Time-Memory Trade-Off for Lattice Enumeration in a Ball

Paul Kirchner[1] and Pierre-Alain Fouque[2]

[1]École normale superieure, France, Paul.Kirchner@ens.fr
[2]Universite Rennes 1 and Institut Universitaire de France,
Pierre-Alain.Fouque@ens.fr

February 29, 2016

Abstract

Enumeration algorithms in lattices are a well-known technique for solving the Short Vector Problem (SVP) and improving blockwise lattice reduction algorithms. Here, we propose a new algorithm for enumerating lattice point in a ball of radius $1.156\lambda_1(\Lambda)$ in time $3^{n+o(n)}$, where $\lambda_1(\Lambda)$ is the length of the shortest vector in the lattice Λ. Then, we show how this method can be used for solving SVP and the Closest Vector Problem (CVP) with approximation factor $\gamma = 1.993$ in a n-dimensional lattice in time $3^{n+o(n)}$. Previous algorithms for enumerating take super-exponential running time with polynomial memory. For instance, Kannan algorithm takes time $n^{n/(2e)+o(n)}$, however ours also requires exponential memory and we propose different time/memory tradeoffs.

Recently, Aggarwal, Dadush, Regev and Stephens-Davidowitz describe a randomized algorithm with running time $2^{n+o(n)}$ at STOC' 15 for solving SVP and approximation version of SVP and CVP at FOCS'15. However, it is not possible to use a time/memory tradeoff for their algorithms. Their main result presents an algorithm that samples an exponential number of random vectors in a Discrete Gaussian distribution with width below the smoothing parameter of the lattice. Our algorithm is related to the hill climbing of Liu, Lyubashevsky and Micciancio from RANDOM' 06 to solve the bounding decoding problem with preprocessing. It has been later improved by Dadush, Regev, Stephens-Davidowitz for solving the CVP with preprocessing problem at CCC'14. However the latter algorithm only looks for one lattice vector while we show that we can enumerate all lattice vectors in a ball. Finally, in these papers, they use a preprocessing to obtain a succinct representation of some lattice function. We show in a first step that we can obtain the same information using an exponential-time algorithm based on a collision search algorithm similar to the reduction of Micciancio and Peikert for the SIS problem with small modulus at CRYPTO' 13.

1 Introduction

A lattice Λ is defined as the set of all integer combinations of some linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ of $\mathbb{R}^n$. Finding a shortest non-zero vector in a lattice is a well-known difficult computational problem, a.k.a. the Shortest Vector Problem (SVP). The SVP problem has also been shown NP-hard to approximate with randomized reduction [5, 22] to constant factor. In [19], Khot shows that SVP is hard to approximate within factor $2^{(\log n)^{1/2-\epsilon}}$ for any $\epsilon > 0$, unless NP has randomized quasipolynomial time algorithms, and by Haviv and Regev [16] to an almost polynomial factor, $2^{(\log n)^{1-\epsilon}}$.

If we turn our attention to finding a short lattice vector within some multiplicative factor exponential in the dimension, the LLL algorithm discovered by Lenstra, Lenstra and Lovász in 1982 solves this problem in polynomial time [20]. Namely, it returns a non-zero vector of length at most $2^n \lambda_1(\Lambda)$, where $\lambda_1(\Lambda)$ is the length of a shortest vector. Then, Schnorr proposed a hierarchy of polynomial-time algorithms achieving better approximation factors but increasing the running time, also known as BKZ or block reduction algorithms in [34]. The approximation factor achieves by Schnorr's algorithms is $2^{\mathcal{O}(n(\log\log n)^2/\log n)}$ and has been improved by Ajtai et al. in [6] to $2^{\mathcal{O}(n \log\log n/\log n)}$ in polynomial-time. The LLL and BKZ algorithms use a lattice reduction technique that applies successive elementary transformations to the input basis in order to make them shorter and more orthogonal.

In 2001, Ajtai, Kumar and Sivakumar propose the first algorithm to solve the SVP problem in $2^{\mathcal{O}(n)}$ time and space using a sieving technique [6]. Ten years later, Micciancio and Voulgaris proposed also exponential $\tilde{O}(4^n)$-time and $\tilde{O}(2^n)$-space algorithm based on computing Voronoi cells [25] and the ListSieve-Birthday runs in time $2^{2.465\,n+o(n)}$ with $2^{1.233\,n+o(n)}$ space [31]. Finally, Aggarwal, Dadush, Regev, Stephens-Davidowitz at STOC 2015 propose in [2] a SVP algorithm running in time $2^{n+o(n)}$ and in [3]. Some heuristic algorithms have been proposed to improve the running time [27, 26, 38, 40]. Very recently, Becker et al. solve heuristically SVP in time $2^{0.292\,n+o(n)}$ and memory $2^{0.208\,n+(n)}$ [9].

Enumeration is a more basic technique to solve lattice problems that dates back to the early 1980s with the work of Pohst [28], Kannan [18], Finke and Pohst [13]. Many improvements have been proposed later [35, 30, 25, 14]. Kannan enumeration runs in time $n^{n/2e+o(n)}$ with poly(n) [15]. This technique consists in performing an exhaustive search for the best integer combination of the basis vectors. They consequently run in exponential-time (or worse) but find the shortest vector and not an approximation. More efficient pruning techniques have been proposed by Gama, Nguyen and Regev in [14]. To date, though standard enumeration has superexponential complexity, it is still the fastest method for low dimensional lattices due to its polynomial memory complexity.

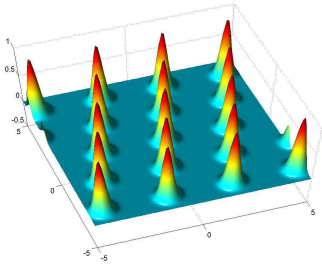
1.1 Related Work

Many recent works on lattice problems [4] use the *periodic Gaussian function* $f : \mathbb{R}^n \to \mathbb{R}^+$ given by $f(\mathbf{t}) = \rho(\Lambda + \mathbf{t})/\rho(\Lambda)$, where $\rho(A) = \sum_{\mathbf{x}\in A} \exp(-\pi||\mathbf{x}||^2)$. This function is described in figure 1. The basic properties of this function are recalled in [11]. For a lattice

Λ of dimension n the f function is about zero for points at distance greater than $\sqrt{n}$, while at distance less than $\sqrt{\log n}$, the function is non-negligible. The function f can be approximated using a list L^* of vectors $\mathbf{w}_i$ in the dual lattice Λ^* of polynomial size N : $f_{L^*}(\mathbf{t}) = (1/N)\sum_{i=0}^{N-1}\cos(2\pi\langle \mathbf{w}_i, \mathbf{t}\rangle)$.

In [24], Micciancio and Peikert describe a reduction between the Short Independent Vector (SIVP) problem and the Short Integer Solution (SIS) problem. Given a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ for $m = \text{poly}(n)$, $q = \text{poly}(n)$ and $\beta > 0$, the SIS problem consists in finding a nonzero integer vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\mathbf{Az} = \mathbf{0} \bmod q$ and $||\mathbf{z}|| \leq \beta$, where $||\cdot||$ denotes the Euclidean norm. Ajtai shows that for some parameters, SIS enjoys a remarquable worst-case/average-case hardness reduction: solving it on average with noticeable probability is at least as hard as approximating some lattice problems in dimension n in the worst-case, to within poly(n) factors. Peikert and Micciancio show that SIS retains its hardness with a modulus q nearly as small as the solution bound β. The main argument of this new reduction is to use Gaussian sampling for reducing a set of independent lattice points at each stage. To generate smaller lattice points in a lattice, random points are combined using the advice given by the collision search oracle.

