A heuristic method to approximate the closest vector problem

MUSTAPHA ELHASSANI, ABDELHAKIM CHILLALI, ALI MOUHIB
Sidi Mohamed Ben Abdellah University
Polydisciplinary Faculty, Engineering sciences laboratory
Oujda Road, B.P. 1223 Taza
MOROCCO

Abstract: The closest vector problem, or CVP for short, is a fundamental lattice problem. The purpose of this challenge is to identify a lattice point in its ambient space that is closest to a given point. This is a provably hard problem to solve, as it is an NP-hard problem. It is considered to be more difficult than the shortest vector problem (SVP), in which the shortest nonzero lattice point is required.
There are three types of algorithms that can be used to solve CVP: Enumeration algorithms, Voronoi cell computation and seiving algorithms. Many algorithms for solving the relaxed variant, APPROX-CVP, have been proposed: The Babai nearest algorithm or the embedding technique.
In this work we will give a heuristic method to approximate the closest vector problem to a given vector using the embeding technique and the reduced centered law.

Key-Words: - Lattice, lattice based cryptography, worst case to average reduction, homomorphic encryption

Received: August 7, 2021. Revised: November 13, 2021. Accepted: December 20, 2021. Published: December 31, 2021.

1 Introduction

A lattice is defined informally as a regular point arrangement in a Euclidean space. Formally An n-Dimensional lattice $\mathcal{L}$ is any subset of $\mathbb{R}^n$ That is both:
1- An additive subgroup: $0 \in \mathcal{L}$ and $-\mathrm{x}, \mathrm{x}+\mathrm{y} \in \mathcal{L}$ for every $\mathrm{x}, \mathrm{y} \in \mathcal{L}$; and
2- Discrete: every $\mathrm{x} \in \mathcal{L}$ has a neighborhood in $\mathbb{R}^n$ in which x is the only lattice point.
It is a $\mathbb{Z}$ free module of finite type and can be defined as the $\mathbb{Z}$ -linear span of a set of linearly independent vectors $\{\mathbf{b}_1, ...\mathbf{b}_m\} \subset \mathbb{R}^n$, these vectors are known as a basis of the lattice let:

$$\mathcal{L}(\mathbf{b}_1, ...\mathbf{b}_m) := \{\sum_{i=1}^{i=m} x_i\mathbf{b}_i, x_1...x_m \in \mathbb{Z}\}$$

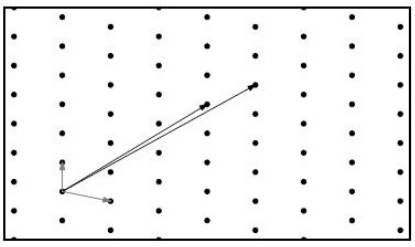

Figure 1: Two dimension lattice

Because the dimension (or rank) of the lattice matches the dimension of the vector subspace $span\mathcal{L}$ spanned by $\mathcal{L}$, all the bases have the same number of elements $dim(\mathcal{L})$. When $dim(\mathcal{L} \geq 2)$, there are an unlimited number of lattice bases (good bases and bad bases). All bases have the same Gramian determinant $det_{1\leq i\leq i\leq m} < \mathbf{b}_i, \mathbf{b}_j >$ since they are connected via a unimodular matrix (integral matrix of determinant ± 1). The volume $vol(\mathcal{L})$ (or determinant) of the lattice is by definition the square root of that Gramian determinant, thus corresponding to the m-dimensional volume of the parallelepiped spanned by the $\mathbf{b}'_i s$. In the important case of full-dimensional lattices where $dim(\mathbf{L}) = n$, the volume is equal to the absolute value of the determinant of any lattice basis (hence the name determinant). If the lattice is further an integer lattice, then the volume is also equal to the index $[\mathbb{Z}^n : \mathcal{L}]$ of $\mathcal{L}$ in $\mathbb{Z}^n$.
Since a lattice is discrete, it has a shortest nonzero vector: the Euclidean norm of such a vector is called the lattice first minimum, denoted by $\lambda_1(\mathcal{L})$ or $\|\mathcal{L}\|$.
More generally, for all $1 \leq i \leq dim(\mathcal{L})$, Minkowski's i-th minimum $\lambda_i(\mathcal{L})$ is defined as the minimum of $max_{1\leq j\leq i}\|\mathbf{v}_j\|$ over all i linearly independent lattice vectors $\mathbf{v}_1, ...\mathbf{v}_i \in \mathcal{L}$. There always exist linearly independent lattice vectors $\mathbf{v}_1, ...\mathbf{v}_d$ reaching the minima, that is $\|\mathbf{v}_i\| = \lambda_i(\mathcal{L}))$. However, surprisingly, as soon as $dim(\mathcal{L}) \geq 4$, such vectors do not necessarily form a lattice basis, and when $dim(\mathcal{L}) \geq 5$, there may not even exist a lattice basis reaching the min-

ima. This is one of the reasons why there exist several notions of basis reduction in high dimension, without any optimal one.

2 The importance of lattice-based cryptography is as follows

Lattice-based encryption is a type of cryptography that takes advantage of the supposed difficulty of lattice problems. Lattice cryptography has a lot of appealing features, which we'll go through presently.[1]
Security against quantum assaults that has been hypothesized:The Diffie- Hellman protocol and the RSA cryptosystem [2] rely on the conjectured hardness of integer factorization or the discrete logarithm problem in certain groups for most number-theoretic cryptography, to solve the discrete logarithm problem on elliptic curves defined over a finite field, some authors have studied elliptic curves defined on rings, see [14, 15]. Shor [3], provided excellent quantum algorithms for all of these problems, making number-theoretic systems insecure in the future when large-scale quantum computers become available. For the challenges commonly encountered in lattice cryptography, however, no efficient quantum techniques exist.
Simplicity, efficiency, and parallelism in algorithms: Lattice-based cryptosystems are typically algorithmically simple and highly parallelizable, relying mostly on linear operations on vectors and matrices modulo tiny integers. Furthermore, architectures based on algebraic lattices over certain rings (e.g., the NTRU cryptosystem [4]) can be extremely efficient, outperforming more traditional systems by a large margin in some circumstances.
Worst-case hardness provides strong security guarantees: Cryptography necessitates average-case intractability, or problems that are difficult to solve for random cases (taken from a defined probability distribution). This is in contrast to the worst-case idea of hardness commonly used in algorithm theory and NP-completeness, in which a problem is considered hard if there are only a few intractable examples. Problems that appear difficult in the worst-case scenario frequently turn out to be easier on average.
Ajtai [5] established a striking relationship between the worst and average cases for lattices in a seminal paper: he demonstrated that some problems are hard on the average (for cryptographically meaningful distributions) if some related lattice problems are hard in the worst case. Unless all cases of particular lattice problems are trivial to solve, one can develop cryptographic structures and establish that they are infeasible to crack using results like these.
Constructions of cryptographic objects that are both adaptable and powerful:Historically, cryptography was used primarily to send encrypted messages. However, during the last few decades, the field has evolved into a science with far broader and more varied goals, embracing practically every scenario involving communication or computation in the face of potentially malevolent conduct. For example, Rivest et al. proposed the powerful concept of fully homomorphic encryption (FHE), which allows an untrusted worker to execute arbitrary computations on encrypted data without learning anything about it. FHE remained an elusive aim for three decades, until Gentry [6] proposed the first candidate FHE architecture, which was based on lattices.
In the field of cryptanalysis: When a system is built on a linear problem or is easily linearizable, lattices are a powerful cryptanalysis tool: The cryptographic instance is transformed into a lattice instance, and cryptanalysis is predicated on the capability of locating a small lattice vector. This is how we can break several knapsack-style protocols as well as some RSA-style protocols.

