4
Códigos Turbo e a Teoria dos Grafos-Fatores

É uma idéia natural a de que uma função complexa envolvendo muitas variáveis pode ser mais facilmente manipulada quando reescrita, caso isto seja possível, como o produto de funções locais mais simples. A teoria de grafos-fatores [25] formaliza essa idéia através do conceito de *grafo-fator*, um grafo bi-particionado que representa de forma concisa a fatoração de uma determinada função global.

A partir de um grafo-fator representando uma função global, funções marginais associadas a essa função global podem ser calculadas eficientemente através de um algoritmo genérico conhecido como *algoritmo Soma-Produto*. Este algoritmo explora a estrutura do grafo-fator — isto é, a maneira na qual a função global é fatorada — para reutilizar cálculos intermediários e assim economizar na complexidade do cálculo das funções marginais.

Como mostrado em [25], o algoritmo BCJR utilizado na decodificação turbo pode ser visto como um caso particular do algoritmo Soma-Produto. Uma das motivações de se avaliar a decodificação turbo através da teoria dos grafos-fatores consiste na vantagem didática que esta teoria apresenta em relação a abordagem convencional desenvolvida no Cap. 3, pois esclarece de maneira simples algumas questões importantes sobre a decodificação turbo.

O objetivo deste capítulo é portanto apresentar a teoria de grafos-fatores, bem como sua aplicabilidade para compreensão da decodificação turbo, segundo uma abordagem moderna que denominamos de abordagem por grafos-fatores.

A história dos grafos-fatores e do algoritmo Soma-Produto tem início em 1962 com o trabalho de Gallager [2], que introduziu os códigos de matriz de paridade de baixa densidade (LDPC) e um algoritmo iterativo para decodificação através do cálculo das probabilidades a posteriori, o qual é considerado a primeira instância do algoritmo Soma-Produto a aparecer na literatura. Em 1981, Tanner [3] introduziu grafos bi-particionados para descrever famílias de códigos que são generalizações dos códigos de Gallager.

Este tipo de grafo é hoje conhecido como grafo de Tanner. Em 1995, Wiberg et al. [8] estendem os grafos de Tanner ao introduzir variáveis de estado, e com isso proporcionam uma conexão com a descrição de códigos através de treliças e com o algoritmo BCJR usado na decodificação turbo. McEliece et al. mostram em [12] que o algoritmo de decodificação turbo pode ser visualizado como uma instância do algoritmo *belief propagation* usado em inteligência artificial. Finalmente, Kschischang and Frey [13] observam que muitos algoritmos importantes, utilizados em diversas áreas como inteligência artificial, processamento de sinais e comunicações digitais, incluindo o algoritmo *belief propagation*, o algoritmo BCJR, o algoritmo de Viterbi e o algoritmo de decodificação turbo, podem todos ser obtidos como instâncias do algoritmo Soma-Produto quando aplicado ao grafo apropriado. Aji e McEliece também obtém resultado semelhante através de uma abordagem diferente [21]. Como mencionado em [25], o conceito de grafos-fatores como uma generalização dos grafos de Tanner foi desenvolvido conjuntamente por um grupo que incluiu, além de Kschischang, Frey e Loeliger, também Forney, Koetter, MacKay, McEliece, Tanner e Wiberg.

Este capítulo organiza-se da seguinte forma. Na Seção 4.1 é abordada a teoria de grafos-fatores, seus principais conceitos e definições. Na Seção 4.2, é mostrado como esta teoria pode ser utilizada para a modelagem do problema da decodificação. Na Seção 4.3 é mostrado que o algoritmo MAP é uma instância do algoritmo Soma-Produto. Por fim, na Seção 4.4 é realizada a interpretação do problema da decodificação turbo nesta abordagem.

4.1 Marginalização Utilizando Grafos-Fatores

4.1.1 Fatoração e os Grafos-Fatores

Considerando $\{x_1, x_2, \ldots, x_N\}$ um conjunto de variáveis, que para cada i, x_i assume valores de um alfabeto $\mathcal{A}_i$. Seja $g(x_1, x_2, \ldots, x_N)$ uma função dessas variáveis, com domínio $S = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_N$ e contradomínio R, onde [31]

Definição 4.1 (Semi-Anel Comutativo) Um semi-anel comutativo $\langle R, +, \cdot \rangle$ é formado por um conjunto R associado a duas operações binárias $+$ e $\cdot$ que satisfaz os seguintes axiomas:

Axioma 1 A operação $+$ é associativa e comutativa, e há uma identidade aditiva "0", de modo que $r + 0 = r$ para todo $r \in R$.

Axioma 2 A operação $\cdot$ é associativa e comutativa, e há uma identidade multiplicativa "1", de modo que $1 \cdot r = r$ para todo $r \in R$.

Axioma 3 A lei distributiva é satifeita: $(r_1 \cdot r_2) + (r_1 \cdot r_3) = r_1 \cdot (r_2 + r_3)$. ■

Anéis são bastante úteis e interessantes para o desenvolvimento de algoritmos de decodificação, conforme será visto na Seção 4.3.1. A Tabela 4.1 apresenta alguns exemplos.

Tabela 4.1: Semi-Anéis Comutativos

Conjunto R	"$(+, 0)$"	"$(\cdot, 1)$"	Nome do Semi-Anel
$[0, \infty)$	$(+, 0)$	$(\cdot, 1)$	Soma-Produto
$(-\infty, \infty]$	$(\min, \infty)$	$(+, 0)$	Mínimo-Soma
$[-\infty, 0)$	$(\max, -\infty)$	$(+, 0)$	Máximo-Soma

O domínio S de g pode receber a denominação de *espaço de configurações*. E cada elemento pertencente a S é conhecido como uma *configuração particular do conjunto de variáveis*, ou simplesmente, uma *configuração de variáveis*.

Considerando que $g(x_1, \ldots, x_N)$ pode ser fatorada como o produto de $|J|$ funções locais, com J sendo um conjunto discreto de índices, tem-se que

$$g(x_1, \ldots, x_N) = \prod_{j \in J} f_j(X_j) \tag{4-1}$$

onde X_j é um subconjunto de $\{x_1, \ldots, x_N\}$ e $f_j(X_j)$, uma função tendo elementos de X_j como argumentos.

A Eq. (4-1) pode ser representada por um grafo-fator cuja definição é apresentada a seguir [25].

Definição 4.2 (grafo-fator) Um grafo-fator $\mathcal{G}$ é um grafo bi-particionado que expressa a estrutura da fatoração em (4-1). Um grafo-fator possui vértices que representam as variáveis x_i, denominados de *vértices ou nós de variáveis*, e vértices que representam as funções f_j, denominados de *vértices ou nós de funções*. Há um ramo ligando um nó de variável a um determinado nó de função, se somente se, x_i é argumento de f_j. ■

Por exemplo, considere que $g(x_1, x_2, x_3, x_4, x_5)$ pode ser fatorada como

$$g(x_1, x_2, x_3, x_4, x_5) = f_1(x_1) f_2(x_2) f_3(x_1, x_2, x_3) f_4(x_3, x_4) f_5(x_3, x_5) \tag{4-2}$$

de modo que $J = \{1, 2, 3, 4, 5\}$, $X_1 = \{x_1\}$, $X_2 = \{x_2\}$, $X_3 = \{x_1, x_2, x_3\}$, $X_4 = \{x_3, x_4\}$ e $X_5 = \{x_3, x_5\}$, o grafo-fator para Eq. (4-2) é mostrado na Fig. 4.1.

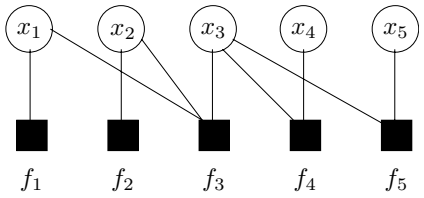

Figura 4.1: Grafo-fator para Eq. (4-2).

Note que o grafo-fator é portanto um ferramenta gráfica que ilustra a relação "é argumento de" que há entre funções locais e variáveis.

4.1.2 O Algoritmo Soma-Produto

O algoritmo Soma-Produto quando aplicado a um grafo-fator $\mathcal{G}$, sem ciclos, que representa a fatoração de uma função global $g(x_1, \ldots, x_N)$, calcula as funções marginais $g_i(x_i)$ exatas, definidas como

$$\begin{aligned} g_i(x_i) &= \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_N} g(x_1, \ldots, x_N) \\ &= \sum_{\sim\{x_i\}} g(x_1, \ldots, x_N) \end{aligned} \tag{4-3}$$

onde $\sum_{\sim\{x_i\}}$ foi denominado de *operador sumário* em [25], e designa de forma compacta os múltiplos somatórios ao longo de todos de os argumentos de g, exceto x_i.

O algoritmo Soma-Produto é dito um algoritmo de passagem de mensagens que opera em $\mathcal{G}$. A propagação de mensagens é regida pelos nós de $\mathcal{G}$ que são concebidos com a finalidade de receber as mensagens provenientes de seus respectivos nós vizinhos, atualizá-las, e transmiti-las. Nesse processo, os ramos agem como canais de comunicação, através do qual as mensagens são propagadas.

A propagação de mensagens realizada pelo algoritmo Soma-Produto deve obedecer ao seguinte princípio básico:

> A mensagem transmitida por um nó v para um nó u através do ramo $e(v, u)$, deve independer de qualquer mensagem proveniente de u.

As mensagens propagadas em $\mathcal{G}$ são funções das variáveis $\{x_1, \ldots, x_N\}$ e contém informações sobre as funções marginais $g_i(x_i)$. Considere que um ramo $e(x, f)$ conecta um nó de variável x a um nó de função f em $\mathcal{G}$, conforme ilustrado na Fig. 4.2.

