Um Algoritmo de Decodificação Iterativa q-ária para Códigos Turbo Produto

Daniel C. Cunha[1], Vagner V. do Nascimento[2] e Jaime Portugheis[3]

***Resumo*— Este artigo considera o estudo de esquemas de codificação turbo produto construídos a partir de códigos Reed-Solomon componentes. Uma proposta de modificação do algoritmo de Chase é desenvolvida para satisfazer requisitos dos símbolos q-ários e utilizada na implementação de um algoritmo de decodificação iterativa q-ária. É verificado que o algoritmo de decodificação iterativa q-ária apresenta uma redução da complexidade quando comparado ao algoritmo turbo binário. Resultados de simulação são apresentados para mostrar a eficácia do algoritmo proposto.**

***Palavras-Chave*— Decodificação de Chase, Códigos produto, Confiabilidade q-ária.**

***Abstract*— This article considers the study of turbo product encoding schemes constructed from components Reed-Solomon codes. A proposal of modification of the Chase algorithm is developed to satisfy q-ary symbols requirements and is used in the implementation of a iterative q-ary decoding algorithm. It is verified that the iterative q-ary decoding algorithm presents a reduction of complexity when compared to the binary turbo algorithm. Simulation results are presented to show the effectiveness of the proposed algorithm.**

***Keywords*— Chase decoding, Product codes, q-ary reliability.**

I. INTRODUÇÃO

Desde a sua introdução em [1], a decodificação iterativa (turbo) de códigos produto ganhou muita importância no desenvolvimento de sistemas de transmissão digital com códigos corretores de erros. Ela já é empregada em sistemas de transmissão por satélite e se tornou uma opção tanto em redes metropolitanas sem fio como também em enlaces ópticos de alta velocidade [2]. Códigos produto são na sua forma padrão construídos através da concatenação serial de dois códigos de bloco lineares binários ou q-ários, separados por um entrelaçador linha-coluna a nível de símbolo. Os códigos RS têm sido empregados como códigos q-ários componentes da concatenação serial. Em [3] e [4] foram analisados códigos RS corretores de erros múltiplos e em [2], códigos RS corretores de erros simples. Entretanto, em todos os trabalhos citados anteriormente, cada símbolo q-ário é representado por bits e uma matriz de confiabilidades dos bits observados é montada no receptor. Sendo assim, o algoritmo de Chase binário pode ser aplicado na decodificação das linhas e colunas do código produto.

1 Departamento de Engenharia Elétrica, POLI, Universidade de Pernambuco (UPE), Recife-PE, Brasil.

2 Gerência de Tecnologia da Informação e Telecomunicações, Petróleo Brasileiro S.A. (PETROBRAS), Aracaju-SE, Brasil.

3 Departamento de Comunicações, FEEC, Universidade Estadual de Campinas (UNICAMP), Campinas-SP, Brasil.

E-mails: dccunha@upe.poli.br, vagnervale@gmail.com e jaime@decom.fee.unicamp.br.

Este trabalho considera sistemas de transmissão digital com códigos produto construídos a partir de códigos RS como códigos componentes. No receptor, uma matriz de confiabilidades dos símbolos q-ários observados é montada. Como consequência disto, um proposta de adaptação do algoritmo de Chase para satisfazer requisitos dos símbolos q-ários foi desenvolvida. Esta adaptação implica em uma redução da complexidade de decodificação quando comparada com a decodificação turbo binária. A adaptação sugerida permite uma realimentação apropriada dos símbolos q-ários e suas confiabilidades a cada iteração do decodificador turbo.

O artigo está organizado como se segue. O algoritmo Chase q-ário é descrito na seção II. Na seção III, o algoritmo turbo q-ário é detalhado. Na seção IV, os algoritmos turbo q-ário e o turbo binário são comparados através do compromisso entre complexidade e desempenho dos mesmos. Por fim, comentários finais são realizados na seção V.

II. ALGORITMO DE CHASE q-ÁRIO

Considere um código de bloco linear $C(n, k, d)$, em que n é o tamanho das palavras-código, k é o número de símbolos de informação e d é a distância de Hamming mínima do código. Com o objetivo de melhorar o desempenho da decodificação de códigos de bloco realizada por decisão abrupta, D. Chase propôs em [5] uma classe de algoritmos de decodificação que utiliza informação de medida de canal. Estes algoritmos são descritos a seguir.

Considere a sequência $\mathbf{Y} = (y_1, ..., y_l, ..., y_n)$, em que $y_l \in$ GF(q), $q = 2^m$, $m = 1$, obtida por decisão abrupta da palavra recebida $\mathbf{R} = (r_1, ..., r_l, ..., r_n)$. Para altos valores de Relação Sinal-Ruído (RSR), a palavra-código correta $\mathbf{C} = (c_1, ..., c_l, ..., c_n)$ é localizada, com probabilidade alta, em uma esfera de raio $(d-1)$ centrada no vetor $\mathbf{Y}$. A partir daí, a sequência $\mathbf{Y}$ pode ser perturbada por um padrão de teste $\mathbf{T}$ para se gerar uma nova sequência

$$\mathbf{Y}' = \mathbf{Y} \oplus \mathbf{T} \ , \tag{1}$$

em que $\oplus$ representa a operação soma módulo 2. A alteração da sequência $\mathbf{Y}$ por $\mathbf{T}$ tem por objetivo encontrar a palavra-código correta $\mathbf{C}$ dentro da esfera de raio $(d-1)$. Usando esta regra, o desempenho da decodificação pode se aproximar do desempenho do processo de decodificação completa sem a complexidade de procura da palavra-código correta em uma esfera de raio $(d-1)$. Este método de decodificação é possível de ser executado devido à informação de medida de canal, ou confiabilidade, extraída a partir da palavra observada na saída do canal. Os métodos completos de decodificação de Chase são descritos em [5].

