Conteúdo

1 Introdução

O presente trabalho foi elaborado no âmbito da unidade curricular de Criptografia, leccionada no 4º Ano do Mestrado Integrado em Engenharia Biomédica — Ramo de Informática Médica, e propõe-se explorar a temática dos problemas *hard* em reticulados, bem como as técnicas criptográficas cuja construção e/ou prova de segurança se baseiam nestes últimos.

1.1 Criptografia de Chave Pública

Numa fase de desenvolvimento inicial, os sistemas criptográficos baseavam-se no conceito de simetria, que partia da utilização da mesma chave por parte do emissor e do receptor da mensagem, o que, pese embora permitisse providenciar segurança da informação a nível teórico, apresentava a séria desvantagem de requerer elevados níveis de aleatoriedade e chaves de tamanho consideravelmente elevado. De forma a contrariar esta premissa, em 1976, Diffie e Hellman desenvolveram um criptosistema que introduzia a noção de funções *trapdoor*, consideradas a base da criptografia assimétrica. Uma função *trapdoor* consiste numa função unidireccional — isto é, uma função f que permite calcular $f(x)$ partindo de x de forma relativamente fácil, mas que dificulta o cálculo de x a partir de $f(x)$ — com a particularidade de acrescentar que existe uma dada informação secreta y que pode ser utilizada para fazer com que o cálculo de x a partir de $f(x)$ seja fácil. Neste sentido, Diffie e Hellman propuseram a Criptografia de Chave Pública, através da qual pretendiam que se criasse um par de chaves — uma pública e uma privada — matematicamente relacionadas. Uma das possíveis utilizações seria a de o emissor de uma mensagem poder utilizar a chave pública do receptor para cifrar uma mensagem, que seria depois decifrada recorrendo à chave privada. A chave pública poderia ser acedida por qualquer utilizador sem que isso comprometesse a segurança da respectiva chave privada, na medida em que derivar a chave privada partindo deste conhecimento seria tão difícil quanto inverter uma função unidireccional. Face à criptografia simétrica, este conceito apresenta uma vantagem inegável, que é a de requerer a criação de um par simétrico único de chaves e a distribuição do mesmo por cada par de utilizadores que desejem comunicar de forma privada. Contudo, uma das desvantagens expostas pelos próprios autores prende-se com o facto de, estando as chaves matematicamente relacionadas, nunca seria possível atingir a segurança teórica de alguns dos sistemas simétricos, dado que um adversário com poder computacional ilimitado conseguiria sempre desvendar a informação privada recorrendo a procuras exaustivas. Assim, os criptosistemas de chave pública almejam não a segurança teórica, mas a segurança de uma perspectiva essencialmente prática, tomando em consideração os limites dos poderes computacionais de um potencial adversário — a designada segurança computacional. Esta premissa assume que quebrar um dado crip-

tosistema deve ser, não impossível, mas antes computacionalmente impraticável, motivo pelo qual se adoptou a prática de basear a segurança de um criptosistema na assumpção de que alguns problemas matemáticos são difíceis de resolver recorrendo à computação. Interessa, neste ponto, mencionar uma variável que se torna essencial nesta conjuntura: o parâmetro de segurança, que determina o tamanho do *input* e é, normalmente, escolhido de forma a que exista um balanço entre a segurança e a eficiência do sistema. A segurança *per se* é normalmente medida em *bits* e, para um segurança de k *bits*, o esforço temporal e logístico que o adversário tem de desenvolver para quebrar o sistema é equivalente a testar todas as chaves possíveis de tamanho k, ou seja, 2^k chaves distintas. De forma a garantir que todos os métodos passíveis de quebrar o sistema são impraticáveis, deve atender-se à complexidade computacional dos problemas que lhe são inerentes [1, 2].

1.2 Teoria da Complexidade Computacional

A Teoria da Complexidade Computacional estende um dos ramos da Teoria da Computação, focando-se especificamente na classificação dos problemas computacionais atendendo à sua dificuldade e no relacionamento entre as classes assim definidas, com o intuito de determinar os limites práticos de operação e funcionalidade dos computadores, entre outros. Considera-se, neste contexto, que um problema é difícil se a solução que dele decorre requer um gasto significativo de recursos — como sejam tempo e capacidade de armazenamento —, quaisquer que sejam as abordagens algorítmicas utilizadas.

Numa primeira fase, interessa explorar a complexidade de uma perspectiva temporal. Neste sentido, torna-se possível distinguir três tipos de complexidade distintos, que se reportam à rapidez (por assim dizer) com que determinados *inputs* de tamanho n são resolvidos: *best-case*, *worst-case* e *average-case*. Os dois primeiros encontram-se associados à solução de um problema para o melhor e o pior *input* de tamanho n, respectivamente, sendo que o último se reporta à complexidade de resolver o problema numa média, tendo em consideração uma distribuição de probabilidades dos *inputs*. No âmbito da computação em tempo real, a vertente *worst-case* constitui um foco de especial atenção, dado que é relevante ter acesso ao tempo necessário para que, no pior dos casos, um algoritmo termine a sua execução atempadamente. Em [3], Ajtai demonstrou que é possível estabelecer uma conexão entre o *worst-case* e o *average-case* entre determinados problemas de reticulados, o que gerou um enorme interesse em volta da utilização destas estruturas como base para criptosistemas, na medida em que a dificuldade *average-case* representa uma propriedade desejável para estes sistemas.

Aos conjuntos de problemas cujas complexidades se encontram relacionadas dá-se o nome

de classes de complexidade. Estas podem ser definidas, por exemplo, através de critérios que constringem o tempo e espaço consumidos por um algoritmo. Interessa, no contexto específico deste trabalho, destacar duas classes em particular:

1.2.1 Classe P

A classe P — de Polinomial — diz respeito a problemas passíveis de serem resolvidos por Deterministic Turing Machines (DTMs) que obedeçam, em termos de tempo de execução, a um limite polinomial, e é, teoricamente, considerada equivalente à noção de 'resolução eficiente', não obstante a existência de excepções [4].

1.2.2 Classe NP

Por sua vez, a classe NP — de Polinomial Não-Determinístico — é definida como o cojunto de todos os problemas resolúveis por intermédio de Nondeterministic Turing Machines (NDTMs) em tempo polinomial. Será, portanto, lógico inferir que a classe P está contida na NP, pese embora esta encerre outros problemas, de entre os quais se destacam os NP-Complete — para os quais não são conhecidas soluções que passem por algoritmos de tempo polinomial [4]. Porém, contrariamente à classe P, os problemas NP não podem ser eficientemente resolvidos, podendo antes ser apenas 'eficientemente verificados' [5].

Uma questão importante na Teoria da Complexidade (e que se encontra, ainda, aberta) prende-se com o facto de não se saber se as classes P e NP são efectivamente coincidentes ou se existem problemas que são exclusivos a NP. Até prova em contrário, assume-se que P $\neq$ NP e considera-se, informalmente, que P é a classe dos problemas 'fáceis', enquanto que NP contém problemas '*hard*' — os já referidos NP-Complete — que, com grande probabilidade, não estão contidos em P. Caso se consiga provar que, matematicamente, quebrar um criptosistema é equivalente à resolução de um problema NP-Complete, é possível inferir uma medida razoável da segurança dessa construção e extrapolar uma prova de segurança.

1.3 Reticulados

Os reticulados são, tipicamente, definidos como um sub-grupo discreto de $\mathbb{R}^n$, o espaço vectorial Euclidiano n-dimensional, sendo que por 'discreto' se pretende implicar que estes não compreendem pontos de acumulação. Isto é, considerando D como um sub-espaço de $\mathbb{R}^n$, para qualquer $\mathbf{x}$ pertencente a D, existe um $r > 0$ tal que a esfera definida como $\mathcal{B}(\mathbf{x}, r) = \{\mathbf{y} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{y}\| < r\}$ satisfaz $\mathcal{B}(\mathbf{x}, r) \cap D = \{\mathbf{x}\}$. Considere-se, ainda, que qualquer sub-conjunto de um conjunto discreto é igualmente discreto.

Definição 1. *Um sub-grupo sob a adição de $(\mathbb{R}^n, +)$ que apresente a propriedade discreta é denominado de reticulado. Com base nesta propriedade, qualquer sub-grupo de um reticulado é, ele mesmo, um reticulado.*

Os reticulados podem ainda ser definidos como sub-conjuntos não vazios $\mathcal{L} \subset \mathbb{R}^n$, fechados a operações de subtracção, de tal forma que exista um $r > 0$ para o qual $\mathcal{B}(0, r) \cap \mathcal{L} = \{0\}$. Neste sentido, as propriedades fundamentais dos reticulados podem ser definidas como:

1. Existe um $r > 0$ tal que, para todo o $\mathbf{x} \in \mathcal{L}$, não existe qualquer outro elemento $\mathbf{y} \in \mathcal{L}$ com $\|\mathbf{x} - \mathbf{y}\| < r$. De forma equivalente, existe um $r > 0$ tal que $\mathcal{B}(\mathbf{x}, r) \cap \mathcal{L} = \{\mathbf{x}\}$ para todo o $\mathbf{x} \in \mathcal{L}$;

2. $\mathcal{L}$ é um conjunto fechado, o que implica necessariamente que não possui pontos de acumulação;

3. Caso $S \subseteq \mathbb{R}^n$ seja limitado, então $S \cap \mathcal{L}$ é finito;

4. $\mathcal{L}$ é contável, isto é, apresenta a mesma cardinalidade de um qualquer sub-conjunto pertencente a $\mathbb{N}$.

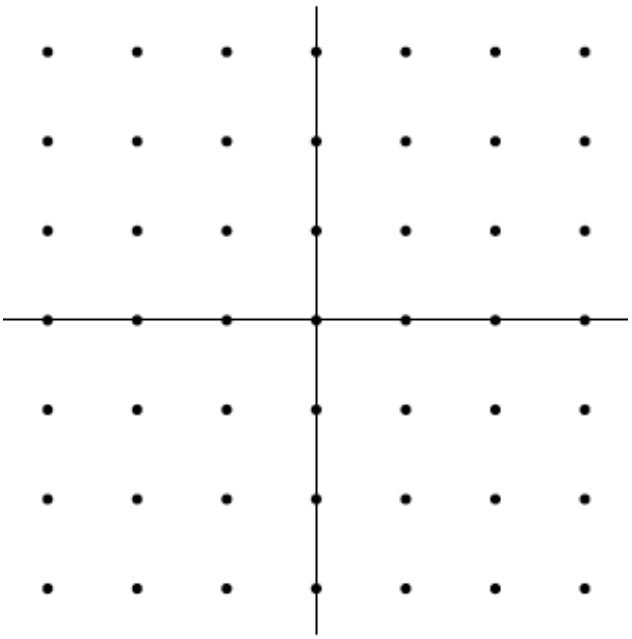

Figura 1: O reticulado $\mathbb{Z}^n$, formado por todos os vectores em $\mathbb{R}^2$ com coeficientes inteiros.

Em espaços vectoriais, um sub-espaço pode ser descrito através da utilização de bases. Considerando m vectores $\mathbf{b}_1, ..., \mathbf{b}_m \in \mathbb{R}^n$ e um conjunto de combinações lineares

$L(\mathbf{b}_1, ..., \mathbf{b}_m)$, estabelece-se que este apenas é discreto (e, por consequência, um reticulado), caso $\mathbf{b}_1, ..., \mathbf{b}_m \in \mathbb{Q}$ ou caso $\mathbf{b}_1, ..., \mathbf{b}_m \in \mathbb{R}^n$ sejam linearmente independentes. O facto de um conjunto finito de vectores poder descrever um reticulado permite postular a seguinte definição:

Definição 2. *Um reticulado $\mathcal{L}$ é dito abrangido pelos geradores $\mathbf{b}_1, ..., \mathbf{b}_m$ se $\mathcal{L} = L(\mathbf{b}_1, ..., \mathbf{b}_m$. Se os $\mathbf{b}_i$ forem também linearmente independentes, então $\mathbf{b}_1, ..., \mathbf{b}_m$ é considerada uma base de $\mathcal{L}$ e, para todo o $\mathbf{x} \in \mathcal{L}$ existem coordenadas $(x_1, ..., x_m)$ inteiras e únicas que verificam:*

$$\mathbf{x} = \sum_{i=1}^{m} x_i \mathbf{b}_i$$

Torna-se ainda relevante realçar a definição de *rank* de um reticulado, na medida em que os sub-espaços lineares de $\mathbb{R}^n$ possuem uma dimensão que limita o número máximo de vectores linearmente independentes e denota o número de vectores numa base desses mesmos espaços.

