Approximate CVP_p in Time $2^{0.802\,n}$

Friedrich Eisenbrand
Ecole Polytechnique Fédérale de Lausanne, Switzerland
friedrich.eisenbrand@epfl.ch

Moritz Venzin
Ecole Polytechnique Fédérale de Lausanne, Switzerland
moritz.venzin@epfl.ch

Abstract

We show that a constant factor approximation of the shortest and closest lattice vector problem w.r.t. any ℓ_p-norm can be computed in time $2^{(0.802+\varepsilon)\,n}$. This matches the currently fastest constant factor approximation algorithm for the shortest vector problem w.r.t. ℓ_2. To obtain our result, we combine the latter algorithm w.r.t. ℓ_2 with geometric insights related to coverings.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases Shortest and closest vector problem, approximation algorithm, sieving, covering convex bodies

Digital Object Identifier 10.4230/LIPIcs.ESA.2020.43

Funding The authors acknowledge support from the Swiss National Science Foundation within the project Lattice Algorithms and Integer Programming (Nr. 200021-185030).

Acknowledgements The authors would like to thank the reviewers for their careful reviews and suggestions. The second author would like to thank Christoph Hunkenschröder, Noah Stephens-Davidowitz and Márton Naszódi for inspiring discussions.

1 Introduction

The *shortest vector problem* (SVP) and the *closest vector problem* (CVP) are important algorithmic problems in the geometry of numbers. Given a rational lattice

$$\mathcal{L}(B) = \{B\mathbf{x} \,:\, \mathbf{x} \in \mathbb{Z}^n\}$$

with $B \in \mathbb{Q}^{n\times n}$ and a target vector $\mathbf{t} \in \mathbb{Q}^n$ the closest vector problem asks for lattice vector $\mathbf{v} \in \mathcal{L}(B)$ minimizing $\|\mathbf{t} - \mathbf{v}\|$. The *shortest vector problem* asks for a *nonzero* lattice vector $\mathbf{v} \in \mathcal{L}(B)$ of minimal norm. When using the ℓ_p norms for $1 \le p \le \infty$, we denote the problems by SVP_p resp. CVP_p.

Much attention has been devoted to the hardness of approximating SVP and CVP. In a long sequence of papers, including [42, 7, 32, 10, 18, 28, 23] it has been shown that SVP and CVP are hard to approximate to within almost polynomial factors under reasonable complexity assumptions. The best polynomial-time approximation algorithms have exponential approximation factors [29, 41, 8].

The first algorithm to solve CVP for any norm that has exponential running time in the dimension only was given by Lenstra [30]. The running time of his procedure is $2^{O(n^2)}$ times a polynomial in the encoding length. In fact, Lenstra's algorithm solves the more general *integer programming* problem. Kannan [27] improved this to $n^{O(n)}$ time and polynomial space. It took almost 15 years until Ajtai, Kumar and Sivakumar presented a randomized algorithm for SVP_2 with time and space $2^{O(n)}$ and a $2^{O(1+1/\varepsilon)n}$ time and space algorithm for $(1+\varepsilon)$-CVP_2 [8, 9]. Here $(1+\varepsilon)$-CVP_2 is the problem of finding a lattice vector, whose

distance to the target is at most $1+\varepsilon$ times the minimal distance. Blömer and Naewe [14] extended the randomized sieving algorithm of Ajtai et al. to solve SVP_p and obtain a $2^{O(n)}$ time and space exact algorithm for SVP_p and an $O(1+1/\varepsilon)^{2n}$ time algorithm to compute a $(1+\varepsilon)$ approximation for CVP_p. For CVP_∞, one has a faster approximation algorithm. Eisenbrand et al. [20] showed how to boost any constant approximation algorithm for CVP_∞ to a $(1+\varepsilon)$-approximation algorithm in time $O(\log(1+1/\varepsilon))^n$. Recently, this idea was adapted in [36] to all ℓ_p norms, showing that $(1+\varepsilon)$ approximate CVP_p can be solved in time $O(1+1/\varepsilon)^{n/\min(2,p)}$ by boosting the deterministic CVP algorithm for general (even asymmetric) norms with a running time of $(1+1/\varepsilon)^n$ that was developed by Dadush and Kun [16].

The first deterministic singly-exponential time and space algorithm for exact CVP_2 (and SVP_2) was developed by [33]. The fastest exact algorithms for SVP_2 and CVP_2 run in time and space $2^{n+o(n)}$ [3, 1, 6]. Single exponential time and space algorithms for exact CVP are only known for ℓ_2. Whether CVP and the more general integer programming problem can be solved in time $2^{O(n)}$ is a prominent mystery in algorithms.

Recently there has been exciting progress in understanding the *fined grained complexity* of exact and constant approximation algorithms for CVP [2, 12, 5]. Under the assumption of the *strong exponential time hypothesis (SETH)* and for $p \neq 0 \pmod 2$, exact CVP_p cannot be solved in time $2^{(1-\varepsilon)d}$. Here d is the *ambient dimension* of the lattice, which is the number of vectors in a basis of the lattice. Under the assumption of a *gap-version* of the strong exponential time hypothesis *(gap-SETH)* these lower bounds also hold for the approximate versions of CVP_p. More precisely, for each $\varepsilon > 0$ there exists a constant $\gamma_\varepsilon > 1$ such that there exits no $2^{(1-\varepsilon)d}$ algorithm that computes a γ_ε-approximation of CVP_p.

Unfortunately, the currently fastest algorithms for CVP_p resp. SVP_p do not match these lower bounds, even for large approximation factors. These algorithms are based on randomized sieving, [8, 9]. Many lattice vectors are generated that are then, during many stages, subtracted from each other to obtain shorter and shorter vectors w.r.t. ℓ_p (resp. any norm) until a short vector is found. However, the algorithm needs to start out with sufficiently many lattice vectors just to guarantee that two of them are close. This issue directly relates to the *kissing number* (w.r.t. some norm) which is the maximum number of unit norm balls that can be arranged so that they touch another given unit norm ball. In the setting of sieving, this is the number of vectors of length r that are needed to guarantee that the difference of two of them is strictly smaller than r. Among all known upper bounds on the kissing numbers, the best (i.e. smallest) upper bound is known for ℓ_2 and equals $2^{0.401n}$, [26]. For ℓ_2 the fastest such approximation algorithms require time $2^{0.802n}$ - the square of the kissing number w.r.t. ℓ_2. For ℓ_∞ the kissing number equals 3^n-1 which is also an upper bound on the kissing number for any norm. The current best constant factor approximation algorithms for SVP_∞ and CVP_∞ require time 3^n, their counterparts w.r.t. ℓ_p require even more time, see [4, 35]. This then suggests the question, originally raised by Aggarwal et al. in [2] for ℓ_∞, whether the kissing number w.r.t. ℓ_p is a natural lower bound on the running time of SVP_p resp. CVP_p.

Our results indicate otherwise. For constant approximation factors, we are able to reduce these problems w.r.t. ℓ_p to another lattice problem but w.r.t. ℓ_2. This directly improves the running time of the algorithms for ℓ_p norms that hinge on the kissing number. Furthermore, given that the development of algorithms for ℓ_2 has been much more dynamic than for arbitrary ℓ_p norms and the difficulty of establishing hardness results for ℓ_2, there is hope to find still faster algorithms for SVP_2 that may not even rely on the kissing number w.r.t. ℓ_2. It is likely that this would then improve the situation for ℓ_p norms as well.