As mensagens transmitidas através de $e(x, f)$ nos sentidos $(x \rightarrow f)$ e $(f \rightarrow x)$, são representadas pela notação $\mu_{x \rightarrow f}(x)$ e $\mu_{f \rightarrow x}(x)$, respectivamente. As mensagens transmitidas através ramos incidentes em x são funções $\mu(x)$, e podem ser especificadas através de um vetor cujos elementos são os valores que a função f assume em todo domínio de x.

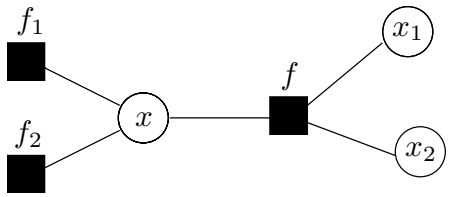

Figura 4.2: Propagação de mensagens entre os nós x e f em $\mathcal{G}$.

Por exemplo, se x é uma variável binária com domínio $A = \{0, 1\}$, tem-se que

$$\mu_{x \rightarrow f}(x) = \begin{pmatrix} \mu(0) \\ \mu(1) \end{pmatrix}. \tag{4-4}$$

Se $\mathcal{G}$ possui E ramos, a propagação de mensagens é realizada em $2E$ passos, visto que são propagadas duas mensagens por ramo, uma em cada sentido.

Considere que a notação $N(x)$ é introduzida para caracterizar o conjunto de nós vizinhos do nó x, e a notação $N(f)$ para representar o conjunto de nós vizinhos do nó f. Se f é vizinho de x, o conjunto que contém todos os vizinhos de x exceto f é representado pela notação $N(x) \backslash f$.

O algoritmo Soma-Produto para um grafo $\mathcal{G}$, sem ciclos, é definido em 3 etapas: uma de *inicialização*, na qual são atribuídos valores às mensagens transmitidas pelas folhas de $\mathcal{G}$ aos seus nós vizinhos; uma de *atualização*, onde todos os nós restantes de $\mathcal{G}$, recebem as mensagens provenientes de seus vizinhos, as atualizam, e as retransmitem aos respectivos vizinhos; e por fim uma etapa de *finalização*, na qual as funções marginais são calculadas. Estas etapas são definidas abaixo:

Algoritmo 4.1 (Soma-Produto)

1) **Inicialização:** A mensagem transmitida pelo nó de variável x ao nó

de função f com $f \in N(x)$, na inicialização, é dada por

$$\mu_{x\to f}(x) = 1 \tag{4-5}$$

enquanto que a mensagem enviada pelo nó de função f ao nó de variável x, $x \in N(f)$, é

$$\mu_{f\to x}(x) = f(x) \tag{4-6}$$

observe que este é o valor da função $f(\cdot)$ obtido quando o valor do argumento é x.

2) **Atualização:** A mensagem atualizada pelo nó x, a ser repassada ao nó vizinho f, é

$$\mu_{x\to f}(x) = \prod_{h\in N(x)\backslash\{f\}} \mu_{h\to x}(x) \tag{4-7}$$

e da mesma maneira a mensagem a ser repassada pelo nó f ao nó vizinho x é

$$\mu_{f\to x}(x) = \sum_{\sim\{x\}} \left(f(X) \prod_{y\in N(f)\backslash\{x\}} \mu_{y\to f}(y) \right). \tag{4-8}$$

Observe que $X = N(f)$ é o conjunto correspondente aos vizinhos (argumentos) de f.

3) **Finalização:** Depois que todos os nós do grafo-fator receberam mensagens provenientes de todos os seus nós vizinhos, a função marginal $\mu(x)$ para a variável x, pode ser obtida por

$$\mu(x) = \prod_{f\in N(x)} \mu_{f\to x}(x). \tag{4-9}$$

4) Fim. ■

Note que a mensagem que um nó de variável x_i pertencente a um grafo bi-particionado[1] transmite ao seu vizinho f_j, é obtida multiplicando-se todas as mensagens recebidas em x_i provenientes de todos os seus vizinhos $f_k \in N(x_i)\backslash f_j$. Já a mensagem que um nó de função f_j transmite a seu vizinho x_i é obtida multiplicando-se f_j por todas as mensagens recebidas em f_j, exceto a mensagem proveniente de x_i, e então realiza-se o somatório

[1] O grafo-fator é um grafo bi-particionado, e por definição os vizinhos de um nó de variável x_i serão todos nós de funções $f_j \in N(x_i)$.

($\sum_{\sim\{x_i\}}$) como indicado em (4-8). Por fim, a função marginal $\mu(x_i)$, para a variável x_i, é obtida multiplicando-se todas as mensagens recebidas no nó x_i.

Foi provado em [25], que a função marginal $\mu(x_i)$ obtida pelo algoritmo Soma-Produto em $\mathcal{G}$ é exatamente a função $g_i(x_i)$. O seguinte teorema encerra este comentário.

Teorema 4.1 - Em um grafo-fator finito, sem ciclos, que representa a fatoração da função $g(x_1, \ldots, x_N)$, a função $\mu(x_i)$ calculada pelo algoritmo Soma-Produto de acordo com Eq. (4-9) é a função marginal $g_i(x_i)$. ■

Exemplo:

Considere o grafo da Fig. 4.3, que representa a fatoração da função $g(x_1, x_2, x_3, x_4, x_5, x_6)$, dada por

$$g(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_4) f_2(x_3, x_4) f_3(x_4, x_5, x_6) f_4(x_5). \quad (4\text{-}10)$$

A seguir a obtenção da função marginal $g_4(x_4)$, usando o Algoritmo 4.1, é detalhada.

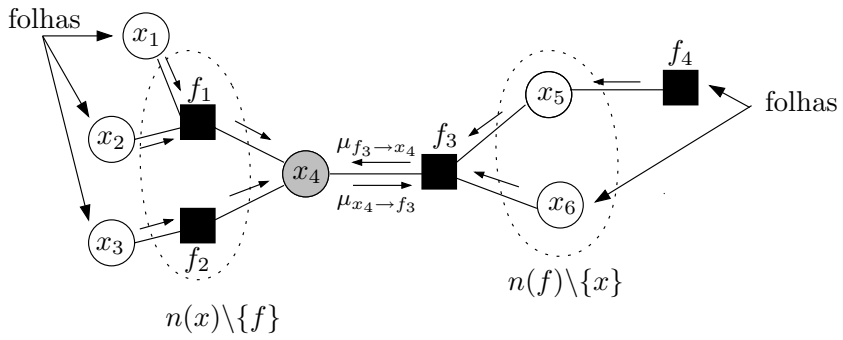

Figura 4.3: Grafo-fator genérico.

Tem-se:

Inicialização:

$$
\begin{aligned}
&x_1 : \mu_{x_1 \to f_1}(x_1) = 1 \\
&x_2 : \mu_{x_2 \to f_1}(x_2) = 1 \\
&x_3 : \mu_{x_3 \to f_2}(x_3) = 1 \\
&x_4 : \text{nó interno} \\
&x_5 : \text{nó interno} \\
&x_6 : \mu_{x_6 \to f_3}(x_6) = 1 \\
&f_1 : \text{nó interno} \\
&f_2 : \text{nó interno} \\
&f_3 : \text{nó interno} \\
&f_4 : \mu_{f_4 \to x_5}(x_5) = f_4(x_5)
\end{aligned}
\tag{4-11}
$$

Atualização:

$$
\begin{aligned}
&x_1 : \text{nó folha} \\
&x_2 : \text{nó folha} \\
&x_3 : \text{nó folha} \\
&x_4 : \mu_{x_4 \to f_3}(x_4) = \mu_{f_1 \to x_4}(x_4) \cdot \mu_{f_2 \to x_4}(x_4) \\
&\qquad\qquad = \sum_{\sim\{x_4\}} f_1(x_1, x_2, x_4) f_2(x_3, x_4) \\
&x_5 : \mu_{x_5 \to f_3}(x_5) = f_4(x_5) \\
&x_6 : \text{nó folha} \\
&f_1 : \mu_{f_1 \to x_4}(x_4) = \sum_{\sim\{x_4\}} f_1(x_1, x_2, x_4)(\mu_{x_1 \to f_1}(x_1) \cdot \mu_{x_2 \to f_1}(x_2)) \\
&\qquad\qquad = \sum_{\sim\{x_4\}} f_1(x_1, x_2, x_4) \\
&f_2 : \mu_{f_2 \to x_4}(x_4) = \sum_{\sim\{x_4\}} f_2(x_3, x_4) \cdot \mu_{x_3 \to f_2}(x_3) \\
&\qquad\qquad = \sum_{\sim\{x_4\}} f_2(x_3, x_4) \\
&f_3 : \mu_{f_3 \to x_4}(x_4) = \sum_{\sim\{x_4\}} f_3(x_4, x_5, x_6)(\mu_{x_5 \to f_3}(x_5) \cdot \mu_{x_6 \to f_3}(x_6)) \\
&\qquad\qquad = \sum_{\sim\{x_4\}} f_3(x_4, x_5, x_6) f_4(x_5) \\
&f_4 : \text{nó folha}
\end{aligned}
\tag{4-12}
$$

Finalização:

$$g_4(x_4) = \mu_{f_1 \to x_4}(x_4) \cdot \mu_{f_2 \to x_4}(x_4) \cdot \mu_{f_3 \to x_4}(x_4) \tag{4-13}$$
$$= \sum_{\sim\{x_4\}} f_1(x_1, x_2, x_4) f_2(x_3, x_4) f_3(x_4, x_5, x_6) f_4(x_5).$$

Note que as mensagens obtidas acima são calculadas uma única vez, e serão reutilizadas nos cálculos de $g_1(x_1)$, $g_2(x_2)$, $g_3(x_3)$, $g_5(x_5)$ e $g_6(x_6)$. A reutilização de cálculos intermediários é a razão pela qual o algoritmo Soma-Produto calcula funções marginais de maneira eficiente. ■

4.2 Modelagem de Sistemas Utilizando Grafos-Fatores

Considere um sistema como sendo um conjunto de variáveis que interagem entre si. Grafos-fatores podem ser utilizados para modelar sistemas, sendo o modelo de um sistema definido por uma função global fatorável cujos argumentos são as variáveis do sistema.