Definição 3. *Seja $\mathcal{L} \subset \mathbb{R}^n$ um reticulado. O rank d de $\mathcal{L}$ é definido como sendo a dimensão do sub-espaço span($\mathcal{L}$), que representa o sub-espaço mínimo contendo $\mathcal{L}$. Caso $n = d$, o reticulado é denominado de full-rank.*

De forma semelhante ao que sucede nos sub-espaços, o *rank* equivale ao número máximo de vectores linearmente independentes do reticulado, bem como ao número de vectores que compõem a sua base. Porém, neste contexto específico, nem todos os conjuntos de vectores linearmente independentes se qualificam para formar uma base, já que uma dada matriz $\mathcal{B}$, composta por d vectores linearmente independentes $(\mathbf{b_1}, ..., \mathbf{b_m})$, constitui uma base de um reticulado $\mathcal{L}$ de *rank* d sse não existem vectores não nulos $\mathbf{x} \in \mathcal{L}$ tais que $\mathbf{x}$ pertença ao paralelepípedo $\mathcal{P}(\mathbf{b_1}, ..., \mathbf{b_m}) = \{\sum_{i=1}^{n} \lambda_1 \mathbf{b_i} : 0 \leqslant \lambda_i < 1\}$. Considerando que todo o reticulado $\mathcal{L}$ de *rank* d compreende, pelo menos, d vectores linearmente independentes, torna-se possível afirmar que qualquer reticulado $\mathcal{L} \subset \mathbb{R}^n$ tem, no mínimo, uma base, podendo ser escrito como as combinações lineares inteiras de d vectores de base.

2 Problemas Hard em Reticulados

Os problemas *hard* baseados em reticulados constituem, conforme já analisado, uma esfera que gira em torno da noção de que não existe uma solução computacional que os resolva de forma eficiente (nem mesmo a computação quântica), o que os torna necessariamente apelativos para aplicações critpográficas, nomeadamente quando comparados com os problemas baseados na Teoria dos Números.

2.1 Conceitos Básicos

No sentido de compreender a dimensão do problema dos vectores curtos, especificamente, torna-se necessário abordar, previamente, duas temáticas que constituem a sua base: os Mínimos Sucessivos de Minkowski e a Constante de Hermite.

2.1.1 Mínimos Sucessivos de Minkowski

Pela sua propriedade discreta, explorada na Secção 1, um reticulado $\mathcal{L}$ de *rank* pelo menos 1 compreende um vector não nulo que se encontra próximo da origem e cuja norma representa a distância mais curta possível entre quaiquer dois vectores distintos do reticulado — o Primeiro Mínimo Sucessivo $\lambda_1(\mathcal{L})$. A Figura 1 ilustra o conceito para um dado reticulado $\mathcal{L}(a, b)$, sendo o vector $\mathbf{x}$ o Primeiro Mínimo Sucessivo respeitante à norma Euclidiana [6, 7, 8].

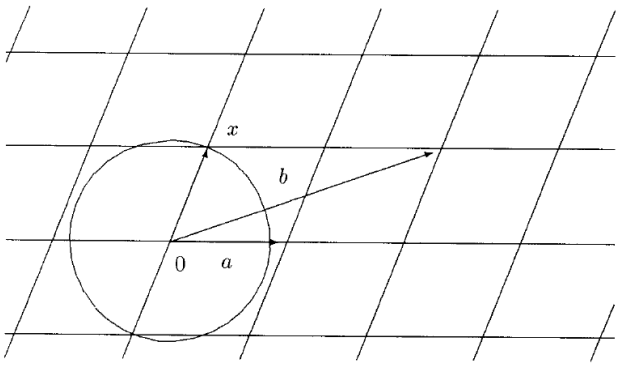

Figura 2: Primeiro mínimo sucessivo $\lambda_1\mathcal{L}(a, b)$.

A aplicação do conceito de Minkowski a outros mínimos sucessivos pode ser generalizada da seguinte forma:

Definição 4. *Para i = 1,...,d, o mínimo sucessivo de ordem i do reticulado $\mathcal{L}$ de rank d é*

$$\lambda_i(\mathcal{L}) = min\{max\|x_1\|,...,\|x_i\|\},\ |x_1, ...x_i| \in \mathcal{L}\ \textit{linearmente independentes.}$$

No seguimento destas assumpções, poder-se-ia assumir que um reticulado deveria ter uma base cujas normas correspondessem precisamente aos mínimos sucessivos. Contudo, e pese embora existam sempre vectores $\mathbf{v_1}, ..., \mathbf{v_d}$ linearmente independentes do reticulado que atingem simultaneamente os mínimos — isto é, $\|\mathbf{v_i}\|=\lambda_i\mathcal{L}$ para todo o i —, assim que $dim(\mathcal{L}) \geqslant 4$, tais vectores não formam necessariamente uma base do reticulado [7]. A título de exemplo, considere-se o reticulado $\mathcal{L}$ definido como o conjunto de todos os $(x_1, x_2, x_3, x_4) \in \mathbb{Z}^4$ de tal forma que $\sum_{i=1}^{4} x_i$ é par. Uma possível base de $\mathcal{L}$ pode ser representada pela matriz $\mathcal{A}$. A matriz $\mathcal{B}$, por outro lado, não cosntitui uma base adequada: não obstante ser composta por vectores linha linearmente independentes que atingem, da mesma forma, todos os mínimos, o seu determinante é igual a 4.

$$\mathcal{A} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} \qquad \mathcal{B} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

Este conceito pode ser reforçado através do caso demonstrativo de multiplicação das duas matrizes, já que existem provas de que duas bases originam o mesmo reticulado se (e só se) as suas matrizes estiverem relacionadas por uma matriz unimodular [9].

$$\mathcal{A}\times\mathcal{B} = \begin{pmatrix} 0 & -2 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}$$

2.1.2 Constante de Hermite

A prova de que a quantidade $\lambda_1(\mathcal{L})/\mathrm{vol}(\mathcal{L})^{1/d}$ pode ser limitada, para todo o reticulado $\mathcal{L}$ de *rank d* foi dada, pela primeira vez, por Hermite, e dita que:

Definição 5. *O supremo de $\lambda_1(\mathcal{L})^2/vol(\mathcal{L})^{2/d}$ para todo o reticulado $\mathcal{L}$ de rank d é representado por γ_d e designado por Constante de Hermite de dimensão d.*

Embora seja assente que γ_d é atingido, o processo de cálculo do seu valor exacto constitui um problema árduo, sendo que este é apenas conhecido para os casos em que $1 \leqslant d \leqslant 8$ e $d = 24$.

Para todas as dimensões $d \geqslant 1$, existe um reticulado $\mathcal{L}$ de *rank d* que verifica os valores da Constante de Hermite explicitados na Tabela 1 — denominado de reticulado crítico. Da mesma forma, todos os reticulados críticos para cada uma das dimensões mencionadas são

Tabela 1: Valores actualmente conhecidos para a Constante de Hermite [7]

d	2	3	4	5	6	7	8	24
γ_d	$2/\sqrt{3}$	$2^{1/3}$	$\sqrt{2}$	$8^{1/5}$	$(64/3)^{1/6}$	$64^{1/7}$	2	4
Aproximação	1.1547	1.2599	1.4142	1.5157	1.6654	1.8114	2	4

conhecidos [7]. As referidas dimensões tornam-se, porém, irrelevantes no caso específico dos problemas *hard*, na medida em que, no âmbito desta problemática, as dimensões de interesse situam-se na ordem das centenas. Poderá referir-se, neste contexto e a título de demonstração da magnitude das grandezas envolvidas nos problemas *hard* em reticulados, um *challenge* actualmente vigente na página da Universidade Técnica de Darmstadt, no qual os participantes são desafiados a encontrar um vector curto em reticulados de dimensões $500 \leqslant d \leqslant 2000$. Até à data, o melhor resultado corresponde à dimensão $d = 825$ [10].

2.2 Vectores Curtos

Nesta subsecção, serão introduzidos os problemas relacionados com vectores curtos, nomeadamente o Shortest Vector Problem, o Shortest Independent Vector Problem e as respectivas variantes aproximativas. As noções aqui abordadas servirão de base à derivação de problemas adicionais (Subsecção 2.4) e ao suporte da prova de segurança do Criptosistema de Ajtai-Dwork (Subsecção 4.1.1).

2.2.1 Shortest Vector Problem

Com base na definição de mínimo postulada em 2.1.1 — que pode ser generalizada como a distância mínima entre dois pontos de um reticulado —, torna-se possível definir a primeira problemática no âmbito dos problemas *hard*: a de encontrar um vector não nulo que atinja este valor — Shortest Vector Problem (SVP) (Figura 3) [11, 12]. Releve-se que o vector mais curto de um reticulado não é único, dado que, para além da possibilidade de existirem outros vectores de norma semelhante, há que considerar que $\|\mathcal{B}\mathbf{x}\| = \|-\mathcal{B}\mathbf{x}\|$ [8]. O número de vectores curtos compreendido num reticulado $\mathcal{L}$ denomina-se de *kissing number* e é superiormente delimitado [7].

Definição 6. *Dado um reticulado $\mathcal{L}(\mathcal{B})$, encontrar um vector não nulo $\mathcal{B}\mathbf{x}$, $\mathbf{x} \in \mathbb{Z}^k$, de comprimento (no máximo) $\|\mathcal{B}\mathbf{x}\| \leqslant \lambda_1$.*

No que concerne a complexidade, o SVP foi inicialmente comprovado como sendo NP-*hard* na norma l_∞ por van Emde Boas, sendo posteriormente definido como NP-*hard* na

norma l_p, para $p < \infty$ por Ajtai [13], mais precisamente na norma l_2, na qual o SVP é normalmente definido na grande maioria das aplicações.

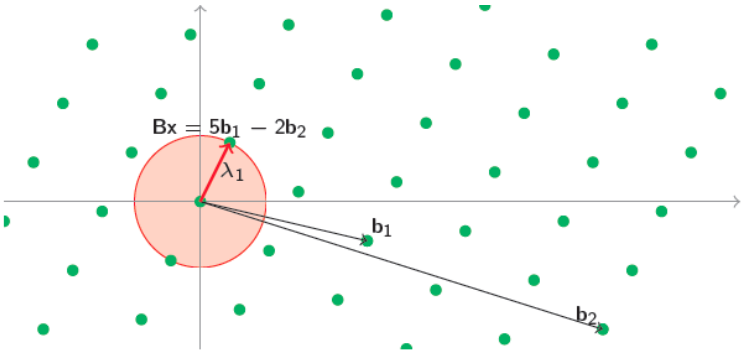

Figura 3: Representação gráfica do Shortest Vector Problem (SVP).

2.2.2 Approximate Shortest Vector Problem

Por vezes, as aplicações não requerem um vector que seja efectivamente 'o mais curto', sendo suficiente encontrar um vector que seja 'suficientemente curto' — Approximate Shortest Vector Problem (SVPγ) (Figura 4) [11, 12].