Our *main results* are resumed in the following theorem.

▶ **Theorem.** *For each $\varepsilon > 0$, there exists a constant γ_ε such that a γ_ε approximate solution to CVP_p, as well as to SVP_p for $p \in [1, \infty]$ can be found in time $2^{(0.802+\varepsilon)n}$.*

Our main idea is to use coverings in order to obtain a constant factor approximation to the shortest resp. closest vector w.r.t. ℓ_p by using a (approximate) shortest vector algorithm w.r.t. ℓ_2. We need to distinguish between the cases $p \in [2, \infty]$ and $p \in [1, 2)$. For $p \in [2, \infty]$, we show that exponentially many short vectors w.r.t. ℓ_2 cannot all have large pairwise distance w.r.t. ℓ_p. This follows from a bound on the number of ℓ_p norm balls scaled by some constant that are required to cover the ℓ_2 norm ball of radius $n^{1/2-1/p}$. The final procedure is then to sieve w.r.t. ℓ_2 and to pick the smallest non zero pairwise difference w.r.t. ℓ_p of the (exponentially many) generated lattice vectors. This yields a constant factor approximation to the shortest resp. closest vector w.r.t. ℓ_p, $p \in [2, \infty]$. For $p \in [1, 2)$, we use a more direct covering idea. There is a collection of at most $2^{\varepsilon n}$ balls w.r.t. ℓ_2, whose union contains the ℓ_p norm ball but whose union is contained in the ℓ_p norm ball scaled by some constant. This leads to a simple algorithm for ℓ_p norms ($p \in [1, 2)$) by using the approximate closest vector algorithm w.r.t. ℓ_2 from this paper.

This paper is organized as follows. In Section 2 we present the main idea for $p = \infty$ that also applies to the case $p \geq 2$. In Section 3 we first reintroduce the list-sieve method originally due to [34] but with a slightly more general viewpoint, we resume this in Theorem 4. We then present in detail our approximate CVP_∞ resp. SVP_∞ algorithm and extend this idea resp. algorithm to ℓ_p, $p \geq 2$. This is Theorem 5. Finally, in Section 4, using the covering technique from Section 2 and our approximate CVP_2 algorithm from Section 3.1, we show how to solve approximate CVP_p for $p \in [1, 2)$. This is Theorem 8.

2 Covering balls with boxes

We now outline our main idea in the setting of an approximate SVP_∞ algorithm. Let us assume that the shortest vector of $\mathcal{L}$ w.r.t. ℓ_∞ is $\mathbf{s} \in \mathcal{L} \setminus \{0\}$. We can assume that the lattice is scaled such that $\|\mathbf{s}\|_\infty = 1$ holds. The euclidean norm of $\mathbf{s}$ is then bounded by $\sqrt{n}$. Suppose now that there is a procedure that, for some constant $\gamma > 1$ independent of n, generates distinct lattice vectors $\mathbf{v}_1, \ldots, \mathbf{v}_N \in \mathcal{L}$ of length at most $\|\mathbf{v}_i\|_2 \leq \gamma\sqrt{n}$.

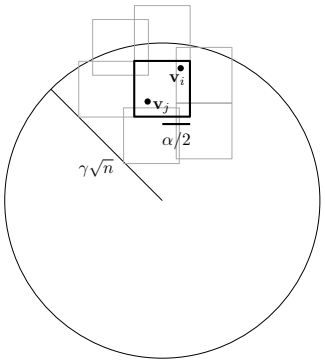

Figure 1 The difference $\mathbf{v}_i - \mathbf{v}_j$ is an α-approximate shortest vector w.r.t. ℓ_∞.

How large does the number of vectors N have to be such that we can guarantee that there exist two indices $i \neq j$ with

$$\|\mathbf{v}_i - \mathbf{v}_j\|_\infty \leq \alpha, \tag{1}$$

where $\alpha \geq 1$ is the approximation guarantee for SVP_∞ that we want to achieve? Suppose that N is larger than the minimal number of copies of the box $(\alpha/2)B_\infty^n$ that are required to cover the ball $\sqrt{n}B_2^n$. Here $B_p^n = \{x \in \mathbb{R}^n \,:\, \|x\|_p \leq 1\}$ denotes the unit ball w.r.t. the ℓ_p-norm. Then, by the pigeon-hole principle, two different vectors $\mathbf{v}_i$ and $\mathbf{v}_j$ must be in the same box. Their difference satisfies (1) and thus is an α-approximate shortest vector w.r.t. ℓ_∞, see Figure 1.

Thus we are interested in the *translative covering number* $N(\sqrt{n}B_2^n, aB_\infty^n)$, which is the number of translated copies of the box aB_∞^n that are needed to cover the ℓ_2-ball of radius $\sqrt{n}$. In the setting above, a is the constant $\alpha/(2\gamma)$. For this procedure to be efficient, we need $N(\sqrt{n}B_2^n, aB_\infty^n)$ to be relatively small for a large enough - this is equivalent to decreasing the number of vectors N we need to generate by worsening (increasing) the approximation guarantee α. Since $2^{O(n)}\operatorname{vol}(B_\infty^n) = \operatorname{vol}(\sqrt{n}B_2^n)$ and by a simple covering argument, we have that $N(\sqrt{n}B_2^n, B_\infty^n) \leq 2^{Cn}$. This gives hope that by taking a large enough (but independent of n), we can decrease $N(\sqrt{n}B_2^n, aB_\infty^n)$ to, say, $2^{0.401n}$ or $2^{\varepsilon n}$ for $\varepsilon > 0$.

Covering problems like these have received considerable attention in the field of convex geometry, see [11, 37]. These techniques rely on the classical *set-cover problem* and the logarithmic integrality gap of its standard LP-relaxation, see, e.g. [43, 15]. To keep this paper self-contained, we briefly explain how this can be applied to our setting.

If we cover the finite set $(1/n)\mathbb{Z}^n \cap \sqrt{n}B_2^n$ with cubes whose centers are on the grid $(1/n)\mathbb{Z}^n$, then by increasing the side-length of those cubes by an additive $1/n$, one obtains a full covering of $\sqrt{n}B_2^n$. Thus we can focus on the corresponding *set-covering problem* with ground set $U = (1/n)\mathbb{Z}^n \cap \sqrt{n}B_2^n$ and sets

$$S_t = U \cap aB_\infty^n + t,\, t \in (1/n)\,\mathbb{Z}^n,$$

ignoring empty sets. An element of the ground set is contained in exactly $|(1/n)\mathbb{Z}^n \cap aB_\infty^n|$ many sets. Therefore, by assigning each element of the ground set the fractional value $1/|(1/n)\mathbb{Z}^n \cap aB_\infty^n|$, one obtains a feasible fractional covering. The weight of this fractional covering is

$$\frac{T}{|(1/n)\mathbb{Z}^n \cap aB_\infty^n|}$$

where T is the number of sets. Clearly, if a cube intersects $\sqrt{n}B_2^n$, then its center is contained in the *Minkowski sum* $\sqrt{n}B_2^n + aB_\infty^n$ and thus the weight of the fractional covering is

$$\frac{|(\sqrt{n}B_2^n + aB_\infty^n) \cap \frac{1}{n}\mathbb{Z}^n|}{|\frac{1}{n}\mathbb{Z}^n \cap aB_\infty^n|} = O\left(\frac{\operatorname{vol}(\sqrt{n}B_2^n + aB_\infty^n)}{\operatorname{vol}(aB_\infty^n)}\right)$$

Since the size of the ground-set is bounded by $n^{O(n)}$ and since the integrality gap of the set-cover LP is at most the logarithm of this size, one obtains

$$N(\sqrt{n}B_2^n, aB_\infty^n) \leq \operatorname{poly}(n)\,\frac{\operatorname{vol}(\sqrt{n}B_2^n + aB_\infty^n)}{\operatorname{vol}(aB_\infty^n)} \tag{2}$$

By *Steiner's formula*, see [22, 40, 24], the volume of $K + tB_2^n$ is a polynomial in t, with coefficients $V_j(K)$ only depending on the convex body K:

$$\operatorname{vol}(K + tB_2^n) = \sum_{j=0}^{n} V_j(K)\operatorname{vol}(B_2^{n-j})t^{n-j}$$

For $K = aB_\infty^n$, $V_j(K) = (2a)^j\binom{n}{j}$. Setting $t = \sqrt{n}$, the resulting expression has been evaluated in [25, Theorem 7.1].

