UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Criptografía Post Cuántica basada en Reticulados

Fundamentos Teóricos y Sistemas de Clave Pública

Bruno Scarone Etchamendi

Instituto de Computación
Facultad de Ingeniería
Universidad de la República

Montevideo – Uruguay
Diciembre de 2018

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Criptografía Post Cuántica basada en Reticulados

Fundamentos Teóricos y Sistemas de Clave Pública

Bruno Scarone Etchamendi

Proyecto de Grado de Ingeniería en Computación presentado al Tribunal Evaluador de la Facultad de Ingeniería de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Ingeniero en Computación.

Director:
Dr. Alfredo Viola

Montevideo – Uruguay
Diciembre de 2018

INTEGRANTES DEL TRIBUNAL DE DEFENSA DE PROYECTO DE GRADO

Dr. Eduardo Canale

Dr. Pablo Romero

Dra. Libertad Tansini

Montevideo – Uruguay
Diciembre de 2018

Agradecimientos

A mis padres María y Claudio y a mi hermano Gastón, por ser un ejemplo de vocación, dedicación y esfuerzo en la vida. Por la paciencia y el apoyo incondicional a lo largo de estos años, junto a las muchas horas de charlas y reflexiones sobre temas tan diversos como interesantes. Siempre han querido lo mejor para mí y me han acompañado en cada una de mis metas.

A mis amigos de Facultad, porque hicieron que este camino sea mucho más disfrutable y llevadero en los momentos más difíciles. A mis otros amigos, por siempre estar dispuestos a escuchar y conversar.

A Brigitte Vallée y Ali Akhavi por sus comentarios sobre el capítulo de reticulados y algoritmos de resolución. En particular a Brigitte por los consejos, la buena disposición y la organización de las reuniones para contribuir al entendimiento de estos temas durante mi estadía en París. A Joachim von zur Gathen por sus comentarios generales sobre complejidad, indistinguibilidad computacional y criptografía basada en reticulados.

A Alfredo Viola (Tuba) por las charlas y el tiempo compartido, las nuevas perspectivas, la forma para abordar los problemas, así como la motivación para buscar nuestro propio camino de vida y la convicción para seguir las experiencias en las que creemos. Por habernos incentivado a participar en instancias como Latincrypt y ponernos en contacto con personas que nos mostraron visiones nuevas sobre estos temas. Por haber sido un apoyo constante en todo este tiempo, incluso en la recuperación de su cirugía.

Con la sensación de que hablaba con O'Brien, y también de que anotaba un importante axioma, escribió:
La libertad es poder decir libremente que dos y dos son cuatro. Si se concede esto, todo lo demás resultará.

George Orwell, 1984.

RESUMEN

El presente trabajo desarrolla los conceptos fundamentales de la criptografía post cuántica basada en reticulados. Para esto, se introducen los fundamentos generales de criptografía y seguridad necesarios, así como de la teoría matemática de reticulados. Con respecto a esta última temática, se presentan las propiedades y características de principal interés relacionadas con estos objetos, para luego estudiar problemas computacionales asociados, su dificultad y el estado del arte en materia algorítmica para su resolución.

Luego, se procede a realizar un relevamiento y análisis de tres de los principales sistemas de clave pública basados en reticulados desarrollados hasta la fecha: los sistemas GGH, NTRU y el primero basado en el problema de "Aprender con Errores". Este problema, que también se presenta en el trabajo, ha permitido desarrollar nuevas construcciones que poseen garantías robustas de seguridad y por tanto resultan de gran interés para la comunidad científica del área. Para cada uno de estos sistemas, se estudian los órdenes de ejecución de las primitivas de encriptado y desencriptado, el espacio de almacenamiento de las claves, así como las condiciones bajo las que los mismos funcionan de manera correcta. También se evalúan distintos ataques sobre los mismos, junto a sus niveles de seguridad.

Palabras claves:
Criptografía, Criptografía Post Cuántica, Reticulados, Seguridad Informática, Complejidad Computacional.

ABSTRACT

The present study develops fundamental concepts of post quantum lattice-based cryptography. To this end, necessary notions concerning cryptography and security are introduced, together with the mathematical theory of lattices. Main properties and quantities of interest related to lattices are presented, to be used later in the study of associated computational problems, their hardness and the state of the art in algorithmic terms for its resolution.

Later, a survey and analysis of three of the most relevant public key cryptosystems based on lattices proposed up to date are described: GGH, NTRU and the first based on the "Learning with Errors" problem. This problem, also presented in this work, has enabled the creation of new constructions that possess strong security guarantees and is therefore of great interest for the scientific community. For each of these systems, the order of execution of their encryption and decryption primitives and the space needed for key storage are quantified, together with the conditions under which they work correctly. Also, different attacks against these systems are evaluated, in conjunction with their security levels.

Keywords:
Cryptography, Post Quantum Cryptography, Lattices, Information Security, Computational Complexity.

Lista de símbolos

Lista de los símbolos más relevantes del proyecto.

$\mathbb{N}$ Conjunto de los números naturales, incluye al cero. 10

$O(g(n))$ Cota superior asintótica. 8

$\Omega(g(n))$ Cota inferior asintótica. 8

$\Theta(g(n))$ Orden exacto asintótico. 8

$\{0,1\}^n$ Conjunto de las tiras binarias de largo n. 9

P Clase de complejidad. 9

NP Clase de complejidad. 10

$\{0,1\}^*$ Conjunto de las tiras binarias de largo arbitrario. 11

$\leq_p$ Reducción en tiempo polinomial. 11

$\mathbb{B} = \{0,1\}$ Conjunto de bits. 15

$\lfloor x \rceil$ Entero más cercano a $x \in \mathbb{R}$. 64

$\xleftarrow{\boxdot}$ Elemento tomado aleatoriamente según una distribución. 80

$\mathbb{Z}[x]$ Conjunto de los polinomios con coeficientes enteros. 93

$\mathbb{1}$ Función indicatriz de un conjunto. 94

$\texttt{poly}(n)$ Función polinomial en n. 108

$\xleftarrow{\chi} S$ Elemento tomado con distribución χ del conjunto S. 109

$adv_{\mathcal{D}}$ Ventaja de un algoritmo distinguidor $\mathcal{D}$. 124

$\mathbb{R}^+$ Conjunto de los números reales positivos. 14

Lista de siglas

Lista de siglas

CVP Problema del vector más cercano 55

GGH Criptosistema Goldreich, Goldwasser y Halevi 6

LWE Problema de "Aprender con Errores" 7

NTRU Criptosistema propuesto por Hoffstein, Pipher y Silverman 6

SVP Problema del vector más corto 54

TPP Tiempo Polinomial Probabilístico 12, 13

Tabla de contenidos

Capítulo 1

Introducción

La criptografía puede ser definida como el estudio de técnicas matemáticas orientadas a garantizar la seguridad de la información digital, sistemas y computación distribuida frente a ataques de adversarios [1]. En este sentido, el área incluye tanto la creación de metodologías y sistemas para lograr los objetivos antes mencionados, así como también para desarrollar el criptoanálisis, estudio y quebrado[1] de los mismos.

La humanidad se ha interesado a lo largo de gran parte de su historia por desarrollar formas de transmisión secreta de mensajes, mucho antes del surgimiento de la computación.[2] El enfoque preponderante antes de 1976, consistía en acordar algún tipo de información secreta (usualmente denominada clave) entre los participantes de la comunicación y mediante su uso poder lograr una comunicación segura. A este esquema se lo denomina criptografía de clave privada o clave simétrica, ya que la misma debe ser compartida por los participantes de la comunicación.

En 1976, Whitfield Diffie y Martin Hellman produjeron en [3] una revolución en el área, proponiendo considerar la construcción de sistemas donde una parte de la clave se mantuviera en secreto y otra se hiciera pública. El paradigma propuesto, denominado criptografía de clave pública, resuelve el desafío de acordar una clave de forma secreta entre dos personas, tarea peligrosa que

[1] Se pueden definir diversas nociones sobre qué constituye "quebrar" un sistema, siendo la más poderosa obtener la clave secreta del mismo. En [2] se puede encontrar una discusión profunda del tema.

[2] Para una exposición introductoria que incluye aspectos históricos del desarrollo de la criptografía consultar [2].

presentó grandes dificultades a lo largo de la historia. Esta contribución dio lugar al enfoque moderno del área. La seguridad de los criptosistemas pasó a ser sostenida por el área de la complejidad computacional. En este sentido, la seguridad se basa en usar versiones de problemas que se suponen no tienen solución "eficiente"[1], como los problemas de factorización de enteros y de logaritmos discretos en ciertos grupos.

Hoy en día, los grandes volúmenes de información almacenados, procesados y transmitidos por computadoras en todas partes del mundo ponen de manifiesto la gran relevancia del campo y la importancia que tiene el continuo desarrollo y análisis de las distintas metodologías y herramientas propuestas por la comunidad científica del área.

Los sistemas criptográficos más utilizados en la actualidad fueron desarrollados en función de construcciones matemáticas de teoría de números, como el problema de factorización de enteros en el que se basa el sistema RSA [4] y el problema de encontrar el logaritmo discreto en alguna clase de grupo utilizado en el protocolo de Diffie y Hellman [3] para el acuerdo de claves. Sin embargo, el desarrollo de la computación cuántica, en particular mediante el algoritmo cuántico propuesto en 1994 por Shor [5], permite resolver de forma "eficiente" los dos problemas antes mencionados para instancias en las que esto no se cree posible mediante computadoras clásicas. Para que este algoritmo pueda ser implementado en la práctica, deben ser desarrolladas computadoras cuánticas de tamaño considerable. Si bien, en la actualidad no se conoce públicamente la construcción exitosa de una computadora cuántica que pueda ser utilizada con estos propósitos, la posibilidad de que en un futuro esto pueda concretarse, genera la necesidad de comenzar a investigar alternativas que permitan mantener la información segura incluso ante ataques implementados con este tipo de computadoras. De aquí surge la idea de comenzar a desarrollar nuevos sistemas, que resulten resistentes a ataques realizados por una computadora cuántica. Criptosistemas que se consideren resistentes a ataques cuánticos, son el centro del área conocida como criptografía post cuántica.

En la Sección 1.1 comentamos brevemente las ideas básicas que se encuentran detrás del modelo de computación cuántica. Esta presentación está basada

[1] En el Capítulo 2, se define formalmente esta noción.

en [6]. Luego en la Sección 1.2 profundizamos en la noción de cripotgrafía post cuántica.

1.1. Computación Cuántica

La unidad fundamental de cómputo en el modelo de computación clásica es el bit, objeto que puede tomar dos estados posibles, usualmente denotados por 0 y 1. Basados en esta unidad, es posible mediante compuertas clásicas realizar la transmision y el almacenamiento de información.

En el modelo de computación cuántica existe el concepto análogo de bit cuántico o qubit. Este objeto puede estar en una superposición de dos estados, usualmente denotados $|0\rangle$ [1] y $|1\rangle$ que suelen llamarse estados de la base computacional. Si realizamos una medición de un qubit para observar su estado, dicha superposición es colapsada y se obtiene uno de los dos valores de la base computacional. Este proceso es no determinista y los distintos valores se obtienen según una distribución de probabilidad definida por la superposición de los estados.

Es posible entonces tomar sistemas de múltiples qubits y mediante el uso de compuertas cuánticas manipularlos de modo de almacenar y transmitir información. Una de las características que diferencia a la computación cuántica de la clásica, es la capacidad de poder realizar el cómputo de forma paralela sobre distintos valores, característica denominada paralelismo cuántico. Dicho paralelismo es posibilitado por la superposición de estados que comentamos previamente. Esto permite que algunos algoritmos cuánticos puedan resolver determinados problemas computacionales de forma drásticamente más eficiente que en el modelo clásico.

Sin embargo, las técnicas de diseño para la concepción de algoritmos cuánticos resultan menos intuitivas y se encuentran menos desarrolladas que las utilizadas en algoritmia clásica [6].

[1] La notación $|\cdot\rangle$ es llamada notación de Dirac y es utilizada como estándar en el área de computación cuántica [6].

1.2. Criptografía Post Cuántica

Como comentamos, el objetivo de esta subárea de la criptografía es el desarrollo de sistemas que se crean resistentes tanto a ataques realizados mediante computadoras clásicas así como cuánticas. En este sentido, debe incorporarse en el proceso de criptoanálisis de las distintas alternativas, la posibilidad de que un atacante tenga acceso a una computadora cuántica para la implementación de las estrategias de ataques.

Además, resulta deseable que las alternativas diseñadas presenten parámetros de implementación eficientes en la práctica, tanto en cuanto a tiempo de ejecución así como espacio de almacenamiento y que no representen un empeoramiento significativo con respecto a las distintas alternativas utilizadas actualmente en la criptografía clásica. Esto resulta de particular interés en contextos donde la eficiencia es un requisito fundamental en la realización de las tareas, como puede ser el caso de algunos servidores de Internet, que deben satisfacer miles de pedidos por segundo [7].

Corresponde sin embargo, cuestionarse si es necesario invertir un esfuerzo considerable en estas líneas de investigación, pese a no tener la certeza de que una computadora cuántica escalable pueda ser construida en la práctica. Con respecto a esto, en [7] se presentan tres razones por las que el autor cree que resulta relevante incentivar el estudio del área en cuestión:

- Se necesita tiempo para mejorar la eficiencia de la criptografía post cuántica.
- Se necesita tiempo para incrementar la confianza en este tipo de sistemas y finalmente,
- Se necesita tiempo para mejorar la usabilidad de los mismos y tomar las consideraciones necesarias para garantizar su seguridad en la práctica.

Es decir, el tiempo de investigación dedicado tiene un carácter preventivo que puede resultar decisivo en el futuro.

Una de las principales clases de sistemas criptográficos post cuánticos está basada en el objeto matemático conocido como reticulado (o *lattice* en inglés).

En particular, como veremos en el trabajo, este tipo de sistemas ofrece pruebas de seguridad robustas basadas en la dificultad de instancias de problemas en el peor caso, implementaciones relativamente eficientes, así como simplicidad en sus construcciones [7].

Estas características son las que han hecho que este tipo de sistemas sea la categoría de propuestas predominantes en el proceso de estandarización que está llevando a cabo el Instituto Nacional de Estándares y Tecnología (NIST por sus siglas en inglés) de Estados Unidos. Dicho proyecto [8] tiene por objetivo solicitar, evaluar y estandarizar una o más propuestas de sistemas de clave pública post cuánticos para su uso futuro, previendo el posible advenimiento de computadoras cuánticas de gran escala.

Desde el descubrimiento del algoritmo de Shor, se han intentado desarrollar distintos algoritmos cuánticos para la resolución eficiente de ciertos problemas computacionales basados en reticulados, sin éxito alguno. Esto da lugar a la conjetura de que no existen algoritmos cuánticos "eficientes" que puedan resolver, incluso de forma aproximada para ciertos factores de aproximación, problemas en reticulados [7]. Todos los motivos antes expuestos justifican el estudio del área de la criptografía post cuántica y en particular la de los sistemas basados en reticulados, que son el objeto central de estudio de este trabajo.

1.3. Objetivos propuestos

Este trabajo busca presentar los conceptos fundamentales de la criptografía post cuántica basada en reticulados. Para lograr este objetivo, se introducen los fundamentos generales de criptografía y seguridad necesarios, así como de la teoría de reticulados. Con respecto a esto último, se presentarán las propiedades y características de principal interés relacionadas con estos objetos, para luego estudiar problemas computacionales asociados, su dificultad y el estado del arte en materia algorítmica para su resolución.

Luego, se procederá a realizar un relevamiento y análisis de los principales sistemas de clave pública desarrollados que hacen uso de reticulados. En particular, se incluirán propuestas vinculadas al surgimiento de este tipo de

sistemas, así como propuestas que resulten eficientes y otra que presenta garantías de seguridad como característica principal del diseño del sistema. Para cada propuesta relevada, se estudiarán los órdenes de ejecución de las primitivas de encriptado y desencriptado, el espacio necesario para el almacenamiento de claves, las condiciones para que los mismos funcionen de manera correcta, evaluándose a su vez distintos ataques y las propiedades en cuanto a seguridad de los mismos.

1.4. Organización del trabajo

El trabajo se encuentra organizado de la siguiente manera:

En el Capítulo 2, se presentan los fundamentos teóricos que incluyen las definiciones y nociones básicas utilizadas a lo largo del trabajo. El Capítulo 3 introduce la teoría de reticulados: las definición básicas, características fundamentales y cantidades de interés para el estudio.

El Capítulo 4 refiere a los problemas computacionales fundamentales asociados a reticulados. Se presentan los problemas principales en sus distintas variantes, las complejidades asociadas y se concluye el capítulo con un estudio de los principales algoritmos existentes para su resolución.

El Capítulo 5 trata uno de los sistemas iniciales de clave pública más relevantes basado en reticulados: el Criptosistema Goldreich, Goldwasser y Halevi (GGH). Este sistema se construye de forma directa en base a uno de los problemas fundamentales presentado en el Capítulo 4, motivo por el que se eligió como el primer sistema de estudio.

El siguiente sistema a considerar es el Criptosistema propuesto por Hoffstein, Pipher y Silverman (NTRU) (Capítulo 6), también dentro de los sistemas inicialmente concebidos que hacen uso de reticulados. Este sistema, a diferencia de GGH, ha mantenido vigencia en la actualidad con algunas modificaciones, siendo una de sus principales características la eficiencia de sus operaciones tanto en tiempo como en espacio.

Concluimos el análisis de los sistemas en el Capítulo 7, con uno que se

deriva de forma directa del Problema de "Aprender con Errores" (LWE) por sus siglas en inglés. Una de las características fundamentales de este sistema es que, a diferencia de los dos anteriores, presenta una prueba de seguridad. Para poder presentar dicha característica, se introduce primero el problema LWE y su dificultad computacional para distintas variantes del mismo, así como también nociones de la teoría de la indistinguibilidad computacional.

Finalizamos el trabajo en el Capítulo 8 con un resumen de los resultados presentados junto con las conclusiones principales del trabajo, comentando también las líneas de trabajo futuro que surgen del mismo.

Capítulo 2

Fundamentos teóricos

En este capítulo se presentan nociones matemáticas básicas necesarias para el desarrollo del trabajo. Comenzamos introduciendo los conceptos utilizados en criptografía y seguridad, para luego recordar las definiciones sobre espacios vectoriales, dado que los reticulados presentan varias similitudes estructurales con estos objetos matemáticos. Concluimos con las nociones básicas de la teoría de la probabilidad. La exposición se basa principalmente en los contenidos e ideas de [9], [1] y [10].

2.1. Complejidad Computacional

Comenzamos definiendo la notación que será usada en el trabajo para caracterizar el comportamiento asintótico de las funciones consideradas.

Definición 1. *Notación asintótica*
Sean $f, g : \mathbb{N} \to \mathbb{R}^+ \cup \{0\}$ *funciones. Entonces decimos que,*

- $f(n) \in O(g(n))$ *si existe una constante* $c \in \mathbb{R}^+$ *y un* $n_0 \in \mathbb{N}$ *tales que*

$$\forall n \geq n_0,\ f(n) \leq c \cdot g(n)$$

- $f(n) \in \Omega(g(n))$ *si existe una constante* $c \in \mathbb{R}^+$ *y un* $n_0 \in \mathbb{N}$ *tales que*

$$\forall n \geq n_0,\ f(n) \geq c \cdot g(n)$$

- $f(n) \in \Theta(g(n))$ *si* $f(n) \in O(n) \cap \Omega(n)$
- $f(n) \in o(g(n))$ *si* $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

El modelo matemático formal de cómputo que suele utilizarse para analizar la complejidad de problemas computacionales es el de la Máquina de Turing, dada su simpleza relativa y la característica de que es posible la simulación de otros modelos mediante esta, sin perder en el proceso una cantidad significativa de eficiencia [9]. En particular, una Máquina de Turing es capaz de simular un lenguaje de programación de propósito general preservando eficiencia [9] y por tanto cuando en las siguientes definiciones nos refiramos a una Máquina de Turing determinada, basta pensar en ella (para los propósitos del trabajo) como un algoritmo que cumpla las mismas características.

En la teoría de la complejidad computacional, los problemas computacionales son modelados como subconjuntos $L \subseteq \{0,1\}^* = \bigcup_{n=0}^{\infty} \{0,1\}^n$ llamados lenguajes, donde para todo $x \in \{0,1\}^*$, o bien $x \in L$ o $x \notin L$. Luego, dado un lenguaje $L \subseteq \{0,1\}^*$ la versión de *decisión* del problema consiste en dar un algoritmo $\mathcal{A}$ que decida el lenguaje, es decir que para toda $x \in \{0,1\}^*$, $\mathcal{A}(x) = 1 \Leftrightarrow x \in L$. Otra variante de problemas son los de *búsqueda*, donde se presenta una instancia, junto con una "estructura" determinada y el objetivo es encontrar dicha "estructura" en la instancia[1]. Veremos ejemplos concretos de ambas variantes en el Capítulo 4.

Luego, los distintos problemas computacionales pueden agruparse en clases de problemas. Las clases estándar consideradas en la teoría de la complejidad están definidas en función de problemas en sus variantes de decisión. Sin embargo, en general es posible considerar variantes de los problemas de búsqueda que los transforman en problemas de decisión [2], posibilitando por tanto la aplicación de la teoría.

Definimos ahora dos de las principales clases de complejidad: **P** y **NP**, junto con otras nociones vinculadas y de interés para el trabajo.

Definición 2. *Clase* **P**

La clase **P** *se define como el conjunto de todos los problemas de decisión que pueden ser decididos por un algoritmo (determinista) en tiempo polinomial. Es decir que, dado un lenguaje* $L \in \mathbf{P}$*, debe existir un algoritmo* $\mathcal{A}$ *con tiempo de*

[1] Por ejemplo, dado un grafo y una propiedad L, encontrar un subconjunto de sus vértices que cumpla la propiedad L.

ejecución $T(n) \in O(n^c)$ para un $c \geq 1$ constante, que decida L.

La idea que busca capturar la clase **P** es la de los problemas que pueden ser resueltos de forma "eficiente" [9]. Siguiendo esta línea tenemos,

Definición 3. *Algoritmo en tiempo polinomial*
Un algoritmo $\mathcal{A}$ ejecuta en tiempo polinomial, si existe un polinomio $p : \mathbb{N} \to \mathbb{N}$ tal que

$$\forall x \in \{0,1\}^* \text{ el cómputo de } \mathcal{A}(x) \text{ termina en a lo sumo } p(|x|) \text{ pasos.}$$

Donde $|x|$ denota el largo de la cadena de caracteres x.

De donde, por lo anterior,

Definición 4. *Algoritmo eficiente*
Un algoritmo es eficiente si ejecuta en tiempo polinomial.

Seguimos con la clase **NP** que está constituida por los problemas cuyas "soluciones" pueden ser verificadas de manera eficiente.

Definición 5. *Clase* **NP**
Un lenguaje de decisión $L \subseteq \{0,1\}^*$ está en **NP** *si existe un polinomio $p : \mathbb{N} \to \mathbb{N}$ y un algoritmo en tiempo polinomial $\mathcal{M}$ (llamado verificador de L) tal que para todo $x \in \{0,1\}^*$,*

$$x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)} \text{ tal que } \mathcal{M}(x,u) = 1$$

Si $x \in L$ y $u \in \{0,1\}^{p(|x|)}$ satisfacen $\mathcal{M}(x,u) = 1$, decimos que u es un certificado para x (con respecto al lenguaje L y al algoritmo $\mathcal{M}$).

Intuitivamente, esta clase busca capturar los problemas para los que existen pruebas "cortas" para verificar la pertenencia al lenguaje[1].

En particular, tenemos **P** $\subseteq$ **NP** dado que el certificado puede ser una cadena de largo 0 (cadena vacía), que se corresponde con el caso $p(|x|) = 0$. Actualmente, no se ha logrado probar si **P** = **NP**, siendo esta la pregunta abierta más relevante de la teoría de la complejidad. Sin embargo, hay mucha evidencia que apunta a que **P** $\neq$ **NP** aunque aún no se ha podido probar

[1] Por ejemplo, dado un grafo y una propiedad T sobre sus vértices, una prueba o certificado puede ser un subconjunto de vértices del grafo que cumpla la propiedad T.

[9]. Esto implica que con frecuencia, se den resultados en base a esta u otras conjeturas, estudiadas pero no resueltas, asumiendo la dificultad de ciertos problemas, cuya resolución implican por ejemplo la igualdad entre las dos clases.

Introducimos ahora la definición de reducción en tiempo polinomial, que permite vincular la dificultad de dos problemas computacionales.

Definición 6. *Reducción en tiempo polinomial*
Un lenguaje $L \in \{0,1\}^*$ *se reduce en tiempo polinomial al lenguaje* $L' \in \{0,1\}^*$*, denotado* $L \leq_p L'$*, si existe una función computable en tiempo polinomial* $f : \{0,1\}^* \to \{0,1\}^*$ *tal que*

$$\forall x \in \{0,1\}^*,\ x \in L \Leftrightarrow f(x) \in L'$$

Es decir, podemos ver la reducción como una transformación en tiempo polinomial, que mapea instancias que pertenecen a L a instancias que también pertenecen a L' y análogamente mapea la no pertenencia de instancias en L a la no pertenencia en L'.

Entonces, si se cumple $L \leq_p L'$ tenemos un procedimiento eficiente para transformar instancias de L en instancias de L' y por tanto si existe un algoritmo eficiente para resolver las instancias de L', tendremos también un algoritmo eficiente para resolver las de L. Es este el motivo por el que decimos que en este caso, L' es al menos tan difícil como L.

Usando la noción del párrafo anterior, caracterizamos ahora a los problemas "más dificiles" de la clase **NP** como los problemas **NP**-completos.

Definición 7. *Problema* **NP***-difícil*
Un lenguaje L' *es* **NP***-difícil si para todo* $L \in$ **NP**, $L \leq_p L'$

Definición 8. *Problema* **NP***-completo*
Un lenguaje L' *es* **NP***-completo si es* **NP***-difícil y* $L' \in$ **NP**

Siguiendo lo anterior, un problema es **NP**-difícil si es al menos tan difícil como cualquier problema en **NP**. Si además un problema de este tipo pertenece a **NP**, tenemos entonces que es uno de los problemas "más difíciles" de toda la clase. La importancia de estos problemas viene dada por el siguiente hecho:

Teorema 1. *[9] Sea L un lenguaje* **NP***-completo. Entonces,*

$$L \in \mathbf{P} \Leftrightarrow \mathbf{P} = \mathbf{NP}$$

Entonces dado que se asume $\mathbf{P} \neq \mathbf{NP}$, resulta poco probable que si un problema es **NP**-completo este admita un algoritmo eficiente para resolver todas sus instancias. Por esto, consideramos que este tipo de problemas son computacionalmente difíciles en el caso general.

Antes de continuar con los fundamentos de criptografía y seguridad, comentaremos el tipo de algoritmos que utilizaremos en el trabajo. Por defecto, trataremos con algoritmos probabilísticos o aleatorizados. Es decir, algoritmos que pueden tener acceso a una fuente de bits aleatorios sin sesgo en cada paso de ejecución. Una forma equivalente de expresar lo anterior, es considerar que el algoritmo, además de sus entradas, tiene acceso a una cinta extra con largo suficiente de bits aleatorios generados independientemente y con distribución uniforme, que puede emplear en su ejecución. En este sentido, diremos que un algoritmo ejecuta en Tiempo Polinomial Probabilístico (TPP)[1] si ejecuta en tiempo polinomial y además para su ejecución, tiene acceso a una fuente de bits uniformes.

2.2. Criptografía y Seguridad

En el trabajo se utiliza la noción de *seguridad computacional* [1], es decir que tomaremos en cuenta en las definiciones de seguridad los límites computacionales del atacante, o equivalentemente de los algoritmos que ejecuta y permitimos una probabilidad de error pequeña. Esta es la noción utilizada actualmente como estándar para todos los propósitos criptográficos [1].

En particular, tenemos que:

- Se garantiza la seguridad solo contra adversarios eficientes que ejecutan por una cantidad de tiempo factible. Es decir que, dado suficiente tiempo o recursos computacionales, un atacante podría ser capaz de violar la seguridad de un sistema. Si logramos que los recursos requeridos para quebrar el sistema sean mayores que los disponibles para cualquier atacante realista, entonces en la práctica el esquema es "inquebrantable".

[1] PPT por sus siglas en inglés.

- Un adversario puede potencialmente tener éxito con muy baja probabilidad. Si podemos lograr que esta sea exponencialmente pequeña, no es necesario preocuparse por este hecho.

En lo que sigue definimos rigurosamente los conceptos antes explicados. Para esto seguimos el *enfoque asintótico*, basado en nociones de la teoría de la complejidad computacional, que comentamos en la Sección 2.1.

Con este propósito, se introduce un *parámetro de seguridad*, denotado por n, que toma valores enteros y parametriza tanto los esquemas criptográficos como los participantes involucrados. En este sentido, se incluyen tanto a los participantes honestos de la comuniciación como a los distintos atacantes. Se asume que el parámetro de seguridad es conocido por cualquier adversario y consideraremos tanto el tiempo de ejecución del adversario como su probabilidad de éxito como funciones de n.

Entonces, formalizando tenemos lo siguiente:

- Consideramos que un adversario es "eficiente" si este ejecuta un algoritmo probabilístico en tiempo polinomial en n. Es decir, que existe un polinomio $p : \mathbb{N} \to \mathbb{N}$ tal que el adversario ejecuta en un tiempo que es a lo sumo $p(n)$.

- También requerimos que para tener eficiencia en la práctica, los participantes honestos ejecuten en tiempo polinomial. Destacamos que un adversario puede poseer muchos más recursos y por tanto, ejecutar por un tiempo mucho mayor que los participantes honestos de la comunicación.

- Asociamos la definición de "probabilidad pequeña" de éxito con probabilidades que son menores que cualquier inverso de un polinomio en n y las llamamos *negligibles* (Definición 9).

Utilizando la noción de TPP, podemos definir seguridad asintótica de forma genérica como sigue [1]:

Un esquema es *seguro* si cualquier adversario en TPP tiene éxito en quebrar el esquema con a lo sumo probabilidad negligible.

Esta noción se dice asintótica, ya que la seguridad depende del comportamiento del esquema para valores suficientemente grandes de n.

De lo anterior, notamos que el parámetro n sirve para "ajustar" la seguridad del sistema hasta un nivel deseado. Notar sin embargo, que dicho ajuste incrementa también el tiempo requerido para ejecutar el esquema, así como el largo de las claves utilizadas. Los participantes honestos querrán que n sea lo más pequeño posible, sujeto a un nivel de seguridad que permita protección contra los ataques que les preocupan. Este enfoque permite que los participantes honestos se defiendan en un contexto donde el poder computacional disponible puede aumentar.

Como usualmente se mide el tiempo de ejecución de un algoritmo en función del largo de su entrada, puede ser conveniente utilizar como entrada la representación unaria del parámetro de seguridad, 1^n que representa una cadena de n unos. Permitiremos entonces, que el tiempo de ejecución de los algoritmos sea polinomial en el largo total de sus parámetros de entrada.

La noción de algoritmo probabilista que presentamos anteriormente es necesaria principalmente por dos motivos: primero, porque como veremos en este trabajo, la aleatoriedad es un ingrediente clave para el desarrollo de la criptografía y segundo porque esta noción es utilizada en la práctica y le da a los atacantes poder adicional. Como nuestro fin es modelar todos los ataques realistas, se opta por una noción más liberal de computación eficiente [1].

Precisamos a continuación la noción de función negligible.

Definición 9. *[1] Función negligible*
Una función $f : \mathbb{N} \to \mathbb{R}^+ \cup \{0\}$ *se dice* negligible *si*

$\forall$ *polinomio positivo* p, $\exists n_0 \in \mathbb{N}$ *tal que* $\forall n \in \mathbb{N}$ *con* $n > n_0$ *se cumple*
$$f(n) < \frac{1}{p(n)}.$$

una formulación equivalente se da si,

$\forall$ *constante* $c \in \mathbb{N}$, $\exists n_0 \in \mathbb{N}$ *tal que* $\forall n > n_0$ *se cumple* $f(n) < n^{-c}$.

Definimos ahora el objeto principal de estudio, un criptosistema de clave pública.

Definición 10. *[1] Criptosistema de clave pública*
Un esquema de encriptado de clave pública es una 3-upla de algoritmos probabilísticos que ejecutan en tiempo polinomial $\Pi = (\boldsymbol{Gen}, \boldsymbol{Enc}, \boldsymbol{Dec})$ *tales que:*

- *El algoritmo generador de claves* ***Gen*** *toma como entrada un parámetro de seguridad* 1^n *y retorna como salida un par de claves* (pk,sk)*. Nos referimos a la primera componente del par como clave pública y a la segunda como clave privada o secreta.*

- *El algoritmo de encriptado* ***Enc*** *toma como entrada la clave pública* pk *y un mensaje* m *de un espacio de mensajes determinado (que puede depender de* pk*). Produce como salida un texto cifrado* c *y notamos* $\mathrm{c} \leftarrow \boldsymbol{Enc}_{\mathrm{pk}}(\mathrm{m})$*, al texto cifrado que se obtiene de ejecutar el algoritmo de encriptado sobre el mensaje m usando la clave pública* pk*.*

- *El algoritmo determinista de desencriptado* ***Dec*** *toma como entrada una clave privada* sk *y un texto cifrado* c*, produciendo como salida un mensaje* m *o un símbolo especial* $\perp$ *denotando una falla. Notamos* $\mathrm{m} := \boldsymbol{Dec}_{\mathrm{sk}}(\mathrm{c})$*, al proceso de obtener un mensaje* m *ejecutando el algoritmo de desencriptado sobre un texto cifrado* c *con clave privada* sk*.*

Se requiere que, excepto por sucesos de probabilidad negligible sobre la salidas de $\boldsymbol{Gen}(1^n)$*,* (pk,sk)*, se tenga* $\boldsymbol{Dec}_{\mathrm{sk}}(\boldsymbol{Enc}_{\mathrm{pk}}(\mathrm{m})) = \mathrm{m}$ *para todo mensaje válido* m*. Si el sistema cumple esta condición, decimos que es* correcto*.*

De la definición notamos que la clave pública es usada para el encriptado, mientras que la clave privada para el proceso de desencriptado. La clave pública, como su nombre indica, está ampliamente distribuida de modo tal que cualquier entidad puede encriptar mensajes para el participante que generó la clave. La clave secreta debe ser mantenida en privado por el receptor para poder garantizar la seguridad del sistema.

En general, tomamos el espacio de mensajes como $\mathbb{B}^n = \{0,1\}^n$ o $\mathbb{B}^* = \{0,1\}^*$ y usualmente este no depende de la clave pública. Si este no es el caso para el espacio de mensajes, debe poder especificarse una codificación de los objetos del conjunto en cadenas de bits. Esta codificación debe poder ser computada de manera eficiente y reversible, de modo que el receptor pueda recuperar la cadena de bits encriptada.

2.2.1. Funciones de un sentido y vínculo con sistemas de clave pública

Podemos vincular la noción de criptosistema con otra construcción utilizada en el área, denominada función de un sentido (*one-way function* en inglés). Esta definción busca capturar la idea de que una función sea fácil de computar, pero difícil de invertir [1].

Formalizamos esta construcción a continuación, siguiendo [1].

Definición 11. *Familia de funciones*
Una tupla $\Pi = (\texttt{Gen}, \texttt{Sampl}, f)$ *de algoritmos en TPP es una* familia de funciones *si se cumple*

1. *El algoritmo de generación de parámetros* `Gen`*, toma como entrada* 1^n *y da como salida parámetros* I *con* $|I| \geq n$*. Para cada* I *definido por* `Gen`*, se definen conjuntos* $\mathcal{D}_I$ *y* $\mathcal{R}_I$ *que constituyen los dominios y rangos de una función* f_I*.*

2. *El algoritmo de muestreo* `Sampl`*, tomando* I *como entrada, retorna un elemento de* $\mathcal{D}_I$ *uniformemente distribuido.*

3. *El algoritmo determinístico de evaluación* f*, dado un* I *como entrada y un* $x \in \mathcal{D}_I$*, da como salida un elemento* $y \in \mathcal{R}_I$*. Notamos* $y := f_I(x)$*.*

Definimos ahora un experimento que captura las ideas antes mencionadas.

Definición 12. *Experimento de inversión* $\texttt{Invert}_{\mathcal{A},\Pi}(n)$
Sea Π *una familia de funciones y* $\mathcal{A}$ *un algoritmo. Definimos el experimento de inversión* $\texttt{Invert}_{\mathcal{A},\Pi}(n)$ *como sigue:*

1. *Se ejecuta* $\texttt{Gen}(1^n)$ *para obtener un* I *y luego ejecutamos* $\texttt{Sampl}(I)$ *para obtener un* $x \in \mathcal{D}_I$ *uniforme. Finalmente computamos* $y := f_I(x)$*.*

2. *Se pasan* I *e* y *como parámetros a* $\mathcal{A}$*, que da como salida* x'*.*

3. *La salida del experimento es 1 si* $f_I(x') = y$*, es decir si* x' *es una preimagen de* y*.*

En base a esto definimos la noción de interés,

Definición 13. *Familia de funciones de un sentido*
Una familia de funciones $\Pi = (\texttt{Gen}, \texttt{Sampl}, f)$ *es* de un sentido *si para todo algoritmo* $\mathcal{A}$ *en TPP, existe una función negligible* $\texttt{negl}$ *tal que*

$$Pr(\texttt{Invert}_{\mathcal{A},\Pi}(n) = 1) \leq \texttt{negl}(n)$$

Si además de I el algoritmo `Gen` da como salida información adicional td que permite que la inversión de f_I se realice de forma eficiente, decimos que Π es una familia de funciones de un sentido con *puerta trasera* (*trapdoor* en inglés). Entonces tenemos

Definición 14. *Familia de funciones con puerta trasera*
Una tupla de algoritmos polinomiales, $\Pi = (\texttt{Gen}, \texttt{Sampl}, f, \texttt{Inv})$ *es una familia de funciones con puerta trasera si:*

- *El algoritmo de generación de parámetros* $\texttt{Gen}$*, tomando como entrada* 1^n *da como salida* (I, td) *con* $|I| \geq n$*. Cada valor de* I *define conjuntos* $\mathcal{D}_I$ *y* $\mathcal{R}_I$ *que constituyen los dominios y rangos de una función* $f_I : \mathcal{D}_I \longrightarrow \mathcal{R}_I$*.*

- *Sea* $\texttt{Gen}_1$ *el algoritmo que resulta de ejecutar* $\texttt{Gen}$ *y dar como salida únicamente* I*. Entonces* $(\texttt{Gen}_1, \texttt{Sampl}, f)$ *es una familia de funciones de un sentido.*

- *Sea* (I, td) *la salida de* $\texttt{Gen}(1^n)$*. El algoritmo de inversión determinista* $\texttt{Inv}$*, tomando como entrada* td *y un* $y \in \mathcal{R}_I$ *da como salida* $x \in \mathcal{D}_I$*. Notamos* $x := \texttt{Inv}_{td}(y)$*. Requerimos que a menos de una probabilidad negligible sobre las salidas de* $\texttt{Gen}(1^n), (I, \text{td})$*, y elección uniforme de* $x \in \mathcal{D}_I$*,*

$$\texttt{Inv}_{td}(f_I(x)) = x$$

Para poder utilizar una familia de funciones de un sentido como un sistema de encriptado, es necesario dar una forma de embeber el mensaje a ser comunicado en los argumentos de estas funciones [11]. Una vez que contamos con este mecanismo asignamos:

- La salida de $\texttt{Gen}(1^n)$, (I, td) se corresponde con el par (pk,sk).

- Para encriptar un mensaje m utilizamos la clave pública I en conjunto con la especificación de la función f, obteniendo $c = f_I(m)$.

- Finalmente para desencriptar utilizamos la puerta trasera td, para poder invertir la función f de manera eficiente $m = Inv_I(c, \text{td})$.

2.2.2. Definiciones básicas de Seguridad

Establecemos ahora las nociones que utilizaremos para razonar sobre la seguridad de los sistemas de manera rigurosa, siguiendo la formulación de [1].

Cualquier definición de seguridad está compuesta por dos elementos: un modelo de amenazas, o equivalentemente del poder que tiene el adversario y un objetivo de seguridad, que usualmente se especifica determinando qué se considera "quebrar" el sistema.

La primera definición que introduciremos, consiste en el modelo de amenaza más sencillo, donde tenemos un adversario pasivo que únicamente observa en el canal de comunicación el encriptado de un único mensaje. Este adversario debe ejecutar en tiempo polinomial pero no se asume ninguna hipótesis en cuanto a las estrategias que puede utilizar para quebrar el texto encriptado considerado. De este modo, la definición asegurará protección contra cualquier adversario en las condiciones anteriores, sin importar el algoritmo concreto que ejecuta.

Al igual que para el caso de funciones de un sentido (Sección 2.2.1), la definición se da en función de un experimento que definimos a continuación.

Definición 15. *Experimento de indistinguibilidad de atacante pasivo* $PubK^{eav}_{\mathcal{A},\Pi}(n)$

1. *Se ejecuta* $\mathtt{Gen}(1^n)$ *para obtener* (pk,sk).
2. *Se le da al adversario* $\mathcal{A}$ *la clave pública* pk *y este da como salida un par de mensajes* m_0 *y* m_1 *del mismo largo, dentro del espacio de mensajes.*
3. *Se selecciona un bit* $b \in \{0,1\}$ *(desconocido para el atacante* $\mathcal{A}$*) de manera uniforme, se computa un texto cifrado* $\mathrm{c} \leftarrow \mathtt{Enc}_{\mathrm{pk}}(\mathrm{m}_b)$ *y este es dado a* $\mathcal{A}$*. Decimos que c es el texto cifrado desafío.*
4. $\mathcal{A}$ *da un bit* b' *como salida. El resultado del experimento es 1 si* $b' = b$ *o 0 si no. En el primer caso decimos que* $\mathcal{A}$ *tiene éxito.*

Notamos que el único requisito sobre los mensaje es que tengan el mismo largo, si bien está implícito que si $\mathcal{A}$ ejecuta en tiempo polinomial en n, en-

tonces el largo de los mensajes también debe cumplir esta restricción.

Basados en este experimento tenemos,

Definición 16. *Un sistema de clave pública* $\Pi = (\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ *presenta indistinguibilidad de encriptado en presencia de un adversario pasivo, si para todo adversario* $\mathcal{A}$ *en TPP, existe una función negligible* $\mathtt{negl}$ *tal que*

$$Pr(PubK^{eav}_{\mathcal{A},\Pi}(n) = 1) \leq \tfrac{1}{2} + \mathtt{negl}(n)$$

Informalmente, esta definición garantiza que el encriptado de un mensaje no revela información sobre el mismo a un atacante pasivo, no pudiendo este distinguir entre ellos con probabilidad significativamente mejor que en el caso de adivinar de forma aleatoria uniforme. En otras palabras, lo mejor que puede hacer $\mathcal{A}$ es tirar una moneda insesgada y en base a esto intentar "adivinar" el bit.

El hecho de que $\mathcal{A}$ tenga acceso a la clave pública, le da acceso a un oráculo de encriptación de forma automática, ya que el algoritmo de encriptado también es públicamente conocido. De esta forma, en el contexto de sistemas de clave pública, la definición anterior puede ser referida como seguridad bajo ataque de texto plano elegido (seguridad CPA por sus siglas en inglés). Entonces diremos que si un sistema de clave pública tiene encriptados indistinguibles en la presencia de un adversario pasivo, entonces es CPA-seguro. También podemos decir que el sistema es semánticamente seguro [1].

Una formulación equivalente para seguridad CPA es que todo adversario en TPP se comporte "de la misma manera" si está en presencia de un encriptado de m_0 o de m_1. Como $\mathcal{A}$ da como salida un único bit, "comportarse de la misma manera" en este contexto significa que da la misma salida (digamos 1) con "casi" la misma probabilidad para ambos casos. Para formalizar estas nociones, sea $PubK^{eav}_{\mathcal{A},\Pi}(n, b)$ el experimento análogo al definido en la Definición 15, excepto que el bit b está dado (en lugar de ser tomado de manera uniforme). Sea entonces $\text{out}_{\mathcal{A}}(PubK^{eav}_{\mathcal{A},\Pi}(n, b))$ el bit b' de salida del algoritmo $\mathcal{A}$ en el experimento $PubK^{eav}_{\mathcal{A},\Pi}(n, b)$. Entonces,

Definición 17. *Un sistema de clave pública* $\Pi = (\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ *presenta indistinguibilidad de encriptado en presencia de un adversario pasivo, si para todo adversario* $\mathcal{A}$ *en TPP, existe una función negligible* $\mathtt{negl}$ *tal que*

$$|\Pr\big(out_{\mathcal{A}}(PubK^{eav}_{\mathcal{A},\Pi}(n,0)) = 1\big) - \Pr\big(out_{\mathcal{A}}(PubK^{eav}_{\mathcal{A},\Pi}(n,1)) = 1\big)| \leq \mathtt{negl}(n)$$

La formulación equivalente dada en la Definición 17 nos permitirá vincular de manera más sencilla la seguridad CPA de un sistema con el concepto de indistinguibilidad computacional, que trataremos en el Capítulo 7.

Comentamos dos ataques de texto plano elegido (CPA) que resultan realistas, para motivar las Definiciones 16 y 17. Como primer escenario [1], consideremos un atacante que escribe en una terminal de uso compartido, que encripta dichos datos y los envía a un servidor remoto utilizando una clave acordada con el servidor (que el atacante no conoce). Aquí el atacante tiene acceso a los textos cifrados correspondientes a distintos mensajes que él escribe (por ejemplo porque observa el tráfico enviado al servidor mientras ingresa el texto en la terminal), sin embargo queremos que cuando otro usuario utilice la terminal, el esquema de encriptado (utilizando la misma clave) siga siendo seguro para el nuevo usuario. Es decir, que si el atacante ve los datos encriptados que se envían al servidor cuando el nuevo usuario escribe en la terminal, no pueda saber qué está escribiendo el nuevo usuario.

Como segundo escenario, analicemos el caso de una votación electrónica entre una serie de candidatos conocidos, donde los distintos votos son encriptados y enviados a una autoridad central para su contabilización. Es deseable que un atacante no pueda identificar qué votos corresponden al mismo candidato, ya que con otra información del contexto (ubicación geográfica del centro de votos o personas que votaron), podría inferir el candidato correspondiente a dicho grupo de votos. En ambos escenarios, la propiedad de seguridad semántica resulta fundamental para asegurar la información de manera correcta. Una propiedad esencial para asegurar estos sistemas es el uso de encriptado aleatorio, donde se agrega en el proceso de encriptado un elemento aleatorio, que permite asociar a un mismo mensaje varios textos cifrados posibles. De aquí que si un sistema presenta un proceso de encriptado determinista, el sistema no será CPA seguro. Profundizaremos estas ideas al estudiar la seguridad de los sistemas expuestos en los Capítulos 5,6 y 7.

