Approx-SVP in Ideal Lattices with Pre-processing

Alice Pellet-Mary, Guillaume Hanrot, and Damien Stehlé

Univ. Lyon, EnsL, UCBL, CNRS, Inria, LIP, F-69342 Lyon Cedex 07,
{alice.pellet-mary,guillaume.hanrot,damien.stehle}@ens-lyon.fr

Abstract. We describe an algorithm to solve the approximate Shortest Vector Problem for lattices corresponding to ideals of the ring of integers of an arbitrary number field K. This algorithm has a pre-processing phase, whose run-time is exponential in $\log|\Delta|$ with Δ the discriminant of K. Importantly, this pre-processing phase depends only on K. The pre-processing phase outputs an "advice", whose bit-size is no more than the run-time of the query phase. Given this advice, the query phase of the algorithm takes as input any ideal I of the ring of integers, and outputs an element of I which is at most $\exp(\widetilde{O}((\log|\Delta|)^{\alpha+1}/n))$ times longer than a shortest non-zero element of I (with respect to the Euclidean norm of its canonical embedding). This query phase runs in time and space $\exp(\widetilde{O}((\log|\Delta|)^{\max(2/3,1-2\alpha)}))$ in the classical setting, and $\exp(\widetilde{O}((\log|\Delta|)^{1-2\alpha}))$ in the quantum setting. The parameter α can be chosen arbitrarily in $[0, 1/2]$. Both correctness and cost analyses rely on heuristic assumptions, whose validity is consistent with experiments.
The algorithm builds upon the algorithms from Cramer *et al.* [EUROCRYPT 2016] and Cramer *et al.* [EUROCRYPT 2017]. It relies on the framework from Buchmann [Séminaire de théorie des nombres 1990], which allows to merge them and to extend their applicability from prime-power cyclotomic fields to all number fields. The cost improvements are obtained by allowing precomputations that depend on the field only.

1 Introduction

The Learning With Errors problem (LWE) introduced by Regev in [Reg05] has proved invaluable towards designing cryptographic primitives. However, as its instance bit-sizes grow at least quadratically with the security parameter to be well-defined, LWE often results in primitives that are not very efficient. In order to improve the efficiency, Stehlé, Steinfeld, Tanaka and Xagawa [SSTX09] introduced the search Ideal-LWE problem which involves polynomials modulo X^n+1 for n a power of two, and Lyubashevsky, Peikert and Regev [LPR10] exhibited the relationship to power-of-two cyclotomic fields, gave a reduction from the latter search problem to a decision variant, and tackled more general rings. This is now referred to as Ring-LWE, and leads to more efficient cryptographic constructions. To support the conjecture that Ring-LWE is computationally intractable, the authors of [SSTX09, LPR10] gave polynomial-time quantum reductions from

the approximate Shortest Vector Problem (approx-SVP) restricted to ideal lattices to Ring-LWE. Approx-SVP consists in finding a non-zero vector of an input lattice, whose norm is within a prescribed factor from the lattice minimum. Ideal lattices are lattices corresponding to ideals of the ring of integers of a number field, for example a power-of-two cyclotomic field in the situation above. When considering a lattice problem for such an ideal, the ideal is implicitly viewed as a lattice via the canonical embedding. A third quantum reduction from approx-SVP for ideal lattices to Ring-LWE was proposed by Peikert, Regev and Stephens-Davidowitz [PRS17]. It has the advantage of working for all number fields.

As is always the case, the value of these reductions highly depends on the intractability of the starting problem, i.e., approx-SVP for ideal lattices: approx-SVP for ideal lattices could even turn out to be computationally easy to solve, hence making these reductions vacuous. We stress that even if this were the case, that would not necessarily mean that there exists an efficient algorithm for Ring-LWE. In this work, we investigate the intractability of ideal approx-SVP for arbitrary number fields.

For arbitrary lattices, the best known trade-off between the run-time and the approximation factor is given by Schnorr's hierarchy of reduction algorithms [Sch87], whose most popular variant is the BKZ algorithm [SE94].

For any real number $\alpha \in [0, 1]$ and any lattice L of dimension n given by an arbitrary basis, it allows one to compute a vector of $L \setminus \{0\}$ which is no more than $2^{\widetilde{O}(n^\alpha)}$ times longer than a shortest one, in time $2^{\widetilde{O}(n^{1-\alpha})}$ (assuming the bit-size of the input basis in polynomial in n). This trade-off is drawn in blue in Figure 1.1.[1] In the case of ideal lattices in a cyclotomic ring of prime-power conductor (i.e., the ring of integers of $\mathbb{Q}(\zeta_m)$ where m is a prime power and ζ_m is a complex primitive m-th root of unity), it has been shown that it is possible to obtain a better trade-off than the BKZ algorithm, in the quantum computation setting. For *principal* ideal lattices, i.e., ideals that can be generated by a single element, the algorithmic blueprint, described in [CGS14, Ber14], consists in first using class group computations to find a generator of the ideal, and then use the so-called log-unit lattice to shorten the latter generator (we note that using the log-unit lattice for this purpose was already suggested in [RBV04]). A quantum polynomial-time algorithm for the first step was provided by Biasse and Song [BS16], building upon the work of [EHKS14]. The second step was carefully analyzed by Cramer, Ducas, Peikert and Regev [CDPR16], resulting in a quantum polynomial-time algorithm for approx-SVP restricted to *principal* ideal lattices, with a $2^{\widetilde{O}(\sqrt{n})}$ approximation factor. (See [HWB17] for a generalization to cyclotomics with degree of the form $p^\alpha q^\beta$, with p and q prime.) This line of works was extended by Cramer, Ducas and Wesolowski [CDW17] to any (not necessarily principal) ideal lattice of a cyclotomic ring of prime-power conductor. Put together, these results give us the trade-off between approximation factor and run-time drawn in red dashes in Figure 1.1. This is better than the BKZ algorithm when the approximation factor is larger than $2^{\widetilde{O}(\sqrt{n})}$. How-

[1] This figure, like all similar ones in this work, is in $(\log_n \log_2)$-scale for both axes.

ever, for smaller approximation factors, Schnorr's hierarchy remains the record holder. One could also hope to improve the trade-off for classical computing, by replacing the quantum principal ideal solver of [BS16] by the classical one of Biasse, Espitau, Fouque, Gélin and Kirchner [BEF+17]. However, this classical principal ideal solver runs in sub-exponential time $2^{\widetilde{O}(\sqrt{n})}$, hence combining it with [CDPR16, CDW17] results in a classical approx-SVP algorithm for a $2^{\widetilde{O}(\sqrt{n})}$ approximation factor in time $2^{\widetilde{O}(\sqrt{n})}$. Up to the $\widetilde{O}(\cdot)$ terms, this is exactly the trade-off obtained using Schnorr's hierarchy. Recently, Ducas, Plançon and Wesolowski [DPW19] experimentally analysed the $\widetilde{O}(\cdot)$ term of the $2^{\widetilde{O}(\sqrt{n})}$ approximation factor of the [CDPR16, CDW17] algorithm. This allows them to determine for which dimension n this quantum algorithm outperforms BKZ.

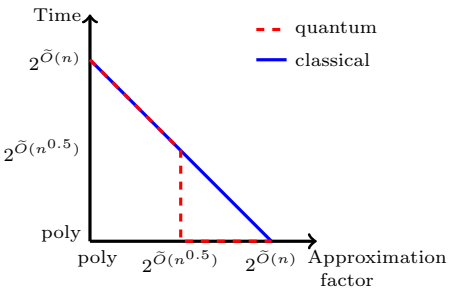

Fig. 1.1. Prior time/approximation trade-offs for ideal approx-SVP in cyclotomic fields of prime-power conductor.

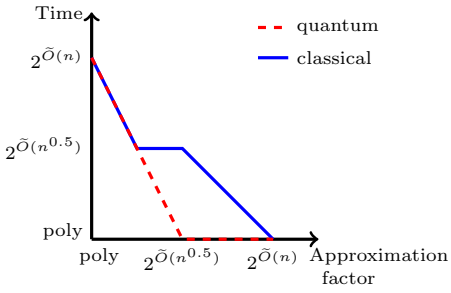

Fig. 1.2. New trade-offs for ideal approx-SVP in the same fields (with a pre-processing of cost $\exp(\widetilde{O}(n))$).

Contributions. We extend the techniques from [CDPR16, CDW17] to all number fields and improve the trade-off above by allowing the algorithm to perform some pre-computations on the number field.

It is a classical fact due to Minkowski [Min67, pp. 261–264] that there exists an absolute constant $c > 1$ such that for all number field K of degree $n \geq 2$ and discriminant Δ, we have $|\Delta| > c^n$. In the sequel, we shall thus state all our upper bounds in terms of $\log|\Delta| \geq \Omega(n)$. Actually, to fix the ideas, one may consider $\log|\Delta| = \widetilde{O}(n)$, which is the case for cyclotomic fields.

Let us consider a number field K of degree n and discriminant Δ. We assume a basis of the ring of integers R of K is given. Our algorithm performs some pre-processing on K, in exponential time $2^{\widetilde{O}(\log|\Delta|)}$. Once this pre-processing phase is completed and for any $\alpha \in [0, 1/2]$, the algorithm can, given any ideal lattice I of R, output a $2^{\widetilde{O}((\log|\Delta|)^{\alpha+1}/n)}$ approximation of a shortest non-zero vector of I in time $2^{\widetilde{O}((\log|\Delta|)^{1-2\alpha})} + T_{\text{c-g}}(K)$. Here $T_{\text{c-g}}(K)$ denotes the time needed to perform class group related computations in K: computing relations between elements of the class group and computing the units of R. Using the results

of [BS16,BEF+17,BF14], we can replace $T_{\text{c-g}}(K)$ by $\text{poly}(\log|\Delta|)$ for a quantum computer, and, for a classical computer, by $2^{\widetilde{O}((\log|\Delta|)^{1/2})}$ if K is a cyclotomic field of prime-power conductor and by $2^{\widetilde{O}((\log|\Delta|)^{2/3})}$ for an arbitrary field K. The three algorithms rely on the Generalized Riemann Hypothesis (GRH) and the two sub-exponential algorithms in the classical setting also require additional heuristic assumptions. The correctness and cost analyses of our algorithm rely on these heuristic assumptions, and others. Our contribution is formalized in the theorem below, which is the main result of this article.

Theorem 1.1 (Heuristic, see Theorems 3.4 and 5.1). *Let $\alpha \in [0,1/2]$ and K be a number field of degree n and discriminant Δ. Assume that a basis of the ring of integers R of K is known. Under some conjectures and heuristics, there exist two algorithms $A_{\text{pre-proc}}$ and A_{query} such that*

- *Algorithm $A_{\text{pre-proc}}$ takes as input the ring R, runs in time $2^{\widetilde{O}(\log|\Delta|)}$ and outputs a hint w of bit-size $2^{\widetilde{O}((\log|\Delta|)^{1-2\alpha})}$;*
- *Algorithm A_{query} takes as inputs any ideal I of R (whose algebraic norm has bit-size bounded by $2^{\text{poly}(\log|\Delta|)}$) and the hint w output by $A_{\text{pre-proc}}$, runs in time $2^{\widetilde{O}((\log|\Delta|)^{1-2\alpha})} + T_{\text{c-g}}(K)$, and outputs an element $x \in I$ such that $0 < \|x\|_2 \leq 2^{\widetilde{O}((\log|\Delta|)^{\alpha+1}/n)} \cdot \lambda_1(I)$.*

The hint output by the pre-processing phase has a bit-size that is bounded by the run-time of the query phase. By considering larger hints, the run-time of the query phase could be drastically improved. We give more details below, at the end of the high-level description of the algorithm.

Considering only the query cost, this result is of interest when $\log|\Delta| \leq \widetilde{O}(n^{4/3})$ for quantum computations and $\log|\Delta| \leq \widetilde{O}(n^{12/11})$ for classical computations. Indeed, in the other cases, the time/quality trade-offs obtained by our algorithm are worse than the ones obtained using Schnorr's hierarchy of algorithms. By letting α vary in $[0,1/2]$ and considering cyclotomic fields of prime-power conductor, we obtain the trade-offs represented in Figure 1.2. For a discussion for more general values of $\log|\Delta|$, we refer to Section 5. Going back to cyclotomic fields of prime-power conductor, these new trade-offs improve upon the prior ones, both for quantum and classical computers. Note that in Figure 1.2, we only plot the time needed for the query phase of the algorithm, but there is a pre-processing phase of exponential time performed before. Also, the new algorithm is no better than Schnorr's hierarchy in the classical setting when the run-time is sufficiently small. Hence, in Figure 1.2, we plotted the trade-offs obtained using Schnorr's hierarchy when they are better than the ones obtained with the new algorithm. The query phase of the new algorithm gives a quantum acceleration for approx-SVP for ideal lattices in cyclotomic fields of prime-power conductor, for all approximation factors $2^{\widetilde{O}(n^\alpha)}$ with $\alpha \in (0,1)$. This extends [CDW17], which obtained such a quantum acceleration for $\alpha \in [1/2,1)$. The query phase of the new algorithm also gives a classical acceleration for these fields, but only for $\alpha \in (0,1/2)$.

Technical overview. Our algorithm is inspired by the algorithms in [CDPR16, CDW17]. Given an ideal I as input, they first allow to find a principal ideal J contained in I (using [CDW17]), and then, they allow to compute a short generator of this ideal J (using [CDPR16]). This short generator is a somehow small element of I. This approach provides a $2^{\widetilde{O}(\sqrt{n})}$ approximation factor for approx-SVP in I. However, it can be shown that we cannot improve this approximation factor using these techniques, even if we increase the run-time of the algorithm. The reason is that, given an arbitrary principal ideal J, it may be that its shortest generator is $2^{\widetilde{O}(\sqrt{n})}$ times longer than its shortest non-zero vector.

