Maiara Francine Bollauf

Códigos, reticulados e aplicações em criptografia

CAMPINAS
2015

UNICAMP

UNIVERSIDADE ESTADUAL DE CAMPINAS

Instituto de Matemática, Estatística
e Computação Científica

MAIARA FRANCINE BOLLAUF

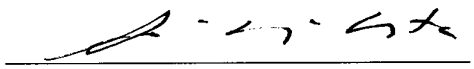
CÓDIGOS, RETICULADOS E APLICAÇÕES EM CRIPTOGRAFIA

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestra em matemática aplicada.

Orientadora: Sueli Irene Rodrigues Costa

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA DISSERTAÇÃO DEFENDIDA PELA ALUNA MAIARA FRANCINE BOLLAUF, E ORIENTADA PELA PROFA. DRA. SUELI IRENE RODRIGUES COSTA.

Assinatura



CAMPINAS
2015

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

B638c

Bollauf, Maiara Francine, 1991-
Códigos, reticulados e aplicações em criptografia / Maiara Francine Bollauf. – Campinas, SP : [s.n.], 2015.

Orientador: Sueli Irene Rodrigues Costa.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Códigos corretores de erros (Teoria da informação). 2. Teoria dos reticulados. 3. Criptografia. I. Costa, Sueli Irene Rodrigues,1949-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Codes, lattices and applications in cryptography
Palavras-chave em inglês:
Error-correcting codes (Information theory)
Lattice theory
Cryptography
Área de concentração: Matemática Aplicada
Titulação: Mestra em Matemática Aplicada
Banca examinadora:
Sueli Irene Rodrigues Costa [Orientador]
João Eloir Strapasson
Julio César López Hernández
Data de defesa: 30-03-2015
Programa de Pós-Graduação: Matemática Aplicada

Dissertação de Mestrado defendida em 30 de março de 2015 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.

Prof.(a). Dr(a). SUELI IRENE RODRIGUES COSTA

Prof.(a). Dr(a). JULIO CÉSAR LÓPEZ HERNÁNDEZ

Prof.(a). Dr(a). JOÃO ELOIR STRAPASSON

Abstract

This dissertation has the aim of approaching the theories of codes and lattices and their recent use to propose public key cryptosystems in the so called post-quantum cryptography. In the first chapter we introduce the theory of error correcting codes, including definitons and particularly properties of larged used codes such as Hamming codes, cyclic codes, BCH codes and Goppa codes. In the second chapter we present a characterization of two hard (NP-complete) problems based on the code structure which are the general decoding problem (GDP) and the syndrome decoding problem (SDP), which underlie algorithms based on the difficulty of solving them, as the McEliece and the Niederreiter cryptosystems. Chapter 3 is devoted to lattice theory, its basic concepts and the characterization of hard problems in this structure – the shortest vector problem (SVP) and the closest vector problem (CVP). We also present a way to obtain lattices from linear codes using the so called Construction A and some tools of geometry of numbers to explain methods to evaluate the implementation of encryption schemes based on lattices. In the last chapter, we describe algorithms of this subarea of cryptography, such as GGH and NTRU. All these fundaments give support to recent research topics in cryptography, intended not only to search for secure systems that will probably resist to the introduction of quantum computers but also to be more efficient considering the the evolution of the classical computers.

Keywords: Error correcting codes (Information theory), Lattice theory, Cryptography.

Resumo

Essa dissertação possui como objetivo abordar as teorias de códigos e de reticulados e a aplicação recente destas na proposição de sistemas criptográficos que fazem o uso de chaves públicas dentro da chamada criptografia pós-quântica. No primeiro capítulo introduzimos a teoria dos códigos corretores de erros, incluindo definições e particularmente propriedades de códigos bastante utilizados como os de Hamming, códigos cíclicos, códigos BCH e códigos de Goppa. No segundo capítulo apresentamos a caracterização de dois problemas difíceis (NP-completos) baseados na estrutura de códigos que são o problema de decodificação geral (GDP) e o problema de decodificação por síndromes (SDP), os quais fundamentam algoritmos baseados na dificuldade de resolvê-los, como os criptossistemas de McEliece e Niederreiter. O Capítulo 3 é dedicado à teoria de reticulados, seus conceitos básicos e à caracterização dos problemas difíceis de se determinar nesta estrutura - o problema do vetor mais curto (SVP) e o problema do vetor mais próximo (CVP). Apresentamos também uma forma de se obter reticulados a partir de códigos lineares, utilizando a chamada Construção A e ferramentas de geometria dos números para explicar métodos que avaliam a implementação da criptografia baseada em reticulados. No último capítulo descrevemos algoritmos desta subárea da criptografia , como os criptossistemas GGH e NTRU. Todos esses fundamentos embasam temas muito recentes de pesquisa em criptografia, que visam não somente a busca de sistemas que possivelmente resistirão à implementação de computadores quânticos mas que sejam mais eficientes na evolução prevista para os computadores clássicos atuais.

Palavras-chave: Códigos corretores de erros (Teoria da informação), Teoria dos reticulados, Criptografia.

Sumário

"Foi o tempo que dedicaste à tua rosa
que fez tua rosa tão importante"
Antoine Saint-Exupéry

Agradecimentos

Agradeço:

- à Deus por me iluminar, me dar saúde e por ser tão generoso quanto às minhas inúmeras preces durante estes dois anos de mestrado;

- à minha família, Rogério, Marisa e Rodrigo, por terem me deixado ir e por terem dividido comigo, dia após dia, esse sonho que exigiu tantas renúncias;

- à professora Sueli por ter confiado em mim desde o início e por ter dividido comigo suas experiências. Agradeço ainda por ter me ensinado tanto sobre matemática, sobre academia e sobre amizade;

- ao Samuel por ter sido a minha fortaleza e fonte de inspiração;

- aos professores João, Julio, Grasiele e José Plínio por terem aceitado o convite para compor a banca e contribuírem com o trabalho. Em especial, agradeço ao professor Julio por ter me incentivado a estudar criptografia e pela sua imensa ajuda na elaboração deste;

- à família campineira, Carol e Marina, por me ajudarem a constituir, mesmo tão longe de casa, um lar divertido e aconchegante;

- às amigas catarinenses, Angela, Eloá, Jaqueline, Aruana, Emanuele, Débora e Marília por sua amizade gratuita e sempre presente;

- às amizades paulistas (ou nem tão paulistas assim), Rachel, Enio, Maria Angélica, Giselle, Eleonesio, Nelson, Ivan e aos integrantes do laboratório de pesquisa do IMECC/UNICAMP, pela cumplicidade e troca de experiências;

- ao Antônio, Aurelio e Rami pelas suas respectivas contribuições neste trabalho;

- à CAPES pelo suporte financeiro.

Lista de Ilustrações

Lista de Tabelas

Lista de Algoritmos

Introdução

Na era digital, estamos constantemente enviando, recebendo e armazenando dados, preocupados com a segurança e confiabilidade da informação. A área de pesquisa interessada neste estudo é a chamada *Teoria da informação* e possui como marco o pioneiro trabalho de C.E. Shannon, de 1948, intitulado "*A Mathematical Theory of Comunication*".

Existem interferências que podem ocorrer durante o processo de envio ou recebimento de dados, que podem ser causadas por problemas no canal ou por falha humana, como a digitação, por exemplo, e tais erros são chamados de ruídos. A fim de garantir que a mensagem chegue ao seu destino de maneira correta surgiram os códigos corretores de erros, cuja principal função é acrescentar bits à mensagem transmitida de maneira que seja possível a detecção e correção dos erros.

Mas, o que podemos falar em relação à segurança da informação? Como garantir que uma mensagem chegue ao destino de maneira sigilosa? Naturalmente existem muitas pessoas, governos e empresas interessados nessas respostas, como bancos, criadores de websites ou cientistas. Surge então a criptografia: do grego *kryptós* que significa "escondido" e mais *graphé*, "escrita". A criptografia era inicialmente definida como a ciência que estudava métodos para que somente o destinatário legítimo de uma mensagem enviada pudesse interpretá-la de maneira correta. Contudo, a criptografia moderna envolve mais do que isso, incluindo também o estudo de técnicas matemáticas para a segurança digital, sistemas e computação contra ataques [41].

O sistema criptográfico mais conhecido é o RSA, introduzido em [60], que é baseado na decomposição de números grandes em fatores primos. A criptografia RSA é bastante usada para a segurança na internet, como por exemplo na transmissão de emails ou compras online. Outros sistemas criptográficos eficientes, como o baseado em curvas elípticas também tem sido largamente utilizados em diferentes contextos. Em 1996, o físico norte americano Peter Shor demonstrou que o problema da fatoração em primos, bem como outros problemas difíceis que dão suporte aos sistemas criptográficos atualmente adotados, poderiam ser resolvidos em tempo polinomial, se fosse utilizado um computador quântico, ou seja, um computador que utiliza as leis da mecânica quântica em seu processamento de dados.

Motivados pelo grande desafio de que um computador quântico poderia decifrar códigos atuais em tempo factível, iniciaram-se as pesquisas em torno da chamada criptografia pós-quântica [11]. Esta inclui a procura por sistemas criptográficos baseados em códigos corretores de erros e reticulados e a apresentação destes a partir do suporte matemático em suas formulações é o objetivo deste trabalho. Por mais que a computação quântica ainda seja um projeto, os estudos acerca desse tema vem sendo muito promissores e têm trazido importantes resultados para a própria

computação clássica.

Assim como todo sistema criptográfico, os algoritmos da criptografia pós-quântica também se baseiam em problemas de difícil resolução, como a fatoração em números primos no caso da criptografia RSA. Dois desses problemas na estrutura de códigos corretores de erros são o problema de decodificação geral e o problema de decodificação por síndromes e estes fundamentam a subárea da criptografia baseada em códigos.

A teoria de reticulados dá suporte a outra subárea de pesquisa em criptografia. De maneira simples, um reticulado é um subgrupo aditivo discreto do $\mathbb{R}^n$ e tal teoria vem sendo aplicada em diversas áreas [69], como em problemas de empacotamento esférico, por exemplo. Também localizamos no estudo de reticulados dois problemas difíceis, o problema do vetor mais curto e o problema do vetor mais próximo, que são o ponto de partida para a chamada criptografia baseada em reticulados.

É possível também relacionar códigos e reticulados por meio da chamada Construção A, na qual partimos de um código linear e obtemos a partir deste um reticulado chamado de reticulado $q-$ário. Uma de nossas perspectivas futuras de pesquisa é obter possíveis aperfeiçoamentos de sistemas criptográficos através da análise dessa conexão.

A abordagem desta dissertação partiu de uma significativa pesquisa bibliográfica onde destacamos como principais referências [55], [17], [31], [36] e [69].

Estruturamos este trabalho em quatro capítulos da seguinte maneira: no primeiro deles introduzimos códigos corretores de erros, bem como exemplos que marcaram o desenvolvimento da teoria de códigos e vem sendo utilizados em criptografia. Entre estes, citamos os códigos de Hamming, códigos cíclicos, códigos BCH e códigos de Goppa.

No segundo capítulo apresentamos sistemas criptográficos baseados em códigos, a dizer, o criptossistema de McElice, o criptossistema de Niederreiter e o esquema de assinatura CFS. Discutiremos aspectos práticos e teóricos da aplicação destes algoritmos, bem como a sua relevância no progresso da criptografia pós-quântica.

No terceiro capítulo abordamos a teoria de reticulados, incluindo suas principais propriedades e possibilidades de aplicação. Além disso, apresentamos a Construção A, descrevendo também uma maneira de decodificar reticulados usando tal construção. Buscamos ainda, por meio de ferramentas de geometria dos números, obter cotas para a resolução dos dois problemas difíceis em reticulados, já citados anteriormente.

O quarto e último capítulo traz alguns criptossistemas baseados em reticulados, como o GGH, que é o sistema mais intuitivo, e ainda o NTRU, que é considerado por diversos autores um sistema promissor. Buscamos nesse capítulo também explorar algumas vantagens e desvantagens da aplicação desses criptossistemas.

Capítulo 1

Códigos corretores de erros

O estudo de códigos corretores de erros integra diversas áreas do conhecimento, como a matemática, a computação e algumas engenharias. O propósito desse estudo é o de intervir todas as vezes que queremos transmitir uma informação, a fim de que a mensagem chegue ao seu destino com o mínimo de erros possível, erros esses que podem ser causados por algum tipo de interferência.

Este primeiro capítulo possui o intuito de apresentar, exemplificar e detalhar aspectos relevantes a algumas classes de códigos binários e $q-$ários em geral, particularmente os códigos de Hamming, códigos cíclicos, códigos BCH, códigos de Reed-Solomon e códigos de Goppa. Tais informações fornecem-nos ferramentas para a elaboração de um sistema criptográfico eficiente e resistente às constantes evoluções computacionais.

Para a elaboração dessa redação, utilizamos como principais referências [42], [31], [43], [29], [61], [39] e [50].

1.1 Conceitos introdutórios

Uma publicação que é considerada um marco para a teoria dos códigos corretores de erros e para a teoria da informação é o artigo escrito por C.E. Shannon em 1948, intitulado *"A Mathematical Theory of Communication"* e concebido durante o seu trabalho no Laboratório Bell de Tecnologia, localizado em Nova Jersey, nos Estados Unidos.

Acredita-se que a construção dos códigos teve inspiração nos idiomas, onde as palavras podem ser interpretadas como sequências de letras. No caso do alfabeto latino que utilizamos como base para a língua portuguesa, tal sequência combina 26 elementos formando uma palavra.

O objetivo essencial dos estudos em códigos, por sua vez, é codificar um dado original adicionando uma informação redundante, de modo que, ao receber o sinal modificado por um possível "ruído"(interferências eletromagnéticas ou erros humanos, como a digitação, por exemplo), o receptor possa recuperar a mensagem original com o mínimo de perdas.

Exemplo 1.1.1. *Vamos considerar um código binário de tripla repetição. Considere um robô que pode se mover em quatro direções: leste(L), norte(N), oeste(O) e sul(S). Tais comandos podem ser codificados como elementos de* $\{0,1\} \times \{0,1\}$ *associando a L, N, O, e S respectivamente as sequências* 00*,* 01*,* 10 *e* 11*. Esse código é chamado de código-fonte.*

Introduzindo as posições redundantes de modo a repetir três vezes o código inicial, temos a seguinte associação:

$$\begin{aligned} L &\longrightarrow 00 \mapsto 000000 \\ N &\longrightarrow 01 \mapsto 010101 \\ O &\longrightarrow 10 \mapsto 101010 \\ S &\longrightarrow 11 \mapsto 111111. \end{aligned}$$

O código introduzido nessa recodificação é chamado de código-canal. Observe que esse código corrige um erro e pode detectar até dois. De fato, digamos que a palavra recebida seja 111010*. Comparando com o código-canal, verificamos que o erro está no segundo elemento e, automaticamente, corrigimos para* 101010*. Assim, ocorrendo apenas um erro em qualquer uma das palavras, podemos identificá-lo e corrigi-lo por meio de comparações.*

Já se a palavra recebida fosse 011010*, o erro seria identificado, mas não corrigido, pois:*

$$011010 = \begin{cases} 101010 & \textit{se foram cometido 2 erros} \\ 111111 & \textit{se foram cometidos 3 erros} \\ 000000 & \textit{se foram cometidos 3 erros} \\ 010101 & \textit{se foram cometidos 4 erros}, \end{cases}$$

e detectamos que no mínimo dois erros foram cometidos.

Definição 1.1.2. *Um* ***código*** $q-$***ário*** *de comprimento n é um subconjunto $\mathcal{C} \subseteq A^n$, onde A é um subconjunto de q elementos chamado de alfabeto. Os elementos de $\mathcal{C}$ são chamados de* ***palavras-código****.*

Usualmente o alfabeto A é tomado como sendo $\mathbb{Z}_q$ [1], que é o anel dos inteiros módulo q, ou ainda como $\mathbb{F}_q$, que é um corpo finito com q elementos, também chamado de corpo de Galois [66]. De modo particular, consideramos nos exemplos e grande parte da teoria deste trabalho $A = \mathbb{Z}_q$, com q primo, o que garante a existência de elemento inverso para a operação de multiplicação e além disso $\mathbb{Z}_q$ é isomorfo à $\mathbb{F}_q$ neste caso. Observamos ainda que, no Exemplo 1.1.1, consideramos $\mathcal{C} \subseteq \mathbb{Z}_2^6$.

Definição 1.1.3. *Seja $\mathcal{C}$ um código corretor de erros de comprimento n. A* ***taxa de informação*** *é o parâmetro que nos permite comparar a eficiência de dois códigos de diferentes tamanhos e ela é definida por:*

$$R(\mathcal{C}) = \frac{\log_q m}{n},$$

donde $m = |\mathcal{C}|$.

[1] [22] O anel $\mathbb{Z}_q = \{\bar{0}, \bar{1}, \ldots, \overline{q-1}\}$ é o chamado anel dos inteiros módulo q, onde a classe de $\bar{a}$ é formada pelos inteiros cuja divisão por q tem resto a. Esse anel tem as operações adição e multiplicação de suas classes induzidas das operações nos inteiros e bem definidas, conforme:

Adição: $\bar{a} + \bar{b} = \overline{a+b}$.

Multiplicação: $\bar{a} \cdot \bar{b} = \overline{a \cdot b}$.

Seja $\mathbb{F}_q$ um corpo finito com q elementos. Então, para cada número natural n temos um $\mathbb{F}_q-$espaço vetorial ${\mathbb{F}_q}^n$ de dimensão n.

Definição 1.1.4. *Um código $\mathcal{C}$ será chamado de* ***código linear*** *se for um subespaço vetorial de $\mathbb{F}_q^n$.*

Observamos que, se q não for primo, então $\mathbb{F}_q \neq \mathbb{Z}_q$. Definimos dessa forma um código linear $\mathcal{C}$ na estrutura do anel $\mathbb{Z}_q$ como:

Definição 1.1.5. *Seja R um anel finito e comutativo com identidade. Um* ***código linear*** *$\mathcal{C}$ de comprimento n sobre R é um submódulo do $R-$módulo livre* [2] *R^n.*

Assim, quando estivermos em $\mathbb{F}_q^n$ usamos a Definição 1.1.4 para caracterizar um código linear, já se estivermos sobre $\mathbb{Z}_q^n$, com q não primo, usamos a Definição 1.1.5.

Se o conjunto $\mathbb{F}_q$ é um corpo finito com q elementos e $\mathcal{C}$ é um espaço vetorial de dimensão k de $\mathbb{F}_q^n$, então $|\mathcal{C}| = q^k$.

Seja $\alpha = \{a_1, \ldots, a_k\}$ uma base para $\mathcal{C}$ sobre $\mathbb{F}_q$. Logo, todo elemento de $\mathbb{F}_q$ se escreve de maneira única:

$$\lambda_1 a_1 + \cdots + \lambda_k a_k,$$

onde cada $\lambda_i \in \mathbb{F}_q$ e $i = 1, \ldots, n$.

Se contarmos todas as combinações possíveis, vem que $|\mathcal{C}| = q^k$. Além disso, temos pela Definição 1.1.3 que:

$$R(\mathcal{C}) = \frac{\log_q q^k}{n} = \frac{k}{n}.$$

Definição 1.1.6. *Seja A um conjunto qualquer e n um número natual. Uma função $d : A^n \times A^n$ é chamada de* ***métrica*** *ou função distância se e somente se satisfaz as seguintes condições:*

i) $d(x, y) \geq 0$ *e* $d(x, y) = 0 \Leftrightarrow x = y$;

ii) $d(x, y) = d(y, x)$;

iii) $d(x, y) + d(y, z) \geq d(x, z)$.

1.2 Distância de Hamming e distância de Lee

Para corrigir erros de códigos em geral, é imprescindível que estejamos munidos de uma noção de distância entre vetores. Nesta seção, apresentamos duas métricas comuns na teoria de códigos: a métrica de Hamming e a métrica de Lee ou métrica da soma, ambas nomeadas em homenagem aos seus autores.

Definição 1.2.1. *Dados dois pontos $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$, ambos pertencentes a $\mathbb{F}_q^n$, definimos a* ***distância de Hamming*** *entre x e y como*

$$d_{Hamming}(x, y) = |\{i : x_i \neq y_i, \quad 1 \leq i \leq n\}|.$$

[2] [33] Um $R-$módulo V é dito livre se admitir uma base, ou seja, um conjunto de vetores linearmente independentes que gera V. Se essa base for finita com n elementos, então V é um $R-$módulo livre com n geradores.

Ou seja, a distância de Hamming conta o número de símbolos diferentes entre duas palavras-código. Por simplicidade de notação, vamos admitir no decorrer do texto, $d_{Hamming}(x, y) = d(x, y)$, ao nos referirmos a distância de Hamming.

Proposição 1.2.2. *[31] A distância de Hamming define uma métrica.*

Demonstração. Considere $x = (x_1, x_2, \ldots, x_n)$, $y = (y_1, y_2, \ldots, y_n)$ e $z = (z_1, z_2, \ldots, z_n)$ e a distância de Hamming conforme explicitada na Definição 1.2.1. Devemos provar que:

(i) $d(x, y) \geq 0$ e $d(x, y) = 0 \Leftrightarrow x = y$

Observe que:

$$d(x, y) = |\{i : x_i \neq y_i\}| > 0, \quad \text{se } x \neq y \quad (x_i \neq y_i, \text{ para algum } i)$$
$$d(x, y) = |\{i : x_i \neq y_i\}| = 0, \quad \text{se e somente se } x = y \quad (x_i = y_i, \text{ para todo } i).$$

(ii) $d(x, y) = d(y, x)$

Note que:

$$d(x, y) = |\{i : x_i \neq y_i\}| = |\{i : y_i \neq x_i\}| = d(y, x).$$

(iii) $d(x, y) + d(y, z) \geq d(x, z)$

Aqui temos dois casos para serem analisados:

Caso 1) $x_i = z_i \quad \forall i \Longrightarrow d(x, z) = 0.$

Então, segue de (i) que:

$$d(x, y) + d(y, z) \geq 0 = d(x, z)$$

Caso 2) $x_i \neq z_i$ para algum $i \Longrightarrow d(x, z) > 0.$

Note que neste caso não podemos ter $x_i = y_i$ e $y_i = z_i$, para este dado i senão $x_i = z_i$ o que contraria a nossa hipótese. Daí, podemos afirmar que:

$$x_i \neq y_i, \text{ para este } i \text{ ou}$$
$$y_i \neq z_i, \text{ para este } i.$$

Disso podemos concluir que a contribuição das i-ésimas coordenadas de x e y para $d(x, y)$ é igual a zero ou igual a um, assim como a contribuição das i-ésimas coordenadas de y e z para $d(y, z)$, excluindo apenas o fato de que elas não podem se anular simultaneamente. Então, a contribuição das i-ésimas coordenadas a $d(x, y) + d(y, z)$ é maior ou igual a um. Daí, segue que $d(x, y) + d(y, z) \geq d(x, z)$, como queríamos.

Como as três propriedades estão satisfeitas, segue que a distância de Hamming define uma métrica. □

Definição 1.2.3. *Seja $a \in \mathbb{F}_q^n$ e $r > 0$ um número real. Definimos a **esfera** e a **bola** de centro em a e raio r como sendo, respectivamente, os conjuntos:*

$$S(a, r) = \{x \in \mathbb{F}_q^n : \quad d(x, a) = r\},$$
$$B(a, r) = \{x \in \mathbb{F}_q^n : \quad d(x, a) \leq r\}.$$

Definição 1.2.4. *Seja $\mathcal{C}$ um código. A* ***distância mínima*** *de $\mathcal{C}$ é dada por:*

$$d_{min}(\mathcal{C}) = d(\mathcal{C}) = min\{d(x,y) : x, y \in \mathcal{C}, \quad x \neq y\}.$$

Definição 1.2.5. *Dado $x \in \mathcal{C} \subset {\mathbb{F}_q}^n$, define-se o* ***peso*** *de x como sendo o número inteiro*

$$\omega(x) = |\{i : \quad x_i \neq 0\}|$$

Ou, de maneira equivalente:

$$\omega(x) = d(x, 0),$$

onde d é a distância de Hamming.

Definição 1.2.6. *Chama-se* ***peso mínimo do código linear*** *$\mathcal{C}$ o número*

$$\omega(\mathcal{C}) := min\{\omega(x) : x \in \mathcal{C}, \quad x \neq 0\}.$$

Exemplo 1.2.7. *Considere o código $\mathcal{C} = \{0000, 1011, 0110, 1101\}$ um subespaço vetorial de $\mathbb{F}_2^4$. O conjunto $\beta = \{1011, 1101\}$ é uma base para $\mathcal{C}$. Temos que*

$$\omega(1011) = 3, \qquad \omega(0110) = 2, \qquad \omega(1101) = 3,$$

e $d_{min}(\mathcal{C}) = 2$.

Proposição 1.2.8. *[31] Seja $\mathcal{C} \subset {\mathbb{F}_q}^n$ um código linear com distância mínima d. Temos que:*

i) $d(x, y) = d(0, x - y) = \omega(x - y) \qquad \forall x, y \in \mathbb{F}_q^n$.

ii) $d = \omega(\mathcal{C})$, *onde d corresponde à distância mínima.*

Demonstração. Para demonstrar (i), observe que:

$$\omega(x - y) = |\{i : x_i - y_i \neq 0\}| = |\{i : x_i \neq y_i\}| = d(x, y).$$

A afirmação em (ii) decorre do fato que, para todo par de elementos $x, y \in \mathcal{C}$, com $x \neq y$, temos que $z = x - y \in \mathcal{C} - \{0\}$ e $d(x, y) = \omega(z)$. Tome, então, $x_1, y_1 \in \mathcal{C}$ tais que $d(x_1, y_1) = d$, onde d é a distância mínima. Neste caso, teremos que $d(x_1, y_1) = d = \omega(\mathcal{C})$. □

A medida para a capacidade de correção de erros de um código linear $\mathcal{C}$ está relacionada à distância mínima de Hamming entre duas palavras-código distintas, como veremos a seguir.

Definição 1.2.9. *O* ***raio de empacotamento*** *de um código $\mathcal{C}$ é o maior número real t tal que as bolas de raio t e centro nas palavras do código são disjuntas.*

Definição 1.2.10. *Se uma palavra distorcida é recebida, ela é decodificada como a palavra do código que está mais próxima dela. Essa proximidade vem da noção de distância baseada na Definição 1.1.6 e à essa técnica damos o nome de* ***princípio de máxima verossimilhança****.*

A Figura 1.1 ilustra o fato de que o raio de empacotamento é o maior inteiro que é estritamente menor que a metade do valor da distância mínima.

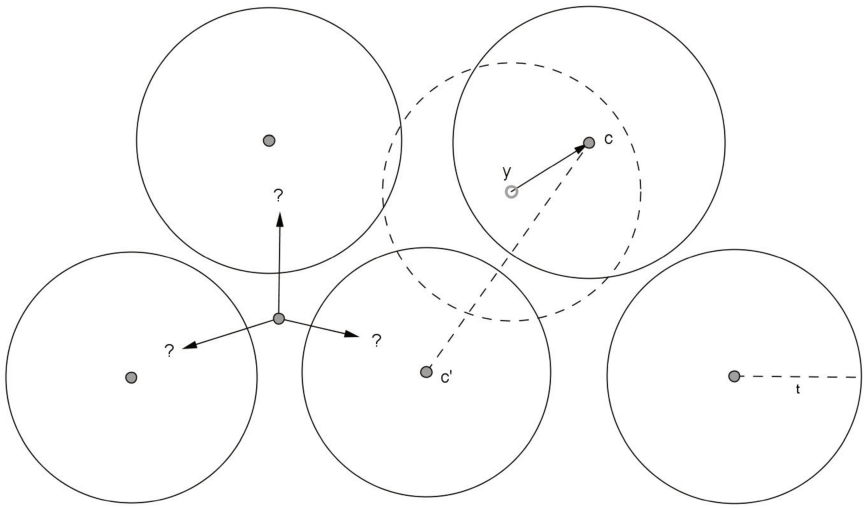


Figura 1.1: O princípio da máxima verossimilhança para decodificação

Fonte: [12], p.14.

Assim, como formalizado no resultado a seguir, é possível corrigir até $t = \lfloor \frac{d-1}{2} \rfloor$ erros em um código linear usando a decodificação por meio do princípio da máxima verossimilhança da seguinte maneira:

(1) Um vetor $y \in \mathbb{F}_q^n$ é dito decodificado em uma palavra $c \in \mathcal{C}$ se c é a palavra-código que está mais próxima à y com respeito à uma dada distância (consideramos a de Hamming nesse caso). Notamos que se existem diversas palavras-código c com essa propriedade, escolhemos uma delas de modo aleatório.

(2) Se uma palavra-código $c \in \mathcal{C}$ foi enviada e não ocorreram mais do que t erros durante a transmissão, o vetor recebido é

$$y = c + e \in \mathbb{F}_q^n,$$

onde e denota o vetor erro e satisfaz $d(c, y) = d(e, 0) \leq t$. Dessa forma, c é o único elemento de $\mathcal{C}$ que mora na bola de raio t ao redor de y. O princípio da máxima verossimilhança devolve c e obtemos a decodificação correta.

Proposição 1.2.11. *[12] Um código pode corrigir até* $\lfloor \frac{d-1}{2} \rfloor$ *erros e detectar até* $d-1$ *erros se sua distância de Hamming mínima for* d.

Demonstração. Suponhamos que a distância mínima do código $\mathcal{C}$ é d. Queremos mostrar, inicialmente, que $\mathcal{C}$ corrige até $\lfloor\frac{d-1}{2}\rfloor$ erros. Seja x um vetor recebido com $d(x,c) \leq \lfloor\frac{d-1}{2}\rfloor$ para algum $c \in \mathcal{C}$, basta provar que $d(x,c') > \lfloor\frac{d-1}{2}\rfloor$ para todo $c' \in \mathcal{C},\ c \neq c'$. Temos:

$$d \leq d(c,c') \leq d(c,x) + d(x,c') \Rightarrow d(x,c') \geq d - d(x,c) > 2\lfloor\frac{d-1}{2}\rfloor - \lfloor\frac{d-1}{2}\rfloor = \lfloor\frac{d-1}{2}\rfloor,$$

como queríamos.

Vamos demonstrar agora que $\mathcal{C}$ detecta até $d-1$ erros. Suponhamos que numa palavra recebida x temos os símbolos de u posições trocados com relação à uma palavra $c \in \mathcal{C}$, com $1 \leq u \leq d-1$. Portanto, $x \neq c'$ para qualquer outra palavra $c' \in \mathcal{C}$, pois $d(x,c') \geq 1$. De fato,

$$d \leq d(c,c') \leq d(c,x) + d(x,c') \Rightarrow d(x,c') \geq d - d(c,x) \geq d - (d-1) = 1.$$

□

Definição 1.2.12. *Dado um conjunto A e uma função distância $d : A^n \times A^n \longrightarrow \mathbb{R}$, dizemos que $\varphi : A^n \longrightarrow A^n$ é uma **isometria** segundo d se for uma aplicação que satisfaz:*

$$d(\varphi(a), \varphi(b)) = d(a,b) \qquad \forall a, b \in A^n.$$

Definição 1.2.13. *Dois códigos $\mathcal{C}$ e $\mathcal{C}'$ em $\mathbb{F}_q^n$ são ditos **equivalentes** segundo uma métrica d se existir uma isometria $\alpha : A^n \to A^n$ tal que $\mathcal{C}' = \alpha(\mathcal{C})$.*

Proposição 1.2.14. *[31] Toda isometria de $\mathbb{F}_q^n$ é uma bijeção de $\mathbb{F}_q^n$.*

Demonstração. Observe que uma isometria é claramente injetiva. Seja $\varphi : \mathbb{F}_q^n \to \mathbb{F}_q^n$ uma isometria. Supondo que $\varphi(a) = \varphi(b)$, segue que $d(a,b) = d(\varphi(a), \varphi(b)) = d(\varphi(a), \varphi(a)) = 0 \Rightarrow d(a,b) = 0 \Rightarrow a = b$.

Como $\mathbb{F}_q^n$ é um conjunto finito e a isometria φ é injetora, segue imediatamente que ela também é sobrejetora. □

Proposição 1.2.15. *[12] Toda translação é uma isometria na métrica de Hamming, ou seja, dados $u, v, w \in \mathbb{F}_q^n$ então*

$$d(u,v) = d(u+w, v+w).$$

Definição 1.2.16. *Dados $a, b \in \mathbb{Z}_q$ definimos a **distância de Lee** ou **distância da soma** como sendo:*

$$d_{Lee}(\bar{a}, \bar{b}) = min\{|a-b|, q - |a-b|\}.$$

Para ilustrar a Definição 1.2.16 consideramos, por exemplo, que em $\mathbb{Z}_{11} = \mathbb{F}_{11}$, temos que $d_{Lee}(\bar{1}, \bar{4}) = 3$ e $d_{Lee}(\bar{1}, \bar{10}) = 2$. Geometricamente, se colocarmos as classes de $\mathbb{Z}_q$ como os vértices de um polígono regular de q lados, a distância de Lee entre dois elementos será o menor número de arestas que conectam estes vértices, como ilustrado na Figura 1.2, considerando $\mathbb{Z}_{11}$.

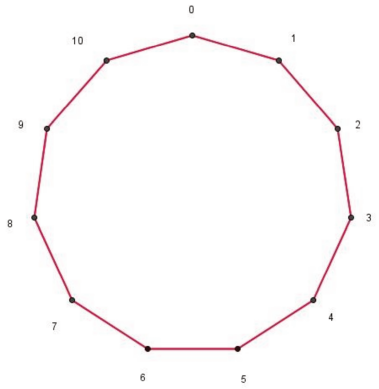


Figura 1.2: Representação geométrica de $\mathbb{Z}_{11}$ com a distância de Lee

Fonte: [42], p.24.