we give a table which gives a comparison between lattice based cryptography and standard cryptography:

Table 1: Table of this comparison

Lattice based cryptography	Standard cryptography
Provably secure	Not always provable
Security based on a worst case problem	Security based on a average case problem
based on hardness of lattice problems	Based on hardness of factoring, discrete Log etc...
Not broken by quantum algorithms	Broken by quantum algorithms
Very simple computations	Require modular exponentiation

3 Complexity

P, for polynomial, and NP, for non-deterministic polynomial, are two notable complexity classes from computational complexity theory. All problems that can be solved by a Deterministic Turing Machine in a time that is bounded from above by a function that is polynomial in the length of its input are included in the class P. All problems that can be solved by a Non-Deterministic Turing Machine, with a time bound by a function polynomial in the length of the input, are classified as NP. The class P is a subclass of the class

NP.
The topic of whether these classes are indeed the same, or whether there are problems in NP that do not exist in P, is a key unresolved question in complexity theory. Informally, P refers to the category of simple problems, whereas NP may include some difficult problems that aren't in P.
Karp [7] developed polynomial-time Karp reductions, which allowed one problem to be reduced to another. Such reductions assume the existence of a subroutine that solves the other problem and can then be applied to the original one. Oracles are the name for these subroutines. Reducing a problem A to another problem B in polynomial-time, thus if B is in P, then A must be as well. Intuitively, this indicates that problem A cannot be more difficult to solve than problem B, because solving B automatically solves A. Karp also demonstrated that there was a subgroup of NP known as NP-complete, in which every problem in that class can be reduced to any other problem in NP. This means that proving that any of these NP-complete problems is genuinely in P automatically establishes $P = NP$.
The subject of whether P = NP is still open, but years of investigation have led to the conclusion that $P \subset NP$. NP-complete problems are regarded difficult to solve until it is proven that $P = NP$. As a result, proving that (mathematically) breaking a cryptosystem is equal to solving an NP-complete problem should provide a reasonable amount of security. Security proofs are examples of such proofs.

3.1 Relationships that exist between lattice problems (Reductions):

Reductions between problems can be used to compare the difficulty of problems. A transformation from the instances of problem A to the instances of problem B is called a reduction from problem A to problem B (A is reduced to B). This indicates that a solution for solving problem B instances can also be used to solve problem A instances. This implies that problem B cannot be easier than problem A or, alternatively, that problem B is at least as difficult as problem A. This is because a solution to problem B automatically leads to a solution to problem A, whereas the opposite is not true: a solution to problem A does not always lead to a solution to problem B.
The typical method for proving that a problem A is NP-hard (and thus unlikely to have a polynomial time solution) is to reduce some other NP-hard problem B to A:$(B \longrightarrow A)$.
In complexity theory, a problem is said to be difficult if it is difficult in the worst-case case, but, in cryptography, a problem is only considered difficult if it is difficult in the average case (i.e, for all but a negligible fraction of the instances).
Two key achievements in the study of lattices, notably from a computational standpoint, were the invention of the LLL lattice reduction method and Ajtai's discovery of a link between the worst-case and average-case hardness of certain lattice problems. The innovative aspect of Ajtai's discovery is that he demonstrates how to construct a cryptographic function that is as difficult to break in the average (e.g., over the random choices of the function instance) as it is to solve the worst case instance of a lattice problem. This achievement is unique to lattice theory at this moment, and it demonstrates that lattices are an excellent source of hardness for cryptographic applications.

4 Computational problems in lattices

4.1 Polynomial problems

We first present some easy problems. These problems are often solved with classical tools of linear algebra whose complexity is polynomial.

Problem 4.1. (base) Let $\mathcal{L}$ be a Lattice defined by a system of vectors that are not necessarily independent. Give a base of $\mathcal{L}$.

It suffices to calculate the Hermite normal form of the system given as input which is obtained in polynomial time from a Gaussian pivot type algorithm.

Problem 4.2. (membership) Let $\mathcal{L}$ be alattice given by a base $\mathbf{B}$ and a vector $\mathbf{v}$. Decide if $\mathbf{v}$ belongs to the $\mathcal{L}$ lattice.

It suffices to solve the system of equations according to $\mathbf{Bx} = \mathbf{v}$ and to check if the solution $\mathbf{x} = (x_1, ..., x_n)$ is indeed integer. To solve the system, we can use the Gaussian pivot method of cubic complexity in dimension.

Problem 4.3. (equivalence) Let $\mathbf{B}$ and $\mathbf{B}'$ be two bases. Decide if these two bases generate the same lattice $\mathcal{L}$.

It suffices to calculate the passage matrix $\mathbf{P}$ and to check if this one is indeed unimodular. In this section, we present the problems in the worst case as well as some problems in the average case to which these first can be reduced.

4.2 Worst-case lattice problems:

Many cryptosystems can be proved secure assuming the hardness of certain lattice problems in the

worst case. In the following, we present the most useful among them and we briefly outline their computational complexity.

SVP- Shortest vector problem:

The first successive minimum of the lattice $\mathcal{L}$is: $\lambda_1(\mathcal{L}) = Min_{\mathbf{v}\in\mathcal{L}^\star}\|\mathbf{v}\|$

Problem 4.4. (Shortest Vector Problem SVP):Given a Basis of $\mathcal{L}$ find $\mathbf{v} \in \mathcal{L}$ such that $\|\mathbf{v}\| = \lambda_1(\mathcal{L})$.

Because there are no efficient algorithms for solving SVP, computer scientists have turned to approximation versions of the problem. It's worth noting that in practice, approximation lattice problems are required to reduce the average case to the worst case. These cases are defined by an approximation factor $\gamma \geq 1$, which is normally determined by the lattice dimension n: $(\gamma = \gamma(n))$. This factor must be polynomial in n in order to be used in actual protocols, i.e. $(\gamma = poly(n))$.

Problem 4.5. (Approximate shortest vector problem, (SVP_γ)): Given a basis of $\mathcal{L}$ and an approximation factor $\gamma \geq 1$, find $\mathbf{v} \in \mathcal{L}$ such that $0 \leq \|\mathbf{v}\| \leq \gamma\lambda_1(\mathcal{L})$.