▶ **Theorem 1** ([25]). *Denote by* H *the binary entropy function and let* $\phi \in (0,1)$ *the unique solution to*

$$\frac{1-\phi^2}{\phi^3} = \frac{2a^2}{\pi} \tag{3}$$

Then

$$\mathrm{vol}(aB_\infty^n + \sqrt{n}B_2^n) = O(2^{n[\mathrm{H}(\phi)+(1-\phi)\log(2a)+\frac{\phi}{2}\log(\frac{2\pi e}{\phi})]})$$

Using this bound in inequality (2) and simplifying, we find

$$N(\sqrt{n}B_2^n, aB_\infty^n) \leq \mathrm{poly}(n)\, 2^{n[\mathrm{H}(\phi)+\frac{\phi}{2}\log(\frac{2\pi e}{\phi})]}$$

Both $\mathrm{H}(\phi)$ and $\frac{\phi}{2}\log(\frac{2\pi e}{\phi})$ decrease to 0 as ϕ decreases to 0. Since ϕ, the unique solution to (3), satisfies $\phi \leq \sqrt[3]{(\pi/2)}a^{-\frac{2}{3}}$, we obtain the following bound.

▶ **Lemma 2.** *For each* $\varepsilon > 0$*, there exists* $a_\varepsilon \in \mathbb{R}_{>0}$ *independent of* n*, such that*

$$N(\sqrt{n}B_2^n, a_\varepsilon B_\infty^n) \leq 2^{\varepsilon n}.$$

Going back to the idea for an approximate SVP_∞ algorithm, we will use Lemma 2 with $\varepsilon = 0.401$. If we generate $2^{0.401n}$ distinct lattice vectors of euclidean length at most $\gamma\sqrt{n}$, then there must exist a pair of lattice vectors with pairwise distance w.r.t. ℓ_∞ shorter than $2\gamma a_{0.401}$. We find it by trying out all possible pairwise combinations, this takes time $2^{0.802n}$.

The main idea for approximate SVP_p is similar. Set $\tilde{\mathbf{s}}$ the shortest vector in $\mathcal{L}$ w.r.t. ℓ_p and scale the lattice so that $\|\tilde{\mathbf{s}}\|_p = 1$. The euclidean norm of $\tilde{\mathbf{s}}$ is bounded by $n^{1/2-1/p}$. Again, we can consider the question of how many different lattice vectors there have to be within a ball of radius $\gamma n^{1/2-1/p}$ so that we can guarantee that there exist two lattice vectors with constant pairwise distance w.r.t. ℓ_p. This leads us to consider the translative covering number $N(n^{1/2-1/p}B_2^n, aB_p^n)$. Since $n^{-1/p}B_\infty^n \subseteq B_p^n$, the following is immediate from Lemma 2.

▶ **Lemma 3.** *For each* $\varepsilon > 0$*, there exists* $a_\varepsilon \in \mathbb{R}_{>0}$ *independent of* n*, such that*

$$N(n^{1/2-1/p}B_2^n, a_\varepsilon B_p^n) \leq 2^{\varepsilon n}.$$

3 Approximate CVP_p for $p \geq 2$

We now describe our main contribution. As we mentioned already, SVP_2 can be approximated up to a constant factor in time $2^{(0.802+\varepsilon)n}$ for each $\varepsilon > 0$. This follows from a careful analysis of the *list sieve* algorithm of Micciancio and Voulgaris [34], see [31, 38]. The running time and space of this algorithm is directly related to the *kissing number* of the ℓ_2-norm. The running time is the square of the best known upper bound by Kabatiansky and Levenshtein [26].

The main insight of our paper is that the current list-sieve variants can be used to approximate SVP_p and CVP_p by testing all pairwise differences of the generated lattice vectors.

3.1 List sieve

We begin by describing the list-sieve method [34] to a level of detail that is necessary to understand our main result. Our exposition follows closely the one given in [38]. Let $\mathcal{L}(B)$ be a given lattice and $\mathbf{s} \in \mathcal{L}$ be an unknown lattice vector. This unknown lattice vector $\mathbf{s}$ is typically the shortest, respectively closest vector in $\mathcal{L}(B)$.

The list-sieve algorithm has two stages. The input to the *first stage* of the algorithm is an LLL-reduced lattice basis B of $\mathcal{L}(B)$, a constant $\varepsilon > 0$ and a guess μ on the length of $\mathbf{s}$ that satisfies

$$\|\mathbf{s}\|_2 \leq \mu \leq (1 + 1/n)\|\mathbf{s}\|_2. \tag{4}$$

The first stage then constructs a list of lattice vectors $L \subseteq \mathcal{L}(B)$ that is random. This list of lattice vectors is then passed on to the second stage of the algorithm.

The *second stage* of the algorithm proceeds by sampling points $\mathbf{y}_1, \ldots, \mathbf{y}_N$ uniformly and independently at random from the ball

$$(\xi_\varepsilon \cdot \mu)B_2^n,$$

where ξ_ε is an explicit constant depending on ε only. It then transforms these points via a deterministic algorithm $\texttt{ListRed}_L$ into lattice points

$$\texttt{ListRed}_L(\mathbf{y}_1), \ldots, \texttt{ListRed}_L(\mathbf{y}_N) \in \mathcal{L}(B).$$

The deterministic algorithm $\texttt{ListRed}_L$ uses the list $L \subseteq \mathcal{L}(B)$ from the first stage.

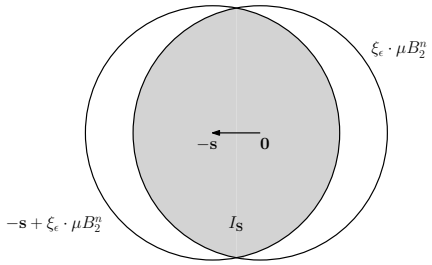

Figure 2 The lens $I_\mathbf{s}$.

As we mentioned above, the list $L \subseteq \mathcal{L}(B)$ that is used by the deterministic algorithm $\texttt{ListRed}_L$ is random. We will show the following theorem in the next section. The novelty compared to the literature is the reasoning about *pairwise differences* lying in *centrally symmetric* sets. In this theorem, $\varepsilon > 0$ is an arbitrary constant, ξ_ε as well as c_ε are explicit constants and K is some centrally symmetric set. Furthermore, we assume that μ satisfies (4).

