition 5.** *A* mapping reduction *from one preprocessing decision problem* X *to another one* Y *is a pair* (R_P, R_Q) *satisfying the following properties:* R_P *is a deterministic function whose output length is polynomial in its input length;* R_Q *is a deterministic algorithm; and for any YES (respectively, NO) instance* (x_P, x_Q) *of* X*, the output pair* (y_P, y_Q) *is a YES (resp., NO) instance of* Y*, where* (y_P, y_Q) *are defined as follows:*

- *first,* R_P *takes* x_P *as input and outputs some* (σ', y_P)*, where* σ' *is some "internal" preprocessed output;*
- *then,* R_Q *takes* (σ', x_Q) *as input and outputs some* y_Q*.*

Additionally, it is a Karp reduction *if* R_Q *runs in time polynomial in the total input length of* R_P *and* R_Q*.*

It is straightforward to see that if X mapping reduces to Y, and there is a deterministic polynomial-time preprocessing algorithm (P_Y, Q_Y) that solves Y, then there is also one (P_X, Q_X) that solves X, which works as follows:

1. the preprocessing algorithm P_X, given a preprocessing input x_p, first computes $(\sigma', y_P) = R_P(x_P)$, then computes $\sigma_Y = P_Y(y_P)$ and outputs $\sigma_X = (\sigma', \sigma_Y)$;
2. the query algorithm Q_X, given $\sigma_X = (\sigma', \sigma_Y)$ and a query input x_Q, computes $y_Q = R_Q(\sigma', x_Q)$ and finally outputs $Q_Y(\sigma_Y, y_Q)$.

2.2 Lattices

A *lattice* is the set of all integer linear combinations of some linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$. It is convenient to arrange these vectors as the columns of a matrix. Accordingly, we define a *basis* $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) \in \mathbb{R}^{d \times n}$ to be a matrix with linearly independent columns, and the lattice generated by basis B as

$$\mathcal{L}(B) := \left\{ \sum_{i=1}^{n} a_i \boldsymbol{b}_i : a_1, \ldots, a_n \in \mathbb{Z} \right\} .$$

Let $\mathcal{B}_p^d$ denote the centered unit ℓ_p ball in d dimensions. Given a lattice $\mathcal{L} \subset \mathbb{R}^d$ of rank n, for $1 \leq i \leq n$ let

$$\lambda_i^{(p)}(\mathcal{L}) := \inf\{r > 0 : \dim(\operatorname{span}(r \cdot \mathcal{B}_p^d \cap \mathcal{L})) \geq i\}$$

denote the ith successive minimum of $\mathcal{L}$ with respect to the ℓ_p norm.

We denote the ℓ_p distance of a vector $\boldsymbol{t}$ to a lattice $\mathcal{L}$ as

$$\operatorname{dist}_p(\boldsymbol{t}, \mathcal{L}) := \min_{\boldsymbol{v} \in \mathcal{L}} \|\boldsymbol{v} - \boldsymbol{t}\|_p .$$

2.3 Bounded Distance Decoding (with Preprocessing)

The primary computational problem that we study in this work is the Bounded Distance Decoding Problem (BDD), which is a version of the Closest Vector Problem (CVP) in which the target vector is promised to be relatively close to the lattice.

▶ **Definition 6.** *For $1 \leq p \leq \infty$ and $\alpha = \alpha(n) > 0$, the α-*Bounded Distance Decoding problem in the ℓ_p norm *($\mathrm{BDD}_{p,\alpha}$) is the (search) promise problem defined as follows. The input is (a basis of) a rank-n lattice $\mathcal{L}$ and a target vector $\boldsymbol{t}$ satisfying $\operatorname{dist}_p(\boldsymbol{t}, \mathcal{L}) \leq \alpha(n) \cdot \lambda_1^{(p)}(\mathcal{L})$. The goal is to output a lattice vector $\boldsymbol{v} \in \mathcal{L}$ that satisfies $\|\boldsymbol{v} - \boldsymbol{t}\|_p \leq \alpha(n) \cdot \lambda_1^{(p)}(\mathcal{L})$.*

The preprocessing (search) promise problem $\mathrm{BDDP}_{p,\alpha}$ is defined analogously, where the preprocessing input is (a basis of) the lattice, and the query input is the target $\boldsymbol{t}$.

We note that in some works, BDD is defined to have the goal of finding a $\boldsymbol{v} \in \mathcal{L}$ such that $\|\boldsymbol{v} - \boldsymbol{t}\|_p = \operatorname{dist}_p(\boldsymbol{t}, \mathcal{L})$. This formulation is clearly no easier than the one defined above. So, our hardness theorems, which are proved for the definition above, immediately apply to the alternative formulation as well.

We also remark that for $\alpha < 1/2$, the promise ensures that there is a *unique* vector $\boldsymbol{v}$ satisfying $\|\boldsymbol{v} - \boldsymbol{t}\|_p \leq \alpha \cdot \lambda_1^{(p)}(\mathcal{L})$. However, BDD is still well defined for $\alpha \geq 1/2$, i.e., above the unique-decoding radius. As in prior work, our hardness results for $\mathrm{BDD}_{p,\alpha}$ are limited to this regime.

To the best of our knowledge, essentially the only previous study of the NP-hardness of BDD is due to [19], which showed the following result.[3]

▶ **Theorem 7** ([19, Corollaries 1 and 2])**.** *For any $p \in [1, \infty)$ and $\alpha > 1/2^{1/p}$, there is no polynomial-time algorithm for $\mathrm{BDD}_{p,\alpha}$ (respectively, with preprocessing) unless* $\mathsf{NP} \subseteq \mathsf{RP}$ *(resp., unless* $\mathsf{NP} \subseteq \mathsf{P/Poly}$*).*

[3] Additionally, [11] gave a reduction from CVP to $\mathrm{BDD}_{2,\alpha}$ but only for some $\alpha > 1$. Also, [26, 20] gave a reduction from GapSVP_γ to BDD, but only for large $\gamma = \gamma(n)$ for which GapSVP is not known to be NP-hard.

Regev and Rosen [28] used norm embeddings to show that almost any lattice problem is at least as hard in the ℓ_p norm, for any $p \in [1, \infty]$, as it is in the ℓ_2 norm, up to an arbitrarily small constant-factor loss in the approximation factor. In other words, they essentially showed that ℓ_2 is the "easiest" ℓ_p norm for lattice problems. (In addition, their reduction preserves the rank of the lattice.) Based on this, [19] observed the following corollary, which is an improvement on the factor α from Theorem 7 for all $p > 2$.

▶ **Theorem 8** ([19, Corollary 3]). *For any $p \in [1, \infty)$ and $\alpha > 1/\sqrt{2}$, there is no polynomial-time algorithm for $\mathrm{BDD}_{p,\alpha}$ (respectively, with preprocessing) unless $\mathsf{NP} \subseteq \mathsf{RP}$ (resp., unless $\mathsf{NP} \subseteq \mathsf{P/Poly}$).*

Figure 1 shows the bounds from Theorems 7 and 8 together with the new bounds achieved in this work as a function of p.