Os algoritmos de Chase propostos em [5] foram aplicados apenas para códigos binários. A utilização da decodificação de Chase para códigos q-ários necessita de conversões bit-símbolo e símbolo-bit que podem afetar a precisão do cálculo de confiabilidade do símbolo, principalmente em sistemas que utilizam um esquema de modulação com mais de dois sinais. Com isso, as conversões mencionadas podem comprometer o desempenho da decodificação de Chase.

Para ilustrar esta perda de desempenho, considere uma aplicação que utiliza o código RS (7,3,5), com capacidade de correção $t_c = \lfloor \frac{d-1}{2} \rfloor = 2$. Suponha que a palavra-código $\mathbf{X}_A$ seja transmitida e que a palavra recebida na entrada do decodificador seja $\mathbf{Y}$, de acordo com a Fig. 1.

Palavra-código transmitida:

$X_A = [\ 3 \quad 1 \quad 7 \quad 7 \quad 5 \quad 1 \quad 5\ ]$

$X_{Abits} = [011\ 001\ 111\ 111\ 101\ 001\ 101]$

Palavra recebida:

$Y = [\ 4 \quad 1 \quad 7 \quad 0 \quad 5 \quad 6 \quad 2\ ]$

$Y_{bits} = [100\ 001\ 111\ 000\ 101\ 110\ 010]$

Erros:

$E = Y - X_A = [\ 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 5 \quad 5\ ]$

$E_{bits} = Y_{bits} - X_{Abits} = [111\ 000\ 000\ 111\ 000\ 111\ 111]$

Fig. 1. Exemplo de erro em uma palavra do código RS(7,3,5) transmitida.

Fazendo-se as conversões símbolo-bit das sequências $\mathbf{X}_A$ e $\mathbf{Y}$, são geradas as sequências $\mathbf{X}_{Abits}$ e $\mathbf{Y}_{bits}$, respectivamente. Considerando a utilização do Algoritmo de Chase II para aumentar a capacidade de correção do código, é possível adicionar $2^{\lfloor d/2 \rfloor} = 2^{\lfloor 5/2 \rfloor} = 4$ padrões de teste $\mathbf{T}$ à palavra $\mathbf{Y}_{bits}$, permitindo, no máximo, a modificação das $p = \lfloor d/2 \rfloor$ posições de menores confiabilidades da palavra recebida $\mathbf{Y}_{bits}$. Entretanto, mesmo com a aplicação do Algoritmo de Chase II à sequência $\mathbf{Y}_{bits}$ e uma posterior conversão bit-símbolo para se obter a nova sequência $\mathbf{Y}'$, o decodificador algébrico não conseguiria decodificar $\mathbf{Y}'$ corretamente, pois ele é capaz de corrigir no máximo $t_c = 2$ erros de símbolos. Para decodificar a palavra recebida $\mathbf{Y}$ de maneira correta, por meio da técnica de Chase, seria necessário manipular a palavra $\mathbf{Y}_{bits}$ em pelo menos 2 grupos de 3 posições (bits) consecutivas, totalizando 6 posições. É possível desenvolver um algoritmo que atenda esta necessidade, porém ele teria uma quantidade grande de padrões de teste e, consequentemente, seria mais complexo.

Com o intuito de melhorar o desempenho da decodificação para códigos q-ários, é proposta uma modificação no método de Chase. Ela é baseada no Algoritmo de Chase II devido à sua boa relação entre desempenho e complexidade. Considere novamente o exemplo do código RS(7,3,5) e assuma que a palavra recebida é a sequência $\mathbf{Y}$ em substituição à sequência $\mathbf{Y}_{bits}$, oriunda da conversão símbolo-bit. Assuma que são gerados padrões de teste que permitam a substituição dos símbolos q-ários das posições com baixa confiabilidade por outros símbolos q-ários mais adequados. Desta maneira, são necessárias substituições em apenas duas posições de $\mathbf{Y}$ para que o decodificador algébrico consiga decodificar a nova sequência $\mathbf{Y}'$ em uma palavra-código válida do código RS(7,3,5). Por exemplo, se o padrão de teste $\mathbf{T} = [7\,0\,0\,7\,0\,0\,0]$ for somado (módulo 8) à palavra recebida $\mathbf{Y}$ indicada na Fig. 1, será obtida a nova palavra $\mathbf{Y}'$, dada por

$$\begin{aligned} \mathbf{Y}' &= \mathbf{Y} + \mathbf{T} \\ &= [4\ 1\ 7\ 0\ 5\ 6\ 2] + [7\ 0\ 0\ 7\ 0\ 0\ 0] \\ &= [3\ 1\ 7\ 7\ 5\ 6\ 2]\ . \end{aligned}$$

Note que a sequência $\mathbf{Y}'$ agora pode ser decodificada pelo decodificador algébrico q-ário, pois possui apenas dois erros de símbolo quando comparada à sequência $\mathbf{X}_A$.