The theorem reasons about an area $I_\mathbf{s}$ that is often referred as the *lens*, see Figure 2. The lens was introduced by Regev as a conceptual modification to facilitate the proof of the original AKS algorithm [39].

$$I_\mathbf{s} = (\xi_\varepsilon \cdot \mu)B_2^n \cap (-\mathbf{s} + (\xi_\varepsilon \cdot \mu)B_2^n) \tag{5}$$

▶ **Theorem 4.** *With probability at least* $1/2$*, the list* L *that was generated in the first stage satisfies the following. If* $\mathbf{y}_1, \cdots, \mathbf{y}_N$ *are chosen independently and uniformly at random within* $B_2^n(0, \xi_\varepsilon\mu)$ *then*

i) *The probability of the event that two different samples* $\mathbf{y}_i, \mathbf{y}_j$ *satisfy*

$$\mathbf{y}_i, \mathbf{y}_j \in I_\mathbf{s} \text{ and } \texttt{ListRed}_L(\mathbf{y}_i) - \texttt{ListRed}_L(\mathbf{y}_j) \in K$$

is at most twice the probability of the event that two different samples $\mathbf{y}_i, \mathbf{y}_j$ *satisfy*

$$\texttt{ListRed}_L(\mathbf{y}_i) - \texttt{ListRed}_L(\mathbf{y}_j) \in K + \mathbf{s}$$

ii) *For each sample* $\mathbf{y}_i$ *the probability of the event*

$$\|\texttt{ListRed}_L(\mathbf{y}_i)\|_2 \leq c_\varepsilon \|\mathbf{s}\|_2 \text{ and } \mathbf{y}_i \in I_\mathbf{s}$$

is at least $2^{-\varepsilon n}$.

The complete procedure, i.e. the construction of the list L in stage one and applying $\texttt{ListRed}_L$ to the N samples $\mathbf{y}_1, \ldots, \mathbf{y}_n$ in stage two takes time $N2^{(0.401+\varepsilon)n} + 2^{(0.802+\varepsilon)n}$ and space $N + 2^{(0.401+\varepsilon)n}$.

The proof of Theorem 4 follows verbatim from Pujol and Stehlé [38], see also [31]. In [38], $\mathbf{s}$ is a shortest vector w.r.t. ℓ_2. But this fact is never used in the proof and in the analysis. Part ii) follows from Lemma 5 and Lemma 6 in [38]. Their probability of a sample being in the lens $I_\mathbf{s} \subseteq \xi \|\mathbf{s}\|_2 B_2^n$ depends only on ξ (corresponding to our ξ_ε). By choosing ξ large enough, this happens with probability at least $2^{-\varepsilon n}$. Their Lemma 6 then guarantees that the list L, with probability $1/2$, when $\mathbf{y}_i \sim I_\mathbf{s}$ is sampled uniformly, returns a lattice vector of length at most $r_0 \|\mathbf{s}\|_2$ (r_0 corresponds to our c_ε). This corresponds to part ii) in our setting. The size of their list (denoted by N_T) is bounded above by $2^{(0.401+\delta)n}$ where $\delta > 0$ decreases to 0 as the ratio r_0/ξ increases, this is their Lemma 4.

Finally, part i) also follows from Pujol and Stehlé [38]. It is in their proof of correctness, Lemma 7, involving the lens $I_\mathbf{s}$. We briefly comment on our general viewpoint. Given $\mathbf{y} \sim (\xi \cdot \mu)B_2^n$, the algorithm computes the linear combination w.r.t. to the lattice basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$

$$\mathbf{y} = \sum_{i=1}^n \lambda_i \mathbf{b}_i$$

and then the *remainder*

$$\mathbf{y} \pmod{\mathcal{L}} = \sum_{i=1}^n \lfloor \lambda_i \rfloor \mathbf{b}_i.$$

The important observation is that this remainder is the same for all vectors $\mathbf{y} + \mathbf{v}$, $\mathbf{v} \in \mathcal{L}$. Next, it keeps reducing (minus) the remainder w.r.t. the list, as long as the length decreases. This results in a vector of the form

$$-\mathbf{y} \pmod{\mathcal{L}} - \mathbf{v}_1 - \cdots - \mathbf{v}_k, \text{ for some } \mathbf{v}_i \in L.$$

The output $\texttt{ListRed}_L(\mathbf{y})$ is then

$$-\mathbf{y} \pmod{\mathcal{L}} - \mathbf{v}_1 - \cdots - \mathbf{v}_k + \mathbf{y} \in \mathcal{L}.$$

The algorithm bases its decisions on $\mathbf{y} \pmod{\mathcal{L}}$ and not on $\mathbf{y}$ directly. This is why one can imagine that, after $\mathbf{y} \pmod{\mathcal{L}}$ has been created, one applies a bijection τ of the ball $\tau(\cdot) : \xi\mu B_2^n \to \xi\mu B_2^n$ on $\mathbf{y}$ with probability $1/2$. For $\mathbf{y} \in I_\mathbf{s}$ one has $\tau(\mathbf{y}) = \mathbf{y} + \mathbf{s}$. We refer to [38] for the definition of τ. Since τ is a bijection and preserves the measure, the result of applying $\tau(\mathbf{y})$ with probability $1/2$ is distributed uniformly. This means that for $\mathbf{y} \in I_s$ this modified but equivalent procedure outputs $\texttt{ListRed}_L(\mathbf{y})$ or $\texttt{ListRed}_L(\mathbf{y}) + \mathbf{s}$, both with probability $1/2$. If $\texttt{ListRed}_L(\mathbf{y}_i) - \texttt{ListRed}_L(\mathbf{y}_j) \in K$, we toss a coin for i and j each. With probability $1/2$, their difference is in $\pm K + \mathbf{s}$.

3.2 Approximation to CVP_p and SVP_p for $p \in [2, \infty]$

We now combine Theorem 4 with the covering ideas presented in Section 2.

▶ **Theorem 5.** *For $p \geq 2$, there is a randomized algorithm that computes with constant probability a constant factor (depending on ε) approximation to CVP_p and SVP_p respectively. The algorithm runs in time $2^{(0.802+\varepsilon)n}$ and it requires space $2^{(0.401+\varepsilon)n}$.*

In short, the algorithm is the standard list-sieve algorithm with a slight twist: *Check all pairwise differences.*
We first present in detail the case $p = \infty$. Even though there is an approximation preserving reduction from SVP to CVP, [21], we present separately the case SVP and CVP to highlight the ideas from Section 2 and Theorem 4. The case $p \geq 2$ then follows from this, we briefly comment on it.

Proof for $p = \infty$. We assume that the list L that was computed in the first stage satisfies the properties described in Theorem 4. Recall that this is the case with probability at least $1/2$.
We first consider SVP_∞. By Lemma 2, there is $a > 0$ such that $N(\sqrt{n}B_2^n, aB_\infty^n) \leq 2^{0.401n}$. Let $\mathbf{s}$ be a shortest vector w.r.t. ℓ_∞ and let $\mu > 0$ such that $\|\mathbf{s}\|_2 \leq \mu < (1 + \frac{1}{n}) \|\mathbf{s}\|_2$ as above. Since $\|\mathbf{s}\|_2 \leq \sqrt{n} \|\mathbf{s}\|_\infty$ we have $N(c_\varepsilon \|\mathbf{s}\|_2 B_2^n, c_\varepsilon a \|\mathbf{s}\|_\infty B_\infty^n) \leq 2^{0.401n}$. This means that, if $\lceil 2^{0.401n} \rceil + 1$ lattice vectors are contained in the ball $c_\varepsilon \|\mathbf{s}\|_2 B_2^n$ at least two of them have ℓ_∞-distance bounded by $2c_\varepsilon a$ which is a constant.