Definição 7. *Dado um reticulado $\mathcal{L}(\mathcal{B})$ e um factor de aproximação $\gamma \geqslant 1$, encontrar um vector não nulo $\mathcal{B}\mathbf{x}$, $\mathbf{x} \in \mathbb{Z}^k$, de comprimento (no máximo) $\|\mathcal{B}\mathbf{x}\| \leqslant \gamma\lambda_1$.*

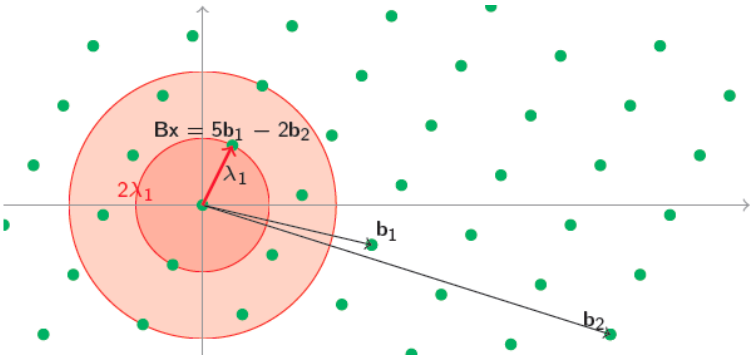

Figura 4: Representação gráfica do Approximate Shortest Vector Problem (SVPγ).

2.2.3 Shortest Independent Vector Problem

No caso particular em que se torna necessário encontrar não um, mas um conjunto de n vectores linearmente independentes — isto é, uma base de $\mathcal{L}$ — o problema denomina-se de Shortest Independent Vector Problem (SIVP) (Figura 5) [11, 14].

Definição 8. *Dado um reticulado $\mathcal{L}(\mathcal{B})$, encontrar um conjunto de n vectores linearmente independentes $\mathcal{B}\mathbf{x}_1, ...\mathcal{B}\mathbf{x}_n$ de comprimento (no máximo) $max_i\|\mathcal{B}\mathbf{x}_i\| \leqslant \lambda_n$.*

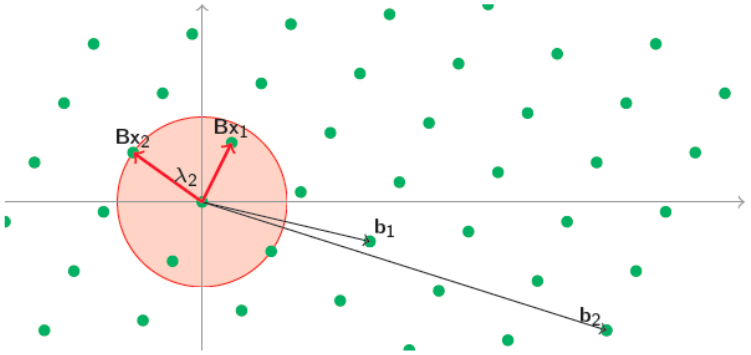

Figura 5: Representação gráfica do Shortest Independent Vector Problem (SIVP).

2.2.4 Approximate Shortest Independent Vector Problem

De forma análoga à variante do SVP exposta em 2.2.2, também o Shortest Independent Vector Problem (SIVPγ) está associado a um problema que visa encontrar, não o conjunto de vectores mais curtos, mas um conjunto de vectores 'suficientemente curtos' (Figura 6) [11, 14].

Definição 9. *Dado um reticulado $\mathcal{L}(\mathcal{B})$, encontrar um conjunto de n vectores linearmente independentes $\mathcal{B}\mathbf{x}_1, ...\mathcal{B}\mathbf{x}_n$ de comprimento (no máximo) $max_i\|\mathcal{B}\mathbf{x}_i\| \leqslant \gamma\lambda_n$.*

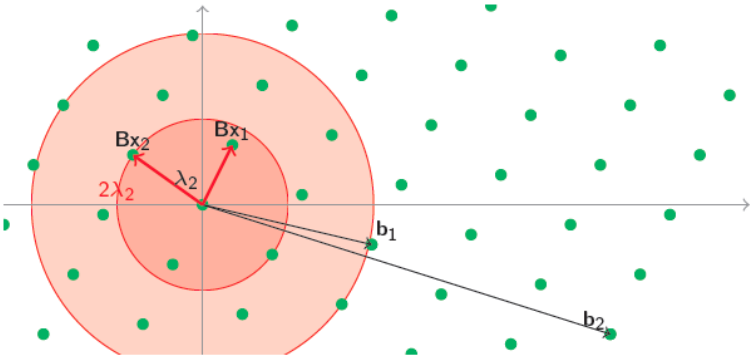

Figura 6: Representação gráfica do Approximate Shortest Independent Vector Problem (SIVPγ).

2.3 Vectores Próximos

Apresentam-se, nesta subsecção, os problemas clássicos relacionados com vectores próximos, nomeadamente o Closest Vector Problem e o Aproximate Closest Vector Problem, o primeiro dos quais será recuperado em 4.1.2, a propósito do Criptosistema de Goldreich-Goldwasser-Halevi.

2.3.1 Closest Vector Problem

Um outro problema *hard* que surge no âmbito dos reticulados diz respeito à noção de vector mais próximo: dado um determinado ponto-alvo $\mathbf{t} \in \mathbb{R}^d$ — não necessariamente pertencente ao reticulado $\mathcal{L}(\mathcal{B})$ —, o vector $\mathcal{B}\mathbf{x}$ mais próximo corresponde ao ponto de $\mathcal{L}(\mathcal{B})$ que minimiza a distância entre os dois (Figura 7) [12, 11, 14].

Definição 10. *Dados um reticulado $\mathcal{L}(\mathcal{B})$ e um ponto-alvo $\mathbf{t}$, encontrar um ponto $\mathbf{x}$ do reticulado que minimize a distância $d(\mathbf{t} - \mathbf{x}) = \|\mathbf{t} - \mathbf{x}\|$, isto é, o vector $\mathcal{B}\mathbf{x}$ à distância $\|\mathcal{B}\mathbf{x} - \mathbf{t}\| \leqslant \mu$ de $\mathbf{t}$.*

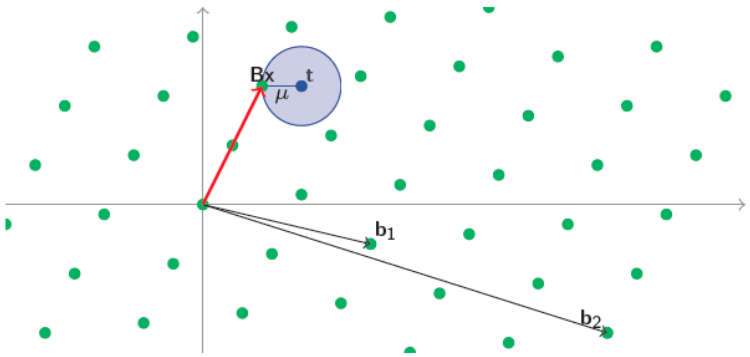

Figura 7: Representação gráfica do Closest Vector Problem (CVP).

De forma semelhante ao SVP, van Emde Boas provou que resolver o problema CVP — denominado, no seu artigo, de 'Nearest Vector Problem' — de forma exacta é também NP-*hard*.

2.3.2 Approximate Closest Vector Problem

Uma relaxação possível para o problema anterior consiste, igualmente, em considerar um factor de aproximação γ que permita obter, não o vector mais próximo, mas um vector 'suficientemente próximo' (Figura 8) [12, 11, 14].

Definição 11. *Dados um reticulado* $\mathcal{L}(\mathcal{B})$ *e um ponto-alvo* $\mathbf{t}$*, encontrar um ponto* $\mathbf{x}$ *do reticulado que minimize a distância* $d(\mathbf{t} - \mathbf{x}) = \|\mathbf{t} - \mathbf{x}\|$*, isto é, o vector* $\mathcal{B}\mathbf{x}$ *à distância* $\|\mathcal{B}\mathbf{x} - \mathbf{t}\| \leqslant \gamma\mu$ *de* $\mathbf{t}$*.*

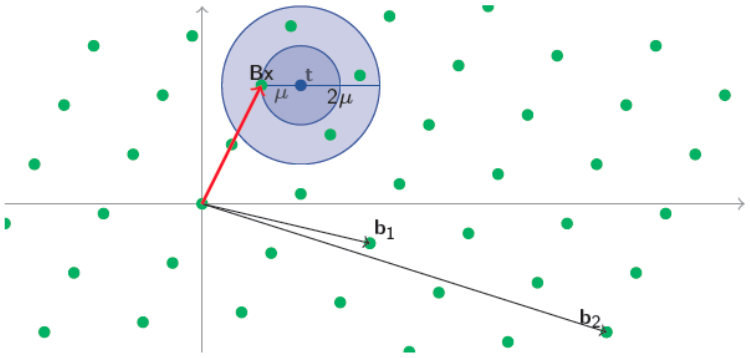

Figura 8: Representação gráfica do Approximate Closest Vector Problem (CVPγ).

2.4 Problemas Adicionais

No contexto das reduções *worst-case* a *average-case*, Ajtai introduziu três problemas de reticulados inerentes à resolução do SVP numa dada classe de reticulados aleatórios — os chamados reticulados modulares ou *q-ary*, que satisfazem $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$ para um dado inteiro q. $\mathcal{L}$ é apenas dito modular nos casos específicos em que q é um múltiplo do volume do reticulado, e assume a seguinte propriedade: se $\mathbf{x} \equiv \mathbf{y} \bmod q$, então $\mathbf{x} \in \mathcal{L}$ sse $\mathbf{y} \in \mathcal{L}$ [3].

Recentemente, surgiu um conjunto de problemas que têm por base aqueles definidos no trabalho de Ajtai — enumerados de seguida —, nomeadamente o Small Integer Solutions, o já explicitado Shortest Independent Vector Problem e o Learning With Errors (que será propositadamente ignorado, no decorrer deste trabalho, por existir um outro trabalho exclusivamente dedicado aos problemas de aprendizagem).

2.4.1 Hermite Shortest Vector Problem

Pese embora não faça parte do conjunto de problemas definidos em [3], o Hermite Shortest Vector Problem destaca-se por constituir uma alternativa ao SVP e é mencionado como ponto motivador da questão colocada no Shortest Length Problem, desenvolvido de seguida. Na sequência do SVP, não existe qualquer certeza (ou forma de comprovar) se um dado vector é efectivamente o vector mais curto pertencente ao reticulado, pelo que o HSVP surge no sentido de colmatar esta lacuna aparente, ao propor-se encontrar não o vector mais curto relativamente ao vector não nulo mais curto, mas relativamente ao volume deste.

Definição 12. *Dada uma base $\mathcal{B}$ de um reticulado $\mathcal{L}$ de rank d que verifique $\mathcal{L} \subseteq \mathbb{Z}^n$ e um factor de aproximação $\gamma > 0$, o Hermite Shortest Vector Problem consiste em encontrar um vector $\mathbf{v}$ não nulo do reticulado tal que a norma de $\mathbf{v}$ satisfaça $\|\mathbf{v}\| \leqslant \gamma(vol(\mathcal{L}))^{1/d}$.*

2.4.2 Shortest Length Problem

Da mesma forma que verificar se um vector não nulo é efectivamente o vector mais curto não constitui um processo trivial, o de determinar o primeiro mínimo sucessivo $\lambda_1(\mathcal{L})$ revela-se igualmente complicado. Nestes termos, é possível definir o Shortest Length Problem:

Definição 13. *Dada uma base $\mathcal{B}$ de um reticulado $\mathcal{L}$ de rank d que verifique $\mathcal{L} \subseteq \mathbb{Z}^n$ e um factor de aproximação $\gamma > 0$, encontrar um valor $\lambda(\mathcal{L})$ tal que $\lambda_1(\mathcal{L}) \leqslant \lambda(\mathcal{L}) \leqslant \gamma\lambda_1(\mathcal{L})$.*

2.4.3 Unique Shortest Vector Problem

Este problema constitui um caso especial do SVP, ao impor determinadas restrições ao reticulado $\mathcal{L}$.

Definição 14. *Dada uma base $\mathcal{B}$ de um reticulado fullrank $\mathcal{L} \subset \mathbb{Z}^n$ e um gap factor γ, encontrar o vector $\mathbf{v} \in \mathcal{L}$ não nulo mais curto, em que $\mathbf{v}$ é único — no sentido de que qualquer outro vector $\mathbf{x} \in \mathcal{L}$ com $\|\mathbf{x}\| \leqslant \gamma\|\mathbf{v}\|$ é um múltiplo inteiro de $\mathbf{v}$.*

Assim, o Unique Shortest Vector Problem representa uma versão do SVP restrita aos reticulados em que o *gap* $\lambda_2(\mathcal{L})/\lambda_1(\mathcal{L}) > \gamma$, representando $\lambda_i(\mathcal{L})$ os mínimos sucessivos de $\mathcal{L}$. Este problema em particular será citado no âmbito deste trabalho, em 4.1, a propósito dos Esquemas de Criptografia de Chave Pública.

