RAFAEL MISOCZKI

**UMA FAMÍLIA DE CÓDIGOS CORRETORES DE
ERRO PARA CRIPTOSSISTEMAS EFICIENTES
BASEADOS EM DECODIFICAÇÃO DE SÍNDROMES**

Dissertação apresentada à Escola Politécnica
da Universidade de São Paulo para obtenção
do Título de Mestre em Ciências – Engenharia
Elétrica.

São Paulo
2010

RAFAEL MISOCZKI

**UMA FAMÍLIA DE CÓDIGOS CORRETORES DE
ERRO PARA CRIPTOSSISTEMAS EFICIENTES
BASEADOS EM DECODIFICAÇÃO DE SÍNDROMES**

Dissertação apresentada à Escola Politécnica
da Universidade de São Paulo para obtenção
do Título de Mestre em Ciências – Engenharia
Elétrica.

Área de Concentração:

Sistemas Digitais

Orientador:

Prof. Dr. Paulo S. L. M. Barreto

São Paulo
2010

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 8 de outubro de 2010.

Assinatura do autor

Assinatura do orientador

FICHA CATALOGRÁFICA

Misoczki, Rafael

Uma família de códigos corretores de erro para criptossistemas eficientes baseados em decodificação de síndromes/ R. Misoczki. – ed.rev. – São Paulo, 2010.

98 p.

Dissertação (Mestrado) — Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais (PCS).

1. Criptologia 2. Segurança de redes 3. Algoritmos I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais (PCS). II. t.

AGRADECIMENTOS

Primeiramente, agradeço à minha mãe, Rozires Luzia Stella, a qual sem seu apoio não seria possível atingir este e tantos outros objetivos, por toda a garra e dedicação empenhadas na criação de seus filhos, me servindo sempre como exemplo de honestidade, superação e sucesso. Também agradeço muito à minha família que, de inúmeras formas, sempre me incentivou na conquista de minhas aspirações.

Ao meu amigo e orientador, Prof. Dr. Paulo S. L. M. Barreto, agradeço por todo o conhecimento, incentivo e, principalmente, dedicação à tarefa de ensinar. Esta universidade e este país, em geral, seriam ainda mais capacitados se outros professores como este estivessem à disposição dos nossos estudantes.

Aos amigos, agradeço pela paciência e motivação durante os momentos difíceis. Também os agradeço por todas as comemorações, nos momentos não tão difíceis assim. Aos demais professores e colegas de mestrado, agradeço por todo o acompanhamento nessa caminhada difícil mas que certamente trará frutos.

RESUMO

A criptografia é uma ciência que tem especial destaque no mundo moderno, evidenciando a necessidade em pesquisar algoritmos e técnicas para seu aperfeiçoamento. Apesar das soluções atualmente empregadas serem baseadas em problemas suficientemente seguros, se comparados ao poderio computacional necessário para resolvê-los, seu emprego futuro é questionado. Pesquisas demonstram potencial vulnerabilidade destes sistemas, caso tenhamos à disposição computadores quânticos sendo utilizados para fraudá-los. Alternativas têm sido estudadas e o criptossistema proposto por Robert J. McEliece se mostra como uma das mais interessantes, visto sua versão convencional, baseada em códigos de Goppa, ter a segurança até o momento inafetada, inclusive se levado em conta a disponibilidade de tecnologia quântica. Entretanto, tal solução sofre com o tamanho de suas grandes chaves criptográficas, representando um entrave para sua aplicação na prática.

Objetivando minimizar esta deficiência, neste trabalho, propomos a classe de códigos de Goppa *quase-diádicos*, que admitem uma matriz de paridade ou matriz geradora com representação muito compacta, produzindo chaves do tipo McEliece que são até um fator $t = \tilde{O}(n)$ menores que as chaves genéricas, em notação que omite fatores logarítmicos, produzidas a partir de códigos de Goppa de comprimento n , para a correção de t erros em característica 2. No caso binário, essa família também mantém a capacidade de corrigir o número total projetado de erros em vez de apenas a metade; esta propriedade falta a todas as tentativas anteriores de construção de códigos compactos para fins de criptografia, incluindo (BERGER et al., 2009). Além disso, a complexidade de todas as operações criptográficas típicas torna-se $\tilde{O}(n)$.

Esta proposta foi submetida e selecionada para publicação e apresentação no *XVI Workshop Selected Areas in Cryptography* (SAC'2009), ocorrido entre 13 e 14 de Agosto de 2009, em Calgary, Alberta, Canadá, de referência bibliográfica (MISOCZKI; BARRETO, 2009). Este evento foi realizado em cooperação com a *International Association for Cryptologic Research* (IACR) e teve seus artigos publicados pela Springer em *Lecture Notes in Computer Science*, volume 5867.

ABSTRACT

Cryptography is a science that is especially prominent in the modern world, highlighting the need to find algorithms and techniques for its improvement. Despite the fact that solutions currently used are based on problems that are sufficiently secure, if compared to the computing power needed to solve them, their future use is questioned. Researches demonstrated the potential vulnerability of these systems, if quantum computers are used to defraud them. Alternatives have been studied and the cryptosystem proposed by Robert J. McEliece has been considered as one of the most interesting, since its conventional version, based on Goppa codes, has the security so far unaffected, even if taken into account the availability of quantum technology. However, this solution suffered with the large size of their cryptographic keys, an obstacle to its implementation in practice.

Aiming to minimize this shortcoming, in this paper, we propose the class of *quasi-dyadic* Goppa codes, which admits a generator or parity-check matrix with very compact representation, producing McEliece keys that are up to a factor $t = \tilde{O}(n)$ lower than the generic keys, in notation which omits logarithmic factors, produced from Goppa codes of length n , to correct t errors in characteristic 2. In the binary case, this family also retains the ability to correct the projected total number of errors, rather than just half; this property lacks in all previous attempts to build compact code for encryption, including (BERGER et al., 2009). Moreover, the complexity of all typical cryptographic operations becomes $\tilde{O}(n)$.

This proposal was submitted and selected for publication and presentation in the *XVI Workshop Selected Areas in Cryptography* (SAC'2009), occurred in August, 13th and 14th, 2009, in Calgary, Alberta, Canada, with bibliographic reference (MISOCZKI; BARRETO, 2009). This event was held in cooperation with the *International Association for Cryptologic Research* (IACR) and had his papers published by *Springer* in *Lecture Notes in Computer Science*, volume 5867.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

1	Introdução	11
1.1	Justificativa	12
1.2	Objetivos	13
1.3	Metodologia	13
1.4	Contribuições originais	14
1.5	Organização do documento	15
2	A criptografia convencional	16
2.1	O que é criptografia	16
2.2	Histórico da criptografia	18
2.3	Esquemas criptográficos	20
2.3.1	Criptografia simétrica	22
2.3.2	Criptografia assimétrica	22
2.4	Criptossistemas convencionais	24
2.4.1	RSA	26
2.4.1.1	Problema da fatoração de números inteiros em primos	26

2.4.1.2	Algoritmos	27
2.4.2	Criptografia baseada em curvas elípticas	28
2.4.2.1	Problema da determinação do logaritmo discreto . .	29
2.4.2.2	Problema da determinação do logaritmo discreto em curvas elípticas	29
2.4.2.3	Algoritmos	32
2.5	Algoritmo quântico de Shor	33
2.6	Resumo	35
3	Criptografia baseada em códigos corretores de erros	36
3.1	Teoria da codificação	36
3.2	Códigos lineares	37
3.3	Códigos GRS	44
3.4	Códigos alternantes	46
3.4.1	Algoritmo estendido de Euclides	47
3.4.2	Método de decodificação alternante	48
3.5	Códigos de Goppa	51
3.5.1	Método de decodificação de Patterson	54
3.6	Criptossistema McEliece	55
3.6.1	Algoritmos	56
3.6.1.1	Geração de chaves	56
3.6.1.2	Cifração	57

3.6.1.3	Decifração	57
3.6.2	Segurança	57
3.6.2.1	Segurança semântica	58
3.6.3	Eficiência algorítmica	60
3.6.4	Tamanho das chaves criptográficas	60
3.7	Resumo	62
4	Códigos de Goppa quase-diádicos	64
4.1	Conceitos algébricos	64
4.2	Códigos de Goppa em forma de Cauchy e em forma diádica	69
4.2.1	Construção de um código de Goppa binário em forma diádica	70
4.2.2	Construção de subcódigos quase-diádicos permutados binários	75
4.2.3	Um exemplo em miniatura	80
4.3	Complexidade computacional de decodificar códigos quase-diádicos .	82
4.3.1	Ataques baseados na solução de sistema de equações multiva- riadas quadráticas	85
4.4	Considerações sobre eficiência	87
4.4.1	Parâmetros sugeridos	88
4.5	Implementação	89
4.5.1	Aplicações	91
4.6	Resumo	91
5	Conclusões	93

LISTA DE FIGURAS

1	Curva Elíptica E e os pontos P , Q , R' e R	30
2	Canal de Comunicação.	37
3	Fragilidade semântica do criptossistema McEliece puro.	59
4	Estrutura diádica.	65
5	Matriz diádica.	66
6	Matriz quase-diádica.	68

LISTA DE TABELAS

1	Classificação dos principais sistemas criptográficos	25
2	Tabela de correção de erros do método máxima probabilidade para o exemplo	41
3	Tabela de esforço para a quebra clássica e quântica	58
4	Complexidade de operação relativa ao comprimento n do código. . . .	88
5	Parâmetros sugeridos para um código de Goppa $[n, k, 2t + 1]$ binário. .	89
6	Benchmarks para geração de chaves. Tempos em ms.	89
7	Benchmarks para cifração. Tempos em ms.	90
8	Benchmarks para decifração. Tempos em ms.	90

1 INTRODUÇÃO

A criptografia é um ramo da ciência da computação que tem sensível importância no nosso cotidiano. Como alguns de seus objetivos, podemos citar o estudo de técnicas que permitam a transmissão de informação de maneira sigilosa, validação de autenticidade de um documento, resumo de informações, entre outras. Inúmeras aplicações presentes no mundo moderno estão embasadas nestes conceitos, demandando contínuo aprimoramento dos mesmos, visto a evolução constante das estratégias destinadas a fraudá-los.

Atualmente, existem diversas soluções para a transmissão de informação de maneira sigilosa, tendo como principais expoentes o sistema criptográfico RSA (RIVEST; SHAMIR; ADLEMAN, 1978) e os baseados em curvas elípticas, ideia originalmente proposta em (MILLER, 1985) e (KOBLOITZ, 1987). Tais sistemas oferecem, para parâmetros usualmente utilizados, recursos de processamento e memória bastante razoáveis, estando presentes na maioria das aplicações práticas.

Com relação à segurança destas propostas, podemos considerá-los atualmente de difícil fraude, uma vez que estão baseados em problemas matemáticos de difícil resolução pelos computadores atuais, a saber, fatoração de números inteiros em primos, no caso do RSA, e de determinação do logaritmo discreto, como Diffie-Hellman, DSA e ElGamal tradicionais ou com curvas elípticas.

Entretanto, em 1994, Peter W. Shor propôs algoritmos quânticos (SHOR, 1994) que resolvem tais problemas em tempo polinomial. Desta forma, computadores quânticos

podem potencialmente quebrar a maioria dos sistemas criptográficos convencionalmente utilizados.

1.1 Justificativa

Levando-se em consideração a possibilidade da computação quântica se apresentar como viável, acarretando na invalidez da segurança dos sistemas atualmente classificados como seguros e largamente utilizados, a necessidade de estudar maneiras alternativas de segurança neste contexto torna-se evidente.

Neste cenário, pesquisas foram realizadas a fim de avaliar alternativas criptográficas que fossem seguras perante estas abordagens de ataques. Com o decorrer dessas pesquisas, foi constatado que certos sistemas criptográficos clássicos, inspirados em problemas computacionais de natureza completamente diferente dos acima expostos e potencialmente muito mais difíceis de serem resolvidos, são uma interessante alternativa. Eles permanecem essencialmente inafetados pela ameaça da computação quântica e, por isso, vêm sendo chamados sugestivamente de criptossistemas “pós-quânticos”. Este grupo inclui criptossistemas baseados em reticulados e criptossistemas baseados em decodificação de síndromes, como McEliece (McEliece, 1978) e Niederreiter (NIEDERREITER, 1986).

Tais sistemas geralmente apresentam uma vantagem de eficiência algorítmica sobre os esquemas convencionais. Por exemplo, tanto McEliece quanto Niederreiter, sobre um código de comprimento n , têm complexidade de tempo $O(n^2)$, enquanto Diffie-Hellman/DSA e RSA (com operações de expoente privado) com chaves de m bits têm complexidade de tempo $O(m^3)$. Por outro lado, eles são afetados pelo fato de suas chaves serem muito grandes em comparação com as soluções convencionais, sendo preteridos para aplicações práticas.

Neste cenário, apresenta-se como de extrema importância a tarefa de reduzir o

tamanho das chaves para criptossistemas pós-quânticos, mantendo um nível de segurança adequado, visando tanto aplicações atuais, se aproveitando de suas qualidades tais como eficiência algorítmica, quanto futuras, em um possível cenário dotado de tecnologia quântica.

1.2 Objetivos

Este trabalho tem por objetivo estudar e propor técnicas que objetivem a redução do tamanho das chaves do criptossistema pós-quântico McEliece, sem comprometer sua segurança. Com isso, espera-se aproximar tal criptossistema do seletivo grupo de criptossistemas empregados em aplicações práticas.

1.3 Metodologia

A metodologia para este trabalho compreende a análise da família de código originalmente proposta para o sistema McEliece, códigos de Goppa, verificando quando e como seria possível reduzir o tamanho das chaves nela baseadas.

Uma ideia bastante promissora se refere à utilização de sub-famílias deste código que tenham características benéficas para sua representação. Esta intuição também está presente em propostas recentes de utilização de famílias particulares de códigos, como em (BERGER et al., 2009).

A fim de construir novas sub-famílias de códigos que permitam chaves criptográficas mais curtas, sem diminuir o nível de segurança, são analisadas estruturas que permitam uma forma sistematicamente mais reduzida para a representação de códigos de Goppa, família de códigos que permanece inafetada por ataques, quando comparados às maneiras convencionais.

Partindo do fato de que códigos de Goppa admitem matriz de verificação de pa-

ridade em forma de Cauchy (Capítulo 4), mas que tal estratégia expõe a estrutura do código privado, são analisadas formas de ocultar esta estrutura de maneira enxuta. Ao disfarçar esta estrutura em forma *quase-diádica* resultados muito satisfatórios são apresentados.

Provas de que, no pior caso, esta escolha não acarreta prejuízos de segurança e considerações sobre o caso médio são apresentadas, além de detalhes sobre a implementação desta proposta.

1.4 Contribuições originais

Neste trabalho, propomos a classe de códigos de Goppa *quase-diádicos*, que admitem uma matriz de paridade ou matriz geradora com representação muito compacta, para instanciar eficientemente criptossistemas baseados em decodificação de síndromes. Ressaltamos que não se propõe aqui um novo sistema criptográfico, mas sim uma técnica para se obter parâmetros e algoritmos eficientes para esses sistemas.

Ao contrário de muitas outras famílias de códigos propostos (GIBSON, 1995; GIBSON, 1996; OTMANI; TILLICH; DALLOT, 2008; SIDELNIKOV; SHESTAKOV, 1992a), códigos de Goppa resistiram muito bem às tentativas de criptoanálise, e apesar dos progressos consideráveis na área (LOIDREAU; SENDRIER, 1998; SENDRIER, 2000) (BERNSTEIN; BUCHMANN; DAHMEN, 2008), permanecem essencialmente incólumes desde que foram sugeridos para a versão original do criptossistema McEliece.

O método proposto neste documento produz chaves do tipo McEliece que são até um fator $t = \tilde{O}(n)$ menores que as chaves genéricas produzidas a partir de códigos de Goppa de comprimento n , para a correção de t erros em característica 2. No caso binário, essa família também mantém a capacidade de corrigir o número total projetado de erros em vez de apenas a metade; esta propriedade falta a todas as tentativas anteriores de construção de códigos compactos para fins de criptografia, incluindo (BERGER et al.,

2009).

Além disso, a complexidade de todas as operações criptográficas típicas torna-se $\tilde{O}(n)$; especificamente, no cenário criptográfico típico em que $t = O(n/\lg n)$, a operação de geração de código tem complexidade assintótica $O(n \lg^3 n)$ e as de cifração e decifração têm complexidade assintótica $O(n \lg n)$.

1.5 Organização do documento

Este documento está organizado da seguinte forma. O capítulo 2 apresenta os conceitos básicos e a história da criptografia, além de descrever a criptografia usualmente utilizada e sua vulnerabilidade em um ambiente tecnologicamente quântico. O capítulo 3 apresenta o embasamento teórico em que a criptografia baseada em códigos corretores de erros apoia sua segurança. Detalhes acerca do sistema criptográfico McEliece, além das recentes evoluções a ele empregadas, também são apresentados. No capítulo 4, descrevemos nossa proposta de utilização de códigos binários de Goppa em forma quase-diádica, como construí-los, análise de sua complexidade computacional, além de sugestões para seus parâmetros e de resultados referentes à sua implementação. Concluímos no capítulo 5.

2 A CRIPTOGRAFIA CONVENCIONAL

Para a correta compreensão deste texto, é necessário explicitar os conceitos básicos acerca da ciência denominada criptografia. Também julgamos pertinente apresentar um breve resumo da utilização da criptografia no desenrolar da história. Na sequência, são apresentados os criptossistemas usualmente empregados, além de sua fragilidade em um ambiente quântico.

Ao final desse capítulo, espera-se que o leitor tenha compreendido a importância da criptografia no desenvolver da sociedade, esteja familiarizado com os conceitos básicos, tais como esquemas simétricos e assimétricos de criptografia, e com os criptossistemas usualmente utilizados, estando ciente de suas fragilidades.

2.1 O que é criptografia

A criptografia pode ser definida como a ciência que propõe o estudo de técnicas que permitam a escrita de informações de maneira que restrinja somente a pessoas autorizadas a capacidade de recuperar a informação original. Está presente nas mais diversas situações, como no uso de serviços bancários, transmissão de dados militares, comércio eletrônico, entre tantas outras.

Em criptografia, utilizamos alguns termos específicos, como o de *texto claro*, que corresponde à mensagem contendo informações em sua forma original, e de *texto cifrado*, correspondente à mensagem com a informação codificada seguindo determinada lógica. Entre esses dois opostos, existem os seguintes processos destinados a levar a

mensagem de uma extremidade à outra, a saber:

1. *Cifrar*: Processo responsável por transformar a mensagem original em código ininteligível.
2. *Decifrar*: Processo responsável por recuperar a mensagem original, a partir da mensagem codificada.

O processo de cifração é obtido a partir de transformações no texto claro. Estas transformações podem ser de dois tipos:

1. *Transposição*: As letras do texto claro são dispostas em ordem diferente da original.
2. *Substituição*: As letras do texto claro são substituídas por outras (não necessariamente do mesmo alfabeto).

A estratégia de transposição é bastante limitada se comparada com a de substituição. Enquanto a transposição apenas trabalha com o deslocamento das letras da mensagem original, a substituição dispõe de um poderoso artifício que é o uso de alfabetos diferentes entre o texto claro e o texto cifrado, permitindo um maior número de variações.

Apesar de geralmente estar associada à finalidade de ocultar informações, denominada como *confidencialidade*, a criptografia também engloba outros três objetivos, a saber:

1. *Confidencialidade*: Ocultar informações de forma que só pessoas autorizadas consigam recuperá-las.
2. *Integridade*: Possibilitar validar se uma mensagem transmitida foi adulterada, após o envio pelo remetente.

3. *Autenticação*: Possibilitar ao destinatário validar o remetente da mensagem.

4. *Irretratabilidade*: Não permitir ao remetente negar a autoria da mensagem.

É preciso ressaltar que quando falamos de um sistema ou técnica criptográfica em especial, geralmente não estamos focando em todos estes objetivos ao mesmo tempo. Para obtermos esse cenário, geralmente devem ser aplicados mais de um processo criptográfico.

2.2 Histórico da criptografia

Atualmente, quando tratamos de criptografia, é natural associarmos computadores, algoritmos matemáticos e telecomunicação de dados. Entretanto, a história da criptografia começa muito antes do primeiro computador, da atual sofisticação dos métodos matemáticos e da tecnologia de transmissão eletrônica de dados.