Set $N = 2 \cdot \lceil 2^{(\varepsilon+0.401)n} + 1 \rceil$ and $\{\mathbf{y}_1, \ldots, \mathbf{y}_N\} \overset{iid}{\sim} B_2^n(0, \xi_\varepsilon \mu)$ uniformly and independently at random. By Theorem 4 ii) and by the Chebychev inequality, see [38], the following event has probability at least $1/2$.

(Event A): There is a subset $S \subseteq \{1, \ldots, N\}$ with $|S| = \lceil 2^{0.401n} \rceil + 1$ such that for each $i \in S$

$$\mathbf{y}_i \in I_{\mathbf{s}} \text{ and } \|\,\mathtt{ListRed}_L(\mathbf{y}_i)\|_2 \leq c_\varepsilon \|\mathbf{s}\|_2. \tag{6}$$

This event is the disjoint union of the event $A \cap B$ and $A \cap \overline{B}$, where B denotes the event where the vectors $\mathtt{ListRed}_L(\mathbf{y}_i)$, $\mathbf{y}_i \in I_{\mathbf{s}}$ are all distinct. Thus

$$\Pr(A) = \Pr(A \cap B) + \Pr(A \cap \overline{B}).$$

The probability of at least one of the events $A \cap B$ and $A \cap \overline{B}$ is bounded below by $1/4$. In the event $A \cap B$, there exists $i \neq j$ such that

$$\|\,\mathtt{ListRed}_L(\mathbf{v}_i) - \mathtt{ListRed}_L(\mathbf{v}_j)\|_\infty \leq 2c_\varepsilon a.$$

By Theorem 4 i) with $K = \{0\}$ one has

$$\Pr(A \cap \overline{B}) \leq 2 \Pr\left(\exists i \neq j \;:\; \mathtt{ListRed}_L(\mathbf{v}_i) - \mathtt{ListRed}_L(\mathbf{v}_j) = \mathbf{s}\right).$$

Therefore, with constant probability, there exist $i, j \in \{1, \ldots, N\}$ with

$$0 < \|\,\mathtt{ListRed}_L(\mathbf{y}_i) - \mathtt{ListRed}_L(\mathbf{y}_j)\|_\infty \leq 2c_\varepsilon a.$$

We try out all the pairs of N elements, which amounts to $N^2 = 2^{(0.802+\varepsilon')n}$ additional time.

We next describe how list-sieve yields a constant approximation for CVP_∞. Let $\mathbf{w} \in \mathcal{L}(B)$ be the closest lattice vector w.r.t. ℓ_∞ to $\mathbf{t} \in \mathbb{R}^n$ and let $\mu > 0$ such that $\|\mathbf{t} - \mathbf{w}\|_2 \leq \mu < (1 + \frac{1}{n}) \|\mathbf{t} - \mathbf{w}\|_2$. We use Kannan's embedding technique [27] and define a new lattice $\mathcal{L}'$ with basis

$$\widetilde{B} = \begin{pmatrix} B & \mathbf{t} \\ 0 & \frac{1}{n}\mu \end{pmatrix} \in \mathbb{Q}^{(n+1)\times(n+1)},$$

Finding the closest vector to $\mathbf{t}$ w.r.t. ℓ_∞ in $\mathcal{L}(B)$ amounts to finding the shortest vector w.r.t. ℓ_∞ in $\mathcal{L}'(\widetilde{B}) \cap \{\mathbf{x} \in \mathbb{R}^{n+1} \ : \ x_{n+1} = \frac{1}{n}\mu\}$. The vector $\mathbf{s} = (\mathbf{t} - \mathbf{w}, \frac{1}{n}\mu)$ is such a vector and its euclidean length is smaller than $(1 + \frac{1}{n})\mu$. Let $a > 0$ be such that

$$N(\sqrt{n}B_2^n, aB_\infty^n) \leq 2^{0.401n}.$$

This means that there is a covering of the n-dimensional ball $(c_\varepsilon \|\mathbf{s}\|_2)B_2^{n+1} \cap \{\mathbf{x} \in \mathbb{R}^{n+1} \ : \ x_{n+1} = 0\}$ by $2^{0.401n}$ translated copies of K, where

$$K = (c_\varepsilon \cdot a(1 + 1/n)\|\mathbf{s}\|_\infty)B_\infty^{n+1} \cap \{\mathbf{x} \in \mathbb{R}^{n+1} \ : \ x_{n+1} = 0\}. \tag{7}$$

(The factor $(1 + 1/n)$ is a reminiscent of the embedding trick, $\mathbf{s}$ is $n + 1$ dimensional.) Similarly, we may cover $(c_\varepsilon \|\mathbf{s}\|_2)B_2^{n+1} \cap \{\mathbf{x} \in \mathbb{R}^{n+1} \ : \ x_{n+1} = k \cdot \frac{\mu}{n}\}$ for all $k \in \mathbb{Z}$ (such that the intersection is not empty) by translates of K. There are only $2c_\varepsilon(n + 1) + 1$ such layers to consider and so $(2c_\varepsilon(n + 1) + 1)2^{0.401n}$ translates of K suffice. The last component of a lattice vector of $\mathcal{L}'$ is of the form $k \cdot \frac{\mu}{n}$ and it follows that these translates of K cover all lattice vectors of euclidean norm smaller than $c_\varepsilon \|\mathbf{s}\|_2$, see Figure 3.

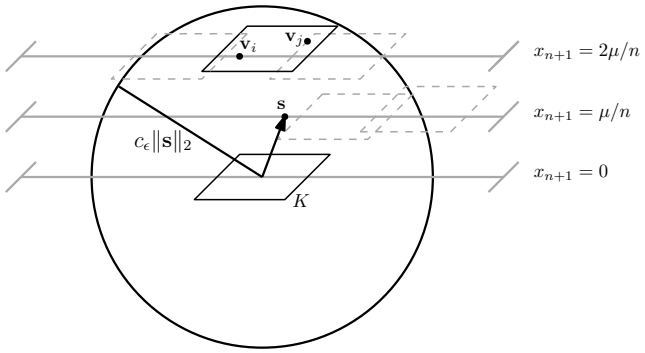

Figure 3 Covering the lattice points with translates of K.

.

Set $N = \lceil (2c_\varepsilon(n+1)+2)2^{(\varepsilon+0.401)n} \rceil$ and sample again $\{\mathbf{y}_1, \ldots, \mathbf{y}_N\} \overset{iid}{\sim} B_2^n(0, \xi_\varepsilon \mu)$ uniformly and independently at random. By Theorem 4 ii) and by the Chebychev inequality, see [38], the following event has a probability at least $1/2$.

