5 CÓDIGOS CONVOLUCIONAIS

Além dos Códigos de Blocos Lineares, os Códigos Convolucionais compõem outra grande família de códigos corretores de erros. Este capítulo descreve os fundamentos dos códigos convolucionais, sendo que os principais tópicos abordados nessas notas de aulas são:

- O codificador convolucional e suas representações;
- Decodificação de Códigos Convolucionais com o algoritmo de Viterbi;
- Capacidade de correção de erros dos Códigos Convolucionais;
- Códigos Convolucionais Catastróficos;
- Códigos Convolucionais sistemáticos; e
- Melhores Códigos Convolucionais Conhecidos.

5.1. INTRODUÇÃO

Considere o bloco de codificação apresentado na Figura 5.1, que representa um codificador convolucional. A mensagem **m** a ser codificada é representada pela seqüência $\mathbf{m} = m_1, m_2, \ldots, m_i, \ldots$, onde cada m_i representa um bit e i é o índice correspondente ao tempo.

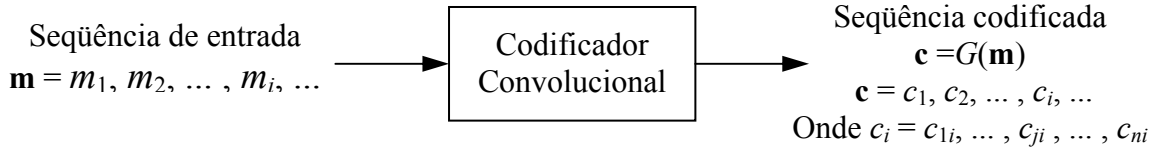


Figura 5.1 – Bloco de codificação convolucional.

Suponha que m_i assuma os valores zero ou um de forma equiprovável e que a presença de um ou outro seja estatisticamente independente, ou seja, o bit presente não depende de seu predecessor nem influencia seu sucessor.

O codificador transforma cada seqüência **m** em uma única seqüência código $\mathbf{c} = G(\mathbf{m})$. A seqüência **U** pode ser segmentada em uma seqüência de *palavras ramos*: $\mathbf{c} = c_1, c_2, \ldots, c_i, \ldots$ Cada palavra ramo c_i é um *símbolo código* binário, freqüentemente chamado de *símbolo do canal*, *bits do canal* ou *bits códigos*. Ao contrário dos bits da mensagem de entrada, os bits dos símbolos códigos não são independentes.

Isto significa que embora uma seqüência **m** defina uma única seqüência **c**, uma característica chave dos códigos convolucionais é que um elemento m_i de uma seqüência de entrada **m** não é suficiente para definir seu símbolo código associado c_i em **U,** uma vez que a codificação de cada elemento m_i não é apenas função do próprio m_i, mas é também do elemento m_{i-1} que o precedeu.

Um codificador convolucional genérico, mostrado na Figura 2.2, é constituído de um registrador de deslocamento de kK estágios e somadores módulo-2, onde K determina a *extensão de influência*, ou *profundidade* do código. A profundidade influência de um código convolucional é definida como sendo o máximo número de bits codificados que podem ser afetados por um único bit de entrada.

A cada unidade de tempo, k bits são deslocados para os primeiros k estágios do registrador; todos os bits no registrador são deslocados k bits para a direita e, as saídas dos n somadores são seqüencialmente amostradas para a obtenção do símbolo código ou bits códigos. Uma vez que n bits códigos são obtidos na saída para cada grupo de k bits de mensagem, a taxa do código é k/n bits de mensagem por bits códigos, onde $k < n$.

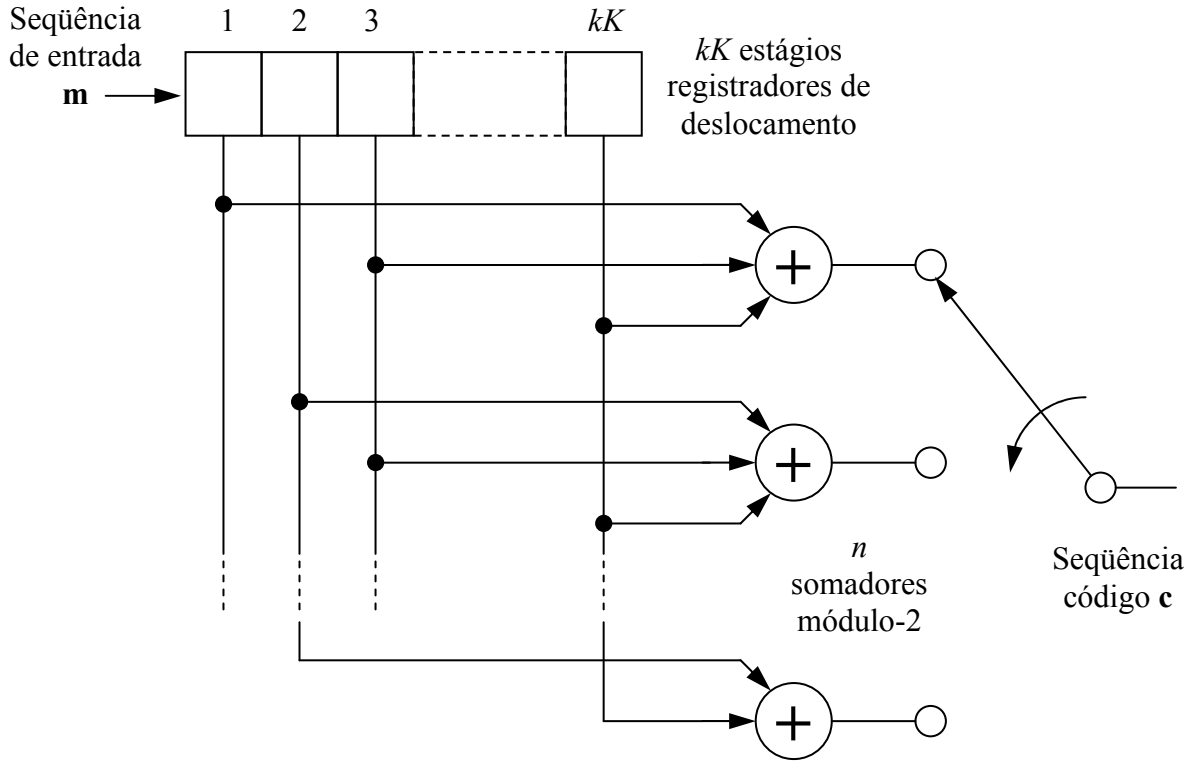


Figura 5.2 - Codificador convolucional.

5.2. REPRESENTAÇÕES E CARACTERÍSTICAS BÁSICAS DO CODIFICADOR [1] [2] [3]

Para descrever um código convolucional é necessário caracterizar a função de codificação $G(\mathbf{m})$ de tal forma que, dada uma seqüência de entrada **m**, seja possível determinar a saída codificada **c**. Uma abordagem muito comum para apresentar o processo da codificação convolucional é por meio de um exemplo. Assim sendo, os tópicos apresentados a seguir mostram as formas de representação mais comuns bem como as características básicas de um codificador convolucional e seu princípio de operação.

5.2.1. REPRESENTAÇÃO PICTORIAL

A representação pictorial nada mais é do que o desenho do codificador. Considere, por exemplo, o codificador convolucional apresentado na Figura 5.3, onde a cada instante de tempo, um bit de entrada ocupa a posição mais a esquerda do registrador de deslocamento (k = 1). Como o codificador possui dois somadores módulo-2, para cada bit que entra no codificador, dois bits códigos são gerados na saída pela amostragem dos resultados dos dois somadores. Logo este codificador gera um código de taxa k/n = 1/2, com profundidade K = 3.

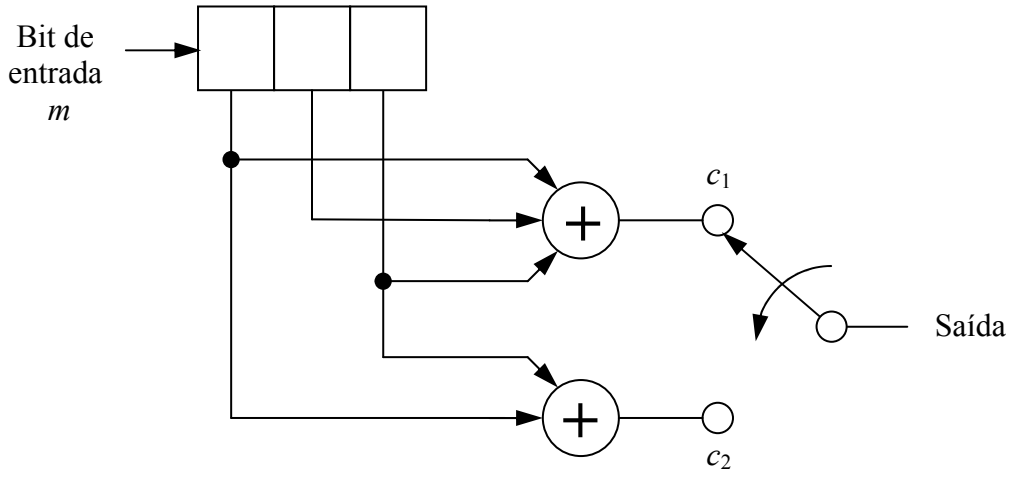


Figura 5.3 - Codificador convolucional (taxa 1/2, K = 3).

A escolha entre das conexões entre os somadores e os estágios do(s) registrador(es) determinam as características do código. Qualquer mudança na escolha das conexões resulta em um código diferente. As conexões não são escolhidas arbitrariamente, entretanto elas afetam as características de distância do código, ou seja, seu desempenho. Esta escolha é complicada e não existe, até o momento, uma regra geral que de escolha de conexões que maximize a distância de um código. Bons códigos, com profundidade menor que 20, têm sido obtidos através de busca computacional.

Ao contrário dos códigos de bloco que possuem palavras com comprimento fixo, um código convolucional não possui um tamanho de bloco particular. No entanto, códigos convolucionais são freqüentemente forçados para uma estrutura de bloco através de um truncamento periódico. Isso requer um número de bits zeros adicionados ao final de uma seqüência com o propósito de esvaziar o conteúdo dos registradores de deslocamento. Como os zeros adicionados não transportam informação, a taxa efetiva de codificação cai abaixo de k/n. Para manter a taxa de codificação próxima de k/n, o período de truncamento deve ser feito tão longo quanto prático.

5.2.2. PRINCÍPIO DE OPERAÇÃO

Suponha agora que a mensagem $\mathbf{m}(X) = 1 + X + X^3$ deve ser codificada pelo codificador da Figura 5.3. O processo de codificação é mostrado passo a passo na Figura 5.4 e as entradas, registro de todos os estados dos registradores e saída à cada deslocamento são apresentados na Tabela 5.1.