We modify the strategy above, as follows. Given any ideal I, we try to find a 'good' principal ideal J contained in I, where we say that a principal ideal is 'good' if its shortest generator is not much larger than its shortest non-zero vector. The precise definition of 'not much larger' will depend on the approximation factor we want to achieve for our approx-SVP instance. Because the Euclidean norm of the shortest non-zero vector of J (broadly) increases with its algebraic norm, we also require that the algebraic norm of J is not much larger than the one of I (note that this was already needed in [CDPR16, CDW17]). To find this 'good' principal ideal J, the main idea of our algorithm is to express the problem as a Closest Vector Problem (CVP) instance in a lattice L depending only on the number field K.

This lattice L is similar to the one appearing in sub-exponential algorithms for computing the class group (see for instance [HM89, Buc88]). More precisely, we first select a set $\mathfrak{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_r\}$ of prime ideals of polynomially bounded algebraic norms, generating the class group. We then compute a generating set of the $\mathfrak{B}$-units, i.e., the set of elements $u \in K$ for which there exists $(e_1, \ldots, e_r) \in \mathbb{Z}^r$ such that $\langle u \rangle = \prod_i \mathfrak{p}_i^{e_i}$. The lattice L is obtained by considering the integer linear combinations of vectors of the form $(\operatorname{Log} u, e_1, \ldots, e_r)^T$, where $\langle u \rangle = \prod_i \mathfrak{p}_i^{e_i}$ and Log is the map applying the logarithm function to the canonical embedding, coefficient-wise. This lattice L only depends on the field K and can then be pre-computed and pre-processed.

Given any ideal I, the query phase of our algorithm computes a target vector t from I, and then solves a CVP instance in L with this target vector t. First, we decompose the ideal I in the class group as a product of the ideals of $\mathfrak{B}$. Concretely, we compute $g \in K$ and $(v_1, \ldots, v_r) \in \mathbb{Z}^r$ such that $I = \prod_i \mathfrak{p}_i^{v_i} \cdot \langle g \rangle$. This principal ideal $\langle g \rangle$ is a candidate for our principal ideal J contained in I (assume for the moment that the v_i's are non-positive, so that $\langle g \rangle$ is indeed contained in I). However, as is, we have no guarantee that $\langle g \rangle$ has a short generator. We also have no guarantee that its algebraic norm is not much larger than the one of I (i.e., that the v_i's are small). Hence, our objective is to multiply the principal ideal $\langle g \rangle$ by other principal ideals, until we have a good candidate for J. To do so, we define the vector $t = (-\operatorname{Log} g, v_1, \ldots, v_r)^T$. Observe that $\langle g \rangle$ would be a good candidate for J if this vector was short (and with $v_i \leq 0$ for all i). Indeed, this would mean that g is a short generator of $\langle g \rangle$ (because $\operatorname{Log} g$ is short), and that $\langle g \rangle = I \cdot \prod_i \mathfrak{p}_i^{-v_i}$ is a small multiple of I (because the $\mathfrak{p}_i$'s

have polynomially bounded norms, and the v_i's are small; the non-positivity of the v_i's is used to ensure that the ideal $\prod_i \mathfrak{p}_i^{-v_i}$ is integral). Also, we can see that adding a vector of L to t amounts to multiply the principal ideal $\langle g \rangle$ by another principal ideal (corresponding to the vector of L we are adding). Hence, we can find a good candidate J (together with a short generator of J) by solving a CVP instance in L with target t.

Finally, we need to solve CVP in the lattice L. We do not know any basis for L which would enable us to solve CVP in it efficiently (as opposed to the lattices considered in [CDPR16, CDW17]). However, the lattice L is fixed for a given number field, hence we can pre-process it. For this, we use a CVP with pre-processing (CVPP) algorithm due to Laarhoven [Laa16]. This leads to the time/approximation trade-offs given in Theorem 1.1. In [Laa16], significant effort is spent towards minimizing the constant factors in the exponents. These have recently been improved in [DLW19]. In this work, we neglect these factors for the sake of simplicity, but these would clearly matter in practice.

Laarhoven's CVPP algorithm is such that the bit-size of the output of the pre-processing phase is no larger than the run-time of the query phase[2] (hence, it is also the case for our algorithm). If we do not require this, we could have the following very simple and efficient algorithm for CVPP. First, it computes a short basis B_{sh} of the lattice. Then, it partitions the fundamental parallelepiped associated to B_{sh} into exponentially many small boxes, such that given any point of the parallelepiped, it is easy to determine to which box it belongs. Then, for each of these boxes, the pre-processing algorithm would compute a closest point of the lattice. The output of the pre-processing phase would then be the small basis B_{sh} and the set of all boxes together with their closest lattice point. Finally, given any vector in the real span of the lattice, the query algorithm would reduce it modulo B_{sh} to obtain a vector in the fundamental parallelepiped, and then determine the box of this reduced vector and its associated lattice vector. All this can be done efficiently (assuming we can efficiently access the database) and provides a small factor approximation for CVP, at the expense of a huge database.

Overall, the correctness and cost analyses of our algorithm rely on several heuristic assumptions. Many of them come from previous works [Laa16, BEF+17, BF14] and were already analysed. We introduce three new heuristic assumptions: Heuristics 4, 5 and 6 in Section 4. We discuss them by providing some mathematical justifications and some experimental results corroborating them. Concurrently to this work, Stephens-Davidowitz [Ste19] obtained a provable variant of the CVPP trade-offs from [Laa16, DLW19] that we use. Relying on it would allow us to make do with Heuristic 1, which was inherited from [Laa16, DLW19], at the expense of replacing Heuristic 4 by a similar one on the smoothing parameter of the lattice under scope (rather than its covering radius).

[2] Laarhoven also describes a variant of his algorithm in which he uses locality-sensitive hashing to reduce the run-time of the query phase below the bit-size of the advice, but we are not considering this variant here.

Impact. The query phase of the new algorithm can be interpreted as a non-uniform algorithm, as it solves approx-SVP for ideals of K, using a hint depending on K only. As the time needed to compute that hint (i.e., the run-time of $A_{\text{pre-proc}}$) is exponential, the concrete impact is limited. Nevertheless, our result should rather be interpreted as a strong indication that ideal approx-SVP is most likely a weaker problem than approx-SVP for arbitrary lattices: for unstructured lattices, there is no known non-uniform algorithm outperforming Schnorr's hierarchy.

Few cryptographic constructions have their security impacted by faster ideal approx-SVP solvers. A notable example is Gentry's fully homomorphic encryption scheme [Gen09], which was later superseded by faster homomorphic schemes relying on better understood hardness assumptions (see, e.g., [BV11b, BV11a]). Another important example is the Garg, Gentry and Halevi candidate construction of cryptographic multilinear map [GGH13] and its extensions. Because of the large pre-processing time, our algorithm does not provide a concrete attack on those schemes.

More importantly, our result strongly suggests that approx-SVP for ideals of the ring of integers R of a number field K may be weaker than Ring-LWE, for a vast family of number fields. Up to some limited parameter losses, Ring-LWE and approx-SVP for R-modules over K (with ranks ≥ 2) reduce to one another [LS15, AD17]. Therefore, a separation between approx-SVP for ideals and Ring-LWE is essentially the same as a separation between approx-SVP for ideals and approx-SVP for R-modules over K.

Open problems. Throughout the article, we keep track of all the different sub-algorithms that compose our approx-SVP solver. The exact formulation of the total cost of the algorithm, as a function of the costs of the sub-algorithms, is given in Theorem 3.4. When instantiated with the run-times of the currently known algorithms for the different sub-tasks, we obtain the values given in Theorem 1.1. However, the formula with all the sub-algorithms allows to see whether an improvement of the run-time of one of them leads to an improvement of the overall cost of the approx-SVP solver. In particular, for the specific choice of cyclotomic fields of prime-power conductor:

- Improving the approx-CVP solver would lead to an improvement of the slope of the curves in Figure 1.2, for approximation factors smaller than $2^{\widetilde{O}(\sqrt{n})}$. In a different direction, removing the pre-processing step needed for this approx-CVP solver would remove the pre-processing of the overall approx-SVP algorithm.
- Designing a classical algorithm that performs class group related computations in time less than $2^{\widetilde{O}(\sqrt{n})}$ would allow to further extend the (classical) segment of Figure 1.2 with slope $-1/2$, until it reaches the cost needed to solve these class group related problems. For example, Biasse described in [Bia17] an algorithm to solve the principal ideal problem in cyclotomic fields of prime-power conductor, with pre-processing. After pre-computations depending on the field only, this algorithm finds a generator of a principal ideal in time less than $2^{\widetilde{O}(\sqrt{n})}$ if the ideal has algebraic norm $\leq 2^{\widetilde{O}(n^{1.5})}$.

Finally, one could wonder whether it is possible to find significantly faster approx-SVP algorithms for specific families of number fields and/or restricted families of ideals. For instance, the Bauch *et al.* algorithm from [BBV+17] and the follow-up algorithm of Biasse and van Vredendaal [BV18] allow to efficiently solve class group related problems in real multiquadratic number fields in the classical setting. This means that in these number fields, the classical version of our algorithm is as efficient as the quantum one (there is no threshold for the query phase in the classical setting). However, the algorithm still requires an exponential pre-processing phase for the approx-CVP solver.

Roadmap. In Section 2, we recall some necessary background on lattices and number fields. Then, in Section 3, we explain how to transform an approx-SVP instance in any ideal into an approx-CVP instance in some lattice depending only on the number field. We detail in Section 4 some properties of the lattice in which we want to solve approx-CVP, and we give the trade-offs obtained when using Laarhoven's algorithm. Finally, in Section 5, we instantiate our main theorem with the best run-times currently known for solving approx-CVPP and class group related problems.

Supplementary material. The code that was used to perform the experiments is available on the webpage of the first author.

2 Preliminaries

We let $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ and $\mathbb{C}$ respectively denote the sets of integer, rational, real and complex numbers. For a positive real number x, we let $\log x$ denote its binary logarithm. For two functions $f(n)$ and $g(n)$, we write $f(n) = \widetilde{O}(g(n))$ if there exists some constant $c > 0$ such that $f(n) = O(g(n) \cdot |\log g(n)|^c)$. We abuse notations by defining $\widetilde{O}(n^\alpha) = O(n^\alpha \operatorname{poly}(\log n))$ even if $\alpha = 0$ (this will simplify some statements). For a vector $v \in \mathbb{R}^n$, we let v_i denote the i-th coordinate of v. We write $\|v\|_1 = \sum_i |v_i|$, $\|v\|_2 = \sqrt{\sum_i v_i^2}$ and $\|v\|_\infty = \max_i |v_i|$. We recall the following inequalities between these three different norms.

$$\|v\|_\infty \leq \|v\|_2 \leq \|v\|_1, \tag{2.1}$$

$$\|v\|_2 \leq \sqrt{n} \cdot \|v\|_\infty, \ \|v\|_1 \leq \sqrt{n} \cdot \|v\|_2. \tag{2.2}$$

Note that only the ℓ_2-norm is invariant under orthonormal transformations.

2.1 Lattice problems

For a lattice L and $i \in \{1, 2, \infty\}$, we let $\lambda_1^{(i)}(L)$ denote the norm of a shortest non-zero vector of L for the ℓ_i-norm. Similarly, for $k \geq 1$, we let $\lambda_k^{(i)}(L)$ denote the smallest real number such that there exist k linearly independent vectors of L whose ℓ_i-norms are no greater than $\lambda_k^{(i)}(L)$. We let $\operatorname{Span}(L)$ denote the real vector space spanned by the vectors of L. For a point $t \in \operatorname{Span}(L)$, we let $\operatorname{dist}^{(i)}(t, L) = \inf_{v \in L} \|t - v\|_i$ be the minimal distance between t and any point

of L. We define the covering radius of L as $\mu^{(i)}(L) = \sup_{t \in \mathrm{Span}(L)} \mathrm{dist}^{(i)}(t, L)$. The determinant (or volume) $\det(L)$ of a full-rank lattice L is the absolute value of the determinant of any of its bases.

Lemma 2.1 (Minkowski's inequality). *For any full-rank lattice L of dimension n, we have $\lambda_1^{(\infty)}(L) \leq \det(L)^{1/n}$. This implies that $\lambda_1^{(2)}(L) \leq \sqrt{n} \cdot \det(L)^{1/n}$.*

We will consider the following algorithmic problems involving lattices.

Definition 2.2 (Approximate Shortest Vector Problem (approx-SVP)). *Given a lattice L and $i \in \{1, 2, \infty\}$, the approximate Shortest Vector Problem in norm ℓ_i, with approximation factor $\gamma \geq 1$, is to find a vector $v \in L \setminus \{0\}$ such that $\|v\|_i \leq \gamma \cdot \lambda_1^{(i)}(L)$.*

Definition 2.3 (Approximate Closest Vector Problem (approx-CVP)). *Given a lattice L, $i \in \{1, 2, \infty\}$ and a target $t \in \mathrm{Span}(L)$, the approximate Closest Vector Problem in norm ℓ_i, with approximation factor $\gamma \geq 1$, is to find a vector $v \in L$ such that $\|v - t\|_i \leq \gamma \cdot \mathrm{dist}^{(i)}(t, L)$.*
In this article, we will be essentially interested in a variant of approx-CVP, in which we ask that $\|v - t\|_i \leq \beta$ for some β, independently of $\mathrm{dist}^{(i)}(t, L)$ (i.e., the distance of the found vector is bounded in absolute terms, independently of whether the target is close to the lattice or not). We call this variant approx-CVP'. For $i \in \{1, 2, \infty\}$, we let $T_{\mathrm{CVP}}(i, L, \beta)$ denote the worst-case run-time of the best known algorithm that solves approx-CVP' for the ℓ_i-norm, in the lattice L, with a bound β.