Sistemas de comunicação que utilizam codificação de canal possuem um grafo-fator que embute em sua estrutura dois modelos combinados: um *comportamental* e outro *probabilístico*.

De maneira qualitativa, um *modelo comportamental* especifica possíveis "comportamentos" que um sistema pode assumir. Isso é realizado através de uma função global denominada de *função indicadora*, que testa quais configurações de variáveis são consideradas válidas para o sistema. A fatoração da função indicadora fornece informações sobre a estrutura do sistema.

No *modelo probabilístico*, o grafo-fator representa a fatoração de uma *função densidade de probabilidade conjunta* das variáveis pertencentes ao sistema. Esta fatoração ilustra a dependência estatística que há entre as variáveis do sistema.

4.2.1 Modelo Comportamental

Seja $\{x_1, x_2, \ldots, x_N\}$ um conjunto de variáveis com espaço de configuração $S = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_N$. Um comportamento em S é definido como qualquer subconjunto B de S. Os elementos de B são considerados como configurações de variáveis válidas.

A função indicadora que determina B é definida como

$$\mathcal{X}_B(x_1,\ldots,x_N) \triangleq \begin{cases} 1, & \text{se } (x_1,\ldots,x_N) \in B \\ 0, & \text{caso contrário} \end{cases} . \tag{4-14}$$

Geralmente a função indicadora $\mathcal{X}_B$ é fatorável, o que significa dizer que o comportamento B pode ser decomposto em vários comportamentos locais. Um comportamento local consiste em um teste que envolve um determinado subconjunto das variáveis pertencentes ao sistema. Neste caso, uma configuração de variáveis é elemento de B e portanto considerada válida, se somente se satisfizer todos os comportamentos locais que compõem B, ou seja, se passar em todos os testes.

Note que sendo $\mathcal{X}_B$ fatorável, esse sistema pode ter seu modelo comportamental representado por um grafo-fator.

Aplicando esse conceito à codificação de canal, e considerando que o domínio de cada variável seja um alfabeto finito A, com espaço de configuração $S = A^N$, tem-se que um comportamento $\mathcal{C} \subset S$ resulta em um código de bloco de tamanho N, sendo cada elemento $c \in \mathcal{C}$ correspondente a uma palavra código. Sua função $\mathcal{X}_\mathcal{C}$ é fatorada em funções locais que estão relacionadas a testes de paridade obtidos através da matriz de paridade do código.

No caso de um código convolucional $\mathcal{C}$ cuja treliça é $\mathcal{T}$, o comportamento $\mathcal{C}$ está associado ao espaço de configurações das variáveis $\{\sigma_0,\ldots,\sigma_N,\boldsymbol{m}_1,\ldots,\boldsymbol{m}_N,\boldsymbol{c}_1,\ldots,\boldsymbol{c}_N\}$[2] e pode ser decomposto em N comportamentos locais, um para cada seção T_t de $\mathcal{T}$, com $t = 1,\ldots,N$.

Os comportamentos locais, neste caso, são responsáveis por testar se determinada configuração de variáveis $(\sigma_{t-1},\boldsymbol{m}_t,\boldsymbol{c}_t,\sigma_t)$ corresponde a um ramo válido na seção T_t da treliça.

A função indicadora $\mathcal{X}_\mathcal{C}$ que define $\mathcal{C}$, é fatorada em N funções locais f_t cujos argumentos são σ_{t-1}, $\boldsymbol{m}_t$, $\boldsymbol{c}_t$ e σ_t. Tem-se então que

$$\mathcal{X}_\mathcal{C} = \prod_{t=1}^{N} f_t(\sigma_{t-1},\boldsymbol{m}_t,\boldsymbol{c}_t,\sigma_t) \tag{4-15}$$

onde

$$f_t(\sigma_{t-1},\boldsymbol{m}_t,\boldsymbol{c}_t,\sigma_t) \triangleq \begin{cases} 1, \text{ se } (\sigma_{t-1},\boldsymbol{m}_t,\boldsymbol{c}_t,\sigma_t) \in T_t \\ 0, \text{ caso contrário.} \end{cases} \tag{4-16}$$

[2]Vide notação na Seção 2.1.2.

Por exemplo, no caso da treliça da Fig. 2.4, a função $f_t(\sigma_{t-1}, m_t, \boldsymbol{c}_t, \sigma_t) = 1$ para as configurações de variáveis: $(00, 0, 00, 00)$, $(00, 1, 11, 10)$, $(01, 0, 11, 00)$, $(01, 1, 00, 10)$, $(10, 0, 01, 01)$, $(10, 1, 10, 11)$, $(11, 0, 10, 01)$ e $(11, 1, 01, 11)$.

O grafo-fator que representa a Eq. (4-15) é visto na Fig. 4.4.

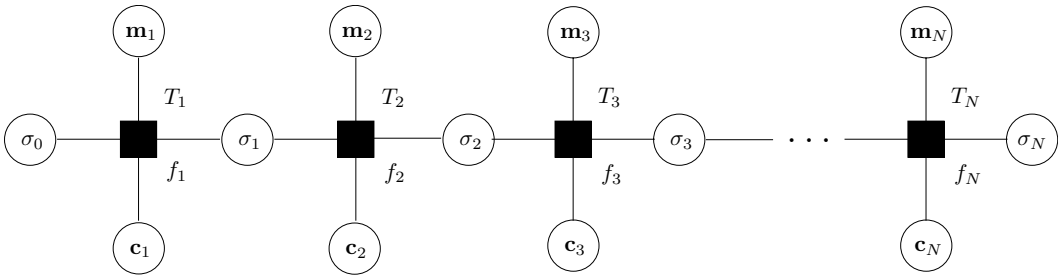

Figura 4.4: Grafo-fator correspondente a um código convolucional.

4.2.2 Modelo Probabilístico

Considere que a seqüência codificada $\boldsymbol{c} = (\ \boldsymbol{c}_1 \ \cdots \ \boldsymbol{c}_N\ )$, gerada por um código convolucional $\mathcal{C}$, foi transmitida através de um canal ruidoso, e seja $\boldsymbol{r} = (\ \boldsymbol{r}_1 \ \cdots \ \boldsymbol{r}_N\ )$ a seqüência observada. Para uma seqüência $\boldsymbol{r}$ fixa, a probabilidade a posteriori $P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}|\boldsymbol{r})$ é proporcional à função densidade de probabilidade conjunta $p(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}, \boldsymbol{r})$. Seja

$$\begin{aligned} g(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}) &\triangleq p(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}, \boldsymbol{r}) \\ &= p(\boldsymbol{r}|\boldsymbol{\sigma}, \boldsymbol{m}, \boldsymbol{c}) P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}) \\ &= p(\boldsymbol{r}|\boldsymbol{c}) P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}) \end{aligned} \tag{4-17}$$

onde $p(\boldsymbol{r}|\boldsymbol{c})$ é a função verossimilhança de $\boldsymbol{c}$ e $P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c})$ é a probabilidade conjunta entre a seqüência de estados $\boldsymbol{\sigma}$, a seqüência de entrada $\boldsymbol{m}$ e $\boldsymbol{c}$. Note que $\boldsymbol{r}$ é considerado parâmetro e não argumento de g, pois foi fixada.

A probabilidade $P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c})$ pode ser escrita como[3]

$$P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}) = P(\sigma_0) \prod_{t=1}^{N} P(\boldsymbol{m}_t) \prod_{t=1}^{N} f_t(\sigma_{t-1}, \boldsymbol{m}_t, \boldsymbol{c}_t, \sigma_t). \tag{4-18}$$

Em um canal sem memória, a função verossimilhança de $\boldsymbol{c}$ pode ainda ser fatorada como

$$p(\boldsymbol{r}|\boldsymbol{c}) = \prod_{t=1}^{N} p(\boldsymbol{r}_t|\boldsymbol{c}_t). \tag{4-19}$$

[3]O desenvolvimento da Eq. (4-18) é apresentado no Apêndice A.

A função $g(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c})$ em 4-17 é expressa então por

$$g(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}) = P(\sigma_0) \prod_{t=1}^{N} P(\boldsymbol{m}_t) \prod_{t=1}^{N} f_t(\sigma_{t-1}, \boldsymbol{m}_t, \boldsymbol{c}_t, \sigma_t) \prod_{t=1}^{N} p(\boldsymbol{r}_t|\boldsymbol{c}_t). \quad (4\text{-}20)$$

onde $f_t(\sigma_{t-1}, \boldsymbol{m}_t, \boldsymbol{c}_t, \sigma_t)$ foi definida na Eq. (4-16).