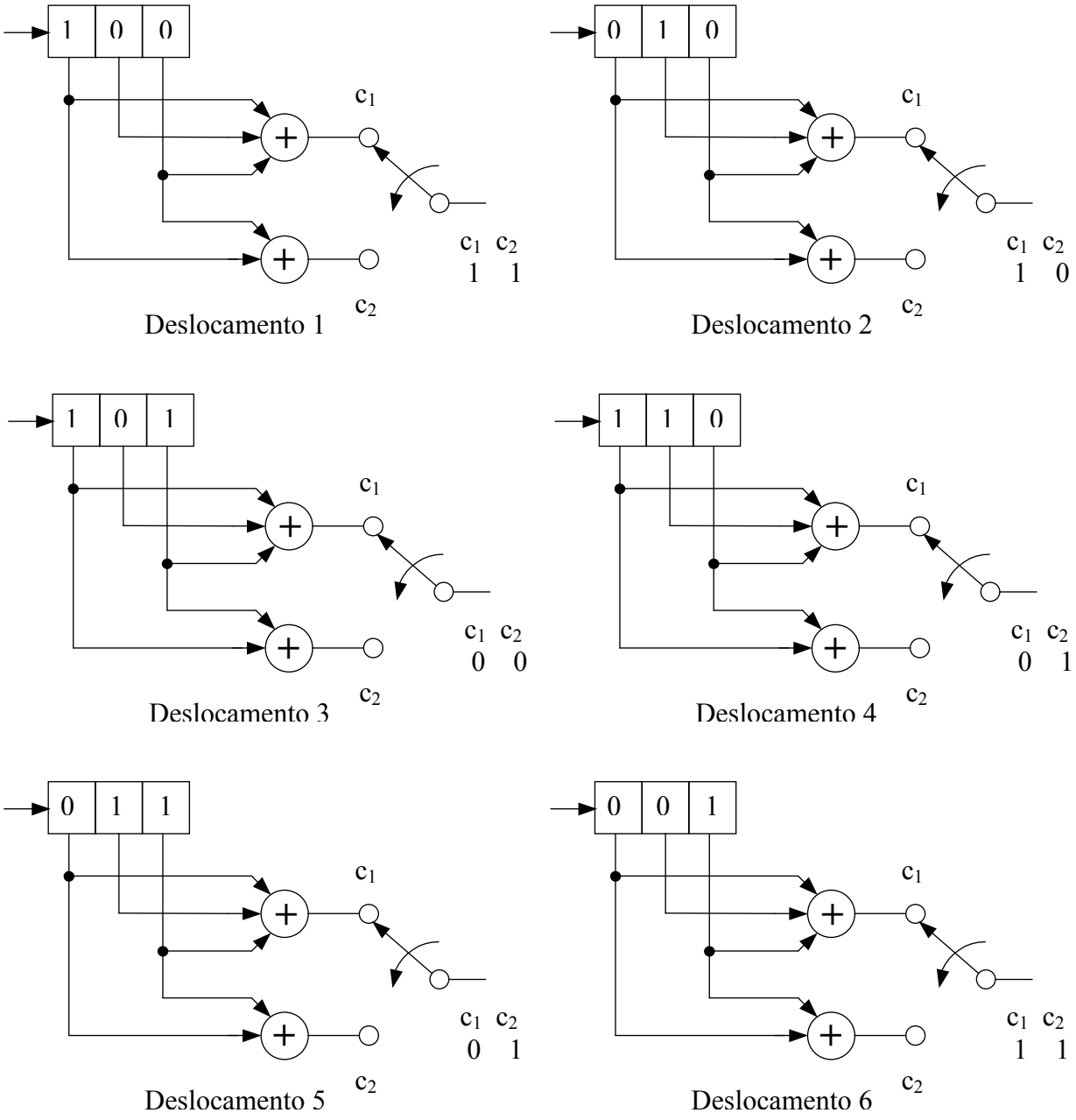


Figura 5.4 – Codificação da mensagem **m** = 1101 através da máquina de estados, obtida passo a passo.

Tabela 5.1 - Codificação da mensagem $\mathbf{m}(X) = 1 + X + X^3$
para o codificador da Figura 5.3.

DESLOCAMENTO	ENTRADA	CONTEÚDO DOS REGISTRADORES			SAÍDA $c_1\ c_2$
0	1	0	0	0	-
1	0	1	0	0	11
2	1	0	1	0	10
3	1	1	0	1	00
4	0	1	1	0	01
5	0	0	1	1	01
6	0	0	0	1	11
7	-	0	0	0	00

Saída: 11 10 00 01 01 11.

Observe, na Tabela 5.1, que foi necessário acrescentar dois zeros para a obtenção da seqüência codificada de saída e três zeros para esvaziar totalmente o registrador de deslocamento. Independentemente do tamanho da seqüência a ser codificada, é necessário acrescentar $k.K$ -1 zeros para a obtenção da seqüência código de saída. Conseqüentemente, o zero do sétimo deslocamento é desnecessário e o primeiro bit de um novo bloco de entrada poderia estar no primeiro estágio do registrador de deslocamento.

A Figura 5.3 mostra um codificador em sua forma pictorial. Outra forma de representar o codificador é através da especificação de um conjunto de n vetores conexão, um para cada somador módulo-2. Cada vetor tem a dimensão $k.K$ e descreve as conexões entre os estágios do registrador de deslocamento e os somadores módulo-2. Um *um* na i-ésima posição do vetor indica que o estágio correspondente do registrador de deslocamento está conectado ao somador representado pelo vetor e um *zero* em uma dada posição indica que aquela posição não é conectada ao somador. Para o codificador da Figura 2.3, pode-se escrever os vetores conexão $\mathbf{g}_1$ e $\mathbf{g}_2$ para os dois somadores como

$$\mathbf{g}_1 = 111$$
$$\mathbf{g}_2 = 101$$

5.2.3. RESPOSTA AO IMPULSO DO CODIFICADOR

Pode-se determinar a *resposta ao impulso* de um codificador convolucional, fazendo um único "1" atravessá-lo desde o primeiro estágio do registrador de deslocamento até o último. Para o codificador da Figura 5.3, a resposta ao impulso pode ser obtida de acordo com a Tabela 5.2.

Tabela 5.2 - Resposta ao impulso do Codificador Convolucional apresentado na Figura 5.3.

DESLOCAMENTO	CONTEÚDO DOS REGISTRADORES			SAÍDA
1	1	0	0	11
2	0	1	0	10
3	0	0	1	11

Resposta ao impulso: 11 10 11. Note que a resposta ao impulso pode ser obtida diretamente dos vetores conexão, conforme mostrado a seguir.

$$\begin{array}{lccc} \mathbf{g}_1 = & 1 & 1 & 1 \\ \mathbf{g}_2 = & 1 & 0 & 1 \\ & 11 & 10 & 11 \end{array}$$

A resposta ao impulso permite a determinação da saída do codificador para uma dada entrada **m**, pela superposição ou adição linear das respostas ao impulso deslocadas no tempo, ou ainda, pela convolução da seqüência de entrada com a resposta ao impulso do codificador. Para a mensagem **m** = 1 1 0 1, a saída do codificador obtida a partir da resposta ao impulso é obtida da forma como se segue.

Entrada **m**	Saída					
1	1 1	1 0	1 1			
0		0 0	0 0	0 0		
1			1 1	1 0	1 1	
1				1 1	1 0	1 1
Soma módulo-2	1 1	1 0	0 0	0 1	0 1	1 1

A resposta ao impulso deslocada pode ser representada também na forma matricial, e a seqüência código **c** pode ser obtida da forma **c** = **m**.*G*, da mesma forma que para códigos de blocos lineares.

$$\mathbf{c} = \mathbf{m} \cdot G = \begin{bmatrix} 1\,0\,1\,1 \end{bmatrix} \begin{bmatrix} 11 & 10 & 11 & & & \\ & 11 & 10 & 11 & & \\ & & 11 & 10 & 11 & \\ & & & 11 & 10 & 11 \end{bmatrix} = 11\,10\,00\,01\,01\,11$$

Note que as dimensões da matriz dependem da resposta ao impulso e do comprimento da mensagem.

É interessante notar ainda que apesar da taxa de codificação do codificador em questão ser igual a ½, uma mensagem de 4 bits produziu uma seqüência codificada de 12 bits, ou seja, uma taxa igual a 4/12 = 1/3. Isso se deve aos $k.K - 1$ zeros necessários para o esvaziamento do registrador de deslocamento, que neste caso são 2 e produz dois pares de bits adicionais à seqüência código. Conseqüentemente a taxa de codificação deve ser entendida como a taxa

quando a mensagem possui comprimento infinito. Por exemplo, para esse mesmo codificador, uma mensagem com 1000 bits produz uma seqüência código com 2000 + 4 bits. Neste caso a taxa é 1000/2004 ≅ ½.

5.2.4. REPRESENTAÇÃO POLINOMIAL

Um codificador convolucional pode ser representado por um conjunto de n polinômios geradores, correspondentes aos n somadores módulo-2 que compõem o codificador. Cada polinômio possui grau igual ou menor que $K - 1$ e descreve as conexões entre os estágios do registrador de deslocamento e seu respectivo somador, da mesma forma que os vetores conexão. Os coeficientes de cada termo de cada polinômio assumem valores 1 ou 0 dependendo da existência ou não de conexão entre uma posição específica do registrador de deslocamento e o seu respectivo somador. Para o codificador da Figura 5.3 os polinômios geradores, em termos de operador D (operador *delay*), são

$$\mathbf{g}_1(D) = 1 + D + D^2$$
$$\mathbf{g}_2(D) = 1 + D^2$$

onde o termo de mais baixa ordem corresponde ao estágio de entrada do codificador. A seqüência de saída do codificador pode ser obtida conforme indicado a seguir.

$$\mathbf{c}(D) = \mathbf{m}(D)\mathbf{g}_1(D) \text{ intercalado com } \mathbf{m}(D)\mathbf{g}_2(D).$$

Note que a mensagem também deve ser escrita em termos de operador atraso, ou seja, $\mathbf{m}(D)$, pois, da mesma forma que em uma transformação de Fourier, a convolução no domínio do tempo torna-se multiplicação no domínio do atraso. Na convenção adotada até agora, o termo de $\mathbf{m}(X)$ "mais cedo" é o termo de mais alta ordem, ou seja, é aquele que corresponde ao primeiro bit a entrar no codificador. Em termos de operador D, o primeiro bit a entrar no codificador é aquele que corresponde ao termo de menor ordem, que corresponde ao termo "mais cedo". Desta forma,

$$\mathbf{m}(X) = 1 + X + X^3 \Leftrightarrow \mathbf{m}(D) = D^3 + D^2 + 1$$

Por exemplo, para a mensagem $\mathbf{m}(D) = 1 + D^2 + D^3$, obtém-se:

$$\mathbf{m}(D)\mathbf{g}_1(D) = (1 + D^2 + D^3)(1 + D + D^2) = 1 + D + D^5$$
$$\mathbf{m}(D)\mathbf{g}_2(D) = (1 + D^2 + D^3)(1 + D^2) = 1 + D^3 + D^4 + D^5$$

$\mathbf{m}(D)\mathbf{g}_1(D) =$	$1 +$	$1\,D +$	$0\,D^2 +$	$0\,D^3 +$	$0\,D^4 +$	D^5
$\mathbf{m}(D)\mathbf{g}_2(D) =$	$1 +$	$0\,D +$	$0\,D^2 +$	$D^3 +$	$D^4 +$	D^5
$\mathbf{c} =$	11	10	00	01	01	11

5.2.5. DIAGRAMA DE ESTADOS

Um codificador convolucional pertence a uma classe de dispositivos conhecidos como *máquinas de estado finito*, que é o nome genérico para maquinas que tem memória dos sinais passados. O termo *finito* refere-se ao fato de que existe apenas um número de estados único e finito que a máquina pode gerar. Em um sentido amplo, um *estado* consiste de uma pequena quantidade de informação que, junto com a informação presente na entrada da máquina, é capaz de determinar a saída. Os estados fornecem algum conhecimento de eventos passados e o restrito conjunto de possíveis saídas no futuro. Um estado futuro fica restringido por um estado passado. Para um codificador convolucional com taxa $1/n$, o estado atual representado pelo tampo t_i é representado pelo conteúdo dos $(K - 1)$ estágios mais a direita, conforme mostrado na Figura 5.5. O estado representado pelo tempo $t_i + 1$ é o estado futuro. O conhecimento do estado e da próxima entrada é necessário e suficiente para determinar a próxima saída.