2.4 Sparsification

A powerful idea, first used in the context of hardness proofs for lattice problems in [17], is that of random lattice sparsification. Given a lattice $\mathcal{L}$ with basis B, we can construct a random sublattice $\mathcal{L}' \subseteq \mathcal{L}$ as

$$\mathcal{L}' = \{\boldsymbol{v} \in \mathcal{L} : \langle \boldsymbol{z}, B^+\boldsymbol{v}\rangle = 0 \pmod{q}\}$$

for uniformly random $\boldsymbol{z} \in \mathbb{Z}_q^n$, where q is a suitably chosen prime.

▶ **Lemma 9.** *Let q be a prime and let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{Z}_q^n \setminus \{\boldsymbol{0}\}$ be arbitrary. Then*

$$\Pr_{\boldsymbol{z} \leftarrow \mathbb{Z}_q^n}[\exists\, i \in [N] \text{ such that } \langle \boldsymbol{z}, \boldsymbol{x}_i\rangle = 0 \pmod{q}] \le \frac{N}{q} \ .$$

Proof. We have $\Pr[\langle \boldsymbol{z}, \boldsymbol{x}_i\rangle = 0] = 1/q$ for each $\boldsymbol{x}_i$, and the claim follows by the union bound. ◀

The following corollary is immediate.

▶ **Corollary 10.** *Let q be a prime and $\mathcal{L}$ be a lattice of rank n with basis B. Then for all $r > 0$ and all $p \in [1, \infty]$,*

$$\Pr_{\boldsymbol{z} \leftarrow \mathbb{Z}_q^n}[\lambda_1^{(p)}(\mathcal{L}') < r] \le \frac{N_p^o(\mathcal{L} \setminus \{\boldsymbol{0}\}, r, \boldsymbol{0})}{q} \ ,$$

where $\mathcal{L}' = \{\boldsymbol{v} \in \mathcal{L} : \langle \boldsymbol{z}, B^+\boldsymbol{v}\rangle = 0 \pmod{q}\}$.

▶ **Theorem 11** ([29, Theorem 3.1]). *For any lattice $\mathcal{L}$ of rank n with basis B, prime q, and lattice vectors $\boldsymbol{x}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_N \in \mathcal{L}$ such that $B^+\boldsymbol{x} \ne B^+\boldsymbol{y}_i \pmod{q}$ for all $i \in [N]$, we have*

$$\frac{1}{q} - \frac{N}{q^2} - \frac{N}{q^{n-1}} \le \Pr_{\boldsymbol{z}, \boldsymbol{c} \leftarrow \mathbb{Z}_q^n}[\langle \boldsymbol{z}, B^+\boldsymbol{x} + \boldsymbol{c}\rangle = 0 \pmod{q} \wedge \langle \boldsymbol{z}, B^+\boldsymbol{y}_i + \boldsymbol{c}\rangle \ne 0 \pmod{q}\ \forall\, i \in [N]] \le \frac{1}{q} + \frac{1}{q^n} \ .$$

We will use only the lower bound from Theorem 11, but we note that the upper bound is relatively tight for $q \gg N$.

▶ **Corollary 12.** *For any $p \in [1, \infty]$ and $r \ge 0$, lattice $\mathcal{L}$ of rank n with basis B, vector $\boldsymbol{t}$, prime q, and lattice vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N \in \mathcal{L}$ such that $\|\boldsymbol{v}_i - \boldsymbol{t}\|_p \le r$ for all $i \in [N]$ and such that all the $B^+\boldsymbol{v}_i \bmod q$ are distinct, we have*

$$\Pr_{\boldsymbol{z}, \boldsymbol{c} \leftarrow \mathbb{Z}_q^n}[\mathrm{dist}_p(\boldsymbol{t} + B\boldsymbol{c}, \mathcal{L}') \le r] \ge \frac{N}{q} - \frac{N(N-1)}{q^2} - \frac{N(N-1)}{q^{n-1}} \ ,$$

where $\mathcal{L}' = \{\boldsymbol{v} \in \mathcal{L} : \langle \boldsymbol{z}, B^+\boldsymbol{v}\rangle = 0 \pmod{q}\}$.

Proof. Observe that for each $i \in [N]$, the events

$$E_i := [\langle \boldsymbol{z}, B^+ \boldsymbol{v}_i \rangle = 0 \pmod q \text{ and } \langle \boldsymbol{z}, B^+ \boldsymbol{v}_j \rangle \neq 0 \pmod q \text{ for all } j \neq i]$$

are disjoint, and by invoking Theorem 11 with $\boldsymbol{x} = \boldsymbol{v}_i$ and the $\boldsymbol{y}_j$ being the remaining $\boldsymbol{v}_k$ for $k \neq i$, we have

$$\Pr_{\boldsymbol{z},\boldsymbol{c}}[E_i] \geq \frac{1}{q} - \frac{N-1}{q^2} - \frac{N-1}{q^{n-1}} \ .$$

Also observe that if E_i occurs, then $\boldsymbol{v}_i + B\boldsymbol{c} \in \mathcal{L}'$ (also $\boldsymbol{v}_j + B\boldsymbol{c} \notin \mathcal{L}'$ for all $j \neq i$, but we will not need this). Therefore,

$$\operatorname{dist}_p(\boldsymbol{t} + B\boldsymbol{c}, \mathcal{L}') \leq \|\boldsymbol{t} + B\boldsymbol{c} - (\boldsymbol{v}_i + B\boldsymbol{c})\| = \|\boldsymbol{t} - \boldsymbol{v}_i\| \leq r \ .$$

So, the probability in the left-hand side of the claim is at least

$$\Pr_{\boldsymbol{z},\boldsymbol{c}}\Big[\bigcup_{i\in[N]} E_i\Big] = \sum_{i\in[N]} \Pr_{\boldsymbol{z},\boldsymbol{c}}[E_i] \geq \frac{N}{q} - \frac{N(N-1)}{q^2} - \frac{N(N-1)}{q^{n-1}} \ . \qquad \blacktriangleleft$$

2.5 Counting Lattice Points in a Ball

Following [5], for any discrete set A of points (e.g., a lattice, or a subset thereof), we denote the number of points in A contained in the closed and open (respectively) ℓ_p ball of radius r centered at a point $\boldsymbol{t}$ as

$$N_p(A, r, \boldsymbol{t}) := |\{\boldsymbol{y} \in A : \|\boldsymbol{y} - \boldsymbol{t}\|_p \leq r\}| \ , \tag{2.1}$$

$$N_p^o(A, r, \boldsymbol{t}) := |\{\boldsymbol{y} \in A : \|\boldsymbol{y} - \boldsymbol{t}\|_p < r\}| \ . \tag{2.2}$$

Clearly, $N_p^o(A, r, \boldsymbol{t}) \leq N_p(A, r, \boldsymbol{t})$.

For $1 \leq p < \infty$ and $\tau > 0$ define

$$\Theta_p(\tau) := \sum_{z\in\mathbb{Z}} \exp(-\tau|z|^p) \ .$$

We use the following upper bound due to Mazo and Odlyzko [21] on the number of short vectors in the integer lattice. We include its short proof for completeness.