Definição 1.2.17. *Dados* $(\bar{a_1}, \bar{a_2}, \ldots, \bar{a_n}), (\bar{b_1}, \bar{b_2}, \ldots, \bar{b_n}) \in {\mathbb{Z}_q}^n$. *A* ***distância de Lee****, neste caso, é definida como sendo a soma das distâncias nas coordenadas:*

$$d_{Lee}((\bar{a_1}, \bar{a_2}, \ldots, \bar{a_n}), (\bar{b_1}, \bar{b_2}, \ldots, \bar{b_n})) = \sum_{i=1}^{n} d_{Lee}(\bar{a_i}, \bar{b_i}).$$

Uma observação pertinente neste caso é que se $q = 2$ ou $q = 3$ a distância de Lee e de Hamming são equivalentes. Para $q = 2$, por exemplo, considere $\overline{x} = (\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$ e $\overline{y} = (\overline{y_1}, \overline{y_2}, \ldots, \overline{y_n})$, onde cada coeficiente $\overline{x_i}, \overline{y_i} \in \mathbb{Z}_2 \ \ \forall\ i$ temos que:

$$\begin{aligned} d_{Lee}(\overline{x}, \overline{y}) &= \sum_{i=1}^{n} d_{Lee}(\overline{x_i}, \overline{y_i}) \\ &= \sum_{i=1}^{n} min\{|x_i - y_i|, 2 - |x_i - y_i|\}. \end{aligned}$$

Note que, $min\{|x_i - y_i|, 2 - |x_i - y_i|\} = \begin{cases} 0, & \text{se} \quad x_i = y_i \\ 1, & \text{se} \quad x_i \neq y_i. \end{cases}$

Com isso, segue que:

$$\begin{aligned} d_{Lee}(\overline{x}, \overline{y}) &= |\{i : x_i \neq y_i, \quad 1 \leq i \leq n\}| \\ &= d_{Hamming}(x, y). \end{aligned}$$

De maneira análoga, podemos demonstrar que, se estamos sob $\mathbb{Z}_3$, as distâncias de Lee e Hamming também coincidem.

Podemos definir os conceitos de esfera e bola, como fizemos com a distância de Hamming.

Definição 1.2.18. *Seja* $a \in {\mathbb{Z}_q}^n$ *e* $r > 0$ *um número real. Definimos a* ***esfera*** *e a* ***bola*** *de centro em* a *e raio* r*, munidas com a distância de Lee, como sendo, respectivamente, os conjuntos:*

$$S_{Lee}(a,r) = \{x \in {\mathbb{Z}_q}^n : \quad d_{Lee}(\overline{x},\overline{a}) = r\},$$
$$B_{Lee}(a,r) = \{x \in {\mathbb{Z}_q}^n : \quad d_{Lee}(\overline{x},\overline{a}) \leq r\}.$$

Para $n = 2$, os pontos de ${\mathbb{Z}_q}^2$ são correspondentes aos vértices de uma malha quadriculada desenhada num quadrado de lado q. Os bordos deste quadrado devem ser identificados e, portanto, esta malha estára sobre um toro. A distância de Lee entre dois vértices é a distância do grafo, ou seja, o número mínimo de arestas nesta malha para ir de um vértice a outro.

A Figura 1.3 nos mostra a representação de ${\mathbb{Z}_5}^2$ e podemos observar que $d_{Lee}((\overline{2},\overline{3}),(\overline{4},\overline{4})) = 3$, por exemplo. De fato, fazendo os cálculos:

$$\begin{aligned} d_{Lee}(\overline{2},\overline{3}),(\overline{4},\overline{4}) &= d_{Lee}(\overline{2},\overline{4}) + d_{Lee}(\overline{3},\overline{4}) \\ &= 2 + 1 = 3. \end{aligned}$$

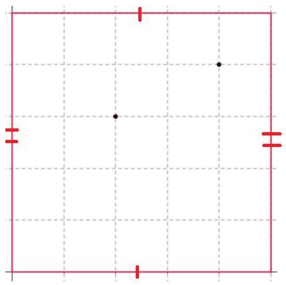


Figura 1.3: Representação geométrica de ${\mathbb{Z}_5}^2$ com a distância de Lee

Fonte: [42], p.24.

Definição 1.2.19. *Um código $\mathcal{C} \in \mathbb{F}_q^n$ com distância mínima d (de Hamming ou de Lee) é dito $t-$**perfeito**, com $t = \lfloor \frac{d-1}{2} \rfloor$ se e somente se $\bigcup_{a \in \mathcal{C}} B(a,t) = \mathbb{F}_q^n$, ou seja, a reunião de bolas disjuntas centradas em palavras do código com raio t recobre todo $\mathbb{F}_q^n$.*

Definição 1.2.20. *Um código $\mathcal{C} \in A^n$ é chamado **geometricamente uniforme** se e somente se, dadas duas palavras quaisquer x e y do código, existe uma isometria $\varphi : \mathbb{F}_q^n \to \mathbb{F}_q^n$ tal que:*

i) $\varphi(\mathcal{C}) = \mathcal{C}$, *ou seja, a isometria leva o código no código.*

ii) $\varphi(x) = y$.

Para finalizar essa seção, vamos citar um exemplo de código $q-$ário e explorar suas propriedades usando a distância de Lee.

Exemplo 1.2.21. *[24] Considere o código:*

$$\mathcal{C} = \{\lambda(1,2) : \lambda = 0,1,2,3,4\} \in {\mathbb{F}_5}^2$$

Note que $\omega_L(1,2) = \omega_L(\lambda(1,2)) = 3 \quad \forall \quad \lambda \neq 0$, *de modo que* $\lfloor \frac{d_L(\mathcal{C})-1}{2} \rfloor = \lfloor \frac{3-1}{2} \rfloor = 1$. *Vamos descrever as bolas unitárias relativas a métrica de Lee, centradas nas palavras do código. Definimos a bola centrada na origem como:*

$$B_L((0,0);1) = \{(0,0),(1,0),(4,0),(0,1),(0,4)\}.$$

As outras bolas são obtidas por translação, ou seja, dados $\lambda(1,2) \in \mathcal{C}$, *temos que:*

$$\begin{aligned} B_L(\lambda(1,2);1) &= \lambda(1,2) + B_L((0,0);1) \\ &= \{\lambda(1,2) + v : v \in B_L((0,0);1)\}. \end{aligned}$$

Por verificação, podemos constatar que:

$$B_L(\lambda(1,2);1) \cap B_L(\alpha(1,2);1) = \emptyset,$$

se $\lambda \neq \alpha$, *de modo que ao tomarmos a união* $\bigcup_{\lambda=0}^{4}(\lambda(1,2),1))$ *destas cinco bolas, obtemos exatamente* 25 *elementos distintos e temos que:*

$$\mathbb{F}_5{}^2 = \bigcup_{\lambda=0}^{4}(\lambda(1,2),1).$$

Isso nos diz que o código $\mathcal{C}$ *é perfeito com a métrica de Lee.*

Observe que se considerarmos a métrica de Hamming ao invés da métrica de Lee, tal conclusão não é válida. De fato:

$$\omega_H(1,2) = \omega_H(\lambda(1,2)) = 2,$$

para todo $\lambda \neq 0$, *de modo que* $\lfloor \frac{d_H(\mathcal{C})-1}{2} \rfloor = \lfloor \frac{2-1}{2} \rfloor = 0$ *e, com isso, concluímos que as bolas unitárias devem obrigatoriamente se interceptar. Note que:*

$$B_H((0,0);1) = \{(0,0),(1,0),(2,0),(3,0),(4,0),(0,1),(0,2),(0,3),(0,4)\},$$

de modo que as bolas unitárias possuem nove elementos cada. Como $|\mathbb{F}_5{}^2| = 5^2 = 25$ *e* 9 *não é divisor de* 25, *a união dessas bolas não pode recobrir todo o espaço sem interseção.*

1.3 Propriedades dos códigos lineares

A classe dos códigos lineares é a mais utilizada na prática e a vantagem de sua utilização se dá pelo fato de eles serem completamente descritos por uma matriz geradora. Isso faz com que o teste feito para verificar se uma palavra dada pertence ao código ocorre facilmente através de uma matriz de paridade, conceitos esses que serão definidos no decorrer desta seção.

É importante ressaltar que todos os códigos que serão apresentados nas seções que seguem nesse primeiro capítulo são exemplos de códigos lineares e, portanto, herdarão as mesmas características básicas explicitadas nesta primeira.

Já definimos códigos lineares na seção 1.1 mas podemos também entender um código linear (que, por conveniência, será utilizada com mais frequência nesse texto) conforme a seguir.

Definição 1.3.1. *Os códigos lineares podem ser obtidos como imagem de uma transformação linear injetiva*

$$\Phi : {\mathbb{F}_q}^k \longrightarrow {\mathbb{F}_q}^n$$
$$(a_1, a_2, \ldots, a_k) \mapsto G_{(n\times k)} \cdot (a_1, a_2, \ldots, a_k)^T,$$

onde $G_{(n\times k)}$ é uma matriz de posto k formada por elementos do corpo $\mathbb{F}_q$. Ou podem também ser dados como o núcleo de uma outra transformação linear

$$\Theta : {\mathbb{F}_q}^n \longrightarrow {\mathbb{F}_q}^{n-k},$$

onde $\Phi(\mathbb{F}_q^k) = Ker(\Theta)$.

Definição 1.3.2. *Sejam $\mathbb{F}_q$ um corpo finito e $\mathcal{C} \subset {\mathbb{F}_q}^n$ um código linear. Chamamos de **parâmetros** do código linear $\mathcal{C}$ à terna de inteiros $[n, k, d]$, onde: n é o comprimento do código, k é a dimensão de $\mathcal{C}$ sobre $\mathbb{F}_q$ e d representa a distância mínima de $\mathcal{C}$, que também é igual ao peso $\omega(\mathcal{C})$ do código $\mathcal{C}$, conforme visto na Proposição 1.2.8.*

Definição 1.3.3. *Observando a Definição 1.3.1, identificamos e definimos a matriz G como sendo a **matriz geradora** do código $\mathcal{C}$ e $\mathcal{C}$ será dito um $[n, k]-$ **código de bloco linear**.*

Além disso, consideramos $\beta = \{v_1, \ldots, v_n\}$ uma base ordenada de $\mathcal{C}$ e seja G a matriz cujas colunas são os vetores $v_i = (v_{1i}, \ldots, v_{ni})$, $\quad i = 1, \ldots, k$, ou seja:

$$G = \begin{pmatrix} v_1 & v_2 & \ldots & v_k \end{pmatrix} = \begin{pmatrix} v_{11} & v_{12} & \ldots & v_{1k} \\ v_{21} & v_{22} & \ldots & v_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ v_{n1} & v_{n2} & \ldots & v_{nk} \end{pmatrix}$$

A matriz G é chamada de matriz geradora de $\mathcal{C}$ associada à base β.

Notamos que a matriz geradora de um código linear $\mathcal{C}$ não é única, já que ela depende da escolha de uma base para $\mathcal{C}$. Recordamos, ainda, que uma base de um espaço vetorial pode ser obtida de uma outra qualquer através de sequências de operações de tipo:

- permutação de dois elementos da base;
- multiplicação de um elemento da base por um escalar não nulo;
- substituição de um vetor da base por ele mesmo somado com um múltiplo escalar de outro vetor da base.

A troca de coordenadas é uma isometria na métrica de Hamming e então, como mostraremos a seguir, a menos de equivalência, sempre existe uma matriz geradora na forma padrão, onde as primeiras k linhas de G formam uma matriz identidade de ordem k. Tal matriz possui o seguinte formato:

$$G = \begin{pmatrix} I_{k\times k} \\ B_{(n-k)\times k} \end{pmatrix}.$$

Proposição 1.3.4. *[31] Dado um código $\mathcal{C}$, existe um código equivalente $\mathcal{C}'$ com matriz geradora na forma padrão.*

Demonstração. Seja G uma matriz geradora de $\mathcal{C}$. Como os k vetores coluna de G são linearmente independentes, a matriz G tem posto k e admite uma submatriz $k \times k$ com determinante não nulo.

Operamos por meio de trocas de linhas em G e obtemos uma matriz $\tilde{G}$ de modo que tal submatriz ocupe as k primeiras linhas, da seguinte maneira:

$$\tilde{G} = \begin{pmatrix} A_{k\times k} \\ B_{(n-k)\times k} \end{pmatrix}.$$

Isto corresponde à troca de coordenadas no código e portanto produz um código equivalente (segundo a distância de Hamming). Em seguida, efetuamos operações por colunas em $\tilde{G}$, obtendo $\hat{G}$ na forma:

$$\hat{G} = \begin{pmatrix} I_{k\times k} \\ \hat{B}_{(n-k)\times k} \end{pmatrix}.$$

Isto é, $\hat{G} = \tilde{G}\left(A^{-1}\right)_{k\times k}$, onde notamos que A uma matriz inversível em $\mathbb{F}_q$ e assim as colunas de $\hat{G}$ são obtidas por combinações lineares de $\tilde{G}$ e vice-versa. Com isso, vemos que o código gerado por $\hat{G}$ é o mesmo que o código gerado por $\tilde{G}$ e ambos equivalentes ao código gerado por G.

□

Exemplo 1.3.5. *Muitos computadores usam a sequência de* 8 *bits (*1 *byte) como unidade de informação. O código $ASCII$ é o mais usado em microcomputadores e representa em uma sequência, as letras do alfabeto maiúscula e minúscula, dígitos de* 0 *a* 9*, entre outros símbolos.*

Para escrever em português ou inglês não precisamos de mais que 80 *símbolos, o que é um número bem menor que o número de bytes,* $256(= 2^8)$*, e assim, podemos utilizar apenas sete dos bits para símbolos (código-fonte) e o oitavo bit para controle da seguinte maneira:*

$$\begin{cases} 0, & \text{se o número de 1's for par} \\ 1, & \text{se o número de 1's for ímpar} \end{cases}$$

A letra A e o número 1*, por exemplo, são respectivamente codificados em $ASCII$ como* 1000001 *e* 1000110*. Adicionando o último dígito conforme a regra acima, obtemos,* 10000010 *e* 10001101*, na devida ordem.*

Vislumbrando tal codificação como uma transformação linear, vem que:

$$\sigma : {\mathbb{F}_2}^7 \longrightarrow {\mathbb{F}_2}^8$$
$$(x_1, x_2, x_3, x_4, x_5, x_6, x_7) \mapsto (x_1, \ldots, x_7, x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7)$$

Assim, uma matriz geradora desse código, na forma padrão, é dada por:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Definição 1.3.6. *A **matriz de verificação** ou **matriz de paridade** $H_{(n-k)\times n}$ de um código linear $\mathcal{C}$, com matriz geradora G, é uma matriz com a propriedade de detectar se um vetor w de ${\mathbb{F}_q}^n$ é uma palavra do código da seguinte maneira:*

$$Hw^T = 0 \in {\mathbb{F}_q}^{n-k} \Leftrightarrow w \in \mathcal{C} \subset {\mathbb{F}_q}^n.$$

Observamos que as linhas de H são vetores de $\mathcal{C}^\perp$. De fato, tome $w \in \mathcal{C} \subset \mathbb{F}_q^n \Rightarrow Hw^T = 0$. Matricialmente, temos que:

$$\begin{pmatrix} h_{11} & h_{12} & \dots & h_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ h_{(n-k)1} & h_{(n-k)2} & \dots & h_{(n-k)n} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = 0.$$

Fixando i, vem que as linhas de H são vetores de $\mathcal{C}^\perp$, pois:

$$\sum_{j=1}^{n} \; h_{ij} \cdot w_j = 0, \quad \forall i,$$

com $w \in \mathcal{C}$.

Além desse fato, podemos notar ainda, utilizando o teorema do núcleo e da imagem que:

$$n = \dim(\mathrm{Im}(H)) + \dim(\mathrm{Ker}(H)) \Rightarrow n - \dim(\mathrm{Ker}(H)) = \dim(\mathrm{Im}(H)) \Rightarrow n - k = \dim(\mathrm{Im}(H)),$$

ou seja, já que $\mathrm{posto}(H) = n - k$ concluímos que as linhas de H são linearmente independentes, portanto, geram um subespaço vetorial de dimensão $n - k$.

Reunindo tais informações e sabendo que o espaço gerado pelas linhas de H está contido em $\mathcal{C}^\perp$ segue que eles coincidem, já que os dois subespaços possuem a mesma dimensão. Assim, provamos que as linhas H geram $\mathcal{C}^\perp$.

Observamos ainda que, se G é uma matriz geradora na forma padrão para $\mathcal{C}$,

$$G = \begin{pmatrix} I_{k\times k} \\ B_{(n-k)\times k} \end{pmatrix},$$

então uma matriz de paridade para $\mathcal{C}$ é

$$H_{(n-k)\times n} = \begin{pmatrix} -B_{(n-k)\times k} & I_{(n-k)\times(n-k)} \end{pmatrix}.$$

Exemplo 1.3.7. *Retornando ao Exemplo 1.3.5, podemos explicitar a matriz de paridade que é dada por*

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \Rightarrow H \cdot \begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \end{pmatrix}^T = a_1 + a_2 + \cdots + a_8 = 0 \Leftrightarrow$$

$$a_8 = -(a_1 + a_2 + \cdots + a_7) = a_1 + a_2 + \cdots + a_7,$$

lembrando que estávamos considerando um código binário.

Munidos de tais informações, podemos afirmar que um código linear $\mathcal{C}$ pode ser descrito por meio de suas matrizes geradora G e de paridade H da seguinte forma:

- $\mathcal{C} = \{Gu^T : u \in \mathbb{F}_q^k\}$.
- $\mathcal{C} = \{c \in \mathbb{F}_q^n : Hc^T = 0_{n-k}\}$.

Proposição 1.3.8. *[31] Seja H a matriz de paridade de um código $\mathcal{C}$. Temos que o peso mínimo de $\mathcal{C}$ é maior do que ou igual a s se, e somente se, quaisquer $s-1$ colunas de H são linearmente independentes.*

Demonstração. ($\Rightarrow$) Vamos supor que $\omega(\mathcal{C}) \geq s$ e que, por absurdo, a matriz de paridade H tenha $s-1$ colunas linearmente dependentes, digamos $h^{i_1}, h^{i_2}, \ldots, h^{i_{s-1}}$. Logo existiriam $c_{i_1}, \ldots, c_{i_{s-1}}$ no corpo, nem todos nulos tais que:

$$c_{i_1} h^{i_1} + \cdots + c_{i_{s-1}} h^{i_{s-1}} = 0.$$

Portanto, $c = (0, \ldots, c_{i_1}, 0, \ldots, c_{i_{s-1}}, 0, \ldots, 0) \in \mathcal{C}$ e, consequentemente, $\omega(c) \leq s-1 < s$, o que é uma contradição.
($\Leftarrow$) Suponhamos que cada conjunto de $s-1$ colunas de H é linearmente independente. Seja $c = (c_1, \ldots, c_n)$ uma palavra não nula de $\mathcal{C}$, e sejam $h^1, \ldots, h^n$ as colunas de H. Como, por definição, $Hc^T = 0$, vem que:

$$0 = Hc^T = \sum c_i h^i. \tag{1.3.1}$$

Visto que $\omega(c)$ é o número de componentes não nulas de c, segue que $\omega(c) \leq s-1$, e teríamos, por 1.3.1 uma combinação nula de um número t, com $1 \leq t \leq s-1$ de colunas de H, o que é contraditório. Logo, $\omega(c) \geq s$ e, por conseguinte, $\omega(\mathcal{C}) \geq s$. □

Proposição 1.3.9. *[31] Seja H a matriz de paridade de um código $\mathcal{C}$. Temos que o peso de $\mathcal{C}$ é igual a s se, e somente se, quaisquer $s-1$ colunas de H são linearmente independentes e existem s colunas linearmente dependentes.*

Demonstração. ($\Rightarrow$) Suponhamos que $\omega(\mathcal{C}) = s$, logo, todos conjunto de $s-1$ colunas de H é linearmente independente. Por outro lado, existem s colunas de H linearmente dependentes, pois, caso contrário, de acordo com a Proposição 1.3.8, teríamos que $\omega(\mathcal{C}) \geq s+1$.
($\Leftarrow$) Admita, agora, que todo conjunto de $s-1$ vetores coluna de H é linearmente independente e existem s colunas linearmente dependentes. Logo, da Proposição 1.3.8, temos que $\omega(\mathcal{C}) \geq s$. Mas, $\omega(\mathcal{C})$ não pode ser maior do que s, pois, nesse caso, usando novamente a Proposição 1.3.8, vem que todo conjunto s de colunas de H é linearmente independente, o que é uma contradição. □

Fazendo uma releitura da Proposição 1.3.9, utilizando o fato de que $\omega(\mathcal{C}) = d$, onde d é a distância mínima. Fazendo tal correspondência, segue o seguinte resultado:

Proposição 1.3.10. *[42] Um código linear $\mathcal{C}$ tem distância mínima d se, e somente se, o número mínimo de vetores coluna linearmente dependentes da matriz de paridade é d.*

Exemplo 1.3.11. *Considere o código linear binário dado pela matriz geradora abaixo:*

$$G = \begin{pmatrix} I_{8\times 8} \\ B_{4\times 8} \end{pmatrix};$$

onde

$$B_{4\times 8} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Observe que $k = 8$ e $n - k = 4 \Rightarrow n = 12$ e que $|\mathcal{C}| = q^k = 2^8 = 256$ palavras, donde $q = 2$, pois o código é binário.

Ainda, observe que a matriz de paridade é:

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Pela Proposição 1.3.10, sabemos que um código linear $\mathcal{C}$ tem distância mínima d se, e somente se, o número mínimo de vetores coluna linearmente dependentes da matriz de paridade é d. Usando essa informação, temos que, neste caso, $d = 3$.

Definição 1.3.12. *Dado $w \in {\mathbb{F}_q}^n$, a imagem deste elemento pela matriz de paridade, Hw^T, é chamada **síndrome** de w.*

Note que os elementos que pertencem ao código linear $\mathcal{C}$ são os que possuem síndrome nula.

Corolário 1.3.13. *[31] (Cota de Singleton) Os parâmetros $[n, k, d]$ de um código linear satisfazem a desigualdade*

$$d \leq n - k + 1.$$

Demonstração. Se H é uma matriz de paridade, ela tem posto $n-k$. Como, pela Proposição 1.3.9, $d - 1$ é menor ou igual ao posto de H, segue a desigualdade que queremos. □

Definição 1.3.14. *Um código será chamado de **MDS** (Maximum Distance Separable) se valer a igualdade para a Cota de Singleton, ou seja, $d = n - k + 1$.*

Corolário 1.3.15. *Códigos binários corrigem pelo menos um erro se, e somente se, todos os vetores da matriz de paridade são distintos.*

Demonstração. ($\Rightarrow$) $\mathcal{C}$ corrige pelo menos um erro $\Rightarrow$ $\lfloor \frac{d-1}{2} \rfloor \geq 1 \Rightarrow d \geq 3$. Da Proposição 1.3.10, vem que o número mínimo de colunas linearmente dependentes é maior ou igual a 3.

($\Leftarrow$) Por hipótese temos que todos os vetores da matriz de paridade são distintos, logo, novamente, pela Proposição 1.3.10, o número mínimo de colunas linearmente dependentes é 3 e d, a distância mínima é no mínimo 3. Para justificar o fato de que o número mínimo de colunas linearmente dependentes é 3, consideramos, sem perda de generalidade, a matriz de paridade da forma

$$H_{(n-k)\times n} = \begin{pmatrix} -B_{(n-k)\times k} & I_{(n-k)\times(n-k)} \end{pmatrix},$$

e então devemos ter todos os valores distintos, pois se tivermos dois iguais teremos duas colunas linearmente dependentes, o que seria um absurdo. De fato: para que dois vetores sejam linearmente dependentes em $\mathbb{F}_2$ (já que estamos considerando um código binário), devemos ter:

$$G = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} = \alpha \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \Rightarrow a_i = 0 \quad \forall i \ \text{ ou } \ a_i = b_i \quad \forall i.$$

Logo, $\lfloor \frac{d-1}{2} \rfloor \geq 1$ e o código corrige pelo menos um erro. □

Exemplo 1.3.16. *Será que é possível que, a um código binário de* 5 *dígitos se acrescente três, e ele corrija um erro? Observe que a matriz geradora desse código será da forma* $G = \begin{pmatrix} I_{5\times5} \\ B_{3\times5} \end{pmatrix}$, *já a matriz de paridade, por sua vez, é dada por*

$$H = \begin{pmatrix} B_{3\times5} & I_{3\times3} \end{pmatrix}. \tag{1.3.2}$$

Vimos no Corolário 1.3.15, que um código corrige pelo menos um erro se e somente se todas as colunas da matriz de paridade são distintas. Note que em um código $\mathcal{C} \subset {\mathbb{F}_2}^3$ *temos no máximo* 7 *vetores distintos, excluindo o vetor nulo. E, para que todas as colunas de* H *fossem distintas,* H *poderia ter no máximo* 7 *colunas, mas possui* 8 *(como podemos ver em 1.3.2.) Logo, esse código não pode corrigir erros.*

Podemos generalizar a ideia do exemplo anterior por meio dos mesmos argumentos, conforme exposto na proposição a seguir, cuja demonstração é direta.

Proposição 1.3.17. *Um código* $\mathcal{C} : {\mathbb{F}_2}^k \longrightarrow {\mathbb{F}_2}^n$, *com* $n = k + r$, r *um número natural fixo, não corrige erros para* $k \geq 2^r - r$.

1.4 Códigos de Hamming

R. W. Hamming observando os erros obtidos por leitores de cartões perfurados, decidiu aperfeiçoar seus estudos sobre códigos que corrigem erros. Assim, em 1950, ele desenvolveu um código de detecção, que permite não apenas constatar a presença do erro, mas localizá-lo. E, tal código ficou conhecido como código de Hamming que até hoje é bastante usado na prática.

1.4.1 Descrição de um código de Hamming

Definição 1.4.1. *Um **código de Hamming** de ordem n sobre $\mathbb{F}_2$ é um código com matriz de paridade $\mathcal{H}_m$, de ordem $m \times n$, cujas colunas são formadas por todos os elementos não nulos de ${\mathbb{F}_2}^m$.*

Definição 1.4.2. *O **comprimento** de um código de Hamming de ordem m é $n = 2^m - 1$ e sua **dimensão** é $k = n - m = 2^m - m - 1$. Logo, por meio desses parâmetros, usamos a notação $[n, k] = [2^m - 1, 2^m - m - 1]-$ código de Hamming.*

Exemplo 1.4.3. *Considere o $[15, 11]-$ código de Hamming $\mathcal{H}_4$. Ou seja:*

$$\mathcal{C} : \mathbb{F}_2^{11} \longrightarrow \mathbb{F}_2^{15}$$

Daí, temos que a matriz geradora é dada por:

$$G = \begin{pmatrix} I_{11\times 11} \\ B_{4\times 11} \end{pmatrix},$$

onde $B = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$.

E, a matriz de paridade, por sua vez:

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Proposição 1.4.4. *[31] Para os códigos de Hamming, verifica-se que:*

i) A distância de Hamming mínima de um código $\mathcal{H}_m$ é 3 e, portanto, ele detecta até dois erros e corrige um.

ii) Todo elemento ${\mathbb{F}_2}^{2^m-1}$ está a uma distância de no máximo um de uma palavra do código, ou seja:

$$\bigcup_{a\in\mathcal{H}_m} B(a, 1) = {\mathbb{F}_2}^{2^m-1}.$$

Demonstração. (i) Tome x e y duas palavras-código de um código de Hamming $\mathcal{C}$ de ordem m, com matriz de paridade H_m. Então, $x - y \in \mathcal{C}$, já que $\mathcal{C}$ é um código linear.

Se $d(x, y) = 1$, então $H_m(x - y)$ é uma coluna de H_m. Sabemos que todas as colunas de H_m são não nulas, mas, se $(x - y)$ é uma palavra do código de Hamming $\mathcal{C}$, temos que $H_m(x - y) = 0$, o que é uma contradição.

Se $d(x, y) = 2$, então $H_m(x - y) = 0$ se, e somente se, existem duas colunas linearmente dependentes em H_m. Este não é o caso, já que $d(x, y) \geq 3$ para todos as palavras-código x e y. Toda matriz de paridade para um código de Hamming binário terá pelo menos três colunas

linearmente dependentes, então, usando a Proposição 1.3.10, algumas palavras-código possuem distância 3 e segue que a distância mínima é 3.

(ii) Em outras palavras, estamos querendo provar neste item que todo código de Hamming é $1-$perfeito. Observe que $\kappa = \lfloor \frac{d-1}{2} \rfloor = 1$. Dado $c \in {\mathbb{F}_2}^n$, temos que:

$$|B(c,1)| = 1 + n$$

Logo, $|\bigcup_{c\in\mathcal{C}} B(c,1)| = [1+n]2^k = [1+2^m-1]2^{n-m} = 2^n$ e, consequentemente, $\bigcup_{c\in\mathcal{C}} B(c,1) = \mathbb{F}_2^n = {\mathbb{F}_2}^{2^m-1}$.

Com isso, provamos que todo código de Hamming é $1-$perfeito e corrige um erro. □

1.4.2 Decodificação de um código de Hamming

Vamos entender agora, por meio de um exemplo simples, como ocorre a decodificação de um código de Hamming utilizando o critério da verossimilhança e considerando que ele é um código que corrige um erro.

Exemplo 1.4.5. *Vamos decodificar um elemento do código de Hamming $\mathcal{H}_3$, assumindo que no máximo um erro foi cometido. Dada uma palavra recebida na forma de um vetor coluna 7×1,*

$$r = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 \end{pmatrix}^T,$$

e a matriz de paridade

$$H_3 = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Temos que:

$$H_3 r = \begin{pmatrix} a_2 + a_3 + a_4 + a_5 \\ a_1 + a_3 + a_4 + a_6 \\ a_1 + a_2 + a_4 + a_7 \end{pmatrix}.$$

Já que todos os elementos de ${\mathbb{F}_2}^3$ compõem colunas de H_3, logo a síndrome $H_3 r$ será o vetor nulo ou será uma coluna de H_3.

Se a síndrome de $H_3 r$ for o vetor nulo, então $r = Gx$ está no código, para algum $x \in \mathbb{F}_2^4$. Senão, tome w como sendo a palavra do código que troca o j-ésimo bit de r, se $H_3 r$ for a j-ésima coluna de H_3. Então, a palavra enviada será a palavra formada pelos quatro primeiros dígitos de r no primeiro caso e de w no segundo.

Para ilustrar o algoritmo, tome, por exemplo $r = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}^T$ Daí, observe que $H_3 r = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}^T$ e esse vetor corresponde a segunda coluna da matriz H_3. Assim, segue que a palavra decodificada é 0100.

Observação 1.4.6. *Um algoritmo mais geral que decodifica códigos lineares se tivermos até um erro, está ilustrado no Algoritmo* 1.1.

Algoritmo 1.1 Decodificação de um código linear que corrige um erro

---

[31] Seja H a matriz de paridade do código $\mathcal{C}$ e seja r um vetor recebido. Suponha que $d \geq 3$.
(1) Calcule Hr^T.
(2) Se $Hr^T = 0$, aceite os k primeiros dígitos de r como sendo a própria palavra do código enviada.
(3) Se $Hr^T = s^T \neq 0$, compare s^T com as colunas de H.
(4) Se existirem i e α tais que $s^T = \alpha h^i$, para $\alpha \in \mathbb{F}_q$, então e é a $n-$upla com α na posição i e zeros nas outras posições. Corrija r pondo $c = r - e$.
(5) Se o contrário de (4) ocorrer, então mais de um erro foi cometido.

Podemos constatar após a observação do algoritmo que ele é válido quando $\omega(e) \leq 1$.

Exemplo 1.4.7. *Para ilustrar o processo, vamos aplicá-lo ao Exemplo 1.1.1. Temos a seguinte codificação:*

$$\begin{aligned} 00 &\mapsto 000000 \\ 01 &\mapsto 010101 \\ 10 &\mapsto 101010 \\ 11 &\mapsto 111111. \end{aligned}$$

Ou seja, o código pode ser definido como $\mathcal{C} : {\mathbb{F}_2}^2 \longrightarrow {\mathbb{F}_2}^6$, *donde:*

$$G = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad e \quad H = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Considere, por exemplo, $r_1 = 110101$ *e observe que* $r_1 \notin \mathcal{C}$. *Seguindo os passos do algoritmo:*
$Hr_1^T = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = 1 \cdot h^1$. *Logo,* $e = 100000$ *e* $c = 010101$. *que é a palavra correta.*
Mas, se considerarmos $r_2 = 100101, r_2 \notin \mathcal{C}$ *temos que:*
$Hr_2^T = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$ $e \nexists \ \ i, \alpha$ *tais que* $Hr_2^T = \alpha h^i$ *e concluímos que mais de um erro foi cometido e não é possível corrigir a palavra.*

1.5 Códigos cíclicos

Os códigos cíclicos são uma subclasse dos códigos lineares cuja construção depende apenas de um polinômio de grau menor do que o comprimento das palavras-código. Tais códigos possuem excelentes algoritmos de decodificação, conforme veremos no decorrer desta seção.

O estudo dos códigos cíclicos iniciou-se com Prange, em 1957 e foi ele quem notou a riqueza presente na estrutura algébrica desse tipo de código. Além disso, ([61], p.19), tais códigos são muito utilizados na prática.

1.5.1 Descrição dos códigos cíclicos

Definição 1.5.1. *Seja $\mathbb{F}_q$ um corpo finito com q elementos. Um código linear $\mathcal{C} \subset \mathbb{F}_q^n$ será chamado de **código cíclico** se, para todo $c = (c_0, \ldots, c_{n-1}) \in \mathcal{C}$, o vetor $(c_{n-1}, c_0, \ldots, c_{n-2})$ pertence ao código $\mathcal{C}$.*

Definição 1.5.2. *O vetor $(c_{n-1}, c_0, \ldots, c_{n-2}) \in {\mathbb{F}_q}^n$ é chamado de **desvio cíclico** de $(c_0, \ldots, c_{n-1})$.*

Portanto, podemos intuir usando essa definição que, um código linear $\mathcal{C}$ é cíclico se contém os desvios cíclicos de todas as palavras-código.