Um codificador convolucional pode ser representado por seu diagrama de estados. Como um exemplo, o diagrama de estados apresentado na Figura 5.6, descreve o comportamento do codificador da Figura 5.5. O diagrama de estados do codificador pode ser obtido a partir da verificação de todas as possibilidades de transição entre os estados do codificador, conforme mostrado na Tabela 5.5.

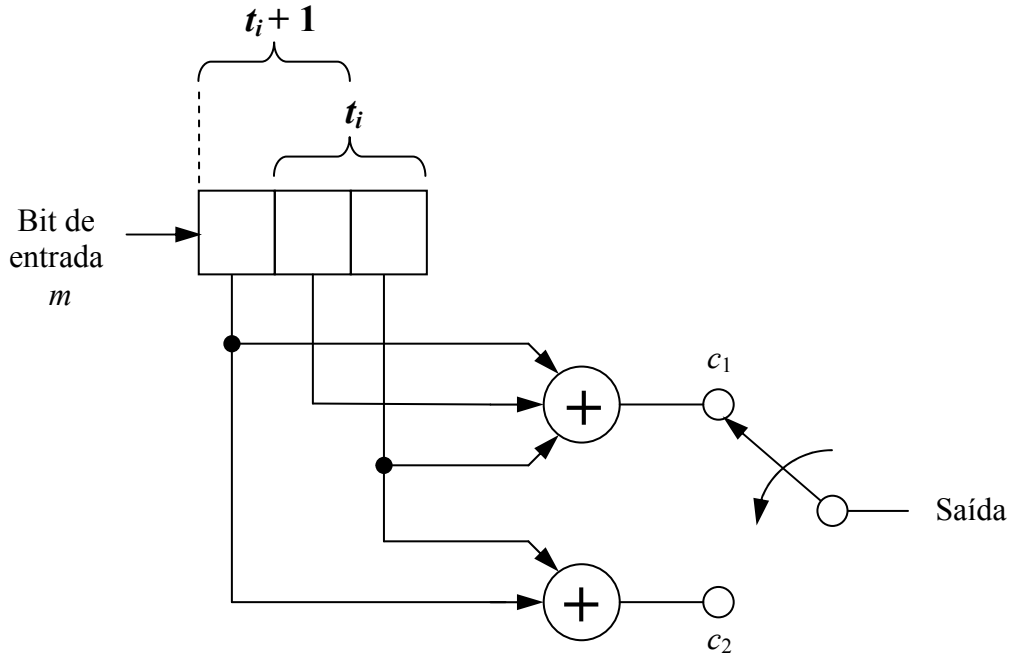


Figura 5.5 – Definição do estado atual (t_i) e estado futuro ($t_i + 1$) para um codificador com $K = 3$.

De acordo com a Figura 5.6(a), o estado "00" é representado por um círculo do qual partem duas transições. Uma transição parte de um estado no tempo t_i e alcança um estado no tempo $t_i + 1$ e é representada por uma seta. Para que isso aconteça, é necessário que esteja presente na entrada um bit m_i que provoca uma saída c_1c_2, representado pela notação m_i/c_1c_2, próximo a cada transição. Conforme mostrado na Tabela 5.5, a partir do estado "00", pode-se permanecer nele se a entrada for "0" ou migrar para o estado "10" se a entrada for "1".

Tabela 5.5 – Construção do diagrama de estados para o codificador da Figura 5.5.

Entrada m_i	Conteúdos dos registradores	Estado em t_i	Estado em $t_i + 1$	Saída em t_i $c_1\ c_2$	Figura
0	000	00	00	00	5.6(a)
1	100	00	10	11	
0	010	10	01	10	5.6(b)
1	110	10	11	01	
0	011	11	01	01	5.6(c)
1	111	11	11	10	
0	001	01	00	11	5.6(d)
1	101	01	10	00	

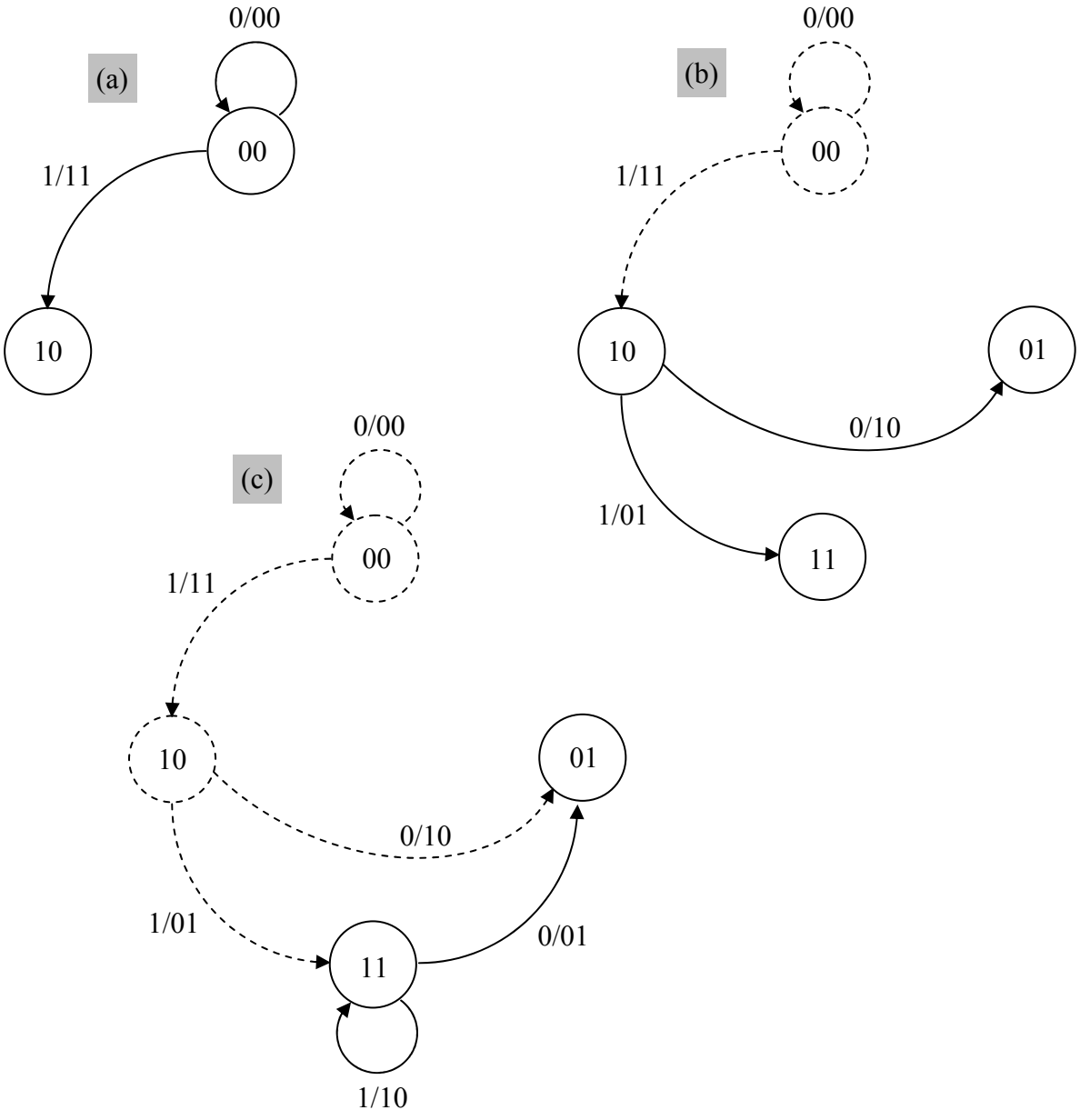


Figura 5.6 (a), (b) e (c)- Construção do diagrama de estados de acordo com a Tabela 5.5.

A Figura 5.6(b) apresenta as transições que partem do estado "10". De acordo com a Tabela 5.5, a partir do estado "10" pode-se migrar para o estado "01" se a entrada for "0" ou migrar para o estado "11" se a entrada for "1".

Da mesma forma, a Figuras 5.6(c) apresenta as transições que partem do estado "11". De acordo com a Tabela 5.5, se a entrada for "1" o próximo estado é o próprio estado "11" e se entrada for "0" ocorre uma transição para o estado "01".

Finalmente, na Figura 5.6(d) são apresentadas as transições que partem do estado "01" e alcançam os estados "10" e "00" se as entradas forem "1" e "0", respectivamente. A Figura 5.6(e) apresenta o diagrama de estados completo.