Figure 1: The periodic Gaussian function f. The picture is taken from [11]



In [11], Dadush, Regev and Stephens-Davidowitz propose a new algorithm to solve the Closest Vector (CVP) problem in lattice with preprocessing. They were able to extend the algorithm of Liu, Lyubashevsky and Micciancio [21] for a point closer to a lattice point than all points. The previous algorithm allows to solve the Bounded Distance Decoding (BDD) problem and the basic idea is to perform a hill climbing from the point close to a lattice point, where the distance is bounded by $\sqrt{\log(n)/n}$, which means that the periodic Gaussian function is non-negligible at this point.

The work of Aggarwal et al. [2] is very interesting since it solves the DGS hard problem: sampling an exponential number of random lattice vectors according to a probability distribution following a Discrete Gaussian with width smaller than the smoothing parameter of the lattice. If the width is above this parameter, the Gaussian behaves as a continuous Gaussian, while below it, the distribution becomes closer to a point distribution in $\mathbf{0}$. They have been able to generate $2^{n/2}$ vectors according to this distribution and they gave a randomized algorithm, since they can generate shortest nonzero vector with a probability larger than $2^{-n/2}$. Consequenly, the overall time and memory complexity of their algorithm is in $2^{n+o(n)}$.

1.2 Our Contributions

We present here a high-level overview of our algorithms. We first show that we can generate the list L^* enabling the approximation of the function f using the output of the Micciancio-Peikert reduction algorithm. The collision finding algorithm required by the Discrete Gaussian Sampling algorithm is the Generalized Birthday Algorithm [36] which requires

exponential time algorithms in this setting. Later, we show that the advice L^* can be used to perform the hill climbing algorithm and perform an enumeration algorithm. This is the first time that enumeration is performed in such a way. From this algorithm, we derive a SVP and CVP algorithm using the results of [11]. Our results are somewhat less efficient than theirs [2] since we need $3^{n+o(n)}$ time and memory, compared to $2^{n+o(n)}$. However, contrary to this result, we also provide a time-memory tradeoff that is not possible with [2]. To do so, it is enough to replace the Generalized Birthday Algorithm by exhaustive search. For a standard deviation of the Gaussian σ above the smoothing parameter of the lattice by a factor $q \geq 2$, we can sample from $D_{\Lambda,\sigma}$ in time $q^{\mathcal{O}(n)}$ and memory $q^{\mathcal{O}(n/q^2)}$ with negligible statistical distance.

Enumeration algorithms are interesting to study since they allow to solve SVP and in [17] to solve the lattice isomorphism problem. Our second contribution is a new algorithm for enumerating all lattice points in a ball of radius $1.156\lambda_1(\Lambda)$ which implies a new algorithm for SVP and γ-CVP in $3^{n+o(n)}$. Contrary to Kannan enumeration, the running time is better but we need exponential memory. We also show that we can have a time/memory tradeoff with polynomial memory which is however worse than Kannan algorithm.

1.3 Techniques

The Discrete Gaussian Sampling (DGS) is a well-known problem introduced in [32] and corresponds to generate discrete Gaussian samples from a lattice. Recently, Aggarwal *et al.* [2] solves it for *any* Gaussian width, even below the smoothing parameter. Here, we only use sampling above the smoothing parameter, while [2] develops a new technique to sample below it. To this end, we use the technique developed by Micciancio and Peikert [24] using efficient collision finding algorithms [36, 12]. We describe an exponential-time algorithm for DGS in time $q^{\mathcal{O}(n)}$ and memory $q^{\mathcal{O}(n/q^2)}$ for some integer $2 \leq q \leq n^{1/5}$. The main idea consists in sampling random lattice vectors, and reducing the length of lattice vectors by iteratively finding smaller and smaller lattice basis. In comparison, the DGS algorithm of [2] runs in $2^{n/2+o(n)}$ time and space and outputs $2^{n/2}$ samples. For $q = 3$, we have a $\mathcal{O}(2^n)$ time algorithm to solve SIS. If we push the idea to its limit, we have an algorithm in time $n^{n/4+o(n)}$ with polynomial memory given a random oracle and using Floyd's cycle algorithm.

Theorem 1.1. *For any $\sigma = \Omega(\sqrt{n/q}/\lambda_1(\Lambda^*))$, $\epsilon = 2^{-\Omega(n)}$ and $q \leq n^{1/5}$, it is possible to sample from $D_{\Lambda,\sigma}$ with statistical distance ϵ, time $q^{\mathcal{O}(n)}$ and memory $q^{\mathcal{O}(n/q^2)}$.*

The idea of our new enumeration algorithm originates from the classical reduction of the Learning With Errors (LWE) problem to bounded decision decoding [33]. A Fourier transform is used to solve LWE, but it does not use properties of the samples enforced by the reduction. We show that one can divide the standard deviation by the modulus p of the LWE, so that the radius of enumeration is multiplied by p.

2 Preliminaries

Any vector $\mathbf{x} \in \mathbb{R}^n$ has an Euclidean norm $||\mathbf{x}||^2 = \sum_{i=0}^{n-1} x_i^2$. We denote by ln the neperian logarithm and log the binary logarithm. To analyze the success probability of our algorithms, we will use the following lemma, which is proven in [29]:

Lemma 2.1. *Let P and Q be two distributions over S, such that $|P(x) - Q(x)| \leq \delta(x)P(x)$ with $\delta(x) \leq 1/4$ for all $x \in S$. Then :*

$$D_{KL}(P||Q) \leq 2\sum_{x \in S} \delta(x)^2 P(x).$$

Theorem 2.2 (Pinsker's inequality)**.** *Let P and Q be two distributions over S, and their statistical distance defines as $D_{SD} = (1/2) \cdot \sum_{x \in S} |P(x) - Q(x)|$. Then $D_{SD} \leq \sqrt{D_{KL}(P||Q)/2}$.*

An algorithm has a negligible probability of failure iff its probability of failure is $2^{-\Omega(n)}$.

2.1 Lattice Background and Problems

A lattice $\Lambda \subset \mathbb{R}^n$ is the set of all integer linear combinations $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \sum_i \mathbf{b_i} \cdot x_i$ (where $x_i \in \mathbb{Z}$) of a set of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ called the basis of the lattice. If $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ is the matrix basis, lattice vectors can be written as $\mathbf{Bx}$ for $\mathbf{x} \in \mathbb{Z}^n$. The length of a shortest non-zero vector in Λ is denoted $\lambda_1(\Lambda)$. For a lattice Λ, the ith successive minimum of Λ is the radius of the smallest ball, centered in $\mathbf{0}$, such that it contains i vectors of the lattice Λ linearly independent. Given a lattice Λ, we define the dual lattice, denoted Λ^* as the set of $\mathbf{x} \in \mathbb{R}^n$ such that $\langle \mathbf{x}, \Lambda \rangle \subset \mathbb{Z}^n$. We have $\Lambda^{**} = \Lambda$.