(Event A'): There is a subset $S \subseteq \{1, \ldots, N\}$ with $|S| = (2c_\varepsilon(n + 1) + 1)2^{0.401n} + 1$ such that for each $i \in S$

$$\mathbf{y}_i \in I_\mathbf{s} \text{ and } \|\,\mathtt{ListRed}_L(\mathbf{y}_i)\|_2 \leq c_\varepsilon \|\mathbf{s}\|_2. \tag{8}$$

In this case, there exists a translate of K that holds at least two vectors $\mathtt{ListRed}_L(\mathbf{y}_i)$ and $\mathtt{ListRed}_L(\mathbf{y}_j)$ for different samples $\mathbf{y}_i$ and $\mathbf{y}_j$, see Figure 3 with $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{L}'$ instead. Thus, with probability at least $1/2$, there are $i, j \in [N]$ with $\mathbf{y}_i, \mathbf{y}_j \in I_\mathbf{s}$ such that

$$\mathtt{ListRed}_L(\mathbf{y}_i) - \mathtt{ListRed}_L(\mathbf{y}_j) \in 2K$$

Theorem 4 i) implies that, with probability at least $1/4$, there exist different samples $\mathbf{y}_i$ and $\mathbf{y}_j$ such that

$$\mathtt{ListRed}_L(\mathbf{y}_i) - \mathtt{ListRed}_L(\mathbf{y}_j) \in 2K + \mathbf{s}$$

In this case, the first n coordinates of $\mathtt{ListRed}_L(\mathbf{y}_i) - \mathtt{ListRed}_L(\mathbf{y}_j)$ can be written of the form $\mathbf{t} - \mathbf{v}$ for $\mathbf{v} \in \mathcal{L}$ and the first n coordinates on the right hand side are of the of the form $(\mathbf{t} - \mathbf{w}) + \mathbf{z}$, where $\mathbf{z} \in \mathcal{L}'$ and $\|\mathbf{z}\|_\infty \leq 2c_\varepsilon(1 + 1/n)a \|\mathbf{s}\|_\infty = 2c_\varepsilon(1 + 1/n)a \|\mathbf{t} - \mathbf{w}\|_\infty$. In particular, the lattice vector $\mathbf{v} \in \mathcal{L}$ is a $2ac_\varepsilon(1 + 1/n) + 1$ approximation to the closest vector to $\mathbf{t}$. Since we need to try out all pairs of the N elements, this takes time $N^2 = 2^{(0.802+\varepsilon')n}$ and space N. ◀

▶ Remark 6. For clarity we have not optimized the approximation factor. There are various ways to do so. We remark that for SVP_∞ we actually get a smaller approximation factor than the one that we describe. Let $\tilde{a}$ be such that $N(\sqrt{n}B_2^n, \tilde{a}B_\infty^n) \leq 2^{0.802n}$, the algorithm described above yields a $2c_\varepsilon\tilde{a}$ approximation instead of a $2c_\varepsilon a$ approximation to the shortest vector. This follows by applying the *birthday paradox* in the way that it was used by Pujol and Stehlé [38]. The same argument also applies to CVP_∞. Finally, we remark that in the case of SVP we have not really used property i) of Theorem 4. We only use this property to ensure that the generated vectors are different. It is plausible that this can be done more efficiently or with a better approximation factor.

Proof continued, $p \geq 2$. For SVP_p, $p \geq 2$, we define $\mathbf{s}$ to be shortest vector w.r.t. ℓ_p instead. Since $\|\mathbf{s}\|_2 \leq n^{1/2-1/p} \|\mathbf{s}\|_p$, we simply use Lemma 3 instead of Lemma 2 to conclude that there is some $a > 0$ such that if we have a set of $2^{0.401n}$ different lattice vectors of (euclidean) length smaller than $c_\varepsilon \|\mathbf{s}\|_2$, then two of them must have pairwise distance smaller than $2c_\varepsilon a$ w.r.t. ℓ_p.
For CVP_p, we define $\mathbf{w}$ to be the closest lattice vector to $\mathbf{t}$ w.r.t. ℓ_p. Both $\mathbf{s}$ and $\mathcal{L}'$ are defined analogously. We will need to replace the convex body K in (7) by

$$K = (c_\varepsilon \cdot a(1 + 1/n)\|\mathbf{s}\|_p)B_p^{n+1} \cap \{\mathbf{x} \in \mathbb{R}^{n+1} \ : \ x_{n+1} = 0\}.$$

The respective algorithms for SVP_p and CVP_p and the proof of correctness now follow from the case $p = \infty$. In particular, we can use the same parameters c_ε and a.

For the important case $p = 2$ we note that we can chose $a = 1$. This yields a approximation to the closest vector with the approximation guarantee c_ε matching that of the fastest approximate shortest vector problem w.r.t. ℓ_2, see [31]. ◀

4 Approximate CVP_p for $p \in [1, 2)$

In the previous section, we have extended the approximate SVP_2 solver to yield constant factor approximations to SVP_p and CVP_p for $p \in [2, \infty]$ in time $2^{(0.802+\varepsilon)n}$. From simple volumetric considerations, the technique from the previous section cannot be adapted to solve SVP_p and CVP_p for $p \in [1, 2)$ (in single exponential time). Instead, we can use a simple covering technique similar to the one considered by Eisenbrand et al. in [20]. We first show that for any constant $\varepsilon > 0$, there is a constant $a_\varepsilon > 0$, so that the crosspolytope B_1^n can be covered by $2^{\varepsilon n}$ balls (w.r.t. ℓ_2) with radius $(a_\varepsilon/\sqrt{n})$ and whose union is contained inside the crosspolytope scaled by a_ε. A similar covering also exists for B_p^n. Using the centers of these balls as targets, we can use the approximate CVP_2 algorithm to solve approximate CVP_1 resp. CVP_p. This is also similar to the technique of Dadush et al. in [17] resp. [16] where they cover general norm balls with M-ellipsoids to solve SVP and CVP w.r.t. to this norm by using the CVP_2 algorithm due to [33]. Unfortunately, there is only an upper bound of 2^{Cn} for some (large) constant $C > 0$ on the number of required M-ellipsoids, for our purpose we need a finer estimate. To achieve this, we rely on the set-covering idea and volume computations as outlined in Section 2. The following analogue to Lemma 2 is shown in the appendix.

▶ **Lemma 7.** *For each $\varepsilon > 0$, there exists $a_\varepsilon \in \mathbb{R}_{>0}$ independent of n such that*

$$\frac{\mathrm{vol}(B_1^n + (a_\varepsilon/\sqrt{n})B_2^n)}{\mathrm{vol}((a_\varepsilon/\sqrt{n})B_2^n)} \leq 2^{\varepsilon n}.$$

We now sketch the covering procedure for CVP_1 and SVP_1. Up to scaling the lattice and a guess on the distance of the closest (resp. shortest) lattice vector $\mathbf{v}$ to the target $\mathbf{t}$, we may assume that $1 - 1/n \leq \|\mathbf{v} - \mathbf{t}\|_1 \leq 1$ (resp. $1 - 1/n \leq \|\mathbf{v}\|_1 \leq 1$). We uniformly sample a point $\mathbf{x}$, [19], within $\mathbf{t} + B_1^n + (a_\varepsilon/\sqrt{n})B_2^n$ (set $\mathbf{t} = 0$ for SVP_1) and place a ball of radius $a_\varepsilon/\sqrt{n}$ around $\mathbf{x}$ (or $\mathbf{x}'$, the closest point to $\mathbf{x}$ in B_1^n, see Fig. 4).

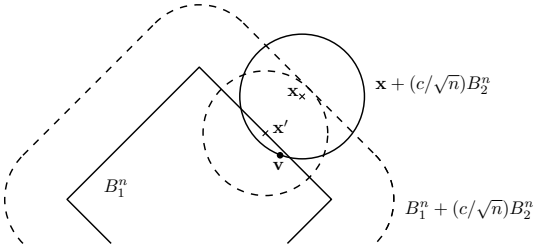

Figure 4 Generating a covering of B_1^n by $(c/\sqrt{n})B_2^n$.