Segundo o exemplo anterior, o método de Chase modificado para suportar a decodificação de códigos q-ários, sem a necessidade de conversões símbolo-bit e bit-símbolo, deve seguir o mesmo algoritmo definido em [5]. Todavia, a obtenção do conjunto de padrões de teste difere das formas empregadas nas três versões dos algoritmos originais de Chase [5], pois agora são considerados símbolos q-ários. Logo, a primeira mudança significativa para a proposta do algoritmo de decodificação q-ário, é a modificação da geração de padrões de teste.

O Algoritmo de Chase II define o conjunto de padrões de teste como todas as possíveis combinações de 1's nas $\lfloor d/2 \rfloor$ posições de menor confiabilidade da palavra recebida e zeros nas demais posições. O número de padrões de teste poderia ser alterado simplesmente aumentando-se a quantidade de posições não confiáveis da palavra $\mathbf{Y}_{bits}$ proporcionalmente ao aumento do número de posições da sequência $\mathbf{Y}$ para a sequência $\mathbf{Y}_{bits}$, ou seja, $\log_2(q)$ vezes. Desta maneira, o número de posições não confiáveis de $\mathbf{Y}_{bits}$ passaria a ser $\lfloor d/2 \rfloor \log_2(q)$, resultando em uma quantidade de padrões de teste dada pela expressão:

$$2^{\lfloor d/2 \rfloor \cdot \log_2(q)} = \left(2^{\log_2(q)}\right)^{\lfloor d/2 \rfloor} = q^{\lfloor d/2 \rfloor}\ . \qquad (2)$$

Apesar desta ser uma solução simples, a expressão (2) indica que a quantidade de padrões de teste tende a crescer exponencialmente com a dimensão do corpo de Galois do código q-ário utilizado. Logo, a aplicação deste algoritmo de Chase modificado seria impraticável devido à sua alta complexidade. Para reforçar o que foi dito, considere novamente o código RS(7,3,5) sobre o GF(8). A alteração do número de padrões de teste sugerida pela expressão (2) geraria um aumento de $2^{\lfloor d/2 \rfloor} = 2^{\lfloor 5/2 \rfloor} = 4$ para $q^{\lfloor d/2 \rfloor} = 8^{\lfloor 5/2 \rfloor} = 64$ padrões de teste.

Para reduzir a complexidade da modificação do algoritmo de Chase proposta, é necessário uma alteração no cálculo da confiabilidade do símbolo recebido. Sabe-se que a sequência $\mathbf{Y}$ é gerada no demodulador por meio de decisão abrupta e que o símbolo escolhido $y_l = \gamma_0$ obedece à seguinte expressão:

$$P(y_l = \gamma_0 | r_l) > P(y_l = \gamma | r_l), \quad \forall\ \gamma \neq \gamma_0, \qquad (3)$$

em que r_l é uma estatística suficiente para o l-ésimo símbolo transmitido e γ_0, $\gamma \in GF(q)$.

Fazendo uso de uma inequação similar à (3), é possível gerar uma segunda sequência $\mathbf{Y}^a$, obtida também por decisão

abrupta, com os símbolos $y_l^a = \gamma_1$ tal que:

$$P(y_l = \gamma_0|r_l) > P(y_l = \gamma_1|r_l) > P(y_l = \gamma|r_l), \quad \forall\ \gamma \neq \gamma_1 \neq \gamma_0\,, \tag{4}$$

ou seja, a sequência $\mathbf{Y}^a$ conterá os símbolos y_l^a com a segunda maior probabilidade de terem sido transmitidos dado que r_l foi observado.

Utilizando $\mathbf{Y}^a$, a sequência $\mathbf{Y}'$ pode assumir, alternativamente, os símbolos provenientes das $\lfloor d/2 \rfloor$ posições menos confiáveis das sequências $\mathbf{Y}$ e $\mathbf{Y}^a$. Sendo assim, os símbolos do padrão de teste são obtidos como

$$t_l = \begin{cases} y_l^a - y_l, & y_l^{'} = y_l^a \\ 0 \quad , & y_l^{'} = y_l \end{cases} . \tag{5}$$

Com isto, teremos um conjunto q-ário com apenas $2^{\lfloor d/2 \rfloor}$ padrões de teste, ou seja, a mesma quantidade do método binário.

Semelhante ao caso binário, a confiabilidade de um símbolo q-ário recebido y_l pode ser calculada por meio de Razão de Log-Verossimilhança (LLR, do inglês *Log-Likelihood Ratio*). Entretanto, para o caso q-ário, são considerados todos os símbolos do alfabeto $GF(q)$ para gerar a seguinte LLR:

$$\Lambda(y_l|r_l) = \ln\left(\frac{P(y_l^{(1)}|r_l)}{\sum_{b=2}^{q} P(y_l^{(b)}|r_l)}\right), \tag{6}$$

em que $y_l^{(b)}$ é o símbolo q-ário com a b-ésima maior probabilidade *a posteriori*.