Segundo (KAHN, 1996), o primeiro uso da criptografia remonta a 1900 a.C., quando um escriba egípcio utilizou hieróglifos fora do padrão para a escrita no túmulo de um nobre. Existem documentos que indicam o uso da criptografia por volta de 1500 a.C. na Mesopotâmia, a fim de ocultar uma fórmula de fabricação de esmaltes para cerâmica, utilizando símbolos especiais que podiam admitir diversos significados. É possível notar que, nestes primeiros casos, o uso da criptografia está imerso no próprio conceito de escrita: transformar em códigos escritos a informação.

A partir de 600 a.C., indícios comprovam o uso da criptografia de maneira mais sofisticada. Para a escrita do Livro de Jeremias, escribas hebreus utilizaram um método (conhecido como *Atbash*) de substituição das letras do alfabeto, seguindo a regra: trocar a primeira letra do alfabeto pela última, a segunda pela penúltima, e assim por diante.

Nos séculos seguintes, diversos métodos de cifração por substituição foram pro-

postos e utilizados. Na maioria dos casos, a motivação para o emprego de técnicas que ocultassem o conteúdo de uma mensagem era de origem militar. Entre diversos exemplos, em 50 a.C., o imperador romano Julio César utilizou em suas mensagens militares uma cifra de substituição baseada na ideia de trocar cada letra do texto claro, pela letra três posições adjacentes, quando consideramos a sequência alfabética. Esta cifra acabou por denominar a classificação de qualquer cifra de substituição cíclica de alfabeto e tem variações sendo utilizadas até os dias atuais. Um exemplo de sua utilização é a cifra conhecida como *ROT-13*, utilizada em fóruns da Internet, que substitui cada letra da mensagem original pela letra 13 posições a frente (operação realizada *mod 26*), tomando-se a ordem lexicográfica.

Uma adaptação bastante famosa deste conceito é a cifra de Vigenère, proposta em 1465, por Leone Battista Alberti. A ideia é aplicar sequencialmente no texto claro cifras de Cesar com diferentes deslocamentos. Uma maneira de visualizar essa estratégia na prática seria montando uma tabela contendo as 26 variações de cifras de Cesar. Para informar quais linhas desta tabela foram utilizadas, o emissor deveria mandar uma palavra de cabeçalho, indicando as letras iniciais das linhas selecionadas.

A partir do início do século XX, a criptografia começou a tomar um papel ainda mais decisivo no contexto militar. Em 1918, a cifra conhecida como *ADFGVX* foi utilizada pelo exército alemão durante a primeira guerra mundial. Chegou a ser quebrada meses depois, mas sem maiores implicações à guerra em si. Entretanto, na segunda guerra mundial, acredita-se que a quebra do sistema criptográfico da Alemanha nazista, implementado na máquina de nome *Enigma*, encurtou o conflito em cerca de um ano.

Apesar de estar sendo cada vez mais utilizada, neste momento, a criptografia apoiava boa parte de sua segurança no próprio método de cifração/decifração. Era necessário encontrar maneiras de minimizar o segredo a ser protegido. Além disso, a criptografia também carecia de padrões para seu emprego. Preenchendo estas lacunas,

em 1976, a NSA (do inglês *National Security Agency*), do governo dos Estados Unidos da América, apresenta o DES (do inglês *Data encryption Standard*) como padrão de cifração simétrica, sendo esta uma variação do algoritmo Lucifer, criado na IBM anos antes. Neste mesmo ano, Whitfield Diffie e Martin Hellman publicaram (DIFFIE; HELLMAN, 1976), propondo o esquema de criptografia assimétrica.

Dois anos depois, Ronald L. Rivest, Adi Shamir e Leonard M. Adleman apresentam o algoritmo de criptografia RSA, posteriormente adotado como padrão de criptografia assimétrica. Em 1986, Victor S. Miller propõe algoritmo que utiliza curvas elípticas para criptografia assimétrica, reduzindo sensivelmente o tamanho das chaves, se comparado com outros sistemas criptográficos assimétrico.

Em 1994, Peter W. Shor propõe algoritmos quânticos capazes de quebrar o RSA e criptossistemas baseados em curvas elípticas em tempo polinomial. Isso significa que, se executados em um computador com tecnologia quântica, seria possível resolver os problemas matemáticos em que o RSA e curvas elípticas apoiam sua segurança, tornando-os suscetíveis a fraudes.

No ano seguinte, o SHA-1, algoritmo de hash, é aprovado pelo governo dos EUA para ser usado por todos os departamentos e agências federais na autenticação de documentos digitais. Em 2000, o algoritmo Rijndael é selecionado para substituir o DES e é denominado AES (do inglês *Advanced Encryption Standard*).

Como podemos notar, a criptografia desde as formas mais rústicas, como no início da civilização moderna, até o seu emprego eletrônico, como nos dias atuais, figura como uma importante ferramenta no desenvolvimento da sociedade.

2.3 Esquemas criptográficos

Tendo introduzido os conceitos básicos de criptografia e demonstrado, através de seu emprego no desenrolar da história, sua importância, podemos passar para um nível

mais detalhista acerca desta ciência.

Segundo (BUNCHAMANN, 2004), o tradicional tópico da criptografia, a cifração, pode ser definido como sendo a tupla (P, K, C, E, D) , com as seguintes propriedades:

- P , chamado de *espaço de textos claros*, é um conjunto com elementos sendo textos claros.
- C , chamado de *espaço de textos cifrados*, é um conjunto com elementos sendo textos cifrados.
- K , chamado de *espaço de chaves*, é um conjunto com elementos sendo chamados de chaves.
- $E = \{E_k : k \in K\}$ é uma família de funções que mapeiam $E_k : P \rightarrow C$, com elementos sendo chamados de *funções de cifração*.
- $D = \{D_k : k \in K\}$ é uma família de funções que mapeiam $D_k : C \rightarrow P$, com elementos sendo chamados de *funções de decifração*.
- Para cada $e \in K$, existe $d \in K$, tal que $D_d(E_e(p)) = p$, para $p \in P$.

Para ilustrar essa definição, tomamos como exemplo a Cifra de Cesar, explicada em 2.2. Representando as 26 letras do alfabeto pelo alfabeto $\Sigma = (0..25)$ e selecionando $k = 3$, como sendo o deslocamento entre uma letra do texto claro e uma letra do texto cifrado, temos a seguinte tupla:

- $P = \{x : x \in \Sigma\}$ é o espaço de textos claros.
- $C = \{x : x \in \Sigma\}$ é o espaço de textos cifrados.
- $k = 3$ é o elemento chave.
- $E_k : \Sigma \rightarrow \Sigma, x \rightarrow (x + k) \bmod 26$ é a função de cifração.

- $D_k : \Sigma \rightarrow \Sigma, x \rightarrow (x - k) \bmod 26$ é a função de decifração.

Como podemos notar, a chave para cifração e decifração é a mesma ($k = 3$). Entretanto, isso não é válido para todos os sistemas criptográficos, como veremos nas seções a seguir.

2.3.1 Criptografia simétrica

A característica que define se uma cifra é simétrica ou não se refere a quantidade de chaves utilizadas entre o processo de cifração e de decifração. Se for utilizada apenas uma mesma chave, tanto para o processo de cifração quanto o de decifração, temos que a cifra é simétrica.

Vale observar que, como a chave de cifração é necessária para o processo de decifração, será necessário que os interlocutores comuniquem, de alguma forma, além do texto cifrado, a própria chave capaz de recuperar o texto claro. Isso confere à criptografia simétrica certas restrições em seu emprego, uma vez que um dos pressupostos para o uso da criptografia de dados é de que o canal conveniente para a comunicação entre o remetente e o destinatário da mensagem não ser um canal seguro.

Outro ponto a ser considerado se refere ao fato de que a chave está associada à relação entre os dois interlocutores. Assim, para cada duas pessoas que desejem se comunicar seguramente, deve existir uma chave. Caso tenhamos n pessoas interessadas em se comunicar duas a duas seguramente, teríamos a necessidade de dispor de $n(n - 1)/2$ chaves.

2.3.2 Criptografia assimétrica

Em 1976, Whitfield Diffie e Martin Hellman propuseram um cenário um pouco mais complexo. Visando sanar algumas dificuldades encontradas no uso da criptografia simétrica, foi proposto um esquema que dispensasse a comunicação de chave entre os

interlocutores, associando cada segredo a um interlocutor e não mais a relação entre dois interlocutores.

Para atingir esse objetivo, foi sugerido que cada interlocutor tivesse um par de chaves. Cada uma das chaves do par tem uma classificação: Pública ou Privada. A chave pública deve ser disponibilizada, por meio de uma entidade de confiança, publicamente. A chave privada deve ser mantida em segredo pelo seu detentor. O par de chaves também deve respeitar duas propriedades, a saber:

1. O texto cifrado por uma das chaves do par, somente a outra chave do par será capaz de decifrar.
2. Não deverá ser possível obter a chave privada a partir da chave pública.

Desta forma, não há mais a necessidade da transmissão de chave entre os interlocutores. Para enviar uma mensagem cifrada, o remetente deverá cifrá-la utilizando a chave pública do destinatário. Como somente o destinatário tem a sua chave privada, somente ele será capaz de recuperar o texto claro, sem existir a necessidade de transmissão de chaves entre os dois.

Ilustrando esse conceito, imagine que Alice queira enviar a Bob uma mensagem cifrada. As chaves públicas de ambos estão disponíveis a qualquer interessado por meio da entidade de confiança para este fim e as chaves privadas mantidas em segredo pelo seu respectivo detentor. Assim, Alice deve obter a chave pública de Bob e a utilizar na cifração do texto claro. Para Bob recuperar o texto claro, ele deverá decifrar a mensagem utilizando sua chave privada.

Com este esquema, outra finalidade atualmente bastante comum em criptografia é possível: garantir a autoria de uma mensagem. Ao utilizar sua chave privada para a cifração de uma mensagem, o remetente assegura ser o autor da mesma, uma vez que somente será possível decifrá-la utilizando sua chave pública.

Apesar de resolver o problema de gerenciamento de chaves, a criptografia assimétrica tem uma desvantagem com relação à criptografia simétrica. Os sistemas de criptografia assimétrica não são tão eficientes algoritmicamente como os de criptografia simétrica. Esse fato motivou a proposta de utilização de ambos os esquemas durante o processo. Desta forma, para Alice enviar uma mensagem com conteúdo cifrado para Bob, ela deverá gerar uma chave simétrica e a utilizar para cifrar o texto claro. Então, obter a chave pública de Bob e a utilizar para cifrar sua chave simétrica. Enviando essas duas informações, Bob será capaz de recuperar o texto claro utilizando sua chave privada e então utilizando a chave simétrica obtida no passo anterior.

Deste modo, tanto o processo de cifração do texto, que utilizou a eficiente criptografia simétrica, quanto o processo de cifração da chave simétrica de Alice, que utilizou criptografia assimétrica em um texto bastante curto, são rápidos.

2.4 Criptossistemas convencionais

Neste capítulo, recapitulamos os criptossistemas convencionais mais empregados atualmente e expomos sua vulnerabilidade diante de ataques quânticos, como contexto à necessidade de alternativas pós-quânticas.

Atualmente, existem à disposição diversos sistemas criptográficos, tendo seu emprego motivado por inúmeras particularidades. Muitos foram os sistemas propostos e quebrados, resguardando a um seleto grupo a classificação de *seguros* (respeitando certas condições). Podemos classificar estes últimos em três categorias, a respeito do problema matemático em que baseiam sua segurança, a saber:

- *Sistemas baseados em fatoração de números inteiros* (também conhecidos apenas por IFS, do inglês *Integer Factorizing Systems*): tem sua segurança baseada no problema da fatoração de números inteiros (também conhecido apenas por IFP, do inglês *Integer Factorizing Problem*).

- *Sistemas baseados no logaritmo discreto* (também conhecidos apenas por DLS, do inglês *Discrete Logarithm Systems*): tem sua segurança baseada no problema da determinação do logaritmo discreto (também conhecido apenas por DLP, do inglês *Discrete Logarithm Problem*).
- *Sistemas baseados em curvas elípticas* (também conhecidos apenas por ECDLS, do inglês *Elliptic Curve Discrete Logarithm Systems*): tem sua segurança baseada no problema da determinação do logaritmo discreto em curvas elípticas (também conhecido apenas por ECDLP, do inglês *Elliptic Curve Discrete Logarithm Problem*).

Na Tabela 1, apresentamos a classificação de alguns dos principais sistemas criptográficos:

Tabela 1: Classificação dos principais sistemas criptográficos

IFS	DLS	ECDLS
RSA	ElGamal	ElGamal (curvas elípticas)
Rabin-Williams	Diffie-Helman key exchange	Diffie-Helman key exchange
	Schnorr	(curvas elípticas)

Entre estes poucos, RSA e criptografia baseada em curvas elípticas em geral (também conhecida como ECC, do inglês *Elliptic Curve Cryptography*) aparecem com maior destaque, visto serem considerados como os mais adaptados às necessidades usuais, além de serem considerados eficientes e seguros. A seguir, apresentamos uma breve descrição desses sistemas, além de analisar um importante resultado obtido em 1994 por Peter W. Shor, afetando a confiabilidade dessas soluções.

2.4.1 RSA

O sistema criptográfico RSA foi proposto em 1978 por Ronald Rivest, Adi Shamir e Leonard Adleman, como uma factível implementação do esquema de criptografia assimétrica. Com ele também é possível implementar um esquema de autenticação. Sua segurança está baseado no problema matemático de fatoração de números inteiros grandes em números primos.

Desde sua proposta até os dias atuais, é considerado como um dos mais adaptados às aplicações que comumente necessitam atentar para questões de segurança. Isso é devido, em grande parte, a seu custo benefício, ou seja, ao seu nível de segurança dado seu custo computacional e de armazenamento.

Os algoritmos referentes à geração de suas chaves, cifração e decifração são de ordem cúbica e são recomendadas chaves de tamanho 1024–2048 bits, dada a atual geração de computadores passíveis de serem empregados para a sua quebra.

2.4.1.1 Problema da fatoração de números inteiros em primos

Sendo o RSA um sistema criptográfico assimétrico, espera-se que calcular sua chave secreta d a partir de sua chave pública (n, e) seja uma tarefa impossível. É importante ressaltar que não é possível fazer tal afirmação, visto estar relacionada diretamente com a capacidade computacional empregada para sua quebra. Entretanto, pode-se tentar reduzir tal tarefa a um problema matemático e então analisar a segurança do sistema por meio da dificuldade em resolvê-lo.

Dada a natureza do problema matemático que a segurança do RSA está baseado, é possível reduzi-lo ao problema de fatoração de números inteiros em primos (no caso, fatorar n nos fatores p e q). Este problema é experimentalmente reconhecido como difícil de ser solucionado, apesar de não existirem provas formais de tal dificuldade. Entretanto, se levarmos em conta o tamanho dos números geralmente utilizados para

o RSA, 1024 bits, tal problema torna-se, pelo poderio computacional e estratégias conhecidas, impossível de ser realizado. É importante ressaltar que esta relação não comprova a segurança do sistema criptográfico como um todo, visto que outros aspectos do sistema podem conter fraquezas não diretamente relacionadas à tarefa de se obter a chave privada a partir da chave pública.

O problema da fatoração de números inteiros está contido na área da matemática conhecida por teoria dos números. Formalmente, consiste em encontrar um divisor não trivial de um número composto. Atualmente, não é conhecido algoritmo que execute tal tarefa em tempo polinomial.

Como desejável, tal problema admite uma única resposta para cada instância, como pode ser conferido em (MENEZES; OORSCHOT; VANSTONE, 1996) no teorema fundamental da aritmética:

Teorema 1. (*Teorema Fundamental da Aritmética*)

Seja $a > 1$ um número inteiro positivo. Então, existem primos positivos $p_1 \leq p_2 \leq \dots \leq p_t$, tais que $a = p_1 p_2 \dots p_t$ e essa decomposição é única.

2.4.1.2 Algoritmos

A seguir apresentamos os algoritmos de geração de chaves, cifração e decifração utilizados pelo RSA.

O processo de geração de chaves pode ser descrito como:

1. Gerar p , um número inteiro aleatório grande.
2. Gerar q , um número inteiro aleatório grande.
3. $n \leftarrow p * q$
4. $\phi(n) \leftarrow (p - 1) * (q - 1)$

5. Escolher e , um número inteiro que satisfaz:

(a) $1 < e < \phi(n)$

(b) $\gcd(e, \phi(n)) = 1$

6. Obter d , um número inteiro tal que:

(a) $1 < d < \phi(n)$

(b) $de \equiv 1 \pmod{\phi(n)}$

Desta forma, temos o par de chaves composto por:

1. Chave pública: (n, e)

2. Chave privada: (d)

O processo de cifração pode ser descrito como:

1. $c \leftarrow m^e \pmod{n}$

O processo de decifração pode ser descrito como:

1. $m \leftarrow c^d \pmod{n}$

2.4.2 Criptografia baseada em curvas elípticas

A criptografia baseada em curvas elípticas, também conhecida como ECC (do inglês *Elliptic Curve Cryptography*) foi proposta, de maneira independente, por Neal Koblitz (KOBLOITZ, 1987) e Victor Miller (MILLER, 1985).

Assim como o RSA, pode ser implementada como um sistema criptográfico assimétrico, e interessantes benefícios são obtidos quando da sua utilização, tais como: chaves curtas e processamento eficiente.

Sua segurança está baseada em problemas relacionados à determinação do logaritmo discreto, apresentado a seguir.

2.4.2.1 Problema da determinação do logaritmo discreto

O problema de determinação do logaritmo discreto em curvas elípticas foi derivado do problema mais geral, sem restrição ao elemento pertencer a uma curva elíptica. Tal problema, poder ser descrito como:

- Seja p um número ímpar.
- Seja $Z_p = \{0, 1, \dots, p-1\}$ um corpo finito.
- Seja $Z_p^* = \{1, \dots, p-1\}$ um grupo multiplicativo cíclico.
- Seja α um gerador para Z_p^* , ou seja:

$$Z_p^* = \{\alpha^0 \bmod p, \alpha^1 \bmod p, \dots, \alpha^{p-2} \bmod p\}$$

- Problema DLP:

Dado algum a , calcular $b \equiv \alpha^a \pmod{p}$ é fácil.

Dado algum b , encontrar a tal que $b \equiv \alpha^a \pmod{p}$ é difícil.

Simplificadamente (omitindo-se o $\bmod p$), pode ser descrito por $a = \log \alpha^b$. Atualmente, não é conhecido algoritmo eficiente que resolva tal problema.

A seguir, apresentamos os conceitos relacionados a curvas elípticas.

2.4.2.2 Problema da determinação do logaritmo discreto em curvas elípticas

Uma curva elíptica E pode ser definida como o conjunto de soluções de uma equação da forma:

$$E : Y^2 = X^3 + AX + B \quad (2.1)$$

Além disso, também deve existir um ponto ϑ e as constantes A e B devem respeitar: $4A^3 + 27B^2 \neq 0$, condições que posteriormente abordaremos.

Uma interessante propriedade dessas curvas se refere à *soma* entre dois pontos pertencentes a uma curva elíptica. Para obtermos o resultado da soma, aqui representada por $\oplus$, entre P e Q , traçamos a reta que liga esses dois pontos e que também intersecta, em geral, a curva mais uma vez. Chamando este terceiro ponto de R' , temos que a soma entre P e Q é a inversão, no eixo Y , do ponto R' , que aqui chamaremos por R . A Figura 1, retirada de (HOFFSTEIN; PIPHER; SILVERMAN, 2008), ilustra essa curva e os pontos P , Q , R' e R :

Figura 1: Curva Elíptica E e os pontos P , Q , R' e R .

Para calcular este terceiro ponto, o primeiro passo é determinar a reta que une os dois pontos P e Q . A mesma pode ser obtida a partir dos dois pontos (x_P, y_P) e

(x_Q, y_Q) , lembrando que $Y - y_1 = \lambda(X - x_1)$, onde $\lambda = (y_1 - y_2)/(x_1 - x_2)$. Aplicando o valor de Y obtido dessa reta na equação 2.1, obtemos uma equação de terceiro grau. Geralmente, resolver uma equação de terceiro grau não é trivial, entretanto, já temos duas de suas raízes à disposição: x_P e x_Q , uma vez que P e Q interceptam a reta. Assim, é fácil a determinação da última raiz $x_{R'}$. Substituindo na equação da reta o valor de $x_{R'}$, determinamos $y_{R'}$ que, por fim, é multiplicado por -1 , determinando o ponto $R = (x_{R'}, -y_{R'})$.