▶ **Proposition 13** ([21]). *For any $p \in [1, \infty)$, $r > 0$, and $n \in \mathbb{N}$,*

$$N_p(\mathbb{Z}^n, r, \boldsymbol{0}) \leq \min_{\tau>0} \exp(\tau r^p) \cdot \Theta_p(\tau)^n \ .$$

Proof. For $\tau > 0$ we have

$$\Theta_p(\tau)^n = \sum_{\boldsymbol{z}\in\mathbb{Z}^n} \exp(-\tau\|\boldsymbol{z}\|_p^p) \geq \sum_{\boldsymbol{z}\in\mathbb{Z}^n\cap r\mathcal{B}_p^n} \exp(-\tau\|\boldsymbol{z}\|_p^p) \geq \exp(-\tau r^p) \cdot N_p(\mathbb{Z}^n, r, \boldsymbol{0}) \ .$$

The result follows by rearranging and taking the minimum over all $\tau > 0$. ◀

2.6 Hardness Assumptions

We recall the Exponential Time Hypothesis (ETH) of Impagliazzo and Paturi [15], and several of its variants. These hypotheses make stronger assumptions about the complexity of the k-SAT problem than the assumption $\mathsf{P} \neq \mathsf{NP}$, and serve as highly useful tools for studying the fine-grained complexity of hard computational problems. Indeed, we will show that strong fine-grained hardness for BDD follows from these hypotheses.

▶ **Definition 14.** *The* (randomized) Exponential Time Hypothesis *((randomized) ETH) asserts that there is no (randomized) $2^{o(n)}$-time algorithm for 3-SAT on n variables.*

▶ **Definition 15.** *The* (randomized) Strong Exponential Time Hypothesis *((randomized) SETH) asserts that for every $\varepsilon > 0$ there exists $k \in \mathbb{Z}^+$ such that there is no (randomized) $2^{(1-\varepsilon)n}$-time algorithm for k-SAT on n variables.*

For proving hardness of lattice problem with preprocessing, we define (Max-)k-SAT with preprocessing as follows. The preprocessing input is a size parameter n, encoded in unary. The query input is a k-SAT formula ϕ with n variables and m (distinct) clauses, together with a threshold $W \in \{0, \dots m\}$ in the case of Max-k-SAT. For k-SAT, it is a YES instance if ϕ is satisfiable, and is a NO instance otherwise. For Max-k-SAT, it is a YES instance if there exists an assignment to the variables of ϕ that satisfies at least W of its clauses, and is a NO instance otherwise.

Observe that because the preprocessing input is just n, a preprocessing algorithm for (Max-)k-SAT with preprocessing is equivalent to a (non-uniform) family of circuits for the problem *without* preprocessing. Also, for any fixed k, because there are only $O(n^k)$ possible clauses on n variables, the length of the query input for (Max-)k-SAT instances having preprocessing input n is $\mathrm{poly}(n)$, so we get the following corollary of Lemma 3.

▶ **Corollary 16.** *If (Max-)k-SAT with preprocessing has a randomized $T(n)$-time algorithm, then it has a deterministic $T(n) \cdot \mathrm{poly}(n)$-time algorithm using $T(n) \cdot \mathrm{poly}(n)$-length preprocessed output.*

Following, e.g., [30, 2], we also define *non-uniform* variants of ETH and SETH, which deal with the complexity of k-SAT with preprocessing. More precisely, non-uniform ETH asserts that no family of size-$2^{o(n)}$ circuits solves 3-SAT on n variables (equivalently, 3-SAT with preprocessing does not have a $2^{o(n)}$-time algorithm), and non-uniform SETH asserts that for every $\varepsilon > 0$ there exists $k \in \mathbb{Z}^+$ such that no family of circuits of size $2^{(1-\varepsilon)n}$ solves k-SAT on n variables (equivalently, k-SAT with preprocessing does not have a $2^{(1-\varepsilon)n}$-time algorithm). These hypotheses are useful for analyzing the fine-grained complexity of preprocessing problems.

One might additionally consider "randomized non-uniform" versions of (S)ETH. However, Corollary 16 says that a randomized algorithm for (Max-)k-SAT with preprocessing can be derandomized with only polynomial overhead, so randomized non-uniform (S)ETH is equivalent to (deterministic) non-uniform (S)ETH, so we only consider the latter.

Finally, we remark that one can define weaker versions of randomized or non-uniform (S)ETH with Max-3-SAT (respectively, Max-k-SAT) in place of 3-SAT (resp., k-SAT). Many of our results hold even under these weaker hypotheses. In particular, the derandomization result in Corollary 16 applies to both k-SAT and Max-k-SAT.

3 Hardness of $\mathrm{BDD}_{p,\alpha}$

In this section, we present our main result by giving a reduction from a known-hard variant GapCVP'_p of the Closest Vector Problem (CVP) to BDD. We peform this reduction in two main steps.

1. First, in Section 3.1 we define a variant of $\mathrm{BDD}_{p,\alpha}$, which we call (S,T)-$\mathrm{BDD}_{p,\alpha}$. Essentially, an instance of this problem is a lattice that may have up to S "short" nonzero vectors of ℓ_p norm bounded by some r, and a target vector that is "close" to – i.e., within distance αr of – at least T lattice vectors. (The presence of short vectors prevents this from being a true $\mathrm{BDD}_{p,\alpha}$ instance.) We then give a reduction, for $S \ll T$, from (S,T)-$\mathrm{BDD}_{p,\alpha}$ to $\mathrm{BDD}_{p,\alpha}$ itself, using sparsification.

2. Then, in Section 3.2 we reduce from GapCVP'_p to (S,T)-$\mathrm{BDD}_{p,\alpha}$ for suitable $S \ll T$ whenever α is sufficiently large as a function of p (and the desired rank efficiency), based on analysis given in Section 3.3 and Lemma 27.

3.1 (S,T)-BDD to BDD

We start by defining a special decision variant of BDD. Essentially, the input is a lattice and a target vector, and the problem is to distinguish between the case where there are few "short" lattice vectors but many lattice vectors "close" to the target, and the case where the target is not close to the lattice. There is a gap factor between the "close" and "short" distances, and for technical reasons we count only those "close" vectors having binary coefficients with respect to the given input basis.

▶ **Definition 17.** *Let $S = S(n), T = T(n) \geq 0$, $p \in [1,\infty]$, and $\alpha = \alpha(n) > 0$. An instance of the decision promise problem (S,T)-$\mathrm{BDD}_{p,\alpha}$ is a lattice basis $B \in \mathbb{R}^{d\times n}$, a distance $r > 0$, and a target $\boldsymbol{t} \in \mathbb{R}^d$.*

- *It is a YES instance if $N_p^o(\mathcal{L}(B) \setminus \{\mathbf{0}\}, r, \mathbf{0}) \leq S(n)$ and $N_p(B \cdot \{0,1\}^n, \alpha r, \boldsymbol{t}) \geq T(n)$.*
- *It is a NO instance if $\mathrm{dist}_p(\boldsymbol{t}, \mathcal{L}(B)) > \alpha r$.*

The search version is: given a YES instance $(B, r, \boldsymbol{t})$, find a $\boldsymbol{v} \in \mathcal{L}(B)$ such that $\|\boldsymbol{v} - \boldsymbol{t}\|_p \leq \alpha r$.