The LLL algorithm was the first to solve (SVP_γ), and it produced a $2^{O(n)}$ approximation with a $poly(n)$ running time. If we want to solve $SVP(i.e : \gamma = 1)$, we can use the LLL algorithm, which has a run time of $2^{O(n^2)}$. The quickest known algorithm for solving the exact SVP has a running time of 2^n.

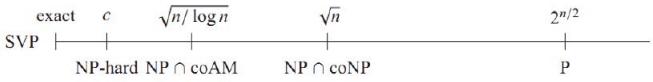

Figure 2: Complexity of SVP_γ

The goal of the shortest vector problem is to find an explicit vector problem. This implies that the shortest vector problem was framed as a search problem with the goal of obtaining something. There are also decisional issues, in which the goal is to figure out whether a statement is true in the setting of a particular problem instance. GapSVP is a decisional form of SVP that checks for the presence of a short vector. $Gap(SVP_\gamma)$ and $SIVP_\gamma$ are two hard lattice problems for $\gamma = poly(n)$, and solving either $Gap(SVP_\gamma)$ or $SIVP_\gamma$ appears to take $2^{O(n)}$ time and space.

Problem 4.6. Decisional shortest vector problem $(Gap(SVP_\gamma))$: Given a Basis of $\mathcal{L}$, where either $\lambda_1(\mathcal{L}) \leq 1$ or $\lambda_1(\mathcal{L}) > \gamma(n)$ determine which is the case.

Problem 4.7. (Approximate Shortest Independent Vectors Problem $(SIVP_\gamma)$): Given a Basis of $\mathcal{L}$, output n linearly independent lattice vectors $\mathbf{u}_i$ where $\|\mathbf{u}_i\| \leq \gamma(n).\lambda_n(\mathcal{L})$ for all i.

USVP is a promise variation of SVP in the sense that the second minimum is guaranteed to be substantially larger than the first minimum. In other words, every vector that is not parallel to the two shortest vectors of norms λ_1 has a greater norm than λ_1. As a consequence, every approximate shortest vector within this gap should be the shortest vector or a multiple of it.

Problem 4.8. (Unique SVP_γ,$uSVP_\gamma$): Let $\gamma \geq 1$. Given as input a lattice basis $\mathbf{B}$ such that $\lambda_2(\mathbf{B}) \geq \gamma.\lambda_1(\mathbf{B})$, the goal is to find a vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ of norm $\lambda_1(\mathbf{B})$.

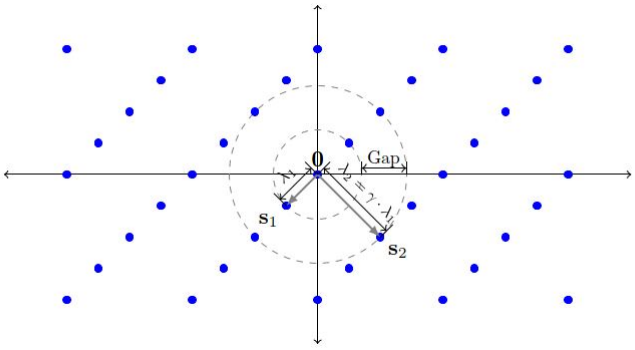

Figure 3: An example of USVP instance

Table 2: History of NP-hardness

Problem	Norm	Hardness	Reference
CVP_1	$\mathbf{L}_p$	NP-hard	[vEB81]
SVP_1	$\mathbf{L}_\infty$	NP-hard	[vEB81]
SVP_1	$\mathbf{L}_2$	NP-hard	[Ajt98]
$SVP_{\sqrt{2}}$	$\mathbf{L}_2$	NP-hard	[Mic98]
SVP_c	$\mathbf{L}_p$	NP-hard	[Kho04]
$SVP_{2^{(logn)^{\frac{2}{2-\epsilon}}}}$	$\mathbf{L}_p$	quasi-NP-hard	[Kho04]
$SVP_{2^{(logn)^{1-\epsilon}}}$	$\mathbf{L}_p$	quasi-NP-hard	[HR07]

CVP- Closest vector problem:

The Closest Vector Problem (CVP) is a lattice-based computer problem similar to SVP. CVP asks for the lattice point closest to the target point t, given a lattice $\mathcal{L}$) and a target point $\mathbf{t}$. In contrast to SVP, CVP can be described in terms of any norm, albeit the Euclidean norm is the most popular. A slightly relaxed variant of the issue (often employed in complexity theory) merely asks for the target's distance from the lattice, rather than the nearest lattice vector.

In many CVP applications, all that is required is to locate a lattice vector that is not too far away

from the objective, even if it is not necessarily the closest. For CVP, a γ approximation algorithm locates a lattice vector within γ times the ideal solution's distance. Babai and Kannan's best-known polynomial-time algorithms for solving CVP are based on lattice reduction and achieve approximation factors that (in the worst case) are essentially exponential in the lattice dimension. When the dimension of the lattice is small enough, heuristics approaches (e.g., the embedding technique) appear to find relatively good approximations to CVP in a reasonable period of time.
CVP is commonly recognized as a far more difficult problem than SVP, both in theory and in reality. Within any constant factor or even a slowly growing (sub-polynomial) function of dimension n, CVP is known to be NP-hard to solve. Goldreich, Micciancio, Safra, and Seifert [8] demonstrated that any algorithm for efficiently approximating CVP can also be used to efficiently approximate SVP with the same approximation factor and essentially the same computational effort, formalizing the intuition that CVP is not an easier (and possibly harder) problem than SVP.
CVP is the foundation of a number of cryptosystems (see lattice-based cryptography), in which the decryption process is basically equivalent to a CVP computation. These cryptosystems are predicated on the notion that any lattice can be represented in a variety of ways (for example, by different bases), and some of them may have superior geometric features than others, allowing them to be used as a decryption trapdoor [9]. However, some lattices accept no good representation, implying that solving CVP (even roughly) is NP-hard regardless of the basis (or other auxiliary information) provided. As a result, the CVP instances utilized by lattice-based cryptosystems (for which CVP may be quickly solved using the decryption key) may be simpler than generic CVP instances.

$d(\mathbf{t}, \mathcal{L})$ denotes the distance of $\mathbf{t} \in \mathbb{R}^n$ to the closest lattice vector.

Problem 4.9. (Closest Vector Problem, CVP): Given a lattice basis $\mathbf{B} \in \mathbb{Z}^{n \times m}$ and a target vector $\mathbf{t} \in \mathbb{Z}^n$, find a lattice vector $\mathbf{Bx}$ close to the target $\mathbf{t}$, i.e, find an integer vector $\mathbf{x} \in \mathbb{Z}^m$ such that:$\|\mathbf{Bx} - \mathbf{t}\| \leq \|\mathbf{By} - \mathbf{t}\|$ for any $\mathbf{t} \in \mathbb{Z}^m$.