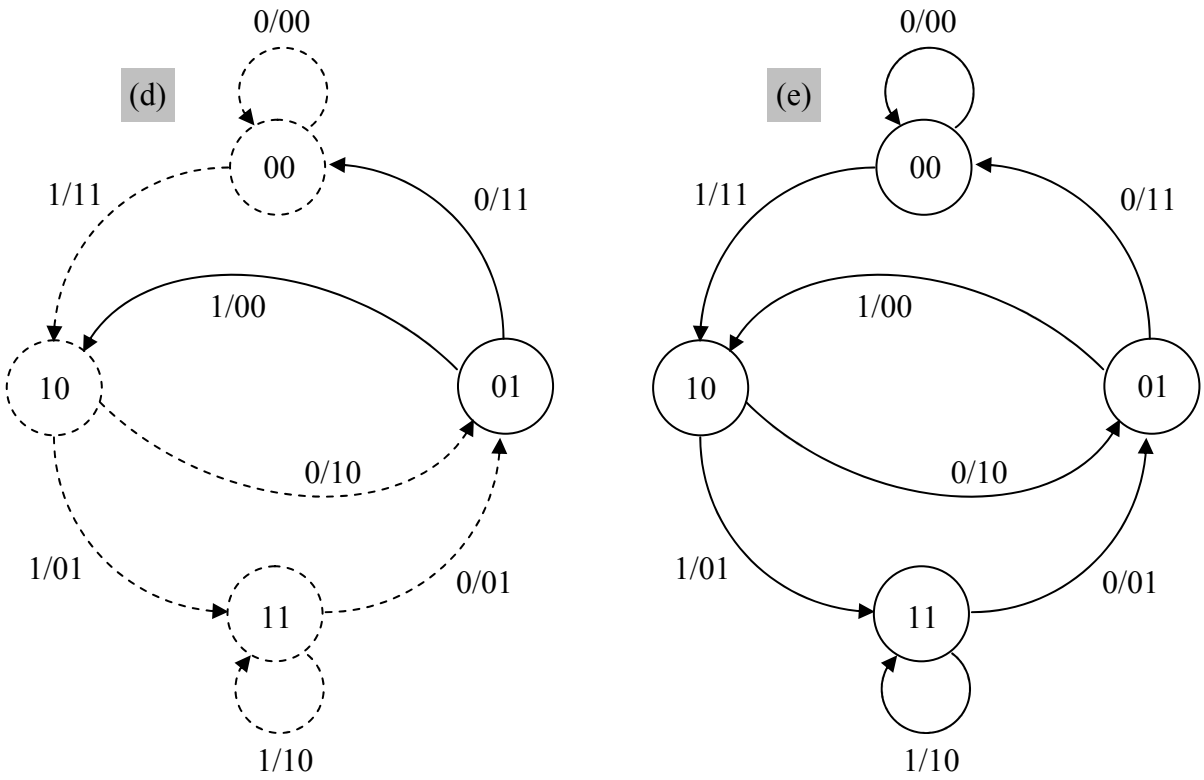


Figura 5.6 (d) e (e) - Construção do diagrama de estados de acordo com a Tabela 5.5.

5.2.6. DIAGRAMA DE ÁRVORE

O diagrama de árvore acrescenta a dimensão de tempo ao diagrama de estado. Ao contrário do diagrama de estado que não permite representar a história do tempo, no diagrama de árvore a evolução das mudanças de estado e a resultado na saída é visível, conforme mostrado na Figura 5.7.

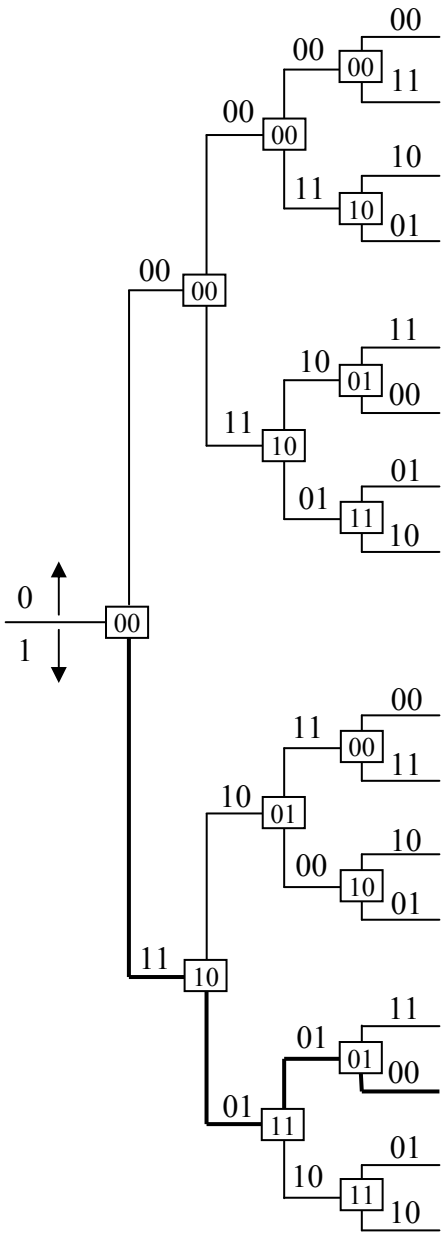


Figura 5.7 – Diagrama de árvore para o codificador convolucional da Figura 5.5.

No diagrama de árvore apresentado pela Figura 5.7, cada ramo vertical representa uma entrada. Um ramo para cima representa uma entrada *zero* enquanto um ramo para baixo representa uma entrada *um*. As saídas do codificador são as palavras binárias colocadas sobre os ramos horizontais. Cada derivação da árvore, ou nó, representa um estado, identificado pela palavra binária colocada na caixa de cada nó. Assim, para a mensagem m = 1101, pode-se verificar pelo trajeto representado pela linha mais espessa na Figura 5.7, que a seqüência de estados correspondentes aos quatro bits da mensagem é 00 10 01 10. As saídas correspondentes a esta trajetória são 11 10 10 01. Note que cada nó representa o instante de tempo em que um estado é atingido. Por isso, este diagrama permite a obtenção de um histórico de transições ao longo do tempo.

5.2.7. DIAGRAMA DE TRELIÇA

Uma observação da Figura 5.7 permite concluir que a partir de um determinado instante de tempo, a estrutura do diagrama torna-se repetitivo. Além disso, para uma observação muitos intervalos de tempo, o diagrama de árvore torna-se pouco prático em função do grande número de ramos do diagrama. Uma alternativa mais prática ao diagrama de árvore é o diagrama de treliça. No diagrama de treliça tem-se um histórico de entradas, transições e saídas sem que o diagrama cresça demasiadamente. A Figura 5.8 apresenta o diagrama de treliça para o codificador da Figura 5.5. Neste diagrama, os estados são representados pelos níveis horizontais e as entradas e saídas são representadas pela mesma convenção utilizada no diagrama de estados, ou seja, m_i/u_1u_2, que são colocados sobre cada braço da treliça, que por sua vez, representa uma transição.

A mensagem m = 1101 estabelece, no diagrama de treliça, a trajetória mostrada na Figura 5.9, resultando, como era de se esperar, nas saídas 11 10 10 01. Note que para esvaziar os registradores do codificador, mais dois zeros na entrada são necessários, resultando em uma saída complementar igual a 01 11. Assim a seqüência de saída completa para a mensagem m = 1101 torna-se 11 10 00 01 01 11. Este resultado é rigorosamente igual aos resultados apresentados anteriormente.

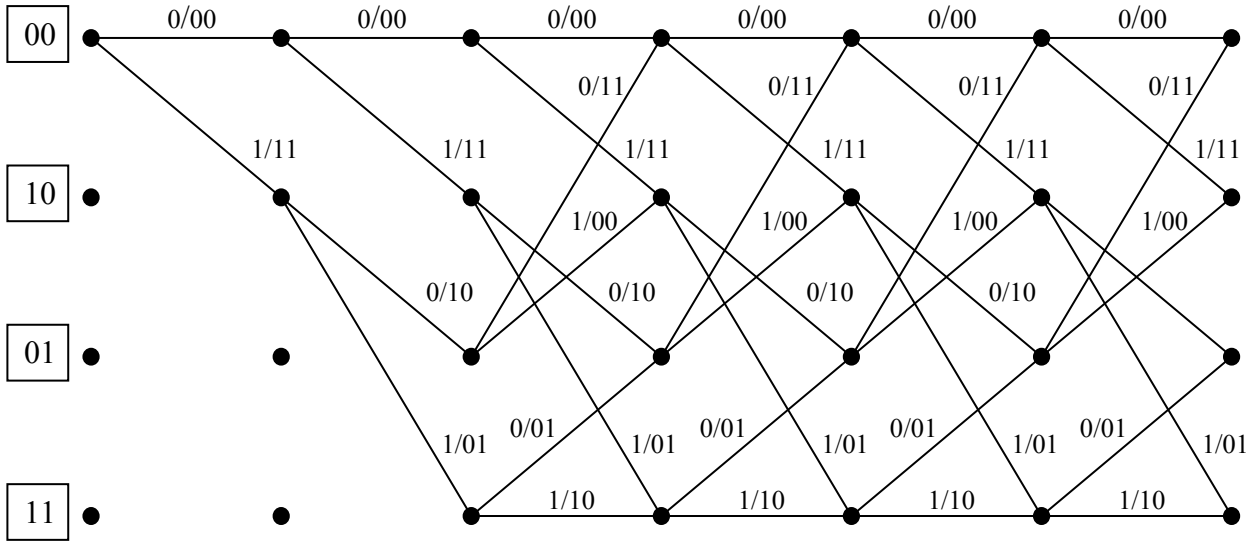


Figura 5.8 – Diagrama de treliça para o codificador convolucional da Figura 5.5.

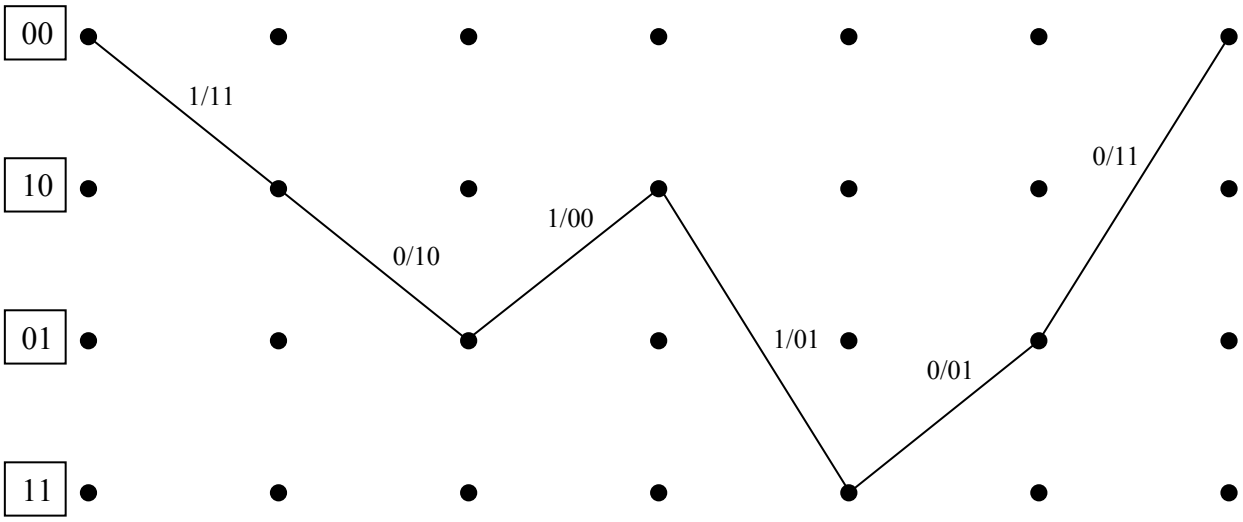


Figura 5.9 – Trajetória para a mensagem m = 1101.