.

By Lemma 7, with probability at least $2^{-\varepsilon n}$, $\mathbf{v}$ is covered by $\mathbf{x} + (a_\varepsilon/\sqrt{n})B_2^n$. Running the c-approximate (randomized) CVP_2 algorithm with target $\mathbf{x}$ (provided $\|\mathbf{v} - \mathbf{x}\|_2 \leq (a_\varepsilon/\sqrt{n})$), a lattice vector $\mathbf{w} \in \mathbf{x} + (c \cdot a_\varepsilon/\sqrt{n})B_2^n \subseteq \mathbf{t} + c \cdot (a_\varepsilon + 1)B_1^n$ is returned. The lattice vector $\mathbf{w}$ is thus a $c \cdot (a_\varepsilon + 1)$ approximation to the closest (resp. shortest) vector. In general, we run the c-approximate CVP_2 algorithm $O(\mathrm{poly}(n)2^{\varepsilon n})$ times with targets uniformly chosen within $\mathbf{t} + B_1^n + (a_\varepsilon/\sqrt{n})B_2^n$ and only output the closest of the resulting lattice vectors if it is within $c \cdot (a_\varepsilon + 1)B_1^n$. This ensures that, if there is lattice vector $\mathbf{v}$ in $\mathbf{t} + B_1^n$, a constant factor approximation to $\|\mathbf{t} - \mathbf{v}\|_1$ is found with high probability.
The same covering technique can be applied to B_p^n, $p \in (1, 2)$. By Hölder's inequality,

$$B_p^n \subseteq n^{1-1/p}B_1^n \text{ and } n^{1/2-1/p}B_2^n \subseteq B_p^n.$$

The first of these inclusions implies that for any $\varepsilon > 0$, we can pick the same constant a_ε as in Lemma 7 and cover B_p^n by at most $2^{\varepsilon n}$ translates of $a_\varepsilon n^{1/2-1/p}B_2^n$.

$$\frac{\mathrm{vol}(B_p^n + cn^{1/2-1/p}B_2^n)}{\mathrm{vol}(cn^{1/2-1/p}B_2^n)} \leq \frac{\mathrm{vol}(n^{1-1/p}B_1^n + cn^{1/2-1/p}B_2^n)}{\mathrm{vol}(cn^{1/2-1/p}B_2^n)} = \frac{\mathrm{vol}(B_1^n + (c/\sqrt{n})B_2^n)}{\mathrm{vol}((c/\sqrt{n})B_2^n)}$$

The second inclusion implies that these translates do not overlap B_p^n by more then a constant factor. It is then straightforward to adapt the boosting procedure described for CVP_1 to CVP_p. Using the approximate CVP_2 algorithm from the previous section then implies the following algorithm.

▶ **Theorem 8.** *There is a randomized algorithm that computes with constant probability a constant (depending on ε) factor approximation to CVP_p, $p \in [1, 2)$. The algorithm runs in time $2^{(0.802+\varepsilon)n}$ and requires space $2^{(0.401+\varepsilon)n}$.*

5 Proof of Lemma 7

Recall that the volume of $K + tB_2^n$ is a polynomial in t, with coefficients $V_j(K)$ that only depend on the convex body K:

$$\mathrm{vol}(K + tB_2^n) = \sum_{j=0}^{n} V_j(K)\,\mathrm{vol}(B_2^{n-j})t^{n-j}$$

The coefficients $V_j(K)$ are known as the intrinsic volumes of K. The intrinsic volumes of the crosspolytope B_1^n were computed by Betke and Henk in [13], and are given by the following formulae:

$$V_n(B_1^n) = \frac{2^n}{n!}$$

and for $0 \le j \le n-1$

$$V_j(B_1^n) = 2^n \binom{n}{j+1} \frac{\sqrt{j+1}}{j!\sqrt{\pi}^{n-j}} \cdot \int_0^\infty e^{-x^2} \left(\int_0^{x/\sqrt{j+1}} e^{-y^2}\,dy \right)^{n-j-1} dx$$

Given that the upper bound of Lemma 7 is exponential in n, we do not care about polynomial factors in n. For the sake of brevity, we will hide these polynomial factors by "$\lesssim$", i.e. $\mathrm{poly}(n) \lesssim 1$. This already simplifies the intrinsic volumes and, for $1 \le j \le n$:

$$V_j(B_1^n) \lesssim \frac{2^j}{j!}\binom{n}{j}$$

The volume of the $k-$dimensional ball B_2^k is given by

$$\mathrm{vol}(B_2^k) = \frac{\pi^{k/2}}{\Gamma(k/2+1)}$$

$\Gamma(\cdot)$ is the Gamma function. For $n \in \mathbb{N}$, we have $\Gamma(n+1) = n!$. By Stirling's formula we have the following estimate on $\Gamma(\cdot)$.

$$\left(\frac{z}{e}\right)^z \lesssim \Gamma(z+1) \lesssim \left(\frac{z}{e}\right)^z$$

With these estimates at hand, we can now prove Lemma 7.

$$\begin{aligned}
\frac{\mathrm{vol}(B_1^n + (c/\sqrt{n})B_2^n)}{\mathrm{vol}((c/\sqrt{n})B_2^n)} &= \frac{\sum_{j=0}^n V_j(B_1^n)\,\mathrm{vol}(B_2^{n-j})(c/\sqrt{n}))^{n-j}}{(c/\sqrt{n})^n\,\mathrm{vol}(B_2^n)} \\
&\lesssim \sum_{j=0}^{n} \frac{2^j\,n!}{j!(n-j)!j!} \frac{n^{j/2}}{c^j} \frac{\mathrm{vol}(B_2^{n-j})}{\mathrm{vol}(B_2^n)} \\
&\lesssim \sum_{j=0}^{n} \frac{(2e)^j\,n^n}{j^j\,(n-j)^{n-j}j^j} \frac{n^{j/2}}{c^j} \frac{n^{n/2}}{(n-j)^{(n-j)/2}(2\pi e)^{j/2}} \\
&\lesssim \sum_{j=0}^{n} \frac{n^{3n/2}n^{j/2}}{j^{2j}(n-j)^{3(n-j)/2}} \left(\frac{2e}{\pi c^2}\right)^{j/2} \\
&\overset{(j=\phi\,n)}{\lesssim} \max_{\phi\in[0,1]} \frac{e^{(3/2)\ln(n)n+\ln(n)n\phi/2}}{e^{2\ln(\phi n)\phi n+(3/2)\ln((1-\phi)n)(1-\phi)n}} \left(\frac{2e}{\pi c^2}\right)^{\phi n/2} \\
&\lesssim \max_{\phi\in[0,1]} e^{-2\ln(\phi)\phi n-2\ln(1-\phi)(1-\phi)n} \left(\frac{2e}{\pi c^2}\right)^{\phi n/2} \\
&= \max_{\phi\in[0,1]} 2^{2\,\mathrm{H}(\phi)n} \left(\frac{2e}{\pi c^2}\right)^{\phi n/2}
\end{aligned}$$

In passing to the second last line, we have added the factor $e^{-(1/2)\ln(1-\phi)(1-\phi)n}$ which is always greater than 1 for $\phi \in [0,1]$. $\mathrm{H}(\cdot)$ is the binary entropy function, i.e. $H(\phi) = -\ln(\phi)\phi - \ln(1-\phi)(1-\phi)$. $\mathrm{H}(\phi) \leq 1$ for $\phi \in [0,1]$ and $\mathrm{H}(\phi) = \mathrm{H}(1-\phi) \to 0$ monotonically as $\phi \to 0$. Thus, for some fixed c, the above expression reaches a maximum for some $\phi \in (0,1)$. If we increase c, we see that the ϕ^* realizing the maximum will decrease which then implies the lemma. This can be shown formally by fixing some c and taking a derivative w.r.t. ϕ. This will then show that the maximum is reached when $\phi^* = \Theta(\frac{1}{\sqrt{c}})$.
Thus, for any $\varepsilon > 0$, we can chose c large enough so that Lemma 7 holds.