O grafo-fator que representa a fatoração da Eq. (4-20) é ilustrado na Fig. 4.5 e tem estrutura similar ao da Fig. 4.4 — a diferença consiste apenas em se ter os nós de funções $p(\boldsymbol{r}_t|\boldsymbol{c}_t)$, $P(\boldsymbol{m}_t)$ e $P(\sigma_0)$, que são anexados devido a inclusão do modelo probabilístico no grafo-fator que representa o código $\mathcal{C}$.

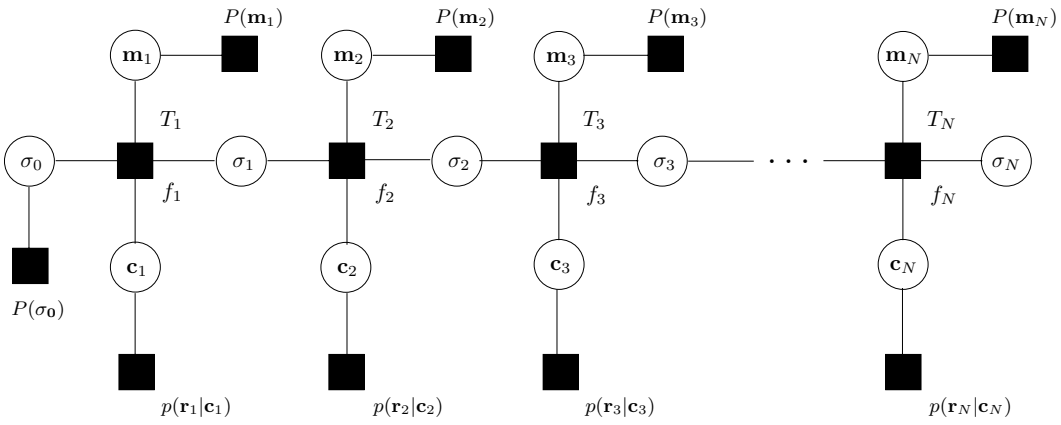

Figura 4.5: Grafo-fator correspondente a um código convolucional, incluindo os modelos comportamental e probabilístico.

Note que os nós de variáveis correspondentes a $\boldsymbol{r}_t$, $t = 1, \ldots, N$, foram omitidos pois $\boldsymbol{r}_t$, está fixada para $\forall t$, sendo desnecessária sua representação no grafo-fator da Fig. 4.5.

Caso o codificador seja terminado, o estado final σ_N e as entradas $\boldsymbol{m}_t$ para $t = N-\nu, \ldots, N$, correspondentes aos bits de cauda, são determinados. A probabilidade conjunta é fatorada agora como

$$P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}) = P(\sigma_0)P(\sigma_N) \prod_{t=1}^{N-\nu} P(\boldsymbol{m}_t) \prod_{t=1}^{N} f_t(\sigma_{t-1}, \boldsymbol{m}_t, \boldsymbol{c}_t, \sigma_t)^4 \quad (4\text{-}21)$$

e dá origem a um grafo-fator semelhante ao apresentado na Fig. 4.5, anexando-se apenas um nó de função $P(\sigma_N)$ ao estado final σ_N, e suprimindo-se os nós de variáveis $P(m_t)$ correspondentes aos bits de cauda.

Na codificação turbo, deseja-se encontrar o grafo-fator correspondente a um código convolucional sistemático $\mathcal{C} = (2, 1, K)$, terminado. O grafo-

[4] Vide Apêndice A.

fator para este código é facilmente obtido através do grafo da Fig. 4.5, estando ilustrado na Fig. 4.6.

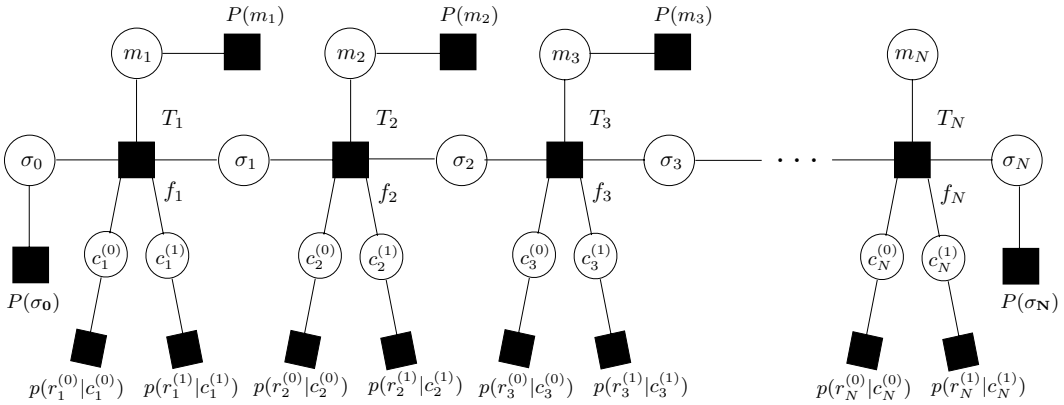

Figura 4.6: Grafo-fator correspondente a um código convolucional sistemático $(2, 1, K)$.

Como $\mathcal{C}$ é sistemático, tem-se que $c_t^{(0)} = m_t$, sendo assim redundante a representação dos nós das variáveis $c_t^{(0)}$, $\forall t$. Dessa forma o grafo-fator da Fig. 4.6 pode ser simplificado, resultando no grafo da Fig. 4.7.

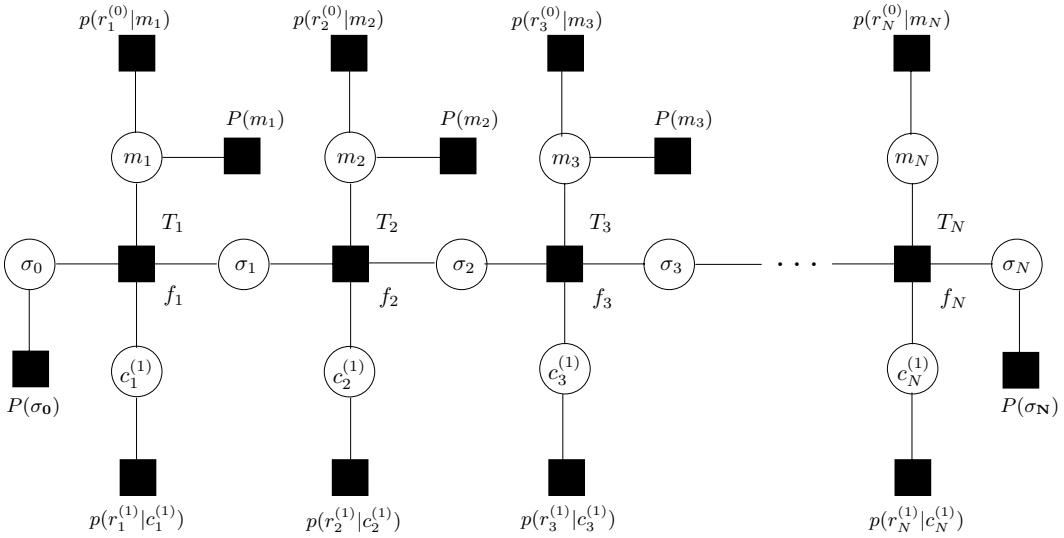

Figura 4.7: Grafo-fator simplificado correspondente a um código convolucional sistemático $\mathcal{C} = (2, 1, K)$.

4.3 O Algoritmo MAP

O algoritmo Soma-Produto, quando aplicado ao grafo-fator que representa um código convolucional $\mathcal{C}$, resulta no algoritmo MAP da Seção 3.1.

Diz-se por essa razão que o algoritmo MAP pode ser visto como uma instância do algoritmo Soma-Produto [25].

O problema da decodificação utilizando o algoritmo MAP, como dito anteriormente, consiste em encontrar as máximas probabilidades *a posteriori* marginais $P(m_t|\boldsymbol{r})$ para cada símbolo transmitido m_t, $t = 1, \ldots, N$, a partir da probabilidade conjunta $P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}|\boldsymbol{r})$.

Como $P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}|\boldsymbol{r}) = P(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c}|\boldsymbol{r})/p(\boldsymbol{r}) \propto g(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c})$, as probabilidades marginais $P(m_t|\boldsymbol{r})$ também são proporcionais às funções marginais $g_t(m_t)$. As funções $g_t(m_t)$ são obtidas através do algoritmo Soma-Produto, a partir do grafo-fator que representa o código.

Considerando que $\mathcal{C} = (2, 1, K)$ seja um código convolucional sistemático recursivo, tem-se para este código: o grafo-fator $\mathcal{G}$ ilustrado na Fig. 4.7, e a função $g(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{c})$ definida em (4-21).