Entretanto, para o caso de somarmos um ponto P a si mesmo devemos utilizar uma outra abordagem. Imaginando a reta que liga dois pontos P e Q , estando Q cada vez mais próximo de P . Quando $P = Q$, tal reta será obtida da derivada da curva no ponto P . A soma de P realizada n vezes é denotada por nP . A condição $4A^3 + 27B^2 \neq 0$ nada mais é do que o determinante da equação cúbica que define a curva elíptica 2.1 ser não nulo, resultando que suas três raízes sejam distintas. Outra condição citada se refere à existência do ponto ϑ , que representa o elemento nulo da soma. Assim, temos que dado um ponto P pertencente à curva elíptica: $P + \vartheta = P$.

Até agora, nada foi dito acerca dos elementos que representam os pontos na curva elíptica. Além dos números reais e complexos, também é possível adaptar essa ideia a elementos de um corpo finito. Neste caso, para o corpo finito $\mathbb{F}_p$, temos a seguinte restrição extra para a curva elíptica denotado por $E(\mathbb{F}_p)$:

$$E(\mathbb{F}_p) = \{(x, y) : x, y \in \mathbb{F}_p, y^2 = x^3 + Ax + B\}$$

Tendo apresentado este embasamento, é possível unir o problema DLP a curvas elípticas. Esta união é feita levando-se em conta:

- Problema ECDLP:

Dado o inteiro n e o ponto P da curva elíptica $E(\mathbb{F}_p)$, calcular $Q \equiv nP$ é fácil.

Dado algum Q , encontrar n tal que $Q \equiv nP$ é difícil.

2.4.2.3 Algoritmos

Existem diversas implementações de operações criptográficas baseadas em curvas elípticas. A fim de ilustrar o conceito, iremos apresentar os algoritmos de geração de chaves, cifração e decifração presentes no sistema criptográfico conhecido como ElGamal (ELGAMAL, 1985), na versão baseada em ECC.

Seja p , um inteiro primo grande, uma curva elíptica E sobre $\mathbb{F}_p$, e um ponto $P \in E(\mathbb{F}_p)$ disponibilizados publicamente. O processo de geração de chaves pode ser descrito como:

1. Escolher um inteiro n .
2. Calcular $Q = nP \in E(\mathbb{F}_p)$.

Desta forma, temos o par de chaves composto por:

1. Chave pública: Q .
2. Chave privada: n .

O processo de cifração é descrito a seguir. Seja $M \in E(\mathbb{F}_p)$ o texto claro.

1. Escolher uma chave intermediária k .
2. Calcular $C_1 = kP \in E(\mathbb{F}_p)$.
3. Calcular $C_2 = M + kQ \in E(\mathbb{F}_p)$.
4. O texto cifrado é (C_1, C_2) .

O processo de decifração é descrito a seguir.

1. Calcular $C'_1 = nC_1$.
2. Calcular $M = C_2 - C'_1$.

2.5 Algoritmo quântico de Shor

Na computação convencional, temos que a unidade básica de informação é o dígito binário, também conhecido por *bit*. Tal informação pode armazenar apenas dois conteúdos, *0* ou *1*. As operações sobre esses valores seguem a lógica booleana, operando em uma par de valores, no caso das operações AND e OR, e em apenas um valor, no caso do NOT. Portas lógicas tratam sequências de bits seguindo tais regras e por meio da utilização de diversas delas é possível efetuar os cálculos realizados nos computadores.

No caso da computação quântica, a unidade básica de informação é conhecida como *qubit* (do inglês *quantum bits*). Sequências de *qubits* são tratadas por portas lógicas quânticas, que simulam as leis da mecânica quântica, como superposição de estados, que possibilita o armazenamento de mais de um estado para um *qubit*, conferindo características únicas a esse tipo de processamento de dados.

A representação para um *qubit* com dois estados é:

$$\alpha|0\rangle + \beta|1\rangle$$

Onde α e β são números complexos que satisfazem $\alpha^2 + \beta^2 = 1$ e $|0\rangle$ representa o estado 0 e $|1\rangle$ o estado 1 do *qubit*. Com a utilização dessa superposição de estados, é possível efetuar um cálculo de uma única vez para diversos valores de entrada.

Aproveitando-se dessa vantagem, Peter Shor propôs em 1994 algoritmos polinomiais que resolvem problemas de extrema importância para a criptografia, tais como a fatoração de números inteiros em primos e de determinação do logaritmo discreto

(inclusive para o caso de curvas elípticas).

A seguir, apresentamos resumidamente os passos para a fatoração de um número n por meio do algoritmo quântico de Shor. Para uma explicação mais detalhada acerca desse processo, consultar (SHOR, 1994).

Este algoritmo baseia-se no fato de que $x^a \bmod n$ é uma função periódica quando x é um inteiro coprimo (máximo divisor comum entre ambos é igual a 1) de n . Uma função periódica tem um período r tal que: $x^0 \bmod n = 1 = x^r \bmod n$. Obter esse período r é um processo exponencial em computadores clássicos, entretanto, utilizando tecnologia quântica para paralelizá-lo, é possível determiná-lo em apenas um passo. Conhecendo r , temos:

$$x^r \equiv 1 \bmod n$$

$$(x^{r/2})^2 = x^r \equiv 1 \bmod n$$

$$(x^{r/2})^2 - 1 \equiv 0 \bmod n$$

Sendo r um número par:

$$(x^{r/2} - 1)(x^{r/2} + 1) \equiv 0 \bmod n$$

Assim, podemos concluir que $(x^{r/2} - 1)(x^{r/2} + 1)$ é um inteiro múltiplo de n . Sendo $|x^{r/2}| \neq 1$, então no mínimo um dos $(x^{r/2} - 1)$ e $(x^{r/2} + 1)$ tem um fator não trivial em comum com n . Desta forma, calculando $\gcd(x^{r/2} - 1, n)$ e $\gcd(x^{r/2} + 1, n)$ iremos obter um fator não trivial de n .

Conforme dito anteriormente, com esse resultado, os criptossistemas convencionais (baseados na fatoração de números inteiros e na determinação do logaritmo discreto) se apresentam como vulneráveis a ataques provenientes de computadores quânticos. Entretanto, outros criptossistemas baseados em diferentes problemas matemáticos

permanecem inatingidos, tais como criptossistemas baseados em códigos corretores de erros.

2.6 Resumo

A criptografia pode ser definida como a ciência que propõe o estudo de técnicas que permitam a escrita de informações de maneira que restrinja somente a pessoas autorizadas a capacidade de recuperar a informação original. Esta ciência tem sido utilizada desde cerca de 1900 a.C. até os dias atuais, tendo desempenhado papel importante em passagens, tais como a segunda guerra mundial.

A criptografia pode ser implementada de maneira simétrica (quando o segredo utilizado, denominado *chave*, é o mesmo tanto no processo de cifração quanto de decifração) ou assimétrica (quando um par de chaves é utilizado). A combinação de esquemas assimétricos com simétricos possibilita inúmeras aplicações.

Os criptossistemas assimétricos convencionalmente utilizados são baseados no problema da fatoração de números inteiros ou no problema da determinação do logaritmo discreto (podendo ser adaptado a curvas elípticas). Como expoente do primeiro grupo temos o RSA e do segundo grupo, como exemplo, o criptossistema de ElGamal. Ambas as soluções possibilitam parâmetros de segurança e eficiência razoáveis para os dias atuais.

Entretanto, resultados obtidos por Peter W. Shor indicam que tais soluções são vulneráveis perante ataques executados em computadores quânticos. Neste cenário, criptossistemas baseados em outros tipos de problemas matemáticos devem ser estudados a fim de garantir a segurança mesmo nessas circunstâncias.

Criptossistemas baseados em códigos corretores de erros têm se mostrado como uma interessante alternativa, permanecendo seguros mesmo perante ataques quânticos.

3 CRIPTOGRAFIA BASEADA EM CÓDIGOS CORRETORES DE ERROS

A criptografia está diretamente ligada a problemas matemáticos de difícil resolução. Identificar a possibilidade de adaptar determinado problema à construção de uma funcionalidade criptográfica não é uma tarefa simples e, para tanto, inúmeros problemas são estudados. Um campo que se mostra versátil nesse sentido é o ligado à teoria da codificação, que tem como objetivo primário a busca pela eficiência na troca de informações.

A seguir, apresentamos a teoria em que a segurança da criptografia baseada em códigos corretores de erros se apoia. Ao final deste capítulo, espera-se que o leitor esteja familiarizado com os conceitos de código linear, mecanismos de correção de erros e, em especial, com os códigos de Goppa e com o criptossistema McEliece.

3.1 Teoria da codificação

A Teoria da Codificação tem como objetivo assegurar que, ao transmitir uma coleção de dados através de um canal sujeito a ruídos (ou seja, à perturbações nos dados enviados), o destinatário dessa transação possa ser capaz de recuperar a mensagem original. Para isso, deve-se encontrar maneiras eficientes de adicionar informação redundante à mensagem original de tal forma que, caso a mensagem chegue ao destinatário contendo erros (existindo inversão em certos bits, para o caso de mensagens binárias), o receptor possa corrigi-la. A Figura 2, baseada em (HUFFMAN; PLESS, 2003), ilustra

esta situação.

Figura 2: Canal de Comunicação.

Como será explicado mais adiante, no contexto da criptografia, esta capacidade de correção de ruídos está intimamente relacionada com a segurança de alguns sistemas criptográficos. A grosso modo, tomando partido do fato de que somente tendo informações específicas sobre o código utilizado é possível corrigir suas palavras de código, tais sistemas criptográficos desafiam a capacidade do interlocutor de correção, distinguindo assim quem deve ter acesso às informações trocadas de quem não deve.

A seguir, introduzimos os conceitos básicos desse ramo de pesquisa.

3.2 Códigos lineares

Um dos conceitos primários da teoria de códigos se refere a *código linear binário*, que pode ser definido como:

Definição 1. *Um subespaço linear de dimensão k , do espaço vetorial $\mathbb{F}_2^n$, onde $\mathbb{F}_2$ é um corpo finito com 2 elementos, é um código linear binário $\Gamma(n, k)$, de tamanho n e posto k .*

Os parâmetros n e k também são conhecidos respectivamente como *comprimento* e *dimensão* do código. Além disso, um vetor que represente um elemento deste código é conhecido como *palavra de código*.

Uma maneira eficiente de representar um código linear $\Gamma(n, k)$ é por meio de uma

matriz geradora. Tal matriz é capaz de gerar qualquer palavra pertencente ao código por meio da combinação linear de suas linhas. Para isso, ela deverá ter dimensões $k \times n$, suas k linhas deverão ser linearmente independentes entre si e seus elementos pertencentes a $\mathbb{F}_{2^n}$. Descrevendo o código Γ em função de uma *matriz geradora* G , teríamos: $\Gamma = \{ uG \mid u \in \mathbb{F}_2^k \}$.

A fim de ilustrar os conceitos, utilizaremos como exemplo a seguinte matriz geradora, capaz de gerar o código conhecido como código de Hamming (7, 4, 3) (HUFFMAN; PLESS, 2003):

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

O processo de codificação é bastante simples, resumindo-se à multiplicação da mensagem pela matriz geradora do código.

Exemplo 1.

Suponha que a mensagem $m = [1 \ 1 \ 0 \ 0]$ deve ser codificada. Para tanto, basta multiplicar o vetor m por G :

$$\begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

onde $m' = [0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1]$ é a palavra de código resultante.

□

Observe que o vetor m' é maior do que o vetor m , ilustrando o fato de termos adicionado informação redundante aos dados originais, o que nos ajudará posteriormente

na tarefa de corrigir mensagens com erros. Analisando a matriz geradora G , note que para uma mensagem original $m = [m1, m2, m3, m4]$, sua mensagem codificada m' seria da forma:

$$m' = m * G = [(m1 + m2 + m3), (m1 + m2 + m4), m2, (m1 + m3 + m4), m3, m4, m1]$$

Expressando a palavra codificada desta maneira, fica claro que se lermos as componentes da palavra de código na ordem: 7ª, 3ª, 5ª e 6ª, determinaremos diretamente a sequência $[m1, m2, m3, m4]$, correspondente a mensagem original m e ilustrando assim um método de decodificação (AU; EUBANKS-TURNER; EVERSON, 2003) para o código de Hamming-(7,4,3).

Vale ressaltar também que se escalonarmos à esquerda a matriz geradora, o código gerado continuará sendo o mesmo e a decodificação de suas palavras de código se tornará direta, como podemos notar no exemplo a seguir, obtido da codificação da palavra de código $m = [1 \ 1 \ 0 \ 0]$ por intermédio da matriz do exemplo anterior escalonada.

Exemplo 2.

Multiplicando o vetor m por G escalonado:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

onde $m' = [1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0]$ é a palavra de código resultante.

□

Repare que as quatro primeiras posições da palavra de código são idênticas ao vetor m e que as últimas três posições atuam como verificadores de erro na mensagem, muitas vezes conhecidos como *checksum*.

Atualmente, existem diversas abordagens para o problema da correção de erros em mensagens binárias. Algumas delas são: método da máxima probabilidade, método da mínima distância e método da decodificação de síndromes. A ideia central destes algoritmos é tentar se aproveitar das informações redundantes geradas na codificação para corrigir possíveis erros.

Para analisar alguns dos métodos de correção de erros, as definições a seguir serão necessárias:

Definição 2. *O peso de um vetor é o número de componentes não-nulas.*

Exemplo 3. Exemplificando o conceito:

$$\text{Peso}([1 \ 0 \ 1]) = 2.$$

$$\text{Peso}([0 \ 0 \ 0]) = 0.$$

□

Definição 3. *O peso mínimo d de um código Γ é o menor peso existente em alguma palavra não totalmente nula (diferente de zero em alguma componente) desse código.*

O peso mínimo é uma informação que caracteriza um código, sendo comum sua utilização junto dos parâmetros n e k para identificar um código em especial. Neste caso, a nomenclatura para identificá-lo segue a forma: $\Gamma(n, k, d)$.

Tendo estas definições sido caracterizadas, é possível explorar o fato enunciado no Teorema 2:

Teorema 2. (MACWILLIAMS; SLOANE, 1977) *Um código linear binário, representado por $\Gamma(n, k, d)$, sendo n seu comprimento, k sua dimensão e d seu peso mínimo, pode corrigir até $t = \lfloor \frac{d-1}{2} \rfloor$ erros.*

Assim, para o código de Hamming (7, 4 3), temos que $n = 7$, $k = 4$ e $d = 3$. Desses parâmetros, podemos concluir que a mensagem codificada terá tamanho 7, as mensagens originais deverão ter tamanho 4 e que este código é capaz de corrigir até $t = \lfloor (d - 1)/2 \rfloor = 1$ erro.

Outra característica relevante dos códigos lineares se refere ao relacionamento entre os parâmetros n , k e d de um código:

Teorema 3. (SINGLETON, 1964) *Seja $\Gamma = (n, k)$ um código linear de peso mínimo d sobre $\mathbb{F}_2$. Então $d + k \leq n + 1$.*

Isso implica que, de certa forma, deve-se optar entre eficiência (k com valor grande) e capacidade de correção de erros (d com valor grande). Para melhor ilustrar esses conceitos, a seguir exemplificamos a utilização do método de correção de erros baseado em máxima probabilidade.

Exemplo 4.

Imagine que desejamos transmitir a mensagem $[1\ 1\ 0\ 0]$, que ao ser codificada no código de Hamming $(7, 4, 3)$ se transforma em $[0\ 0\ 1\ 1\ 0\ 0\ 1]$. Porém, durante a transmissão dos dados, uma perturbação inverteu um dos bits, acarretando no fato de que a mensagem recebida foi $[0\ 0\ 1\ 1\ 1\ 0\ 1]$. Como este código tem $t = 1$, podemos nos assegurar de que ele é capaz de corrigir esse bit alternado. Para isso, o método de máxima probabilidade se utiliza da Tabela 2 contendo cada palavra de código associada a todos as 7 possíveis variações (correspondentes a um erro em cada um dos bits):

Tabela 2: Tabela de correção de erros do método máxima probabilidade para o exemplo

Palavra	0000000	1101001	1110000	0011001	...
	0000001	1101000	1110001	0011000	...
	0000010	1101011	1110010	0011011	...
	0000100	1101101	1110100	0011101	...
	0001000	1100001	1111000	0010001	...
	0010000	1111001	1100000	0001001	...
	0100000	1001001	1010000	0111001	...
	1000000	0101001	0110000	1011001	...

Para corrigir a mensagem recebida $[0\ 0\ 1\ 1\ 1\ 0\ 1]$, o algoritmo encontra sua referência na tabela e a substitui pela palavra da primeira linha de sua coluna.

□

Note que esse tipo de abordagem sempre retornará o código que mais se assemelha à mensagem contendo os erros, ou seja, o de maior probabilidade de acerto (uma vez que se espera probabilidade inferior a 50% na ocorrência de erro na transmissão de um bit). Vale ressaltar que este é considerado um dos melhores métodos de correção de erros, porém, a construção de tabelas para códigos maiores do que o exemplificado é dispendiosa e ineficiente. A fim de melhorar esse aspecto, apresentaremos o método de *decodificação de síndromes*, posteriormente às seguintes definições.

Definição 4. *Seja Γ um código (n, k, d) . Definimos $\Gamma^\perp := \{ v \in V \mid v \cdot c = 0, \forall c \in \Gamma \}$, onde $\cdot$ representa o produto escalar. $\Gamma^\perp$ é chamado de dual do código Γ e tem dimensão $n - k$.*

Definição 5. *Uma matriz de verificação de paridade de um código Γ é uma matriz de dimensões $(n - k) \times n$, onde as linhas são linearmente independentes entre si e tais que o produto (módulo 2) de cada uma delas por qualquer palavra do código Γ resulte em 0. Essa matriz gera o código dual de Γ .*

Exemplo 5. A matriz de verificação de paridade para o Código de Hamming $(7, 4, 3)$ seria:

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Multiplicando pela palavra de código $[0\ 0\ 1\ 1\ 0\ 0\ 1]$, verificamos:

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

□

Como existe apenas um código dual para cada código linear, podemos considerar a matriz de verificação de paridade como uma forma alternativa de representação de um código linear.

Definição 6. *Seja $v \in \mathbb{F}_q^n$. Multiplicando a matriz de verificação de paridade de um código Γ por v^t (a transposição do vetor v), o vetor resultante de dimensão $(n - k)$ é chamado de Síndrome de v em Γ .*

Fato 1. *Seja m uma mensagem enviada pela origem e $\bar{m}$ a mensagem recebida pelo destino. Caso a síndrome de $\bar{m}$ for não nula, a mensagem contém erros e sua síndrome é igual à síndrome do vetor de erros e , tal que $\bar{m} = m + e$.*

Isso se deve a Hm^t corresponder a um vetor nulo, já que m é uma palavra de código, então: $H\bar{m}^t = H(m + e)^t = Hm^t + He^t = He^t$. Assim, a maneira para corrigir uma mensagem $\bar{m}$, codificada e contendo erros, pelo método de *decodificação de síndromes* pode ser descrita como:

1. Calcular e armazenar as síndromes de todos os possíveis vetores de erros.
2. Calcular a síndrome da mensagem recebida, a qual corresponde à síndrome do vetor de erros contido na mensagem.

3. Identificar o vetor de erro através de sua síndrome.
4. Somar o vetor de erros identificado pelo passo anterior à mensagem recebida, obtendo a mensagem original.

Observe que novamente temos a utilização de uma estrutura armazenando dados pré-calculados a fim de determinarmos qual palavra de código a mensagem recebida representa. Porém, neste caso, apenas calculamos e armazenamos a *síndrome* de cada possível erro, ao invés de todas as variações de cada uma das palavras códigos. Essa diferença acarreta em um grande ganho de desempenho, visto que a matriz contendo as possíveis palavras e suas perturbações ocupa 2^n entradas, enquanto a matriz contendo a síndrome dos erros tem tamanho 2^{n-k+1} .

O código de Hamming (7, 4, 3), utilizado até o momento para a explicação dos conceitos, não é o utilizado pelo criptossistema McEliece, que, por razões de maior adaptabilidade às práticas criptográficas, utiliza os códigos da família conhecida por códigos de Goppa, uma subclasse dos códigos alternantes.

3.3 Códigos GRS

Como dito anteriormente, é possível definir um código linear por meio de sua matriz de verificação de paridade. Existem códigos em que tal matriz admite uma forma sistemática para sua construção. A seguir, apresentamos o caso dos códigos GRS, os quais admitem a construção de sua matriz de verificação de paridade por meio do produto de duas outras matrizes.