Concluimos esta sección con las pruebas de seguridad por Reducción.

Pruebas de seguridad por Reducción

Un método formal para probar que una construcción dada es computacionalmente segura consiste en basarse en la hipótesis (ampliamente aceptada) de que otro problema computacional o primitiva de bajo nivel criptográfico es segura. Usualmente esto requiere asumir conjeturas no probadas de la teoría de complejidad computacional como axiomas, por ejemplo $\mathbf{P} \neq \mathbf{NP}$, como explicamos en la Sección 2.1. La comunidad científica cree que la evidencia disponible que los respalda es suficientemente fuerte y por tanto se consideran un indicio de que en efecto las construcciones en ellas basadas son robustas.

La prueba de que una construcción criptográfica es segura, siempre y cuando el problema subyacente es difícil en general se realiza dando una reducción, que muestra un procedimiento para transformar cualquier adversario eficiente $\mathcal{A}$ que tenga éxito en "quebrar" la construcción, en un adversario eficiente $\mathcal{A}'$ que tenga éxito en la resolución del problema que se asume difícil [1].

Damos ahora la explicación en alto nivel del proceso, presentada en [1], que guarda una relación estrecha con la noción de reducción en tiempo polinomial vista en la Sección 2.1. Comenzamos asumiendo que un problema $\mathtt{X}$ no puede ser resuelto, en un sentido definido precisamente, por ningún algoritmo en tiempo polinomial, a menos de una probabilidad negligible. Queremos probar que una construcción Π, por ejemplo un criptosistema, es seguro, en algún sentido definido precisamente. La prueba consiste en los siguientes pasos:

1. Fijar un adversario eficiente $\mathcal{A}$ que ataca Π, denotando por $\epsilon(n)$ su probabilidad de éxito.

2. Construir otro algoritmo eficiente $\mathcal{A}'$, que llamaremos "reducción", que intenta resolver el problema $\mathtt{X}$ usando $\mathcal{A}$ como subrutina. Observar que para $\mathcal{A}'$, $\mathcal{A}$ funciona como una "caja negra" (o subrutina ya implementada), no conociendo el primero ningún detalle de funcionamiento interno del segundo. Entonces, dada como entrada una instancia $\mathtt{x}$ del problema $\mathtt{X}$, genera una instancia de Π para $\mathcal{A}$ tal que

    a. Desde la perspectiva de $\mathcal{A}$, el algoritmo está interactuando con Π, es decir que no debe percibir a $\mathcal{A}'$ cuando es ejecutado como subrutina de este.

b. Si $\mathcal{A}$ tiene éxito en "quebrar" la instancia de Π que generó $\mathcal{A}'$, esto debe permitir que $\mathcal{A}'$ resuelva la instancia $\mathtt{x}$ dada, con al menos probabilidad inversa de un polinomio, $\frac{1}{p(n)}$.

3. Los dos subpuntos a. y b. implican que $\mathcal{A}'$ resuelve $\mathtt{X}$ con probabilidad $\frac{\epsilon(n)}{p(n)}$. Si $\epsilon(n)$ es no negligible, entonces tampoco es $\frac{\epsilon(n)}{p(n)}$. Además, si $\mathcal{A}$ es eficiente, obtenemos un algoritmo eficiente $\mathcal{A}'$ que resuelve $\mathtt{X}$ con probabilidad no neglibile, contradiciendo la hipótesis de que $\mathtt{X}$ era un problema difícil.

4. Dado nuestro supuesto sobre $\mathtt{X}$, concluimos que ningún adversario eficiente $\mathcal{A}$ puede tener éxito en el quebrado de Π con probabilidad no neglibile. Dicho de otro modo, Π es computacionalmente seguro.

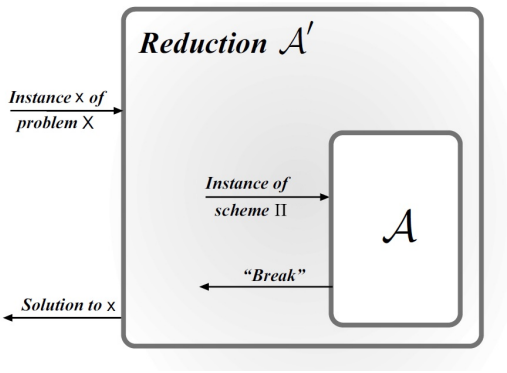


Figura 2.1: Proceso de reducción en alto nivel [1]

La Figura 2.1 ilustra el proceso antes descrito. Formalizamos la idea de prueba de seguridad por reducción mediante la Definición 6, diciendo que el problema $\mathtt{X}$ se reduce en tiempo polinomial al problema de "quebrar" el sistema Π. Donde la noción de "quebrar" el sistema está bien definida.

2.3. Fundamentos matemáticos

Comenzamos recordando nociones básicas de álgebra lineal que guardan relación con los reticulados.

2.3.1. Espacios vectoriales

Restringimos la exposición a espacios vectoriales sobre el cuerpo de los números reales $\mathbb{R}$, con las operaciones usuales de suma y multiplicación por escalar, ya que esto resulta suficiente para desarrollar los conceptos del trabajo. Notaremos los escalares con letras minúsculas, los vectores (columna) en **negrita** y las matrices tanto como los conjuntos con letras mayúsculas.

Definición 18. *Espacio Vectorial*
Sea $V \subseteq \mathbb{R}^m$ conjunto no vacío con $m \in \mathbb{N} - \{0\}$ y sean $+ : V \times V \to V$, $\cdot : \mathbb{R} \times V \to V$ las operaciones de suma y multiplicación por escalar usuales.

$(V, \mathbb{R}, +, \cdot)$ es un espacio vectorial real si se verifican las siguientes propiedades:

- *(Asociatividad de la suma)* $\forall \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w} \in V,\ (\boldsymbol{u} + \boldsymbol{v}) + \boldsymbol{w} = \boldsymbol{u} + (\boldsymbol{v} + \boldsymbol{w})$
- *(Conmutatividad de la suma)* $\forall \boldsymbol{u}, \boldsymbol{v} \in V,\ \boldsymbol{u} + \boldsymbol{v} = \boldsymbol{v} + \boldsymbol{u}$
- *(Existencia de neutro suma)* $\exists \boldsymbol{0} \in V$ *tal que* $\forall \boldsymbol{v} \in V,\ \boldsymbol{v} + \boldsymbol{0} = \boldsymbol{0} + \boldsymbol{v} = \boldsymbol{v}$
- *(Existencia de inverso suma)* $\forall \boldsymbol{v} \in V\ \exists (\boldsymbol{-v}) \in V$ *tal que* $\boldsymbol{v} + (\boldsymbol{-v}) = \boldsymbol{0}$
- $\forall \alpha, \beta \in \mathbb{R},\ \forall \boldsymbol{v} \in V\ \ \alpha(\beta \boldsymbol{v}) = (\alpha\beta)\boldsymbol{v}$
- *(Existencia de neutro multiplicación)* $\exists \boldsymbol{1} \in \mathbb{R}$ *tal que* $\forall \boldsymbol{v} \in V\ \ \boldsymbol{1v} = \boldsymbol{v1} = \boldsymbol{v}$
- *(Distributividad de multiplicación con respecto a suma en $\mathbb{R}$)*
  $\forall \alpha, \beta \in \mathbb{R}\ \forall \boldsymbol{v} \in V\ \ (\alpha + \beta)\boldsymbol{v} = \alpha \boldsymbol{v} + \beta \boldsymbol{v}$
- *(Distributividad de multiplicación con respecto a suma vectorial)*
  $\forall \alpha \in \mathbb{R}\ \forall \boldsymbol{u}, \boldsymbol{v} \in V\ \ \alpha(\boldsymbol{u} + \boldsymbol{v}) = \alpha \boldsymbol{u} + \alpha \boldsymbol{v}$

Definición 19. *Subespacio vectorial*
Sea $(V, \mathbb{R}, +, \cdot)$ espacio vectorial. Un subconjunto $S \subset V$ es subespacio vectorial de V si

- $S \neq \emptyset$
- $\forall \boldsymbol{v}, \boldsymbol{w} \in S,\ \boldsymbol{v} + \boldsymbol{w} \in S$

- $\forall \alpha \in \mathbb{R}\ \forall \boldsymbol{v} \in S,\ \alpha \boldsymbol{v} \in S$

Es decir si S es no vacío y cerrado bajo las operaciones de suma y multiplicación por escalar.

Seguimos con el concepto de combinación lineal e independencia de un conjunto de vectores.

Definición 20. *Combinación Lineal*
Sean $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k \in V$.
Una combinación lineal de estos vectores viene dada por

$$\boldsymbol{w} = \alpha_1 \boldsymbol{v}_1 + \alpha_2 \boldsymbol{v}_2 + \cdots + \alpha_k \boldsymbol{v}_k \text{ donde } \alpha_1, \alpha_2, \ldots, \alpha_k \in \mathbb{R}.$$

Llamamos subespacio generado al conjunto de todas las combinaciones lineales de un conjunto de vectores $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k \in V$, y lo denotamos

$$[\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k\}] = \operatorname{span}(\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k\}) = \{\boldsymbol{w} \in V : \boldsymbol{w} = \alpha_1 \boldsymbol{v}_1 + \alpha_2 \boldsymbol{v}_2 + \cdots + \alpha_k \boldsymbol{v}_k \text{ donde } \alpha_1, \alpha_2, \ldots, \alpha_k \in \mathbb{R}\}$$

Definición 21. *Independencia Lineal*
Un conjunto de vectores $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k \in V$ se dice linealmente independiente (l.i.) si la única forma de obtener

$$\alpha_1 \boldsymbol{v}_1 + \alpha_2 \boldsymbol{v}_2 + \cdots + \alpha_k \boldsymbol{v}_k = \boldsymbol{0},$$

es tener $\alpha_1 = \alpha_2 = \cdots = \alpha_k = 0$. Contrariamente, el conjunto se dice linealmente dependiente si podemos obtener el vector nulo con una combinación lineal donde al menos uno de los escalares sea no nulo.

Continuamos con la caracterización de conjuntos que generan el espacio vectorial.

Definición 22. *Generador*
Dado V $\mathbb{R}$-espacio vectorial, decimos que $X \subseteq V$ es un generador de V si

$$\forall \boldsymbol{v} \in V,\ \boldsymbol{v} \text{ es combinación lineal de los elementos de } X.$$

Definición 23. *Base*
Dado V $\mathbb{R}$-espacio vectorial, decimos que $B \subseteq V$ es una base de V si es un conjunto linealmente independiente que genera V.

Proposición 1. *Sea $V \subseteq \mathbb{R}^m$ espacio vectorial. Entonces,*

- *Existe una base de V.*
- *Dos bases de V tienen necesariamente la misma cardinalidad.*
- *Sea $B = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m\}$ una base de V y sea $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_m\}$ otro conjunto de m vectores de V. Escribimos cada $\boldsymbol{w}_j$ con $j \in \{1, \ldots, m\}$ como una combinación lineal de los elementos de la base,*

$$\begin{aligned}
\boldsymbol{w}_1 &= \alpha_{11}\boldsymbol{v}_1 + \alpha_{12}\boldsymbol{v}_2 + \cdots + \alpha_{1m}\boldsymbol{v}_m \\
\boldsymbol{w}_2 &= \alpha_{21}\boldsymbol{v}_1 + \alpha_{22}\boldsymbol{v}_2 + \cdots + \alpha_{2m}\boldsymbol{v}_m \\
&\vdots \\
\boldsymbol{w}_m &= \alpha_{m1}\boldsymbol{v}_1 + \alpha_{m2}\boldsymbol{v}_2 + \cdots + \alpha_{mm}\boldsymbol{v}_m
\end{aligned}$$

Entonces $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_m\}$ también es base de V si y solo si

$$\det \begin{pmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1m} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mm} \end{pmatrix} \neq 0$$

Reescribiendo el sistema del último punto de la proposición en forma matricial tenemos,

$$\begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_m \end{pmatrix} = \begin{pmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1m} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mm} \end{pmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_m \end{pmatrix}$$

y como el determinante de la matriz de coeficientes es no nulo, sabemos que es invertible y por tanto

$$\begin{pmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1m} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mm} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_m \end{pmatrix} = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_m \end{pmatrix}$$

De aquí queda claro que podemos escribir cada $\mathbf{v}_i$ de la base como combinación lineal de los $\mathbf{w}_j$ y por tanto $\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m\}$ también es un conjunto de vectores de cardinalidad mínima que genera el espacio.

Definición 24. *Dimensión*
Definimos la dimensión de un espacio vectorial $V \subseteq \mathbb{R}^m$ como el número de elementos de una base de V. Si B es una base de V, notamos $\dim(V) = |B|$ *a la dimensión de V.*

Observar que por la proposición anterior, la dimensión de un espacio vectorial es independiente de la base concreta elegida.

Presentamos ahora una noción de distancia en $\mathbb{R}^n$ y conceptos relacionados de medidas sobre el espacio.

Definición 25. *Producto interno usual*
Sean $\boldsymbol{v}, \boldsymbol{w} \in V \subseteq \mathbb{R}^m$ tales que $\boldsymbol{v} = (x_1, x_2, \ldots, x_m)^t$ y $\boldsymbol{w} = (y_1, y_2, \ldots, y_m)^t$. Definimos el producto interno (usual) entre $\boldsymbol{v}$ y $\boldsymbol{w}$ como el operador $\langle , \rangle : V \times V \to \mathbb{R}$ tal que

$$\langle \boldsymbol{v}, \boldsymbol{w} \rangle = x_1 y_1 + x_2 y_2 + \cdots + x_m y_m$$

En base a esto, definimos la noción de ortogonalidad de vectores.

Definición 26. *Dado $V \subseteq \mathbb{R}^m$, decimos que dos vectores $\boldsymbol{v}, \boldsymbol{w} \in V$ son ortogonales si $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$.*

Definición 27. *Norma Euclídea*
Dado $\boldsymbol{v} \in \mathbb{R}^m$ tal que $\boldsymbol{v} = (x_1, \ldots, x_m)^t$. Definimos la norma euclídea de $\boldsymbol{v}$ como

$$\|\boldsymbol{v}\| = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} = \sqrt{x_1^2 + x_2^2 + \cdots + x_m^2}$$

Vinculamos ahora el concepto de base y ortogonalidad de vectores.

Definición 28. *Base ortogonal y ortonormal*
Sea $B = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ una base de $V \subseteq \mathbb{R}^n$ espacio vectorial. Decimos que la base es ortogonal si

$$\forall \boldsymbol{v}_i, \boldsymbol{v}_j \in B \text{ con } i \neq j \text{ se cumple } \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle = 0$$

Si además se cumple que $\forall \boldsymbol{v} \in B$, $\|\boldsymbol{v}\| = 1$ *decimos que la base es ortonormal.*

Notar que si $B = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ base ortogonal, se cumple

$$\|\mathbf{w}\|^2 = \|a_1\mathbf{v}_1 + \cdots + a_n\mathbf{v}_n\|^2 = \langle a_1\mathbf{v}_1 + \cdots + a_n\mathbf{v}_n, a_1\mathbf{v}_1 + \cdots + a_n\mathbf{v}_n\rangle = \\ = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i \cdot a_j\langle \mathbf{v}_i, \mathbf{v}_j\rangle = \sum_{i=1}^{n} a_i^2\langle \mathbf{v}_i, \mathbf{v}_i\rangle = \sum_{i=1}^{n} a_i^2\|\mathbf{v}_i\|^2$$

donde la penúltima igualdad se da porque al ser B conjunto ortogonal, $\langle \mathbf{v}_i, \mathbf{v}_j\rangle = 0$ para todo $i \neq j$.

Definición 29. *Complemento ortogonal*
Sea $V \subseteq \mathbb{R}^n$ *un espacio vectorial con producto interno y sea* $S \subset V$*. Definimos el complemento ortogonal de* S *como el conjunto*

$$S^{\perp} = \{\boldsymbol{v} \in V : \forall \boldsymbol{s} \in S, \langle \boldsymbol{v}, \boldsymbol{s}\rangle = 0\}$$

Notamos que $S^{\perp}$ es subespacio de V, independientemente de que S lo sea.

Definición 30. *Proyección ortogonal*
Sea V *un* $\mathbb{R}$ *espacio vectorial con producto interno y* S *un subespacio de* V *tal que dado* $\boldsymbol{v} \in V$*, existen únicos* $\boldsymbol{v}_s \in S$ *y* $\boldsymbol{v}_{s^{\perp}} \in S^{\perp}$ *con* $\boldsymbol{v} = \boldsymbol{v}_s + \boldsymbol{v}_{s^{\perp}}$*. Sea* $B_S = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_p\}$ *una base ortogonal de* S*, entonces definimos la proyección ortogonal de* $\boldsymbol{w}$ *sobre* S *como la transformación lineal* $P_S : V \to S$ *tal que*

$$P_S(w) = \frac{\langle \boldsymbol{w}, \boldsymbol{s}_1\rangle}{\langle \boldsymbol{s}_1, \boldsymbol{s}_1\rangle}\boldsymbol{s}_1 + \cdots + \frac{\langle \boldsymbol{w}, \boldsymbol{s}_n\rangle}{\langle \boldsymbol{s}_n, \boldsymbol{s}_n\rangle}\boldsymbol{s}_n$$

Observar que P_S es la identidad para elementos de S y que $P_S(S^{\perp}) = \mathbf{0}$.

Culminamos con la presentación de un algoritmo que permite, a partir de una base cualquiera, generar una base ortogonal del espacio. La prueba de correctitud del mismo se puede consultar en [10].

Teorema 2. *Algoritmo de Gram-Schmidt*
Sea $B = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\}$ *una base para un espacio vectorial* $V \subseteq \mathbb{R}^n$*. La salida del siguiente algoritmo computa una base ortogonal* $B^* = \{\boldsymbol{v}_1^*, \boldsymbol{v}_2^*, \ldots, \boldsymbol{v}_n^*\}$ *de* V*.*

Algoritmo 1 Proceso de Ortogonalización de Gram-Schmidt

Entrada: $B = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ base de espacio vectorial $V \subseteq \mathbb{R}^n$
Salida: $B^* = \{\mathbf{v}_1^*, \ldots, \mathbf{v}_n^*\}$ base ortogonal de V

1: $\mathbf{v}_1^* \leftarrow \mathbf{v}_1$
2: **para** $i = 2$ hasta n **hacer**
3: **para** $j = 1$ hasta $i - 1$ **hacer**
4: $\mu_{i,j} \leftarrow \frac{\langle \mathbf{v}_i, \mathbf{v}_j^* \rangle}{\left\| \mathbf{v}_j^* \right\|^2}$
5: **fin para**
6: $\mathbf{v}_i^* \leftarrow \mathbf{v}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{v}_j^*$
7: **fin para**

Basta con normalizar los vectores de B^* para obtener una base ortonormal de V. Probamos ahora dos propiedades que cumplen los vectores producidos en el proceso.

Proposición 2. *Sea $B = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\}$ una entrada al proceso de Gram-Schmidt y $B^* = \{\boldsymbol{v}_1^*, \boldsymbol{v}_2^*, \ldots, \boldsymbol{v}_n^*\}$ la salida correspondiente. Entonces se cumple,*

- $\forall i, j \in \{1, \ldots, n\} : i < j,\ \langle \boldsymbol{v}_i, \boldsymbol{v}_j^* \rangle = 0$
- $\forall j \in \{1, \ldots, n\},\ \langle \boldsymbol{v}_j, \boldsymbol{v}_j^* \rangle = \langle \boldsymbol{v}_j^*, \boldsymbol{v}_j^* \rangle = \left\| \boldsymbol{v}_j^* \right\|^2$

Demostración. Para el primer punto, usando la definición de v_i tenemos,

$$\begin{gathered} \langle \mathbf{v}_i, \mathbf{v}_j^* \rangle = \langle \mathbf{v}_i^* + \sum_{k=1}^{i-1} \mu_{i,k} \mathbf{v}_k^*, \mathbf{v}_j^* \rangle = \\ = \langle \mathbf{v}_i^* + \mu_{i,1} \mathbf{v}_1^* + \mu_{i,2} \mathbf{v}_2^* + \cdots + \mu_{i,i-1} \mathbf{v}_{i-1}^*, \mathbf{v}_j^* \rangle = \\ = \langle \mathbf{v}_i^*, \mathbf{v}_j^* \rangle + \mu_{i,1} \langle \mathbf{v}_1^*, \mathbf{v}_j^* \rangle + \mu_{i,2} \langle \mathbf{v}_2^*, \mathbf{v}_j^* \rangle + \cdots + \mu_{i,i-1} \langle \mathbf{v}_{i-1}^*, \mathbf{v}_j^* \rangle = 0 \end{gathered}$$

ya que el conjunto B^* es ortogonal.

Para el segundo aplicando una idea análoga,

$$\begin{gathered} \langle \mathbf{v}_j, \mathbf{v}_j^* \rangle = \langle \mathbf{v}_j^* + \sum_{k=1}^{j-1} \mu_{j,k} \mathbf{v}_k^*, \mathbf{v}_j^* \rangle = \\ = \langle \mathbf{v}_j^* + \mu_{j,1} \mathbf{v}_1^* + \mu_{j,2} \mathbf{v}_2^* + \cdots + \mu_{j,j-1} \mathbf{v}_{j-1}^*, \mathbf{v}_j^* \rangle = \\ = \langle \mathbf{v}_j^*, \mathbf{v}_j^* \rangle + \mu_{j,1} \langle \mathbf{v}_1^*, \mathbf{v}_j^* \rangle + \mu_{j,2} \langle \mathbf{v}_2^*, \mathbf{v}_j^* \rangle + \cdots + \mu_{j,j-1} \langle \mathbf{v}_{j-1}^*, \mathbf{v}_j^* \rangle = \langle \mathbf{v}_j^*, \mathbf{v}_j^* \rangle \end{gathered}$$

□

2.3.2. Teoría de la Probabilidad

Recordamos las nociones básicas de la teoría de la probabilidad. Tomamos como referencia las definiciones de [2] y [9].

Definición 31. *Espacio de probabilidad finito*
Un espacio de probabilidad finito es un conjunto $\Omega = \{\omega_1, \ldots, \omega_n\}$ *finito no vacío con una distribución de probabilidad, que es una función* $\Pr : \Omega \to [0,1] \subset \mathbb{R}$ *tal que* $\sum_{\omega_i \in \Omega} \Pr(\omega_i) = 1$.

Cuando Pr es tal que para todo $\omega \in \Omega$, $\Pr(\omega) = \frac{1}{\#\Omega}$, decimos que es la distribución uniforme en Ω.

Definición 32. *Evento*
Un evento es un subconjunto $A \subseteq \Omega$ *y se define la probabilidad de un evento como* $\Pr(A) = \sum_{\omega \in A} \Pr(\omega)$.

Tenemos $\Pr(\emptyset) = 0$, $\Pr(\Omega \backslash A) = 1 - \Pr(A)$ y $\Pr(A \cup B) = \Pr(A) + \Pr(B) - \Pr(A \cap B)$ donde $A, B \subseteq \Omega$. En el trabajo también identificamos la unión y la intersección de eventos con los operadores $\vee$ y $\wedge$ respectivamente.

Definición 33. *Variable aleatoria*
Una variable aleatoria (v.a.) X *sobre un espacio de probabilidad finito* $(\Omega, \Pr)$ *es una función* $X : \Omega \to \mathbb{R}$. *Notamos por* $\mathcal{X} = X(\Omega)$ *el conjunto imagen de la variable aleatoria.*

Definición 34. *Dada* X *v.a. sobre* $(\Omega, \Pr)$*, definimos su valor esperado como*

$$E(X) = \sum_{\omega \in \Omega} X(\omega) \cdot \Pr(\omega) = \sum_{x \in \mathcal{X}} x \cdot \Pr(X^{-1}(x))$$

Dadas dos v.a. X, Y sobre $(\Omega, \Pr)$ y $\alpha, \beta \in \mathbb{R}$, notamos que el valor esperado es lineal, $\mathrm{E}(\alpha X + \beta Y) = \alpha \mathrm{E}(X) + \beta \mathrm{E}(Y)$.

Definición 35. *Dada* X *v.a. sobre* $(\Omega, \Pr)$*, definimos su varianza como* $Var(X) = E(X^2) - E(X)^2$. *También definimos su desviación estándar como* $\sigma(X) = \sqrt{Var(X)}$.

La función de probabilidad $\Pr_X : \mathcal{X} \subset \mathbb{R} \to [0,1]$ definida por $\Pr_X(X = x) = \Pr_X(x) = \Pr(X^{-1}(x))$ se llama función de distribución de probabilidad de la v.a. X. En particular, si X es discreta ($\mathcal{X}$ es un conjunto numerable)

llamamos a esta función como función de probabilidad (puntual)[1]. Observamos que $(X(\Omega), \Pr_X)$ es también un espacio de probabilidad finito. En general, el espacio Ω está implícito y hablamos de la distribución de la variable aleatoria sin hacer referencia a él.

En ocasiones en el trabajo identificamos la variable aleatoria con su distribución de manera indistinta. También hacemos uso de manera indistinta de las notaciones $\Pr(x \leftarrow X)$, $\Pr\left(x \xleftarrow{\boxdot} X\right)$ o $\Pr(X = x)$ para denotar la probabilidad de que la variable aleatoria X tome el valor $x \in \mathcal{X}$. Al notar $\xleftarrow{p} S$ con S conjunto, nos referimos a tomar un elemento de S según la distribución p y con $s \xleftarrow{\boxdot} S$ a tomar de forma aleatoria (según alguna distribución) un elemento del conjunto S.

[1]Si X es continua, la llamamos función de densidad. No profundizamos esta noción por ser los espacios de probabilidad finitos el centro del trabajo.

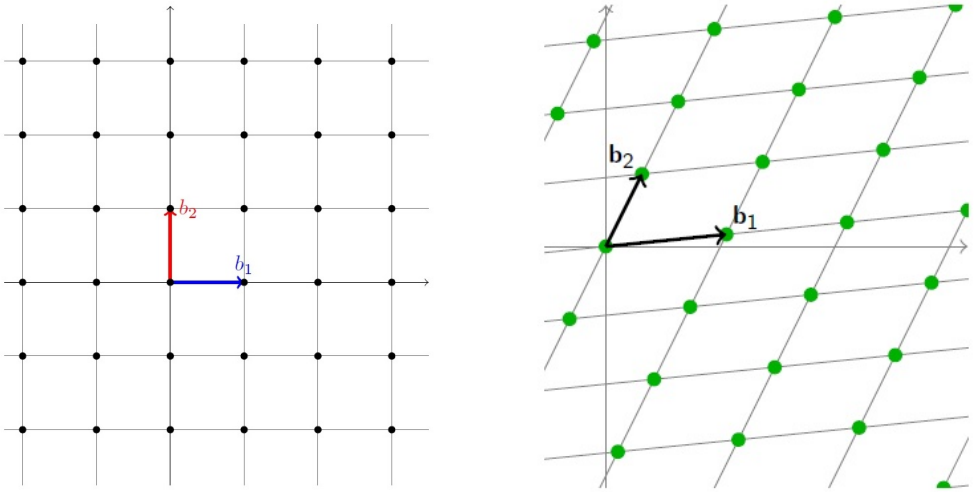
Capítulo 3

Reticulados

Continuamos con la descripción del objeto matemático que nos permitirá la construcción de distintos criptosistemas en los que se basará el estudio: el reticulado. La presentación sigue la línea de [10] y [12].

3.1. Definiciones básicas

Los reticulados son objetos geométrios que pueden ser descritos visualmente como el conjunto de puntos de intersección de una grilla infinita rectangular, no necesariamente ortogonal, de n dimensiones.



(a) Reticulado $\mathbb{Z}^2$ **(b)** Reticulado con base no ortogonal

Figura 3.1: Ejemplos de reticulados 2 dimensionales [13].

Formalmente, son definidos como sigue

Definición 36. *Reticulado*
Sea $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\} \subset \mathbb{R}^m$ *un conjunto de* n *vectores linealmente independientes* $(m \geqslant n)$*.*
El reticulado $\mathcal{L} \subset \mathbb{R}^m$ *generado por* $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *es el conjunto*

$$\mathcal{L}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) = \Big\{ \sum_{i=1}^{n} x_i \boldsymbol{b}_i : x_i \in \mathbb{Z} \Big\}$$

de todas las combinaciones lineales de $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *con coeficientes enteros.*

Decimos que n *es el rango del reticulado y* m *su dimensión. Llamamos al conjunto* $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *la base del reticulado, que puede ser representada de forma matricial como sigue*

$$B = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n] \in \mathbb{R}^{m \times n}$$

Dando esto lugar a una representación equivalente,

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

Podemos interpretar la representación alternativa como el hecho de que cada reticulado puede ser obtenido aplicando alguna transformación lineal, representada por la matriz $B \in \mathbb{R}^{m \times n}$ al reticulado entero $\mathbb{Z}^n$.

Cuando $n = m$, decimos que el reticulado tiene rango máximo. Tenemos entonces,

Proposición 3. *Dada* $B \in \mathbb{R}^{m \times n}$ *con columnas linealmente independientes. Entonces,* $\mathcal{L}(B) \subset \mathbb{R}^m$ *es de rango máximo si y solo si*

$$\text{span}(B) = \{B\boldsymbol{x} : \boldsymbol{x} \in \mathbb{R}^n\} = \mathbb{R}^m$$

Esto surge de observar que si $\text{span}(B) = \mathbb{R}^m$, entonces el cardinal del conjunto generador debe ser m y por tanto $n = m$.

De aquí podemos caracterizar el rango de un reticulado $\mathcal{L} \subset \mathbb{R}^m$ como la dimensión del espacio que genera su base,

$$\text{rango}(\mathcal{L}(B)) = \dim(\text{span}(B))$$

Como en general, existe más de una base que genera el espacio, es decir que si B y B' generan el mismo reticulado entonces $\text{span}(B) = \text{span}(B')$, entonces podemos para cualquier reticulado $\Lambda = \mathcal{L}(B)$ definir el espacio generado por los vectores del reticulado $\text{span}(\Lambda)$ sin hacer referencia a una base en particular [12].

En los casos en los que tenemos una base dada del reticulado, no perdemos generalidad al asumir que el reticulado es de rango máximo, ya que de lo contrario podríamos restringir el estudio al espacio generado por los vectores de la base, resultando el tratamiento algorítmico equivalente a menos de factores multiplicativos en la cantidad de bits usados para representar los vectores, que es mayor en el caso en el que el rango no sea máximo.

Continuamos con la noción de subreticulado.

Definición 37. *[12] Subreticulado*
Sean $\mathcal{L}' = \mathcal{L}(B')$ y $\mathcal{L} = \mathcal{L}(B)$ reticulados generados por bases B' y B respectivamente, si $\mathcal{L}' \subseteq \mathcal{L}$ decimos que $\mathcal{L}'$ es un subreticulado de $\mathcal{L}$.

Observamos que existe una gran similitud entre la noción de reticulado y la de espacio vectorial que presentamos en el Capítulo 2: mientras que el segundo toma combinaciones lineales de la base con coeficientes reales, el primero restringe estos al dominio de los números enteros.

Si B es una base para el reticulado $\mathcal{L}(B)$, entonces B también es una base para el espacio vectorial $\text{span}(B)$. Sin embargo, no todo conjunto de vectores $S \subset \mathcal{L}(B)$ que sea base del espacio vectorial $\text{span}(B)$ constituye una base del reticulado.

En particular, dada $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ una base de $\text{span}(B)$, consideramos el conjunto $2B = \{2\mathbf{b}_1, \ldots, 2\mathbf{b}_n\}$ que se mantiene base para $\text{span}(B)$, sin embargo solo genera $2\mathcal{L}(B) = \{2\mathbf{v} : \mathbf{v} \in \mathcal{L}(B)\} \subset \mathcal{L}(B)$.

3.2. Relaciones entre bases, caracterización algebraica

Analizamos, de forma análoga a lo que hicimos en el caso de espacios vectoriales, la relación entre dos bases para el mismo reticulado.

Supongamos que $B = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ es base del reticulado $\mathcal{L}(B)$ y sea $B' = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n\}$ otro conjunto de vectores en $\mathcal{L}$. Podemos escribir cada $\mathbf{w}_j \in B'$ como combinación lineal de la base B,

$$\begin{aligned}
\mathbf{w}_1 &= a_{11}\mathbf{v}_1 + a_{12}\mathbf{v}_2 + \cdots + a_{1n}\mathbf{v}_n \\
\mathbf{w}_2 &= a_{21}\mathbf{v}_1 + a_{22}\mathbf{v}_2 + \cdots + a_{2n}\mathbf{v}_n \\
&\vdots \\
\mathbf{w}_n &= a_{n1}\mathbf{v}_1 + a_{n2}\mathbf{v}_2 + \cdots + a_{nn}\mathbf{v}_n
\end{aligned}$$

Como se trata de un reticulado, sabemos que $a_{ij} \in \mathbb{Z}$ para $i, j \in \{1, \ldots, n\}$.

Entonces, al igual que en el caso de espacios vectoriales, si queremos expresar los $\mathbf{v}_i$ en términos de los $\mathbf{w}_j$ debemos poder invertir la matriz de coeficientes

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

Como los $\mathbf{v}_i \in \mathcal{L}$, los coeficientes de la combinación lineal de los elementos de B' deben ser también enteros, es decir que además de garantizar la existencia de A^{-1} esta debe tener entradas enteras. Por tanto,

$$\begin{gathered}
1 = \det(I) = \det(AA^{-1}) = \det(A)\det(A^{-1}) \\
\text{donde } \det(A), \det(A^{-1}) \in \mathbb{Z}
\end{gathered}$$

Recordando el siguiente resultado,

Proposición 4. *Dada una matriz entera M, su inversa M^{-1}, es también una matriz con entradas enteras si y solo si* $\det(M) = \pm 1$.

Por tanto debe ser $\det(A) = \pm 1$.

Introducimos un término para definir este tipo de matrices,

Definición 38. *Matriz Unimodular*
Una matriz $U \in \mathbb{Z}^{n\times n}$ *se dice unimodular si* $\det(U) = \pm 1$.

De lo anterior obtenemos la siguiente proposición,

Proposición 5. *Dadas dos bases* $B, B' \in \mathbb{R}^{m\times n}$ *para un reticulado* $\mathcal{L} \subset \mathbb{R}^m$, *existe una matiz unimodular* $U \in \mathbb{Z}^{n\times n}$ *tal que*

$$B' = BU$$

Este resultado nos da un procedimiento eficiente algebraico para determinar si dos bases generan el mismo reticulado.

Para el caso en el que el reticulado sea de rango máximo y por tanto B invertible, basta verificar que $B^{-1}B' = U$ sea unimodular. Otro corolario que se desprende de este resultado es que el conjunto las bases de $\mathbb{Z}^n$ es el de todas las matrices unimodulares $U \in \mathbb{Z}^{n\times n}$. Esto se obtiene de observar que la matriz identidad $I \in \mathbb{Z}^{n\times n}$ es una base para $\mathbb{Z}^n$.

Como veremos más adelante, una gran variedad de los problemas algorítmicos sobre reticulados se reducen al problema de transformar una base arbitraria dada para un reticulado en otra para el mismo con algunas propiedades especiales. De aquí surge el interés de estudiar este tipo de propiedades sobre cambios de base para un reticulado.

En este sentido, al considerar aspectos algorítmicos, son de particular interés los reticulados enteros y racionales, que son subconjuntos de $\mathbb{Z}^n$ y $\mathbb{Q}^n$ respectivamente. Esto se debe a que estos conjuntos pueden ser representados con una cantidad finita de elementos. Notar que no todos los reticulados cumplen esta propiedad, por ejemplo $\sqrt{2}\mathbb{Z}$ entre otros. De hecho, estos dos tipos de reticulados resultan equivalentes a menos de un factor de ajuste, ya que el hecho de que se trate de conjuntos discretos garantiza que para todo reticulado racional existe un $d \in \mathbb{Z}$ tal que $d\mathcal{L}$ sea entero. Podemos pensar en d como el "común denominador" de todos los elementos del reticulado racional. Expandiremos a continuación esta noción, cuando presentemos una definición equivalente de los reticulados como subgrupos aditivos.

3.3. Definición alternativa

Definición 39. *Definición Alternativa Reticulado*
Un subconjunto $\mathcal{L} \subset \mathbb{R}^m$ es un reticulado si y solo si es un subgrupo aditivo discreto.

Definimos ahora las nociones mencionados en la definición alternativa y analizamos la equivalencia de las dos definiciones presentadas.

Definición 40. *Grupo*
Un grupo es un conjunto G con una operación binaria $* : G \times G \to G$ tal que se cumple

- *(Asociatividad) $\forall x, y, z \in G$, $x * (y * z) = (x * y) * z$*
- *(Existencia de neutro) $\exists$ un elemento $e \in G$ tal que $e * x = x * e$ para todo $x \in G$*
- *(Existencia de inverso) $\forall g \in G$, $\exists g^{-1} \in G$ tal que $g * g^{-1} = g^{-1} * g = e$*

Un grupo puede ser notado por G cuando la operación y el neutro son claros, o bien $(G, *)$ o $(G, *, e)$ para indicar estos elementos de manera explícita.

Definición 41. *Subgrupo*
Dado un grupo $(G, *, e)$, un subconjunto $H \subseteq G$ es un subgrupo de G si cumple

- *(Cerrado bajo operación) $\forall h, h' \in H$, $h * h' \in H$*
- *(Existencia de neutro) $e \in H$*
- *(Cerrado bajo inversos) Si $h \in H$ entonces $h^{-1} \in H$*

Notamos $H < G$ cuando H es un subgrupo de G.

Entonces, según la definición alternativa un reticulado $\mathcal{L}$ es un subgrupo aditivo de $\mathbb{R}^m$, es decir que $\mathcal{L} < \mathbb{R}^m$ tomando como operación la suma usual de vectores. De acuerdo a la definición debe cumplirse entonces,

- $\mathbf{0} \in \mathcal{L}$
- $\forall \mathbf{x}, \mathbf{y} \in \mathcal{L}$, $\mathbf{x} + \mathbf{y} \in \mathcal{L}$
- Si $\mathbf{x} \in \mathcal{L}$ entonces $-\mathbf{x} \in \mathcal{L}$

Una noción equivalente exige únicamente que el conjunto sea cerrado bajo la operación de sustracción, ya que podemos obtener los inversos restando el neutro y expresar la suma como $\mathbf{x} + \mathbf{y} = \mathbf{x} - (-\mathbf{y})$.

Además de la definición alternativa el conjunto de vectores de un reticulado debe ser discreto, noción que definimos a continuación.

Definición 42. *Conjunto Discreto*
Un conjunto $S \subset \mathbb{R}^m$ es discreto si existe un $\epsilon > 0$ tal que

$$\forall \boldsymbol{x} \in S,\ S \cap \{\boldsymbol{w} \in \mathbb{R}^m : \|\boldsymbol{x} - \boldsymbol{w}\| < \epsilon\} = \{\boldsymbol{x}\}$$

Entonces, para el caso de un reticulado debe cumplirse también que exista un $\epsilon > 0$ tal que $\forall \mathbf{v} \in \mathcal{L}$, $\mathcal{L} \cap B(\mathbf{v}, \epsilon) = \{\mathbf{v}\}$. Donde $B(\mathbf{v}, \epsilon)$ denota la bola de centro $\mathbf{v}$ y radio ϵ, $B(\mathbf{v}, \epsilon) = \{\mathbf{w} \in \mathbb{R}^m : \|\mathbf{v} - \mathbf{w}\| < \epsilon\}$.

Para presentar la equivalencia de ambas definiciones debemos introducir dos conceptos más, que resultan fundamentales en el estudio de los reticulados.

3.4. Relaciones entre bases, caracterización geométrica

Comenzamos presentando la noción de paralelepípedo fundamental de un reticulado.

Definición 43. *Paralelepípedo fundamental*
Dados n vectores linealmente independientes $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_n \in \mathbb{R}^m$, definimos su paralelepípedo fundamental como

$$\mathcal{P}(\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_n) = \left\{ \sum_{i=1}^{n} x_i \boldsymbol{b}_i : x_i \in [0, 1) \right\}$$

Gráficamente, un paralelepípedo fundamental es la región semiabierta delimitada por los vectores $\mathbf{b}_1, \ldots, \mathbf{b}_n$. Observamos que en particular, los vectores que conforman la base de un reticulado tienen asociado un paralelepípedo fundamental y que dos bases distintas generan paralelepípedos fundamentales diferentes como podemos ver en la Figura 3.2.

Enunciamos a continuación un resultado que manifiesta una idea importante en relación a su estructura.

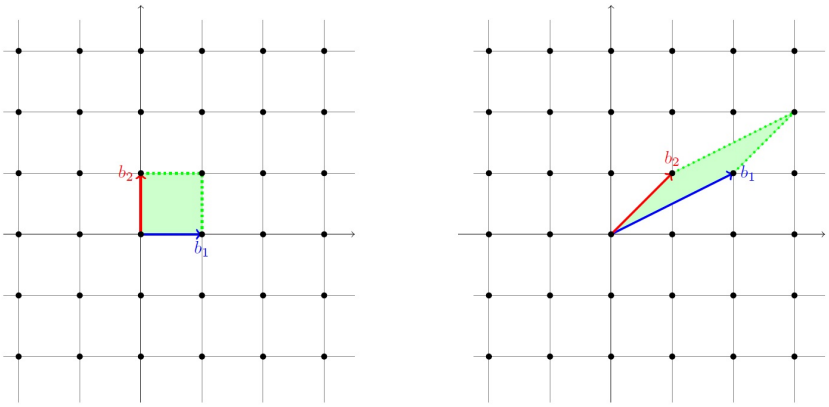


Figura 3.2: Reticulado $\mathbb{Z}^2$ con dos bases asociadas $B = \{(0,1),(1,0)\}$ y $B' = \{(1,1),(2,1)\}$ junto a sus paralelepípedos fundamentales [13].

Proposición 6. *Sea $\mathcal{L} \subset \mathbb{R}^n$ un reticulado de rango n y $\mathcal{P}(B)$ un paralelepípedo fundamental asociado. Entonces, para todo $\boldsymbol{w} \in \mathbb{R}^n$, existe un único $\boldsymbol{t} \in \mathcal{P}(B)$ y un único $\boldsymbol{v} \in \mathcal{L}$ tales que $\boldsymbol{w} = \boldsymbol{t} + \boldsymbol{v}$.*

La demostración puede consultarse en [10].

La idea esencial de la proposición es capturar que, si consideramos traslaciones de $\mathcal{P}(B)$ sobre los distintos vectores del reticulado cubrimos el espacio de forma exacta, es decir sin solapamientos. Esto puede ser expresado de la siguiente forma,

$$\biguplus_{\mathbf{v}\in\mathcal{L}} \mathcal{P}(B) + \mathbf{v} = \biguplus_{\mathbf{v}\in\mathcal{L}} \{\mathbf{t} + \mathbf{v} : \mathbf{t} \in \mathcal{P}(B)\} = \mathbb{R}^n$$

Presentamos ahora una condición necesaria para que un conjunto de vectores linealmente independiente sea una base de un reticulado, que se basa en el estudio del paralelepípedo fundamental.

Teorema 3. *Sea $\mathcal{L} \subset \mathbb{R}^n$ un reticulado de rango máximo, n dimensional y sea $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\} \subset \mathcal{L}$ un conjunto de vectores linealmente independientes. Entonces B es una base de $\mathcal{L}$ si y solo si*

$$\mathcal{P}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) \cap \mathcal{L} = \{\boldsymbol{0}\}$$

Demostración. ($\Rightarrow$) Supongamos que B es base de $\mathcal{L}$. Consideremos un vector $\mathbf{t} \in \mathcal{L} \cap \mathcal{P}(B)$. Escribimos $\mathbf{t} = \sum_{i=1}^{n} x_i \mathbf{b}_i$ en términos de la base.

Ahora como $\mathbf{t} \in \mathcal{L}$ debe cumplirse por definición que $x_i \in \mathbb{Z}$ para todo $i \in \{1, \ldots, n\}$. A su vez como $\mathbf{t} \in \mathcal{P}(B)$, $x_i \in [0, 1)$ para todo $i \in \{1, \ldots, n\}$. De aquí solo puede ser $x_i = 0$ para todo i y por tanto $\mathbf{t} = \mathbf{0}$.

($\Leftarrow$) Supongamos que $\mathcal{P}(B) \cap \mathcal{L} = \{\mathbf{0}\}$. Queremos probar que B es base del reticulado.

Sea $\mathcal{L}(B)$ el conjunto de combinaciones lineales enteras de los vectores de B, es decir el reticulado generado por B. Como $B \subset \mathcal{L}$, $\mathcal{L}(B) \subset \mathcal{L}$.

Para ver la inclusión en el otro sentido, tomemos un $\mathbf{v} \in \mathcal{L}$,

$$\mathbf{v} = \sum_{i=1}^{n} x_i \mathbf{b}_i \text{ con } x_i \in \mathbb{R}$$

Consideremos ahora el siguiente vector,

$$\mathbf{v}' = \sum_{i=1}^{n} \lfloor x_i \rfloor \mathbf{b}_i \in \mathcal{L}(B)$$

Entonces, como vimos en la definición alternativa también debe ser $\mathbf{v} - \mathbf{v}' \in \mathcal{L}(B)$ con

$$\mathbf{v} - \mathbf{v}' = \sum_{i=1}^{n} (x_i - \lfloor x_i \rfloor) \mathbf{b}_i$$

que pertenece al paralelepípedo $\mathcal{P}(B)$ pues $x_i - \lfloor x_i \rfloor \in [0, 1)$ para todo i.

Luego tenemos que $\mathbf{v} - \mathbf{v}' \in \mathcal{L}(B) \cap \mathcal{P}(B)$ y por hipótesis sabemos entonces que $\mathbf{v} - \mathbf{v}' = \mathbf{0}$. Ahora como $\mathbf{b}_1, \ldots, \mathbf{b}_n$ son l.i.,

$$\mathbf{0} = \mathbf{v} - \mathbf{v}' = \sum_{i=1}^{n} (x_i - \lfloor x_i \rfloor) \mathbf{b}_i$$

debe ser $x_i - \lfloor x_i \rfloor = 0$ para todo $i \in \{1, \ldots, n\}$. Esto implica que $x_i = \lfloor x_i \rfloor \in \mathbb{Z}$ para todo i.

Entonces, tenemos que para un $\mathbf{v} \in \mathcal{L}$ arbitrario se cumple $\mathbf{v} \in \mathcal{L}(B)$ que implica $\mathcal{L} \subset \mathcal{L}(B)$.

De ambas tenemos $\mathcal{L} = \mathcal{L}(B)$. □

Continuamos con una noción relacionada al paralelepípedo fundamental de un reticulado.