We introduce three different formulations of CVP:

Problem 4.10. (Decisional version: GapCVP): Given integer lattice $\mathbf{B}$, target vector t and a rational r, determine whether $dist(\mathbf{t}, \mathcal{L}) \leq r$ or $dist(\mathbf{t}, \mathcal{L}) > r$.

Problem 4.11. (CVP, Optimisation version): Given integer lattice $\mathbf{B}$ and target vector $\mathbf{t}$, compute $dist(\mathbf{t}, \mathcal{L}(\mathbf{B}))$.

Problem 4.12. (CVP Search version): Given integer lattice $\mathbf{B}$ and target vector $\mathbf{t}$, find a lattice vector $\mathbf{Bx}$ such that $\|\mathbf{Bx} - \mathbf{t}\|$ is minimum.

Decision versus Search.
As seen below, each of these problems can be simply reduced to the next. Given a search oracle that returns a lattice vector $\mathbf{Bx}$ that is close to $\mathbf{t}$, one may compute the distance between $\mathbf{t}$ and the lattice by evaluating $|\mathbf{Bx} - \mathbf{t}|$. Surprisingly, the search version of CVP is not significantly more difficult than the optimization or decisional variants, given an oracle to resolve the decision problem associated with CVP, the search problem can be solved in polynomial time. Micciancio established in [10] that the search version of CVP may be solved in polynomial time by calling a polynomial number of oracles to answer the decisional CVP issue. This demonstrates that the decisional, optimization, and search variants of (exact) CVP are polynomially comparable, and that decisional CVP reflects the problem's hardness.

NP-Completeness:

Definition 4.13. The subset sum problem (SS) is the following: Given $n+1$ integers $(a_1, ..., a_n, s)$, find a subset of the $a_i's$ (if one exists) that adds up to s, or equivalently, find coefficient $x_i \in \{0, 1\}$ such that$\sum_i a_i x_i = s$. In the decision version of the problem one is given $(a_1, ..., a_n, s)$ and must decide if there exist coefficients $x_i \in \{0, 1\}$ such that $\sum_i a_i x_i = s$.

For a proof of the NP-hardness of the subset sum see [10] (Garey and Johnson, 1979).

Theorem 4.14. For any $p \geq 1$, GapCVP (i.e. the decision problem associated to solving CVP) in the $\mathbf{L}_p$ norm is NP-complete.

For a proof of the NP hardness of the GapCVP problem, i.e., any other problem in NP (or, equivalently, some specific NP-complete problem) can be efficiently reduced to GapCVP. micciancio gave a reduction from the subset sum problem to GapCVP .(SS $\rightarrow$ GapCVP), the GapCVP is at least as hard as the subset sum problem. So GapCVP is NP-hard, afterwards CVP est NP hard.

Problem 4.15. (CVP_γ)-Approximate closest vector problem: Given a basis of $\mathcal{L}$ and an approximation factor $\gamma \geq 1$, find $v \in \mathcal{L}$ such that $\|\mathbf{v} - \mathbf{t}\| \leq \gamma d(\mathbf{t}, \mathcal{L})$.

CVP_γ is provably NP-Hard for $\gamma = 2^{log^{1-\epsilon}n}$. Babai nearest plane (based on LLL) solves CVP_γ in polynomial time for $\gamma = 2(2/\sqrt{3})^n$.

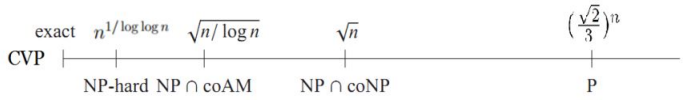

Figure 4: Complexity of CVP_γ

Problem 4.16. (Bounded Distance Decoding, BDD_α). Let $\alpha > 0$. Given as inputs a lattice basis $\mathbf{B}$ and a vector $\mathbf{t}$ such that $dist(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq \alpha.\lambda_1(\mathbf{B})$, the goal is to find a lattice $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ closest to $\mathbf{t}$.

Unlike CVP, which allows the target vector to be as far away from the lattice as feasible, the BDD problem guarantees that the target vector will be within a defined distance of the lattice. It's worth noting that the range of α in some works is limited to $(0, \frac{1}{2})$. This ensures that in the ball of radius $\alpha.\lambda_1(\mathbf{B})$ centered on t, there is precisely one element of $\mathcal{L}$.

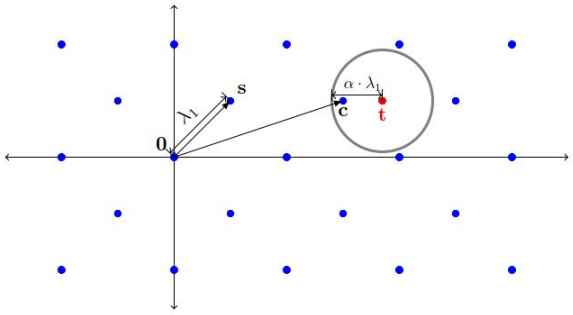

Figure 5: An example of BDD instance

4.3 Average case problems:

An m dimensional lattice is defined as a discrete additive subgroup of $(\mathbb{R}^m, +)$, generated by a basis by forming linear combinations with integer coefficients. suppose $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is a full rank matrix with $m \geq n$. In this work, we consider two lattices of dimension m.
$\Lambda_q(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m / \exists \mathbf{s} \in \mathbb{Z}^n : \mathbf{z} = \mathbf{A}^T\mathbf{s}(modq)\}$ and $\Lambda_q^\perp(\mathbf{A}) = \{z \in \mathbb{Z}^m / \mathbf{A}z = 0(modq)\}$. The first lattice $\Lambda_q(\mathbf{A})$ is formed by linear integer combinations of the rows of $\mathbf{A}modq$. vectors in the second lattice $\Lambda_q^\perp(\mathbf{A})$ are all orthogonal to the rows of $\mathbf{A}$.

A hard average-case problem: Short integer solution (SIS)

The Short Integer Solution was defined by Ajtai in [11] and used to develop a conjectured one-way and collision resistant hash function known as Ajtai function. Being this work the first example of worst-case to average-case reduction involving lattice problems, its importance goes well beyond the hash function itself, which actually turns out to be quite inefficient. Many different cryptographical tools, like identification scheme and digital signature schemes have been based on the SIS.