Definição 7. Dado $t > 0$ e uma sequência $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$, a matriz de

Vandermonde $\text{vdm}(t, L)$ é a matriz $t \times n$ com elementos $\text{vdm}(t, L)_{ij} = L_j^i$.

$$\text{vdm}(t, L) = \begin{bmatrix} 1 & \dots & 1 \\ L_0 & \dots & L_{n-1} \\ L_0^2 & \dots & L_{n-1}^2 \\ \vdots & \ddots & \vdots \\ L_0^{t-1} & \dots & L_{n-1}^{t-1} \end{bmatrix}.$$

Definição 8. Dada uma sequência $D = (D_0, \dots, D_{n-1}) \in \mathbb{F}_q^n$, a matriz diagonal $\text{diag}(D)$ é a matriz $n \times n$ com elementos $\text{diag}(D)_{ii} = D_i$.

$$\text{diag}(D) = \begin{bmatrix} D_0 & & & & \\ & D_1 & & & \\ & & \ddots & & \\ & & & D_{n-2} & \\ & & & & D_{n-1} \end{bmatrix}.$$

A partir destas duas matrizes, é possível construir uma matriz de verificação de paridade para um código GRS, o definindo sem ambiguidade.

Definição 9. Dada uma sequência $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$ de elementos distintos, a qual chamaremos de suporte do código, e uma sequência $D = (D_0, \dots, D_{n-1}) \in \mathbb{F}_q^n$ de elementos não nulos, o código Reed-Solomon generalizado $\text{GRS}_r(L, D)$ é o código linear corretor de erros $[n, n - r]$, sendo $n - r$ sua dimensão, definido pela matriz de paridade

$$H = \text{vdm}(r - 1, L) \cdot \text{diag}(D).$$

Os métodos de codificação e decodificação para códigos GRS são de pouca utilidade à nossa proposta, sendo omitidos nesse texto. Para o leitor que tenha interesse, sugerimos (MACWILLIAMS; SLOANE, 1977, Cap. 10). A seguir, recapitulamos a defini-

ção e as propriedades referentes aos códigos alternantes, família de códigos importante para criptossistemas baseados em códigos corretores de erros, derivada da família de códigos GRS, por meio do método de construção conhecido como *construção por traço*.

3.4 Códigos alternantes

Os códigos que primariamente nos interessam são os alternantes, os quais são definidos a partir de uma especialização dos códigos GRS. A idéia dessa especialização é selecionar a parcela desse código que também esteja presente em determinado subcorpo do corpo original. Essa especialização é conhecida como subcódigo de subcorpo e pode ser atingida por meio do processo conhecido como *construção por traço*, descrito a seguir.

Definição 10. (*Construção por traço*) Seja Γ um código construído sobre o corpo F_q e F_s um subcorpo de F_q (isto é, $s \mid q$). Seja também $m = q/s$. A partir da matriz de verificação de paridade $H^{r \times n} = H_{ij}$ do código Γ , é possível definirmos a matriz de verificação de paridade $\overline{H}$, do subcódigo que está em F_s decompondo cada elemento H_{ij} em m componentes:

$$\overline{H} = \begin{bmatrix} H_{111} & H_{121} & \cdots & H_{1n1} \\ H_{112} & H_{122} & \cdots & H_{1n2} \\ \vdots & \vdots & \ddots & \vdots \\ H_{11m} & H_{12m} & \cdots & H_{1nm} \\ \vdots & \vdots & \ddots & \vdots \\ H_{r1m} & H_{r2m} & \cdots & H_{rnm} \end{bmatrix}.$$

Este processo, quando aplicado em um código GRS resulta na criação de um código alternante. A seguir, apresentamos a definição formal desse subcódigo.

Definição 11. *Seja $\mathbb{F}_s$ um subcorpo de $\mathbb{F}_q$ (isto é, $s \mid q$), então o código alternante de $GRS_r(L, D)$ no subcorpo $\mathbb{F}_s$ é $GRS_r(L, D) \cap \mathbb{F}_s$.*

Isto significa que para chegar no código alternante, deve-se tomar um código GRS, com matriz de verificação de paridade H , definido sobre determinado corpo F_q . Além disso, F_s deverá ser um subcorpo de F_q (isto é, $s \mid q$). O subcódigo de Γ definido em F_s contém palavras de código a que respeitam a relação $Ha^t = 0$, além de ter suas componentes pertencentes a F_s , resultando no código alternante desejado.

Uma vez tendo definido os códigos alternantes, é desejável explicitar o funcionamento do seu método de decodificação capaz de corrigir possíveis erros. Para essa tarefa, será utilizado o algoritmo estendido de Euclides, o qual será explicado primariamente.

3.4.1 Algoritmo estendido de Euclides

Um dos passos mais importantes no processo de decodificação de códigos alternantes se dá por meio do algoritmo estendido de Euclides, o qual tem por objetivo encontrar o máximo divisor comum (m.d.c.) de dois inteiros ou de dois polinômios. No caso polinomial, dado $f(z)$ e $g(z)$, encontrar o m.d.c. de $f(z)$ e $g(z)$ significa encontrar o polinômio de máximo grau que divida tanto $f(z)$ quanto $g(z)$. A seguir, apresentamos sua descrição em forma de teorema, que pode ser verificado em (MACWILLIAMS; SLOANE, 1977, Cap. 12, §8).

Teorema 4. *(Algoritmo estendido de Euclides para polinômios)*

Dados dois polinômios $r_{-1}(z)$ e $r_0(z)$ com $\text{grau}(r_0) \leq \text{grau}(r_{-1})$, fazemos repetidas

divisões para obter a série de equações:

$$\begin{aligned}
 r_{-1}(z) &= q_1(z)r_0(z) + r_1(z), \text{ grau}(r_1) < \text{grau}(r_0) \\
 r_0(z) &= q_2(z)r_1(z) + r_2(z), \text{ grau}(r_2) < \text{grau}(r_1) \\
 &\dots \\
 r_{j-2}(z) &= q_j(z)r_{j-1}(z) + r_j(z), \text{ grau}(r_j) < \text{grau}(r_{j-1}) \\
 r_{j-1}(z) &= q_{j+1}(z)r_j(z)
 \end{aligned}$$

Assim, o último resto não nulo $r_j(z)$ no processo de divisão será o m.d.c. de $r_1(z)$ e $r_0(z)$.

Além disso, outro teorema também se faz importante no processo de decodificação de códigos alternantes.

Teorema 5. *Seja $r_{-1}(z)$ e $r_0(z)$ polinômios de grau $(r_0) \leq \deg(r_{-1})$ e com m.d.c $h(z)$. Então existem polinômios $U(z)$ e $V(z)$, com $\text{grau}(U)$ e $\text{grau}(V)$ menores do que o $\text{grau}(r_{-1})$, tais que:*

$$U(z)r_{-1}(z) + V(z)r_0(z) = h(z)$$

Tais polinômios U e V podem ser construídos, conforme elucidado em (MACWILLIAMS; SLOANE, 1977, Cap. 12, §8), a partir de:

$$\begin{aligned}
 U_{-1}(z) &= 0, \quad U_0(z) = 1, \\
 V_{-1}(z) &= 1, \quad V_0(z) = 0, \\
 U_i(z) &= q_i(z)U_{i-1}(z) + U_{i-2}(z), \\
 V_i(z) &= q_i(z)V_{i-1}(z) + V_{i-2}(z)
 \end{aligned}$$

3.4.2 Método de decodificação alternante

O processo de decodificação de um código alternante pode ser dividido em três etapas.

1. Cálculo da síndrome da mensagem.
2. Determinação dos polinômios localizador e avaliador de erros.
3. Localização e correção dos erros.

Conforme explicado anteriormente, para efetuar o cálculo da síndrome de uma mensagem, basta obter o resultado da multiplicação da matriz de verificação de paridade do código utilizado pelo vetor que representa a mensagem.

A segunda etapa contém o processo central desse método, baseado na determinação de dois polinômios: polinômio de localização de erros e o polinômio avaliador de erros. O primeiro polinômio, como o próprio nome sugere, é capaz de localizar a posição dos erros dentro das mensagens. O segundo polinômio é pertinente ao caso de mensagens não-binárias, conforme explicado a seguir. Enquanto no caso de mensagens binárias, ao localizar o erro, automaticamente determinamos qual o valor correto do bit (uma vez que bastaria inverter seu valor de zero para um, ou vice-versa), no caso não-binário, se faz necessário saber além da posição do erro qual o seu valor, função desempenhada pelo polinômio avaliador de erros.

A seguir, apresentamos o algoritmo capaz de gerar o polinômio localizador de erros e o polinômio avaliador de erros, conforme descrito em (MACWILLIAMS; SLOANE, 1977, Cap. 12, §9). O valor da síndrome calculada no passo anterior será representado por $S(z)$, lembrando se tratar de sua representação polinomial.

1. $r_{-1}(z) = z^r$, $r_0(z) = S(z)$
2. Executar o algoritmo estendido de Euclides até encontrar $r_k(z)$ tal que:
 $\text{grau}(r_{k-1}(z)) \geq \frac{1}{2}r$ e $\text{grau}(r_k(z)) \leq \frac{1}{2}r - 1$
3. $\sigma(z) = \delta U_k(z)$, sendo δ escolhido tal que $\sigma(0) = 1$
4. $\omega(z) = (-1)^k \delta r_k(z)$, com δ do passo anterior

Segundo (MACWILLIAMS; SLOANE, 1977, Cap. 12, §8), os polinômios ainda devem satisfazer:

$$\begin{aligned}\omega(z) &= \sigma(z)S(z) \bmod z^r \\ \text{grau}(\sigma(z)) &\leq \frac{1}{2}r \\ \text{grau}(\omega(z)) &\leq \frac{1}{2}r - 1\end{aligned}$$

Estando de posse desses dois polinômios, a terceira etapa se refere a determinar o local e o valor dos erros. Para isso, fazemos uso do suporte do código.

No caso da localização de erros, devemos avaliar o polinômio localizador de erros calculado no inverso de todos os valores contidos no suporte. A cada elemento L_i que resultar em $\sigma(\frac{1}{L_i}) = 0$, indica que existe um erro na posição i da mensagem. Ou seja, para a mensagem m contendo erros, teríamos:

$$\sigma(\frac{1}{L_i}) = 0 \Leftrightarrow m_i \text{ contém erro.}$$

Para o caso do polinômio avaliador de erros ω , segundo (MACWILLIAMS; SLOANE, 1977, Cap. 12, §9), obtemos o valor do erro Y_μ , referente à posição μ , a partir do suporte L e da diagonal D (utilizada na construção da matriz de verificação de paridade do código), calculando:

$$Y_\mu = \frac{\omega(1/L_\mu)}{D_{i_\mu} \prod_{v \neq \mu} (1 - L_v/L_\mu)}$$

A seguir, recapitulamos a subfamília de códigos alternantes conhecida como códigos de Goppa, de especial interesse à nossa proposta.

3.5 Códigos de Goppa

Os códigos de Goppa (GOPPA, 1970) são uma subfamília dos códigos alternantes. Um código de Goppa de tamanho n , com elementos em $\mathbb{F}_q$, é definido por intermédio de um polinômio $g(z)$, conhecido como *polinômio de Goppa*, tendo coeficientes em $\mathbb{F}_{q^m}$, para algum inteiro positivo m , e pelo subconjunto, conhecido como *suporte do código*, $L = \{L_1, \dots, L_n\}$ de $\mathbb{F}_{q^m}$, tal que $g(L_i) \neq 0$, para todo $L_i \in L$.

Estes códigos têm interessantes características, como o fácil cálculo de seu peso mínimo (problema de difícil resolução para outras famílias de código), o qual determina a capacidade de correção de erros de um código (ver Teorema 2) e está relacionado com o seu peso mínimo. Para o caso geral, temos:

Fato 2. *Seja t o grau do polinômio de Goppa do código Γ , o peso mínimo de Γ é maior ou igual a $t + 1$.*

Resultando na capacidade de correção de até $\frac{t}{2}$ erros. Já para o caso binário e com $g(z)$ não tendo raízes múltiplas, os códigos de Goppa apresentam uma particularidade que dobram a sua capacidade corretora, a saber:

Fato 3. *O código gerado por $\Gamma(L, g)$ binário, livre de raízes múltiplas, é o mesmo código gerado por $\Gamma(L, g^2)$.*

Como o peso mínimo do código (e consequentemente sua capacidade de correção) estão diretamente ligados ao grau do polinômio de Goppa, ao elevá-lo ao quadrado, seu grau duplica, assegurando que seu peso mínimo passa a ser maior ou igual a $2t + 1$. Desta forma, temos que:

Fato 4. *A capacidade de correção de erros de um código de Goppa binário $\Gamma(L, g)$, livre de raízes múltiplas, com o polinômio g tendo grau t é de t erros.*

A seguir, apresentamos a definição de síndrome que caracteriza os códigos de Goppa.

Definição 12. O código de Goppa $\Gamma(L, g)$ consiste de todos os vetores $v = \{v_1, \dots, v_n\}$ sobre $\mathbb{F}_q$, tais que

$$\sum_{i=1}^n \frac{v_i}{z - L_i} \equiv 0 \pmod{g(z)}.$$

Sistematizando esta relação em forma matricial, chegamos à condição de que um vetor $v \in \Gamma(L, g) \Leftrightarrow Hv_t = 0$, onde $\{g_1, \dots, g_n\}$ são os coeficientes do polinômio g e a matriz H é definida a seguir:

$$H = \begin{pmatrix} g_t g(L_0)^{-1} & \cdots & g_t g(L_{n-1})^{-1} \\ (g_{t-1} + g_t L_0)g(L_0)^{-1} & \cdots & (g_{t-1} + g_t L_{n-1})g(L_{n-1})^{-1} \\ \vdots & \ddots & \vdots \\ (\sum_{j=1}^t g_j L_0^{j-1})g(L_0)^{-1} & \cdots & (\sum_{j=1}^t g_j L_{n-1}^{j-1})g(L_{n-1})^{-1} \end{pmatrix}$$

É possível decompor esta matriz no produto de outras três, $H = TVD$, onde:

$$T = \begin{pmatrix} g_t & 0 & 0 & \cdots & 0 \\ g_{t-1} & g_t & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & g_3 & \cdots & g_t \end{pmatrix}, V = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ L_0 & L_1 & \cdots & L_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ L_0^{t-1} & L_1^{t-1} & \cdots & L_{n-1}^{t-1} \end{pmatrix},$$

$$e \quad D = \begin{pmatrix} \frac{1}{g(L_0)} & & & \\ & \frac{1}{g(L_1)} & & \\ & & \ddots & \\ & & & \frac{1}{g(L_{n-1})} \end{pmatrix}$$

A matriz V é a matriz de Vandermonde, já utilizada na definição de códigos GRS e alternantes. A matriz D segue a mesma idéia da matriz diagonal utilizada na definição de códigos GRS e alternantes, restringindo seus valores à relação: $D_{ii} = g(L_i)^{-1}$. Já a matriz T segue a estrutura matricial de Toeplitz (para detalhes, consultar (GRAY, 2005)). Esta estrutura matricial apresenta valores constantes em suas

diagonais e, neste caso, são utilizados os coeficientes do polinômio g .

Apesar dessa sistemática forma de construir a matriz de verificação de paridade para códigos de Goppa, existe uma alternativa que se assemelha ainda mais com a matriz de verificação de paridade para códigos alternantes. Partindo da interessante propriedade de códigos lineares de que H e H' geram o mesmo código, quando $H = CH'$, com C sendo uma matriz inversível, podemos reescrever a matriz de verificação de paridade para códigos de Goppa da forma:

$$H = \underbrace{\begin{bmatrix} 1 & \dots & 1 \\ L_0 & \dots & L_{n-1} \\ L_0^2 & \dots & L_{n-1}^2 \\ \vdots & \ddots & \vdots \\ L_0^{t-1} & \dots & L_{n-1}^{t-1} \end{bmatrix}}_V \underbrace{\begin{bmatrix} \frac{1}{g(L_0)} & & & \\ & \frac{1}{g(L_1)} & & \\ & & \ddots & \\ & & & \frac{1}{g(L_{n-2})} \\ & & & & \frac{1}{g(L_{n-1})} \end{bmatrix}}_D$$

Repare que descevendo a matriz H desta forma $H = VD$, ou seja, da forma da matriz de verificação de paridade de códigos alternantes, apenas impondo a restrição de que os valores da diagonal da matriz D sejam: $D_i = g(L_i)^{-1}$, explicitamos claramente que os códigos de Goppa são uma subfamília de códigos alternantes.

Tendo definido os códigos de Goppa e como representá-los, nos resta explicitar como corrigir os erros deste código. Sabendo se tratar de uma subfamília de códigos alternantes, podemos utilizar o método de decodificação alternante explicado em 3.4.2. Vale ressaltar que este método é capaz de corrigir até $\lfloor \frac{t-1}{2} \rfloor$, onde t é o peso mínimo do código (ver seção 2) e o grau do polinômio de Goppa.

Para o caso binário, onde o polinômio de Goppa é livre de raízes múltiplas, sabemos que seu peso mínimo dobra, permitindo a correção de t erros (por exemplo, aplicando o método de decodificação alternante ao código $\Gamma(L, g^2)$, ao invés de $\Gamma(L, g)$). Entretanto, existe um método de correção de erros que se baseia apenas em $\Gamma(L, g)$ e

mesmo assim corrige t erros (PATTERSON, 1975), sendo explicada a seguir.

3.5.1 Método de decodificação de Patterson

Assim como no método de decodificação alternante, podemos dividir em três etapas a tarefa de decodificar mensagens com o método de Patterson, a saber:

1. Cálculo da síndrome da mensagem.
2. Determinação do polinômio localizador de erros.
3. Localização e correção dos erros.

A primeira etapa é realizada multiplicando a matriz de verificação de paridade pelo vetor que representa a mensagem. A segunda etapa contém a principal diferença desse método. A seguir, apresentamos o algoritmo de Patterson para a geração do polinômio localizador de erros σ , capaz de corrigir t erros de um código de Goppa binário, com polinômio de Goppa livre de raízes múltiplas.

Algoritmo 1 Método de decodificação de Patterson

INPUT: $s(x)$ (síndrome da mensagem), $g(x)$ (polinômio de goppa).

OUTPUT: Polinômio localizador de erros $\sigma(x)$.

```

1:  $v(x) \leftarrow \sqrt{x+1}/s(x) \bmod g(x)$  ▶ Este passo pode ser executado com o algoritmo estendido de Euclides.
2:  $F \leftarrow v, G \leftarrow g, B \leftarrow 1, C \leftarrow 0, t \leftarrow \text{grau}(g)$ 
3: while  $\text{grau}(G) > \lfloor t/2 \rfloor$  do
4:    $F \leftrightarrow G, B \leftrightarrow C$ 
5:   while  $\text{grau}(F) \geq \text{grau}(G)$  do
6:      $j \leftarrow \text{grau}(F) - \text{grau}(G), h \leftarrow F_{\text{grau}(F)}/G_{\text{grau}(G)}$ 
7:      $F \leftarrow F - hx^jG, B \leftarrow B - hx^jC$ 
8:   end while
9: end while
10:  $\sigma(x) \leftarrow G^2(x) + xC^2(x)$ 

```

Conforme explicado anteriormente, de posse do polinômio localizador de erros é possível encontrar quais são as posições que contêm erro na mensagem. Como um dos pré-requisitos para a utilização do método de Patterson se trata do código utilizado ser binário, ao localizar a posição do erro, automaticamente determinamos seu valor correto (bastando inverter o valor do bit errado).

Para a determinação da posição do erro na mensagem, o raciocínio é bastante parecido com o utilizado no método de decodificação alternante, avaliando o polinômio em valores relacionados ao suporte. No caso, avaliamos o polinômio localizador de erros σ , calculado no valor do suporte. Cada elemento L_i que resultar em $\sigma(L_i) = 0$ indica que existe um erro na posição i da mensagem. Ou seja, para a mensagem m contendo erros, teríamos:

$$\sigma(L_i) = 0 \Leftrightarrow m_i \text{ contém erro.}$$

3.6 Criptossistema McEliece

O criptossistema McEliece é um sistema de chaves assimétricas que representa uma interessante alternativa às soluções criptográficas convencionais. Sua segurança está relacionada à tarefa de decodificar códigos lineares genéricos, problema NP-difícil quando não são conhecidas as características do código utilizado.