Definition 2.4 (Approx-CVP with Pre-processing (approx-CVPP)). *This problem is the same as approx-CVP, except that the algorithm can perform some pre-processing on the lattice L before it gets the target vector t. Approx-CVPP' is defined analogously. We will then consider the pre-processing time (performed when knowing only L) and the query time (performed once we get the target t). For $i \in \{1, 2, \infty\}$, we let $T_{\mathrm{CVP}}^{\text{pre-proc}}(i, L, \beta)$ (resp. $T_{\mathrm{CVP}}^{\text{query}}(i, L, \beta)$) denote the worst-case run-time of the pre-processing phase (resp. query phase) of the best algorithm that solves approx-CVPP' for the ℓ_i-norm, in the lattice L, with a bound β.*

In the following, we will always be interested in the approximate versions of these problems, so we will sometimes omit the 'approx' prefix.

In [Laa16], Laarhoven gives a heuristic algorithm for solving approx-CVPP. The following result is not explicitly stated in [Laa16] (only the two extreme values are given), but the computations can be readily adapted.

Theorem 2.5 ([Laa16, Corollaries 2 and 3]). *Let $\alpha \in [0, 1/2]$. Then, under Heuristic 1 below, there exists an algorithm that takes as pre-processing input an n-dimensional lattice L (given by a basis whose bit-size is polynomial in n) and as query input any vector $t \in \mathrm{Span}(L)$ (with bit-size that is polynomial in n) and outputs a vector $v \in L$ with $\|t - v\|_2 \leq O(n^\alpha) \cdot \mathrm{dist}^{(2)}(t, L)$, with pre-processing time $2^{O(n)}$ and query time $\mathrm{poly}(n) \cdot 2^{O(n^{1-2\alpha})}$ (the memory needed during the query phase is also bounded by $\mathrm{poly}(n) \cdot 2^{O(n^{1-2\alpha})}$).*

The heuristic assumption used in Laarhoven's algorithm states that the lattice L is somehow dense and behaves randomly.

Heuristic 1. There exists a constant $c > 0$ such that the ball of radius $c \cdot \lambda_1^{(2)}(L)$ (in ℓ_2-norm) contains at least 2^n points of L. Moreover, once renormalized, these points 'behave' as uniformly and independently distributed points on the unit sphere.

We can weaken this heuristic assumption by taking $c = \text{poly}(\log n)$, in which case the approximation factor in Laarhoven's algorithm becomes $\widetilde{O}(n^\alpha)$ (the pre-processing and query costs remain the same).

We will use this algorithm to heuristically solve approx-CVPP' in Euclidean norm for $\alpha \in [0, 1/2]$, achieving $T_{\text{CVP}}^{\text{pre-proc}}(2, L, O(n^\alpha) \cdot \mu^{(2)}(L)) = 2^{O(n)}$ and $T_{\text{CVP}}^{\text{query}}(2, L, O(n^\alpha) \cdot \mu^{(2)}(L)) = 2^{\widetilde{O}(n^{1-2\alpha})}$.

2.2 Number fields and ideals

We let K denote any number field of degree n and R be its ring of integers (i.e., elements of K which are roots of a monic polynomial with integer coefficients). The ring R is a free $\mathbb{Z}$-module of rank n. Let $\sigma_1, \ldots, \sigma_n$ be the n distinct embeddings from K to $\mathbb{C}$ ordered such that for $i \in \{1, \ldots, r_1\}$ we have $\sigma_i : K \to \mathbb{R}$ and for $i \in \{r_1 + 1, \ldots, r_2\}$ we have $\sigma_i = \overline{\sigma_{i+r_2}}$. We have r_1 real embeddings and r_2 pairs of complex conjugate embeddings, with r_1 and r_2 satisfying $r_1 + 2r_2 = n$. We let Δ denote the discriminant of K, i.e., $\Delta = [\det(\sigma_i(b_j))_{i,j}]^2$ for $b_1, \ldots, b_n$ any basis of the $\mathbb{Z}$-module R. Recall from Section 1 that Minkowski's bound gives us the following inequality:

$$\log |\Delta| \geq \Omega(n). \tag{2.3}$$

In the following, most of the costs will be expressed in term of $\log |\Delta|$.

We let $R^\times$ denote the group of units of R, that is $R^\times = \{u \in R \mid \exists v \in R,\ uv = 1\}$. Dirichlet's unit theorem states that $R^\times$ is isomorphic to the Cartesian product of a finite cyclic group (formed by the roots of unity contained in K) with the additive group $\mathbb{Z}^{r_1+r_2-1}$.

We associate to an element $x \in K$ the vector $(\sigma_1(x), \ldots, \sigma_{r_1}(x), \text{Re}(\sigma_{r_i+1}(x)), \text{Im}(\sigma_{r_1+1}(x)), \ldots, \text{Re}(\sigma_{r_1+r_2}(x)), \text{Im}(\sigma_{r_1+r_2}(x)))^T \in \mathbb{R}^n$, which we will call the canonical embedding of x. In the following, we will only consider the canonical embedding for the elements of K and R. We will abuse notation by considering that the elements of K and R are real vectors of the above form. Using this representation, the ring R becomes an n-dimensional lattice of $\mathbb{R}^n$. The volume of the lattice R is given by $\det(R) = 2^{-r_2}\sqrt{|\Delta|}$.

A fractional ideal I of K is a subset of K which is stable by addition, and by multiplication with any element of R, and such that $dI \subseteq R$ for some $d \in \mathbb{Z} \setminus \{0\}$. An ideal I is said to be integral if it is contained in R. A non-zero fractional ideal $I \subseteq R$ can be seen as a full-rank lattice in $\mathbb{R}^n$, via the canonical embedding. For an element $g \in K$, we write $\langle g \rangle = gR$, the smallest fractional ideal containing g.

Such an ideal is said to be principal. An integral ideal $I \subseteq R$ is said to be prime if the ring R/I is an integral domain. The product of two fractional ideals I and J is defined by $I \cdot J = \{x_1 y_1 + \cdots + x_r y_r \mid r \geq 0, x_1, \ldots, x_r \in I, y_1, \ldots, y_r \in J\}$.

The algebraic norm $\mathcal{N}(I)$ of a non-zero fractional ideal $I \subseteq R$ is the determinant of I when seen as a lattice in $\mathbb{R}^n$ (via the canonical embedding), divided by $\det(R) = 2^{-r_2}\sqrt{|\Delta|}$ (and $\mathcal{N}(\langle 0 \rangle)$ is defined as 0). If I is integral, this is also equal to $|R/I|$. The algebraic norm of a prime ideal is a power of a prime number. For two fractional ideals I and J, the algebraic norm of their product satisfies $\mathcal{N}(I \cdot J) = \mathcal{N}(I) \cdot \mathcal{N}(J)$. The algebraic norm of an element $r \in R$ is defined by $\mathcal{N}(r) = \prod_{i=1}^{n} \sigma_i(r)$. For any element $r \in R$, we have that $\mathcal{N}(\langle r \rangle) = |\mathcal{N}(r)|$, so in particular $\mathcal{N}(r) \in \mathbb{Z}$.

Let I be a non-zero fractional ideal seen as a lattice. By definition of the norm of I and Minkowski's inequality, we know that $\lambda_1^{(\infty)}(I) \leq \mathcal{N}(I)^{1/n} \cdot |\Delta|^{1/(2n)}$. We also have the following lower bound

$$\lambda_1^{(\infty)}(I) \geq \mathcal{N}(I)^{1/n}. \tag{2.4}$$

This lower bound comes from the fact that if $x \in I$ is such that $\|x\|_\infty = \lambda_1^{(\infty)}(I)$, then we have $|\mathcal{N}(x)| = \prod_i |\sigma_i(x)| \geq \mathcal{N}(I)$ (because $\langle x \rangle$ is a sub-lattice of I). This implies that at least one of the $|\sigma_i(x)|$'s is no smaller than $\mathcal{N}(I)^{1/n}$, hence the inequality. When $\log |\Delta| = \widetilde{O}(n)$, these two inequalities imply that $\lambda_1^{(\infty)}(I)$ is essentially $\mathcal{N}(I)^{1/n}$, up to a $2^{\mathrm{poly}(\log n)}$ factor. When $|\Delta|$ increases, so does the gap between the two bounds.

2.3 The class group

We let $\mathcal{I}_K$ denote the set of non-zero fractional ideals of K and $\mathcal{P}_K \subseteq \mathcal{I}_K$ denote the subset of non-zero principal fractional ideals. One can prove that for every non-zero fractional ideal I, there is a fractional ideal I^{-1} such that $I \cdot I^{-1} = R$. This gives $\mathcal{I}_K$ a group structure, for which $\mathcal{P}_K$ is a subgroup.

The class group of K is defined as the quotient $Cl_K = \mathcal{I}_K/\mathcal{P}_K$. For any non-zero ideal I of K, we let $[I]$ denote the equivalence class of I in the class group. In particular, we have $\mathcal{P}_K = [R]$. The class group is a finite abelian group and its cardinality h_K is called the class number. We have the following bound:

$$\log h_K = \widetilde{O}(\log |\Delta|). \tag{2.5}$$

This can be derived from the proof of Equation (2.3), this proof being based on the fact that any class of the class group contains an integral ideal whose norm is bounded as $2^{\widetilde{O}(\log |\Delta|)}$. We also justify it later using Equation (2.6) (which is significantly stronger).

We know, thanks to a result of Bach [Bac90] that the class group can be generated by ideals of polynomially bounded norms.

Theorem 2.6 (Theorem 4 of [Bac90]). *Under the GRH, the class group of a number field of discriminant Δ is generated by the prime ideals of algebraic norms $\leq 12 \log^2 |\Delta|$.*

Moreover, computing all prime ideals of norms $\leq 12\log^2|\Delta|$ can be done in time polynomial in $\log|\Delta|$. Indeed, these prime ideals can be obtained by factoring all ideals $\langle p\rangle$ where $p \in \mathbb{Z}$ is a prime no greater than $12\log^2\Delta$. Further, factoring such an ideal can be done in polynomial time (either using the Kummer-Dedekind theorem if p does not divide the index of $\mathbb{Z}[\theta]/R$, where θ is an algebraic integer such that $K = \mathbb{Q}(\theta)$, or using an algorithm due to Buchmann and Lenstra if p divides $|\mathbb{Z}[\theta]/R|$, see [Coh13, Section 4.8.2] for the former and [Coh13, Section 6.2] for the latter).

We will use the following lemma.

Lemma 2.7. *Let $\mathfrak{B}$ be any finite set of fractional ideals that generates the class group Cl_K. Then we can extract a subset $\mathfrak{B}'$ of $\mathfrak{B}$, of cardinality at most $\log h_K$, which also generates the class group. Moreover, this can be done efficiently if we are given the relations between the elements of $\mathfrak{B}$, in the form of a basis of $\ker(f_{\mathfrak{B}})$ where $f_{\mathfrak{B}} : (e_1, \dots e_r) \in \mathbb{Z}^r \mapsto \prod_i [\mathfrak{p}_i^{e_i}] \in Cl_K$, with $\mathfrak{B} = \{\mathfrak{p}_1, \dots, \mathfrak{p}_r\}$.*

We did not find this exact lemma in previous work, so we give a proof of it for the sake of completeness (even if the technique used to prove it is far from new).

Proof. We know that $\ker(f_{\mathfrak{B}})$ is a lattice of volume h_K contained in $\mathbb{Z}^r$ (it is stable by addition and subtraction, and $|\mathbb{Z}^r/\ker(f_{\mathfrak{B}})| = |Cl_K| = h_K$). Let $R_{\mathfrak{B}} \in \mathbb{Z}^{r\times r}$ be a basis of this lattice, with column vectors. From this basis, we can efficiently compute the Hermite Normal Form (HNF) of the lattice, which we will write $H_{\mathfrak{B}}$. This basis matrix is triangular, and each column corresponds to a relation between the elements of $\mathfrak{B}$ (each row corresponds to an ideal of $\mathfrak{B}$). So we can remove from the set $\mathfrak{B}$ any ideal whose row in $H_{\mathfrak{B}}$ has a 1 on the diagonal. Indeed, if row i has a 1 on the diagonal, this means that we have a relation of the form $[p_i \cdot \prod_{j>i} \mathfrak{p}_j^{e_j}] = [R]$. Hence the ideal class $[\mathfrak{p}_i]$ is in the group generated by $\{[\mathfrak{p}_j]\}_{j>i}$, and so it is not needed to generate the class group. But we know that $\det(H_{\mathfrak{B}}) = \det(\ker(f_{\mathfrak{B}})) = h_K$ is the product of the diagonal elements (which are integers). So we have at most $\log h_K$ ideals with diagonal entries different from 1. Hence, after removing from $\mathfrak{B}$ all ideals whose corresponding row in $H_{\mathfrak{B}}$ has a 1 on the diagonal, we obtain a set $\mathfrak{B}'$ of cardinality at most $\log h_K$ and which still generates the class group. This proof is an efficient algorithm if we are given an initial basis $R_{\mathfrak{B}}$, because we only need to compute an HNF basis, which can be done in time polynomial in the size of the input matrix. □

Theorem 2.6 states that the class group can be generated by integral ideals of polynomially bounded norms, but this does not give us the existence of many small-norm integral ideals. For instance, if the class group is trivial (i.e., all ideals are principal), then it is generated by $[R]$. More generally, the class group could be generated by a very small number of ideals. In the following, we will need the existence of $\widetilde{\Omega}(\log|\Delta|)$ distinct integral ideals of polynomially bounded norms.