As mensagens propagadas pelo algoritmo Soma-Produto para obtenção das funções marginais g_t são obtidas através dos passos descritos no Algoritmo 4.1. Em resumo tem-se

1) A transmissão de mensagens é iniciada pelas folhas, vide Fig. 4.7, que transmitem[5] $P(m_t)$, $p(r_t^{(0)}|m_t)$, $p(r_t^{(1)}|c_t)$, $P(\sigma_0)$ e $P(\sigma_N)$ como mensagens para os nós m_t, c_t, σ_0 e σ_N, respectivamente.
2) As funções $p(r_t^{(0)}|m_t)$ e $p(r_t^{(1)}|c_t)$ são obtidas através do modelamento do canal. Caso o codificador seja iniciado e terminado no estado zero[6], ou seja, $\sigma_0 = \sigma_N = 0$, tem-se que $P(\sigma_0 = 0) = P(\sigma_N = 0) = 1$. Assume-se que os bits m_t são equiprováveis, resultando em $P(m_t = 0) = P(m_t = 1) = 0.5$, $\forall t$.
3) Os nós de variáveis σ_0 e σ_N por serem vértices de grau 2, apenas repassam as mensagens recebidas para os nós f_1 e f_N.
4) O mesmo vale para os nós c_t, $\forall t$.
5) Os nós m_t transmitem as mensagens $p(r_t^{(0)}|m_t)P(m_t)$ para os nós f_t, obtidas multiplicando-se todas as mensagens recebidas em m_t, excetuando-se a mensagem proveniente do ramo $e(m_t, f_t)$.
6) Os nós f_1 e f_N após terem recebido as mensagens provenientes de seus vizinhos (σ_0, m_1, c_1) e (σ_N, m_N, c_N), estão aptos a transmitirem mensagens para os nós de variáveis σ_1 e σ_{N-1}, inciando uma propagação de mensagens em dois sentidos: crescente e decrescente de t. As mensa-

[5] A fim de simplificar a notação optou-se por representar $c_t^{(1)}$ apenas por c_t.

[6] Caso o codificador não seja terminado, as variáveis aleatórias σ_0 e σ_N são uniformemente distribuídas.

gens transmitidas no sentido crescente são denominadas na literatura por α, enquanto que no sentido decrescente, por β.

7) A propagação de mensagens prossegue até que tenham sido transmitidas duas mensagens em todos os ramos $e(\sigma_t, f_t)$.

As métricas α e β a serem calculadas são obtidas aplicando-se as equações (4-7) e (4-8) do Algoritmo 4.1, e desta forma tem-se

$$\alpha(\sigma_t) = \sum_{\sim\{\sigma_t\}} f_t(\sigma_{t-1}, m_t, c_t, \sigma_t)\alpha(\sigma_{t-1})p(r_t^{(0)}|m_t)p(r_t^{(1)}|c_t)P(m_t) \quad (4\text{-}22)$$

$$\beta(\sigma_{t-1}) = \sum_{\sim\{\sigma_{t-1}\}} f_t(\sigma_{t-1}, m_t, c_t, \sigma_t)\beta(\sigma_t)p(r_t^{(0)}|m_t)p(r_t^{(1)}|c_t)P(m_t)$$

Denominando o fator $p(r_t^{(0)}|m_t)p(r_t^{(1)}|c_t)P(m_t)$ de $\gamma(\sigma_{t-1}, \sigma_t)$, como foi feito em (3-33) no Capítulo 3, e considerando $e(\sigma_{t-1}, m_t, c_t, \sigma_t)$ o ramo na qual $f_t(e) = 1$. Para cada ramo e, tem-se que $\alpha(\sigma_{t-1}) = \alpha(e)$, $\beta(\sigma_t) = \beta(e)$ e $\gamma(\sigma_{t-1}, \sigma_t) = \gamma(e)$. Utilizando a notação $E_t(\sigma_t)$ para caracterizar o conjunto dos ramos que incidem no estado σ_t pertencente à seção T_t da treliça, permite que α e β sejam reescritas

$$\alpha(\sigma_t) = \sum_{e \in E_t(\sigma_t)} \alpha(e)\gamma(e) \quad (4\text{-}23)$$

$$\beta(\sigma_{t-1}) = \sum_{e \in E_t(\sigma_{t-1})} \beta(e)\gamma(e)$$

que corresponde às equações (3-20) e (3-22), obtidas na Seção 3.1. Na Fig. 4.8 está ilustrada a propagação de mensagens realizada pelo algoritmo MAP na seção T_t de $\mathcal{G}$.

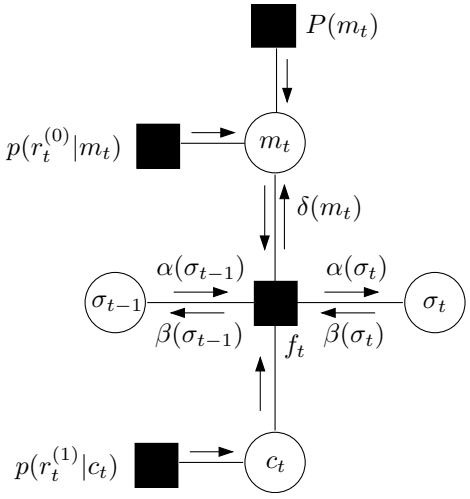

Figura 4.8: Propagação de mensagens na seção T_t de $\mathcal{G}$.

Dado que o grafo-fator que representa a treliça de um codificador convolucional é um grafo sem ciclos, o algoritmo Soma-Produto obtém as funções marginais exatas para cada variável m_t, lembrando que as funções marginais são proporcionais às probabilidade *a posteriori* $P(m_t|\boldsymbol{r})$.

Por fim, as marginais $g_t(m_t)$ são obtidas multiplicando-se todas as mensagens recebidas pelos nós m_t

$$g_t(m_t) = p(r_t^{(0)}|m_t)P(m_t)\delta\left(m_t\right) \tag{4-24}$$

onde

$$\delta(m_t) = \sum_{\sim\{m_t\}} f_t(\sigma_{t-1}, m_t, c_t, \sigma_t)\alpha(\sigma_{t-1})p(r_t^{(1)}|c_t)\beta(\sigma_t). \tag{4-25}$$

Considerando que a notação $F_t(m_t)$ representa o conjunto de ramos associados à entrada m_t pertencentes à seção T_t da treliça, tem-se

$$\delta\left(m_t\right) = \sum_{e\in F_t(m_t)} \alpha\left(e\right)p(r_t^{(0)}|m_t)\beta\left(e\right) \tag{4-26}$$

Sendo $P_{\text{int},t}(m_t) = p(r_t^{(0)}|m_t)$, $P_{\text{pri},t}(m_t) = P(m_t)$ e $P_{\text{ext},t}(m_t) = \delta(m_t)$, tem-se que

$$g_t(m_t) = P_{\text{int},t}(m_t) \cdot P_{\text{pri},t}(m_t) \cdot P_{\text{ext},t}(m_t) \tag{4-27}$$

que corresponde à Eq. (3-35) obtida na Seção 3.1.1, comprovando que o algoritmo Soma-Produto quando aplicado ao grafo-fator referente a um codificador convolucional, resulta no algoritmo MAP.

4.3.1 Os Algortimos Max-Log-MAP e Log-MAP

Apesar de sua excelente performance, o algoritmo MAP é um algoritmo de grande complexidade se comparado, por exemplo, ao algoritmo de Viterbi. Por essa razão há um grande interesse em se utilizar algoritmos de complexidade inferior a do algoritmo MAP, mesmo que isto implique em perda no seu desempenho. Em [7], foram propostos como alternativas de menor complexidade com perda de desempenho admissível em relação ao algoritmo MAP, os algoritmos Max-Log-MAP e Log-MAP, ambos versões do algoritmo MAP no domínio logarítmico.

O algoritmo MAP pode ser enunciado, no domínio logarítmico, utilizando-se o logaritmo de certas quantidades. Considere a seguinte notação:

$$\begin{aligned}\bar{P}(m_t) &= \ln(P(m_t)) \\ \bar{p}(r_t^{(0)}|m_t) &= \ln(p(r_t^{(0)}|m_t)) \\ \bar{p}(r_t^{(1)}|c_t) &= \ln(p(r_t^{(1)}|c_t)) \\ \bar{\alpha}(\sigma_{t-1}) &= \ln(\alpha(\sigma_{t-1})) \\ \bar{\beta}(\sigma_t) &= \ln(\beta(\sigma_t)).\end{aligned} \tag{4-28}$$

Para calcular recursivamente $\bar{\alpha}$ e $\bar{\beta}$, aplica-se o logaritmo neperiano a ambos os lados das equações em (4-23), e obtém-se

$$\bar{\alpha}\left(\sigma_t\right) = \ln\left(\sum_{e\in E_t(\sigma_t)} \alpha\left(e\right)\gamma\left(e\right)\right) \tag{4-29}$$

e

$$\bar{\beta}\left(\sigma_{t-1}\right) = \ln\left(\sum_{e\in E_t(\sigma_{t-1})} \beta\left(e\right)\gamma\left(e\right)\right). \tag{4-30}$$

Através das relações definidas em 4.3.1, tem-se que

$$\begin{aligned}\alpha\left(e\right) &= e^{\bar{\alpha}(e)} \\ \beta\left(e\right) &= e^{\bar{\beta}(e)} \\ \gamma\left(e\right) &= e^{\overline{P}(m_t)} e^{\overline{p}(r_t^{(0)}|m_t)} e^{\overline{p}(r_t^{(1)}|c_t)} \\ \bar{\gamma}\left(e\right) &= \ln(\gamma\left(e\right)) = \bar{P}(m_t) + \bar{p}(r_t^{(0)}|m_t) + \bar{p}(r_t^{(1)}|c_t)\end{aligned} \tag{4-31}$$

assim (4-29) e (4-30) podem então ser reescritas como

$$\begin{aligned}\bar{\alpha}\left(\sigma_t\right) &= \ln\left(\sum_{e\in E_t(\sigma_t)} \alpha\left(e\right)\gamma\left(e\right)\right) = \ln\left(\sum_{e\in E_t(\sigma_t)} e^{\bar{\alpha}(e)} e^{\bar{\gamma}(e)}\right) \\ &= \ln\left(\sum_{e\in E_t(\sigma_t)} e^{(\bar{\alpha}(e)+\bar{\gamma}(e))}\right) \\ \bar{\beta}\left(\sigma_{t-1}\right) &= \ln\left(\sum_{e\in E_t(\sigma_{t-1})} \beta\left(e\right)\gamma\left(e\right)\right) = \ln\left(\sum_{e\in E_t(\sigma_{t-1})} e^{\bar{\beta}(e)} e^{\bar{\gamma}(e)}\right) \\ &= \ln\left(\sum_{e\in E_t(\sigma_{t-1})} e^{(\bar{\beta}(e)+\bar{\gamma}(e))}\right).\end{aligned} \tag{4-32}$$

Duas variações do algoritmo MAP denominadas de algoritmo Log-

MAP e algoritmo Max-Log-MAP são obtidas a partir de como a expressão

$$\ln\left(e^{x_1}+\cdots+e^{x_n}\right) \tag{4-33}$$

é solucionada. O algoritmo Max-Log-MAP é obtido quando em (4-33) a aproximação

$$\ln\left(e^{x_1}+\cdots+e^{x_n}\right) \approx \max_t\left(x_t\right) \tag{4-34}$$

é utilizada. Enquanto que o algoritmo Log-MAP é obtido quando a definição de logaritmo Jacobiano

$$\ln(e^{x_1}+e^{x_2}) = \max\left(x_1,x_2\right) + \ln\left(1+e^{-|x_1-x_2|}\right) \tag{4-35}$$

é usada.