Note que o símbolo $y_l^{(1)}$ representa a l-ésima posição obtida a partir da sequência $\mathbf{Y}$ e o símbolo $y_l^{(2)}$ é a l-ésima posição obtida a partir da sequência $\mathbf{Y}^a$. Aplicando a regra de Bayes e considerando $P(y_l^{(b)})$ constante, a equação (6) pode ser escrita como

$$\Lambda(y_l|r_l) = \ln\left(\frac{P(r_l|y_l^{(1)})}{\sum_{b=2}^{q} P(r_l|y_l^{(b)})}\right) . \tag{7}$$

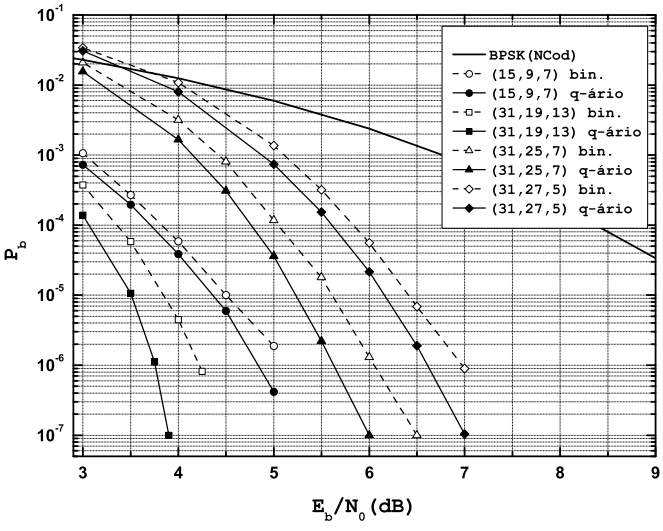

Fig. 2. Comparação de desempenho das decodificações de Chase binária e q-ária para códigos RS típicos.

A Fig. 2 ilustra o desempenho dos algoritmos de decodificação de Chase binário (Chase II) e q-ário para os códigos Reed-Solomon (RS) $(15, 9, 7)$, $(31, 19, 13)$, $(31, 25, 7)$ e $(31, 27, 5)$ por meio da curva de taxa de erro de bit (P_b) versus E_b/N_0, em que E_b é a energia média por bit de informação e N_0, a densidade espectral de potência do ruído AWGN. A modulação BPSK (do inglês, *Binary Phase Shift Keying*) é considerada. Para o código RS $(15, 9, 7)$ e considerando que $E_b/N_0 = 5\,\text{dB}$, verifica-se uma redução de 5 vezes na taxa de erro de bit em favor do algoritmo proposto. Em relação aos códigos RS $(31, 25, 7)$ e $(31, 27, 5)$, tem-se uma redução da P_b de aproximadamente 10 vezes para $E_b/N_0 = 6$ e 7 dB, respectivamente. Por fim, o código RS $(31, 19, 13)$ apresenta uma redução de 20 vezes em P_b para $E_b/N_0 = 3,75\,\text{dB}$ ao se utilizar o algoritmo de decodificação q-ário.

III. DECODIFICAÇÃO TURBO q-ÁRIA

A. Códigos Produto

Códigos produto foram introduzidos por Elias [6] e representam uma maneira simples e relativamente eficiente para a construção de códigos de bloco longos a partir de códigos componentes curtos. Códigos produto bidimensionais clássicos são obtidos a partir da concatenação serial de dois códigos de bloco lineares $C_1(n_1, k_1, d_1)$ e $C_2(n_2, k_2, d_2)$ sobre GF(q). O código produto resultante $P(n_p, k_p, d_p)$ possui comprimento $n_p = n_1 n_2$, $k_p = k_1 k_2$ símbolos de informação e distância mínima $d_p = d_1 d_2$. As n_1 linhas do código produto são palavras-código de C_1 e as n_2 colunas são palavras-código de C_2 [7]. Neste trabalho, consideraremos a utilização de códigos produto Reed-Solomon (RS) compostos por códigos componentes RS sobre GF(q) idênticos.

B. Decodificação Turbo q-ária de Códigos Produto

As linhas e colunas de um código produto P podem ser decodificadas por um método iterativo chamado decodificação turbo [1]. Seja $\mathbf{\Lambda}(\mathbf{Y})$ a matriz de confiabilidade dos símbolos recebidos. O diagrama do decodificador turbo binário é ilustrado na Fig. 3 (conforme as linhas cheias).

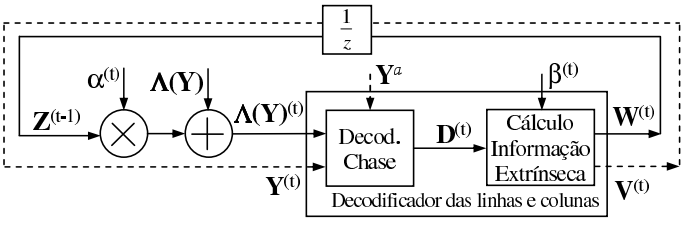

Fig. 3. Diagrama do decodificador turbo produto.