The preprocessing search and decision problems (S,T)-$\mathrm{BDDP}_{p,\alpha}$ are defined analogously, where the preprocessing input is B and r, and the query input is $\boldsymbol{t}$.

We stress that in the preprocessing problems BDDP, the distance r is part of the preprocessing input; this makes the problem no harder than a variant where r is part of the query input. So, our hardness results for the above definition immediately apply to that variant as well. However, our reduction *from* (S,T)-BDDP (given in Lemma 18) critically relies on the fact that r is part of the preprocessing input.

Clearly, there is a trivial reduction from the decision version of (S,T)-$\mathrm{BDD}_{p,\alpha}$ to its search version (and similarly for the preprocessing problems): just call the oracle for the search problem and test whether it returns a lattice vector within distance αr of the target. So, to obtain more general results, our reductions involving (S,T)-BDD will be *from* the search version, and *to* the decision version.

Reducing to BDD

We next observe that for $S(n) = 0$ and any $T(n) > 0$, there is *almost* a trivial reduction from (S,T)-$\mathrm{BDD}_{p,\alpha}$ to ordinary $\mathrm{BDD}_{p,\alpha}$, because YES instances of the former satisfy the $\mathrm{BDD}_{p,\alpha}$ promise. (See below for the easy proof.) The only subtlety is that we want the $\mathrm{BDD}_{p,\alpha}$ oracle to return a lattice vector that is within distance αr of the target; recall that the definition of $\mathrm{BDD}_{p,\alpha}$ only guarantees distance $\alpha \cdot \lambda_1^{(p)}(\mathcal{L}(B))$. This issue is easily resolved by modifying the lattice to *upper bound* its minimum distance by r, which increases the lattice's rank by one. (For the alternative definition of BDD described after Definition 6, the trivial reduction works, and no increase in the rank is needed.)

▶ **Lemma 18.** *For any $T(n) > 0$, $p \in [1,\infty]$, and $\alpha = \alpha(n) > 0$, there is a deterministic Cook reduction from the search version of $(0, T(n))$-$\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank n to $\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank $n+1$.*

Proof. The reduction works as follows. On input $(B, r, \boldsymbol{t})$, call the $\mathrm{BDD}_{p,\alpha}$ oracle on

$$B' := \begin{pmatrix} B & 0 \\ 0 & r \end{pmatrix}, \quad \boldsymbol{t}' := \begin{pmatrix} \boldsymbol{t} \\ 0 \end{pmatrix} ,$$

and (without loss of generality) receive from the oracle a vector $\boldsymbol{v}' = (\boldsymbol{v}, zr)$ for some $\boldsymbol{v} \in \mathcal{L}$ and $z \in \mathbb{Z}$. Output $\boldsymbol{v}$.

We analyze the reduction. Let $\mathcal{L} = \mathcal{L}(B)$ and $\mathcal{L}' = \mathcal{L}(B')$. Because the input is a YES instance, we have $N_p^o(\mathcal{L} \setminus \{\boldsymbol{0}\}, r, \boldsymbol{0}) = 0$ and hence $\lambda_1^{(p)}(\mathcal{L}) \geq r$, so $\lambda_1^{(p)}(\mathcal{L}') = r$. Moreover, $N_p(B \cdot \{0,1\}^n, \alpha r, \boldsymbol{t}) > 0$ implies that $\mathrm{dist}_p(\boldsymbol{t}', \mathcal{L}') = \mathrm{dist}(\boldsymbol{t}, \mathcal{L}) \leq \alpha r = \alpha \cdot \lambda_1^{(p)}(\mathcal{L}')$. So, $(B', \boldsymbol{t}')$ satisfies the $\mathrm{BDD}_{p,\alpha}$ promise, hence the oracle is obligated to return some $\boldsymbol{v}' = (\boldsymbol{v}, zr) \in \mathcal{L}'$ where $\boldsymbol{v} \in \mathcal{L}$ and $\alpha r = \alpha \lambda_1^{(p)}(\mathcal{L}') \geq \|\boldsymbol{v}' - \boldsymbol{t}'\|_p \geq \|\boldsymbol{v} - \boldsymbol{t}\|_p$. Therefore, the output $\boldsymbol{v}$ of the reduction is a valid solution.

Finally, observe that all of the above also constitutes a valid reduction for the preprocessing problems, because B' depends only on the preprocessing part B, r of the input. ◀

We now present a more general randomized reduction from (S,T)-$\mathrm{BDD}_{p,\alpha}$ to $\mathrm{BDD}_{p,\alpha}$, which works whenever $T(n) \geq 10S(n)$. The essential idea is to sparsify the input lattice, so that with some noticeable probability no short vectors remain, but at least one vector close to the target does remain. In this case, the result will be an instance of $(0,1)$-$\mathrm{BDD}_{p,\alpha}$, which reduces to $\mathrm{BDD}_{p,\alpha}$ as shown above.

We note that the triangle inequality precludes the existence of (S,T)-$\mathrm{BDD}_{p,\alpha}$ instances with $T > S+1$ and $\alpha \leq 1/2$, so with this approach we can only hope to show hardness of $\mathrm{BDD}_{p,\alpha}$ for $\alpha > 1/2$, i.e., the unique-decoding regime remains out of reach.

▶ **Theorem 19.** *For any $S = S(n) \geq 1$ and $T = T(n) \geq 10S$ that is efficiently computable (for unary n), $p \in [1, \infty]$, and $\alpha = \alpha(n) > 0$, there is a randomized Cook reduction with no false positives from the search version of (S,T)-$\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank n to $\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank $n+1$.*

Proof. By Lemma 18, it suffices to give such a reduction to $(0,1)$-$\mathrm{BDD}_{p,\alpha}$ in rank n, which works as follows. On input $(B, r, \boldsymbol{t})$, let $\mathcal{L} = \mathcal{L}(B)$. First, randomly choose a prime q where $10T \leq q \leq 20T$. Then sample $\boldsymbol{z}, \boldsymbol{c} \in \mathbb{Z}_q^n$ independently and uniformly at random, and define

$$\mathcal{L}' := \{\boldsymbol{v} \in \mathcal{L} : \langle \boldsymbol{z}, B^+ \boldsymbol{v} \rangle = 0 \pmod{q}\} \text{ and } \boldsymbol{t}' := \boldsymbol{t} + B\boldsymbol{c} .$$

Let B' be a basis of $\mathcal{L}'$. (Such a basis is efficiently computable from B, $\boldsymbol{z}$, and q. See, e.g., [29, Claim 2.15].) Invoke the $(0,1)$-$\mathrm{BDD}_{p,\alpha}$ oracle on $(B', r, \boldsymbol{t}')$, and output whatever the oracle outputs.