Exemplo 1.5.3. *O código explicitado no Exemplo 1.1.1 é um código cíclico. De fato, temos que $\mathcal{C} = \{000000, 010101, 101010, 111111\}$ e todos os desvios cíclicos das palavras-código claramente pertencem ao código $\mathcal{C}$.*

Consideramos uma palavra do código $c = (c_0, c_1, \ldots, c_{n-1}) \in \mathbb{F}_q^n$. Podemos associar c a um polinômio por meio da seguinte transformação linear:

$$\phi : \mathbb{F}_q^n \longrightarrow R = \mathbb{F}_q[x]/I$$

$$(c_0, c_1, \ldots, c_{n-1}) \mapsto c_0 + c_1 x + \cdots + c_i x^i + \cdots + c_{n-1} x^{n-1} = c(x),$$

onde $I =< x^n - 1 >$ e R é o anel dos polinômios com coeficientes em $\mathbb{F}_q$. Podemos enxergar os códigos lineares como subconjuntos do anel R graças à correspondência estabelecida acima.

Definição 1.5.4. *Se c é uma palavra-código de $\mathcal{C}$, chamamos $c(x)$ de **código polinomial.***

Proposição 1.5.5. *[4] Seja $\mathcal{C}$ um $[n, k]-$código linear. Então $\mathcal{C}$ é um código cíclico se e somente se $\mathcal{C}$ é um ideal de R.*

Demonstração. ($\Rightarrow$) Note que, o desvio cíclico $\tilde{c}$ possui o seguinte código polinomial associado a ele:

$$\begin{aligned} \tilde{c}(x) &= xc(x) - c_{n-1}(x^n - 1) \\ &= c_0 x + c_1 x^2 + \cdots + c_{n-1} x^n - c_{n-1} x^n + c_{n-1} \\ &= c_{n-1} + c_0 x + c_1 x^2 + \cdots + c_{n-2} x^{n-1}. \end{aligned}$$

Dessa forma, notamos que $\tilde{c}(x)$ possui grau menor que n e é igual ao resto da divisão de $xc(x)$ por $x^n - 1$. Em outras palavras:

$$\tilde{c}(x) \equiv xc(x) (mod\ x^n - 1).$$

Então, segue que $\tilde{c}(x)$ e $xc(x)$ são iguais sob o anel de polinômios $\mathbb{F}_q[x](mod\ x^n-1)$. Se $c(x)$ é o código polinomial associado a alguma palavra-código c de $\mathcal{C}$, então, por um abuso de notação, podemos dizer que $c(x) \in \mathcal{C}$.

Se $f(x)$ é um polinômio qualquer sobre $\mathbb{F}_q[x]$ cujo resto da divisão por x^n-1 pertence a $\mathcal{C}$, podemos escrever

$$f(x) \in \mathcal{C}(mod\ x^n-1).$$

Tendo isso em mente, vem que a definição de código cíclico pode ser estendida a um anel de polinômios como:

$$c(x) \in \mathcal{C}(mod\ x^n-1) \quad \text{ se, e somente se } \quad xc(x) \in \mathcal{C}(mod\ x^n-1).$$

Sem perda de generalidade, temos que:

$$x^i c(x) \in \mathcal{C}(mod\ x^n-1) \quad \forall i.$$

Por linearidade, para todo $a_i \in \mathbb{F}_q$:

$$a_i x^i c(x) \in \mathcal{C}(mod\ x^n-1) \text{ e } \sum_{i=0}^{d} a_i x^i c(x) \in \mathcal{C}(mod\ x^n-1), \quad d \in \mathbb{N} \text{ qualquer.}$$

Isso nos diz que para todo polinômio $a(x) = \sum_{i=0}^{d} a_i x^i \in \mathbb{F}_q$, o produto $a(x)c(x) \in \mathcal{C}$. E com isso, provamos que $\mathcal{C}$ é um ideal de polinômios em R.

($\Leftarrow$) Se $\mathcal{C}$ é um ideal de R, então sabemos que para todo polinômio $k(x) \in R$ e para todo $c(x) \in (C)$ vale que $k(x)c(x) \in \mathcal{C}$. Tome $k(x) = x$ e o resultado segue. □

Proposição 1.5.6. *[29] Seja $\mathcal{C} \neq 0$ um código cíclico de comprimento n sobre $\mathbb{F}_q$. Afirmamos então que:*

i) Se $g(x)$ é um código polinomial mônico de grau mínimo em $\mathcal{C}$, então $g(x)$ é determinado unicamente em $\mathcal{C}$ e

$$\mathcal{C} = \{q(x)g(x) : q(x) \in \mathbb{F}_q[x]_{n-r}\},$$

onde $r = deg(g(x))$. Particularmente, $\mathcal{C}$ possui dimensão $n-r$.

ii) O polinômio $g(x)$ divide x^n-1 em $\mathbb{F}_q[x]$.

Demonstração. Como $\mathcal{C} \neq 0$, ele contém pelo menos um código polinomial não nulo, o qual possui um único múltiplo escalar mônico. Além disso, existe um único polinômio mônico $g(x)$ em $\mathcal{C}$ de grau mínimo. Vamos supor que esse grau seja r, que é único a menos que $g(x)$ não o seja.

Pela Proposição 1.5.5, o conjunto dos polinômios

$$C_0 = \{q(x)g(x) : q(x) \in \mathbb{F}_q[x]_{n-r}\}$$

está contido em $\mathcal{C}$, já que ele é composto pelos múltiplos do código polinomial $g(x)$ com a propriedade de possuir grau menor que n e C_0 é um $\mathbb{F}_q-$ espaço vetorial de dimensão $n-r$. O polinômio $g(x)$ é o único polinômio mônico de grau r em C_0.

(i) Devemos provar aqui que todo código polinomial $c(x)$ é um $\mathbb{F}_q[x]_{n-r}$ múltiplo de $g(x)$ e pertence ao conjunto C_0.

Pelo algoritmo da divisão de Euclides, temos que:

$$c(x) = q(x)g(x) + r(x),$$

para algum $q(x), r(x) \in \mathbb{F}_q[x]$ com $deg(r(x)) < r = deg(g(x))$. Ou ainda:

$$r(x) = c(x) - q(x)g(x)$$

Por definição, $c(x) \in \mathcal{C}$ e $q(x)g(x) \in \mathcal{C}$, já que $\mathcal{C}$ é um ideal, de acordo com a Proposição 1.5.5, logo $r(x) \in \mathcal{C}$. Se $r(x)$ fosse diferente de zero, teríamos um múltiplo escalar mônico pertencente a $\mathcal{C}$ e com grau menor que r, o que é uma contradição devido a escola de $g(x)$. Assim, $r(x) = 0$ e $c(x) = q(x)g(x)$, como queríamos. Com isso $\mathcal{C} = C_0$.

(ii) Seja

$$x^n - 1 = h(x)g(x) + s(x),$$

para algum $s(x)$ de grau menor que $deg(g(x))$. Então, como feito anteriormente:

$$s(x) = (-h(x)g(x))(mod\ x^n - 1)$$

pertence a $\mathcal{C}$. Novamente, se $s(x)$ é não nulo, então existe um múltiplo escalar mônico pertencente a $\mathcal{C}$ e de grau menor do que $g(x)$, uma contradição. Logo, $s(x) = 0$ e $g(x)h(x) = x^n - 1$. □

Definição 1.5.7. *O polinômio $g(x)$ definido conforme na Proposição 1.5.6 é chamado* ***polinômio gerador*** *do código $\mathcal{C}$. O polinômio $h(x) \in \mathbb{F}_q[x]$ é determinado por*

$$g(x)h(x) = x^n - 1$$

e é chamado de ***polinômio verificador*** *de $\mathcal{C}$.*

Exemplo 1.5.8. *Considere o código binário cíclico de comprimento 7, temos então a seguinte fatoração em polinômios irredutíveis:*

$$x^7 - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1).$$

Como estamos trabalhando com códigos binários, podemos substituir os sinais negativos por positivos. Assim:

$$x^7 + 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1).$$

Observe que temos três fatores irredutíveis, logo, existem $2^3 = 8$ códigos cíclicos (incluindo 0 e $\mathbb{F}_2^7$). Tais códigos possuem os seguintes polinômios geradores:

i) 1;

ii) $x + 1$;

iii) $x^3 + x + 1$;

iv) $x^3 + x^2 + 1$;

v) $(x+1)(x^3 + x + 1) = x^4 + x^3 + x^2 + 1$;

vi) $(x+1)(x^3 + x^2 + 1) = x^4 + x^2 + x + 1$;

vii) $(x^3 + x + 1)(x^3 + x^2 + 1) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$;

viii) $(x+1)(x^3 + x + 1)(x^3 + x^2 + 1) = x^7 + 1$.

Notamos que em (i) *o polinômio* 1 *gera todo* $\mathbb{F}_2^7$. *Em* (ii) *encontramos o código de verificação de paridade e em* (vii) *o código de repetição. Em* $(viii)$ *vemos que o código* 0 *é gerado por* $x^7 + 1$. *Os polinômios de* (iii) *e* (iv) *possuem grau* 3 *e por isso geram códigos equivalentes ao* $[7, 4]-$*código de Hamming.*

Proposição 1.5.9. *[29] Consideramos o código cíclico* $\mathcal{C}$ *com o polinômio gerador* $g(x)$. *Então,* $\mathcal{C}$ *está contido no código de soma zero se, e somente se* $g(1) = 0$.

Demonstração. Um código binário de soma zero é aquele onde a soma de todos as suas entradas resultam em zero sob o corpo onde o mesmo está definido. Assim:
$(\Rightarrow)$ Considere a palavra-código $c = (c_0, c_1, \ldots, c_{n-1})$ pertencente ao código de soma zero. Já que $\mathcal{C} = \{q(x)g(x) : q(x) \in \mathbb{F}_q[x]_{n-r}\}$, tome $q(x) = 1$. Logo, vem que $g(x) \in \mathcal{C}$, que, por hipótese está contido em um código de soma zero. Dessa forma, os coeficientes de $g(x)$ são os bits de uma palavra-código, c, por exemplo. Daí:

$$g(x) = c_0 + c_1 x + \cdots + c_{n-1} c^{n-1}.$$

Portanto, $g(1) = c_0 + c_1 + \cdots + c_{n-1} = 0$.
$(\Leftarrow)$ $g(1) = 0 \Rightarrow c_0 + c_1 + \cdots + c_{n-1} = 0$, por definição de $g(x)$. Considere $c \in \mathcal{C}$ tal que $c = (c_0, c_1, \ldots, c_{n-1})$. Daí, concluímos que c é uma palavra-código que pertence a um código de soma zero, por conseguinte, $\mathcal{C}$ está contido em um código de soma zero. □

Proposição 1.5.10. *[29] Se* $\mathcal{C}$ *é um código cíclico de comprimento* n, *com um polinômio verificador* $h(x)$, *então* $\mathcal{C} = \{c(x) \in \mathbb{F}_q[x]_n : c(x)h(x) = 0(mod\ x^n - 1)\}$.

Demonstração. Queremos mostrar que

$$\mathcal{C} = \{c(x) \in \mathbb{F}_q[x]_n : c(x)h(x) = 0(mod\ x^n - 1\}.$$

Para isso, devemos provar que:

(i) $\mathcal{C} \subset \{c(x) \in \mathbb{F}_q[x]_n : c(x)h(x) = 0(mod\ x^n - 1)\}$.
Tome $c(x) \in \mathcal{C}$, então, pela Proposição 1.5.6, existe $q(x)$ tal que $c(x) = q(x)g(x)$. Mas,

$$c(x)h(x) = q(x)g(x)h(x) = q(x)(x^n - 1) = 0(mod\ x^n - 1).$$

(ii) $\{c(x) \in \mathbb{F}_q[x]_n : c(x)h(x) = 0(mod\ x^n - 1)\} \subset \mathcal{C}$.

Consideramos $c(x) \in \mathbb{F}_q[x]_n$ um polinômio qualquer tal que:

$$c(x)h(x) = p(x)(mod\ x^n - 1)$$

Então, $c(x)h(x) = p(x)(mod\ x^n - 1) = p(x)g(x)h(x)$. E, $(c(x) - p(x)g(x))h(x) = 0$.

Como $g(x)h(x) = x^n - 1$, não podemos ter $h(x) = 0$. Assim, $c(x) - p(x)g(x) = 0$ e $c(x) = p(x)g(x)$, como queríamos. □

Conhecendo o polinômio gerador $g(x) = \sum_{j=0}^{r} g_j x^j$ do código cíclico $\mathcal{C}$, podemos construir uma matriz geradora para $\mathcal{C}$.

Proposição 1.5.11. *[16] Um código cíclico $\mathcal{C}$ com polinômio gerador $g(x) = \sum_{j=0}^{r} g_j x^j$ possui dimensão $n - r$ e matriz geradora G onde,*

$$G^T = \begin{pmatrix} g(x) \\ xg(x) \\ \vdots \\ x^{n-r-2}g(x) \\ x^{n-r-1}g(x) \end{pmatrix} = \begin{pmatrix} g_0 & g_1 & \dots & g_r & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & \dots & g_r & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & g_0 & g_1 & \dots & g_r & 0 \\ 0 & 0 & \dots & 0 & g_0 & g_1 & \dots & g_r \end{pmatrix}.$$

Demonstração. Seja $c(x) \in \mathcal{C}$, onde $\mathcal{C}$ é um código cíclico com polinômio gerador $g(x)$. Sabemos, da Proposição 1.5.6 que $\mathcal{C} =< g(x) >$, ou seja, $c(x) = q(x)g(x)$ para algum polinômio $q(x)$. Observe que $deg(q(x)) < n - r$, já que $deg(c(x)) < n$. De fato:

$$c(x) = (q_0 + q_1x + \cdots + q_{n-r-1}x^{n-r-1})g(x) = q_0g(x) + q_1xg(x) + \cdots + q_{n-r-1}x^{n-r-1}g(x),$$

que é uma combinação linear das $n - r$ linhas $g(x), xg(x), \dots, x^{n-r-1}g(x)$ de G. As entradas não nulas g_0 garantem que as linhas de G são linearmente independentes. Logo, o espaço gerado pelas linhas de G é o $(n - r)-$dimensional código $\mathcal{C}$. □

Observando ainda tal construção como uma transformação linear σ, vemos que:

$$\sigma : \mathbb{F}^{n-r=k} \longrightarrow \mathbb{F}^n$$
$$(c_0, c_1, \dots, c_{n-r-1}) \mapsto G_{(n\times n-r)} \cdot (c_0, c_1, \dots, c_{n-r-1})^T.$$

Aplicando na base canônica, notamos finalmente que:

$$G_{(n\times n-r)} \cdot {e_1}^T = (g_0, g_1, \dots, g_{r-1}, g_r, 0, \dots, 0),$$
$$G_{(n\times n-r)} \cdot {e_2}^T = (0, g_0, g_1, \dots, g_{r-1}, g_r, 0, \dots, 0),$$
$$\vdots$$
$$G_{(n\times n-r)} \cdot {e_{n-r-1}}^T = (0, \dots, 0, g_0, \dots, g_{r-1}, g_r, 0),$$
$$G_{(n\times n-r)} \cdot {e_{n-r}}^T = (0, \dots, 0, 0, g_0, \dots, g_{r-1}, g_r).$$

Analisando essa construção, vemos que a transformação linear σ define o código $\mathcal{C}$ como sendo sua imagem, exatamente da maneira que queríamos.

Exemplo 1.5.12. *Considere $\mathcal{C}$ um $[7,4]-$ código binário cíclico com polinômio gerador $1+x+x^3$. Queremos encontrar uma matriz geradora para esse código.*

Note que o grau do polinômio gerador é $n-k=3$. Como $n=7$, temos que $k=4$ e uma base para o código $\mathcal{C}$ será

$$\begin{aligned} g(x) &= 1+x+x^3 \\ xg(x) &= x+x^2+x^4 \\ x^2g(x) &= x^2+x^3+x^5 \\ x^3g(x) &= x^3+x^4+x^6. \end{aligned}$$

Então, uma matriz geradora para tal código será:

$$G^T = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

Proposição 1.5.13. *[16] Um código cíclico com polinômio verificador $h(x) = \sum_{j=0}^{k} h_j x^j$ possui dimensão k e matriz de paridade H :*

$$H = \begin{pmatrix} h_k & h_{k-1} & \dots & h_0 & 0 & 0 & \dots & 0 \\ 0 & h_k & h_{k-1} & \dots & h_0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & h_k & h_{k-1} & \dots & h_0 & 0 \\ 0 & 0 & \dots & 0 & h_k & h_{k-1} & \dots & h_0 \end{pmatrix}.$$

Demonstração. Como o grau do polinômio gerador $g(x)$ é $r=n-k$, pela Proposição 1.5.11, a dimensão do código cíclico $\mathcal{C}$ deve ser k. De acordo com a Proposição 1.5.10, sabemos que uma palavra-código $c(x) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1}$ deve satisfazer $c(x)h(x) = 0$. Em particular, os coeficientes $x^k, x^{k+1}, \dots, x^{n-1}$ do produto $c(x)h(x)$ devem ser zero. Então, para $l = k, \dots, n-1$, temos que:

$$0 = \sum_{i+j=l} c_i h_j.$$

Mas, como cada uma dessas equações é uma das $n-k$ linhas da matriz:

$$H = \begin{pmatrix} h_k & h_{k-1} & \dots & h_0 & 0 & 0 & \dots & 0 \\ 0 & h_k & h_{k-1} & \dots & h_0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & h_k & h_{k-1} & \dots & h_0 & 0 \\ 0 & 0 & \dots & 0 & h_k & h_{k-1} & \dots & h_0 \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix},$$

vemos que as palavras-código são ortogonais a todos os desvios cíclicos do vetor $(h_k\, h_{k-1} \ldots h_0\, 0 \ldots 0)$. As palavras-código também são ortogonais a todas as combinações lineares das linhas de H. Isso significa que $\mathcal{C}^{\perp}$ contém o espaço gerado pelas linhas de H. Mas $h_k = 1$, logo, vemos que H possui posto $n-k$ e gera exatamente o espaço $\mathcal{C}^{\perp}$. Ou seja, H é a matriz de paridade para o código com polinômio verificador $h(x)$. □

1.5.2 Codificação e decodificação de códigos cíclicos

Consideramos um código cíclico $\mathcal{C}$ gerado por um polinômio $g(x)$, tal que $deg(g(x)) = n-k$. Seja $m = (m_0, \ldots, m_{k-1})$ uma mensagem, que pode ser representada também pelo polinômio $m(x) = m_0 + m_1 x + \cdots + m_{k-1}x^{k-1}$.

A codificação da mensagem $m(x)$ ocorre simplesmente pela multiplicação $c(x) = m(x)g(x)$, que também corresponde à $c = Gm^T$, se tratarmos na forma matricial.

Exemplo 1.5.14. *Seja $\mathcal{C}$ um código binário com $n = 7$ e $k = 4$, tal que $g(x) = x^3 + x^2 + 1$ seja o polinômio gerador desse código. Queremos codificar a mensagem $m = (1010)$.*

Temos que $m(x) = x^2 + 1$. Então, vem que $m(x) \hookrightarrow c(x) = m(x)g(x) = (x^2+1)(x^3+x^2+1) = x^5 + x^4 + x^3 + 1$, ou seja, $m = (1010) \hookrightarrow c = (1001110)$.

Vamos agora apresentar a decodificação de códigos cíclicos. Seja $\mathcal{C}$ um código cíclico binário gerado por $g(x)$. Suponha que a mensagem $c(x)$ tenha sido enviada e $w(x) = c(x) + e(x)$ tenha sido recebida, já que, devido ao ruído do canal, vetor recebido pode não ser o mesmo que o transmitido. Na decodificação de códigos lineares vista anteriormente, usamos a síndrome para efetuar o processo. No caso dos códigos cíclicos ocorre da mesma forma.

Definição 1.5.15. *Sob tais hipóteses, a **síndrome polinomial** é definida por $s(x) = w(x)\ mod\ g(x)$, onde $g(x)$ é o polinômio gerador do código $\mathcal{C}$. Em outras palavras*

$$w(x) = p(x)g(x) + s(x),$$

para algum $p(x)$.

Assumindo que $g(x)$ tenha grau igual a $n-k$, então $deg(s(x)) < n-k$. Como $w(x) = c(x)+e(x)$ e $c(x) = m(x)g(x)$, temos:

$$\begin{aligned} e(x) &= w(x) + c(x) \\ &= p(x)g(x) + s(x) + m(x)g(x) \\ &= (p(x) + m(x))g(x) + s(x). \end{aligned}$$

Assim, se $s(x) = 0$ então a palavra recebida $w(x)$ é a palavra-código enviada, caso contrário, $s(x) = w(x)\ mod\ g(x)$, ou ainda, pela definição original de síndrome, $s = Hw^T$.

Devido a estrutura cíclica do código, a síndrome polinomial $s(x)$ possui a seguinte propriedade:

Proposição 1.5.16. *[43] Seja $s(x)$ a síndrome de uma palavra recebida $r(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_{n-1}x^{n-1}$. Então, o resto $s^{(1)}$ que é resultado da divisão de $xs(x)$ pelo polinômio gerador $g(x)$ é a síndrome de $r^{(1)}(x)$, que é o desvio cíclico de $r(x)$.*

A demonstração pode ser encontrada em [43].

Exemplo 1.5.17. *Considere o* $[7,4]-$*código cíclico gerado por* $g(x) = 1 + x + x^3$. *Admita que o vetor* $r = (0010110)$ *tenha sido recebido. A síndrome de* r *é* $s = (101)$. *De fato, se considerarmos* r *e* s *como polinômios* $r(x)$ *e* $s(x)$ *respectivamente, notamos que:*

$$\begin{aligned} r(x) &= a(x)g(x) + s(x) \\ x^5 + x^4 + x^2 &= (x^2 + x + 1)(x^3 + x + 1) + x^2 + 1. \end{aligned}$$

e $s(x) = x^2 + 1$, *ou ainda,* $s = (101)$.

Usando o resultado explicitado na Proposição 1.5.16, podemos encontrar a síndrome $s^{(1)}$ *do desvio cíclico de* r, *denotado por* $r^{(1)} = (0001011)$. *De fato, sabemos que* $s^{(1)}(x)$ *é o resto da divisão de* $xs(x) = x(x^2 + 1) = x^3 + x$ *por* $g(x)$. *Temos que:*

$$\begin{aligned} xs(x) &= \tilde{a}(x)g(x) + s^{(1)}(x) \\ x^3 + x &= (x^3 + x + 1) + 1. \end{aligned}$$

Com isso, concluímos que $s^{(1)}(x) = 1$, *ou,* $s = (100)$.

Ao invés de construirmos um algoritmo de decodificação padrão para códigos cíclicos, basta aproveitarmos a grande vantagem oferecida pela estrutura presente em tais códigos. Vimos que $s(x) = e(x)\ mod\ g(x)$, então $x^i s(x) \equiv x^i e(x)\ mod\ g(x)$ e, usando a Proposição 1.5.16, a síndrome dos desvios cíclicos de $e(x)$ são fáceis de serem calculadas. É importante notar que se $deg(e(x)) < deg(g(x))$ então $e(x) = e(x)\ mod\ g(x)$ e a síndrome é o próprio erro.

No algoritmo 1.2 apresentamos uma maneira de decodificar um código cíclico, admitindo que a palavra $w(x)$ tenha sido recebida.

Algoritmo 1.2 Decodificação de um código cíclico

[30] Calcule a síndrome polinomial $s(x) = w(x)\ mod\ g(x)$, onde $w(x)$ é a palavra recebida.

Para cada $i \geq 0$, calcule $s_i \mapsto s_i(x) = x^i s(x)\ mod\ g(x)$ (que é a síndrome polinomial do $i-$ésimo desvio cíclico de w), até que seja encontrada uma síndrome s_j tal que $\omega(s_j) \leq t$, onde t é o número de erros cometidos.

Então, o erro polinomial provável é dado por $e(x) = x^{n-j} s_j(x)\ mod\ (x^n + 1)$.

Exemplo 1.5.18. *Seja* $\mathcal{C}$ *um código cíclico corretor de um erro (*$t = 1$*) gerado por* $g(x) = 1+x+x^3$ *com* $n = 7$. *Se* $w(x) = x^3 + x^2$ *é a palavra recebida, então*

$$s(x) = w(x)\ mod\ g(x) = x^3 + x^2\ mod\ (1 + x + x^3) = 1 + x + x^2$$

é a síndrome polinomial.

Calculamos também:

$$\begin{aligned} s_1(x) = xs(x)\ mod\ g(x) = x(1 + x + x^2)\ mod\ g(x) = 1 + x^2 \\ s_2(x) = x^2 s(x)\ mod\ g(x) = x(1 + x^2)\ mod\ g(x) = 1, \end{aligned}$$

que possui pedo $\omega = 1 \leq t$. Assim, $j = 2$ e, usando o Algoritmo 1.2:

$$e(x) = x^{7-2}s_2(x) \ mod \ (x^7 + 1) = x^5.$$

Logo, $c(x) = w(x) + e(x) = (x^2 + x^3) + x^5$ é a palavra-código mais provável de ter sido enviada.

Para terminar essa seção, vamos apresentar um exemplo que relaciona códigos de Hamming e códigos cíclicos.

Exemplo 1.5.19. *a) Para o $[7, 4]-$código binário cíclico $\mathcal{D}$ com polinômio gerador $g(x) = x^3 + x^2 + 1$, vamos encontrar a matriz cíclica padrão. Temos que $r = 7 - 4 = 3$, daí:*

$x^3 = (x^3 + x^2 + 1) + x^2 + 1$
$x^4 = (x^3 + x^2 + 1)(x + 1) + x^2 + 1$
$x^5 = (x^3 + x^2 + 1)(x^2 + x + 1) + x + 1$
$x^6 = x^3x^3 = (x^3 + x^2 + 1)(x^3 + x^2 + 1) + x^2 + x.$

Por meio dessas construções, temos que a matriz geradora na forma padrão é:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

b) Vamos encontrar agora a matriz geradora padrão para o $[15, 11]-$ código binário cíclico $\mathcal{E}$ que possui polinômio gerador $g(x) = x^4 + x + 1$. Observe que $r = 15 - 11 = 4$, logo:

$x^4 = (x^4 + x + 1) + x + 1$
$x^5 = (x^4 + x + 1)x + x^2 + 1$
$x^6 = (x^4 + x + 1)x^2 + x^3 + x^2$
$x^7 = (x^4 + x + 1)(x^3 + 1) + x^3 + x + 1$
$x^8 = (x^4 + x + 1)(x^4 + x + 1) + x^2 + 1$
$x^9 = (x^4 + x + 1)(x^5 + x^2 + x) + x^3 + x$
$x^{10} = (x^4 + x + 1)(x^6 + x^3 + x^2 + 1) + x^2 + x + 1$
$x^{11} = (x^4 + x + 1)(x^7 + x^4 + x^3 + x) + x^3 + x^2 + x$
$x^{12} = (x^4 + x + 1)(x^8 + x^5 + x^4 + x^2 + 1) + x^3 + x^2 + x + 1$
$x^{13} = (x^4 + x + 1)(x^9 + x^6 + x^5 + x^3 + x + 1) + x^3 + x^2 + 1$
$x^{14} = (x^4 + x + 1)(x^{10} + x^7 + x^6 + x^4 + x^2 + x + 1) + x^3 + 1.$

Traduzindo tais informações matricialmente, temos que a matriz geradora cíclica na forma padrão é:

$$G = \begin{pmatrix} I_{11\times 11} \\ B_{4\times 11} \end{pmatrix}, \ com \ B = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

c) Agora vamos mostrar que $\mathcal{D}$ e $\mathcal{E}$ são códigos de Hamming.

De fato, comparando as matrizes geradoras e observando que a matriz de paridade de $\mathcal{D}$ é

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = \mathcal{H}_3,$$

já que as colunas de H são todos os elementos não nulos de ${\mathbb{F}_2}^3$. Logo, $\mathcal{D}$ é um código de Hamming.

Para $\mathcal{E}$, por sua vez, temos a seguinte matriz de paridade:

$$H' = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = \mathcal{H}_4,$$

pois as colunas de H' compreendem todos os elementos de ${\mathbb{F}_2}^4 - \{0\}$. Assim, $\mathcal{E}$ é um código de Hamming, por definição.

Um código equivalente a um código cíclico não precisa ser necessariamente cíclico. Por exemplo, existem 30 códigos binários distintos $[7, 4]-$códigos de Hamming (a menos de reordenação de bits, o $[7, 4]-$código de Hamming é único, mas podemos obter versões distintas de um mesmo código por reordenação), mas, conforme vimos anteriormente, somente dois deles são cíclicos.

1.6 Códigos BCH

Os códigos BCH(Bose-Chaudhuri-Hocquenghem) formam uma vasta classe de códigos corretores de erros aleatórios. Os códigos BCH são, na verdade, uma generalização dos códigos de Hamming para correção de múltiplos erros [61]. Eles foram inicialmente descobertos por A. Hocqueghem [32] em 1959 e de maneira independente foram estudados por R. C. Bose e D. K. Ray-Chaudhuri [13] em 1960. Além disso, existe uma subclasse dos códigos BCH muito explorada, chamada de códigos de Reed-Solomon [59].

Nesta seção, vamos caracterizar e estudar as propriedades dos códigos BCH e elucidar um exemplo de um código BCH, que é o código de Reed-Solomon, usando como referências [43] e [12].

1.6.1 Definição e parâmetros de um código BCH

Os códigos BCH são um caso particular de códigos cíclicos, para os quais conseguimos uma cota para a sua distância mínima. Essa fato é útil não somente para se estimar a capacidade de correção de erros, mas também para se definir características particulares desse código. Aqui, usaremos a estrutura de corpos finitos, ou seja $\mathbb{F}_{q^m}$, com q primo e $m \in \mathbb{N}$.

Iniciamos com algumas definições algébricas importantes para a posterior definição de código BCH. Seja n o comprimento das palavras-código e assumimos que n não é divisível pela característica p do corpo finito $\mathbb{F}_q$.

A ordem $\mathcal{O}_n(q)$ no grupo das unidades de $\mathbb{Z}_n$ é indicada por m, assim, $q^m \equiv 1 \ mod \ n$.

De um elemento primitivo $\beta \in \mathbb{F}_{q^m}$, obtemos uma raiz $n-$ésima primitiva da unidade, ou seja,

$$\alpha := \beta^{(q^m-1)/n}.$$

Observamos que $\alpha^n = (\beta^{(q^m-1)/n})^n = \beta^{(q^m-1)} = 1$.

O conjunto de todas as raízes $n-$ésimas da unidade em $\mathbb{F}_{q^m}$ é

$$\mathcal{U}_n =< \alpha >= \{\kappa \in \mathbb{F}_{q^m} : \kappa^n = 1\}.$$

Definição 1.6.1. *Um subconjunto $W \subseteq \mathcal{U}_n$ é chamado **consecutivo** (com respeito a α) se existem inteiros $b \geq 0$ e $\delta \geq 2$ tais que*

$$W = \{\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta-2}\}.$$

Definição 1.6.2. *Seja $f \in \mathbb{F}_q[x]$ um polinômio não nulo. O conjunto de todas as raízes de f (em uma extensão de corpo de $\mathbb{F}_q$ conveniente) é chamada de **variedade** $V(f)$ **de** f.*

Definição 1.6.3. *Seja $\mathcal{C}$ um código cíclico sobre $\mathbb{F}_q$ com polinômio gerador $g(x)$. A variedade de g é também chamada de **variedade de $\mathcal{C}$ sobre** $\mathbb{F}_q$ e denotada por*

$$V(\mathcal{C}) := V(g).$$

Todo elemento de $V(\mathcal{C})$ é chamado de raiz de $\mathcal{C}$ sobre $\mathbb{F}_q$.

Isso se justifica, já que cada $a \in V(g)$ é uma raiz de todo $c(x), c \in \mathcal{C}$. Reciprocamente, $V(\mathcal{C})$ é formado por todas as raízes de todas as palavras-código de $\mathcal{C}$, em particular, as raízes de $g(x)$ estão compreendidas em $V(\mathcal{C})$.