2.4.4 Shortest Basis Problem

O Shortest Basis Problem assemelha-se, em parte, a um problema de redução de bases. Considerando $\mathcal{B}(\mathcal{L})$ o conjunto de todas as bases do reticulado $\mathcal{L}$:

Definição 15. *Dado um reticulado $\mathcal{L} \subset \mathbb{Z}^n$ e o factor de aproximação $\lambda > 0$, encontrar uma base $\mathcal{B} = [\mathbf{b}_1, ...\mathbf{b}_d]$ tal que $\max_{1\leqslant i\leqslant d} \|\mathbf{b}_i\| \leqslant \gamma \max_{1\leqslant i\leqslant d} \|\mathbf{b}_i{}'\|$ para todas as bases $\mathcal{B}' = [\mathbf{b_1}', ..., \mathbf{b_d}']$.*

2.5 Redução entre Problemas Hard

A dificuldade dos diversos problemas *hard* pode, por vezes, ser comparada através de reduções entre estes, partindo do princípio de que um dado Problema A pode ser reduzido

ao Problema B se um qualquer método que permita resolver o Problema B possa ser utilizado para resolver as instâncias do Problema A [8]. Intuitivamente, infere-se, nestes moldes, que o Problema B é pelo menos tão *hard* quanto o Problema A. No entanto, embora uma solução para o Problema B veicule de imediato uma solução para o Problema A, esta relação não é invertível [15].

A Figura 9 representa as relações existentes entre os problemas *hard* em reticulados actualmente definidos e assume, por questões de simplificação, que cada problema se encontra num reticulado *full-rank* em $\mathbb{Z}^n$ ($\mathbb{Z}^m$ no caso específico dos problemas SIS e LWE). Uma seta entre dois problemas indica que o problema na ponta de origem pode ser reduzido ao problema na ponta de destino (em tempo polinomial). Os pormenores técnicos das reduções esquematizadas, como seja a justificação dos factores de aproximação, não são detalhados por não se considerar relevante para o contexto deste trabalho, podendo ser consultados na íntegra em [15].

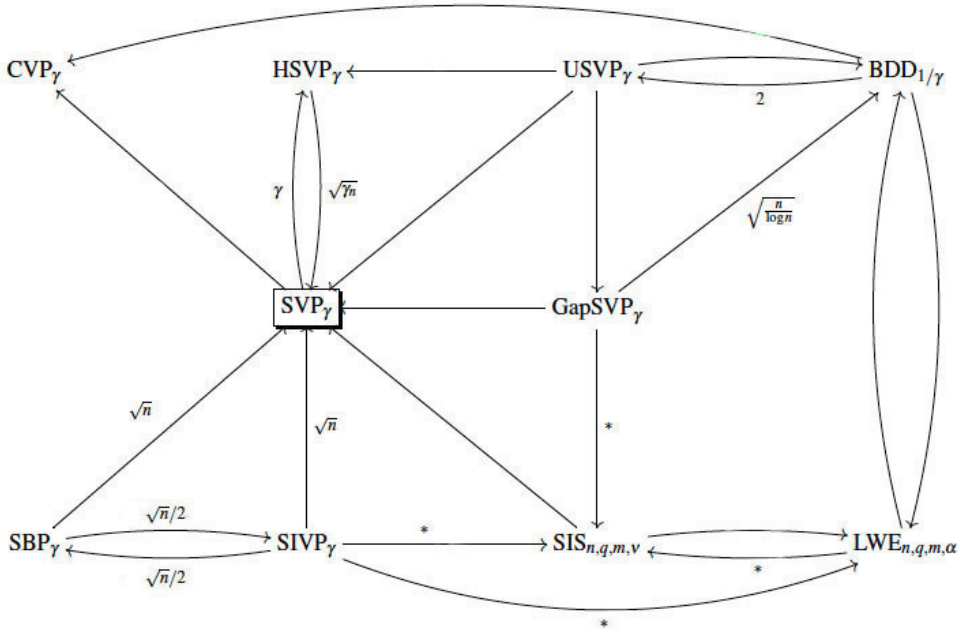

Figura 9: Esquema ilustrativo das relações entre os problemas *hard* em reticulados.

3 Resolução de Problemas Hard em Reticulados

Apresentar-se-ão, nesta secção, apenas métodos de resolução desenvolvidos e actualmente utilizados no âmbito dos vectores curtos, nomeadamente do SVP. Pese embora se depreendesse, à partida, que não existiam algoritmos capazes de levar a cabo a tarefa de encontrar um vector curto em tempo polinomial — dado que este problema apresenta uma complexidade NP-*hard* —, inferiu-se que, para dimensões d consideradas razoáveis, este processo é computacionalmente viável, nomeadamente através de técnicas como a Enumeração e os algoritmos de *sieving*.

3.1 Enumeração

A resolução do SVP por intermédio da técnica de enumeração baseia-se no teste de todas as combinações possíveis de vectores de uma base, destacando, entre todas as possibilidades, o vector mais curto. Naturalmente, testar todas as combinações possíveis implicaria uma quantidade infinita de vectores, pelo que se torna necessário equacionar e explorar um método associado à enumeração que veicule um limite a esta quantidade — a ortagonalização de Gram Schmidt [11, 12].

3.1.1 Ortogonalização de Gram Schmidt

A existência de uma base ortonormal facilita a procura dos coeficientes necessários à descrição de um vector enquanto combinação linear dos vectores $\mathbf{v_1}, ..., \mathbf{v_n}$, pertencentes à base de um espaço vectorial.

$$v = \frac{(v, v_1)}{\|v_1\|^2}\mathbf{v_1} + ... + \frac{(v, v_n)}{\|v_n\|^2}\mathbf{v_n} \tag{1}$$

Neste sentido, o processo de Gram Schmidt surge como um método de ortonormalização de bases, através do escalonamento de um vector da base de forma a que este tenha comprimento igual a um. Os vectores seguintes são iterativamente projectados no complemento ortogonal do *span* de vectores anteriores e igualmente escalonados. Quando os vectores não são escalonados no decorrer deste procedimento, a base resultante será ortogonal em vez de ortonormal, isto é, embora o produto interno entre todos os vectores da base seja $u.v = 0$, estes não são vectores unitários.

No que concerne a aplicação deste método no âmbito dos reticulados, não existe qualquer garantia de que a projecção de um vector do reticulado no complemento ortogonal de um outro vector do reticulado se encontre necessariamente no reticulado, nem tão pouco é sempre possível escalonar um vector nas condições anteriormente referidas, pelo que nem sempre é

possível ortogonalizar as bases. Não obstante, o processo de Gram Schmidt desempenha um papel fundamental nos métodos de redução e enumeração aplicados a bases de reticulados.

1 Ortogonalização de Gram Schimdt

Dada uma base $B = b_1, ..., b_d$ de um reticulado $\mathcal{L}$, calcular, para cada $i = 1, ..., d$:

1. Para cada $j < i$, calcular $\mu_{i,j} = s_j^{-1}(b_i \cdot b_i^*$, sendo $s_j = b_j^* \cdot b_j^*$;
2. $b_j^* = b_1 - \sum_{j<i} \mu_{i,j} b_j^*$, onde $\mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\|b_j^*\|^2}$, para todo $1 \leqslant j < i \leqslant d$.

Uma forma simplificada de representar o algoritmo de enumeração é por intermédio de uma árvore de pesquisa (Figura 10), cujos nós correspondem a um qualquer vector. Considerando que a raiz se encontra no nível 0, o nível i da árvore é composto por todos os vectores de π_{d-i+1}, para $0 \leqslant i \leqslant d$. Sendo v um nó do nível i, isto é, $v \in \pi_{d-i+1}$, a respectiva descendência consiste no conjunto de todos os vectores $u \in \pi_{d-i+2}(\mathcal{L})$ projectados em v: $v = \pi_{d-i+1}(u)$. No seguimento deste raciocínio, infere-se que a raiz consiste do vector π_{d+1}, o primeiro nível no conjunto de vectores pertencentes a $\pi_d(\mathcal{L}) = \mathcal{L}(b_d^*)$ de norma, no máximo, R, e assim sucessivamente, até ao nível d, que contém todos os vectores de $\pi_1(\mathcal{L})$ de norma, no máximo, R.

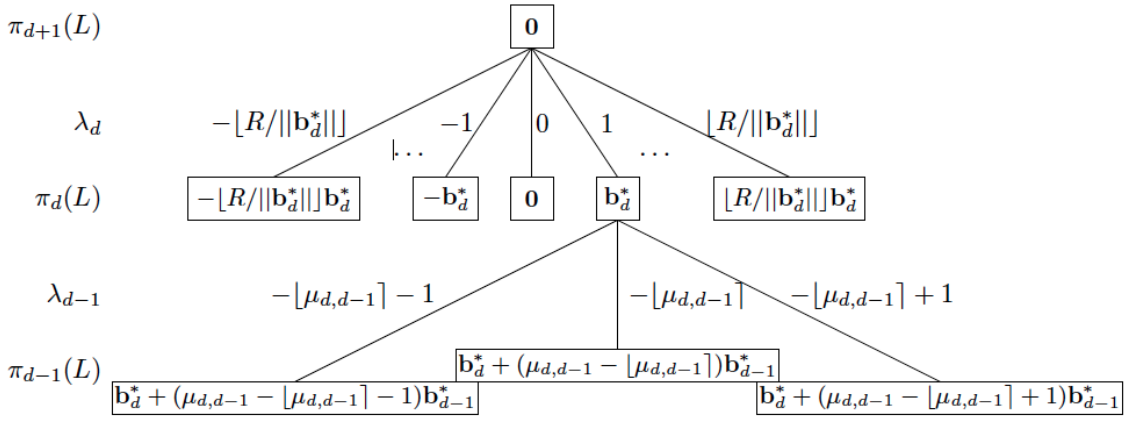

Figura 10: Esquematização dos dois primeiros níveis de uma árvore de enumeração.

Uma outra forma de descrever a enumeração de forma simplificada é através de um algortimo (Algoritmo 1), recorrendo à noção de árvore supracitada.

Dependendo do grau de precisão do delimitador R, as árvores de enumeração tendem a desenvolver-se exponencialmente, pelo que, pese embora correspondam a um método capaz de retornar uma solução exacta do SVP, o tempo de execução não é polinomial e, por essa

mesma razão, estes algoritmos são normalmente precedidos de algoritmos de redução. Em alternativa, Schnorr e Hörner propuseram uma técnica de enumeração melhorada, designada de *pruning* [16], recentemente elevada a outro patamar por Gama, Nguyen e Regev, num conceito denominado de *extreme pruning* [17].

Algoritmo 1 Enumeração

Requer: uma base reduzida $\{b_1, ..., b_d\}$ de $\mathcal{L}$ e os respectivos coeficientes de Gram-Schmidt $\mu_{i,j}$ e normas $\|b_i^*\|$

Garante: o vector de *output* do reticulado $\sum_i u_i b_i \in \mathcal{L}$ é um vector curto de $\mathcal{L}$

1: **repetir**
2: **se** a norma do nó actual for menor do que o limite, **então**
3: descer um nível, até ao descendente de norma mínima
4: **else**
5: subir um nível, até ao vizinho não visitado do ascendente de menor norma
6: **end if**
7: **até** que todos os nós tenham sido percorridos

3.2 Sieving

Em contraposição — ou como alternativa — à superexponencialidade da enumeração surgem os algoritmos de *sieving*, que revelam tempos de execução exponenciais. Porém, mesmo a variante prática mais rápida deste algoritmo conhecida é ultrapassada, em termos de *performance*, pelo algoritmo de enumeração de Schnorr-Hörner, até $n \approx 50$ (dimensão acima da qual este se torna impraticável) [8, 18]. Abaixo, apresenta-se um possível algoritmo de *sieving* — o algoritmo de Micciancio-Voulgaris (Algoritmo 2).