Definition 4.17. (Short Integer Solution: $SIS_{n,q,\beta,m}$)
Given m uniformly random vectors $\mathbf{a}_i \in \mathbb{Z}_q^n$, grouped as the columns of a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, find a non zero integer vector $\mathbf{z} \in \mathbb{Z}^m, \|\mathbf{z}\| \leq \beta < q$ such that $\mathbf{Az} = \sum_{i=1}^m a_i zi = 0 \in \mathbb{Z}_q^n$

to solve this problem we consider the lattice: $\Lambda_q^\perp(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m : \mathbf{Az} = 0\}\}$, is discrete because we are looking for integer solution and it is subgroup because it is the kernel of a linear transformation. Without an upper bound on $\|\mathbf{z}\|$, the problem can be solved in polynomial time by applying Gaussian reduction (It is easy to find arbitrary integer solution :solving the linear system). To make this problem hard Ajtai impose that the solution is short vector in the lattice ($\mathbf{z} \in \{0,1\}^n$).

Worst-case to Average-case reduction

Theorem 4.18. For $m = poly(n)$, any $\beta > 0$ and any sufficiently large $q \geq \beta.poly(n)$, solving $SIS_{n,q,\beta,m}$ is at least as hard as solving the decisional approximate shortest vector problem $GapSVP_\gamma$ and the approximate shortest independent vectors $SIVP_\gamma$ on arbitrary n-dimensional lattices for some $\gamma = \beta.poly(n)$.

A hard average-case problem: Learning With Error problem (LWE)

In his seminal work from 2005 [Reg2005] [12] , Regev introduced the average-case problem known as Learning With Errors problem, a generalisation of the Learning parity with Noise problem, has been proven to be equally hard to solve as worst-case lattice problem. It has therefore become an important building block in modern cryptographic systems and popular topic in present-research. In addition to its significance in post-quantum cryptography, the LWE problem also has promising applications, such as fully-homomorphic and identity-based encryption. With a fully-homomorphic encryption scheme it is possible to perform calculations on encrypted data, which opens up the opportunity to outsource private computations to third

parties.

Definition 4.19. (informal) Solve a random system of m noisy linear equations and n unknowns modulo an integer q, with $m \geq n$

Let $q \geq 2$ be an integral modulus, let$\mathbf{s} \in \mathbb{Z}^n$ be an n-dimensional vector and let χ be a probability distribution on $\mathbb{Z}_q$. Now, define $\mathbf{A}_{\mathbf{s},\chi}$ as the probability distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$, where samples of this distribution are obtained by the following procedure: $\mathbf{A}_{\mathbf{s},\chi}$: take $a \in \mathbb{Z}_q^n$ and take $e \in \mathbb{Z}_q$ according to the distribution χ return the tuple $(\mathbf{a}, < \mathbf{a}, \mathbf{s} > +e)$ mod q. The learning with errors can be formulated as follows:

Definition 4.20. (Learning with Errors(LWE)). Given a size parameter $n \geq 1$, a modulus $q \geq 2$. a probability distribution χ on $\mathbb{Z}_q$ and an arbitrary number of independent samples from the distribution $\mathbf{A}_{\mathbf{s},\chi}$, find s.

As with the SIS problem LWE can be described in terms of lattices. Consider m LWE samples $(\mathbf{a}_i, \mathbf{b}_i) = (\mathbf{a}_i, < \mathbf{a}_i, \mathbf{s} > +e_i)$ from $\mathbf{A}_{\mathbf{s},\chi}$ for $1 \leq i \leq m$. let $\mathbf{A}$ the $n \times m$ matrix that has the vectors $\mathbf{a}_i$ as his columns. Now, the matrix $\mathbf{A}$ has rank n with high probability. The rows of $\mathbf{A}$ give rise to the lattice: $\Lambda_q(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m / \exists \mathbf{s} \in \mathbb{Z}^n : \mathbf{z} = \mathbf{A}^T\mathbf{s}(modq)\}$.
A secret $\mathbf{s} \in \mathbb{Z}^n$ in the LWE problem now corresponds to the lattice vector $\mathbf{A}^T\mathbf{s} \in \Lambda_q(\mathbf{A})$. The i'th entry of the vector $\mathbf{A}^T\mathbf{s}$ consists of the inner product $< \mathbf{a}_i, \mathbf{s} >$ for $1 \leq i \leq m$. Thus, writing $\mathbf{b} = (\mathbf{b}_1, ..., \mathbf{b}_m)$ and $\mathbf{e} = (\mathbf{e}_1, ..., \mathbf{e}_m)$, the LWE samples gives rise to the equation: $\mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{e}$. The goal of the LWE problem is to find $\mathbf{s}$. This equivalent to find $\mathbf{A}^T\mathbf{s}$, because the matrix $\mathbf{A}$ has rank n with high probability. Since $\mathbf{A}^T\mathbf{s}$ is a lattice vector of $\Lambda_q(\mathbf{A})$, LWE can be described as a closest problem on this lattice. Depending on the choice of the error vector $\mathbf{e}$. $\mathbf{A}^T\mathbf{s}$ will be the closest vector to $\mathbf{b}$ in the lattice $\Lambda_q(\mathbf{A})$. For practical application, the error distribution χ is chosen such that $\mathbf{e}$ is bounded with high probability. This means that LWE can be described as an instance of BDD problem, rather than the more general case of CVP. Thus, the LWE problem is essentially a bounded distance decoding problem in the lattice $\Lambda_q(\mathbf{A})$.

Definition 4.21. (LWE Decision problem). Given $(\mathbf{A}, \mathbf{b})$ with $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{b} \in \mathbb{Z}_q^m$, determine whether $\mathbf{b}$ is chosen uniformly at random from $\mathbb{Z}_q^m$ or $\mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{e}$ mod q.

Definition 4.22. (LWE search problem). Given $(\mathbf{A}, \mathbf{b})$ with $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{b} \in \mathbb{Z}_q^m$, find $\mathbf{s} \in \mathbb{Z}_q^n$ such that $\mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{e}$ mod q.

The Hardness of (LWE)
Regev gives a reduction from approximate version of SVP and SIVP to LWE ($SIVP \rightarrow LWE$). However this reduction uses a quantum computer, this reduction bases the hardness of LWE on the quantum hardness of SVP and SIVP. This implies that a solution to LWE provide a solution to GapSVP, making LWE at least as hard as hard GapSVP.

Theorem 4.23. :If there exists an efficient algorithm that solves the LWE search problem, then there is an efficient quantum algorithm that approximate the GapSVP and the SVP in the worst case.

As no such algorithms to solve GapSVP or SIVP exist, we may indeed assume that the LWE search problem is hard and can even resist to quantum adversaries.