A introdução dos códigos BCH foi motivada pelo seguinte resultado:

Proposição 1.6.4. *[12] Seja $\mathcal{C}$ um código cíclico de comprimento n sobre $\mathbb{F}_q$, onde q e n são coprimos. Assuma que uma variedade $V(\mathcal{C})$ contenha $\delta - 1$ potências consecutivas de α, que é uma raiz primitiva da unidade, com $\delta \geq 2$. Então a distância mínima de $\mathcal{C}$ é pelo menos δ. Em outras palavras,*

$$W := \{\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta-2}\} \subseteq V(\mathcal{C}) \Rightarrow d_{min}(\mathcal{C}) \geq \delta.$$

Demonstração. Consideramos a seguinte matriz de dimensão $(\delta - 1) \times n$:

$$\tilde{\Delta} := \begin{pmatrix} 1 & \alpha^b & \alpha^{2b} & \ldots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \ldots & \alpha^{(n-1)(b+1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha^{b+\delta-2} & \alpha^{2(b+\delta-2)} & \ldots & \alpha^{(n-1)(b+\delta-2)} \end{pmatrix}.$$

Observamos que essa matriz é uma matriz sobre a extensão $\mathbb{F}_{q^m}$ que contém α, onde $m = \mathcal{O}_n(q)$ e para cada $c \in \mathcal{C}$ temos que:

$$c \cdot \tilde{\Delta}^T = (c(\alpha^b), \ldots, c(\alpha^{b+\delta-2})) = (0, \ldots, 0) = 0.$$

Vamos verificar que cada subconjunto de $\delta - 1$ colunas de $\tilde{\Delta}$ é linearmente independente sobre $\mathbb{F}_q$. Para isso, consideramos a seguinte submatriz formada por $\delta - 1$ colunas de $\tilde{\Delta}$:

$$\begin{pmatrix} \alpha^{i_1 b} & \alpha^{i_2 b} & \ldots & \alpha^{i_{\delta-1} b} \\ \alpha^{i_1 (b+1)} & \alpha^{i_2 (b+1)} & \ldots & \alpha^{i_{\delta-1} (b+1)} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{i_1 (b+\delta-2)} & \alpha^{i_2 (b+\delta-2)} & \ldots & \alpha^{i_{\delta-1} (b+\delta-2)} \end{pmatrix},$$

com $0 \leq i_1 < i_2 < \cdots < i_{\delta-1} \leq n-1$. O determinante dessa matriz é $\alpha^{(i_1+i_2+\cdots+i_{\delta-1})b}$ vezes o determinante de Vandermonde da matriz

$$\begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha^{i_1} & \alpha^{i_2} & \ldots & \alpha^{i_{\delta-1}} \\ \alpha^{2i_1} & \alpha^{2i_2} & \ldots & \alpha^{2i_{\delta-1}} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{(\delta-2)i_1} & \alpha^{(\delta-2)i_2} & \ldots & \alpha^{(\delta-2)i_{\delta-1}} \end{pmatrix}.$$

Assim, o determinante é diferente de 0 já que α^{i_j} são dois a dois distintos. Isso mostra que quaisquer $(\delta-1)$ colunas de $\tilde{\Delta}$ são linearmente independentes sobre $\mathbb{F}_{q^m}$. Dessa forma, $\tilde{\Delta}$ é a matriz de paridade de um código $\tilde{\mathcal{C}}$ sobre $\mathbb{F}_{q^m}$ que possui distância mínima

$$d_{min}(\tilde{\mathcal{C}}) \geq \delta.$$

Mais do que isso, como $c \cdot \tilde{\Delta}^T = 0$ para todo $c \in \mathcal{C}$, obtemos a inclusão $\mathcal{C} \subseteq \tilde{\mathcal{C}}$ e temos ainda

$$d_{min}(\mathcal{C}) \geq d_{min}(\tilde{\mathcal{C}}) \geq \delta,$$

como queríamos. □

Códigos BCH são definidos como sendo o código cíclico maximal que contém um conjunto consecutivo W em sua variedade.

Definição 1.6.5. *Seja $W := \{\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta-2}\}$ um subconjunto consecutivo de $\mathcal{U}_n$, para algum $b \geq 0$ e algum δ com $n > \delta \geq 2$. Defina o polinômio*

$$g := mmc\{M_{\alpha^{b+i}} : i \in \delta - 1\},$$

onde $M_{\alpha^{b+i}}$ é o polinômio minimal de α^{b+i} em $\mathbb{F}_q$. O código $\mathcal{C}$ com polinômio gerador g é chamado ***código BCH gerado por*** *W. O valor de δ é a* ***distância designada*** *de $\mathcal{C}$, já que $d_{min}(\mathcal{C}) \geq \delta$. Se $n = q^m - 1$, o código é chamado de* ***primitivo****, já que nesse caso α é também um elemento primitivo para $\mathbb{F}_{q^m}$. Além disso, se $b = 1$, dizemos que $\mathcal{C}$ é um* ***código BCH no sentido estrito****.*

1.6.2 Códigos de Reed-Solomon: um exemplo de código BCH

Definição 1.6.6. *Um código BCH de comprimento $n = q - 1$ sobre $\mathbb{F}_q$ é chamado de* ***código de Reed-Solomon****.*

Códigos de Reed-Solomon são particularmente fáceis de serem criados, já que no caso de $n = q - 1$ os polinômios minimais são lineares. Especialmente nesse caso, o corpo $\mathbb{F}_q$ contém todas as raízes $n-$ésimas da unidade e $M_{\alpha^i} = (x - \alpha^i) \ \ \forall\ i$.

Exemplo 1.6.7. *Vamos caracterizar um código BCH que pode corrigir até dois erros. Para isso, precisamos de distância mínima de pelo menos* 5*, ou seja, definimos a distância designada como* $\delta = 5$.

Decidimos usar um código de Reed-Solomon e considerar $q = n - 1$, logo, $q - 1 = n > \delta = 5$. Escolhemos então $n = 6$ e $q = 7$, por exemplo. Um elemento primitivo módulo 7 é $\beta = 3$ e considerando $b = 1$ temos:

$$W := \{3, 3^2, 3^3, 3^4\} = \{3, 2, 6, 4\} \subset \mathbb{F}_7.$$

Portanto, o código Reed-Solomon $\mathcal{C}$ gerado pelo conjunto consecutivo W possui polinômio gerador:

$$\begin{aligned} g(x) &= (x-3)(x-3^2)(x-3^3)(x-3^4) \\ &= (x-3)(x-2)(x-6)(x-4) \\ &= x^4 + 6x^3 + 3x^2 + 2x + 4. \end{aligned}$$

Como $deg(g(x) = 4 = n - k$, temos que esse código é um $[6, 2]-$código cíclico, com matriz geradora:

$$G = \begin{pmatrix} 4 & 0 \\ 2 & 4 \\ 3 & 2 \\ 6 & 3 \\ 1 & 6 \\ 0 & 1 \end{pmatrix}.$$

Sabemos que os códigos BCH são códigos cíclicos, então podemos aplicar a teoria vista na seção anterior para encontrar o polinômio verificador e também a matriz de paridade desse código. Temos que:

$$\begin{aligned} g(x)h(x) &= x^6 - 1 \\ (x^4 + 6x^3 + 3x^2 + 2x + 4)h(x) &= x^6 - 1 \\ \Rightarrow h(x) &= x^2 + x + 5. \end{aligned}$$

Assim, segue que a matriz de paridade é dada por:

$$H = \begin{pmatrix} 1 & 1 & 5 & 0 & 0 & 0 \\ 0 & 1 & 1 & 5 & 0 & 0 \\ 0 & 0 & 1 & 1 & 5 & 0 \\ 0 & 0 & 0 & 1 & 1 & 5 \end{pmatrix}.$$

1.7 Códigos de Goppa

Os códigos de Goppa, propostos por V. G. Goppa em 1970, formam uma classe de códigos corretores de erros com características peculiares que permitem, finalmente, fundamentar um sistema criptográfico. É importante destacar que os códigos de Goppa são uma generalização dos códigos BCH, com o diferencial de oferecerem um número muito maior de códigos, fato esse que permite a implementação de criptossistemas como o de McEliece, introduzido por R. J. McEliece, no ano de 1978.

Nesta seção, vamos descrever e caracterizar os códigos de Goppa, bem como apresentar seus métodos de codificação e decodificação, usando como base as obras de [39], [38], [31] e [12].

1.7.1 Definição e parâmetros de um código de Goppa

Definição 1.7.1. *Considere o polinômio $g(x)$ de grau t sobre uma extensão $\mathbb{F}_{q^m}$ do corpo $\mathbb{F}_q$, q primo, de modo que:*

$$g(x) = g_0 + g_1 x + \cdots + g_t x^t = \sum_{i=0}^{t} g_i x^i$$

*Então, $g(x)$ é chamado de **polinômio de Goppa**.*

O código de Goppa $\Gamma(L, g(x))$ é definido por $g(x)$, que é o polinômio de Goppa conforme enunciado acima, e por $L = \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \subset \mathbb{F}_{q^m}$, tal que $g(\alpha_i) \neq 0 \ \ \forall \alpha_i \in L$.

Definição 1.7.2. *A um vetor $c = (c_0, c_1, \ldots, c_n) \in \mathbb{F}_q^n$ associamos a função:*

$$R_c(x) = \sum_{i=1}^{n} \frac{c_i}{x - \alpha_i} \tag{1.7.1}$$

*na qual $\dfrac{1}{x-\alpha_i}$ é o único polinômio que satifaz $(x-\alpha_i)\dfrac{1}{(x-\alpha_i)} \equiv 1 (mod \ \ g(x))$. Tal função é chamada de **função síndrome**.*

Munidos de tais informações, podemos finalmente definir um código de Goppa:

Definição 1.7.3. *O **código de Goppa** $\Gamma(L, g(x))$ consiste em todos os vetores $c \in \mathbb{F}_q^n$ tais que*

$$R_c(x) \equiv 0 (mod \ \ g(x)). \tag{1.7.2}$$

Isso significa que, quando o lado esquerdo da Equação 1.7.2 for escrito como uma função racional, o numerador deve ser um múltiplo de $g(x)$ (trabalhar em módulo $g(x)$ é como trabalhar no anel $\mathbb{F}_{q^m}[x]/ < g(x) >$ e a hipótese que $g(\alpha_i) \neq 0 \ \ \forall \alpha_i \in L$ garante que $x - \alpha_i$ é invertível nesse anel).

Vamos encontrar agora a matriz de paridade de um código de Goppa $\Gamma(L, g(x))$ para, em seguida, falar sobre a decodificação de tal código.

Observamos que:

$$\frac{1}{x-\alpha_i} \equiv -\frac{1}{g(\alpha_i)} \frac{g(x) - g(\alpha_i)}{x - \alpha_i} (mod \ \ g(x)),$$

já que, comparando os numeradores, temos $1 \equiv -g(\alpha_i)^{-1}(g(x) - g(\alpha^i))(mod \ \ g(x))$.

Pela Definição 1.7.2, sabemos que $c = (c_0, c_1, \ldots, c_n) \in \Gamma(L, g(x))$ se e somente se

$$\sum_{i=1}^{n} \frac{c_i}{x - \alpha_i} \equiv 0(mod \ \ g(x)),$$

ou seja,

$$\sum_{i=1}^{n} c_i \frac{(g(x) - g(\alpha_i))}{x - \alpha_i} g(\alpha_i)^{-1} \equiv 0(mod \ \ g(x)).$$

Sabemos que $g(x) = \sum_{j=0}^{t} g_j x^j$, com $g_j \in \mathbb{F}_{q^m}$, onde $t = deg(g(x)$. Então

$$\begin{aligned} \frac{(g(x) - g(\alpha_i))}{x - \alpha_i} g(\alpha_i)^{-1} &= g(\alpha_i)^{-1} \sum_{j=0}^{t} g_j \sum_{k=0}^{j-1} x^k \alpha_i^{j-1-k} \\ &= g(\alpha_i)^{-1} \sum_{k=1}^{t-1} x^k \left(\sum_{j=k+1}^{t} g_j \alpha_i^{j-1-k} \right). \end{aligned}$$

Fazendo os coeficientes de x^k iguais a zero na ordem $k = t-1, t-2, \ldots, 0$, temos que $c \in \Gamma(L, g(x))$ se e somente se $Hc^t = 0$, onde

$$H = \begin{pmatrix} h_0 g_t & h_1 g_t & \ldots & h_{n-1} g_t \\ h_0(g_{t-1} + g_t \alpha_1) & h_1(g_{t-1} + g_t \alpha_2) & \ldots & h_{n-1}(g_{t-1} + g_t \alpha_n) \\ \vdots & \vdots & \vdots & \vdots \\ h_0 \sum_{j=1}^{t} \alpha_1^{j-1} & h_1 \sum_{j=1}^{t} \alpha_2^{j-1} & \ldots & h_{n-1} \sum_{j=1}^{t} \alpha_n^{j-1} \end{pmatrix},$$

com $h_i = g(\alpha_i)^{-1}$. Por meio de operações por linhas, é possível reduzir H para uma matriz H' com dimensão $t \times n$, tal que:

$$H' = \begin{pmatrix} g(\alpha_1)^{-1} & g(\alpha_2)^{-1} & \ldots & g(\alpha_n)^{-1} \\ g(\alpha_1)^{-1}\alpha_1 & g(\alpha_2)^{-1}\alpha_2 & \ldots & g(\alpha_n)^{-1}\alpha_n \\ \vdots & \vdots & \vdots & \vdots \\ g(\alpha_1)^{-1}\alpha_1^{t-1} & g(\alpha_2)^{-1}\alpha_2^{t-1} & \ldots & g(\alpha_n)^{-1}\alpha_n^{t-1} \end{pmatrix}.$$

As entradas de H' estão em $\mathbb{F}_{q^m}$. Escolhendo uma base de $\mathbb{F}_{q^m}$ sobre $\mathbb{F}_q$, cada elemento de $\mathbb{F}_{q^m}$ pode ser representado como um vetor coluna $m \times 1$ sobre $\mathbb{F}_q$. Substituindo cada entrada de H' por seu vetor coluna correspondente, obtemos uma matriz $mt \times n$ H'' sobre $\mathbb{F}_q$ com a propriedade de que $c \in \mathbb{F}_q^n$ está em $\Gamma(L, g(x))$ se e somente se $H''c^T = 0$.

O comprimento n das palavras-código c de um código de Goppa são fixados por L e para os outros dois parâmetros, limites inferiores podem ser derivados por meio de suas cotas, conforme veremos a seguir.

Proposição 1.7.4. *[38] O código de Goppa $\Gamma(L, g(x))$ de tamanho n é um código linear sobre $\mathbb{F}_q$ com as seguintes propriedades:*

i) a dimensão do código satisfaz $k \geq n - mt$;

ii) a distância mínima do código satisfaz $d \geq t + 1$.

Demonstração. As linhas de H'' podem ser dependentes, já que o posto dessa matriz é no máximo mt. Além disso, $\Gamma(L, g(x))$ possui dimensão de pelo menos $n - mt$. Se uma palavra-código $c \in \Gamma(L, G(x))$ possui peso t ou menos, então o lado esquerdo da Equação 1.7.2 pode ser escrito como uma função racional, onde o numerador possui grau $t - 1$ ou menos. Porém, esse numerador deve ser um múltiplo de $g(x)$, o que é impossível, já que $deg(g(x)) = t$. □

Agora, vamos analisar um caso específico para o código de Goppa. Queremos que a distância mínima seja a maior possível para um código poder corrigir r erros se $2r + 1 \leq d$. Isso ocorre quando $\Gamma(L, g(x))$ é um código de Goppa binário com polinômio $g(x)$ sobre $\mathbb{F}_{2^m}$ de grau t que não possui raízes com multiplicidade maior do que um. Um polinômio com essas características é chamado de *separável*.

Proposição 1.7.5. *[39] Seja* $\Gamma(L, g(x))$ *um código de Goppa binário com polinômio separável* $g(x)$ *de grau* t. *Então,* $\Gamma(L, g(x))$ *possui distância mínima* d *de pelo menos* $2t + 1$.

A matriz de paridade H é utilizada para corrigir erros, mas além dela, precisamos conhecer a matriz geradora para codificar e decodificar mensagens. Uma palavra-código é formada pela multiplicação de G por uma mensagem $m = (m_1, m_2, \ldots, m_k)$. Depois disso, a palavra-código pode ser corrigida usando a igualdade

$$c \cdot H^T = 0 \quad \text{ou} \quad H \cdot c^T = 0,$$

para todo $c \in \Gamma(L, g(x))$.

Por conseguinte, a equação determinada por

$$GH^T = 0 \quad \text{ou} \quad HG^T = 0,$$

pode ser usada para encontrar G a partir de H. Os vetores do espaço nulo de H módulo q formam o espaço das linhas de G.

Para explicar esse processo, nada melhor do que ilustrar a teoria com um exemplo.

Exemplo 1.7.6. *[12] O polinômio* $g(x) = x^2 + x + 1 \in \mathbb{F}_2[x]$ *é separável, ou seja, não possui raízes múltiplas. Para construirmos um código de Goppa, consideramos uma extensão* $\mathbb{F}_8 = \mathbb{F}_{2^3}$ *de* $\mathbb{F}_2$. *Usamos o polinômio irredutível* $x^3 + x^2 + 1$ *para gerar esse corpo. Seja* α *uma raiz desse polinômio, tal que* $\alpha^3 = \alpha^2 + 1$. *Escrevemos então os elementos de* $\mathbb{F}_8$ *como:*

$$\begin{aligned}
&\kappa_0 = 0,\\
&\kappa_1 = \alpha^0 = 1,\\
&\kappa_2 = \alpha^1 = \alpha,\\
&\kappa_3 = \alpha^2 = \alpha^2,\\
&\kappa_4 = \alpha^3 = \alpha^2 + 1,\\
&\kappa_5 = \alpha^4 = \alpha^3 + \alpha = \alpha^2 + \alpha + 1,\\
&\kappa_6 = \alpha^5 = \alpha^3 + \alpha^2 + \alpha = \alpha + 1,\\
&\kappa_7 = \alpha^6 = \alpha^2 + \alpha.
\end{aligned}$$

O código de Goppa com $L = \{\kappa_0, \ldots, \kappa_7\}$ *e* $g(x) = x^2 + x + 1$ *é o conjunto de* $c \in \mathbb{F}_8^8$ *tal que*

$$\sum_{i \in 8} \frac{c_i}{x - \kappa_i} \equiv 0 (mod \ \ g(x)).$$

Para encontrar a matriz de paridade desse código, observamos que módulo $g(x)$*, temos:*

$$\frac{1}{x} \equiv x + 1, \ \frac{1}{x+1} \equiv x, \ \frac{1}{x+\alpha} \equiv \alpha^3 x + \alpha, \ \frac{1}{x+\alpha^2} \equiv \alpha^6 x + \alpha^2,$$

$$\frac{1}{x+\alpha^3} \equiv \alpha^6 x + \alpha, \ \frac{1}{x+\alpha^4} \equiv \alpha^5 x + \alpha^4, \ \frac{1}{x+\alpha^5} \equiv \alpha^3 x + \alpha^4,$$

$$\frac{1}{x+\alpha^6} \equiv \alpha^5 x + \alpha^2 (mod \ \ g(x)).$$

Assim, a matriz de paridade é construída por:

$$\begin{aligned} H'' &= \begin{pmatrix} 1 & 0 & \alpha & \alpha^2 & \alpha & \alpha^4 & \alpha^4 & \alpha^2 \\ 1 & 1 & \alpha^3 & \alpha^6 & \alpha^6 & \alpha^5 & \alpha^3 & \alpha^5 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & \alpha & \alpha^2 & \alpha & \alpha^2+\alpha+1 & \alpha^2+\alpha+1 & \alpha^2 \\ 1 & 1 & \alpha^2+1 & \alpha^2+\alpha & \alpha^2+\alpha & \alpha+1 & \alpha^2+1 & \alpha+1 \end{pmatrix}. \end{aligned}$$

Usando uma base de $\mathbb{F}_8$ *sobre* $\mathbb{F}_2$ *que consiste nos elementos* $\alpha^2, \alpha, 1$*, podemos escrever a matriz* H'' *sobre* $\mathbb{F}_2$ *como:*

$$H' = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

Essa matriz possui posto 6 *e usando eliminação de Gauss podemos ainda reescrevê-la como:*

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix},$$

e a matriz geradora para o código de Goppa é dada por:

$$G = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}.$$

É fácil observar que o código tem distância mínima 5, *o que atende o resultado da Proposição 1.7.5. Assim, encontramos um* $(8, 2, 5)-$ *código de Goppa.*

1.7.2 Codificação e decodificação de um código de Goppa

Seja $\Gamma(L, g(x))$ um código de Goppa definido por $g(x)$, um polinômio de grau t sobre $\mathbb{F}_{q^m}$ e um conjunto $L \subset \mathbb{F}_{q^m}$ de tamanho n. Seja k a dimensão de $\Gamma(L, g(x))$ e d sua distância mínima.

Codificar uma mensagem significa escrevê-la em blocos com k símbolos e multiplicar cada um desses blocos pela matriz geradora G, ou seja,

$$(c_0, c_1, \ldots, c_n) = G \cdot (m_0, m_1, \ldots, m_k).$$

Seja y uma palavra recebida, contendo r erros,com $2r + 1 \leq d$ ou $r \leq \lfloor \frac{t}{2} \rfloor$ no caso em que $d = t + 1$, que é a cota inferior da distância mínima. Então

$$(y_1, y_2, \ldots, y_n) = (c_1, c_2, \ldots, c_n) + (e_1, e_2, \ldots, e_n),$$

com $e_i \neq 0$ em exatamente r posições. Para corrigir a palavra e encontrar a palavra-código retificada c, devemos encontrar o vetor do erro e e depois exibir

- o conjunto de localização dos erros $B = \{i : e_i \neq 0\}$;
- os valores correspondentes ao erro e_i para $i \in B$.

Para atingir esse objetivo, vamos definir os seguintes polinômios:

Definição 1.7.7. *O* ***polinômio localizador*** *do erro* $\sigma(x)$ *e o* ***polinômio avaliador do erro*** $\varepsilon(x)$ *são dados, respectivamente, por:*

$$\sigma(x) = \prod_{i \in B} (x - \alpha_i) \tag{1.7.3}$$

$$\varepsilon(x) = \sum_{i \in B} e_i \prod_{j \in B,\ j \neq i} (x - \alpha_j) \tag{1.7.4}$$

Detalhando tal definição, é fácil ver que a localização dos erros está inteiramente relacionada com as raízes de $\sigma(x)$, já que $B = \{i : \alpha_i$ é uma raiz de $\sigma(x)\}$. Na proposição a seguir, algumas propriedades de $\sigma(x)$ e $\varepsilon(x)$ serão expostas e apresentaremos uma fórmula para descrever os valores dos erros, bem como uma relação entre $\sigma(x), \varepsilon(x)$ e a síndrome $s(x)$ da palavra recebida, onde

$$\begin{aligned} s(x) &= \sum_{i=1}^{n} \frac{y_i}{x - \alpha_i} = \sum_{i=1}^{n} \frac{c_i + e_i}{x - \alpha_i} = \sum_{i=1}^{n} \frac{c_i}{x - \alpha_i} + \sum_{i=1}^{n} \frac{e_i}{x - \alpha_i} \\ &\equiv \sum_{i \in B} \frac{e_i}{x - \alpha_i} (mod\ g(x)). \end{aligned}$$

Proposição 1.7.8. *[39] Seja e o erro de um vetor de peso r tal que* $r \leq \lfloor \frac{t}{2} \rfloor$. *Sejam* $\sigma(x), \varepsilon(x)$ *e* $s(x)$ *os polinômios definidos conforme acima. Então, as seguintes propriedades são válidas:*

i) $deg(\sigma(x)) = r$;

ii) $deg(\varepsilon(x)) \leq r - 1$;

iii) $mdc\ (\sigma(x), \varepsilon(x)) = 1$;

iv) $e_k = \dfrac{\varepsilon(\alpha_k)}{\sigma'(\alpha_k)}, \quad k \in B$;

v) $\sigma(x)s(x) \equiv \varepsilon(x)(mod\ g(x))$.

Para corrigir erros em uma palavra-código, devemos resolver a seguinte equação

$$\sigma(x)s(x) \equiv \varepsilon(x)(mod\ g(x)). \tag{1.7.5}$$

Como $g(x)$ é conhecido e a síndrome $s(x)$ pode ser calculada, precisamos resolver um sistema de t equações com $2r$ incógnitas, a dizer $\sigma_0, \sigma_1, \ldots, \sigma_{r-1}$ e $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{r-1}$, onde $\sigma(x) = \sigma_0 + \sigma_1 x + \cdots + \sigma_{r-1}x^{r-1} + x^r$ e $\varepsilon(x) = \varepsilon_0 + \varepsilon_1 x + \cdots + \varepsilon_{r-1}x^{r-1}$. Já que $2r \leq t$, sabemos que a solução é única. Portanto, estamos preparados para desenvolver um algoritmo geral para a correção de erros de um código de Goppa, que será exposto no Algoritmo 1.3.

Note que as observações no passo (2) do algoritmo a seguir decorrem das definições explicitadas em 1.7.7, usando $e_i = 1 \ \ \forall\ i \in B$, desde que o código seja binário.

Algoritmo 1.3 Corrigindo $r \leq \lfloor \frac{t}{2} \rfloor$ erros em um código de Goppa

---

[39] Seja $y = (y_1, \ldots, y_n)$ uma palavra recebida, que contém r erros com $2r \leq t$.

(1) Calcule a síndrome $s(x) = \displaystyle\sum_{i=1}^{n} \frac{y_i}{x - \alpha_i}$.

(2) Resolva a equação chave $\sigma(x)s(x) \equiv \varepsilon(x)(mod\ g(x))$, escrevendo

$$\sigma(x) = \sigma_0 + \sigma_1 x + \cdots + \sigma_{r-1}x^{r-1} + x^r,$$

$$\varepsilon(x) = \varepsilon_0 + \varepsilon_1 x + \cdots + \varepsilon_{r-1}x^{r-1}$$

e resolvendo o sistema auxiliar de t equações e $2r$ incógnitas. Se o código é binário, podemos tomar $\varepsilon(x) = \sigma'(x)$.

(3) Determine o conjunto de localização dos erros $B = \{i : \sigma(\alpha_i) = 0\}$.

(4) Calcule os valores dos erros $e_i = \dfrac{\varepsilon(\alpha_i)}{\sigma'(\alpha_i)}, \ \ \forall\ i \in B$.

(5) O vetor $e = (e_1, \ldots, e_n)$ é definido por e_i para $i \in B$ e 0 nas demais posições.

(6) A palavra-código enviada é $c = y - e$.

Capítulo 2

Criptografia baseada em códigos: o sistema criptográfio McEliece

Vamos agora abordar uma aplicação dos códigos corretores de erros em criptografia. O primeiro algoritmo desenvolvido nesse formato, ou seja, usando códigos corretores de erros em sua fundamentação, foi proposto por Robert J. McEliece, em 1978, em [45] e leva seu nome. Curiosamente esse mesmo algoritmo, após alguns ajustes, continua sendo um dos mais interessantes no momento, mesmo considerando algumas desvantagens em sua implementação, já que nenhum ataque conhecido representa um risco concreto à sua segurança, respeitando certos parâmetros.

Iniciamos esse capítulo com uma breve introdução à criptografia de chaves públicas e a seguir buscamos compreender a teoria por trás do criptossistema McEliece, explorar as vantagens e desvantagens desse sistema já antigo, porém tão atual visto sob a ótica de sua eficiência algorítmica e apresentar algumas variações desse método que foram desenvolvidas com o propósito de amenizar as desvantagens do sistema original. Como embasamento teórico, foram usadas as referências [55], [46], [39] e [8].

2.1 Criptografia de chaves públicas

Vamos iniciar esse capítulo de aplicações estabelecendo algumas noções iniciais de criptografia, noções essas que possuem papel essencial no que se segue. Até agora muito se mencionou sobre a criptografia de chaves públicas. Nos questionamos então: como ela funciona?

Seja $\{E_e : e \in \kappa\}$ um conjunto de transformações de encriptação e $\{D_d : d \in \kappa\}$ um conjunto de transformações de decriptação correspondente, onde κ é o espaço da chaves. Consideramos um par de transformações de encriptação e decriptação associados (E_e, D_d) e admitimos que cada par possua a seguinte propriedade:

> Dado um texto cifrado $c \in \mathcal{C}$, onde $\mathcal{C}$ é o espaço dos textos crifrados, se conhecermos E_e deve ser computacionalmente inviável encontrar a mensagem $m \in \mathcal{M}$, onde $\mathcal{M}$ é o espaço das mensagens, tal que $E_e(m) = c$.

Essa propriedade indica que dado e é inviável determinar a decriptação correspondente d.

Sob essas hipóteses considere uma comunicação entre Bob e Alice conforme ilustrada na Figura 2.1:

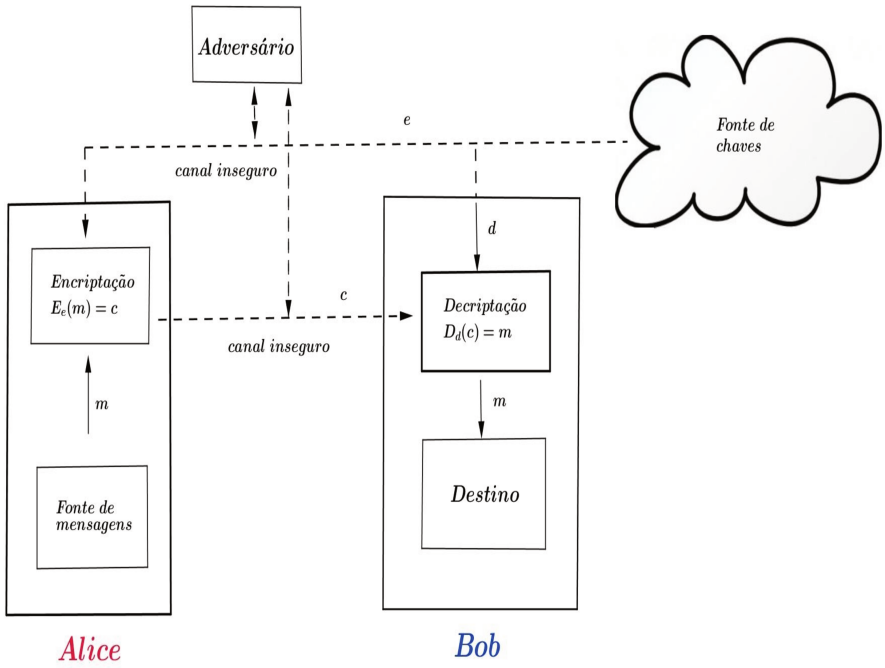


Figura 2.1: Criptografia de chaves públicas

Fonte: [46], p.26.

Bob seleciona seu par de chaves (e, d) e envia a chave de encriptação e (chamada de *chave pública*) para Alice por meio de um canal qualquer, mas ele mantém em segredo a chave de decriptação d (chamada de *chave privada*) segura e secreta. Alice, em seguida, envia o texto cifrado c para Bob que foi obtido a partir de sua mensagem original m, aplicando a transformação de encriptação determinada pela chave pública de Bob, ou seja, $c = E_e(m)$. Bob ao receber a mensagem, decripta o texto cifrado c aplicando a transformação inversa D_d unicamente determinada por d.

Outra noção importante à nossa pesquisa é a de complexidade. Os problemas computacionais, por sua vez, podem ser divididos em três classes [19]:

P: É a classe de problemas que podem ser resolvidos por algum algoritmo em um certo número de passos que está limitado por um polinômio. De modo mais específico, os problemas pertencentes à essa classe podem ser resolvidos em tempo $\mathcal{O}(n^k)$ para alguma constante k, onde n é o tamanho dos dados de entrada do problema. Aqui se encontra, por exemplo, a resolução de sistemas lineares.

NP: A classe NP consiste nos problemas que são verificáveis em tempo polinomial, onde denotamos por problema verificável um problema onde, se estivéssemos de alguma forma

munidos de uma possível solução, poderíamos verificar se ela está correta em tempo polinomial. Problemas clássicos de combinatória, como o problema de decisão associado ao problema do caixeiro viajante, se enquadram nessa classe.

Observe que todo problema P é NP já que se um problema está na classe P então podemos resolvê-lo em tempo polinomial sem que seja necessário uma possível solução, ou seja, P $\subseteq$ NP. A questão aberta é se P é ou não um subconjunto próprio de NP.

Para falar da última classe, definimos os problemas NP-difíceis. Um problema X é NP-díficil se for possível reduzir em tempo polinomial qualquer problema da classe NP ao problema X. Assim, se X for resolvido será também possível resolver todo problema NP.

NP-completo: Um problema está na classe dos problemas NP-completos se ele for NP e se for NP-difícil.

O termo "NP-completo" é essencial ao nosso estudo, pois se estabelecermos um problema na classe dos NP-completos fornecemos boas evidências para a sua intratabilidade.

2.2 A proposta original de McEliece

A versão original, proposta por McEliece, em 1978 tem sua ideia principal em primeiro selecionar um código particular para o qual se possui um algoritmo eficaz de decodificação (quando comparado ao processo de decodificação geral) e então disfarçá-lo como um código linear geral.

Códigos aleatórios não aparecem com qualquer estrutura que os caracterize, logo, um atacante que não conhece a chave secreta se depara com o *problema de decodificação geral* (GDP - *General Decoding Problem*):

> Seja $\mathbb{F}_q$ um corpo finito e seja (G, w, c) uma tripla consistindo de uma matriz geradora $G \in \mathbb{F}_q^{k\times n}$, um número inteiro $w < n$ e um vetor $c \in \mathbb{F}_q^n$. O problema de decodificação geral consiste em determinar se existe um vetor $m \in \mathbb{F}_q^k$ tal que $e = c - mG$ tenha peso $\omega(e) \leq w$.

Ou seja, o atacante deve encontrar a palavra-código mais próxima em um código linear $\mathcal{C}$ a um vetor dado, assumindo que exista uma única palavra-código que satisfaça essa condição.

Além desse problema, na estrutura de códigos corretores de erros encontramos também o *problema de decodificação por síndromes* (SDP - *Sydrome Decoding Problem*):

> Seja $\mathbb{F}_q$ um corpo finito e seja (H, w, s) uma tripla consistindo de uma matriz de paridade $H \in \mathbb{F}_q^{r\times n}$, um número inteiro $w < n$ e um vetor $s \in \mathbb{F}_q^r$. O problema de decodificação por síndromes consiste em determinar se existe um vetor $e \in \mathbb{F}_q^n$ com peso de Hamming $\omega(e) \leq w$ tal que $He^t = s^t$.

Berlekamp, McEliece e Van Tilborg provaram em [10], que o *problema de decodificação geral* e o *problema de decodificação por síndromes* são NP-completos. Além disso, pesquisadores consideram ainda esses problemas como sendo de difícil resolução na média e não somente para os piores parâmetros ([57]).

O criptossistema de McEliece está baseado no problema GDP e se apropria da descrição do código original como chave privada e a descrição do código transformado é usada como chave pública.