Sumariamente, o algoritmo procura exaurir o espaço de vectores curtos do reticulado, adicionando sucessivamente vectores curtos a uma lista Q até que os vectores $\mathbf{v_i}$ e $\mathbf{v_j}$ em Q estejam afastados, no máximo, à distância μ ou se tenham usado demasiadas amostras. Caso se encontrem dois vectores $\mathbf{v_1}, \mathbf{v_j} \in \mathcal{L}$ nas condições referidas, então, o vector $\mathbf{v} = \mathbf{v_i} - \mathbf{v_j}$ também pertence ao reticulado e tem um comprimento, no máximo, de μ, e a solução retornada pelo algoritmo é $\mathbf{v}$. Em caso contrário, novos vectores são adicionados à lista — cujo tamanho é limitado por uma constante fixa. No cômputo geral, é possível garantir, com elevada probabilidade, que, a dada altura, se abandona o ciclo representado no Algoritmo 2, ao encontrar um vector curto do reticulado [8].

Actualmente, para além de os próprios Micciancio e Voulgaris terem proposto uma melhoria ao seu algoritmo de *sieving* — o denominado Algoritmo de GaussSieve —, Ajtai et

al. [19] propuseram uma abordagem distinta, que vai ao encontro do conceito de *sieving* de forma mais explícita: em vez de exaurir o espaço de vectores curtos numa lista de vectores de dimensão substancialmente elevada, começa-se por uma lista de vectores longos, que vai sendo reduzida em tamanho, paralelamente à redução das normas dos vectores nela contidos, através de sucessivas iterações. Na mesma linha de construção, inserem-se as propostas de Nguyen e Vidick [20], e Wang et al. [21], a título de exemplo, que não serão exploradas no presente trabalho.

Algoritmo 2 Algoritmo de *sieving* de Micciancio-Voulgaris

Requer: uma base reduzida $\{b_1, ..., b_d\}$ de $\mathcal{L}$ e um valor $\mu \in \mathbb{R}$

Garante: se $\mu > \lambda_1(\mathcal{L})$, então, com grande probabilidade, o algoritmo encontra um vector $\mathbf{v} \in \mathcal{L}$ com $\|\mathbf{v}\| \leqslant \mu$

$Q \leftarrow \{0\}$
$\xi \leftarrow 0.685$
$N \leftarrow poly(d) \cdot 2^{3.199d}$
for $i = 1$ to N **do**
  $\mathbf{e_i} \in_R \mathbf{B_d}(\mathbf{0}, \xi, \mu)$
  $\mathbf{r_i} \leftarrow \mathbf{e_i} \bmod \mathbf{B}$
  while $\exists \mathbf{v_j} \in Q : \|\mathbf{r_i} - \mathbf{v_j}\| \leqslant (1 - \frac{1}{d}\|\mathbf{v_i}\|$ **do**
    $\mathbf{r_i} \leftarrow \mathbf{r_i} - \mathbf{v_j}$
  end while
  $\mathbf{v_i} \leftarrow \mathbf{r_i} - \mathbf{e_i}$
  se $\mathbf{v_i} \notin Q$ **then**
    se $\exists \mathbf{v_j} \in Q : \|\mathbf{v_i} - \mathbf{v_j}\| < \mu$ **then**
      return $\mathbf{v_i} - \mathbf{v_j}$
    end if
    $\mathbf{Q} \leftarrow \mathbf{Q} \cup \{\mathbf{v_i}\}$
  end if
end for
return $\perp$

4 Técnicas Criptográficas

Conforme vem sido explorado ao longo do presente trabalho, as construções criptográficas baseadas em reticulados representam uma promessa no âmbito da criptografia pós-quântica, na medida em que são comprovadamente eficientes, de implementação simples e seguras contra computadores quânticos. Nesta secção, serão apresentados essencialmente dois tipos de construção: esquemas de criptografia de chave pública e esquemas de assinatura digital.

4.1 Esquemas de Criptografia de Chave Pública

A descoberta da conexão entre o *worst-case* e o *average-case* dos problemas *hard* de reticulados potenciou a sua utilização no âmbito da criptografia de chave pública. Serão descritos, nesta secção, os três primeiros criptosistemas baseados em reticulados, nomeadamante os Criptosistemas de Ajtai-Dwork, Goldreich-Goldwasser-Halevi (GGH) e NTRU.

4.1.1 Criptosistema de Ajtai-Dwork

O criptosistema de Ajtai-Dwork foi o primeiro, no seio da criptografia de chave pública, a admitir uma prova de segurança baseada em assumpções de dificuldade *worst-case* em problemas de reticulados. Assim, embora o sistema não tenha sido apresentado recorrendo explicitamente à utilização de reticulados, a referida prova de segurança evidencia que cada instância do uSVP pode ser transformada, com elevada probabilidade, numa instância aleatória do criptosistema.

Tabela 2: Parâmetros do Criptosistema de Ajtai-Dwork.

Parâmetro	Descrição	Conhecimento
n	Dimensão	Público
$m = n^3$	Inteiro	Público
$r_n = n^n$	Inteiro	Público
$\mathbf{u}$	Vector n-dimensional	Privado
$\mathbf{w}_1, ..., \mathbf{w}_n, \mathbf{v}_1, ..., \mathbf{v}_m$	$n + m$ vectores n-dimensionais	Público

Relativamente aos parâmetros inerentes ao sistema, o parâmetro de segurança n determina a dimensão a dimensão do espaço vectorial em que o criptosistema está inserido. Sendo $m = n^3$, $r_n = n^n$ e $c > 0$, considerem-se um cubo $\mathcal{B}_n$ de lados de comprimento r_n e uma esfera $\mathcal{S}_n$ de raio n^{-c}, ambos n-dimensionais:

$$\mathcal{B}_n = \{x \in \mathbb{R}^n : |x_i| \leqslant r_n/2, \forall i\}$$

$$\mathcal{S}_n = \{x \in \mathbb{R}^n : \|x\| \leqslant n^{-c}\}$$

Neste contexto, a chave privada corresponde a um vector $\mathbf{u}$ escolhido de forma aleatória a partir de $\mathcal{S}_n$, com base na qual uma distribuição $\mathcal{H}_u$ é definida em $\mathcal{B}_n$. Esta construção é definida em três passos: extrair $\mathbf{x}$ de $\{\mathbf{x} \in \mathcal{B} : \langle \mathbf{x}, \mathbf{u} \rangle \in \mathbb{Z}\}$ de forma uniforme e aleatória; desenhar n vectores de erro $y_1, ..., y_n$ a partir de $\mathcal{S}_n$, independente e uniformemente, também de forma aleatória; e gerar $\mathbf{v} = \mathbf{x} + \sum_{i=1}^{n} \mathbf{y_i}$. Por outro lado, a chave pública é obtida retirando $n + m$ vectores $\mathbf{w_1}, ..., \mathbf{w_n}$, $\mathbf{v_1}, ..., \mathbf{v_m}$ de $\mathcal{H}_u$ aleatoriamente.

O processo de cifragem é levado a cabo *bit* a *bit*, sendo que, para cifrar um *bit* 0, devem retirar-se $b_1, ..., b_m$, uniforme e aleatoriamente, de {0,1} e reduzir o vector $\sum_i b_i \mathbf{v}_i$ módulo o paralelepípedo $\mathcal{P}(\mathbf{w}_1, ..., \mathbf{w}_n)$ — que corresponde ao paralelepípedo abrangido pelos $\mathbf{w}_i$. O resultado será um vector n-dimensional, que corporiza o texto cifrado $\mathbf{c}$. A cifragem de um *bit* 1, por outro lado, implica que se escolha aleatoriamente um vector n-dimensional do paralelepípedo referido, que constituirá o texto cifrado $\mathbf{c}$. O processo inverso, o de decifragem, compreende o cálculo do produto interno do vector n-dimensional correspondente ao *bit* em análise, recorrendo à chave privada $\mathbf{u}$. Caso dist($\langle \mathbf{c}, \mathbf{u} \rangle$,$\mathbb{Z} \leqslant n^{-1}$), $\mathbf{c}$ é decifrado como 0; em caso contrário, $\mathbf{c}$ é decifrado como 1 [22].

Dado que, à data deste trabalho, não se reúnem o conhecimento nem os meios necessários à construção de exemplos de funcionamento destes sistemas, para efeitos de exemplificação, recorrer-se-á aos exemplos construídos por Joop van De Pol em [12] para todos os criptosistemas apresentados.

Exemplo 1. Para uma dimensão $n = 3$, os parâmetros do sistema fixam-se em $m = 27$ e $r_3 = 27$ (Tabela 2). Numa primeira fase, gerou-se a chave privada de forma aleatória a partir da esfera $\mathcal{S}_n$:

$$\mathbf{u} = (-0.524655, 0.606024, 0.36764)$$

A chave pública ($\mathbf{w}_1$,$\mathbf{w}_2$,$\mathbf{w}_3$,$\mathbf{v}_1$,...,$\mathbf{v}_{27}$), por sua vez, é derivada retirando aleatoriamente vectores do cubo $\mathcal{B}_n$ compreendido nos hiperplanos definidos por $\mathbf{u}$ e adicionando três vectores da esfera tri-dimensional de raio $n^{-c} = 3^{-8} = 1/6561$. Por motivos de simplificação de exposição, a tabela correspondente à chave pública não é aqui incluída. A sua estrutura compreende um tripleto para cada $\mathbf{w_i}$ e $\mathbf{v_i}$. Considerando uma mensagem $\mathbf{m} = (1,0,1,1,1,1,1,0,1)$, cifrada *bit* a *bit*, o texto cifrado resultante é:

$$\begin{aligned}\mathbf{c} = ((&9.986, 3.746, -2.791), (1.365, 1.417, -3.108), (-16.955, -1.992, 9.227),\\ &(-5.223, -1.139, 1.278), (5.590, -3.151, -6.728), (-7.319, 9.134, 17.364),\\ &(-3.014, 3.752, 2.509), (-9.874, 4.964, 7.645), (-9.039, 4.727, 5.035)).\end{aligned}$$

Em **c**, cada tripleto corresponde à cifragem de um *bit*. Para proceder à decifragem, o produto interno de cada tripleto com **u** é calculado e, de seguida, determina-se se este se encontra suficientemente perto de $\mathbb{Z}$, isto é, se a distância ao inteiro mais próximo é menor do que $n^{-1} = 1/3$. Neste caso específico, os produtos internos obtidos foram:

Tabela 3: Produtos internos do texto cifrado **c** com **u** [12].

$\mathbf{c}_i$	$\langle \mathbf{c}_i, \mathbf{u} \rangle$
$\mathbf{c}_1$	3.994
$\mathbf{c}_2$	1.000
$\mathbf{c}_3$	-11.08
$\mathbf{c}_4$	2.519
$\mathbf{c}_5$	7.316
$\mathbf{c}_6$	-15.76
$\mathbf{c}_7$	-4.777
$\mathbf{c}_8$	-10.99
$\mathbf{c}_9$	-9.458

Todos os valores que se encontrem no intervalo $(z-1/3, z+1/3)$ para um dado inteiro z são decifrados como 0's, sendo os restantes decifrados como 1's, resultando na mensagem **m'** = (0,0,0,1,0,0,0,0,1), que não corresponde à mensagem original, conforme se esperava. Apenas os 0's são decifrados de forma correcta, sendo que se verificaram 5 erros de decifragem, pelo que se conclui que, para $n = 3$, a taxa de fallha neste processo é de, aproximadamente, 2/3.

O criptosistema de Ajtai-Dwork constitui, das três instâncias abordadas no decorrer deste trabalho, o único acompanhado de uma prova de segurança, inspirada na descoberta seminal de Ajtai de que existe efectivamente uma conexão entre a complexidade *worst-case* e *average-case* do SVP, através da qual este provou que este problema é `NP-Hard`, quando sujeite a reduções aleatórias [3]. Infelizmente, este criptosistema não demonstra ser muito eficiente, sobretudo devido à expansão da mensagem e ao elevado espaço requerido para o armazenamento da chave pública.