3.5. Determinante de un Reticulado

Definición 44. *Determinante*
Dado un reticulado $\mathcal{L} \subset \mathbb{R}^m$ definimos su determinante, denotado $\det(\mathcal{L})$ *como el volumen n dimensional de su paralelepípedo fundamental $\mathcal{P}(B)$.*

Esta cantidad es un invariante del reticulado, ya que no depende de la base elegida. En el caso donde el reticulado es de rango máximo, que implica que B es cuadrada, tenemos que

$$\det(\mathcal{L}) = |\det(B)|$$
con B representación matricial de una base de $\mathcal{L}$ [1]

Veamos ahora que los paralelepípedos asociados a distintas bases tienen el mismo volumen. Por la relación algebraica presentada en la Sección 3.2 para distintas bases, tenemos que para dos bases B y B' de $\mathcal{L}$, $B' = BU$ donde U es una matriz unimodular.

Entonces,

$$|\det(B')| = |\det(BU)| = |\det(B)| \cdot |\det(U)| = |\det(B)|$$

donde la segunda igualdad se debe a que el determinante es multiplicativo y la última a la definición de matriz unimodular.

Entonces, vimos que todos los paralelepípedos fundamentales asociados a un reticulado tienen el mismo volumen y que cubren la totalidad del espacio en el caso de rango máximo. De aquí, surge la interpretación de que el determinante de un reticulado es inversamente proporcional a la densidad de puntos del mismo por unidad de volumen del espacio. Cuanto más grande es el determinante, menos concentrados están los puntos del reticulado en el espacio.

Comentamos ahora dos alternativas eficientes para el cómputo del determinante de un reticulado. La primera opción depende de la base de Gram-Schmidt correspondiente. Entonces, primero es preciso notar

Proposición 7. *El Algoritmo 1, que dada una base de un reticulado B, computa la base de Gram-Schmidt correspondiente B^* puede implementarse en tiempo polinomial.*

[1] La demostración puede consultarse en [10]

La demostración puede consultarse en [14].

Entonces tenemos el siguiente resultado,

Proposición 8. *Cómputo eficiente del determinante I*
Dado un reticulado $\mathcal{L} \subset \mathbb{R}^n$ de rango máximo con base B tenemos

$$\det(\mathcal{L}) = \prod_{i=1}^{n} \|\boldsymbol{b}_i^*\|$$

donde $B^* = [\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*]$ es la base de Gram-Schmidt correspondiente.

Demostración. Reescribimos el proceso de ortogonalización de Gram-Schmidt visto en la Sección 2.3.1 de forma matricial,

$$B = [\mathbf{b}_1, \ldots, \mathbf{b}_n] = [\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*] \cdot \begin{pmatrix} 1 & \mu_{21} & \mu_{31} & \ldots & \mu_{n1} \\ 0 & 1 & \mu_{32} & \ldots & \mu_{nu} \\ 0 & 0 & 1 & \ldots & \mu_{n3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$= \left[\tfrac{\mathbf{b}_1^*}{\|\mathbf{b}_1^*\|}, \ldots, \tfrac{\mathbf{b}_n^*}{\|\mathbf{b}_n^*\|}\right] \cdot \begin{pmatrix} \|\mathbf{b}_1^*\| & \mu_{21}\|\mathbf{b}_1^*\| & \mu_{31}\|\mathbf{b}_1^*\| & \ldots & \mu_{n1}\|\mathbf{b}_1^*\| \\ 0 & \|\mathbf{b}_2^*\| & \mu_{32}\|\mathbf{b}_2^*\| & \ldots & \mu_{nu}\|\mathbf{b}_2^*\| \\ 0 & 0 & \|\mathbf{b}_3^*\| & \ldots & \mu_{n3}\|\mathbf{b}_3^*\| \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \|\mathbf{b}_n^*\| \end{pmatrix}$$

Entonces como el determinante es multiplicativo,

$$\det(B) = \det\left(\left[\tfrac{\mathbf{b}_1^*}{\|\mathbf{b}_1^*\|}, \ldots, \tfrac{\mathbf{b}_n^*}{\|\mathbf{b}_n^*\|}\right]\right) \cdot \det \begin{pmatrix} \|\mathbf{b}_1^*\| & \mu_{21}\|\mathbf{b}_1^*\| & \mu_{31}\|\mathbf{b}_1^*\| & \ldots & \mu_{n1}\|\mathbf{b}_1^*\| \\ 0 & \|\mathbf{b}_2^*\| & \mu_{32}\|\mathbf{b}_2^*\| & \ldots & \mu_{nu}\|\mathbf{b}_2^*\| \\ 0 & 0 & \|\mathbf{b}_3^*\| & \ldots & \mu_{n3}\|\mathbf{b}_3^*\| \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \|\mathbf{b}_n^*\| \end{pmatrix}$$

Como los vectores $\frac{\mathbf{b}_i^*}{\|\mathbf{b}_i^*\|}$ son ortonormales, el determinante de la matriz con dichos vectores es 1 o -1 (1 en valor absoluto) y al ser la segunda matriz triangular superior obtenemos,

$$\det(\mathcal{L}) = |\det(B)| = \prod_{i=1}^{n} \|\mathbf{b}_i^*\|$$

□

El segundo resultado que presentamos no requiere el cálculo de la base de Gram-Schmidt y resulta también más general.

Proposición 9. *Cómputo eficiente del determinante II*
Para cualquier reticulado con base $B \in \mathbb{R}^{m\times n}$ *tenemos*

$$\det(\mathcal{L}(B)) = \sqrt{\det(B^t B)}$$

En particular, si $B \in \mathbb{R}^{n\times n}$ *es una matriz cuadrada invertible entonces* $\det(B) = \det(B^t)$ *y por tanto* $\det(\mathcal{L}(B)) = |\det(B)|$ *como vimos en la Proposición 8.*

Demostración. Nuevamente basamos la demostración en el proceso de ortogonalización de Gram-Schmidt en su forma matricial. En particular como vimos en la prueba de la Proposición 8, podemos escribir a la matriz compuesta por los vectores de la base como $B = B^*T$ donde B^* es la matriz de vectores ortogonales y T una matriz triangular superior con unos en la diagonal. Entonces,

$$\sqrt{\det(B^t B)} = \sqrt{\det((B^*T)^t B^*T)} = \sqrt{\det(T^t B^{*t} B^* T)} = \\ = \sqrt{\det(T^t)\det(B^{*t}B^*)\det(T)}$$

Ahora como las matrices T y T^t son triangular superior e inferior respectivamente, sus determinantes pueden ser calculados como el producto de los elementos de sus diagonales, que es 1 en ambos casos. En el caso de $B^{*t}B^*$, como las columnas de B^* son ortogonales tenemos que $B^{*t}B^*$ es matriz diagonal y por tanto su determinante es el producto de los elementos de esta,

$$\det(B^{*t}B^*) = \prod_{i=1}^{n} \langle \mathbf{b}_i^*, \mathbf{b}_i^* \rangle = \prod_{i=1}^{n} \|\mathbf{b}_i^*\|^2 = \\ = (\prod_{i=1}^{n} \|\mathbf{b}_i^*\|)^2 = \det(\mathcal{L}(B))^2$$

De aquí se obtiene que $\det(\mathcal{L}(B)) = \sqrt{\det(B^t B)}$ como queríamos probar.
□

Definimos una cantidad que indica qué tan ortogonales son los elementos de una base, que será de utilidad en la evaluación del criptosistema del Capítulo 5.

Definición 45. *Defecto ortogonal*
Dada una base $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *de un reticulado de rango máximo, definimos su defecto ortogonal como*

$$def\_ort(B) = \frac{\prod_{i=1}^{n} \|\boldsymbol{b}_i\|}{|\det(B)|}$$

Notar que como $\|\mathbf{b}_i^*\| \leq \|\mathbf{b}_i\|$ para todo elemento de la base, entonces $\det(B) = \prod_{i=1}^{n} \|\mathbf{b}_i^*\| \leq \prod_{i=1}^{n} \|\mathbf{b}_i\|$ y por tanto

$$1 \leq \frac{\prod_{i=1}^{n} \|\mathbf{b}_i\|}{|\det(B)|}$$

con igualdad si y solo si B es base ortogonal.

3.6. Ejemplos

Consideramos ahora algunos ejemplos para familiarizarnos con las definiciones equivalentes (Definición 36 y Definición 39) de reticulados.

- El conjunto que contiene únicamente al vector nulo $\{\mathbf{0}\} \subset \mathbb{R}^m$ es un reticulado para cualquier $m \in \mathbb{Z}^+$.
- El conjunto de los números enteros, $\mathbb{Z} \subset \mathbb{R}$ constituye un reticulado unidimensional y de igual manera, $\mathbb{Z}^n \subset \mathbb{R}^n$ también es un reticulado.
- El conjunto $\mathbb{Z} \times \{0\} = \{(k, 0) : k \in \mathbb{Z}\} \subset \mathbb{R}^2$ es un reticulado de dimensión 2, aunque su rango es 1, ya que $B_1 = \{(1, 0)\} \subset \mathbb{R}^2$ es base y $dim(B_1) = 1$, en particular genera el subespacio $\{(k, 0) : k \in \mathbb{R}\}$.
- $\mathbb{Q}^n$ es un subgrupo de $\mathbb{R}^n$, sin embargo no es un reticulado por no ser discreto. Para ver esto, observar por ejemplo que existen vectores racionales arbitrariamente cercanos al vector nulo.
- Los enteros impares $2\mathbb{Z} + 1$ no forman un reticulado, porque si bien son discretos no son subgrupo de $\mathbb{R}$, por no ser cerrados bajo la operación de suma. Por otro lado, el conjunto de los números pares $2\mathbb{Z}$ si es un reticulado, una base posible es $\{2\}$.

Notamos que salvo por el caso degenerado $\{\mathbf{0}\}$, un reticulado siempre viene dado por un conjunto de cardinalidad infinita.

3.7. Mínimos sucesivos

Otro de los parámetros que veremos resulta de interés al considerar un reticulado es el largo de un vector más corto no nulo del mismo. Consideramos vectores no nulos, porque el vector nulo pertenece a todo reticulado, evitando así obtener siempre esta solución trivial.

Como comentamos con anterioridad, usaremos la norma Euclídea en este trabajo para definir el largo de los vectores. Notamos que en general, pueden existir muchos vectores "más cortos" en un reticulado. En particular, en un reticulado no trivial (distinto del compuesto únicamente por el vector nulo) deben existir al menos dos, ya que si $\mathbf{v}$ es un vector más corto, $-\mathbf{v}$ también lo será. Formalicemos esta noción,

Definición 46. *[13] Distancia Mínima*
Dado un reticulado $\mathcal{L}$ definimos la distancia mínima de $\mathcal{L}$ como

$$\lambda_1(\mathcal{L}) = \min_{\boldsymbol{v}\in\mathcal{L}\setminus\{\boldsymbol{0}\}} \|\boldsymbol{v}\| = \min_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{L}:\boldsymbol{x}\neq\boldsymbol{y}} \|\boldsymbol{x}-\boldsymbol{y}\|$$

Para la segunda igualdad, considerar que para dos vectores distintos del reticulado, $\boldsymbol{x}-\boldsymbol{y}\in\mathcal{L}$ y es no nulo.

En general, se define el mínimo sucesivo i-ésimo $\lambda_i(\mathcal{L})$ como el menor $r\in\mathbb{R}$ tal que $\mathcal{L}$ tiene i vectores linealmente independientes de largo a lo sumo r. Formalizando esta noción tenemos,

Definición 47. *Mínimos Sucesivos*
Dado un reticulado $\mathcal{L}\subset\mathbb{R}^m$ de rango n, definimos para todo $1\leq i\leq n$,

$$\lambda_i(\mathcal{L}) = \inf\{r\in\mathbb{R} : B(\boldsymbol{0},r) \text{ contiene al menos } i \text{ vectores del reticulado l.i.}\}$$

Notar que los mínimos sucesivos forman una secuencia no decreciente $\lambda_1\leq\lambda_2\leq\cdots\leq\lambda_n$.

Estudiaremos ahora cotas superiores e inferiores para $\lambda_1(\mathcal{L})$, siguiendo los contenidos de [13] y [12].

3.7.1. Cotas inferiores sobre $\lambda_1(\mathcal{L})$

Teorema 4. *Cota inferior sobre* $\lambda_1(\mathcal{L})$
Sea B *la base de un reticulado de rango* n *y* B^* *la base de Gram-Schmidt correspondiente. Entonces,*

$$\lambda_1(\mathcal{L}(B)) \geq \min_{i\in\{1,\dots,n\}} \|\boldsymbol{b}_i^*\| > 0$$

Demostración. Sea $\mathbf{x} \in \mathbb{Z}^n$ un vector de coeficientes enteros no nulo. Queremos acotar el largo del vector $B\mathbf{x} \in \mathcal{L}(B)$.

Para esto, calculamos la cantidad $|\langle Bx, b_j^*\rangle|$ de dos maneras.

Sea $j \in \{1,\dots,n\}$ el mayor índice tal que $x_j \neq 0$. Tenemos entonces,

$$|\langle B\mathbf{x}, \mathbf{b}_j^*\rangle| = |\langle \sum_{i=1}^{n} x_i\mathbf{b}_i, \mathbf{b}_j^*\rangle| = |\sum_{i=1}^{n} x_i\langle \mathbf{b}_i, \mathbf{b}_j^*\rangle| = |x_j\langle \mathbf{b}_j, \mathbf{b}_j^*\rangle|$$

Donde la segunda igualdad resulta de la linealidad del producto interno, la tercera de que para una base de Gram-Schmidt por construcción, $\forall i < j, \langle \mathbf{b}_i, \mathbf{b}_j^*\rangle = 0$ y finalmente por definición de j tenemos que $\forall i > j, x_i = 0$ para justificar la última igualdad.

Notamos ahora que,

$$|x_j\langle \mathbf{b}_j, \mathbf{b}_j^*\rangle| = |x_j\langle \mathbf{b}_j^* + \sum_{k<j} \mu_{jk}\mathbf{b}_k^*, \mathbf{b}_j^*\rangle| =$$
$$= |x_j\langle \mathbf{b}_j^*, \mathbf{b}_j^*\rangle + \sum_{k<j} x_j\mu_{jk}\langle \mathbf{b}_k^*, \mathbf{b}_j^*\rangle| = |x_j\langle \mathbf{b}_j^*, \mathbf{b}_j^*\rangle| = |x_j|\left\|\mathbf{b}_j^*\right\|^2$$

De aquí, $|\langle B\mathbf{x}, \mathbf{b}_j^*\rangle| = |x_j|\left\|\mathbf{b}_j^*\right\|^2$.

Por otro lado, por la desigualdad de Cauchy-Schwarz,

$$|\langle B\mathbf{x}, \mathbf{b}_j^*\rangle| \leq \|B\mathbf{x}\| \cdot \left\|\mathbf{b}_j^*\right\|$$

De ambas obtenemos,

$$|\langle B\mathbf{x}, \mathbf{b}_j^*\rangle| = |x_j|\left\|\mathbf{b}_j^*\right\|^2 \leq \|B\mathbf{x}\| \cdot \left\|\mathbf{b}_j^*\right\|$$
$$\Leftrightarrow \|B\mathbf{x}\| \geq |x_j|\left\|\mathbf{b}_j^*\right\| \geq \left\|\mathbf{b}_j^*\right\| \geq \min_{i\in\{1,\dots,n\}} \|\mathbf{b}_i^*\|$$

Donde la tercera desigualdad se sigue del hecho de que x_j es un entero no nulo. De aquí se cumple lo enunciado. □

Como corolario obtenemos que el reticulado es un conjunto discreto, ya que si tomamos dos $\mathbf{x}, \mathbf{y} \in \mathcal{L}$, $\mathbf{x} - \mathbf{y} \in \mathcal{L}$ y $\|\mathbf{x} - \mathbf{y}\| \geq \lambda_1(\mathcal{L}) > 0$. Tomando $\epsilon = \lambda_1(\mathcal{L})$ se cumple la definición de conjunto discreto.

3.7.2. Cota superior sobre $\lambda_1(\mathcal{L})$

Nos proponemos ahora acotar el largo $\lambda_1(\mathcal{L})$ superiormente. Trivialmente, se tiene que para cualquier base B del reticulado, $\lambda_1(\mathcal{L}) \leq \min_i \|\mathbf{b}_i\|$, por estar todo elemento de la base en el reticulado y ser no nulo. Sin embargo, resulta de interés encontrar una cota que no dependa de la base utilizada. Para esto enunciamos ahora dos resultados, ambos debidos al matemático alemán Hermann Minkowski.

Teorema 5. *Primer Teorema de Minkowski*
Para todo reticulado $\mathcal{L} \subset \mathbb{R}^n$ de rango máximo,

$$\lambda_1(\mathcal{L}) \leq \sqrt{n} \cdot \det(\mathcal{L})^{\frac{1}{n}}$$

Observar que además de no depender de la base, esta cota escala linealmente junto al reticulado, pues si escalamos el reticulado por un factor $c \in \mathbb{R}$ no nulo, tenemos que $\lambda_1(c\mathcal{L}) = c\lambda_1(\mathcal{L})$ y además $\det(c\mathcal{L})^{\frac{1}{n}} = (c^n \det(\mathcal{L}))^{\frac{1}{n}} = c \cdot \det(\mathcal{L})^{\frac{1}{n}}$.

El segundo resultado, da condiciones necesarias para asegurar la existencia de vectores del reticulado en una región del espacio. Antes de enunciarlo debemos presentar algunas nociones previas sobre caracterización de conjuntos.

Definición 48. *Conjunto acotado*
Un conjunto $S \subset \mathbb{R}^n$ es acotado si $\exists r > 0$ tal que $S \subseteq B(\boldsymbol{0}, r)$.

Definición 49. *Conjunto cerrado*
Un conjunto $S \subset \mathbb{R}^n$ es cerrado si

Para todo $\boldsymbol{a} \in \mathbb{R}^n$ tal que toda $B(\boldsymbol{a}, r)$ contiene un vector de S entonces $\boldsymbol{a} \in S$.

Definición 50. *Conjunto convexo*
Un conjunto $S \subset \mathbb{R}^n$ es convexo si $\forall \boldsymbol{x}, \boldsymbol{y} \in S$ con $\boldsymbol{x} \neq \boldsymbol{y}$

$$\forall \alpha \in [0, 1],\ \alpha \boldsymbol{x} + (1 - \alpha)\boldsymbol{y} \in S$$

Informalmente, la definición es equivalente a decir que si tomamos dos puntos distintos del conjunto, el segmento de recta entre ellos debe estar también contenido en él.

Definición 51. *Conjunto simétrico*
Un conjunto $S \subset \mathbb{R}^n$ *es simétrico si*

$$\forall \boldsymbol{x} \in S,\ -\boldsymbol{x} \in S$$

Pasamos ahora a enunciar el resultado,

Teorema 6. *Teorema del Cuerpo Convexo de Minkowski*
Para todo reticulado de rango máximo $\mathcal{L} \subset \mathbb{R}^n$*, dado un conjunto* $S \subset \mathbb{R}^n$ *acotado, convexo y simétrico con* $\mathrm{Vol}(S) > 2^n \det(\mathcal{L})$, S contiene un vector no nulo del reticulado.

Si el conjunto es cerrado, basta con $\mathrm{Vol}(S) \geq 2^n \det(\mathcal{L})$.

Entonces, tomando conjuntos acotados, convexos y simétricos del espacio, como un hipercubo o una hiperesfera, de volumen adecuado podemos acotar el largo del vector más corto del reticulado.

En particular, la prueba del Primer Teorema consiste en un caso particular del que acabamos de enunciar [10]. Consideramos un reticulado arbitrario $\mathcal{L} \subset \mathbb{R}^n$ y el conjunto S como el hipercubo en $\mathbb{R}^n$ centrado en $\mathbf{0}$ con lados de longitud $2l$,

$$S = \{(x_1, \ldots, x_n)^t \in \mathbb{R}^n : -l \leq x_i \leq l \text{ para todo } i \in \{1, \ldots, n\}\}$$

Entonces, S es cerrado, simétrico, convexo y acotado y su volumen es $\mathrm{Vol}(S) = (2l)^n$. Tomando, $l = \det(\mathcal{L})^{\frac{1}{n}}$ tenemos que $\mathrm{Vol}(S) = 2^n \det(\mathcal{L})$ y por el Teorema del Cuerpo Convexo sabemos que existe un vector no nulo $\mathbf{v} = (v_1, \ldots, v_n)^t \in S \cap \mathcal{L}$.

Ahora, como $\mathbf{v} \in S$ tenemos

$$\begin{gathered} \forall i \in \{1, \ldots, n\} - l \leq v_i \leq l \\ \Leftrightarrow v_1^2 + \cdots + v_n^2 \leq l^2 + \cdots + l^2 = n \cdot l^2 \\ \Leftrightarrow \|\mathbf{v}\| = \sqrt{v_1^2 + \cdots + v_n^2} \leq \sqrt{n} l = \sqrt{n} \cdot \det(\mathcal{L})^{\frac{1}{n}} \end{gathered}$$

que completa la prueba del Primer Teorema.

La demostración del Teorema 6 puede consultarse en [10].

Notamos que un reticulado $\mathcal{L}$ puede contener vectores arbitrariamente más cortos que la cota de Minkowski, $\sqrt{n}\det(\mathcal{L})^{\frac{1}{n}}$. Para ver que esto es posible, considerar el reticulado bidimensional generado por $\{(1,0),(0,D)\}$ con $D \gg 1$. Entonces el reticulado contiene un vector de largo $\lambda_1 = 1$, sin embargo el determinante es D y la cota de Minkowski $\sqrt{2D}$ que es mucho más grande que 1 [15].

3.8. Equivalencia de definiciones

Proposición 10. *Ambas definiciones dadas para reticulados son equivalentes.*

Demostración. (Definición $\Rightarrow$ Definición Alternativa)
Supongamos que, dado un conjunto de n vectores linealmente independiente $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\} \subset \mathbb{R}^m$, definimos un reticulado $\mathcal{L}$ como el conjunto de todas las combinaciones lineales enteras de los vectores de B.

Tenemos entonces que $\mathbf{0} = 0\mathbf{b}_1 + \cdots + 0\mathbf{b}_n \in \mathcal{L}$. Además, si $\mathbf{v} \in \mathcal{L}$, $\exists \alpha_1, \ldots, \alpha_n \in \mathbb{Z}$ tales que $\mathbf{v} = \alpha_1\mathbf{b}_1 + \cdots + \alpha_n\mathbf{b}_n$ y por tanto como $-\alpha_1, \ldots, -\alpha_n \in \mathbb{Z}$ entonces $-\mathbf{v} = -\alpha_1\mathbf{b}_1 - \cdots - \alpha_n\mathbf{b}_n \in \mathcal{L}$. Además es claro que la suma de vectores del reticulado está también en el reticulado, ya que la suma de enteros es entera. Por tanto, $\mathcal{L} < \mathbb{R}^m$.

Por el corolario de la cota inferior de la distancia mínima (Teorema 4), tenemos que el conjunto también es discreto.

(Definición Alternativa $\Rightarrow$ Definición)

Damos el algoritmo propuesto en [13].

Recordamos que la clausura del conjunto $\mathcal{P}(b_1, \ldots, b_n)$, es

$$\overline{\mathcal{P}}(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} x_i\mathbf{b}_i : x_i \in [0,1] \right\}$$

Algoritmo 2 Algoritmo de equivalencia, Definición Alternativa ⇒ Definición

Entrada: Subgrupo aditivo discreto $\mathcal{L}$ de $\mathbb{R}^m$

Salida: $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ base base de $\mathcal{L}$ según Definición 36.

1: Tomar un $\mathbf{y} \in \mathcal{L}$ tal que no exista ningún vector entre $\mathbf{0}$ e $\mathbf{y}$.
2: $\mathbf{b}_1 \leftarrow \mathbf{y}$
3: **para** $i \in \{1, \ldots, n-1\}$ **hacer**
4: {Ya se han elegido $\mathbf{b}_1, \ldots, \mathbf{b}_i$}
5: Tomar $\mathbf{y} \notin \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_i)$
6: Tomar $\mathbf{z} \in \mathcal{L} \cap \overline{\mathcal{P}}(\mathbf{b}_1, \ldots, \mathbf{b}_i, \mathbf{y}) \setminus \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_i)$ tal que $\text{dist}(\mathbf{z}, \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_i)$ sea mínimo
7: $\mathbf{b}_{i+1} \leftarrow \mathbf{z}$
8: **fin para**
9: retornar $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$

Observar que $\overline{\mathcal{P}}(\mathbf{b}_1, \ldots, \mathbf{b}_i, \mathbf{y})$ tiene al menos un vector del reticulado (contiene a $\mathbf{y} \in \mathcal{L}$) y que contiene una cantidad finita de vectores del mismo, por ser el conjunto discreto. Notar también que al elegir $\mathbf{z}$ también consideramos una cantidad finita de vectores.

Veamos ahora que la salida de este algoritmo es realmente una base del reticulado. Como $\mathcal{L} < \mathbb{R}^m$ tenemos que $\mathbf{b}_i \in \mathbb{R}^m$ para todo i y por construcción son un conjunto de vectores linelmente independientes. Entonces $\{\sum x_i \mathbf{b}_i : x_i \in \mathbb{Z}\} \subseteq \mathcal{L}$.

Resta ver que $\mathcal{L} \subseteq \{\sum x_i \mathbf{b}_i : x_i \in \mathbb{Z}\}$.

Sea $\mathbf{z} = \sum_{i=1}^{n} z_i \mathbf{b}_i \in \mathcal{L}$ arbitrario. Supongamos que los $z_i \in \mathbb{R}$. Sea $z' = \sum_{i=1}^{n} \lfloor z_i \rfloor \mathbf{b}_i \in \{\sum x_i \mathbf{b}_i : x_i \in \mathbb{Z}\} \subseteq \mathcal{L}$. Por ser $\mathcal{L}$ subgrupo aditivo,

$$z - z' = \sum_{i=1}^{n} (z_i - \lfloor z_i \rfloor) \mathbf{b}_i \in \mathcal{L}$$

Veremos que estos coeficientes deben ser enteros.

Reescribimos primero,

$$\mathbf{z} - \mathbf{z}' = (z_n - \lfloor z_n \rfloor) \mathbf{b}_n + \mathbf{y} \text{ con } \mathbf{y} \in \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$$

Entonces, como el espacio generado por los elementos de la base de Gram-Schmidt, $\mathbf{b}_i^*$, es el mismo que los de la base original tenemos

$$\mathbf{z} - \mathbf{z}' = (z_n - \lfloor z_n \rfloor)\mathbf{b}_n^* + \mathbf{y} \text{ con } \mathbf{y} \in \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}) = \text{span}(\mathbf{b}_1^*, \ldots, \mathbf{b}_{n-1}^*)$$

Donde además, $0 \leq z_n - \lfloor z_n \rfloor < 1$.

Tenemos entonces,

$$\text{dist}(\mathbf{z} - \mathbf{z}', \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})) = (z_n - \lfloor z_n \rfloor)\|\mathbf{b}_n^*\|$$

dado que la distancia es igual a la componente ortogonal de $\mathbf{z} - \mathbf{z}'$ al espacio generado por $\{\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}\}$. Esto se desprende del hecho de que $\mathbf{b}_n^*$ es por construcción ortogonal a $\text{span}(\mathbf{b}_1^*, \ldots, \mathbf{b}_{n-1}^*) = \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$.

Además,

$$\text{dist}(\mathbf{b}_n, \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})) = \|\mathbf{b}_n^*\|$$

Ya que por construcción, $\mathbf{b}_n^*$ es la componente de $\mathbf{b}_n$ que es ortogonal a $\mathbf{b}_1^*, \ldots, \mathbf{b}_{n-1}^*$.

Entonces, como $0 \leq (z_n - \lfloor z_n \rfloor) < 1$,

$$\text{dist}(\mathbf{z} - \mathbf{z}', \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}) = (z_n - \lfloor z_n \rfloor)\|\mathbf{b}_n^*\| < \|\mathbf{b}_n^*\| = \text{dist}(\mathbf{b}_n, \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}))$$

Pero como $\mathbf{b}_n$ se eligió en el algoritmo como el vector más cercano a $\text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$, debe ser por tanto,

$$\text{dist}(\mathbf{z} - \mathbf{z}', \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})) = 0$$

y por tanto $z_n - \lfloor z_n \rfloor = 0 \Leftrightarrow z_n \in \mathbb{Z}$, por ser $\mathbf{b}_n \neq \mathbf{0}$.

Repitiendo este proceso para $\mathbf{z}_{nuevo} = \mathbf{z} - z_i\mathbf{b}_i \in \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$ para todo $i \in \{1, \ldots, n\}$ obtenemos que $\forall j \in \{1, \ldots, n\}, z_j \in \mathbb{Z}$ y así probamos que ambas definiciones resultan equivalentes. □

3.9. Reticulado Dual

En esta sección definimos la noción de reticulado dual, que será utilizada en el trabajo en el Capítulo 5, junto con algunas propiedades del mismo.

Definición 52. *Reticulado Dual*
Dado un reticulado $\mathcal{L}$, definimos su dual como sigue,

$$\mathcal{L}^* = \{\boldsymbol{y} \in \text{span}(\mathcal{L}) : \forall \boldsymbol{x} \in \mathcal{L}, \langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}\}$$

Para el caso de un reticulado de rango máximo tenemos,

$$\mathcal{L}^* = \{\boldsymbol{y} \in \mathbb{R}^n : \forall \boldsymbol{x} \in \mathcal{L}, \langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}\}$$

Es decir, el dual de $\mathcal{L}$ es el conjunto de vectores en el espacio generado por una base del reticulado, cuyo producto interno con cualquier elemento de $\mathcal{L}$ es un número entero. Es posible verificar que efectivamente $\mathcal{L}^*$ es un reticulado y dar una expresión para computar una base a partir de una del reticulado primal.

Teorema 7. *El dual de un reticulado con base $B = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n] \in \mathbb{R}^{m \times n}$ es un reticulado con base $D = B(B^tB)^{-1}$*[1]

Demostración. Hacemos la prueba por doble inclusión de los conjuntos. Comenzamos probando que $\mathcal{L}(D) \subseteq \mathcal{L}^*(B)$. Primero, observamos que dado $\mathbf{y} \in \mathbb{Z}^n$ arbitrario, considerando $D\mathbf{y} \in \mathcal{L}(D)$ tenemos,

- $D\mathbf{y} = B(B^tB)^{-1}\mathbf{y} = B((B^tB)^{-1}\mathbf{y}) \in \text{span}(B)$, dado que $B^tB \in \mathbb{R}^{n \times n}$ y por tanto $(B^tB)^{-1}\mathbf{y} \in \mathbb{R}^m$.
- Para todo $x \in \mathbb{Z}^n$, $B\mathbf{x} \in \mathcal{L}(B)$, tenemos $(D\mathbf{y})^t(B\mathbf{x}) = \mathbf{y}^t\mathbf{x} \in \mathbb{Z}$ ya que ambos tienen coordenadas enteras.

De lo anterior $D\mathbf{y} \in \mathcal{L}^*(B)$ por definición y por tanto probamos la primera inclusión.

Ahora, sea $\mathbf{v} \in \mathcal{L}^*(B)$ arbitrario. Por definición de dual, $\mathbf{v} \in \text{span}(B)$ y además $B^t\mathbf{v} \in \mathbb{Z}^n$ dado que $(B\mathbf{x})^t\mathbf{v} \in \mathbb{Z} \Leftrightarrow \mathbf{x}^tB^t\mathbf{v} \in \mathbb{Z}$ y $\mathbf{x} \in \mathbb{Z}^n$. Entonces de estos hechos,

[1] La matriz $G = (B^tB)$ es a veces denominada matriz de Gram y es invertible si y solo si las columnas de B son linealmente independientes. En nuestro caso como B es base del reticulado esto último se cumple y por tanto D está bien definida.

- $\mathbf{v} = B\mathbf{w}$ para un $\mathbf{w} \in \mathbb{R}^n$
- $\mathbf{v} = B\mathbf{w} = (B(B^tB)^{-1}B^t)B\mathbf{w} = (DB^t)B\mathbf{w} = D\mathbf{w} = D(B^t\mathbf{v})$ donde la última igualdad utiliza el primer punto y por tanto $\mathbf{v} = D(B^t\mathbf{v}) \in \mathcal{L}(D)$.

□

Notar que si el reticulado es de rango máximo, entonces $D = (B^t)^{-1}$.

Proposición 11. *Para todo reticulado $\mathcal{L}$,*

- $(\mathcal{L}^*)^* = \mathcal{L}$
- $\det(\mathcal{L}^*) = \frac{1}{\det(\mathcal{L})}$

Definimos ahora la noción de defecto ortogonal para el caso del reticulado dual.

Definición 53. *Defecto ortogonal dual*
Dada una base $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ de un reticulado de rango máximo, definimos su defecto ortogonal dual como

$$\text{def\_ort}^*(B) = \frac{\prod_{i=1}^{n} \|\hat{\boldsymbol{b}}_i\|}{|\det(B^{-1})|} = |\det(B)| \cdot \prod_{i=1}^{n} \left\|\hat{\boldsymbol{b}}_i\right\|$$

donde $\hat{\boldsymbol{b}}_i$ es el vector correspondiente a la i-ésima fila de B^{-1}, es decir el i-ésimo vector columna de $(B^{-1})^t = (B^t)^{-1}$.

Notamos que el defecto ortogonal dual para una base no es más que el defecto ortogonal de la base del reticulado dual correspondiente, dado que al ser $(B^t)^{-1} = (B^{-1})^t$ base del dual, cuyos vectores son los $\hat{\mathbf{b}}_i^t$,

$$\text{def\_ort}((B^t)^{-1}) = \frac{\prod_{i=i}^{n} \left\|\hat{\mathbf{b}}_i^t\right\|}{|\det((B^t)^{-1})|} = \prod_{i=i}^{n} \left\|\hat{\mathbf{b}}_i\right\| \cdot |det(B^t)| = \prod_{i=i}^{n} \left\|\hat{\mathbf{b}}_i\right\| \cdot |det(B)|$$

Capítulo 4

Problemas Computacionales en Reticulados y Algoritmos de Resolución

Como vimos en el Capítulo 3, los teoremas de Minkowski (Teorema 5 y Teorema 6) permiten acotar el largo del vector más corto de un reticulado, sin embargo pueden existir vectores más cortos arbitrariamente más pequeños en algunos reticulados. Como la prueba del primer teorema (Teorema 5) no es constructiva, es decir que no propone un método por el que se pueda obtener un vector más corto, resulta de interés desde un punto de vista computacional desarrollar métodos eficientes para encontrar estos vectores y también estudiar su dificultad en términos computacionales.

Para el abordaje de estos temas nos basamos en [12], [16], [10] y [17].

4.1. Problemas Computacionales Fundamentales

Para formalizar estas ideas se definen distintos problemas computacionales (concepto introducido en el Capítulo 2) vinculados a reticulados que presentamos a continuación. Tenemos los siguientes problemas fundamentales [12],

Definición 54. *Problema del vector más corto (SVP)*[1]
Dada una base $B \in \mathbb{Z}^{m \times n}$ *de un reticulado* $\mathcal{L}$*,*

- *Búsqueda: Encontrar un vector* $\boldsymbol{v} \in \mathcal{L} \setminus \{\boldsymbol{0}\}$ *tal que* $\|\boldsymbol{v}\| = \lambda_1(\mathcal{L})$.
- *Optimización: Encontrar* $\lambda_1(\mathcal{L})$.
- *Decisión: Dado un número racional* $r \in \mathbb{Q}$*, determinar si* $\lambda_1(\mathcal{L}) \leq r$ *o no.*

Observamos que en la definición anterior y en las que siguen nos restringimos a tomar vectores enteros como elementos de la base en lugar de vectores reales arbitrarios, para evitar tener problemas de representación. En este sentido, nos aseguramos que la entrada de los problemas que consideramos pueda ser representada con una cantidad finita de bits. Como mencionamos antes, podríamos también trabajar con vectores racionales, ya que podemos multiplicando por un factor de ajuste adecuado transformarlos al primer caso de manera eficiente.

Resulta claro de la Definición 54, que la versión de decisión no es más difícil que la de optimización y que esta no es más difícil que la de búsqueda. En este caso, se puede probar también el recíproco, que la versión de búsqueda no es más difícil que la de optimización y que esta no lo es más que la de decisión. De esto podemos afirmar que las tres variantes son esencialmente equivalentes [16].

Otra variante de estos problemas que resulta de interés, en particular cuando las instancias exactas resultan difíciles de resolver, es considerar aproximaciones en las soluciones. En general, para los problemas de aproximación, se toma como factor de aproximación un valor funcional $\gamma(n)$, donde γ es función del rango del reticulado n. Esto busca usualmente capturar el hecho de que el problema se vuelve más difícil a medida que n aumenta [12].

Definición 55. *Problema del vector más corto - Aproximación (*SVP_γ*)*
Dada una base $B \in \mathbb{Z}^{m \times n}$ *de un reticulado* $\mathcal{L}$ *y un* $\gamma(n) > 1$*,*

- *Búsqueda: Encontrar un vector* $\boldsymbol{v} \in \mathcal{L} \setminus \{\boldsymbol{0}\}$ *tal que* $\|\boldsymbol{v}\| = \gamma(n)\lambda_1(\mathcal{L})$.

[1]Sigla proveniente del término en inglés, *Shortest Vector Problem.*

- *Optimización: Encontrar* $d \in \mathbb{Q}$ *tal que* $d \leq \lambda_1(\mathcal{L}) \leq \gamma(n) \cdot d$.

- *Promesa: Dado un* $r \in \mathbb{Q}$ *y los siguientes conjuntos disjuntos de instancias,*

$$SI = \{(B, r) : \lambda_1 \leq r\}$$
$$NO = \{(B, r) : \lambda_1 > \gamma(n) \cdot r\}$$

  el problema consiste en distinguir correctamente dada una instancia $I \in SI \cup NO$*, a cuál de estos conjuntos pertenece* I.

La última variante suele ser denotada por GapSVP_γ. Es, como mencionamos en la definición, un problema de *promesa*. En este tipo de problemas tenemos dos conjuntos de instancias disjuntos, uno llamado SI y otro NO y el objetivo es distinguir de qué conjunto fue tomada la instancia recibida. A diferencia de los problemas de decisión, la unión $SI \cup NO$ no debe contener todas las posibles instancias del problema. En el caso de que la instancia no pertenezca a $SI \cup NO$, se toma como válida cualquiera de las dos respuestas [12].

Al igual que en el caso anterior, tenemos que para cualquier $\gamma(n) > 1$, la variante de promesa no es más difícil que la de optimización y que esta no lo es más que la de búsqueda. También se cumple que la variante de optimización no es más difícil que la de promesa, sin embargo es un problema abierto si la variante de búsqueda no es más difícil que la de optimización [16].

Seguimos con otro de los problemas fundamentales, similar al anterior, donde el objetivo es encontrar el vector del reticulado más cercano a otro del espacio. En las definiciones $\text{dist}(\mathcal{L}, \mathbf{t}) = \min_{\mathbf{v} \in \mathcal{L}} \|\mathbf{v} - \mathbf{t}\|$.

Definición 56. *Problema del vector más cercano (CVP)*[1]
Dada una base $B \in \mathbb{Z}^{m \times n}$ *de un reticulado* $\mathcal{L}$ *y un* $\boldsymbol{t} \in \mathbb{Z}^m$,

- *Búsqueda: Encontrar un vector* $\boldsymbol{v} \in \mathcal{L}$ *tal que* $\text{dist}(\boldsymbol{v}, \boldsymbol{t}) = \|\boldsymbol{v} - \boldsymbol{t}\|$ *sea mínima.*

- *Optimización: Encontrar* $\text{dist}(\mathcal{L}, \boldsymbol{t})$.

[1] Sigla proveniente del término en inglés, *Closest Vector Problem*.

- *Decisión: Dado un racional* $r \in \mathbb{Q}, r > 0$ *decidir si existe un* $\boldsymbol{v} \in \mathcal{L} \setminus \{\boldsymbol{0}\}$ *tal que* $\operatorname{dist}(\mathcal{L}, \boldsymbol{t}) \leq r$.

También tenemos las variantes de aproximación asociadas,

Definición 57. *Problema del vector más cercano - Aproximación (CVP*$_\gamma$*)*
Dada una base $B \in \mathbb{Z}^{m \times n}$ *de un reticulado* $\mathcal{L}$*, un* $\boldsymbol{t} \in \mathbb{Z}^m$ *y* $\gamma(n) > 1$*,*

- *Búsqueda: Encontrar un vector* $\boldsymbol{v} \in \mathcal{L} \setminus \{\boldsymbol{0}\}$ *tal que* $\operatorname{dist}(\boldsymbol{v}, \boldsymbol{t}) = \|\boldsymbol{v} - \boldsymbol{t}\| \leq \gamma(n) \operatorname{dist}(\mathcal{L}, \boldsymbol{t})$.
- *Optimización: Encontrar un* d *tal que* $d \leq \operatorname{dist}(\boldsymbol{t}, \mathcal{L}) \leq \gamma(n) \cdot d$.
- *Promesa: Dado un* $r \in \mathbb{Q}$ *y los siguientes conjuntos disjuntos de instancias,*

$$SI = \{(B, t, r) : \operatorname{dist}(\mathcal{L}, \boldsymbol{t}) \leq r\}$$
$$NO = \{(B, t, r) : \operatorname{dist}(\mathcal{L}, \boldsymbol{t}) > \gamma(n) \cdot r\}$$

el problema consiste en distinguir correctamente dada una instancia $I \in SI \cup NO$*, a cuál de estos conjuntos pertenece* I.

A la última versión se la denomina GapCVP$_\gamma$.

Como veremos más adelante, tanto SVP así como CVP son problemas computacionalmente difíciles, incluso con frecuencia cuando consideramos sus versiones aproximadas.

Concluimos con un problema vinculado al mínimo sucesivo de orden n.

Definición 58. *Problema de vectores independientes más cortos - Búsqueda (SIVP)*[1]
Dada una base $B \in \mathbb{Z}^{n \times n}$ *de un reticulado* $\mathcal{L}$*, encontrar un conjunto de* n *vectores independientes* $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ *tales que* $\forall i \in \{1, \ldots, n\}$ $\|\boldsymbol{v}_i\| \leq \lambda_n(\mathcal{L})$.

Definición 59. *Problema de vectores independientes más cortos - Aproximación (SIVP*$_\gamma$*)*
Dada una base $B \in \mathbb{Z}^{n \times n}$ *de un reticulado* $\mathcal{L}$ *y un* $\gamma(n) > 1$*, encontrar un conjunto de* n *vectores independientes* $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ *tales que* $\forall i \in \{1, \ldots, n\}$*,* $\|\boldsymbol{v}_i\| \leq \gamma(n)\lambda_n(\mathcal{L})$.

Continuamos analizando la complejidad de estos problemas.

[1] Sigla proveniente del término en inglés, *Shortest Independent Vectors Problem.*

4.2. Complejidades Asociadas

Comenzamos con el problema del vector más cercano.

Teorema 8. *[18] Complejidad de CVP*
*CVP en su versión de decision es **NP**-completo.*

Para mostrar que el problema está en **NP** (Definición 5), notamos que para cualquier instancia $(B, \mathbf{t}, r)$ tal que $\text{dist}(\mathbf{t}, \mathcal{L}(B)) \leq r$ podemos tomar como "certificado" a un vector $\mathbf{x} \in \mathcal{L}$ que cumpla $\|\mathbf{x} - \mathbf{t}\| \leq r$. Verificar si $\mathbf{x} \in \mathcal{L}$ y $\|\mathbf{x} - \mathbf{t}\| \leq r$ puede realizarse en tiempo polinomial y además podemos representar $\mathbf{x}$ con una cantidad polinomial de bits en el tamaño de la entrada, ya que es un vector a coeficientes enteros cuyas entradas son a lo sumo $\|\mathbf{t}\| + r$ en valor absoluto [12].

Para probar que CVP es **NP**-difícil, se muestra que SUBSETSUM $\leq_p$ CVP, dado que SUBSETSUM es **NP**-difícil. Una instancia de SUBSETSUM viene dada por $n + 1$ enteros $a_1, \ldots, a_n, S \in \mathbb{Z}$ y el objetivo es decidir si existe un $A \subseteq \{1, \ldots, n\}$ tal que $\sum_{i \in A} a_i = S$ [12].

El siguiente resultado indica que para cualquier $\gamma \geq 1$ encontrar soluciones a SVP_γ no es más difícil que encontrar soluciones a CVP_γ.

Teorema 9. *[16]* $\forall \gamma \geq 1$ *dado acceso a un oráculo que resuelva* $GapCVP_\gamma$*, es posible resolver* $GapSVP_\gamma$ *en tiempo polinomial.*

Esto en un principio podría parecer intuitivo, ya que dada una instancia de SVP se nos podría ocurrir utilizar una rutina que resuelva CVP con vector objetivo igual al vector nulo $\mathbf{0}$. Sin embargo, esto no funciona, ya que para todo reticulado $\mathcal{L}, \mathbf{0} \in \mathcal{L}$ y por tanto, el vector más cercano será él mismo. En [16] también se explica el problema con otro enfoque posible, que consiste en generar un "agujero" en el reticulado, eliminando un vector del mismo y luego usar el oráculo CVP con el vector eliminado como objetivo. El problema con este enfoque, es que podríamos estar eliminando elementos que hagan que el conjunto resultante deje de ser un reticulado. El enfoque de la prueba se base en repetir este proceso, considerando las precauciones para evitar lo anterior, hasta lograr las condiciones adecuadas.

Para el caso aproximado tenemos el siguiente resultado,

Teorema 10. *[19] Para toda constante* $c > 0$ *y* $\gamma(n) = n^{\frac{c}{\log(\log(n))}}$,

$$GapCVP_{\gamma} \in \textbf{\textit{NP}}\textit{-difícil}$$

En cuanto al problema del vector más corto, GapSVP_{γ}, se conjetura [20] que no existen algoritmos clásicos que puedan lograr soluciones aproximadas con factores de aproximación polinomiales. Oded Regev, autor de [20], incluso considera la conjetura de que no existan algoritmos cuánticos que puedan resolver este problema con factores de aproximación polinomiales. Las mismas conjeturas aplican para el caso del problema SIVP.

Observamos que cuando un problema es **NP**-completo, esto refiere a su dificultad en el *peor caso*. Por tanto, esto no es una condición suficiente para obtener construcciones criptográficas seguras, dado que en general como veremos en los Capítulos 5, 6 y 7, necesitaremos que las instancias de los problemas subyacentes utilizadas sean difíciles en el *caso promedio*. Dicho de otra forma, queremos que al tomar una instancia del problema según una distribución particular, esta sea difícil a menos de probabilidad negligible. Entonces que un problema sea **NP**-completo, es un buen indicio (ver Sección 2.1) para la no existencia de algoritmos eficientes que resuelvan *todas* las instancias, pero no es suficiente para construir esquemas seguros. Notar que esto es distinto a cuando realizamos una prueba de seguridad por reducción (Sección 2.2.2) a un problema determinado, donde establecemos que quebrar el sistema de encriptado es al menos tan difícil como resolver el problema computacional, ambos en el peor caso.