A estrutura desse sistema, que envolve regras para geração de chaves, encriptação de mensagens e decriptação de textos cifrados, se dá conforme descrito no Algoritmo 2.1.

Algoritmo 2.1 O sistema criptográfico McEliece para um código binário arbitrário

---

[46] Seja Γ um código binário arbitrário $t-$corretor de erros de comprimento n e dimensão k sobre $\mathbb{F}_2$ para o qual se conhece um algoritmo eficiente de decodificação.

Geração de chaves: Calcule a matriz geradora G com elementos em $\mathbb{F}_2$ e dimensão $k \times n$ para o código Γ. Escolha uma matriz S com coeficientes em $\mathbb{F}_2$, não singular e com dimensão $k \times k$. Escolha uma matriz de permutação P com dimensão $n \times n$. Calcule $\hat{G} = SGP$ e defina:

Chave privada: S, G, P

Chave pública: $\hat{G}, t$.

Encriptação de uma mensagem $m \in {\mathbb{F}_2}^k$: escolha um vetor arbitrário $e \in {\mathbb{F}_2}^n$ com até t erros e calcule $c = m\hat{G} + e \in {\mathbb{F}_2}^n$.

Decriptação de um texto cifrado $c \in {\mathbb{F}_2}^n$: calcule $\hat{c} = cP^{-1}$ e use o algoritmo eficiente de decodificação para o código Γ gerado por G para decodificar $\hat{c}$ para $\hat{m}$. Calcule $m = \hat{m}S^{-1}$.

Observamos que tal decodificação de fato funciona, pois:

$$\hat{c} = cP^{-1} = (m\hat{G} + e)P^{-1} = (mSGP + e)P^{-1} = (mS)G + eP^{-1},$$

e eP^{-1} é um vetor com até t erros por definição então o algoritmo de decodificação para o respectivo código gerado por G corrige $\hat{c}$ para $\hat{m} = mS$. Finalmente, $\hat{m}S^{-1} = m$ e recuperamos a mensagem original.

Vamos ilustrar tal processo por meio de um exemplo.

Exemplo 2.2.1. *Consideramos o código de Hamming $\mathcal{H}_3$ corretor de um erro ($t = 1$) com $n = 7$ e $k = 4$. Vamos considerar para esse exemplo a matriz geradora em sua forma linha para que não sejam necessárias alterações no algoritmo original.*

A matriz geradora desse código é dada por:

$$G = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right).$$

Escolhemos as matrizes S (inversível e com coeficientes em $\mathbb{F}_2$) e P (matriz de permutação) como sendo:

$$S = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \qquad P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Calculamos agora o valor de $\hat{G} = SGP$*, que será parte da nossa chave pública. Temos que:*

$$\hat{G} = SGP = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Munidos de tais informações, podemos agora identificar os dados desse sistema criptográfico:

Chave privada: S, G, P.

Chave pública: $\hat{G}, t = 1$.

Alice deseja enviar a mensagem $m = (1101)$ *para Bob, que já enviou à ela a chave pública* $\hat{G}, t$*. Alice inicia escolhendo um vetor com no máximo um erro* $e = (1000000)$ *e encripta sua mensagem calculando:*

$$c = m\hat{G} + e = (0111000) + (1000000) = (1111000)$$

e envia o texto cifrado c *para Bob.*

Ao receber c*, Bob calcula* $\hat{c} = cP^{-1} = cP^T = (1100101)$ *e sabe como decodificar códigos que corrigem um erro, como é o caso do código de Hamming* $\mathcal{H}_3$*. A partir da sua chave privada* G*, ele obtém a matriz de paridade*

$$H = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Calculando a síndrome $cH = (111)$*, Bob observa que tal vetor corresponde à primeira linha de* H*, assim, o erro ocorreu na primeira coordenada e ele corrige* $\hat{c}$ *para* $\hat{m} = (0100)$*, considerando somente os quatro primeiros bits. Por fim, basta calcular* $m = \hat{m}S^{-1}$*, onde*

$$S^{-1} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Portanto, $m = (1101)$*, que foi exatamente a mensagem enviada por Alice.*

Note que esse exemplo é algo simples e o fato do código corrigir um erro facilitou a decodificação, que se dá usando a própria matriz de paridade (que pode ser obtida imediatamente a partir da matriz geradora). Contudo, de um modo geral os esquemas criptográficos baseados em códigos estão fundamentados em propriedades muito específicas do código quanto à decodificação.

Um dos códigos comumente usados no sistema McEliece, é o código de Goppa, por possuir uma vasta família de códigos, proporcionar um grande número de chaves públicas em potencial e também, por admitir um algoritmo de decodificação eficiente que opera em tempo polinomial.

Muitos pesquisadores vem buscando alternativas para substituir os códigos de Goppa por outros códigos. Por exemplo, em 1986, Niederreiter pensou em usar os códigos de Reed-Solomon para embasar o criptossistema de McEliece, conforme exposto em [53], porém, aproximadamente um ano depois, tal sistema foi quebrado, por mostrar-se vulnerável e inseguro.

Vamos elencar agora algumas vantagens e desvantagens do uso e implementação do sistema criptográfico McElice.

VANTAGENS

- [55] O sistema é elegante é de fácil compreensão.
- [55] Sua segurança tem sido estudada desde sua introdução, em 1978 e até hoje nenhum ataque representou uma ameaça potencial ao seu desempenho.
- [55] A implementação de sua encriptação e decriptação é muito rápida e possui baixa complexidade, comparada a outros criptossistemas baseados em códigos.

DESVANTAGENS

- [39] O sistema é facilmente quebrado quando encriptações múltiplas de uma mesma mensagem são feitas. Uma das desvantagens do sistema criptográfico McEliece é que ele não é seguro para o envio de uma mesma mensagem repetidas vezes com a mesma matriz de encriptação $\hat{G}$.
- [68] O tamanho das chaves públicas usadas no sistema é muito grande comparado à outros criptossistemas, como o RSA, por exemplo. De fato, as chaves do sistema McEliece possuem um tamanho que varia de 100 kilobytes a vários megabytes, enquanto as chaves do clássico RSA medem de 1024 a 2048 bits.

Tabela 2.1: Comparando RSA e McEliece
Fonte: Yan (2009, p. 349)

	RSA	McEliece
Velocidade de encriptação	$n^2 \approx n^3$	n^2
Velocidade de decriptação	n^3	n^2
Tamanho da chave pública	n	n^2
Tamanho da chave privada	n	n^2

Por exemplo, se temos uma chave pública no sistema RSA de tamanho 256 bits, no sistema McEliece ela será de aproximadamente 65536 bits, o que é um aumento muito inconveniente.

- [55] Não existe um caminho óbvio para usar tal esquema para assinaturas digitais (usadas para garantir a autenticidade de uma mensagem ou documento), como ocorre novamente com o RSA.
- [55] Aplicações práticas de criptografia baseada em códigos ainda não são conhecidas e acredita-se que isso se dá pelo motivo de que tais soluções alternativas não possuem execução urgente.

2.3 Variantes do sistema criptográfio McEliece

Visando minimizar as desvantagens apresentadas pelo sistema McEliece original, muitas variantes foram propostas ao longo dos anos. Vamos agora compreender duas das mais relevantes destas: o criptossistema de Niederreiter [53] e a assinatura CFS [21].

2.3.1 O criptossistema de Niederreiter

O esquema apresentado por Niederreiter, em 1986, é uma variação do sistema de McEliece, chamada de **variante dual**. Tal método ambicionava, basicamente, reduzir o tamanho das chaves públicas para, então, viabilizar a implementação do criptossistema. Para atingir tal objetivo, o autor propôs, inicialmente, substituir os códigos de Goppa por códigos de Reed-Solomon generalizados. No entanto, tal estrutura foi quebrada em 1991, por Sidelnikov e Shestakov, com um ataque estrutural baseado na relação entre códigos de Goppa e códigos de Reed-Solomon (a demonstração desse fato pode ser encontrada em [63]). Surgiu então uma modificação sugerida por Gabidulin [25], que propunha a utilização de códigos menores,com o propósito de transformar o sistema em algo mais prático. Contudo, existe um ataque conhecido que quebrou esse sistema para os parâmetros propostos, destruindo as principais vantagens do sistema McEliece.

Niederreiter sugeriu também, em seu artigo [53], a utilização da matriz de paridade ao invés da matriz geradora como chave pública do sistema de McEliece, o que, posteriormente, foi demonstrado que é equivalente em termos de segurança. Vamos conferir esse resultado no Algoritmo 2.2.

Vale ressaltar que o algoritmo referente ao criptossistema de Niederreiter não possue exatamente a mesma estrutura do que o original. Esse novo esquema realiza as operações de geração de chaves, encriptação e decriptação da mesma maneira, porém, até o presente momento, é a formatação mais eficiente e segura que pode ser apresentada, segundo Barreto [7].

A principal diferença entre os criptossistemas de McElice e de Niederreiter, de acordo com Baldi [6], é que o segundo requer chaves públicas menores e isso se dá pelo fato de que no algoritmo de Niederreiter a versão encriptada de cada mensagem é uma síndrome ao invés da própria palavra-código.

Além disso, é possível comparar também as taxas de encriptação dos dois sistemas. Enquanto o processo descrito pelo algoritmo de McEliece possui a mesma taxa de encriptação do código em que ele se baseia, ou seja, $R_{McEliece} = R = \frac{k}{n}$, o criptossistema de Niederreiter, por sua vez, encripta mensagens de n bits com peso t em síndromes de r bits, fazendo com que sua taxa de

Algoritmo 2.2 O sistema criptográfico Niederreiter para um código arbitrário

[7] Seja Γ um código arbitrário $t-$corretor de erros de comprimento n e dimensão k sobre $\mathbb{F}_2$ que admite um alçapão para decodificação, que vamos admitir como sendo uma matriz de verificação de paridade $\hat{H} \in \mathbb{F}_2^{r\times n}$ e um algoritmo eficiente de decodificação $\mathcal{D} : \mathbb{F}_2^r \to B(0^n, t)$. Considere também uma cifra simétrica semanticamente segura $\varepsilon : B(0^n, t) \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$.

Geração de chaves: Calcule a matriz de paridade H com elementos em $\mathbb{F}_2$ e dimensão $r \times n$ tal que $\hat{H} = \hat{M}H$ para alguma matriz não singular $\hat{M} \in \mathbb{F}_2^{r\times r}$.

Chave privada: $\hat{M}, \hat{H}$.

Chave pública: H, t.

Encriptação de uma mensagem $m \in \{0,1\}^*$: escolha $e \in B(0^n, t)$ e calcule $s = eH^T$. Defina $d = \varepsilon(e, m)$ e por fim, a mensagem encriptada será dada por $c = (s, d)$.

Decriptação de um texto cifrado $(s, d) \in \mathbb{F}_2^r \times \{0,1\}^*$: calcule $\hat{s} = s\hat{M}^T = (eH^T)\hat{M}^T = e(\hat{M}H)^T = e\hat{H}^T$, com isso vemos que $\hat{s}$ será decodificado usando a chave privada $\hat{H}$ para e. Faça $e = \mathcal{D}(\hat{s})$ e então $m = \varepsilon^{-1}(e, d)$. Aceite a mensagem m se e somente se $m \neq \perp$.

encriptação seja:

$$R_{Niederreiter} = \frac{\log_2 \left(\begin{array}{c} n \\ t \end{array} \right)}{r}.$$

2.3.2 Assinatura CFS

Por outro ângulo, pesquisadores defendem que o tamanho das chaves não deveria ser considerado um problema nos dias de hoje, tendo em vista a grande capacidade de memória dos computadores e baixo custo que elas possuem, argumentando assim, que o maior problema do sistema de McEliece é o fato de não poder ser usado em assinaturas digitais. Dessa forma, as pesquisas em torno do sistema criptográfico de Niederreiter não cessaram. Em 2001, Courtois, Finiasz e Sendrier [21], mostraram que é possível construir um esquema de assinatura baseado no criptossistema de Niederreiter.

Um esquema de assinatura digital busca promover um modo de assinar qualquer documento de maneira a identificar unicamente o autor da mensagem e que apresente um algoritmo público eficiente de verificação de assinatura. O esquema baseado no criptossistema de Niederreiter, por sua vez, faz o uso de funções hash. Estas mapeiam dados grandes e de tamanho variável para pequenos dados de tamanho fixo e por esse motivo, essas funções são conhecidas por resumirem dados, tendo como principal aplicação comparar informações grandes ou secretas.

O conhecimento da chave privada permite ao decodificador resolver esse problema para um certo número de palavras arbitrárias c. A ideia do algoritmo CFS é aplicar a função hash repetidamente ao documento, realizando esse processo de maneira aleatória por um contador de comprimento r bits, até que o resultado seja um criptograma decriptável. O assinante usa sua chave privada para determinar o vetor erro correspondente. Juntamente com o valor atual do contador, esse vetor erro servirá de assinatura.

Vamos esquematizar o processo de assinatura digital, explicitando o modo de gerar as chaves,

assinar um documento e verificar a assinatura, descrito no Algoritmo 2.3.

Algoritmo 2.3 O sistema de assinatura digital CFS

[7] Seja Γ um código arbitrário $t-$corretor de erros de comprimento n e dimensão k sobre $\mathbb{F}_2$ equipado com uma matriz de verificação de paridade $\hat{H} \in \mathbb{F}_2^{r\times n}$ e um algoritmo eficiente de decodificação $\mathcal{D} : \mathbb{F}_2^r \to B(0^n, t)$.

Escolha um oráculo aleatório $\mathcal{H} : \{0,1\}^* \times \mathbb{N} \to (\mathbb{F}_2)^r$.

Geração de chaves: Calcule a matriz de paridade H com elementos em $\mathbb{F}_2$ e dimensão $r \times n$ tal que $\hat{H} = \hat{M}H$ para alguma matriz não singular $\hat{M} \in \mathbb{F}_2^{r\times r}$.

Defina a chave privada como sendo $(\hat{M}, \hat{H})$ e a chave pública como sendo (H, t).

Assinando uma mensagem $m \in \{0,1\}^*$**:** encontre $i \in \mathbb{N}$ tal que para $c = \mathcal{H}(m, i)$ e $\hat{c} = c\hat{M}$, seja possível decodificar $\hat{c}$ usando a matriz $\hat{H}$. Faça $e = \mathcal{D}(\hat{c})$ e defina $\sigma = (e, i)$. Observe que $c\hat{M}^T = \hat{c} = e\hat{H}^T = e(\hat{M}H)^T = (eH^T)\hat{M}^T \Rightarrow c = e\hat{H}^T$, ou seja, c é a síndrome do vetor e por $\hat{H}$.

Verificando a assinatura (e, i) : calcule $c = e\hat{H}^T$.

Aceite a assinatura se e somente se $c = \mathcal{H}(m, i)$.

Traduzindo de maneira simples: escolhemos um código linear $\mathcal{C}$ e o CFS usa uma função *hash* pública para resumir o documento a ser assinado no vetor $\mathcal{H}(m)$. Decodificando esse resumo com o algoritmo de correção de erros do respectivo código, obtemos um vetor c, que corresponde a assinatura da mensagem m. Para a verificação da assinatura, basta encriptar c, recebido junto da mensagem m, e verificar se corresponde ao cálculo do *hash* da mensagem m.

É importante ressaltar que encontrar uma função que transforme rapidamente dados que foram resumidos usando hash para uma síndrome corrigível é um trabalho muito difícil. No trabalho original [21] esse problema é encarado de duas maneiras: incluindo um contador à mensagem ou decodificando completamente a mensagem. Segundo Baldi [6], esses dois processos requerem uma escolha particular de códigos para que essa implementação se torne razoavelmente rápida.

Por isso, é possível afirmar, usando ainda as palavras de Baldi [6], que não existe ainda nenhum sistema eficiente de assinatura digital baseado nos algoritmos de McEliece e Niederreiter.

Capítulo 3

Reticulados

Os primeiros registros do uso da estrutura de reticulados foram encontrados no século $XVIII$ e usados até meados do século XX por grandes matemáticos como Gauss e Lagrange para resolver problemas em teoria dos números, como o teorema da reciprocidade quadrática e o teorema dos quatro quadrados.

Aspectos computacionais envolvendo reticulados, por sua vez, começaram a ser estudados somente no início dos anos 80, onde eles eram utilizados com sucesso para quebrar alguns esquemas criptográficos. No entanto, somente a partir do final de 1990 tais estruturas foram empregados para embasar um sistema criptográfico e atualmente, existem muitas pesquisas sendo feitas acerca de suas vantagens em tais aplicações.

Neste capítulo, vamos definir matematicamente os reticulados, bem como explorar suas principais características, visando apresentar o modo de como eles vem sendo utilizados em criptografia. Para isso, tomamos como referência as obras [56], [36], [69] e [17].

3.1 Conceitos básicos e propriedades

Definição 3.1.1. *Um conjunto $\mathcal{L} \in \mathbb{R}^n$ é um* ***reticulado*** *se existem m vetores linearmente independentes $u_1, u_2, \ldots, u_m$ em $\mathbb{R}^n$, com $m \leq n$ tal que*

$$x \in \mathcal{L} \Leftrightarrow x = a_1u_1 + a_2u_2 + \cdots + a_mu_m,$$

com $a_i \in \mathbb{Z}$ para todo i. Além disso, o conjunto $\beta = \{u_1, u_2, \ldots, u_m\}$ é chamado de ***base do reticulado*** *$\mathcal{L}$.*

Podemos ainda entender essa definição de outra maneira, ou seja, um reticulado é um subgrupo aditivo discreto de $\mathbb{R}^n$. De fato, um reticulado $\mathcal{L}$ é um subgrupo de $\mathbb{R}^n$, segundo a operação adição usual de $\mathbb{R}^n$ componente a componente. Vale lembrar que $\mathcal{L}$ é um subgrupo se contém o elemento identidade $0 \in \mathcal{L}$ e se para todo $x, y \in \mathcal{L}$ temos que $-x \in \mathcal{L}$ e $x + y \in \mathcal{L}$.

Além disso, um reticulado é discreto: isso significa que para todo $x \in \mathcal{L}$ existe uma vizinhança de x na qual x é o único ponto, ou seja, para todo $x \in \mathcal{L}$ existe $\epsilon > 0$ tal que $B(x; \epsilon) \cap \mathcal{L} = \{x\}$, onde $B(x; \epsilon)$ denota uma bola aberta de centro em x e raio ϵ. Note que como $\mathcal{L}$ é um grupo, existe algum $\epsilon > 0$ que satisfaz tal condição para a origem $0 \in \mathcal{L}$, logo, também satisfaz para todo $x \in \mathcal{L}$.

Reciprocamente, pode-se mostrar que todo subgrupo aditivo discreto de $\mathbb{R}^n$ satisfaz a definição acima [67].

Podemos interpretar a base β do reticulado como sendo uma matriz $B \in \mathbb{R}^{n\times m}$ não singular cujas colunas ordenadas são $u_1, \ldots, u_m$. Com isso, $\mathcal{L} = B \cdot \mathbb{Z}^m = \{Bx : x \in \mathbb{Z}^m\}$. Seja $x = k_1u_1 + k_2u_2 + \cdots + k_mu_m \in \mathcal{L}$, então, escrevendo os vetores na forma de colunas, com as coordenadas na base canônica, obtemos:

$$x = k_1 \begin{pmatrix} u_{11} \\ \vdots \\ u_{1n} \end{pmatrix} + \cdots + k_m \begin{pmatrix} u_{m1} \\ \vdots \\ u_{mn} \end{pmatrix} = \begin{pmatrix} u_{11} & \ldots & u_{m1} \\ \vdots & & \vdots \\ u_{1n} & \ldots & u_{mn} \end{pmatrix} \begin{pmatrix} k_1 \\ \vdots \\ k_m \end{pmatrix}$$

Isso nos mostra que $\mathcal{L}$ é a imagem de $\mathbb{Z}^m$ pela matriz $B = (u_{ij})$, ou seja, todo $x \in \mathcal{L}$ é da forma Bv^t, para algum $v = (k_1, \ldots, k_m)$ em $\mathbb{Z}^m$.

Definição 3.1.2. *A matriz B, com as características explicitadas acima, é chamada de* ***matriz geradora*** *do reticulado $\mathcal{L}$.*

Observação 3.1.3. *Observamos que uma base não é necessariamente um subconjunto de um conjunto gerador. Por exemplo, em duas dimensões os três pontos $(1,4), (4,1)$ e $(4,4)$ geram o reticulado $\mathbb{Z}^2$, mas nenhum par desses pontos gera tal reticulado. Isso acontece porque para gerar um reticulado precisamos de combinações lineares inteiras de vetores e não mais reais, o que ocasiona em uma pequena modificação na nossa definição usual de vetores linearmente independentes. Naturalmente uma base de $\mathbb{Z}^2$ é dada por $\beta = \{(1,0), (0,1)\}$.*

Por exemplo, 7 e 8, são vetores linearmente dependentes quando os consideramos sobre o corpo dos reais $\mathbb{R}$, mas o mesmo não ocorre sob $\mathbb{Z}$, que, por não ser corpo, não goza da operação de divisão. Com isso, podemos notar ainda que 7 e 8 geram todo o reticulado $\mathbb{Z}$, mas cada um desses pontos não consegue gerá-lo. Observamos também que $\beta = \{1\}$ é uma base de $\mathbb{Z}$.

Contudo, estaremos considerando como base um conjunto gerador com um número mínimo de vetores. Um reticulado com posto ou dimensão k como definido a seguir sempre possui uma base com k elementos.

Definição 3.1.4. *O* ***posto*** *k de um reticulado $\mathcal{L} \in \mathbb{R}^n$ é a dimensão do seu espaço gerado, ou seja, $k = \dim(span(\mathcal{L}))$. Quando $k = n$ dizemos que tal reticulado possui* ***posto completo****.*

Na maioria dos resultados explicitados daqui em diante estaremos considerando reticulados com posto completo. Se existirem casos particulares onde o reticulado não possua posto completo, deixaremos esse fato claro para o leitor.

Exemplo 3.1.5. *Podemos notar que no reticulado $\mathbb{Z}^2 \subset \mathbb{R}^2$, que é formado pelas coordenadas inteiras do plano, a base $\beta = \{(0,1), (1,0)\}$ não é única. De fato $\beta' = \{(1,3), (0,1)\}$ também é uma base de $\mathbb{Z}^2$, já que o reticulado gerado pela base β' está contido em $\mathbb{Z}^2$ e, por outro lado*

$$(m,n) = m(1,3) + (n-3m)(0,1),$$

para $(m,n) \in \mathbb{Z}^2$.

A caracterização de bases que geram o mesmo reticulado em $\mathbb{R}^n$ é dada a seguir.

Proposição 3.1.6. *[42] Dado um reticulado $\mathcal{L}$ com posto completo gerado por uma base β. Uma base α de $\mathbb{R}^n$ também é base deste reticulado, se e somente se $\alpha \subset \mathcal{L}$ e a matriz de mudança de base M possui entradas inteiras e determinante ± 1.*

Demonstração. Sejam $\alpha = \{u_1, \ldots, u_n\}$ e $\beta = \{v_1, \ldots, v_n\}$ bases de $\mathcal{L}$, $[T]_\alpha^\beta$ a matriz mudança de base de β para α e $[T]_\beta^\alpha = ([T]_\alpha^\beta)^{-1}$ a matriz de mudança de base de α para β. Temos então que, para $w \in \mathcal{L}$ podemos escrever matricialmente:

$$[w]_\beta = [T]_\beta^\alpha [w]_\alpha.$$

Ao considerarmos $[w]_\alpha = e_i$, onde e_i é o vetor canônico de $\mathbb{R}^n$, concluímos que $[T]_\beta^\alpha$ deve ser formada somente por elementos inteiros e como essa matriz é inversível, seu determinante deve ser ± 1. A mesma afirmação vale para $[T]_\alpha^\beta$.

Reciprocamente, sabemos que como a matriz mudança de base possui apenas entradas inteiras e determinante ± 1, então todo elemento da base β pode ser escrito como combinação linear inteira dos elementos da base α. Portanto, todo elemento do reticulado gerado por β pertence ao reticulado gerado por α e vice-versa. □

Definição 3.1.7. *Sejam $\mathcal{L}, \mathcal{L}' \subset \mathbb{R}^n$ dois reticulados de mesmo posto. Se $\mathcal{L}' \subseteq \mathcal{L}$ dizemos que $\mathcal{L}'$ é um **subreticulado** de $\mathcal{L}$.*

Denotamos por $B(u, \rho) \subset \mathbb{R}^n$ a **bola** aberta de centro em $u \in \mathbb{R}^n$ e raio ρ, com a distância euclidiana.

Definição 3.1.8. *O **raio de empacotamento** de um reticulado $\mathcal{L}$ é o maior raio ρ tal que $B(u, \rho) \cap B(v, \rho) = \emptyset$, para todo $u, v \in \mathcal{L}$ tais que $u \neq v$.*

Exemplo 3.1.9. *Se tomarmos bolas de raio maior do que $\frac{1}{2}$ centradas em cada ponto v do reticulado $\mathbb{Z}^2$ haverá sobreposições. Portanto, $\frac{1}{2}$ é o maior raio possível para um empacotamento de bolas com centros em pontos de $\mathbb{Z}^2$.*

Outra definição importante nesse contexto é a de norma mínima, que está intimamente relacionada com o problema matemático concernente ao sistema criptográfico baseado em reticulados e como veremos no decorrer deste capítulo.

Definição 3.1.10. *A **norma mínima** de um reticulado $\mathcal{L}$ é a menor norma dentre os elementos não nulos de $\mathcal{L}$, ou seja:*

$$\eta = min\{||x|| : x \in \mathcal{L}, x \neq 0\}.$$

O resultado a seguir relaciona o raio de empacotamento e distância mínima. Observamos que a norma mínima de um vetor não nulo do reticulado possui o mesmo valor que a distância mínima entre dois pontos distintos do reticulado.

Proposição 3.1.11. *[42] O raio de empacotamento ρ de um reticulado $\mathcal{L}$ é igual a metade da distância mínima d entre pontos do reticulado.*

Demonstração. Basta demonstrarmos que se escolhermos $r = \frac{d}{2}$ vale que $B(u,r) \cap B(v,r) = \emptyset$, para todo $u, v \in \mathcal{L}$ e depois, mostramos que $r = \frac{d}{2}$ é o maior raio possível.

Consideramos $r = \frac{d}{2}$. Se tomarmos $x \in \mathbb{R}^n$ qualquer tal que $x \in B(u,r) \cap B(v,r)$. Daí,

$$d(x,u) < r = \frac{d}{2}$$

e

$$d(x,v) < r = \frac{d}{2}.$$

Isso nos mostra que $d(x,u) + d(x,v) < \frac{d}{2} + \frac{d}{2} = d$. Mas, a desigualdade triangular nos diz que $d(x,u) + d(x,v) \geq d(u,v) \geq d$, e temos uma contradição. Logo, $B(u,r) \cap B(v,r) = \emptyset$.

Resta provar que $r = \frac{d}{2}$ é o maior raio possível. Suponha $r > \frac{d}{2}$. Sejam $u, v \in \mathcal{L}$ tais que $d(u,v) = d$. Então, para $h = \dfrac{u+v}{2} \in \mathbb{R}^n$, temos que $d(h,u) = d(h,v) = \frac{d}{2} < r$. Portanto, existiria $r \in \mathbb{R}^n$ tal que $h \in B(u,r) \cap B(v,r)$. Assim, $r = \frac{d}{2} = \rho$ é o maior raio possível. □

Definição 3.1.12. *Se $x \in \mathcal{L}$, a **região de Voronoi** de x é o conjunto*

$$R(x) = \{x \in \mathbb{R}^n : d(x,v) \leq d(u,v), u, v \in \mathcal{L}\}.$$

Em outras palavras, a região de Voronoi de um ponto $x \in \mathcal{L}$ é o conjunto dos pontos de $\mathbb{R}^n$ que estão mais próximos de x, segundo uma métrica d, do que de qualquer outro ponto de $\mathcal{L}$.

Exemplo 3.1.13. *A região de Voronoi de um ponto v de $\mathbb{Z}^2$ é o conjunto $R(v)$ dos pontos de $\mathbb{R}^2$ que estão mais próximos de v do que de qualquer outro ponto de $\mathbb{Z}^2$, que neste caso, está representada por um quadrado unitário centrado no ponto v.*

Proposição 3.1.14. *[42] Para todo $v \in \mathcal{L}$, onde $\mathcal{L}$ é um reticulado, vale que*

$$R(v) = v + R(0) = \{v + x \in \mathbb{R}^n : x \in R(0)\}.$$

Definição 3.1.15. *Um conjunto $\mathcal{F} \subset \mathbb{R}^n$ é uma **região fundamental** de um reticulado $\mathcal{L}$ se suas translações $x + \mathcal{F} = \{x + y : y \in \mathcal{F}\}$ consideradas sobre todo $x \in \mathcal{L}$ formam uma partição de $\mathbb{R}^n$.*

Em outras palavras, uma região fundamental é um conjunto fechado $\mathcal{F}$ do $\mathbb{R}^n$ que ladrilha o $\mathbb{R}^n$ via o reticulado $\mathcal{L}$, isto é, tomando todas as translações $x + \mathcal{F}$, com $x \in \mathcal{L}$ conseguimos cobrir todo o $\mathbb{R}^n$ de modo que dois ladrilhos não possuam interseção ou, se ela ocorrer, será apenas nos bordos.

Exemplo 3.1.16. *A região de Voronoi $R(0)$ é um exemplo de região fundamental de $\mathcal{L}$, isso significa que podemos ladrilhar o $\mathbb{R}^n$ considerando translações de $R(0)$. Notamos que esse fato segue da Proposição 3.1.14.*

Exemplo 3.1.17. *Os intervalos $[0,1)$ e $[-\frac{1}{2}, \frac{1}{2})$ formam regiões fundamentais do reticulado inteiro $\mathbb{Z}$. De fato, qualquer $x \in \mathbb{R}$ é a única translação $\lfloor x \rfloor + [0,1)$ ou $\lfloor x \rceil + [-\frac{1}{2}, \frac{1}{2})$, respectivamente, onde $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ denota a aproximação ao menor inteiro.*

Exemplo 3.1.18. *O cubos* $[0,1)^n$ *e* $[-\frac{1}{2},\frac{1}{2})^n$ *são regiões fundametais do* $\mathbb{Z}^n$.

Proposição 3.1.19. *O volume de qualquer região fundamental de* $\mathcal{L}$ *é o mesmo.*

A demonstração para esse fato pode ser encontrada em [9].

Definição 3.1.20. *Seja um reticulado* $n-$*dimensional* $\mathcal{L} \subset \mathbb{R}^n$. *Definimos a* ***densidade de empacotamento esférico*** *do reticulado* $\mathcal{L}$ *como sendo*

$$\Delta = \frac{volume\ de\ uma\ esfera\ em\ \mathbb{R}^n}{volume\ de\ uma\ região\ fundamental} = \frac{volS(0,\rho)}{(\det \mathcal{L})^{\frac{1}{2}}}.$$

Podemos calcular a densidade de um empacotamento esférico de um reticulado $\mathcal{L}$ tendo em mãos uma base de $\mathcal{L}$ e sua distância mínima. Vamos ilustrar tal definição com alguns exemplos clássicos.

Exemplo 3.1.21. *Para* $n = 2$ *temos que o melhor empacotamento é o dado pelo reticulado hexagonal, cada círculo tocando seis outros círculos. Os centros dessas esferas são os pontos do reticulado* A_2 *(que será ilustrado a seguir). A densidade* ρ *desse empacotamento, que é a fração do plano ocupado pelas esferas, é de* $\frac{\pi}{\sqrt{12}} \simeq 0.9069...$

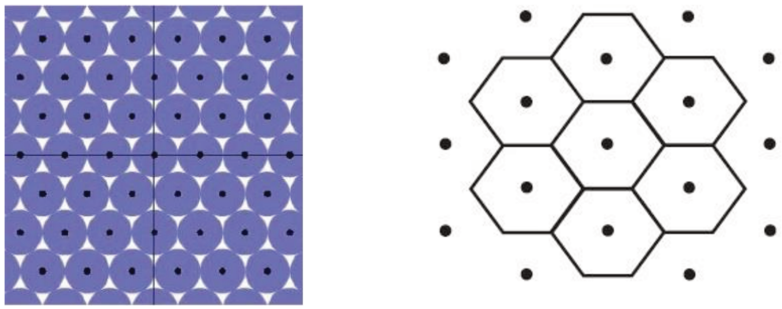


Figura 3.1: Reticulado A_2 e sua região de Voronoi

Fontes: [20], p.4 e [69], p.18.

Exemplo 3.1.22. *A melhor densidade de empacotamento para um reticulado em* $\mathbb{R}^3$ *é* $\frac{\pi}{\sqrt{18}} \simeq 0.74$ *que é atingida no reticulado que pode ser visualizado como uma "pilha de laranjas". Este reticulado é o* $A_3 = D_3$, *que possui como base os vetores que são centro das faces de um cubo (face-centered cubic - FCC)[17] e está ilustrado na Figura 3.2. Em 1998, Tomas Hales anunciou uma prova que mesmo para o empacotamento de esferas cujos centros não estão em reticulados, esta seria a melhor densidade possível em* $\mathbb{R}^3$.