Theorem 2.8 (Theorem 8.7.4 of [BS96]). *Assume the GRH. Let $\pi_K(x)$ be the number of prime integral ideals of K of norm $\leq x$. Then there exists an absolute constant C (independent of K and x) such that*

$$|\pi_K(x) - \mathrm{li}(x)| \leq C \cdot \sqrt{x}\,(n\log x + \log|\Delta|)\,,$$

where $\mathrm{li}(x) = \int_2^x \frac{\mathrm{d}t}{\ln t} \sim \frac{x}{\ln x}$ *(and* ln *refers to the natural logarithm).*

Instantiating this theorem with $x = (\log|\Delta|)^\kappa$ for some constant $\kappa > 4$, we obtain the following corollary. The bounds in this corollary can be improved, but they suffice for our needs.

Corollary 2.9. *Assume the GRH. Let* $\kappa > 4$. *For* $\log|\Delta|$ *sufficiently large, there are* $\geq (\log|\Delta|)^{\kappa-2}$ *distinct prime integral ideals of norm smaller than* $(\log|\Delta|)^\kappa$.

Proof. We apply Theorem 2.8 with $x = (\log|\Delta|)^\kappa$. As $\mathrm{li}(x) \sim \frac{x}{\ln x}$, we have that $\mathrm{li}(x) \geq (\log|\Delta|)^{\kappa-1}$ holds for $\log|\Delta|$ sufficiently large. Recall that $\log|\Delta| > cn$ for some (explicit) constant c. Hence, the right hand side of the inequality of Theorem 2.8 can be bounded as

$$C \cdot \sqrt{x}\,(n\log x + \log|\Delta|) \leq C(\kappa/c+1)\cdot(\log|\Delta|)^{\kappa/2+1}\cdot\log\log|\Delta|.$$

But, as we chose κ such that $\kappa - 1 > \kappa/2 + 1$, we have, for $\log|\Delta|$ sufficiently large:

$$(\log|\Delta|)^{\kappa-1} - C(\kappa/c+1)\cdot(\log|\Delta|)^{\kappa/2+1}\cdot\log\log|\Delta| \geq (\log|\Delta|)^{\kappa-2},$$

hence proving the corollary. □

We use Theorem 2.6, Corollary 2.9 and Lemma 2.7, to obtain the following.

Corollary 2.10. *Assume the GRH. Then, for* $\log|\Delta|$ *sufficiently large and for any integer* $r \geq \log h_K$, *there exists a set* $\mathfrak{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_r\}$ *of prime integral ideals generating the class group, with* $\mathcal{N}(\mathfrak{p}_i) = \mathrm{poly}(\log|\Delta|, r)$ *for all* i.

Proof. Combining Theorem 2.6 and Lemma 2.7, we know that there exists a set $\mathfrak{B}$ of cardinality at most r, generating the class group and containing only prime ideals of norms $\leq 12\log^2|\Delta|$. We can then add prime ideals to this set $\mathfrak{B}$, until its cardinality reaches r. Thanks to Corollary 2.9, we know that there are enough prime ideals of norm smaller than $\mathrm{poly}(\log|\Delta|, r)$ (for some fixed poly) to increase the cardinality of $\mathfrak{B}$ up to r. □

2.4 The log-unit lattice

We define $\mathrm{Log}\, x = (\log|\sigma_1(x)|, \ldots, \log|\sigma_n(x)|)^T \in \mathbb{R}^n$, for any $x \in K \setminus \{0\}$. Observe that this is not the usual definition of the logarithmic embedding. The function Log is often defined either as $(\log|\sigma_1(x)|, \ldots, \log|\sigma_{r_1+r_2}(x)|)^T \in \mathbb{R}^{r_1+r_2}$ [Sam13, Section 4.4] or as $(\log|\sigma_1(x)|, \ldots, \log|\sigma_{r_1}(x)|, 2\log|\sigma_{r_1+1}(x)|, 2\log|\sigma_{r_1+r_2}(x)|)^T \in \mathbb{R}^{r_1+r_2}$ [Coh13, Definition 4.9.6]. Indeed, for $i > r_1 + r_2$, the $\log|\sigma_i(x)|$'s are redundant because $|\sigma_i(x)| = |\sigma_{i-r_2}(x)|$. However, in our case, it will be more convenient to work with the logarithms of all the embeddings.

Let $E = \{x \in \mathbb{R}^n : x_i = x_{i+r_2}, \forall r_1 < i \leq r_2\}$. We have $\mathrm{Log}(K \setminus \{0\}) \subseteq E$. We let H be the hyperplane of $\mathbb{R}^n$ defined by $H = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 0\}$ and $\mathbf{1}$ be the vector, orthogonal to H, defined as $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^n$. We

write $\pi_H : \mathbb{R}^n \to H$ the orthogonal projection on H, parallel to $\mathbf{1}$. We define $\Lambda = \{\operatorname{Log} u, u \in R^\times\}$, which is a lattice of dimension $r_1 + r_2 - 1$ contained in $H \cap E$ (thanks to Dirichlet's unit theorem), called the *log-unit lattice*. We have the following upper bound:

$$\det(\Lambda) \cdot h_K \leq 2^{O(\log|\Delta| + n \log\log|\Delta|)} = 2^{\widetilde{O}(\log|\Delta|)}. \tag{2.6}$$

This upper bound comes from the relation between $\det(\Lambda)$, h_K and the residue of the zeta-function ζ_K at $s = 1$ (see [Lou00]). The latter is known considering Λ defined by the logarithmic embedding $(\log|\sigma_1(x)|, \dots, \log|\sigma_{r_1}(x)|, 2\log|\sigma_{r_1+1}(x)|, 2\log|\sigma_{r_1+r_2}(x)|)^T \in \mathbb{R}^{r_1+r_2}$. However, it can be seen that if one multiplies our lattice Λ by a matrix with blocks of the form $\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$ (in order to add and subtract $\log|\sigma_i(x)|$ and $\log|\sigma_{i+r_2}(x)|$ for $r_1 < i \leq r_2$), one obtains the log-unit lattice defined by the logarithmic embedding considered in [Lou00]. As multiplying by such a matrix increases the determinant by a factor 2^{r_2}, Inequality (2.6) remains valid in our setup.

This bound, combined with a lower bound on $\det(\Lambda)$ also gives Equation (2.5). Indeed, using a result of Zimmert [Zim80], we have that $\det(\Lambda) > 0.02 \cdot 2^{-r_2}$ (handling again our unusual definition of Λ).

For any $x \in K$, there exists a unique vector $h \in H \cap E$ and a unique real number a such that $\operatorname{Log} x = h + a\mathbf{1}$. In the following, we recall relationships between (h, a) and x. These results are standard (e.g., they are used freely in [CDPR16, Section 6]).

Lemma 2.11. *Let $r \in K$. Then we have $\operatorname{Log} r = h + \frac{\log|\mathcal{N}(r)|}{n}\mathbf{1}$, for some $h \in H \cap E$.*

For the sake of completeness, and because we are using an unusual definition of Log, we give a proof of this result below.

Proof. Write $\operatorname{Log} r = h + a\mathbf{1}$ for some $h \in H \cap E$ and $a > 0$. First, as $\mathbf{1}$ is orthogonal to H, we have that $\langle \mathbf{1}, \operatorname{Log} r\rangle = \langle \mathbf{1}, a\mathbf{1}\rangle = a \cdot n$. But using the definition of $\operatorname{Log} r$, we also have that

$$\langle \mathbf{1}, \operatorname{Log} r\rangle = \sum_i \log|\sigma_i(r)| = \log|\mathcal{N}(r)|,$$

where we used the fact that $\mathcal{N}(r) = \prod_i \sigma_i(r)$. This completes the proof. □

The following lemma gives a bound on the Euclidean norm of an element $r \in R$ in terms of its decomposition $\operatorname{Log} r = h + a\mathbf{1}$.

Lemma 2.12. *For any $r \in K$, if $\operatorname{Log} r = h + a\mathbf{1}$ with $h \in H \cap E$ and $a \in \mathbb{R}$, then we have $\|r\|_\infty \leq 2^a \cdot 2^{\|h\|_\infty}$. In particular, this implies that*

$$\|r\|_2 \leq \sqrt{n} \cdot 2^a \cdot 2^{\|h\|_2} = \sqrt{n} \cdot |\mathcal{N}(r)|^{1/n} \cdot 2^{\|h\|_2}.$$

Proof. The second inequality follows from the first one by using Equations (2.1) and (2.2) (and Lemma 2.11 for the equality). For the first inequality, recall that by definition of Log, we have that $(\operatorname{Log} r)_i = \log|\sigma_i(r)| = h_i + a$ for all i. So, by definition of $\|r\|_\infty = \max_i |\sigma_i(r)|$, we have $\|r\|_\infty = \max_i 2^{h_i + a} \leq 2^a \cdot 2^{\|h\|_\infty}$. □

2.5 Algorithmic problems related to class group computations

Let $\mathfrak{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_r\}$ be a set of prime integral ideals generating the class group, obtained for example using Corollary 2.10.

We will be interested in computing the lattice of all the relations between the ideals of $\mathfrak{B}$, i.e., the kernel of the map

$$f_{\mathfrak{B}} : e = (e_1, \ldots, e_r) \in \mathbb{Z}^r \mapsto [\prod_i \mathfrak{p}_i^{e_i}] \in Cl_K.$$

Recall that $\ker(f_{\mathfrak{B}})$ is a full-rank sub-lattice of $\mathbb{Z}^r$ of volume $|\mathbb{Z}^r / \ker(f_{\mathfrak{B}})| = |Cl_K| = h_K$. Let $N_{\mathfrak{B}} = \max_i \mathcal{N}(\mathfrak{p}_i)$. We let $T_{\text{rel}}(N_{\mathfrak{B}}, r)$ denote the time needed to compute a basis of $\ker(f_{\mathfrak{B}})$, together with generators of the corresponding principal ideals, given as input the set $\mathfrak{B}$. We write $T_{\text{decomp}}(N, N_{\mathfrak{B}}, r)$ for the time needed, given $\mathfrak{B}$ and a fractional ideal I of norm $\mathcal{N}(I) = N$, to find a vector $e \in \mathbb{Z}^r$ and an element $g \in K$ such that $I = \prod_i \mathfrak{p}_i^{e_i} \cdot \langle g \rangle$. Note that this decomposition always exists but might not be unique (we only require that $\mathfrak{B}$ generates the class group). Finally, we let $T_{\text{log-unit}}$ be the time needed to compute a basis of the log-unit lattice of K.

The three problems above are usually solved by computing S-units[3] for a well-chosen set S. This is why, in the following, the same cost bounds hold for the three of them.

In the *quantum* setting, Biasse and Song [BS16] showed that these three problems can be solved in polynomial time for any number field (under GRH). More precisely, they showed that

- $T_{\text{rel}}(N_{\mathfrak{B}}, r) = \text{poly}(\log |\Delta|, r, \log N_{\mathfrak{B}})$;
- $T_{\text{decomp}}(N, N_{\mathfrak{B}}, r) = \text{poly}(\log |\Delta|, \log N_{\text{num}}, \log N_{\text{denom}}, r, \log N_{\mathfrak{B}})$;
- $T_{\text{log-unit}} = \text{poly}(\log |\Delta|)$;

where N_{num} and N_{denom} refer to the numerator and denominator of N (i.e., $N = N_{\text{num}}/N_{\text{denom}} \in \mathbb{Q}$ with $N_{\text{num}}, N_{\text{denom}}$ in $\mathbb{Z}_{>0}$ and coprime).

In the *classical* setting, these three problems can be solved heuristically in sub-exponential time (under GRH). The first sub-exponential algorithm for all number fields (and which allows n to tend to infinity with $\log |\Delta|$) is due to Biasse and Fieker [BF14]:

- $T_{\text{rel}}(N_{\mathfrak{B}}, r) = \text{poly}(r, \log N_{\mathfrak{B}}) \cdot 2^{\widetilde{O}((\log |\Delta|)^{2/3})}$;
- $T_{\text{decomp}}(N, N_{\mathfrak{B}}, r) = \text{poly}(\log N_{\text{num}}, \log N_{\text{denom}}, r, \log N_{\mathfrak{B}}) \cdot 2^{\widetilde{O}((\log |\Delta|)^{2/3})}$;
- $T_{\text{log-unit}} = 2^{\widetilde{O}((\log |\Delta|)^{2/3})}$.

Biasse and Fieker actually claim $2^{O((\log |\Delta|)^{2/3+\varepsilon})}$ run-times. Tracing back the source of this ε leads to Biasse's [Bia14, Proposition 3.1]. A careful reading of the proof of the latter shows that the $(\log |\Delta|)^{\varepsilon}$ term is actually a power of $\log \log |\Delta|$,

[3] Given a set $S = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_r\}$ of prime integral ideals, the S-units are the elements $\alpha \in K$ such that there exist $e_1, \ldots, e_r \in \mathbb{Z}$ with $\prod_i \mathfrak{p}_i^{e_i} = \langle \alpha \rangle$.

hence, in our notations, it is absorbed by the $\widetilde{O}$ notation. In addition to the GRH, the algorithm of Biasse and Fieker requires two heuristic assumptions, referred to as Heuristic 1 and Heuristic 3 in [BF14]. We recall these two heuristic assumptions below (see [BF14] for more details).