Resumidamente, sua ideia central é de selecionar um código linear que tenha um método de decodificação eficiente e disfarçá-lo como sendo um código linear genérico. Desta forma, apenas os que tiverem o conhecimento da forma original do código (a qual dispõe de um método de decodificação eficiente) serão capazes de utilizá-lo em sua plenitude, implicando em um eficiente esquema de chaves assimétricas: a chave pública é o código disfarçado e a chave privada é o código original.

Diferentes famílias de códigos lineares podem ser utilizadas para instanciar criptossistemas baseados na decodificação de códigos lineares aleatórios (NIEDERREITER, 1986; GABIDULIN; PARAMONOV; TRETJAKO, 1991; SIDELNIKOV, 1994; JANWA; MORENO, 1996; SHOKROLLAHI; MONICO; ROSENTHAL, 2000). A proposta original do criptossistema McEliece sugere a utilização dos códigos de Goppa. Propostas baseadas em outras famílias de código têm demonstrado gerarem fraquezas aos criptossis-

temas, como pode ser verificado em (SIDELNIKOV; SHESTAKOV, 1992b; GIBSON, 1995; MINDER; SHOKROLLAHI, 2007; OVERBECK, 2008).

3.6.1 Algoritmos

A seguir, recapitulamos a definição dos algoritmos do criptossistema McEliece, em sua versão convencional.

3.6.1.1 Geração de chaves

O algoritmo de geração de chaves pode ser dividido em duas etapas: A primeira se refere à escolha do código a ser utilizado e à representação desse código através de uma matriz geradora, já a segunda, à tarefa de disfarçar este código.

1. Escolha do Código:

- (a) Escolher uma sequência aleatória secreta $L \in (\mathbb{F}_{2^m})^n$ de elementos distintos e um polinômio secreto $g(X) \in \mathbb{F}_{2^m}[X]$ de grau t tal que $g(L) \neq 0$.
- (b) Construir o Código de Goppa $\Gamma = (L, g(X))$ capaz de corrigir até t erros.
- (c) Construir a matriz de paridade $t \times n$ decodificável para esse código, na forma $H_0 = TVD$ (Definição 12).

2. Disfarce do Código:

- (a) Escalonar a matriz de paridade binária H_0 , resultando na matriz $H = [M^t \mid I]$, que também se refere ao código Γ (mas em formato não decodificável).
- (b) Obter a matriz geradora $G = [I \mid M] \in (\mathbb{F}_2)^{k \times n}$ do subcódigo binário de Γ .

Assim, as chaves seriam:

- Chave Privada: $(L, g(X))$ (e, implicitamente, H_0);
- Chave Pública: (G, t) .

3.6.1.2 Cifração

Para a cifração de uma mensagem $m \in \mathbb{F}_2^k$ em um vetor $c \in \mathbb{F}_2^n$:

1. Calcular $m' = mG$.
2. Gerar um vetor uniformemente aleatório $e \in \mathbb{F}_2^n$ de peso t , e somá-lo a m' , obtendo assim o vetor $c = m' + e$.

3.6.1.3 Decifração

Para a decifração de um criptograma $c \in \mathbb{F}_2^n$, temos os seguintes passos:

1. Calcular a síndrome $s^t = H_0 c^t = H_0 e^t$.
2. Decodificar a síndrome s , obtendo o erro e .
3. Extrair a mensagem m das k primeiras colunas de $c + e$.

3.6.2 Segurança

A segurança da versão original do criptossistema McEliece está baseada em dois principais pressupostos, a saber:

1. Dificuldade de decodificação de um código linear genérico.
2. Indistinguibilidade dos códigos de Goppa.

O primeiro é um problema antigo da teoria da codificação, para o qual apenas soluções exponenciais são conhecidas (BERLEKAMP; MCELIECE; TILBORG, 1978; BARG, 1998). Já o segundo ainda não pode ser classificado como um problema extensivamente estudado, entretanto, está relacionado a problemas antigos da teoria da codificação algébrica, levando à crença de também ser válido (BERNSTEIN; BUCHMANN; DAHMEN, 2008).

Atualmente, tal proposta permanece inquebrada (apesar de em (CANTEAUT; SENDRIER, 1998), terem sido demandados ajustes em seus parâmetros originais). Vale ressaltar que esta análise é feita considerando também a possibilidade de ataques provenientes de computadores quânticos, característica que falta aos criptossistemas convencionalmente empregados.

Na Tabela 3, baseada em (BERNSTEIN; BUCHMANN; DAHMEN, 2008), explicitamos o esforço de quebra do criptossistema McEliece, tanto a partir de computadores clássicos, quanto a partir de computadores quânticos, demonstrando sua robustez em ambos os cenários. Os parâmetros m e t são relativos ao código de Goppa utilizado, sendo m a extensão do corpo finito onde o código está baseado e t o grau do polinômio de Goppa deste código. As colunas *custo clássico* e *custo quântico* apresentam o número de passos computacionais necessários para a quebra nas respectivas plataformas e a última coluna indica em qual nível padrão de segurança o criptossistema McEliece se enquadraria.

Tabela 3: Tabela de esforço para a quebra clássica e quântica

m,t	Custo clássico	Custo quântico	Nível de segurança
11,32	2^{91}	2^{86}	80
11,40	2^{98}	2^{94}	88
12,22	2^{93}	2^{87}	80
12,45	2^{140}	2^{133}	128

3.6.2.1 Segurança semântica

Apesar de considerado um sistema seguro, a utilização direta do McEliece apresenta fragilidades. A seguir, descrevemos um cenário indesejado para um criptossistema e que demanda uma camada extra de cuidados para viabilizar sua utilização prática.

Da definição 3.6.1, sabemos que a matriz geradora G do código utilizado se apresenta em forma escalonada. Desta forma, quando da sua utilização para a cifração de uma mensagem m , teríamos que o vetor resultante $c = mG + e$ deixaria transparecer a maior parte dos bits de m , que são visíveis nas primeiras k colunas de c (com alguns erros), como ilustra a Figura 3.

Figura 3: Fragilidade semântica do criptossistema McEliece puro.

Isso torna a descrição básica de McEliece inaceitável enquanto criptossistema. Vale ressaltar que em muitos textos a melhor descrição para o mecanismo de McEliece seria, na verdade, uma função de mão única com alçapão (FMUA).

Para transformá-la num criptossistema completo, semanticamente seguro, é necessário integrá-lo num esquema do tipo (FUJISAKI; OKAMOTO, 1999), que essencialmente pode ser descrito como:

1. Usa a FMUA para cifrar uma cadeia aleatória.
2. Deriva uma chave simétrica a partir dessa cadeia com uma função de *hash* apropriada (de modo que, restando um erro sequer na cadeia, a chave simétrica correta não poderá ser obtida).
3. Usa essa chave simétrica para cifrar e autenticar a mensagem propriamente dita.

3.6.3 Eficiência algorítmica

Além de ser seguro em cenários clássico e quântico, tal criptossistema ainda provê ganho de eficiência algorítmica se comparado aos criptossistemas RSA e os baseados em curvas elípticas. Em (CANTEAUT; SENDRIER, 1998), estima-se que o criptossistema McEliece seja cerca de 50 vezes mais eficiente no processo de cifração e cerca de 100 vezes mais eficiente no processo de decifração, quando comparado ao RSA-1024 (levando-se em conta o número de operações binárias necessárias para processar um bit de informação).

Muito desse ganho é de fácil entendimento, uma vez que os passos computacionais executados pelo criptossistema McEliece nesses processos, em boa parte, se resumem a multiplicações matriz por vetor, de custo computacional quadrático (enquanto que as operações do RSA/ECC atingem ordem cúbica).

3.6.4 Tamanho das chaves criptográficas

Porém, nem todas as particularidades deste sistema são benéficas. O tamanho de suas chaves, representadas por grandes matrizes, sempre foi um considerável problema quando comparadas às chaves do RSA e ECC. Este tem sido o fator preponderante para sua não empregabilidade em larga escala. Para se ter uma ideia dessa magnitude, o criptossistema McEliece utilizando códigos de Goppa binário demandam cerca de 58KB para sua chave pública (BERNSTEIN; LANGE; PETERS, 2008) e para assinaturas seguras, cerca de 720KB (FINIASZ; SENDRIER, 2009), para um mesmo nível de segurança do RSA-1024.

Levando-se em conta este problema, aliado com a necessidade de termos à disposição soluções pós-quânticas, fica evidente a extrema importância em buscarmos formas de reduzir o tamanho das chaves destes criptossistemas, mantendo-se o nível de segurança.

Nesse sentido, os primeiros passos focando este objetivo foram dados em (MONICO; ROSENTHAL; SHOKROLLAHI, 2000), utilizando códigos de paridade de baixa densidade (também conhecidos pela sigla *LDPC*, do inglês *Low density parity check*), em (GABORIT, 2005), usando códigos quase-cíclicos, e em (BALDI; CHIARALUCE, 2007), usando uma combinação de ambos.

No primeiro, no próprio texto, os autores não recomendam seu uso, dadas as fragilidades inerentes ao modelo proposto. Nas duas outras propostas, o alçapão capaz de decodificar e corrigir mensagens é protegido essencialmente por nada mais que uma permutação privada do código. No entanto, descobriu-se que essas propostas apresentam vulnerabilidades (OTMANI; TILLICH; DALLOT, 2008).

A estratégia de ataque consiste em obter um sistema linear solúvel, tal que os componentes da matriz de permutação devem satisfazer. Como esta permutação secreta deve preservar a estrutura quase-cíclica, apresentando uma variedade muito restrita de permutações, e aliada ao fato de que o código secreto é um subcódigo de um código público, tal estratégia tem êxito.

Uma correção dos problemas em (BALDI; CHIARALUCE, 2007) é proposta em (BALDI; CHIARALUCE; BODRATO, 2008). Mais recentemente, (BERGER et al., 2009) mostraram como contornar as desvantagens do esquema original de (GABORIT, 2005) e eliminar os pontos fracos indicados em (OTMANI; TILLICH; DALLOT, 2008) por meio de duas técnicas:

1. Trabalhar com subcódigos definidos sobre um subcorpo intermediário entre o corpo-base e a extensão adotada pelo código original.
2. Extrair códigos públicos encurtados por blocos a partir de códigos privados muito longos.

Estas duas técnicas foram aplicadas com sucesso aos códigos quase-cíclicos. A

primeira veio a se mostrar inefetiva para proteger o código privado. Já a segunda baseia-se no seguinte teorema:

Teorema 6. (WIESCHEBRINK, 2006) *Distinguir um código encurtado em blocos de seu código original é um problema NP-completo.*

A aplicabilidade desta última técnica não se restringe somente à classe de códigos quase-cíclicos, como veremos no capítulo 4.

3.7 Resumo

Neste capítulo, recapitulamos os conceitos básicos da teoria da codificação, área em que se apoia a segurança de criptossistemas baseados em códigos. Um código linear pode ser representado por uma matriz geradora, possibilitando codificar uma mensagem ao multiplicá-la por esta matriz. O peso mínimo de um código é o menor número de componentes não nulo de uma palavra desse código. Um código linear de peso mínimo d pode corrigir até $\lfloor \frac{d-1}{2} \rfloor$ erros. Existem diversos métodos para correção de erros de códigos lineares.

Outra maneira de definir um código linear Γ é por intermédio da matriz de verificação de paridade, a qual gera o código dual de Γ . Ao multiplicá-la por uma mensagem, o resultado é conhecido como síndrome da mensagem e as palavras de código de Γ devem apresentar síndrome nula. A família de códigos $GRS(L, D)$ é definida por sua matriz de verificação de paridade, obtida do produto de uma matriz de Vandermonde, baseada em uma sequência $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$, por uma matriz diagonal, baseada em uma sequência $D = (D_0, \dots, D_{n-1}) \in \mathbb{F}_q^n$.

Utilizando o método de construção por traço, pode-se obter a partir da matriz verificação de paridade dos códigos GRS a subfamília de código conhecida como alternante. De especial interesse à nossa proposta, temos a subfamília dos códigos alternantes conhecida como códigos de Goppa. Tais códigos são definidos impondo restrições

na construção da matriz de verificação de paridade alternante relacionadas a um polinômio, conhecido como polinômio de Goppa. Esta família de código, em seu caso binário, tem a interessante característica de possibilitar corrigir até o dobro dos erros que um código alternante.

Baseado em códigos de Goppa, o criptossistema McEliece se apresenta como uma interessante alternativa aos criptossistemas convencionais. Além de resistente perante ataques de computadores quântico, este criptossistema é bastante eficiente. Entretanto, como principal fator para o seu não emprego em larga escala, temos o grande tamanho de suas chaves criptográficas.

Recentes propostas visaram minimizar esse problema. Uma interessante estratégia se refere a extrair códigos públicos encurtados por blocos a partir de códigos privados muito longos, originalmente proposta para códigos quase-cíclicos. Esta estratégia se mostra bastante promissora e não está restrita somente a esta família de código, conforme veremos no capítulo a seguir.

4 CÓDIGOS DE GOPPA QUASE-DIÁDICOS

A seguir, descrevemos uma maneira de reduzir significativamente o tamanho das chaves do criptossistema McEliece, além de melhorar a eficiência das operações criptográficas para tempo $\tilde{O}(n)$, notação que omite fatores logarítmicos. Para isso, apresentamos uma nova subclasse dos códigos de Goppa: os códigos de Goppa quase-diádicos.

Esta proposta foi submetida e selecionada para publicação e apresentação no *XVI Workshop Selected Areas in Cryptography* (SAC'2009), ocorrido entre 13 e 14 de Agosto de 2009, na cidade de Calgary, Alberta, Canadá. Este evento foi realizado em cooperação com a *International Association for Cryptologic Research* (IACR) e teve seus artigos publicados pela *Springer* em *Lecture Notes in Computer Science*, volume 5867. A referência para a publicação de nossa proposta é (MISOCZKI; BARRETO, 2009).

Ao final deste capítulo, espera-se que o leitor tenha compreendido o processo de construção de códigos de goppa quase-diádicos, sua segurança, as implicações de seu uso junto ao sistema criptográfico McEliece, além do entendimento dos conceitos teóricos em que está fundamentada nossa proposta.

4.1 Conceitos algébricos

A seguir, apresentamos os conceitos algébricos necessários ao entendimento da nossa proposta. No que segue, todos os índices de vetores e de matrizes são numerados de zero em diante.

Definição 13. Dado um anel $\mathcal{R}$ e um vetor $h = (h_0, \dots, h_{n-1}) \in \mathcal{R}^n$, a matriz diádica $\Delta(h) \in \mathcal{R}^{n \times n}$ é a matriz simétrica com componentes $\Delta_{ij} = h_{i \oplus j}$, onde $\oplus$ denota a operação de ou-exclusivo, aplicada bit a bit sobre as representações binárias dos índices.

Quando n é uma potência de 2, pode-se caracterizar recursivamente uma matriz diádica: qualquer matriz 1×1 é diádica, e para $k > 0$ qualquer matriz diádica M de dimensão $2^k \times 2^k$ tem a forma

Figura 4: Estrutura diádica.

onde A e B são matrizes diádicas $2^{k-1} \times 2^{k-1}$.

A Figura 5 expande essa estrutura, a fim de facilitar a compreensão do leitor.

Definição 14. A sequência $h = (h_0, \dots, h_{n-1}) \in \mathcal{R}^n$, que define a matriz diádica $\Delta(h) \in \mathcal{R}^{n \times n}$, é chamada de assinatura de $\Delta(h)$.

A assinatura h de uma matriz diádica coincide com a sua primeira linha, como pode ser verificado no exemplo a seguir.

Figura 5: Matriz diádica.

Exemplo 6. Para uma matriz diádica $\Delta(h) \in \mathcal{R}^{4 \times 4}$, com $h = (1, 2, 3, 4) \in \mathcal{R}^4$, temos:

$$\Delta(h) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

□

Assim, uma maneira bastante enxuta de representarmos matrizes diádicas é por meio de sua assinatura, reduzindo a memória necessária de custo quadrático $O(n^2)$ na forma matricial, para custo linear $O(n)$.

Definição 15. O conjunto de matrizes diádicas $n \times n$ sobre $\mathcal{R}$ denota-se $\Delta(\mathcal{R}^n)$.

Um fato importante acerca dessas matrizes se refere ao fato de formarem um subanel comutativo de $\mathcal{R}^{n \times n}$, sempre que $\mathcal{R}$ for comutativo (GULAMHUSEIN, 1973).

Definição 16. Dado $t > 0$, $\Delta(t, h)$ denota $\Delta(h)$ truncada às primeiras t linhas.

Exemplo 7. Utilizando a matriz diádica $\Delta(h) \in \mathcal{R}^{4 \times 4}$ do exemplo anterior, $\Delta(2, h)$

seria:

$$\Delta(2, h) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}$$

□

Definição 17. Uma permutação diádica é uma matriz diádica $\Pi^i \in \Delta(\{0, 1\}^n)$ cuja assinatura é a i -ésima linha da matriz identidade de dimensões $n \times n$.

Exemplo 8. Para $n = 4$ e $i = 3$, ou seja, $\Pi^3 \in \Delta(\{0, 1\}^4)$, temos:

$$\Pi^3 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

□

Fato 5. Uma permutação diádica é uma involução, ou seja:

$$(\Pi^i) * (\Pi^i) = I$$

Exemplo 9. A partir da permutação diádica Π^3 do último exemplo, temos:

$$(\Pi^3) * (\Pi^3) = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} * \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I$$

□

Fato 6. A i -ésima linha (ou equivalentemente a i -ésima coluna) de uma matriz diádica definida por uma assinatura h pode ser escrita como $\Delta(h)_i = h\Pi^i$.

Exemplo 10. Selecionando a terceira linha da matriz do Exemplo 6: $(3, 4, 1, 2)$,

comprovamos que:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \end{pmatrix} * \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 3 & 4 & 1 & 2 \end{pmatrix}$$

□

Definição 18. *Uma matriz quase-diádica é uma matriz (possivelmente não-diádica) de blocos, tais que estes blocos componentes são submatrizes diádicas.*

A seguir, apresentamos a Figura 6, ilustrando essa estrutura matricial.

Figura 6: Matriz quase-diádica.

Matrizes quase-diádicas estão no centro de nossa proposta. Neste texto, tratamos principalmente do caso $\mathcal{R} = \mathbb{F}_q$, o corpo finito com q elementos, sendo q uma potência de um número primo. A seguir, recapitulamos outra matriz relevante ao entendimento de nossa proposta.

Definição 19. *Dadas duas sequências disjuntas $z = (z_0, \dots, z_{t-1}) \in \mathbb{F}_q^t$ e $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$ de elementos distintos, a matriz de Cauchy $C(z, L)$ é a matriz $t \times n$ com elementos $C_{ij} = 1/(z_i - L_j)$, ou seja,*

$$C(z, L) = \begin{pmatrix} \frac{1}{z_0 - L_0} & \cdots & \frac{1}{z_0 - L_{n-1}} \\ \vdots & \ddots & \vdots \\ \frac{1}{z_{t-1} - L_0} & \cdots & \frac{1}{z_{t-1} - L_{n-1}} \end{pmatrix}$$

Matrizes de Cauchy admitem uma representação muito compacta, bastando armazenar as sequências $z = (z_0, \dots, z_{t-1}) \in \mathbb{F}_q^t$ e $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$, ao invés de toda a matriz. Desta forma, assim como as matrizes diádicas, seu custo de armazenamento cai de $O(n^2)$ para $O(n)$.

Além disso, matrizes de Cauchy têm a propriedade de que todas as suas submatrizes são não-singulares (SCHECHTER, 1959). Assim, para cada matriz $C(z, L)$, existe uma matriz $C(z, L)^{-1}$, tal que:

$$C(z, L) * C(z, L)^{-1} = I$$

4.2 Códigos de Goppa em forma de Cauchy e em forma diádica

Em geral, matrizes diádicas não são de Cauchy e vice-versa. Entretanto, a intersecção dessas duas classes não é vazia em característica 2. Além disso, os códigos de Goppa possuem uma propriedade fundamental para a nossa proposta, a saber, eles admitem uma matriz de paridade na forma de Cauchy:

Teorema 7. (TZENG; ZIMMERMANN, 1975) *O código de Goppa gerado por um polinômio mônico $g(x) = (x - z_0) \dots (x - z_{t-1})$ sem raízes múltiplas admite uma matriz de paridade da forma $H = C(z, L)$, i.e. $H_{ij} = 1/(z_i - L_j)$, $0 \leq i < t$, $0 \leq j < n$.*

Este teorema, que também aparece em (MACWILLIAMS; SLOANE, 1977, Cap. 12, §3, Pr. 5), é inteiramente geral quando se considera a fatoração do polinômio de Goppa sobre o seu corpo de decomposição, em cujo caso, uma única raiz de g basta para caracterizar completamente o código. Por simplicidade, restringiremos nossa atenção ao caso em que todas as raízes daquele polinômio estão no próprio $\mathbb{F}_q$.