A matrix $\mathbf{B}$ can be Gram-Schmidt orthogonalized in $\widetilde{\mathbf{B}}$, and its norm $||\mathbf{B}||$ is the maximum of the norm of its columns. Gram-Schmidt orthogonalization of a non-singular square matrix $\mathbf{B}$ is the unique decomposition as $\mathbf{B} = \mathbf{L} \cdot \widetilde{\mathbf{B}}^*$, where $\mathbf{L}$ is a lower triangular matrix with unit diagonal and $\widetilde{\mathbf{B}}^*$ consists of mutually orthogonal rows. For each $i \in [1, n]$, we call π_i the orthogonal projection over $\text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})^\perp$. In particular, one has $\pi_i(\mathbf{b}_i) = \mathbf{b}_i^*$, which is the ith row of $\widetilde{\mathbf{B}}$.

Definition 2.3. *The search problem SVP (Shortest Vector Problem) is defined as follows: The input is a basis $\mathbf{B}$ for a lattice $\Lambda \subset \mathbb{Z}^n$. The goal is to output a vector $\mathbf{y} \in \Lambda$ satisfying $\|\mathbf{y}\| = \lambda_1(\Lambda)$.*

Definition 2.4. *For any approximation parameter $\gamma = \gamma(n) \geq 1$, the search problem γ-CVP (Closest Vector Problem) is defined as follows: The input is a basis $\mathbf{B}$ for a lattice $\Lambda \subset \mathbb{Z}^n$ and a point $\mathbf{x}$. The goal is to output a vector $\mathbf{y} \in \Lambda$ satisfying $\|\mathbf{x} - \mathbf{y}\| \leq \gamma \cdot d(\Lambda, \mathbf{x})$.*

Definition 2.5. *The search problem r-Enum (Enumeration Problem) is defined as follows: The input is a basis $\mathbf{B}$ for a lattice $\Lambda \subset \mathbb{Z}^n$, a point $\mathbf{x}$ and a radius r. The goal is to output all vectors $\mathbf{y} \in \Lambda$ satisfying $\|\mathbf{x} - \mathbf{y}\| \leq r \cdot \lambda_1(\Lambda)$.*

If $r \geq 1$, then the size of output might be exponential [39] but is at most $2(\lceil 2r \rceil)^n$. For simplicity, we assume that the size of the input is polynomial in the dimension.

2.2 Discrete Gaussian distribution

We define $\rho_s(\mathbf{x}) = \exp(-\pi||\mathbf{x}||^2/s^2)$ and $\rho_s(S) = \sum_{\mathbf{x}\in S} \rho_s(\mathbf{x})$ (and similarly for other functions). The discrete gaussian distribution $D_{E,s}$ over a set E and of parameter s is such that the probability $D_{E,s}(\mathbf{x})$ of drawing $\mathbf{x} \in E$ is equal to $\rho_s(\mathbf{x})/\rho_s(E)$. For simplicity, D_E denotes the distribution $D_{E,1}$. We define $f_s(\mathbf{x}) = \frac{\rho_s(\Lambda+\mathbf{x})}{\rho_s(\Lambda)}$. Using a Poisson summation on both term, it is immediate to prove that

$$f_s(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in\Lambda^*} \cos(2\pi\langle\mathbf{x},\mathbf{y}\rangle)\rho_{1/s}(\mathbf{y})}{\sum_{\mathbf{y}\in\Lambda^*} \rho_{1/s}(\mathbf{y})} \leq 1.$$

Definition 2.6. *The smoothing parameter η_ϵ of the lattice Λ is the smallest s such that $\rho_{1/s}(\Lambda^*) = 1 + \epsilon$.*

The following lemma was proved in [7] and for completeness we recall its proof in appendix.

Lemma 2.7. *For a lattice Λ, $\mathbf{c} \in \mathbb{R}^n$, and $t \geq 1$,*

$$\frac{\rho((\Lambda+\mathbf{c}) \setminus \mathcal{B}(\mathbf{0}, t\sqrt{\frac{n}{2\pi}}))}{\rho(\Lambda)} \leq \exp\big(-n(t^2 - 2\ln t - 1)/2\big) \leq \exp\big(-n(t-1)^2/2\big).$$

The following lemmata were proved in [24, 11].

Lemma 2.8 ([**11**])**.** *For $\epsilon = 2^{-\Omega(n)}$, we have $\eta_\epsilon = \Theta(\sqrt{\log(1/\epsilon)}/\lambda_1(\Lambda^*))$.*

Proof. We have $\rho_{1/s}(\Lambda^*) \geq 1 + 2\exp(-\pi s^2\lambda_1(\Lambda^*)^2)$, so that $\eta_\epsilon \geq \sqrt{\ln(2/\epsilon)/\pi}/\lambda_1(\Lambda^*)$. The previous lemma gives for $t = 1 + \sqrt{2\ln((1+\epsilon)/\epsilon)/n}$ that if $\lambda_1(\Lambda^*) \geq t\sqrt{n/2\pi}$, then $\rho(\Lambda^*) \geq 1+\epsilon$. By scaling the lattice, we can see that $\eta_\epsilon \leq t\sqrt{n/2/\pi}/\lambda_1(\Lambda^*) = \frac{\sqrt{\ln((1+\epsilon)/\epsilon)/\pi} + \sqrt{n/2/\pi}}{\lambda_1(\Lambda^*)}$. The result follows from these two bounds. □

Lemma 2.9 ([**11**])**.** *For $\epsilon \leq 1$ and $t > 1$, we have $\eta_{\epsilon^{t^2}} \leq t\eta_\epsilon$.*

Proof. Since $\epsilon \leq 1$,

$$\sum_{\mathbf{x}\in\Lambda^*\setminus\{\mathbf{0}\}} \exp(-\pi\eta_\epsilon^2||\mathbf{x}||^2)^{t^2} < \left(\sum_{\mathbf{x}\in\Lambda^*\setminus\{\mathbf{0}\}} \exp(-\pi\eta_\epsilon^2||\mathbf{x}||^2)\right)^{t^2} = \epsilon^{t^2}.$$

□

Lemma 2.10 ([**24, Theorem 3.3**])**.** *Let Λ be a lattice, $\epsilon < 1/4$, $\mathbf{z} \in \mathbb{Z}^m$ and non zero, $s \geq \sqrt{2}||\mathbf{z}||_\infty\eta_\epsilon$. Let $\mathbf{y}_i$ be independent vectors with distributions $D_{\Lambda+\mathbf{c}_i,s}$. Then the distribution of $\sum_{i=0}^{m-1} z_i\mathbf{y}_i$ is close to $D_{\gcd(\mathbf{z})\Lambda+\sum_{i=0}^{m-1}\mathbf{c}_i,||\mathbf{z}||s}$ with KL-divergence of $2\epsilon^2$.*

Proof. Micciancio and Peikert proved in [24] that we can apply Theorem 2.1 with $\delta = \epsilon$. □

Lemma 2.11 ([**2**])**.** *There is an algorithm that takes as input a lattice $\Lambda \subset \mathbb{R}^n$ and a parameter $s \geq \sqrt{2}\eta_{1/2}(\Lambda)$ and outputs $2^{n/2}$ i.i.d. samples from a distribution whose statistical distance to $D_{\Lambda,s}$ is at most $2^{-\Omega(n)}$ using $2^{n/2+o(n)}$ time and space.*

2.3 Lattice algorithms

Lemma 2.12 ([**10, Lemma 2.3**]). *Given a basis* $\mathbf{A}$ *of a lattice* Λ *and a parameter* $s \geq ||\widetilde{\mathbf{A}}||\sqrt{\ln(2n+4)/\pi}$, *the* SAMPLE *algorithm, samples in polynomial time a point according to the distribution* $D_{\Lambda,s}$.