Las construcciones criptográficas basadas en reticulados que hacen uso de las versiones aproximadas de los problemas de esta sección, toman factores de aproximación al menos lineales [13] como la que veremos en el Capítulo 7. Además, observando los resultados de la dificultad computacional de estos problemas, no resulta sorprendente que como veremos en la siguiente sección, no existan algoritmos eficientes para resolver dichos problemas en sus formas exactas e incluso para algunos tipos de factores de aproximación.

Lo expuesto en este capítulo es evidencia que respalda la conjetura (aceptada por la comunidad científica) [7] comentada en la Sección 1.2, que indica que no existen algoritmos clásicos ni cuánticos que puedan resolver este tipo de

problemas en reticulados para factores de aproximación al menos polinomiales en el rango de los mismos.

4.3. Algoritmos de Resolución

Presentamos ahora tres algoritmos que resuelven instancias de SVP_γ y CVP_γ. Veamos primero qué condiciones son deseables sobre los elementos de la base de un reticulado para que pueda ser utilizada para la resolución de estos problemas. [10]

Notamos que, dado un reticulado $\mathcal{L} \subset \mathbb{R}^n$, si su base $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ consta de vectores que son ortogonales entre ellos, es decir que $\langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0$ para los $i \neq j$, entonces resulta sencillo resolver SVP y CVP en sus versiones exactas. Veamos cómo resolverlos a continuación.

Para el primer problema, observamos que $\forall \mathbf{w} \in \mathcal{L}$, $\exists \alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{Z}$ tal que,

$$\|\mathbf{w}\|^2 = \|\alpha_1 \mathbf{b}_1 + \alpha_2 \mathbf{b}_2 + \cdots + \alpha_n \mathbf{b}_n\|^2$$

utilizando la definición de norma, las propiedades del producto interno y que la base es ortogonal tenemos,

$$\begin{aligned} \|\mathbf{w}\|^2 &= \langle \alpha_1 \mathbf{b}_1 + \alpha_2 \mathbf{b}_2 + \cdots + \alpha_n \mathbf{b}_n, \alpha_1 \mathbf{b}_1 + \alpha_2 \mathbf{b}_2 + \cdots + \alpha_n \mathbf{b}_n \rangle = \\ &= \alpha_1 \alpha_1 \langle \mathbf{b}_1, \mathbf{b}_1 \rangle + \alpha_2 \alpha_2 \langle \mathbf{b}_2, \mathbf{b}_2 \rangle + \cdots + \alpha_n \alpha_n \langle \mathbf{b}_n, \mathbf{b}_n \rangle = \\ &= \alpha_1^2 \|\mathbf{b}_1\|^2 + \alpha_2^2 \|\mathbf{b}_2\|^2 + \cdots + \alpha_n^2 \|\mathbf{b}_n\|^2 \end{aligned}$$

Como los coeficientes deben ser enteros, tenemos entonces que los vectores más cortos en $\mathcal{L}$ deben ser los más cortos del conjunto $\{\pm\mathbf{b}_1, \ldots, \pm\mathbf{b}_n\}$.

Con un enfoque análogo podemos buscar el vector más cercano del reticulado a un cierto $\mathbf{t} \in \mathbb{R}^n$,

$$\mathbf{t} = \beta_1 \mathbf{b}_1 + \beta_2 \mathbf{b}_2 + \cdots + \beta_n \mathbf{b}_n \text{ con } \beta_1, \beta_2, \ldots, \beta_n \in \mathbb{R}$$

Entonces dado un $\mathbf{w} \in \mathcal{L}$,

$$\mathbf{w} = \alpha_1 \mathbf{b}_1 + \alpha_2 \mathbf{b}_2 + \cdots + \alpha_n \mathbf{b}_n$$

tenemos

$$\|\mathbf{w}-\mathbf{t}\|^2 = \|(\alpha_1-\beta_1)\mathbf{b}_1+\cdots+(\alpha_n-\beta_n)\mathbf{b}_n\|^2 = \\ = \langle(\alpha_1-\beta_1)\mathbf{b}_1+\cdots+(\alpha_n-\beta_n)\mathbf{b}_n, (\alpha_1-\beta_1)\mathbf{b}_1+\cdots+(\alpha_n-\beta_n)\mathbf{b}_n\rangle \\ = (\alpha_1-\beta_1)^2\|\mathbf{b}_1\|^2+\cdots+(\alpha_n-\beta_n)^2\|\mathbf{b}_n\|^2$$

y como los $\alpha_i \in \mathbb{Z}$, tenemos que esta expresión se minimiza si tomamos $\alpha_i = \lfloor \beta_i \rceil$, es decir si lo tomamos como el entero más cercano al β_i correspondiente.

De lo anterior queda claro que cuanto más cortos y ortogonales son los vectores de la base, se podrán obtener mejores soluciones a los problemas del vector más corto y más cercano usando las estrategias explicadas.

Nos interesamos entonces por encontrar algoritmos que permitan reducir una base arbitraria de un reticulado a una que cumpla las condiciones que mencionamos anteriormente. Denominamos a este proceso de reducción de base de un reticulado.

4.3.1. Algoritmo LLL

Para el caso de bases de rango máximo en dos dimensiones, existe un algoritmo para reducción de bases de reticulados desarrollado por Lagrange y Gauss, que puede consultarse en [17] [10]. En nuestra exposición, nos enfocamos en una generalización de dicho algoritmo, propuesto originalmente en 1982 por Lenstra, Lenstra y Lovász, conocido como algoritmo LLL [21].

Dicho algoritmo hace uso del proceso de ortogonalización de Gram-Schmidt que presentamos en el Capítulo 2. Observamos que en general los coeficientes utilizados en el proceso, $\mu_{i,j}$, no son números enteros y por lo tanto en general los vectores resultantes no forman parte del reticulado considerado. El algoritmo LLL se asegura que las combinaciones lineales usadas para actualizar los vectores del reticulado posean todas coeficientes en $\mathbb{Z}$ [17].

Base LLL-reducida

Damos comienzo a la exposición introduciendo la definición central de [21], que busca caracterizar la base a ser producida por el algoritmo.

Definición 60. *Base LLL-reducida*
Sea $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *una base para un reticulado* $\mathcal{L}$ *y* $B^* = \{\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*\}$ *la base de Gram-Schmidt correspondiente. Fijando un* $\delta : \frac{1}{4} < \delta < 1$*, decimos que* B *es una base LLL-reducida si se cumplen las dos condiciones siguientes,*

- ***Reducción de tamaño****:* $|\mu_{i,j}| = \frac{|\langle \boldsymbol{b}_i, \boldsymbol{b}_j^* \rangle|}{\|\boldsymbol{b}_j^*\|^2} \leq \frac{1}{2}$ *para todo* $1 \leq j < i \leq n$
- ***Condición de Lovász****:* $\|\boldsymbol{b}_i^*\|^2 \geq (\delta - \mu_{i,i-1}^2)\|\boldsymbol{b}_{i-1}^*\|^2$ *para todo* $2 \leq i \leq n$

Tradicionalmente, se toma $\delta = \frac{3}{4}$*, siendo un valor mayor usado actualmente en la práctica [17].*

El resultado fundamental de Lenstra, Lenstra y Lovász es que una base LLL-reducida tiene propiedades deseables para resolver SVP_γ y CVP_γ para factores exponenciales y que es posible computarla en tiempo polinomial. Veamos algunos resultados con respecto a estas propiedades.

Teorema 11. *[10] Propiedades base LLL-reducida*
Sea $\mathcal{L}$ *un reticulado de dimensión* n*. Entonces tomando* $\delta = \frac{3}{4}$*, cualquier base LLL-reducida* $B = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_n\}$ *de* $\mathcal{L}$ *cumple las siguientes propiedades,*

1. $\prod_{i=1}^{n} \|\boldsymbol{b}_i\| \leq 2^{\frac{n(n-1)}{4}} |\det(\mathcal{L})|$
2. $\|\boldsymbol{b}_j\| \leq 2^{\frac{(i-1)}{2}} \|\boldsymbol{b}_i^*\|$ *para todo* $1 \leq j \leq i \leq n$

Además el primer vector de la base LLL-reducida cumple,

3. $\|\boldsymbol{b}_1\| \leq 2^{\frac{(n-1)}{4}} \cdot |\det(\mathcal{L})|^{\frac{1}{n}}$
4. $\|\boldsymbol{b}_1\| \leq 2^{\frac{(n-1)}{2}} \cdot \lambda_1$

De lo anterior se obtiene como corolario que una base LLL-reducida resuelve SVP_γ con $\gamma = 2^{\frac{n-1}{2}}$.

Demostración. Por la condición de tamaño tenemos para $j = i-1$, $|\mu_{i,i-1}| \leq \frac{1}{2}$. Junto a la condición de Lovász resulta,

$$\|\mathbf{b}_i^*\|^2 \geq (\tfrac{3}{4} - \mu_{i,i-1}^2)\|\mathbf{b}_{i-1}^*\|^2 \geq \tfrac{1}{2} \cdot \|\mathbf{b}_{i-1}^*\|^2$$

Por tanto, como $\|\mathbf{b}_i^*\|^2 \geq 2^{-1}\|\mathbf{b}_{i-1}^*\|^2$, repitiendo esto por inducción se tiene

$$\|\mathbf{b}_i^*\|^2 \geq 2^{j-i}\left\|\mathbf{b}_j^*\right\|^2 \Leftrightarrow 2^{i-j}\|\mathbf{b}_i^*\|^2 \geq \left\|\mathbf{b}_j^*\right\|^2$$

Por el proceso de Gram-Schmidt, tenemos

$$\mathbf{b}_i = \mathbf{b}_i^* + \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^*$$

por tanto,

$$\|\mathbf{b}_i\|^2 = \left\|\mathbf{b}_i^* + \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^*\right\|^2 = \langle \mathbf{b}_i^* + \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^*, \mathbf{b}_i^* + \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^* \rangle$$

y como $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$ son ortogonales,

$$\|\mathbf{b}_i\|^2 = \|\mathbf{b}_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2\left\|\mathbf{b}_j^*\right\|^2 \leq \|\mathbf{b}_i^*\|^2 + \sum_{j=1}^{i-1} \tfrac{1}{4}\left\|\mathbf{b}_j^*\right\|^2 \text{ ya que } |\mu_{i,j}| \leq \tfrac{1}{2}$$

Ahora, usando como vimos anteriormente que $\left\|\mathbf{b}_j^*\right\|^2 \leq 2^{i-j}\|\mathbf{b}_i^*\|^2$,

$$\begin{aligned}\|\mathbf{b}_i\|^2 \leq \|\mathbf{b}_i^*\|^2 + \sum_{j=1}^{i-1} \tfrac{1}{4}2^{i-j}\|\mathbf{b}_i^*\|^2 &= \|\mathbf{b}_i^*\|^2 + \sum_{j=1}^{i-1} 2^{i-j-2}\|\mathbf{b}_i^*\|^2 \\ &= (1 + \sum_{j=1}^{i-1} 2^{i-j-2})\|\mathbf{b}_i^*\|^2\end{aligned}$$

Como $\sum_{j=1}^{i-1} 2^{i-j} = 2^i - 2$, el término entre paréntesis resulta,

$$\begin{aligned}1 + \sum_{j=1}^{i-1} 2^{i-j-2} = 1 + 2^{-2}\sum_{j=1}^{i-1} 2^{i-j} = 2^{-2}(2^2 + \sum_{j=1}^{i-1} 2^{i-j}) = \\ = 2^{-2}(2^2 + 2^i - 2) = 2^{-2}(2 + 2^i) = \tfrac{1+2^{i-1}}{2}\end{aligned}$$

Luego, $\|\mathbf{b}_i\|^2 \leq \dfrac{1 + 2^{i-1}}{2}\|\mathbf{b}_i^*\|^2 \leq 2^{i-1}\|\mathbf{b}_i^*\|^2$.

Esto último se desprende de

$$\forall i \geq 1, 1 \leq 2^{i-1} \Leftrightarrow \tfrac{1}{2} \leq \tfrac{2^{i-1}}{2} \Leftrightarrow \tfrac{1}{2} + \tfrac{2^{i-1}}{2} \leq 2^{i-1} \Leftrightarrow \tfrac{1+2^{i-1}}{2} \leq 2^{i-1}$$

Entonces multiplicando $\|\mathbf{b}_i\|^2$ para $i \in \{1, \ldots, n\}$ obtenemos

$$\prod_{i=1}^{n} \|\mathbf{b}_i\|^2 \leq \prod_{i=1}^{n} 2^{i-1}\|\mathbf{b}_i^*\|^2 = 2^{\frac{(n-1)n}{2}} \prod_{i=1}^{n} \|\mathbf{b}_i^*\|^2$$

ya que $\prod_{i=1}^{n} 2^{i-1} = 2^{\sum_{i=1}^{n} i-1} = 2^{\sum_{j=0}^{n-1} j} = 2^{\frac{(n-1)n}{2}}$.

Luego, como vimos en el Capítulo 3, se cumple $\det(\mathcal{L}) = \prod_{i=1}^{n} \|\mathbf{b}_i^*\|$, entonces resulta,

$$\prod_{i=1}^{n} \|\mathbf{b}_i\|^2 \leq 2^{\frac{(n-1)n}{2}} \det(\mathcal{L})^2$$

Tomando raíz cuadrada obtenemos $\prod_{i=1}^{n} \|\mathbf{b}_i\| \leq 2^{\frac{(n-1)n}{4}} |\det(\mathcal{L})|$, punto 1 del teorema.

Para ver el punto 2 tomamos primero el resultado anterior,

$$\|\mathbf{b}_j\|^2 \leq 2^{j-1} \left\|\mathbf{b}_j^*\right\|$$

y lo combinamos con $\left\|\mathbf{b}_j^*\right\|^2 \leq 2^{i-j} \|\mathbf{b}_i^*\|^2$ para $j \leq i$ obteniendo

$$\|\mathbf{b}_j\|^2 \leq 2^{j-1} \cdot 2^{i-j} \|\mathbf{b}_i^*\|^2 = 2^{i-1} \|\mathbf{b}_i^*\|^2$$

Tomando raíz cuadrada tenemos el punto 2.

En cuanto al punto 3 fijamos $j = 1$ en el resultado del punto 2 y considerando la multiplicación para los distintos $i \in \{1, \ldots, n\}$ tenemos

$$\|\mathbf{b}_1\|^n \leq \prod_{i=1}^{n} 2^{\frac{i-1}{2}} \|\mathbf{b}_i^*\|$$

Tomando raíz n-ésima,

$$\|\mathbf{b}_1\| \leq (\prod_{i=1}^{n} 2^{\frac{i-1}{2}})^{1/n} \cdot |\det(\mathcal{L})|^{1/n} = 2^{(\frac{(n-1)n}{4})^{1/n}} |\det(\mathcal{L})|^{1/n}$$

Luego,

$$\|\mathbf{b}_1\| \leq 2^{\frac{(n-1)}{4}} |\det(\mathcal{L})|^{1/n}$$

Para probar el punto 4, comenzamos recordando la cota inferior presentada en el Capítulo 3 para el largo del vector más corto,

$$\lambda_1(\mathcal{L}(B)) \geq \min_{i \in \{1,\ldots,n\}} \|\mathbf{b}_i^*\| > 0$$

Según el punto 2, $\forall 1 \leq j \leq i \leq n, \|\mathbf{b}_j\| \leq 2^{\frac{i-1}{2}} \|\mathbf{b}_i^*\|$. En particular, para $j = 1$,

$$\|\mathbf{b}_1\| \leq 2^{\frac{i-1}{2}} \|\mathbf{b}_i^*\| \text{ para toda } i \in \{1, \ldots, n\}$$

$$\Leftrightarrow 2^{\frac{1-i}{2}} \|\mathbf{b}_1\| \leq \|\mathbf{b}_i^*\|$$

Luego,

$$\lambda_1 \geq \min_{i \in \{1,\ldots,n\}} \|\mathbf{b}_i^*\| \geq \min_{i \in \{1,\ldots,n\}} 2^{\frac{1-i}{2}} \|\mathbf{b}_1\| = 2^{\frac{(1-n)}{2}} \|\mathbf{b}_1\|$$

De donde,

$$2^{\frac{n-1}{2}} \cdot \lambda_1 \geq \|\mathbf{b}_1\|$$

□

Algoritmo LLL y tiempo de ejecución

Presentamos el algoritmo LLL para reducción de bases a continuación.

Algoritmo 3 Algoritmo LLL

Entrada: $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ base de un reticulado $\mathcal{L} \subset \mathbb{R}^n$, $\delta \in (\frac{1}{4}, 1)$
Salida: $\hat{B} = \{\hat{\mathbf{b}_1}, \hat{\mathbf{b}_2}, \ldots, \hat{\mathbf{b}_n}\}$ base LLL-reducida de $\mathcal{L}$

1: Computar la base de Gram-Schmidt $B^* = \{\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*\}$ de B y coeficientes $\mu_{i,j}$ para $1 \leq j < i \leq n$
2: $B_i \leftarrow \|\mathbf{b}_i^*\|^2 = \langle \mathbf{b}_i^*, \mathbf{b}_i^* \rangle$ para $1 \leq i \leq n$
3: $k \leftarrow 2$
4: **mientras** $k \leq n$ **hacer**
5: {Reduccción de tamaño}
6: **para** $j = k - 1$ hasta 1 **hacer**
7: $q_j \leftarrow \lfloor \mu_{k,j} \rceil$
8: $\mathbf{b}_k \leftarrow \mathbf{b}_k - q_j \cdot \mathbf{b}_j$
9: Actualizar valor de $\mu_{k,j}$ para todo $1 \leq j < k$
10: **fin para**
11: {Verificar Condición de Lovász}
12: **si** $B_k \geq (\delta - \mu_{k,k-1}^2) B_{k-1}$ **entonces**
13: $k \leftarrow k + 1$
14: **si no**
15: {Paso de intercambio}
16: Intercambiar $\mathbf{b}_k$ con $\mathbf{b}_{k-1}$
17: Actualizar $\mathbf{b}_k^*, \mathbf{b}_{k-1}^*, B_k, B_{k-1}, \mu_{k-1,j}$ para $1 \leq j < k$
18: Actualizar $\mu_{i,k}, \mu_{i,k-1}$ para $k < i \leq n$
19: $k \leftarrow \text{máx}\{2, k - 1\}$
20: **fin si**
21: **fin mientras**
22: **devolver** $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$

En la línea 7, $\lfloor x \rceil$ denota al entero más cercano a $x \in \mathbb{R}$. Como comentamos al comienzo de esta sección (4.3.1), el resultado principal es el siguiente,

Teorema 12. *Algoritmo LLL - Correctitud y cantidad de iteraciones*
Sea $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *una base para un reticulado* $\mathcal{L} \subset \mathbb{Z}^n$[1] *y sea* $T =$

[1]Trabajamos con números enteros o racionales para evitar problemas de representación.

$\max_{i\in\{1,\dots,n\}} \|\boldsymbol{b}_i\|$. *Entonces el algoritmo LLL (Algoritmo 3) retorna una base LLL-reducida para $\mathcal{L}$ y el ciclo principal (líneas 4 a 21, Algoritmo 3) se ejecuta no más de $O(n^2 \log(T))$ iteraciones.*

Primero, enunciamos una proposición que prueba la correctitud del algoritmo, asumiendo que este termina.

Proposición 12. *Si el algoritmo LLL (Algoritmo 3) termina, la salida es una base LLL-reducida.*

Demostración. El hecho de que la salida sea efectivamente una base para el reticulado se desprende de que solo realizamos operaciones de la forma $\mathbf{b}_i \leftarrow \mathbf{b}_i + k\mathbf{b}_j$ para $i \neq j$ y $k \in \mathbb{Z}$ sobre los elementos de la base original. [16] Como dicho tipo de operación se corresponde con la aplicación de una transformación unimodular sobre los elementos de la base original, estos preservan la base. En forma matricial, podemos ver esto como la multiplicación de la matriz de vectores de la base con la matriz unimodular correspondiente a la transformación. Como la multiplicación de matrices unimodulares es, como vimos en el Capítulo 2, también unimodular tenemos que la aplicación sucesiva de estas transformaciones preservan la base. En la demostración de la Proposición 13 vemos como construir la matriz de este tipo de transformación.

La condición de Lovász se verifica trivialmente dado el **si** de la línea 12. En particular, el hecho de que $k = n + 1$ al finalizar la ejecución asegura que todos los vectores han pasado la condición del **si** de la línea 12, ya que esta es la única forma de incrementar el valor de k.

Para la satisfacibilidad de la condición de tamaño, realizamos la demostración en tres pasos (ejercicio en [17]). Sean $j, k \in \{1, \dots, n\}$ tales que $1 \leq j < k$ y $\mathbf{b}'_k = \mathbf{b}_k - \lfloor \mu_{k,j} \rceil \mathbf{b}_j$ entonces,

1. $\forall j < i$, $\langle \mathbf{b}_j, \mathbf{b}^*_j \rangle = \langle \mathbf{b}^*_j, \mathbf{b}^*_j \rangle$ y $\langle \mathbf{b}_j, \mathbf{b}^*_i \rangle = 0$

   Ambas fueron demostradas en el Capítulo 2 cuando tratamos el proceso de Gram-Schmidt.

2. Definiendo $\mu'_{k,j} = \frac{\langle \mathbf{b}'_k, \mathbf{b}^*_j \rangle}{\langle \mathbf{b}^*_j, \mathbf{b}^*_j \rangle}$ tenemos $|\mu'_{k,j}| \leq \frac{1}{2}$ para $1 \leq j < k$.
   Desarrollando con la definición de $\mathbf{b}'_k$,

$$\mu'_{k,j} = \frac{\langle \mathbf{b}'_k, \mathbf{b}^*_j \rangle}{\langle \mathbf{b}^*_j, \mathbf{b}^*_j \rangle} = \frac{\langle \mathbf{b}_k - \lfloor \mu_{k,j} \rceil \mathbf{b}_j, \mathbf{b}^*_j \rangle}{\langle \mathbf{b}^*_j, \mathbf{b}^*_j \rangle} = \frac{\langle \mathbf{b}_k, \mathbf{b}^*_j \rangle}{\langle \mathbf{b}^*_j, \mathbf{b}^*_j \rangle} - \lfloor \mu_{k,j} \rceil \frac{\langle \mathbf{b}_j, \mathbf{b}^*_j \rangle}{\langle \mathbf{b}_j, \mathbf{b}^*_j \rangle}$$

Donde en la última igualdad usamos el punto 1.

Luego, por definición tenemos

$$\mu'_{k,j} = \mu_{k,j} - \lfloor \mu_{k,j} \rceil$$

Que cumple por definición de $\lfloor \rceil$,

$$|\mu'_{k,j}| = |\mu_{k,j} - \lfloor \mu_{k,j} \rceil| \leq \tfrac{1}{2}$$

3. $j < i < k$, $\mu'_{k,i} = \mu_{k,i}$

   Usando el mismo enfoque que en 2,

$$\mu'_{k,i} = \frac{\langle \mathbf{b}'_k, \mathbf{b}^*_i \rangle}{\langle \mathbf{b}^*_i, \mathbf{b}^*_i \rangle} = \frac{\langle \mathbf{b}_k - \lfloor \mu_{k,j} \rceil \mathbf{b}_j, \mathbf{b}^*_i \rangle}{\langle \mathbf{b}^*_i, \mathbf{b}^*_i \rangle} = \frac{\langle \mathbf{b}_k, \mathbf{b}^*_i \rangle}{\langle \mathbf{b}^*_i, \mathbf{b}^*_i \rangle} - \lfloor \mu_{k,j} \rceil \frac{\langle \mathbf{b}_j, \mathbf{b}^*_i \rangle}{\langle \mathbf{b}^*_i, \mathbf{b}^*_i \rangle} = \frac{\langle \mathbf{b}_k, \mathbf{b}^*_i \rangle}{\langle \mathbf{b}^*_i, \mathbf{b}^*_i \rangle} = \mu_{k,i}$$

   ya que por el punto 1, $\langle \mathbf{b}_j, \mathbf{b}^*_i \rangle = 0$.

□

Previo a la demostración del Teorema 12, veamos una propiedad de la reducción de tamaño.

Proposición 13. *Vectores de Gram-Schmidt son invariantes a la reducción de tamaño.*
La reducción de tamaño (líneas 6 a 10 del Algoritmo 3) no altera los vectores de la base de Gram-Schmidt.

Demostración. Para probar este hecho notamos que podemos codificar el paso $\mathbf{b}_k \leftarrow \lfloor \mu_{k,j} \rceil \mathbf{b}_j$ en forma matricial como $B \leftarrow B \cdot W$, donde W es una matriz diagonal superior con a lo sumo una entrada no nula fuera de su diagonal con valor $\mu_{k,j}$ en la posición (j, k).

Como ejemplo veamos el caso para $n = 3$ correspondiente a $\mathbf{b}_3 \leftarrow \mathbf{b}_3 - \lfloor \mu_{3,1} \rceil \mathbf{b}_1$,

$$\begin{pmatrix} b_{11} & b_{21} & b_{31} - \lfloor \mu_{3,1} \rceil b_{11} \\ b_{12} & b_{22} & b_{32} - \lfloor \mu_{3,1} \rceil b_{12} \\ b_{13} & b_{23} & b_{33} - \lfloor \mu_{3,1} \rceil b_{13} \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & -\lfloor \mu_{3,1} \rceil \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Entonces, describiendo el proceso de Gram-Schmidt en forma matricial como vimos en el capítulo 3 tenemos que $B = B^* \cdot U$ con U matriz triangular superior conteniendo los $\mu_{i,j}$ en las entradas arriba de la diagonal. Si esta es la descomposición previa a alguna de las iteraciones del algoritmo entonces cada paso de la reducción de tamaño puede escribirse como $B \leftarrow B \cdot W = (B^*U) \cdot W = B^* \cdot (UW)$. Como UW es producto de matrices triangulares superiores, esta también lo es y por tanto concluimos que los vectores de Gram-Schmidt serán los mismos luego de cada paso del ciclo. □

Continuamos con la demostración del teorema principal (Teorema 12).

Demostración. Por hipótesis $\mathcal{L} \subset \mathbb{Z}^n$ y por la proposición anterior, sabemos que si el algoritmo termina entonces su salida será una base LLL-reducida.

Resta ver que en efecto, la ejecución siempre termina. Esto no es evidente, ya que si bien en el paso 13 el valor de k se incrementa, en el paso 19 el mismo disminuye, por lo que podría pasar que nunca se cumpla $k > n$.

Probaremos ahora que el paso 19 solo se ejecuta una cantidad finita de veces. Como en cada iteración del ciclo del paso 4, se ejecuta o bien la línea 13 o la 18, esto asegura que eventualmente el valor de k superará el de n y el algoritmo terminará su ejecución.

Sea $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ base de $\mathcal{L}$ recibida como entrada y $B^* = \{\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*\}$ la base de Gram-Schmidt correspondiente. Para cada $l \in \{1, 2, \ldots, n\}$ definimos L_l como el reticulado generado por $\{\mathbf{b}_1, \ldots, \mathbf{b}_l\} \subset \mathbb{Z}^n$ y definimos las siguientes cantidades asociadas,

$$d_l = \prod_{i=1}^{l} \|\mathbf{b}_i^*\|^2$$

$$\mathcal{D}(B) = \prod_{l=1}^{n-1} d_l = \prod_{l=1}^{n-1} \prod_{i=1}^{l} \|\mathbf{b}_i^*\|^2 = \prod_{l=1}^{n-1} \|\mathbf{b}_1^*\|^2 \|\mathbf{b}_2^*\|^2 \cdots \|\mathbf{b}_l^*\|^2 = \prod_{l=1}^{n-1} \|\mathbf{b}_l^*\|^{2(n-l)}$$

Esto surge de que para $l = 1$ solo multiplicamos $\|\mathbf{b}_1^*\|^2$, para $l = 2$ $\|\mathbf{b}_1^*\|^2$ y $\|\mathbf{b}_2^*\|^2$ y en general para $l = i$ los primeros i, $\|\mathbf{b}_i^*\|^2$. Por tanto, sumando los exponentes para los distintos $\|\mathbf{b}_i^*\|$ obtenemos el resultado.

Utilizando la Proposición 9 del Capítulo 3 tenemos que $\det(L_l)^2 = \prod_{i=1}^{l} \|\mathbf{b}_i^*\|^2 = d_l$. Notamos que los reticulados L_l no son de rango máximo y por tanto debemos usar este método para el cálculo del determinante. Observamos también que no incluimos $d_n = \det(\mathcal{L})^2$ por ser este invariante en toda la ejecución, como vimos en el Capítulo 3.

Entonces, tenemos que $\mathcal{D}(B)$ es una función no negativa de los elementos de la base B [1]. Para establecer que el algoritmo termina, es decir que la cantidad de iteraciones es finita, debemos verificar tres condiciones:

1. En cada iteración el potencial asociado a B, $\mathcal{D}(B)$, disminuye en un factor constante $\delta < 1$. Es decir, que si $\mathcal{D}$ y $\mathcal{D}'$ son los valores antes y después de una iteración, debe cumplirse $\mathcal{D}' \leq \delta\mathcal{D} \Leftrightarrow \frac{\mathcal{D}'}{\mathcal{D}} \leq \delta$.

2. Acotar superiormente el valor inicial de la función de potencial, $\mathcal{D}_{inicial}$.

3. Acotar inferiormente el valor final de la función de potencial, $\mathcal{D}_{fin}$.

Notamos que el punto 1 depende exclusivamente del algoritmo, sin embargo los puntos 2 y 3 refieren a cotas sobre los vectores del reticulado. Estas pueden no ser únicas y su variación puede conducir a resultados más o menos ajustados.

Comencemos con la cota superior sobre el valor inicial. Sea $\mathcal{D}_{inicial}$ el valor inicial de $\mathcal{D}(B)$. Entonces por el proceso de Gram-Schmidt tenemos $\|\mathbf{b}_i^*\| \leq \|\mathbf{b}_i\|$ para todo valor de i,

$$\mathcal{D}_{inicial} \leq \prod_{l=1}^{n} \|\mathbf{b}_l^*\|^{2(n+1-l)} \leq (\max_{1\leq l\leq n} \|\mathbf{b}_l\|)^{2(1+2+\cdots+n)} = (\max_{1\leq l\leq n} \|\mathbf{b}_l\|)^{n(n+1)}$$

Donde la segunda desigualdad surge del hecho que el paso $l = 1$ aporta un factor de $2 \cdot n$ al producto, el $l = 2$ aporta $2 \cdot (n-1)$ y en general el paso i-ésimo aporta $2 \cdot (n - i + 1)$ factores, así hasta el paso n, que contribuye $2 \cdot 1$. Notar además que $(\max_{1\leq l\leq n} \|\mathbf{b}_l\|)^{n(n+1)}$ es una constante que solo depende de la dimensión del reticulado y no de la cantidad de iteraciones.

[1] Se puede referir a este tipo de argumentos como *argumentos de potencial* [13], donde la función no negativa considerada es llamada potencial

Recordando que $d_l = \det(L_l)^2 = |\det(B_l^t B_l)|$ con $B_l = [\mathbf{b}_1, \cdots, \mathbf{b}_l]$ y que los vectores de la base tienen coordenadas enteras, tenemos que $d_l \geq 1$ y es inicialmente un entero positivo. Por tanto tenemos también que $\mathcal{D} \geq 1$ entero positivo, para todo estado de B durante el algoritmo. Observar que el único caso donde $\mathcal{D} = 1$ es en el que todos los vectores de Gram-Schmidt tienen norma 1.[1]Es decir, esta cota inferior en general no es ajustada.

Como $\mathcal{D}$ es función de los vectores de Gram-Schmidt de la base y como probamos (Proposición 13), estos no cambian durante la reducción de tamaño, durante la ejecución de LLL (Algoritmo 3). Dicho valor solo puede cambiar en el paso de intercambio (líneas 15 a 19, Algoritmo 3). En caso de que un cambio se produzca (línea 16, Algoritmo 3), este afectará únicamente el valor de d_l para $l = k - 1$, ya que si $l < k - 1$ entonces d_l no involucra ni a $\mathbf{b}_{k-1}^*$ ni a $\mathbf{b}_k^*$ y si $l \geq k$ entonces el producto que define a d_l incluye tanto a $\mathbf{b}_{k-1}^*$ como también a $\mathbf{b}_k^*$.

Siguiendo la idea de [10], estimamos el cambio en d_{k-1} notando que cuando se da un intercambio, la condición de Lovász no se cumple en la línea 12, entonces resulta,

$$\|\mathbf{b}_k^*\|^2 < (\delta - \mu_{k,k-1}^2)\left\|\mathbf{b}_{k-1}^*\right\|^2 \leq \delta\left\|\mathbf{b}_{k-1}^*\right\|^2 \Leftrightarrow \frac{\left\|\mathbf{b}_k^*\right\|^2}{\left\|\mathbf{b}_{k-1}^*\right\|^2} \leq \delta$$

Entonces al intercambiar $\mathbf{b}_{k-1}^*$ con $\mathbf{b}_k^*$ en una iteración obtenemos,

$$\begin{aligned} d_{k-1}^{nvo} &= \|\mathbf{b}_1^*\|^2\|\mathbf{b}_2^*\|^2 \cdots \left\|\mathbf{b}_{k-2}^*\right\|^2\|\mathbf{b}_k^*\|^2 \\ &= \|\mathbf{b}_1^*\|^2\|\mathbf{b}_2^*\|^2 \cdots \left\|\mathbf{b}_{k-2}^*\right\|^2\left\|\mathbf{b}_{k-1}^*\right\|^2 \frac{\left\|\mathbf{b}_k^*\right\|^2}{\left\|\mathbf{b}_{k-1}^*\right\|^2} \\ &= d_{k-1}^{ant} \frac{\left\|\mathbf{b}_k^*\right\|^2}{\left\|\mathbf{b}_{k-1}^*\right\|^2} \leq d_{k-1}^{ant} \cdot \delta \end{aligned}$$

donde d_{k-1}^{ant} y d_{k-1}^{nvo} son los valores de d_{k-1} antes y después de realizar el intercambio respectivamente.

Si el paso de intercambio se ejecuta m veces, el valor de $\mathcal{D}$ se reduce por un factor de al menos δ^m (notar que $\delta < 1$), ya que en cada intercambio se reduce el valor de alguno de los d_l por al menos un factor de δ y $\mathcal{D}$ es el producto de

[1]Recordar que los vectores iniciales tienen coeficientes enteros y que luego pueden tener coeficientes racionales

los d_l.

Notando $\mathcal{D}_{fin}$ al valor de $\mathcal{D}$ una vez que el algoritmo finaliza su ejecución y nuevamente m la cantidad de veces que se ejecuta el paso de intercambio, tenemos [1]

$$1 \leq \mathcal{D}_{fin} \leq \delta^m \mathcal{D}_{inicial} \tag{4.1}$$

y por tanto el Algoritmo 3 termina, ya que de lo contrario como $\lim_{m \to +\infty} \delta^m \mathcal{D}_{inicial} = 0$, no se cumpliría 4.1.

Por otro lado,

$$\tfrac{1}{\delta^m} = (\tfrac{1}{\delta})^m \leq \mathcal{D}_{inicial}$$

tomando logaritmos en la expresión y observando que como $\delta < 1$ entonces $\frac{1}{\delta} > 1$ resulta,

$$\log\big((\tfrac{1}{\delta})^m\big) \leq \log(\mathcal{D}_{inicial}) \Leftrightarrow m \cdot \log\big(\tfrac{1}{\delta}\big) \leq \log(\mathcal{D}_{inicial}) \Leftrightarrow m \leq \tfrac{\log(\mathcal{D}_{inicial})}{\log\left(\frac{1}{\delta}\right)}$$

Luego, $m \in O(\log(D_{inicial}))$.

Usando la cota superior, $\mathcal{D}_{inicial} \leq (\max_{1 \leq l \leq n} \|\mathbf{b}_l\|)^{n(n+1)}$ y notando $k = \max_{1 \leq l \leq n} \|\mathbf{b}_l\|$ tenemos,

$$\log(D_{inicial}) \leq \log\left(k^{n^2+n}\right) = (n^2+n)\log(k) \leq 2n^2 \log(k)$$

Entonces, $\log(D_{inicial}) \in O(n^2 \log(k))$ como queríamos probar.

□

Para ver que cada iteración puede implementarse en tiempo polinomial y que el tamaño de los valores computados también lo es consultar [16].

[1] Observamos que el **mientras** de la línea 4 se ejecuta a lo sumo $2m+n$ veces, ya que por cada paso de intercambio el ciclo debe ejecutarse nuevamente y este debe además realizarse n veces al menos, para que el valor de k pueda superar n.

4.3.2. Algoritmos de Babai

Presentamos a continuación dos algoritmos propuestos por Babai [22] que pueden ser utilizados como herramienta, en conjunto con el algoritmo LLL, para resolver versiones aproximadas de CVP. Los factores de aproximación logrados con estos métodos son exponenciales en la dimensión del reticulado (asumiendo reticulados de rango máximo).

Algoritmo del hiperplano más cercano

Comenzamos con las ideas que motivan el algoritmo. Sea $\mathcal{L} \subset \mathbb{R}^n$ de rango máximo, $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ y $B^* = \{\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*\}$ una base de $\mathcal{L}$ y su base de Gram-Schmidt correspondiente. Consideramos un $\mathbf{w} \in \mathbb{R}^n$ arbitrario.

Sea $H = \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1})$ y consideramos el reticulado generado por $\{\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}\}$, $\mathcal{L}' = \mathcal{L} \cap H$. Nuestro objetivo será encontrar un vector $\mathbf{y} \in \mathcal{L}$ tal que $\text{dist}(\mathbf{w}, H + \mathbf{y})$ sea mínima. Observar que $H + \mathbf{y}$ es un hiperplano[1] trasladado según $\mathbf{y}$. Luego se define un nuevo vector $\mathbf{w}'$ como la proyección ortogonal de $\mathbf{w}$ sobre $H + \mathbf{y}$, es decir que $\mathbf{w}' \in H + \mathbf{y}$ y $\mathbf{w} - \mathbf{w}' \in H^\perp$, donde $H^\perp$ denota el complemento ortogonal de H.

Ahora, tomando $\mathbf{w}'' = \mathbf{w}' - \mathbf{y} \in H$, notamos que si $\mathbf{w} \notin \mathcal{L}$, entonces $\mathbf{w}''$ tampoco. Para ver esto último, como $H = \{\mathbf{v} \in \mathbb{R}^n : \mathbf{v} = \alpha_1\mathbf{b}_1 + \cdots + \alpha_{n-1}\mathbf{b}_{n-1}, \alpha_i \in \mathbb{R}\}$ y para un $\mathbf{y} \in \mathcal{L}$ fijo $H + \mathbf{y} = \{\mathbf{v} + \mathbf{y} : \mathbf{v} \in H\}$, entonces como $\mathbf{w}' \in H + \mathbf{y}$ tenemos

$$\mathbf{w}' = \mathbf{v} + \mathbf{y} = (\alpha_1\mathbf{b}_1 + \cdots + \alpha_{n-1}\mathbf{b}_{n-1}) + (a_1\mathbf{b}_1 + \cdots + a_n\mathbf{b}_n) = (\alpha_1 + a_1)\mathbf{b}_1 + \cdots + (\alpha_{n-1} + a_{n-1})\mathbf{b}_{n-1} + a_n\mathbf{b}_n$$
$$\text{donde } \alpha_i \in \mathbb{R} \text{ y } a_i \in \mathbb{Z}$$

ahora como $\mathbf{w} \notin \mathcal{L}$, existe al menos un $\alpha_i \in \mathbb{R} \setminus \mathbb{Z}$, que implica que también existe $(\alpha_i + a_i) \in \mathbb{R} \setminus \mathbb{Z}$ y por tanto $\mathbf{w}' \notin \mathcal{L}$. Como el reticulado es subgrupo aditivo $\mathbf{w}' - \mathbf{y} = \mathbf{w}''$ tampoco formará parte del reticulado en estas condiciones.

Entonces, la idea del algoritmo es resolver el subproblema CVP asociado al caso de dimensión más baja usando $\mathbf{w}''$ como vector objetivo en $\mathcal{L}'$, obteniendo como resultado un $\mathbf{y}' \in \mathcal{L}'$ y dando como salida $\mathbf{v} = \mathbf{y} + \mathbf{y}'$ para el problema

[1] Subespacio cuya dimensión es una menos que la del espacio que lo incluye.

CVP original. Notar que como $\mathbf{y}' \in \mathcal{L}' = \mathcal{L} \cap H \Rightarrow \mathbf{y}' \in \mathcal{L}$ de modo que $\mathbf{y} + \mathbf{y}' = \mathbf{v} \in \mathcal{L}$. Queda asegurar que este vector se encuentra "relativamente cerca" del vector objetivo y especificar cómo obtener los vectores $\mathbf{y}$ y $\mathbf{w}'$.

Proposición 14. *Sea* $\boldsymbol{w} = \sum_{j=1}^{n} l_j \boldsymbol{b}_j^*$ *con* $l_i \in \mathbb{R}$, $\boldsymbol{y} = \lfloor l_n \rceil \boldsymbol{b}_n \in \mathcal{L}$ *y* $\boldsymbol{w'} = \sum_{j=1}^{n-1} l_j \boldsymbol{b}_j^* + \lfloor l_n \rceil \boldsymbol{b}_n^*$. *Entonces* $\boldsymbol{y}$ *es tal que minimiza* $\operatorname{dist}(H + \boldsymbol{y}, \boldsymbol{w})$. *Además* $\boldsymbol{w'}$ *es la proyección ortogonal de* $\boldsymbol{w}$ *sobre* $H + \boldsymbol{y}$.

Demostración. Por definición tenemos

$$\operatorname{dist}(H + \mathbf{y}, \mathbf{w}) = \min_{k \in H + \mathbf{y}} \|\mathbf{w} - \mathbf{k}\| = \min_{\mathbf{h} \in H} \|\mathbf{w} - (\mathbf{h} + \mathbf{y})\|$$

Sea un $\mathbf{y} \in \mathcal{L}$ arbitrario. Tenemos que $\mathbf{y} = \sum_{j=1}^{n} l'_j \mathbf{b}_j$ con $l'_j \in \mathbb{Z}$. Ahora como $\operatorname{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}, \mathbf{b}_n) = \operatorname{span}(\mathbf{b}_1^*, \ldots, \mathbf{b}_{n-1}^*, \mathbf{b}_n^*)$ podemos escribir $\mathbf{y} = \sum_{j=1}^{n-1} l''_j \mathbf{b}_j^* + l'_n \mathbf{b}_n$ y $\mathbf{b}_n = \mathbf{b}_n^* + \sum_{j=1}^{n-1} \mu_{n,j} \mathbf{b}_j^*$, tenemos entonces

$$\begin{aligned} \mathbf{y} &= \sum_{j=1}^{n-1} l''_j \mathbf{b}_j^* + l'_n (\mathbf{b}_n^* + \sum_{j=1}^{n-1} \mu_{n,j} \mathbf{b}_j^*) = \sum_{j=1}^{n-1} l''_j \mathbf{b}_j^* + l'_n \mathbf{b}_n^* + \sum_{j=1}^{n-1} l'_n \mu_{n,j} \mathbf{b}_j^* \\ &= \sum_{j=1}^{n-1} (l''_j + l'_n \mu_{n,j}) \mathbf{b}_j^* + l'_n \mathbf{b}_n^* = \sum_{j=1}^{n-1} \hat{l}_j \mathbf{b}_j^* + l'_n \mathbf{b}_n^* \end{aligned}$$

donde $\hat{l}_j \in \mathbb{R}$ y $l'_n \in \mathbb{Z}$.

Entonces,

$$\mathbf{h}_{min} = \arg\min_{\mathbf{h} \in H} \|\mathbf{w} - (\mathbf{h} + \mathbf{y})\|^2 = \sum_{j=1}^{n-1} (l_j - \hat{l}_j) \mathbf{b}_j^*$$

De modo que utilizando la propiedad vista en el Capítulo 2 para evaluar la norma de un vector usando una base ortogonal,

$$\operatorname{dist}(\mathbf{w}, H + \mathbf{y})^2 = \|\mathbf{w} - (\mathbf{h}_{min} + \mathbf{y})\|^2 = \left\| (l_n - \hat{l}_n) \mathbf{b}_n^* \right\|^2 = (l_n - \hat{l}_n)^2 \|\mathbf{b}_n^*\|^2$$

en esta expresión el componente variable es $\hat{l}_n$ que forma parte de $\mathbf{h}_{min}$ y debemos tomar $\hat{l}_n = \lfloor l_n \rceil$ para minimizar dicha distancia. Observar que podemos tomar cualquier $\mathbf{t} \in \mathcal{L}'$ y considerar $\mathbf{y} = \lfloor l_n \rceil + \mathbf{t}$ obteniendo el mismo resultado, pudiendo esta adición ser removida con la elección de $\mathbf{h}$.

Además, por definición de $\mathbf{w}'$ e $\mathbf{y}$ se cumple,

$$\mathbf{w}' - \mathbf{y} = \sum_{j=1}^{n-1} l_j \mathbf{b}_j^* + \lfloor l_n \rceil \mathbf{b}_n^* - \lfloor l_n \rceil \mathbf{b}_n = \sum_{j=1}^{n-1} l_j \mathbf{b}_j^* + \lfloor l_n \rceil (\mathbf{b}_n^* - \mathbf{b}_n)$$

y como $\mathbf{b}_n^* - \mathbf{b}_n = \mathbf{b}_n - \sum_{j=1}^{n-1} \mu_{n,j} \mathbf{b}_j^* - \mathbf{b}_n = -\sum_{j=1}^{n-1} \mu_{n,j} \mathbf{b}_j^* \in H$

$$\mathbf{w}' - \mathbf{y} = \sum_{j=1}^{n-1} (l_j - \mu_{n,j}) \mathbf{b}_j^* \in H$$

Por tanto, $\mathbf{w}' \in H + \mathbf{y}$.

También se cumple,

$$\mathbf{w} - \mathbf{w}' = \sum_{j=1}^{n-1} l_j \mathbf{b}_j^* - \sum_{j=1}^{n-1} l_j \mathbf{b}_j^* - \lfloor l_n \rceil \mathbf{b}_n^* = (l_n - \lfloor l_n \rceil) \mathbf{b}_n^*.$$

Dado que $(l_n - \lfloor l_n \rceil) \leq \frac{1}{2}$ y por definición $\mathbf{b}_n^*$ es ortogonal a $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_{n-1}^*\}$ y por tanto también a $H = \text{span}(\mathbf{b}_1^*, \ldots, \mathbf{b}_{n-1}^*)$, tenemos que $\mathbf{w}'$ es por definición la proyección ortogonal de $\mathbf{w}$ sobre $H + \mathbf{y}$. □

A continuación presentamos el algoritmo [17], donde se nota en cada paso $\mathbf{y}_n = \mathbf{y}$, $\mathbf{w}_n = \mathbf{w}$ y $\mathbf{w}_{n-1} = \mathbf{w}''$.