4.4 A summary of several reductions between the different lattice problems

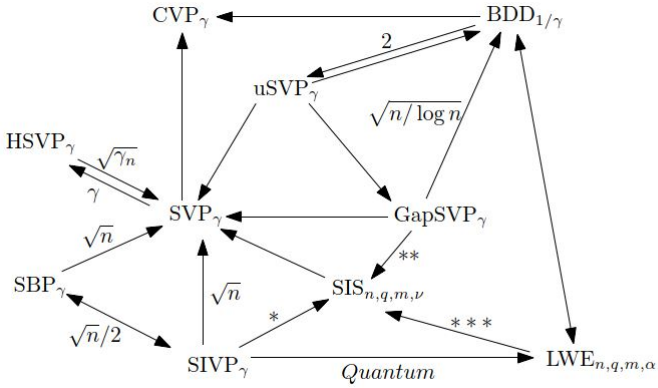

Figure 6: Relation among lattice problems

GapSVP: Compute (or approximate) the value λ_1 without necessarily finding a shortest vector.
GapSIVP: Compute (or approximate) the value λ_n without necessarily finding short linearly independent vectors.
($GapSVP \approx GapSIVP \approx BDD$) approximating λ_1 or approximating λ_n or solving BDD are all equivalent up to poynomial factor. these class of problem are those from wich we knew how to build public key cryptography function.
SIVP: Finding n linearly independent short vectors or solving ADD are equivalent ($SIVP \approx ADD$) from those problem we knew how to build private key cryptography function.

($BDD \leq CVP$) the reduction from BDD to CVP is trivial, because CVP is simply BDD without the distance bound.
($GapSVP \leq SVP$) The reduction from GapSVP to SVP is trivial, because the solution of the search problem solves the decision problem.
($GapSVP \leq Search-LWE \leq decision-LWE$) The learning with errors problem is equivalent to a bounded distance decoding. Regev gives a reduction from approximate versions of SVP and SIVP to LWE. This reduction uses a quantum computer and bases the hardness on the quantum hardness fo SVP and SIVP.
($GapSVP \leq SIS$) Micciancio and Regev described the reduction from SIVP and GapSVP to SIS. SIS can be trivially reduced to SVP.
($LWE \leq SIS$) SIS is at least as hard as LWE

5 Algorithms for lattice problems

A lattice has an infinity of bases, all of which are algebraically equivalent. But from a Euclidean point of view, this is no longer the case, and some of these bases have more interesting Euclidean properties. The objective of the reduction is to find in one reasonable time a basis with fairly good Euclidean properties, formed by fairly orthogonal vectors, and sufficiently short to give approximations for successive minima. But, as we have already seen, and from dimension 5, the successive minima do not necessarily form a base of the lattice.
The LLL method, invented in 1982 by Lenstra, Lenstra, and Lovasz, is the most well-known and extensively examined algorithm for lattice problems. This is a polynomial-time SVP (and most other fundamental lattice problems) algorithm that achieves a $2^{\Theta(n)}$ approximation factor, where n is the lattice dimension.
The LLL algorithm, as bad as it may appear, is surprisingly helpful, with applications ranging from factoring polynomials over rational numbers to integer programming, as well as numerous cryptanalysis applications (such as attacks on knapsack-based cryptosystems and special cases of RSA).
Schnorr published an extension of the LLL method in 1987 that resulted in somewhat better approximation factors. The primary idea behind Schnorr's algorithm is to replace the LLL method's core, which contains blocks, with larger blocks. At the cost of higher running time, increasing the block size improves the approximation factor (i.e., leads to shorter vectors). There are several variations of Schnorr's algorithm, including one developed recently by Gama and Nguyen that is quite natural and elegant. Regrettably, all of these alternatives attain a similar level of exponential approximation.
The best known algorithm has a running time of $2^{O(n)}$ if one insists on a precise solution to SVP, or even only an approximation to within poly(n) factors. Unfortunately, this algorithm's space need is likewise exponential, making it virtually impractical. Other techniques use only polynomial space but take $2^{O(nlogn)}$ time to complete. The above debate leads us to the following hypothesis:
Conjecture: There is no polynomial-time algorithm that can solve lattice problems inside a polynomial factor.

6 Lattices and quantum algorithms:

As we've seen, lattice problems are extremely difficult. The most well-known algorithms either take an infinite amount of time or have poor approximation ratios. The field of lattice-based cryptography arose from the belief that lattice problems are difficult. Is lattice-based cryptography, on the other hand, appropriate for a post-quantum world? Is it true that lattice problems are intractable even for quantum computers?.
The short answer is probably yes: there are currently no quantum algorithms for solving lattice problems that outperform the best known classical (non-quantum) techniques. This is despite the fact that lattice problems appear to be a natural candidate for solving with quantum algorithms: they aren't thought to be NP-hard for typical approximation factors, they have a periodic structure, and the Fourier transform, which is widely used in quantum algorithms, is closely linked to the concept of lattice duality.
Since Shor's discovery of the quantum factoring algorithm in the mid-1990s, attempts to solve lattice problems using quantum algorithms have met with little success, if any at all. The fundamental issue is that the periodicity discovery technique employed in Shor's factoring algorithm and other quantum algorithms does not appear to work for lattice problems. As a result, it is only natural to explore the following conjecture to justify the adoption of lattice-based cryptography for post-quantum encryption: Conjecture: There isn't a polynomial time quantum algorithm that can solve lattice problems inside polynomial factors.

7 Reduction of the lattice basis

As a result, lattice reduction methods can be utilized to approximate the shortest vector problem by providing a foundation of sufficiently short vectors. However, the findings of lattice reduction

can also be used to develop an algorithm that approximates the closest vector problem. Here are two methods for approximating the nearest vector problem. The embedding approach [13] and Babai's rounding technique.

7.1 The Babai's Rounding technique

Given a target $\mathrm{w} \in \mathbb{R}^n$ we can write $\mathrm{w} = \sum_{i=1}^{n} \mathrm{l}_i \mathrm{b}_i$ with $\mathrm{l}_i \in \mathbb{R}$.One computes the coefficient l_i by solving the system of linear equation (since the lattice is full rank we can also compute the vector $(\mathrm{l}_1, ..., \mathrm{l}_n)$ as $\mathrm{w}\mathbf{B}^{-1}$). The rounding method technique is simply to set $\mathrm{v} = \sum_{i=1}^{n} \lfloor \mathrm{l}_i \rceil \mathrm{b}_i$. Where $\lfloor \mathrm{l}_i \rceil$ means take the closest integer to the real number l_i.