Heuristic 2 ([BF14, Heuristic 1]). The probability $P(x, y)$ that an integral ideal of R produced by the Biasse-Fieker [BF14] algorithm, of norm bounded by x, can be factored as a product of prime ideals of norms bounded by y satisfies

$$P(x, y) \geq e^{-(1+o_{x\to\infty}(1))\cdot u \log u} \quad \text{for} \quad u = \frac{\log x}{\log y}.$$

Heuristic 3 ([BF14, Heuristic 3]). Given a set of r elements generating the class group, the algorithm only needs to find $r^{O(1)}$ relations between these elements to generate the full lattice of relations, with probability close to 1.

Smaller cost bounds are known for specific families of number fields. For prime-power cyclotomic fields, the $2^{\widetilde{O}((\log|\Delta|)^{2/3})}$ bounds can be replaced by $2^{\widetilde{O}((\log|\Delta|)^{1/2})}$ [BEF+17]. This algorithm is again heuristic and relies on the same assumptions as [BF14]. For real multiquadratic number fields, efficient classical algorithms allow to solve these three problems [BBV+17, BV18]. Finally, we note that the exponent 2/3 was recently lowered to 3/5 in [Gel17] and can even be decreased further in some cases.

3 From Ideal SVP to CVP in a Fixed Lattice

The main idea of our algorithm is, given an input ideal I, to find a principal ideal $\langle g\rangle \subseteq I$ with a short generator g. This is very similar to [CDW17], where the authors find a $2^{O(\sqrt{n})}$ approximation of a shortest non-zero vector of the ideal I by computing a principal ideal contained in I and then finding a short generator of this principal ideal. The limitation of this approach is that, if we consider any principal ideal contained in I, we cannot hope to find a better approximation than the $2^{O(\sqrt{n})}$ approximation obtained above in the worst case. This is due to the fact that in some principal ideals (including for prime-power cyclotomic fields), the shortest generator can be $2^{O(\sqrt{n})}$ times longer than a shortest non-zero element of the ideal (see [CDPR16]). Instead of looking for any principal ideal contained in I, we consider only those with a 'good' generator (i.e., a generator which is also a very short element of the corresponding principal ideal).

In order to find such an ideal, we merge the two steps of [CDPR16, CDW17] (consisting in first finding a principal multiple of I and then computing a small generator of the principal ideal), by introducing a lattice L that is very similar to the one used for class group computations. This lattice only depends on the number field (and not on the ideal I). We describe it in the next subsection. We then show how to express the problem of finding a principal multiple of I with a small generator as a CVP instance for this fixed lattice.

3.1 Definition of the lattice L

In this subsection, we define the lattice L which we will use in order to transform our ideal-SVP instance into a CVPP' instance. We also give an algorithm to compute a basis of L and analyze its run-time. The lattice L we are considering is not new. It was already used in previous sub-exponential algorithms computing the class group of a number field [HM89, Buc88, BF14, BEF+17, Gel17]. However, these algorithms usually choose a sub-exponential set of ideals, hence resulting in a lattice L of sub-exponential dimension. Our lattice L will have a dimension which is polynomial in $\log|\Delta|$.

In the following, we fix some integer r such that $\log h_K \leq r$ and $r \leq \mathrm{poly}(\log|\Delta|)$ (looking forward, the integer r will be related to the dimension of the lattice in which we will solve CVPP', so it would be undesirable to set it too large). Let us also fix a set of prime integral ideals $\mathfrak{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_r\}$ as given by Corollary 2.10. We consider the lattice L of dimension $\nu := r + r_1 + r_2 - 1$, generated by the columns of the following matrix:

$$B_L := \begin{array}{|c|c|} \hline c \cdot B_\Lambda & c\cdot\widetilde{h}_{g_1}, \ldots, c\cdot\widetilde{h}_{g_r} \\ \hline 0 & v_1 \quad v_2 \quad \cdots \quad v_r \\ \hline \end{array} \quad \begin{array}{l} \overleftrightarrow{r_1+r_2-1} \\ r_1+r_2-1 \\ r \\ \nu \end{array} \quad \begin{array}{l} \overleftrightarrow{\qquad r \qquad} \\ \\ \\ \\ \end{array}$$

where:

- the scaling parameter $c > 0$ is to be chosen later;
- the matrix $B_\Lambda = (f_{H\cap E}(b_1), \ldots, f_{H\cap E}(b_{r_1+r_2-1}))$ is a basis of $f_{H\cap E}(\Lambda)$, where Λ is the log-unit lattice and $f_{H\cap E} : H \cap E \subset \mathbb{R}^n \to \mathbb{R}^{r_1+r_2-1}$ is an isometry;[4]
- the matrix consisting of the vectors $v_i = (v_{1i}, \ldots, v_{ri})^T$ is a basis of $\ker(f_{\mathfrak{B}})$ (in particular, the ideals $\prod_j \mathfrak{p}_j^{v_{ji}}$ are principal for all i);
- the column vectors $\widetilde{h}_{g_i}$ are of the form $f_{H\cap E}(\pi_H(\mathrm{Log}\, g_i))$ for $g_i \in K$ a generator of the fractional principal ideal associated with the relation v_i, i.e., we have $\prod_j \mathfrak{p}_j^{v_{ji}} = \langle g_i \rangle$.

[4] As Λ is not full rank in $\mathbb{R}^n$, we change the ambient space such that $f_{H\cap E}(\Lambda)$ becomes full rank in $H \cap E = \mathbb{R}^{r_1+r_2-1}$. Note however that the ℓ_2-norm is preserved by this transformation (this is not the case for the ℓ_1 and ℓ_∞ norms).

We will explain how to construct L below. This lattice enjoys the following property, which will be used later.

Lemma 3.1. *Let w be a vector of L and parse it as $w = (h, v)^T$ with h of dimension $r_1 + r_2 - 1$ and $v = (v_1, \ldots, v_r)$ of dimension r. Then there exists an element $g \in K \setminus \{0\}$ such that $h = c \cdot f_{H\cap E}(\pi_H(\mathrm{Log}\, g))$ and $\prod_j \mathfrak{p}_j^{v_j} = \langle g \rangle$.*

Proof. We first observe that the result holds for the vectors of the basis B_L. For the r vectors on the right of B_L, this holds by construction. For the $r_1 + r_2 - 1$ vectors on the left, we have that $\prod_j \mathfrak{p}_j^0 = R = \langle u \rangle$ for any unit $u \in R$. So by definition of B_Λ, the property of Lemma 3.1 also holds for the $r_1 + r_2 - 1$ first vectors of B_L.

To complete the proof, it suffices to observe that the property of Lemma 3.1 is preserved by addition (if g_1 corresponds to a vector w_1 and g_2 corresponds to a vector w_2, then $g_1 g_2$ corresponds to the vector $v_1 + v_2$) and by multiplication by -1 (if g corresponds to a vector w, then g^{-1} corresponds to the vector $-w$). All these elements g are invertible as they are obtained by multiplying and inverting non-zero elements of K. □

3.2 Computation of the lattice L

The lattice L described above only depends on the number field we are working on. A basis of it can then be computed in a pre-processing phase, before the knowledge of the ideal in which we want to find a short non-zero vector. In this subsection, we give an algorithm to compute the lattice L and we show that this algorithm can be performed in time at most exponential in $\log |\Delta|$. As we shall see, this will even be sub-exponential in $\log |\Delta|$. The costly part of the pre-processing phase will be the pre-processing used for the CVPP algorithm.

Algorithm 3.1 Computes a basis B_L as described above

Input: A number field K and an integer $r = \mathrm{poly}(\log |\Delta|)$ such that $\log h_K \leq r$.
Output: The basis B_L described in Section 3.1.

1: Compute the set $\mathfrak{B}'$ of all prime ideals of algebraic norm $\leq 12 \log^2 |\Delta|$.
2: Compute all the relations between the elements of $\mathfrak{B}'$ and the log-unit lattice Λ.
3: Use the relations to extract a set $\mathfrak{B}'' \subseteq \mathfrak{B}'$ generating the class group with $|\mathfrak{B}''| \leq \log h_K$.
4: Compute the set $\mathfrak{P}$ of all prime ideals of norms smaller than some $\mathrm{poly}(\log |\Delta|)$ (choose the bound so that $|\mathfrak{P}| > r$).
5: Create a set $\mathfrak{B}$ by adding to $\mathfrak{B}''$ ideals taken uniformly in $\mathfrak{P}$, until the cardinality of $\mathfrak{B}$ reaches r.
6: Compute a basis of $\ker(f_{\mathfrak{B}})$ and generators g_i of the fractional principal ideals corresponding to the relations computed.
7: Create the matrix B_L from these r relations, the corresponding g_i and the log-unit lattice Λ computed at Step 2.
8: **return** B_L.

Lemma 3.2. *Assume GRH. Then Algorithm 3.1 outputs a matrix B_L as described above, in time at most*

$$T_{\text{log-unit}} + 2 \cdot T_{\text{rel}}(\text{poly}(\log|\Delta|), \text{poly}(\log|\Delta|)) + \text{poly}(\log|\Delta|).$$

Proof. We analyze the cost of each step of the algorithm, and provide some details for the correctness when needed.

Step 1. We have already seen in Section 2 that computing all prime ideals of norm $\leq 12\log^2|\Delta|$ can be performed in time polynomial in $\log|\Delta|$. There are $\text{poly}(\log|\Delta|)$ such ideals.

Step 2. Computing all the relations between the elements of $\mathfrak{B}'$ and the log-unit lattice Λ can be performed in time at most $T_{\text{rel}}(\text{poly}(\log|\Delta|), \text{poly}(\log|\Delta|)) + T_{\text{log-unit}}$. The relations between the elements of $\mathfrak{B}'$ are represented as an invertible matrix (whose columns span the kernel of the function $f_{\mathfrak{B}'}$ defined in Section 2.5).

Step 3. Extracting a generating set $\mathfrak{B}''$ from $\mathfrak{B}'$ of cardinality at most $\log h_K$ can be done using Lemma 2.7. Because we already have the matrix of relations between the elements of $\mathfrak{B}'$ (and because the size of this matrix is polynomial in $\log|\Delta|$), this can be done in polynomial time (as stated in the lemma).

Step 4. As in Step 1, this can be done in polynomial time, because the bound on the norms of the ideals is polynomial. We obtain a set $\mathfrak{P}$ whose cardinality is polynomial in $\log|\Delta|$.

Step 5. Picking uniform elements in a set of polynomial size can be done efficiently, so this step can be performed in polynomial time (recall that $r = \text{poly}(\log|\Delta|)$). Note that in the previous step, we had that the cardinality of $\mathfrak{B}'$ was at most $\log h_K \leq r$, so we can indeed add ideals to it to reach a set of cardinality r.

Step 6. As in Step 2, computing the kernel of $f_{\mathfrak{B}}$ can be done in time at most $T_{\text{rel}}(\text{poly}(\log|\Delta|), \text{poly}(\log|\Delta|)$. Together with the relations, we also get generators of the corresponding principal ideals.

Step 7. Finally, to compute the matrix B_L, we just need to compute the functions π_H and $f_{H\cap E}$ on the g_i's computed in Step 6. We then put it together with the matrix of relations computed in Step 6 and the log-unit lattice computed in Step 2. This can be done in polynomial time. □

3.3 From SVP in ideal lattices to CVP in L

We now explain how to transform the problem of finding a short non-zero vector in a fractional ideal I of R, into solving a CVP instance with respect to the lattice L described in Section 3.1. As explained above, the main idea is to multiply

the ideal I by ideals of the set $\mathfrak{B}$, until we obtain a 'good' principal ideal (i.e., with a short generator). In terms of lattice operations in L, our initial lattice I will give us a target vector in the real vector space spanned by L. Multiplying it by ideals of $\mathfrak{B}$ will be the same as adding to the target a vector of the lattice L. Finally, checking whether the resulting ideal is a good principal ideal can be done by checking whether the obtained vector is short. Overall, we are indeed solving a CVP instance in L. We first describe the algorithm, and then prove its correctness and bound its cost.

Algorithm 3.2 Solves ideal SVP using an oracle to solve CVP in L

Input: A non-zero fractional ideal $I \subseteq R$ (given by some basis), the basis B_L defined above and some parameter $\beta = \beta(n) > 0$.

Output: A somehow short non-zero element in I.

1: Compute $v_1, \ldots, v_r \in \mathbb{Z}$ and $g \in K$ such that $I = \prod_j \mathfrak{p}_j^{v_j} \cdot \langle g \rangle$.
2: Let $t = (-c \cdot f_{H \cap E}(h_g), v_1 + \beta, \ldots, v_r + \beta)^T$, where $h_g = \pi_H(\operatorname{Log} g)$.
3: Compute $w \in L$ such that $\|t - w\|_\infty \leq \beta$ (see Section 4).
4: Let $g' \in K$ be the element associated to w as in Lemma 3.1.
5: **return** $g \cdot g'$.

Theorem 3.3. *Let us fix $c = n^{1.5}/r$. Let $\beta = \beta(n) > 0$. Then, for any non-zero fractional ideal I of R, Algorithm 3.2 runs in time at most*

$$T_{\text{decomp}}(\mathcal{N}(I), \text{poly}(\log|\Delta|), \text{poly}(\log|\Delta|)) + T_{\text{CVP}}(\infty, L, \beta) + \text{poly}(\log|\Delta|)$$

and outputs a non-zero element $x \in I$ such that $\|x\|_2 \leq 2^{O(\frac{\beta \cdot r \cdot \log\log|\Delta|}{n})} \cdot \mathcal{N}(I)^{1/n}$.