Note que a diferença entre as equações Eq. (4-35) e (4-34) é a função de correção $f_c(\Delta) = \ln\left(1+e^{-\Delta}\right)$, onde $\Delta = |x_1 - x_2|$. Esta função que proporciona o aumento na complexidade[7] do algoritmo Log-MAP em relação ao algoritmo Max-Log-MAP, e é também o que garante ao algoritmo o mesmo desempenho que o do algoritmo MAP.

Foi mostrado em [7], que $f_c(\Delta)$ não precisa ser calculada repetidamente, podendo ser armazenada em um tabela pré-calculada, o que diminui a complexidade do algoritmo Log-MAP. A tabela pré-computada é dada abaixo

Tabela 4.2: Tabela para função de correção $f_c(\Delta)$.

$\Delta \in [a,b]$	$f_c(\Delta)$
$[0, 0.625)$	0.54905486
$[0.625, 1.25)$	0.33045821
$[1.25, 1.875)$	0.19029914
$[1.875, 2.5)$	0.10633724
$[2.5, 3.125)$	0.05832048
$[3.125, 3.75)$	0.03163911
$[3.75, 4.375)$	0.01705960
$[4.375, \infty]$	0.00916753

Considere a função

$$g(x_1,x_2) = \max\left(x_1,x_2\right) + \ln\left(1+e^{-|x_1-x_2|}\right) \tag{4-36}$$

[7] A complexidade de cada algoritmo pode ser vista na Seção 5.3.

e defina

$$\max_t *(x_1, \ldots, x_N) \triangleq g(x_N, g(x_{N-1}, \ldots, g(x_3, g(x_2, x_1)))). \tag{4-37}$$

Como foi definido na Seção 4.1.1, é necessário que o contradomínio R da função g cuja fatoração é representada por um grafo-fator $\mathcal{G}$, seja tal que $\langle R, \max, + \rangle$, ou $\langle R, \max *, + \rangle$, seja um semi-anel.

A razão para esta necessidade está no fato de que os cálculos efetuados pelo algoritmo Soma-Produto através de $\mathcal{G}$, fundamenta-se na lei distributiva e sendo $\langle R, +, \cdot \rangle$ um semi-anel, garante-se que este possui duas operações, "+" e "$\cdot$", bem definidas, que satisfazem a lei distributiva.

É fácil ver que o conjunto $R = [-\infty, 0]$ associado às operações "+" e "max" resulta em um semi-anel[8], satisfazendo à lei

$$\forall a, b, c \in R, \quad a + \max(b, c) = \max(a + b, a + c) \tag{4-38}$$

o mesmo é válido quando R é associado às operações "+" e "max $*$".

Dado o anel $\langle R, \max, + \rangle$, as equações em (4-32), são obtidas pelo algoritmo Max-Log-MAP como

$$\begin{aligned} \bar{\alpha}(\sigma_t) &= \max_{e \in E_t(\sigma_t)} (\bar{\alpha}(e) + \bar{\gamma}(e)) \\ \bar{\beta}(\sigma_{t-1}) &= \max_{e \in E_t(\sigma_{t-1})} (\bar{\beta}(e) + \bar{\gamma}(e)) \end{aligned} \tag{4-39}$$

e pela Eq. (4-26), tem-se que

$$\begin{aligned} \bar{\delta}(m_t) &= \ln(\delta(m_t)) = \ln\left(\sum_{e \in F_t(m_t)} e^{\bar{\alpha}(e) + \bar{\beta}(e) + \bar{p}(r_t^{(1)}|c_t)} \right) \\ \bar{\delta}\,(m_t) &= \max_{e \in F_t(m_t)} \left(\bar{\alpha}\,(e) + \bar{\beta}\,(e) + \bar{p}(r_t^{(1)}|c_t) \right) \end{aligned} \tag{4-40}$$

e

$$\begin{aligned} \bar{g}_t(m_t) &= \bar{p}(r_t^{(0)}|m_t) + \bar{P}(m_t) + \bar{\delta}(m_t) \\ &= \bar{P}_{\text{int},t}(m_t) + \bar{P}_{\text{pri},t}(m_t) + \bar{P}_{\text{ext},t}(m_t). \end{aligned} \tag{4-41}$$

[8]Note que o conjunto R associado às operações $+$ e max resulta no semi-anel máximo-soma definido na Tabela 4.1.

Na Fig. 4.9 está ilustrada a propagação de mensagens realizada pelos algoritmos Log-MAP e Max-Log-MAP.

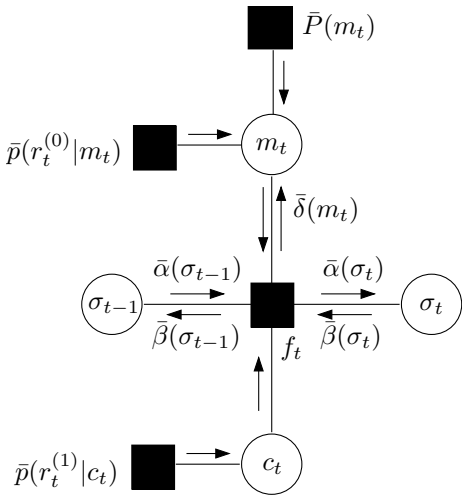

Figura 4.9: Propagação de mensagens no domínio logarítmico na seção T_t de $\mathcal{G}$.

O algoritmo Log-MAP é obtido através do mesmo desenvolvimento, trocando-se apenas a operação "max" pela operação "max$*$", o semi-anel utilizado no algoritmo Log-MAP é definido como $\langle R, \max*, + \rangle$.

Conclui-se que trabalhar no domínio logaritmo é trabalhar nos semi-anéis $\langle R, \max, + \rangle$ para o algoritmo Max-Log-MAP e $\langle R, \max*, + \rangle$ para o algoritmo Log-MAP.

4.4 Decodificação Turbo e sua Interpretação em Grafos-Fatores

O grafo-fator $\mathcal{G}$ de um código turbo $\mathcal{C}$ representa a fatoração de uma função global[9] $g(\boldsymbol{m}, \tilde{\boldsymbol{m}}, \boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}, \boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$ proporcional à função densidade de probabilidade conjunta $p(\boldsymbol{m}, \tilde{\boldsymbol{m}}, \boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}, \boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)})$, e resulta em um grafo composto pelos grafos fatores $\mathcal{G}_1$ e $\mathcal{G}_2$, correspondentes ao primeiro e segundo codificadores, respectivamente, interligados através de um entrelaçador Π, conforme é ilustrado na Fig.4.10.

Note que em $\mathcal{G}$, os registradores de ambos os codificadores são inicializados, usualmente no estado zero, $\sigma_0^{(1)} = 0$ e $\sigma_0^{(2)} = 0$, com apenas o primeiro codificador sendo terminado, pois devido ao entrelaçador Π, os bits de cauda utilizados para terminar o primeiro codificador, depois de entrelaçados, com

[9]Nesta função, $\boldsymbol{\sigma}^{(1)}$ e $\boldsymbol{\sigma}^{(2)}$ correspondem às seqüências de estados descritas pelos registradores dos codificadores 1 e 2, respectivamente.