7.2 Embedding technique

Although SVP can be reduced to CVP, there also exists a heuristic method to convert instances of CVP to an instance of SVP in a lattice in a similar dimension. This method, known as the "embedding technique" allows for reasonable approximations to CVP.
The embedding technique works as follows. Take an instance of the CVP with a basis $\{\mathrm{b}_1, ..., \mathrm{b}_n\}$ for the lattice $\mathcal{L}$ and with the target vector $\mathbf{c}$. Now construct the $(n+1)$-rank lattice $\mathcal{L}'$ using the rows of the following matrix as basis vectors:

$$\mathbf{B}' = \begin{pmatrix} b_{11} & b_{12} & ... & b_{1i} & ... & b_{1n} & 0 \\ b_{21} & b_{22} & ... & b_{2i} & ... & b_{2n} & 0 \\ ... & ... & ... & ... & ... & ... & 0 \\ b_{i1} & b_{i2} & ... & b_{ii} & ... & b_{in} & 0 \\ ... & ... & ... & ... & ... & ... & 0 \\ b_{n1} & b_{n2} & ... & b_{ni} & ... & b_{nn} & 0 \\ c_1 & c_2 & ... & c_i & ... & c_n & 1 \end{pmatrix}$$

The determinant of $\mathbf{B}'$ is the same as the determinant of $\mathbf{B}$ of $\mathcal{L}$. Thus the volume of the lattice $\mathcal{L}'$ is the same as that of $\mathcal{L}$. Furthermore, the rank is nearly the same, since the rank of $\mathcal{L}$ is n and the rank of $\mathcal{L}'$ is $n+1$. The expectation is that the length of the shortest lattice vector is approximately the same in both lattices. Now, consider the vector that is closest to c as an linear combination of the $\mathrm{b}_i's$,$\mathrm{x} = \sum_i \lambda_i \mathrm{b}_i$. The idea is that $(\mathrm{c} - \mathrm{x})$ will have relatively small entries(depending on how close c to the lattice), and therefore $(\mathrm{c} - \mathrm{x}, 1)$ will be a short vector in $\mathcal{L}'$.

8 Our method to approximate the closest vector problem

Definition 8.1. Random variables: A random variable is any rule(i.e,function)that associates a number with each outcome in the sample space.
The set of possible values that a random variable $\mathbf{X}$ can takes is called the range of $\mathbf{X}$. A random variable $\mathbf{X}$ is said to be discrete if its range consists of finite or countable number of values.
The probability function of a discrete a random variable $\mathbf{X}$ is the function $p(x)$ satisfying:

$$p(x) = Pr(\mathbf{X} = x)$$

for all x in the range of $\mathbf{X}$.
The mean or expectation of a discrete random variable $\mathbf{X}$, $\mathbf{E}(\mathbf{X})$ is defined as

$$\mathbf{E}(\mathbf{X}) = \sum_x x Pr(\mathbf{X} = x)$$

Variance:

$$Var(\mathbf{X}) = \mathbf{E}(\mathbf{X}^2) - \{\mathbf{E}(\mathbf{X})\}^2$$

Standard deviation:

$$Sd(\mathbf{X}) = \sqrt{Var(\mathbf{X})}$$

.

Our approach to reducing the closest vector to the shortest problem is as follows. Assume we want to find the point in a lattice $\mathcal{L}(\mathbf{B})$ where $\mathbf{B} = (\{\mathrm{b}_1, ..., \mathrm{b}_n\})$ (approximately) closest to some target $\mathbf{c}$ is to embed the vectors $[\mathbf{B}|\mathbf{c}]$ in a higher dimensional space (using the Embedding Technique) and add to $\mathbf{c}$ a component orthogonal to $\mathbf{B}$. In other words we consider the lattice generated by the matrix.

$$\mathbf{B}' = \begin{pmatrix} \mathbf{B} & 0 \\ \mathbf{c} & 1 \end{pmatrix}$$

Notice that if $\mathbf{B}$ is a basis of $\mathcal{L}(\mathbf{B})$, then the rows of matrix $\mathbf{B}'$ are linearly independent,i.e $\mathbf{B}'$ is a basis of $\mathcal{L}(\mathbf{B}')$. The determinant of $\mathbf{B}'$ is the same as the determinant of $\mathbf{B}$ of $\mathcal{L}$. Thus the volume of the lattice $\mathcal{L}'$ is the same as that of $\mathcal{L}$. Furthermore,the rank is nearly the same, since the rank of $\mathcal{L}$ is n and the rank of $\mathcal{L}'$ is $n+1$.The expectation is that the length of the shortest lattice vector is approximately the same in both lattices.

$$\mathcal{L}(\mathbf{B}') = \begin{pmatrix} b_{11} & b_{12} & ... & b_{1i} & ... & b_{1n} & 0 \\ b_{21} & b_{22} & ... & b_{2i} & ... & b_{2n} & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ b_{i1} & b_{i2} & ... & b_{ii} & ... & b_{in} & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ b_{n1} & b_{n2} & ... & b_{ni} & ... & b_{nn} & 0 \\ c_1 & c_2 & ... & c_i & ... & c_n & 1 \end{pmatrix}$$

we will approximate the shortest problem in the lattice $\mathcal{L}(\mathbf{B}')$ by the reduced centred law. On each column i we will do the following calculations.

The mean: $m_i = \frac{b_{1i}+b_{2i}+...+b_{ni}+c_i}{n+1}$; $\sigma_i = (\frac{b_{1i}^2+b_{2i}^2+...+b_{ni}^2+c_i^2+}{n+1} - m_i^2)^{\frac{1}{2}}$
$x_i = \frac{c_i - m_i}{\sigma_i}$, we then obtain an approximation of the shortest vector $\mathbf{x} = (\lceil x_1 \rfloor, \lceil x_2 \rfloor, ..., \lceil x_n \rfloor)$. And finally the approximation of the closest vector:

$$\mathbf{t} = (c_1 - \lceil x_1 \rfloor, c_2 - \lceil x_2 \rfloor, ..., c_n - \lceil x_n \rfloor)$$

Example 8.2. Consider the basis matrix:

$$\mathbf{B} = \begin{pmatrix} 35 & 72 & -100 \\ -10 & 0 & -25 \\ -20 & -279 & 678 \end{pmatrix}$$

of a lattice in $\mathbb{R}^3$. We solve the closest vector problem instance with $\mathbf{w} = (100, 100, 100)$. Apply the method of the reduced centred law to the basis :

$$\mathbf{B}' = \begin{pmatrix} 35 & 72 & -100 & 0 \\ -10 & 0 & -25 & 0 \\ -20 & -279 & 678 & 0 \\ 100 & 100 & 100 & 1 \end{pmatrix}$$

The mean:

$$(26, 25; -26, 75; 163, 25; 0, 25)$$

The variance:

$$(2242, 1875; 22540, 6875; 93426, 6875; 0, 1875)$$

The standard deviation:

$$(47, 35174231; 150, 1355637; 305, 6577948; 0, 433012702)$$

The reduced centered matrix:

$$\mathbf{c} = \begin{pmatrix} 0,184787287 & 0,657738896 & -0,861257277 & -0,577350269 \\ -0,765547332 & 0,178172309 & -0,615884833 & -0,577350269 \\ -0,976732803 & -1,680148219 & 1,684072871 & -0,577350269 \\ 1,557492848 & 0,844237014 & -0,206930761 & 1,732050808 \end{pmatrix}$$

The method give an approximation du shortest vector problem of the lattice $\Lambda(\mathbf{B})'$ is $(0, 1, 0, 1)$, so we know that $(0, 1, 0)$ is the difference between $\mathbf{w}$ and a lattice point $\mathbf{v}$. On verifies that:

$$\mathbf{v} = (100, 100, 100) - (0, 1, 0) = (100, 99, 100),$$

is the lattice vector close to $\mathbf{w}$.