Algoritmo 4 Algoritmo del hiperplano más cercano de Babai

Entrada: $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ base del reticulado $\mathcal{L}$, $\mathbf{w} \in \mathbb{R}^n$
Salida: $\mathbf{v} \in \mathcal{L}$

1: Computar la base de Gram-Schmidt $B^* = \{\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*\}$ de B
2: Asignar $\mathbf{w}_n \leftarrow w$
3: **para** $i = n$ hasta 1 **hacer**
4: $\quad l_i \leftarrow \frac{\langle \mathbf{w}_i, \mathbf{b}_i^* \rangle}{\langle \mathbf{b}_i^*, \mathbf{b}_i^* \rangle}$
5: $\quad \mathbf{y}_i \leftarrow \lfloor l_i \rceil \mathbf{b}_i$
6: $\quad \mathbf{w}_{i-1} \leftarrow \mathbf{w}_i - (l_i - \lfloor l_i \rceil) \mathbf{b}_i^* - \lfloor l_i \rceil \mathbf{b}_i$
7: **fin para**
8: **devolver** $\mathbf{v} = \mathbf{y}_1 + \cdots + \mathbf{y}_n$

Observar que de la demostración anterior (Proposición 14) tenemos que $\mathbf{w} - \mathbf{w}' = (l_n - \lfloor l_n \rceil) \mathbf{b}_n^* \Leftrightarrow \mathbf{w}' = \mathbf{w} - (l_n - \lfloor l_n \rceil) \mathbf{b}_n^*$ y como $\mathbf{w}'' = \mathbf{w}' - \mathbf{y} = \mathbf{w}' - \lfloor l_n \rceil \mathbf{b}_n = \mathbf{w} - (l_n - \lfloor l_n \rceil) \mathbf{b}_n^* - \lfloor l_n \rceil \mathbf{b}_n$ obtenemos que efectivamente $\mathbf{w}''$ se corresponde con $\mathbf{w}_{n-1}$ en el algoritmo.

Seguimos con un resultado que será utilizado en la demostración de correctitud del algoritmo.

Proposición 15. *Sea* $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ *una base LLL-reducida con factor* $\delta = \frac{3}{4}$ *de* $\mathcal{L}$ *de rango máximo. Si* $\boldsymbol{v}$ *es la salida del algoritmo del hiperplano más cercano de Babai (Algoritmo 4), dado* $\boldsymbol{w}$ *como entrada entonces se cumple*

$$\operatorname{dist}(\boldsymbol{w}, \boldsymbol{v})^2 = \|\boldsymbol{w} - \boldsymbol{v}\|^2 \leq \tfrac{1}{4}(2^n - 1)\|\boldsymbol{b}_n^*\|^2$$

donde $\boldsymbol{b}_n^*$ *es el* n*-ésimo vector de la base de Gram-Schmidt correspondiente.*

Demostración. Hacemos la prueba por inducción en la dimensión del reticulado.

Paso Base) Si $n = 1$, $B = \{b_1\} \subset \mathbb{R}$, el algoritmo devuelve $v = y = \lfloor l_1 \rceil b_1$ con $l_1 = \frac{w \cdot b_1^*}{b_1^* \cdot b_1^*} = \frac{w \cdot b_1}{b_1 \cdot b_1} = \frac{w}{b_1}$. Además podemos escribir $w = \frac{w}{b_1} b_1$ y como por definición $\lfloor l_1 \rceil$ es el entero más cercano a l_1 se tiene,

$$\operatorname{dist}(w, v)^2 = \operatorname{dist}(w, y)^2 = \left|\tfrac{w}{b_1} b_1 - \lfloor \tfrac{w}{b_1} \rceil b_1\right|^2 = \left|(\tfrac{w}{b_1} - \lfloor \tfrac{w}{b_1} \rceil) b_1\right|^2 \leq \left|\tfrac{1}{2} b_1\right|^2 = \tfrac{1}{4}|b_1|^2 = \tfrac{1}{4}|b_1^*|^2$$

Paso Inductivo) Ahora suponemos que la proposición se cumple en dimensión $k \geq 1$ y probamos el resultado para $k + 1$.

Como salida del algoritmo obtenemos, $\mathbf{v} = \mathbf{y} + \mathbf{y}'$ con $\mathbf{y} \in \mathcal{L}$ tal que minimiza $\operatorname{dist}(\mathbf{w}, H + \mathbf{y})$, $\mathbf{w}'$ es la proyección ortogonal de $\mathbf{w}$ sobre $H + \mathbf{y}$ e $\mathbf{y}'$ es la salida del algoritmo para la entrada $\mathbf{w}'' = \mathbf{w}' - \mathbf{y} \in \mathcal{L}'$.

Por hipótesis inductiva tenemos,

$$\operatorname{dist}(\mathbf{w}'', \mathbf{y}')^2 = \|\mathbf{w}'' - \mathbf{y}'\|^2 \leq \tfrac{1}{4}(2^{n-1} - 1)\left\|\mathbf{b}_{n-1}^*\right\|^2$$

Por tanto,

$$\operatorname{dist}(\mathbf{w}, \mathbf{y} + \mathbf{y}')^2 = \|\mathbf{w} - (\mathbf{y} + \mathbf{y}')\|^2 = \|\mathbf{w} - \mathbf{w}' + \mathbf{w}' - (\mathbf{y} + \mathbf{y}')\|^2 = \|\mathbf{w} - \mathbf{w}' + (\mathbf{w}' - \mathbf{y}) - \mathbf{y}'\|^2 = \|(\mathbf{w} - \mathbf{w}') + (\mathbf{w}'' - \mathbf{y}')\|^2$$

Ahora por la Proposición 14, sabemos que $\mathbf{w} - \mathbf{w}'$ es ortogonal a H y como $\mathbf{w}'' \in H$ que implica $\mathbf{w}'' - \mathbf{y}' \in H - \mathbf{y}'$. Entonces $(\mathbf{w} - \mathbf{w}')$ es ortogonal a $\mathbf{w}'' - \mathbf{y}'$ y entonces,

$$\begin{aligned}\operatorname{dist}(\mathbf{w}, \mathbf{y}+\mathbf{y}')^2 = \|(\mathbf{w}-\mathbf{w}')+(\mathbf{w}''-\mathbf{y}')\|^2 = \|\mathbf{w}-\mathbf{w}'\|^2 + \|\mathbf{w}''-\mathbf{y}'\|^2 \\ \leq \left\|\frac{1}{2}\mathbf{b}_n^*\right\|^2 + \frac{1}{4}(2^{n-1}-1)\left\|\mathbf{b}_{n-1}^*\right\|^2 \end{aligned} \tag{4.2}$$

donde en la última desigualdad usamos la última cuenta realizada en la demostración de la Proposición 14 y la hipótesis inductiva.

Como B es LLL-reducida, tenemos por el Teorema 11 que $\left\|\mathbf{b}_j^*\right\|^2 \leq 2^{i-j}\|\mathbf{b}_i^*\|^2$ para todo $j < i$, en particular $\left\|\mathbf{b}_{n-1}^*\right\|^2 \leq 2\|\mathbf{b}_n^*\|^2$, aplicando este resultado en la ecuación 4.2,

$$\operatorname{dist}(\mathbf{w}, \mathbf{y}+\mathbf{y}')^2 \leq \tfrac{1}{4}\|\mathbf{b}_n^*\|^2 + \tfrac{1}{4}(2^{n-1}-1)\left\|\mathbf{b}_{n-1}^*\right\|^2 \leq$$
$$\tfrac{1}{4}\|\mathbf{b}_n^*\|^2 + \tfrac{1}{4}(2^{n-1}-1)2\|\mathbf{b}_n^*\|^2 = \tfrac{1}{4}\|\mathbf{b}_n^*\|^2(1+(2^{n-1}-1)2) = \tfrac{1}{4}(2^n-1)\|\mathbf{b}_n^*\|^2$$

□

Finalizamos con la prueba de correctitud del Algoritmo 4.

Teorema 13. *Correctitud algoritmo del hiperplano más cercano de Babai*
Sea $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ una base LLL-reducida con factor $\delta = \frac{3}{4}$ de un reticulado $\mathcal{L}$. Entonces la salida del algoritmo del hiperplano más cercano de Babai (Algoritmo 4), tomando $\boldsymbol{w} \in \mathbb{R}^n$ como entrada, es un vector $\boldsymbol{v}$ tal que

$$\operatorname{dist}(\boldsymbol{v}, \boldsymbol{w}) = \|\boldsymbol{v}-\boldsymbol{w}\| < 2^{\frac{n}{2}}\|\boldsymbol{u}-\boldsymbol{w}\| = 2^{\frac{n}{2}}\operatorname{dist}(\boldsymbol{u}, \boldsymbol{w}) \text{ para todo } \boldsymbol{u} \in \mathcal{L}$$

Demostración. La demostración se hace por inducción en n.

Paso Base) Para $n = 1$, vimos en la demostración de la Proposición 15 que $v = y_1 = \lfloor \frac{w}{b_1} \rceil b_1$ es efectivamente el punto del reticulado más cercano a $w = \frac{w}{b_1}b_1$, es decir es la solución exacta de CVP para w.

Paso Inductivo) Asumimos que el resultado se cumple para $k \geq 1$ y lo probamos para $k+1$. Sea $\mathbf{c} \in \mathcal{L}$ un vector más cercano a $\mathbf{w}$. Sea $\mathbf{y}$ el vector elegido en la primera iteración del algoritmo, existen dos casos posibles:

- Caso $\mathbf{c} \in H + \mathbf{y}$. Por el teorema de Pitágoras, $\|\mathbf{c}-\mathbf{w}\|^2 = \|\mathbf{c}-\mathbf{w}'\|^2 + \|\mathbf{w}'-\mathbf{w}\|^2$ y como por ser $\mathbf{w}'$ la proyección ortogonal de $\mathbf{w}$ sobre $H+\mathbf{y}$ la distancia de $\mathbf{w}$ a $\mathbf{w}'$ es la mínima con respecto a $H+\mathbf{y}$, tenemos que

$\mathbf{c}$ también debe ser un vector más cercano para $\mathbf{w}'$. Entonces, $\mathbf{c}-\mathbf{y}$ es un vector más cercano para $\mathbf{w}'-\mathbf{y}=\mathbf{w}''\in H$.

Sea entonces $\mathbf{y}'$ la salida del algoritmo para $\mathbf{w}''$, por hipótesis inductiva tenemos

$$\begin{aligned}\operatorname{dist}(\mathbf{y}',\mathbf{w}'') = \|\mathbf{y}'-\mathbf{w}''\| &< 2^{\frac{n-1}{2}}\|\mathbf{c}-\mathbf{y}-\mathbf{w}''\| \\ &= 2^{\frac{n-1}{2}}\operatorname{dist}(\mathbf{c}-\mathbf{y},\mathbf{w}'')\end{aligned} \tag{4.3}$$

Usando que $\mathbf{w}''=\mathbf{w}'-\mathbf{y}$ en 4.3,

$$\operatorname{dist}(\mathbf{y}',\mathbf{w}'') = \|\mathbf{y}+\mathbf{y}'-w'\| < 2^{\frac{n-1}{2}}\|c-w'\| \tag{4.4}$$

Entonces,

$$\begin{aligned}\operatorname{dist}(\mathbf{w},\mathbf{v})^2 = \|\mathbf{v}-\mathbf{w}\|^2 = \|\mathbf{y}+\mathbf{y}'-\mathbf{w}+\mathbf{w}'-\mathbf{w}'\|^2 = \\ = \|\mathbf{y}+\mathbf{y}'-\mathbf{w}'\|^2 + \|\mathbf{w}'-\mathbf{w}\|^2 \\ < (2^{\frac{n-1}{2}})^2\|\mathbf{u}-\mathbf{w}'\|^2 + \|\mathbf{w}'-\mathbf{w}\|^2 = 2^{n-1}\|\mathbf{c}-\mathbf{w}'\|^2 + \|\mathbf{w}'-\mathbf{w}\|^2\end{aligned} \tag{4.5}$$

donde la tercera igualdad en 4.5 se da porque $\mathbf{y}+\mathbf{y}'-\mathbf{w}'=\mathbf{y}-\mathbf{w}''$ y $\mathbf{w}'-\mathbf{w}$ son ortogonales, dado que $\mathbf{w}'-\mathbf{w}$ ortogonal a $H+\mathbf{y}$ e $\mathbf{y}-\mathbf{w}''\in H+\mathbf{y}$. La desigualdad en 4.5, resulta de aplicar 4.4.

Ahora como $\mathbf{c}\in H+\mathbf{y}$ y $\mathbf{w}'$ es la proyección ortogonal de $\mathbf{w}$ sobre $H+\mathbf{y}$, $\operatorname{dist}(\mathbf{c},\mathbf{w}') = \|\mathbf{c}-\mathbf{w}'\| < \|\mathbf{c}-\mathbf{w}\| = \operatorname{dist}(\mathbf{c},\mathbf{w})$ y $\operatorname{dist}(\mathbf{w}',\mathbf{w}) = \|\mathbf{w}'-\mathbf{w}\| < \|\mathbf{c}-\mathbf{w}\| = \operatorname{dist}(\mathbf{c},\mathbf{w})$ Usando esto en la desigualdad 4.5,

$$\begin{gathered}\operatorname{dist}(\mathbf{w},\mathbf{v})^2 = \|\mathbf{v}-\mathbf{w}\|^2 < 2^{n-1}\|\mathbf{c}-\mathbf{w}\|^2 + \|\mathbf{c}-\mathbf{w}\|^2 = \\ (2^{n-1}+1)\|\mathbf{c}-\mathbf{w}\|^2 \le 2^n\|\mathbf{c}-\mathbf{w}\|^2\end{gathered}$$

Tomando raíz cuadrada, $\operatorname{dist}(\mathbf{w},\mathbf{v}) = \|\mathbf{v}-\mathbf{w}\| < 2^{\frac{n}{2}}\|\mathbf{c}-\mathbf{w}\|$

- Caso $\mathbf{c}\notin H+\mathbf{y}$. Como vimos en la prueba de la Proposición 14, $\operatorname{dist}(\mathbf{w},H+\mathbf{y}) = \operatorname{dist}(\mathbf{w},\mathbf{w}') = \|\mathbf{w}-\mathbf{w}'\| \le \frac{1}{2}\|\mathbf{b}_n^*\|$. Como $\mathbf{c}$ no pertenece al hiperplano más cercano, pero si al reticulado, $\operatorname{dist}(\mathbf{w},\mathbf{c}) \ge \frac{1}{2}\|\mathbf{b}_n^*\|$. Según la Proposición 15, $\sqrt{\frac{4}{2^n-1}}\|\mathbf{w}-\mathbf{v}\| \le \|\mathbf{b}_n^*\|$, entonces

$$\|\mathbf{w}-\mathbf{c}\| \geq \tfrac{1}{2}\|\mathbf{b}_n^*\| \geq \tfrac{1}{2}\sqrt{\tfrac{4}{2^n-1}}\|\mathbf{w}-\mathbf{v}\| = \tfrac{1}{\sqrt{2^n-1}}\|\mathbf{w}-\mathbf{v}\|$$
$$\Leftrightarrow (2^n-1)^{\frac{1}{2}}\|\mathbf{w}-\mathbf{c}\| \geq \|\mathbf{w}-\mathbf{v}\|$$

Finalmente como $2^n - 1 < 2^n$,

$$2^{\frac{n}{2}}\|\mathbf{w}-\mathbf{c}\| > (2^n-1)^{\frac{1}{2}}\|\mathbf{w}-\mathbf{c}\| \geq \|\mathbf{w}-\mathbf{v}\| = \text{dist}(\mathbf{w},\mathbf{v})$$

□

Este algoritmo puede ser implementado en una cantidad polinomial de operaciones en función del número de bits de la entrada [17]. Por tanto, dicho algoritmo es eficiente y resuelve el problema CVP_γ para un factor de aproximación exponencial $\gamma(n) = 2^{\frac{n}{2}}$.

Algoritmo del vértice más cercano

La otra propuesta algorítmica dada por Babai en [22] es el algoritmo del vértice más cercano que presentamos a continuación.

Algoritmo 5 Algoritmo del vértice más cercano de Babai

Entrada: $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ base del reticulado $\mathcal{L}$, $\mathbf{w} \in \mathbb{R}^n$
Salida: $\mathbf{v} \in \mathcal{L}$
1: Computar los $l_i \in \mathbb{R}$ tales que $\mathbf{w} = l_1\mathbf{b}_1 + \cdots + l_n\mathbf{b}_n$
2: **devolver** $\mathbf{v} = \lfloor l_1 \rceil \mathbf{b}_1 + \cdots + \lfloor l_n \rceil \mathbf{b}_n$

Este algoritmo resulta más sencillo, tanto conceptualmente como también en cuanto al cómputo necesario, ya que no requiere calcular la base de Gram-Schmidt. Sin embargo, su análisis es más complejo [17].

Babai probó en [22] que si se utiliza una base LLL-reducida en conjunto con el algoritmo, el vector que se obtiene como salida se encuentra a un factor exponencial de la distancia mínima de $\mathbf{w}$ al reticulado. Formalizamos esto mediante el siguiente teorema,

Teorema 14. *Correctitud algoritmo del vértice más cercano de Babai*
Sea $B = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ una base LLL-reducida con factor $\delta = \frac{3}{4}$ de un reticulado $\mathcal{L}$. Entonces la salida del algoritmo del vértice más cercano de Babai (Algoritmo 5) tomando $\boldsymbol{w} \in \mathbb{R}^n$ como entrada es un vector $\boldsymbol{v}$ tal que

$$\text{dist}(\boldsymbol{v}, \boldsymbol{w}) = \|\boldsymbol{v} - \boldsymbol{w}\| < (1 + 2n(\tfrac{9}{2})^{\frac{n}{2}})\|\boldsymbol{u} - \boldsymbol{w}\| \text{ para todo } \boldsymbol{u} \in \mathcal{L}$$

La demostración puede consultarse en [22] y utiliza resultados geométricos sobre la forma del paralelepípedo fundamental de la base LLL-reducida. Este algoritmo también ejecuta en tiempo polinomial en función del tamaño de la entrada.[22] Notar que para los tiempos de ejecución se asumen entradas racionales.

Conceptualmente, el algoritmo toma el vértice del paralelepípedo fundamental (eventualmente trasladado) más cercano a $\mathbf{w}$. Este enfoque funciona bien si los vectores de la base son relativamente ortogonales entre sí y no produce buenas soluciones si el ángulo entre los mismos es pequeño, que se condice con la noción de "propiedades deseables" de una base que discutimos en el Teorema 11. La siguiente Figura 4.1 ilustra un ejemplo en dimensión 2.

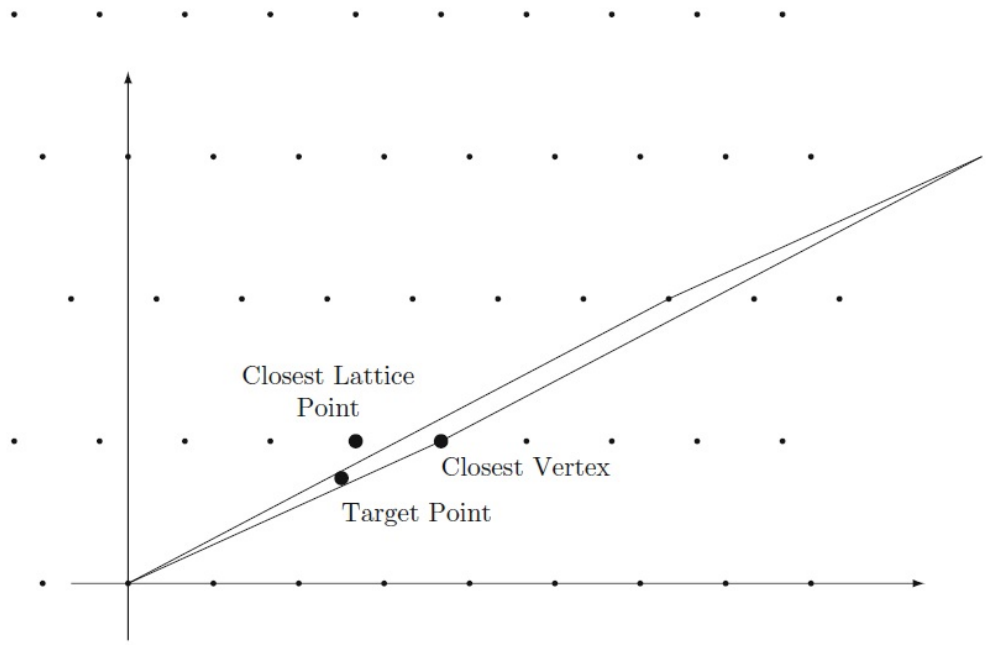


Figura 4.1: Vértice más cercano cuando la base no está cerca de ser ortogonal [10]

Si consideramos los algoritmos junto con una base LLL-reducida, utilizar el algoritmo del hiperplano más cercano tiende a dar mejores resultados que los obtenidos haciendo uso del algoritmo del vértice más cercano [10].

Capítulo 5

Criptosistema Goldreich, Goldwasser y Halevi (GGH)

La forma en la que los reticulados pueden ser utilizados en criptografía fue descubierta por Ajtai en [23], donde propuso una conexión entre la dificultad del problema del vector más corto (SVP) en una clase aleatorizada de reticulados y la dificultad de otros tres problemas para todo reticulado. Es decir, estableció que resolver un problema SVP en el caso promedio en una clase de reticulados determinada en $\mathbb{Z}^n$, equivale a resolver tres problemas en el peor caso en todo reticulado (con probabilidad exponencialmente cercana a 1). En base a la dificultad en el peor caso de uno de estos problemas, propuso junto a Dwork en [24] un sistema de clave pública, que fue el primero con este tipo de garantías de seguridad, si bien el mismo resultaba poco eficiente.

Notamos que la existencia de un algoritmo en el peor caso es más fuerte que la existencia de un algoritmo en el caso promedio y por tanto, brinda más garantías de seguridad a las construcciones.

Luego durante mediados de 1990, varios criptosistemas basados en SVP y/o CVP en reticulados con dimensiones altas fueron propuestos. Dentro de los más relevantes se encuentran el criptosistema GGH creado por Goldreich, Goldwasser y Halevi [11], así como el criptosistema NTRU propuesto por Hoffstein, Pipher y Silverman [25]. En el presente capítulo describimos el primer sistema y en el siguiente nos ocupamos del segundo. Comenzamos con GGH, ya que el sistema se deriva de forma relativamente intuitiva del problema del

vector más cercano, tratado en el Capítulo 4. Como referencia principal se utiliza el artículo original [11].

5.1. Descripción del sistema

5.1.1. Función con puerta trasera

La construcción del criptosistema surge de forma directa de la aplicación de una familia de funciones de un sentido con puerta trasera (noción definida en el Capítulo 2) que construyen los autores. Especificamos dicha familia a continuación.

Algoritmo generador de claves (`Gen`)

Este algoritmo toma como entrada un parámetro 1^n y genera como salida dos bases R y B del mismo reticulado de rango máximo $\mathcal{L} \subset \mathbb{Z}^n$, junto con un número real positivo σ. Las bases vienen dadas por matrices $n \times n$ compuestas por los vectores de la base en las columnas de las mismas, $R = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_n]$ y $B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n]$. R y B se eligen de forma tal que el defecto ortogonal dual (Capítulo 3, Definición 53) de la primera sea bajo y el de la segunda alto. Llamaremos a R la "base privada" y a B la "base pública".

Siguiendo la definicón de función con puerta trasera del Capítulo 2, tenemos que $I = (B, \sigma)$ describe una función $f_{(B,\sigma)} : \mathbb{Z}^n \times \mathbb{R}^n \to \mathbb{R}^n$ y $td = R$.

Algoritmo de muestreo (`Sampl`)

Dada $I = (B, \sigma)$, da como salida vectores $\mathbf{v} \in \mathbb{Z}^n$ y $\mathbf{e} \in \mathbb{R}^n$ elegidos de la siguiente manera:

- El vector $\mathbf{v}$ se toma de un cubo incluido en $\mathbb{Z}^n$, una alternativa concreta propuesta en el trabajo es tomar cada entrada de $\mathbf{v}$ de manera uniforme en el conjunto $\ell = \{-n^2, -n^2 - 1, \ldots, n^2 - 1, n^2\} \subset \mathbb{Z}$. Es decir, $v_i \xleftarrow{\boxdot} \ell$ uniforme $\Leftrightarrow v \xleftarrow{\boxdot} \ell^n$ uniforme. El tamaño del cubo, en particular la elección de n^2 como límite fue elegida de forma empírica, en base al tamaño de los enteros con los que se deseaba trabajar.

- El vector $\mathbf{e}$ es elegido de forma aleatoria de $\mathbb{R}^n$ de modo que cada entrada e_i tenga media nula y varianza σ^2. Las dos alternativas propuestas son:

  1. Tomar $e_i \xleftarrow{\boxdot} \pm\sigma$, cada uno con probabilidad $\frac{1}{2}$. De este forma $E(e_i) = \sigma Pr(e_i = \sigma) + (-\sigma)Pr(e_i = -\sigma) = 0$ y $Var(e_i) = E(e_i^2) - E(e_i)^2 = E(e_i^2) = \sigma^2 Pr(e_i^2 = \sigma^2) = \sigma^2$.

  2. Si queremos que e tenga coordenadas enteras, podemos tomar $e_i \leftarrow \{\lceil\sigma\rceil, -\lceil\sigma\rceil, 0\}$ con $Pr(e_i = \lceil\sigma\rceil) = Pr(e_i = -\lceil\sigma\rceil) = \frac{\sigma^2}{2\lceil\sigma\rceil^2}$ y $Pr(e_i = 0) = 1 - \frac{\sigma^2}{\lceil\sigma\rceil^2}$.
     Entonces,

     $$E(e_i) = \lceil\sigma\rceil \tfrac{\sigma^2}{2\lceil\sigma\rceil^2} - \lceil\sigma\rceil \tfrac{\sigma^2}{2\lceil\sigma\rceil^2} + 0(1 - \tfrac{\sigma^2}{\lceil\sigma\rceil^2}) = 0;$$
     $$Var(e_i) = E(e_i^2) - E(e_i)^2 = E(e_i^2) =$$
     $$\lceil\sigma\rceil^2 Pr(e_i^2 = \lceil\sigma\rceil^2) + 0Pr(e_i^2 = 0) = \lceil\sigma\rceil^2 Pr(e_i = \lceil\sigma\rceil \vee e_i =$$
     $$-\lceil\sigma\rceil) = \lceil\sigma\rceil^2 (Pr(e_i = \lceil\sigma\rceil) + Pr(e_i = -\lceil\sigma\rceil)) = \lceil\sigma\rceil^2 \cdot 2\tfrac{\sigma^2}{2\lceil\sigma\rceil^2} = \sigma^2$$

     donde $Pr(e_i = \lceil\sigma\rceil \vee e_i = -\lceil\sigma\rceil) = Pr(e_i = \lceil\sigma\rceil) + Pr(e_i = -\lceil\sigma\rceil)$ debido a que las entradas del vector se eligen de manera independiente.

Algoritmo de evaluación f

Dado $I = (B, \sigma)$ obtenido mediante `Gen`, $\mathbf{v}$ y $\mathbf{e}$ obtenidos mediante `Sampl`, computa y da como salida $\mathbf{c} = f_{(B,\sigma)}(\mathbf{v}, \mathbf{e}) = B\mathbf{v} + \mathbf{e}$.

Algoritmo de inversión (`Inv`)

Dados $td = R$ y $\mathbf{c} \in \mathcal{R}_f = \mathbb{R}^n$, utilizamos el algoritmo del vértice más cercano de Babai para invertir f. Es decir, escribimos $\mathbf{c}$ como combinación lineal de los elementos de R y redondeamos los coeficientes de las entradas a su entero más cercano para obtener un vector del reticulado. Luego, escribimos dicho vector en términos de B para obtener el parámetro $\mathbf{v}$. Una vez obtenido $\mathbf{v}$, también es posible computar $\mathbf{e}$.

En forma matricial tenemos, notando $T = B^{-1}R$,

$$\mathbf{v} \leftarrow T\lceil R^{-1}\mathbf{c}\rfloor \text{ y } \mathbf{e} \leftarrow \mathbf{c} - B\mathbf{v}$$

Observar que $R^{-1}\mathbf{c} = \text{coord}_R(\mathbf{c})$, ya que $\mathbf{c} = R \cdot \text{coord}_R(\mathbf{c})$.

5.1.2. Elección de parámetros

Comentamos ahora los distintos parámetros vinculados y las formas de selección propuestas en [11].

Dimensión del reticulado

El parámetro principal del sistema es la dimensión del reticulado a ser utilizado, es decir el valor de n. Cuanto mayor sea dicho valor, esperamos que el sistema resulte más seguro, ya que el problema CVP subyacente incrementa su dificultad. Por otro lado, el espacio de almacenamiento para las claves, así como el tiempo de ejecución involucrados en la evaluación de la función y su inversión crecen al menos como $\Omega(n^2)$, ya que los métodos de almacenamiento y operaciones para bases generales de reticulados requieren al menos estas cantidades de recursos [7]. La fijación de este parámetro se realiza de forma empírica, en función de los ataques conocidos (Sección 5.3).

Base privada

Luego de fijar la dimensión del reticulado, es necesario tomar una distribución para elegir la base privada del sistema. Las dos alternativas propuestas en [11] son:

- Elegir un reticulado "aleatorio": Dado un $a \in \mathbb{Z}$, elegimos R de manera uniforme en $\{-a, \ldots, a\}^{n \times n}$. Los autores reportaron que en sus experimentos el valor de a no tuvo efecto alguno sobre la calidad de las bases obtenidas y por tanto eligieron trabajar con números pequeños ($a \in \{-4, \ldots, 4\}$).

- Elegir un reticulado "rectangular": Se comienza con la base kI donde $I \in \mathbb{R}^{n \times n}$ es la matriz identidad y $k \in \mathbb{Q}$. Luego se agrega "ruido" a esta matríz, tomando otra matriz R' uniformemente distribuida en $\{-a, \ldots, a\}^{n \times n}$ con un $a \in \mathbb{Z}$ con las mismas características que en el punto anterior y computando $R \leftarrow R' + kI$.

En el trabajo [11] no se dan valores concretos para las cotas superiores e inferiores de $\text{def\_ort}^*(R)$ y $\text{def\_ort}^*(B)$. Únicamente se dan relaciones entre $\text{def\_ort}^*(R)$ y la probabilidad de error en la inversión de la función (Sección 5.2), así como con el tamaño del espacio de búsqueda para realizar un ataque por búsqueda exhaustiva (Sección 5.3.1).

Generación base pública

Una vez fijada la base privada R, debemos dar una forma para construir la base pública de manera eficiente a partir de esta. Como cada base de $\mathcal{L}(R)$, por lo visto en el Capítulo 3, se obtiene con $B = RT$ con $T \in \mathbb{Z}^{n\times n}$ unimodular, elegir B una vez determinada R equivale a elegir una matriz unimodular (pseudo)aleatoria. En [11] esto se consigue multiplicando varias matrices que llaman "elementales". Algunas entradas de estas matrices son elegidas de manera aleatoria y se refieren a la cantidad de matrices utilizadas como cantidad de "pasos de mezclado"[1]. El tipo de matrices busca asegurar, junto a cotas para las magnitudes de los valores numéricos utilizados en las entradas de las matrices, que los coeficientes de B se mantengan en un rango determinado y no "exploten" en tamaño.

5.1.3. Descripción del sistema y eficiencia operacional

Describimos ahora la forma en la que los elementos de la función con puerta trasera son utilizados para crear el criptosistema. Esto sigue la idea general descrita en el Capítulo 2. Además damos los órdenes de tiempo de ejecución de las operaciones de encriptado y desencriptado del sistema.

Generación de claves. Dado 1^n como entrada ejecutamos $\texttt{Gen}(1^n)$ para obtener (B, R, σ), definiendo $pk = (B, \sigma)$ como clave pública y $sk = (R^{-1}, T)$ como clave privada donde $T = B^{-1}R$.

Encriptado. Tomando como entrada un mensaje m y la clave pública $pk = (B, \sigma)$, primero se aplica una función de codificación que puede ser aleatorizada, $\mathbf{v} \leftarrow \text{Cod}(m)$, que mapea m a un vector $\mathbf{v} \in \mathbb{Z}^n$. También debemos dar una función Decod que invierta este proceso. Ambas funciones deben ser públicas, eficientes de computar y cumplir $\text{Decod}(\text{Cod}(m)) = m$. Como vimos en el Capítulo 2, estas funciones son las únicas que no quedan directamente determinadas por la función con puerta trasera antes descrita. En [11], se propone un método de codificación que consiste en tomar bits específicos de $\mathbf{v}$ para transmitir bits del mensaje, evitando que los bits elegidos expongan información del mismo a un atacante a menos de probabilidad negligible. El

[1] Para profundizar en el tipo concreto de matrices, ver [11].

resto de los bits se eligen de manera aleatoria uniforme. En particular, las entradas que se consideran "débiles" son las correspondientes a las filas de B^{-1} con norma pequeña, como veremos en la sección de ataques (Sección 5.3).

Luego de obtener $\mathbf{v}$, utilizamos `Sampl` para obtener el vector de error $\mathbf{e} \in \mathbb{R}^n$ y para obtener el texto cifrado aplicamos $\mathbf{c} \leftarrow f_{B,\sigma}(\mathbf{v}, \mathbf{e}) = B\mathbf{v} + \mathbf{e}$.

Operaciones involucradas en el encriptado de un mensaje: (1) Codificarlo como un vector en $\mathbb{Z}^n$ (2) Elegir un vector (pseudo)aleatorio (3) Realizar una multiplicación entre un vector y una matriz y una suma entre dos vectores. (1) puede realizarse en $O(n)$, (2) en $O(n)$, ya que consiste en tomar n números pseudoaleatorios de un conjunto finito[1] y (3) puede realizarse en $O(n^2)$ por lo que el proceso es $O(n^2)$.

Desencriptado. Para desencriptar $\mathbf{c}$ hacemos uso de la clave privada $sk = (R^{-1}, T)$ para invertir $f_{B,\sigma}$ mediante $\mathbf{v} \leftarrow T\lceil R^{-1}\mathbf{c}\rfloor$. Luego hacemos $m = \text{Decod}(\mathbf{v})$. Es decir, la operación requiere dos multiplicaciones matriz-vector y una operación de redondeo sobre un vector, que puede implementarse en $O(n^2)$.

5.2. Correctitud

Presentamos ahora resultados que dan cotas sobre σ de modo que el algoritmo de inversión `Inv` tenga éxito con probabilidad exponencialmente alta. Las cotas se dan en términos de la norma $\|\cdot\|_\infty$ en $\mathbb{R}^n$, definida como $\|\mathbf{x}\|_\infty = \max_{i\in\{1,\dots,n\}} |x_i|$.

Comenzamos probando cuándo existe un error en el proceso de inversión.

Proposición 16. *Sea R la base privada utilizada en la inversión de $f_{B,\sigma}(\boldsymbol{v}, \boldsymbol{e})$, entonces existe error de inversión si y solo si $\lceil R^{-1}\boldsymbol{e}\rfloor \neq \boldsymbol{0}$.*

Demostración. Sea T la matriz unimodular $T = B^{-1}R$. El algoritmo de inversión da como salida $\mathbf{v}' = T\lceil R^{-1}\mathbf{c}\rfloor$ y $\mathbf{e}' = \mathbf{c} - B\mathbf{v}$. Observar que si $\mathbf{v}$ se computa correctamente, es sencillo obtener $\mathbf{e} = \mathbf{c} - B\mathbf{v}$. Recordando que $\mathbf{c} = B\mathbf{v} + \mathbf{e}$,

[1] Como la generación de un número pseudoaleatorio no depende del largo de la entrada, tenemos que se realiza en $O(1)$.

$$\mathbf{v}' = T\lceil R^{-1}\mathbf{c}\rfloor = T\lceil R^{-1}(B\mathbf{v}+\mathbf{e})\rfloor = T\lceil R^{-1}B\mathbf{v}+R^{-1}\mathbf{e}\rfloor =$$
$$T\lceil (BT)^{-1}B\mathbf{v}+R^{-1}\mathbf{e}\rfloor = T\lceil T^{-1}B^{-1}B\mathbf{v}+R^{-1}\mathbf{e}\rfloor = T\lceil T^{-1}\mathbf{v}+R^{-1}\mathbf{e}\rfloor$$

Como $T \in \mathbb{Z}^{n\times n}$ es unimodular, $T^{-1} \in \mathbb{Z}^{n\times n}$ también lo es y como $\mathbf{v} \in \mathbb{Z}^n$, tenemos $T^{-1}\mathbf{v} \in \mathbb{Z}^n$. Por tanto

$$\mathbf{v}' = T\lceil T^{-1}\mathbf{v}+R^{-1}\mathbf{e}\rfloor = T(T^{-1}\mathbf{v}+\lceil R^{-1}\mathbf{e}\rfloor = \mathbf{v}+T\lceil R^{-1}\mathbf{e}\rfloor$$

Luego $\mathbf{v}' = \mathbf{v} \Leftrightarrow \lceil R^{-1}\mathbf{e}\rfloor = \mathbf{0}$ □

Para lo que sigue asumimos que las entradas e_i del vector de error se eligen de manera equiprobable de $\{\sigma, -\sigma\}$.

Utilizaremos el siguiente hecho,

Teorema 15. *[26] Desigualdad de Hoeffding*
Sean $X_1, \ldots, X_n$ variables aleatorias independientes con $X_i \in [a_i, b_i]$ y sea $S_n = X_1 + \cdots + X_n$, entonces

$$Pr(S_n - E(S_n) \geq t) \leq 2e^{-\frac{2t^2}{\sum_{i=1}^n (b_i-a_i)^2}}$$

El siguiente resultado acota σ, de modo de garantizar una probabilidad de error baja.

Teorema 16. *Sea R la base privada utilizada en la inversión de $f_{B,\sigma}(\boldsymbol{v}, \boldsymbol{e})$. Notamos el máximo de las normas infinito de los vectores de $(R^{-1})^t$ (filas de R^{-1}) como $\frac{\gamma}{\sqrt{n}}$. Entonces tenemos,*

$$Pr(\textit{error de inversión usando } R) < 2n \cdot e^{-\frac{1}{8\sigma^2\gamma^2}}$$

Demostración. Sea $\mathbf{d} := R^{-1}\mathbf{e}$ y notamos la i-ésima entrada de $\mathbf{d}$ por δ_i. Análogamente, sea ϵ_i la i-ésima entrada de $\mathbf{e}$, $\hat{\mathbf{r}}_i$ el vector correspondiente a la fila i-ésima de R^{-1} y $\rho_{i,j}$ la entrada i, j de R^{-1}.

Por la Propisición 16, existe un error en el proceso de inversión si y solo si $\lceil R^{-1}\mathbf{e}\rfloor \neq \mathbf{0}$, es decir si y solo si $|\delta_i| > \frac{1}{2}$.

Fijamos entonces un i y evaluamos $\Pr\big(|\delta_i| \geq \frac{1}{2}\big)$. Por definición de $\mathbf{d}$ tenemos que $\delta_i = \sum_{j=1}^{n} \rho_{i,j}\epsilon_j$. Como para todo $j \in \{1, \ldots, n\}$, $|\rho_{i,j}| \leq \frac{\gamma}{\sqrt{n}}$ fijos y $\epsilon_j \in \{\sigma, -\sigma\}$ con probabilidad $\frac{1}{2}$ cada uno, entonces para todo $i, j \in$

$\{1, \ldots, n\}, \rho_{i,j}\epsilon_j$ son variables aleatorias independientes con media nula en $[-\frac{\sigma\gamma}{\sqrt{n}}, \frac{\sigma\gamma}{\sqrt{n}}]$. Estamos entonces en las hipótesis de la desigualdad de Hoeffding,

$$\Pr\big(|\delta_i| > \tfrac{1}{2}\big) = Pr(|\sum_{j=0}^{n} \rho_{i,j}\epsilon_j| > \tfrac{1}{2}) < 2\exp\Big(-\frac{2(\frac{1}{2})^2}{\sum_{i=1}^{n} 4\frac{\sigma^2\gamma^2}{n}}\Big) = 2\exp\Big(-\frac{2\frac{1}{4}}{4\frac{\sigma^2\gamma^2}{n}\cdot n}\Big) = 2\exp\Big(-\frac{1}{8\sigma^2\gamma^2}\Big)$$

Luego,

$$\Pr(\text{error de inversión usando R}) = \Pr\big(|\delta_1| > \tfrac{1}{2} \vee |\delta_2| > \tfrac{1}{2} \vee \cdots \vee |\delta_n| > \tfrac{1}{2}\big) = \sum_{i=1}^{n} \Pr\big(|\delta_i| > \tfrac{1}{2}\big) < 2n \cdot \exp\Big(-\frac{1}{8\sigma^2\gamma^2}\Big)$$

Notamos que $e^{-\frac{1}{8\sigma^2\gamma^2}} \in (0, 1)$ para todo valor de σ y γ. Además, para tener una probabilidad de error (fijada R) menor que un $\epsilon > 0$, basta tomar $\sigma \leq \frac{1}{\gamma\sqrt{8 \cdot ln(\frac{2n}{\epsilon})}}$. □

Entonces, cuanto mayor sea el valor de σ, más difícil se espera que resulte la instancia de CVP subyacente al sistema. Sin embargo, el teorema anterior muestra que σ debe ser pequeño para que el proceso de desencriptado tenga éxito. En la práctica, uno toma una base privada R, observa la norma de las columnas en R^{-1} y luego toma el máximo valor posible para σ [27]. Analizando la cota sobre la probabilidad de error, notamos que si tanto σ como γ tienden a 0, esta también. Esto se relaciona de manera directa con la elección de R, dado que un defecto ortogonal dual pequeño implica que la norma de los vectores fila de R^{-1} sea pequeña (intuitivamente, un valor de γ pequeño y de $\sqrt{n}$ grande).

5.3. Ataques

Los posibles ataques pasivos que surgen naturalmente al considerar estas construcciones son los siguientes:

- Obtener la base privada, o una base equivalente, de la base pública. Con equivalente, nos referimos a una base con bajo defecto ortogonal dual que tenga esencialmente las mismas propiedades que la base privada.

- Resolver el problema del vector más cercano a $\mathbf{c}$ con respecto al reticulado dado.

- Obtener información del mensaje a partir del texto cifrado $\mathbf{c}$, cuando el vector de error es pequeño.

Describimos estas alternativas en lo que sigue.

5.3.1. Ataques sobre función con puerta trasera

En [11] se presentan dos ataques sobre la función con puerta trasera, junto a análisis estimados de la carga de trabajo que conllevan. En particular, la estimación que realizan los autores para la carga asociada a ataques de fuerza bruta es proporcional al defecto ortogonal dual de la base pública. Este es el motivo por el que se elige una base con estas características como pública.

Como primer paso natural en la ejecución de cualquier ataque del sistema, se considera reducir la base pública B con un algoritmo de reducción de bases (como LLL, Capítulo 4) para obtener una base reducida B' y luego utilizar B' para invertir la función.

Notar que la única característica de R utilizada en el análisis de la probabilidad de error en la inversión de la función, fue que la norma infinito de las filas de R^{-1} era pequeña. Esto implica que las filas de R^{-1} tienen norma pequeña y es equivalente a pedir que R tenga un defecto ortogonal dual bajo. Es decir, que si podemos hallar otra base del reticulado con estas propiedades, esta también puede ser usada. Sin embargo, hallar una base con estas características es un problema que se asume es computacionalmente difícil [11]. Por tanto asumimos como en [11], que luego de aplicar el proceso de reducción de base, la base obtenida B' sigue teniendo un defecto ortogonal dual relativamente alto.

Entonces el ataque consiste en utilizar esta nueva base B' como R para invertir la función y luego aplicar una búsqueda exhaustiva sobre los posibles $\mathbf{d} = B'^{-1}\mathbf{e}$. Para el análisis en [11], se asume que el atacante utiliza B (dado que se supone no puede obtener una base significativamente mejor) y por tanto $B^{-1}\mathbf{c} = B^{-1}(B\mathbf{v} + \mathbf{e}) = \mathbf{v} + B^{-1}\mathbf{e}$. Luego, asumiendo que las entradas de $\mathbf{d}$ son variables aleatorias Gaussianas independientes, se obtiene un espacio de búsqueda exponencial en la dimensión del reticulado ($O(c^n)$ con $c > 1$) y también proporcional al defecto ortogonal dual de la base B.

También se propone junto al proceso de reducción de base que comentábamos al comienzo de esta sección, utilizar mejores algoritmos para resolver CVP de forma aproximada, en particular utilizando el algoritmo del hiperplano más cercano que describimos en la Sección 4.3.2. La seguridad frente a estos ataques (primeros dos puntos mencionados al comienzo de la sección) se realiza de forma empírica, fijando una dimensión suficientemente alta de modo que los mismos no puedan ser realizados en un tiempo factible en la práctica.

5.3.2. Ataques sobre el criptosistema

Otro tipo de ataque que tiene solo sentido en el contexto del criptosistema (no es un ataque sobre la función de un sentido considerada), es utilizar el hecho de que $\mathbf{c} = B\mathbf{v} + \mathbf{e}$ junto con la hipótesis de que $\mathbf{e}$ es un vector con entradas pequeñas, para obtener información parcial del mensaje. Para esto computamos $B^{-1}\mathbf{c} = \mathbf{v} + B^{-1}\mathbf{e}$ y tratamos de estimar algunas entradas del término $B^{-1}\mathbf{e}$. En particular, si la norma de la fila j-ésima de B^{-1} es pequeña en valor absoluto, como asumimos que $\mathbf{e}$ tiene entradas también pequeñas, la entrada j-ésima de $B^{-1}\mathbf{e}$ también lo será. Entonces, $c_j \approx v_j$ es una buena aproximación para la j-ésima entrada del mensaje. Utilizando estas aproximaciones podemos tratar de inferir partes más extensas del mensaje encriptado.

Para contrarrestar este ataque, se debe evitar la utilización de funciones de codificación que mapeen de manera directa el mensaje a un vector de $\mathbb{Z}^n$. Esto como comentamos en la Sección 5.1.1, utilizando únicamente algunos bits específicos para transmitir información o completando algunas de las entradas con bits aleatorios[1].

Ataque de Nguyen

En [27], Nguyen estableció que la elección del vector de error $\mathbf{e}$ en el criptosistema, lo hace extremadamente vulnerable a un ataque por él desarrollado. Explicamos la idea general a continuación.