5.3. Decodificação de Códigos Convolucionais pelo Algoritmo de Viterbi [1] [2] [3]

O algoritmo de Viterbi é um algoritmo de máxima verossimilhança com baixa carga computacional em função da utilização da estrutura dos diagramas de treliça dos códigos convolucionais. A vantagem da decodificação de Viterbi, comparado com a decodificação por força bruta, é que a complexidade de um decodificador não é função do número de símbolos da seqüência código, mas função de uma medida de *similaridade* ou *distância* entre o sinal recebido em um tempo t_i e todos os braços da treliça que entram em cada estado no tempo t_i. Quando dois braços entram no mesmo estado em um tempo t_i, o que possuir melhor métrica ou maior semelhança com o sinal recebido é escolhido e chamado de *caminho sobrevivente*.

Existem, basicamente, duas distâncias que podem ser utilizadas no algoritmo de Viterbi para a medida de similaridade entre a seqüência recebida pelo decodificador e as seqüências possíveis sobre a treliça: a distância de Hamming e a distância Euclidiana. A distância de Hamming é utilizada em um processo de decisão chamado de *decisão abrupta* (*hard decision*) enquanto a distância Euclidiana, ou um parâmetro derivado dela, é utilizado em um processo de decisão chamado de *decisão suave* (*soft decision*). A decisão abrupta destaca-se pela simplicidade e baixa carga computacional enquanto que a decisão suave requer uma carga computacional maior em troca de melhor desempenho em termos de capacidade de correção de erros. As subseções apresentadas a seguir descrevem os dois processos de decodificação.

5.3.1. Decodificação Abrupta pelo Algoritmo de Viterbi

Para facilitar o entendimento do algoritmo de Viterbi, considere a seqüência recebida 11 10 10 01 01 11 codificada pelo codificador da Figura 5.5, cujo diagrama de treliça está apresentado na Figura 5.8. Note que foi introduzido um erro no terceiro par de bits.

Passo 1: Partindo do estado 00, compara-se a distância de Hamming do primeiro par de bits da seqüência código (11) com as distâncias de Hamming das duas transições possíveis a partir do estado 00. As distâncias de Hamming obtidas são armazenadas (valores entre parênteses), conforme apresentado na Figura 5.10(a).

Passo 2: A distância de Hamming do próximo par de bits da seqüência código (10) é comparada com as distâncias de Hamming das transições possíveis a partir dos estados alcançados após o passo anterior. As distâncias de Hamming encontradas são somadas àquelas obtidas nas transições anteriores. Veja Figura 5.10(b).

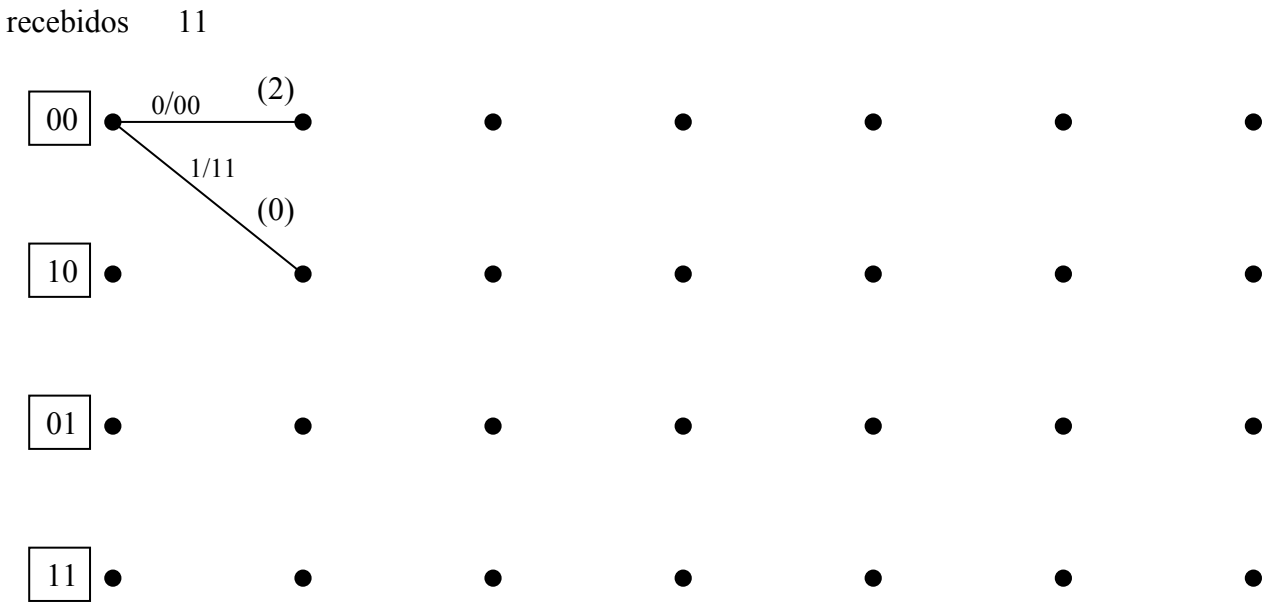


Figura 5.10(a) – Situação após o passo 1 do exemplo de decodificação.

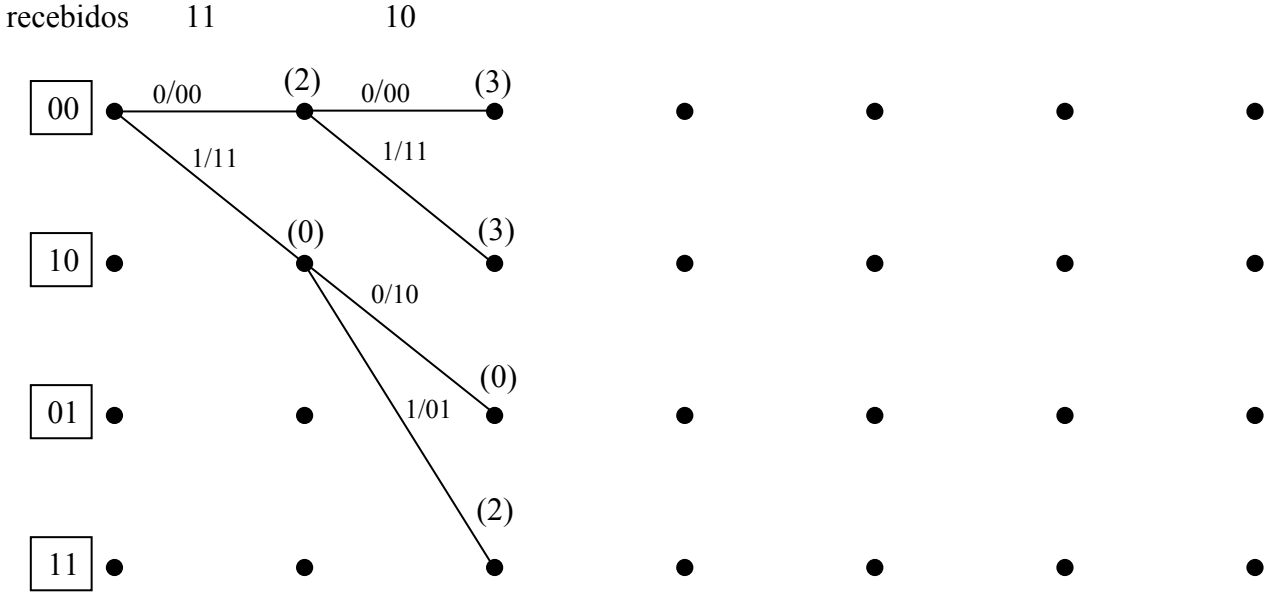


Figura 5.10(b) – Situação após o Passo 2 do exemplo de decodificação.

Passo 3: O mesmo procedimento apresentado no Passo 2 é repetido para o próximo par da seqüência código (10). Note que neste terceiro passo alguns dos estados são alcançados por dois caminhos diferentes. O caminho sobrevivente deve ser aquele que apresenta a menor distância de Hamming acumulada. Na Figura 5.10(c), os caminhos sobreviventes estão representados por uma linha mais espessa.

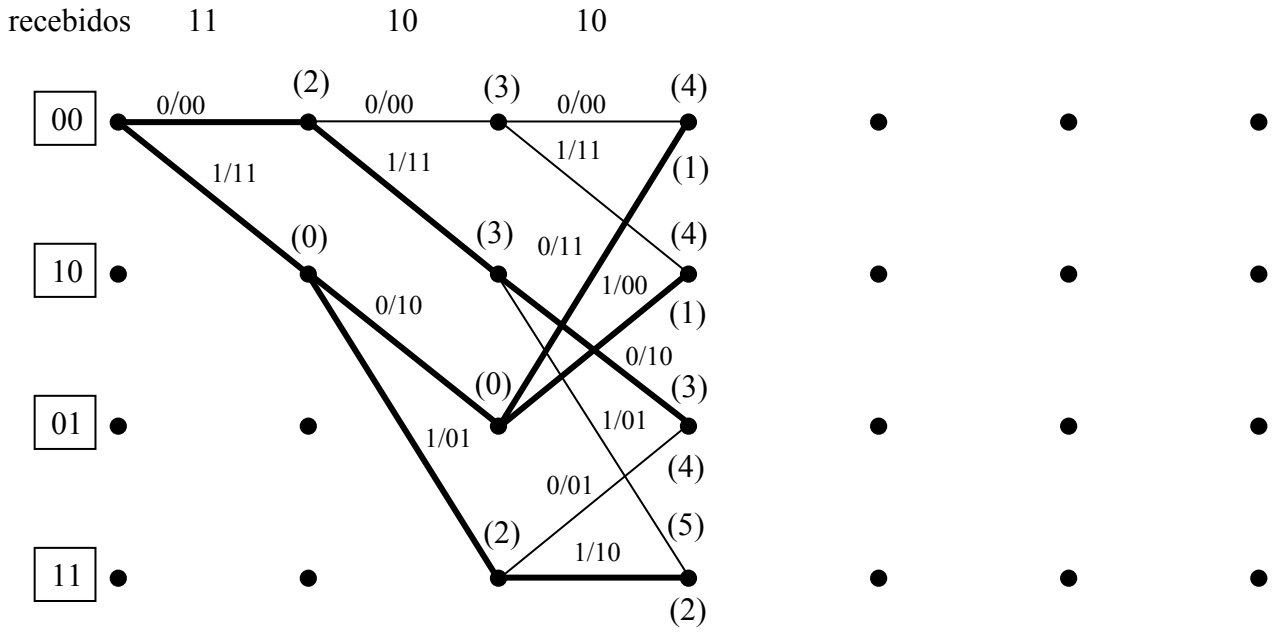


Figura 5.10(c) – Situação após o Passo 3 do exemplo de decodificação.

Passo 4: O mesmo procedimento é repetido até o final da seqüência. Para cada par do sinal recebido deve-se inspecionar a treliça e eleger o caminho sobrevivente. As Figuras 5.10(d), (e) e (f) apresentam a evolução da treliça, mostrando apenas os caminhos sobreviventes. Note que ao final da seqüência código recebida, apenas um caminho é o caminho sobrevivente final.