4.1.2 Criptosistema de Goldreich-Goldwasser-Halevi (GGH)

Pese embora represente um interesse inegável, sobretudo a nível teórico, o criptosistema anteriormente exposto revela, em parte, ser pouco eficiente, devido à expansão da mensagem e ao elevado espaço requerido para armazenar a chave pública, para além de apresentar fragilidades ao nível da segurança. Comparativamente com este sistema, o Criptosistema GGH surge associado a uma eficiência bastante superior, não obstante não ser acompanhado de uma prova de segurança.

A base que suporta o Criptosistema GGH é essencialmente composta por dois parâmetros: a dimensão n do reticulado e um parâmetro de segurança σ, sendo este último responsável por determinar a dificuldade do CVP derivado do processo de cifragem. A chave privada, neste caso, consiste numa matriz secreta $\mathcal{R}$, cujas colunas formam uma base do reticulado $\mathcal{L}$. Por sua vez, a chave pública consiste numa matriz $\mathcal{B}$, que representa uma base para $\mathcal{L}$ distinta da anterior e é gerada aleatoriamente a partir da base secreta $\mathcal{R}$.

Tabela 4: Parâmetros do Criptosistema GGH.

Parâmetro	Descrição	Conhecimento
n	Dimensão	Público
σ	Parâmetro de Segurança	Público
$\mathcal{R}$	Matriz $n \times n$	Privado
$\mathcal{B}$	Matriz $n \times n$	Privado

A cifragem da mensagem é realizada através da codificação da mesma como um vector $\mathbf{m} \in \mathbb{Z}^n$, seguida da criação de um vector de erro e — retirado aleatoriamente da conjunto $\{-\sigma,\sigma\}^n$ — e da computação do texto cifrado:

$$\mathbf{c} = \mathcal{B}\mathbf{m} + e$$

A mensagem $\mathbf{m}$ pode ser recuperada por intermédio do processo de decifragem do texto cifrado:

$$\mathbf{m} = \mathcal{B}^{-1}\mathcal{R}\lfloor\mathcal{R}^{-1}\mathbf{c}\rceil$$

Cada mensagem $\mathbf{m} \in \mathbb{Z}^n$ corresponde a um ponto do reticulado $\mathbf{m}_\mathcal{L} = \mathcal{B}\mathbf{m}$ e, paralelamente, cada ponto $\mathbf{m}_\mathcal{L}$ corresponde a uma dada mensagem $\mathbf{m} = \mathcal{B}^{-1}\mathbf{m}_\mathcal{L}$. No processo de cifragem, a mensagem $\mathbf{m}$ é transformada no ponto $\mathbf{m}_\mathcal{L}$ correspondente e o texto cifrado $\mathbf{c}$ é obtido a partir da adição de um vector de erro e. Posteriormente, σ e e são escolhidos de forma a que $m_\mathcal{L}$ seja o vector do reticulado mais próximo de $\mathbf{c}$. No sentido de obter $\mathbf{m}_\mathcal{L}$ e, consequentemente, $\mathbf{m}$, torna-se necessário resolver o CVP, nomeadamente através do Método de Babai — método de redução que não é discutido no âmbito deste trabalho —, partindo-se do princípio de que este será relativamente eficaz utilizando a base privada $\mathcal{R}$, mas não suficientemente eficaz utilizando a base pública $\mathcal{B}$. Conforme mencionado anteriormente, o parâmetro σ determina a dificuldade do problema a resolver nesta fase, pelo que devem ser tidas em consideração as implicações que advêm de um aumento do mesmo: quanto maior o valor de σ, maior a distância entre o vector $\mathbf{m}_\mathcal{L}$ e o texto $\mathbf{c}$ — logo, maior a dificuldade — e maior a probabilidade de surgirem erros no processo de decifragem [23].

Exemplo 2. Para uma dimensão $n = 4$ e um parâmetro de segurança $\sigma = 1$, a matriz privada $\mathcal{R}$ é gerada retirando aleatoriamente os seus *inputs* — números inteiros — do intervalo [-4,4]. Por sua vez, a matriz pública $\mathcal{B}$ corresponde à Forma Normal de Hermite de $\mathcal{R}$.

$$\mathcal{R} = \begin{pmatrix} 3 & 1 & -1 & 3 \\ 3 & -4 & 3 & 1 \\ 3 & -3 & -4 & -3 \\ -2 & -3 & -2 & 3 \end{pmatrix} \qquad \mathcal{B} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -349 & 311 & 321 & 851 \end{pmatrix}$$

Posteriormente, a mensagem $\mathbf{m}$ = (-2,0,-4,-1) é cifrada recorrendo ao vector de erro $\mathbf{e}$ = (-1,1,1,-1), correspondendo o texto cifrado a $\mathbf{c} = \mathcal{B}\mathbf{m}+\mathbf{e} = (-3, 1, -3, -1438)$. O processo de decifragem da mensagem $\mathbf{m}$ é levado a cabo por intermédio do Método de Babai, utilizando a base privada $\mathcal{R}$:

$$\mathbf{x} = \lfloor \mathcal{R}^{-1}\mathbf{c} \rceil = (143, 152, 112, -157)$$
$$\mathbf{m}_{\mathcal{L}} = \mathcal{R}\mathbf{x} = (-2, 0, -4, -1437)$$
$$\mathbf{m} = \mathcal{B}^{-1}\mathbf{m}_{\mathcal{L}} = (-2.0, -4, -1)$$

Assim, a mensagem decifrada é equivalente à mensagem original, pelo que se deduz que não ocorreram quaisquer erros de decifragem. Se, por outro lado, se utilizar a base $\mathcal{B}$, o resultado não é a mensagem original, conforme se esperaria.

Em suma, o criptosistema GGH revela um interesse considerável na medida em que é explicitamente baseado em em reticulados, embora a função de *trapdoor* utilizada no CVP introduza comprovadamente fragilidades estruturais que podem ser exploradas por intermédio de algoritmos de redução capazes de quebrar o sistema para dimensões reduzidas. Não obstante, apresenta maior eficiência do que o criptosistema de Ajtai-Dwork.

4.1.3 Criptosistema NTRU

De forma semelhante ao criptosistema GGH, o criptosistema NTRU é orientado à prática, descurando também a existência de uma prova de segurança que assegure que quebrar o sistema é pelo menos tão *hard* quanto resolver um dado problema de reticulados que lhe seja inerente. Não obstante este criptosistema ser baseado em anéis (nomeadamente o anel de convolução de polinómios), pode ser descrito de forma equivalente utilizando reticulados com uma estrutura especial.

Embora, conforme explicitado anteriormente, o sistema NTRU seja descrito como um criptosistema de anel polinomial, a relação entre as chaves pública e privada define um

reticulado — denominado de reticulado NTRU — cuja base (uma das possíveis) pode ser derivada a partir da chave pública e cuja chave secreta corresponde a determinados vectores curtos pertencentes a esse mesmo reticulado. O sistema tem por base quatro parâmetros fundamentais: o grau n (que, por motivos de segurança, deve ser primo), os módulos q e p (que devem ser co-primos), e os limites inteiros d_f, d_g e d_r.

Tabela 5: Parâmetros do Criptosistema NTRU.

Parâmetro	Descrição	Conhecimento
n	Grau	Público
q	Módulo Maior	Público
p	Módulo Menor	Público
d_f	Limite inteiro para f	Público
d_g	Limite inteiro para g	Público
d_r	Limite inteiro para r	Público
$\mathbf{f}$	Vector coeficiente de f	Privado
$\mathbf{g}$	Vector coeficiente de g	Privado
$\mathcal{H} = p_q^{-1}[\mathcal{C}^*\mathbf{h}] \pmod q$	Vector coeficiente de h	Público
$\mathbf{h} = p[\mathcal{C}^*\mathbf{f}]_q^{-1}\mathbf{g}) \pmod q$	Vector coeficiente de h	Público

Considerando $\mathcal{C}$ a rotação cíclica que envia um vector $(x_1, x_2, ..., x_n)^T$ para $(x_n, x_1, ..., x_{n-1})^T$ e definindo, para um dado $x \in \mathbb{R}^n$, a matriz circulante de x como:

$$[\mathcal{C}^*x] = [x, \mathcal{C}x, ..., \mathcal{C}^{n-1}x] = \begin{pmatrix} x_1 & x_n & \cdots & x_2 \\ x_2 & 0 & \cdots & x_3 \\ \vdots & \vdots & \ddots & \vdots \\ x_n & x_{n-1} & \cdots & x_1 \end{pmatrix}$$

O produto da convolução dos polinómios $f * g$ é equivalente à multiplicação matricial $[\mathcal{C}^*\mathbf{f}]\mathbf{g}$, representando $\mathbf{f}$ e $\mathbf{g}$ os vectores coeficiente de f e g, respectivamente. Neste seguimento, a estrutura do NTRUEncrypt tem início na escolha da chave privada ($\mathbf{f}$,$\mathbf{g}$), que corresponde a um vector curto em $\mathbb{Z}^{2n}$, e cujos vectores $\mathbf{f}$ e $\mathbf{g}$, secretos, devem apresentar as seguintes características:

- A matriz $[\mathcal{C}^*\mathbf{f}]$ deve ser invertível mod q e mod p, sendo as respectivas inversas $[\mathcal{C}^*\mathbf{f}]_q^{-1}$ e $[\mathcal{C}^*\mathbf{f}]_p^{-1}$.
- $\mathbf{f}$ e $\mathbf{g}$ devem ser pequenos.

A chave pública $\mathbf{h}$ pode, então, ser derivada partindo da chave privada através da equação:

$$[\mathcal{C}^*\mathbf{f}]\mathbf{h} \equiv p\mathbf{g} \pmod q \Rightarrow \mathbf{h} = p[\mathcal{C}^*\mathbf{f}]^{-1}\mathbf{g} \bmod q$$

Sendo $H = p_q^{-1}[\mathcal{C}^*\mathbf{h}]$, em que p_q^{-1} corresponde à inversa de p mod q, o reticulado NTRU é definido por todos os vectores $(x,y) \in \mathbb{Z}^{2n}$ que satisfaçam $y \equiv Hx \pmod{q}$ e é abrangido pelas colunas da matriz L, em que I_n representa a matriz identidade $n \times n$ e O_n a matriz nula $n \times n$.

$$L = \begin{pmatrix} I_n & O_n \\ H & qI_n \end{pmatrix}$$

Com base nos parâmetros do sistema definidos, torna-se possível explorar os processos de cifragem e decifragem. A cifragem de uma mensagem passa por codificá-la como um vector $\mathbf{m} \in \mathbb{Z}^n$ com coeficientes módulo p, gerar aleatoriamente um *blinding factor* $\mathbf{r} \in \{-1, 0, 1\}^n$ e, por último, calcular o texto cifrado:

$$\mathbf{c} = [\mathcal{C}^*\mathbf{h}]\mathbf{r} + \mathbf{m} \pmod{q}$$

Por sua vez, a decifragem do texto $\mathbf{c}$ implica a redução de $\mathbf{a} \equiv [\mathcal{C}^*\mathbf{f}]\mathbf{c} \pmod{q}$ de tal forma que todos os coeficientes de $\mathbf{a}$ se encontrem no intervalo $[-q/2, q/2)$, seguida do cálculo que permite obter a mensagem:

$$\mathbf{m} = [\mathcal{C}^*\mathbf{f}]_p^{-1}\mathbf{a} \pmod{p}$$

No que concerne a funcionalidade do sistema, os parâmetros d_f, d_g e d_r, que estabelecem os limites inteiros, devem ser escolhidos de forma a que mesmo os produtos da convolução tenham coeficientes pequenos, contando que, quanto maiores forem estes parâmetros, maior é a probabilidade de as entradas de x $= \mathcal{C}^*\mathbf{f}]\mathbf{m} + p[\mathcal{C}^*\mathbf{g}]\mathbf{r}$ não se encontrarem no intervalo $[-q/2, q/2)$. Caso esta situação se verifique, ocorrerão erros no processo de decifragem que poderão ser potencialmente utilizados para conseguir extrair a chave privada, em determinadas situações [12, 24].