Primeiramente, o decodificador executa a decodificação suave (decodificação de Chase) das linhas (ou colunas) de P utilizando $\mathbf{\Lambda}(\mathbf{Y})$ como a matriz de entrada. Assim, a matriz decodificada por Chase, $\mathbf{D}^{(t=1)}$, é gerada depois da primeira iteração ($t = 1$). A entrada suave para a t-ésima iteração do decodificador é dada por

$$\Lambda(\mathbf{Y})^{(t)} = \Lambda(\mathbf{Y}) + \alpha^{(t)}\mathbf{Z}^{(t-1)}, \tag{8}$$

em que $\alpha^{(t)}$ é um fator de escala (vide [8] e [9]) que varia no intervalo $[0,0;1,0]$ de acordo com [1].

De acordo com [10], a saída suave $\mathbf{\Lambda}(d_{ij})^{(t)}$ pode ser aproximada por

$$\mathbf{\Lambda}(d_{ij})^{(t)} = \beta^{(t)}\mathbf{\Phi}(d_{ij})^{(t)}, \quad (9)$$

em que $\beta^{(t)}$ é um fator de confiabilidade que varia no intervalo $[0,2;1,0]$ segundo [1] e $\mathbf{\Phi}(d_{ij})^{(t)}$ é a LLR da decisão $d_{ij}^{(t)}$, que é definida por:

$$\mathbf{\Phi}(d_{ij})^{(t)} = \ln\left(\frac{P(d_{ij}^{(t)}|x_{ij}=\gamma_0)}{\sum_{h=0 \quad h\neq\gamma_0}^{q-1} P(d_{ij}^{(t)}|x_{ij}=h)}\right), \quad (10)$$

em que $\mathbf{X}$ é a matriz obtida na saída do codificador produto e $\gamma_0 \in GF(q)$.

Desta maneira, a informação extrínseca é dada por:

$$\mathbf{W}^{(t)} = \mathbf{\Lambda}(d_{ij})^{(t)} - \mathbf{\Lambda}(\mathbf{Y})^{(t)}. \quad (11)$$

O algoritmo utiliza a matriz de sequências $\mathbf{Y}^a$ e a matriz $\mathbf{\Lambda}(\mathbf{Y})$ que são obtidas de acordo com as equações (4) e (7), respectivamente. No caso binário, podemos identificar o bit y_{ij} pelo sinal da sua confiabilidade $\mathbf{\Lambda}(y_{ij})$ (positivo ou negativo). Entretanto, no caso q-ário, o valor do símbolo não está implícito na confiabilidade, porque o símbolo pode assumir q valores de acordo com o alfabeto utilizado. Assim, o decodificador turbo q-ário deve ser adicionalmente alimentado com as matrizes de sequências $\mathbf{Y}^{(0)} = \mathbf{Y}$ e $\mathbf{Y}^a$.

Em cada iteração do decodificador turbo q-ário, o valor do símbolo $y_{ij}^{(t)}$ da t-ésima matriz de entrada $\mathbf{Y}^{(t)}$ deve ser comparado com o valor do símbolo $d_{ij}^{(t)}$ da t-ésima matriz decodificada $\mathbf{D}^{(t)}$. Se $y_{ij}^{(t)} = d_{ij}^{(t)}$, significa que o decodificador de Chase q-ário não corrigiu o símbolo de entrada. Desta maneira, a informação extrínseca $\mathbf{W}^{(t)}$ para a próxima iteração deve ser obtida conforme é ilustrado na Fig. 3 (linhas tracejadas). Observa-se ainda que a t-ésima matriz de símbolos decodificados $\mathbf{V}^{(t)}$ deve retornar à entrada do decodificador q-ário. Sendo assim, quando $y_{ij}^{(t)} \neq d_{ij}^{(t)}$, um método diferente para a realimentação do decodificador turbo q-ário é proposto conforme ilustrado na Fig. 4 e descrito a seguir.

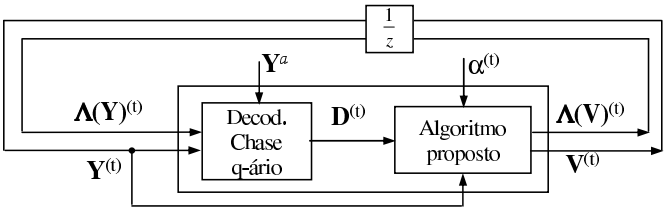

Fig. 4. Decodificador turbo q-ário para $y_{ij}^{(t)} \neq d_{ij}^{(t)}$.