Para evitar problemas de representación asumimos $(n, \sigma) \in \mathbb{N}^2$. Sea $\vec{\sigma} = (\sigma, \sigma, \ldots, \sigma) \in \mathbb{Z}^n$. La observación principal en la que se basa el ataque es

[1] Esta técnica recibe el nombre de "relleno aleatorio" o *randomised padding* en inglés.

que si $\mathbf{c}$ es un texto cifrado del sistema, entonces $\mathbf{c} + \vec{\sigma} = (B\mathbf{m} + \mathbf{e}) + \vec{\sigma} \equiv B\mathbf{m} \pmod{2\sigma}$. Luego, si B es invertible módulo 2σ, podemos obtener los restos $\mathbf{m} \equiv B^{-1}(\mathbf{c} + \vec{\sigma}) \pmod{2\sigma}$. Entonces, $\mathbf{m} = B^{-1}(\mathbf{c} + \vec{\sigma}) + k \cdot 2\vec{\sigma}$ para un $k \in \mathbb{Z}$ es información significativa que se filtra del sistema.

Además, si se computa $\mathbf{m}_{2\sigma} \equiv \mathbf{m} \pmod{2\sigma}$ de forma exitosa, entonces podemos simplificar la instancia CVP del problema. Para esto, primero observar que $\mathbf{c} - B\mathbf{m}_{2\sigma} = B\mathbf{m} + \mathbf{e} - B\mathbf{m}_{2\sigma} = B(\mathbf{m} - \mathbf{m}_{2\sigma}) + \mathbf{e}$ donde $\mathbf{m} - \mathbf{m}_{2\sigma}$ es de la forma $2\sigma\mathbf{m}'$ donde $\mathbf{m}' \in \mathbb{Z}^n$. Luego, tenemos

$$\frac{\mathbf{c} - B\mathbf{m}_{2\sigma}}{2\sigma} = B\mathbf{m}' + \frac{\mathbf{e}}{2\sigma}$$

Resolviendo esta nueva instancia podemos computar $\mathbf{m} = \mathbf{m}_{2\sigma} + 2\sigma\mathbf{m}'$. En particular, como $\frac{\mathbf{e}}{2\sigma} \in \{\frac{1}{2}\}^n$ es un vector mucho más corto que $\mathbf{e}$[1], es posible que los algoritmos para resolver CVP que no fueron exitosos para la instancia original sí lo sean en la versión más sencilla.

5.4. Seguridad

El sistema GGH no cuenta con una reducción de seguridad que muestre que quebrar el sistema sea al menos tan difícil como resolver un problema subyacente en reticulados [7]. En particular, los valores recomendados para la fijación de parámetros se basan en evaluaciones y experimentaciones empíricas. Las elecciones concretas de los parámetros no se incluyen en este trabajo, por no ser este el foco del mismo.

Notamos como comentamos antes, que si se mapea directamente el mensaje a un $\mathbf{m} \in \mathbb{Z}^n$ el criptosistema no es semánticamente seguro. Se relevaron dos enfoques distintos para justificar este hecho en la bibliografía. En [7], se considera que el proceso de encriptado es determinista, que implica que la elección del vector de error es realizada por la entidad que trasmite el mensaje. Entonces en este caso, para una instancia del experimento $PubK^{eav}_{\mathcal{A},GGH}(n)$ definido en el capítulo 2, el adversario $\mathcal{A}$ tiene disponible la información $pk = (B, \sigma)$ y $(\mathbf{m}_0, \mathbf{e}_0), (\mathbf{m}_1, \mathbf{e}_1)$ dos mensajes dentro del espacio junto a los vectores de error correspondientes. Luego al recibir el texto cifrado desafío, basta con que

[1] El nuevo vector de error tiene largo $\sqrt{\frac{n}{4}}$, mientras que en el caso anterior contábamos con largo $\sigma\sqrt{n}$

$\mathcal{A}$ realice $\mathbf{c}_0 \leftarrow \mathtt{Enc}_{B,\sigma}(\mathbf{m}_0, \mathbf{e}_0)$ y $\mathbf{c}_1 \leftarrow \mathtt{Enc}_{B,\sigma}(\mathbf{m}_1, \mathbf{e}_1)$ y devuelva 0 si $\mathbf{c} = \mathbf{c}_0$ y 1 si no. Por tanto, trivialmente obtenemos $\Pr\big(PubK^{eav}_{\mathcal{A},GGH}(n) = 1\big) = 1$ y por tanto GGH no es CPA-seguro. Por otro lado, si asumimos la elección del vector de error como dentro del proceso de encriptado como en [27], podemos igualmente obtener los vectores de error haciendo $\mathbf{c} - B\mathbf{m}$ teniendo así nuevamente toda la información para determinar el mensaje correspondiente al texto cifrado desafío.

En la práctica, el mensaje puede ser rellenado con contenido aleatorio en el proceso de encriptado para resolver este problema, aunque esto no está rigurosamente justificado [7].

Si bien no se conocen ataques efectivos asintóticamente sobre el sistema, el ataque de Nguyen comentado con anterioridad quiebra el mismo en la práctica, para valores moderadamente grandes del parámetro de seguridad. Este ataque puede ser evitado aumentando el valor del parámetro antes mencionado, sin embargo esto hace que el criptosistema sea impráctico [7]. El principal motivo para esto es, como ya comentamos, que el almacenamiento de bases arbitrarias de reticulados requiere $\Omega(n^2)$ espacio y por tanto implica que las operaciones de encriptado y desencriptado ejecuten en al menos $\Omega(n^2)$. En el siguiente capítulo estudiaremos un sistema más eficiente, que hace uso de reticulados con una estructura especial.

Capítulo 6

Criptosistema NTRU

Continuamos en este capítulo con el sistema NTRU propuesto en [25] por Hoffstein, Pipher y Silverman. Este constituye uno de los sistemas más competitivos en cuanto a eficiencia basados en reticulados, sin embargo tampoco está respaldado (al igual que GGH) por una prueba de seguridad [7]. Originalmente, el sistema fue propuesto en términos de anillos de polinomios, sin embargo puede ser también interpretado en términos de reticulados con una estructura particular. En este capítulo daremos ambas descripciones, estudiando luego el sistema en función de estas.

6.1. Anillos de polinomios

Comenzamos definiendo la noción de anillo conmutativo, según la que definiremos el sistema posteriormente. Esta sección sigue las ideas de [10].

Definición 61. *Anillo conmutativo*
Un anillo conmutativo $(R, +, *)$ *viene dado por un conjunto* R *y dos operaciones* $+, * : R \times R \to R$ *tales que se cumplen*

Propiedades de $+$

- *(Existencia de neutro) Existe un elemento* $e_+ \in R$ *tal que* $e_+ + a = a + e_+ = a$ *para todo* $a \in R$*. Usualmente notamos con* 0 *al inverso aditivo.*
- *(Existencia de inverso) Para todo* $a \in R$*, existe un inverso aditivo* $b \in R$ *tal que* $a + b = b + a = 0$*. Podemos notar el inverso como* $-a$*.*

- *(Asociatividad)* $a + (b + c) = (a + b) + c$ *para todo* $a, b, c \in R$
- *(Conmutatividad)* $a + b = b + a$ *para todo* $a, b \in R$

Es decir que, si consideramos $(R, +, e_+)$*, este es un grupo aditivo conmutativo*[1] *con inverso aditivo* e_+*.*

Propiedades de $*$

- *(Existencia de neutro) Existe un elemento* $e_* \in R$ *tal que* $e_* * a = a * e_* = a$ *para todo* $a \in R$*. Usualmente notamos con* 1 *al inverso multiplicativo.*
- *(Asociatividad)* $a * (b * c) = (a * b) * c$ *para todo* $a, b, c \in R$*.*
- *(Conmutatividad)* $a * b = b * a$ *para todo* $a, b \in R$

La única propiedad que le falta a $(R, *, e_*)$ *para ser un grupo conmutativo es la existencia de inversos multiplicativos.*

Distributividad $a * (b + c) = a * b + a * c$ *para todo* $a, b, c \in R$*.*

Seguimos con un tipo particular de anillos, definidos en base a las clases de equivalencia dadas por las congruencias en el anillo al fijar un elemento del mismo.

Definición 62. *Anillo cociente*
Sea $(R, +, *)$ *un anillo y* $m \in R$ *con* $m \neq e_+$*. Para un* $a \in R$*, notamos* $\overline{a} = \{a' \in R : a' \equiv a \bmod m\}$ *como la clase de equivalencia de* a*. Además, definimos* $R/(m)$ *como el conjunto de clases de equivalencia de* R *según* m*, es decir* $R/(m) = \{\overline{a} : a \in R\}$*.*

Luego definimos dos operaciones $+$ *y* $*$ *sobre* $R/(m)$ *de la siguiente forma,*

$$\overline{a} + \overline{b} = \overline{a + b} \text{ y } \overline{a} * \overline{b} = \overline{a * b}$$

Definimos $(R/(m), +, *)$ *como el* ***anillo cociente*** *de* R *por* m*.*

Notamos que las operaciones anteriores junto con el conjunto $R/(m)$ definen efectivamente un anillo [10].

[1] Equivalente a decir, que es un grupo y además la operación correspondiente satisface la ley de conmutatividad. Estos grupos también son llamados abelianos.

Introducimos ahora el tipo concreto de anillo usado en NTRU: los anillos de polinomios con convolución, que son un caso particular de anillos cociente.

Definición 63. *Anillo de polinomios con convolución*
Sea $n \in \mathbb{Z}^+$. Entonces llamamos ***anillo de polinomios con convolución*** *a la terna $(R, +, *)$ donde $R = \frac{\mathbb{Z}[x]}{(x^n-1)}$ y las operaciones se corresponden a la suma y multiplicación usuales entre polinomios.*

De forma análoga, se define a $(R_q, +, *)$ como el anillo de polinomios con convolución módulo q donde $R_q = \frac{\mathbb{Z}_q[x]}{(x^n-1)}$ y las operaciones se corresponden a la suma y multiplicación usuales módulo q.

Es posible probar [10] que cualquier clase de equivalencia $\overline{a}$ en R y R_q tiene un único elemento representativo de la forma $p(x) = a_{n-1}x^{n-1} + a_{n-2}x + \cdots + a_0$ con a_i enteros o enteros módulo q respectivamente.

Además notamos que el hecho de tomar el elemento $x^n - 1 \in \mathbb{Z}[x]$ hace que el procedimiento de reducción de los polinomios en $\mathbb{Z}[x]$ sea sencillo. En particular, como $\alpha x^n \equiv \alpha \bmod (x^n - 1)$ y $x^j \equiv x^j \bmod (x^n - 1)$ si $j < n$, tenemos para un $k = i \cdot n + j$ con $j < n$

$$\alpha x^k = \alpha x^i \cdot n + j = \alpha (x^n)^i x^j \equiv \alpha 1^i x^j = \alpha x^j \bmod (x^n - 1) \tag{6.1}$$

Es decir, que dado un polinomio $p(x) \in \mathbb{Z}[x]$, basta reducir el exponente de sus monomios módulo n. Dado que la suma de monomios tiene menor igual grado que los sumandos, basta luego con sumar estos resultados para obtener la reducción de $p(x)$ módulo $x^n - 1$.

Notamos que un polinomio $p(x) = a_{n-1}x^{n-1} + a_{n-2}x + \cdots + a_0 \in R$ puede ser también representado por su vector de coeficientes $(a_0, a_1, \ldots, a_{n-1}) \in \mathbb{Z}^n$. Esta idea puede aplicarse de forma análoga para R_q. Veamos ahora una proposición que determina una forma de calcular el producto de dos elementos de R.

Proposición 17. *Producto de convolución*
Dados $p(x) = p_{n-1}x^{n-1} + \cdots + p_1 x + p_0$ y $q(x) = q_{n-1}x^{n-1} + \cdots + q_1 x + q_0$ en R, su producto de convolución está dado por

$$p(x) * q(x) = c(x) = c_{n-1}x^{n-1} + \cdots + c_1 x + c_0 \text{ donde}$$

$$c_k = \sum_{i=0}^{k} \sum_{j=0}^{k} \mathbb{1}_{\{i+j \equiv k \ mod \ n\}} p_i q_{k-i} = \sum_{\substack{i,j \in \{0,\dots,k\}: \\ i+j \equiv k \ mod \ n}} p_i q_{k-i}$$

Demostración. Primero computamos el producto usual de polinomios entre $p(x)$ y $q(x)$,

$$p(x) * q(x) = \Big(\sum_{i=0}^{n-1} p_i x^i \Big) * \Big(\sum_{j=0}^{n-1} q_j x^j \Big) = \sum_{k=0}^{2n-2} \Big(\sum_{l=0}^{k} p_l q_{k-l} \Big) x^k$$

Luego, separamos la última suma y reducimos los exponentes de los monomios con grado mayor o igual a n como vimos en 6.1,

$$\begin{aligned} p(x) * q(x) &= \sum_{k=0}^{n-1} \Big(\sum_{l=0}^{k} p_l q_{k-l} \Big) x^k + \sum_{k=n}^{2n-2} \Big(\sum_{l=0}^{k} p_l q_{k-l} \Big) x^{k-n} = \\ &= \sum_{k=0}^{n-1} \Big(\sum_{\substack{l,j \in \{0,\dots,k\}: \\ l+j=k \Leftrightarrow j=k-l}} p_l q_j \Big) x^k + \sum_{k=n}^{2n-2} \Big(\sum_{\substack{l,j \in \{0,\dots,k\}: \\ l+j=k \Leftrightarrow j=k-l}} p_l q_j \Big) x^{k-n} = \\ &= \sum_{k=0}^{n-1} \Big(\sum_{\substack{l,j \in \{0,\dots,k\}: \\ l+j=k \Leftrightarrow j=k-l}} p_l q_j \Big) x^k + \sum_{k'=0}^{n-2} \Big(\sum_{\substack{l,j \in \{0,\dots,k'+n\}: \\ l+j=k'+n \Leftrightarrow j=(k'+n)-l}} p_l q_j \Big) x^{k'} \end{aligned}$$

donde en el término x^{k-n} se cumple $k - n \in \{0, 1, \dots, n-2\}$, dado que $k \in \{n, \dots, 2n-3, 2n-2\}$. Agrupando los índices de ambas sumas módulo n obtenemos,

$$p(x) * q(x) = \sum_{k=0}^{n-1} \Big(\sum_{\substack{l,j \in \{0,\dots,k\}: \\ l+j \equiv k \text{ mod n}}} p_l q_j \Big) x^k = \sum_{k=0}^{n-1} \Big(\sum_{\substack{l,j \in \{0,\dots,k\}: \\ l+j \equiv k \text{ mod n}}} p_l q_{k-l} \Big) x^k$$

□

Observamos que, si en vez de trabajar en R lo hacemos en R_q, basta con reducir los coeficientes de la última expresión módulo q. Además, si trabajamos con la representación vectorial de polinomios, definimos la convolución entre vectores de manera análoga,

$$(p_0, p_1, \dots, p_{n-1}) * (q_0, q_1, \dots, q_{n-1}) = (c_0, c_1, \dots, c_{n-1})$$

con los c_k definidos en la proposición anterior.

Recordamos que un homomorfismo de anillos entre $(R, +_R, *_R)$ y $(S, +_S, *_S)$ es una función $f : R \to S$ tal que

- $f(a +_R b) = f(a) +_S f(b)$ para todos $a, b \in R$
- $f(a *_R b) = f(a) *_S f(b)$ para todos $a, b \in R$

- $f(e_{*,R}) = e_{*,S}$ y $f(e_{+,R}) = e_{+,S}$

Luego, reducir los coeficientes de un polinomio módulo q es un homomorfismo de anillos natural entre R y R_q con las operaciones de cada anillo, es decir que para todos $a(x), b(x) \in R$,

$$(a(x) + b(x)) \bmod \mathrm{q} = (a(x) \bmod \mathrm{q}) +_q (b(x) \bmod \mathrm{q})$$
$$(a(x) * b(x)) \bmod \mathrm{q} = (a(x) \bmod \mathrm{q}) *_q (b(x) \bmod \mathrm{q})$$

Fijamos también un mapeo $i : R_q \to R$ para ir en la otra dirección. De los múltiples posibles definimos siguiendo [10], dado un $p(x) \in R_q$, $i(p(x)) = p'(x) \in R$ como el único $p'(x) \in R$ que cumple

$$p'(x) \bmod \mathrm{q} = p(x) \text{ con } p'_j \in (-\tfrac{q}{2}, \tfrac{q}{2}] \text{ para todo } j \in \{0, \ldots, n-1\}$$

Observamos que i no es un homomorfismo de anillos.

Enunciamos ahora una propiedad, que permite caracterizar la existencia de inversos multiplicativos en R_q para valor de q primos.

Proposición 18. *[10]*
Sea q primo. Entonces $p(x) \in R_q$ tiene inverso multiplicativo si y solo si $\gcd(p(x), x^n - 1) = 1$ *en* $\mathbb{Z}_q[x]$.

6.2. Descripción del sistema usando anillo de polinomios

Al igual que GGH, NTRU es un sistema de encriptado probabilista, es decir que el proceso de encriptado incluye un elemento aleatorio. Esto implica que cada mensaje del espacio dado tiene varios encriptados posibles [25]. La descripción se hace en base a la publicación [25] que introdujo el sistema.

Una instancia del sistema viene dada por tres parámetros enteros positivos $(n, p, q) \in (\mathbb{Z}^+)^3$, donde n primo, $\gcd(n, q) = \gcd(p, q) = 1$ y $q \gg p$. Además se fijan cuatro conjuntos $\mathcal{T}_f, \mathcal{T}_g, \mathcal{T}_r, \mathcal{M}$, donde para definir los primeros tres es necesario elegir $d_f, d_g, d \in \mathbb{Z}^+$ y la noción de polinomios ternarios.

El espacio de mensajes viene dado por el conjunto de los polinomios $\mathcal{M} = \{m(x) \in R : m_i \in (-\frac{p}{2}, \frac{p}{2}]\} \cong R_p$. Es decir, que el texto plano $m(x)$ es un polinomio en R tal que $m(x) = i(p(x))$ para un $p(x) \in R_p$. Continuamos ahora con la definición de polinomios ternarios.

Definición 64. *Polinomios Ternarios*
Dados $d_1, d_2 \in \mathbb{Z}^+$, definimos el conjunto de polinomios ternarios como sigue

$$\mathcal{T}(d_1, d_2) = \{a(x) \in R : a(x) \text{ tiene } d_1 \text{ coeficientes iguales a } 1,$$
$$d_2 \text{ coeficientes iguales a } -1 \text{ y el resto iguales a } 0\}$$

Llamamos a los polinomios en el conjunto antes definido como ternarios, debido a que los coeficientes son tomados del conjunto $\{-1, 0, 1\}$. Observamos que $\mathcal{T}(d_1, d_2) \subset R$.

Utilizando lo anterior definimos,

$$\mathcal{T}_f = \mathcal{T}(d_f + 1, d_f)$$
$$\mathcal{T}_g = \mathcal{T}(d_g, d_g)$$
$$\mathcal{T}_r = \mathcal{T}(d, d)$$

Observar que no es posible tomar $\mathcal{T}(d_f, d_f)$ ya que los elementos de este conjunto nunca tienen inversos en R_q [25]. Estudiemos el caso en el que q es un número primo, haciendo uso de la Proposición 18.

Proposición 19. *(Ejercicio en [10]) Sea $a(x) \in \mathbb{Z}_q[x]$ con q primo. Entonces,*

1. *$a(1) \equiv 0 \pmod q \Leftrightarrow (x-1)|a(x)$ en $\mathbb{Z}_q[x]$*
2. *Si $a(1) \equiv 0 \pmod q$ entonces $a(x)$ no es invertible en R_q.*

Demostración.

1. ($\Rightarrow$) Sabemos por hipótesis que existe un $k \in \mathbb{Z}$ tal que $a(1) = kq$. Luego $x = 1$ es raíz de $a(x)$ módulo q y por tanto $(x-1)$ es factor de $a(x)$ en $\mathbb{Z}_q[x]$. ($\Leftarrow$) Por hipótesis tenemos, $a(x) = (x-1)b(x)$ en $\mathbb{Z}_q[x]$ y por tanto $a(1) \equiv 0 \pmod q$.

2. Por 1. tenemos que $(x-1)$ es factor de $a(x)$ en $\mathbb{Z}_q[x]$ y por tanto $\gcd(a(x), x^n - 1) = \gcd((x-1)b(x), x^n - 1) \neq 1$, ya que es posible probar por inducción que $(x-1)|x^n - 1$ para todo $n \in \mathbb{Z}^+$. Utilizando la Proposición 18, $a(x)$ no es invertible en R_q.

□

6.2.1. Algoritmos del sistema

Damos ahora la terna de algoritmos probabilistas que componen el sistema.

Algoritmo generador de claves (Gen)

Dado un parámetro de seguridad 1^k, que determina los parámetros (n, p, q, d, d_f, d_g) con las condiciones definidas al cominezo de la Sección 6.2, el algoritmo elige de forma aleatoria uniforme, $f(x) \xleftarrow{\cdot} \mathcal{T}_f$ y $g(x) \xleftarrow{\cdot} \mathcal{T}_g$, donde $f(x)$ debe tener inversos módulo p y q.[1] Notamos por F_p y F_q a estos inversos, es decir que $F_p * f \equiv 1$ en R_p y $F_p * f \equiv 1$ en R_q. Luego, la salida del algoritmo es un par (pk, sk) donde $pk = h(x) \equiv F_q * g$ en R_q y $sk = (f, F_p)$.

Si q o p son primos, resulta sencillo calcular los inversos siguiendo la idea de la Proposición 18, haciendo uso del Algoritmo Extendido de Euclides[2]. De esta forma si $\gcd(f(x), x^n - 1) = 1$ en $\mathbb{Z}_q[x]$, utilizando el algoritmo, se obtienen polinomios $u(x), v(x) \in \mathbb{Z}_q[x]$ que cumplen, $u(x)f(x) + v(x)(x^n - 1) = 1$ en $\mathbb{Z}_q[x]$. Finalmente, tenemos $u(x)f(x) = 1$ en R_q, que implica $F_q = u(x)$. La misma idea puede ser aplicada si p primo. En [10] se describe una generalización de este método para computar inversos módulo p^e con p primo, no incluimos su descripción por exceder esta el alcance del trabajo.

Algoritmo de encriptado ($\texttt{Enc}_{pk}$)

Dado un mensaje $m(x) \in \mathcal{M}$, se elige $r(x) \xleftarrow{\cdot} \mathcal{T}_r$ uniforme y utilizando la clave pública correspondiente, $pk = h$, computa

$$c(x) = \texttt{Enc}_h(m(x)) = (p \cdot r(x) * h(x)) + m(x) \pmod q \in R_q$$

El cálculo de un producto de convolución requiere $O(n^2)$ multiplicaciones, sin embargo en [25] los autores destacan que en NTRU se cumple en general que uno de los parámetros de la multiplicación tiene coeficientes pequeños de modo que el cómputo puede realizarse rápidamente. De aquí que el encriptado de un mensaje requiere $O(n^2)$ operaciones.

[1] Observamos que en [25] se aclara que la elección de p y q se realiza de modo que esto sea cierto para la mayoría de las elecciones de f posibles.

[2] Consultar [2] para ver la definición del algoritmo.

Algoritmo de desencriptado ($\texttt{Dec}_{sk}$)

Dado un texto cifrado $c(x)$, se utiliza la clave privada $sk = (f, F_p)$ para desencriptar el mensaje primero calculando

$$a(x) \equiv f(x) * c(x) \pmod{q} \in R_q$$

A continuación, se toma $a'(x) = i(a(x)) \in R$ con $a'_i \in (-\frac{q}{2}, \frac{q}{2}]$. Luego, $b(x) \equiv F_p(x) * a'(x) \pmod{p}$ y finalmente $\texttt{Dec}_{(f,F_p)}(c(x)) = i(b(x)) \in R$ con coeficientes en $(-\frac{p}{2}, \frac{p}{2}]$.

Resulta claro que el proceso de desencriptado también requiere $O(n^2)$ operaciones.

Notamos que en los productos de convolución $r * h$ y$f * e$, no es necesario realizar multiplicaciones, ya que uno de los factores es un polinomio ternario y por tanto basta con realizar sumas y restas de coeficientes para computar la operación. Los autores de [10] también resaltan que, si bien $f(x)$ tiene coeficientes pequeños por definición, sus inversos módulos p y q parecen estar uniformemente distribuidos.

6.3. Correctitud

Para evaluar los casos en los que el desencriptado se produce de manera exitosa analicemos la siguiente expresión,

$$\begin{aligned} a(x) \equiv f(x) * c(x) &\equiv f(x) * (p \cdot r(x) * h(x) + m(x)) \pmod{q} \\ &= f(x) * pr(x) * h(x) + f(x) * m(x) \pmod{q} \\ &= f(x) * pr(x) * F_q(x) * g(x) + f(x) * m(x) \pmod{q} \\ &= pr(x) * g(x) + f(x) * m(x) \pmod{q} \end{aligned}$$

Donde primero sustituimos la definición de $c(x)$, luego aplicamos la propiedad distributiva seguido de la definición de $h(x)$ y concluimos simplificando los inversos con sus elementos correspondientes.

Luego, para que el desencriptado recupere el mensaje de manera correcta, debe ocurrir que todos los coeficientes en la expresión $pr(x) * g(x) + f(x) * m(x)$

se mantengan menores que q, de modo que estos no cambien al hacerse la reducción.

En [10], escrito por los mismos autores del criptosistema, se propone la siguiente proposición para estudiar una condición bajo la que podemos asegurar que no existe error en el proceso de desencriptado. Para esto, se asume la simplificación $d = d_f = d_g$.

Proposición 20. *Condición suficiente para un desencriptado correcto*
Sean (n, p, q) y $d = d_f = d_g$ los parámetros elegidos para el sistema, cumpliendo las condiciones de la Sección 6.2. Si además se cumple

$$q > (6d + 1)p$$

entonces el desencriptado $\texttt{Dec}_{(f,F_p)}(\texttt{Enc}_h(m))$ recupera el mensaje $m(x)$.

Demostración. Como describimos anteriormente tenemos,

$$\texttt{Dec}_{(f,F_p)}(c(x)) = a(x) \equiv f(x)*c(x) \pmod q = pr(x)*g(x)+f(x)*m(x) \pmod q \tag{6.2}$$

Ahora vamos a considerar la expresión $pr(x) * g(x) + f(x) * m(x)$ en R y acotar el coeficiente más grande en la misma.

Por hipótesis tenemos que $g(x), r(x) \in \mathcal{T}(d, d)$ y por tanto en $r(x) * g(x)$ el máximo valor posible para un coeficiente viene dado por $2d$. Este valor se obtiene cuando en el producto de convolución existe un coeficiente donde se multiplican todos los d coeficientes de $r(x)$ iguales a 1 con los d coeficientes de $g(x)$ iguales a 1 y lo mismo para los coeficientes iguales a -1. El primer caso contribuye con d términos a la suma y el segundo con otros d términos, todos iguales a 1.

Siguiendo un razonamiento análogo, como $f(x) \in \mathcal{T}(d + 1, d)$ y los coeficientes de $m(x)$ se toman de $(-\frac{p}{2}, \frac{p}{2}]$, el coeficiente de mayor magnitud posible en $f(x) * m(x)$ se dará si todos los $(d + 1)$ coeficientes iguales a 1 de $f(x)$ se multiplican con coeficientes en $m(x)$ iguales a $\frac{p}{2}$ y los d iguales a -1 con coeficientes en $m(x)$ iguales a $-\frac{p}{2}$[1]. Así, es posible acotar el mayor coeficiente

[1]En realidad, los coeficientes en m pueden ser a lo sumo $-\lceil \frac{p}{2} - 1 \rceil$, pero asumimos que pueden tomar el valor $-\frac{p}{2}$ para simplificar la expresión resultante, que sigue siendo válida como cota superior.

en $f(x) * m(x)$ por $(d+1)\frac{p}{2} + d\frac{p}{2} = (2d+1)\frac{p}{2}$.

Por tanto, el mayor coeficiente de la expresión 6.2 está acotado por

$$p \cdot 2d + (2d+1)\tfrac{p}{2} = (3d + \tfrac{1}{2})p$$

y si se cumple además que $q > (6d+1)p \Leftrightarrow \frac{q}{2} > (3d + \frac{1}{2})p$, tenemos que el mayor coeficiente de toda la expresión está acotado por $\frac{q}{2}$. De aquí que en el cómputo de $a(x)$ módulo q seguido de la aplicación de i, se recuperarán exactamente los valores de $pr(x) * g(x) + f(x) * m(x)$ en R.

El resto de las operaciones se sigue de forma algebraica,

$$\begin{aligned} b(x) &\equiv F_p(x) * a(x) \pmod{p} \\ &\equiv F_p * (pg(x) * r(x) + f(x) * m(x)) \pmod{p} \\ &\equiv F_p * pg(x) * r(x) + F_p * f(x) * m(x) \pmod{p} \\ &\equiv F_p * f(x) * m(x) \pmod{p} \\ &\equiv m(x) \pmod{p} \end{aligned}$$

Finalmente como $b(x) \equiv m(x) \pmod{p}$, tenderemos que $i(b(x))$ recupera efectivamente el mensaje original. □

Es preciso resaltar que el caso donde los coeficientes de $g(x)$ y $r(x)$ cumplen la condición necesaria para que se alcance la cota de $2d$ para el máximo coeficiente de $r(x) * p(x)$, resulta poco probable, dados que ambos son tomados de manera uniforme de $\mathcal{T}(d, d)$. Esto se cumple también para el caso $f(x) * m(x)$. Por tanto, relajar un poco la condición $q > (6d+1)p$, es decir tomar valores de q más pequeños, no aumenta de gran manera la probabilidad de error al desencriptar [10].

6.4. Descripción del sistema usando reticulados

Daremos ahora una descripción equivalente del sistema haciendo uso de reticulados con una estructura particular, integrando ideas de [7] y [28]. Comenzamos notando que existe un reticulado de dimensión n, que puede ser asociado de forma natural con cada polinomio $r(x) \in R$. Para expresar la base

del mismo en función de su vector de coeficientes $\mathbf{r} \in \mathbb{Z}^n$, definimos la transformación lineal $T : \mathbb{Z}^n \to \mathbb{Z}^n$ que rota las coordenadas de un vector de forma cíclica un lugar y puede representarse de forma matricial como sigue

$$T = \left(\begin{array}{c|c} \mathbf{0}^t & 1 \\ \hline \mathbf{I} & \mathbf{0} \end{array}\right)$$

donde I y $\mathbf{0}$ son la matriz identidad y el vector nulo de tamaño $n-1$.

Notando por T^* la transformación que asocia a un vector $\mathbf{v} \in \mathbb{Z}^n$, su matriz de rotaciones cíclicas $T^*(\mathbf{v}) = [\mathbf{v}, T\mathbf{v}, \ldots, T^{n-1}\mathbf{v}]$ tenemos que la base asociada al reticulado antes descrito es

$$B_r = T^*(\mathbf{r}) = \begin{pmatrix} r_0 & r_{n-1} & \cdots & r_1 \\ r_1 & r_0 & \cdots & r_2 \\ \vdots & \vdots & \ddots & \vdots \\ r_{n-1} & r_{n-2} & \cdots & r_0 \end{pmatrix}$$

Decimos que esta asociación es natural en el sentido de que es un homomorfismo de anillo para R [28], es decir que si B_r y B_s son dos matrices asociadas a $r, s \in R$[1], entonces la matriz asociada a $r * s \in R$ es $B_r \cdot B_s$ y lo mismo se cumple para la suma en R.

Dado un $q \in \mathbb{Z}$ y un $h \in R$, definimos el conjunto

$$M_{h,q} = \{(u, v) \in R^2 : u \in \mathcal{T}_f, v \in \mathcal{T}_g \text{ con } u * h \equiv v \pmod q\}$$

como un *reticulado NTRU*. Los $h \in R$ que consideraremos son las claves públicas del sistema que definimos en la sección anterior. Luego, asumiendo como antes que $f \in \mathcal{T}_f$ invertible en R_q y $g \in \mathcal{T}_g$, el par $(f, g)^t \in M_{h,q}$ si y solo si $h \equiv F_q * g \pmod q$. Una base de $M_{q,h}$ es $\{(1, h)^t, (0, q)^t\} \subset R^2$ y por tanto $\dim(M_{h,q}) = 2n$.

Podemos hacer uso de la transformación T^* para representar la base de $M_{h,q}$ como sigue,

[1]Notar que cuando nos referimos a $r \in R$ hacemos referencia al polinomio $r(x)$ con vector de coeficientes $\mathbf{r} \in \mathbb{Z}^n$. Para no sobrecargar la notación, en ocasiones no escribimos la variable de los polinomios.

$$M = \left(\begin{array}{c|c} \mathbf{I} & \mathbf{O} \\ \hline T^*(h) & q\mathbf{I} \end{array}\right) \in \mathbb{Z}^{2n\times 2n} \text{ equivalente a } M = \begin{pmatrix} 1 & 0 \\ h(x) & q \end{pmatrix} \in R^{2\times 2}$$

donde $\mathbf{I}$ y $\mathbf{O}$ son la matriz identidad y la matriz nula de tamaño n.

Veamos ahora la equivalencia de las primitivas criptográficas del sistema con estos cambios.

Generación de claves. Los vectores privados $(\mathbf{f}, \mathbf{g})^t \in \mathbb{Z}^{2n}$ se eligen de la misma forma que en la descripción anterior, computándose también $\mathbf{h}$ de la misma manera. Observamos que $(f, g)^t \in \mathcal{L}(M)$ dado que

$$\begin{pmatrix} 1 & 0 \\ h(x) & q \end{pmatrix} \begin{pmatrix} f(x) \\ -u(x) \end{pmatrix} = \begin{pmatrix} f(x) \\ f(x) * h(x) - q \cdot u(x) \end{pmatrix} = \begin{pmatrix} f(x) \\ g(x) \end{pmatrix}$$

Luego, la clave pública viene dada por la matriz M, que puede ser representada únicamente por el vector de coeficientes $\mathbf{h} \in \mathbb{Z}_q^n$, dado que el resto de sus componentes están fijas.

Encriptado. Dado un texto cifrado $\mathbf{c} \equiv p\mathbf{r} * \mathbf{h} + \mathbf{m} \pmod q$ tenemos que

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{c} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ p\mathbf{r} * \mathbf{h} + \mathbf{m} \pmod q \end{pmatrix} \equiv \begin{pmatrix} p\mathbf{r} \\ p\mathbf{r} * \mathbf{h} \pmod q \end{pmatrix} + \begin{pmatrix} -p\mathbf{r} \\ \mathbf{m} \pmod q \end{pmatrix}$$

Donde el primer vector es $\begin{pmatrix} p\mathbf{r} \\ \mathbf{c} - \mathbf{m} \end{pmatrix}$ que pertenece a $\mathcal{L}(M)$, dado que puede obtenerse tomando el vector de coeficientes $\begin{pmatrix} p\mathbf{r} \\ \mathbf{b} \end{pmatrix}$ para un $\mathbf{b}$ arbitrario y usualmente $\begin{pmatrix} -p\mathbf{r} \\ \mathbf{m} \pmod q \end{pmatrix}$ es un vector relativamente corto.

Desencriptado. Resulta análogo a la descripción anterior con anillos de polinomios utilizando la nueva representación.

6.5. Ataques

Comenzamos esta sección, siguiendo la idea de [10], observando una relación que orienta la búsqueda de la clave privada $f(x)$.

Como la clave pública cumple, como vimos en la Sección 6.2.1,

$$h(x) \equiv F_q(x) * g(x) \pmod{q} \Leftrightarrow f(x) * h(x) \equiv f(x) * F_q(x) * g(x) \pmod{q}$$
$$\Leftrightarrow f(x) * h(x) \equiv g(x) \pmod{q}$$

Entonces el problema de recuperar la clave $f(x)$ puede ser formulado como sigue: Dado $h(x)$, encontrar polinomios ternarios $f(x) \in \mathcal{T}_f$ y $g(x) \in \mathcal{T}_g$ tales que $f(x)*h(x) \equiv g(x) \pmod{q}$. Además, notamos que si $(f(x), g(x))$ es una solución a dicho problema, entonces para cada $k : 0 \leq k < n$, $(x^k * f(x), x^k * g(x))$ también lo es, en el sentido de que al utilizar $x^k * f(x)$ como clave, obtenemos $x^k * m(x)$. Llamamos a $x^k * f(x)$ rotación de $f(x)$, dado que sus coeficientes son los de $f(x)$ rotados k lugares utilizando la representación vectorial de los polinomios.

6.5.1. Ataques por fuerza bruta

Evaluamos ahora el tamaño del espacio de claves, para determinar que una búsqueda exhaustiva en el mismo no es posible en la práctica. Calculemos el tamaño de un conjunto de polinomios ternarios parametrizado por $d_1, d_2 \in \mathbb{Z}^+$,

$$\#\mathcal{T}(d_1, d_2) = \binom{n}{d_1}\binom{n-d_1}{d_2} = \frac{n!}{d_1!(n-d_1)!}\frac{(n-d_1)!}{d_2!(n-d_1-d_2)!} = \frac{n!}{d_1!d_2!(n-d_1-d_2)!}$$

Donde la expresión surge de elegir primero d_1 coeficientes que serán 1 y luego, de los $n - d_1$ restantes elegir d_2 para que tomen el valor -1. Esta expresión se maximiza con $d_1 = d_2 \approx \frac{n}{3}$ [10].

De lo anterior tenemos que el espacio de búsqueda es

$$\#\mathcal{T}(d_f + 1, d_f) \cdot \frac{1}{n}$$

dado que para toda $f(x)$ sus n rotaciones también son soluciones válidas.

Haremos uso de la aproximación de Striling, $n! \approx \sqrt{2\pi n}(\frac{n}{e})^n > (\frac{n}{e})^n$, para estimar el máximo espacio de búsqueda.

$$\max \#\mathcal{T}(d_f + 1, d_f) \approx \frac{n!}{(\frac{n}{3})!(\frac{n}{3})!(\frac{n}{3})!} = \frac{n!}{(\frac{n}{3}!)^3} \approx (\frac{n}{e})^n \cdot ((\frac{n}{3e})^{\frac{n}{3}})^{-3}$$
$$= (\frac{n}{e})^n \cdot \frac{(3e)^n}{n^n} = 3^n$$

Otro ataque posible consiste en interceptar un texto cifrado $c(x)$ y probar todos los posibles $r(x) \in \mathcal{T}_r$, computando $c(x) - pr(x) * h(x) \pmod{q}$. El tamaño de $\mathcal{T}_r = \mathcal{T}(d, d)$ puede estudiarse con las mismas ideas antes descritas.

6.5.2. Ataque por múltiple transmisión

Si un atacante logra capturar varios textos cifrados que corresponden al mismo mensaje $m(x)$, estando en conocimiento de esto, puede realizar lo que se conoce como un ataque por múltiple transmisión. El atacante puede determinar si algunos de los textos cifrados en una lista $c_1(x), c_2(x), \ldots, c_t(x)$ tienen el mismo texto plano si

$$p^{-1}\big(c_i(x) - c_j(x)\big) \equiv \big(r_i(x) - r_j(x)\big) + p^{-1}\big(m_i(x) - m_j(x)\big)h^{-1}(x) \pmod{q}$$

tiene coeficientes pequeños en todas sus entradas. Suponemos que el adversario puede calcular $h^{-1}(x)$ en R_q y p^{-1} módulo q[1].

Supongamos entonces que $m(x) \in \mathcal{M}$ es encriptado y transmitido varias veces usando la clave pública $h(x)$ y distintos $r_1(x), r_2(x), \ldots, r_t(x) \in \mathcal{T}_r = \mathcal{T}(d, d)$ [29]. Sean $c_1(x), c_2(x), \ldots, c_t(x)$ estos distintos textos cifrados, donde por definición del sistema

$$c_i(x) \equiv p \cdot r_i(x) * h(x) + m(x) \pmod{q} \text{ para todo } i \in \{1, \ldots, t\}$$

Entonces, el atacante puede calcular

$$p^{-1} \cdot \big(h^{-1}(x) * \big(c_i(x) - c_1(x)\big)\big) \equiv r_i(x) - r(x) \pmod{q}$$

como $q \gg p$ y además los coeficientes de $r_i(x) - r_1(x)$ se encuentran en $\{-2, \ldots, 2\}$, el atacante puede recuperar estos valores de forma exacta.

Luego, siguiendo la idea de [29], podemos estudiar para cada coeficiente de $r_i(x) - r_1(x)$, las distintas alternativas posibles para los coeficientes de $r_i(x)$ y $r_1(x)$ respectivamente. Obtenemos entonces los valores de la Tabla 6.1. Podemos utilizar estos datos para determinar o acotar el espacio de búqueda (dependiendo de los valores observados) de los coeficientes de $r_1(x)$ y una vez determinado este podremos desencriptar el mensaje $m(x)$.

Notamos que si un mensaje es desencriptado de forma exitosa utilizando esta técnica, *a priori* esto no revela ninguna información adicional que pueda afectar la seguridad de otros mensajes.

[1] Si no existe el inverso de p módulo q, como $q \gg p$ se puede aplicar el mismo procedimiento

$r_{i,j} \backslash r_{1,j}$	-1	0	1
-1	0	-1	-2
0	1	0	-1
1	2	1	0

Tabla 6.1: Valores posibles para coeficientes de $r_i(x) - r_1(x)$

6.5.3. Ataques basados en reticulados

Como se explicó antes, podemos describir el sistema de forma equivalente utilizando un tipo particular de reticulados. En particular, haciendo uso de esta descripción se pueden desarrollar ataques sobre el sistema.

Uno de los puntos resaltados en la descripción mediante reticulados de la Sección 6.4, es que el vector $(f, g)^t$ de polinomios privados pertenece al reticulado considerado, $\mathcal{L}(M)$. Además, es posible ver [10] que para elecciones típicas de los parámetros, si n es suficientemente grande, con alta probabilidad los vectores más cortos del reticulado serán $(f, g)^t$ y sus rotaciones. De aquí es posible enfocar el ataque de recuperación de la clave privada como un problema de vector más corto (SVP) en el reticulado correspondiente y podemos hacer uso de las herramientas para la resolución de este problema que presentamos en el Capítulo 4 para realizar dicho ataque.

También observamos que el proceso de encriptado puede modelarse como un vector, cuyas últimas n coordenadas se corresponden con los coeficientes de $\mathbf{c}$ y que este puede ser descompuesto en la suma de un vector en el reticulado, $(p\mathbf{r}, \mathbf{c} - \mathbf{m})^t$, más otro vector, $(-p\mathbf{r}, \mathbf{m})^t$, que es típicamente pequeño. Es decir, que el vector $(p\mathbf{r}, \mathbf{c} - \mathbf{m})^t$ es un vector cercano al vector conocido $(\mathbf{0}, \mathbf{c})^t$ y por tanto resolver este problema de vector más cercano (CVP) permite recuperar el mensaje $\mathbf{m}$. Nuevamente podemos utilizar los algoritmos del Capítulo 4 para resolución de CVP para realizar este ataque. Además si en vez de utilizar la base M propuesta como clave pública, en su lugar tomamos

$$M' = \begin{pmatrix} 1 & 0 \\ p\mathbf{h} & q \end{pmatrix}$$

como base del reticulado considerado [10], reducimos la distancia entre el vector objetivo y el reticulado por un factor p. Notamos que el determinante de ambos reticulados permanece invariante, lo que implica que la estimación del largo

del vector más corto utilizada en [10] no cambia.[1]

6.6. Seguridad

Al igual que en el caso de GGH tratado en el Capítulo 5, no existe una reducción de seguridad que respalde al sistema, siendo esta establecida de forma empírica en base a los mejores ataques algorítmicos conocidos [7].

Otra similitud con respecto al sistema anterior es que el mismo no es CPA-seguro. Análogamente a lo discutido en el Capítulo 5, tenemos dos formas de razonar sobre esto. Por un lado, podemos considerar que el proceso de encriptado es determinista [7] si la elección de r es realizada por la entidad que desea encriptar el mensaje y no internamente en el proceso de encriptado. En este caso la verificación es inmediata.[2] Sin embargo, si asumimos que solo el mensaje es pasado como parámetro al algoritmo de encriptado, también es posible mostrar que el sistema no cumple la condición de seguridad exigida. Consideremos una instancia arbitraria de $PubK^{eav}_{\mathcal{A},\Pi}(n)$, donde el adversario recibe $h(x)$ como clave pública del sistema y luego da como salida los mensajes $m_0(x) = 0$ y $m_1(x) = 1$. Al recibir $c(x)$, el adversario calcula $c(1) = pr(1) * h(1) + m(1) = m(1)$, dado que $r(x) \in \mathcal{T}(d, d)$ que implica $r(1) = 0$. De aquí, el adversario puede distinguir siempre entre los dos mensajes dados y por tanto el sistema no resulta CPA-seguro. Al igual que en el caso del sistema anterior, si utilizamos un sistema de rellenado o padding de bits sobre el mensaje, previo a su enriptación, de forma tal que no siempre se verifique $c(1) = m(1)$, si no que este se corresponda con $m_{pad}(1)$ y esto no revela información sobre el mensaje original se podría solucionar este problema en la práctica. Notamos que esto no implica una justificación rigurosa de la seguridad del sistema.

Existe una variante del sistema propuesta por Stehlé y Steinfeld [30] que resulta CPA-segura, aunque menos eficiente en la práctica. Esta se basa en la dificultad del problema de Ring-LWE, variante del que presentaremos en

[1] Esta estimación llamada Heurística Gaussiana se basa en la idea de que para valores grandes de n, $\#(B(\mathbf{0}, r) \cap \mathcal{L})$ puede aproximarse por la cantidad de copias del paralelepípedo fundamental incluidas en $B(\mathbf{0}, r)$.

[2] Observamos que ningún proceso de encriptado determinista puede ser CPA-seguro por esta misma razón.

el siguiente capítulo. Esto aporta valor, según los autores de [30], dado que bajo modificaciones menores del sistema es posible probar formalmente bajo hipótesis razonables que este es seguro. Dado que al interpretarlo utilizando reticulados los mismos tienen una estructura particular, esto da la posibilidad de que se explote esta estructura adicional para realizar ataques que sean más efectivos que en el caso general y el hecho de que pueda probarse que una variante que no cuente con mayores modificaciones es segura, da buenos indicios sobre el diseño del sistema.

Capítulo 7

El problema de "Aprender con Errores"

En este capítulo tratamos el problema de "Aprender con Errores", o LWE por sus siglas en inglés, introducido por Regev en [20] en el año 2005. Dicho problema ha dado lugar a un tipo de criptosistemas eficientes que además admiten una reducción de seguridad a problemas computacionales basados en reticulados, tratados en el Capítulo 4. Esta es una de las grandes diferencias con los sistemas GGH y NTRU, que como vimos en los Capítulos 5 y 6, no estaban respaldados por una prueba de seguridad. En particular, este tipo de sistemas se encuentran dentro de los más efcientes que admiten una prueba de seguridad teórica y sus niveles de eficiencia son suficientemente buenos para considerar su implementación en la práctica [7]. En particular, el sistema que presentaremos en este capítulo resulta semánticamente seguro (CPA seguro) bajo la hipótesis de que el problema LWE es difícil.