The algorithm MAKEBASIS corresponds to a slight modification of [23].

Lemma 2.13 ([**23, lemma 7.1**]). *Given a basis of a lattice, the* MAKEBASIS *algorithm searches for* n *linearly independent lattice points amongst the first* n^2 *points of its input and outputs in polynomial time a basis* $\mathbf{A}$ *of the lattice such that* $||\widetilde{\mathbf{A}}||$ *is at most the maximum norm of the input vectors.*

Lemma 2.14. *If it is possible to solve* r*-Enum for* $r \geq 1$, *then it is possible to solve SVP and* $\sqrt{1+1/(r^2-1)}$*-CVP.*

Proof. Enumerating in a ball centered at the origin and of radius λ_1 is clearly enough to solve SVP. The second statement is immediate from [11, theorem 6.1]. □

3 Discrete Gaussian Sampling (DGS) Algorithm

Micciancio and Peikert propose in [24] a new Discrete Gaussian Sampling using a SIS solver. We describe an exponential-time algorithm for this task in time $q^{\mathcal{O}(n)}$ and memory $q^{\mathcal{O}(n/q^2)}$ for some integer $2 \leq q \leq n^{1/5}$ by instantiating the SIS solver with a subexponential-time algorithm similar to Wagner's algorithm [36]. The main idea consists in sampling random lattice vectors, and reducing the length of lattice vectors by iteratively finding smaller and smaller lattice basis. In comparison, the DGS algorithm of Dadush, Regev and Davidowitz-Stephens runs in $2^{n/2+o(n)}$ time and space and outputs $2^{n/2}$ samples.

We use the Micciancio and Peikert lemma 2.10 to estimate the distribution of a linear combination of independent lattice vectors generated by some Gaussian distributions with width larger than the smoothing parameter of the lattice. In the SAMPLING algorithm, we use recursively this lemma to combine lattice vectors and reduce their length.

The next lemma and theorem show that in the SAMPLING algorithm, we can reduce the width of the Gaussian distribution by using the GBA algorithm (Generalized Birthday Algorithm), a.k.a. Wagner algorithm. The input of the GBA algorithm is a list $\mathcal{L}_0$ of

$m = 2^{wr}q^{n/w}$ pairs $(\mathbf{x}_i, \mathbf{a}_i)_{0 \leq i \leq m-1}$ where $\mathbf{x}_i$ is sampled as a Gaussian vector in the lattice generated by $\mathbf{A}$ with width s and $\mathbf{a}_i = \mathbf{A}^{-1}\mathbf{x}_i$. The aim of this algorithm is to find a vector $\mathbf{z} = (z_0, \ldots, z_{m-1})$ so that the second component of $\sum_{i=0}^{m-1} z_i\mathbf{a}_i$ cancels and $||\mathbf{z}||$ as small as possible.

Lemma 3.1. *In* SAMPLING, *if* $s \geq \sqrt{2}\eta_\epsilon q(q/2^{w/2})^{r-1}$, *then* $|L_r| \geq q^{n/w}$ *and the statistical distance between the elements of* L_r *and* $D_{\Lambda,s}$ *is less than* $r\epsilon\sqrt{m}$, *where* $m = 2^{wr}q^{n/w}$.

Algorithm 1 Generalized Birthday Algorithm

function GBA($\mathcal{L}_0$)
  $w = \lfloor \log(q^2 - 1) \rfloor$
  $\ell = n/w$ ▷ Assume this is an integer
  for $i = 0$ to $w - 1$ **do**
    Empty t
    for all $(\mathbf{x}, \mathbf{a}) \in \mathcal{L}_i$ **do**
      $\mathbf{r} = (\mathbf{a}_{i\ell}, \ldots, \mathbf{a}_{i\ell+\ell-1}) \mod q$
      if $t[\mathbf{r}] = \varnothing$ **then**
        $t[\mathbf{r}] \leftarrow (\mathbf{x}, \mathbf{a})$
      else
        $\mathcal{L}_{i+1} \leftarrow ((\mathbf{x}, \mathbf{a}) - t[\mathbf{r}]) :: \mathcal{L}_{i+1}$
        $t[\mathbf{r}] \leftarrow \varnothing$
      end if
    end for
  end for
  for all $(\mathbf{x}, \mathbf{a}) \in \mathcal{L}_w$ **do**
    $\mathcal{L}_{out} \leftarrow (\mathbf{x}/q, \mathbf{a}/q) :: \mathcal{L}_{out}$
  end for
  return $\mathcal{L}_{out}$
end function

Algorithm 2 Gaussian sampling

function SAMPLING($\mathbf{A}$,σ)
  $w \leftarrow \lfloor \log(q^2 - 1) \rfloor$
  $r \leftarrow \lceil \log(C\sqrt{n \log(n)}) / \log(q/2^{w/2}) \rceil$
  $m \leftarrow 2^{wr+1} q^{n/w}$
  repeat
    $s \leftarrow \max(||\widetilde{\mathbf{A}}|| C\sqrt{\log(n)}, \sigma(q/2^{w/2})^r)$
    $L_0 \leftarrow \varnothing$
    for $i = 0$ to $m - 1$ **do**
      $\mathbf{x} \leftarrow$ SAMPLE($\mathbf{A}, s$)
      $L_0 \leftarrow (\mathbf{x}, \mathbf{A}^{-1}\mathbf{x}) :: L_0$
    end for
    for $j = 0$ to $r - 1$ **do**
      $L_{j+1} \leftarrow \mathrm{GBA}(L_j)$
    end for
    $\mathbf{A} \leftarrow$ MAKEBASIS($\mathbf{A}, L_r$)
  until $s = \sigma(q/2^{w/2})^r$
  return L_r
end function

Proof. We can note that in the GBA algorithm, we have

$$|\mathcal{L}_i| \geq (|\mathcal{L}_{i-1}| - q^{n/w})/2.$$

Then, by induction, we can show that $|\mathcal{L}_i| + q^{n/w} \geq (|\mathcal{L}_0| + q^{n/w})/2^i$. Thus, we get that in the SAMPLING procedure, $|L_r| \geq (m + q^{n/w})/2^{wr} > q^{n/w}$ since $|L_0| = m = 2^{wr+1}q^{n/w}$.