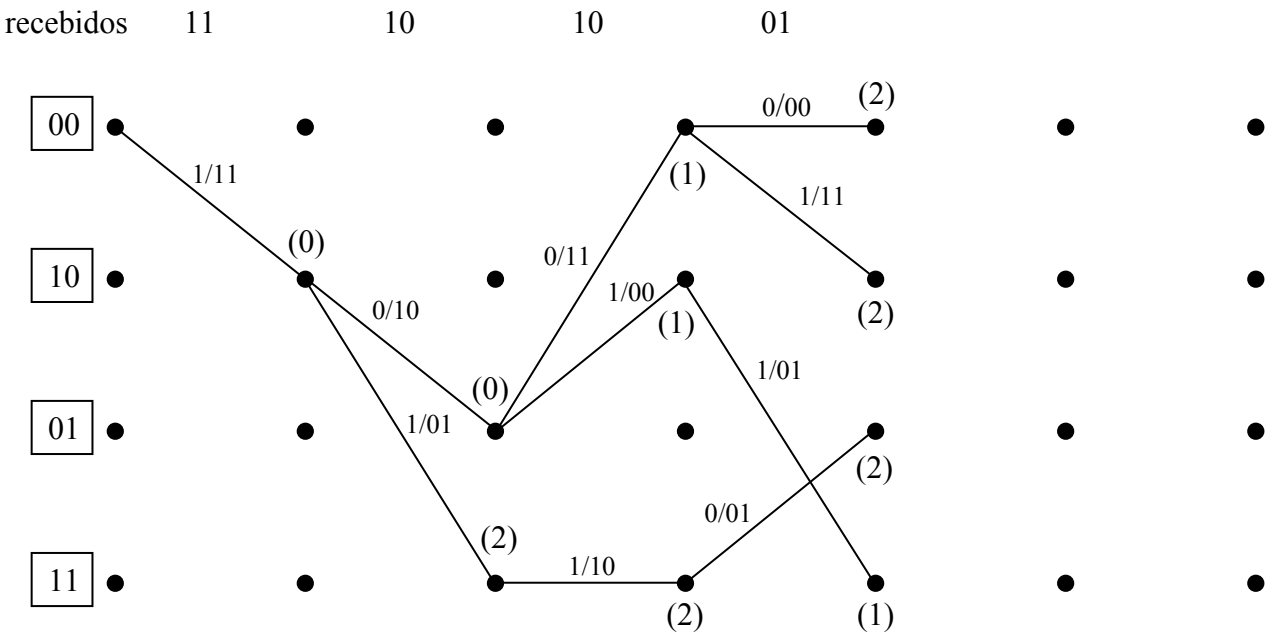


Figura 5.10(d) – Situação após a análise do 4º par de bits recebidos do exemplo de decodificação.

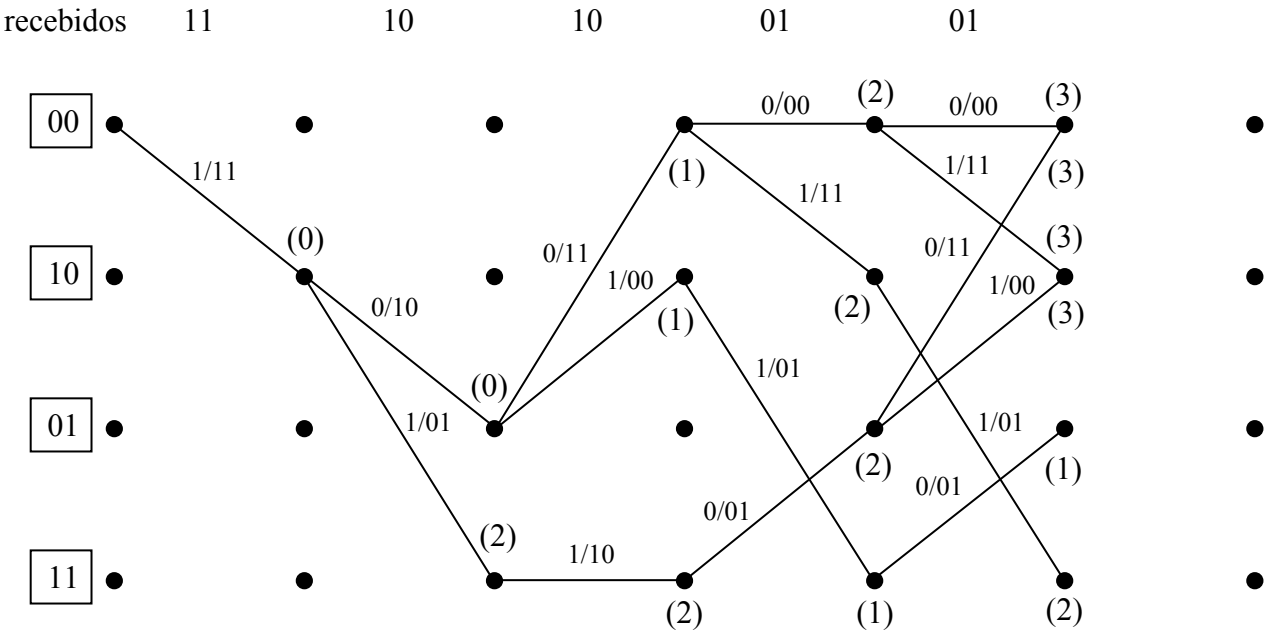


Figura 5.10(e) – Situação após a análise do 5º par de bits recebidos do exemplo de decodificação.

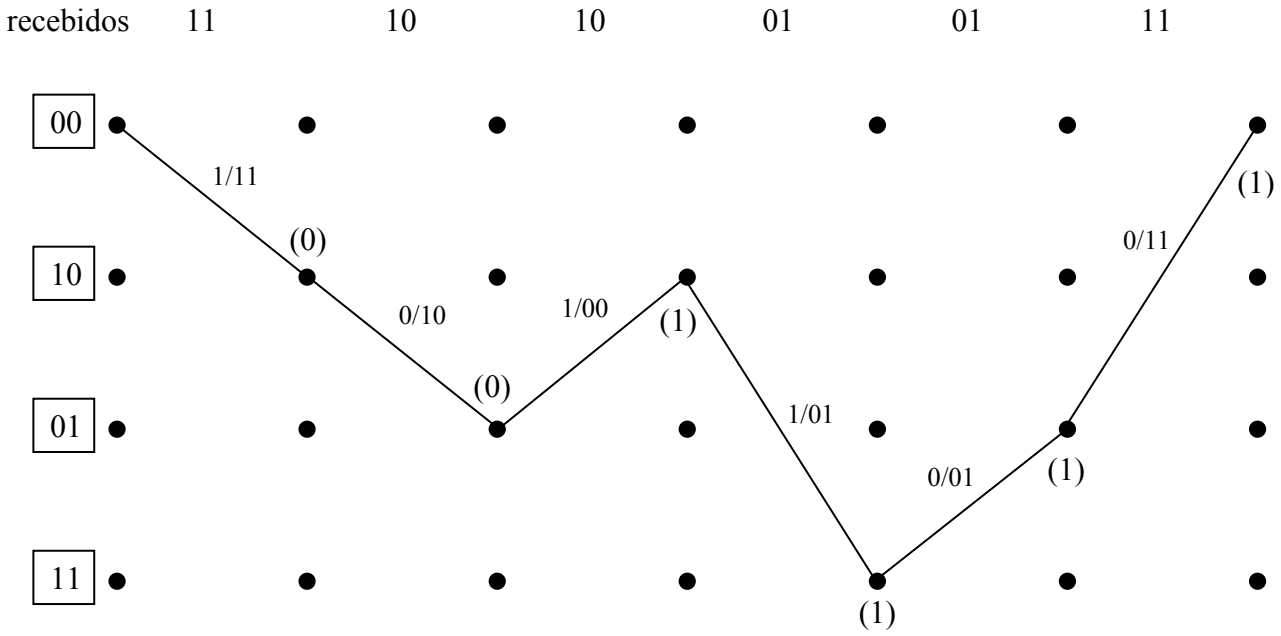


Figura 5.10(f) – Situação após a análise do 6º par de bits recebidos do exemplo de decodificação.

Conclusão: Após a análise do último par de bits recebidos, o caminho sobrevivente aponta uma distância de Hamming acumulada igual a 1. Este resultado mostra que a seqüência decodificada com maior probabilidade de ter sido a seqüência transmitida é a seqüência 11 10 00 01 01 11, correspondente a mensagem m = 101100. Neste caso a seqüência recebida apresenta um erro em relação à seqüência decodificada, correspondente à distância de Hamming acumulada igual a 1.

Note que, no exemplo apresentado a seqüência analisada é curta e o resultado da decodificação torna-se óbvio. Entretanto, quando a seqüência recebida é extremamente longa, o uso do algoritmo de Viterbi requer a utilização de uma quantidade de memória muito alta. A abordagem usualmente usada é "truncar" a seqüência decodificada conforme os passos descritos a seguir.

1. Uma *janela de decodificação* de comprimento l é especificada.
2. O algoritmo opera sobre um quadro correspondente ao comprimento l, parando sempre após l intervalos de tempo.
3. A decisão é tomada sobre o melhor caminho e o símbolo associado com o primeiro ramo do caminho é liberado para o usuário.
4. O símbolo associado com o último ramo do caminho é desconsiderado.
5. A *janela de decodificação* é movida um intervalo de tempo para frente e a decisão sobre o melhor caminho sobre o novo quadro é feita, e assim por diante.

A decisão de decodificação feita dessa maneira não é verdadeiramente máxima verossimilhança, mas ela pode ser feita quase tão boa desde que a janela de decodificação seja suficientemente grande. Experiência e análises têm demonstrado que resultados satisfatórios são obtidos se a janela de decodificação tem comprimento l da ordem de cinco vezes ou mais do que a *extensão de influência* K do código convolucional.

5.3.2. DECODIFICAÇÃO SUAVE PELO ALGORITMO DE VITERBI

A decodificação com o algoritmo de Viterbi apresentada anteriormente tem por finalidade encontrar a seqüência código ou seqüência válida que mais se assemelha à seqüência binária recebida. Isso é feito por meio do cálculo da distância de Hamming acumulada em todos os caminhos possíveis da treliça de forma que através da eleição dos caminhos sobreviventes, seja possível a determinação do caminho sobrevivente final, que é aquele que apresenta maior verossimilhança com a seqüência recebida. Portanto, o parâmetro utilizado para a eleição dos caminhos sobreviventes é a distância de Hamming.

Isso pressupõe que os símbolos recebidos foram determinados previamente por um processo de decisão de máxima verossimilhança, onde os símbolos recebidos no espaço de sinais são aproximados para os símbolos mais próximos, ou seja, o parâmetro utilizado neste processo de decisão é a distância Euclidiana. Neste caso, verifica-se que o processo de decisão e o processo de decodificação são dois processos realizados independentemente. Esta decodificação é chamada de *decodificação abrupta* (*hard-decision*).