No caso binário, o bit é identificado através da confiabilidade. Desta maneira, pode-se considerar que o valor da confiabilidade $\mathbf{\Lambda}(y_{ij})$ varia dentro de um segmento de reta de intervalo [-1;+1], de acordo com a Fig. 5a. Devido à diversidade de símbolos, não é possível fazer esta mesma consideração no caso q-ário. Logo, quando $y_{ij}^{(t)} \neq d_{ij}^{(t)}$, o método proposto, ilustrado na Fig. 5b, é aplicado para calcular o valor da confiabilidade para próxima iteração, agora denotada por $\Lambda(v_{ij})^{(t)}$. Na Fig. 5b, para a t-ésima iteração, o ponto A no eixo das abscissas representa o valor da confiabilidade $\Lambda(y_{ij})^{(t)}$ do símbolo de entrada, enquanto o ponto B no eixo das ordenadas representa o valor da confiabilidade $\Lambda(d_{ij})^{(t)}$ do símbolo decodificado pelo algoritmo de Chase q-ário. Podemos encontrar o ponto C traçando uma reta perpendicular ao segmento $\overline{AB}$ de tal forma que o segmento $\overline{AC}$ obedeça à expressão $\overline{AC} = \alpha^{(t)} \times \overline{AB}$. Por meio de alguns cálculos trigonométricos básicos, podemos obter o segmento $\overline{OD}$, cujo valor representará a confiabilidade $\Lambda(v_{ij})^{(t)}$ da próxima iteração. Note que, dependendo de A, B e $\alpha^{(t)}$, o ponto D poderá estar nos eixos das abscissas $\Lambda(y_{ij})^{(t)}$ ou das ordenadas $\Lambda(d_{ij})^{(t)}$ e sua posição definirá se o símbolo de entrada da próxima iteração v_{ij} será igual a y_{ij} ou d_{ij}, respectivamente.

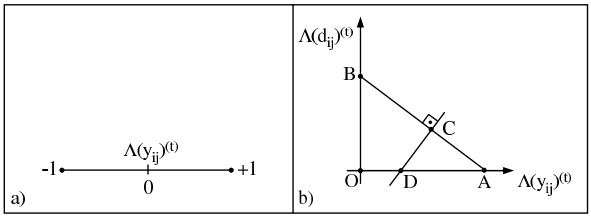

Fig. 5. Projeção da decodificação turbo: a) binária e b) q-ária.

IV. COMPARAÇÃO DE COMPLEXIDADE VERSUS DESEMPENHO

Na seção II, uma modificação do algoritmo de Chase foi proposta para códigos q-ários. Este algoritmo foi empregado em uma nova proposta de algoritmo de decodificação turbo apresentada na seção III. A partir deste momento, faremos uma comparação de complexidade e desempenho do algoritmo turbo q-ário proposto e o algoritmo turbo binário definido por Pyndiah em [1].

Como referência para a comparação, foram considerados os resultados obtidos em [4] para o algoritmo turbo binário ([1]). Foram utilizados os fatores $\alpha^{(s)}$ e $\beta^{(s)}$ variando a cada meia iteração s (linha/coluna) do decodificador e assumindo os valores estabelecidos em [1]. Os códigos produto considerados foram compostos pelos códigos RS $(31,27,5)$ e $(63,57,7)$. O algoritmo de Berlekamp-Massey (BM) foi utilizado para a decodificação de decisão abrupta e o parâmetro p do algoritmo de Chase foi considerado igual a 4. Por fim, $t = 4$ iterações completas do decodificador turbo foram realizadas.

Para o algoritmo turbo q-ário proposto neste trabalho, foram usados os mesmos códigos RS componentes mencionados anteriormente, assim como o algoritmo de BM. Em relação aos fatores α e β, foram considerados os valores $\alpha^{(s)} = [0,0;0,2;0,3;0,5;0,7;0,9;1,0;1,0]$ e $\beta^{(s)} = [0,2;0,4;0,6;0,8;1,0;1,0;1,0;1,0]$. Finalmente, o parâmetro p do algoritmo de Chase q-ário foi estabelecido de acordo com o algoritmo de Chase original, ou seja, $p = \lfloor d/2 \rfloor$.

Sejam N_s, N_m, N_c e N_g, as operações de soma, multiplicação, comparação e soma módulo 2, respectivamente. Estas operações foram selecionadas para avaliar a

complexidade computacional de cada algoritmo, baseado no que foi empregado em [11]. Na Tabela I, são indicadas as operações referentes ao algoritmo turbo binário. É importante ressaltar que a análise realizada em [11] levou em consideração o uso de códigos de Hamming estendidos. Para o nosso caso, em que códigos RS são considerados, alguns passos do algoritmo foram adaptados.

Com o intuito de computar a quantidade de operações do algoritmo turbo q-ário, este será descrito conforme os seguintes passos:

1) Encontrar os dois símbolos com as duas maiores probabilidades *a posteriori* $P(y_l^{(1)}|r_l)$ e $P(y_l^{(2)}|r_l)$, a partir das probabilidades *a posteriori* dos bits recebidos, e gerar os símbolos referentes às seqüências $\mathbf{Y}$ e $\mathbf{Y}^a$, respectivamente.
2) Calcular a confiabilidade de cada símbolo da seqüência $\mathbf{Y}$

$$\Lambda(y_l|r_l) = \ln\left(\frac{P(y_l^{(1)}|r_l)}{1-P(y_l^{(1)}|r_l)}\right), l = 1, \ldots, n\,.$$