We can easily check that if GBA outputs $(\mathbf{x}, \mathbf{a})$, then $\mathbf{x} = \mathbf{A}\mathbf{a}$ and since GBA depends only on the input coefficient vectors modulo q, we can apply lemma 2.10 to show that if the input distribution of $\mathbf{x}$ is $D_{\Lambda,s}$, the output distribution has a KL-divergence of $2\epsilon^2$ with $D_{\Lambda,s2^{w/2}/q}$ if $s \geq \sqrt{2}q\eta_\epsilon$. By induction, we can then show using Theorem 2.2 that the statistical distance between L_i and $D_{\Lambda,s(2^{w/2}/q)^i}$ is at most $i\sqrt{m}\epsilon$. □

Theorem 3.2. *Assuming $\sigma \geq \sqrt{2}2^{w/2}\eta_\epsilon$ for $\epsilon < 1/10$,* SAMPLING *returns in time $n^{\mathcal{O}(1)}m$ more than $q^{n/w}$ elements such that the statistical distance between their distribution and $D_{\Lambda,\sigma}$ is less than $r\epsilon\sqrt{m}$ with probability of failure smaller than $rn\epsilon\sqrt{m} + 2^{-\Omega(n)}$.*

Proof. At each reduction step of GBA, the size of the list is divided by at least two, so that we can compute L_r from L_0 in time $\mathcal{O}(mn)$. Then, we can set C to 0.68, so that SAMPLE takes polynomial time.

If the elements of L_r were sampled as $D_{\Lambda,s}$ with $s \geq \sqrt{2}\eta_\epsilon$, then [32, corollary 3.16] shows that MAKEBASIS would fail with probability at most $2^{-\Omega(n)}$. Also, using Theorem 2.7, the norms of the first n^2 lattice points of L_r would be at most $s\sqrt{n/\pi}$, except with probability at most $2^{-\Omega(n)}$.

We can deduce that each iteration of SAMPLING in this idealized model divides $||\widetilde{\mathbf{A}}||$ by at least $(q/2^{w/2})^r/(C\sqrt{n\log n/\pi}) \geq 2$. Therefore, there are only a polynomial number of iterations before $s = \sigma(q/2^{w/2})^r$. We conclude with the previous lemma. □

Corollary 3.3. *For any $\sigma \geq \sqrt{2}2^{w/2}\eta_\epsilon/k$, $q = o(\sqrt{n}/\log n)$ and positive integer k, it is possible to sample $q^{n/w}$ elements from $D_{\Lambda,\sigma}$ in time $(k + o(k))^n q^{n/w}$ with statistical distance and probability of failure bounded by $\epsilon 2^{o(n)}q^{n/w}$.*

Proof. We filter the output of SAMPLING: samples in $k\Lambda$ are divided by k, others are discarded. The probability for $\mathbf{x}$ sampled according to $D_{\Lambda,\sigma}$ to be in the coset $\mathbf{x} + k\Lambda$ is proportional to $f(\mathbf{x})$ which is maximized in $\mathbf{0}$, so that $\Pr[\mathbf{x} \in k\Lambda] \geq k^{-n}$.

As $\log(q/2^{w/2}) \geq \log(q/\sqrt{q^2-1}) = -\log(1-1/q^2)/2 \geq 1/(3q^2)$, we have $r = o(n/\log n)$ since $r \leq \log(C\sqrt{n\log(n)/\pi})/\log(q/2^{w/2}) < 3q^2\log(C\sqrt{n\log(n)/\pi})$ and $q = o(\sqrt{n}/\log n)$. Hence, $m = 2^{o(n)}q^{n/w}$ and the result follows by repeating SAMPLE k^n times. □

If we replace GBA by exhaustive search (or more efficient algorithms [12, 37]) where vectors from the previous solutions are removed, we have the following result.

Theorem 3.4. *For any $\sigma \geq \sqrt{2}q\eta_\epsilon/k$, $q = o((n/\log(n))^{1/4})$ and positive integer k, it is possible to sample $q^{n/q^2}/k^n$ elements from $D_{\Lambda,\sigma}$ in time q^{5n}, with memory q^{5n/q^2} with statistical distance and probability of failure bounded by $\epsilon q^{5n/q^2}$.*

Algorithm 3 Exhaustive search

function ES($\mathcal{L}$)
  Let B be the $n \times k$ matrix of the concatenation of the $\mathbf{a}$ for all $(\mathbf{x}, \mathbf{a}) \in \mathcal{L}$
  $w \leftarrow q^2 - 1$
  while $k \geq q^{2+n/\lfloor w/2 \rfloor}$ **do**
    for all $y \in \{-1, 0, 1\}^k$, $0 < ||y|| < q$ **do**
      if $By = 0 \bmod q$ **then**
        $\mathcal{L}_{out} \leftarrow (\sum_i \mathbf{x_i} y_i/q, Ay/q) :: \mathcal{L}_{out}$
        Remove $(\mathbf{x}_i, \mathbf{a}_i)$ from $\mathcal{L}$ and $\mathbf{a}_i$ from B for all i with $y_i \neq 0$
      end if
    end for
  end while
  return $\mathcal{L}_{out}$
end function

Proof. By the pigeonhole principle, ES will find a new output if $\binom{k}{\lfloor w/2 \rfloor} \geq q^n + 1$, so that it terminates.

One may then redo the previous proofs with $r = \lceil \log(C\sqrt{n \log n})/\log(q^2/w) \rceil = o(\sqrt{n \log(n)})$ and $m = 2w^r q^{n/\lfloor w/2 \rfloor}$. Since $\lfloor (q^2-1)/2 \rfloor \geq q^2/4$, and $w^r = 2^{o(\sqrt{n \log^3(n)})}$ while $q^{n/\lfloor w/2 \rfloor} = 2^{\Omega(\sqrt{n \log^3(n)})}$, we have $m \leq q^{5n/q^2}$ for a sufficiently large n. Hence, the running time is bounded by $n^{\mathcal{O}(1)}(2m)^w \leq q^{5n}$. □

For example, with $q = 3$, we can consider $w = 8$, hope that lists of size $q^{n/(w-1)+o(n)} = \mathcal{O}(2^{0.226n})$ are sufficiently large, and solve the SIS problem in time $q^{5n/8+o(n)} = \mathcal{O}(2^{1.333n})$ using the algorithm of [12]. Using a tower of lattices as in [2, Section 5], it might be possible to use $w = q^2$ while the SIS problem is over a space a bit larger than q^n. This shows that the algorithm may be more practical that it seems to be.

Corollary 3.5. *For any $\sigma = \Omega(\sqrt{n/q}/\lambda_1(\Lambda^*))$, $\epsilon = 2^{-\Omega(n)}$ and $q \leq n^{1/5}$, it is possible to sample from $D_{\Lambda,\sigma}$ with statistical distance ϵ, time $q^{\mathcal{O}(n)}$ and memory $q^{\mathcal{O}(n/q^2)}$.*

Proof. Theorem 2.7 proves that, with $\epsilon' = \epsilon 2^{-2n} k^{-n}$, $\eta_{\epsilon'} = \mathcal{O}(\sqrt{nk}/\lambda_1(\Lambda^*))$. We then apply the previous theorem with $k = \mathcal{O}(q)$ sufficiently large and ϵ', and repeat the corresponding algorithm nk^n times. □

If we push the previous idea to its limit, we can even use polynomial memory using Floyd's cycle finding algorithm to replace ES.