Exemplo 3. Considerando os parâmetros $n = 7$, $q = 32$, $p = 3$, $d_f = 3$ e $d_g = d_r = 2$, a chave privada gerada de forma aleatória consiste em $\mathbf{f}$ = (-1,0,0,1,1,-1,1) e $\mathbf{g}$ = (1,-1,0,0,-1,1,0). Com base na equação que deriva a chave pública $\mathbf{h}$ a partir da chave privada, obtém-se $\mathbf{h}$ = (0,24,26,27,10,12,29). Neste contexto, as inversas de $[\mathcal{C}^*\mathbf{f}]$ módulo q e p são, respectivamente:

$$[\mathcal{C}^*\mathbf{f}]_q^{-1} = \begin{pmatrix} 17 & 28 & 16 & 1 & 30 & 9 & 28 \\ 28 & 17 & 28 & 16 & 1 & 30 & 9 \\ 9 & 28 & 17 & 28 & 16 & 1 & 30 \\ 30 & 9 & 28 & 17 & 28 & 16 & 1 \\ 1 & 30 & 9 & 28 & 17 & 28 & 16 \\ 16 & 1 & 30 & 9 & 28 & 17 & 28 \\ 28 & 16 & 1 & 30 & 9 & 28 & 17 \end{pmatrix} \qquad [\mathcal{C}^*\mathbf{f}]_p^{-1} = \begin{pmatrix} 2 & 1 & 2 & 2 & 0 & 1 & 2 \\ 2 & 2 & 1 & 2 & 2 & 0 & 1 \\ 1 & 2 & 2 & 1 & 2 & 2 & 0 \\ 0 & 1 & 2 & 2 & 1 & 2 & 2 \\ 2 & 0 & 1 & 2 & 2 & 1 & 2 \\ 2 & 2 & 0 & 1 & 2 & 2 & 1 \\ 1 & 2 & 2 & 0 & 1 & 2 & 2 \end{pmatrix}$$

Atendendo a um *blinding factor* $\mathbf{r}$ = (1,1,0,0,0,-1,-1) escolhido de forma aleatória, a mensagem $\mathbf{m}$ = (0,-1,1,-1,1,0,-1) é cifrada, dando lugar ao texto cifrado $c = [\mathcal{C}^*\mathbf{h}]\mathbf{r} + \mathbf{m} \equiv (11, 2, 14, 30, 29, 25, 16) \pmod{q}$. A decifragem é levada a cabo multiplicando $\mathbf{c}$ por $[\mathcal{C}^*\mathbf{f}]$ e reduzindo-o mod q de forma a que os respectivos coeficientes se encontrem no intervalo [-q/2,q/2):

$$[\mathcal{C}^*\mathbf{f}]\mathbf{c} \equiv (4, 4, -4, 1, -7, -4, 5) \pmod{q}$$

A mensagem original é obtida multiplicando por $[\mathcal{C}^*\mathbf{f}]_p^{-1}$ e reduzindo mod q:

$$[\mathcal{C}^*\mathbf{f}]_p^{-1} \cdot (4, 4, -4, 1 - -7, -4, 5) \equiv (0, -1, 1, -1, 1, 0, -1)(mod\mathrm{p}) = \mathbf{m}$$

Em suma, pese embora o criptosistema NTRUEncrypt seja definido utilizando polinómios truncados, os ataques mais eficazes tiram partido da estrutura das chaves, através do uso de algoritmos de redução de reticulados. Contrariamente aos criptosistemas mencionados previamente, o NTRU mostra-se suficientemente eficiente para elevar os parâmetros de segurança ao mesmo tempo que mantém a competitividade.

4.2 Esquemas de Assinatura Digital

O conceito de assinatura digital pode ser encarado como o dual da criptografia de chave pública — que compreende a cifragem de mensagens com recurso a uma chave pública e decifragem utilizando a chave privada correspondente —, na medida em que a mensagem é assinada utilizando a chave privada e, posteriormente, a assinatura assim gerada pode ser verificada recorrendo à respectiva chave pública. Neste sentido, as assinaturas digitais podem ser utilizadas tanto para efeitos de autenticação como de avaliação da integridade das mensagens e de não-repudiação. Formalmente:

Definição 16. *Um esquema de assinatura consiste num tripleto de algoritmos de tempo polinomial (G, S, V) tais que, para cada par de outputs (s,v) de $G(1^n)$ e uma qualquer mensagem m de n-bits,*

$$Pr[V(v, m, S(s, m)) = 1] = 1$$

em que a probabilidade é dominada pelo carácter aleatório dos algoritmos de assinatura (S) e de verificação (V). Neste contexto, G é o algoritmo de key-generation e s e v são as chaves de assinatura e verificação, respectivamente.

Considera-se que um esquema de assinatura é seguro se for desprezável a probabilidade de um qualquer *forger*, após visualizar assinaturas de mensagens à sua escolha, conseguir assinar uma mensagem cuja assinatura ainda não tenha visto. Isto é:

Definição 17. *Um esquema de assinatura (G, S, V) é dito seguro se, para todo o forger $\mathcal{F}$ de tempo polinomial (e possivelmente aleatório), a probabilidade de, depois de visualizar a chave pública e $\{(\mu_1, S(s,\mu_1)),\ldots,((\mu_q, S(s,\mu_q))\}$, para quaisquer q (polinomial em n) mensagens μ_1 à sua escolha, $\mathcal{F}$ conseguir produzir $(\mu \neq \mu_1, \sigma)$ tal que $V(v,\mu,\sigma=1)$ seja negligentemente pequena.*

De entre as diversas tentativas de construção de esquemas de assinatura digital baseados em reticulados, apenas a recente proposta de Lyubashevsky [25] se mantém, ainda, inquebrável, sendo que abordagens como aquelas derivadas dos criptosistemas GGH ou NTRU — o NTRU Signature Scheme, quebrado por Gentry et al., e o NTRUSign, quebrado por Nguyen e Regev, após a respectiva aproximação a um problema de aprendizagem — provaram ser falíveis para determinadas dimensões.

4.2.1 Esquema de Assinatura de Lyubashevsky

O esquema de Lyubashevsky baseia-se em reticulados ideais e recorre à transformada de Fiat-Shamir para transformar esquemas de identificação em esquemas de assinatura digital, abordagem que permite minorar um defeito associado ao esquema de identificação inicialmente construído: considerando que uma resposta correcta por parte do *prover* é suficiente para a autenticação, numa dada fracção constante de tempo, o *prover* não é capaz de responder correctamente ao desafio do *verifier* e é forçado a abortar o protocolo, o que implica a repetição dos passos de *commit* e *challenge* do esquema de identificação várias vezes, de forma a assegurar que um *prover* válido é aceite contra uma probabilidade aceitável. Ao anular qualquer interacção até que a assinatura seja efectivamente emitida, deixa de ser necessário expor as tentativas falhadas de resposta ao desafio — que correspondem às tentativas de assinar —, pelo que, ainda que as sucessivas falhas sejam *time-consuming*, o comprimento da assinatura final é tão curto quanto seria caso o *signer* tivesse tentado assinar apenas uma vez e tivesse imediadamente sucedido. Assim, e contando que a probabilidade de falha associada ao processo é diminuta e constante ($\approx 2/3$), o protocolo de assinatura sucederia ao fim de apenas três repetições [25, 12]. A segurança dos esquemas de identificação e de assinatura assim obtidos baseia-se na *worst-case hardness* de aproximar o vector mais curto a um factor de $\tilde{O}(n^2)$ em reticulados correspondentes a ideais no anel $\mathbb{Z}[x]/\langle x^n + 1\rangle$ e distancia-se dos demais devido ao facto de a assinatura poder ser concluída em tempo $\tilde{O}(n)$ [25].

4.2.2 Esquemas Baseados Em Funções de Hash Resistentes a Colisões

Por definição, uma função de *hash* resistente a colisões consiste numa função $h : D \rightarrow$ R que mapeia um domínio D a um conjunto consideravelmente mais reduzido R, tal que se torna computacionalmente difícil encontrar colisões. Neste contexto, entende-se por colisão a existência de um par $x_1, x_2 \in D$ tal que se verifique $x_1 \neq x_2$ e, ainda assim, $h(x_1) = h(x_2)$. Considerando uma colecção de funções $\{h_k : D \rightarrow R\}$ e o parâmetro k aleatório, torna-se possível assegurar que nenhum atacante é capaz de encontrar uma colisão em h_k de forma eficiente, pese embora tais colisões existam, dado que $|D| \gg |R|$. Neste sentido, as funções de *hash* resistentes a colisões assumem um papel importante (ou, pelo menos, útil) na construção de primitivas criptográficas, permitindo comprimir uma mensagem extensa $\mathbf{x} \in D$ numa entidade sumária $h(x) \in R$ que se encontra computacionalmente associada a um $\mathbf{x}$ único devido à propriedade de resistência à colisão anteriormente descrita [24]. Pese embora as funções de *hash* possam ser construídas com base em problemas *standard* da Teoria dos Números, estas construções pecam por serem facilmente quebradas por computadores quânticos, pelo que uma abordagem mais robusta passa por baseá-las em reticulados.

Ajtai apresentou, no seu trabalho seminal [3], a primeira construção baseada em reticulados, reportando-se a uma família de funções unidireccionais cuja segurança assenta sobre a dificuldade *worst-case* do n^c-ASVP para uma dada constante $c > 0$, implicitando que a tentativa de inverter uma das funções desta família é, em termos de complexidade, equivalente à resolução de qualquer instância deste problema *hard*. Atentando no Algoritmo 3, que traduz os elementos envolvidos na construção de uma função de *hash* segundo a teoria de Ajtai, torna-se importante realçar que a escolha do parâmetro n determina o nível de segurança da função. Em termos de *bits*, a função mapeia $m \log q$ *bits* em $n \log q$ *bits*, pelo que o parâmetros m deve ser escolhido de forma a verificar $m > n \log q / \log d$ e a garantir, assim, que a função obtida é capaz de comprimir o *input* [24].

Algoritmo 3 Função de *hash* baseada na construção de Ajtai

Parâmetros: Inteiros n, m, q, $d \geqslant 1$.

Chave: Matriz $\mathcal{A}$ escolhida uniformemente de $\mathbb{Z}_q^{n \times m}$.

Função: $f_{\mathcal{A}} : \{0, ..., d-1\}^m \rightarrow \mathbb{Z}_q^n$ dada por $f_{\mathcal{A}}(\mathbf{y}) = \mathcal{A}\mathbf{y} \bmod q$.

Não obstante a simplicidade de implementação das funções de *hash* proposta por Ajtai, as construções conformes aos termos explicitados apresentam uma lacuna considerável, ao nível da eficiência, na medida em que o tamanho da chave cresce, pelo menos, de forma

quadrática em n. A título de exemplo, cita-se a situação equacionada por Micciancio e Regev [24]: considerando um cenário com $d = 2$, $q = n^2$ e $m = 2\log q = 4n\log n$, a correspondente função teria uma chave composta por $mn = 4n^2 \log n$ elementos de $\mathbb{Z}_q$ e a respectiva avaliação requereria, aproximadamente, um número semelhante de operações aritméticas. As colisões são dadas pelos vectores em $\Lambda_q^{\perp}(\mathcal{A})$ — o reticulado q-ary m-dimensional $\Lambda_q^{\perp}(\mathcal{A}) = \{\mathbf{y} \in \mathbb{Z}^m : \mathcal{A}\mathbf{y} = 0 \bmod q\}$, que contém todos os vectores ortogonais módulo q às linhas de $\mathcal{A}$ — com entradas em {1,0,-1}. Assim, considerando um ataque de complexidade $L = 3^{m/16} \approx 2^{m/10}$ e almejando obter 100 *bits* de segurança ($L \approx 2^{100}$), os parâmetros m e n teriam de ser fixados em $m \approx 1000$ e $n \geqslant 46$, o que resultaria numa função de *hash* com uma chave de tamanho $mn \log q \approx 500,000$ *bits* e um tempo de computação na ordem das $mn \approx 50,000$ operações aritméticas, o que é inconcebível no contexto das funções de *hash* resistentes a colisões.