Com base no Teorema 7 e somando o fato do criptossistema McEliece, em sua inquebrada versão original, ter suas chaves criptográficas baseadas na representação

de códigos de Goppa, uma interessante estratégia para diminuir o espaço necessário para representá-las seria utilizando matrizes de Cauchy. Dessa forma, ao invés de existir a necessidade de armazenar toda a matriz referente ao código de Goppa, apenas as sequências z e L seriam necessárias para sua definição.

Porém, o fato de poder representar códigos de Goppa por intermédio dessas matrizes não é o suficiente para sua aplicação junto ao criptossistema McEliece de maneira a obter ganhos no tamanho de suas chaves. Para ter alguma vantagem em sua utilização, também seria necessário poder operar sobre essas matrizes sem que sua estrutura de Cauchy fosse perdida, o que, infelizmente, não é possível. Entretanto, como ressaltamos em 4.1, matrizes diádicas formam um subanel comutativo, permitindo operar sobre ela sem perda de sua estrutura diádica.

Sabendo desse fato, uma interessante alternativa para representar códigos de Goppa seria por meio da combinação de matrizes de Cauchy e diádicas, possibilitando uma grande economia no espaço em memória para sua representação.

4.2.1 Construção de um código de Goppa binário em forma diádica

Mostraremos a seguir como construir um código de Goppa binário que admite uma matriz de verificação de paridade em forma diádica. Para isso, buscamos uma maneira de construir matrizes que respeitassem tanto as restrições diádicas quanto as de Cauchy. O teorema seguinte caracteriza todas as matrizes que satisfazem essas duas restrições.

Teorema 8. *Seja $H \in \mathbb{F}_q^{n \times n}$, com $n > 1$, uma matriz simultaneamente diádica, $H = \Delta(h)$ para algum $h \in \mathbb{F}_q^n$, e de Cauchy, $H = C(z, L)$ para duas sequências disjuntas $z \in \mathbb{F}_q^n$ e $L \in \mathbb{F}_q^n$ de elementos distintos. Então $\mathbb{F}_q$ tem característica 2, h satisfaz*

$$\frac{1}{h_{i \oplus j}} = \frac{1}{h_i} + \frac{1}{h_j} + \frac{1}{h_0}, \quad (4.1)$$

e $z_i = 1/h_i + \omega$, $L_j = 1/h_j + 1/h_0 + \omega$ para algum $\omega \in \mathbb{F}_q$.

Demonstração. Uma vez que uma matriz diádica é simétrica, e levando em conta a restrição de Cauchy $H_{ij} = 1/(z_i - L_j)$, as sequências que a definem devem satisfazer:

$$\frac{1}{(z_i - L_j)} = \frac{1}{(z_j - L_i)}$$

Logo $L_j = z_i + L_i - z_j$ deve ser válido todo i e j . Assim, $z_i + L_i$ deve ser uma constante α , e tomando $i = 0$, em particular, podemos simplificar para $L_j = \alpha - z_j$. Substituindo na definição $H_{ij} = 1/(z_i - L_j)$ observa-se que:

$$H_{ij} = 1/(z_i + z_j + \alpha)$$

Entretanto, matrizes diádicas também possuem a diagonal constante, a saber, $H_{ii} = 1/(2z_i + \alpha) = h_0$. Isto só é possível se todos os z_i forem iguais (contradizendo a definição de matriz de Cauchy), ou então se a característica do corpo for 2, conforme estipulado.

Nesse caso, observa-se que $\alpha = 1/h_0$, por conseguinte:

$$H_{ij} = \frac{1}{(z_i + z_j + 1/h_0)}$$

Substituindo de volta na definição $H_{ij} = h_{i \oplus j}$, obtém-se:

$$\frac{1}{H_{ij}} = \frac{1}{h_{i \oplus j}} = z_i + z_j + \frac{1}{h_0}$$

Tomando $j = 0$, em particular, temos $1/h_i = z_i + z_0 + 1/h_0$, ou simplesmente $z_i = 1/h_i + 1/h_0 + z_0$. Substituindo mais uma vez obtém-se, como esperado:

$$\begin{aligned} \frac{1}{h_{i \oplus j}} &= z_i + z_j + \frac{1}{h_0} \\ &= \frac{1}{h_i} + \frac{1}{h_0} + z_0 + \frac{1}{h_j} + \frac{1}{h_0} + z_0 + \frac{1}{h_0} \\ &= \frac{1}{h_i} + \frac{1}{h_j} + \frac{1}{h_0} \end{aligned}$$

Finalmente, definindo $\omega = 1/h_0 + z_0$ e substituindo nas relações $z_i = 1/h_i + 1/h_0 + z_0$ e $L_j = \alpha - z_j$ acima, obtém-se, conforme desejado:

$$z_i = 1/h_i + \omega$$

$$L_j = 1/h_j + 1/h_0 + \omega$$

□

Assim, precisamos de um método para resolver a Equação 4.1. A técnica que propomos consiste em simplesmente escolher aleatoriamente os valores h_0 e h_i , todos distintos e não nulos, com i varrendo todas as potências de dois menores que n , e definir todos os demais valores como:

$$h_{i+j} \leftarrow \frac{1}{\frac{1}{h_i} + \frac{1}{h_j} + \frac{1}{h_0}}$$

para $0 < j < i$ (de modo que $i + j = i \oplus j$), desde que esse valor não acarrete em interseções entre os conjuntos z e L , uma vez que devem ser disjuntos. Para isso, devemos ter, para todo j e i :

$$z_i \neq L_j$$

$$1/h_i + \omega \neq 1/h_j + 1/h_0 + \omega$$

$$h_j \neq \frac{1}{(1/h_i + 1/h_0)}$$

Esta preocupação é devido ao fato das sequências z e L precisarem ser disjuntas, uma vez que a matriz de verificação de paridade a ser construída deverá satisfazer a restrição de Cauchy $H_{ij} = 1/(z_i + L_j)$.

O Algoritmo 2 retrata essa ideia. O tempo de execução é de $O(n)$ passos, uma vez que cada elemento da assinatura h recebe um valor exatamente uma vez. A notação $u \stackrel{s}{\leftarrow} U$ significa que a variável u é escolhida uniformemente ao acaso a partir do conjunto U . Por conveniência, define-se também a *essência* de h como sendo a sequência

$\eta_s = 1/h_{2^s} + 1/h_0$ para $s = 0, \dots, \lceil \lg n \rceil - 1$ juntamente com $\eta_{\lceil \lg n \rceil} = 1/h_0$, de modo que:

$$\frac{1}{h_i} = \eta_{\lceil \lg n \rceil} + \sum_{k=0}^{\lceil \lg n \rceil - 1} i_k \eta_k$$

para $i = \sum_{k=0}^{\lceil \lg n \rceil - 1} i_k 2^k$.

Algoritmo 2 Construção de um código de Goppa binário em forma diádica

INPUT: q (uma potência de 2), $n \leq q/2$, t .

OUTPUT: Suporte L , polinômio gerador g , matriz diádica de paridade H para um código de Goppa binário $\Gamma(L, g)$ de comprimento n e distância de projeto $2t + 1$ sobre $\mathbb{F}_q$, e a essência η da assinatura de H .

```

1:  $U \leftarrow \mathbb{F}_q \setminus \{0\}$ 
2: ▶ Escolha uma assinatura diádica  $(h_0, \dots, h_{n-1})$ .
3:  $h_0 \xleftarrow{\$} U$ 
4:  $\eta_{\lceil \lg n \rceil} \leftarrow 1/h_0$ 
5:  $U \leftarrow U \setminus \{h_0\}$ 
6: for  $s \leftarrow 0$  to  $\lceil \lg n \rceil - 1$  do
7:    $i \leftarrow 2^s$ 
8:    $h_i \xleftarrow{\$} U$ 
9:    $\eta_s \leftarrow 1/h_i + 1/h_0$ 
10:   $U \leftarrow U \setminus \{h_i, 1/(1/h_i + 1/h_0)\}$ 
11:  for  $j \leftarrow 1$  to  $i - 1$  do
12:     $h_{i+j} \leftarrow 1/(1/h_i + 1/h_j + 1/h_0)$ 
13:     $U \leftarrow U \setminus \{h_{i+j}, 1/(1/h_{i+j} + 1/h_0)\}$ 
14:  end for
15: end for
16:  $\omega \xleftarrow{\$} \mathbb{F}_q$ 
17: ▶ Monte o polinômio gerador de Goppa:
18: for  $i \leftarrow 0$  to  $t - 1$  do
19:    $z_i \leftarrow 1/h_i + \omega$ 
20: end for
21:  $g(x) \leftarrow \prod_{i=0}^{t-1} (x - z_i)$ 
22: ▶ Calcule o suporte:
23: for  $j \leftarrow 0$  to  $n - 1$  do
24:    $L_j \leftarrow 1/h_j + 1/h_0 + \omega$ 
25: end for
26:  $h \leftarrow (h_0, \dots, h_{n-1})$ 
27:  $H \leftarrow \Delta(t, h)$ 
28: return  $L, g, H, \eta$ 

```

Vale ressaltar que sempre que h_j , com $j > 0$, é definido, são extraídos de U o valor h_j , evitando elementos repetidos na assinatura h . Além disso, o elemento $1/(1/h_i + 1/h_0)$ também é removido a cada interação, evitando intersecções entre os conjuntos z e L , conforme explicado anteriormente.

Uma vez demonstrada a possibilidade de ser possível a construção de tal código, se faz necessário estimar a quantidade de possíveis códigos de Goppa em forma diádica.

Caso este fosse um número restrito, sua aplicação criptográfica seria questionada e ataques se aproveitando desse fato rapidamente surgiriam. Entretanto, este não é o caso, como demonstra o teorema 9.

Teorema 9. *O Algoritmo 2 produz até $\prod_{i=0}^{\lceil \lg n \rceil} (q - 2^i)$ códigos de Goppa em forma diádica, onde q é uma potência de 2.*

Demonstração. Cada assinatura diádica produzida pelo Algoritmo 2 é inteiramente determinada pelos valores h_0 e h_{2^s} escolhidos nos passos 3 e 8, para $s = 0, \dots, \lceil \lg n \rceil - 1$ (variando ω , só produziremos códigos equivalentes). Ao longo do laço na linha 6, exatamente $2i = 2^{s+1}$ elementos são apagados de U , correspondendo às escolhas de $h_{2^s} \dots h_{2^{s+1}-1}$. No final do laço, $2 + 2 \sum_{\ell=0}^s 2^\ell = 2^{s+2}$ elementos são apagados no total. Assim, no início de cada passo do laço, apenas 2^{s+1} elementos terão sido apagados de U , ou seja, há $q - 2^{s+1}$ elementos em U dentre os quais se pode escolher h_{2^s} , e $q - 1$ possibilidades para h_0 . Assim, a construção produz até $(q - 1) \prod_{s=0}^{\lceil \lg n \rceil - 1} (q - 2^{s+1}) = \prod_{i=0}^{\lceil \lg n \rceil} (q - 2^i)$ códigos.

□

O Teorema 9 estabelece, na verdade, o número de essências distintas de assinaturas diádicas correspondentes a matrizes de Cauchy. As raízes do polinômio de Goppa são completamente especificadas pelos primeiros $\lceil \lg t \rceil$ elementos da essência η e por $\eta_{\lceil \lg n \rceil}$, a saber, $z_i = \eta_{\lceil \lg n \rceil} + \sum_{k=0}^{\lceil \lg t \rceil - 1} i_k \eta_k$, ignorando o termo ω que está implícito na escolha de $\eta_{\lceil \lg n \rceil}$. Nota-se que qualquer permutação dos elementos $\eta_0, \dots, \eta_{\lceil \lg t \rceil - 1}$ da essência apenas muda a ordem dessas raízes. Uma vez que o próprio polinômio de Goppa é definido por suas raízes, independentemente da ordem, o número total de polinômios possíveis de Goppa assim construídos é, portanto, $\left(\prod_{i=0}^{\lceil \lg t \rceil} (q - 2^i) \right) / \lceil \lg t \rceil! \approx (q - t) \binom{q}{\lceil \lg t \rceil}$.

Para $n \approx q/2$ o número de códigos diádicos pode ser aproximado como $q^m Q = 2^{m^2} Q$, onde $Q = \prod_{i=1}^{\infty} (1 - 1/2^i) \approx 0.2887881$. Conforme veremos, o número de

códigos quase-diádicos, que descreveremos a seguir como proposta para aplicações criptográficas, é consideravelmente maior que isso.

Antes de procedermos, porém, é interessante notar que uma das razões pelas quais o ataque proposto em (OTMANI; TILLICH; DALLOT, 2008) funciona contra certos códigos quase-cíclicos, além da estrutura limitada da permutação aplicada, é que aqueles esquemas partem de um código BCH ou Reed-Solomon conhecido, o qual é único a menos da escolha de um elemento primitivo do corpo finito subjacente. Com isso, naquelas propostas, um código inicial sobre $\mathbb{F}_{2^m}$ será, na melhor das hipóteses, escolhido de um conjunto de $O(2^m)$ códigos. Em contraste, nossa proposta parte de um código secreto escolhido de uma família muito maior de $O(2^{m^2})$ códigos. Por exemplo, enquanto aqueles esquemas possuem apenas 2^{15} pontos de partida sobre $\mathbb{F}_{2^{16}}$, nosso esquema pode amostrar uma família com mais de 2^{254} códigos sobre o mesmo corpo. A proteção principal do alçapão oculto, é claro, é o processo de puntura por blocos e a permutação de maneira mais complexa do código secreto inicial, conforme detalhado a seguir.

4.2.2 Construção de subcódigos quase-diádicos permutados binários

Um criptossistema não pode ser definido com segurança sobre um código de Goppa especificado diretamente por uma matriz de paridade em forma de Cauchy e em forma diádica, pois isso revelaria imediatamente o polinômio de Goppa $g(x)$: bastaria resolver o sistema linear, determinado $z_i - L_j = 1/H_{ij}$ que consiste de tn equações em $t + n$ incógnitas.

Conforme vimos na seção 4.2.1, o Algoritmo 2 gera códigos puramente diádicos, além de respeitar a restrição de Cauchy. Mostraremos agora como integrar as técnicas de (BERGER et al., 2009) com o Algoritmo 2 de modo a construir subcódigos de subcorpo quase-diádicos, cuja matriz de paridade é não-diádica, mas composta de blo-

cos diádicos. O princípio proposto aqui é *selecionar e permutar* as colunas da matriz original de paridade de maneira a preservar a quase-diadicidade do subcódigo alvo, impossibilitando a determinação do alçapão através dessa matriz de paridade.

Para o nível de segurança desejado (ver discussão na Seção 4.4.1), escolha $q = 2^m$ para algum m , um código de comprimento n e número projetado de erros corrigíveis t tal que $n = \ell t$ para algum $\ell > m$. Por simplicidade, assumiremos que t é uma potência de 2, mas o método de construção a seguir pode ser modificado para trabalhar com outros valores.

Execute o Algoritmo 2 para produzir um código sobre $\mathbb{F}_q$ cujo comprimento $N \gg n$ seja um múltiplo grande de t sem exceder o maior comprimento possível $q/2$, de modo que a matriz de paridade $t \times N$ construída, $\hat{H}$, possa ser vista como uma sequência de N/t blocos diádicos $[B_0 \mid \cdots \mid B_{N/t-1}]$ de tamanho $t \times t$. Selecione uniformemente ao acaso ℓ blocos distintos $B_{i_0}, \dots, B_{i_{\ell-1}}$ em qualquer ordem a partir de $\hat{H}$, juntamente com ℓ permutações diádicas $\Pi^{j_0}, \dots, \Pi^{j_{\ell-1}}$ de tamanho $t \times t$. Seja $\hat{H}' = [B_{i_0}\Pi^{j_0} \mid \cdots \mid B_{i_{\ell-1}}\Pi^{j_{\ell-1}}] \in (\mathbb{F}_q^{t \times t})^\ell$.

Para completar a construção, é necessário escolher uma matriz geradora compacta para o subcódigo sobre o subcorpo base. Representando cada elemento do corpo numa base de $\mathbb{F}_q$ sobre o subcorpo base $\mathbb{F}_2$, pode-se ver H como uma superposição de $m = [\mathbb{F}_q : \mathbb{F}_2]$ matrizes binárias diádicas distintas, e cada uma delas pode ser armazenada numa assinatura diádica separada. Embora a matriz de paridade H construída pelo Algoritmo 2 seja diádica sobre $\mathbb{F}_q$, a construção comum por traço leva a uma matriz que provavelmente viola a simetria diádica, como exemplificado a seguir.

Exemplo 11. Seja $\mathbb{F}_{2^4}$ o corpo finito operando sobre seus elementos módulo o polinômio irreduzível $f(x) = x^4 + x^3 + 1$. Sendo α raiz do polinômio irreduzível,

listamos seus valores em forma binária:

0 : 0000	α^3 : 1000	α^7 : 0111	α^{11} : 1101
1 : 0001	α^4 : 1001	α^8 : 1110	α^{12} : 0011
α : 0010	α^5 : 1011	α^9 : 0101	α^{13} : 0110
α^2 : 0100	α^6 : 1111	α^{10} : 1010	α^{14} : 1100

Seja M a matriz-diádica 2×2 com elementos no corpo finito $\mathbb{F}_{2^4}$:

$$\begin{pmatrix} \alpha^4 & \alpha^9 \\ \alpha^9 & \alpha^4 \end{pmatrix}$$

Sabendo que $\alpha^4 = 1001$ e $\alpha^9 = 0101$, a construção por traço gera:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{1} \\ 0 & 1 \\ 1 & 0 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{1} \end{pmatrix}$$

Repare que os blocos 2×2 em negrito violam a simetria diádica. □

Sendo assim, é necessário utilizar outra forma de construção de subcódigo de subcorpo que preserve a simetria diádica dos blocos da matriz de verificação de paridade. Para isso, foi utilizada a *construção por co-traço*, explicada a seguir e que atinge esse objetivo.

Definição 20. (*Construção por co-traço*) Seja p uma potência de um número primo, $q = p^d$ para algum d , e $\mathbb{F}_q = \mathbb{F}_p[x]/b(x)$ para algum polinômio irredutível $b(x) \in \mathbb{F}_p[x]$ de grau d . Dado um código especificado por uma matriz de paridade $H \in \mathbb{F}_q^{t \times n}$, a construção por co-traço deriva a partir dele um subcódigo sobre o subcorpo $\mathbb{F}_p$, com

matriz de paridade $T'_d(H) \in \mathbb{F}_p^{dt \times n}$, escrevendo os coeficientes de $\mathbb{F}_p$ dos termos de igual grau extraídos de todos os componentes de uma coluna de H sobre linhas sucessivas de $T'_d(H)$.

Assim, dados os elementos $u_i(x) = u_{i,0} + \dots + u_{i,d-1}x^{d-1} \in \mathbb{F}_q = \mathbb{F}_p[x]/b(x)$, a construção por traço mapeia a coluna $(u_0, \dots, u_{t-1})^\top$ de H à coluna $(u_{0,0}, \dots, u_{t-1,0}; \dots; u_{0,d-1}, \dots, u_{t-1,d-1})^\top$ da matriz de co-traço $T'_d(H)$. A matriz de paridade de *co-traço*, equivalente a $T_d(H)$ por uma permutação à esquerda.

Exemplo 12. Seja M a matriz-diádica 2×2 com elementos no corpo finito $\mathbb{F}_{2^4}$, ambos do exemplo anterior:

$$\begin{pmatrix} \alpha^4 & \alpha^9 \\ \alpha^9 & \alpha^4 \end{pmatrix}$$

Sendo $\alpha^4 = 1001$ e $\alpha^9 = 0101$, a construção por co-traço gera:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Com todos os blocos 2×2 respeitando a simetria diádica. □

Finalizando o processo de construção de subcódigos quase-diádicos permutados binários, calcule a matriz de co-traço $H' = T'_m(\hat{H}') \in (\mathbb{F}_2^{t \times t})^{m \times \ell}$ e a coloque em forma sistemática H (por exemplo, utilizando eliminação gaussiana).