7.1. Descripción del problema

En esta sección definimos el problema LWE y comentamos algunas de sus características. Comenzamos con la definición del mismo, siguiendo la línea de [20].

Sea $p = p(n) \leq \texttt{poly}(n)$ con $n \geq 1$, p número entero primo[1] y $\mathbf{s} \in \mathbb{Z}_p^n$

[1]El requisito de la primalidad de p solo es necesario para la reducción del problema en su versión de búsqueda a el de decisión, que trataremos en la Sección 7.2.1. En el resto de los resultados, basta con pedir $p \in \mathbb{Z}^+$, sujeto a las otras restricciones exigidas.

vector secreto. Dados $\mathbf{a}_i \xleftarrow{\square} \mathbb{Z}_p^n$ uniformes y $b_i \in \mathbb{Z}_p$ para todo $i \in \{1, \ldots, n\}$, consideramos una lista de m "ecuaciones con error" cuya incógnica es $\mathbf{s}$,

$$\begin{aligned}
\langle \mathbf{s}, \mathbf{a}_1 \rangle &\approx_\chi b_1 (\text{mod } p) \\
\langle \mathbf{s}, \mathbf{a}_2 \rangle &\approx_\chi b_2 (\text{mod } p) \\
&\vdots \\
\langle \mathbf{s}, \mathbf{a}_i \rangle &\approx_\chi b_i (\text{mod } p) \\
&\vdots \\
\langle \mathbf{s}, \mathbf{a}_m \rangle &\approx_\chi b_m (\text{mod } p)
\end{aligned}$$

El error en las ecuaciones viene dado por la distribución de probabilidad $\chi : \mathbb{Z}_p \to \mathbb{R}^+$ sobre $\mathbb{Z}_p$. Es decir que tenemos para cada $i \in \{1, \ldots, n\}$, $b_i = \langle \mathbf{s}, \mathbf{a}_i \rangle + e_i$ donde cada $e_i \xleftarrow{\chi} \mathbb{Z}_p$ es tomado de forma independiente según χ. Informalmente, el problema de "Aprender con Errores" consiste en recuperar o "aprender" el vector $\mathbf{s}$ de la lista de ecuaciones con error dadas.

Observamos que, si las ecuaciones no tuvieran error y el sistema de ecuaciones correspondiente es de rango máximo, el problema se resuelve trivialmente de manera eficiente aplicando escalerización Gaussiana al sistema de ecuaciones. Sin embargo, una vez agregados los términos de error en las ecuaciones, el problema aumenta su dificultad de manera significativa.

Por ejemplo, si en el caso con errores aplicamos escalerización Gaussiana al sistema, en cada paso del proceso al reemplazar una ecuación por la combinación lineal de otras dos del sistema, amplificaremos el error en las ecuaciones resultantes. Para tener una idea cuantitativa de la expansión del error, asumimos que tenemos m ecuaciones, consideramos que para todo término de error e_i, $|e_i| < \delta$ para un $\delta \in \mathbb{R}^+$ e ignoramos el factor multiplicativo en la combinación lineal (asumimos que solo debemos sumar las ecuaciones para escalerizar el sistema). Entonces, tenemos que al sumar la primera ecuación con las $m-1$ restantes, estas $m-1$ ecuaciones pasarán de tener error menor que δ a error menor que 2δ. En el siguiente paso, ampliamos la cota de error de las ecuaciones $3, \ldots, m$ de 2δ a 4δ. Entonces, al haber considerado $\frac{n}{2}$ variables en la escalerización, las ecuaciones restantes tendrán errores que pueden alcanzar el valor $2^{\frac{n}{2}}\delta$. A menos de que δ sea muy pequeño y q realmente grande (caso

en el que la diferencia entre trabajar de forma modular o no desaparece) las ecuaciones que se obtienen como resultado tienen un error de tal magnitud, que no obtenemos prácticamente ninguna información sobre las incógnitas del sistema [31].

Formalizamos ahora el problema como un problema computacional, en sus versiones de búsqueda y decisión, al igual que como hicimos en el Capítulo 4. Comenzamos con el proceso de obtención de las ecuaciones con error, que modelamos con una distribución de probabilidad siguiendo la descripción del problema.

Definición 65. *[20][32] Distribución LWE*
Sea χ una distribución sobre $\mathbb{Z}_p$ y $\boldsymbol{s} \in \mathbb{Z}_p^n$ un vector que llamamos secreto. Entonces la distribución LWE que notamos $A_{\boldsymbol{s},\chi}$ sobre $\mathbb{Z}_p^n \times \mathbb{Z}_p$, se obtiene tomando $\boldsymbol{a} \xleftarrow{\square} \mathbb{Z}_p^n$ de manera uniforme, $e \xleftarrow{\chi} \mathbb{Z}_p$ y dando como salida $(\boldsymbol{a}, b = \langle \boldsymbol{s}, \boldsymbol{a} \rangle + e \ mod \ p)$.

En las definiciones que siguen $n, p \in \mathbb{Z}^+$ y χ es una distribución sobre $\mathbb{Z}_p$.

La versión de búsqueda del problema que definimos ahora consiste en encontrar el vector secreto dadas las muestras correspondientes.

Definición 66. *[32] Búsqueda-LWE$_{n,p,\chi,m}$*
Dadas m muestras independientes $(\boldsymbol{a}_i, b_i) \xleftarrow{A_{\boldsymbol{s},\chi}} \mathbb{Z}_p^n \times \mathbb{Z}_p$ y un $\boldsymbol{s} \xleftarrow{\square} \mathbb{Z}_p^n$ uniforme fijo para todas las muestras, el objetivo es encontrar $\boldsymbol{s}$.

La versión de decisión correspondiente consiste en distinguir entre muestras de la distribución LWE y la uniforme.

Definición 67. *[32] Decisión-LWE$_{n,p,\chi,m}$*
Dadas m muestras independientes $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_p^n \times \mathbb{Z}_p$ que son tomadas según

1. *$A_{\boldsymbol{s},\chi}$ para un $\boldsymbol{s} \in \mathbb{Z}_p^n$ uniforme (fijo para todas las muestras), o*

2. *La distribución uniforme (en $\mathbb{Z}_p^n \times \mathbb{Z}_p$).*

El objetivo es distinguir cuál es el caso de manera exitosa con probabilidad no negligible.

Típicamente se toma la cantidad de muestras m de forma tal que su valor sea suficientemente grande para asegurar que el secreto $\mathbf{s}$ esté determinado de forma única con alta probabilidad [32]. En particular en [20] se indica que bastan $O(n)$ ecuaciones para que esto se cumpla.

Podemos considerar, como hicimos en el Capítulo 3, el problema de forma matricial tomando $A = [\mathbf{a}_1 | \cdots | \mathbf{a}_m] \in \mathbb{Z}_p^{n \times m}$ donde las columnas son los vectores $\mathbf{a}_i$, y un vector $\mathbf{b} \in \mathbb{Z}_p^m$ (cuyas entradas son los $b_i \in \mathbb{Z}_p$), de modo que las muestras LWE cumplen

$$\mathbf{b}^t = \mathbf{s}^t A + \mathbf{e}^t \pmod{\text{p}} \Leftrightarrow \mathbf{b} = (\mathbf{s}^t A + \mathbf{e}^t)^t = A^t \mathbf{s} + \mathbf{e} \pmod{\text{p}}$$

con $\mathbf{e}$ vector de error, tomado según χ^m.

Observamos que analizando la forma matricial, es claro que la versión de búsqueda del problema puede formularse como un problema de vector más cercano, donde el vector $\mathbf{b}$ se encuentra relativamente cerca a un vector en el reticulado que llamaremos *reticulado LWE*,

$$\Lambda_p(A) := \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y} = A^t \mathbf{s} \pmod{\text{p}} \text{ donde } \mathbf{s} \in \mathbb{Z}^n\}$$

Dicho reticulado es el generado por las filas de la matriz A (columnas de A^t) módulo p.

Siguiendo un razonamiento análogo para la versión de decisión, en el caso en el que las muestras sean tomadas de la distribución uniforme, tendremos que $\mathbf{b}$ estará alejado de todos los vectores de $\Lambda_p(A)$ con alta probabilidad [32].

7.2. Dificultad computacional del problema

En esta sección comentamos el resultado principal de [20], que vincula la dificultad de resolver el problema LWE para ciertas elecciones de los parámetros con la de resolver problemas computacionales sobre reticulados en el peor caso.

Como vimos en la Sección 7.1, uno de los parámetros fundamentales para definir el problema LWE es la distribución en base a la que se eligen los términos

de error que serán agregados a las muestras. Daremos ahora la distribución de error utilizada en [20] para luego enunciar el resultado central del trabajo sobre la dificultad del problema. Esta distribución está basada en la distribución normal que recordamos a continuación.

Definición 68. *Distribución Normal*
Una variable aleatoria X tiene distribución normal o Gaussiana con media μ y varianza σ^2 si su función de distribución $f_X : \mathbb{R} \to \mathbb{R}^+ \cup \{0\}$ viene dada por,

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

En este caso notamos $X \sim \mathcal{N}(\mu, \sigma^2)$. Además, recordamos que la desviación estándar de esta variable aleatoria es $\sigma = \sqrt{\sigma^2}$.

Luego, la distribución de error usada en la reducción es la siguiente:

Definición 69. *[20][7] Distribución $\bar{\Psi}_\alpha$*
Dado $\alpha \in \mathbb{R}^+$, $\bar{\Psi}_\alpha$ es la distribución que se obtiene sobre $\mathbb{Z}_p$, tomando muestras de una distribución normal con media 0 y desviación estándar $\frac{\alpha p}{\sqrt{2\pi}}$, luego redondeando el resultado al entero más cercano y reduciéndolo módulo p.

Observamos que $\bar{\Psi}_\alpha$ es una discretización de una distribución normal. En base a esto, el resultado principal es el siguiente.

Teorema 17. *[20] Sean $n, p, m \in \mathbb{Z}^+$ con $p, m \leq poly(n)$ y $\alpha \in (0, 1)$ tales que $\alpha p > 2\sqrt{n}$. Si existe un algoritmo eficiente que resuelva Decisión-LWE$_{n,p,\bar{\Psi}_\alpha,m}$, entonces existe un algoritmo cuántico[1] eficiente que resuelve GapSVP$_\gamma$ y SIVP$_\gamma$ en reticulados arbitrarios n-dimensionales con $\gamma \in \tilde{O}\big(\frac{n}{\alpha}\big) = O\big(\frac{n}{\alpha}\log(n)\big)$.*

Es decir, que el resultado indica que resolver el problema Decisión-LWE$_{n,p,\bar{\Psi}_\alpha,m}$ es al menos tan difícil como resolver de forma cuántica los problemas GapSVP$_\gamma$ y SIVP$_\gamma$ en reticulados arbitrarios n-dimensionales con $\gamma \in \tilde{O}\big(\frac{n}{\alpha}\big)$. Entonces, el teorema está dando una reducción *cuántica* en tiempo polinomial que hace uso de un oráculo LWE para resolver GapSVP$_\gamma$ y SIVP$_\gamma$ en el peor caso, que notamos

$$\text{GapSVP}_\gamma \leq_p \text{Decisión-LWE}_{n,p,\bar{\Psi}_\alpha,m}$$
$$\text{SIVP}_\gamma \leq_p \text{Decisión-LWE}_{n,p,\bar{\Psi}_\alpha,m}$$

[1] Observamos que la construcción dada en [20] es casi enteramente clásica, siendo solo un paso cuántico. Transformar este paso al modelo clásico, transformaría también toda la reducción.

Esto permite transformar cualquier algoritmo eficiente (clásico o cuántico) que resuelva LWE en un algoritmo cuántico también eficiente para los problemas en reticulados [32].

Como vimos en el Capítulo 4, GapSVP y SIVP son dos de los problemas principales tratados en reticulados. Ambos problemas, para factores de aproximación polinomiales, no se creen **NP**−difíciles sin embargo la comunidad supone que son difíciles (no existen algoritmos eficientes para resolverlos). Los mejores algoritmos conocidos para resolverlos ejecutan en tiempo $2^{O(n)}$ y las computadoras cuánticas no parecen mejorar estos resultados de forma significativa [33]. De aquí que se conjetura que no existe un algoritmo clásico o cuántico en tiempo polinomial que los resuelva de forma aproximada para factores polinomiales y por tanto el resultado antes enunciado es un muy buen indicador de que el problema LWE es computacionalmente difícil de resolver. Por las razones antes mencionadas, LWE es un buen problema candidato sobre el cual basar construcciones criptográficas.

La prueba del Teorema 17 forma la mayor parte de [20] y los detalles técnicos de la misma exceden los contenidos del trabajo. En [32] se da un esquema de la prueba, que ilustra el proceso de la reducción en alto nivel.

Peikert logró transformar parcialmente la parte cuántica de la reducción al modelo de computación clásico en [34]. Decimos que la transformación fue parcial dado que obtuvo una reducción clásica únicamente de GapSVP_γ a LWE, nuevamente con $\gamma \in \tilde{O}\left(\frac{n}{\alpha}\right)$. Esto refuerza la confianza en la dificultad del problema LWE, dado que considerando este resultado un algoritmo clásico eficiente que resuelva LWE, implica un algoritmo clásico eficiente para resolver el problema GapSVP_γ. Además otro problema de la reducción clásica, destacado en [32], es que ésta requiere un módulo exponencial $p \geq 2^{\frac{n}{2}}$ para el problema LWE, mientras que por lo visto antes la reducción cuántica solo requiere $p > 2\frac{\sqrt{n}}{\alpha}$. Considerar un módulo más grande implica la necesidad de usar más bits para representar las muestras de la distribución LWE y esto impacta en los tamaños de claves y la eficiencia de los criptosistemas que se construyan en base al problema. Todo esto indica que todavía se puede progresar en varios aspectos en relación a este tema y sigue siendo considerado un problema abierto en el área como indicó Regev en una reedición de su trabajo original

[35].

7.2.1. Variantes del Problema

En esta sección consideramos la dificultad relativa entre distintas variantes del problema LWE. Seguimos los contenidos de [35] y [13], expandiendo la sección de análisis probabilístico de las pruebas de la Proposiciones 21 y 22.

Comenzamos con el vínculo entre la dificultad de resolver el caso promedio de la versión de decisión del problema con el de resolverlo en el peor caso. Sorprendentemente, la resolución del problema en el caso promedio con probabilidad no negligible, implica una solución en el peor caso también con probabilidad no negligible como veremos a continuación. Cuando un problema cumple esta propiedad, donde resolver cualquier instancia puede ser reducido a la resolución de instancias aleatorias uniformes, decimos que el problema es reducible aleatoriamente a si mismo[1] [9].

Proposición 21. *Peor Caso* $\leq_p$ *Caso Promedio*
Sean $n, p, m \in \mathbb{Z}^+$ *con* $p, m \leq poly(n)$ *y* χ *una distribución en* $\mathbb{Z}_p$*. Si existe un distinguidor eficiente con probabilidad no negligible* $\mathcal{D}_{avg}$ *para el caso promedio (el secreto es elegido de manera uniforme al azar) de Decisión-LWE*$_{n,p,\chi,m}$*, entonces existe un distinguidor eficiente con probabilidad no negligible* $\mathcal{D}_{worst}$ *para el peor caso (el secreto puede ser tomado de cualquier distribución) de Decisión-LWE*$_{n,p,\chi,m}$*.*

Demostración. Dado el algoritmo $\mathcal{D}_{avg}$ antes descrito, cuya salida indica si las muestras de la instancia dada son de la distribución $A_{\mathbf{s},\chi}$ definida en la sección anterior o la uniforme, construimos $\mathcal{D}_{worst}$ como sigue:

Algoritmo 6 $\mathcal{D}_{worst}$ en base a $\mathcal{D}_{avg}$

Entrada: $\mathcal{D}_{avg}$, $(A, \mathbf{b})$ instancia de Decisión-LWE$_{n,p,\chi,m}$ con $\mathbf{s}$ tomado de una distribución arbitraria.
Salida: Muestras tomadas de $A_{\mathbf{s},\chi}$ o de la distribución uniforme.

1: **s'** $\xleftarrow{\square}$ $\mathbb{Z}_p^n$ uniforme
2: **devolver** $\mathcal{D}_{avg}(A, \mathbf{b} + A^t\mathbf{s'})$

Primero observamos que si $(A, \mathbf{b})$ es una muestra LWE (proveniente de la distribución LWE, $A_{\mathbf{s},\chi}$), es decir si $\mathbf{b} = A^t\mathbf{s} + \mathbf{e}$ (mod p), entonces

[1] *Random-self-reducible* en inglés.

$$(A, \mathbf{b} + A^t\mathbf{s'}) = (A, A^t\mathbf{s} + \mathbf{e} + A^t\mathbf{s'}) = (A, A^t(\mathbf{s} + \mathbf{s'}) + \mathbf{e}) \text{ (mod p)}$$

es una muestra de $A_{\mathbf{s}+\mathbf{s'},\chi}$, esto significa una muestra LWE con vector secreto $\mathbf{s} + \mathbf{s'}$. Como $\mathbf{s'}$ es tomado de la distibucion uniforme y $\mathbf{s}$ fijo, tenemos que el nuevo vector secreto $\mathbf{s} + \mathbf{s'}$ también se distribuye de manera uniforme. Para ver esto en el caso general, sea A variable aleatoria en $\mathbb{Z}_p$ con distribución arbitraria y B variable aleatoria con distribución uniforme en $\mathbb{Z}_p$. Consideremos la variable aleatoria $C = A + B$ en $\mathbb{Z}_p$, entonces para todo $j \in \{0, \ldots, p-1\}$

$$\Pr(C = j) = \sum_{i=0}^{p-1} \Pr(A = i) \cdot \Pr(B = j - i) = \tfrac{1}{p} \sum_{i=0}^{p-1} \Pr(A = i) = \tfrac{1}{p}$$

y por tanto C tiene distribución uniforme en $\mathbb{Z}_p$. De aquí que la entrada de $\mathcal{D}_{avg}$ es una instancia correspondiente al caso promedio de Decisión-LWE$_{n,p,\chi,m}$. Luego, en el otro caso, donde $(A, \mathbf{b})$ es tomado de la distribución uniforme, la entrada $(A, \mathbf{b} + A^t\mathbf{s'})$ también se distribuye de manera uniforme.

Entonces, por hipótesis tenemos que $\mathcal{D}_{avg}$ distinguirá de forma exitosa con probabilidad no negligible entre el caso de que $(A, \mathbf{b} + A^t\mathbf{s'})$ sea una muestra LWE con secreto $\mathbf{s} + \mathbf{s'}$ o provenga de la distribución uniforme. Como mostramos que estos casos se corresponden con el caso en el que la entrada al algoritmo construido $\mathcal{D}_{worst}$, $(A, \mathbf{b})$, sea de la distribución LWE con secreto $\mathbf{s}$ o uniforme respectivamente, tendremos que $\mathcal{D}_{worst}$ distingue entre estos dos casos también con probabilidad no negligible. □

Concluimos con la reducción del problema en su versión de búsqueda a la de decisión. Esta también es una característica que resulta sorpendente, dado que resolver la versión de decisión es más sencillo *a priori* que resolver la de búsqueda. Sin embargo, a diferencia del caso anterior, en la reducción veremos que necesitaremos un número mayor de muestras en la instancia de búsqueda, que las usadas para resolver la instancia de decisión para lograr la resolución con probabilidad no negligible.

Proposición 22. *Búsqueda* $\leq_p$ *Decisión*
Sea $2 \leq p \leq poly(n)$ *primo,* $n \in \mathbb{Z}^+$, $m \in \mathbb{Z}^+$ *con* $m < p$ *y* χ *una distribución sobre* $\mathbb{Z}_p$.

Si existe un algoritmo eficiente que resuelve Decisión-LWE$_{n,p,\chi,m}$ *con probabilidad no negligible, entonces existe un algoritmo eficiente con probabilidad no negligible que resuleve Búsqueda-LWE*$_{n,p,\chi,m'}$ *donde* $m' \in O\left(\frac{nmp}{\alpha(n)}\right)$ *con* $\alpha(n)$ *función no negligible.*

Demostración. Sea $\mathcal{D}$ un algoritmo distinguidor eficiente para Decisión-LWE$_{n,p,\chi,m}$. Supongamos sin pérdida de generalidad que

$$\begin{aligned}\Pr\Big(\mathbf{s} \xleftarrow{\cdot} \mathbb{Z}_p^n \wedge A \xleftarrow{\cdot} \mathbb{Z}_p^{n\times m} \wedge \mathbf{e} \xleftarrow{\chi^m} \mathbb{Z}_p^m \wedge \mathcal{D}(A, A^t\mathbf{s}+\mathbf{e}) = 1\Big) \\ - \Pr\Big(A \xleftarrow{\cdot} \mathbb{Z}_p^{n\times m} \wedge \mathbf{b} \xleftarrow{\cdot} \mathbb{Z}_p^m \wedge \mathcal{D}(A, \mathbf{b}) = 1\Big) = \alpha(n)\end{aligned} \tag{7.1}$$

si este no es el caso, basta con invertir la salida del algoritmo para que esto se cumpla. En 7.1 tanto A,$\mathbf{s}$ y $\mathbf{b}$ fueron tomados de manera uniforme. Observamos que

$$\Pr\Big(A \xleftarrow{\cdot} \mathbb{Z}_p^{n\times m} \wedge \mathbf{b} \xleftarrow{\cdot} \mathbb{Z}_p^m \wedge \mathcal{D}(A, \mathbf{b}) = 1\Big) \geq \frac{1}{2}, \tag{7.2}$$

dado que el algoritmo tiene dos salidas posibles y una es la correcta.

El enfoque de la prueba consiste en "adivinar" el vector secreto coordenada a coordenada, dada una instancia de Búsqueda-LWE$_{n,p,\chi,m'}$. Sea entonces $\mathbf{s} = (s_1, \ldots, s_n)^t \in \mathbb{Z}_p^n$ el vector secreto considerado en la instancia dada. Utilizamos el Algoritmo 7 para predecir las coordenadas de $\mathbf{s}$.

Algoritmo 7 Predictor de la coordenada i de $\mathbf{s}$

Entrada: $(A, \mathbf{b})$ instancia de Búsqueda-LWE$_{n,p,\chi,m'}$; $\mathcal{D}$ algoritmo distinguidor eficiente para Decisión-LWE$_{n,p,\chi,m}$ con probabilidad no negligible; $i \in \{1, \ldots, n\}$ índice de la coordenada a predecir de $\mathbf{s}$.

Salida: Predicción para la coordenada i de $\mathbf{s}$

1: **para** $j = 0$ hasta $p-1$ **hacer**
2: **para** $\ell = 1$ hasta L **hacer**
3: $c_\ell \xleftarrow{\cdot} \mathbb{Z}_p^m$ uniforme
4: Construir matriz $C_{\ell,i} \in \mathbb{Z}_p^{n\times m}$ con fila i igual a c_ℓ y el resto de las entradas nulas
5: $A_\ell \leftarrow A + C_{\ell,i}$
6: $\mathbf{b}_\ell \leftarrow \mathbf{b} + j \cdot c_\ell$
7: $d_\ell \leftarrow \mathcal{D}(A_\ell, \mathbf{b}_\ell)$
8: **fin para**
9: **si** la mayoría de $(d_1, \ldots, d_L)$ son iguales a 1 **entonces**
10: **devolver** j
11: **fin si**
12: **fin para**

Analicemos ahora el algoritmo propuesto: su correctitud y probabilidad de éxito.

Supongamos que para una coordenada s_i dada, la predicción del algoritmo

es correcta. Es decir, notando por p_i a la salida del algoritmo tenemos que $s_i = p_i$ para un $p_i \in \{0, \ldots, p-1\}$. Observamos ahora que en este caso, las entradas $(A_\ell, \mathbf{b}_\ell)$ dadas a $\mathcal{D}$ son muestras de la distribución LWE dado que,

$$\begin{aligned}
\mathbf{b}_\ell &= \mathbf{b} + s_i \cdot \mathbf{c}_\ell = A^t\mathbf{s} + \mathbf{e} + s_i \cdot \mathbf{c}_\ell && (7.3)\\
&= (A^t\mathbf{s} + s_i \cdot \mathbf{c}_\ell) + \mathbf{e} && (7.4)\\
&= (A^t + C^t_{\ell,i})\mathbf{s} + \mathbf{e} && (7.5)\\
&= A^t_\ell\mathbf{s} + \mathbf{e} \pmod{\text{p}} && (7.6)
\end{aligned}$$

Damos ahora la justificación para los distintos pasos. En la linea 7.3 utilizamos primero la definición de $\mathbf{b}_\ell$ y luego la de $\mathbf{b}$, para luego, reagrupar términos en 7.4. La ecuación 7.5 se obtiene observando que por contrucción se cumple $C^t_{\ell,i}\mathbf{s} = \mathbf{c}_\ell \cdot s_i$. Finalmente en la linea 7.6, se aplica la definición de A_ℓ.

Estudiemos ahora el caso en el que la predicción no coincide con el valor de s_i, es decir cuando $p_i \neq s_i$. En este caso, las entradas $(A_\ell, \mathbf{b}_\ell)$ dadas a $\mathcal{D}$ se distribuyen de manera uniforme. Para verificar esto primero tenemos

$$\begin{aligned}
\mathbf{b}_\ell &= \mathbf{b} + p_i \cdot \mathbf{c}_\ell = A^t\mathbf{s} + \mathbf{e} + p_i \cdot \mathbf{c}_\ell && (7.7)\\
&= A^t\mathbf{s} + p_i \cdot \mathbf{c}_\ell + \mathbf{e} && (7.8)\\
&= (A_\ell - C_{\ell,i})^t\mathbf{s} + p_i \cdot \mathbf{c}_\ell + \mathbf{e} && (7.9)\\
&= A^t_\ell\mathbf{s} - C^t_{\ell,i}\mathbf{s} + p_i \cdot \mathbf{c}_\ell + \mathbf{e} && (7.10)\\
&= A^t_\ell\mathbf{s} - \mathbf{c}_\ell \cdot s_i + p_i \cdot \mathbf{c}_\ell + \mathbf{e} && (7.11)\\
&= A^t_\ell\mathbf{s} + \mathbf{c}_\ell \cdot (p_i - s_i) + \mathbf{e} \pmod{\text{p}} && (7.12)
\end{aligned}$$

donde en 7.7 aplicamos la definición de $\mathbf{b}_\ell$ y la de $\mathbf{b}$ y reordenamos términos en 7.8. Luego, en 7.9 aplicamos la identidad $A = A_\ell - C_{\ell,i}$ y en 7.10 transponemos la resta de las matrices. La linea 7.11 surge, como en el caso anterior, de observar $C^t_{\ell,i}\mathbf{s} = \mathbf{c}_\ell \cdot s_i$. En la última linea tomamos factor común.

Luego, como en $\mathbb{Z}_p$ con p primo todo elemento no nulo tiene inverso multiplicativo, $m < p$ y $p_i - s_i \neq 0 \pmod{\text{p}}$, tenemos

$$c_\ell \cdot (p_i - s_i) = 0 \pmod{\text{p}} \Leftrightarrow c_\ell = 0 \pmod{\text{p}}$$

Por esto y el hecho de que $\mathbf{c}_\ell$ es uniformemente aleatorio, $\mathbf{c}_\ell \cdot (p_i - s_i)$ también lo es. Por tanto, dados $A_\ell^t \mathbf{s}$ y $\mathbf{e}$ fijos[1], tenemos que $\mathbf{b}_\ell$ uniformemente aleatorio. Con un razonamiento análogo tenemos que $A_\ell = A + C_{\ell,i}$ también resulta uniforme, dado que A es matriz uniforme y $C_{\ell,i}$ tiene una fila uniforme y el resto de las entradas nulas.

Entonces por hipótesis sabemos que el algoritmo $\mathcal{D}$ podrá distinguir entre estos dos casos con probabilidad no negligible. En particular, usando 7.1 y la observación 7.2, tendremos que *en una ejecución* de $\mathcal{D}$, la probabilidad de que $\mathcal{D}$ de como salida 1 en el caso $p_i = s_i$ (donde las muestras son de la distribución LWE, que es el que nos interesa dado que queremos adivinar s_i) es mayor o igual que $\frac{1}{2} + \alpha(n)$.

Veamos ahora que cuando $s_i = p_i$, la probabilidad de que la cantidad de unos en $(d_1, \ldots, d_L)$ sean menor o igual a $\frac{L}{2}$ es muy baja. Como repetimos L veces la ejecución de $\mathcal{D}$, usaremos la cota de Chernoff que enunciamos a continuación en el Teorema 18 para estimar las probabilidades que mencionamos al comienzo del párrafo.

Teorema 18. *[36] Cota de Chernoff*
Sean $X_1, \ldots, X_n$ variables aleatorias independientes tomando valores en $\mathbb{B} = \{0, 1\}$ con $\Pr(X_i = 1) = t_i$. Entonces si $\mu = E\Big(\sum_{i=1}^{n} X_i\Big) = \sum_{i=1}^{n} E(X_i) = \sum_{i=1}^{n} t_i$ y $\delta \in (0, 1]$ tenemos,

$$\Pr\Big(\sum_{i=1}^{n} X_i \leq (1-\delta)\mu\Big) \leq e^{\frac{-\mu\delta^2}{2}}$$

Observemos primero que las salidas del algoritmo $\mathcal{D}$, d_i, son variables aleatorias en $\mathbb{B}$ y que como consideramos el caso en el que $s_i = p_i$, $t_i = \Pr(d_i = 1) \geq \frac{1}{2} + \alpha(n)$. Equivalentemente dado que estamos en el caso $s_i = p_i$, donde las muestras dadas a $\mathcal{D}$ son LWE, tenemos que si $d_i = 1$ el algoritmo $\mathcal{D}$ distinguió de forma correcta y no tuvo éxito de lo contrario, es decir

$$d_i = \begin{cases} 1 \text{ si } \mathcal{D} \text{ responde con éxito,} \\ 0 \text{ si no} \end{cases}$$

La probabilidad de que el Algoritmo 7 falle en este caso, es entonces

[1] Formalmente, consideramos el condicionamiento de estas variables aleatorias a los valores del vector secreto y de las muestras dadas.

$$\Pr(\text{Algoritmo 7 falla}) = \Pr\left(\#\{d_i \in (d_1, \ldots, d_L) : d_i = 1\} \leq \frac{L}{2}\right) \tag{7.13}$$

$$= \Pr\left(\sum_{i=1}^{L} d_i \leq \frac{L}{2}\right) \tag{7.14}$$

Aplicando la cota de Chernoff tomando $\frac{L}{2} = (1-\delta)\mu$ tenemos que

$$\Pr\left(\sum_{i=1}^{L} d_i \leq (1-\delta)\mu\right) \leq e^{\frac{-\mu\delta^2}{2}} \tag{7.15}$$

Ahora como $\mu = \sum_{i=1}^{L} t_i \geq L \cdot (\frac{1}{2} + \alpha)$, asumiendo que se cumple la igualdad, despejamos el valor de δ,

$$\frac{L}{2} = (1-\delta)\mu \Leftrightarrow \frac{L}{2} = (1-\delta)L(\frac{1}{2}+\alpha) \Leftrightarrow \delta = \frac{\alpha}{\frac{1}{2}+\alpha} \tag{7.16}$$

Sustituyendo

$$\begin{cases} \mu = \frac{L}{2(1-\delta)} \\ \delta = \frac{\alpha}{\frac{1}{2}+\alpha} \end{cases}$$

en 7.15 obtenemos que

$$\Pr(\text{Algoritmo 7 falla}) \leq e^{-\frac{L\alpha^2}{1+2\alpha}}$$

dado que

$$\frac{\mu\delta^2}{2} = \frac{L(\frac{1}{2}+\alpha)}{2} \cdot \frac{\alpha^2}{(\frac{1}{2}+\alpha)^2} = \frac{L\alpha^2}{2(\frac{1}{2}+\alpha)} = \frac{L\alpha^2}{1+2\alpha} \tag{7.17}$$

Entonces, tenemos que la probabilidad de falla del Algoritmo 7 decrece de manera exponencial en el grado de α y L. Si queremos que la probabilidad de falla del algoritmo sea menor que un $\epsilon > 0$ debemos tomar

$$e^{-\frac{L\alpha^2}{1+2\alpha}} < \epsilon \Leftrightarrow -\frac{L\alpha^2}{1+2\alpha} < \ln(\epsilon) \Leftrightarrow L > \frac{1+2\alpha}{\alpha^2}\ln\left(\frac{1}{\epsilon}\right)$$

Tenemos entonces que la cantidad de iteraciones L debe estar en el orden de $L \in \Omega\left(\frac{1}{\alpha}\right)$ para asegurar que nuestro algoritmo fallará con una probabilidad extremadamente baja.

Resta analizar el tiempo de ejecución del algoritmo. La hipótesis $p \leq \text{poly}(n)$ asegura que el algoritmo propuesto ejecuta en tiempo polinomial, dado que este número acota la cantidad de iteraciones del **para** de la linea 1. En cuanto a la cantidad de instancias necesarias m', debemos ejecutar el Algoritmo 7 n veces, una vez para cada coordenada del vector $\mathbf{s}$. Luego en cada ejecución del Algoritmo 7, consideramos las m muestras por cada iteración del **para** de la linea 2, que se ejecuta $p \cdot L$ veces. De lo anterior tenemos $m' \in O(nmpL)$ y como vimos que basta tomar una cantidad de muestras del orden de $1\backslash\alpha$, se sigue que $m' \in O(\frac{nmp}{\alpha})$. □

7.3. Criptosistema de clave pública

Presentamos ahora un sistema de clave pública basado en LWE, presentado por Regev en [20].

7.3.1. Descripción y eficiencia operacional

Damos la terna de algoritmos que componen el sistema. Como en los casos anteriores (GGH y NTRU) tendremos un parámetro de seguridad $n \in \mathbb{Z}^+$ y además dos enteros $m, p \in \mathbb{Z}^+$ junto con una distribución de probabilidad χ sobre $\mathbb{Z}_p$.

Algoritmo generador de claves (`Gen`)

Dado el parámetro de seguridad 1^n como entrada, se toman

- **Clave Privada**: $sk = \mathbf{s} \xleftarrow{\cdot} \mathbb{Z}_p^n$ uniforme.
- **Clave Pública**: Para $i = 1, \ldots, m$ se eligen m vectores $\mathbf{a}_i \xleftarrow{\cdot} \mathbb{Z}_p^n$ uniformes de forma independiente. Además, se eligen también de forma independiente m términos de error $e_i \xleftarrow{\chi} \mathbb{Z}_p$. Luego la clave pública viene dada por las m parejas $(\mathbf{a}_i, b_i)$ donde $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \pmod{\text{p}}$. Notamos esto como $pk = (\mathbf{a}_i, b_i)_{i=1}^m$.

Observamos que, como vimos en las secciones anteriores, podemos obtener una formulación equivalente en forma matricial si tomamos $A \xleftarrow{\square} \mathbb{Z}_p^{n\times m}$ uniforme, cuyas columnas representan los vectores $\mathbf{a}_i$ antes mencionados, $\mathbf{e} \xleftarrow{\chi^m} \mathbb{Z}_p^m$ vector con los términos de error y $pk = (A, \mathbf{b})$ con $\mathbf{b} = A^t\mathbf{s} + \mathbf{e}$ (mod p).

El algoritmo da como salida la pareja (pk, sk).

Algoritmo de encriptado ($\texttt{Enc}_{pk}$)

El proceso de encriptado se utiliza para encriptar un único bit. Si se quiere encriptar una cadena de bits, se aplica el procedimiento a cada bit de la misma. Este algoritmo toma como entradas el texto plano que viene dado como un bit $\mu \in \mathbb{B}$ y la clave pública $pk = (A, \mathbf{b})$.

Entonces, para encriptar un bit $\mu \in \mathbb{B} = \{0, 1\}$, primero se elige un conjunto S de manera uniforme de los 2^m subconjuntos posibles de $\{1, \ldots, m\}$, Luego si $\mu = 0$, se da como salida $\big(\sum_{i\in S} \mathbf{a}_i, \sum_{i\in S} b_i\big)$ (mod p) y si $\mu = 1$, $\big(\sum_{i\in S} \mathbf{a}_i, \lfloor \frac{p}{2} \rfloor + \sum_{i\in S} b_i\big)$ (mod p).

En forma matricial esto se corresponde con tomar un vector binario $\mathbf{r} \xleftarrow{\square} \mathbb{B}^m$ de manera uniforme y dar como salida el texto cifrado

$$\texttt{Enc}_{(A,\mathbf{b})}(\mu) = (A\mathbf{r}, \mathbf{b}\mathbf{r}^t + \mu \cdot \lfloor \tfrac{p}{2} \rfloor) \text{ (mod p)}$$

Operaciones involucradas en el encriptado de un mensaje: (1) Tomar un vector (pseudo)aleatorio de m componentes ($\mathbf{r}$), (2) Realizar un producto matriz-vector ($A\mathbf{r}$), (3) un producto entre dos vectores ($\mathbf{b}\mathbf{r}^t$) (4) junto a un producto y una suma de dos números. Luego, (1) puede implementarse en $O(m)$, (2) en $O(nm)$, (3) en $O(m)$ y (4) en $O(1)$. Por tanto, el encriptado puede implementarse en $O(m(1 + n))$. Asumiendo $m = \text{poly}(n)$, es decir $m \in O(n^\alpha)$ con $\alpha \geq 1$ tenemos $O(n^{\alpha+1})$. La operación de "piso" tanto como la reducción módulo p también se consideran $O(1)$. En particular, como se explicó en la Sección 7.1, bastan $m \in O(n)$ muestras y por tanto el encriptado resulta $O(n^2)$.

Algoritmo de desencriptado ($\texttt{Dec}_{sk}$)

El algoritmo de desencriptado toma como parámetros la clave privada $sk = \mathbf{s}$ y un texto cifrado $c = (\mathbf{a}, b) \in \mathbb{Z}_p^n \times \mathbb{Z}_p$ para dar como salida

$$\mu' = \begin{cases} 0 \text{ si } |b - \langle \mathbf{a}, \mathbf{s} \rangle \pmod{\text{p}}| < \frac{\lfloor \frac{p}{2} \rfloor}{2} \\ 1 \text{ si no} \end{cases}$$

En otras palabras, el desencriptado del par $(\mathbf{a}, b)$ es 0 si $b - \langle \mathbf{a}, \mathbf{s} \rangle = b - \mathbf{a}^t\mathbf{s} \pmod{\text{p}}$ se encuentra a menor distancia de 0 que de $\frac{p}{2}$ y 1 en el caso contrario.

Operaciones involucradas en el desencriptado de un texto cifrado: (1) Producto vector-vector ($\langle \mathbf{a}, \mathbf{s} \rangle$) (2) Resta de dos números (3) Reducción módulo p, aplicación de valor absoluto y comparación con $\frac{\lfloor \frac{p}{2} \rfloor}{2}$. Entonces como (1) puede implementarse en $O(n)$, (2) y (3) en $O(1)$ tenemos que el desencriptado es $O(n)$.

7.3.2. Correctitud

Damos ahora una condición sobre los términos de error que aseguran el correcto desencriptado de un bit cifrado. Para esto definimos siguiendo [33] la distribución $\chi^{\star k}$ vinculada a la suma de términos de error.

Definición 70. *Distribución $\chi^{\star k}$*
Dada una distribución χ sobre $\mathbb{Z}_p$ y un $k \in \mathbb{N}$, definimos $\chi^{\star k}$ como la distribución que se obtiene de sumar k muestras independientes de χ donde la suma se realiza módulo p. Para $k = 0$ definimos $\chi^{\star 0}$ como la distribución constante 0.

Es decir, que la distribución da elementos $e_1 + \cdots + e_k \pmod p$, donde $e_i \xleftarrow{\chi} \mathbb{Z}_p$ independientes.

Damos en la siguiente proposición la condición que debe cumplir $\chi^{\star k}$ para que no exista error en el desencritado.

Proposición 23. *[33] Correctitud*
Sea $\delta > 0$. Si para cualquier $k \in \{0, 1, \ldots, m\}$, $\chi^{\star k}$ satisface

$$\Pr\left(e \xleftarrow{\chi^{\star k}} \mathbb{Z}_p \wedge |e| < \frac{\lfloor \frac{p}{2} \rfloor}{2} \right) > 1 - \delta \tag{7.18}$$

entonces, la probabilidad de que exista un error en el desencritpado es a lo sumo δ.

Demostración. Consideremos primero el caso donde $\mu = 0$. En este caso el texto cifrado viene dado por $(\mathbf{a}, b)$ donde $\mathbf{a} = \sum_{i \in S} \mathbf{a}_i$ y

$$b = \sum_{i \in S} b_i = \sum_{i \in S} \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i = \langle \mathbf{a}, \mathbf{s} \rangle + \sum_{i \in S} e_i$$

Por tanto, en este caso, $b - \langle \mathbf{a}, \mathbf{s} \rangle = \sum_{i \in S} e_i$ que tiene distribución $\chi^{\star \# S}$. Por hipótesis, con probabilidad mayor que $1 - \delta$ se cumple $|\sum_{i \in S} e_i| < \frac{\lfloor \frac{p}{2} \rfloor}{2}$. En este caso $b = \sum_{i \in S} e_i$ se encuentra a menor distancia de 0 que de $\lfloor \frac{p}{2} \rfloor$ y por tanto $\mathtt{Dec}_{\mathbf{s}}(\mathbf{a}, b) = 0 = \mu$ con probabilidad mayor que $1 - \delta$.

Para el caso $\mu = 1$, tendremos por un razonamiento análogo al anterior $b = \lfloor \frac{p}{2} \rfloor + \sum_{i \in S} e_i$ cuyo valor absoluto es menor que $\lfloor \frac{p}{2} \rfloor + \frac{\lfloor \frac{p}{2} \rfloor}{2} = \frac{3}{2} \lfloor \frac{p}{2} \rfloor$ con probabilidad $1 - \delta$, que se encuentra a menor distancia de $\lfloor \frac{p}{2} \rfloor$ que de 0 con la misma probabilidad. □

7.4. Seguridad e Indistinguibilidad Computacional

En esta sección mostraremos que el sistema de clave pública presentado en la Sección 7.3 es CPA seguro, asumiendo que el problema LWE es computacionalmente difícil. Recordamos del Capítulo 2, que la noción de que un sistema presente seguridad CPA, refiere a que éste tenga la propiedad de que un atacante que posee la clave pública y ve un texto cifrado correspondiente a uno de dos mensajes posibles conocidos, no pueda distinguir cuál es el mensaje correspondiente al texto cifrado dado.

7.4.1. Indistinguibilidad Computacional

Formalizaremos a continuación la idea de "no poder distinguir" entre los dos mensajes, mediante el concepto de indistinguibilidad computacional. Este tema resulta mucho más extenso que la exposición que se hará en el trabajo; para profundizar en las nociones tratadas consultar [2][37][9]. Comenzamos entonces con la definición de indistinguibilidad computacional.

Definición 71. *[38] Indistinguibilidad Computacional*
Sean X_n e Y_n dos variables aleatorias sobre $\{0,1\}^n$. Decimos que X_n e Y_n son computacionalmente indistinguibles si para todo algoritmo $\mathcal{D} : \{0,1\}^n \to \{0,1\}$ en TPP, existe una función negligible $\epsilon(n)$ tal que

$$|\Pr\Big(t \xleftarrow{\cdot} X_n \wedge \mathcal{D}(t) = 1\Big) - \Pr\Big(t \xleftarrow{\cdot} Y_n \wedge \mathcal{D}(t) = 1\Big)| < \epsilon(n)$$

En este caso notamos $X_n \approx Y_n$.

Es decir que dos variables aleatorias serán indistinguibles, si ningún distinguidor eficiente $\mathcal{D}$ puede distinguirlas con una probabilidad mejor que negligible. Usualmente, se denomina a la diferencia anterior como ventaja del distinguidor [1] y se nota $adv_{\mathcal{D}}(X_n, Y_n)$.

Observar que también se cumple,

$$\Pr\Big(b \xleftarrow{\cdot} \mathcal{D}(X)\Big) = \sum_{x \in \mathbb{B}^n} \Pr\Big(x \xleftarrow{\cdot} X\Big) \cdot \Pr\Big(b \xleftarrow{\cdot} \mathcal{D}(x)\Big)$$

y el valor esperado de $\mathcal{D}$ sobre X resulta,

$$E_{\mathcal{D}}(X) = \sum_{b \in \mathbb{B}} b \cdot \Pr\Big(b \xleftarrow{\cdot} \mathcal{D}(X)\Big) = \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(X)\Big)$$

Por lo que en este contexto (cadenas binarias) también se cumple,

$$adv_{\mathcal{D}}(X_n, Y_n) = |E_{\mathcal{D}}(X_n) - E_{\mathcal{D}}(Y_n)|$$

Si para un $\epsilon > 0$, $adv_{\mathcal{D}}(X_n, Y_n) \geq \epsilon$ entonces decimos que $\mathcal{D}$ es un ϵ-distinguidor entre X_n e Y_n y que ambas son ϵ-distinguibles.

La noción de indistinguibilidad computacional fue introducida por Goldwasser y Micali en [39], donde también se introduce la noción de sistema de encriptado probabilístico. Ya hemos comentado esta noción con anterioridad en el trabajo, dado que GGH y NTRU forman parte de este tipo de sistemas.

[1] Ya consideramos la ventaja de un algoritmo en la ecuación 7.1 de la prueba de la reducción de la versión de Búsqueda a la de Decisión de LWE.

7.4.2. Propiedades de la Indistinguibilidad Computacional

Explicamos ahora algunas propiedades generales de la indistinguibilidad computacional definida en la Sección 7.4.1, siguiendo las ideas presentadas en [38]. Comenzamos con la idea de que si dos distribuciones son similares en el sentido de la Definición 71, entonces lo seguirán siendo si aplicamos cualquier tipo de procesamiento eficiente sobre ellas. Formalizamos esta idea en la siguiente proposición.

Proposición 24. *[38] Clausura bajo operaciones eficientes*
Sean dos distribuciones tales que $X_n \approx Y_n$, entonces para todo algoritmo $\mathcal{M}$ en TPP se cumple $\mathcal{M}(X_n) \approx \mathcal{M}(Y_n)$.