Os problemas anteriormente apontados podem ser resolvidos de forma relativamente simples, adoptando matrizes com estruturas específicas, como seja, no Algoritmo 3, a substituição da matriz $\mathcal{A} \in \mathbb{Z}_q^{n \times m}$ por uma matriz bloco em que cada bloco seja uma matriz circulante, o que se repercute de imediato no espaço de armazenamento requerido para a chave — que passa de nm elementos de $\mathbb{Z}_q$ para apenas m elementos — e no tempo de execução — uma vez que a multiplicação por uma matriz circulante pode ser implementada em tempo $\tilde{O}(n)$ utilizando a Transformada Rápida de Fourier (Fast Fourier Transform (FFT)). Embora, sempre que se efectue uma modificação numa construção teórica no sentido do aumento da eficiência, se deva ponderar quais as implicações dessa modificação ao nível da segurança, Micciancio [26] provou que é razoável assumir que resolver problemas de reticulados neste tipo de reticulados é tão *hard* quanto no caso geral.

O Algoritmo 3 pode ser adaptado (Algoritmo 4) no sentido de incorporar chaves estruturadas $\mathcal{A}$, considerando como parâmetros os inteiros n, m, q, d e um vector $\mathbf{f} \in \mathbb{Z}^n$. Neste caso, a escolha aleatória de $\mathcal{A}$ a partir do conjunto de todas as matrizes é substituída pela declaração de $\mathcal{A}$ como uma matriz bloco com blocos estruturados $\mathcal{A}^{(i)} = \mathbf{F}^*\mathbf{a}^{(i)}$.

Algoritmo 4 Função de *hash* baseada em reticulados ideais.

Parâmetros: Inteiros n, m, q, d com $n|m$, e um vector $\mathbf{f} \in \mathbb{Z}^n$

Chave: m/n vectores $\mathbf{a}_1, ..., \mathbf{a}_{m/n}$ escolhidos independente e uniformemente de forma aleatória de $\mathbb{Z}_q^n$.

Função: $f_{\mathcal{A}} : \{0, ..., d-1\}^m \to \mathbb{Z}_q^n$ dada por $f_{\mathcal{A}}(\mathbf{y}) = [\mathbf{F}^*\mathbf{a}_1|...|\mathbf{F}^*\mathbf{a}_{m/n}]\mathbf{y} \bmod q$.

Na função representada no Algoritmo 4, as chaves são representadas por apenas m ele-

mentos de $\mathbb{Z}_q$ e esta pode ser avaliada recorrendo a $\tilde{O}(m)$ operações aritméticas utilizando a Transformada Rápida de Fourier (FTT) em $\mathbb{C}$. As colisões correspondem a vectores curtos no reticulado $\Lambda_q^{\perp}([\mathbf{F}^*\mathbf{a}_1|...|\mathbf{F}^*\mathbf{a}_{m\times n}])$ e, obedecendo a determinadas condicionantes, asseguram que as funções de *hash* construídas apresentam garantias de segurança *worst-case* [24, 26].

No sentido de contrariar o *overhead* que, na prática, se regista aquando da efectiva utilização das funções acima descritas, Lyubashevsky et al. [27] propuseram uma versão optimizada destas últimas: a família de funções de *hash* SWIFFT (Algoritmo 5), que tira partido da estrutura do algoritmo da FTT para atingir uma *performance* superior. Esta optimização é atingida considerando um vector binário como *input* da Transformada, o que limita necessariamente o número possível de entradas e viabiliza o cálculo prévio e o armazenamento das iterações iniciais numa tabela; e considerando o facto de que o algoritmo da FTT consiste em operações repetidas em paralelo sobre várias parcelas de dados, para as quais os microprocessadores actuais estão configurados [27, 24].

Algoritmo 5 Função de *hash* SWIFFT.

Parâmetros: Inteiros n, m, q, d tal que n é uma potência de 2, q é primo, $2n|(q-1)$ e $n|m$.

Chave: m/n vectores $\tilde{\mathbf{a}}_\mathbf{1}, ..., \tilde{\mathbf{a}}_{\mathbf{m/n}}$ escolhidos independente e uniformemente de forma aleatória de $\mathbb{Z}_q^n$.

Input: m/n vectores $\mathbf{y}^(1), ..., \mathbf{y}^{(m/n)} \in \{0, ..., d-1\}^n$.

Output: o vector $\sum_{i=1}^{m/n} \tilde{\mathbf{a}}^(\mathbf{i}) \odot (\mathbf{W}\mathbf{y}^{(i)}) \in \mathbb{Z}_q^n$, em que $\odot$ representa o produto do vector *component-wise*.

Com base nas construções apresentadas, Lyubashevsky e Micciancio desenvolveram um esquema de assinatura aparentemente óptimo, na medida em que admite uma prova de segurança baseada em assumpções de complexidade *worst-case* e revela uma eficiência assimptótica, sendo que o tamanho da chave e o tempo de assinatura e verificação são quase lineares na dimensão do reticulado inerente ao esquema. O sistema baseia-se num esquema de assinatura *one-time* — que permite a assinatura em segurança de uma única mensagem —, por sua vez baseado em funções de *hash* resistentes a colisões, que é posteriormente transformado num esquema de assinatura completo. Conforme explicitado na exposição acerca desta família de funções, encontrar colisões numa função *hash* h é computacionalmente *hard*, pelo que se parte de h e de um par de vectores $\mathbf{x}_1, ..., \mathbf{x}_m/n \in \mathbb{Z}_q^n$ e $\mathbf{y}_1, ..., \mathbf{y}_m/n \in \mathbb{Z}_q^n$ escolhidos de forma aleatória e de acordo com uma dada distribuição que gera vectores curtos com probabilidade elevada. As imagens destes vectores de *input* na função de *hash* $X = h(\mathbf{x}_1, ..., \mathbf{x}_m/n)$,

$Y = h(\mathbf{y}_1, ..., \mathbf{y}_m/n)$ correspondem à chave pública, sendo que as mensagens a assinar são representadas por vectores curtos $\mathbf{m} \in \mathcal{Z}_q^n$ e a assinatura consiste simplesmente no cálculo:

$$\sigma = (\sigma_1, ..., \sigma_{m/n}) = [\mathbf{F}^*\mathbf{m}]\mathbf{x}_1 + \mathbf{y}_1, ..., [\mathbf{F}^*\mathbf{m}]\mathbf{x}_m/n + \mathbf{y}_m/n \bmod q.$$

A construção proposta assegura não só o conceito básico de segurança, mas ainda uma noção mais robusta de segurança, denominada de *strong unforgeability*, que requer adicionalmente que um *forger* $\mathcal{F}$ não seja sequer capaz de apresentar uma assinatura diferente para uma mensagem de cuja assinatura este já teve conhecimento [28].

5 Conclusão

As construções criptográficas com base em reticulados representam uma promessa significativa no âmbito da criptografia pós-quântica, uma vez que não existem, actualmente, algoritmos quânticos cujo desempenho seja melhor do que o dos algoritmos clássicos, não obstante os reticulados apresentarem características que os tornam favoráveis (ou elegíveis) à resolução por intermédio deste tipo de algoritmos. O facto de serem de implementação menos complexa e de assumirem graus de eficiência relativamente equiparáveis aos das melhores alternativas conhecidas, aliado ao facto de, em certos casos, admitirem garantias de segurança sólidas baseadas na dificuldade *worst-case* de problemas de reticulados, fazem com que este tipo de construções sejam particularmente interessantes num futuro próximo, no sentido de proteger informação enviada através de canais inseguros.

Da análise dos três criptosistemas considerados — Ajtai-Dwork, GGH e NTRU — retira-se que apenas os dois últimos são explicitamente baseados em reticulados, sendo que o primeiro apenas se enquadra nesta dimensão ao nível da prova de segurança, directamente baseada em problemas *hard*. No que concerne o sistema NTRU, o facto de a relação entre as chaves pública e privada assentar sobre uma estrutura de reticulados torna-o especialmente vulnerável a ataques que utilizem técnicas de redução, mas este é irrefutavelmente, de entre os três, o mais eficiente. Todos os criptosistemas descritos almejam atingir a noção de segurança semântica, que não se revela suficientemente forte para casos em que o adversário não seja passivo. Nestas situações, há que considerar esquemas com garantias de segurança mais robustas, a grande maioria dos quais baseados no problema *hard* LWE, que não foi explorado no âmbito deste trabalho.

Por último, os esquemas de assinatura digital baseados nos conceitos subjacentes às construções anteriormente referidas, nomeadamente nos criptosistema GGH e NTRU, provaram ser facilmente quebráveis (pelo menos, na sua versão básica), facto indeferivelmente explanável pelo facto de nenhum dos dois ser acompanhado de uma prova de segurança. Pese embora se apresentem dois sistemas considerados bastante eficazes e seguros — um dos quais apresenta mesmo *strong unforgeability* —, os esquemas de assinatura baseados em reticulados não atingiram ainda o patamar de desenvolvimento das restantes construções criptográficas.

Referências

[1] S. Goldwasser. New directions in cryptography: twenty some years later (or cryptograpy and complexity theory: a match made in heaven). In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, page 314. IEEE Computer Society, 1997.

[2] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theor.*, 22(6):644–654.

[3] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *In Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 99–108, 1996.

[4] D. Johnson. Handbook of theoretical computer science. pages 67–161. MIT Press, 1990.

[5] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[6] C. Dwork. Lattices and their application to cryptography. Lecture Notes, 1998. `http://theory.stanford.edu/~csilvers/cs359/`.

[7] P. Q. Nguyen. *Hermite's Constant and Lattice Algorithms*. Information Security and Cryptography. Springer, 2009.

[8] T. Laarhoven, J. van de Pol, and B. Weger. Solving hard lattice problems and the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2012/533, 2012. `http://eprint.iacr.org/`.

[9] Geometry of numbers: Determinant of the lattice and the fundamental parallelepiped. Number Theory Reading Group, 2008. `http://numbertheoryreadinggroup.wordpress.com/2008/04/24/`.

[10] Lattice challenge. University of Technology of Darmstadt. `http://www.latticechallenge.org/`.

[11] Department of Computer Science and Engineering, University of California. *The Geometry of Lattice Cryptography*, 2011.

[12] J. van de Pol. Lattice-based criptography. Master's thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, 2011.

[13] M. Ajtai. The shortest vector problem in l2 is np-hard for randomized reductions (extended abstract). In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 10–19. ACM, 1998.

[14] J. Valença. *Introdução à Segurança da Informação, Códigos e Criptografia*. Departamento de Informática, Universidade do Minho, 2012.

[15] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, 2002.

[16] C. Schnorr, H. Hörner, and J. Wolfgang. Attacking the chor-rivest cryptosystem by improved lattice reduction. Springer-Verlag, 1995.

[17] N. Gama, P. Nguyen, and O. Regev. Lattice enumeration using extreme pruning. In *Advances in Cryptology - Proceedings of EUROCRYPT '10*, volume 6110 of *LNCS*. Springer, 2010.

[18] D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10. Society for Industrial and Applied Mathematics, 2010.

[19] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *In STOC*. ACM, 2001.

[20] P. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. *J. of Mathematical Cryptology*, 2(2), 2008.

[21] X. Wang, C. Tian M. Liu, and J. Bi. Improved nguyen-vidick heuristic sieve algorithm for shortest vector problem. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM, 2011.

[22] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. pages 284–293, 1997.

[23] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. pages 112–131. Springer-Verlag, 1996.

[24] D. Micciancio and O. Regev. Lattice based cryptography. In *Encyclopedia of Cryptography and Security*. 2005.

[25] V. Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In *ASIACRYPT*, pages 598–616, 2009.

[26] D. Micciancio. Generalized compact knapsaks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007.

[27] V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. Swifft: A modest proposal for fft hashing.

[28] V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures.