

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE SÃO JOÃO DA BOA VISTA

LUIS ANTONIO BORGES JUNIOR

Implementação Alternativa de Códigos Corretores de Erros para Sistemas de Comunicações
Ópticas de 400 Gbps : Código de Hamming Duplamente Estendido com Algoritmo DEPT para
Decodificação

São João da Boa Vista

2022

Luis Antonio Borges Junior

Implementação Alternativa de Códigos Corretores de Erros para Sistemas de Comunicações Ópticas de 400 Gbps : Código de Hamming Duplamente Estendido com Algoritmo DEPT para Decodificação

Trabalho de Graduação apresentado ao Conselho de Curso de Graduação em Engenharia Eletrônica e de Telecomunicações do Campus de São João da Boa Vista, Universidade Estadual Paulista, como parte dos requisitos para obtenção do diploma de Graduação em Engenharia Eletrônica e de Telecomunicações .

Orientador: Profº Dr. Ivan Aritz Aldaya Garde

São João da Boa Vista

2022

B732i

Borges Junior, Luis Antonio

Implementação Alternativa de Códigos Corretores de Erros para Sistemas de Comunicações Ópticas de 400 Gbps : Código de Hamming Duplamente Estendido com Algoritmo DEPT para Decodificação / Luis Antonio Borges Junior. -- São João da Boa Vista, 2022

49 p. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Engenharia de Telecomunicações) - Universidade Estadual Paulista (Unesp), Faculdade de Engenharia, São João da Boa Vista

Orientador: Ivan Aritz Aldaya Garde

1. Comunicações ópticas. 2. Códigos corretores de erros (Teoria da informação). 3. Comunicações digitais. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Engenharia, São João da Boa Vista. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

**UNIVERSIDADE ESTADUAL PAULISTA “JÚLIO DE MESQUITA FILHO”
FACULDADE DE ENGENHARIA - CÂMPUS DE SÃO JOÃO DA BOA VISTA
GRADUAÇÃO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES**

TRABALHO DE CONCLUSÃO DE CURSO

**IMPLEMENTAÇÃO ALTERNATIVA DE CÓDIGOS CORRETORES DE ERROS
PARA SISTEMAS DE COMUNICAÇÕES ÓPTICAS DE 400 GBPS: CÓDIGO DE
HAMMING DUPLAMENTE ESTENDIDO COM ALGORITMO DEPT PARA
DECODIFICAÇÃO**

Aluno: Luís Antonio Borges Junior
Orientador: Prof. Dr. Ivan Aritz Aldaya Garde

Banca Examinadora:

- Ivan Aritz Aldaya Garde (Orientador)
- Cintya Wink de Oliveira Benedito (Examinadora)
- Rafael Abrantes Penchel (Examinador)

A ata da defesa com as respectivas assinaturas dos membros encontra-se no prontuário do aluno (Expediente nº 058/2022)
--

São João da Boa Vista, 16 de novembro de 2022

Dedico este trabalho a todos aqueles que de alguma forma me apoiaram.

AGRADECIMENTOS

Gostaria de expressar minha sincera gratidão ao Prof. Dr. Ivan Aldaya Aritz Garde, Luciana Shiroma Montali e Matheus Martins Rodrigues por seu apoio contínuo, paciência e incentivo durante o período de graduação e de estágio. Sem o suporte dos mesmos, este trabalho não seria possível.

Gostaria de agradecer à Profa. Dra. Cintya Wink de Oliveira Benedito e ao Prof. Dr. Rafael Abrantes Penchel por revisar minha monografia e fazer parte da minha banca avaliadora.

Muito obrigado aos meus amigos da graduação e do estágio, pelo incentivo, sugestões e momentos agradáveis.

Por fim, gostaria de expressar minha profunda gratidão à minha mãe Alessandra Martins Gomes Borges e às minhas irmãs Karoline Martins Borges e Letícia Martins Borges, por seu apoio e encorajamento incondicionais.

*“Geðenc hwelc witu us ða becomon for ðisse worulde, ða ða we hit nohwæðer ne selfe ne lufodon ne
eac oðrum monnum ne lefdon.”
(Alfredo, o Grande)*

RESUMO

Essa monografia apresenta o processo de codificação e decodificação do código de correção de erros interno para atingir taxas de 400 Gbps do acordo de implementação OIF-400ZR. O código em questão é o Hamming (128, 119) duplamente estendido com entrada *soft* e saída *hard*. O processo de decodificação é realizado implementando um algoritmo de decodificação *soft* alternativo aos algoritmos de Chase. O código implementado atinge os parâmetros necessários para um sistema de 400 Gbps impostos pela norma OIF-400ZR. Baseado em uma otimização de parâmetros a qual é realizada e apresentada. Assim, o código corretor de erros atingiu um ganho de codificação de 1.18 dB com uma razão de erro de bits de saída de 4.28×10^{-3} para uma razão de erro de entrada de 1.25×10^{-2} .

PALAVRAS-CHAVE: Códigos corretores de erros; FEC; OIF; 400ZR; Hamming (128,119); Algoritmo DEPT; Decodificação *soft* e *hard*; Sistemas de comunicações ópticas; 400 Gbps; decodificador Hamming.

ABSTRACT

This monograph presents the process of encoding and decoding of the internal forward error correction (FEC) to achieve rates of 400 Gbps from implementation agreement OIF-400ZR. The FEC in question is the double-extended Hamming code (128, 119) with *soft* input and *hard* output. The decoding process is performed by implementing an alternative *soft* decoding algorithm to Chase's algorithms. The implemented FEC reaches the necessary parameters for a 400 Gbps system imposed by the OIF-400ZR standard. Based on a parameter optimization which is performed and presented. Thus, the FEC achieved a net coding gain of 1.18 dB with an output bit error ratio of 4.28×10^{-3} at an input error ratio of 1.25×10^{-2} .

KEYWORDS: Error correcting codes; FEC; OIF; 400ZR; Hamming (128,119); DEPT Algorithm; Hard and soft decoding; Optical communications systems; 400 Gbps; Hamming decoder.

LISTA DE ILUSTRAÇÕES

Figura 1	Arquitetura de um sistema de comunicação com staircase/Hamming FEC concatenado.	13
Figura 2	Representação de um sistema de comunicação óptica coerente com a presença de um DSP.	17
Figura 3	Representação de uma palavra-código na forma sistemática.	22
Figura 4	Constelação 16-QAM com símbolos mapeados usando codificação de Gray. . .	29
Figura 5	Regiões de decisão para o bit 0.	31
Figura 6	Regiões de decisão para o bit 1.	32
Figura 7	Regiões de decisão para o bit 2.	32
Figura 8	Regiões de decisão para o bit 3.	33
Figura 9	Curvas dos valores de LLR para os bits da polarização: (a) bit 0, (b) bit 1, (c) bit 2 e (d) bit 3.	33
Figura 10	Diagrama de blocos da implementação do decodificador com o algoritmo DEPT. .	36
Figura 11	Unidade (1,2,3) computa a soma de quatro LLRs absolutas que correspondem ao padrão de erro definido pela PEP (1,2,3).	37
Figura 12	Diagrama do fluxo de dados para obtenção dos resultados.	41
Figura 13	Efeito do <i>threshold</i> no algoritmo DEPT com quantificação de 5-bits.	42
Figura 14	Efeito do <i>threshold</i> no algoritmo DEPT com quantificação de 6-bits.	42
Figura 15	Curvas obtidas para o algoritmo DEPT com quantização de 5-bits.	44
Figura 16	Curvas obtidas para o algoritmo DEPT com quantização de 6-bits.	44
Figura 17	Sistema sem FEC e sistema com FEC.	46

LISTA DE TABELAS

Tabela 1 – Multiplicação e adição sobre $GF(2^m)$	19
Tabela 2 – Representação do corpo $GF(2^3)$	19
Tabela 3 – Função de permutação Π_d	27
Tabela 4 – Função de permutação inversa Π_d^{-1}	28
Tabela 5 – Amplitude do símbolo em fase (I) e em quadratura (Q)	30
Tabela 6 – Padrões de erros parciais (PEPs).	36
Tabela 7 – Dados obtidos para quantização de 5-bits	43
Tabela 8 – Dados obtidos para quantização de 6-bits	43
Tabela 9 – Tempo de simulação para os diferentes padrões e formas de quantização.	45
Tabela 10 – Probabilidade de erro de bit de saída para a especificação do 400ZR.	46

LISTA DE ABREVIATURAS E SIGLAS

ITU	União Internacional de Telecomunicações (<i>International Telecommunication Union</i>)
OIF	Forúm de Interconexão Óptica (<i>Optical Internetworking Forum</i>)
DWDM	Multiplexação por Divisão de Comprimento de Onda Densa (<i>Dense Wavelength Division Multiplexing</i>)
FEC	Correção de Erro Avançada (<i>Forward Error Correction</i>)
CFEC	Correção de Erro Avançada Concatenado (<i>Concatenated Forward Error Correction</i>)
BCH	Bose–Chaudhuri–Hocquenghem
BER	Razão de Erro de Bit (<i>Bit Error Ratio</i>)
NCG	Ganho de Codificação (<i>Net Coding Gain</i>)
DEPT	Teste de Padrão de Erros Direto (<i>Direct Error-Pattern Testing</i>)
ADC	Conversor Analógico/Digital (<i>Analog to Digital Converter</i>)
DSP	Processador Digital de Sinais (<i>Digital Signal Processor</i>)
DAC	Conversor Digital/Analógico (<i>Digital to Analog Converter</i>)
WDM	Multiplexação por Divisão de Comprimento de Onda (<i>Wavelength Division Multiplexing</i>)
EDFA	Amplificador de Fibra Dopado com Érbio (<i>Erbium-Doped Fiber Amplifier</i>)
Tx	Transmissor
Rx	Receptor
SNR	Relação Sinal-Ruído (<i>Signal to Noise Ratio</i>)
GF	Corpo de Galois ou Corpo Finito (<i>Galois Field</i>)
Cb(n , k)	Código de bloco, com n bits codificados e k bits de mensagem
Ham(n , k , d)	Código de Hamming, com n bits codificados, k bits de mensagem e distância mínima de Hamming d
LLR	Verossimilhança Logarítmica (<i>Log-likelihood Ratio</i>)
AWGN	Ruído Aditivo Gaussiano Branco (<i>Additive White Gaussian Noise</i>)
PEP	Padrão de Erro Parcial (<i>Partial Error Pattern</i>)

SUMÁRIO

1	INTRODUÇÃO	13
1.1	Motivação	13
1.2	Contribuição	14
1.3	Organização da Monografia	15
2	REFERENCIAL TEÓRICO	16
2.1	Códigos corretores de erros em sistemas de comunicações ópticas	16
2.1.1	Sistemas de comunicações ópticas	16
2.1.2	Códigos Corretores de Erros	17
2.1.3	Métricas de desempenho de um FEC	18
2.2	Estrutura básica de códigos corretores de erros	18
2.2.1	Construção do Corpo Finito	18
2.2.2	Códigos de Bloco	19
2.2.2.1	Matriz Geradora $G_{[k \times n]}$	21
2.2.2.2	Matriz verificadora de paridade $H_{[(n-k) \times k]}$	22
2.2.3	Detecção de Erro por Síndrome	22
2.3	Códigos de Hamming	23
2.3.1	Códigos de Hamming Convencionais	23
2.3.2	Códigos de Hamming Estendidos	24
2.3.3	Algoritmo de decodificação de Chase	24
2.4	Decodificação hard e soft	25
3	DESENVOLVIMENTO	27
3.1	Codificador do código de Hamming duplamente estendido	27
3.2	Mapeador 16-QAM	28
3.3	Demapeador <i>soft</i> 16-QAM	30
3.4	Decodificador Hamming	34
3.5	Exemplo de codificação e decodificação	37
4	RESULTADOS	41
5	CONCLUSÃO	48
	REFERÊNCIAS	49

1 INTRODUÇÃO

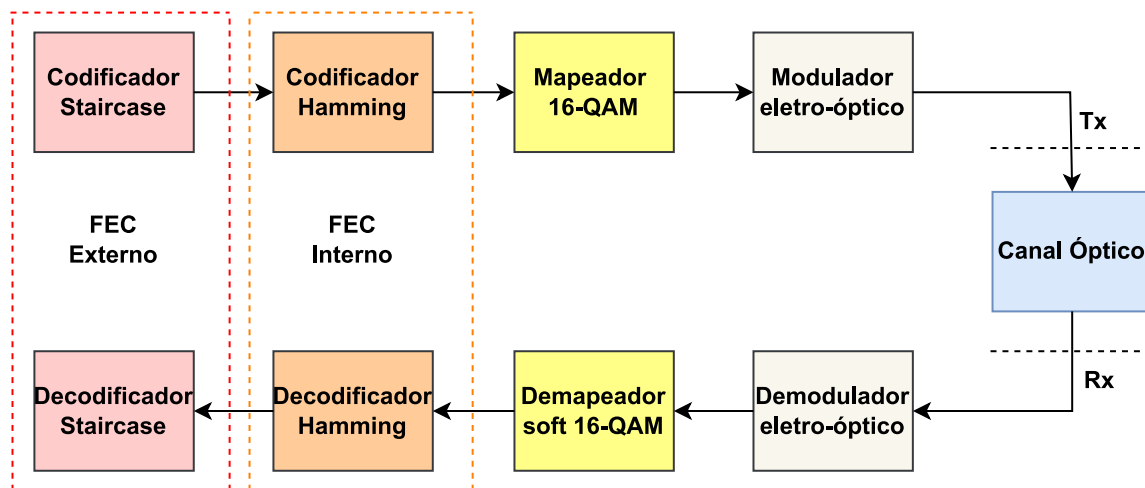
1.1 MOTIVAÇÃO

Devido ao aumento de requerimento de largura de banda nos centros de dados, influenciados pelo crescimento da computação em nuvem, tanto a indústria como a academia vem dedicando um esforço significativo no desenvolvimento dos sistemas de comunicações ópticas ultra rápidos. Nesse contexto, o padrão de rede óptica de longo alcance para 400 Gbps foi estabelecido pela União Internacional de Telecomunicações (ITU - *International Telecommunication Union*)(UNION, 2020).

Com base nesse padrão, o acordo de implementação 400ZR foi desenvolvido pelo Fórum de Interconexão Óptica (OIF - *Optical Internetworking Forum*) para que se projetem módulos ópticos compatíveis para interconexão dos centros de dados (OIF-400ZR-01.0, 2020). Tais módulos ópticos são utilizados para aumentar a largura de banda desses sistemas. O acordo de implementação do 400ZR estabelece valores para os parâmetros e para os requerimentos básicos de um sistema de comunicações ópticas coerente visando taxas de transmissão de 400 Gbps. Em particular, seu objetivo é padronizar o design de transceptores ópticos para aplicações empregando multiplexação por divisão de comprimento de onda densa (DWDM - *Dense Wavelength Division Multiplexing*) de curto alcance (menos de 120 km). Os requisitos para se projetar um transceptor óptico baseado na norma 400ZR incluem alto rendimento (400 Gbps em portadora única) e baixo consumo de energia.

O uso de processador digital de sinais (DSP - *Digital Signal Processor*) de alto desempenho permite a implementação de sofisticados códigos corretores de erros. Estas técnicas são cruciais, pois quando a informação é transmitida, o canal introduz distorções no sinal que pode acabar por corromper esta informação e diminuir a confiabilidade do sistema de comunicação (MOREIRA; FARRELL, 2006). A utilização dos códigos de correção de erros avançada (FEC - *Forward Error Correction*) é amplamente explorada nesses sistemas por viabilizar a implementação de *hardware* eficiente e consequentemente a adoção em escala comercial.

Figura 1 – Arquitetura de um sistema de comunicação com staircase/Hamming FEC concatenado.



fonte: (TRUHACHEV et al., 2020).

Assim, o acordo de implementação define um FEC concatenado (CFEC - *Concatenated FEC*) para o sistema 400ZR. O CFEC é composto por dois códigos de bloco sendo eles: o código de Hamming duplamente estendido que atua como um passo intermediário entre o demapeador e o código *staircase*, como ilustrado na Figura 1. O código de Hamming possui uma capacidade de correção de pelo menos um erro ($d_{min} = 4$), podendo corrigir uma maior quantidade de erros devido sua característica *soft*. Seu decodificador com entrada *soft* e saída *hard*, tem a função de reduzir a razão de erros de bit (BER - *Bit Error Ratio*) da saída do demapeador de constelação para um nível em que o decodificador do *staircase* opera de forma satisfatória, resultando em uma comunicação praticamente livre de erros, provendo uma BER abaixo de 10^{-15} . Desta forma o CFEC pode atingir um ganho de codificação (NCG - *Net Coding Gain*) de 10,8 dB (TRUHACHEV et al., 2020).

O desempenho dos códigos FEC e sua capacidade para satisfazer os requerimentos de padrões específicos são limitados especialmente por sua implementação em *hardware*. Portanto o desenvolvimento eficiente desses códigos é de suma importância. Os decodificadores dos códigos de Hamming com entrada *soft* e saída *hard*, costumam utilizar do algoritmo de Chase (CHASE, 1972) ou algumas de suas variações como, por exemplo, o algoritmo de decodificação Chase II, onde a etapa de decodificação de distância limitada é realizada para um número estendido de padrões de teste em sua implementação (CAI et al., 2019). Por conta das restrições impostas pela eficiência energética, inerentes da implementação em *hardware*, ainda existem poucas publicações sobre algoritmos para o decodificador que sejam compatíveis para o projeto. Portanto, algoritmos alternativos são explorados e analisados. O algoritmo Teste de Padrão de Erros Direto (DEPT - *Direct Error-Pattern Testing*) é apresentado em (TRUHACHEV et al., 2020), sendo um algoritmo promissor para a implementação no decodificador *soft* do código de Hamming proposto.

Existem três principais parâmetros que governam o desempenho do algoritmo DEPT: os padrões de testes, o número de bits de resolução (quantização) e o limiar utilizado. Deve se realizar uma otimização deste limiar, analisando seu efeito no número de bits da resolução. Esse parâmetro requer uma análise mais detalhada durante o desenvolvimento deste trabalho.

1.2 CONTRIBUIÇÃO

1. Nessa monografia implementamos e analisamos o processo de codificação e decodificação do código de Hamming duplamente estendido para o sistema de comunicação óptica 400ZR da recomendação ITU-T 709.2/Y.1331.2 do padrão 400 Gbps.
2. Apresentamos um algoritmo de decodificação DEPT com entrada *soft* e saída *hard*, alternativo aos algoritmos de Chase para a implementação do decodificador do código de *Hamming*.
3. Comprovamos que o algoritmo proposto é compatível, atingindo os resultados satisfatórios que atendem a normas dos sistemas de comunicações ópticas 400ZR, padronizados pela OIF.
4. O algoritmo proposto utiliza de uma abordagem diferente aos padrões de testes, permitindo a escolha de padrões menores, de acordo com a necessidade de operação. Além de possuir um parâmetro que foi otimizado para um melhor desempenho de acordo com a aplicação. Com esses aspectos a eficiência energética do sistema pode melhorar.

1.3 ORGANIZAÇÃO DA MONOGRAFIA

1. O capítulo 2 revisa códigos corretores de erros aplicados aos sistemas de comunicações ópticas. Diferentes tipos de códigos de correção de erros relacionados ao acordo de implementação OIF-400ZR também são revisados, assim como alguns conceitos e algoritmos relacionados.
2. O capítulo 3 apresenta o processo de codificação e decodificação detalhado do código de Hamming duplamente estendido, com a utilização do algoritmo DEPT para decodificação. Assim como, apresenta uma implementação simplificada do mapeador 16-QAM e do demapeador *soft* 16-QAM.
3. O capítulo 4 demonstra os resultados obtidos das simulações e a estrutura utilizada para a geração dos mesmos.
4. O capítulo 5 conclui a monografia.

2 REFERENCIAL TEÓRICO

2.1 CÓDIGOS CORRETORES DE ERROS EM SISTEMAS DE COMUNICAÇÕES ÓPTICAS

2.1.1 Sistemas de comunicações ópticas

Os sistemas de comunicação por fibra óptica modernos são geralmente sistemas de comunicação digital. A principal diferença para os sistemas tradicionais é o meio em que a informação é transmitida, onde os sistemas tradicionais utilizam cobre. Os sistemas ópticos usam a luz como forma de transmitir informação, de um transmissor para um receptor, por meio de um canal de fibra óptica. Existem diversas vantagens neste sistema de comunicação quando comparados aos sistemas tradicionais, isso inclui uma largura de banda muito maior, transmissão mais rápida e um custo de implementação menor. Além do canal de fibra óptica ser imune às interferências eletromagnéticas, às quais são um fator comum de ruído nos sistemas que utilizam o cobre como meio de transmissão (AGRAWAL, 2010).

Em um sistema de comunicação óptica, após a transmissão do sinal, o mesmo pode ser detectado no receptor por duas formas, uma sendo a detecção direta e a outra a detecção coerente. Na forma direta, a informação do sinal é obtida pela amplitude do sinal óptico recebido, utilizando de fotodiodos para a conversão do domínio óptico para o elétrico. A detecção coerente, diferente da direta, pode-se obter a informação do sinal transmitido através de outras características além da amplitude, sendo elas a frequência, a fase e a polarização. Essas grandezas são convertidas em um sinal elétrico. Assim, utilizando-se de um conversor analógico/digital (ADC - *Analog to Digital Converter*), o sinal pode vir a ser processado por um processador digital de sinais (DSP) que é responsável por compensar digitalmente os efeitos não lineares degradantes inseridos na propagação do sinal pela fibra óptica. Os DSPs podem evitar desperdícios de recursos no tratamento analógico do sinal, além de suportar diversas formas de modulação (ZHOU; XIE, 2016).

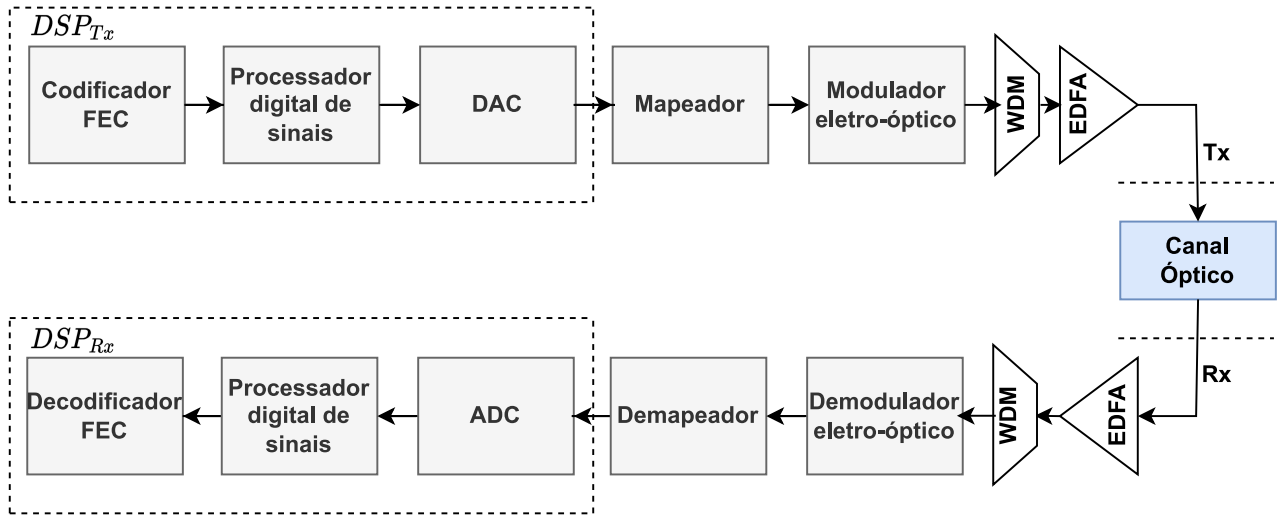
Portanto, nos sistemas de comunicação óptica com detecção coerente, a informação que será transmitida passa por um codificador FEC para, em seguida, ser processada por um DSP no lado da transmissão (Tx). O sinal de saída será convertido da forma digital para analógica por meio de um conversor digital/analógico (DAC - *Digital to Analog Converter*) onde sua saída é mapeada e será utilizada nos moduladores que são responsáveis por realizar a modulação do sinal óptico. No domínio óptico, o sinal passa por uma multiplexação por divisão de comprimento de onda (WDM - *Wavelength Divison Multiplexing*) e então é amplificado por um amplificador de fibra dopado com érbio (EDFA - *Erbium-Doped Fiber Amplifier*). Seguindo o processo, no receptor, o sinal óptico é demodulado em um sinal elétrico analógico e é convertido para digital por meio de um ADC. Permitindo assim que o mesmo seja processado por um DSP no lado da recepção (Rx), sendo esse responsável por compensar de forma digital os efeitos degradantes.

A BER, representada pela Equação 1, é utilizada como uma forma de medir o nível de degradação do sinal transmitido, sendo uma métrica importante para o projeto de um sistema de comunicação. De forma simplificada, o sistema de comunicação óptica coerente com a presença de um DSP, pode ser representado por um conjunto blocos no lado do transmissor e no lado do receptor, como mostrado na

Figura 2.

$$BER = \frac{\text{número de bits incorretos}}{\text{número de bits transmitidos}} \quad (1)$$

Figura 2 – Representação de um sistema de comunicação óptica coerente com a presença de um DSP.



fonte: Produção do próprio autor.

Portanto, no projeto de um DSP, o mesmo é responsável por entregar um valor máximo de BER, que será o valor máximo de erros na transmissão de bits que o código corretor de erros será capaz de corrigir. Esse valor é conhecido por ser o limite do FEC (ZHOU; XIE, 2016).

2.1.2 Códigos Corretores de Erros

Nos sistemas de comunicações, a presença de maneiras de corrigir erros é essencial. A informação transmitida pode vir a ser alterada no processo de transmissão pelo canal, acarretando assim uma informação errada, que por sua vez acaba por comprometer a confiabilidade do sistema de comunicação. O método aplicado para a correção de erros em sistemas de comunicações digitais, são os códigos FEC. Esses códigos auxiliam o sistema a detectar e corrigir determinado número de dados corrompidos recebidos pelo receptor, sem a necessidade de retransmissão (MOREIRA; FARRELL, 2006).

O codificador FEC adiciona bits de redundância, ou bits de paridade, na informação a ser transmitida, através de um processo chamado de codificação. Após esse processo, o codificador gera uma palavra-código (*codeword*). No caso de um sistema de comunicação coerente digital, a palavra-código (informação codificada) será processada pelo DSP_{Tx}, mapeada e modulada por um modulador eletro-óptico para, enfim, ser transmitida pelo canal óptico até o receptor.

No receptor, o caminho inverso será realizado. A palavra-código, corrompida pelo ruído, é recebida e será demodulada por um demodulador eletro-óptico, desmapeada, processada pelo DSP_{Rx} e então decodificada pelo decodificador FEC. A palavra-código passará por um processo de correção de erros realizado pelo decodificador e, no final desse processo, seus bits de redundância serão removidos e a informação será finalmente recuperada, livre de erros. A Figura 2 demonstra a presença do código FEC em um sistema de comunicação óptica.

Os dois tipos de códigos FEC mais utilizados são os códigos de bloco e os códigos convolucionais. A diferença entre eles está na codificação dos mesmos. Os códigos de bloco codificam os bits de informação em um bloco com tamanho fixo, seguido pelos bits de paridade. Os códigos convolucionais utilizam da informação atual e também da informação anterior para realizar a sua codificação (MOON, 2005). Exemplos de códigos de blocos são: os códigos de Hamming e o código Bose–Chaudhuri–Hocquenghem codes (BCH).

2.1.3 Métricas de desempenho de um FEC

As formas de avaliar os códigos FEC incluem a taxa do código, o ganho do código (NCG - *Net Coding Gain*), sua complexidade e a razão de erros de bit (BER) de entrada que um código corretor de erros precisa para atingir uma determinada BER de saída.

A taxa do código se refere à proporção de dados que são de informação e não de paridade. O NCG é a medida na diferença entre os níveis de relação sinal-ruído (SNR - *Signal to Noise Ratio*), entre o sistema não codificado e o sistema codificado necessário para atingir os mesmos níveis de BER quando usado com o código de correção de erro (LI, 2019). O NCG possui dois parâmetros relacionados à perda de codificação: o primeiro sendo o aumento de potência devido aos bits de paridade adicionados no codificador e que são necessários para manter a mesma BER que o canal não codificado; o segundo sendo o ganho de transferência do código que é a diminuição de potência, devido à correção de erro do decodificador, necessária para obter um BER de saída do decodificador igual ao BER de entrada do decodificador.

$$NCG = \text{ganho de transferência} - \text{perda de codificação}, \quad (2)$$

$$NCG[dB] = \text{ganho de transferência}[dB] - 10 \cdot \log_{10}(n/k). \quad (3)$$

O limite de Shannon define o máximo de taxa de informação possível em um sistema (SHANNON, 1998).

2.2 ESTRUTURA BÁSICA DE CÓDIGOS CORRETORES DE ERROS

2.2.1 Construção do Corpo Finito

Um corpo finito ou corpo de Galois é definido como um corpo que possui um número finito de elementos. O corpo é uma estrutura algébrica que segue certas propriedades de adição e multiplicação (MOREIRA; FARRELL, 2006).

Para um determinado número primo p , um corpo de Galois pode ser definido e denotado como $GF(p^m)$ para qualquer m inteiro. O polinômio $b_{m-1}x^{m-1} + \dots + b_1x^1 + b_0$ é usado para representar o corpo $GF(p^m)$ onde o coeficiente b_k está no intervalo de $\{0, 1, \dots, p-1\}$ (LI, 2019).

Em termos da implementação do corpo Galois nos códigos de correção de erros, o corpo binário $GF(2^m)$ é comumente aplicado. De fato, as operações presentes neste trabalho, são realizadas sobre esse corpo. A multiplicação e a adição são duas operações aritméticas frequentemente usadas sobre

$GF(2^m)$. A multiplicação sobre $GF(2^m)$ é a mesma que uma multiplicação módulo-2 ou uma operação lógica *XOR*. Já a adição sobre $GF(2^m)$ é uma operação lógica *AND*, ou equivalentemente, uma adição de módulo-2.

Tabela 1 – Multiplicação e adição sobre $GF(2^m)$.

x	0	1
0	0	0
1	0	1

+	0	1
0	0	1
1	1	0

fonte: (MOREIRA; FARRELL, 2006)

Seja α a raiz de um polinômio primitivo $p(X)$ (polinômio cujos coeficientes tem máximo divisor comum igual a um) de grau m . Os elementos não nulos de $GF(2^m)$ podem então ser representados por elementos de corpo $\{0, 1, \alpha, \alpha^2, \dots, \alpha^{2^m-2}\}$. Esse polinômio pode ser representado como $p(X) = p_0 + p_1X + p_2X^2 + \dots + p_mX^m$, onde cada coeficiente p_i é binário e $p_0 = 1$, pois caso contrário, o polinômio será fatorável por X , assim não sendo irredutível nem primitivo. Para se ter a ordem m é necessário que $p_m = 1$. O polinômio irredutível é um polinômio que não se pode ser fatorado, portanto o mesmo só é primitivo se e somente se o menor número inteiro positivo n para o qual $p(X)$ é um divisor de $X^n + 1$ para $n = 2^m - 1$ (MOON, 2005).

Por exemplo, para compor um $GF(2^3)$, podemos utilizar um polinômio primitivo de grau $m = 3$: $p(X) = 1 + X + X^3$. Portanto, os elementos de corpo ou forma exponencial do corpo é: $GF(2^3) = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$, onde $2^m - 2 = 2^3 - 2 = 6$ é a última potência de α . A representação polinomial é obtida através de $\alpha^i \text{ mod } (p(X))$ onde $i = 1, 2, \dots, 6$ e $X = \alpha$. A Tabela 2 demonstra as representações do corpo $GF(2^3)$ (MOREIRA; FARRELL, 2006).

Tabela 2 – Representação do corpo $GF(2^3)$.

Representação Exp.	Representação Pol.	Representação Vet.
0	0	0 0 0
1	1	1 0 0
α	α	0 1 0
α^2	α^2	0 0 1
α^3	$1 + \alpha$	1 1 0
α^4	$\alpha + \alpha^2$	0 1 1
α^5	$1 + \alpha + \alpha^2$	1 1 1
α^6	$1 + \alpha^2$	1 0 1

fonte: (MOREIRA; FARRELL, 2006).

2.2.2 Códigos de Bloco

A codificação em bloco de um conjunto de bits baseia-se em agrupar e processar este conjunto em blocos de k bits, denominados bits de mensagem. Assim, temos 2^k mensagens possíveis. O codificador é responsável por processar cada bloco de k bits e converter o mesmo em um bloco mais longo de $n > k$ bits codificados, os quais são denominados de palavra código. Nesse procedimento existem $(n - k)$ bits que são adicionados pelo codificador à palavra de mensagem. Esses bits são de paridade ou de redundância. O controle de erros na codificação requer o uso desse mecanismo para adicionar

redundância à mensagem. Essa redundância pode ser adicionada de diferentes maneiras. Porém, em todo caso, deve permitir que na sua operação inversa (decodificação) seja possível que a informação da mensagem seja recuperada com sucesso. No decodificador, é realizado o procedimento de correção de erro e, em seguida, o descarte dos bits de paridade, pois eles não possuem informação de mensagem (MOREIRA; FARRELL, 2006).

Os códigos com tal estrutura são chamados de códigos de bloco e são denotados por $Cb(n, k)$. A taxa do código é $Rc = k/n$, sendo essa uma medida do nível de redundância aplicada em um determinado código. Ela é a razão dos bits codificados que representam a informação ou a mensagem e está diretamente relacionada com o aumento de largura de banda necessária na transmissão quando a codificação é usada (MOREIRA; FARRELL, 2006).

Definição 1: Um código de bloco $Cb(n, k)$ sobre um alfabeto $\mathcal{A}$ de q símbolos é um conjunto de q^k vetores chamados de palavras-código. Associado ao código está um codificador que mapeia uma mensagem, uma k -tupla $m \in \mathcal{A}^k$, para sua palavra código associada. (MOON, 2005, p.83)

Para uma dada taxa de transmissão, a ocupação espectral é sempre maior no caso de uma mensagem codificada. De forma geral, armazenar informações codificadas exigirá mais espaço do que o da informação não codificada. Assim, uma consideração prática é manter a taxa do código em um determinado nível, embora em geral seja levado em consideração uma troca com respeito à capacidade de correção de erros do código (MOREIRA; FARRELL, 2006).

Para que um código de bloco seja útil para fins de correção de erros, deve haver uma correspondência entre uma mensagem m e sua palavra-código c . No entanto, para um determinado código $Cb(n, k)$, pode haver mais de uma maneira possível de se mapear mensagens para palavras-código.

Um código de bloco pode ser representado como uma lista exaustiva, mas para um k muito grande isso seria extremamente complexo para armazenar e decodificar. A complexidade pode ser reduzida impondo algum tipo de estrutura matemática no código.

O requisito mais comum é a linearidade, que nos leva aos códigos de bloco lineares (MOON, 2005). Nesses códigos a informação que será codificada e agrupada em um bloco de k -bits, que constitui uma mensagem genérica $m = (m_0, m_1, m_2, \dots, m_{k-1})$ sendo uma das 2^k mensagens possíveis. O codificador recebe tal mensagem e gera uma palavra código de n componentes $c = (c_0, c_1, c_2, \dots, c_{n-1})$, $n > k$, adicionando redundância. A linearidade das operações simplifica o procedimento de codificação das mensagens.

Codificar é relacionar as 2^k mensagens binárias de k bits a algum dos 2^n vetores de n bits. Como sempre $k < n$, existem mais vetores de n bits do que vetores de k bits. Então a seleção de vetores de n bits deve ser feita utilizando o nível mais baixo de redundância e maximizando a distância entre as palavras-código.

Definição 2: Um código de bloco de comprimento n e 2^k mensagens é dito ser um código de bloco linear $C_b(n, k)$ se as 2^k palavras do código formam um subespaço vetorial, de dimensão k , do espaço vetorial V_n de todos os vetores de comprimento n com componentes no corpo $GF(2)$. (MOREIRA; FARRELL, 2006, p.50)

O conjunto de 2^k palavras-código constitui um subespaço vetorial do conjunto de palavras de n bits. Como uma consequência de sua definição, um código de bloco linear é caracterizado pelo fato de que a soma de qualquer uma das duas palavras de código também é uma palavra-código.

2.2.2.1 Matriz Geradora $\mathbf{G}_{[k \times n]}$

Um código de bloco linear $C_b(n, k)$ é um espaço vetorial de dimensão k , portanto haverá k vetores linearmente independentes, que por sua vez, são palavras-código $g_0, g_1, \dots, g_{k-1}$, tal que cada palavra de código c em $C_b(n, k)$ pode ser representado como uma combinação linear desses vetores:

$$\mathbf{c} = m_0 \bullet \mathbf{g}_0 \oplus m_1 \bullet \mathbf{g}_1 \oplus \dots \oplus m_{k-1} \bullet \mathbf{g}_{k-1} \quad (4)$$

Assim, agrupando os valores de g_k , onde $g_{k,n}$ é o n -ésimo elemento da k -ésima palavra. Podemos definir a matriz $\mathbf{G}$ como:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{0,0} & \mathbf{g}_{0,1} & \cdots & \mathbf{g}_{0,n-1} \\ \mathbf{g}_{1,0} & \mathbf{g}_{1,1} & \cdots & \mathbf{g}_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \mathbf{g}_{k-1,0} & \mathbf{g}_{k-1,1} & \cdots & \mathbf{g}_{k-1,n-1} \end{bmatrix}. \quad (5)$$

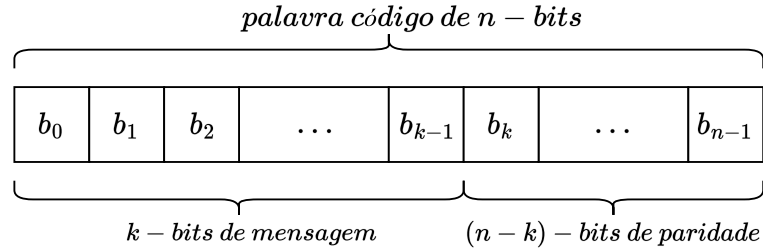
Então dado um vetor mensagem $\mathbf{m} = (m_0, m_1, \dots, m_{k-1})$, a sua palavra-código correspondente pode ser obtida através da multiplicação matricial:

$$\mathbf{c} = \mathbf{m} \circ \mathbf{G}_{[k \times n]} = m_0 \bullet \mathbf{g}_0 \oplus m_1 \bullet \mathbf{g}_1 \oplus \dots \oplus m_{k-1} \bullet \mathbf{g}_{k-1}. \quad (6)$$

O símbolo ” $\circ$ ” representa o produto interno entre os vetores ou matrizes, já o símbolo ” $\bullet$ ” representa a multiplicação por um escalar no corpo finito $GF(2)$ e ” $\oplus$ ” representa a adição no corpo $GF(2)$. Essas operações foram definidas na subseção 2.2.1.

O código linear $C_b(n, k)$ é caracterizado pela matriz $\mathbf{G}$. Portanto $\mathbf{G}_{[k \times n]}$ é chamada de matriz geradora de $C_b(n, k)$. A Equação 6 pode ser pensada como uma operação de codificação da mensagem. É conveniente gerar $\mathbf{G}_{[k \times n]}$ em sua forma sistemática. Nessa forma as palavras-código $\mathbf{c}$ serão constituídas por $(n - k)$ bits de paridade seguidos de k bits de mensagem. A Figura 3 demonstra a estrutura final de uma palavra-código, a convenção da posição dos bits de paridade não modifica as propriedades do código de blocos.

Figura 3 – Representação de uma palavra-código na forma sistemática.



fonte: Produção do próprio autor.

A forma sistemática do código de blocos linear $C_b(n, k)$ é especificada por uma matriz geradora composta por duas sub-matrizes, sendo uma matriz de paridade $\mathbf{P}$ e a matriz identidade $\mathbf{I}_k$.

$$\mathbf{G}_{[k \times n]} = \begin{bmatrix} \mathbf{P}_{[k \times (n-k)]} & \mathbf{I}_{[k \times k]} \end{bmatrix}. \quad (7)$$

2.2.2.2 Matriz verificadora de paridade $\mathbf{H}_{[(n-k) \times k]}$

A matriz verificadora de paridade $\mathbf{H}$ de um código corretor $Cb(n, k)$ é uma matriz crítica para o sistema. Sua principal função é verificar se uma palavra-código pertence ou não ao seu código Cb .

Se considerarmos a **Definição 2**, apresentada na subseção 2.2.2, as linhas das matrizes $\mathbf{G}$ e $\mathbf{H}$ são ortogonais (MOREIRA; FARRELL, 2006). Portanto, a matriz verificadora de paridade $\mathbf{H}$ em sua forma sistemática é formada pela matriz de paridade transposta $\mathbf{P}^T$ combinada com uma matriz identidade $\mathbf{I}_{(n-k)}$:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}_{[(n-k) \times (n-k)]} & \mathbf{P}_{[(n-k) \times k]}^T \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 & p_{0,0} & p_{1,0} & \cdots & p_{k-1,0} \\ 0 & 1 & \cdots & 0 & p_{0,1} & p_{1,1} & \cdots & p_{k-1,1} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & p_{0,n-k-1} & p_{1,n-k-1} & \cdots & p_{k-1,n-k-1} \end{bmatrix}. \quad (8)$$

Como demonstrado em (MOON, 2005), pelo fato de que as linhas de $\mathbf{G}$ e $\mathbf{H}$ serem vetores linearmente independentes, temos as seguintes expressões:

$$\mathbf{G} \circ \mathbf{H}^T = 0, \quad (9)$$

$$\mathbf{c} \circ \mathbf{H}^T = \mathbf{m} \circ \mathbf{G} \circ \mathbf{H}^T = 0. \quad (10)$$

2.2.3 Detecção de Erro por Síndrome

Como consequência de transmitir uma palavra-código $\mathbf{c} = (c_0, c_1, \dots, c_{n-1}) \in GF(2)$ por um canal ruidoso, a palavra-código recebida, pode ter alguns erros. Ela será diferente da transmitida e então denotamos a palavra-código recebida como $\mathbf{r} = (r_0, r_1, \dots, r_{n-1})$, a qual também $\in GF(2)$.

Um evento de erro pode ser modelado como um vetor de erro ou padrão de erro $\mathbf{e} = (e_0, e_1, \dots, e_{n-1})$, cujos componentes também são definidos sobre o corpo binário, $e \in GF(2)$, e que está relacionado à palavra de código e os vetores recebidos da seguinte forma:

$$\mathbf{e} = \mathbf{r} \oplus \mathbf{c}. \quad (11)$$

O padrão de erros possui componentes não nulos nas posições onde o erro ocorreu. Portanto, com o padrão de erros determinado, o decodificador tem a função de estimar uma palavra-código válida, através da seguinte expressão:

$$\mathbf{c} = \mathbf{r} \oplus \mathbf{e}. \quad (12)$$

Sabendo que toda a palavra-código segue a condição da Equação 10, uma forma de detectar erros pode ser implementada, baseada na seguinte expressão:

$$\mathbf{S} = \mathbf{r} \circ \mathbf{H}^T = (s_0, s_1, \dots, s_{n-k-1}). \quad (13)$$

Esse vetor $\mathbf{S}$ é denominado vetor de síndrome. Quando o mesmo apresenta valores não nulos em suas componentes, significa que houve erro na transmissão da palavra-código. Caso o vetor de síndrome seja totalmente nulo $\mathbf{S} = 0$, temos uma palavra-código válida.

2.3 CÓDIGOS DE HAMMING

2.3.1 Códigos de Hamming Convencionais

Os códigos de Hamming $Ham(n, k, d_{min})$, criados por Richard W. Hamming (HAMMING, 1950), são a primeira classe importante de códigos de correção de erros lineares. Esses códigos têm sido amplamente utilizados para controle de erros em comunicação digital e armazenamento de dados. Eles têm propriedades interessantes que facilitam as operações de codificação e decodificação. Discutiremos sobre os códigos de Hamming binários, definidos sobre o corpo $GF(2)$.

Para qualquer inteiro $m \geq 3$, existe um código de Hamming capaz de corrigir qualquer erro único ao longo do comprimento do bloco e possuem as seguintes características:

- Comprimento: $n = 2^m - 1$;
- Número de bits de mensagem: $k = 2^m - m - 1$;
- Número de bits de paridade: $n - k = m$;
- Distância mínima de Hamming: $d_{min} = 3$;
- Capacidade de correção de erro: $t = \frac{d_{min}-1}{2} = 1$, onde t é o menor inteiro mais próximo.

A matriz de paridade $\mathbf{H}$ desses códigos é formada por colunas de m bits e podem ser implementadas na forma sistemática:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}_m & \mathbf{Q} \end{bmatrix}. \quad (14)$$

A submatriz de identidade $\mathbf{I}_m$ é uma matriz quadrada de tamanho $m \times m$ e a submatriz $\mathbf{Q}$ consiste de $2^m - m - 1$ colunas formadas de vetores de tamanho 2 ou maior (MOREIRA; FARRELL, 2006).

Na matriz verificadora de paridade $\mathbf{H}$, a soma de três colunas pode resultar em um vetor de zeros e não é possível que a soma de duas colunas dê o mesmo resultado, portanto, a distância mínima do código é $d_{min} = 3$. Isso significa que eles podem ser usados para corrigir qualquer padrão de erro que contenha um erro ou detectar qualquer padrão de erro que possua até dois erros (MOREIRA; FARRELL, 2006). A matriz geradora pode ser construída usando a submatriz $\mathbf{Q}$ da Equação 14, ficando da seguinte forma:

$$\mathbf{G} = \begin{bmatrix} \mathbf{Q}^T & \mathbf{I}_{2^m-m-1} \end{bmatrix}. \quad (15)$$

2.3.2 Códigos de Hamming Estendidos

Um código Hamming $Ham(n, k, d_{min})$ pode ser facilmente estendido adicionando um bit de paridade extra para cada uma de suas palavras-chave de forma a se obter um código $Ham(n', k, d'_{min})$ chamado de código de Hamming estendido, $Ham(m)^*$. O código estendido pode ter uma melhor capacidade de detecção de erros, devido ao aumento da distância de Hamming que implica diretamente na capacidade de correção de um código de bloco.

Podemos estender o código de Hamming, aumentando suas palavras-chave em uma posição. Isso é feito adicionando um novo bit de checagem de paridade, $x_0 \in [0, 1]$, de modo que o peso da nova palavra-código seja par. O resultado dessa adição é o código de Hamming estendido (IBRAHEEM, 2009). O novo bit de paridade, x_0 , é chamado de bit de checagem de paridade. Dada uma palavra-código $\mathbf{c} = (c_0 \ c_1 \ \dots \ c_{n-1}) \in Ham(n, k, d_{min})$, sua versão estendida seria uma nova palavra-código $\mathbf{c}' = (c_0 \ c_1 \ \dots \ c_{n-1} \ x_0) \in Ham(n', k, d'_{min})$ onde n' é um número par.

Teorema: Um código de Hamming estendido $Ham(n', k, d'_{min})$ de um código $Ham(n, k, 3)$ é um código linear $(n', k', 4)$. Onde $n = 2^m - 1, k = 2^m - 1 - m, n' = n + 1, k' = k$ os números 3 e 4 sendo a distância mínima (d_{min}) de cada código respectivamente (IBRAHEEM, 2009).

2.3.3 Algoritmo de decodificação de Chase

O algoritmo de Chase para decodificação *soft* de um código de Hamming é inicializado com uma sequência de n -bits denominado v_{hd} , representando uma decodificação de decisão *hard* realizada nos valores de $\mathbf{I}$, o qual é um vetor das verossimilhanças logarítmicas (LLR - *Log-likelihood Ratio*) recebidas. Por padrão, o valor de LLR nulo (região de indecisão) é decodificado para bit 1. Então, podemos computar os 9-bits de síndrome usando a matriz verificadora de paridade $\mathbf{H}$ do código. Onde as operações de adição e multiplicação estão no campo binário $GF(2)$ (TRUHACHEV et al., 2020).

$$\mathbf{s} = (s_1, s_2, \dots, s_9) = v_{hd} \mathbf{H}^T. \quad (16)$$

No caso $s = 0$, é determinado que v_{hd} é uma palavra-código válida e o algoritmo entende que a saída será $\hat{v} = v_{hd}$. Se $s \neq 0$, o algoritmo parte para uma busca sobre um conjunto de possíveis padrões de erro que se relacionam com as posições dos menores valores entre as LLRs absolutas $|l|$.

Os padrões de erro são gerados em duas etapas. Primeiro um conjunto Γ de padrões de teste t é gerado, em seguida, uma decodificação de distância mínima é realizada em $v_{hd} + t$, executando os seguintes passos (TRUHACHEV et al., 2020):

1. Gerar um conjunto de padrões de teste Γ que consiste em vetores t de n -bits, com um peso de Hamming baixo, que serão utilizados para criar pequenas perturbações em v_{hd} .
2. Passar por todos os padrões de teste $t \in \Gamma$. Para cada $y' = v_{hd} + t$, executar a decodificação de distância mínima de y' para se obter uma palavra-código $\hat{v}$. Isso identifica um padrão de erro e que se relaciona v_{hd} com a palavra-código decodificada $\hat{v}$, ou seja, $e = v_{hd} + \hat{v}$.
3. Identificar o peso analógico de e em termos de LLRs. Se e tem 1's nas posições $j_1, \dots, j_N$ o peso analógico é igual à soma dos valores absolutos das respectivas LLRs:

$$w(e) = |LLR_{j_1}| + |LLR_{j_2}| + |LLR_{j_3}| + \dots + |LLR_{j_{N-1}}| + |LLR_{j_N}|. \quad (17)$$

4. Identificar um padrão de erro e com o menor $w(e)$ entre todos os padrões gerados, através das etapas 2) e 3). Executar a correção de erro com o respectivo padrão de erro e produzindo a saída $\hat{v} = v_{hd} + e$.

O algoritmo de Chase I, faz uma varredura sobre um conjunto de Γ de todos os possíveis padrões de erro que têm peso de Hamming de $[d_{min}/2] = 2$, onde a distância mínima do código de Hamming duplamente estendido é igual a $d_{min} = 4$. A versão Chase II identifica o conjunto L_2 medições com as menores LLRs absolutas com $[d_{min} = 2]$. Então o conjunto de padrões de teste Γ é gerado, tal que, para qualquer padrão de teste $t \in \Gamma$ com no máximo $[(d_{min} - 1)/2] = 1$ que estão localizados fora de L_2 (CHASE, 1972).

2.4 DECODIFICAÇÃO HARD E SOFT

Uma tarefa difícil na correção de erros é decodificar as palavras código que foram recebidas por meio de canais ruidosos. Antes que os dados sejam transmitidos, o codificador adiciona bits de paridade às mensagens que formam a palavra-código. As palavras-código são então transmitidas através de canais. O receptor verifica as palavras-código recebidas e executa o processo de decodificação ou correção de erros para recuperar os dados originais.

Se não houver erros, ou seja, as palavras-código encontrarem uma correspondência exata ($S = 0$ na Equação 13), será fácil decodificar os dados, basta apenas eliminar os bits de paridade da palavra-código recebida. No entanto, se uma correspondência não for encontrada, serão adotados mecanismos de decodificação mais complexos. Curiosamente, para qualquer canal, existem dois limites, um para decodificação usando "*hard data*" e outro para decodificação usando "*soft data*". Essas formas de decodificação são conhecidas como decodificação *hard* e decodificação *soft*.

O sinal transmitido é distorcido pela combinação de transmissor, canal óptico e receptor. O decodificador *hard* toma uma decisão com base na tensão limite. A cada instante de amostragem no receptor, o detector de decisão *hard* determina o estado do bit como "0" se o nível de tensão recebida for inferior ao limite e "1" se o nível de tensão for superior ao limite. Então, se considerarmos um limite de tensão de $2.2V$, qualquer tensão igual ou acima de $2.2V$ é considerado como bit 1 e qualquer tensão abaixo de $2.2V$ é considerado bit 0.

Na decodificação de decisão *soft*, a palavra-código recebida é comparada com todas as palavras-código possíveis e a palavra-código que fornece os melhores parâmetros é selecionada. Assim, a decodificação de decisão *soft* melhora o processo de tomada de decisão, fornecendo informações adicionais de confiabilidade (distância euclidiana calculada ou razão de probabilidade logarítmica calculada) (DOKMANIC et al., 2015). Existem diversos algoritmos de decodificação *soft*. O decodificador de decisão *soft* calcula as distâncias euclidianas entre as tensões recebidas e as tensões das palavras-código recebidas. Em seguida, encontra a distância euclidiana mínima ou a razão de probabilidade logarítmica e seleciona a palavra-código correspondente.

3 DESENVOLVIMENTO

Neste capítulo, os módulos responsáveis pelo processo de codificação e decodificação do código de Hamming (128,119) duplamente estendidos serão apresentados e discutidos. Dois módulos auxiliares, mapeador e demapeador, necessários para a validação do código FEC, também são apresentados.

3.1 CODIFICADOR DO CÓDIGO DE HAMMING DUPLAMENTE ESTENDIDO

Neste trabalho, a principal tarefa do codificador de Hamming é computar a matriz geradora do código baseada na recomendação do 400ZR da OIF (OIF-400ZR-01.0, 2020). A matriz geradora G e a matriz verificadora de paridade H do código de Hamming duplamente estendido (128,119) são definidas e utilizadas em sua forma sistemática.

A norma define que a matriz verificadora de paridade, considerando uma função g , representada pela Equação 1, a qual mapeia um inteiro i , $0 \leq i \leq 127$ que é permutado pela função Π_d , definida pela Tabela 3, em um vetor coluna:

$$g(i) = \begin{bmatrix} s_{0,i} \\ s_{1,i} \\ \vdots \\ s_{6,i} \\ s_{7,i} \\ 1 \end{bmatrix}, \quad (1)$$

$$i = 64s_{6,i} + 32s_{5,i} + 16s_{4,i} + 8s_{3,i} + 4s_{2,i} + 2s_{1,i} + s_{0,i}, \quad (2)$$

$$s_{7,i} = (s_{0,i} \wedge s_{2,i}) \vee (\bar{s}_{0,i} \wedge \bar{s}_{1,i} \wedge \bar{s}_{2,i}) \vee (s_{0,i} \wedge s_{1,i} \wedge \bar{s}_{2,i}). \quad (3)$$

Sendo $\wedge$ uma operação binária *AND*, $\vee$ a operação binária *OR* e $\bar{x}$ um *NOT* binário.

Portanto a matriz verificadora de paridade é uma matriz binária 9×128 :

$$H = \begin{bmatrix} g(\Pi_d(0)) & g(\Pi_d(1)) & g(\Pi_d(2)) & \dots & g(\Pi_d(127)) \end{bmatrix}. \quad (4)$$

Tabela 3 – Função de permutação Π_d

$\Pi_d(0 : 62) = 0 : 62$	$\Pi_d(63 : 93) = 64 : 94$	$\Pi_d(94 : 108) = 96 : 110$
$\Pi_d(109 : 115) = 112 : 118$	$\Pi_d(116) = 120$	$\Pi_d(117) = 122$
$\Pi_d(118) = 124$	$\Pi_d(119) = 63$	$\Pi_d(120) = 95$
$\Pi_d(121) = 111$	$\Pi_d(122) = 119$	$\Pi_d(123) = 121$
$\Pi_d(124) = 123$	$\Pi_d(125 : 127) = 125 : 127$	

fonte: (OIF-400ZR-01.0, 2020)

Sendo que $\Pi_d(a : b)$ representa:

$$[\Pi_d(a), \Pi_d(a+1), \Pi_d(a+2), \dots, \Pi_d(b)]. \quad (5)$$

Tabela 4 – Função de permutação inversa Π_d^{-1}

$\Pi_d^{-1}(0 : 62) = 0 : 62$	$\Pi_d^{-1}(63) = 119$	$\Pi_d^{-1}(64 : 94) = 63 : 93$
$\Pi_d^{-1}(95) = 120$	$\Pi_d^{-1}(96 : 110) = 94 : 108$	$\Pi_d^{-1}(111) = 121$
$\Pi_d^{-1}(112 : 118) = 109 : 115$	$\Pi_d^{-1}(119) = 122$	$\Pi_d^{-1}(120) = 116$
$\Pi_d^{-1}(121) = 123$	$\Pi_d^{-1}(122) = 117$	$\Pi_d^{-1}(123) = 124$
$\Pi_d^{-1}(124) = 118$	$\Pi_d^{-1}(125 : 127) = 125 : 127$	

fonte: (OIF-400ZR-01.0, 2020)

Para se obter a matriz geradora $\mathbf{G}$ do codificador, os seguinte cálculos devem ser realizados:

$$\mathbf{B} = \left[g(63), g(95), g(111), g(119), g(121), g(123), g(125) : g(127) \right]^{-1}, \quad (6)$$

$$\mathbf{P} = \mathbf{B} \left[g(0) : g(62), g(64) : g(94), g(96) : g(110), g(112) : g(118), g(120), g(122), g(124) \right]. \quad (7)$$

Finalmente, a matriz geradora do código de Hamming é:

$$\mathbf{G} = \begin{bmatrix} \mathbf{I} & \mathbf{P}^T \end{bmatrix}. \quad (8)$$

Se temos uma mensagem de 119 bits $\mathbf{m} = [m_0, m_1, \dots, m_{118}]$, podemos obter uma palavra-código $\mathbf{c}$ por meio da seguinte expressão:

$$\mathbf{c} = \mathbf{m} \circ \mathbf{G} = [c_0, c_1, \dots, c_{118}, c_{119}, \dots, c_{127}]. \quad (9)$$

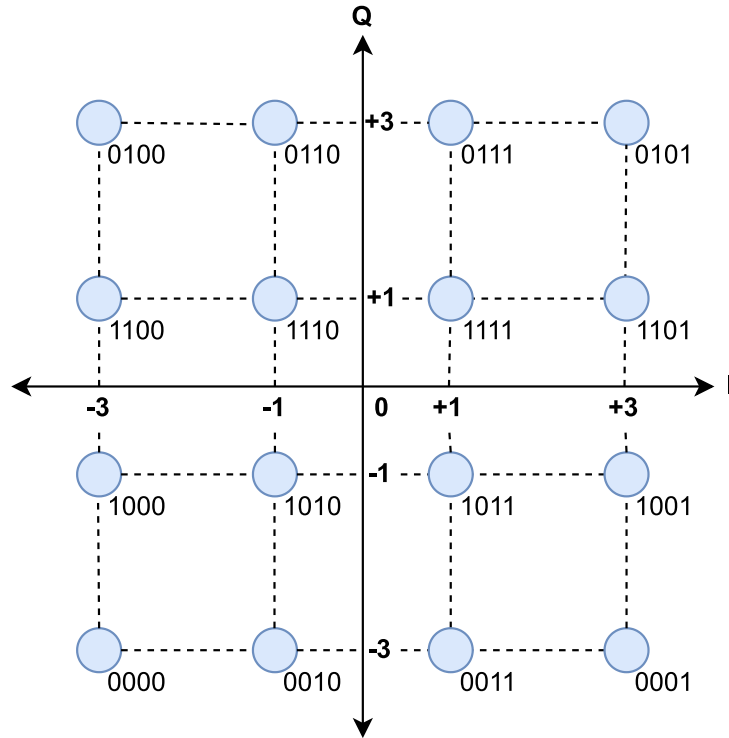
Dessa forma obtemos uma palavra-código de 128 bits do código de Hamming duplamente estendido, onde os bits mais significativos são referentes aos bits de paridade.

3.2 MAPEADOR 16-QAM

Pelo fato do sistema 400ZR (OIF-400ZR-01.0, 2020) utilizar de modulação 16-QAM, onde os símbolos são mapeados em um diagrama de fase (I) e quadratura (Q), cada símbolo apresenta uma distância específica da origem do diagrama e essa distância representa sua amplitude (energia do sinal). Isto significa que as informações são inseridas nos parâmetros de amplitude e quadratura da onda portadora. O mapeador 16-QAM é responsável por realizar o mapeamento dos símbolos em uma constelação 16-QAM seguindo a codificação de Gray (OIF-400ZR-01.0, 2020).

O mapeamento da palavra-código do Hamming (128, 119) duplamente estendido possui 16 símbolos de 8-bits. Cada um dos símbolos representa duas polarizações X e Y , cada polarização recebe 4-bits e são mapeados nas componentes $X(I)$, $X(Q)$, $Y(I)$ e $Y(Q)$. Cada componente da polarização é alocada em um dos 16 quadrantes do diagrama de uma constelação 16-QAM. A Figura 4 é um exemplo da constelação 16-QAM seguindo o mapeamento Gray.

Figura 4 – Constelação 16-QAM com símbolos mapeados usando codificação de Gray.



fonte: Produção do próprio autor.

Assim, para cada palavra-código teremos 16 símbolos DP-16QAM.

$$S = [S_0, S_1, \dots, S_{15}]. \quad (10)$$

Como cada símbolo possui 8 bits correspondentes a uma parcela da palavra-código, podemos representá-los da seguinte forma:

$$S_i = [c_{8i}, c_{8i+1}, c_{8i+2}, \dots, c_{8i+7}], \quad (11)$$

com $i = 0, 1, \dots, 15$. Em que,

- (c_{8i}, c_{8i+1}) mapeia para a componente em fase (I) da polarização X de S_i .
- (c_{8i+2}, c_{8i+3}) mapeia para a componente em quadratura (Q) da polarização X de S_i .
- (c_{8i+4}, c_{8i+5}) mapeia para a componente em fase (I) da polarização Y de S_i .
- (c_{8i+6}, c_{8i+7}) mapeia para a componente em quadratura (Q) da polarização Y de S_i .

Em cada dimensão de sinalização, definimos o seguinte mapeamento do rótulo binário para a amplitude do símbolo:

$$(0, 0) \rightarrow -3; (0, 1) \rightarrow -1; (1, 1) \rightarrow +1; (1, 0) \rightarrow +3. \quad (12)$$

A Tabela 5 detalha o mapeando dos símbolos:

Tabela 5 – Amplitude do símbolo em fase (I) e em quadratura (Q)

$(c_{8i}, c_{8i+1}, c_{8i+2}, c_{8i+3})$ ou $(c_{8i+4}, c_{8i+5}, c_{8i+6}, c_{8i+7})$	I	Q
(0, 0, 0, 0)	−3	−3
(0, 0, 0, 1)	−3	−1
(0, 0, 1, 0)	−3	3
(0, 0, 1, 1)	−3	1
(0, 1, 0, 0)	−1	−3
(0, 1, 0, 1)	−1	−1
(0, 1, 1, 0)	−1	3
(0, 1, 1, 1)	−1	1
(1, 0, 0, 0)	3	−3
(1, 0, 0, 1)	3	−1
(1, 0, 1, 0)	3	3
(1, 0, 1, 1)	3	1
(1, 1, 0, 0)	1	−3
(1, 1, 0, 1)	1	−1
(1, 1, 1, 0)	1	3
(1, 1, 1, 1)	1	1

fonte: (OIF-400ZR-01.0, 2020)

3.3 DEMAPEADOR *SOFT* 16-QAM

Um demapeador é um dispositivo que desmapeia um símbolo em um grupo de bits, revertendo a operação de um mapeador. No caso de um demapeador *hard*, as informações de confiabilidade de cada bit são perdidas: se um símbolo recebido estiver muito próximo do limite entre duas saídas possíveis é equivalente ao caso que o símbolo estiver muito longe de um limiar. Agora, um caso é muito mais provável de ter erros do que o outro.

Um demapeador *soft* é uma extensão de um demapeador comum que fornece, além do grupo de bits, uma medida de quão confiável cada bit é, dependendo de quão longe o símbolo recebido estava de um limite. Essa informação *soft* é chamada de verossimilhança logarítmica (LLR - *Log-likelihood Ratio*). O demapeador *soft* fornece, como grande diferencial, a informação de quão confiável é a estimativa de cada bit. Nesse sentido, se o símbolo recebido está próximo do limiar de decisão, há uma grande incerteza relacionada com o bit transmitido. Essa informação extra da confiabilidade de cada bit transmitido, possui um efeito positivo no desempenho de um código FEC (TOSATO; BISAGLIA, 2004).

O demapeador computa a métrica de confiabilidade baseada em probabilidades, sendo a razão logarítmica da probabilidade do bit ser um zero sobre o bit ser um definido pela equação (ALLPRESS; LUSCHI; FELIX, 2004):

$$LLR_k = \log \left(\frac{P(b_k = 0 | s_{rx})}{P(b_k = 1 | s_{rx})} \right), \quad (13)$$

estando k no intervalo de 0 a $\log_2(M) - 1$, onde M é o número de símbolos na constelação e s_{rx} é o símbolo recebido.

Se um modelo de ruído aditivo gaussiano branco (AWGN - *Additive White Gaussian Noise*) é

assumido para o canal, junto a probabilidades independentes para um bit ser zero ou um, as LLRs podem ser calculadas seguindo a seguinte expressão (ALLPRESS; LUSCHI; FELIX, 2004):

$$LLR_k = \log \left(\frac{\sum_{s \in S_{b_k=0}} \exp\left(\frac{-1}{\sigma^2} |s_{rx} - s|^2\right)}{\sum_{s \in S_{b_k=1}} \exp\left(\frac{-1}{\sigma^2} |s_{rx} - s|^2\right)} \right), \quad (14)$$

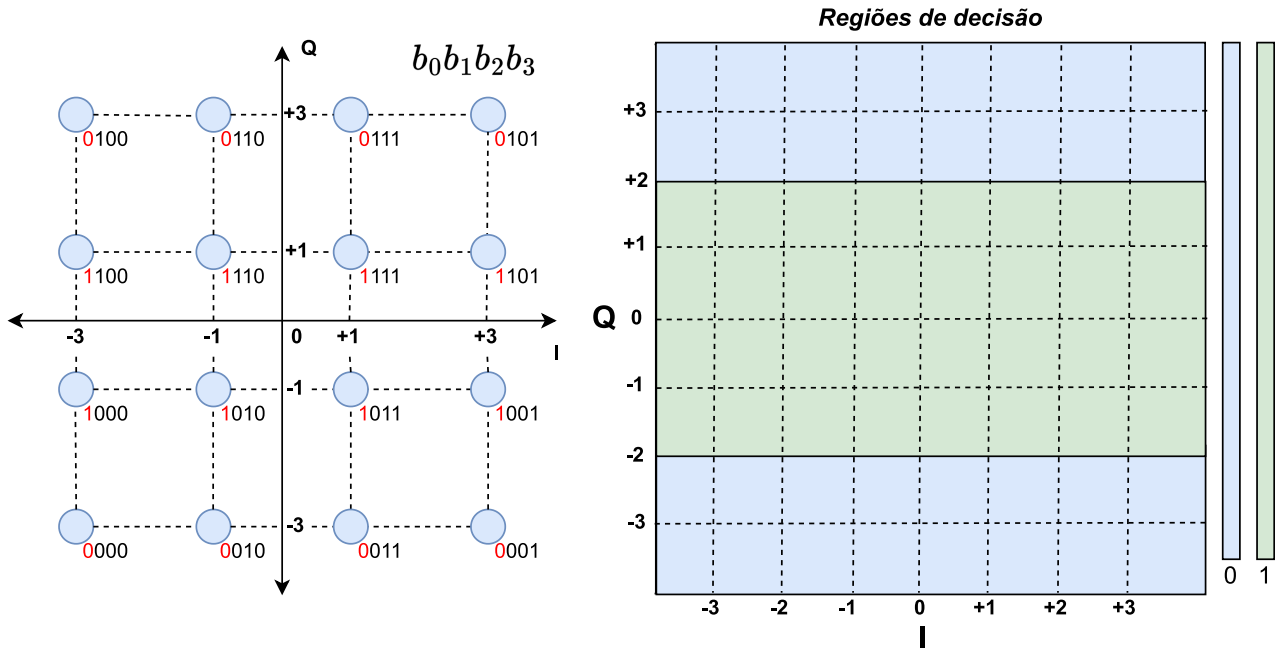
onde σ é o desvio padrão do canal, $S_{b_k=1}$ corresponde ao conjunto de símbolos para o qual o bit k é um e $S_{b_k=0}$ para o conjunto de símbolos onde o bit k é zero.

Se considerarmos que o sinal s_{rx} é recebido em um plano, dividido em regiões que definem os símbolos mais próximos, a Equação 14 pode ser simplificada termos quadráticos mais simples ou constantes. A LLR se torna uma função linear das coordenadas de s_{rx} (TOSATO; BISAGLIA, 2004).

É assim que o demapeador *soft* é implementado. Primeiro, a região na qual o sinal recebido está localizado é calculada e, então, a equação linear correta é aplicada. Os valores de LLR são alocados na forma de magnitude que representa a confiabilidade do bit e um valor de sinal (0 ou 1), que representa a sinalização do mesmo. O valor da confiabilidade, geralmente possui uma resolução (quantização) que determinará os valores possíveis.

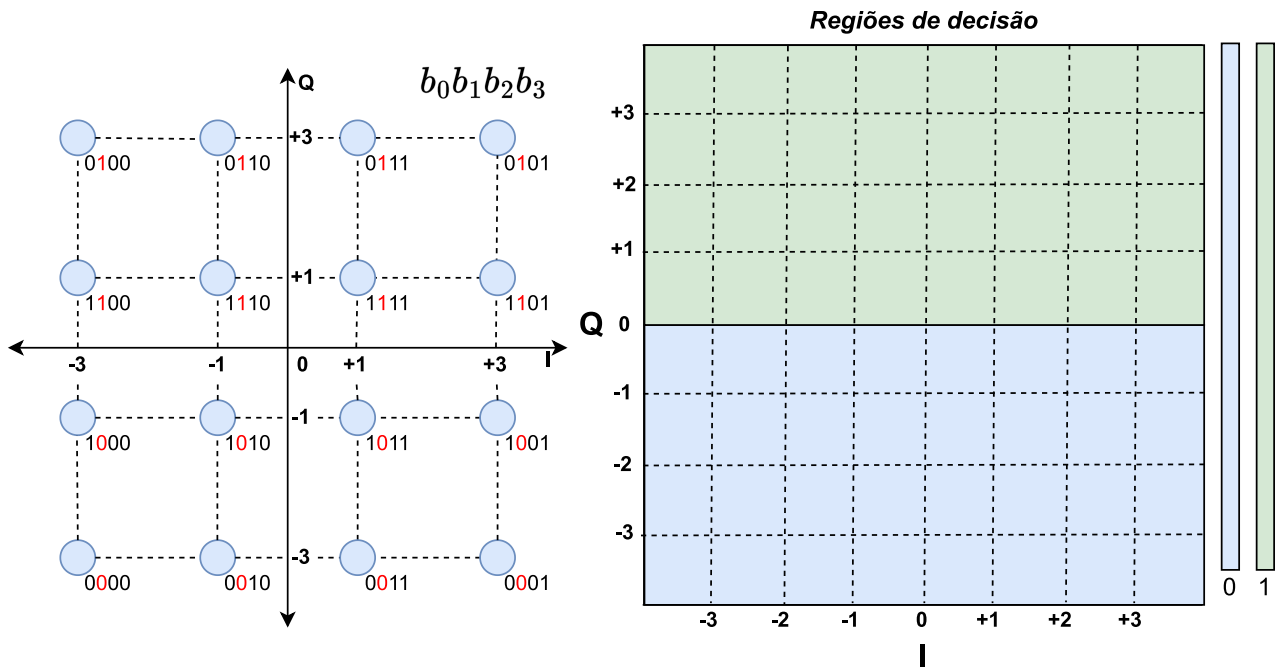
Considerando a constelação 16-QAM definida na subseção 3.2, em conjunto com a adição de um ruído aditivo gaussiano branco (AWGN), as seguintes regiões foram definidas para cada um dos dois conjuntos de quatro bits que compõem o símbolo (TOSATO; BISAGLIA, 2004).

Figura 5 – Regiões de decisão para o bit 0.



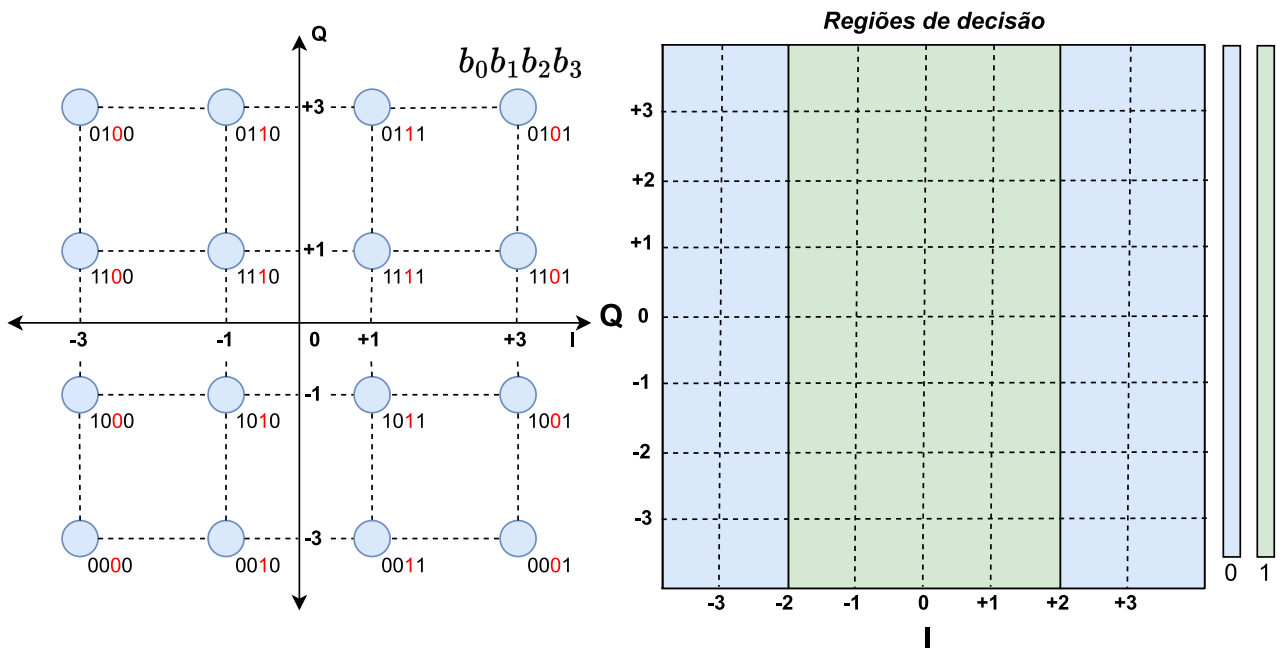
fonte: Produção do próprio autor.

Figura 6 – Regiões de decisão para o bit 1.



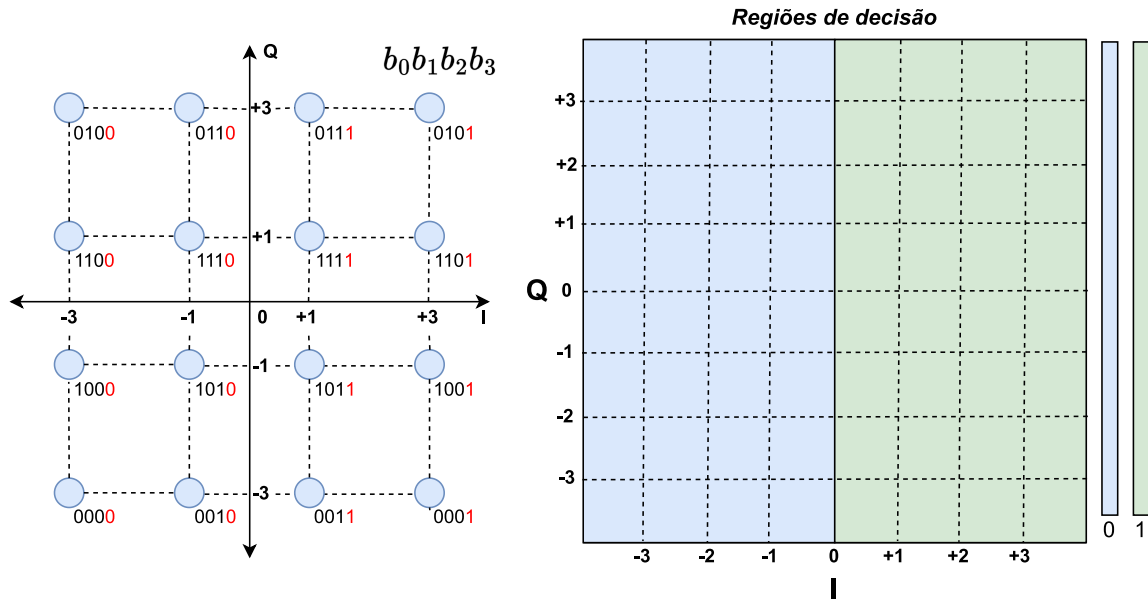
fonte: Produção do próprio autor.

Figura 7 – Regiões de decisão para o bit 2.



fonte: Produção do próprio autor.

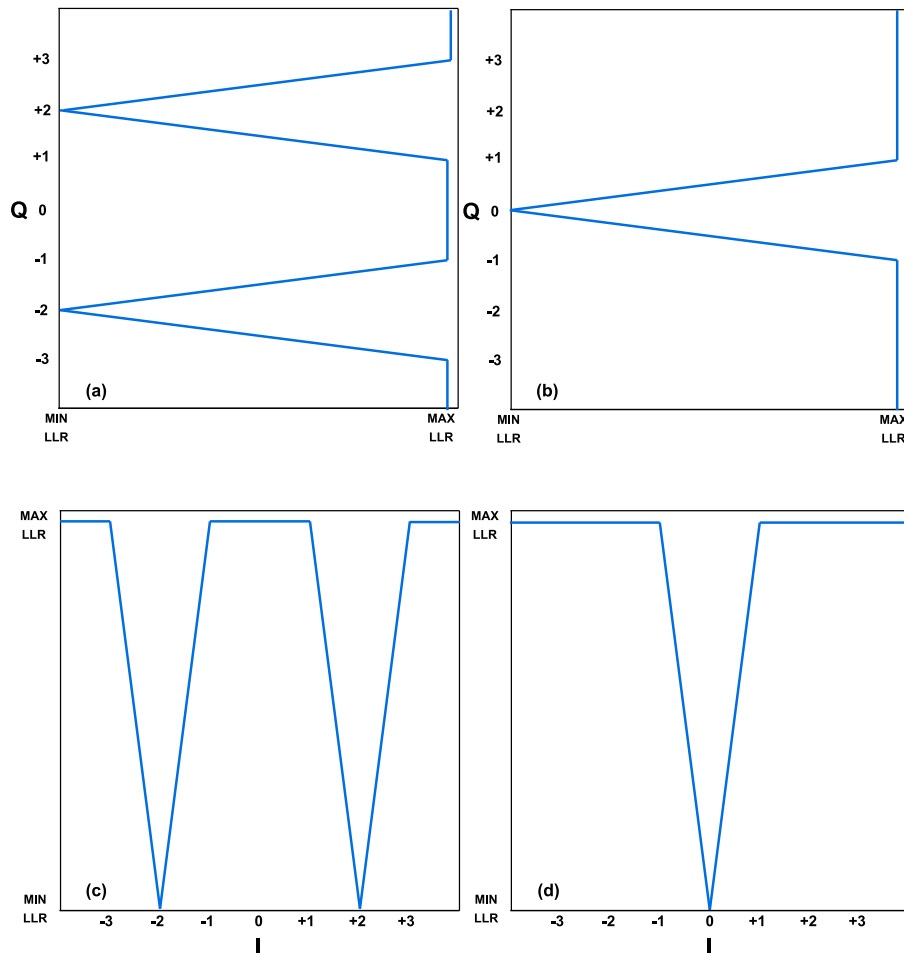
Figura 8 – Regiões de decisão para o bit 3.



fonte: Produção do próprio autor.

Com as regiões definidas, a equação linear se torna as curvas da Figura 9:

Figura 9 – Curvas dos valores de LLR para os bits da polarização: (a) bit 0, (b) bit 1, (c) bit 2 e (d) bit 3.



fonte: Produção do próprio autor.

Os bits 4, 5, 6 e 7 do símbolo mapeado na outra polarização, seguem as mesmas regiões e curvas de LLR dos bits 0, 1, 2 e 3 respectivamente.

O valor de magnitude da confiabilidade (amplitude da LLR) e o valor de sinal (sinalização da LLR) alocados para um determinado bit pelo demapeador. São definidos de acordo com a quantização N definida. Portanto, se N for um inteiro > 2 um vetor binário $\mathbf{q}$ de N -bits é definido.

$$\mathbf{q} = [q_0, q_1, \dots, q_{N-1}]. \quad (15)$$

Com o vetor binário de dimensão N definido (Equação 15). O valor da magnitude é obtido através das $N - 1$ posições menos significativas e o bit de sinalização será igual ao bit da posição mais significativa:

$$llr_{amp} = q_0 \cdot 2^0 + q_1 \cdot 2^1 + \dots + q_{N-2} \cdot 2^{N-2}, \quad (16)$$

$$LLR_{sign} = q_{N-1}. \quad (17)$$

Então se $N = 6$, teremos valores de confiabilidade, Equação 16, que vão de 0 (menos confiável) até 31 (mais confiável). O bit de sinalização, Equação 17, informa se a magnitude representa bit 0 ($LLR_{sign} = 0$) ou bit 1 ($LLR_{sign} = 1$). Assim, quanto maior for o nível de quantização N , maior será a precisão em que podemos representar a confiabilidade (LLR_{amp}) do bit recebido.

3.4 DECODIFICADOR HAMMING

O comprimento do código de Hamming estendido é relativamente baixo, enquanto o raio de correção de erros do código é igual a 1. Os primeiros sete bits das colunas da matriz verificadora de paridade $\mathbf{H}$ (Equação 4) representa todos os vetores binários possíveis de comprimento sete.

A última linha da matriz $\mathbf{H}$ pode ser utilizada para identificar a paridade do total de erros. Se a síndrome (Equação 13 da subseção 2.2.3) não for um vetor nulo, ou seja, $\mathbf{s} = (s_0, s_1, \dots, s_8) \neq \mathbf{0}$ e $s_8 = 0$ o algoritmo vai realizar a decodificação, sabendo que o número de erros é par. Caso $s \neq \mathbf{0}$ e $s_8 = 1$, o número de erros é ímpar. Se soubermos as posições de todos os 1's em um padrão de erro, exceto o último, então conheceremos um padrão de erro parcial (PEP - *Partial Error Pattern*). A posição do último erro pode ser encontrada baseada na síndrome $\mathbf{s}$ e nas colunas da matriz $\mathbf{H}$ correspondentes ao bits de padrão de erro conhecidos por meio de uma função inversa da permutação (Tabela 4). A ideia por trás é identificar o padrão de erro previamente com base na BER visada. O último membro de cada padrão de erro será executado durante o algoritmo e será dependente da síndrome $\mathbf{s}$.

O algoritmo se utiliza de dois conjuntos de PEPs: os conjuntos ímpares (ε_{impar}) e os conjuntos pares (ε_{par}), definidos na Tabela 6. O PEP é definido em termos da posição do valor absoluto menos confiável da LLR. Na notação um padrão (3, 4) por exemplo, identifica um vetor de erros e com peso dois, que possui 1's nas posições do terceiro e quarto valor menos confiável da LLR absoluta. A expressão $(i, j - k)$ denota um conjunto de padrões $(i, j), (i, j + 1), \dots, (i, k)$. No caso do padrão ser igual 0, o algoritmo vai identificar uma posição de erro j , onde $\mathbf{h}_j = \mathbf{s}$ se existir.

O algoritmo Teste de Padrão de Erro Direto (DEPT - *Direct Error-Pattern Testing*) pode levar a uma melhor implementação em *hardware*, pois diferente dos algoritmos de Chase (CHASE, 1972), permite escolher os padrões de teste que serão utilizados (TRUHACHEV et al., 2020).

Executamos o algoritmo para o caso em que $s \neq 0$:

1. O algoritmo é executado para os padrões ímpares caso $s_8 = 1$ ou para os padrões pares $s_8 = 0$, percorrendo as PEPs definidas na Tabela 6.
2. Para cada PEP identificar as posições $\mathbf{j}_1, \dots, \mathbf{j}_n$ de acordo com os valores menos confiáveis da LLR. Cada $\mathbf{j}$ será passado por uma função $\pi_+(\mathbf{j})$ que mapeia um índice de coluna $j \in \{0, 1, \dots, 127\}$ em uma coluna respectiva $\mathbf{h}_j$ da matriz verificadora de paridade $\mathbf{H}$ definida na subseção 3.1.

$$\pi_+(\mathbf{j}) = [\bar{\pi}, g_7(j), 1] = \mathbf{h}_j. \quad (18)$$

Onde $g_7(j)$ é obtido através da Equação 3 e $\bar{\pi}$ é o índice de coluna.

3. O próximo passo do algoritmo é realizar uma soma binária de todos os $\mathbf{h}_j$ de um padrão de erro parcial com o seu vetor de síndrome $\mathbf{s}$ respectivo. O resultado será um vetor de 9-bits $\mathbf{s}' = (s'_0, s'_1, \dots, s'_8)$.

$$\mathbf{s}'_{[0,8]} = (h_{j1} + h_{j2} + \dots + h_{jn} + \mathbf{s}). \quad (19)$$

4. Definimos uma função denominada Π_+^{-1} , que receberá os 9-bits de $\mathbf{s}'$. Tal função será responsável por retornar o valor de j_* através de uma comparação do valor de $j' = \mathbf{s}'_{[0,6]}$, onde $\mathbf{s}'_{[0,6]}$ é um número decimal definido pelos bits menos significativos de $\mathbf{s}'$.

$$j_* = \Pi_+^{-1}(s'), \quad (20)$$

$$\Pi_+^{-1}(s') = \begin{cases} j', & \text{se } s'_7 = g_7(j') \text{ e } s'_8 = 1; \\ -1, & \text{caso contrário.} \end{cases} \quad (21)$$

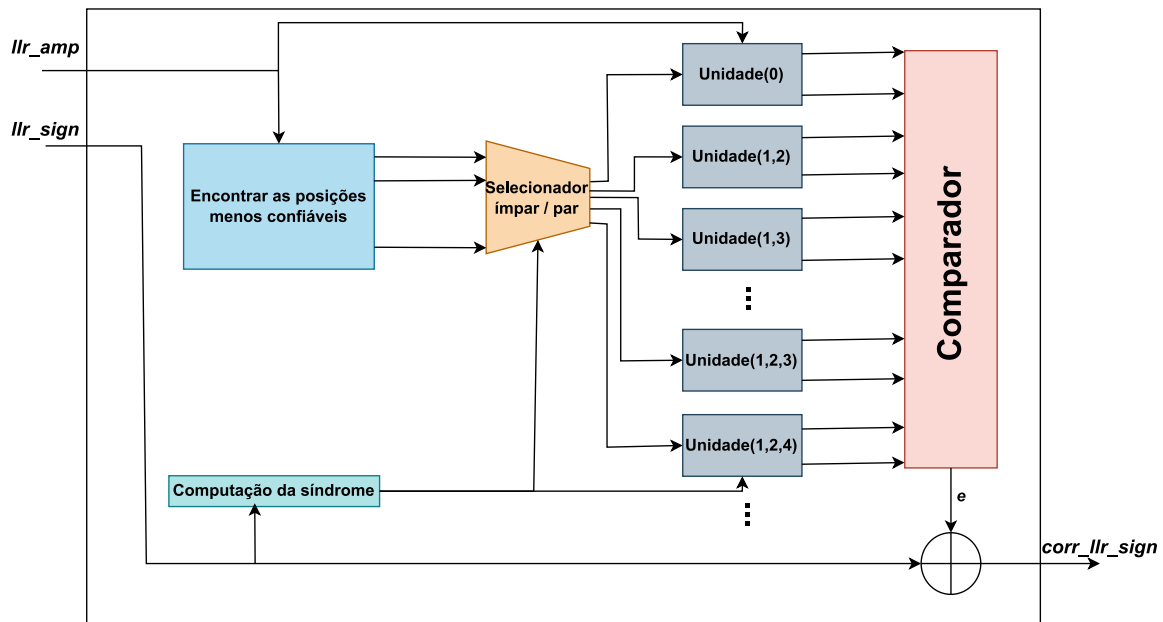
Então, $\Pi_+^{-1}(s') = -1$ indica que não há índice de coluna j tal que $\mathbf{h}_j = \mathbf{s}'$. No caso em que $j_* \neq -1$, deve-se calcular a soma dos valores absolutos das respectivas LLRs (peso analógico do padrão de erro).

$$SUM = |LLR_{j_1}| + |LLR_{j_2}| + |LLR_{j_3}| + \dots + |LLR_{j_n}| + |LLR_{j_*}|. \quad (22)$$

5. Identificar o padrão de erro e com a menor soma absoluta entre todos os padrões gerados nos passos anteriores. Caso essa soma seja menor que um determinado valor de *threshold*, deve-se realizar a correção de erros com o padrão de erro respectivo $\mathbf{e}$. Esse valor de *threshold* deve ser

escolhido com cuidado, pois é um valor que permite correção de mais erros e ao mesmo tempo inibe a adição de erros.

Figura 10 – Diagrama de blocos da implementação do decodificador com o algoritmo DEPT.



fonte: (TRUHACHEV et al., 2020)

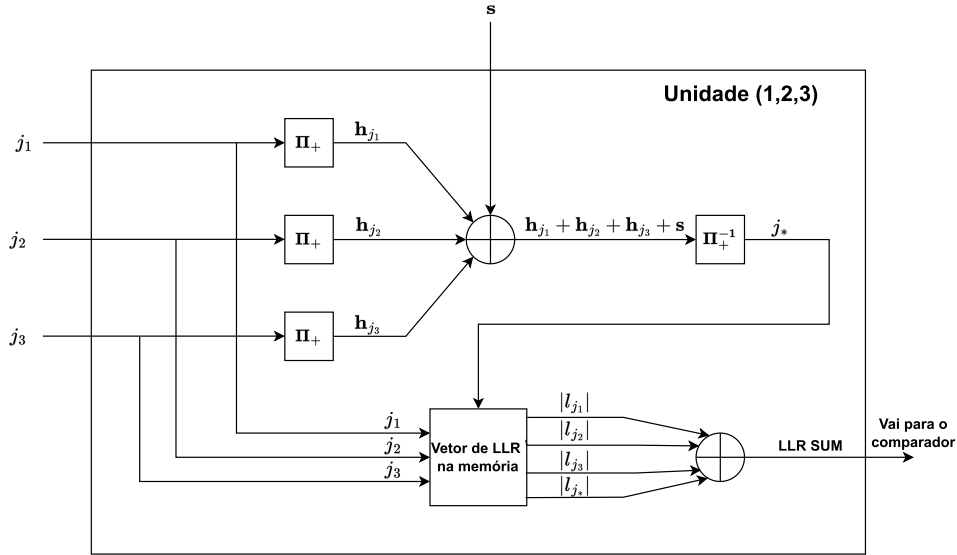
Tabela 6 – Padrões de erros parciais (PEPs).

	Padrões de Erros Parciais Ímpares	Padrões de Erros Parciais Pares
1	0; (1, 2 - 5); (2, 3 - 5); (3, 4)	(1 - 6); (1, 2, 3 - 4)
2	0; (1, 2 - 5); (2, 3 - 5); (3, 4)	(1 - 6); (1, 2, 3 - 4) (1, 3, 4); (2, 3, 4)
3	0; (1, 2 - 6); (2, 3 - 5); (3, 4 - 5)	(1 - 6); (1, 2, 3 - 4) (1, 3, 4); (2, 3, 4)
4	0; (1, 2 - 6); (2, 3 - 6); (3, 4 - 5); (1, 2, 3, 4); (4, 5)	(1 - 6); (1, 2, 3 - 5) (1, 3, 4); (2, 3, 4)
5	0; (1, 2 - 7); (2, 3 - 6); (3, 4 - 6); (1, 2, 3, 4); (4, 5)	(1 - 7); (1, 2, 3 - 5) (1, 3, 4 - 5); (2, 3, 4 - 5)
6	0; (1, 2 - 8); (2, 3 - 7); (3, 4 - 7); (4, 5 - 6); (5, 6); (1, 2, 3, 4 - 5)	(1 - 8); (1, 2, 3 - 8) (1, 3, 4 - 6); (2, 3, 4 - 5); (1, 4, 5)
7	0; (1, 2 - 9); (2, 3 - 9); (3, 4 - 8); (4, 5 - 8); (5, 6 - 7); (1, 2, 3, 4 - 7); (1, 2, 3, 5 - 6)	(1 - 9); (1, 2, 3 - 8) (1, 3, 4 - 7); (2, 3, 4 - 7); (1, 4, 5 - 7); (2, 4, 5 - 7); (3, 4, 5 - 7)

fonte: (TRUHACHEV et al., 2020)

Portanto, para a realizar a correção de erros utilizando o algoritmo DEPT, a estrutura demonstrada na Figura 10 é utilizada. Os blocos em azul e verde claro são responsáveis por preparar os dados necessários onde os valores de LLRs são divididos em LLRs absolutas e bits de sinalização. Já os blocos em laranja, cinza e vermelho são referentes ao algoritmo DEPT. A saída do comparador, somada (*XOR*) aos bits de sinalização, correspondem à palavra-código estimada.

Figura 11 – Unidade (1,2,3) computa a soma de quatro LLRs absolutas que correspondem ao padrão de erro definido pela PEP (1,2,3).



fonte: (TRUHACHEV et al., 2020)

A Figura 11 ilustra o fluxo de dados de um padrão de erro qualquer do algoritmo DEPT. A saída do comparador depende de um valor de *threshold* que visa limitar a inserção de erros na estimação da palavra-código. Esse valor deve ser otimizado de acordo com a resolução (quantização) utilizada no demapeador.

3.5 EXEMPLO DE CODIFICAÇÃO E DECODIFICAÇÃO

Um exemplo de codificação e decodificação do código de Hamming (128,119) duplamente estendido, utilizando o algoritmo DEPT 2 com quantização de 6-bits e *threshold*=26 é apresentado. O exemplo em questão adiciona três erros nas posições 0, 5 e 10 (em verde) do vetor de mensagem.

1. Seja a mensagem $\mathbf{m} = [m_0 m_1 \dots m_{118}]$ aleatória de 119-bits:

$\mathbf{m} = [1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0].$

2. Quando codificamos a mensagem $\mathbf{m}$ (Equação 9) é adicionado 9-bits de paridade. A mensagem $\mathbf{m}$ codificada se torna uma palavra-código $\mathbf{c} = [c_0 c_1 \dots c_{127}]$ de 128-bits:

$c = [1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0].$

Onde os bits em azul representam os bits de paridade.

- Essa palavra-código c é então mapeada e transmitida pelo canal com ruído AWGN. Durante esse processo ruído é adicionado na palavra-código. Quando ela for demapeada, o vetor de LLR_{sign} (o qual pode ser visto como uma decodificação *hard*) apresentará diferenças (em vermelho) se comparado com a palavra-código c do item anterior:

$LLR_{sign} = [0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0].$

O demapeador também fornece os valores de confiabilidade (LLR_{amp}) de acordo com a quantização utilizada (nesse exemplo $N = 6$):

$LLR_{amp} = [4\ 17\ 30\ 24\ 18\ 5\ 15\ 26\ 6\ 24\ 4\ 9\ 22\ 11\ 13\ 22\ 17\ 12\ 16\ 12\ 14\ 24\ 10\ 17\ 22\ 16\ 14\ 31\ 23\ 22\ 15\ 15\ 19\ 13\ 14\ 23\ 22\ 31\ 11\ 15\ 6\ 26\ 12\ 9\ 6\ 29\ 28\ 28\ 13\ 28\ 18\ 17\ 22\ 27\ 18\ 17\ 14\ 25\ 28\ 26\ 17\ 10\ 30\ 17\ 7\ 31\ 26\ 15\ 14\ 26\ 20\ 20\ 13\ 23\ 27\ 14\ 18\ 20\ 23\ 10\ 19\ 22\ 30\ 27\ 8\ 24\ 13\ 22\ 13\ 9\ 31\ 28\ 22\ 11\ 19\ 27\ 12\ 20\ 7\ 26\ 16\ 22\ 29\ 7\ 14\ 25\ 25\ 21\ 18\ 20\ 11\ 26\ 16\ 26\ 22\ 16\ 29\ 23\ 17\ 6\ 8\ 9\ 11\ 30\ 19\ 13\ 24\ 15].$

Os valores em amarelo representam as posições 0, 10, 5, 8, 40 e 44. Essas são as seis posições menos confiáveis (menores valores) em ordem crescente. Nota-se que as posições dos erros também são posições não confiáveis.

- Os valores de LLR_{sign} e LLR_{amp} são processados pelo decodificador. Com LLR_{sign} se calcula o vetor de síndrome $s = [s_0\ s_1\ \dots\ s_8]$, Equação 13:

$$s = [1\ 1\ 1\ 1\ 0\ 0\ 0\ 1]. \quad (23)$$

Se o vetor de síndrome fosse totalmente nulo ($s = 0$) o algoritmo termina, sua saída é o próprio vetor LLR_{sign} sem os bits de paridade.

Como o vetor de síndrome é diferente de zero ($s \neq 0$), o algoritmo sabe que existe erro na informação recebida e devemos definir quais PEPs utilizaremos de acordo com s_8 . Se $s_8 = 1$ temos um número ímpar de erros e usamos o conjunto de PEPs ímpares. Caso $s_8 = 0$ temos um número par de erros e utilizamos o conjunto par de PEPs. Portanto, como $s_8 = 1$ o conjunto de PEPs ímpares é utilizado.

- Considerando os valores menos confiáveis e as posições dos PEPs ímpares do DEPT 2, apresentado na Tabela 6. Teremos o seguinte conjunto de PEPS (unidades da Figura 10):

$$PEPs = ([\]; [0, 10]; [0, 5]; [0, 8]; [0, 40]; [10, 5]; [10, 8]; [10, 40]; [5, 8]). \quad (24)$$

6. As operações definidas na Figura 11 serão executadas. Cada posição j presente em um PEP, retornará valores de um índice de coluna h_j , através da Equação 18. Portanto, para cada unidade de PEPs teremos:

- Unidade (): Não possui índices de colunas, pois é um conjunto vazio de PEPs. Logo, o vetor binário de 9-bits $s' = [s'_0 \ s'_1 \ \cdots \ s'_8]$ (Equação 19) será igual o vetor de síndrome obtido no item 4:

$$s' = s = [1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1]. \quad (25)$$

Esse vetor s' é então passado pela função Π_+^{-1} (Equação 21) que fornece a posição do último erro j_* . Sabemos que j' é representado pelos sete primeiros bits de s' :

$$j' = s'_{[0,6]} = [1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0] = 15, \quad (26)$$

$$h_{j'} = [1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1] \neq s'. \quad (27)$$

Caso as condições da Equação 21 forem totalmente satisfeitas, j_* será igual a j' . Se as condições não forem satisfeitas, não há índice de coluna para j tal que $h_j = s'$, então o PEP em questão não é válido e pode ser descartado. Neste caso em específico, as condições não são válidas, pois $s'_7 \neq g_7(j')$. Portanto, esse padrão não pode ser utilizado.

- Unidade (0, 10): Essa unidade possui dois índices de coluna, $j_0 = 0$ e $j_1 = 10$. Portanto, esses índices retornam através da Equação 18, os vetores de 9-bits, h_{j_0} e h_{j_1} que são somados (*XOR*) junto ao vetor de síndrome s , para assim se obter s' :

$$s' = h_{j_0} + h_{j_1} + s, \quad (28)$$

$$s' = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1] + [0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1] + [1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0] = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1]. \quad (29)$$

O vetor s' é passado pela função Π_+^{-1} (Equação 21) fornecendo a posição do último erro j_* . Onde j' é composto pelos sete primeiros bits de s' :

$$j' = s'_{[0,6]} = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0], \quad (30)$$

$$h_{j'} = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1] = s'. \quad (31)$$

Se as condições da Equação 21 são válidas, j_* será igual a j' e caso não forem satisfeitas, não existirá índice de coluna para j tal que $h_j = s'$, assim o padrão não pode ser utilizado. Para esse caso, as condições foram satisfeitas, pois $s'_7 = g_7(j')$ e $s'_8 = 1$.

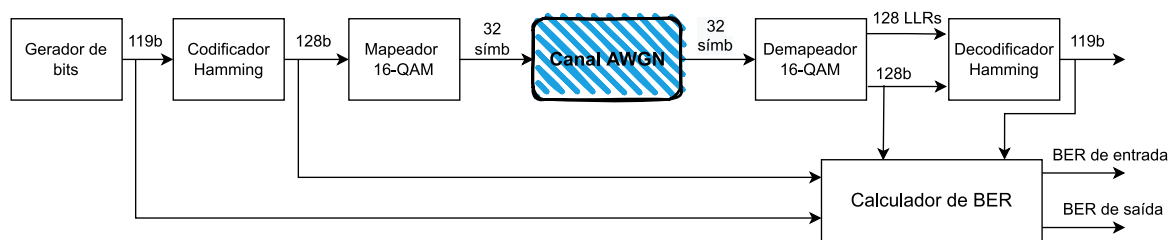
$$j_* = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0] = 5, \quad (32)$$

4 RESULTADOS

Para se obter resultados para a validação do algoritmo DEPT, foi necessário a integração das seções do Capítulo 3 e a adição de um gerador de mensagens, o qual fornece 119-bits aleatórios. Esses bits serão processados pelo codificador do código de Hamming (128,119). Assim, fornecendo uma palavra-código composta por 128-bits. Em seguida, os 128-bits resultantes da codificação, são transferidos para o mapeador 16-QAM descrito na subseção 3.2, o qual segue a codificação de Gray.

O mapeador é responsável por mapear a palavra-código em 16 símbolos de 8-bits. Esses símbolos são transmitidos pelo canal óptico, o qual é modelado como um canal com ruído aditivo gaussiano branco (AWGN). Os sinal distorcido sera processado pelo demapeador *soft* 16-QAM, discutido na subseção 3.3, responsável por desmapear e fornecer os 128 valores de confiabilidade quantizados em N-bits e os 128-bits de sinalização respectivos. No final, os valores de confiabilidade (LLR) e os bits de sinalização são finalmente transferidos para o decodificador, apresentado na subseção 3.4, onde a correção do erro é realizada. Sua saída corresponde a 119-bits de mensagem, obtidos através da decodificação da palavra-código. Caso a correção dos erros seja totalmente realizada, os 119-bits da saída do decodificador serão correspondentes aos 119-bits de mensagem gerados no início do teste. Ao final do processo, um contador de erros é adicionado. O mesmo é responsável por calcular a BER de entrada e de saída do FEC. Esse cálculo é realizado utilizando a Equação 1, onde o número de erros para a BER de entrada é obtido pela diferença entre os 119-bits que entram no codificador e os 119-bits correspondentes à mensagem dos 128-bits que entram no decodificador. Para a BER de saída, o número de erros é obtido pela diferença entre os 119-bits que entram no codificador e os 119-bits que saem do decodificador. A estrutura do diagrama de blocos da Figura 12 foi utilizada para obtenção de todos os resultados.

Figura 12 – Diagrama do fluxo de dados para obtenção dos resultados.



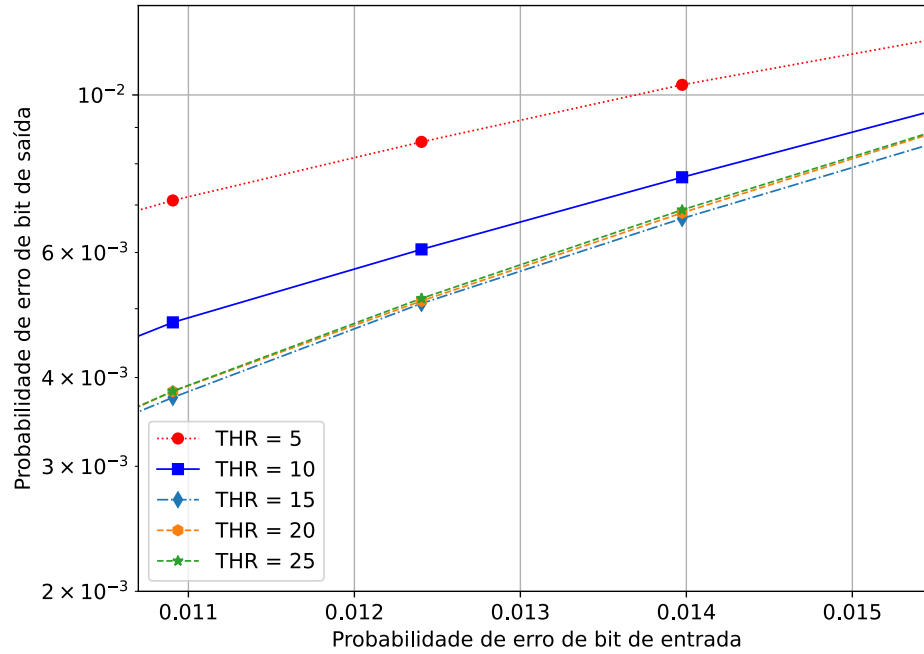
fonte: Produção do próprio autor.

Com a estrutura estabelecida, tem-se a prioridade de encontrar o valor ótimo de *threshold* (*thr*). O valor ótimo de *threshold* foi obtido fazendo uma varredura deste para diferentes valores de SNR e considerando diferentes resoluções (quantização) do discretizador.

Na Figura 13 e Figura 14 mostramos a BER de saída em termos da BER de entrada, considerando uma quantização com resolução de 5 e outra com 6 bits. Alguns valores de *threshold* são testados para cada tipo de resolução. Analisando as figuras, nota-se que o valor ótimo está entre 15 e 20 para a quantização de 5-bits e entre 25 e 30 para a quantização de 6-bits. A busca pelo valor ótimo tem grande importância no resultado, pois é um valor que permite correção de mais erros e ao mesmo tempo inibe

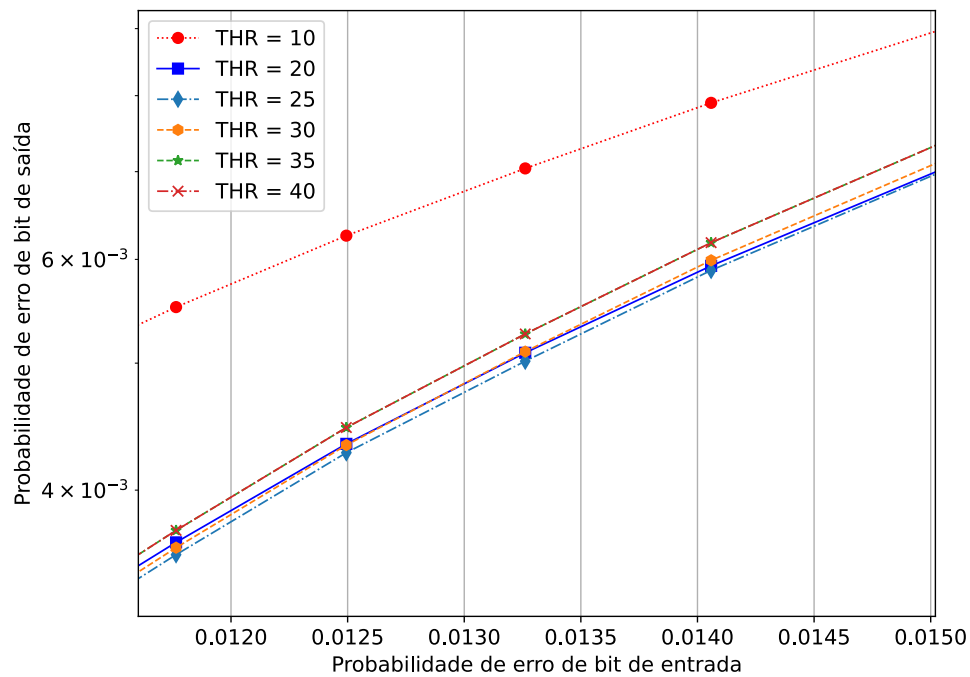
a adição de erros. O melhor valor de *threshold* encontrado foi de $thr = 16$ para quantização de 5-bits e $thr = 26$ para a quantização de 6-bits.

Figura 13 – Efeito do *threshold* no algoritmo DEPT com quantificação de 5-bits.



fonte: Produção do próprio autor.

Figura 14 – Efeito do *threshold* no algoritmo DEPT com quantificação de 6-bits.



fonte: Produção do próprio autor.

Com o valor ótimo de *threshold*, foi-se realizado o levantamento da curva de BER de entrada pela BER de saída, referentes a cada um dos padrões de erros parciais (PEPs) apresentados na Tabela 6. Como o algoritmo permite que os valores de confiabilidade (LLR) recebidos sejam quantizados em diferentes níveis, as Tabelas 7 e 8, mostram os valores de BER de entrada e saída, obtidos para quantização de 5-bits e 6-bits respectivamente. Com os resultados obtidos, nota-se a diferença entre os padrões de erros parciais (PEPs) utilizados. Aqueles, com o maior número de PEPs, apresentam o melhor desempenho se considerado como parâmetro a BER de saída. Isso se deve ao fato de o número de comparações realizadas na decodificação serem maiores.

Tabela 7 – Dados obtidos para quantização de 5-bits

BER de entrada	BER de saída com thr = 16						
	DEPT 1	DEPT 2	DEPT 3	DEPT 4	DEPT 5	DEPT 6	DEPT 7
0,00975	0,00275	0,00269	0,00266	0,00260	0,00257	0,00254	0,00251
0,01107	0,00387	0,00377	0,00374	0,00366	0,00361	0,00356	0,00352
0,01176	0,00455	0,00445	0,00442	0,00431	0,00425	0,00419	0,00414
0,01212	0,00493	0,00482	0,00478	0,00468	0,00462	0,00456	0,00450
0,01249	0,00533	0,00522	0,00518	0,00507	0,00501	0,00495	0,00489
0,01287	0,00577	0,00565	0,00561	0,00550	0,00543	0,00536	0,00530
0,01326	0,00619	0,00607	0,00601	0,00590	0,00584	0,00577	0,00569
0,01406	0,00714	0,00701	0,00696	0,00683	0,00676	0,00668	0,00660
0,01576	0,00931	0,00918	0,00910	0,00895	0,00886	0,00875	0,00864
0,01756	0,01178	0,01162	0,01154	0,01138	0,01127	0,01114	0,01102

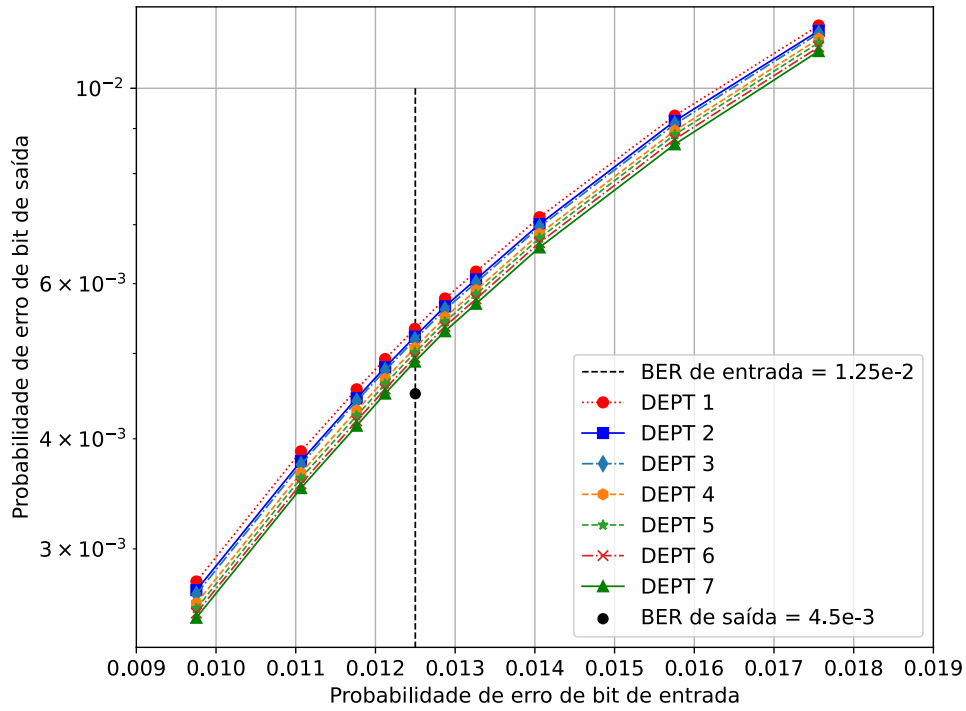
fonte: Produção do próprio autor.

Tabela 8 – Dados obtidos para quantização de 6-bits

BER de entrada	BER de saída com thr = 26						
	DEPT 1	DEPT 2	DEPT 3	DEPT 4	DEPT 5	DEPT 6	DEPT 7
0,00975	0,00209	0,00202	0,00198	0,00190	0,00185	0,00179	0,00175
0,01107	0,00309	0,00298	0,00292	0,00282	0,00274	0,00265	0,00257
0,01176	0,00369	0,00358	0,00353	0,00339	0,00330	0,00320	0,00311
0,01212	0,00403	0,00392	0,00386	0,00372	0,00362	0,00352	0,00343
0,01249	0,00440	0,00428	0,00422	0,00408	0,00399	0,00386	0,00377
0,01287	0,00478	0,00465	0,00459	0,00444	0,00434	0,00422	0,00411
0,01326	0,00515	0,00503	0,00495	0,00480	0,00470	0,00458	0,00446
0,01406	0,00603	0,00590	0,00582	0,00566	0,00555	0,00543	0,00530
0,01576	0,00808	0,00796	0,00786	0,00768	0,00756	0,00741	0,00726
0,01756	0,01052	0,01036	0,01027	0,01010	0,00996	0,00980	0,00962

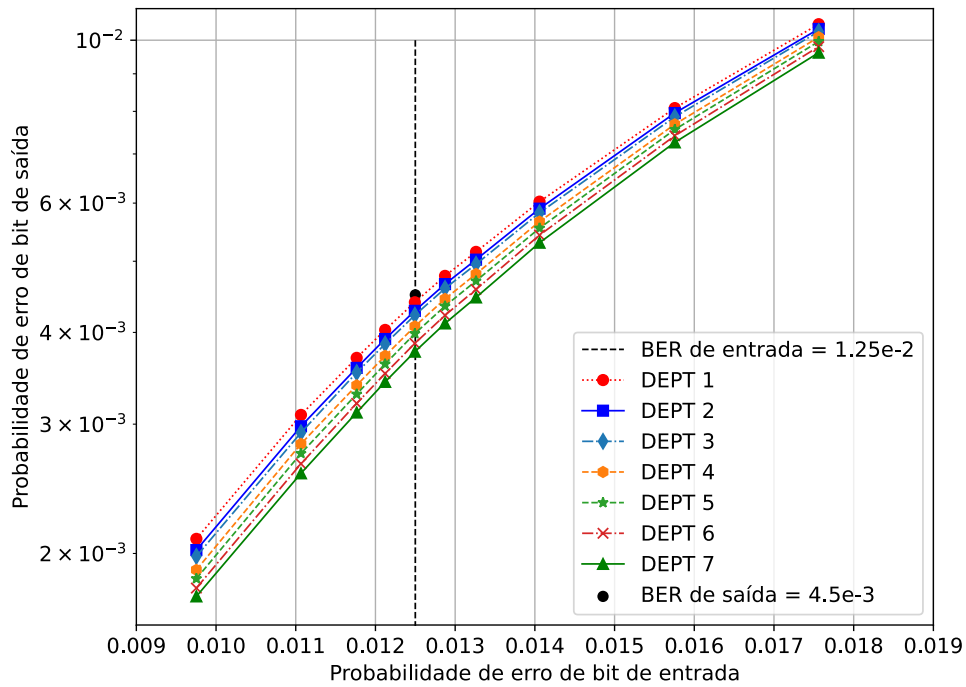
fonte: Produção do próprio autor.

Figura 15 – Curvas obtidas para o algoritmo DEPT com quantização de 5-bits.



fonte: Produção do próprio autor.

Figura 16 – Curvas obtidas para o algoritmo DEPT com quantização de 6-bits.



fonte: Produção do próprio autor.

Considerando a proposta da OIF para o sistema 400ZR (OIF-400ZR-01.0, 2020), onde o CFEC possui transmissão praticamente livre de erros se, e somente se, a BER de entrada for igual ou menor a $1,25 \times 10^{-2}$. O código externo é o *staircase* FEC que consegue atingir BER de saída em torno de 10^{-15} (praticamente livre de erros). Essa BER, em específico, só é alcançada caso a BER de entrada do código seja igual ou menor a $4,5 \times 10^{-3}$. Portanto, a BER de saída do FEC interno (Hamming) deve ser igual a BER de entrada do FEC externo, ou seja, a BER de saída do código interno deve ser igual ou menor a $4,5 \times 10^{-3}$.

Analisando os resultados obtidos para a quantização de 5-bits, todas as versões do algoritmo DEPT não atenderam aos requisitos impostos pela OIF. Olhando para a quantização de 6-bits, todas as versões do algoritmo atendem os requisitos propostos pela OIF. Isso é notado nas Figura 15 e Figura 16 que representam as curvas para as diferentes versões do algoritmo DEPT para cada quantização. No primeiro caso (quantização de 5-bits), todas as curvas ficaram acima do ponto de referência (ponto em preto). Portanto, nem uma versão do algoritmo DEPT atende a especificação mínima. Para o segundo caso (quantização de 6-bits), todas as curvas ficaram abaixo da referência, ou seja, todas as versões do algoritmo são válidas para implementação do FEC interno. Essa diferença entre quantização, se deve ao nível de precisão que o demapeador consegue atingir, ou seja, quanto maior for a quantização N melhor será o resultado obtido.

Outro parâmetro que deve ser analisado é o tempo de simulação. A Tabela 9 apresenta as informações sobre o tempo médio para se decodificar uma palavra-código. Nota-se, de forma clara, que quanto maior é o número de bits de quantificação e o padrão de erro parcial utilizado maior será o tempo de simulação em termos de *software*.

Tabela 9 – Tempo de simulação para os diferentes padrões e formas de quantização.

	Quantização de 5-bits		Quantização de 6-bits		Diferença entre as quantizações (%)
	Tempo (s)	Diferença (%)	Tempo (s)	Diferença (%)	
DEPT 1	0,002170826	—	0,002171152	—	0,015017
DEPT 2	0,002171826	0,04604	0,002193501	1,101886	0,988135
DEPT 3	0,002281767	4,86206	0,002291649	5,25809	0,431230
DEPT 4	0,002409177	5,28853	0,002411957	9,98381	0,115273
DEPT 5	0,002482048	12,53891	0,002510248	13,50847	1,123418
DEPT 6	0,002767536	21,56105	0,002789765	22,174377	0,796809
DEPT 7	0,003320654	34,62656	0,003383158	25,82469	1,847493

fonte: Produção do próprio autor.

Pensando em uma futura implementação no nível de *hardware*, devemos considerar que uma quantização com mais bits e um padrão de erro parcial (PEP) grande levará em aumento de processamento, consumo de energia e área utilizada. Portanto, com base nas informações obtidas e, considerando as especificações dos códigos FECs do sistema 400ZR, para uma BER de entrada menor ou igual a $1,25 \times 10^{-2}$, a melhor combinação do algoritmo DEPT para aplicação nos sistemas 400ZR é o DEPT 2 com quantização de 6-bits. Essa é uma combinação com poucos padrões de erro parciais (PEPs) e consegue atingir a BER de saída especificada pela OIF (igual ou menor a $4,5 \times 10^{-3}$) com uma certa margem para pequenas variações da BER de entrada, diferente do DEPT 1, que está muito em cima da especificação não possuindo margem para erro. Essa escolha atende as especificações. A mesma

poderia levar uma implementação em *hardware* com bom desempenho e margem de segurança para *burst* de erros.

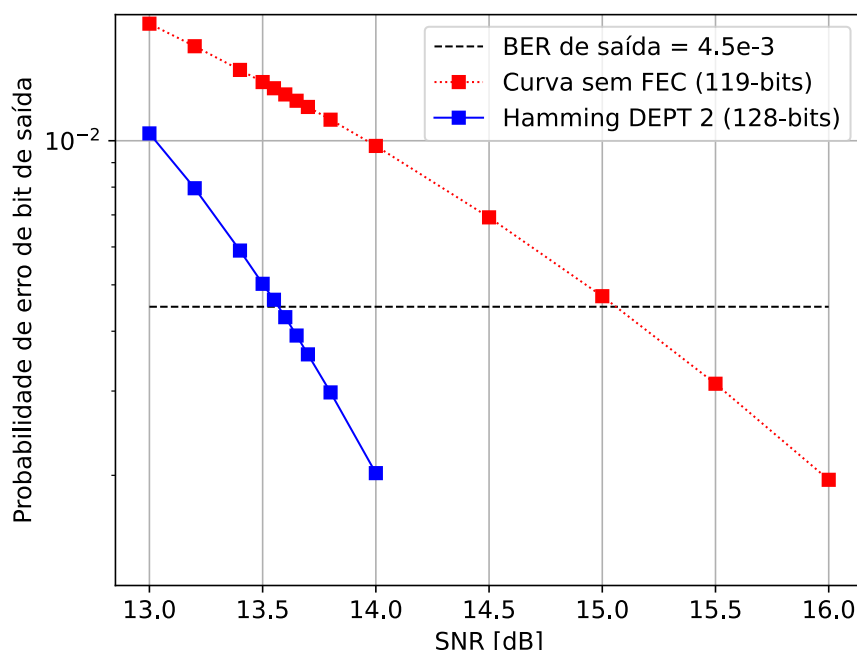
Para uma melhor visualização a Tabela 10 foi levantada para a BER de entrada do FEC interno da norma 400ZR da OIF. Nela podemos ver os valores de BER de saída do decodificador de Hamming referentes aos padrões de erros parciais e o tipo de quantização utilizada. Lembrando que o código de Hamming duplamente estendido (128, 119), deve atingir a BER de saída no valor de 4.5×10^{-3} para a BER de entrada estipulada, assim os requisitos descritos na norma do 400ZR serão atendidos.

Tabela 10 – Probabilidade de erro de bit de saída para a especificação do 400ZR.

BER de Entrada = 0,00125	BER de Saída	
	Quantização de 5-bits	Quantização de 6-bits
DEPT 1	0,00533	0,00440
DEPT 2	0,00522	0,00428
DEPT 3	0,00518	0,00422
DEPT 4	0,00507	0,00408
DEPT 5	0,00501	0,00399
DEPT 6	0,00495	0,00386
DEPT 7	0,00489	0,00377

fonte: Produção do próprio autor.

Figura 17 – Sistema sem FEC e sistema com FEC.



fonte: Produção do próprio autor.

Um último parâmetro para se avaliar seria a diferença do sistema com a presença do código corretor de erros Hamming duplamente estendido (128,119) para o sistema sem a presença de um FEC. Na Figura 17 mostramos duas curvas: uma sendo para o sistema sem a presença de um FEC e outra para o sistema com o código de Hamming, utilizando o algoritmo DEPT 2 e uma quantização de 6-bits. Uma

diferença de aproximadamente 1.5 dB é notada, subtraindo um valor referente a perda de codificação dessa diferença, Equação 3 da subseção 2.1.3, podemos obter o $NCG = 1.18$ dB, 0.1 dB acima da especificação da norma da OIF.

$$NCG = 1.5 \text{ dB} - 10 \cdot \log_{10}(n/k) = 1.5 \text{ dB} - 10 \cdot \log_{10}(128/119) = 1.18 \text{ dB} \quad (1)$$

5 CONCLUSÃO

Nessa monografia, o processo de codificação e decodificação do código de Hamming duplamente estendido é apresentando baseado no acordo de implementação OIF-400ZR, para sistemas de comunicação óptica de 400 Gbps. Um algoritmo de decodificação alternativo denominado DEPT com entrada *soft* e saída *hard* é apresentado.

O codificador e o decodificador foram implementados em alto nível (Python) com o intuito de avaliar o algoritmo proposto e atingir os valores da norma especificados pelo acordo OIF-400ZR. Para atingir os valores definidos pela norma, foi necessário analisar valores para a resolução da LLR, junto a uma otimização do parâmetro de limite do algoritmo de decodificação. Essas análises foram apresentadas e discutidas. Também se obteve o tempo de simulação para as diferentes combinações de padrões de teste e resolução da LLR, mostrando o custo em processamento para se ter um desempenho melhor na correção de erros. Os gráficos e tabelas apresentados demonstram o comportamento do algoritmo com suas diferentes combinações.

Por fim, o cálculo do NCG, para uma BER de entrada de 1.25×10^{-2} é realizado e apresentado junto à curva comparativa de um sistema com a presença do código FEC Hamming (128, 119) duplamente estendido, decodificado com o DEPT 2, e a curva de um sistema sem FEC. Onde o valor obtido foi de 1.18 dB, o qual se encontra 0.1 dB acima do valor especificado pelo acordo de implementação.

REFERÊNCIAS

- AGRAWAL, G. P. **Fiber-Optical Communication Systems**. Hoboken, New Jersey - Estados Unidos da America: John Wiley and Sons, 2010.
- ALLPRESS, S.; LUSCHI, C.; FELIX, S. V. Exact and approximated expressions of the log-likelihood ratio for 16-qam signals. **Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers**, v. 1, p. 794 – 798, 2004.
- CAI, Y. et al. Fpga investigation on error-flare performance of a concatenated staircase and hamming fec code for 400g inter-data center interconnect. **Journal of Lightwave Technology**, v. 37, n. 1, p. 188 – 195, 2019.
- CHASE, D. Class of algorithms for decoding block codes with channel measurement information. **IEEE Transactions on Information Theory**, v. 18, n. 1, p. 170 – 182, 1972.
- DOKMANIC, I. et al. Euclidean distance matrices: Essential theory, algorithms and applications. **IEEE Signal Processing**, v. 32, n. 6, p. 12 – 30, 2015.
- HAMMING, R. W. Error detecting and error correcting codes. **The Bell System Technical Journal**, v. 29, n. 2, p. 147 – 160, 1950.
- IBRAHEEM, N. H. I. **Error-Detecting and Error-Correcting Using Hamming and Cyclic Codes**. Nablus, West Bank - Palestina: [s.n.], 2009.
- LI, S. **Low-power staircase encoder implementation for high-throughput fiber-optical communications**. Halifax, Nova Scotia - Canadá: [s.n.], 2019.
- MOON, T. K. **Error Correction Coding: Mathematical Methods and Algorithms**. Hoboken, New Jersey - Estados Unidos da America: John Wiley and Sons, 2005.
- MOREIRA, J. C.; FARRELL, P. G. **Essentials of error-control coding**. Chichester, West Sussex - Inglaterra: John Wiley Sons, 2006.
- OIF-400ZR-01.0. **Implementation Agreement 400ZR**. 2020. Disponível em: <https://www.oiforum.com/wp-content/uploads/OIF-400ZR-01.0_reduced2.pdf>.
- SHANNON, C. E. Communication in the presence of noise. **Proceedings of the IEEE**, v. 86, n. 2, p. 447 – 457, 1998.
- TOSATO, F.; BISAGLIA, P. Simplified soft-output demapper for binary interleaved cofdm with application to hiperlan/2. **IEEE International Conference on Communications**, v. 2, p. 664 – 668, 2004.
- TRUHACHEV, D. et al. Efficient implementation of 400 gbps optical communication fec. **IEE Transactions on Circuits and Systems I: Regular Papers**, v. 68, n. 1, p. 496 – 509, 2020.
- UNION, I. T. **Recommendation: ITU-T G.709.3/Y.1331.3 Flexible OTN long-reach interfaces**. 2020. Disponível em: <<https://www.itu.int/rec/T-REC-G.709.3/en>>.
- ZHOU, X.; XIE, C. **Enabling Technologies for High Spectral-efficiency Coherent Optical Communication Networks**. Hoboken, New Jersey - Estados Unidos da America: John Wiley and Sons, 2016.