Observe that in the statement of the run-time, the term $T_{\text{CVP}}(\infty, L, \beta)$ will be infinite if β is smaller than $\mu^{(\infty)}(L)$ (no algorithm can find a point of L at distance at most β given any target input). In this case, the run-time of our algorithm might also be infinite (i.e. the algorithm fails).

Proof. **Correctness.** Let us define the fractional ideal $J = \langle g \cdot g' \rangle$. This will be our 'good' principal ideal, i.e., a principal ideal with a small generator, and contained in I. Let us first prove that J is a multiple of I. By Lemma 3.1, we have $w = (c \cdot f_{H \cap E}(\pi_H(\operatorname{Log} g')), v'_1, \ldots, v'_r)^T$ with $\langle g' \rangle = \prod_j \mathfrak{p}_j^{v'_j}$. We can then write

$$\begin{aligned} J &= I \cdot \prod_j \mathfrak{p}_j^{-v_j} \cdot \langle g' \rangle && \text{by definition of } g \text{ and the } v_j\text{'s} \\ &= I \cdot \prod_j \mathfrak{p}_j^{-v_j} \cdot \prod_j \mathfrak{p}_j^{v'_j} && \text{by Lemma 3.1} \\ &= I \cdot \prod_j \mathfrak{p}_j^{v'_j - v_j}. \end{aligned}$$

Further, we know that $\|t - w\|_\infty \leq \beta$, and hence we have $v_j \leq v'_j \leq v_j + 2\beta$ for all j. In particular, we have that $v'_j - v_j \geq 0$ and so the ideal $\prod_j \mathfrak{p}_j^{v'_j - v_j}$ is an integral ideal. We conclude that J is contained in I, and in particular $g \cdot g'$ is indeed an element of I. Also, because $g' \neq 0$ (see Lemma 3.1) and $g \neq 0$ (we chose I to be non-zero), then $g \cdot g'$ is a non-zero element of I.

Let us now show that $g \cdot g'$ is short. We will do so by using Lemma 2.12. Let $\operatorname{Log} g = h_g + a_g \mathbf{1}$ and $\operatorname{Log} g' = h_{g'} + a_{g'} \mathbf{1}$ with h_g and $h_{g'} \in H \cap E$ (note that because $g, g' \in K$, we do not necessarily have $a_g, a_{g'} > 0$). We then have that $\operatorname{Log}(gg') = (h_g + h_{g'}) + (a_g + a_{g'})\mathbf{1}$. By Lemma 2.12, we know that $\|gg'\|_2 \leq \sqrt{n} \cdot |\mathcal{N}(gg')|^{1/n} \cdot 2^{\|h_g + h_{g'}\|_2}$. Therefore, it suffices to bound the two terms $|\mathcal{N}(gg')|^{1/n}$ and $\|h_g + h_{g'}\|_2$.

Let us start by $|\mathcal{N}(gg')|^{1/n}$. By multiplicativity of the algebraic norm, we have that $|\mathcal{N}(gg')|^{1/n} = \mathcal{N}(J)^{1/n} = \mathcal{N}(I)^{1/n} \cdot \prod_j \mathcal{N}(\mathfrak{p}_j)^{\frac{v'_j - v_j}{n}}$. We have chosen the ideals $\mathfrak{p}_j$ with polynomially bounded algebraic norms, and we have seen that $0 \leq v'_j - v_j \leq 2\beta$. Thus, we obtain that $\mathcal{N}(\mathfrak{p}_j)^{\frac{v'_j - v_j}{n}} = 2^{O(\frac{\beta \log\log|\Delta|}{n})}$. By taking the product of the r ideals $\mathfrak{p}_j$, we obtain

$$|\mathcal{N}(gg')|^{1/n} = \mathcal{N}(I)^{1/n} \cdot 2^{O(\frac{\beta \cdot r \cdot \log\log|\Delta|}{n})}.$$

We now consider the term $\|h_g + h_{g'}\|_2$. Recall that $\|w - t\|_\infty \leq \beta$, so in particular, if we consider only the first $r_1 + r_2 - 1$ coefficients of the vectors, we have that $\|c \cdot f_{H \cap E}(h_{g'}) + c \cdot f_{H \cap E}(h_g)\|_\infty \leq \beta$. And if we consider the ℓ_2-norm, we obtain $\|f_{H \cap E}(h_{g'}) + f_{H \cap E}(h_g)\|_2 \leq \sqrt{n}\beta/c$. Using the fact that the ℓ_2-norm is invariant by $f_{H \cap E}$, we conclude that $\|h_{g'} + h_g\|_2 \leq \sqrt{n}\beta/c$.

Finally, combining the two upper bounds above and replacing c by $n^{1.5}/r$, we obtain that

$$\|gg'\|_2 \leq \sqrt{n} \cdot 2^{O(\frac{\beta \cdot r \cdot \log\log|\Delta|}{n})} \cdot \mathcal{N}(I)^{1/n}.$$

Cost. Step 1 can be performed in time $T_{\text{decomp}}(\mathcal{N}(I), \text{poly}(\log|\Delta|), \text{poly}(\log|\Delta|))$. Step 2 can be performed in polynomial time. Step 3 uses a CVP solver and can be done in time $T_{\text{CVP}}(\infty, L, \beta)$. Finally, Step 4 only consists in recovering g' from the vector w, it can be done in polynomial time. Note that for this last step, if we only have the vector w, then we know $\pi_H(\operatorname{Log} g')$, but it might not be possible to recover g' from it. On the other hand, the lower part of the vector w also gives us the ideal $\langle g' \rangle$, but then computing g' from it would be costly. In order to perform this step in polynomial time, when creating the matrix B_L we keep in memory the elements g_i corresponding to the different columns. Then, when we obtain w, we only have to write it as a linear combination of the vectors of B_L and we can recover g' as a product of the g_i's. This can also be done in polynomial time. □

Combining Algorithm 3.2 with the pre-processing phase (i.e., computing B_L with Algorithm 3.1 and pre-processing it for approx-CVPP'), we obtain the following theorem.

Theorem 3.4. *Let K be any number field of dimension n and discriminant Δ. Let $\alpha \in [0,1]$, $r = \mathrm{poly}(\log|\Delta|)$ be such that $\log h_K \leq r$, and $\nu := r + r_1 + r_2 - 1$. Then, under GRH, there exist two algorithms $A_{pre\text{-}proc}$ and A_{query} such that*

- *Algorithm $A_{pre\text{-}proc}$ takes as inputs the field K and a basis of its ring of integers R, runs in time*

  $$T_{\mathrm{CVP}}^{\text{pre-proc}}(\infty, L, \nu^\alpha) + T_{\log\text{-unit}} + 2 \cdot T_{\mathrm{rel}}(\mathrm{poly}(\log|\Delta|), \mathrm{poly}(\log|\Delta|)) + \mathrm{poly}(\log|\Delta|)$$

  and outputs a hint w of bit-size at most $T_{\mathrm{CVP}}^{\mathrm{query}}(\infty, L, \nu^\alpha)$;
- *Algorithm A_{query} takes as inputs the hint w output by $A_{pre\text{-}proc}$ and any fractional ideal I of R such that the numerator and denominator of $\mathcal{N}(I)$ have bit-sizes bounded by $\mathrm{poly}(\log|\Delta|)$; it runs in time*

  $$T_{\mathrm{decomp}}(\mathcal{N}(I), \mathrm{poly}(\log|\Delta|), \mathrm{poly}(\log|\Delta|)) + T_{\mathrm{CVP}}^{\mathrm{query}}(\infty, L, \nu^\alpha) + \mathrm{poly}(\log|\Delta|)$$

  and outputs a non-zero element $x \in I$ such that

  $$\|x\|_2 \leq 2^{O(\frac{\nu^\alpha \cdot r \cdot \log\log|\Delta|}{n})} \cdot \lambda_1^{(2)}(I).$$

The lattice L is as defined in Section 3.1 and only depends on the field K. The memory consumption of both algorithms is bounded by their run-times.

Note that we used the fact that $\lambda_1^{(2)}(I) \geq \lambda_1^{(\infty)}(I) \geq \mathcal{N}(I)^{1/n}$ (see Inequality (2.4)) to replace the $\mathcal{N}(I)^{1/n}$ term in Theorem 3.3 by $\lambda_1^{(2)}(I)$.

4 Solving CVP' with Pre-processing

In this section, we describe a possible way of solving approx-CVP' in the lattice L defined previously. Even if our lattice L has some structure, it does not seem easy to solve approx-CVP' in it (not necessarily easier than solving the approx-SVP instance directly for the initial lattice I). However, the lattice L only depends on the field K and not on the ideal I. Hence, in this section, we focus on solving approx-CVP' with pre-processing on the lattice (to which we refer as CVPP'). Combining it with the result of Section 3, this will provide an algorithm to solve approx-SVP in ideals, with pre-processing on the field K.

4.1 Properties of the lattice L

Recall that our lattice L is given by the basis matrix $B_L = \left(\begin{array}{c|c} c \cdot B_\Lambda & A_{\mathfrak{B}} \\ \hline 0 & R_{\mathfrak{B}} \end{array}\right) \in \mathbb{R}^{\nu \times \nu}$, where we let $A_{\mathfrak{B}}$ denote the top-right block of B_L consisting of the vectors $c \cdot \widetilde{h}_{g_i}$, and $R_{\mathfrak{B}}$ be the bottom-right block of B_L containing the relations of the elements of $\mathfrak{B}$. Recall that $R_{\mathfrak{B}}$ is a basis of the kernel of $f_{\mathfrak{B}} : (e_1, \ldots, e_r) \in \mathbb{Z}^r \mapsto [\prod_j \mathfrak{p}_j^{e_j}] \in Cl_K$. Hence we have $\det(R_{\mathfrak{B}}) = |\mathbb{Z}^r / \ker(f_{\mathfrak{B}})| = h_K$.

Equation (2.6) gives that $\det(\Lambda) \cdot h_K \leq 2^{O(\log|\Delta|+n\log\log|\Delta|)}$. Hence, we have that $\det(L) = c^{r_1+r_2-1} \cdot 2^{O(\log|\Delta|+n\log\log|\Delta|)}$. We chose $c = n^{1.5}/r$ in Theorem 3.3. We then obtain the following upper bound on $\det(L)$:

$$\det(L) = \left(\frac{n^{1.5}}{r}\right)^{r_1+r_2-1} \cdot 2^{O(\log|\Delta|+n\log\log|\Delta|)} = 2^{O(\log|\Delta|+n\log\log|\Delta|)}.$$

We still have some freedom for the choice of the parameter r (and hence the dimension $\nu = r + r_1 + r_2 - 1$ of the lattice L), as long as $\log h_K \leq r$. We will choose it sufficiently large to ensure that the root determinant of L is at most constant. On the other hand, the dimension of L should be as small as possible as it impacts the cost of the CVP computations. We fix

$$r = \max(\log h_K, \log|\Delta| + n\log\log|\Delta|).$$

This choice of r satisfies $r \geq \log h_K$ and $\det(L)^{1/\nu} \leq O(1)$. Note that as $\log h_K = \widetilde{O}(\log|\Delta|)$ (see Equation (2.5)), we have $r \leq \widetilde{O}(\log|\Delta|)$.

In the following, we view the lattice L as random, where the randomness comes from the choice of the set $\mathfrak{B}$ (the initial set $\mathfrak{B}''$ in Algorithm 3.1 is fixed, but then we add to it random prime ideals of polynomially bounded norms to create the set $\mathfrak{B}$). If the created lattice L does not satisfy the conditions we want, we can try another lattice by sampling a new set $\mathfrak{B}$. As we chose r so that $\det(L)^{1/\nu} = O(1)$, we know by Minkowski's inequality that $\lambda_1^{(\infty)}(L) = O(1)$. Then, because L is somehow random, we also expect that all successive minima $\lambda_i^{(\infty)}(L)$ and the covering radius in infinity norm are constant. Hence, we expect to be able to take β as small as $O(1)$ in Algorithm 3.2. We summarize this assumption below.

Heuristic 4. With good probability over the choice of $\mathfrak{B}$, the ℓ_∞-norm covering radius of L satisfies $\mu^{(\infty)}(L) = O(1)$ (and hence $\mu^{(2)}(L) = O(\sqrt{\nu})$).

This heuristic calls for a few comments. First, it is better analyzed as two separate conjectures, one on the log-unit lattice, the other one on the class group lattice. Concerning the latter, assume that the class number is a prime p. Then we can choose $\mathfrak{p}_1$ to be a generator of the class group, and the relation matrix is of the form

$$\begin{pmatrix} p & a_1 & a_2 & \dots & a_r \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix},$$

where the a_i's in $[0, p-1]$ characterize the elements of the ideal class group. In our setting where each $\mathfrak{p}_i$ (for $i \geq 2$) is picked randomly among small prime ideals, we can thus reasonably assume that the a_i's are uniformly distributed in $[0, p-1]$. Hence, for $(e_i)_{2\leq i\leq r} \in [-B, B]^{r-1}$ for some constant $B \geq 1$, we can expect that one among the $(2B+1)^{r-1} = p^c$ (with $c > 1$) fractional ideals

$\prod_{i\geq 2} \mathfrak{p}_i^{e_i}$ is in a class $[\mathfrak{p}_1]^a$ for some $a = O(1)$, which implies that the ℓ_∞-norm covering radius is $O(1)$.