References

1. D. Aggarwal, D. Dadush, and N. Stephens-Davidowitz. Solving the closest vector problem in 2^n time – the discrete gaussian strikes again! In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 563–582, October 2015. doi:10.1109/FOCS.2015.41.
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). *arXiv preprint*, 2019. arXiv:1911.02440.
3. Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the shortest vector problem in 2n time using discrete gaussian sampling. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 733–742, 2015.
4. Divesh Aggarwal and Priyanka Mukhopadhyay. Faster algorithms for SVP and CVP in the infinity norm. *CoRR*, abs/1801.02358, 2018. arXiv:1801.02358.
5. Divesh Aggarwal and Noah Stephens-Davidowitz. (gap/s)eth hardness of svp. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 228–238, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188840.
6. Divesh Aggarwal and Noah Stephens-Davidowitz. Just take the average! An embarrassingly simple 2ˆn-time algorithm for SVP (and CVP). In *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*, pages 12:1–12:19, 2018. doi:10.4230/OASIcs.SOSA.2018.12.
7. Miklós Ajtai. The shortest vector problem in l2 is np-hard for randomized reductions (extended abstract). In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 10–19, New York, NY, USA, 1998. Association for Computing Machinery. doi:10.1145/276698.276705.
8. Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 601–610, 2001. doi:10.1145/380752.380857.
9. Miklós Ajtai, Ravi Kumar, and D. Sivakumar. Sampling short lattice vectors and the closest lattice vector problem. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity, Montréal, Québec, Canada, May 21-24, 2002*, pages 53–57, 2002. doi:10.1109/CCC.2002.1004339.
10. Sanjeev Arora. *Probabilistic Checking of Proofs and Hardness of Approximation Problems*. PhD thesis, University of California at Berkeley, Berkeley, CA, USA, 1995. UMI Order No. GAX95-30468.
11. Shiri Artstein-Avidan and Boaz A Slomka. On weighted covering numbers and the levi-hadwiger conjecture. *Israel Journal of Mathematics*, 209(1):125–155, 2015.
12. H. Bennett, A. Golovnev, and N. Stephens-Davidowitz. On the quantitative hardness of cvp. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 13–24, October 2017. doi:10.1109/FOCS.2017.11.
13. Ulrich Betke and Martin Henk. Intrinsic volumes and lattice points of crosspolytopes. *Monatshefte für Mathematik*, 115(1):27–33, 1993. doi:10.1007/BF01311208.

14 Johannes Blömer and Stefanie Naewe. Sampling methods for shortest vectors, closest vectors and successive minima. *Theor. Comput. Sci.*, 410(18):1648–1665, 2009. doi:10.1016/j.tcs.2008.12.045.

15 V. Chvatal. A greedy heuristic for the set-covering problem. *Math. Oper. Res.*, 4(3):233–235, August 1979. doi:10.1287/moor.4.3.233.

16 Daniel Dadush and Gábor Kun. Lattice sparsification and the approximate closest vector problem. *Theory of Computing*, 12(1):1–34, 2016. doi:10.4086/toc.2016.v012a002.

17 Daniel Dadush, Chris Peikert, and Santosh S. Vempala. Enumerative lattice algorithms in any norm via m-ellipsoid coverings. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 580–589. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.31.

18 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. doi:10.1007/s00493-003-0019-y.

19 Martin E. Dyer, Alan M. Frieze, and Ravi Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *J. ACM*, 38(1):1–17, 1991. doi:10.1145/102782.102783.

20 Friedrich Eisenbrand, Nicolai Hähnle, and Martin Niemeier. Covering cubes and the closest vector problem. In *Proceedings of the 27th ACM Symposium on Computational Geometry, Paris, France, June 13-15, 2011*, pages 417–423, 2011. doi:10.1145/1998196.1998264.

21 O. Goldreich, D. Micciancio, S. Safra, and Jean-Pierre Seifert. Approximating shortest lattice vectors is not harder than approximating closet lattice vectors. *Inf. Process. Lett.*, 71(2):55–61, July 1999. doi:10.1016/S0020-0190(99)00083-6.

22 Peter Gruber. *Convex and Discrete Geometry*. Encyclopedia of Mathematics and its Applications. Springer, 2007.

23 Ishay Haviv and Oded Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 469–477, 2007.

24 Martin Henk, Jürgen Richter-Gebert, and Günter M Ziegler. Basic properties of convex polytopes. In *Handbook of discrete and computational geometry*, pages 243–270. CRC Press, 1997.

25 Varun Jog and Venkat Anantharam. A geometric analysis of the awgn channel with a (σ, ρ)-power constraint. *IEEE Transactions on Information Theory*, April 2015. doi:10.1109/TIT.2016.2580545.

26 Grigorii Anatol'evich Kabatiansky and Vladimir Iosifovich Levenshtein. On bounds for packings on a sphere and in space. *Problemy Peredachi Informatsii*, 14(1):3–25, 1978.

27 Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, 1987. doi:10.1287/moor.12.3.415.

28 Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, September 2005. doi:10.1145/1089023.1089027.

29 A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982. doi:10.1007/BF01457454.

30 Hendrik W. Lenstra. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538.

31 Mingjie Liu, Xiaoyun Wang, Guangwu Xu, and Xuexin Zheng. Shortest lattice vectors in the presence of gaps. *IACR Cryptology ePrint Archive*, 2011:139, 2011.

32 Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM journal on Computing*, 30(6):2008–2035, 2001.

33 Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 351–358, 2010. doi:10.1145/1806689.1806739.

34 Daniele Micciancio and Panagiotis Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, page 1468–1480, USA, 2010. Society for Industrial and Applied Mathematics.

35 Priyanka Mukhopadhyay. Faster provable sieving algorithms for the shortest vector problem and the closest vector problem on lattices in ℓ_p norm. *CoRR*, abs/1907.04406, 2019. `arXiv:1907.04406`.

36 Márton Naszódi and Moritz Venzin. Covering convex bodies and the closest vector problem. *arXiv preprint*, 2019. `arXiv:1908.08384`.

37 Márton Naszódi. On some covering problems in geometry. *Proceedings of the American Mathematical Society*, 144, April 2014. `doi:10.1090/proc/12992`.

38 Xavier Pujol and Damien Stehlé. Solving the shortest lattice vector problem in time 2 2.465n. *IACR Cryptology ePrint Archive*, 2009:605, January 2009.

39 Oded Regev. Lattices in computer science, lecture 8: $2^{O(n)}$ algorithm for svp, 2004.

40 Rolf Schneider. *Convex Bodies: The Brunn–Minkowski Theory*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2 edition, 2013. `doi:10.1017/CBO9781139003858`.

41 Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical computer science*, 53(2-3):201–224, 1987.

42 P. van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. *Technical Report 81-04, Mathematische Instituut, University of Amsterdam*, 1981.

43 Vijay V Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.