Outra noção bastante importante para o nosso estudo é:

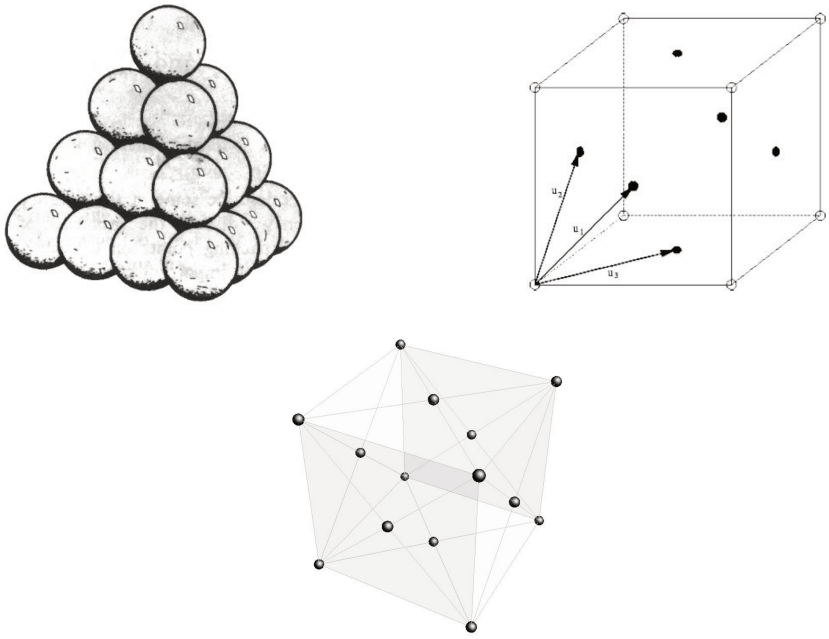


Figura 3.2: Face centered cubic (FCC)

Fontes: [17], p.2, [20], p.5 e *produção do próprio autor.*

Definição 3.1.23. *Dada uma base* $\beta = \{u_1, u_2, \ldots, u_n\}$, *definimos o* ***politopo fundamental*** *gerado por essa base como sendo o sólido*

$$P = P(\beta) = \beta \cdot [0,1)^n = \{\sum_{i=1}^{n} a_i u_i : 0 \leq a_i < 1\}.$$

Lema 3.1.24. *[56] Seja* $\mathcal{F}$ *uma região fundamental de* $\mathbb{Z}^n$ *e* β *uma base de um reticulado de posto máximo em* $\mathbb{R}^n$. *Considere matriz* $B \in \mathbb{R}^{n\times n}$ *como sendo a matriz geradora de* $\mathcal{L}$. *Então* $P = B \cdot \mathcal{F} = \{Bx : x \in \mathcal{F}\}$ *é uma região fundamental de* $\mathcal{L}$. *Em particular,* P *é uma região fundamental de* $\mathcal{L}$.

Demonstração. Queremos mostrar que cada $x \in \mathbb{R}^n$ está em exatamente uma translação $v + B\mathcal{F}$, onde $v = Bz \in \mathcal{L}$ para algum $z \in \mathbb{Z}^n$.

Como B é uma matriz não singular, temos que $x \in Bz + B\mathcal{F}$ se e somente se $B^{-1}x \in z + \mathcal{F}$. Já que $B^{-1}x \in \mathbb{R}^n$ e $\mathcal{F}$ é uma região fundamental de $\mathbb{Z}^n$ por hipótese, existe exatamente um $z \in \mathbb{Z}^n$ para o qual a última inclusão é válida e isso prova o que queríamos. □

Definição 3.1.25. *O* ***volume*** *(ou determinante) de um reticulado* $\mathcal{L}$, *denotado por* $\det(\mathcal{L}) = |\det B|$, *onde* B *é uma matriz geradora de* $\mathcal{L}$. *Observamos que* $\det(\mathcal{L}) = vol(\mathcal{F})$, *onde* $\mathcal{F}$ *é a região fundamental de* $\mathcal{L}$.

Definição 3.1.26. *Se* B *é uma matriz geradora de* $\mathcal{L}$, *a* ***matriz de Gram*** *associada é* $G = B^T B$.

Como um reticulado pode admitir várias bases distintas que o geram, ele também admite várias matrizes de Gram distintas, mas, o determinante de cada uma delas é o mesmo e só depende do reticulado.

De fato, considere as bases $\beta = \{u_1, u_2, \ldots, u_n\}$ e $\beta' = \{v_1, v_2, \ldots, v_n\}$ de $\mathcal{L}$ e sejam A e B as matrizes geradoras associadas. Como β é base de $\mathcal{L}$, podemos escrever:

$$v_j = a_{1j}u_1 + a_{2j}v_2 + \cdots + a_{nj}v_n,$$

para $j = 1, 2, \ldots, n$, onde cada $a_{ij} \in \mathbb{Z}$. A transformação linear $T : \mathbb{R}^n \to \mathbb{R}^n$, que leva u_j em v_j faz a mudança de base e possui matriz M com determinante ± 1. Logo:

$$B = MA$$

e $\det(B^T B)$ é igual a

$$\det(A^T M^T M A) = \det(A^T)\det(M^T)\det(M)\det(A) = det(A^T A).$$

Vamos citar agora alguns exemplos de reticulados importante e as suas principais características.

Exemplo 3.1.27. *O* ***reticulado*** $\mathbb{Z}^n$ *é definido por*

$$\mathbb{Z}^n = \{(x_1, x_2, \ldots, x_n) : x_i \in \mathbb{Z} \ \ \forall 1 \leq i \leq n\}.$$

Observe que a matriz geradora de $\mathbb{Z}^n$ *é a matriz identidade de ordem* n*, logo, o determinante desse reticulado é igual a* 1.

Além disso, a norma mínima desse reticulado é 1*, já que seus vetores de norma mínima são todos da forma* $v = (0, 0, \ldots, \pm 1, \ldots, 0)$*. E, o raio de empacotamento é* $\rho = \frac{1}{2}$.

Exemplo 3.1.28. *O* ***reticulado*** A_n *é definido por*

$$A_n = \{(x_1, \ldots, x_{n+1}) \in \mathbb{Z}^{n+1} : x_1 + \cdots + x_{n+1} = 0\}.$$

É um reticulado que apresenta as melhores densidades euclidianas possíveis nas dimensões 2 *e* 3*. O reticulado* A_2 *é o equivalente ao reticulado hexagonal na métrica euclidiana.*

A norma mínima de A_n*, por sua vez, é* $\sqrt{2}$*, já que todos os seus vetores de norma mínima são da forma* $v = (1, -1, \ldots, 0, 0, \ldots, 0)$*. O raio de empacotamento é* $\rho = \dfrac{1}{\sqrt{2}}$.

Exemplo 3.1.29. *O* ***reticulado*** D_n *é definido por:*

$$D_n = \{(x_1, \ldots, x_n) \in \mathbb{Z}^n : x_1 + \cdots + x_n \text{ é par}\} \subset \mathbb{R}^n$$

é o reticulado que possui a melhor densidade euclidiana possível nas dimensões 3 *e* 4.

Exemplo 3.1.30. *O* ***reticulado*** E_8*, definido como:*

$$E_8 = \{(x_1, \ldots, x_8) : \text{todo } x_i \in \mathbb{Z} \text{ ou todo } x_i \in \mathbb{Z} + 1/2, \ \text{e} \ \ \sum x_i \text{ é par}\}$$

é o reticulado com maior densidade euclidiana na dimensão 8.

3.2 Construção A: relação entre reticulados e códigos

É possível estabelecer uma relação entre códigos e reticulados e a construção A nos mostra como isso pode ser feito. Na verdade, ela nos apresenta um modo de se obter reticulados a partir de códigos lineares $\mathcal{C} \subset \mathbb{Z}_q^n$. Vamos utilizar como referência principal os textos de [17] e [69].

Alguns reticulados conhecidos, como o E_8 e o $D_n, n \geq 3$, já citados na seção anterior, podem ser obtidos por meio da Construção A aplicada a códigos $q-$ários e tais processos serão ilustrados no decorrer desta seção.

Proposição 3.2.1. *[17] Considere a aplicação sobrejetora:*

$$\phi : \mathbb{Z}^n \to \mathbb{Z}_q^n$$

$$(x_1, \ldots, x_n) \mapsto (\overline{x_1}, \ldots, \overline{x_n})$$

onde $\overline{x_i}$ é obtido pela redução módulo q para todo $1 \leq i \leq n$. Temos que $\mathcal{C} \subseteq \mathbb{Z}_q^n$ é um código linear $q-$ário se, e somente se, $\phi^{-1}(\mathcal{C}) \subseteq \mathbb{Z}^n$ é um reticulado em $\mathbb{R}^n$. Além disso, $q\mathbb{Z}^n \subseteq \phi^{-1}(\mathcal{C})$.

Demonstração. Seja $\mathcal{C}$ um código linear $q-$ário. Como $\phi^{-1}(\mathcal{C}) \subseteq \mathbb{Z}_q^n$ é um conjunto discreto, basta provar que $\phi^{-1}(\mathcal{C})$ é um subgrupo aditivo e então, as condições para que $\phi^{-1}(\mathcal{C})$ seja um reticulado estarão satisfeitas. De fato.

- $0 \in \phi^{-1}(\mathcal{C})$, pois $\phi(0) = \overline{0} \in \mathcal{C}$;
- Se $a, b \in \phi^{-1}(\mathcal{C})$, então $\phi(a) = \overline{a}$ e $\phi(b) = \overline{b} \in \mathcal{C}$. Logo, $\phi(a-b) = \overline{a-b} = \overline{a} - \overline{b} \in \mathcal{C}$. Assim, $a - b \in \phi^{-1}(\mathcal{C})$, e, com isso, mostramos que $\phi^{-1}(\mathcal{C})$ é um subgrupo aditivo de $\mathbb{R}^n$.

Reciprocamente, seja $\mathcal{C} \subseteq \mathbb{Z}_q^n$ tal que $\phi^{-1}(\mathcal{C})$ é um reticulado. Temos então que $\mathcal{C} = \phi(\phi^{-1}(\mathcal{C}))$ é um subgrupo de $\mathbb{Z}_q^n$, pois é a imagem de um subgrupo de $\mathbb{Z}^n$ via um homomorfismo de grupos. Logo, $\mathcal{C}$ é um código linear $q-$ário. □

Definição 3.2.2. *A aplicação ϕ que relaciona um código linear $q-$ário $\mathcal{C}$ a um reticulado $\phi^{-1}(\mathcal{C})$ é chamada de* ***Construção A****. Além disso, o reticulado $\mathcal{L}_\mathcal{C} = \phi^{-1}(\mathcal{C})$, ou seja, $\mathcal{L}_\mathcal{C} = \{x \in \mathbb{Z}^n : x \ (mod\ q) \in \mathcal{C}\}$ é chamado de* ***reticulado $q-$ário****. De maneira equivalente, $\mathcal{L}_\mathcal{C}$ pode ser escrito como $\mathcal{L}_\mathcal{C} = q\mathbb{Z}^n + \mathcal{C}$.*

Note que em [55], um reticulado $q-$ário é definido como sendo um reticulado $\mathcal{L}$ tal que $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$, para algum $q \in \mathbb{N}$. Contudo, tal definição é equivalente à dada anteriormente, de acordo com [40]. De fato, se $\mathcal{L}$ é um reticulado $q-$ário segundo a definição 3.2.2, temos que $\phi^{-1}(\{\overline{0}\}) = q\mathbb{Z}^n$ e segue que $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$. Seja $\mathcal{L}$ um reticulado que satisfaz $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$, para algum $q \in \mathbb{N}$. Seja $\mathcal{C} = \phi(\mathcal{L})$. Se $x \in \phi^{-1}(\mathcal{C})$, então $\phi(x) \in \mathcal{C}$, o que nos diz que existe $y \in \mathcal{L}$ tal que $\phi(x) = \phi(y)$ e assim, $x - y \in \mathrm{Ker}(\phi) = q\mathbb{Z}^n \subseteq \mathcal{L}$. Como $y \in \mathcal{L}$ e $x - y \in \mathcal{L}$ segue que $x \in \mathcal{L}$ e, portanto, $\phi(\mathcal{C}) \subseteq \mathcal{L}$. Como $\phi^{-1}(\mathcal{C}) = \phi^{-1}\phi(\mathcal{L}) \subset \mathcal{L}$, vem que $\mathcal{L} = \phi^{-1}(\mathcal{C})$. Com isso, concluímos que as duas definições são de fato equivalentes e usaremos a que for mais conveniente durante o desenvolvimento deste texto.

Antes de explorarmos exemplos de reticulados obtidos via Construção A, vamos apresentar mais alguns resultados importantes.

Proposição 3.2.3. *[69] Propriedades de um reticulado $q-$ário:*

(*i*) *A distância mínima de quaisquer dois pontos em $\mathcal{L}(\mathcal{C})$ é no máximo q. Para $q = 2$ temos que $d_{min}(\mathcal{L}(\mathcal{C})) = min\{\sqrt{2}, d\}$, onde d é a distância de Hamming mínima do código gerado $\mathcal{C}$.*

(*ii*) *O reticulado $\mathcal{L}(\mathcal{C})$ é gerado por uma "matriz geradora estendida"de dimensão $n \times (n + k)$, dada por*

$$G_{\mathcal{L}(\mathcal{C})} = \left(\begin{array}{c|c} G & qI_n \end{array} \right), \tag{3.2.1}$$

onde G é a matriz geradora do código $\mathcal{C}$ e I_n é a matriz identidade $n \times n$. (Observamos que nesse item, a matriz geradora do código não está necessariamente em sua forma padrão e $G_{\mathcal{L}(\mathcal{C})}$ não contém necessariamente uma base do reticulado em suas colunas).

(*iii*) *A matriz estendida conforme em 3.2.1 pode ser reduzida a uma matriz na forma padrão $n \times n$ para $\mathcal{L}(\mathcal{C})$, que é obtida por meio da matriz geradora padrão de $\mathcal{C}$. Tal representação sempre existe em um alfabeto primo (q primo). De modo mais específico, admita uma matriz geradora na forma $G = [I_k|B^T]^T$, onde I_k é a matriz identidade $k \times k$ e B é uma matriz $(n - k) \times k$. Ou seja, cada palavra-código $c = Gw$ consiste em um vetor de informação w concatenado com as informações da matriz de paridade H. Logo:*

$$G_{\mathcal{L}(\mathcal{C})} = \left(\begin{array}{c|c} I_k & 0 \\ B & qI_{n-k} \end{array} \right), \tag{3.2.2}$$

é uma matriz geradora na forma padrão para $\mathcal{L}(\mathcal{C})$.

Exemplo 3.2.4. *Vamos construir a partir da teoria desenvolvida o reticulado $D_n,\ n \geq 3$.*

Seja $\mathcal{C}_n$ um código definido como:

$$\mathcal{C}_n = \{(x_1, x_2, \ldots, x_n) \in \mathbb{Z}_2^n : \sum_{i=1}^{n} x_i = 0\},$$

com parâmetros $[n, n-1, 2]$. Então, o reticulado D_n é igual a $\mathcal{L}(\mathcal{C}_n) = \phi^{-1}(\mathcal{C}_n)$. Vale lembrar que $D_n = \{(x_1, \ldots, x_n) \in \mathbb{Z}^n : x_1 + \cdots + x_n$ é par$\}$.

Vamos mostrar que $D_n = \phi^{-1}(\mathcal{C}_n)$, onde

$$\phi : \mathbb{Z}^n \to \mathbb{Z}_2^n$$

$$(x_1, \ldots, x_n) \mapsto (\overline{x_1}, \ldots, \overline{x_n}).$$

(*i*) *$D_n \subset \phi^{-1}(\mathcal{C}_n)$*

Tome $(x_1, \ldots, x_n) \in D_n$. Então, $\phi(x_1, \ldots, x_n) = (\overline{x_1}, \ldots, \overline{x_n})$. Observe que $(\overline{x_1}, \ldots, \overline{x_n}) \in \mathcal{C}_n$, pois

$$\overline{x_1} + \cdots + \overline{x_n} = \overline{x_1 + \cdots + x_n} \equiv 0 (mod\ 2)$$

Logo, $\phi(x_1, \ldots, x_n) = (\overline{x_1}, \ldots, \overline{x_n}) \in \mathcal{C}_n$. Portanto, $(x_1, \ldots, x_n) \in \phi^{-1}(\mathcal{C}_n)$ e com isso provamos que $D_n \subset \phi^{-1}(\mathcal{C}_n)$.

(*ii*) $\phi^{-1}(\mathcal{C}_n) \subset D_n$

Considere $(x_1, \ldots, x_n) \in \phi^{-1}(\mathcal{C}_n)$, *isso significa que* $\phi(x_1, \ldots, x_n) \in \mathcal{C}_n$, *e, por definição,* $(\overline{x_1}, \ldots, \overline{x_n}) \in \mathcal{C}_n$. *Então*

$$\overline{x_1} + \cdots + \overline{x_n} \equiv 0 (mod\ 2) \Rightarrow \overline{x_1 + \cdots + x_n} \equiv 0 (mod\ 2).$$

Daí, vem que $x_1 + \cdots + x_n$ *é par e, portanto,* $x_1 + \cdots + x_n \in D_n$. *Com isso, mostramos que* $\phi^{-1}(\mathcal{C}_n) \subset D_n$.

Exemplo 3.2.5. *[42] Recorde que:*

$$E_8 = \{(x_1, \ldots, x_8) : todo\ x_i \in \mathbb{Z}\ ou\ todo\ x_i \in \mathbb{Z} + 1/2,\ e\ \ \sum x_i\ é\ par\}.$$

Considere o $[8, 4, 4]-$ *código de Hamming* ($q = 2,\ n = 8,\ k = 4$ *e* $d_{min} = 4$). *Ele é construído sob a estrutura do* $[7, 4, 3]-$ *código de Hamming acrescentando às palavras desse código um bit de modo que a soma de todas as entradas da palavra seja sempre par. Considere a matriz de paridade do* $[7, 4, 3]-$*código de Hamming:*

$$\mathcal{H}_3 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

A partir dela, obtemos a matriz de paridade do $[8, 4, 4]-$*código de Hamming* $\tilde{H}_3$ *como sendo:*

$$\mathcal{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Então, uma base usual para esse código é dada pelas colunas da matriz geradora:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix},$$

que provém da matriz de paridade.

Considere $(x_1, x_2, \ldots, x_8) \in \tilde{H}_3$. *Observamos que o código de Hamming estendido* $\tilde{H}_3$ *possui 14 palavras de peso igual a 4, onde 7 destas palavras possuem* $x_8 = 1$ *e as outras 7 possuem* $x_8 = 0$. *Para a construção do* E_8, *vamos considerar o seguinte resultado auxiliar:*

Lema 3.2.6. *[42] Dados $u, v \in \mathbb{Z}_2^n$, sendo $\omega(u)$ o peso de u e $< u, v >$ o produto interno de u com v em $\mathbb{R}^n$, temos*

$$\omega(u+v) = \omega(u) + \omega(v) - 2 < u, v > .$$

Assim, como os pesos não nulos que $\tilde{H}_3$ atinge são 4 e 8, se u e v tem peso 4 e $u \neq v$, então $\omega(u+v) = 8 - 2 < u, v >$. Dessa forma, temos duas possibilidades: ou $\omega(u+v) = 8$ o que implica em $< u, v >= 0$ ou $\omega(u, v) = 4$ e $< u, v >= 2$. Considerando apenas os vetores que contém a última coordenada não nula, obtemos sempre $< u, v >= 2$.

Admitindo primeiramente os vetores $u_1, u_2, \ldots, u_7$ tais que a última coordenada é 1, buscamos encontrar uma base para o reticulado E_8 a partir de tais vetores. Tome $f_i = \frac{1}{\sqrt{2}} u_i$ e, conforme visto anteriormente:

$$< f_i, f_j >= 1, \text{se } i \neq j$$
$$< f_i, f_i >= 2, \text{para cada } i.$$

Consideramos agora os seguintes vetores:

$$\begin{aligned} e_1 &= f_1, \\ e_2 &= f_2 - f_1, \\ &\vdots \\ e_7 &= f_7 - f_6. \end{aligned}$$

Usando as relações anteriores entre os $f_i's$, vem que:

$$\begin{aligned} < e_i, e_{i+1} >= -1, \text{para } i = 1, \ldots, 6, \\ < e_i, e_{i-1} >= -1, \text{para } i = 2, \ldots, 7, \\ < e_i, e_j >= 0, \text{se } |i-j| \geq 2, \end{aligned}$$

e todos os produtos internos de pares de vetores em $\{e_1, e_2, \ldots, e_7\}$ coincidem com os vetores que formam uma base para E_8 definidos em [17]. Nos resta apenas definir mais um vetor e_8 tal que valha $< e_8, e_8 >= 2$, $< e_8, e_5 >= -1$ e $< e_8, e_i >= 0$, para $i \neq 5, 8$. Resolvendo esse sistema de equações, obtemos o vetor

$$e_8 = \frac{1}{\sqrt{2}}(-1, -1, 0, 0, 1, 0, -1, 0).$$

Notamos que esse vetor pertence à $\mathcal{L}_{\tilde{H}_3}$ e concluímos que $E_8 = \mathcal{L}_{\tilde{H}_3}$.

Exemplo 3.2.7. *Considere o código de Goppa ilustrado em 1.7.6. Lembre que tínhamos um código de Goppa $\Gamma(L, g(x))$ definido por*

$$g(x) = x^2 + x + 1$$
$$L = \{\kappa_i : 0 \leq i \leq 7\},$$

em $\mathbb{F}_{2^3}$.

A matriz geradora binária de $\Gamma(L, g(x))$, por sua vez, era dada por:

$$G^T = \left(\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{array} \right) \sim \left(\begin{array}{cc|cccccc} 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{array} \right) = [I_2 | B^T]^T,$$

onde B é uma matriz 2×6. Com isso, obtemos um código equivalente ao código de Goppa $\Gamma(L, g(x))$ do Exemplo 1.7.6.

Munidos dessas informações e usando o resultado exposto no item (iii) da Proposição 3.2.3, temos que um reticulado $\mathcal{L}_\Gamma$ obtido via Construção A a partir do código equivalente ao código de Goppa $\Gamma(L, g(x))$ definido conforme acima será gerado por:

$$G_{\mathcal{L}(\Gamma)} = \left(\begin{array}{c|c} I_{2\times 2} & 0_{2\times 6} \\ B_{2\times 6} & 2I_{2\times 2} \end{array} \right).$$

3.3 Problemas envolvendo reticulados

Vamos apresentar agora, os problemas existentes no estudo de reticulados, a dizer: o problema do vetor mais curto (SVP - *shortest vector problem*) e o problema do vetor mais próximo (CVP - *closest vector problem*), usando como referência os textos de [47] e [36].

As construções em criptografia baseada em reticulados são elaboradas usando problemas que são considerados difíceis em reticulados, são eles:

PROBLEMA DO VETOR MAIS CURTO (SVP): Consiste em encontrar o vetor não nulo mais curto em um reticulado $\mathcal{L}$, isto é, encontrar um vetor não nulo $v \in \mathcal{L}$ que minimiza a norma euclidiana $||v||$.

PROBLEMA DO VETOR MAIS PRÓXIMO (CVP): Dado um vetor $w \in \mathbb{R}^n$, mas não está no reticulado $\mathcal{L}$, busca-se encontrar o vetor $v \in \mathcal{L}$ que é mais próximo a w, isto é, encontrar um vetor $v \in \mathcal{L}$ que minimiza a norma euclidiana $||w - v||$.

Observação 3.3.1. *Note que pode existir mais de um vetor não nulo que satisfaça o SVP em um reticulado. Por exemplo,em $\mathbb{Z}_2$, todos os vetores na forma $(0, \pm 1)$ e $(\pm 1, 0)$ são soluções para o SVP. Esse é o motivo pelo qual o problema SVP exige "um" vetor mais curto e não "o" vetor mais curto. Uma observação similar se aplica ao CVP.*

Ambos SVP e CVP são problemas profundos e ambos se tornam computacionalmente difíceis a medida que a dimensão n do reticulado cresce. Por outro lado, até as soluções aproximadas para os problemas SVP e CVP possuem suas aplicações em diferentes áreas. Sabemos ainda que o problema CVP é $NP-$completo, como provado em [23] e o problema SVP é $NP-$difícil sob reduções arbitrárias, conforme demonstrado por Ajtai [2], mas observe que esse fato é mais fraco do que dizer que o problema SVP é $NP-$difícil. A demonstração de que o SVP é $NP-$completo é uma conjectura e continua sendo um problema aberto na área de computação.

Observação 3.3.2. *Em geral, ambos SVP e CVP são considerados como sendo problemas difícies em reticulados, mas na prática, é complicado atingir a "generalidade completa" tão idealizada, de acordo com [36]. Em um cenário real, criptossistemas baseados em problemas $NP-$completos tendem a recair em uma subclasse particular de problemas, seja para alcançar eficiência ou permitir a criação de um alçapão. Quando isso acontece, existe sempre a possibilidade de alguma propriedade especial da subclasse de problemas tornar o problema inicial mais fácil de ser resolvido comparado ao caso geral.*

Vamos descrever algumas variações dos problemas SVP e CVP que são aplicados tanto na teoria quanto na prática.

PROBLEMA DA BASE MAIS CURTA (SBP): Consiste em encontrar uma base $v_1, \ldots, v_n$ para um reticulado tal que ela seja a mais curta em algum parâmetro. Por exemplo, podemos querer que

$$\max_{1 \leq i \leq n} ||v_i|| \quad \text{ou} \quad \sum_{i=1}^{n} ||v_i||^2$$

sejam mínimos. Existem diferentes versões desse problema, dependendo dos parâmetros usados para medir o tamanho da base.

PROBLEMA DO VETOR MAIS CURTO APROXIMADO (APPRSVP): Seja $\psi(n)$ uma função de n. Em um reticulado $\mathcal{L}$ de dimensão n, buscamos encontrar um vetor não nulo tal que não seja maior do que $\psi(n)$ vezes maior do que qualquer outro vetor não nulo mais curto. Em outras palavras, se v_s é o vetor mais curto em $\mathcal{L}$, queremos encontrar um vetor não nulo $v \in \mathcal{L}$ que satisfaz:

$$||v|| \leq \psi(n)||v_s||.$$

E cada escolha distinta para $\psi(n)$ define um problema APPRSVP distinto.
Por exemplo, considere um algoritmo que queira encontrar $v \in \mathcal{L}$ satisfazendo

$$||v|| \leq 3\sqrt{n}||v_s|| \quad \text{ou} \quad ||v|| \leq 2^{n/2}||v_s||.$$

Claramente um algoritmo que resolve o primeiro dos problemas é mais forte do que um algoritmo que resolve o segundo.

PROBLEMA DO VETOR MAIS PRÓXIMO APROXIMADO (APPRCVP): É o igual ao APPRSVP, mas agora estamos procurando um vetor que aproxime a solução do CVP.

A seguir, vamos desenvolver a teoria necessária para apresentar dois importantes resultados que definem cotas superiores e inferiores para reticulados em temos de $\det(\mathcal{L})$ e $dim(\mathcal{C})$ para o vetor mais curto em $\mathcal{L}$.
A fim de explicitar o Teorema de Minkowski, vamos introduzir algumas definições:

Definição 3.3.3. *Para todo $a \in \mathbb{R}^n$ e $r > 0$, a **bola fechada** com centro em a e raio r é o conjunto:*

$$B[a, r] = \{x \in \mathbb{R}^n : ||x - a|| \leq r\}.$$

Definição 3.3.4. *Seja S um subconjunto de $\mathbb{R}^n$.*

- *S é **limitado** se o comprimento de todos os vetores de S forem limitados. De maneira equivalente, S é limitado se existe um raio r tal que S esteja contido inteiramente dentro da bola $B[0, r]$.*
- *S é **simétrico** se para todo ponto $a \in S$, o ponto $-a \in S$.*

- *S é **convexo** se quaisquer dois pontos a e b em S puderem ser ligados por um segmento de reta que esteja totalmente contido em S.*
- *S é **fechado** se ele satisfizer a seguinte propriedade: se $a \in \mathbb{R}^n$ é um ponto tal que toda bola $B[a, r]$ contém um ponto de S, então $a \in S$.*

Proposição 3.3.5. *[36] Seja $\mathcal{L} \subset \mathbb{R}^n$ um reticulado de dimensão n e seja $\mathcal{F}$ uma região fundamental de $\mathcal{L}$. Então, todo vetor $w \in \mathbb{R}^n$ pode ser escrito na forma*

$$w = t + v,$$

para únicos $t \in \mathcal{F}$ e $v \in \mathcal{L}$. De maneira equivalente, a união de translações da região fundamental

$$\mathcal{F} + v = \{t + v : t \in \mathcal{F}\},$$

com v variando sobre os vetores do reticulado $\mathcal{L}$, cobre o espaço $\mathbb{R}^n$. Tal fato está ilustrado na Figura 3.3.

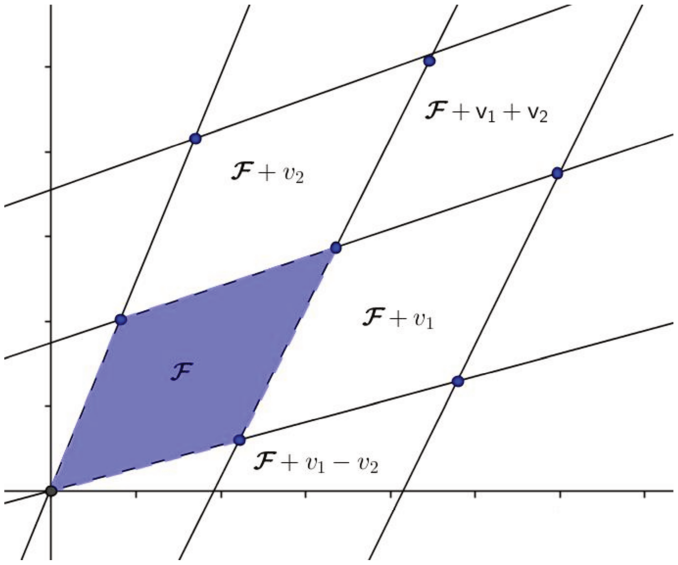


Figura 3.3: Translações de $\mathcal{F}$ por vetores em $\mathcal{L}$ cobre o plano $\mathbb{R}^2$

Fonte: [36], p. 368

Demonstração. Seja $v_1, \ldots, v_n$ uma base do reticulado $\mathcal{L}$ que gera a região fundamental $\mathcal{F}$. Então $v_1, \ldots, v_n$ são linearmente independentes em $\mathbb{R}^n$, sendo também, uma base de $\mathbb{R}^n$. Isso nos diz que qualquer $w \in \mathbb{R}^n$ pode ser escrito na forma:

$$w = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n,$$

para $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$. Agora, vamos escrever cada α_i como:

$$\alpha_i = t_i + a_i, \quad \text{com} \quad 0 \leq t_i \leq 1 \quad \text{e} \quad a_i \in \mathbb{Z}.$$

Então,

$$w = t_1v_1 + t_2v_2 + \cdots + t_nv_n + a_1v_1 + a_2v_2 + \cdots + a_nv_n,$$

onde os primeiros n formam um vetor $t \in \mathcal{F}$ e os próximos formam um vetor $v \in \mathcal{L}$. Isso mostra que w pode ser escrito como queríamos.

Agora, suponha que $w = t + v = t' + v'$ possua duas representações como soma de vetores de $\mathcal{F}$ com vetores de $\mathcal{L}$. Então:

$$(t_1 + a_1)v_1 + \cdots + (t_n + a_n)v_n = ({t_1}' + {a_1}')v_1 + \cdots + ({t_n}' + {a_n}')v_n.$$

Como $v_1, \ldots, v_n$ são linearmente independentes, segue que:

$$t_i + a_i = {t_i}' + {a_i}' \ \ \forall\ i = 1, 2, \ldots, n.$$

Além disso,

$$t_i - {t_i}' = {a_i}' - a_i \in \mathbb{Z}$$

é um inteiro. Mas também sabemos que t_i e ${t_i}'$ são maiores ou iguais a zero e estritamente menores do que 1, logo, a única maneira de $t_i - {t_i}'$ ser um inteiro é se $t_i = {t_i}'$. Assim, $t = t'$ e também:

$$v = w - t = w - t' = v'.$$

Isso completa a prova de que $t \in \mathcal{F}$ e $v \in \mathcal{L}$ são unicamente determinados por w. □

Teorema 3.3.6. *[36]* ***(Teorema de Minkowski)*** *Seja $\mathcal{L} \subset \mathbb{R}^n$ um reticulado de dimensão n e seja $S \subset \mathbb{R}^n$ um conjunto convexo e simétrico cujo volume satisfaz:*

$$vol(S) > 2^n \det(\mathcal{L}).$$

Então, S contém um vetor não nulo do reticulado $\mathcal{L}$.

Demonstração. Seja $\mathcal{F}$ uma região fundamental de $\mathcal{L}$. Logo, de acordo com a Proposição 3.3.5, sabemos que todo vetor $a \in S$ pode ser escrito de forma única como:

$$a = v_a + w_a,$$

com $v_a \in \mathcal{L}$ e $w_a \in \mathcal{F}$. "Dilatamos"S por um fator de $\frac{1}{2}$:

$$\frac{1}{2}S = \left\{\frac{1}{2}a : a \in S\right\}$$

e consideramos a aplicação:

$$\begin{aligned} \frac{1}{2}S &\longrightarrow \mathcal{F} \\ \frac{1}{2}a &\mapsto w_{\frac{1}{2}a}. \end{aligned} \tag{3.3.1}$$

Ao encolhermos S por um fator de 2, seu volume é modificado por um fator de 2^n, então:

$$vol\left(\frac{1}{2}S\right) = \frac{1}{2^n}vol(S) > \det(\mathcal{L}) = vol(\mathcal{F}).$$

A aplicação apresentada em 3.3.1 é dada por uma coleção finita de aplicações de translações (esse é o motivo pelo qual assumimos que S é limitado), portanto, a aplicação 3.3.1 preserva volume. Além disso, o fato de que o domínio $\frac{1}{2}S$ possui volume estritamente maior do que o volume da imagem de $\mathcal{F}$ implica que existem pontos distintos $\frac{1}{2}a_1$ e $\frac{1}{2}a_2$ com a mesma imagem em $\mathcal{F}$.