We now analyze the reduction. We are promised that $(B, r, \boldsymbol{t})$ is a YES instance of (S,T)-$\mathrm{BDD}_{p,\alpha}$, and it suffices to show that $(B', r, \boldsymbol{t}')$ is a YES instance of $(0,1)$-$\mathrm{BDD}_{p,\alpha}$, i.e., $\lambda_1^{(p)}(\mathcal{L}') \geq r$ and $\mathrm{dist}_p(\boldsymbol{t}', \mathcal{L}') \leq \alpha r$, with some positive constant probability. By Corollary 10 we have

$$\Pr[\lambda_1^{(p)}(\mathcal{L}') < r] \leq \frac{N_p^o(\mathcal{L} \setminus \{\boldsymbol{0}\}, r, \boldsymbol{0})}{q} \leq \frac{S}{q} \leq \frac{1}{100} .$$

Furthermore, because there are T vectors $\boldsymbol{v}_i \in \mathcal{L}$ for which $\|\boldsymbol{v}_i - \boldsymbol{t}\|_p \leq \alpha r$, and their coefficient vectors $B^+ \boldsymbol{v}_i \in \{0,1\}^n$ are distinct (as integer vectors, and hence also modulo q), by Corollary 12 we have

$$\Pr[\mathrm{dist}_p(\boldsymbol{t}', \mathcal{L}') \leq \alpha r] \geq \frac{T}{q} - \frac{T^2}{q^2} - \frac{T^2}{q^{n-1}} \geq \frac{1}{20} - \frac{1}{400} - \frac{1}{400 q^{n-3}} .$$

Therefore, by the union bound we have

$$\Pr[\lambda_1^{(p)}(\mathcal{L}') \geq r \text{ and } \operatorname{dist}_p(\boldsymbol{t}', \mathcal{L}') \leq \alpha r] \geq \frac{1}{20} - \frac{1}{400} - \frac{1}{400 q^{n-3}} - \frac{1}{100} \geq \frac{1}{40}$$

for all $n \geq 3$, as desired.

Finally, the above also constitutes a valid reduction for the preprocessing problems (in the sense of Definition 4), because B' depends only on B from the preprocessing part of the input and the reduction's own random choices (and r remains unchanged). ◀

3.2 GapCVP' to (S, T)-BDD

Here we show that a known-hard variant of the (exact) Closest Vector Problem reduces to (S, T)-BDD (in its decision version).

▶ **Definition 20.** *For $p \in [1, \infty]$, the (decision) promise problem* GapCVP'_p *is defined as follows: an instance consists of a basis $B \in \mathbb{R}^{d \times n}$ and a target vector $\boldsymbol{t} \in \mathbb{R}^d$.*
- *It is a YES instance if there exists $\boldsymbol{x} \in \{0,1\}^n$ such that $\|B\boldsymbol{x} - \boldsymbol{t}\|_p \leq 1$.*
- *It is a NO instance if $\operatorname{dist}_p(\boldsymbol{t}, \mathcal{L}(B)) > 1$.*

The preprocessing (decision) promise problem $\mathrm{GapCVPP}'_p$ *is defined analogously, where the preprocessing input is B and the query input is $\boldsymbol{t}$.*

Observe that for GapCVP'_p the distance threshold is 1 (and not some instance-dependent value) without loss of generality, because we can scale the lattice and target vector. The same goes for $\mathrm{GapCVPP}'_p$, with the caveat that any instance-dependent distance threshold would need to be included in the *preprocessing* part of the input, not the query part. (See Remark 26 below for why this is essentially without loss of generality, under a mild assumption on the $\mathrm{GapCVPP}'_p$ instances.) We remark that some works define these problems with a stronger requirement that in the NO case, $\operatorname{dist}_p(z\boldsymbol{t}, \mathcal{L}(B)) > r$ for all $z \in \mathbb{Z} \setminus \{0\}$. We will not need this stronger requirement, and some of the hardness results for GapCVP' that we rely on are not known to hold with it, so we use the weaker requirement.

We next describe a simple transformation on lattices and target vectors: we essentially take a direct sum of the input lattice with the integer lattice of any desired dimension n and append an all-$\frac{1}{2}$s vector to the target vector.

▶ **Lemma 21.** *For any $n' \leq n$, define the following transformations that map a basis B' of a rank-n' lattice $\mathcal{L}'$ to a basis B of a rank-n lattice $\mathcal{L}$, and a target vector $\boldsymbol{t}'$ to a target vector $\boldsymbol{t}$:*

$$B := \begin{pmatrix} \frac{1}{2}B' & 0 \\ I_{n'} & 0 \\ 0 & I_{n-n'} \end{pmatrix}, \qquad \boldsymbol{t} := \frac{1}{2}\begin{pmatrix} \boldsymbol{t}' \\ \mathbf{1}_{n'} \\ \mathbf{1}_{n-n'} \end{pmatrix}, \tag{3.1}$$

and define

$$s_p = s_p(n) := \tfrac{1}{2}(n+1)^{1/p} \text{ for } p \in [1, \infty), \text{ and } s_\infty := 1/2. \tag{3.2}$$

Then:
1. *$N_p^o(\mathcal{L}, r, \mathbf{0}) \leq N_p^o(\mathbb{Z}^n, r, \mathbf{0})$ for all $r \geq 0$;*
2. *if there exists an $\boldsymbol{x} \in \{0,1\}^{n'}$ such that $\|B'\boldsymbol{x} - \boldsymbol{t}'\|_p \leq 1$, then $N_p(B \cdot \{0,1\}^n, s_p, \boldsymbol{t}) \geq 2^{n-n'}$;*
3. *if $\operatorname{dist}_p(\boldsymbol{t}', \mathcal{L}') > 1$ then $\operatorname{dist}_p(\boldsymbol{t}, \mathcal{L}) > s_p$.*

Proof. Item 1 follows immediately by construction of B, because vectors $\boldsymbol{v}' = (\frac{1}{2}B'\boldsymbol{x}, \boldsymbol{x}, \boldsymbol{y}) \in \mathcal{L}$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^n$ correspond bijectively to vectors $\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z}^n$, and $\|\boldsymbol{v}\|_p \leq \|\boldsymbol{v}'\|_p$.

For Item 2, for every $\boldsymbol{y} \in \{0,1\}^{n-n'}$, the vector $\boldsymbol{v} := (\frac{1}{2}B'\boldsymbol{x}, \boldsymbol{x}, \boldsymbol{y}) \in \mathcal{L}$ satisfies

$$\|\boldsymbol{v} - \boldsymbol{t}\|_p^p = \frac{\|B'\boldsymbol{x} - \boldsymbol{t}'\|_p^p}{2^p} + \frac{n}{2^p} \leq s_p^p$$

for finite p, and $\|\boldsymbol{v} - \boldsymbol{t}\|_\infty = \max(\frac{1}{2}\|B'\boldsymbol{x} - \boldsymbol{t}'\|_\infty, \frac{1}{2}) = \frac{1}{2} = s_\infty$. The claim follows.