A matriz de paridade resultante define um código binário de comprimento n e dimensão $k = n - mt$ e, uma vez que todas as operações por blocos realizadas durante

a eliminação gaussiana são realizadas no anel $\Delta(\mathbb{F}_2^t)$, o resultado ainda é formado por submatrizes diádicas que podem ser representadas por assinaturas de comprimento t . Segue deste fato que a matriz H como um todo pode ser armazenada em espaço menor que o de uma matriz genérica por um fator t .

No entanto, as submatrizes diádicas que aparecem neste processo não são necessariamente não-singulares, pois não estão mais associadas a matrizes de Cauchy. Caso todas as submatrizes de uma mesma coluna forem singulares (acima ou abaixo da diagonal, de acordo com a direção deste processo) de maneira que nenhum pivoteamento seja possível, o bloco inteiro que contém essa coluna pode ser substituído por outro bloco B_j escolhido aleatoriamente de $\hat{H}$.

Para cada código produzido pelo Algoritmo 2, o número de códigos gerados por esta construção é calculado a partir de:

Número possível de conjuntos de ℓ blocos $\times$ número possível de ordenações dos
 ℓ blocos $\times$ número possível de permutações diádicas de dimensão t .

Que é equivalente a:

$$\binom{N/t}{\ell} \times \ell! \times t^\ell$$

Logo, $\binom{N/t}{\ell} \times \ell! \times t^\ell \times \prod_{i=0}^{\lceil \lg N \rceil} (q - 2^i) \approx N^\ell \times 2^{m^2} Q$ códigos são possíveis em princípio. A informação do alçapão, que consiste na essência η de h , na sequência $(i_0, \dots, i_{\ell-1})$ de blocos, e na sequência $(j_0, \dots, j_{\ell-1})$ de identificadores de permutação diádica, relaciona o código público definido por H com o código privado definido por $\hat{H}$. O espaço ocupado pelo alçapão é, portanto, $m^2 + \ell \lg N$ bits. Se o maior valor possível $N = 2^{m-1}$ for adotado, isto simplifica-se para o tamanho máximo de $m^2 + \ell(m - 1)$ bits.

O espaço total ocupado pela parte essencial da matriz geradora (ou de paridade) binária resultante é de $mt \times (n - mt)/t = mk$ bits, ou seja, um fator t melhor que códigos de Goppa genéricos, que ocupam $k(n - k) = mkt$ bits. Se t não fosse uma potência de 2, digamos, $t = 2^\nu v$ onde $v > 1$ é ímpar, o custo de multiplicar matrizes $t \times t$ seria

em geral $O(2^u uv^3)$ em vez de simplesmente $O(2^u u)$, e a matriz final seria comprimida por um fator 2^u apenas.

4.2.3 Um exemplo em miniatura

Seja $\mathbb{F}_{2^5} = \mathbb{F}_2[u]/(u^5 + u^2 + 1)$. A assinatura diádica

$$h = (u^{20}, u^3, u^6, u^{28}, u^9, u^{29}, u^4, u^{22}, u^{12}, u^5, u^{10}, u^2, u^{24}, u^{26}, u^{25}, u^{15})$$

e o offset $\omega = u^{21}$ definem um código de Goppa de comprimento $N = 16$ capaz de corrigir 2 erros, com $g(x) = (x - u^{12})(x - u^{15})$ e suporte $L = (u^{21}, u^{29}, u^{19}, u^{26}, u^6, u^{16}, u^7, u^5, u^{25}, u^3, u^{11}, u^{28}, u^{27}, u^9, u^{22}, u^2)$. A matriz de paridade associada, construída segundo o Teorema 7, é

$$\hat{H} = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} u^{20} & u^3 & u^6 & u^{28} & u^9 & u^{29} & u^4 & u^{22} & u^{12} & u^5 & u^{10} & u^2 & u^{24} & u^{26} & u^{25} & u^{15} \\ u^3 & u^{20} & u^{28} & u^6 & u^{29} & u^9 & u^{22} & u^4 & u^5 & u^{12} & u^2 & u^{10} & u^{26} & u^{24} & u^{15} & u^{25} \end{array} \right],$$

com 8 blocos $B_0, \dots, B_7$ de tamanho 2×2 conforme indicado.

Suponha que o subcódigo tenha comprimento $n = 14$. Logo, da equação $n = \ell t$, sabemos que devemos selecionar $\ell = 7$ blocos. Ao acaso, selecionamos os blocos $B_7, B_5, B_1, B_2, B_3, B_6$ e B_4 .

Para as permutações diádicas, temos à disposição apenas duas opções, a saber:

$$\Pi^0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \Pi^1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Suponha que selecionamos ao acaso as seguintes permutações: $\Pi^0, \Pi^1, \Pi^0, \Pi^1, \Pi^0, \Pi^1, \Pi^0$.

Assim, temos a sequência encurtada, rearranjada e permutada por blocos $\hat{H}' = [B_7\Pi^0 \mid B_5\Pi^1 \mid B_1\Pi^0 \mid B_2\Pi^1 \mid B_3\Pi^0 \mid B_6\Pi^1 \mid B_4\Pi^0]$, ou seja:

$$\hat{H}' = \left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc} u^{25} & u^{15} & u^2 & u^{10} & u^6 & u^{28} & u^{29} & u^9 & u^4 & u^{22} & u^{26} & u^{24} & u^{12} & u^5 \\ u^{15} & u^{25} & u^{10} & u^2 & u^{28} & u^6 & u^9 & u^{29} & u^{22} & u^4 & u^{24} & u^{26} & u^5 & u^{12} \end{array} \right],$$

cuja matriz de co-traço sobre $\mathbb{F}_2$ é:

$$\hat{H}'' = \left[\begin{array}{cccccccccccccccc} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{array} \right],$$

e escalonada tem a forma sistemática:

$$H = \left[\begin{array}{cccc|cccccccccccc} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] = [M^T \mid I_{n-k}],$$

da qual se obtém imediatamente a matriz geradora $k \times n = 4 \times 14$ em forma sistemática

$$G = \left[\begin{array}{cccc|cccccccccccc} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{array} \right] = [I_k \mid M].$$

As matrizes G e H compartilham a parte essencial

$$M = \left[\begin{array}{cccccccccccc} \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{array} \right],$$

que é inteiramente especificada pelos elementos em negrito, e podem portanto ser armazenadas em 20 bits em vez de, respectivamente, $4 \cdot 14 = 56$ e $10 \cdot 14 = 140$ bits.

4.3 Complexidade computacional de decodificar códigos quase-diádicos

O esquema McEliece convencional (ou, equivalentemente, o esquema Niederreiter convencional) é descrito mais apropriadamente como uma *função de mão única com alçapão*, em vez de um criptossistema completo de chave pública. Funções dessa natureza são usadas em criptografia em muitos cenários diferentes, cada um com diferentes requisitos de segurança, e tais aplicações fogem ao escopo deste trabalho. Portanto, iremos nos concentrar na questão de inverter a função de alçapão, em outras palavras, na tarefa de decodificação.

Conforme assinalamos no capítulo 3.6, a classe de códigos de Goppa continua a ser uma das melhores escolhas para instanciar criptossistema baseados em códigos corretores de erros. Embora nossa proposta seja, em última análise, baseada em códigos de Goppa, pode-se perguntar se a natureza altamente composta do polinômio gerador

de Goppa $g(x)$, ou a estrutura peculiar das matrizes de paridade e geradores quase-diádicas, deixam vaziar alguma informação que possa facilitar a decodificação sem o conhecimento do alçapão.

Contudo, qualquer código alternante pode ser escrito de maneira similar a um código de Goppa, usando o componente diagonal de sua matriz de paridade padrão (vide Definição 9) para interpolar um polinômio gerador (não necessariamente do grau de t), que é, com alta probabilidade, composto. Não conhecemos nenhuma maneira em que este fato poderia ser aproveitado para facilitar a decodificação, sem pleno conhecimento da estrutura do código. Claramente, qualquer resultado neste sentido afetaria a maioria dos códigos alternantes proposto para fins criptográficos até os dias de hoje.

O ataque de (OTMANI; TILLICH; DALLOT, 2008) contra códigos quase-cíclicos poderia ser modificado para funcionar contra códigos de Goppa em forma diádica. Por este motivo, adotamos as mesmas contramedidas propostas por (BERGER et al., 2009) para defender códigos quase-cíclicos, a saber, trabalhar com um subcódigo encurtado por blocos de um código muito longo, tal como descrito na Seção 4.2.2. Esta ideia também se aproveita do trabalho de Wieschebrink (WIESCHEBRINK, 2006), que provou que decidir se um código é equivalente a um código encurtado é NP-completo. No nosso caso, o resultado é esconder a estrutura de Cauchy do código privado em uma estrutura diádica geral, ao invés de disfarçar uma código quase-cíclico como outro código de mesma simetria.

Apresentaremos agora uma redução do problema de decodificação da classe especial de códigos quase-diádicos ao problema bem conhecido da decodificação de síndromes, que é clássico em teoria da codificação e sabidamente NP-completo (BERLEKAMP; MCELIECE; TILBORG, 1978).

Definição 21 (Decodificação de Síndromes). *Seja $\mathbb{F}_q$ um corpo finito, e seja (H, w, s) uma tripla consistindo em uma matriz $H \in \mathbb{F}_q^{r \times n}$, um inteiro $w < n$, e um vetor $s \in \mathbb{F}_q^r$. Existe um vetor $e \in \mathbb{F}_q^n$ com peso de Hamming $\text{wt}(e) \leq w$ tal que $He^T = s^T$?*

O problema correspondente para matrizes quase-diádicas é:

Definição 22 (Decodificação de Síndromes Quase-Diádicas). *Seja $\mathbb{F}_q$ um corpo finito, e seja (H, w, s) uma tripla consistindo em uma matriz quase-diádica $H \in \Delta(\mathbb{F}_q^\ell)^{r \times n}$, um inteiro $w < \ell n$, e um vetor $s \in \mathbb{F}_q^{\ell r}$. Existe um vetor $e \in \mathbb{F}_q^{\ell n}$ com peso de Hamming $\text{wt}(e) \leq w$ tal que $He^\top = s^\top$?*

Teorema 10. *O problema da decodificação de síndromes quasi-diádicas (QD-SDP) é polinomialmente equivalente ao problema da decodificação de síndromes (SDP). Em outras palavras, decodificar códigos quase-diádicos é tão difícil no pior caso quanto decodificar códigos gerais.*

Demonstração. O QD-SDP, sendo uma instância do SDP restrito a uma classe particular de códigos, é claramente um problema de decisão em NP.

Consideremos agora uma instância genérica $(H', w', s') \in \mathbb{F}_q^{r \times n} \times \mathbb{Z} \times \mathbb{F}_q^r$ do SDP. Suponhamos que exista um oráculo que resolve o QD-SDP sobre $\Delta(\mathbb{F}_q^\ell)$ para algum $\ell > 0$ dado. Seja $v_\ell \in \mathbb{F}_q^\ell$ o vetor totalmente unitário, i.e. $(v_\ell)_j = 1$ para todo j . Definamos a matriz quase-diádica $H = H' \otimes I_\ell \in \Delta(\mathbb{F}_q^\ell)^{r \times n}$ com blocos $H_{ij} = H'_{ij} I_\ell$, o vetor $s = s' \otimes v_\ell \in (\mathbb{F}_q^\ell)^r$ com blocos $s_i = s'_i v_\ell$, e $w = \ell w'$. É evidente que a instância $(H, w, s) \in \Delta(\mathbb{F}_q^\ell)^{r \times n} \times \mathbb{Z} \times (\mathbb{F}_q^\ell)^r$ do QD-SDP pode ser construída em tempo polinomial.

Suponhamos agora que exista um vetor $e \in \mathbb{F}_q^{\ell n}$ com peso de Hamming $\text{wt}(e) \leq w$ tal que $He^\top = s^\top$. Para todo $0 \leq i < \ell$, seja $e'_i \in \mathbb{F}_q^n$ o vetor com elementos $(e'_i)_j = e_{i+j\ell}$, $0 \leq j < n$, de modo que os componentes e'_i são intercalados para compor e . Obviamente, pelo menos um dos e'_i tem peso de Hamming não excedendo $w/\ell = w'$, e pela construção de H qualquer um deles satisfaz $He'_i{}^\top = s'^\top$, constituindo uma solução da instância dada do SDP. Isto efetivamente reduz o SDP ao QD-SDP para qualquer ℓ dado em tempo polinomial. Portanto, o QD-SDP é ele mesmo NP-completo. $\square$

Embora este teorema não diga nada sobre a complexidade computacional no caso médio, ele reforça a nossa posição de que a família de códigos que propomos é, em

princípio, não menos adequada para aplicações criptográficas que um código genérico, no sentido de que, caso o QD-SDP venha a ser eficientemente solúvel no pior caso, *todos* os criptossistemas baseados em decodificação de síndromes serão definitivamente descartados, independentemente de qual código seja usado para instanciá-los. Felizmente, o tempo esperado de execução de todos os algoritmos conhecidos para o SDP (e para o QD-SDP) é exponencial, proporcionando evidência empírica de que o caso médio desses problemas também seja muito difícil.

A seguir, analisamos um ataque contra códigos de Goppa quase-diádicos, o qual não afeta a confiabilidade de sua versão utilizando corpos finitos binários (como explicado em todo o texto). De qualquer forma, vale ressaltar que criptossistemas baseados em códigos quase-diádicos dependerão, via de regra, de hipóteses mais específicas de segurança, cuja análise transcende o escopo deste trabalho.

4.3.1 Ataques baseados na solução de sistema de equações multivariadas quadráticas

Recentemente, em (FAUGÈRE et al., 2010) foi proposto a redução do problema de decodificação para códigos quase-diádicos (entre outros, tais como códigos quase-cíclicos) ao problema de resolver sistemas de equações quadráticas multivariadas.

A idéia central deste ataque é buscar diretamente um decodificador alternante para o código público, ou seja, escrever a matriz de verificação de paridade como $H = VD$, para uma matriz de Vandermonde V desconhecida e uma matriz diagonal D definida sobre o corpo finito $\mathbb{F}_{2^d}$, sendo $d \mid m$. Uma técnica relacionada é adotada em (UMANA; LEANDER, 2009). As componentes desconhecidas de V e D nas equações definidas em $H = VD$ dão origem a uma instância do problema de equações quadráticas multivariadas. Tratando cuidadosamente a estrutura de H , muitas equações componentes tornam-se lineares e as quadráticas restantes apenas envolvem um número pequeno de variáveis, reduzindo significativamente a complexidade do sistema resultante. Desta

forma, os autores são capazes de quebrar todos os parâmetros propostos em (BERGER et al., 2009) e (MISOCZKI; BARRETO, 2009) sobre corpos finitos, mas não sobre parâmetros binários.

Além disso, desconsiderando o fato da complexidade do ataque aumentar rapidamente quando os códigos são definidos sobre corpos menores, ainda argumentamos que essa estratégia não obtém resultado contra códigos quase-diádicos binários, até mesmo em caso de obterem sucesso contra códigos quase-cíclicos. A razão para isso se refere ao princípio do ataque de construir diretamente um decodificador alternante a partir do código público definido por H , o qual não é, com esmagadora probabilidade, um código de Goppa.

Esse decodificador pode ser utilizado para corrigir até $t/2$ erros no máximo, onde t é o número de erros estipulado. Para todos os códigos alternantes, à exceção de códigos de Goppa binários, esse é exatamente o número de erros que podem ser introduzidos e que o decodificador privado seria capaz de corrigir. Entretanto, para o caso específico de códigos de Goppa binários, incluindo códigos de Goppa quase-diádicos binários, esse ataque somente é capaz de corrigir a metade dos erros que podem ser introduzidos e corrigidos pelo decodificador de Goppa privado.

Sabendo que, ao aplicar um decodificador capaz de corrigir $t/2$ erros em um mensagem com t erros, produziremos lixo (em vez de uma mensagem com apenas os $t/2$ erros restantes), o atacante seria obrigado a adivinhar $t/2$ erros antes de usar este decodificador alternante. Isso significa repetir a tentativa de decodificar $\binom{n}{t/2} / \binom{t}{t/2}$ vezes, o que é inviável para parâmetros práticos.

Assim, nós concluímos que ataques baseados na solução de instâncias do problema de equações quadráticas multivariadas falham contra códigos quase-diádicos corretamente escolhidos.

4.4 Considerações sobre eficiência

Devido à sua estrutura, as matrizes da nossa proposta podem ser mantidas como vetores simples, não só para o armazenamento de longo prazo ou para transmissão, mas também para processamento.

A operação de multiplicar um vetor por uma matriz (quase-)diádica está no centro do criptossistema de McEliece. A abordagem via transformada rápida de Walsh-Hadamard (FWHT) (GULAMHUSEIN, 1973) para a convolução diádica através de *lifting*¹ para característica 0 produz complexidade assintótica $O(n \lg n)$ para esta operação, e consequentemente também para a codificação. O método de decodificação de Sarwate (SARWATE, 1977) também estabelece um custo assintótico para aquela operação de aproximadamente $O(n \lg n)$ na configuração $t = O(n / \lg n)$, típica em criptografia.

A inversão diádica, por outro lado, exige apenas $O(n)$ passos: pode-se mostrar por indução que uma matriz diádica binária $\Delta(h)$ de dimensão n satisfaz $\Delta^2 = (\sum_i h_i)^2 I$, e por conseguinte sua inversa, quando existe, é $\Delta^{-1} = (\sum_i h_i)^{-2} \Delta$, que pode ser calculada em $O(n)$ passos uma vez que é inteiramente determinada por sua primeira linha.

A conversão de uma matriz quase-diádica para forma sistemática (escalonada) envolve uma eliminação gaussiana que acarreta cerca de $m^2 \ell$ produtos de submatrizes diádicas $t \times t$, implicando uma complexidade $O(m^2 \ell t \lg t) = O(m^2 n \lg n)$, que se simplifica para $O(n \lg^3 n)$ num cenário criptográfico típico com $m = O(\lg n)$.

A Tabela 4 resume as complexidades assintóticas de geração de código (devido principalmente à formatação sistemática), codificação e decodificação, que coincidem com as complexidades de geração de chave, cifração e decifração em criptossistemas típicos baseados em códigos.

¹Agradecemos a Dan Bernstein por sugerir a técnica de *lifting* para emular a FWHT em característica 2.

Tabela 4: Complexidade de operação relativa ao comprimento n do código.

operação	genérica	proposta
geração de código	$O(n^3)$	$O(n \lg^3 n)$
codif. / decodif.	$O(n^2)$	$O(n \lg n)$

4.4.1 Parâmetros sugeridos

A Tabela 5 contém uma variedade de parâmetros balanceados para níveis práticos de segurança. Embora não os recomendemos para adoção antes da realização de análises mais profundas, estes parâmetros foram escolhidos para ressaltar as possibilidades da nossa proposta, dando uma impressão realista do que se poderia de fato escolher na prática. O número de erros é sempre uma potência de 2 para otimizar a redução de tamanho, e o código original de onde se extrai o código binário de Goppa é sempre definido sobre $\mathbb{F}_{2^{16}}$. Os níveis efetivos de segurança², calculados de acordo com a estratégia de ataque em (BERNSTEIN; LANGE; PETERS, 2008) são, respectivamente, 84.3, 112.3, 136.5, 201.6, e 265.1, enquanto o número de códigos possíveis varia entre $2^{669.6}$ e $2^{792.4}$.

O nível alvo de segurança, que corresponde aproximadamente ao custo logarítmico estimado do melhor ataque conhecido segundo as orientações em (BERNSTEIN; LANGE; PETERS, 2008), é mostrado na coluna ‘nível’. A coluna ‘tamanho’ contém a quantidade de bits necessária para armazenar uma matriz geradora ou de paridade quase-diádica em forma sistemática. O tamanho de uma matriz sistemática equivalente para um código de Goppa genérico aproximadamente no mesmo nível de segurança conforme sugerido em (BERNSTEIN; LANGE; PETERS, 2008) é dado na coluna ‘genérico’. Em ambos os casos leva-se em conta somente a parte essencial da chave em forma sistemática (a prova de que isto não afeta o nível de segurança encontra-se em (ENGELBERT; OVERBECK; SCHMIDT, 2007)). A coluna ‘redução’ traz a razão de tamanhos entre uma matriz genérica e uma matriz quase-diádica equivalente.

²Somos gratos a Christiane Peters pela gentileza de prover estas estimativas.