Theorem 3.6. *Let $\mathcal{D}$ be a distribution over a set E of size N. Let $g : E \to E$ such that all $g(x)$ are sampled independently with the distribution $\mathcal{D}$. Then, using $\mathcal{O}(\sqrt{nN})$ calls to g, we can find $x \neq y$ such that $g(x) = g(y)$ with polynomial memory, except with probability $2^{-\Omega(n)} + \mathcal{O}(\sqrt[4]{n/N})$.*

Proof. The algorithm samples uniformly $u_0 \in E$. Then, we use Floyd's cycle finding algorithm with the sequence defined by $u_{i+1} = g(u_i)$. Let $K = 2\lceil\sqrt{nN}\rceil$. If the sequence does not cycle in its first K elements, or if u_0 is in cycle, the algorithm fails. Else, we can deduce from the size of the cycle x and y in K calls to g.
Let $p = \sum_{i \leq K/2} \mathcal{D}(u_i)$. Then, $\Pr[\mathcal{D}(u_i) \leq 1/(3N)] \leq 1/(3\sum_{j<i} \mathcal{D}(u_j))$ for any $i \leq K/2$ and these events are independents. Therefore, either $p \geq \frac{2}{3}$, or, using the Hoeffding inequality, at most $2K/3 - 1$ events occured except with probability $2^{-\Omega(n)}$. Thus, we have $p \geq \min(\sqrt{n/N}/9, \frac{2}{3})$.

The probability that there is no collision amongst the first K elements is therefore at most $(1-p)^{K/2} \leq \exp(pK/2)$. If $n \geq N$, then the theorem is vacuous, else $\exp(pK/2) = 2^{-\Omega(n)}$. The probability that there exists $1 \leq i \leq K$ such that $u_0 = u_i$ is at most $K\mathcal{D}(u_0)$. We have $\Pr[\mathcal{D}(u_0) \geq 1/\sqrt{KN}] \leq \sqrt{K/N}$ and therefore, the probability that u_0 is in the cycle is at most $2\sqrt{K/N} = \mathcal{O}(\sqrt[4]{n/N})$. □

Corollary 3.7. *For any $\sigma \geq n^{1/2-o(1)}/\lambda_1(\Lambda^*)$ and $\epsilon = n^{-\Omega(n)}$, given a random oracle, it is possible to sample from $D_{\Lambda,\sigma}$ with statistical distance ϵ, time $n^{n/4+o(n)}$ and polynomial memory.*

Proof. Let $g(x) = \mathbf{A}^{-1}\textsc{Sample}_{H(x)}(\mathbf{A}, s) \bmod q$ where the random bits used in Sample come from $H(x)$ and H is a function from $(\mathbb{Z}/q\mathbb{Z})^n$ to the set of bit strings of suitable polynomial length, sampled uniformly. Using the previous theorem, we can find a collision in g in $\mathcal{O}(\sqrt{n}q^{n/2})$ calls to g, except with negligible probability which reveals $\mathbf{x}$ and $\mathbf{y}$ sampled independently under $D_{q\Lambda+\mathbf{c},s}$ for some $\mathbf{c}$. Then, using Theorem 2.10, the distribution of $(\mathbf{x}-\mathbf{y})/q$ is close to $D_{\Lambda,\sqrt{2}s/q}$. Setting $q = \lceil C\sqrt{2n\log n}\rceil$, we may now proceed just like in Sampling. Finally, choosing some $k = n^{o(1)}$ to filter the samples gives the result. □

In practice, we would use any sufficiently hard pseudo-random function as a random oracle. We can even build a pseudo-random function assuming worst-case hardness of lattice problems with a much higher dimension [8].

4 Enumeration algorithms

In this section, we first describe the Enumerate algorithm, which allows to enumerate all lattice vectors within a ball of radius about $\lambda_1(\Lambda)$ and then, we apply it to solve the SVP problem in time $3^{n+o(n)}$. The time-memory trade-off algorithms are simplified version of it, with a different sampler.
Let L^* be a list with N independent samples from D_{Λ^*} and $f_{L^*}(\mathbf{x}) = \frac{1}{N}\sum_{i=0}^{N-1} \cos(2\pi\langle \mathbf{w}_i, \mathbf{x}\rangle)$, $\rho(\Lambda) = 1+\epsilon$ with $\epsilon \leq 2^{-n}$, $s_\epsilon = \sqrt{\ln(1/\epsilon)/\pi} \leq n$ and select some $N = \mathcal{O}(n\log(1/\epsilon)/\sqrt{\epsilon})$.

The following theorem was proven in [11] (s_ϵ was taken slightly larger than in our definition, but the theorem is still correct) :

Theorem 4.1 (**[11, Proposition 3.2]**). *Except with probability $2^{-\Omega(n)}$,*

$$\left|\left| \frac{\nabla f_{L^*}(\mathbf{x})}{2\pi f_{L^*}(\mathbf{x})} + \mathbf{x} \right|\right| \leq \epsilon^{(1-2\delta(\mathbf{x}))/4}||x||$$

holds simultaneously for all $\mathbf{x} \in \mathbb{R}^n$ *with* $\delta(\mathbf{x}) = \max(1/8, ||\mathbf{x}||/s_\epsilon)$ *and* $||\mathbf{x}|| \leq (1/2 - 2/(\pi s_\epsilon^2))s_\epsilon$.

Combined with rounding, we can deduce the following theorem :

Theorem 4.2. *For* $\epsilon \leq 2^{-n}$*, except with probability* $2^{-\Omega(n)}$*, for all* $\mathbf{x}$ *such that* $\mathrm{dist}(\mathbf{x}, \Lambda) \leq (1/2 - 3\log(n)/n)s_\epsilon$*,*

$$\textsc{Round}\left(L^*, \frac{\nabla f_{L^*}(\mathbf{x})}{2\pi f_{L^*}(\mathbf{x})} + \mathbf{x}\right)$$

is the closest lattice point to $\mathbf{x}$*, where* Round *takes polynomial time.*

Proof. First of all, the Round algorithm selects n linearly independent vectors within L^*. Then, using algorithm of lemma 2.13, one can build a matrix $\mathbf{A}$, such that $||\mathbf{A}|| \leq \sqrt{n}/2$, except with probability $2^{-\Omega(n)}$. Finally, we round according to the lattice the input point: $\textsc{Round}(L^*, \mathbf{t}) = \mathbf{A}^{-t}\lfloor \mathbf{A}^t\mathbf{t} \rceil$.

One can check that when $\mathbf{x}$ is shifted by a lattice point, the result is shifted by this lattice point ; so that we only have to prove the result when $\mathbf{0}$ is the closest lattice point. If $||\mathbf{x}|| \leq (1/2 - 5\log(n)/n)s_\epsilon$, we have

$$\left|\left|\frac{\nabla f_{L^*}(\mathbf{x})}{2\pi f_{L^*}(\mathbf{x})} + \mathbf{x}\right|\right| \leq \epsilon^{1/4 - \delta(\mathbf{x})/2}||\mathbf{x}||.$$

If $||\mathbf{x}|| \leq s_\epsilon/8$, the right hand side is smaller than $\epsilon^{3/16}n \leq n^{-1/2}$ as soon as $n \geq 8$. Else, it is smaller than $\epsilon^{3\log(n)/(2n)}n \leq n^{-1/2}$. As a result, using the Cauchy-Schwarz inequality, $\left\lfloor \mathbf{A}^t\left(\frac{\nabla f_{L^*}(\mathbf{x})}{2\pi f_{L^*}(\mathbf{x})} + \mathbf{x}\right)\right\rceil = \mathbf{0}$. □

The Enumerate algorithm allows to find all lattice points close to $\mathbf{x}$ within a distance bounded by $(1/2 - 3\log(n)/n)s_\epsilon$ given the succinct representation of the dual lattice and the function f_{L^*}. The main idea consists in performing a gradient ascent from $\mathbf{x}$ as in [21, 11].