Example 8.3. Consider the basis matrix:

$$\mathbf{B} = \begin{pmatrix} 7 & 69 & -990 & 425 & 512 & -346 \\ 56 & 575 & -8514 & 934 & 1345 & 3 \\ -77 & -644 & 8019 & 66 & 156 & -33 \\ 17 & -275 & 4514 & -34 & 45 & 365 \\ 516 & -75 & 14 & -634 & 137 & -31 \\ 230 & -5 & 14 & -2334 & 845 & -63 \end{pmatrix}$$

of a lattice in $\mathbb{R}^6$. We solve the closest vector problem instance with

$\mathbf{w} = (282759, 2639330, -2132526, -1039397, 491124, 1762598)$.

Apply the method of the reduced centred law to the basis (using the embeding technique to the lattice $\mathcal{L}(\mathbf{B}')$:

$$\mathbf{B}' = \begin{pmatrix} 7 & 69 & -990 & 425 & 512 & -346 & 0 \\ 56 & 575 & -8514 & 934 & 1345 & 3 & 0 \\ -77 & -644 & 8019 & 66 & 156 & -33 & 0 \\ 17 & -275 & 4514 & -34 & 45 & 365 & 0 \\ 516 & -75 & 14 & -634 & 137 & -31 & 0 \\ 230 & -5 & 14 & -2334 & 845 & -63 & 0 \\ 282759 & 2639330 & -2132526 & -1039397 & 491124 & 1762598 & 1 \end{pmatrix}$$

Calculations are made with the Maple application: The inverse matrix

$$c = \begin{bmatrix} \frac{50174428484600}{35137707797785443} & \frac{-4412600415054}{11712569265928481} & \frac{-34720702968460}{35137707797785443} & \frac{47552443077155}{35137707797785443} & \frac{69562227293795}{35137707797785443} & \frac{-16731266418952}{35137707797785443} \\ \frac{3090460880447 14}{35137707797785443} & \frac{-35198605266615}{11712569265928481} & \frac{-227194242191708}{35137707797785443} & \frac{272300523042463}{35137707797785443} & \frac{-84702090305455}{70275415595570886} & \frac{30261885450455}{70275415595570886} \\ \frac{123144201326636}{175688538988927215} & \frac{-29293348626183}{117125692659284810} & \frac{-71625235846399}{175688538988927215} & \frac{110318625507818}{175688538988927215} & \frac{-13383302581714}{175688538988927215} & \frac{9685706205841}{35137707797785443 0} \\ \frac{57381467398681}{175688538988927215} & \frac{5251755934531}{585628463296424 05} & \frac{-1381287686954}{175688538988927215} & \frac{44881642636618}{175688538988927215} & \frac{50562651914837}{35137707797785443 0} & \frac{-132160665561199}{35137707797785443 0} \\ \frac{13891446619330}{35137707797785443} & \frac{9292449713817}{23425138531856962} & \frac{8071282535230}{35137707797785443} & \frac{14985072829714}{35137707797785443} & \frac{-10486971894277}{70275415595570886} & \frac{9541632015370}{35137707797785443} \\ \frac{-74725471641610}{35137707797785443} & \frac{18876970687067}{23425138531856962} & \frac{6582185752100}{35137707797785443} & \frac{25334717496671}{35137707797785443} & \frac{-928163907100}{70275415595570886} & \frac{-4413187432091}{70275415595570886} \end{bmatrix}$$

Figure 7: Inverse matrix

We obtain the shortest vector:

$$\mathrm{e} = (0, 1, 0, -1, 1, 0)$$

The lattice vector v close to w is:

$$\mathrm{v} := \begin{bmatrix} 282759 \\ 2639331 \\ -2132526 \\ -1039398 \\ 491125 \\ 1762598 \end{bmatrix}$$

For verification we solve the equation: $\mathbf{xB} = \mathbf{v}$, we find that

$$\mathrm{x} := \begin{bmatrix} 243 \\ 456 \\ -234 \\ -512 \\ 631 \\ 253 \end{bmatrix}$$

Indeed $\mathbf{v}$ is the lattice vector close to $\mathbf{w}$.

9 Conclusion

Cryptography is used to protect the integrity and confidentiality of messages, as well as to authenticate their source. The security of most cryptographic primitives relies on number theory problems. In this work we have dealt with the problem of the closest vector on a Lattice, we have used a statistical method.

References:

[1] C. Peikert, Public-key cryptosystems from the worst-case shortest vector problem, In STOC, 2009, pp. 333342.

[2] J. H. Van de Pol, Lattice- based cryptography, PHD thesis, July 2011.

[3] P. W. Shor, Algorithms for quantum computation, In Proceedings of the 35^{th} Annual Symposium on Foundations of Computer Science, 1994.

[4] J. Hoffstein, J. Pipher, and J. Silverman, NTRU: A new high speed public key cryptosystem, Algorithic number theory, Lecture note in computer science, 1998.

[5] M. Ajtai, Generating Hard Instances of Lattice Problems, Proceedings of the 28th Annual ACM Symposium on Theory of Computing, 1996, or Electronic Colloquium on Computational Complexity, 1996. http://www.eccc.uni-trier.de/eccc/

[6] C. Gentry, Fully homomorphic encryption using ideal lattices, In Procceding of the 41^{st} annual ACM symposium on theory of computing, 2009, pp. 169-178.

[7] R. M Karp, Reducibility among combinatorial problems, Complexity of computations proceedings, 1972, pages 58-104.

[8] O. Goldreich, D. Micciancio, S. Seifert and S. Safra, Approximating shortest lattice vectors is not harder than approximating closest lattice vectors, Information Processing Letters, 71(2), 1999, pp. 55-61.

[9] D. Micciancio and C. Peikert, Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller, Cryptology ePrint Archive, Report 2011/501, 2011. https://ia.cr/2011/501

[10] Garey, Michael R. and Johnson, David S., Computers and Intractability; A Guide to the Theory of NP-Completeness, W. H. Freeman and Co., USA, 1990.

[11] M. Ajtai, Generating hard instances of lattice problems (extend abstract), Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing, July,1996, pp. 99-108. https://doi.org/10.1145/237814.237838

[12] O. Regev, On lattices, learning with errors random linear code and cryptography, In Proc of Stoc, 56(6), 2009 .

[13] L. Babai, On Lvasz lattice reduction and the nearest lattice point problem, Combinatorica 6, 1986, pp. 1-13.

[14] M. Sahmoudi, A. Chillali, Elliptic Curve on a Family of Finite Ring, WSEAS Transactions on Mathematics,Volume 18, 2019, pp. 415-422.

[15] S. Abdelalim, A. Chillali, S. Elhajji, Point of Infinite Order on an Elliptic Curve over a Quadratic Field, WSEAS Transactions on Mathematics,Volume 13, 2014, pp. 428-430.