The general case is analogous to this first intuition. Let $\mathfrak{B} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_s, \mathfrak{p}_{s+1}, \ldots, \mathfrak{p}_r\}$ with $\{\mathfrak{p}_1, \ldots, \mathfrak{p}_s\}$ the prime ideals coming from the set $\mathfrak{B}''$ (hence fixed) and $\{\mathfrak{p}_{s+1}, \ldots, \mathfrak{p}_r\}$ the ideals uniformly chosen among prime ideals of norm bounded by some polynomial. Because the set $\mathfrak{B}''$ generates the class group, we can find a basis of L of the following form, by taking the HNF matrix for the bottom-right part of B_L.

$$B'_L := \begin{array}{c} \overset{r_1+r_2-1}{\longleftrightarrow}\ \overset{r}{\longleftrightarrow} \\ \left(\begin{array}{cc|c} c \cdot B_\Lambda & c \cdot \widetilde{h}_{g_1}, \ldots, c \cdot \widetilde{h}_{g_r} & \\ \hline & R_{\mathfrak{B}''} & v_{s+1} \cdots v_r \\ \hline 0 & & \begin{array}{ccc} 1 & & \\ & \ddots & \\ & & 1 \end{array} \end{array}\right) \end{array} \begin{array}{l} \updownarrow r_1+r_2-1 \\ \updownarrow s \\ \updownarrow r \end{array}$$

In this matrix, the block matrices B_Λ and $R_{\mathfrak{B}''}$ are fixed, as well as the vectors $\widetilde{h}_{g_i}$ for i in $\{1, \ldots, s\}$. However, the vectors v_i and $\widetilde{h}_{g_i}$ for $s < i \leq r$ depend on our choices of $\{\mathfrak{p}_{s+1}, \ldots, \mathfrak{p}_r\}$. The vectors of $\mathbb{Z}^s/R_{\mathfrak{B}''}$ are in bijection with the elements of the ideal class group (because $\mathfrak{B}''$ generates the class group). So if we assume that the class of a uniform prime ideal of norm polynomially bounded is uniform in the class group, then we would have that the vectors v_i of the matrix above are uniform in $\mathbb{Z}^s/R_{\mathfrak{B}''}$. In a similar way, we will assume that the vectors $\widetilde{h}_{g_i}$ are somehow uniform in $\mathbb{R}^{r_1+r_2-1}/\Lambda$ (recall that they correspond to the projection over H of $\operatorname{Log} g_i$ for g_i a generator of the principal ideal associated with the lower part of the vector). Let us now explain why, given any target vector $t \in \mathbb{R}^\nu$, we expect to find a vector $v \in L$ at distance $O(1)$ from t. Write $t = (c \cdot \widetilde{h}, v, w)^T$ with $\widetilde{h}$ of dimension $r_1 + r_2 - 1$, v of dimension s and w of dimension $r - s$. We can assume, without loss of generality, that $|w_i| < 1/2$ for all i (using the last $r-s$ columns of B'_L to reduce t if needed). By taking the subset sums of the last $r - s$ columns of B'_L, we obtain 2^{r-s} vectors of L of the form $t' = (c \cdot \widetilde{h'}, v', w')^T$, with $w' \in \{0, 1\}^{r-s}$. Because we assumed that the v_i and $\widetilde{h}_{g_i}$ for $s < i \leq r$ were somehow uniform modulo $R_{\mathfrak{B}''}$ and Λ respectively, we also expect the vectors $\widetilde{h'}$ and v' created above to be somehow uniform modulo $R_{\mathfrak{B}''}$ and Λ. Recall that we chose r so that $(\det(c\Lambda) \cdot \det(R_{\mathfrak{B}''}))^{1/r} = O(1)$, hence the volume of $c\Lambda$ and $R_{\mathfrak{B}''}$ satisfies $\det(c\Lambda) \cdot \det(R_{\mathfrak{B}''}) \leq 2^{O(r)}$. We can then assume that we have $2^{r-s} > \det(c\Lambda) \cdot \det(R\mathfrak{B}'')$ (if needed, we can multiply r by a constant factor, which will not change the asymptotics). This means that we expect to find one of the 2^{r-s} vector $t' = (c \cdot \widetilde{h'}, v', w')^T$ satisfying

$\|(c \cdot \widetilde{h}, v) - (c \cdot \widetilde{h'}, v')\|_\infty = O(1)$. And because $|w_i| < 1/2$ and $w'_i \in \{0, 1\}$ we also have $\|t - t'\|_\infty = O(1)$.

We experimentally computed the lattice L for some cyclotomic fields (using Algorithm 3.1). For each lattice L, we then computed an empirical covering radius. To do so, we picked 21 random target vectors t_i in the real span of the lattice. These vectors were sampled from a continuous Gaussian distribution with standard deviation $\sigma = 100$. We then solved the CVP instances in L for these target vectors t_i and let v_i be a closest vector in L (for the ℓ_2-norm). We defined $\widetilde{\mu}^{(2)}(L)$ to be $\max_i \|t_i - v_i\|_2$ and $\widetilde{\mu}^{(\infty)}(L)$ to be $\max_i \|t_i - v_i\|_\infty$.[5] The approximated values of $\mu^{(2)}(L)$ and $\mu^{(\infty)}(L)$ are given in Table 4.1. We observe that, while $\widetilde{\mu}^{(2)}(L)$ increases with the dimension (we expect that it increases as $\sqrt{\nu}$), the approximate covering radius in ℓ_∞-norm $\widetilde{\mu}^{(\infty)}(L)$ seems to remain constant around 1. These experimental results are consistent with Heuristic 4. The code is available as supplementary material.

Conductor of K	Dimension of L	$\widetilde{\mu}^{(2)}(L)$	$\widetilde{\mu}^{(\infty)}(L)$
18	9	1.13	0.755
16	16	1.50	0.899
36	28	1.79	0.795
40	41	2.15	0.893
48	42	2.19	0.840
32	44	2.26	0.794
27	49	2.36	0.901
66	54	2.47	0.989
44	57	2.53	0.815
70	67	2.72	1.03
84	68	2.74	1.27
90	68	2.71	0.814
78	70	2.81	0.882
72	73	2.90	1.00

Fig. 4.1. Approximate covering radii in ℓ_2 and ℓ_∞ norms for the lattice L, for cyclotomic number fields of different conductors.

4.2 Using Laarhoven's algorithm

We now consider Laarhoven's algorithm, which solves approx-CVPP in Euclidean norm (we only found algorithms for CVPP with Euclidean norm in the literature, and not for infinity norm). Recall from Section 2 that, for a ν-

[5] As we solved CVP in L for the ℓ_2-norm, the quantity $\mu^{(\infty)}(L)$ may be over-estimated, but this should not be over-estimated by too much. Further, as we want an upper bound on $\mu^{(\infty)}(L)$, this is not an issue.

dimensional lattice L, Laarhoven's (heuristic) algorithm gives, for any $\alpha \in [0, 1/2]$:

$$T_{\mathrm{CVP}}^{\text{pre-proc}}(2, L, O(\nu^\alpha) \cdot \mu^{(2)}(L)) = 2^{O(\nu)},$$
$$T_{\mathrm{CVP}}^{\text{query}}(2, L, O(\nu^\alpha) \cdot \mu^{(2)}(L)) = 2^{\widetilde{O}(\nu^{1-2\alpha})}.$$

As we have assumed (Heuristic 4) that $\mu^{(2)}(L) = O(\sqrt{\nu})$ with good probability over the choice of L, this implies that Laarhoven's algorithm achieves

$$T_{\mathrm{CVP}}^{\text{pre-proc}}(2, L, O(\nu^{1/2+\alpha})) = 2^{O(\nu)} \quad \text{and} \quad T_{\mathrm{CVP}}^{\text{query}}(2, L, O(\nu^{1/2+\alpha})) = 2^{\widetilde{O}(\nu^{1-2\alpha})}.$$

We now have an algorithm that, given any input $t \in \mathrm{Span}(L)$, outputs a vector $v \in L$ such that $\|t - v\|_2 \leq O(\nu^{1/2+\alpha})$, while we would like to have $\|t - v\|_\infty \leq O(\nu^\alpha)$. But when we take a random vector of Euclidean norm bounded by $O(\nu^{1/2+\alpha})$, we expect that with good probability, its coefficients are somehow balanced. Hence we expect its infinity norm to be approximately $\sqrt{\nu}$ times smaller than its Euclidean norm. This is the meaning of the following heuristic assumptions.

First, because we want the output of Laarhoven's algorithm to be somehow random, we argue that we can randomize the input vector of the CVPP algorithm.

Heuristic 5. We assume that in our algorithm, the target vector t given as input to Laarhoven's algorithm behaves like a random vector sampled uniformly in $\mathrm{Span}(L)/L$.

This assumption that t is distributed uniformly in $\mathrm{Span}(L)/L$ may be justified by the fact that, in Algorithm 3.2, we can somehow randomize our target vector t by multiplying our initial fractional ideal I by an integral ideal of small algebraic norm (statistically independent of the $\mathfrak{p}_i$'s chosen for $\mathfrak{B}$).

Heuristic 6. With non-negligible probability over the input target vector t, distributed uniformly in $\mathrm{Span}(L)/L$, the vector v output by Laarhoven's algorithm satisfies $\|t - v\|_\infty \leq \widetilde{O}(\|t - v\|_2/\sqrt{\nu})$.

In order to motivate Heuristic 6, we recall that if a vector is drawn uniformly at random on a sphere, then its ℓ_∞-norm is smaller than its ℓ_2-norm by a factor $O(\log n/\sqrt{n})$, with good probability.

Lemma 4.1. *Let x be sampled uniformly on the unit sphere S^{n-1} in $\mathbb{R}^n$. Then* $\Pr(\|x\|_\infty \geq \frac{\sqrt{8}\ln n}{\sqrt{n}}) \leq O(\frac{1}{\sqrt{\ln n}})$.

Proof. Sampling x uniformly in S^{n-1} is the same as sampling y from a centered spherical (continuous) Gaussian distribution of parameter 1 and then normalizing it by setting $x = \frac{y}{\|y\|_2}$. So we have $\|x\|_\infty = \frac{\|y\|_\infty}{\|y\|_2}$, and it is sufficient to find an upper bound on $\|y\|_\infty$ and a lower bound on $\|y\|_2$. We know that for a centered spherical Gaussian distribution of parameter 1, we have $\Pr(\|y\|_\infty > 2\ln n) = \Pr(\exists i : |y_i| > 2\ln n) \leq \frac{1}{2\sqrt{2\pi \ln n}}$. Moreover, we also have that $\Pr(\|y\|_2 < \sqrt{n/2}) \leq \mathrm{e}^{-n/8}$ (see for instante [LM00, Lemma 1]). By the union bound, we finally obtain that $\Pr(\|y\|_\infty/\|y\|_2 > \frac{\sqrt{8}\ln n}{\sqrt{n}}) \leq O(\frac{1}{\sqrt{\ln n}})$. □

Note that the proof also shows that for a continuous Gaussian vector y of dimension n, $\|y\|_\infty/\|y\|_2 = O(\log n/\sqrt{n})$ with good probability. We also have experimental results corroborating Heuristic 6. We implemented our algorithm in Magma, both the generation of the lattice L and the CVP phase using Laarhoven's algorithm (the code is available as supplementary material). We tested our implementation for different cyclotomic fields. The maximum conductor achieved was 90. The maximum dimension of the lattice L that we achieved was 73, for a cyclotomic field of conductor 72. For these cyclotomic fields, we computed the lattice L. Then, we sampled target vectors t in the real span of L, using a Gaussian distribution of parameter $\sigma = 100$, and we ran Laarhoven's CVP algorithm to obtain a vector $v \in L$. We then computed the ratios $\frac{\|t-v\|_2}{\|t-v\|_\infty}$, which we expect to be around $O(\sqrt{\nu}/\log \nu)$. Because we are working in small dimensions, the $\log \nu$ term has a non-negligible impact. So, instead of plotting $\log(\frac{\|t-v\|_2}{\|t-v\|_\infty})$ as a function of $\log \nu$, we compared our ratios with the ones we would have obtained if the vectors were Gaussian vectors. On Figure 4.2, the blue dots represent the logarithms of the ratios $\frac{\|t-v\|_2}{\|t-v\|_\infty}$ obtained when choosing a random Gaussian vector t as input of our algorithm. For every fixed conductor, we have several vertically aligned points, because we tried Laarhoven's algorithm for different approximation factors (i.e., different choices of α). The green '+' are obtained by computing $\log(\|x\|_2/\|x\|_\infty)$ for some Gaussian vectors of dimension ν. The red crosses are obtained by taking the median point of a large number of green '+' (not all of them are plotted on the figure).

We observe that the ratios obtained with our algorithm are well aligned with the red crosses. Moreover, even if we have some variance within the blue dots, it is comparable to the variance observed within the green '+'. So Heuristic 6 seems consistent with our empirical experiments (recall that Gaussian vectors provably satisfy Heuristic 6 with good probability).

We conclude that, under Heuristics 4, 5 and 6, and Heuristic 1 present in [Laa16], for any $\alpha \in [0, 1/2]$, Laarhoven's algorithm solves approx-CVPP' with

$$T_{\mathrm{CVP}}^{\text{pre-proc}}(\infty, L, \nu^\alpha) = 2^{O(\nu)} \quad \text{and} \quad T_{\mathrm{CVP}}^{\text{query}}(\infty, L, \nu^\alpha) = 2^{\widetilde{O}(\nu^{1-2\alpha})}. \tag{4.1}$$

5 Summary

We now instantiate Theorem 3.4 with $\nu = \widetilde{O}(\log \Delta)$ and the values given in Section 2 and in Equation (4.1) for $T_{\text{log-unit}}, T_{\text{decomp}}, T_{\text{rel}}, T_{\mathrm{CVP}}^{\text{pre-proc}}$ and $T_{\mathrm{CVP}}^{\text{query}}$.