A seguir será apresentado o processo de decodificação suave, onde a decisão dos símbolos recebidos leva em conta, simultaneamente, a distância Euclidiana e o código corretor de erro utilizado, ou seja, o processo de decisão por meio de distâncias Euclidianas e a decodificação com o algoritmo de Viterbi são fundidos em um único processo. Para isso, considere o esquema de codificação e modulação apresentado na Figura 5.11.

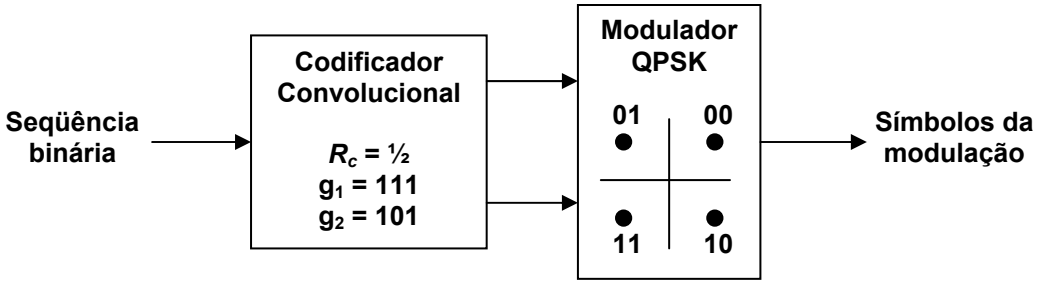


Figura 5.11 - Exemplo de um esquema de modulação e codificação a decodificação suave.

Admita que a constelação recebida, na ausência de ruído, tenha seus simbolos posicionados no espaço de sinais de acordo com as coordenadas apresentadas na Figura 5.12.

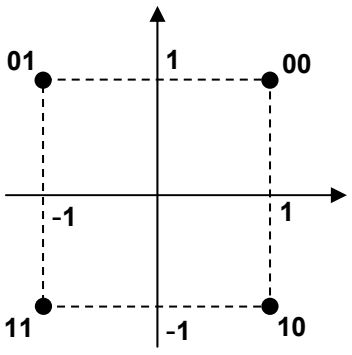


Figura 5.12 - Coordenadas dos símbolos de modulação no espaço de sinais.

Na decisão por máxima verossimilhança, um ponto recebido contaminado por ruído é decidido como sendo o símbolo mais próximo do espaço de sinais. Ou seja, decide-se pelo símbolo que apresenta a menor distância Euclidiana para o ponto recebido. A distância Euclidiana E_i de um ponto recebido para um símbolo S_i de uma constelação pode ser calculada, em função de suas coordenadas ortogonais, pela expressão

$$E_i = \sqrt{(i_i - i')^2 + (q_i - q')^2} \tag{5.1}$$

onde i_i é a coordenada do símbolo i no eixo I, i' é a coordenada do ponto recebido no eixo I, q_i é a coordenada do símbolo i no eixo Q e q' é a coordenada do ponto recebido no eixo Q, conforme mostrado na Figura 5.13.

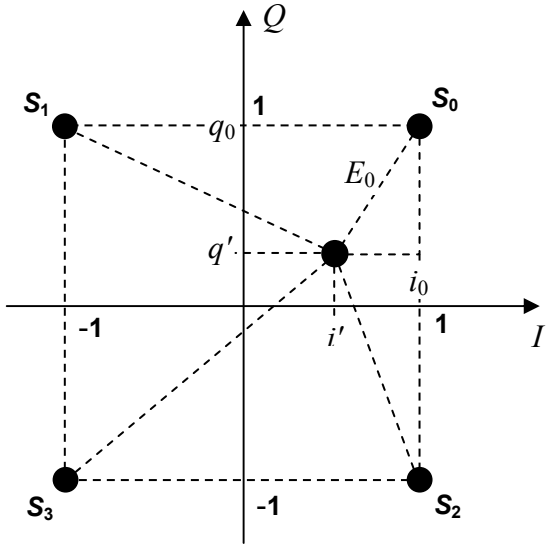


Figura 5.13 - Distância Euclidiana E_0 de um ponto recebido para o símbolo S_0.

Para a modulação em questão, os símbolos, seus valores binários e suas coordenadas são mostrados na Tabela 5.6.

Tabela 5.6 - Símbolos, seus valores binários e suas coordenadas

Símbolo	Valor binário	Coordenadas (i_i, q_i)
S_0	00	(1, 1)
S_1	01	(-1, 1)
S_3	11	(-1, -1)
S_2	10	(1, -1)

Suponha agora que seis pontos tenham sido transmitidos através de um canal ruidoso, de acordo com o esquema de codificação e modulação apresentado na Figura 5.11. Admita que no receptor tenham chegado seis pontos com as seguintes coordenadas:

$$r_1 = (-0{,}93;\ -0{,}03);\ r_2 = (0{,}55;\ 0{,}11);\ r_3 = (0{,}35;\ 1{,}13);$$
$$r_4 = (-0{,}97;\ -0{,}02);\ r_5 = (0{,}20;\ 0{,}42);\ r_6 = (-0{,}41;\ -0{,}25).$$

Em um processo de **decisão abrupta**, os pontos recebidos são aproximados para o símbolo da constelação mais próximo, resultando na seqüência

$$S_3,\ S_0,\ S_0,\ S_3,\ S_0,\ S_3,$$

que correspondem aos valores binários

$$11,\ 00,\ 00,\ 11,\ 00,\ 11.$$

Esta seqüência binária, quando decodificada de acordo com o algoritmo de Viterbi resulta na treliça mostrada na Figura 4.

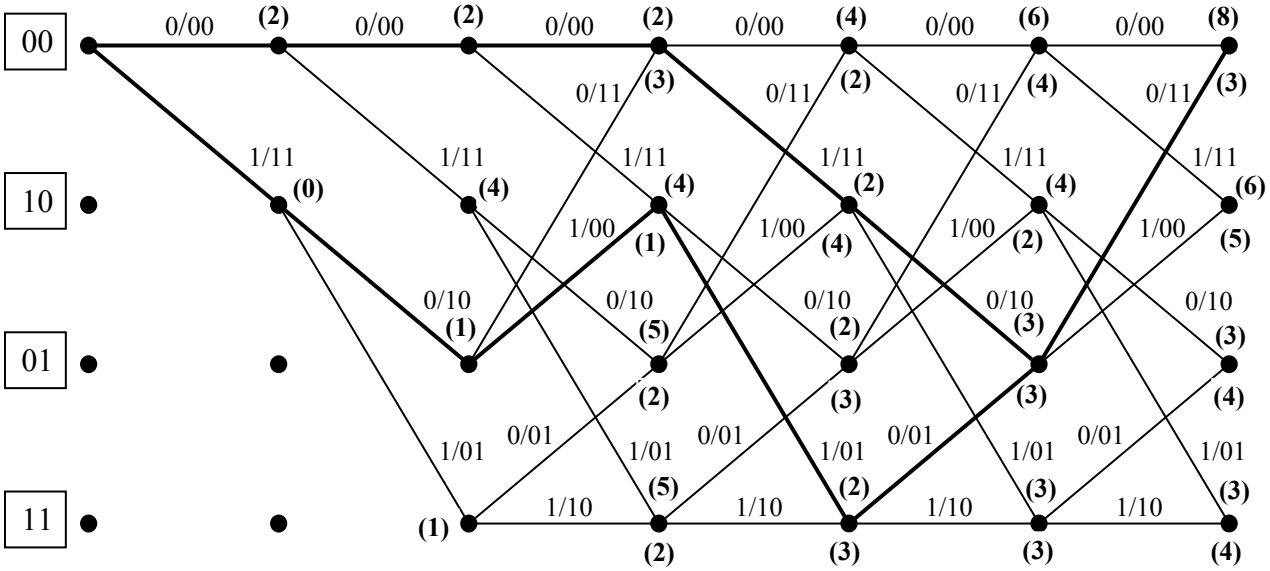


Figura 5.14 - Caminhos sobreviventes correspondentes à decodificação abrupta da seqüência de pontos $r_1, r_2, \ldots, r_6$.

De acordo com o resultado apresentado na Figura 5.14, observa-se que existem duas possibilidades para o caminho sobrevivente final, que são:

00 00 00 11 10 11 ou 11 10 00 01 01 11.

Desde que a capacidade de correção de erros do código não tenha sido excedida pelo número de erros introduzido pelo canal, então uma das duas seqüências é a seqüência transmitida. Note que, neste caso, a chance de se eleger a seqüência de máxima verossimilhança é de 50%.

O algoritmo de Viterbi para a decodificação suave é o mesmo utilizado para a decodificação abrupta. Entretanto, no processo de decodificação suave, as distâncias acumuladas ao longo da treliça não são as distâncias de Hamming e sim as distâncias Euclidianas dos pontos recebidos em relação aos símbolos representados por cada braço da treliça. Desta forma, na decodificação suave, cada braço da treliça deve ser identificado com o par de coordenadas do símbolo obtido na saída do codificador, de acordo com as correspondências entre coordenadas e valores binários apresentados na Tabela 5.6.

A partir do exposto acima, pode-se redesenhar a treliça para a decodificação suave da forma como mostrada na Figura 5.15.

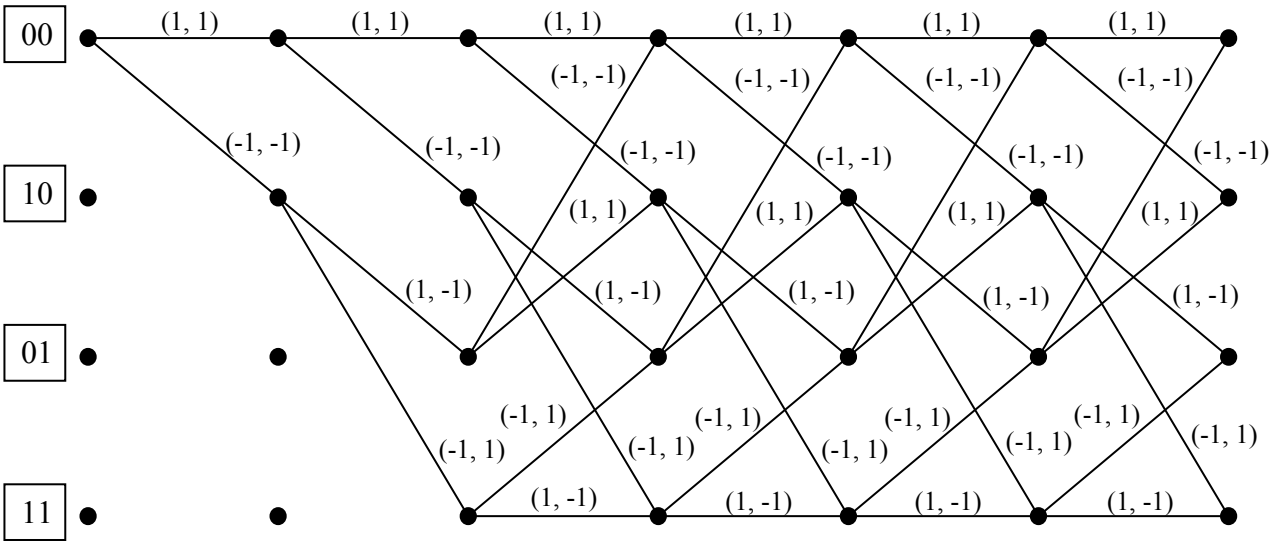


Figura 5.15 - Treliça para a decodificação suave.