Com isso, encontramos pontos distintos tais que:

$$\frac{1}{2}a_1 = v_1 + w \quad \text{e} \quad \frac{1}{2}a_2 = v_2 + w,$$

com $v_1, v_2 \in \mathcal{L}$ e $w \in \mathcal{F}$.

Subtraindo tais vetores, temos que:

$$\frac{1}{2}a_1 - \frac{1}{2}a_2 = v_1 - v_2 \in \mathcal{L}.$$

Observamos então que:

$$\underbrace{\frac{1}{2}a_1 + \overbrace{(-\frac{1}{2}a_2)}^{-a_2 \in S \text{ por simetria}}}_{\in S \text{ por convexidade}}$$

Com isso, $0 \neq v_1 - v_2 \in S \cap \mathcal{L}$ e construímos um ponto não nulo do reticulado $\mathcal{L}$ em S.

Isso completa a prova do Teorema de Minkowski assumindo que o volume de S é estritamente maior do que $2^n \det(\mathcal{L})$.

□

Definição 3.3.7. *Dizemos que K é um **corpo convexo** se K for um conjunto convexo limitado e tal que $int(S) \neq \emptyset$.*

Definição 3.3.8. *Seja K um corpo convexo com $0 \in int(K)$. Então, para cada $x \in \mathbb{R}^n$, existe um número positivo λ tal que $x \in \lambda K$. Definimos então uma função F_K da seguinte maneira:*

$$F_K(x) = inf\{\lambda : \lambda > 0 \ e \ x \in \lambda K\}.$$

*Tal função é chamada de **função distância** ou **função de Gauge**.*

Definição 3.3.9. *Definimos o **primeiro mínimo de $\mathcal{L}$** com relação à um corpo convexo K como*

$$\lambda_1 = \lambda_{1,K}(\mathcal{L}) = \min_{x \in \mathcal{L}\setminus\{0\}} F_K(x).$$

Em criptografia, o valor do primeiro mínimo $\lambda_{1,K}$ é importante para estimar parâmetros para reticulados que não são munidos de alguma estrutura especial e decidir se o algoritmo baseado em tal reticulado terá sucesso em sua implementação. Experimentos vem sendo desenvolvidos por Gama, Nguyen e Regev ([26]) e mostram que é suficiente, para reticulados aleatórios, estimar o valor do primeiro mínimo $\lambda_{1,K}$, sem precisar resolver o problema SVP, por exemplo.

É possível ainda, de acordo com [52], reescrever o SVP e o apprSVP usando o valor do primeiro mínimo da seguinte maneira:

PROBLEMA DO VETOR MAIS CURTO (SVP): Dada uma base de um reticulado $\mathcal{L}$, de posto d encontre $u \in \mathcal{L}$ tal que $||u|| = \lambda_1(\mathcal{L})$.

PROBLEMA DO VETOR MAIS CURTO APROXIMADO (APPRSVP): Dada uma base de um reticulado $\mathcal{L}$, de posto d e um fator de aproximação $f \geq 1$, encontre um vetor não nulo $u \in \mathcal{L}$ tal que $||u|| \leq f\lambda_1(\mathcal{L})$.

A proposição a seguir nos dá uma estimativa para o primeiro mínimo de $\mathcal{L}$ em corpos convexos.

Proposição 3.3.10. *[14] Seja $K \subset \mathbb{R}^n$ um corpo convexo e $\mathcal{L}$ um reticulado. Então,*

$$\lambda_{1,K} \leq 2\left(\frac{\det(\mathcal{L})}{vol(K)}\right)^{1/n}.$$

Demonstração. Se $\lambda_{1,K} > 2\left(\frac{\det(\mathcal{L})}{vol(K)}\right)^{1/n}$, então teríamos que:

$$vol(\lambda_{1,K}K) = \lambda_K^n vol(K) > 2^n\left(\frac{\det(\mathcal{L})}{vol(K)}\right)vol(K).$$

Isso nos diz que existe $x \in \mathcal{L}$ tal que $x \in \lambda K$. Logo, $F_K(x) < \lambda_{1,K}$, o que é um absurdo, já que $\lambda_{1,K}$ é o primeiro mínimo. □

Proposição 3.3.11. *[14] Seja $\mathcal{L}$ um reticulado. Existe $x \in \mathcal{L}$ tal que*

$$||x||_p \leq n^{1/p}(\det(\mathcal{L}))^{1/n}.$$

Demonstração. Sabemos que

$$||x||_p \leq n^{1/p}||x||_\infty,$$

e, portanto, temos que $B_\infty(r) \subseteq B_p(n^{1/p}r)$, onde $B_\infty(r)$ denota a bola de raio r na norma infinito e $B_p(n^{1/p}r)$, denota a bola de raio $n^{1/p}r$ com a norma p, ou seja,

$$B_\infty(r) = \{(x_1, \ldots, x_n) : |x_i| \leq r\} \text{ e } B_p(n^{1/p}r) = \{(x_1, \ldots, x_n) : \sum |x_i|^p \leq n^{1/p}r\}.$$

Tome $r = (\det(\mathcal{L}))^{1/n}$. Então, o volume de $B_\infty(r)$ é o volume do cubo de lado $2r$ centrado em x, dado por:

$$vol(B_\infty(r)) = 2^n r^n = 2^n \det(\mathcal{L}).$$

Assim, $vol(B_p(n^{1/p}r)) \geq 2^n \det(\mathcal{L})$, e, do Teorema 3.3.6, existe $x \in \mathcal{L}$ tal que $||x||_p \leq n^{1/p}\det(\mathcal{L})^{1/n}$. □

Se tomarmos $p = 2$, temos então o seguinte resultado:

Proposição 3.3.12. *[36] Todo reticulado $\mathcal{L}$ de dimensão n contém um vetor não nulo $x \in \mathcal{L}$ que satisfaz:*

$$||x||_2 \leq \sqrt{n}\det(\mathcal{L})^{1/n}.$$

Note que no caso em que $p = 2$ temos uma aproximação ainda melhor para a cota estabelecida pela Proposição 3.3.12. De fato, temos que o corpo convexo K da Proposição 3.3.10 é tal que $K = B_2(r)$, ou seja, é a bola com a norma 2 de raio r e vale que ([17], p. 9):

$$vol(K) = \frac{\pi^{n/2}}{\left(\frac{n}{2}\right)!}.$$

Com isso, $vol(K)^{1/n} = \dfrac{\sqrt{\pi}}{\left(\left(\frac{n}{2}\right)!\right)^{1/n}}$. Utilizando a aproximação de Stirling, temos que:

$$\left(\frac{n}{2}\right)! \approx (\sqrt{2\pi e})e^{-n/2}\left(\frac{n}{2}\right)^{\frac{n}{2}-\frac{1}{2}}$$

$$\left(\left(\frac{n}{2}\right)!\right)^{1/n} \approx (\sqrt{2\pi e})^{1/n}e^{-1/2}\left(\frac{n}{2}\right)^{\frac{1}{2}-\frac{1}{2n}}.$$

Se n é assintótico:

$$\left(\left(\frac{n}{2}\right)!\right)^{1/n} \approx \left(\frac{n}{2e}\right)^{1/2}.$$

Então, temos o seguinte resultado:

$$(vol(K))^{1/n} \approx \frac{\sqrt{2\pi}}{\sqrt{n}}e \leq \frac{2\sqrt{n}}{\sqrt{2\pi e}}(\det(\mathcal{L}))^{1/n}$$

e a constante $\frac{2}{\sqrt{2\pi e}} \simeq 0.4839$ nos fornece um limitante melhor do que o da Proposição 3.3.12.

Definição 3.3.13. *Para uma dada dimensão n, a constante de Hermite γ_n é definida como sendo o menor valor tal que todo reticulado $\mathcal{L}$ de dimensão n contém um vetor não nulo $v \in \mathcal{L}$ que satisfaz:*

$$||v||^2 \leq \gamma_n \det(\mathcal{L})^{2/n}.$$

Observe que na versão apresentada na Proposição 3.3.12, temos que $\gamma_n \leq n$. O valor exato de γ_n, de acordo com [36], é conhecido somente para $1 \leq n \leq 8$ e para $n = 24$:

$$\gamma_2^2 = \frac{4}{3},\ \ \gamma_3^3 = 2,\ \ \gamma_4^4 = 4,\ \ \gamma_5^5 = 8,\ \ \gamma_6^6 = \frac{64}{3},\ \ \gamma_7^7 = 64,\ \ \gamma_8^8 = 256 \ \text{ e } \ \gamma_{24} = 4.$$

É importante ressaltar que encontrar o valor exato para a constante de Hermite γ_n é um problema muito difícil e foi um dos principais tópicos do estudo de Geometria dos Números por

Minkowski em [49]. Para as propostas criptográficas, o valor de γ_n é interessante quando n for muito grande ou assintótico. Para valores de n consideravelmente grandes, sabe-se que a constante de Hermite satisfaz:

$$\frac{n}{2\pi e} \leq \gamma_n \leq \frac{n}{\pi e},$$

onde $\pi = 3.14159...$ e $e = 2.71828...$ são as constantes conhecidas.

Desta forma, foi proposto um novo problema semelhante ao SVP que é chamado de HSVP [52], definido como:

PROBLEMA DO VETOR MAIS CURTO DE HERMITE (HSVP): Dada uma base de um reticulado $\mathcal{L}$ com posto d e um fator de aproximação $f > 0$, encontrar um vetor não nulo $u \in \mathcal{L}$ tal que $||u|| \leq fvol(\mathcal{L})^{1/d}$.

Ao contrário do SVP e do apprSVP, é possível averiguar facilmente a solução do HSVP: de fato, dado $u, \mathcal{L}$ e f podemos verificar em tempo polinomial se u pertence ou não ao reticulado $\mathcal{L}$ e se $||u|| \leq fvol(\mathcal{L})^{1/d}$. Além disso, segundo [52], se for possível resolver o apprSVP com um fator de aproximação f, então o HSVP poderá ser resolvido com um fator $f\sqrt{\gamma_d}$, em um reticulado com posto d.

3.4 Algoritmos importantes em reticulados

Nesta seção vamos apresentar dois importantes algoritmos que facilitam a resolução de problemas envolvendo reticulados: o algoritmo de Babai e o algoritmo LLL. O primeiro deles resolve o problema do vetor mais próximo (CVP) quando é dada uma base boa para um reticulado e o segundo é utilizado para encontrar uma base de vetores curtos no reticulado, ou seja, ele resolve o problema no vetor mais curto (SVP) de maneira aproximada, o que permite reduzir a complexidade de outros problemas em reticulados.

Esses dois algoritmos possuem mais destaque ainda, pois eles estão envolvidos no ataque à segurança de criptossistemas baseados em reticulados, que é o tema do próximo capítulo. Para a redação dessa seção, usamos os textos de [36] e [52].

3.4.1 Algoritmo de Babai

Se um reticulado $\mathcal{L}$ possui uma base $v_1, v_2, \ldots, v_n$ formada por vetores dois a dois ortogonais, ou seja,

$$v_i \cdot v_j = 0, \quad \forall i \neq j$$

então é fácil resolver os problemas SVP e CVP. Nesta seção vamos, então, compreender o algoritmo desenvolvido por Babai [5] em 1985 para abordar esses problemas difíceis em reticulados considerando tais hipóteses.

Para resolver SVP, observe que o comprimento de qualquer vetor em um reticulado $\mathcal{L}$ com base ortogonal é dado pela fórmula:

$$||a_1v_1 + a_2v_2 + \cdots + a_nv_n||^2 = {a_1}^2||v_1||^2 + {a_2}^2||v_2||^2 + \cdots + {a_n}^2||v_n||^2.$$

Como $a_1, \ldots, a_n \in \mathbb{Z}$, temos que os vetores mais curtos não nulos em $\mathcal{L}$ são simplesmente os vetores mais curtos no conjunto $\{\pm v_1, \pm v_2, \ldots, \pm v_n\}$.

De modo similar, suponha que queremos encontrar o vetor em $\mathcal{L}$ que está mais próximo a um dado vetor $w \in \mathbb{R}^n$. Escrevemos, primeiramente:

$$w = t_1 v_1 + t_2 v_2 + \cdots + t_n v_n, \quad \text{com} \quad t_1, \ldots, t_n \in \mathbb{R}.$$

Então, para $v = a_1 v_1 + a_2 v_2 + \cdots + a_n v_n \in \mathcal{L}$, temos:

$$||v - w|| = (a_1 - t_1)^2 ||v_1||^2 + (a_2 - t_2)^2 ||v_2||^2 + \cdots + (a_n - t_n)^2 ||v_n||^2. \tag{3.4.1}$$

Como a_i deve ser inteiro, a Equação 3.4.1 será minimizada se tomarmos cada a_i como sendo o inteiro mais próximo ao correspondente t_i.

Observando esse processo, é difícil não querer utilizá-lo com uma base qualquer não ortogonal, porém, se a base não for ortogonal, o algoritmo não irá funcionar corretamente, como veremos a seguir.

A base $\{v_1, v_2, \ldots, v_n\}$ para o reticulado $\mathcal{L}$ determina uma região fundamental. A Proposição 3.3.5 nos diz que as translaçoes de $\mathcal{F}$ por elementos de $\mathcal{L}$ preenchem todo o espaço do $\mathbb{R}^n$, logo, cada $w \in \mathbb{R}^n$ está em uma única translação $\mathcal{F} + v$ de $\mathcal{F}$ por um elemento $v \in \mathcal{L}$. Consideramos o vértice do paralelogramo $\mathcal{L} + v$ que está mais próximo a w como sendo nossa solução hipotética para o problema CVP. É fácil encontrar o vértice mais próximo, já que:

$$w = v + \varepsilon_1 v_1 + \varepsilon_2 v_2 + \cdots + \varepsilon_n v_n \quad \text{para algum} \quad 0 \leq \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n < 1,$$

e então, somente substituímos ε_i por 0 se ele for menor do que $\frac{1}{2}$ e substituímos por 1 se ele for maior ou igual a $\frac{1}{2}$.

Vamos introduzir uma noção ainda não mencionada envolvendo reticulados, que são as bases "boas" e as bases "ruins". Uma base boa para um reticulado, é uma base que consiste em vetores curtos e ortogonais entre si e uma base ruim para um reticulado, por sua vez, é a formada por vetores longos que geralmente apontam para a mesma direção (ou oposta) ou ainda por vetores que possuem o ângulo entre eles muito pequenos.

Se tentarmos resolver o problema CVP com uma base ruim, enfrentaremos problemas ao encontrar o vértice mais próximo à um dado vetor que se encontra dentro do paralelogramo definido por essa base. E é importante ressaltar que os problemas começam a ficar muito piores a medida que a dimensão do reticulado vai aumentando.

O método geral para encontrar a solução do problema CVP, chamado de Algoritmo de Babai para o Vértice ou Vetor mais Próximo, está explicitado no Algoritmo 3.1.

Em geral, se os vetores da base são razoavelmente ortogonais uns aos outros, o algoritmo resolve algumas versões do APPRCVP, mas se os vetores da base forem altamente não ortogonais, então o vetor devolvido pelo algoritmo estará muito longe do vetor mais próximo ao vetor w.

3.4.2 Algoritmo LLL

O algoritmo LLL para redução de bases foi criado por Arjen Lenstra, Hendrik Lenstra e László Lováz em 1982. Como não é conhecido um algoritmo eficiente que resolva o problema do vetor

Algoritmo 3.1 Algoritmo de Babai para o Vetor mais Próximo

Seja $\mathcal{L} \subset \mathbb{R}^n$ um reticulado com base $v_1, v_2, \ldots, v_n$ e seja $w \in \mathbb{R}^n$ um vetor arbitrário. Se os vetores da base forem suficientemente ortogonais um ao outro, então resolvemos o CVP da seguinte maneira:
(1) Escreve $w = t_1 v_1 + t_2 v_2 + \cdots + t_n v_n$ com $t_1, \ldots, t_n \in \mathbb{R}$.
(2) Defina $a_i = \lfloor t_i \rceil$ para $i = 1, 2, \ldots, n$.
(3) Devolva o vetor $v = a_1 v_1 + a_2 v_2 + \cdots + a_n v_n$.

mais curto (SVP) em tempo polinomial em reticulados de dimensão grande, o algoritmo LLL é usado para encontrar uma aproximação para o vetor mais curto e essa aproximação é suficiente para muitas aplicações. Além disso, o algoritmo LLL não só encontra um vetor mais curto em um reticulado, mas sim uma base mais curta para tal reticulado. Aqui apresentaremos uma versão resumida do algoritmo LLL e mais detalhes podem ser encontrados em [36].

Seja $\beta = \{v_1, \ldots, v_n\}$ uma base de um reticulado $\mathcal{L}$ de posto completo. Note que, existem infinitas possíveis escolhas para base de um reticulado. É possível, como já visto anteriormente na Proposição 3.1.6, obter uma outra base a partir de uma transformação unimodular. Reciprocamente, quaisquer duas bases de um reticulado diferem uma da outra apenas por uma transformação unimodular.

Para algumas aplicações, estamos interessados em bases cujos vetores sejam tão ortogonais quanto queiramos. Uma maneira de medir a ortogonalidade de uma base é observar o valor do chamado *defeito de ortogonalidade*, dado por:

$$\frac{||v_1|| \cdot ||v_2|| \cdot \ldots ||v_n||}{\det(\mathcal{L})}.$$

Observe então o seguinte resultado:

Proposição 3.4.1. *[52] (Desigualdade de Hadamard) O determinante de um reticulado é menor ou igual ao produto da norma dos vetores da base, ou seja:*

$$\det(\mathcal{L}) \leq \prod_{i=1}^{n} ||v_i||.$$

Devido à esse resultado, note que o defeito de ortogonalidade é sempre maior ou igual a 1 e a igualdade vale se os vetores da base são ortogonais. Assim, é possível formular um problema para tentar encontrar uma base para o reticulado tal que essa base minimize o defeito de ortogonalidade. Uma maneira de obter uma base com um defeito de ortogonalidade pequeno é aplicando o algoritmo LLL.

O algoritmo de redução LLL inicia com a aplicação do processo de ortogonalização de Gram-Schmidt na base para obter n vetores ortogonais. Como sabemos, o processo de ortogonalização funciona da seguinte maneira: defina $v_1^* = v_1$ e calcule os outros vetores recursivamente a partir da seguinte fórmula:

$$v_i^* = v_i - \sum_{j=1}^{i-1} \frac{< v_i, v_j^* >}{||v_j^*||^2} v_j^*,$$

para $i = 2, 3, \ldots, n$.

Sejam $\mu_{i,j}$ os coeficientes referentes à $\dfrac{< v_i, v_j^* >}{||v_j^*||^2}$. Como tais coeficientes, em geral, não são inteiros, o processo de Gram-Schmidt não produz uma base para o reticulado.

Note que os coeficientes $\mu_{i,j}$ seriam todos nulos se os vetores da base fossem ortogonais. Nesse caso, o defeito de ortogonalidade atingiria seu valor mínimo, que é 1. Esse é o método do algoritmo LLL que chamamos de redução de base: podemos subtrair um múltiplo inteiro de um vetor da base de outro vetor da base tal que $|\mu_{i,j}| \leq \frac{1}{2}$ para $1 \leq j < i \leq n$. Se $|\mu_{i,j}| > \frac{1}{2}$, substituímos v_i por

$$v_i \leftarrow v_i - \lfloor \mu_{i,j} \rceil v_j,$$

onde $\lfloor \mu_{i,j} \rceil$ é o inteiro mais próximo à $\mu_{i,j}$. Após esse passo de redução, se retornarmos ao processo de Gram-Schmidt, notamos que $|\mu_{i,j}| \leq \frac{1}{2}$, para todo $\mu_{i,j}$.

O segundo passo do algoritmo LLL é manter a condição de Lovász:

$$||v_k^*||^2 \geq (\delta - \mu_{k,k-1}^2)||v_{k-1}^*||^2.$$

Aqui, δ é um parâmetro que geralmente vale $\frac{3}{4}$. Note que a condição de Lovász deve valer para pares de vetores consecutivos. Como v_k^* e v_{k-1}^* são ortogonais por construção, a condição de Lovász é equivalente à:

$$||v_k^* + \mu_{k,k-1} v_{k-1}^*||^2 \geq \delta ||v_{k-1}^*||^2.$$

O vetor $v_k^* + \mu_{k,k-1} v_{k-1}^*$ pode ser interpretado como a projeção do vetor v_k no complemento ortogonal do espaço vetorial gerado por v_1 até v_{k-2}, enquanto o vetor v_{k-1}^* é a projeção correspondente à v_{k-1}. Em outras palavras, a condição de Lovász nos diz que deve existir um intervalo entre as normas dessas duas projeções.

No algoritmo LLL trocamos v_k e v_{k-1} se a condição de Lovász falhar para v_k^* e v_{k-1}^* e repetimos o primeiro passo, que é o passo de redução. Esse algoritmo termina em tempo polinomial e a base resultante possui defeito de ortogonalidade limitado por:

$$1 \leq \frac{||v_1|| \cdot ||v_2|| \cdot \ldots ||v_n||}{\det(\mathcal{L})} \leq 2^{\frac{n(n-1)}{4}}.$$

3.5 Decodificação de reticulados q-ários via Construção A

Estudaremos por fim um processo de decodificação para reticulados construídos a partir de códigos binários por meio da Construção A. Vamos mostrar também que decodificar um código binário $\mathcal{C} \subset \mathbb{Z}_q^n$ corresponde a decodificar o reticulado binário $\mathcal{L}_2 \subseteq \mathbb{R}^2$ na métrica euclidiana. Esta redação está baseada nos trabalhos de [40] e [15].

Seja $\mathcal{C} \subset \mathbb{Z}_q^n$ um código $q-$ário. Devido ao isomorfismo existente entre $\mathcal{L}_q(\mathcal{C})/q\mathbb{Z}^n \simeq \mathcal{C}$ não faremos distinção entre os elementos de $\mathcal{L}_q(\mathcal{C})/q\mathbb{Z}^n$ e das palavras-código de $\mathcal{C}$. Seja $r \in \mathbb{R}^n$ um vetor recebido e seja $z \in \mathcal{L}_q(\mathcal{C})$ o vetor mais próximo a r, considerado sob a métrica de Lee. Vamos apresentar uma maneira de encontrar via Construção A uma representação para $\overline{z} \in \mathcal{C}$. Em outras palavras, vamos mostrar como resolver o CVP usando a métrica de Lee em um reticulado $q-$ário

obtido via Construção A a partir de um código linear $\mathcal{C}$ e ainda apresentar um algoritmo que decodifica palavras nesse código $\mathcal{C}$.

Proposição 3.5.1. *[15] Seja $\mathcal{L}_q(\mathcal{C})$ um reticulado $q-$ário e $r = (r_1, \ldots, r_n)^T \in \mathbb{R}^n$ o vetor recebido. Dado um elemento $\overline{x} \in \mathcal{L}_q(\mathcal{C})/q\mathbb{Z}^n$, $x = (x_1, \ldots, x_n)^T$, a representação de $z = (z_1, \ldots, z_n)^T \in \mathcal{L}_q(\mathcal{C})$ de $\overline{x}$ é mais próxima a r em $\mathcal{L}_q(\mathcal{C})$, considerando que a métrica de Lee é dada por $z_i = x_i + qw_i$, onde $w_i = \lceil \frac{r_i - x_i}{q} \rfloor$.*

Demonstração. Uma representação para a classe de x é dada por $z = x + qw$, onde $w \in \mathbb{Z}^n$ e a distância de Lee $d(r, z) = \sum_{i=1}^{n} |r_i - x_i - qw_i|$ é mínima quando $w_i = \lceil \frac{r_i - x_i}{q} \rfloor$. □

Proposição 3.5.2. *[15] Seja $\mathcal{L}_q(\mathcal{C})$ um reticulado $q-$ário e $r = (r_1, \ldots, r_n)^T \in \mathbb{R}^n$ o vetor recebido. Seja $r(mod\ q) \in [0, q)^n$, o vetor obtido a partir de r por reduções módulo q em cada entrada do vetor. Se $\overline{x} \in \mathcal{C}$ é o elemento de $\mathcal{C}$ mais próximo a $r(mod\ q)$, considerando a métrica de Lee, então $z \in \mathcal{L}_q(\mathcal{C})$, dado pela Proposição 3.5.1, tal que $\overline{z} = \overline{x}$, é o ponto do reticulado mais próximo a r.*

Demonstração. Seja $r(mod\ q) = r^*$, isto é, $r = (r_1, \ldots, r_n) = (r_1^*, \ldots, r_n^*) + q(t_1, \ldots, t_n)$, onde $0 \leq r_i^* \leq q$ e $t_i \in \mathbb{Z}$ para $i = 1, \ldots, n$. Considere $\overline{x} \in \mathcal{C}$ com $x = (x_1, \ldots, x_n)$, onde $0 \leq x_i \leq q - 1,\ i = 1, \ldots, n$, como sendo o ponto mais próximo de $r(mod\ q)$, admitindo a métrica de Lee.

Vamos provar que o ponto mais próximo de $r(mod\ q)$ está na mesma classe de x em $\mathcal{L}_q(\mathcal{C})/q\mathbb{Z}^n$. Para cada classe $\overline{a} \in \mathcal{C}$ com $a = (a_1, \ldots, a_n)$, de acordo com a Proposição 3.5.1, conseguimos encontrar o representante a^* mais próximo de r na métrica de Lee.

Mostraremos que $d(r, a^*) = d(r(mod\ q), \overline{a})$. Para a métrica de Lee, temos:

$$d(r, a^*) = \sum_{i=1}^{n} |{r_i}^* - a_i - \alpha_i q|,$$

onde $\alpha_i = ([\frac{{r_i}^* - a_i}{q} + t_i] - t_i)$. Então, $-1 \leq \frac{r_i^* - a_i}{q} \leq 1$, pois $|r_i^* - x_i| \leq q, \alpha_i \in \{-1, 0, 1\}$.

Assim, podemos observar que:

- Se $\alpha_i = 0$, para algum i, então $-\frac{q}{2} \leq r_i^* - a_i \leq \frac{q}{2}$ e isso implica que

$$min\{|r_i^* - a_i|, q - |r_i^* - a_i|\} = |r_i^* - a_i|.$$

- Se $\alpha_i = 1$, para algum i, então $-\frac{q}{2} \leq r_i^* - a_i \leq q$ e então

$$min\{|r_i^* - a_i|, q - |r_i^* - a_i|\} = q - |r_i^* - a_i| \text{ e } |r_i^* - a_i| = r_i^* - a_i.$$

- Se $\alpha_i = -1$, para algum i, então $-q \leq r_i^* - a_i \leq -\frac{q}{2}$, logo

$$min\{|r_i^* - a_i|, q - |r_i^* - a_i|\} = q - |r_i^* - a_i| \text{ e } |r_i^* - a_i| = -(r_i^* - a_i).$$

Portanto,

$$d(r,a^*) = \sum_{i=1}^{n} |{r_i}^* - a_i - \alpha_i q| = \sum_{i=1}^{n} min\{|r_i^* - a_i|, q - |r_i^* - a_i|\} = d(r(mod\ q), a).$$

Como z é o ponto mais próximo de r, então $\overline{z}$ minimiza $d(r(mod\ q), z)$, ou seja, $\overline{z} = \overline{x}$. □

Vamos fazer agora um breve resumo dos passos necessários para, dado $r = (r_1, \ldots, r_n) \in \mathbb{R}^n$, encontrar um ponto de $\mathcal{L}_q(\mathcal{C})$ mais próximo de r na métrica de Lee:

1. Calcule $r(mod\ q)$, fazendo a redução módulo q em todas as entradas do vetor r.
2. Como $r(mod\ q) \in [0,q)^n$, calcule o ponto $\overline{x} = (\overline{x_1}, \ldots, \overline{x_n})$ do código $\mathcal{C}$ mais próximo de $r(mod\ q)$ na métrica de Lee.
3. Para $i = 1, \ldots, n$, calcule $w_i = \lceil \frac{r_i - x_i}{q} \rfloor$ e tome $w = (w_1, \ldots, w_n)$.
4. Tome $z = (x_1, \ldots, x_n) + q(w_1, \ldots, w_n)$ como sendo o ponto mais próximo a r com a métrica de Lee.

Note que nem todo código $q-$ário possui um algoritmo eficiente de decodificação utilizando a métrica de Lee. A maioria dos algoritmos conhecidos é desenvolvido para decodificar vetores com entradas em $\mathbb{Z}_q^n$, chamado de "hard decoding". Um exemplo de códigos que possuem um algoritmo desse tipo são os códigos BCH definidos sobre $\mathbb{Z}_q$, onde q é uma potência de número primo.

Exemplo 3.5.3. *[40] Seja $\mathcal{C}$ o código BCH definido em $\frac{\mathbb{Z}_4[x]}{<f(x)>}$, onde $f(x) = x^3 + x + 1$, $\alpha = \beta^2$, tal que $f(\beta) = 0$ e com matriz de paridade e matriz geradora em $\mathbb{Z}_4$ dadas, respectivamente, por:*

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \alpha^5 & \alpha & 1 & \alpha^3 & \alpha^6 & \alpha^2 & \alpha^4 \end{pmatrix}, \qquad G = \begin{pmatrix} 3 & 3 & 2 & 3 & 0 & 0 & 1 \\ 3 & 1 & 1 & 2 & 0 & 1 & 0 \\ 1 & 2 & 1 & 3 & 1 & 0 & 0 \end{pmatrix},$$

Uma matriz geradora para o reticulado $4-$ário $\mathcal{L} = \phi^{-1}(\mathcal{C})$ é:

$$G^T = \begin{pmatrix} 1 & 0 & 0 & 2 & 1 & 1 & 3 \\ 0 & 1 & 0 & 1 & 3 & 1 & 2 \\ 0 & 0 & 1 & 3 & 2 & 1 & 1 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 \end{pmatrix}.$$

Seja $r = (0, 7, 4, 8, 0, 12, 0)$ um vetor recebido. Temos que $r(mod\ 4) = (0, 3, 0, 0, 0, 0, 0)$ e o ponto de $\mathcal{C}$ mais próximo de $r(mod\ 4)$ segundo a métrica de Lee é $\overline{x} = (0, 0, 0, 0, 0, 0, 0)$. Vamos calcular o vetor w. Temos que $w_1 = 0, w_2 = [\frac{7}{4}] = 2, w_3 = [\frac{4}{4}] = 1, w_4 = [\frac{8}{4}] = 2, w_5 = 0, w_6 = [\frac{12}{4}] = 3$ e $w_7 = 0$. Assim, temos que $z = (0, 0, 0, 0, 0, 0, 0) + 4(0, 2, 1, 2, 0, 3, 0) = (0, 8, 4, 8, 0, 12, 0)$.

Com isso, a Proposição 3.5.2 nos diz que, se o reticulado $q-$ário apresentar um algoritmo eficiente de decodificação na métrica de Lee, teremos um algoritmo eficiente de decodificação para o código associado. A complexidade de tal algoritmo está diretamente relacionada com a complexidade de decodificar o reticulado $q-$ário na métrica da soma.

Assim, dado $\mathcal{C} \subset \mathbb{Z}_q^n$ um código linear $q-$ário, podemos efetuar, de acordo com [40], a sua decodificação conforme descrito no Algoritmo 3.2 :

Algoritmo 3.2 Decodificação de um código $\mathcal{C} \subset \mathbb{Z}_q^n$ via reticulados $q-$ários usando a métrica de Lee

---

Seja $\mathcal{C} \subset \mathbb{Z}_q^n$ um código linear $q-$ário e suponha que o vetor $\overline{y} \in \mathbb{Z}_q^n$ tenha sido recebido.

(1) Utilize o reticulado $\mathcal{L}_q(\mathcal{C})$ para decodificar y com a métrica de Lee, encontando $z \in \mathcal{L}_q(\mathcal{C})$ como sendo o ponto mais próximo a y.

(2) Encontre o elemento $\overline{z} \in \mathcal{L}_q(\mathcal{C})/q\mathbb{Z}^n$ fazendo as reduções módulo q em cada uma das entradas do vetor z.

O ponto de $\mathcal{C}$ mais próximo de $\overline{y}$ com a métrica de Lee será o ponto $\overline{z} \in \mathcal{C}$.

Capítulo 4

Criptografia baseada em reticulados: GGH e NTRU

Sistemas criptográficos baseados em reticulados são vistos, atualmente, como uma promissora alternativa para a evolução da criptografia clássica e seu estudo iniciou-se com a publicação de Ajtai, em 1996, intitulada "*Generating hard instances of Lattice problems*", encontrada em [1].

Essa afirmação, segundo [62], é fortemente defendida por meio de vários estudos acerca da sua provável resistência à computadores quânticos. Tal fato se dá devido à grande dificuldade em encontrar, na estrutura de um reticulado em dimensões altas, um vetor que esteja mais próximo a um vetor recebido do $\mathbb{R}^n$, ou, em outras palavras, devido à grande complexidade de se resolver os problemas difíceis em reticulados, a dizer o SVP e CVP.

Neste capítulo vamos abordar sobre dois dos principais criptossistemas usando chaves públicas desenvolvidos utilizando a estrutura de reticulados, o criptossistema GGH e NTRU, e também, assim como feito no Capítulo 2, apontar as vantagens e desvantagens na aplicação e segurança de tais criptossistemas e também mencionar algumas questões abertas nessa área de pesquisa. Foram usados como referências os textos [36], [54], [47] e [48].

4.1 O criptossistema GGH

Os problemas SVP e CVP, considerados difíceis em um reticulado de dimensão n quando n é consideravelmente grande, serviram de base para vários criptossistemas introduzidos desde meados dos anos 90. Os mais conhecidos deles são: o criptossistema de Ajtai-Dwork [3], o criptossistema GGH atribuído a Goldreich, Goldwasser and Halevi [28] e o criptossistema NTRU proposto por Hoffstein, Pipher e Silverman [35].