Theorem 5.1. *Let K be any number field of dimension n and discriminant Δ. Let $\alpha \in [0, 1/2]$. Then, under GRH and Heuristics 1-6,there exist two algorithms $A_{pre\text{-}proc}$ and A_{query} such that*

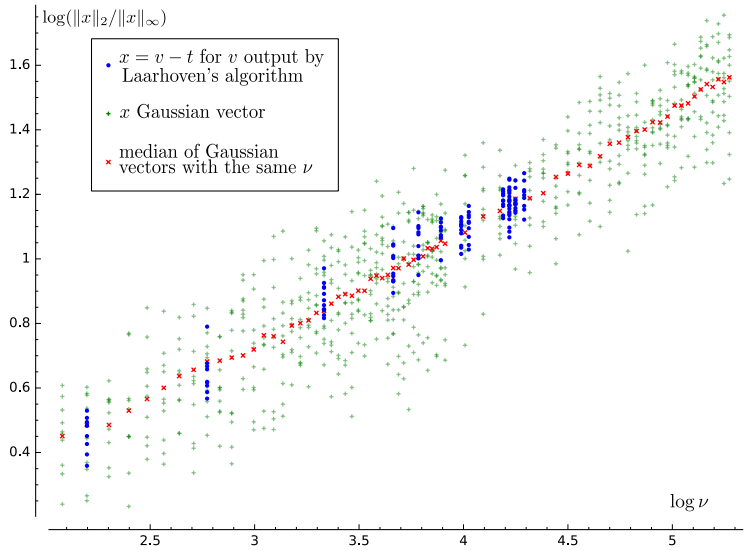

Fig. 4.2. Comparison of $\log(\|x\|_2/\|x\|_\infty)$ as a function of $\log \nu$ for x a Gaussian vector or $x = t - v$ with t a random target and v the approx-CVP solution output by Laarhoven's algorithm (on our lattice L, in selected cyclotomic fields).

- *Algorithm $A_{\text{pre-proc}}$ takes as inputs the field K and a basis of its integer ring R, runs in time $2^{\widetilde{O}(\log|\Delta|)}$ and outputs a hint w of bit-size at most $2^{\widetilde{O}((\log|\Delta|)^{1-2\alpha})}$,*
- *Algorithm A_{query} takes as inputs the hint w output by $A_{\text{pre-proc}}$ and any fractional ideal I of R such that the numerator and denominator of $\mathcal{N}(I)$ have bit-sizes bounded by $\text{poly}(\log|\Delta|)$. It runs in classical time $2^{\widetilde{O}((\log|\Delta|)^{\max(2/3,1-2\alpha)})}$ or in quantum time $2^{\widetilde{O}((\log|\Delta|)^{1-2\alpha})}$ and outputs an element $x \in I$ such that $0 < \|x\|_2 \leq 2^{\widetilde{O}(\frac{(\log|\Delta|)^{\alpha+1}}{n})} \cdot \lambda_1^{(2)}(I)$.*

The memory consumption of both algorithms is bounded by their run-times.

In the case where $\log|\Delta| = \widetilde{O}(n)$, we can replace $\log|\Delta|$ by n in all the equations of Theorem 5.1, and we obtain an element x which is a $2^{\widetilde{O}(n^\alpha)}$ approximation of a shortest non-zero vector of I (see Figure 5.2). On the other hand, if $\log|\Delta|$ becomes significantly larger than n, then both the run-time and the approximation factor degrade. The cost of the pre-computation phase also becomes larger than $2^{O(n)}$. However, the query phase still improves upon the BKZ algorithm, for some choices of α, as long as $\log|\Delta| = \widetilde{O}(n^{12/11})$ in the classical setting or $\log|\Delta| = \widetilde{O}(n^{4/3})$ in the quantum setting (see Figure 5.3). In Figures 5.1, 5.2 and 5.3, we plot the ratios between time and approximation factor for the BKZ algorithm and the query phase of our algorithm, in the different regimes $\log|\Delta| = \widetilde{O}(n)$ and $\log|\Delta| = \widetilde{O}(n^{1+\varepsilon})$ for some $\varepsilon > 0$.

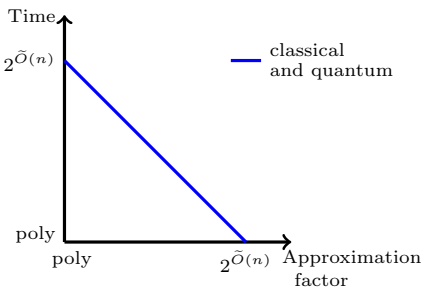

Fig. 5.1. Prior time/approximation trade-offs for approx-SVP in ideal lattices in any number field of degree n (using the BKZ algorithm).

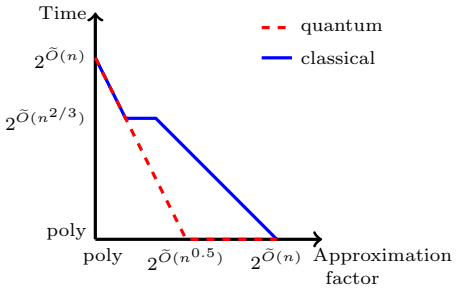

Fig. 5.2. New trade-offs for ideal lattices in number fields satisfying $\log|\Delta| = \widetilde{O}(n)$ (with a pre-processing of cost $\exp(\widetilde{O}(n))$).

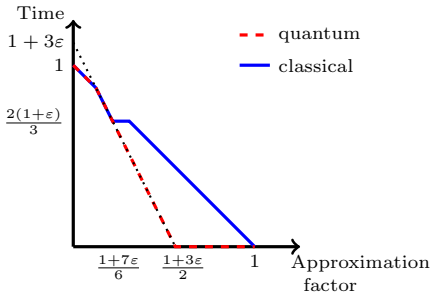

Fig. 5.3. New trade-offs for ideal lattices in number fields satisfying $\log|\Delta| = \widetilde{O}(n^{1+\varepsilon})$ for some $\varepsilon > 0$ (with a pre-processing of cost $\exp(\widetilde{O}(n^{1+\varepsilon}))$).

In the case of prime-power cyclotomic fields, we know that $\log|\Delta| = \widetilde{O}(n)$. Moreover, there is a heuristic algorithm of Biasse *et al.* [BEF+17] satisfying $T_{\text{rel}}, T_{\text{decomp}}, T_{\text{log-unit}} \approx 2^{\widetilde{O}(n^{1/2})}$. Hence, we obtain the trade-offs shown in Figure 1.2 (in the introduction) when applying our algorithm to prime-power cyclotomic fields. Recall that in this special case, we already had an improvement upon the BKZ algorithm in the quantum setting, using the results of [CDPR16] and [CDW17], see Figure 1.1.

Acknowledgments. We thank Léo Ducas for his suggestion to use Laarhoven's CVPP algorithm. We thank Oded Regev and Noah Stephens-Davidowitz for illustrating the importance of limiting the witness size by the run-time of the query phase, by pointing out the faster algorithm with exponential-size witness described in the introduction. We also thank Dan Bernstein, Elena Kirshanova and Alexandre Wallet for helpful discussions.

This work was supported in part by BPI-France in the context of the national project RISQ (P141580), by the European Union PROMETHEUS project (Horizon 2020 Research and Innovation Program, grant 780701) and by the ERC Starting Grant ERC-2013-StG-335086-LATTAC.

References

AD17. Martin R. Albrecht and Amit Deo. Large modulus ring-LWE $\geq$ module-LWE. In *ASIACRYPT 2017*, pp. 267–296. Springer, 2017.

Bac90. Eric Bach. Explicit bounds for primality testing and related problems. *Mathematics of Computation*, 55(191):355–380, 1990.

BBV+17. Jens Bauch, Daniel J. Bernstein, Henry de Valence, Tanja Lange, and Christine van Vredendaal. Short generators without quantum computers: the case of multiquadratics. In *Eurocrypt*, pp. 27–59. Springer, 2017.

BEF+17. Jean-François Biasse, Thomas Espitau, Pierre-Alain Fouque, Alexandre Gélin, and Paul Kirchner. Computing generator in cyclotomic integer rings. In *Eurocrypt*, pp. 60–88. Springer, 2017.

Ber14. D. J. Bernstein. A subfield-logarithm attack against ideal lattices: Computational algebraic number theory tackles lattice-based cryptography. The cr.yp.to blog, 2014. https://blog.cr.yp.to/20140213-ideal.html.

BF14. Jean-François Biasse and Claus Fieker. Subexponential class group and unit group computation in large degree number fields. *LMS Journal of Computation and Mathematics*, 17(A):385–403, 2014.

Bia14. Jean-François Biasse. Subexponential time ideal decomposition in orders of number fields of large degree. *Adv. Math. Comm.*, 8(4):407–425, 2014.

Bia17. Jean-François Biasse. Approximate short vectors in ideal lattices of $\mathbb{Q}(\zeta_{p^e})$ with precomputation of $\mathrm{Cl}(\mathcal{O}_k)$. In *SAC*, pp. 374–393. Springer, 2017.

BS96. Eric Bach and Jeffrey Outlaw Shallit. *Algorithmic Number Theory: Efficient Algorithms*, vol. 1. MIT press, 1996.

BS16. Jean-François Biasse and Fang Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *SODA*, pp. 893–902. Society for Industrial and Applied Mathematics, 2016.

Buc88. Johannes Buchmann. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. *Séminaire de théorie des nombres, Paris*, 1989(1990):27–41, 1988.

BV11a. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS 2011*, pp. 97–106. IEEE Computer Society, 2011.

BV11b. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO 2011*, pp. 505–524. Springer, 2011.

BV18. Jean-François Biasse and Christine van Vredendaal. Fast multiquatratic S-unit computation and application to the calculation of class groups. In *ANTS-XIII*. Springer, 2018.

CDPR16. Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. Recovering short generators of principal ideals in cyclotomic rings. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, pp. 559–585. Springer, Heidelberg, May 2016.

CDW17. Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short Stickelberger class relations and application to ideal-SVP. In *Eurocrypt*, pp. 324–348. Springer, 2017.

CGS14. Peter Campbell, Michael Groves, and Dan Shepherd. Soliloquy: A cautionary tale, 2014. Available at http://docbox.etsi.org/Workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves_Annex.pdf.

Coh13. Henri Cohen. *A course in computational algebraic number theory*, vol. 138. Springer Science & Business Media, 2013.

DLW19. Emmanouil Doulgerakis, Thijs Laarhoven, and Benne de Weger. Finding closest lattice vectors using approximate Voronoi cells. In *PQCRYPTO*. Springer, 2019. To appear.

DPW19. Léo Ducas, Maxime Plançon, and Benjamin Wesolowski. On the Shortness of Vectors to be found by the Ideal-SVP Quantum Algorithm, 2019. To appear.

EHKS14. Kirsten Eisenträger, Sean Hallgren, Alexei Kitaev, and Fang Song. A quantum algorithm for computing the unit group of an arbitrary degree number field. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pp. 293–302. ACM Press, May / June 2014.

Gel17. Alexandre Gelin. *Calcul de groupes de classes d'un corps de nombres et applications à la cryptologie*. PhD thesis, Paris 6, 2017.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pp. 169–178. ACM Press, May / June 2009.

GGH13. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT 2013*, pp. 1–17. Springer, 2013.

HM89. James L. Hafner and Kevin S. McCurley. A rigorous subexponential algorithm for computation of class groups. *Journal of the American mathematical society*, 2(4):837–850, 1989.

HWB17. Patrick Holzer, Thomas Wunderer, and Johannes A. Buchmann. Recovering short generators of principal fractional ideals in cyclotomic fields of conductor $p^\alpha q^\beta$. In *International Conference in Cryptology in India*, pp. 346–368. Springer, 2017.

Laa16. Thijs Laarhoven. Sieving for closest lattice vectors (with preprocessing). In *International Conference on Selected Areas in Cryptography*, pp. 523–542. Springer, 2016.

LM00. Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pp. 1302–1338, 2000.

Lou00. Stéphane Louboutin. Explicit bounds for residues of dedekind zeta functions, values of l-functions at $s = 1$, and relative class numbers. *Journal of Number Theory*, 85(2):263–282, 2000.

LPR10. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, pp. 1–23. Springer, Heidelberg, May 2010.

LS15. Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.

Min67. Hermann Minkowski. *Gesammelte Abhandlungen*. Chelsea, New York, 1967.

PRS17. Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-LWE for any ring and modulus. In *STOC 2017*, pp. 461–473. ACM, 2017.

RBV04. Ghaya Rekaya, Jean-Claude Belfiore, and Emanuele Viterbo. A very efficient lattice reduction tool on fast fading channels. In *ISITA*, 2004.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pp. 84–93. ACM Press, May 2005.

Sam13. Pierre Samuel. *Algebraic Theory of Numbers: Translated from the French by Allan J. Silberger*. Courier Corporation, 2013.

Sch87. Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical computer science*, 53:201–224, 1987.

SE94. Claus-Peter Schnorr and Martin Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66:181–199, 1994.

SSTX09. Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, pp. 617–635. Springer, Heidelberg, December 2009.

Ste19. Noah Stephens-Davidowitz. A time-distance trade-off for GDD with preprocessing – instantiating the DLW heuristic, 2019. Personal communication.

Zim80. Rainer Zimmert. Ideale kleiner Norm in Idealklassen und eine Regulatorabschätzung. *Inventiones mathematicae*, 62(3):367–380, 1980.