Lemma 4.3. *Given* $\mathbf{x}$*,* Enumerate *returns all* $\mathbf{As}$ *such that* $||\mathbf{As} - \mathbf{x}|| \leq p(1/2 - 3\log n/n)s_\epsilon$ *in time* $\mathcal{O}(n^2p^n + n^3/\sqrt{\epsilon})$.

Proof. Let $\mathbf{z}$ be sampled according to D_{Λ^*} and $\mathbf{v} = \mathbf{As} - \mathbf{x}$. Therefore, we can write $\langle \mathbf{z}, \mathbf{x}\rangle = \langle \mathbf{z}, \mathbf{As}\rangle - \langle \mathbf{z}, \mathbf{v}\rangle = \langle \mathbf{A}^t\mathbf{z}, \mathbf{s}\rangle - \langle \mathbf{z}, \mathbf{v}\rangle$ and since $\mathbf{z} \in \Lambda^*$, $\mathbf{A}^t\mathbf{z} \in \mathbb{Z}^n$.
Since $\mathbb{E}[\exp(2i\pi(\langle \mathbf{A}^t\mathbf{z}, \mathbf{s}\rangle - \langle \mathbf{z}, \mathbf{x}\rangle)/p)] = \mathbb{E}[\exp(2i\pi\langle \mathbf{z}, \mathbf{v}/p\rangle)]$, we get $\Re(f[\mathbf{s} \bmod p]) = f_{L^*}(\mathbf{v}/p)$.
Moreover, $\Re(grad[j][\mathbf{s} \bmod p]) = (2\pi\nabla f_{L^*}(\mathbf{v}/p))_j$, and therefore, if we have $||\mathbf{As} - \mathbf{x}|| \leq p(1/2 - 3\log n/n)s_\epsilon$, $p\textsc{Round}(L^*, \mathbf{y} + (\mathbf{x} - \mathbf{A}(\mathbf{s} \bmod p))/p) + \mathbf{A}(\mathbf{s} \bmod p) = \mathbf{As}$. □

Theorem 4.4. *Let* $t \geq 1$ *such that* $a = (t^2 - 2\ln t - 1)/4$*. Assume we have an oracle which samples from* $D_{\Lambda^*,\sigma}$ *for any* $\sigma \geq \eta_{\exp(-2an)}$*. Then, it is possible to enumerate all lattice points within a ball of radius* $(1 + o(1))p\sqrt{a}/t\lambda_1$ *in time* $\exp(an + o(n))$ *and negligible failure probability.*

Algorithm 4 Enumerate

function ENUMERATE($\mathbf{x}$,L^*)
  for all $\mathbf{z} \in L^*$ **do**
    $\mathbf{a} \leftarrow \mathbf{A}^{-1}\mathbf{z} \bmod p$
    $f[\mathbf{a}] \leftarrow f[\mathbf{a}] + \exp(2i\pi\langle \mathbf{z}, \mathbf{x}\rangle/p)$
    for $j = 0$ to $n-1$ **do**
      $g[j][\mathbf{a}] \leftarrow g[j][\mathbf{a}] + x_j \exp(2i\pi\langle \mathbf{z}, \mathbf{x}\rangle/p)$
    end for
  end for
  $f \leftarrow$ FASTFOURIERTRANSFORM(f)
  for $j = 0$ to $n-1$ **do**
    $grad[j] \leftarrow$ FASTFOURIERTRANSFORM($g[j]$)
  end for
  $L_{out} \leftarrow \varnothing$
  for all $\mathbf{s} \in (\mathbb{Z}/p\mathbb{Z})^n$ **do**
    for $j = 0$ to $n-1$ **do**
      $\mathbf{y}_j \leftarrow \Re(grad[j][\mathbf{s}])/\Re(f[\mathbf{s}])$
    end for
    $L_{out} \leftarrow (p\text{ROUND}(L^*, \mathbf{y} + (\mathbf{x} - \mathbf{A}\mathbf{s})/p) + \mathbf{A}\mathbf{s}) :: L_{out}$
  end for
  return L_{out}
end function

Proof. Let $\epsilon = \exp(-2an)/2$. Using LLL [20], it is possible to find $\lambda \in [1, 2^{n/2}]\lambda_1(\Lambda)$. Because of Theorem 2.8, we can deduce $\eta \in [1, 2^n]\eta_\epsilon(\Lambda^*)$. We then run the previous algorithm by scaling the problem of a factor $\eta(1+1/n)^i$ for i integer from 0 to n^2. Let $i \leq n^2$ be the largest integer such that $\eta_\epsilon(\Lambda^*) \geq \eta(1+1/n)^i = \eta_{\epsilon'}(\Lambda^*)$. Because $\eta_\epsilon \leq (1+1/n)\eta_{\epsilon'}(\Lambda^*)$, we have using Theorem 2.9 $\epsilon \geq \epsilon'^{(1+1/n)^2}$. Therefore, $s_{\epsilon'} \geq s_\epsilon/(1+1/n)$. Using Theorem 2.7 and the previous lemma, it is possible to enumerate in a ball of radius

$$p(1/2 - 3\log(n)/n)\sqrt{2na/\pi}\eta_\epsilon/(1+1/n) \geq (1+o(1))p\sqrt{a}/t\lambda_1.$$

□

Corollary 4.5. *It is possible to solve* 1.156*-Enum in time* $3^{n+o(n)}$*. Also, for any* $r = \omega(1)$*, we can solve* r*-Enum in time* $(2r)^{n+o(n)}$*.*

Proof. Use the previous theorem with $p = 3$ and $t = 2.7194 > \sqrt{2}$, and the *exact* fast gaussian sampler of [2] recalled in Theorem 2.11. For the second claim, use $t = \sqrt{\min(r, n)} = \omega(1)$ so that $\sqrt{a}/t = 1/2 + o(1)$ and $s_\epsilon \leq n$, and we can then set $p = (2 + o(1))r$. □

Corollary 4.6. *For any* $q = o((n/\log n)^{1/4})$*,* $q \geq 2$*, it is possible to solve* r*-Enum in time* $(rq)^{\mathcal{O}(n)}$*, using a memory of size* $q^{\mathcal{O}(n/q^2)}$ *and with negligible probability of failure. Also, with a random oracle, we can solve* r*-Enum in time* $(rn^{1/4})^{n+o(n)}$ *and polynomial memory.*

Proof. Use $p = \mathcal{O}(r)$, $t = 2$, in Theorem 4.4 where the Fourier transforms are computed naively, with polynomial memory. Repeat the Fourier transforms p times, so that the failure probability is $p^{\mathcal{O}(n)}2^{-p\Omega(n)} = 2^{-\Omega(n)}$. The sampler is given by Theorem 3.5.