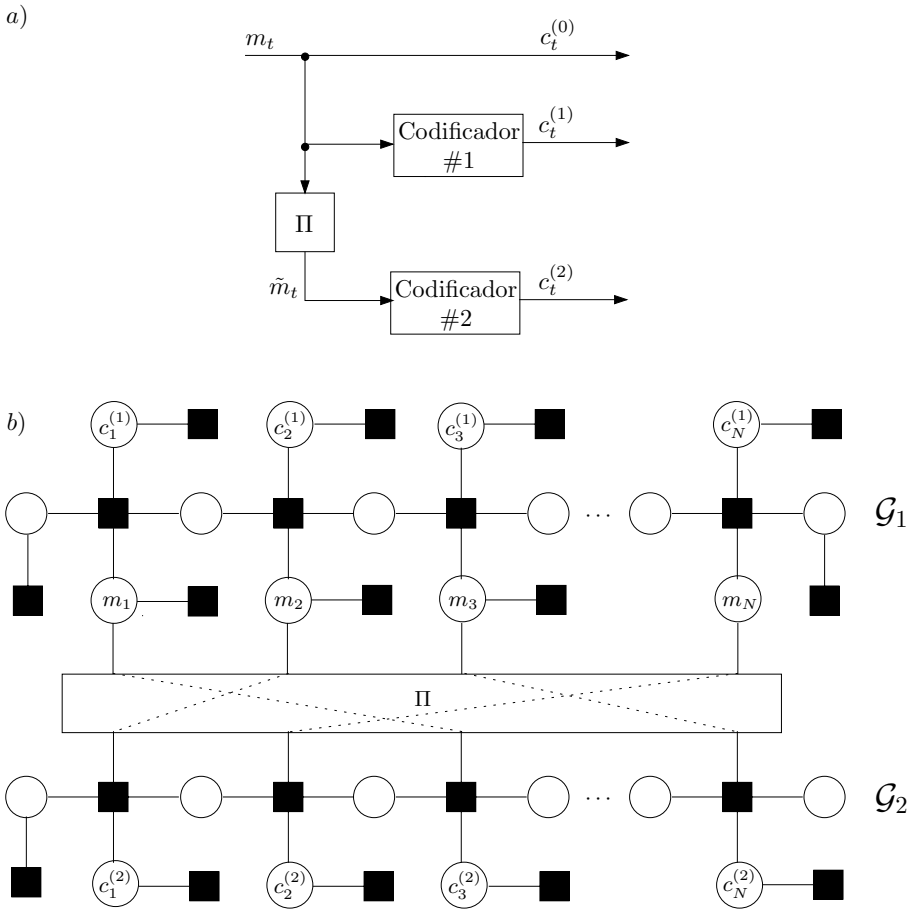

Figura 4.10: a) Codificador turbo $\mathcal{C}$; b) Grafo-fator $\mathcal{G}$, correspondente a $\mathcal{C}$.

alta probabilidade não correspondem ao bits de cauda necessários para terminar o segundo codificador, que por isso não é terminado[10].

Ainda que os grafos-fatores $\mathcal{G}_1$ e $\mathcal{G}_2$ sejam árvores[11], quando os mesmos são interligados para compor $\mathcal{G}$, resultam em um grafo-fator com ciclos. Quando o algoritmo Soma-Produto é aplicado a $\mathcal{G}$, devido a presença dos ciclos, os seguintes comentários se aplicam:

- O algoritmo obtém aproximações das funções marginais e não mais as funções exatas, apresentando portanto um desempenho subótimo;
- O princípio básico para a propagação de mensagens não é mais obedecido, os nós recebem mensagens que contém informações sobre si;
- A propagação de mensagens efetuada pelo algoritmo dentro dos ciclos de $\mathcal{G}$, prossegue indefinidamente, pois os nós pertencentes aos ciclos

[10]Na literatura foi proposta uma técnica para terminar ambos os codificadores concomitantemente, mas o ganho em desempenho devido à técnica, não justifica o aumento de complexidade que ela implica [31].

[11]Vide Definição 2.14.

estarão sempre recebendo e transmitindo mensagens, sendo por isto necessária a utilização de um critério de parada com o intuito de finalizar o algoritmo, e de um cronograma de execução que lide com a propagação de mensagens ao longo dos ciclos.

Em relação ao princípio básico da propagação de mensagens, notou-se que um grafo-fator que possua ciclos "longos"[12] apresenta um desempenho próximo do ótimo, como é o caso do grafo que representa codificadores turbo, pois mensagens propagadas através de ramos distantes de um determinado nó, mesmo contendo informações sobre este nó, possuem pouco influência sobre o mesmo, o que permite que o grafo-fator seja considerado sem ciclos nas vizinhanças do nó [22].

Em $\mathcal{G}$, o componente que determina os comprimentos dos ciclos é o entrelaçador Π, ao embaralhar a seqüência de entrada do segundo codificador, ele acaba alongando os comprimentos dos ciclos se comparados aos comprimentos que os ciclos teriam caso não houvesse um entrelaçador entre os codificadores.

Note que o aumento do tamanho de Π, contribui para o aumento nos comprimentos dos ciclos de $\mathcal{G}$, o que melhora a aproximação da estrutura de $\mathcal{G}$ em uma estrutura de um grafo-fator sem ciclos, melhorando o desempenho do codificador turbo.

A propagação de mensagens indefinidas que o algoritmo Soma-Produto ocasiona em $\mathcal{G}$, cria a necessidade de um critério de parada, que no caso dos códigos turbo consiste em estabelecer um número pré-definido de iterações; e de um cronograma de execução que será responsável por determinar quais ramos estão ativos no grafo e em quais direções as mensagens estão sendo propagadas para cada instante de tempo[13].

O cronograma de execução que organiza a propagação de mensagens na decodificação turbo é definido, para uma iteração, de acordo com as seguintes etapas, vide Fig. 4.11:

1) Primeiramente, os nós m_t e $c_t^{(1)}$ enviam mensagens para os nós f_t, $\forall t$, iniciando a propagação de mensagens que resulta na execução do algoritmo MAP para o primeiro código componente;

2) Terminada a Etapa 1, os nós f_t enviam as mensagens $\delta^{(1)}(m_t)$ para os nós m_t, $\forall t$, finalizando a primeira fase da iteração;

3) Os nós m_t, enviam direcionados pelo entrelaçador Π, juntamente com os nós $c_{t'}^{(2)}$, mensagens para os nós $f_{t'}$, $\forall t'$, iniciando a propagação de

[12] O tamanho de um ciclo é igual ao número de ramos que o compõem.
[13] Considera-se que as mensagens são transmitidas em instantes discretos de tempo.

mensagens que resulta na execução do algoritmo MAP para o segundo código componente;

4) Por fim, os nós $f_{t'}$ enviam as mensagens $\delta^{(2)}(m_t)$ para os nós m_t, $\forall t$, completando uma iteração.

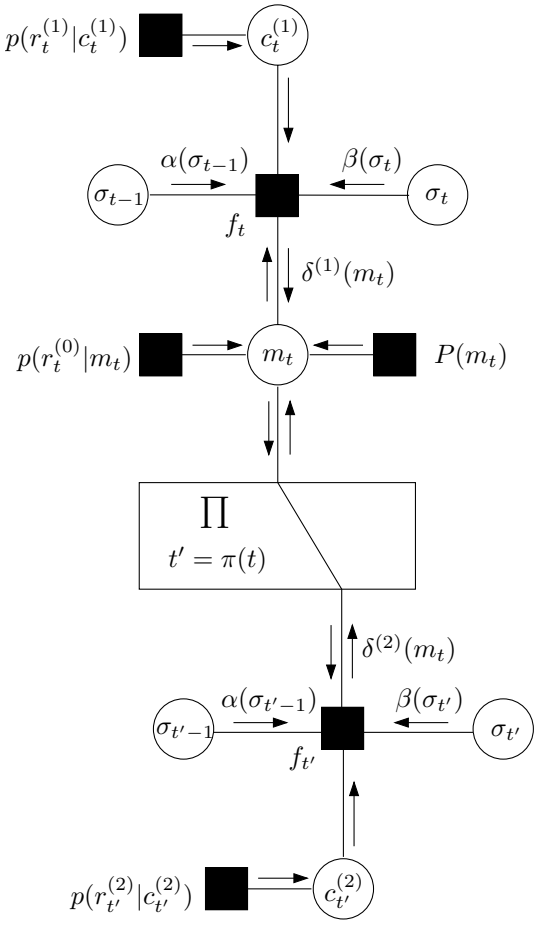

Figura 4.11: Propagação de mensagens para decodificação turbo iterativa.

O cronograma de execução acima é repetido até que o número máximo especificado de iterações seja alcançado. Finalizada a propagação de mensagens em $\mathcal{G}$, multiplica-se todas as mensagens recebidas pelos nós m_t e obtém-se

$$g_t(m_t) = p(r^{(0)}|m_t)P(m_t)\delta^{(1)}(m_t)\delta^{(2)}(m_t). \tag{4-42}$$

Vale lembrar que $g_t(m_t)$ não corresponde à função marginal exata, sendo apenas uma aproximação para a probabilidade $P(m_t|\boldsymbol{r})$.

Nesta seção pôde-se observar que alguns pontos da decodificação turbo são explicados de maneira simples através da observação da estrutura de $\mathcal{G}$. As mensagens, métricas na abordagem convencional, são obtidas de maneira mais intuitiva pela abordagem por grafos-fatores que pela abordagem convencional. Por estas razões, consideramos a abordagem por

grafos-fatores uma abordagem mais didática que a abordagem convencional, desenvolvida no Capítulo 3.

4.4.1 Decodificação Turbo e Outras Modulações

A decodificação turbo proposta em [4], foi desenvolvida para um canal DMC obtido pelo uso da modulação BPSK em um canal AWGN. Entretanto, para aplicações que possuam limitação em banda, é de interesse que códigos turbos estejam associados a modulações eficientes em relação ao uso da banda disponível.

Considere o codificador turbo $\mathcal{C}$ associado ao modulador $\mathcal{M}$ ilustrado na Fig. 4.12. As saídas do codificador turbo são entregues a um conversor paralelo/série que apresenta em sua saída

$$\boldsymbol{c} = \begin{pmatrix} m_1 & c_1^{(1)} & c_1^{(2)} & \cdots & m_N & c_N^{(1)} & c_N^{(2)} \end{pmatrix}. \tag{4-43}$$

O vetor codificado $\boldsymbol{c}$ é entrada para um entrelaçador de bits $\Pi_{\mathcal{M}}$ que disponibiliza ao modulador $\mathcal{M}$, a versão entrelaçada $\tilde{\boldsymbol{c}}$ de $\boldsymbol{c}$. O modulador $\mathcal{M}$ possui uma constelação b-ária, unidimensional, $\mathcal{A}^x = \{\pm 1, \ldots, \pm(2^b - 1)\}$, e mapeia um bloco de b bits, $\tilde{\boldsymbol{c}}_\ell$, de $\tilde{\boldsymbol{c}}$ no símbolo x_ℓ, ou seja, $x_\ell = \mathcal{M}(\tilde{\boldsymbol{c}}_\ell)$, onde $\tilde{\boldsymbol{c}} = \begin{pmatrix} \tilde{\boldsymbol{c}}_1 & \cdots & \tilde{\boldsymbol{c}}_\ell & \cdots & \tilde{\boldsymbol{c}}_M \end{pmatrix}$ e $M = \frac{N}{Rb}$. O símbolo x_ℓ assume valores na constelação $\mathcal{A}^x$.