Demostración. Supongamos por absurdo que existe un distinguidor $\mathcal{D}$ que ejecuta en TPP y una función no negligible $\mu(n)$ tal que $\mathcal{D}$ distingue $\mathcal{M}(X_n)$ de $\mathcal{M}(Y_n)$ con probabilidad mayor que $\mu(n)$, es decir

$$|\Pr\Big(t \xleftarrow{\boxdot} \mathcal{M}(X_n) \wedge \mathcal{D}(t) = 1\Big) - \Pr\Big(t \xleftarrow{\boxdot} \mathcal{M}(Y_n) \wedge \mathcal{D}(t) = 1\Big)| > \mu(n)$$

Una descripción equivalente de lo anterior sería primero obtener el resultado del experimento correspondiente a la variable aleatoria y luego aplicar $\mathcal{M}$ a este, de donde obtenemos,

$$|\Pr\Big(t \xleftarrow{\boxdot} X_n \wedge \mathcal{D}(\mathcal{M}(t)) = 1\Big) - \Pr\Big(t \xleftarrow{\boxdot} Y_n \wedge \mathcal{D}(\mathcal{M}(t)) = 1\Big)| > \mu(n)$$

Entonces, si consideramos el algoritmo en TPP $\mathcal{D}' = \mathcal{D}(\mathcal{M})$, éste puede distinguir con ventaja mayor que $\mu(n)$ entre las familias X_n e Y_n y esto contradice que ambas son indistinguibles. □

Seguimos con el hecho de que la relación de indistinguibilidad también cumple la propiedad transitiva,

Proposición 25. *Transitividad de $\approx$*
Sean A_n, B_n y C_n variables aleatorias tales que $A_n \approx B_n$ y $B_n \approx C_n$, entonces $A_n \approx C_n$.

Demostración. Dado un algoritmo distinguidor $\mathcal{D}$, se cumple,

$$adv_{\mathcal{D}}(A, C) = |\Pr(1 \leftarrow \mathcal{D}(A)) - \Pr(1 \leftarrow \mathcal{D}(C))| = \tag{7.19}$$

$$= |\Pr(1 \leftarrow \mathcal{D}(A)) - \Pr(1 \leftarrow \mathcal{D}(B)) + \Pr(1 \leftarrow \mathcal{D}(B)) - \Pr(1 \leftarrow \mathcal{D}(C))| \tag{7.20}$$

$$\leq |\Pr(1 \leftarrow \mathcal{D}(A)) - \Pr(1 \leftarrow \mathcal{D}(B))| + |\Pr(1 \leftarrow \mathcal{D}(B)) - \Pr(1 \leftarrow \mathcal{D}(C))| \tag{7.21}$$

$$\leq 2\epsilon \tag{7.22}$$

donde en 7.19 aplicamos la definición de ventaja, en 7.20 sumamos y restamos $\Pr(1 \leftarrow \mathcal{D}(B))$ y en 7.21 hicimos uso de la desigualdad triangular. Finalmente en 7.22 usamos las hipótesis de donde el primer y el segundo sumando en la equación son menores o iguales que ϵ_1 y ϵ_2 negligibles respectivamente. Tomando $\epsilon = \text{máx}\{\epsilon_1, \epsilon_2\}$ se cumple la propiedad. □

Generalizamos el argumento anterior para una cantidad polinomial de variables aleatorias.

Proposición 26. *Transitividad polinomial - Argumento híbrido*
Sean $X_1, X_2, \ldots, X_m$ una secuencia de variables aleatorias con sus distribuciones asociadas, donde $m = poly(n)$. Si para todo $i \in \{1, \ldots, m-1\}$, $X_i \approx X_{i+1}$ también se cumple $X_1 \approx X_m$.

Demostración. De una forma análoga a la proposicon anterior, para un distinguidor $\mathcal{D}$ entre X_1 y X_m,

$$adv_{\mathcal{D}}(X_1, X_m) = |\Pr(1 \leftarrow \mathcal{D}(X_1)) - \Pr(1 \leftarrow \mathcal{D}(X_m))| = \tag{7.23}$$

$$= |\Pr(1 \leftarrow \mathcal{D}(X_1)) - \Pr(1 \leftarrow \mathcal{D}(X_2)) + \Pr(1 \leftarrow \mathcal{D}(X_2)) + \cdots - \Pr(1 \leftarrow \mathcal{D}(X_{m-1})) + \Pr(1 \leftarrow \mathcal{D}(X_{m-1})) - \Pr(1 \leftarrow \mathcal{D}(X_m))| \tag{7.24}$$

$$\leq |\Pr(1 \leftarrow \mathcal{D}(X_1)) - \Pr(1 \leftarrow \mathcal{D}(X_2))| + |\Pr(1 \leftarrow \mathcal{D}(X_2)) - \Pr(1 \leftarrow \mathcal{D}(X_3))| + \cdots + |\Pr(1 \leftarrow \mathcal{D}(X_{m-1})) - \Pr(1 \leftarrow \mathcal{D}(X_m))| \leq m \cdot \epsilon \tag{7.25}$$

□

En [40] se explica una aplicación de esta proposición que describimos a continuación. Supongamos que queremos probar que $X \approx X'$, pero que ambas distribuciones asociadas parecen diferentes *a priori*. Entonces, si es posible definir $X_1, \ldots, X_m$ con m constante o polinomial en n de forma tal que se verifique

- $X_1 = X$ y $X' = X_m$
- Para todo $i \in \{1, \ldots, m-1\}$, $X_i \approx X_{i+1}$

es posible concluir que $X \approx X'$. Esta técnica ("argumento híbrido") está vinculada con las variables aleatorias híbridas que definimos a continuación.

Sean X y K variables aleatorias. Para $0 \leq i \leq n$ definimos $\pi_i : \mathbb{B}^n \to \mathbb{B}^i$ como la proyeccción sobre las primeras i coordenadas del vector correspondiente y $\hat{\pi}_j : \mathbb{B}^n \to \mathbb{B}^j$ la proyección de las últimas j coordenadas, entonces notamos

$$Y_i = \pi_i(X) \times \hat{\pi}_{n-i}(K)$$

como la v.a. en $\mathbb{B}^n$ donde las primeras i coordenadas son generadas según X y las otras $n-i$ según K. Decimos que las Y_i son *híbridas*, dado que están formadas por valores de la distribución de X y de K y varían entre los extremos $Y_0 = K$ e $Y_n = X$.

Utilizando las v.a. híbridas probamos ahora, siguiendo [37] que el hecho de no poder distinguir dos distribuciones con una única muestra implica no poder hacerlo con una cantidad polinomial de muestras. En este sentido, la noción de indistinguibilidad computacional que presentamos implica el caso donde consideramos una cantidad polinomial de muestras de las distribuciones en cuestión. Dicho resultado es válido si ambas familias pueden construirse en tiempo polinomial, es decir si podemos generar valores con la misma distribución en tiempo polinomial.

Definimos primero el concepto de indistinguibilidad en presencia de múltiples muestras.

Definición 72. *[37] Indistinguibilidad bajo múltiples muestras*
Sea $s : \mathbb{N} \to \mathbb{N}$ polinomio. Dos v.a. X_n e Y_n son indistinguibles dadas $s(n)$ muestras (todas o bien de X_n o de Y_n) si para todo algoritmo $\mathcal{D}$ en TPP, existe una función negligible $\epsilon(n)$ tal que

$$adv_{\mathcal{D}}(X_n^1, \ldots, X_n^{s(n)}, Y_n^1, \ldots, Y_n^{s(n)})$$
$$= |\Pr\left(1 \xleftarrow{\cdot} \mathcal{D}(X_n^1, \ldots, X_n^{s(n)})\right) - \Pr\left(1 \xleftarrow{\cdot} \mathcal{D}(Y_n^1, \ldots, Y_n^{s(n)})\right)| < \epsilon(n)$$

Proposición 27. *[37] Sean dos v.a. X_n e Y_n construibles en tiempo polinomial y $s : \mathbb{N} \to \mathbb{N}$ polinomio. Si $X_n \approx Y_n$, entonces X_n e Y_n son también indistinguibles dadas $s(n)$ muestras.*

Demostración. En la prueba notamos $X_n = X$ e $Y_n = Y$. Supongamos por absurdo que existe un algoritmo eficiente $\mathcal{D}$, que puede distinguir entre X e Y dadas $s(n)$ muestras de una de las distribuciones. Mostraremos que esto implica la existencia de un algoritmo eficiente $\mathcal{D}'$, que distingue entre ambas familias dada una muestra de una de las distribuciones.

Sea $H_i = \pi_i(X) \times \hat{\pi}_{s(n)-i}(Y)$ la variable híbrida donde las primeras i coordenadas son generadas según X y las $s(n) - i$ restantes según Y. Tenemos $H_0 = Y$ y $H_{s(n)} = X$. Por hipótesis $\mathcal{D}$ distingue entre H_0 y $H_{s(n)}$ con ventaja no negligible $\delta(n)$.

Para $1 \leq i \leq n$ notamos $e_i = E_{\mathcal{D}}(H_i)$. Por definicón, como $\mathcal{D}$ tiene ventaja δ, $|e_0 - e_{s(n)}| \geq \delta$ y en particular a menos de una inversión de las salidas del algoritmo podemos asumir $e_{s(n)} - e_0 \geq \delta \Leftrightarrow E_{\mathcal{D}}(X_n) - E_{\mathcal{D}}(Y_n) \geq \delta$.

Mostramos que la ventaja esperada es al menos $\frac{\delta(n)}{s(n)}$.

$$E_i(e_i - e_{i-1}) = \sum_{k=1}^{s(n)} \Pr\Big(k \xleftarrow{\boxdot} i\Big) \cdot (e_k - e_{k-1}) = \tfrac{1}{s(n)} \sum_{i=1}^{s(n)} (e_i - e_{i-1}) = (e_1 - e_0) +$$
$$(e_2 - e_1) + \cdots + (e_{s(n)-1} - e_{s(n)-2}) + (e_{s(n)} - e_{s(n)-1}) = \tfrac{1}{s(n)}(e_{s(n)} - e_0) \geq \tfrac{\delta(n)}{s(n)}$$

Por tanto, para el i aleatorio uniforme, el valor esperado de $e_i - e_{i-1}$ es al menos $\frac{\delta}{s(n)}$. Entonces, tenemos

$$E_{i \in \{0,\ldots,s(n)-1\}}\Big(\Pr\Big(1 \xleftarrow{\boxdot} \mathcal{D}(H_{i+1})\Big) - \Pr\Big(1 \xleftarrow{\boxdot} \mathcal{D}(H_i)\Big)\Big) \geq \frac{\delta(n)}{s(n)} \tag{7.26}$$

Construimos ahora un algoritmo que usando $\mathcal{D}$ toma como entrada una muestra y distingue entre ambas distribuciones.

Algoritmo 8 $\mathcal{D}'$ Distinguidor 1-muestra en base a Distinguidor múltiples-muestras

Entrada: z o bien tomada según X o Y, $\mathcal{D}$ distinguidor en base a múltiples muestras

Salida: $b \in \mathbb{B}$

1: Elegir $i \xleftarrow{\boxdot} \{0, \ldots, s(n) - 1\}$
2: Generar i muestras de X e $s(n) - i - 1$ muestras de Y
3: $r \leftarrow \mathcal{D}(x_1, x_2, \ldots, x_i, z, y_{i+2}, \ldots, y_{s(n)})$
4: **devolver** r

Notamos que la hipótesis de la construcción de las distribuciones en tiempo polinomial es utilizada en el paso donde se generan muestras de ambas en el algoritmo.

Observamos que cuando el algoritmo se invoca para $z \xleftarrow{\cdot} X$, al ser i elegido tenemos que la linea 3 ejecuta $\mathcal{D}(H_{i+1})$ y cuando $z \xleftarrow{\cdot} Y$, $\mathcal{D}(H_i)$. Entonces, en este caso $\mathcal{D}$ distingue en media entre H_i y H_{i+1} con al menos ventaja $\frac{\delta(n)}{s(n)}$ no negligible, por tanto tenemos que el nuevo algoritmo $\mathcal{D}'$ distinguirá entre X e Y con ventaja no negligible, de donde resulta la prueba.

Formalmente tenemos,

$$E_{\mathcal{D}'}(X) = \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}'(X)\Big) = \tfrac{1}{s(n)} \sum_{i=0}^{s(n)-1} \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(H_{i+1})\Big)$$

y análogamente

$$E_{\mathcal{D}'}(Y) = \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}'(Y)\Big) = \tfrac{1}{s(n)} \sum_{i=0}^{s(n)-1} \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(H_i)\Big)$$

Usamos estos resultados para calcular la ventaja de $\mathcal{D}'$,

$$\begin{aligned} adv_{\mathcal{D}'}(X,Y) &= |\Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}'(X)\Big) - \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}'(Y)\Big)| \\ &= \tfrac{1}{s(n)} |\sum_{i=0}^{s(n)-1} \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(H_{i+1})\Big) - \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(H_i)\Big)| \\ &= \tfrac{1}{s(n)} |\Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(H_{s(n)})\Big) - \Pr\Big(1 \xleftarrow{\cdot} \mathcal{D}(H_0)\Big)| \\ &\geq \tfrac{1}{s(n)} \cdot \tfrac{\delta(n)}{s(n)} \end{aligned}$$

Esto concluye la prueba. □

7.4.3. Seguridad del sistema LWE

Probamos ahora, haciendo uso de los resultados de la Sección 7.4.1, que el sistema de clave pública LWE es CPA-seguro asumiendo la dificultad del problema LWE.

Queremos probar que para cualquier $k = \text{poly}(n)$, dados k textos planos $\mu_1, \ldots, \mu_k$ que se corresponden a un elemento de $\mathbb{B}$ y una clave pública del

sistema pk obtenida ejecutando el algoritmo $\texttt{Gen}$, los textos cifrados correspondientes usando pk serán indistinguibles de k textos cifrados de $\neg\mu_1, \ldots, \neg\mu_k$ usando pk.[1] Es decir,

$$(\texttt{Enc}_{pk}(\mu_1), \ldots, \texttt{Enc}_{pk}(\mu_k)) \approx (\texttt{Enc}_{pk}(\neg\mu_1), \ldots, \texttt{Enc}_{pk}(\neg\mu_k)) \tag{7.27}$$

Debido a la Proposición 27 probada en la sección anterior, tenemos que para esto basta probar

$$\texttt{Enc}_{pk}(1) \approx \texttt{Enc}_{pk}(0) \tag{7.28}$$

Mostramos que 7.28 se cumple utilizando un argumento híbrido explicado en términos generales en la Sección 7.4.2. Siguiendo [13], probamos que las variables híbridas definidas son indistinguibles dos a dos. En las siguientes descripciones, cada variable híbrida estará compuesta por una componente pk que hace referencia a la clave pública y c al texto cifrado. En lo que sigue, $n, p, m \in \mathbb{Z}^+$ con $p, m \leq \text{poly}(n)$ y χ es una distribución sobre $\mathbb{Z}_p$.

Variable híbrida 1 (H_1)

- $pk = (A, \mathbf{b}) = (A, A^t\mathbf{s} + \mathbf{e})$ donde $A \xleftarrow{\$} \mathbb{Z}_p^{n\times m}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_p^n$ uniformes y $\mathbf{e} \xleftarrow{\chi^m} \mathbb{Z}_p^m$.
- $c = \texttt{Enc}_{pk}(0) = (A\mathbf{r}, \mathbf{b}\mathbf{r}^t)$ para un $\mathbf{r} \xleftarrow{\$} \mathbb{B}^m$ uniforme.

Ahora, para obtener la variable híbrida 2, sustituímos el vector $\mathbf{b}$ de H_1 por un vector uniformemente aleatorio.

Variable híbrida 2 (H_2)

- $pk = (A, \mathbf{b})$ donde $A \xleftarrow{\$} \mathbb{Z}_p^{n\times m}$, $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_p^m$ uniformes.
- $c = \texttt{Enc}_{pk}(0) = (A\mathbf{r}, \mathbf{b}\mathbf{r}^t)$ para un $\mathbf{r} \xleftarrow{\$} \mathbb{B}^m$ uniforme.

Para obtener la tercera variable híbrida sustituimos el texto cifrado $\texttt{Enc}_{pk}(0) = (A\mathbf{r}, \mathbf{b}\mathbf{r}^t)$ del caso anterior por un vector aleatorio uniforme.

Variable híbrida 3 (H_3)

[1] Esto es lo que se conoce como seguridad CPA dadas múltiples muestras. [1]

- $pk = (A, \mathbf{b})$ donde $A \xleftarrow{\boxdot} \mathbb{Z}_p^{n \times m}$, $\mathbf{b} \xleftarrow{\boxdot} \mathbb{Z}_p^m$ uniformes.
- $c = (\mathbf{u}, v) \xleftarrow{\boxdot} \mathbb{Z}_p^n \times \mathbb{Z}_p$ uniforme.

La cuarta variable híbrida se obtiene sustituyendo el vector aleatorio uniforme agregado en H_3 por el encriptado de un 1.

Variable híbrida 4 (H_4)

- $pk = (A, \mathbf{b})$ donde $A \xleftarrow{\boxdot} \mathbb{Z}_p^{n \times m}$, $\mathbf{b} \xleftarrow{\boxdot} \mathbb{Z}_p^m$ uniformes.
- $c = \mathtt{Enc}_{pk}(1) = (A\mathbf{r}, \mathbf{b}\mathbf{r}^t + \lfloor \frac{p}{2} \rfloor)$ para un $\mathbf{r} \xleftarrow{\boxdot} \mathbb{B}^m$ uniforme.

Finalmente, volvemos a transformar la clave pública en una clave bien formada del sistema.

Variable híbrida 5 (H_5)

- $pk = (A, \mathbf{b}) = (A, A^t\mathbf{s} + \mathbf{e})$ donde $A \xleftarrow{\boxdot} \mathbb{Z}_p^{n \times m}$, $\mathbf{s} \xleftarrow{\boxdot} \mathbb{Z}_p^n$ uniformes y $\mathbf{e} \xleftarrow{\chi^m} \mathbb{Z}_p^m$.
- $c = \mathtt{Enc}_{pk}(1) = (A\mathbf{r}, \mathbf{b}\mathbf{r}^t + \lfloor \frac{p}{2} \rfloor)$ para un $\mathbf{r} \xleftarrow{\boxdot} \mathbb{B}^m$ uniforme.

Para probar que $H_i \approx H_{i+1}$ para todo $i \in \{1, \ldots, 4\}$ debemos introducir dos nociones preliminares. Comenzamos con una medida estándar usada para medir distancia entre dos distribuciones.

Definición 73. *[2] Distancia estadística*
Sean X e Y dos variables aleatorias discretas que toman valores en un conjunto Ω. La distancia estadística entre las distribuciones de X e Y viene dada por

$$\Delta(X, Y) = \frac{1}{2} \sum_{\omega \in \Omega} |\Pr(X = \omega) - \Pr(Y = \omega)| \tag{7.29}$$

Notar que $0 \leq \Delta(X, Y) \leq 1$. Luego, decimos que si $\Delta(X, Y) \leq \epsilon$, X e Y son ϵ-cercanas.

También podemos asociar a este concepto la noción de indistinguibilidad estadística.

Definición 74. *[2] Indistinguibilidad estadística*
Sean X_n e Y_n dos variables aleatorias discretas que toman valores en un conjunto Ω. Decimos que X_n e Y_n son estadísticamente indistinguibles, si existe una función negligible $\epsilon(n)$ tal que

$$\Delta(X_n, Y_n) \leq \epsilon(n)$$

Como la distancia estadística no aumenta bajo la aplicación de funciones, es decir que para toda función $f : \Omega \to \Omega'$ con Ω' conjunto finito, $\Delta(f(X), f(Y)) \leq \Delta(X, Y)$ [9], tenemos que si dos distribuciones son estadísticamente indistinguibles, entonces también son computacionalmente indistinguibles.

Podemos enunciar ahora el segundo resultado necesario para finalizar la prueba de seguridad del sistema.

Proposición 28. *[13] Sean* $n, p \in \mathbb{Z}^+$,$A \xleftarrow{\square} \mathbb{Z}_p^{n\times m}$ *según una distribución arbitraria,* $\boldsymbol{r} \xleftarrow{\square} \mathbb{B}^m$ *y* $\boldsymbol{u} \xleftarrow{\square} \mathbb{Z}_p^n$ *uniformes.*

Entonces dado un $\epsilon > 0$*, si* $m > n\log(p) + 2\log\left(\frac{1}{\epsilon}\right)$*, la distribución de* $(A, A\boldsymbol{r})$ *y la de* $(A, \boldsymbol{u})$ *son* ϵ*-cercanas.*

En particular, por lo anterior tenemos $(A, A\mathbf{r}) \approx (A, \mathbf{u})$. Intuitivamente, esto nos indica que podemos cambiar, dado un m suficientemente grande, un elemento aleatorio $\mathbf{u}$ por $A\mathbf{r}$ con las condiciones expuestas en la Proposición 28. Usaremos esta proposición para probar el resultado principal:

Proposición 29. *Considerando las variables híbridas antes definidas, se cumple que* $H_1 \approx H_5$.

Demostración. Probaremos que $H_i \approx H_{i+1}$ para todo $i \in \{1, \ldots, 4\}$, de donde utilizando la propiedad transitiva se cumple el resultado.

- $H_1 \approx H_2$: Se cumple asumiendo la dificultad computacional de Decisión-LWE$_{n,p,\chi,m}$, es decir que no existe un algoritmo $\mathcal{D}$ en TPP que para un $\mathbf{s} \xleftarrow{\square} \mathbb{Z}_p^n$ uniforme verifique

$$adv_{\mathcal{D}}(A_{\mathbf{s},\chi}, U) = |\Pr\Big((A, \mathbf{b}) \xleftarrow{A_{\mathbf{s},\chi}} \mathbb{Z}_p^n \times \mathbb{Z}_p \wedge \mathcal{D}(A, \mathbf{b}) = 1\Big) - \Pr\Big((A, \mathbf{b}) \xleftarrow{\square} \mathbb{Z}_p^n \times \mathbb{Z}_p \wedge \mathcal{D}(A, \mathbf{b}) = 1\Big)| > \epsilon(n)$$

  con $\epsilon(n)$ negligible. Aquí U denota la distribución uniforme en $\mathbb{Z}_p^n \times \mathbb{Z}_p$.

- $H_2 \approx H_3$: Se cumple por la Proposición 28.

- $H_3 \approx H_4$: El cambio de $\mathbf{u}$ por $A\mathbf{r}$ se cumple por la Proposición 28 y el del v uniforme por $\mathbf{br}^t + \lfloor \frac{p}{2} \rfloor$ debido a la dificultad computacional de Decisión-LWE$_{n,p,\chi,m}$.

- $H_4 \approx H_5$: Nuevamente, asumiendo la dificultad computacional de Decisión-LWE$_{n,p,\chi,m}$.

□

Además como $H_1 = (pk, \texttt{Enc}_{pk}(0))$ y $H_5 = (pk, \texttt{Enc}_{pk}(1))$, entonces esto prueba 7.28, asumiendo la dificultad computacional de Decisión-LWE$_{n,p,\chi,m}$. ¿Qué tan seguros podemos estar de la hipótesis sobre la dificultad de Decisión-LWE$_{n,p,\chi,m}$? La robustez de dicha hipótesis viene dada por el teorema principal de [20] (Teorema 17) y los contenidos de este capítulo, donde si tomamos $\chi = \bar{\Psi}_\alpha$ junto a parámteros adecuados (siguiendo las hipótesis de los distintos resultados discutidos en este capítulo) tenemos que el problema es al menos tan difícil como problemas clásicos en reticulados que han sido extensamente estudiados por la comunidad científica y parecen ser de gran dificultad computacional. Lo anterior vale incluso considerando algoritmos cuánticos, de donde resulta que el criposistema LWE es CPA-seguro incluso bajo ataques cuánticos para parámetros apropiados. Esta es la característica distintiva de este sistema como construcción criptográfica.

Capítulo 8

Conclusiones y Trabajo Futuro

Este proyecto se elaboró con el objetivo de brindar una introducción, que busca ser auto-contenida, a la criptografía post cuántica basada en reticulados. Podemos dividir los contenidos del trabajo en tres partes que describimos a continuación. La primera parte (Capítulos 1 y 2) contextualiza el interés por el estudio de la criptografía en general y también el tipo de sistemas post cuánticos (Capítulo 1), además de presentar (Capítulo 2) los fundamentos necesarios utilizados en el resto del trabajo.

En la segunda parte (Capítulos 3 y 4), se introdujeron los reticulados y sus propiedades fundamentales (Capítulo 3), dado que son los objetos matemáticos que resultan transversales a todo el trabajo. Luego en el Capítulo 4, se exploraron los principales problemas computacionales asociados a los mismos en sus distintas variantes y se indicaron resultados que caracterizaban su dificultad. En este sentido, se constató que, incluso para factores de aproximación polinomiales, la dificultad de los mismos es elevada, o de forma equivalente en el contexto del trabajo, que no existen algoritmos eficientes para su resolución. A continuación del estudio de la complejidad de dichos problemas computacionales, se relevaron las principales propuestas algorítmicas para resolverlos (Sección 4.3). El algoritmo central de esta sección es el algoritmo LLL, que ha sido desde su invención en 1982 el avance más significativo en el área en cuanto a aspectos algorítmicos y por tanto su entendimiento resulta un componente relevante a la hora de analizar las distintas construcciones criptográficas basadas en reticulados. En particular como vimos en el trabajo, una aplicación inmediata del LLL, es resolver el problema del vector más corto en

Sistema \ Operación	$\texttt{Enc}_{pk}$	$\texttt{Dec}_{sk}$
GGH	$O(n^2)$	$O(n^2)$
NTRU	$O(n^2)$	$O(n^2)$
LWE	$O(m(1+n)) \in O(n^2)$	$O(n)$

Tabla 8.1: Órdenes asintóticos operaciones de encriptado y desencriptado.

un reticulado para factores de aproximación exponenciales en la dimensión del mismo. Para reforzar la utilidad concreta del algoritmo LLL, se presentaron los dos algoritmos de Babai (vértice más cercano e hiperplano más cercano) en la Sección 4.3.2, dado que son algoritmos relativamente sencillos de entender y que, utilizados en conjunto con LLL, generan soluciones al problema del vector más cercano, también con factores de aproximación exponenciales. Tanto LLL como los algoritmos de Babai fueron luego referenciados en los Capítulos 5 y 6 para diseñar ataques contra los sistemas GGH y NTRU respectivamente.

En la tercera parte del trabajo (Capítulos 5 a 7) se estudiaron tres sistemas de clave pública basados en reticulados junto con nociones asociadas: GGH (Capítulo 5), NTRU (Capítulo 6) y el primero basado en el problema LWE (Capítulo 7). Para cada uno de estos, junto a su descripción evaluamos el orden de ejecución asintótico de las operaciones de encriptado y desencriptado. Mostramos los resultados en la Tabla 8.1. Recordamos que tanto para GGH como NTRU, los órdenes se dan en función del parámetro n. En el caso de GGH, este parámetro se corresponde con la dimensión del reticulado utilizado (coincide con el parámetro de seguridad del sistema) y en NTRU representa el tamaño de los mensajes (cantidad de entradas necesarias para representar de forma vectorial polinomios en $R = \mathbb{Z}[x]/(x^n - 1)$). Observamos que en el último caso, el parámetro n queda determinado por el parámetro de seguridad del sistema. En el caso de LWE, m representa la cantidad de muestras/ecuaciones consideradas y n la cantidad de incógnitas del sistema (se corresponde con el parámetro de seguridad del sistema) y como vimos en la Sección 7.1, basta tomar $m \in O(n)$ de modo que el encriptado es $O(n^2)$. En cuanto al espacio necesario para almacenar las claves, tenemos los resultados mostrados en la Tabla 8.2.

Los resultados en la Tabla 8.2 refieren a la cantidad de números a ser almacenados, para obtener el orden de la cantidad de bits necesarios se debe

Sistema	Tamaño clave pública	Tamaño clave privada
GGH	$O(n^2)$	$O(n^2)$
NTRU	$O(n)$	$O(n)$
LWE	$O(nm) \in O(n^2)$	$O(n)$

Tabla 8.2: Órdenes asintóticos almacenamiento de claves.

multiplicar por el factor logarítmico correspondiente. Para indicar esto en ocasiones [7][20] se expresa el orden con $\tilde{O}$ que esconde los factores logarítmicos en la expresión. Observamos, como comentamos en el Capítulo 5, que en GGH el tamaño de las claves se debe a que es necesario almacenar la base para un reticulado genérico de dimensión n. Por otro lado, para la clave pública de NTRU como vimos en el Capítulo 6, basta con almacenar un único vector para describir la base del reticulado correspondiente. Entonces, decimos que los reticulados utilizados en NTRU tienen "más estructura" que en el caso general y de aquí la mejora en cuanto al espacio de almacenamiento. Esto también genera la pregunta de si esta estructura adicional no afecta en algún sentido la seguridad del sistema, es decir si esto no disminuye la dificultad de los problemas computacionales en los que se basa el sistema.

En cuanto al nivel de seguridad alcanzado por las propuestas presentadas, únicamente el sistema basado en LWE logra alcanzar la seguridad semántica (seguridad CPA). Como vimos en el Capítulo 2, este nivel de seguridad implica que un adversario en TPP no puede obtener información significativa sobre un mensaje observando el encriptado del mismo. Notamos que para probar la seguridad del sistema en el Capítulo 7, fue necesario basar la prueba en la dificultad computacional del problema de LWE, que a su vez está respaldada por la reducción dada por Regev en [20], que la vincula con la dificultad de resolver problemas fundamentales en reticulados en el peor caso. Este tipo de garantías teóricas de seguridad son una de las características distintivas de la criptografía basada en reticulados, dado que aseguran que los ataques sobre las construcciones son asintóticamente robustos (probablemente si son efectivos, lo serán solamente para elecciones pequeñas de los parámetros del sistema) y por tanto, que no tienen fallas fundamentales en cuanto a diseño [7].

Otra medida que podemos considerar de los sistemas es la expansión del tamaño de un mensaje que se genera al encriptarlo. Definimos la expansión de

un sistema como el cociente entre el tamaño del texto cifrado y el del mensaje. Tanto para NTRU como para LWE resulta sencillo calcular esta cantidad como sigue:

$$\text{expansión}_{NTRU} = \frac{\text{tamaño(c)}}{\text{tamaño}(m)} = \frac{n\log_2(q)bits}{n\log_2(p)bits} = \frac{\log_2(q)}{\log_2(p)} = \frac{\log_p(q)}{\log_p(2)}\frac{\log_p(2)}{\log_p(p)} = \log_p(q)$$

$$\text{expansión}_{LWE} = (n+1)\log_2(p)$$

Tanto en el caso de GGH, así como en el de NTRU, al no ser los sistemas semánticamente seguros comparar su expansión con la de LWE resulta poco representativo. En particular, para que dicha comparación tenga sentido, se debería incluir en el análisis de la expansión procedimientos de rellenado de bits bajo los que los sistemas (GGH y NTRU) logren propiedades de seguridad equivalentes o al menos similares.

De lo anterior, podemos concluir que la criptografía basada en reticulados presenta alternativas para sistemas criptográficos de clave pública que cumplen las siguientes características: (1) La comunidad científica cree que las construcciones son seguras contra ataques tanto cuánticos como clásicos; (2) Pueden implementarse de manera relativamente eficiente tanto en términos del tiempo de ejecución de las primitivas del sistema, así como del espacio de almacenamiento de claves. En particular, como resumimos en esta sección, las operaciones resultan particularmente eficientes y el tamaño de claves manejables. Esto se debe principalmente a que los sistemas resultan algebraicos y para su implementación, basta con utilizar operaciones lineales de multiplicación y suma de matrices y vectores; (3) Presentan garantías teóricas robustas en cuanto a seguridad. Además, el enfoque geométrico de los reticulados permite diversificar el tipo de construcciones criptográficas (y sus problemas subyacentes), resultando esto de interés en sí mismo, dado que ante el eventual quebrado de un tipo particular de sistemas (por ejemplo los basados en factorización o un determinado problema en teoría de números), seguirán existiendo diversas alternativas para asegurar la información.

Estos motivos son los que han hecho que este tipo de sistemas sea la categoría de propuestas predominantes en el proceso de estandarización que está llevando a cabo el Instituto Nacional de Estándares y Tecnología (NIST por sus siglas en inglés) de Estados Unidos. Dicho proyecto [8] tiene por objetivo

solicitar, evaluar y estandarizar una o más propuestas de sistemas de clave pública post cuánticos para su uso futuro, previendo el posible advenimiento de computadoras cuánticas de gran escala.

La primera ronda, finalizada en Noviembre de 2017 incluyó 45 propuestas consideradas válidas para sistemas de encriptado y 21 de estas son basadas en reticulados [41]. A su vez, dentro de las propuestas que hacen uso de reticulados, existen varias basadas en NTRU y LWE, lo que muestra la relevancia del diseño de estas construcciones.

Varias líneas de trabajo futuro surgen de este proyecto. Por un lado, existen otras alternativas relativas a criptografía post cuántica, que no tienen que ver con reticulados. En [7] se introducen las otras alternativas y sería interesante entender las propuestas principales involucradas, sus propiedades y cómo se relacionan con las que presentan las alternativas propuestas en el trabajo o más en general con las que se encuentran incluídas en el área basada en reticulados.

Otra línea posible de investigación cociste en evaluar la seguridad de los sistemas para valores concretos de los parámetros de los mismos. Este enfoque implica concentrarse en implementaciones concretas tanto en software como hardware de los sistemas en cuestión y resulta una extensión natural del estudio teórico realizado en el trabajo. En particular, porque tiene sentido evaluar la seguridad concreta de un sistema cuando se ha realizado un análisis asintótico del mismo y se tiene evidencia de que este presenta un diseño aceptable en estos términos. En cuanto a las implementaciones, de particular interés resulta la de la generación pseudoaleatoria de números, elemento que como vimos en los Capítulos 5, 6 y 7, es fundamental en criptografía.

En los Capítulos 5 y 6, comentamos que la seguridad de los sistemas GGH y NTRU podía ser mejorada utilizando un sistema de rellenado de bits. Resultaría interesante relevar las distintas propuestas existentes para realizar esta tarea y estudiar el impacto que las mismas tienen tanto en la práctica, así como a nivel del análisis teórico de la seguridad.

También existen nociones más fuertes de seguridad, como la seguridad

CCA[1] o de indistinguibilidad bajo ataque de texto cifrado elegido, que ameritan su estudio. Además de los recursos que el atacante tiene en el experimento de seguridad CPA descrito en el Capítulo 2, se le da acceso a un oráculo de desencriptado. Haciendo uso de este oráculo, puede pedir el desencriptado de cualquier texto cifrado, salvo el texto cifrado desafío. La primera construcción basada en reticulados que cumple esta característica fue dada en [42] por Peikert y Waters en 2008. En particular, una de las alternativas que proponen los autores para esta construcción está basada en el problema LWE, descrito en el Capítulo 7.

Finalmente, los reticulados han posibilitado la concepción de formas de encriptado con otra clase de propiedades que resultan de interés. En particular, se destaca el encriptado homomórfico, que permite la realización de cómputo sobre datos encriptados. Esta idea es muy interesante, dado que posibilita que las tareas de cómputo sobre datos sensibles que fueron encriptados, puedan ser realizadas por cualquier entidad (aunque esta no sea de confianza para la que posee los datos), sin que esta pueda obtener información alguna de los mismos. Un vínculo directo surge, por ejemplo, con los esquemas de cómputo "en la nube".

A nivel personal, el mayor desafío involucrado en el proyecto fue lograr comprender y brindar un enfoque unificado para los temas abordados. El hecho de que la criptografía basada en reticulados esté teniendo, desde hace unos años, un crecimiento sostenido con avances novedosos recientes, hace que sea un verdadero desafío tener una idea actualizada y comprensiva de los resultados más relevantes hasta el momento. Además, dado que el área integra contenidos de diversas disciplinas (teoría de la complejidad, álgebra lineal, probabilidad, análisis de algoritmos, criptografía, sin ser exhaustivos), es un aporte personal muy valioso poder estudiar estas materias en un mismo contexto, entendiendo sus interacciones e implicancias. Debido a su dinámica actual, con frecuencia no existe un enfoque unificado de los temas tratados y por tanto es necesario entender y adaptar los contenidos de las distintas publicaciones científicas, para expresar los conceptos que se desean transmitir de forma adecuada. Todo esto se considera de fundamental valor para la formación académica personal brindada por el proyecto.

[1] La definición formal del concepto se encuentra en el Anexo 1.

Referencias bibliográficas

[1] J. Katz and Y. Lindell, *Introduction to modern cryptography*, 2015, oCLC: 893721520.

[2] J. v. Zur Gathen, *CryptoSchool*. Heidelberg New York Dordrecht: Springer, 2015, oCLC: 918615833.

[3] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976. [Online]. Available: http://ieeexplore.ieee.org/document/1055638/

[4] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Available: http://portal.acm.org/citation.cfm?doid=359340.359342

[5] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997. [Online]. Available: http://epubs.siam.org/doi/10.1137/S0097539795293172

[6] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th ed. Cambridge ; New York: Cambridge University Press, 2010.

[7] I. W. on Post Quantum Cryptography, *Post-quantum cryptography*, D. J. Bernstein, J. Buchmann, and E. Dahmen, Eds. Berlin Heidelberg: Springer, 2009, oCLC: 551314023.

[8] I. T. L. Computer Security Division, "Post-Quantum Cryptography | CSRC," Jan. 2017. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography

[9] S. Arora and B. Barak, *Computational complexity: a modern approach*. Cambridge ; New York: Cambridge University Press, 2009, oCLC: ocn286431654.

[10] J. Hoffstein, J. C. Pipher, and J. H. Silverman, *An introduction to mathematical cryptography*, second edition ed., ser. Undergraduate texts in mathematics. New York: Springer, 2014, oCLC: ocn891676484.

[11] O. Goldreich, S. Goldwasser, and S. Halevi, "Public-key cryptosystems from lattice reduction problems," in *Advances in Cryptology — CRYPTO '97*, G. Goos, J. Hartmanis, J. van Leeuwen, and B. S. Kaliski, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, vol. 1294, pp. 112–131. [Online]. Available: http://link.springer.com/10.1007/BFb0052231

[12] D. Micciancio and S. Goldwasser, *Complexity of lattice problems: a cryptographic perspective*, softcover reprint of the hardcover 1. ed ed., ser. The Kluwer international series in engineering and computer science. New York: Springer Science + Business Media, 2002, no. SECS 671, oCLC: 935601384.

[13] "6.876j Advanced Topics in Cryptography (Fall 2015)." [Online]. Available: http://people.csail.mit.edu/vinodv/6876-Fall2015/index.html

[14] "CSE206a: Lattice Algorithms and Applications (Winter 2010)." [Online]. Available: https://cseweb.ucsd.edu/classes/wi10/cse206a/

[15] "CSE206a: Lattices Algorithms and Applications (Fall 2017)." [Online]. Available: http://cseweb.ucsd.edu/classes/fa17/cse206A-a/

[16] "Lattices in Computer Science (Fall 2009)." [Online]. Available: https://cims.nyu.edu/~regev/teaching/lattices_fall_2009/

[17] S. D. Galbraith, *Mathematics of public key cryptography*. Cambridge ; New York: Cambridge University Press, 2012.

[18] P. van Emde Boas, "Another NP-complete problem and the complexity of computing short vectors in a lattice," vol. Technical Report 81–04, Math. Inst. Univ. Amsterdam, 1981.

[19] I. Dinur, G. Kindler, R. Raz, and S. Safra, "Approximating CVP to Within Almost-Polynomial Factors is NP-Hard," *Combinatorica*, vol. 23, no. 2, pp. 205–243, Apr. 2003. [Online]. Available: http://link.springer.com/10.1007/s00493-003-0019-y

[20] O. Regev, "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography," in *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, ser. STOC '05. New York, NY, USA: ACM, 2005, pp. 84–93. [Online]. Available: http://doi.acm.org/10.1145/1060590.1060603

[21] A. K. Lenstra, H. W. Lenstra, and L. Lov\'asz, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, Dec. 1982. [Online]. Available: http://link.springer.com/10.1007/BF01457454

[22] L. Babai, "On Lovász' lattice reduction and the nearest lattice point problem," *Combinatorica*, vol. 6, no. 1, pp. 1–13, Mar. 1986. [Online]. Available: http://link.springer.com/10.1007/BF02579403

[23] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*. Philadelphia, Pennsylvania, United States: ACM Press, 1996, pp. 99–108. [Online]. Available: http://portal.acm.org/citation.cfm?doid=237814.237838

[24] M. Ajtai and C. Dwork, *A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence (Extended Abstract)*, 1996.

[25] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring-based public key cryptosystem," in *Algorithmic Number Theory*, G. Goos, J. Hartmanis, J. van Leeuwen, and J. P. Buhler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, vol. 1423, pp. 267–288. [Online]. Available: http://link.springer.com/10.1007/BFb0054868

[26] R. Vershynin, *High-dimensional probability: an introduction with applications in data science*, ser. Cambridge series in statistical and probabilistic mathematics. Cambridge: Cambridge University Press, 2018, no. 47.

[27] G. Goos, J. Hartmanis, J. van Leeuwen, and P. Nguyen, "Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto '97," in *Advances in Cryptology — CRYPTO' 99*, M. Wiener, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, vol. 1666, pp. 288–304. [Online]. Available: http://link.springer.com/10.1007/3-540-48405-1_18

[28] J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. H. Silverman, and W. Whyte, "NTRUSign: Digital Signatures Using the NTRU Lattice," in *Topics in Cryptology — CT-RSA 2003*, G. Goos, J. Hartmanis, J. van Leeuwen, and M. Joye, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, vol. 2612, pp. 122–140. [Online]. Available: http://link.springer.com/10.1007/3-540-36563-X_9

[29] J. H. Joseph H. Silverman, "Implementation Notes for NTRU PKCS Multiple Transmissions," NTRU Cryptosystems, Inc., Reporte técnico 6, May 1998.

[30] D. Stehlé and R. Steinfeld, "Making NTRU as Secure as Worst-Case Problems over Ideal Lattices," in *Advances in Cryptology – EUROCRYPT 2011*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and K. G. Paterson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6632, pp. 27–47. [Online]. Available: http://link.springer.com/10.1007/978-3-642-20465-4_4

[31] B. Barak, *An Intensive Introduction To Cryptography*, Apr. 2018. [Online]. Available: https://intensecrypto.org/public/lnotes_book.pdf

[32] C. Peikert, "A Decade of Lattice Cryptography," *Found. Trends Theor. Comput. Sci.*, vol. 10, no. 4, pp. 283–424, Mar. 2016. [Online]. Available: http://dx.doi.org/10.1561/0400000074

[33] O. Regev, "The Learning with Errors Problem (Invited Survey)," in *2010 IEEE 25th Annual Conference on Computational Complexity*, Jun. 2010, pp. 191–204.

[34] C. Peikert, "Public-key Cryptosystems from the Worst-case Shortest Vector Problem: Extended Abstract," in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, ser. STOC '09. New York, NY, USA: ACM, 2009, pp. 333–342. [Online]. Available: http://doi.acm.org/10.1145/1536414.1536461

[35] O. Regev, "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography," *J. ACM*, vol. 56, no. 6, pp. 34:1–34:40, Sep. 2009. [Online]. Available: http://doi.acm.org/10.1145/1568318.1568324

[36] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, Jan. 2005, google-Books-ID: 0bAYl6d7hvkC.

[37] O. Goldreich, *Computational Complexity: a Conceptual Perspective*. Leiden: Cambridge University Press, 2008, oCLC: 437209191. [Online]. Available: http://public.eblib.com/choice/publicfullrecord.aspx?p=343545

[38] A. S. Rafael Pass, *A Course In Cryptography*, Cornell University, Jan. 2010.

[39] S. Goldwasser and S. Micali, "Probabilistic encryption & how to play mental poker keeping secret all partial information," in *Proceedings of the fourteenth annual ACM symposium on Theory of computing - STOC '82*. San Francisco, California, United States: ACM Press, 1982, pp. 365–377. [Online]. Available: http://portal.acm.org/citation.cfm?doid=800070.802212

[40] "CSCI-GA.3210-001/MATH-GA.2170-001: Introduction To Cryptography." [Online]. Available: https://cs.nyu.edu/courses/spring12/CSCI-GA.3210-001/

[41] C. S. D. Information Technology Laboratory, "Let's Get Ready to Rumble -The NIST PQC 'Competition'," Apr. 2018. [Online]. Available: https://csrc.nist.gov/Presentations/2018/Let-s-Get-Ready-to-Rumble-The-NIST-PQC-Competiti

[42] C. Peikert and B. Waters, "Lossy Trapdoor Functions and Their Applications," in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, ser. STOC '08. New York, NY, USA: ACM, 2008, pp. 187–196. [Online]. Available: http://doi.acm.org/10.1145/1374376.1374406

ANEXOS

Anexo 1

Nociones más fuertes de seguridad

Presentamos una noción más fuerte de seguridad, donde el adversario puede obtener el desencriptado de textos cifrados arbitrarios (distintos del que recibe como desafío). Esta noción recibe el nombre de seguridad contra ataques de texto cifrado elegido (seguridad-CCA). [1] Comenzamos con el experimento correspondiente como en el caso anterior,

Definición 75. *Experimento de indistinguibilidad CCA,* $PubK^{cca}_{\mathcal{A},\Pi}(n)$

1. *Se ejecuta* $\mathtt{Gen}(1^n)$ *para obtener* (pk, sk).

2. *Se le otorga la clave pública pk al adversario* $\mathcal{A}$ *y también acceso a un oráculo de desencriptado* $\mathtt{Dec}_{sk}(\cdot)$. *Entonces* $\mathcal{A}$ *da como salida un par de mensajes* m_0,m_1 *del mismo largo, incluídos en el espacio de mensajes asociados a pk.*

3. *Se elige un bit de manera uniforme* $b \in \{0, 1\}$ *y se computa el texto cifrado desafío* $c \leftarrow \mathtt{Enc}_{pk}(m_b)$ *y este es dado a* $\mathcal{A}$.

4. $\mathcal{A}$ *puede continuar interactuando con el oráculo de desencriptado, sin pedir el desencriptado de c. Finalmente,* $\mathcal{A}$ *da un bit* b' *como salida.*

5. *La salida del experimento se define como 1 si* $b' = b$ *y 0 en caso contrario.*

Nuevamente en base a esto tenemos,

Definición 76. *Un sistema de clave pública* $\Pi = (\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *tiene indistinguibilidad de encriptado bajo ataque de texto cifrado elegido (o es CCA-seguro) si para todo adversario* $\mathcal{A}$ *en TPP, existe una función negligible* $\texttt{negl}$ *tal que*

$$PrPubK^{cca}_{\mathcal{A},\Pi}(n) = 1) \leq \frac{1}{2} + \texttt{negl}(n)$$

Entonces de las definiciones resulta que si un sistema es CCA-seguro también debe ser CPA-seguro (sección 2.2.2), ya que en el primer caso el adversario tiene más recursos que en el segundo y si uno logra asegurar los ataques contra el primer tipo también lo hará contra uno del segundo.