O sistema de Ajtai-Dwork é particularmente interessante, de acordo com [36], pois os autores mostraram que o criptosssitema desenvolvido é provavelmente seguro a não ser que o pior caso do problema em reticulados possa ser resolvido por um algoritmo em tempo polinomial. Em compensação, esse importante resultado teórico ainda nos diz que o tamanho das chaves é aproximado por $\mathcal{O}(n^4)$, o que implica em chaves enormes.

De modo informal, o sistema GGH se explica da seguinte forma: a chave privada de Alice

é uma base boa β_{boa} para um reticulado $\mathcal{L}$ e sua chave pública é uma base ruim β_{ruim} para o mesmo reticulado $\mathcal{L}$. A mensagem de Bob é um vetor binário m, que ele utiliza para formar uma combinação linear $\sum m_i v_i$ dos vetores $v_i \in \beta_{\text{ruim}}$. Bob então perturba essa soma adicionando à ela um pequeno vetor r aleatório. O resultado é um vetor w que difere de um vetor v do reticulado por um vetor r. Como Alice conhece uma base boa para $\mathcal{L}$, ela pode utilizar o algoritmo de Babai para encontrar v e expressá-lo em termos da base ruim β_{ruim} e recuperar m. Se um atacante, Tom, conhece somente a base β_{ruim}, ele é incapaz de resolver o problema CVP em $\mathcal{L}$ em um tempo viável.

Munidos de tais informações, podemos esquematizar um algoritmo que explique o funcionamento do criptossistema GGH, conforme apresentado no Algoritmo 4.1

Algoritmo 4.1 O sistema criptográfico de chave pública GGH

Escolha um reticulado $\mathcal{L}$.

Chave privada: $\{v_1, \ldots, v_n\}$ como sendo uma base boa e curta para o reticulado $\mathcal{L}$.

Chave pública: $\{w_1, \ldots, w_n\}$ como sendo uma base ruim e longa para o reticulado $\mathcal{L}$.

Encriptação de uma mensagem $m \in {\mathbb{F}_2}^n$: Escolha um vetor de perturbação r. A mensagem cifrada, por sua vez, é dada por: $e = m_1 w_1 + m_2 w_2 + \cdots + m_n w_n + r$. Note que a mensagem cifrada $e \notin \mathcal{L}$.

Decriptação: Encontre um vetor $u \in \mathcal{L}$ que é próximo a e. Se r for pequeno o suficiente, então $u = m_1 w_1 + m_2 w_2 + \cdots + m_n w_n$, com isso, basta resolver o CVP para e em $\mathcal{L}$ para recuperar m. A chave privada, que é a base boa para o reticulado $\mathcal{L}$, pode ser usada para encontrar u. Escreva, primeiramente, $e = \mu_1 v_1 + \mu_2 v_2 + \cdots + \mu_n v_n$, com $\mu_1, \mu_2, \ldots, \mu_n \in \mathbb{R}$. A aproximação $\mu_1, \mu_2, \ldots, \mu_n$ para o menor inteiro: $\lfloor \mu_1 \rceil v_1 + \cdots + \lfloor \mu_n \rceil v_n$ será igual a u.

Como pudemos notar, o criptossistema GGH é dos sistemas mais intuitivos baseados em reticulados e a sua segurança reside na dificuldade em resolver o CVP usando uma base altamente não ortogonal. O criptossistema GGH foi submetido a ataques criptoanalíticos [51] com parâmetros de segurança relativamente grandes e pode ser considerado, de certa forma, inseguro de um ponto de vista prático, de acordo com [47].

Uma outra forma de atacar o criptossistema GGH é tentar melhorar a base ruim, que é a chave pública, afim de tornar seus vetores menores e mais ortogonais, usando o algoritmo LLL, por exemplo. Em dimensão 2 esse problema pode ser facilmente resolvido [8], já em dimensões bem maiores esse problema é considerado difícil.

Além disso, vimos que a chave pública do sistema GGH é uma base de um reticulado $\mathcal{L}$, assim, seu tamanho é de aproximadamente $\mathcal{O}(n^2)$ bits, que é muito grande, tornando o uso desse criptossistema ineficiente em termos práticos.

4.2 O criptossistema NTRU

Acredita-se que o criptossistema NTRU seja um dos mais práticos e promissores algoritmos baseados em reticulado e sua patente pertence à empresa *NTRU Cryptosystem, Inc*, fundada por seus idealizadores Jeffrey Hoffstein, Jill Pipher e Joseph H. Silverman [35], juntamente com Daniel

Lieman. A segurança do NTRU está baseada na dificuldade em resolver o problema SVP em reticulados.

Criptossistemas baseados nas dificuldades sugeridas pela fatoração inteira ou no problema do logaritmo discreto são baseadas em teoria de grupos, pois seus problemas difíceis requerem somente uma operação. Para o RSA consideramos, por exemplo, o grupo das unidades módulo m para algum módulo m que pode ser primo ou composto e o grupo da multiplicação módulo m. Já para os esquemas baseados em curvas elípticas, o grupo em questão é o grupo dos pontos de uma curva elíptica módulo p e o grupo da adição de curvas elípticas.

Quando necessitamos de mais de uma operação, pensamos em anéis, já que um anel é uma estrutura algébrica munido das operações de soma e multiplicação, além da lei da distributividade. O sistema NTRU é baseado originalmente em anéis, mas pode ser descrito de maneira equivalente usando reticulados, que é o que faremos nessa seção, usando como referência os textos de [35], [36], [8] e [47].

A proposta original do sistema NTRU, por sua vez, foi desenvolvida usando a estrutura de anéis convolucionais de polinômios, ou seja, o anel R definido por:

$$R = \frac{\mathbb{Z}[x]}{(x^N - 1)},$$

onde os elementos desse anel são polinômios da forma:

$$a(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{N-1} x^{N-1}.$$

Definição 4.2.1. *A multiplicação no anel R é dada pela operação de convolução ou operação convolucional:*

$$\left(\sum_{i=0}^{N-1} a_i x^i\right) * \left(\sum_{j=0}^{N-1} b_j x^j\right) = \left(\sum_{k=0}^{N-1} c_k x^k\right),$$

onde $c_k = \sum_{i+j \equiv k (mod\ N)} a_i b_j = \sum_{i=0}^{k} a_i b_{k-i} + \sum_{i=k+1}^{N-1} a_i b_{N+k-i}$.

Notamos que esta multiplicação é diferente da multiplicação usual nos anéis de polinômios, que, por sua vez, seria dada por $\left(\sum_{i=0}^{N-1} a_i x^i\right)\left(\sum_{j=0}^{N-1} b_j x^j\right) = \sum_{k=0}^{2N-2}\left(\sum_{i+j=k} a_i b_j\right) x^k$.

A priori a multiplicação convolucional de $f * g$ requer em torno de N^2 operações, de acordo com [35]. Contudo, para as multiplicações comuns feitas no algoritmo NTRU que utilizam coeficientes pequenos, tal cálculo é feito de maneira consideravelmente rápida.

Um fato curioso é que, ao contrário do criptossistema GGH, o nome NTRU não deriva dos sobrenomes dos seus autores, mas, por sua vez, significa "Anel de polinômios truncados de grau N,"do inglês, "N^{th} *Degree Truncated Polynomial Ring.*"

O criptossistema NTRU depende de três parâmetros (N, p, q) e de quatro conjuntos $\mathcal{F}_f, \mathcal{F}_g, \mathcal{F}_\phi$ e $\mathcal{F}_m$. Tais parâmetros devem satisfazer as seguintes condições:

- Os valores de p e q não precisam ser necessariamente números primos, mas devem satisfazer $mdc(p, q) = 1$. Além disso, q deve ser sempre "muito maior"do que p.

- $\mathcal{F}_f$ e $\mathcal{F}_g$ serão conjuntos com polinômios de grau até $N-1$ de onde serão extraídas as chaves públicas e privadas do sistema.
- $\mathcal{F}_m$ é o espaço de onde serão extraídas as mensagens. Podemos definir $\mathcal{F}_m$ como:

$$\mathcal{F}_m = \{m(x) \in R : \text{todos os coeficientes de m moram em}[-\frac{p-1}{2}, \frac{p-1}{2}]\}.$$

  Tais exigências para os coeficientes de $m(x)$ estão bem detalhadas em [35]. *(Por exemplo, se $p = 3$, então os polinômios que pertencem à $\mathcal{F}_m$ possuem grau até $N-1$ e seus coeficientes estão em $\{-1, 0, 1\}$.)*
- $\mathcal{F}_\phi$ é o conjunto de polinômios de onde será selecionado um valor sigiloso que será utilizado durante a encriptação.

Assim, o criptossistema NTRU sob o anel convolucional de polinômios opera da seguinte maneira:

GERAÇÃO DE CHAVES: Bob escolhe dois polinômios arbitrários $f \in \mathcal{F}_f$ e $g \in \mathcal{F}_g$. O polinômio f deve satisfazer ainda a propriedade adicional de possuir inversa módulo p e módulo q, que serão denotadas, respectivamente, por f_p^{-1} e f_q^{-1}. Para escolhas corretas de parâmetros, tais inversas sempre podem ser encontradas para grande parte das escolhas de f. Então, temos:

$$f_q^{-1} * f \equiv 1(mod\ q) \qquad f_p^{-1} * f \equiv 1(mod\ p).$$

Bob, por sua vez, calcula:

$$h \equiv f_q^{-1} * g(mod\ q).$$

Dessa forma, definimos:
Chave pública: h.
Chave privada: f, mas na prática é importante guardar também o valor de f_p^{-1}.

ENCRIPTAÇÃO: Suponha que Alice deseja enviar uma mensagem para Bob. Ela inicia escolhendo sua mensagem $m \in \mathcal{F}_m$. Em seguida, ela escolhe um polinômio aleatório $\phi \in \mathcal{F}_\phi$ e usa a chave pública h para calcular

$$c \equiv p\phi * h + m(mod\ q).$$

E c é a mensagem encriptada que será enviada à Bob.

DECRIPTAÇÃO: Suponha que Bob recebeu a mensagem c de Alice e deseja decriptá-la usando sua chave privada f. Para fazer isso de maneira eficiente é importante que Bob já tenha em suas mãos o valor de f_p^{-1}. Assim, para decriptar c, Bob calcula inicialmente:

$$a \equiv f * c(mod\ q),$$

onde ele toma os coeficientes do polinômio a no intervalo de $-q/2$ a $q/2$. Agora, considerando a como um polinômio com coeficientes inteiros, Bob recupera a mensagem m calculando:

$$f_p^{-1} * a(mod\ p).$$

Nos perguntamos então, a decriptação conforme descrita acima de fato funciona?

Note que o polinômio a que Bob calculou satisfaz

$$\begin{aligned} a \equiv f * c &\equiv f * p\phi * h + f * m(mod\ q) \\ &= f * p\phi * f_q^{-1} * g + f * m(mod\ q) \\ &= p\phi * g + f * m(mod\ q). \end{aligned}$$

Para escolhas apropriadas de parâmetros, é possível garantir que todos os coeficientes de a estejam entre $-q/2$ e $q/2$ e isso garante que não ocorrerá nenhuma alteração quando for feita a redução módulo q. Isso significa que quando Bob reduz os coeficientes de $f*c$ módulo q no intervalo de $-q/2$ a $q/2$, ele recupera exatamente o polinômio

$$a = p\phi * g + f * m.$$

Fazendo redução módulo p, Bob obtém $f * m(mod\ p)$ e multiplicando por f_p^{-1}, ele consegue finalmente recuperar $m(mod\ p)$.

Tratando-se das escolhas de parâmetros apropriadas que tanto foram comentadas nesse texto, não entraremos em detalhes, mas é possível encontrar mais informações em [35].

A segurança do NTRU está, por sua vez, baseada no seguinte problema de fatoração:

> Dado um polinômio $h \in R$, encontre dois polinômios f e $g \in R$ tais que $h = f_q^{-1} * g(mod\ q)$, onde f_q^{-1} é a inversa de f em $\dfrac{\mathbb{Z}_q[x]}{(x^N - 1)}$.

O primeiro artigo dedicado ao estudo da segurança do criptossistema NTRU, que foi escrito por Copersmith e Shamir [18] em 1997, propunha um ataque baseado em reticulados. Para isso, foi definido inicialmente um reticulado baseado nos parâmetros exigidos pelo próprio criptossistemas NTRU, que são N, q e h. Mostrou-se então que recuperar a chave privada f a partir da chave pública h se resumia a resolver um problema do vetor mais curto (SVP) nesse reticulado.

Tal reticulado, chamado de *reticulado convolucional modular* ou *reticulado NTRU*, é definido como:

Definição 4.2.2. *Considere que todo polinômio $p \in R$ tal que $deg(p) \leq N-1$ com a forma $p(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{N-1}x^{(N-1)}$ possa ser identificado com o vetor $(a_0, a_1, a_2, \ldots, a_{N-1}) \in \mathbb{Z}^N$. Seja $h = (h_0, h_1, h_2, \ldots, h_{N-1})$ a chave pública do criptossitema NTRU. O* ***reticulado NTRU*** *de dimensão $2N$ é o reticulado gerado pelas linhas da matriz com a seguinte forma:*

$$G_{\mathcal{L}} = \left(\begin{array}{c|c} lI_N & H \\ 0 & qI_N \end{array}\right) = \left(\begin{array}{cccc|cccc} l & 0 & \ldots & 0 & h_0 & h_1 & \ldots & h_{N-1} \\ 0 & l & \ldots & 0 & h_{N-1} & h_2 & \ldots & h_{N-2} \\ \vdots & \ldots & \ddots & \vdots & \ldots & \ldots & \ddots & \ldots \\ 0 & 0 & \ldots & l & h_1 & h_2 & \ldots & h_0 \\ 0 & 0 & \ldots & 0 & q & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 & 0 & q & \ldots & 0 \\ \vdots & \ldots & \ddots & \vdots & \ldots & \ldots & \ddots & \ldots \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & q \end{array}\right).$$

Observe que a matriz H apresentada na matriz é a chamada *matriz circulante*, que é construída a partir de desvios cíclicos, ou seja, cada uma de suas linhas é um desvio cíclico da linha anterior, iniciando com $h = (h_0, h_1, h_2, \ldots, h_{N-1})$.

Como $h = f_q^{-1} * g(mod\ q)$, então $f * qu = g$ para algum $u \in R$, usando uma extensão do algoritmo de Euclides. Além disso,

$$(f, -u) * G_{\mathcal{L}} = (f, -u) * \left(\begin{array}{c|c} lI_N & H \\ 0 & qI_N \end{array} \right) = (lf, g).$$

Com isso, notamos que o vetor (lf, g) é um vetor curto no reticulado NTRU $\mathcal{L}$, que possui grandes chances de ser o vetor mais curto de $\mathcal{L}$ e com isso, o atacante usa um algoritmo de redução de bases para encontrar (f, g) a partir de $\mathcal{L}$. Dessa forma, o atacante consegue, ao resolver o SVP, recuperar a chave privada f.

O algoritmo LLL, que é utilizado para resolver de forma aproximada o problema do vetor mais curto e também o problema do vetor mais próximo, apresenta bons resultados para a resolução do problema apresentado acima para dimensões menores do que 100. Pesquisadores como Hoffstein ([34]) vem estudando maneiras de mensurar até qual ponto o algoritmo LLL se apresenta eficaz na resolução desse problema.

Observamos ainda que para maximizar a probabilidade de quebrar o sistema NTRU usando o algoritmo LLL, o atacante deve escolher o valor de l como sendo $\frac{||g||}{||f||}$, de acordo com estudos apresentados em [54].

Resumimos a seguir algumas vantagens e desvantagens do uso e implementação do sistema criptográfico NTRU que tem sido apontadas por diversos autores.

VANTAGENS

- [64] O tamanho da chave pública utilizada no criptossistema NTRU é muito menor do que a do GGH, usando aproximadamente $\frac{1}{2}\log_2(n) \simeq \mathcal{O}(n)$ bits.
- [47] A segurança do sistema NTRU ainda não pôde ser comprovada, porém, sua credibilidade está baseada no fato de que todos os ataques conhecidos até o momento não foram suficientes para quebrá-lo, já que ele depende da dificuldade de resolução do SVP, que acredita-se que seja um problema $NP-$completo.
- [37] É possível implementar um sistema de assinatura digital usando NTRU, que é baseado na resolução do problema APPRCVP.
- [44] As operações do criptossistema NTRU requerem poucas operações de memória, o que torna o sistema ideal para a implementação em dispositivos restritos como *smart cards*.

DESVANTAGENS

- [47] Até o momento, nenhuma prova que comprove a segurança é conhecida.

- [65] O NTRU é prático e considerado seguro para reticulados com dimensão variando de 500 a 1000, já que as técnicas do algoritmo LLL não são capazes de lidar com reticulados desse tamanho. Isso é visto como desvantagem, já que ele não parece eficiente para reticulados com dimensão menor.
- [58] Mesmo apresentando uma boa velocidade de decriptação, o RSA ainda é mais veloz nesse aspecto. Esses e outros parâmetros podem ser comparados conforme na Tabela 4.1, que é uma tabela comparativa com características teóricas de operação dos sistemas NTRU, RSA, McEliece e GGH. Em todos os casos o valor de n representa um parâmetro natural de comprimento.

Tabela 4.1: Comparando sistemas criptográficos de chaves públicas
Fonte: Ranjan, Baghel e Kumar, 2012 [58]

	NTRU	RSA	McEliece	GGH
Velocidade de encriptação	n^2	n^2	n^2	n^2
Velocidade de decriptação	n^2	n^3	n^2	n^2
Tamanho das chaves	n	n	n^2	n^2

Observamos também que o NTRU apresenta melhores resultados em todas as comparações com os criptossistemas de McEliece e GGH, o que reforça a percepção de que ele seja um dos algoritmos mais eficientes e promissores que utilizam chaves públicas dentro da criptografia pós-quântica.

Contribuições e perspectivas de estudo e pesquisa

A contribuição desta dissertação de mestrado é a elaboração de um texto conciso em língua portuguesa abordando conceitos matemáticos associados à teoria de códigos corretores de erros e reticulados com foco em tópicos de uma subárea atual de pesquisa que é conhecida como criptografia pós-quântica.

Considerando a relevância desse tema e a perspectiva de continuar os estudos acerca do mesmo, surgiram questionamentos e tópicos que provavelmente serão um ponto de partida para futuras pesquisas. Podemos citar aqui, por exemplo, a Construção A, que foi estudada com o propósito de ser utilizada para pesquisar relações entre a criptografia baseada em códigos e a criptografia baseada em reticulados.

Listamos a seguir outros tópicos que nos fornecem objetos potenciais de pesquisa, dando continuidade ao que aqui foi iniciado:

Construções B e D

As construções B e D, assim como a construção A, mostram-nos maneiras de obter reticulados a partir de certos códigos.

Vamos definir, brevemente, tais construções:

Definição 4.2.3. *[40] Sejam $q = 2^r b, r \geq 0, b \in \mathbb{N}$ ímpar e $\mathcal{C} \subset \mathbb{Z}_q^n$ um código linear $q-$ário tal que:*

$$2^r \text{ divide } \sum_{i=1}^{n} c_i \text{ para todo } \bar{c} = (\bar{c_1}, \ldots, \bar{c_n}) \in \mathcal{C}.$$

*Definimos a **Construção B** estendida para o código $\mathcal{C}$ como*

$$\mathcal{L}_B(\mathcal{C}) = \{z = c + qw : w \in \mathbb{Z}^n, \bar{c} \in \mathcal{C} \text{ e } 2^{r+1} \text{ divide } \sum_{i=1}^{n} z_i\}.$$

Definição 4.2.4. *[17] Sejam $\mathcal{C}_0 \supseteq \mathcal{C}_1 \supseteq \cdots \supseteq \mathcal{C}_a$ uma família de $a+1$ códigos lineares onde $\mathcal{C}_l[n, k_l, d^l]$ para $1 \leq l \leq a$ e $\mathcal{C}_0[n, n, 1]$ é um código linear trivial em $\mathbb{F}_2^n$. Definimos a **Construção D** de $\mathcal{L} \in \mathbb{R}^n$ como todos os vetores na forma:*

$$z + \sum_{l=1}^{a} \sum_{j=1}^{kl} \beta_j^{(l)} \frac{c_j}{2^{l+1}},$$

$com\ z \in 2\mathbb{Z}^n\ \ e\ \beta_j^{(l)} = 0\ \ ou\ 1.$

Uma linha possível de investigação que consideramos é a de explorar conexões entre os criptossistemas baseados em códigos corretores de erros e os baseados em reticulados usando tais contruções.

Reticulado NTRU

Ao definirmos o reticulado NTRU por meio de sua matriz geradora e compará-lo com a matriz geradora de um reticulado obtido a partir da Construção A, notamos uma grande semelhança entre tais estruturas. Surgem então algumas questões interessantes de serem investigadas, tais como explorar o possível código gerador do reticulado NTRU para obter melhorias no ataque ao criptossistema NTRU e aprimorar a busca ao vetor mais curto em tal reticulado.

Possíveis aplicações de códigos e reticulados para segurança em redes de armazenamento

Dentro de nossas perspectivas incluímos também o estudo e pesquisa de possíveis aplicações de teoria de códigos e reticulados na área de segurança para "computação em nuvem", ou seja, uma computação baseada em internet, através da qual, os recursos compartilhados, informação e software são fornecidos para o uso em computadores ou outros dispositivos.

Um estudo recente [27] aborda diversas questões associadas à segurança da computação em nuvem, como por exemplo, privacidade e confidencialidade, armazenamento, cópia e recuperação dos dados e também disponibilidade de dados, e nesse contexto, o criptossistema NTRU é discutido como uma possibilidade de aplicação para garantir tal segurança, já que ele permite manipular dados já encriptados sem comprometer o que foi armazenado.

Aspectos de implementação

Tendo em vista a abordagem teórica que fizemos, consideramos fundamental nas pesquisas futuras compreender e explorar aspectos práticos de implementação dos algoritmos computacionais aqui expostos, como os criptossistemas de McEliece, Niederreiter, GGH e NTRU. A importância dessas implementações reside, principalmente, no fato de visualizar as vantagens e desvantagens de cada sistema, comparando com a teoria que foi apresentada nesta dissertação e também na possível proposta de alternativas para tais sistemas.

Referências

[1] M. Ajtai. "Generating Hard Instances of Lattice Problems (Extended Abstract)". Em: *In Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*. ACM, 1996, pp. 99–108.

[2] M. Ajtai. "The Shortest Vector Problem in L2 is NP-hard for Randomized Reductions (Extended Abstract)". Em: *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*. STOC '98. ACM, 1998, pp. 10–19.

[3] M. Ajtai e C. Dwork. "A public-key cryptosystem with worst-case/average-case equivalence". Em: *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*. STOC '97. ACM, 1997, pp. 284–293.

[4] D. Augot, E. Orsini e E. Betti. "An introduction to linear and cyclic codes". Em: *Journal of Symbolic Computation* (2009), pp. 47–68.

[5] L. Babai. "On Lovász' Lattice Reduction and the Nearest Lattice Point Problem (Shortened Version)". Em: *Proceedings of the 2Nd Symposium of Theoretical Aspects of Computer Science*. STACS '85. London: Springer-Verlag, 1985, pp. 13–20.

[6] M. Baldi. *QC-LDPC Code-Based Cryptography*. Springer, 2014.

[7] P. Barreto. *Introduction to Code-Based Cryptography*. Presented in: SP Coding School, Brazil. 2015.

[8] P. Barreto, F.P. Biasi, R. Dahab et al. *Introdução à criptografia pós-quântica*. 2013. URL: `http://dainf.ct.utfpr.edu.br/~maziero/lib/exe/fetch.php/ceseg:2013-sbseg-mc2.pdf`.

[9] A.F. Beardon. *The Geometry of Discrete Groups*. Graduate texts in mathematics. New York: Springer, 1995.

[10] E. Berlekamp, R. McEliece e H. Van Tilborg. "On the inherent intractability of certain coding problems". Em: *IEEE Transactions on Information Theory, v.24, n.3* (1978), pp. 384–386.

[11] D.J. Bernstein, J. Buchmann e E.Dahmen. *Post-Quantum Cryptography*. Dordrecht: Springer, 2009.

[12] A. Betten, M. Braun, H. Fripertinger et al. *Error-correcting linear codes : classification by isometry and applications*. Algorithms and computation in mathematics. Berlin: Springer, 2006.

[13] R.C. Bose e D.K. Ray-Chaudhuri. "On a class of error correcting binary group codes". Em: *Information and Control, v.3, n.1* (1960), pp. 68–79.

[14] A. Campello. *Notas de aula: reticulados, teoremas de Minkowski revisitados e teoremas de somas de quadrados*. 2014.

[15] A. C. de A. Campello, G. C. Jorge e S. I. R. Costa. "Decoding q-ary lattices in the Lee metric". Em: *CoRR* (2011).

[16] H. Chen. *Coding theory: cyclic codes*. 2011.

[17] J.H. Conway e N.J.A. Sloane. *Sphere Packings, Lattices and Groups*. 3ª ed. New York: Springer Verlag, 1998.

[18] D. Coppersmith e A. Shamir. "Lattice Attacks on NTRU". Em: *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Germany*. 1997, pp. 52–61.

[19] T. H. Cormen et al. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2001.

[20] S. Costa e A. Campello. *Lattice Tools in Comunication*. Presented in: International Telecommunications Simposium, Brazil. 2014.

[21] N. Courtois, M. Finiaz e N. Sendrier. "How to achieve a McEliece-based Digital Signature Scheme". Em: *Advances in cryptology—ASIACRYPT* (2001), pp. 157–174.

[22] H. Domingues e G. Iezzi. *Álgebra Moderna*. 4ª ed. São Paulo: Atual, 2003.

[23] P. van Emde Boas. *Another NP-Complete Problem and the Complexity of Computing Short Vectors in a Lattice*. Rel. téc. Mathematische Instituut, Universiry of Amsterdam, 1981.

[24] M. Firer. *Notas de aula: Códigos Corretores de Erros*. 2007.

[25] E. M. Gabidulin. "Public-key cryptosystems based on linear codes over large alphabets: efficiency and weakness". Em: *4^{th} IMA conference on cryptography and coding, the Institute of Mathematics and its Applications*. Ed. por P. G. Farrell. 1993, pp. 17–31.

[26] N. Gama, P. Q. Nguyen e O. Regev. "Lattice Enumeration using Extreme Pruning". Em: *Advances in Cryptology - Proceedings of EUROCRYPT '10, v.6110*. LNCS. Springer, 2010.

[27] A. K. Gill e C. Singh. "Security of N-Tier Architecture using NTRU". Em: *International Journal of Advanced Research in Computer Science and Software Engineering, v.4, n.7* (2014), pp. 1018–1022.

[28] O. Goldreich, S. Goldwasser e S. Halevi. "Public-Key Cryptosystems from Lattice Reduction Problems". Em: *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*. CRYPTO '97. London: Springer-Verlag, 1997, pp. 112–131.

[29] J. I. Hall. *Notes on Coding Theory*. `http://www.mth.msu.edu/~jhall/classes/codenotes/coding-notes.html`. 2012.

[30] D.C. Hankerson et al. *Coding Theory and Cryptography: The Essentials, Second Edition*. Chapman & Hall/CRC Pure and Applied Mathematics. New York: Taylor & Francis, 2000.

[31] A. Hefez e M. L. T. Villela. *Códigos Corretores de Erros*. 1ª ed. Rio de Janeiro: IMPA, 2002.

[32] A. Hocquenghem. “Codes correcteurs d’erreurs”. Em: *Chiffres, v.2* (1959), pp. 147–158.

[33] K. Hoffman e R.A. Kunze. *Linear algebra*. Prentice-Hall mathematics series. Englewood Cliffs: Prentice-Hall, 1971.

[34] J. Hoffstein. *A history of the development of NTRU*. Presented in: EUROCRYPT 2014, Copenhagen. 2014. URL: `http://http://ec14.compute.dtu.dk/talks/6.pdf`.

[35] J. Hoffstein, J. Pipher e J. H. Silverman. “NTRU: A Ring-Based Public Key Cryptosystem”. Em: *Lecture Notes in Computer Science*. Springer-Verlag, 1998, pp. 267–288.

[36] J. Hoffstein, J. Pipher e J.H. Silverman. *An Introduction to Mathematical Cryptography*. 1ª ed. New York: Springer, 2008.

[37] J. Hoffstein, N. Howgrave-graham, J. Pipher et al. “NTRUSign: Digital Signatures Using the NTRU Lattice”. Em: *Proceedings of the 2003 RSA Conference on The Cryptographers’ Track*. CT-RSA’03. Berlin: Springer Verlag, 2003, pp. 122–140.

[38] W.C. Huffman e V. Pless. *Fundamentals of Error-correcting Codes*. New York: Cambridge University Press, 2003.

[39] E. Jochemsz. “Goppa Codes and McEliece Cryptosystem”. Tese de doutorado. Vrije Univesiteit Amsterdam, 2002.

[40] G. C. Jorge. “Reticulados q-ários e algébricos”. Tese de doutorado. Universidade Estadual de Campinas, 2012.

[41] J. Katz e Y. Lindell. *Introduction to Modern Cryptography*. Chapman & Hall/Crc Cryptography and Network Security Series. Chapman & Hall/CRC, 2007.

[42] C.C. Lavor, M.M.S. Alves, R.M. de Siqueira et al. *Uma Introdução à Teoria de Códigos*. 2012.

[43] S. Lin e D. J. Costello. *Error Control Coding: Fundamentals and Applications*. New Jesey: Prentice Hall, 1983.

[44] J. Lopez e J. Zhou. “Wireless Sensor Network Security”. Em: Cryptology and Information Security Series. IOS Press, 2007.

[45] R.J. McEliece. “A Pubic-Key Criptosystem Based on Algebraic Coding Theory”. Em: *DSN Progress Report* (1978), pp. 42–44.

[46] A. J. Menezes, S. A. Vanstone e P. C. Van Oorschot. *Handbook of Applied Cryptography*. 1ª ed. Boca Raton: CRC Press, 1996.

[47] D. Micciancio e O. Regev. “Post-Quantum Cryptography”. Em: ed. por D.J. Bernstein, J. Buchmann e E.Dahmen. Springer, 2009. Cap. Lattice-based Cryptography, pp. 147–191.

[48] A. Mihata e E. Simion. “New Trends in Lattice-Based Cryptography”. Em: *Acta Universitatis Apulensis* 29 (2012), pp. 53–64.

[49] H. Minkowski. *Geometrie der Zahlen*. Teubner: Leipzig, 1896.

[50] R. Misoczki e P. S.L.M. Barreto. *Criptografia Pós-Quântica com Códigos Corretores de Erros*. VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. 2008.

[51] P. Q. Nguyen e Jacques S. "Cryptanalysis of the Ajtai-Dwork Cryptosystem." Em: *CRYPTO, v. 1462*. Ed. por Hugo Krawczyk. Lecture Notes in Computer Science. Springer, 1998, pp. 223–242.

[52] P. Q. Nguyen e B. Valle. *The LLL Algorithm: Survey and Applications*. 1st. Berlin: Springer, 2009.

[53] H. Niederreiter. "Knapsack-type cryptosystems and algebraic coding theory". Em: *Problems Control Information Theory* (1986), pp. 159–166.

[54] A. Nitaj. "Cryptanalysis of NTRU with two Public Keys". Em: *I. J. Network Security, v.16, n.2* (2014), pp. 112–117.

[55] R. Overbeck e N. Sendrier. "Post-Quantum Cryptography". Em: ed. por D.J. Bernstein, J. Buchmann e E.Dahmen. Springer, 2009. Cap. Code-based Cryptography, pp. 95–145.

[56] C. Peikert. *Lattices in Cryptography*. `http://www.cc.gatech.edu/~cpeikert/lic13/`. 2013.

[57] E. Persichetti. "Compact McEliece keys based on quasi-dyadic Srivastava codes". Em: *J. Mathematical Cryptology, v.6, n.2* (2012), pp. 149–169.

[58] A. S. Baghel R. Ranjan e S. Kumar. "Improvement of NTRU Cryptosystem". Em: *International Journal of Advanced Research in Computer Science and Software Engineering, v.2, n.9* (2012), pp. 79–84.

[59] I. S. Reed e G. Solomon. "Polynomial codes over certain finite fields". Em: *Journal of the Society for Industrial and Applied Mathematics, v.8* (1960), pp. 300–304.

[60] R.L. Rivest, A. Shamir e L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Em: *Communications of the ACM, v.21* (1978), pp. 120–126.

[61] V.C. da Rocha. *Elements of Algebraic Coding Systems*. New York: Momentum Press, 2014.

[62] M. Rückert e M. Schneider. *Estimating the Security of Lattice-based Cryptosystems*. Cryptology ePrint Archive, Report 2010/137. 2010.

[63] V.M. Sidelnikov e O. Shestakov. "On insecurity of cryptosystems based on generalized Reed-Solomon codes". Em: *Discrete Mathematics and Applications* (2009), pp. 439–444.

[64] J. H. Silverman. *An introduction to the theory of lattices and applications to cryptography*. 2006.

[65] J. H. Silverman. "Lattices, Cryptography, and the NTRU Public Key Cryptosystem". Em: *Unusual Applications of Number Theory: DIMACS Workshop*. DIMACS Series (2004).

[66] I. Stewart. *Galois Theory*. 3ª ed. Chapman Hall/CRC Mathematics Series. Boca Raton: Taylor & Francis, 2003.

[67] I. Stewart e D. Tall. *Algebraic Number Theory and Fermat's Last Theorem: Third Edition*. Ak Peters Series. Taylor & Francis, 2001.

[68] Song Y. Yan. *Primality Testing and Integer Factorization in Public-Key Cryptography*. 2ª ed. Springer, 2008.

[69] R. Zamir. *Lattice Coding of Signals and Networks: A Structured Coding Approach to Quantization, Modulation and Multi-user Information Theory*. Cambridge University Press, 2014.