For Item 3, for finite p we have

$$\mathrm{dist}_p(\boldsymbol{t}, \mathcal{L})^p \geq \frac{\mathrm{dist}_p(\boldsymbol{t}', \mathcal{L}')^p}{2^p} + \frac{n}{2^p} > \frac{n+1}{2^p} = s_p^p \ ,$$

and for $p = \infty$ we immediately have $\mathrm{dist}_\infty(\boldsymbol{t}, \mathcal{L}) \geq \frac{1}{2}\,\mathrm{dist}_\infty(\boldsymbol{t}', \mathcal{L}') > \frac{1}{2} = s_\infty$, as needed. ◀

▶ **Corollary 22.** *For any $p \in [1, \infty]$, $\alpha > 0$, and $\mathrm{poly}(n')$-bounded $n \geq n'$, there is a deterministic Karp reduction from GapCVP'_p (resp., with preprocessing) in rank n' to the decision version of (S,T)-$\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank n, where $S(n) = N_p^o(\mathbb{Z}^n \setminus \{\boldsymbol{0}\}, s_p/\alpha, \boldsymbol{0})$ for s_p as defined in Equation (3.2), and $T(n) = 2^{n-n'}$.*

Proof. Given an input GapCVP'_p instance $(B', \boldsymbol{t}')$, the reduction simply outputs $(B, r = s_p/\alpha, \boldsymbol{t})$, where $B, \boldsymbol{t}$ are as in Equation (3.1). Observe that this is also valid for the preprocessing problems because B and r depend only on B'. Correctness follows immediately by Lemma 21. ◀

3.3 Setting Parameters

We now investigate the relationship among the choice of ℓ_p norm (for finite p), the BDD relative distance α, and the rank ratio $C := n/n'$, subject to the constraint

$$N_p^o(\mathbb{Z}^n, s_p/\alpha, \boldsymbol{0}) \leq 2^{n-n'}/10 = T(n)/10 \ , \tag{3.3}$$

so that the reductions in Corollary 22 and Theorem 19 can be composed. For $p \in [1, \infty)$ and $C > 1$, define

$$\alpha_{p,C}^* := \inf\{\alpha^* > 0 : \min_{\tau > 0} \exp(\tau/(2\alpha^*)^p) \cdot \Theta_p(\tau) \leq 2^{1-1/C}\} \ , \tag{3.4}$$

$$\alpha_p^* := \lim_{C \to \infty} \alpha_{p,C}^* = \inf\{\alpha^* > 0 : \min_{\tau > 0} \exp(\tau/(2\alpha^*)^p) \cdot \Theta_p(\tau) \leq 2\} \ . \tag{3.5}$$

These quantities are well defined because for any $C > 1$ we have $2^{1-1/C} > 1$, so the inequality in Equation (3.4) is satisfied for sufficiently large τ and α^*. Moreover, it is straightforward to verify that $\alpha_{p,C}^*$ is strictly decreasing in both p and C, and α_p^* is strictly decreasing in p. Although it is not clear how to solve for these quantities in closed form, it is possible to approximate them numerically to good accuracy (see Figure 1), and to get quite tight closed-form upper bounds (see Lemma 27). We now show that to satisfy Equation (3.3) it suffices to take any constant $\alpha > \alpha_{p,C}^*$.

▶ **Corollary 23.** *For any $p \in [1, \infty)$, $C \geq 1$, and constant $\alpha > \alpha_{p,C}^*$ (Equation (3.4)), there is a deterministic Karp reduction from GapCVP'_p (resp., with preprocessing) in rank n' to the decision version of (S,T)-$\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank $n = Cn'$, where $S(n) = T(n)/10$ and $T(n) = 2^{(1-1/C)n}$.*

Proof. Recalling that $s_p = \frac{1}{2}(n+1)^{1/p}$, by Proposition 13, $N_p^o(\mathbb{Z}^n, s_p/\alpha, \mathbf{0})$ is at most

$$\begin{aligned} N_p(\mathbb{Z}^n, s_p/\alpha, \mathbf{0}) &\le \min_{\tau>0} \exp(\tau \cdot (s_p/\alpha)^p) \cdot \Theta_p(\tau)^n \\ &= \min_{\tau>0} \exp(\tau \cdot (n+1)/(2\alpha)^p) \cdot \Theta_p(\tau)^n \\ &= \Big(\min_{\tau>0} \exp(\tau/(n(2\alpha)^p)) \cdot \exp(\tau/(2\alpha)^p) \cdot \Theta_p(\tau)\Big)^n . \end{aligned}$$

Because $\alpha > \alpha^*_{p,C}$, we have that $\min_{\tau>0} \exp(\tau/(2\alpha)^p) \cdot \Theta_p(\tau)$ is a constant strictly less than $2^{1-1/C}$. So, $N_p^o(\mathbb{Z}^n, s_p/\alpha, \mathbf{0}) \le 2^{(1-1/C)n}/10 = T(n)/10$ for all large enough n. The claim follows from Corollary 22. ◀

▶ **Theorem 24.** *For any $p \in [1, \infty)$, $C \ge 1$, and constant $\alpha > \alpha^*_{p,C}$, there is a randomized Cook reduction with no false positives from GapCVP'_p (resp., with preprocessing) in rank n' to $\mathrm{BDD}_{p,\alpha}$ (resp., with preprocessing) in rank $n = Cn' + 1$. Furthermore, the same holds for $p = \infty$, $C = 1$, $\alpha = 1/2$, and the reduction is deterministic.*

Proof. For finite p, we simply compose the reductions from Corollary 23 and Theorem 19, with the trivial decision-to-search reduction for (S,T)-$\mathrm{BDD}_{p,\alpha}$ in between.

For $p = \infty$, we first invoke the deterministic reduction from Corollary 22, from GapCVP'_∞ in rank n' to (S,T)-$\mathrm{BDD}_{\infty,1/2}$ in rank $Cn' = n'$, where $S = N_\infty^o(\mathbb{Z}^n \setminus \{\mathbf{0}\}, 1, \mathbf{0}) = 0$ and $T = 2^0 > 0$. By Lemma 18, the latter problem reduces deterministically to $\mathrm{BDD}_{\infty,1/2}$ in rank $n' + 1$.

Lastly, all of these reductions work for the preprocessing problems as well, because their component reductions do. ◀

3.4 Putting it all Together

We now combine our reductions from GapCVP' to BDD with prior hardness results for GapCVP' (stated below in Theorem 25) to obtain our ultimate hardness theorems for BDD. We first recall relevant known hardness results for GapCVP'_p and $\mathrm{GapCVPP}'_p$.

▶ **Theorem 25** ([23, 10, 2])**.** *The following hold for GapCVP'_p and $\mathrm{GapCVPP}'_p$ in rank n:*

1. *For every $p \in [1, \infty]$, GapCVP'_p is* NP*-hard, and $\mathrm{GapCVPP}'_p$ has no polynomial-time (preprocessing) algorithm unless* NP $\subseteq$ P/Poly.
2. *For every $p \in [1, \infty]$, there is no $2^{o(n)}$-time randomized algorithm for GapCVP'_p unless randomized ETH fails.*
3. *For every $p \in [1, \infty] \setminus \{2\}$, there is no $2^{o(n)}$-time algorithm for $\mathrm{GapCVPP}'_p$, and there is no $2^{o(\sqrt{n})}$-time algorithm for $\mathrm{GapCVPP}'_2$, unless non-uniform ETH fails.*
4. *For every $p \in [1, \infty] \setminus 2\mathbb{Z}$ and every $\varepsilon > 0$, there is no $2^{(1-\varepsilon)n}$-time randomized algorithm for GapCVP'_p (respectively, $\mathrm{GapCVPP}'_p$) unless randomized SETH (resp., non-uniform SETH) fails.*

▶ Remark 26. Several of the above results are stated slightly differently from what appears in [23, 10, 2]. First, all of the above results for GapCVP'_p (respectively, $\mathrm{GapCVPP}'_p$) are instead stated for GapCVP_p (resp., $\mathrm{GapCVPP}_p$). However, inspection shows that the reductions are indeed to GapCVP'_p or $\mathrm{GapCVPP}'_p$, so this difference is immaterial.

Second, the above statements ruling out *randomized* algorithms for GapCVP'_p assuming randomized (S)ETH are instead phrased in [10, 2] as ruling out *deterministic* algorithms for GapCVP'_p assuming deterministic (S)ETH. However, because these results are proved via deterministic reductions, randomized algorithms for GapCVP'_p have the consequences claimed above.

Third, the above results for $\mathrm{GapCVPP}'_p$ follow from the reductions given in (the proofs of) [23], [2, Theorem 4.3], [10, Theorem 1.4 and Lemma 6.1], and [2, Theorem 4.6]. However, those reductions all prove hardness for the variant of $\mathrm{GapCVPP}'_p$ where the distance threshold r is part of the *query* input, rather than the preprocessing input. Inspection of [2, Theorem 4.6] shows that r is fixed in the output instance, so this difference is immaterial in that case. We next describe how to handle this difference for the remaining cases. Below we give, for any $p \in [1, \infty)$, a straightforward rank-preserving mapping reduction (in the sense of Definition 5) from the variant of $\mathrm{GapCVPP}'_p$ where the distance threshold r is part of the query input to the variant where it is part of the preprocessing input, assuming that r is always at most some r^* that depends only on B, and whose length $\log r^*$ is polynomial in the length of B. Inspection shows that such an r^* does indeed exist for the reductions given in [23], [2, Theorem 4.3], and [10, Lemma 6.1], which handles the second difference for those cases.

The mapping reduction (R_P, R_Q) in question maps $(B, (\boldsymbol{t}, r)) \mapsto ((B', r^*), \boldsymbol{t}')$ as follows. First, R_P takes B as input, and sets $B' := \binom{B}{\boldsymbol{0}^t}$; it also outputs $\sigma' = r^*$ as side information for R_Q. Then, R_Q takes $(\boldsymbol{t}, r)$ and r^* as input, and outputs $\boldsymbol{t}' := (\boldsymbol{t}, ((r^*)^p - r^p)^{1/p})$. Using the guarantee that $r^* \geq r$, it is straightforward to check that the output instance $((B', r^*), \boldsymbol{t}')$ is a YES instance (respectively, NO instance) if the input instance $(B, (\boldsymbol{t}, r))$ is a YES instance resp., NO instance, as required.

Finally, we again remark that several of the hardness results in Theorem 25 in fact hold under weaker versions of randomized or non-uniform (S)ETH that relate to Max-3-SAT (respectively, Max-k-SAT), instead of 3-SAT (resp. k-SAT). Therefore, it is straightforward to obtain corresponding hardness results for BDD(P) under these weaker assumptions as well.

We can now prove our main theorem, restated from the introduction:

▶ **Theorem 1.** *The following hold for* $\mathrm{BDD}_{p,\alpha}$ *and* $\mathrm{BDDP}_{p,\alpha}$ *in rank n:*

1. *For every $p \in [1, \infty)$ and constant $\alpha > \alpha_p^*$ (where $\alpha_p^* \leq \frac{1}{2} \cdot 4.6723^{1/p}$), and for $(p, \alpha) = (\infty, 1/2)$, there is no polynomial-time algorithm for $\mathrm{BDD}_{p,\alpha}$ (respectively, $\mathrm{BDDP}_{p,\alpha}$) unless $\mathsf{NP} \subseteq \mathsf{RP}$ (resp., $\mathsf{NP} \subseteq \mathsf{P/Poly}$).*
2. *For every $p \in [1, \infty)$ and constant $\alpha > \min\{\alpha_p^*, \alpha_2^*\}$, and for $(p, \alpha) = (\infty, 1/2)$, there is no $2^{o(n)}$-time algorithm for $\mathrm{BDD}_{p,\alpha}$ unless randomized ETH fails.*
3. *For every $p \in [1, \infty) \setminus \{2\}$ and constant $\alpha > \alpha_p^*$, and for $(p, \alpha) = (\infty, 1/2)$, there is no $2^{o(n)}$-time algorithm for $\mathrm{BDDP}_{p,\alpha}$ unless non-uniform ETH fails.*
   Moreover, for every $p \in [1, \infty]$ and $\alpha > \alpha_2^*$ there is no $2^{o(\sqrt{n})}$-time algorithm for $\mathrm{BDDP}_{p,\alpha}$ unless non-uniform ETH fails.
4. *For every $p \in [1, \infty) \setminus 2\mathbb{Z}$ and constants $C > 1$, $\alpha > \alpha_{p,C}^*$, and $\epsilon > 0$, and for $(p, C, \alpha) = (\infty, 1, 1/2)$, there is no $2^{n(1-\epsilon)/C}$-time algorithm for $\mathrm{BDD}_{p,\alpha}$ (respectively, $\mathrm{BDDP}_{p,\alpha}$) unless randomized SETH (resp., non-uniform SETH) fails.*

Proof. For BDD, each item of the theorem follows from the corresponding item of Theorem 25, followed by Theorem 24 and then (where needed) rank-preserving norm embeddings from ℓ_2 to ℓ_p [28]. (Also, Lemma 27 below provides the upper bound on α_p^*.) The claims for BDDP follow similarly, combined with the well-known fact that $\mathsf{P/Poly} = \mathsf{BPP/Poly}$ and Corollary 16.[4] ◀

[4] In fact, $\mathsf{P/Poly} = \mathsf{BPP/Poly}$ also follows as a corollary of the more general derandomization result in Lemma 3.

3.5 An Upper Bound on $\alpha^*_{p,C}$ and α^*_p

We conclude with closed-form upper bounds on $\alpha^*_{p,C}$ and α^*_p. The main idea is to replace $\Theta_p(\tau)$ with an upper bound of $\Theta_1(\tau)$ (which has a closed-form expression) in Equations (3.4) and (3.5), then directly analyze the value of $\tau > 0$ that minimizes the resulting expressions. This leads to quite tight bounds (and also yields tighter bounds than the techniques used in the proof of [5, Claim 4.4], which bounds a related quantity). For example, $\alpha^*_2 \approx 1.05006$, and the upper bound in Lemma 27 gives $\alpha^*_2 \le 1.08078$; similarly, $\alpha^*_5 \approx 0.672558$ and the upper bound in Lemma 27 gives $\alpha^*_5 \le 0.680575$.

▶ **Lemma 27.** *Define*

$$g(\sigma, \tau) := \exp(\tau/\sigma) \cdot \left(\frac{2}{1 - \exp(-\tau)} - 1 \right)$$

and $\tau^*(\sigma) := \operatorname{arcsinh}(\sigma) = \ln(\sigma + \sqrt{1 + \sigma^2})$. *Let* σ^* *and* σ^*_C *for* $C > 1$ *be the (unique) constants for which* $g(\sigma^*, \tau^*(\sigma^*)) = 2$ *and* $g(\sigma^*_C, \tau^*(\sigma^*_C)) = 2^{1-1/C}$. *Then for any* $p \in [1, \infty)$, *we have*

$$\alpha^*_{p,C} \le \frac{1}{2} \cdot (\sigma^*_C)^{1/p} \quad \text{and} \quad \alpha^*_p \le \frac{1}{2} \cdot (\sigma^*)^{1/p} \le \frac{1}{2} \cdot 4.6723^{1/p} \ .$$

In particular, $\alpha^*_{p,C} \to 1/2$ *as* $p \to \infty$ *for any fixed* $C > 1$, *and therefore* $\alpha^*_p \to 1/2$ *as* $p \to \infty$.

Proof. For any $\tau > 0$, by the definition of $\Theta_p(\tau)$ and the formula for summing geometric series we have

$$\Theta_p(\tau) \le \Theta_1(\tau) = 1 + 2 \sum_{i=1}^{\infty} \exp(-\tau)^i = \frac{2}{1 - \exp(-\tau)} - 1 \ . \tag{3.6}$$

Define the objective function

$$f(p, \alpha) := \min_{\tau > 0} \exp(\tau/(2\alpha)^p) \cdot \Theta_p(\tau)$$

to be the expression that is upper-bounded in Equations (3.4) and (3.5). For any fixed $\alpha > 0$, set $\sigma := (2\alpha)^p$. Applying Equation (3.6), it follows that $f(p, \alpha) \le g(\sigma, \tau)$ for any $\tau > 0$. This implies that if there exists some $\tau > 0$ satisfying $g(\sigma, \tau) \le 2$ then $\alpha^*_p \le \frac{1}{2}\sigma^{1/p}$, and similarly, if $g(\sigma, \tau) \le 2^{1-1/C}$ then $\alpha^*_{p,C} \le \frac{1}{2}\sigma^{1/p}$.

By standard calculus,

$$\frac{\partial g}{\partial \tau} = \frac{e^{\tau/\sigma}}{1 - e^{-\tau}} \cdot \Big((1 + e^{-\tau})/\sigma - 2e^{-\tau}/(1 - e^{-\tau}) \Big) \ .$$

Setting the right-hand side of the above expression equal to 0 and solving for τ yields the single real solution

$$\tau = \tau^*(\sigma) = \operatorname{arcsinh}(\sigma) = \ln(\sigma + \sqrt{1 + \sigma^2}) \ ,$$

which is a local minimum, and therefore a global minimum of $g(\sigma, \tau)$ for any fixed $\sigma > 0$.

Define the univariate function $g^*(\sigma) := g(\sigma, \tau^*(\sigma))$. The fact that σ^* and σ^*_C exist and are unique follows by noting that $\lim_{\sigma \to 0^+} g^*(\sigma) = \infty$, that $\lim_{\sigma \to \infty} g^*(\sigma) = 1$, and that $g^*(\sigma)$ is strictly decreasing in $\sigma > 0$. By definition of σ^* (respectively, σ^*_C), it follows that $g^*(\sigma^*) = 2$ for $\alpha = \frac{1}{2}(\sigma^*)^{1/p}$, and $g^*(\sigma^*_C) = 2^{1-1/C}$ for $\alpha = \frac{1}{2}(\sigma^*_C)^{1/p}$, as desired. Moreover, one can check numerically that $\sigma^* \le 4.6723$. ◀

References

1 Leonard M. Adleman. Two theorems on random polynomial time. In *FOCS*, pages 75–83, 1978.

2 Divesh Aggarwal, Huck Bennett, Alexander Golovnev, and Noah Stephens-Davidowitz. Fine-grained hardness of CVP(P)— Everything that we can prove (and nothing else), 2019. arXiv:1911.02440.

3 Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the shortest vector problem in 2^n time using discrete Gaussian sampling: Extended abstract. In *STOC*, pages 733–742, 2015.

4 Divesh Aggarwal, Daniel Dadush, and Noah Stephens-Davidowitz. Solving the closest vector problem in 2^n time – the discrete Gaussian strikes again! In *FOCS*, pages 563–582, 2015.

5 Divesh Aggarwal and Noah Stephens-Davidowitz. (Gap/S)ETH hardness of SVP. In *STOC*, pages 228–238, 2018.

6 Divesh Aggarwal and Noah Stephens-Davidowitz. Just take the average! An embarrassingly simple 2^n-time algorithm for SVP (and CVP). In *Symposium on Simplicity in Algorithms*, volume 61, pages 12:1–12:19, 2018.

7 Miklós Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In *STOC*, pages 10–19, 1998.

8 Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.*, 54(2):317–331, 1997. doi:10.1006/jcss.1997.1472.

9 Shi Bai, Damien Stehlé, and Weiqiang Wen. Improved reduction from the bounded distance decoding problem to the unique shortest vector problem in lattices. In *ICALP*, pages 76:1–76:12, 2016.

10 Huck Bennett, Alexander Golovnev, and Noah Stephens-Davidowitz. On the quantitative hardness of CVP. In *FOCS*, 2017.

11 Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. On the closest vector problem with a distance guarantee. In *IEEE Conference on Computational Complexity*, pages 98–109, 2014.

12 N. D. Elkies, A. M. Odlyzko, and J. A. Rush. On the packing densities of superballs and other bodies. *Inventiones mathematicae*, 105:613–639, December 1991.

13 Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

14 Ishay Haviv and Oded Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. *Theory of Computing*, 8(1):513–531, 2012. Preliminary version in STOC 2007.

15 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

16 Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, 2005. Preliminary version in FOCS 2004.

17 Subhash Khot. Hardness of approximating the shortest vector problem in high ℓ_p norms. *J. Comput. Syst. Sci.*, 72(2):206–219, 2006.

18 Ravi Kumar and D. Sivakumar. On the unique shortest lattice vector problem. *Theor. Comput. Sci.*, 255(1-2):641–648, 2001.

19 Yi-Kai Liu, Vadim Lyubashevsky, and Daniele Micciancio. On bounded distance decoding for general lattices. In *APPROX-RANDOM*, pages 450–461, 2006.

20 Vadim Lyubashevsky and Daniele Micciancio. On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In *CRYPTO*, pages 577–594, 2009.

21 J. E. Mazo and A. M. Odlyzko. Lattice points in high-dimensional spheres. *Monatshefte für Mathematik*, 110:47–61, March 1990.

22 Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM J. Comput.*, 30(6):2008–2035, 2000. Preliminary version in FOCS 1998.

23 Daniele Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Trans. Information Theory*, 47(3):1212–1215, 2001. doi:10.1109/18.915688.

24 Daniele Micciancio. Efficient reductions among lattice problems. In *SODA*, pages 84–93, 2008.

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

26 Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342, 2009.

27 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):1–40, 2009. Preliminary version in STOC 2005.

28 Oded Regev and Ricky Rosen. Lattice problems and norm embeddings. In *STOC*, pages 447–456, 2006.

29 Noah Stephens-Davidowitz. Discrete Gaussian sampling reduces to CVP and SVP. In *SODA*, pages 1748–1764, 2016. doi:10.1137/1.9781611974331.ch121.

30 Noah Stephens-Davidowitz and Vinod Vaikuntanathan. SETH-hardness of coding problems. In *FOCS*, pages 287–301, 2019.

31 Peter van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical Report 81-04, University of Amsterdam, 1981.