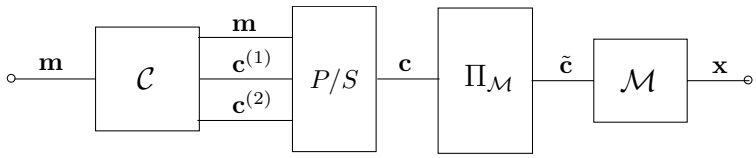

Figura 4.12: Codificador turbo $\mathcal{C}$ associado ao modulador $\mathcal{M}$.

O grafo-fator $\mathcal{G}$ de um codificador turbo associado a $\mathcal{M}$, representa a fatoração de uma função $g(\boldsymbol{m}, \tilde{\boldsymbol{m}}, \boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}, \boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}, \boldsymbol{x})$ definida como

$$g(\boldsymbol{m}, \tilde{\boldsymbol{m}}, \boldsymbol{\sigma}^{(1)}, \boldsymbol{\sigma}^{(2)}, \boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}, \boldsymbol{x}) \triangleq p(\boldsymbol{r}|\boldsymbol{x}) P(\boldsymbol{x}|\boldsymbol{c}) P(\boldsymbol{m}, \boldsymbol{\sigma}^{(1)}, \boldsymbol{c}^{(1)}) P(\tilde{\boldsymbol{m}}, \boldsymbol{\sigma}^{(2)}, \boldsymbol{c}^{(2)}) \tag{4-44}$$

onde $P(\boldsymbol{m}, \boldsymbol{\sigma}^{(1)}, \boldsymbol{c}^{(1)})$ e $P(\tilde{\boldsymbol{m}}, \boldsymbol{\sigma}^{(2)}, \boldsymbol{c}^{(2)})$ podem ainda ser fatoradas de acordo com as equações (4-21) e (4-18), respectivamente.

Assumindo um canal sem memória, tem-se ainda

$$p(\boldsymbol{r}|\boldsymbol{x}) = \prod_{\ell=1}^{M} p(r_\ell|x_\ell) \tag{4-45}$$
$$P(\boldsymbol{x}|\boldsymbol{c}) = \prod_{\ell=1}^{M} P(x_\ell|\tilde{\boldsymbol{c}}_\ell).$$

A função densidade de probabilidade $p(r_\ell|x_\ell)$ é obtida através do modelamento do canal. As probabilidades $P(x_\ell|\tilde{\boldsymbol{c}}_\ell)$ são dadas por

$$P(x_\ell|\tilde{\boldsymbol{c}}_\ell) = \begin{cases} 1, \text{se } \mathcal{M}(\tilde{\boldsymbol{c}}_\ell) = x_\ell \\ 0, \text{caso contrário} \end{cases} . \tag{4-46}$$

O grafo $\mathcal{G}$ diferencia-se do ilustrado na Fig. 4.10, apenas pela substituição dos nós $p(r_t^{(0)}|m_t)$, $p(r_t^{(1)}|c_t^{(1)})$ e $p(r_{t'}^{(0)}|c_{t'}^{(2)})$ pelos nós $p(r_\ell|x_\ell)$, $P(x_\ell|\tilde{\boldsymbol{c}}_\ell)$ e x_ℓ.

A inclusão dos nós $P(x_\ell|\tilde{\boldsymbol{c}}_\ell)$ cria novos ciclos em $\mathcal{G}$, sendo conveniente utilizar um entrelaçador de bits $\Pi_{\mathcal{M}}$ entre os nós $\{m_t, c_t^{(1)}, c_{t'}^{(2)}\}$ e x_ℓ, a fim de aumentar os comprimentos destes ciclos e melhorar conseqüentemente o desempenho do algoritmo Soma-Produto.

Exemplo:

Considere a modulação ASK-4 cujo grafo está ilustrada na Fig. 4.13. Tem-se que x_ℓ pertence à constelação $\mathcal{A}^x = \{\pm 1, \pm 3\}$, $\tilde{\boldsymbol{c}}_\ell \in \{00, 01, 10, 11\}$, e $P(x_\ell|\tilde{\boldsymbol{c}}_\ell) = 1$ para $(x_\ell, \tilde{\boldsymbol{c}}_\ell) \in \{(1, 00), (3, 01), (-1, 10), (-3, 11)\}$.

De acordo com a Fig. 4.13, tem-se que $x_1 = \mathcal{M}(m_3, c_2^{(1)})$, utilizando o algoritmo Soma-Produto descrito na Seção 4.1.2, as mensagens que o nó de função $P(x_1|m_3, c_2^{(1)})$ envia para os nós de variáveis m_3 e $c_2^{(1)}$ são obtidas por

$$\mu_{P_1 \to m_3}(m_3) = \sum_{\sim\{m_3\}} P(x_1|m_3, c_2^{(1)}) p(r_1|x_1) \tag{4-47}$$
$$\mu_{P_1 \to c_2^{(1)}}(c_2^{(1)}) = \sum_{\sim\{c_2^{(1)}\}} P(x_1|m_3, c_2^{(1)}) p(r_1|x_1).$$

onde por simplicidade utiliza-se a notação $P_\ell = P(x_\ell|\tilde{\boldsymbol{c}}_\ell)$, ou seja, $P_1 = P(x_1|m_3, c_2^{(1)})$.

O cronograma de execução para Fig. 4.13, é o mesmo descrito na Seção 4.4, incluindo apenas uma etapa inicial, denominada de etapa 0, descrita

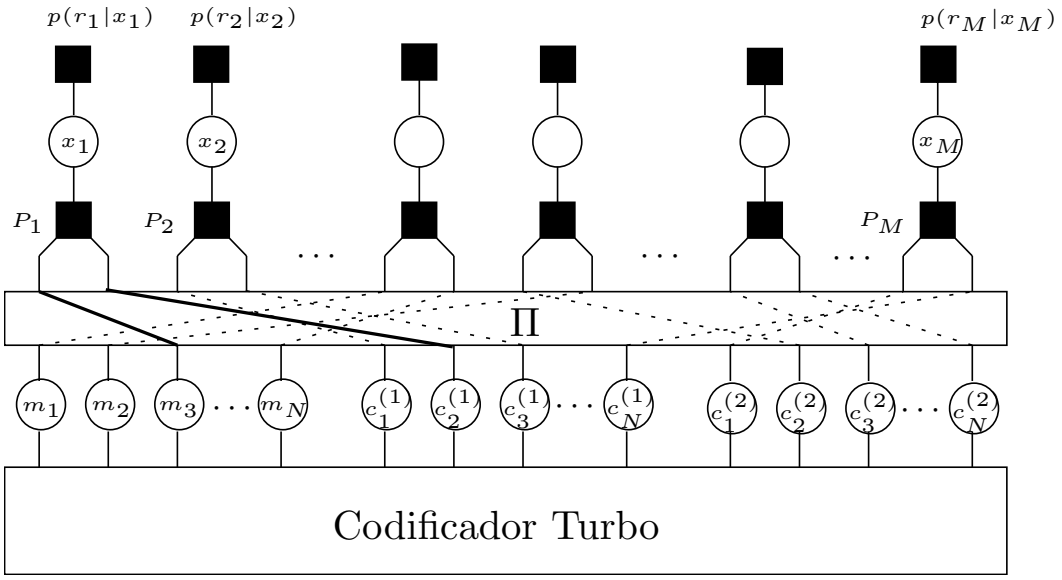

Figura 4.13: Grafo-fator de um codificador turbo associado à modulação ASK-4.

como:

0) Os ramos que conectam os nós m_t, $c_t^{(1)}$ aos nós f_t, m_t e $c_{t'}^{(2)}$ aos nós $f_{t'}$, ficam inativos nesta etapa. Os nós x_ℓ, $\ell = 1, \ldots, M$, recebem as mensagens $p(r_\ell|x_\ell)$ e apenas as repassam aos nós P_ℓ, que por fim enviam suas mensagens aos nós m_t, $c_t^{(1)}$ e $c_{t'}^{(2)}$, para $t = 1, \ldots, N$, $t' = 1, \ldots, N$. Os ramos que foram desativados são reativados para etapa 1. ■

Note que o desenvolvimento realizado nesta seção, pode ser facilmente expandido para o caso da modulação QAM-16, pois para o canal AWGN essa modulação pode ser implementada com o uso de dois canais AWGN em paralelo, cada um com um modulador[14] ASK-4. O grafo-fator utilizado para a decodificação será o mesmo da Fig. 4.13, mas com o dobro do número de nós para P_ℓ, x_ℓ e $p(r_\ell|x_\ell)$, onde agora $\ell = 1, \ldots, 2M$.

[14] O mesmo vale para as modulações QAM-M e ASK-N, onde $M = N \times N$, por exemplo, QAM-4(QPSK) e ASK-2(BPSK).