3) Determinar as p posições menos confiáveis da seqüência $\mathbf{Y}$.
4) Gerar os 2^p padrões de teste e suas respectivas seqüências perturbadas $\mathbf{Y}'$. Para isso, é necessário apenas substituir os símbolos das posições menos confiáveis de $\mathbf{Y}$ pelos símbolos referentes em $\mathbf{Y}^a$ de acordo com os padrões de teste.
5) Decodificar a sequência perturbada $\mathbf{Y}'$ por meio do algoritmo de BM.
6) Calcular o peso analógico

$$W_\Lambda(\mathbf{Z}^e) = \sum_{i=1}^{n} \Lambda_i \cdot z_i^e\,,$$

em que $\mathbf{Z}^e$ é o padrão de erro q-ário $\mathbf{E}$ com as posições não nulas substituídas por 1.
7) Obter a palavra decodificada $\mathbf{D}$ por Chase a partir do padrão de erro de menor peso analógico $(\mathbf{E}_{min})$.
8) Calcular a confiabilidade da decodificação, considerando o canal AWGN. Observe que, conforme [10], calculamos as confiabilidades para os bits

$$\Lambda(d_{ij,f}) = \beta\sqrt{E_b}\,,$$

em que f varia de 0 a $m-1$. Em seguida, geramos as confiabilidades para os símbolos.
9) Calcular os valores das confiabilidades e dos símbolos para próxima iteração segundo o procedimento proposto na seção III.

Na Tabela II, são indicadas as operações contabilizadas para o algoritmo turbo q-ário. Vale ressaltar que as operações referentes aos dois algoritmos consideram apenas a complexidade computacional de uma palavra-código componente. Além disso, os passos 4 (Tabela I) e 5 (Tabela II) se referem à decodificação de BM e, por gerar o número de operações nos dois casos, não são levados em conta para a comparação de complexidade.

TABELA I

COMPLEXIDADE DE DECODIFICAÇÃO DO ALGORITMO TURBO BINÁRIO.

Passo	N_s	N_m	N_c	N_g
1	-	-	$mn(p+1)$	-
2	-	-	-	$p2^p$
3	-	-	-	$mn2^p$
4	-	-	-	-
5	$mn2^p$	$mn2^p$	-	-
6	-	-	2^p	-
7	$2mn$	$2mn$	$mn(2^p-1)$	-

Para consolidar a comparação entre as complexidades dos algoritmos turbo binário e q-ário, definimos a Razão de Complexidade (RC) como a razão entre os números totais de operações $(N_s + N_m + N_c + N_g)$ dos algoritmos binário e q-ário. Para o código RS$(31, 27, 5)^2$, foi obtido um valor de RC aproximadamente igual a $8,6$, enquanto para o código RS $(63, 57, 7)^2$, o valor de RC aproximado foi de $7,7$, resultando em uma menor complexidade do algoritmo turbo q-ário.

TABELA II

COMPLEXIDADE DE DECODIFICAÇÃO DO ALGORITMO TURBO q-ÁRIO.

Passo	N_s	N_m	N_c	N_g
1	-	$(m-1)n$	$(2m-1)n$	-
2	n	n	-	-
3	-	-	$np - \sum_{i=1}^{p} i$	-
4	-	-	$(2^p-1)p$	-
5	-	-	-	-
6	$(n-1)2^p$	$n2^p$	-	-
7	-	-	2^p-1	-
8	-	$(2m-1)n$	-	-
9	$2n$	$4n$	$2n$	-

Na Fig. 6 é ilustrado o desempenho dos algoritmos turbo binário e q-ário em um canal AWGN. As curvas referentes à decodificação turbo binária foram obtidas a partir de [4]. A modulação BPSK foi considerada. Os códigos produto RS$(31, 27, 5)^2$ e RS$(63, 57, 7)^2$ foram utilizados e as curvas P_b versus E_b/N_0 da decodificação turbo q-ária foram geradas através de simulações.

Para o código produto RS$(31, 27, 5)^2$, foram transmitidos $\sim 2,2 \cdot 10^8$ bits de informação. Observe que, para alcançar uma $P_b = 10^{-5}$, a decodificação turbo q-ária necessita de uma $E_b/N_0 \simeq 4,25\,dB$. A decodificação turbo binária alcança esta mesma P_b para $E_b/N_0 \simeq 4,05\,dB$. Apesar do algoritmo turbo binário oferecer um ganho de aproximadamente $0,2\,dB$, a sua complexidade da decodificação mostra-se maior que a complexidade q-ária. Além disso, uma possível justificativa para o melhor desempenho do algoritmo turbo binário foi o fato de ele ter utilizado o parâmetro p do algoritmo de Chase maior do que o previsto pelo algoritmo original definido em [5], ou seja, o algoritmo turbo binário utilizou um valor maior $(p = 4)$ do que $\lfloor d/2 \rfloor$.

Já para o código produto RS$(63, 57, 7)^2$, foram transmitidos $\sim 1,2 \cdot 10^9$ bits de informação. Para $E_b/N_0 \simeq$

$4,5\,dB$, percebe-se que o algoritmo turbo q-ário oferece, com menor complexidade computacional, uma redução de aproximadamente 5 vezes em P_b comparado ao algoritmo turbo binário. Vale ressaltar que, assim como no caso do código RS$(31,27,5)^2$, o algoritmo turbo binário para o código RS$(63,57,7)^2$ utilizou o parâmetro $p = 4$, diferente do algoritmo turbo q-ário, que utilizou $p = 3$, de acordo com o algoritmo original definido em [5].

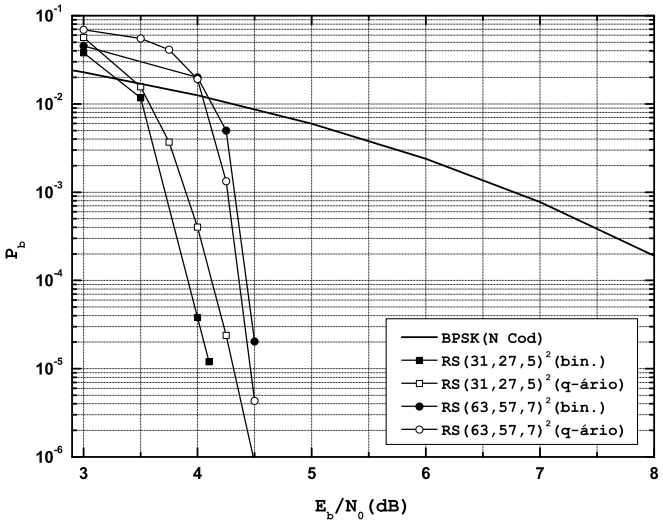

Fig. 6. Desempenho dos algoritmos de decodificação turbo binária e q-ária para os códigos produto RS $(31,27,5)^2$ e $(63,57,7)^2$.

V. COMENTÁRIOS FINAIS

Este trabalho considerou o estudo de esquemas de codificação produto construídos a partir de códigos RS componentes. Uma proposta de modificação do algoritmo de Chase binário foi desenvolvida para satisfazer requisitos dos símbolos q-ários. Segundo os resultados apresentados na seção II, o método de decodificação de Chase modificado para símbolos q-ários superou em desempenho o método original de Chase para os códigos RS apresentados, sem incrementar a complexidade de decodificação.

A modificação do algoritmo de Chase foi utilizada na implementação de um algoritmo de decodificação iterativa q-ária. Foi realizada uma comparação da complexidade computacional versus desempenho dos algoritmos de decodificação turbo binária e q-ária. A complexidade computacional foi avaliada por meio do cálculo do número de operações matemáticas efetuadas em cada um dos algoritmos. Foi observado que o algoritmo turbo q-ário se mostrou menos complexo que o algoritmo turbo binário para os códigos produto RS$(31,27,5)^2$ e RS$(63,57,7)^2$. Adicionalmente, foi verificado que o desempenho do algoritmo turbo binário foi ligeiramente superior ao algoritmo q-ário para o código RS$(31,27,5)^2$, justificado pela utilização de um maior conjunto de padrões de teste no algoritmo de Chase. Por fim, ficou evidenciado o melhor desempenho do algoritmo turbo q-ário frente ao algoritmo binário para o código RS$(63,57,7)^2$.

VI. AGRADECIMENTOS

Este trabalho foi parcialmente financiado pela Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE).

REFERÊNCIAS

[1] R. Pyndiah, "Near Optimum Decoding of Product Codes: Block Turbo Codes," *IEEE Trans. on Communications*, vol. 2, n. 8, pp. 1003-1010, Ago 1998.

[2] R. Zhou, R.L. Bidan, R. Pyndiah e A. Goalic, "Low-Complexity High-Rate Reed-Solomon Block Turbo Codes," *IEEE Trans. on Communications*, vol. 55, n. 9, pp. 1656-1660, Set 2007.

[3] O. Aitsab e R. Pyndiah, "Performance of Reed-Solomon Block Turbo Codes," In: *Proceedings of the IEEE Global Telecommunications Conference 1996 (GLOBECOM'96)*, pp. 121-125, Londres, Inglaterra, Nov 1996.

[4] C. Argon, S.W. McLaughlin e T. Souvignier, "Iterative Application of the Chase Algorithm on Reed-Solomon Product Codes," In: *Proceedings of the IEEE Int. Conf. on Communications 2001 (ICC'01)*, vol.1, pp. 320-324, Helsinque, Finlândia, Jun 2001.

[5] D. Chase, "A Class of Algorithms for Decoding Block Codes with Channel Measurement Information," *IEEE Trans. on Information Theory*, vol. IT-18, n. 1, pp. 170-182, Jan 1972.

[6] P. Elias, "Error-free Coding," *IEEE Trans. on Information Theory*, vol. IT-4, pp. 29-37, 1954.

[7] F.J. MacWilliams e N.J. Sloane, *The Theory of Error Correcting Codes*. Amsterdam, The Netherlands: North-Holland, 1978.

[8] C. Berrou, A. Glavieux e P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding:Turbo-Codes(1)," *IEEE Int. Conf. on Communications (ICC'93)*, vol. 2/3, pp. 1064-1071, Mai 1993.

[9] C. Berrou e A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Trans. on Communications*, vol. 44, pp. 1261-1271, Out 1996.

[10] D. C. Cunha e J. Portugheis, "Decodificação Iterativa (Turbo) de Códigos Produto em Canais Não-Gaussianos," In: *Anais do XX Simpósio Brasileiro de Telecomunicações 2003 (SBrT'03)*, Rio de Janeiro-RJ, Brasil, Out 2003.

[11] C. Xu, Y.-C. Liang e W.S. Leon, "A Low Complexity Decoding Algorithm for Extended Turbo Product Codes," *IEEE Trans. on Wireless Communications*, vol. 7, n.1, pp. 43-47, Jan 2008.