Tabela 5: Parâmetros sugeridos para um código de Goppa $[n, k, 2t + 1]$ binário.

nível	n	k	t	tamanho	genérico	redução
80	2304	1280	64	20480	460647	22.5
112	3584	1536	128	24576	1047600	42.6
128	4096	2048	128	32768	1537536	46.9
192	6912	2816	256	45056	4185415	92.9
256	8192	4096	256	65536	7667855	117.0

4.5 Implementação

Para os parâmetros na Tabela 5, observaram-se os tempos relatados nas Tabelas 6, 7 e 8 (medidos em ms) para códigos de Goppa genéricos e quase-diádicos (QD), e também para o algoritmo RSA para avaliar a eficiência relativa a um criptossistema pré-quântico muito comum. Não se fez nenhuma tentativa séria de otimizar agressivamente a implementação, que foi feita em C++ e testada numa plataforma AMD Turion 64X2 2.4 GHz. *Benchmarks* para RSA-15360 foram omitidos devido ao enorme tempo necessário para gerar parâmetros adequados.

Tabela 6: Benchmarks para geração de chaves. Tempos em ms.

nível	RSA	genérico	QD
80	563	375	17.2
112	1971	1320	18.7
128	4998	2196	20.5
192	628183	13482	47.6
256	–	27161	54.8

Para a geração de chaves, os tempos obtidos com a implementação quase-diádica são entre 21 (nível de segurança 80 bits) e 495 (nível de segurança de 256 bits) vezes mais eficientes do que a implementação genérica. Com relação ao RSA, essa melhora é ainda mais notável, sendo entre 32 e 13197 vezes mais eficientes, para o nível de segurança de 192 bits (vale lembrar que não foi considerado o nível de segurança

de 256 bits, dado enorme tempo que demandaria). Observa-se também um salto de tempo demandado pela proposta quase-diádica no nível de 192 bits, fato que se repete nas demais tabelas de *benchmark*. Isto se deve à dependência em n , que aumentou proporcionalmente na escolha de parâmetros (vide Tabela 5).

Tabela 7: Benchmarks para cifração. Tempos em ms.

nível	RSA	genérico	QD
80	0.431	0.736	0.817
112	1.548	1.696	1.233
128	3.467	2.433	1.575
192	22.320	6.872	4.695
256	–	12.176	6.353

Para a cifração, no comparativo entre a implementação quase-diádica e a genérica, só é notado um ganho a partir do nível de 112 bits de segurança, chegando um desempenho 1,9 vezes superior, no caso de 256 bits de segurança. No comparativo com o RSA, também só é notada uma melhora a partir de 80 bits de segurança, mas chegando a 4,7 vezes mais eficiente, no caso de 192 bits de segurança.

Tabela 8: Benchmarks para decifração. Tempos em ms.

nível	RSA	genérico	QD
80	15.61	1.016	3.685
112	110.34	2.123	4.463
128	349.91	3.312	5.261
192	5094.10	8.822	17.783
256	–	15.156	21.182

Por fim, para a decifração, a implementação quase-diádica não propicia melhorias, quando comparada à implementação genérica. Entretanto, chega a ser 286 vezes mais

eficientes do que o RSA, para o nível de segurança de 192 bits.

4.5.1 Aplicações

Nossa proposta possibilita diversos cenários de aplicações. Baseando-se em sua eficiência algorítmica, principalmente quando comparada às soluções criptográficas extensamente utilizadas, podemos citar os dispositivos com baixo recurso de processamento. Além de demandarem algoritmos extremamente eficientes pelo fato de terem pouca capacidade de processamento, em muitas vezes, tais dispositivos são móveis, exigindo também o máximo de economia de energia.

Vale ressaltar que outras soluções criptográficas pós-quânticas também apresentam essa vantagem algorítmica, mas pecam pelo tamanho de suas chaves criptográficas. Esse problema foi bastante minimizado com a utilização de nossa proposta, ressaltando sua adequabilidade a este cenário.

Com relação às aplicações sem restrições de processamento, podemos citar o fato de que criptossistemas pós-quânticos, tais como de McEliece, são prioritariamente preferidos em relação às soluções convencionais pelo fato de terem chaves extensas. Conforme ressaltado, este problema foi bastante minimizado, aproximando esta solução inclusive do cenário tradicional de aplicações.

Por fim, não podemos esquecer de sua aplicabilidade em um eventual ambiente dotado de tecnologia quântica. Neste cenário, soluções criptográficas pós-quânticas deverão substituir as convencionais e aquelas que apresentarem menor impacto em sua adoção ganharão preferência.

4.6 Resumo

Neste capítulo, apresentamos os códigos de Goppa quase-diádicos, capazes de reduzir sensivelmente o tamanho das chaves dos criptossistemas McEliece.

Matrizes diádicas são completamente definidas apenas por sua primeira linha $h = (h_0, \dots, h_{n-1}) \in \mathcal{R}^n$, conhecida como assinatura diádica, e que define os demais elementos pela relação $H_{ij} = h_{i \oplus j}$. Estas matrizes formam um subanel comutativo. Uma permutação diádica é uma matriz diádica, cuja assinatura é uma linha da matriz identidade. Uma matriz quase-diádica é uma matriz possivelmente não diádica composta por blocos, tais que estes blocos são matrizes diádicas.

Matrizes de Cauchy são completamente definidas apenas pelas duas sequências disjuntas $z = (z_0, \dots, z_{t-1}) \in \mathbb{F}_q^t$ e $L = (L_0, \dots, L_{n-1}) \in \mathbb{F}_q^n$ de elementos distintos e que definem os demais elementos pela relação $C_{ij} = 1/(z_i - L_j)$. O código de Goppa gerado por um polinômio mônico $g(x) = (x - z_0) \dots (x - z_{t-1})$ sem raízes múltiplas admite uma matriz de paridade da forma $H = C(z, L)$, i.e. $H_{ij} = 1/(z_i - L_j)$, $0 \leq i < t$, $0 \leq j < n$. Logo, seria possível representar chaves do criptossistema McEliece por intermédio de matrizes de Cauchy. Entretanto, tais matrizes não formam um subanel comutativo, não permitindo operar sobre elas sem perder sua estrutura de Cauchy. Uma alternativa seria utilizar matrizes de Cauchy que também fossem diádicas.

Esta construção é possível e apresentada no Algoritmo 2. Entretanto, ataques estruturais poderiam obter sucesso, dadas as restritas relações que definem essas matrizes. Se valendo da estratégia de extrair códigos públicos encurtados por blocos a partir de códigos privados muito longos, é possível resguardar a estrutura do código, mantendo uma representação compacta para o mesmo. Isto é conseguido selecionando e permutando diadicamente os blocos que compõem a matriz de verificação de paridade.

Como resultado, as chaves criptográficas do criptossistema McEliece têm seu tamanho reduzido entre 22,5 (para o nível de segurança de 80 bits) e 117 vezes (para o nível de segurança de 256 bits), se comparadas às chaves genéricas, minimizando consideravelmente o maior entrave para o seu emprego.

5 CONCLUSÕES

Descrevemos neste trabalho como gerar códigos de Goppa em forma quase-diádica para aplicações criptográficas. Os tamanhos correspondentes das chaves num criptosistema típico do tipo McEliece são um fator $t = \tilde{O}(n)$ menores que o que se obtém com códigos genéricos de Goppa, e as chaves podem ser mantidas neste tamanho compacto não só para armazenamento e transmissão, mas também para processamento. Códigos quase-diádicos binários podem corrigir todos os erros projetados, em vez de apenas a metade como a maioria dos códigos alternantes. Também fornecem uma alternativa aos códigos convencionais cíclicos e quase-cíclicos e se beneficiam das mesmas técnicas de ocultação de alçapão propostas por Wieschebrink (WIESCHEBRINK, 2006) e por (BERGER et al., 2009). A complexidade de todas as operações no criptosistema McEliece e outros relacionados é reduzido a $\tilde{O}(n)$.

Outros criptosistemas podem se beneficiar de códigos diádicos, e.g. protocolos de identificação de entidade e de assinaturas digitais para os quais códigos duplamente circulantes foram propostos (GABORIT; GIRALT, 2007) poderiam usar códigos diádicos, até mesmo códigos gerados aleatoriamente, sem um alçapão de Goppa.

Uma linha adicional de investigação futura consiste em determinar se podem ser combinadas de forma segura as técnicas propostas em (BALDI; CHIARALUCE; BORDATO, 2008) com a nossa para definir códigos quase-diádicos com matriz de paridade de baixa densidade (QD-LDPC) adequados para fins criptográficos e potencialmente mais compactos que códigos quase-diádicos simples.

Curiosamente, é igualmente possível definir criptossistemas baseados em *reticulados diádicos* com chaves curtas, de modo inteiramente análogo a reticulados ideais cíclicos como propostos por Micciancio (MICCIANCIO, 2007) e conseguindo redução comparável de tamanho. Omitimos aqui, porém, qualquer tratamento adicional desta linha de investigação, uma vez que está fora do escopo deste trabalho.

Por fim, vale ressaltar que a proposta apresentada neste documento foi submetida e selecionada para publicação e apresentação no *XVI Workshop Selected Areas in Cryptography* (SAC'2009), ocorrido entre 13 e 14 de Agosto de 2009, em Calgary, Alberta, Canadá, de referência bibliográfica (MISOCZKI; BARRETO, 2009). Este evento foi realizado em cooperação com a *International Association for Cryptologic Research* (IACR) e teve seus artigos publicados pela *Springer* em *Lecture Notes in Computer Science*, volume 5867.

REFERÊNCIAS

- AU, S.; EUBANKS-TURNER, C.; EVERSON, J. *The McEliece Cryptosystem*. 2003. Unpublished manuscript.
- BALDI, M.; CHIARALUCE, F. Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC code. In: *IEEE International Symposium on Information Theory – ISIT’2007*. Nice, France: IEEE, 2007. p. 2591–2595.
- BALDI, M.; CHIARALUCE, F.; BODRATO, M. A new analysis of the mceliece cryptosystem based on qc-ldpc codes. In: *Security and Cryptography for Networks – SCN’2008*. [S.l.]: Springer, 2008. (Lecture Notes in Computer Science, v. 5229), p. 246–262.
- BARG, A. Complexity issues in coding theory. In: PLESS, V.; HUFFMAN, W. (Ed.). *Handbook of coding theory*. [S.l.]: North-Holland, 1998. p. 649–754.
- BERGER, T. P. et al. Reducing key length of the McEliece cryptosystem. In: *Progress in Cryptology – Africacrypt’2009*. [S.l.]: Springer, 2009. (Lecture Notes in Computer Science). To appear. Preliminary (2008) version at http://www.unilim.fr/pages_perso/philippe.gaborit/reducing.pdf.
- BERLEKAMP, E.; MCELIECE, R.; TILBORG, H. van. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, IEEE, v. 24, n. 3, p. 384–386, 1978.
- BERNSTEIN, D. J.; BUCHMANN, J.; DAHMEN, E. *Post-Quantum Cryptography*. [S.l.]: Springer, 2008.
- BERNSTEIN, D. J.; LANGE, T.; PETERS, C. Attacking and defending the McEliece cryptosystem. In: *Post-Quantum Cryptography Workshop – PQCrypto’2008*. [S.l.]: Springer, 2008. (Lecture Notes in Computer Science, v. 5299), p. 31–46. <http://www.springerlink.com/content/68v69185x478p53g>.
- BUNCHAMANN, J. A. *Introduction to Cryptography*. 2nd. ed. [S.l.]: Springer, 2004.
- CANTEAUT, A.; SENDRIER, S. Cryptanalysis of the original mceliece cryptosystem. In: *Advances in Cryptology – ASIACRYPT’98*. [S.l.]: Springer, 1998. (Lecture Notes in Computer Science, v. 1514), p. 187–199.
- DIFFIE, W.; HELLMAN, M. New directions in cryptography. *IEEE Transactions on Information Theory*, IEEE, v. 22, n. 6, p. 644–654, 1976.
- ELGAMAL, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IEEE, IT-31, n. 4, p. 469–472, 1985.

ENGELBERT, D.; OVERBECK, R.; SCHMIDT, A. A summary of McEliece-type cryptosystems and their security. *Journal of Mathematical Cryptology*, v. 1, p. 151–199, 2007.

FAUGÈRE, J.-C. et al. Algebraic cryptanalysis of McEliece variants with compact keys. In: *Advances in Cryptology – Eurocrypt’2010*. [S.l.]: Springer, 2010. (LNCS). To appear.

FINIASZ, M.; SENDRIER, N. Security bounds for the design of code-based cryptosystems. In: MATSUI, M. (Ed.). *Asiacrypt 2009*. Springer, 2009. (Lecture Notes in Computer Science, v. 5912), p. 88–105. Disponível em: <http://www-roc.inria.fr/secret/Matthieu.Finiasz/research/2009/finiasz-sendrier-asiacrypt09.pdf>.

FUJISAKI, E.; OKAMOTO, T. Secure integration of asymmetric and symmetric encryption schemes. In: *Advances in Cryptology – CRYPTO’1999*. Gold Coast, Australia: Springer, 1999. (Lecture Notes in Computer Science, v. 1666), p. 537–554.

GABIDULIN, E.; PARAMONOV, A.; TRETJAKO, O. Ideals over a non-commutative ring and their applications to cryptography. In: *Workshop on the Theory and Application of Cryptographic Techniques*. [S.l.: s.n.], 1991. (Proceedings), p. 482–489.

GABORIT, P. Shorter keys for code based cryptography. In: *International Workshop on Coding and Cryptography – WCC’2005*. Bergen, Norway: ACM Press, 2005. p. 81–91.

GABORIT, P.; GIRAULT, M. Lightweight code-based authentication and signature. In: *IEEE International Symposium on Information Theory – ISIT’2007*. Nice, France: IEEE, 2007. p. 191–195.

GIBSON, J. K. Severely denting the Gabidulin version of the McEliece public key cryptosystem. *Designs, Codes and Cryptography*, v. 6, n. 1, p. 37–45, 1995.

_____. The security of the Gabidulin public key cryptosystem. In: *Advances in Cryptology – Eurocrypt’1996*. Zaragoza, Spain: Springer, 1996. (Lecture Notes in Computer Science, v. 1070), p. 212–223.

GOPPA, V. D. A new class of linear error-correcting codes. *Problems of Information Transmission*, v. 6, p. 207–212, 1970.

GRAY, R. Toeplitz and circulant matrices: a review. *Foundations and Trends in Communications and Information Theory*, v. 2, n. 3, p. 155–329, 2005.

GULAMHUSEIN, M. N. Simple matrix-theory proof of the discrete dyadic convolution theorem. *Electronics Letters*, v. 9, n. 10, p. 238–239, 1973.

HOFFSTEIN, J.; PIPHER, J.; SILVERMAN, J. *An Introduction to Mathematical Cryptography*. 1st. ed. [S.l.]: Springer, 2008.

HUFFMAN, W. C.; PLESS, V. *Fundamentals of Error-Correcting Codes*. 1st. ed. [S.l.]: Cambridge University Press, 2003.

JANWA, H.; MORENO, O. McEliece public key cryptosystems using algebraic-geometric codes. *Designs Codes and Cryptography*, v. 8, n. 3, p. 293–307, 1996.

KAHN, D. *The Codebreakers - The Story of Secret Writing Handbook of Applied Cryptography*. 1st. ed. [S.l.]: Scribner Book Company, 1996.

KOBLITZ, M. Elliptic curve cryptosystems. In: KOBLITZ, M. (Ed.). *Math Comp.* [S.l.]: Publishing Press, 1987. p. 203–209.

LOIDREAU, P.; SENDRIER, N. Some weak keys in McEliece public-key cryptosystem. In: *IEEE International Symposium on Information Theory – ISIT'1998*. Boston, USA: IEEE, 1998. p. 382.

MACWILLIAMS, F. J.; SLOANE, N. J. A. *The theory of error-correcting codes*. [S.l.]: North-Holland Mathematical Library, 1977.

McEliece, R. J. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep Space Network Progress Report*, v. 44, p. 114–116, jan. 1978.

MENEZES, A.; OORSCHOT, P.; VANSTONE, S. *Handbook of Applied Cryptography*. 1st. ed. [S.l.]: CRC Press, 1996.

MICCIANCIO, D. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, v. 16, n. 4, p. 365–411, 2007.

MILLER, V. Uses of elliptic curves in cryptography. In: *Advances in Cryptology, Crypto 85*. [S.l.]: Springer, 1985, (Lecture Notes in Computer Science). p. 417–426.

MINDER, L.; SHOKROLLAHI, A. Cryptanalysis of the sidelnikov cryptosystem. In: *Advances in Cryptology – Eurocrypt'2007*. Barcelona, Spain: Springer, 2007. (Lecture Notes in Computer Science, v. 4515), p. 347–360.

MISOCZKI, R.; BARRETO, P. Compact McEliece keys from Goppa codes. In: *Selected Areas in Cryptography SAC2'2009*. Calgary, Canada: Springer, 2009. (Lecture Notes in Computer Science, v. 5867), p. 376–392.

MONICO, C.; ROSENTHAL, J.; SHOKROLLAHI, A. Using low density parity check codes in the McEliece cryptosystem. In: *IEEE International Symposium on Information Theory – ISIT'2000*. Sorrento, Italy: IEEE, 2000. p. 215.

NIEDERREITER, H. Knapsack-type crytosystems and algebraic coding theory. *Prob. Contr. Inform. Theory*,, p. 157–166, 1986.

OTMANI, A.; TILLICH, J.-P.; DALLLOT, L. *Cryptanalysis of Two McEliece Cryptosystems Based on Quasi-Cyclic Codes*. 2008. Preprint. <http://arxiv.org/abs/0804.0409v2>.

OVERBECK, R. Structural attacks for public key cryptosystems based on gabidulin codes. *Journal of Cryptology*, v. 21, n. 2, p. 280–301, 2008.

- PATTERSON, N. J. The algebraic decoding of Goppa codes. *IEEE Transactions on Information Theory*, IEEE, v. 21, n. 2, p. 203–207, 1975.
- RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. M. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, v. 21, n. 2, p. 120–126, 1978.
- SARWATE, D. V. On the complexity of decoding Goppa codes. *IEEE Transactions on Information Theory*, IEEE, v. 23, n. 4, p. 515–516, 1977.
- SCHECHTER, S. On the inversion of certain matrices. *Mathematical Tables and Other Aids to Computation*, v. 13, n. 66, p. 73–77, 1959. <http://www.jstor.org/stable/2001955>.
- SENDRIER, N. Finding the permutation between equivalent linear codes: the support splitting algorithm. *IEEE Transactions on Information Theory*, IEEE, v. 46, n. 4, p. 1193–1203, 2000.
- SHOKROLLAHI, A.; MONICO, C.; ROSENTHAL, J. Using low density parity check codes in the mceliece cryptosystem. In: *IEEE International Symposium on Information Theory (ISIT 2000)*. [S.l.: s.n.], 2000. (Proceedings), p. 215.
- SHOR, P. W. Algorithms for quantum computation: discrete logarithms and factoring. In: SHOR, P. W. (Ed.). *35th Annual Symposium on Foundations of Computer Science (Santa Fe, NM, 1994)*. [S.l.]: IEEE Comput. Soc. Press, 1994. p. 124–134.
- SIDELNIKOV, V. A public-key cryptosystem based on reed-muller codes. *Discrete Mathematics and Applications*, v. 4, n. 3, p. 191–207, 1994.
- SIDELNIKOV, V.; SHESTAKOV, S. On cryptosystems based on generalized Reed-Solomon codes. *Discrete Mathematics*, v. 4, n. 3, p. 57–63, 1992.
- _____. On the insecurity of cryptosystems based on generalized reed-solomon codes. *Diskretnaya Mat.*, v. 4, n. 3, 1992.
- SINGLETON, R. C. Maximum distance q-nary codes. *IEEE Transactions on Information Theory*, IEEE, v. 10, n. 2, p. 116–118, 1964.
- TZENG, K. K.; ZIMMERMANN, K. On extending Goppa codes to cyclic codes. *IEEE Transactions on Information Theory*, IEEE, v. 21, p. 721–716, 1975.
- UMANA, V. G.; LEANDER, G. *Practical Key Recovery Attacks On Two McEliece Variants*. 2009. Cryptology ePrint Archive, Report 2009/509. <http://eprint.iacr.org/>.
- WIESCHEBRINK, C. Two NP-complete problems in coding theory with an application in code based cryptography. In: *IEEE International Symposium on Information Theory – ISIT'2006*. Seattle, USA: IEEE, 2006. p. 1733–1737.