For the second claim, use the sampler of Theorem 3.7. □

To illustrate this tradeoff, we can remark that if $q = \log n$, we have a time complexity of $2^{\mathcal{O}(n \log\log n)}$ and memory complexity of $2^{\mathcal{O}\left(\frac{n \log\log n}{\log^2 n}\right)}$ which is more efficient than Kannan but uses much more memory.

For comparison, Kannan algorithm can solve r-Enum in time $\mathcal{O}(r)^n n^{n/(2\mathrm{e})+o(n)}$ or for $r \geq \sqrt{n}$, in time $\mathcal{O}(r)^n$[15].

References

[1] *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*. IEEE Computer Society, 2004.

[2] Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the shortest vector problem in 2^n time using discrete gaussian sampling: Extended abstract. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015*, pages 733–742, 2015.

[3] Divesh Aggarwal, Daniel Dadush, and Noah Stephens-Davidowitz. Solving the closest vector problem in 2^n time - the discrete gaussian strikes again! In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015*, pages 563–582, 2015.

[4] Dorit Aharonov and Oded Regev. Lattice problems in NP cap conp. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 362–371, 2004.

[5] Miklós Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 10–19, 1998.

[6] Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing*, pages 601–610, 2001.

[7] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, 1993.

[8] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *Advances in Cryptology–EUROCRYPT 2012*, pages 719–737. Springer, 2012.

[9] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*, pages 10–24, 2016.

[10] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Symposium on Theory of Computing Conference, STOC'13*, pages 575–584, 2013. `http://perso.ens-lyon.fr/damien.stehle/downloads/LWE.pdf`.

[11] Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. On the closest vector problem with a distance guarantee. In *Computational Complexity (CCC), 2014 IEEE 29th Conference on*, pages 98–109. IEEE, 2014. `http://arxiv.org/pdf/1409.8063`.

[12] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Efficient dissection of composite problems, with applications to cryptanalysis, knapsacks, and combinatorial search problems. In *Advances in Cryptology–CRYPTO 2012*, pages 719–740. 2012. `http://eprint.iacr.org/2012/217.pdf`.

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

[14] Nicolas Gama, Phong Q. Nguyen, and Oded Regev. Lattice enumeration using extreme pruning. In *Advances in Cryptology - EUROCRYPT 2010. Proceedings*, pages 257–278, 2010.

[15] Guillaume Hanrot and Damien Stehlé. Improved analysis of kannan's shortest lattice vector algorithm. In *Advances in Cryptology - CRYPTO 2007. Proceedings*, pages 170–186, 2007.

[16] Ishay Haviv and Oded Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, California, USA*, pages 469–477, 2007.

[17] Ishay Haviv and Oded Regev. On the lattice isomorphism problem. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 391–404, 2014.

[18] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, Boston, Massachusetts, USA*, pages 193–206, 1983.

[19] Subhash Khot. Hardness of approximating the shortest vector problem in lattices. In *45th Symposium on Foundations of Computer Science (FOCS 2004), Proceedings* [1], pages 126–135.

[20] Arjen Lenstra, Jr. Hendrik Lenstra, and Laszlo Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

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

[22] Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 92–98. IEEE Computer Society, 1998.

[23] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems: a cryptographic perspective*, volume 671. Springer Science & Business Media, 2002.

[24] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In *Advances in Cryptology - CRYPTO 2013. Proceedings, Part I*, pages 21–39, 2013.

[25] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, USA*, pages 351–358, 2010.

[26] Daniele Micciancio and Panagiotis Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, USA*, pages 1468–1480, 2010.

[27] Phong Q. Nguyen and Thomas Vidick. Sieve algorithms for the shortest vector problem are practical. *J. Mathematical Cryptology*, 2(2):181–207, 2008.

[28] M. Pohst. On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications. *SIGSAM Bull*, 15(1):37–44, 1981.

[29] Thomas Pöppelmann, Léo Ducas, and Tim Güneysu. Enhanced lattice-based signatures on reconfigurable hardware. *IACR Cryptology ePrint Archive*, 2014:254, 2014. `https://eprint.iacr.org/2014/254.pdf`.

[30] Xavier Pujol and Damien Stehlé. Rigorous and efficient short lattice vectors enumeration. In *Advances in Cryptology - ASIACRYPT 2008. Proceedings*, pages 390–405, 2008.

[31] Xavier Pujol and Damien Stehlé. Solving the shortest lattice vector problem in time $2^{2.465n}$. *IACR Cryptology ePrint Archive*, 2009:605, 2009.

[32] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009. `http://www.cims.nyu.edu/~regev/papers/qcrypto.pdf`.

[33] Oded Regev. The learning with errors problem (invited survey). In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, June 9-12, 2010*, pages 191–204. IEEE Computer Society, 2010.

[34] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987.

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

[36] David Wagner. A generalized birthday problem. In *Advances in Cryptology - CRYPTO 2002. Proceedings*, pages 288–303, 2002.

[37] Joshua R. Wang. Space-efficient randomized algorithms for K-SUM. In *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*, pages 810–829, 2014.

[38] Xiaoyun Wang, Mingjie Liu, Chengliang Tian, and Jingguo Bi. Improved Nguyen-Vidick heuristic sieve algorithm for shortest vector problem. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2011*, pages 1–9, 2011.

[39] A. D. Wyner. Capabilities of bounded discrepancy decoding. *Bell System Technical Journal*, 44(6):1061–1122, 1965.

[40] Feng Zhang, Yanbin Pan, and Gengran Hu. A three-level sieve algorithm for the shortest vector problem. In *Selected Areas in Cryptography - SAC 2013*, pages 29–47, 2013.

A Proof for lemma 2.7

Proof. For any $s \geq 1$, using Poisson summation :

$$\begin{aligned}\frac{\rho_s(\Lambda + \mathbf{c})}{\rho(\Lambda)} =& s^n \frac{\sum_{\mathbf{x}\in\Lambda^*} \rho_{1/s}(\mathbf{x}) \exp(2i\pi\langle \mathbf{x}, \mathbf{c}\rangle)}{\rho(\Lambda^*)} \\ &\leq s^n \frac{\sum_{\mathbf{x}\in\Lambda^*} \rho_{1/s}(\mathbf{x})}{\sum_{\mathbf{x}\in\Lambda^*} \rho(\mathbf{x})} \\ &\leq s^n.\end{aligned}$$

Then,

$$\begin{aligned}s^n \rho(\Lambda) \geq& \rho_s\left((\Lambda + \mathbf{c}) \setminus \mathcal{B}\left(\mathbf{0}, t\sqrt{\frac{n}{2\pi}}\right)\right) \\ &\geq \exp(t^2(1 - 1/s^2)n/2)\rho\left((\Lambda + \mathbf{c}) \setminus \mathcal{B}\left(\mathbf{0}, t\sqrt{\frac{n}{2\pi}}\right)\right).\end{aligned}$$

And therefore, using $s = t$:

$$\begin{aligned}\frac{\rho((\Lambda + \mathbf{c}) \setminus \mathcal{B}(\mathbf{0}, t\sqrt{\frac{n}{2\pi}}))}{\rho(\Lambda)} &\leq \exp(-n(t^2(1 - 1/s^2) - 2\ln s)/2) \\ &= \exp(-n(t^2 - 2\ln t - 1)/2) \\ &\leq \exp(-n(t - 1)^2/2),\end{aligned}$$

where the last inequality stems from $\ln t \leq t - 1$. □