O início da decodificação suave consiste na determinação da distância do ponto recebido para os símbolos das transições que saem do estado 00, ou seja, os ramos cujos símbolos estão nas coordenadas (1, 1) e (-1, -1). A partir dos estados alcançados, calculam-se novamente as distâncias para os estados seguintes acumulando-as com as anteriores e assim por diante, conforme mostrado a seguir.

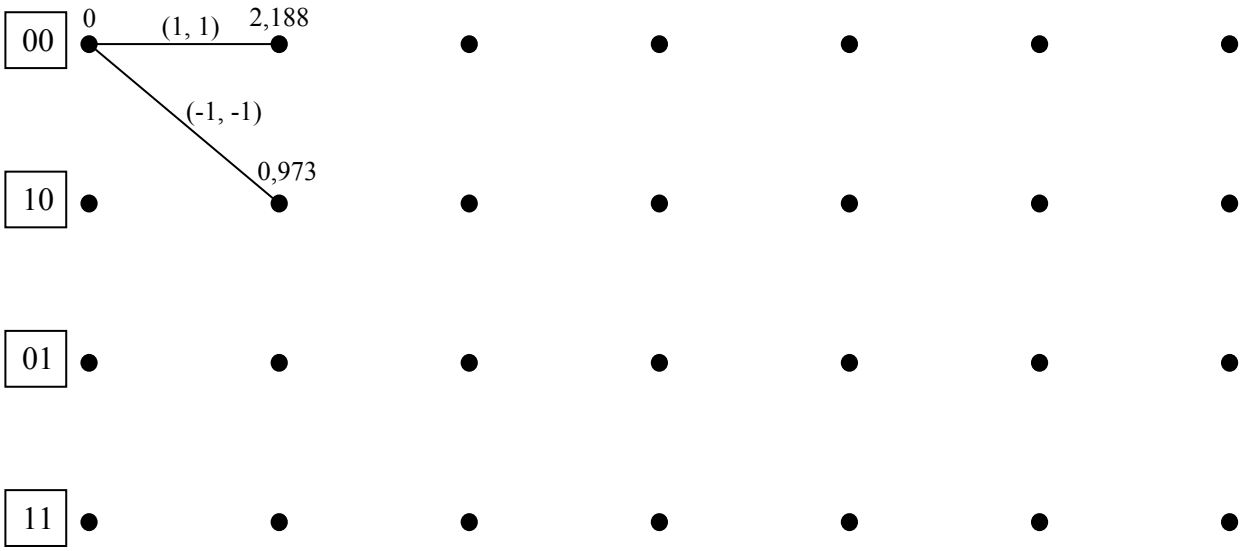
De r_1 = (-0,93; -0,03) para:

$$S_0 = (1,\ 1) \quad \Rightarrow \quad E_0 = \sqrt{(1+0{,}93)^2 + (1+0{,}03)^2} \qquad E_0 = 2{,}188$$

$$S_3 = (-1,\ -1) \quad \Rightarrow \quad E_3 = \sqrt{(-1+0{,}93)^2 + (-1+0{,}03)^2} \qquad E_3 = 0{,}973$$

$$\sum\nolimits_{00-00} = 0 + 2{,}188 = 2{,}188$$

$$\sum\nolimits_{00-10} = 0 + 0{,}973 = 0{,}973$$



De $r_2 = (0{,}55;\ 0{,}11)$ para:

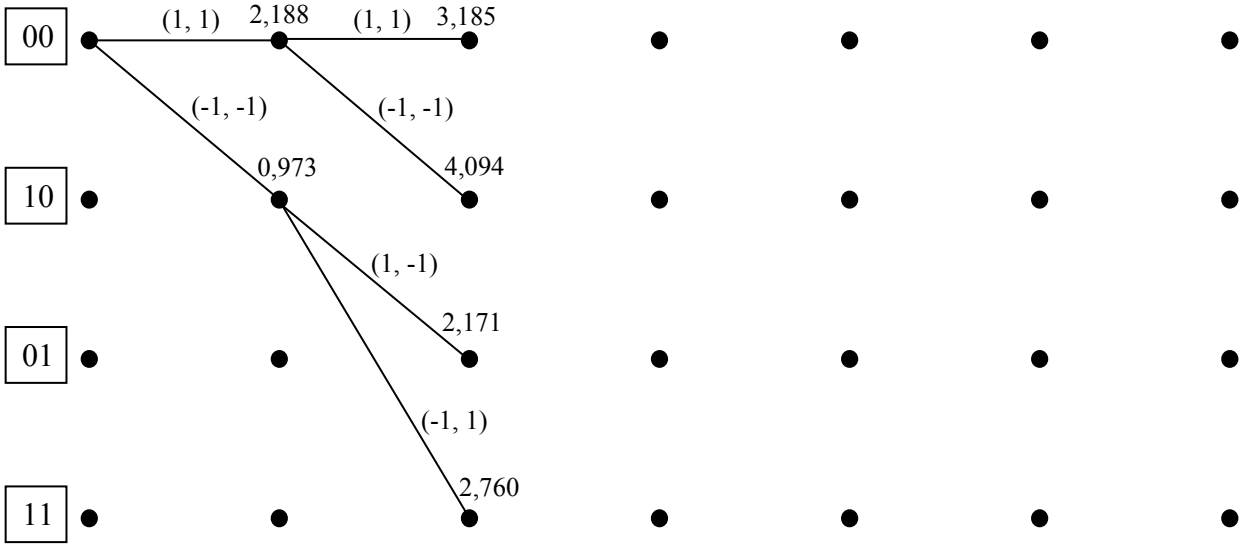
$$S_0 = (1,\ 1) \quad \Rightarrow \quad E_0 = \sqrt{(1-0{,}55)^2 + (1-0{,}11)^2} \qquad E_0 = 0{,}997$$

$$S_3 = (-1,\ -1) \quad \Rightarrow \quad E_3 = \sqrt{(-1-0{,}55)^2 + (-1-0{,}11)^2} \qquad E_3 = 1{,}906$$

$$S_2 = (1,\ -1) \quad \Rightarrow \quad E_2 = \sqrt{(1-0{,}55)^2 + (-1-0{,}11)^2} \qquad E_2 = 1{,}198$$

$$S_1 = (-1,\ 1) \quad \Rightarrow \quad E_1 = \sqrt{(-1-0{,}55)^2 + (1-0{,}11)^2} \qquad E_1 = 1{,}787$$

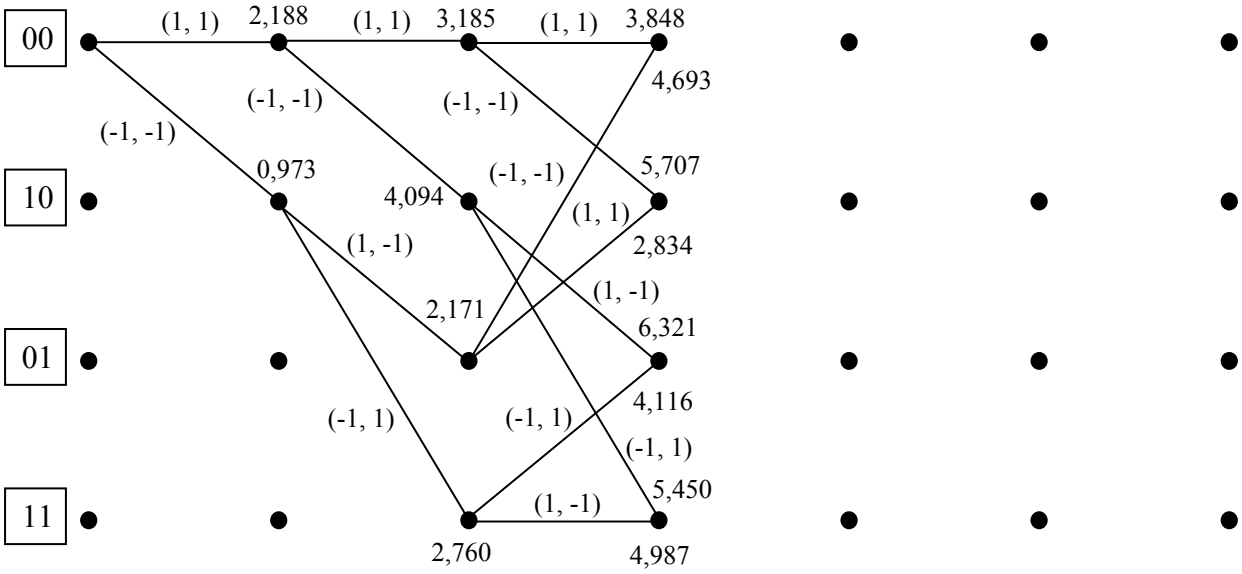
$$\sum\nolimits_{00-00} = 2{,}188 + 0{,}997 = 3{,}185$$
$$\sum\nolimits_{00-10} = 2{,}188 + 1{,}906 = 4{,}094$$
$$\sum\nolimits_{10-01} = 0{,}973 + 1{,}198 = 2{,}171$$
$$\sum\nolimits_{10-11} = 0{,}973 + 1{,}787 = 2{,}760$$



De $r_3 = (0{,}35;\ 1{,}13)$ para:

$$S_0 = (1,\ 1) \quad \Rightarrow \quad E_0 = \sqrt{(1-0{,}35)^2 + (1-1{,}13)^2} \qquad E_0 = 0{,}663$$
$$S_3 = (-1,\ -1) \quad \Rightarrow \quad E_3 = \sqrt{(-1-0{,}35)^2 + (-1-1{,}13)^2} \qquad E_3 = 2{,}522$$
$$S_2 = (1,\ -1) \quad \Rightarrow \quad E_2 = \sqrt{(1-0{,}35)^2 + (-1-1{,}13)^2} \qquad E_2 = 2{,}227$$
$$S_1 = (-1,\ 1) \quad \Rightarrow \quad E_1 = \sqrt{(-1-0{,}35)^2 + (1-1{,}13)^2} \qquad E_1 = 1{,}356$$

$$\sum_{00-00} = 3{,}185 + 0{,}663 = 3{,}848 \qquad \sum_{01-10} = 2{,}171 + 0{,}663 = 2{,}834$$
$$\sum_{00-10} = 3{,}185 + 2{,}522 = 5{,}707 \qquad \sum_{01-00} = 2{,}171 + 2{,}522 = 4{,}693$$
$$\sum_{10-01} = 4{,}094 + 2{,}227 = 6{,}321 \qquad \sum_{11-11} = 2{,}760 + 2{,}227 = 4{,}987$$
$$\sum_{10-11} = 4{,}094 + 1{,}356 = 5{,}450 \qquad \sum_{11-01} = 2{,}760 + 1{,}356 = 4{,}116$$

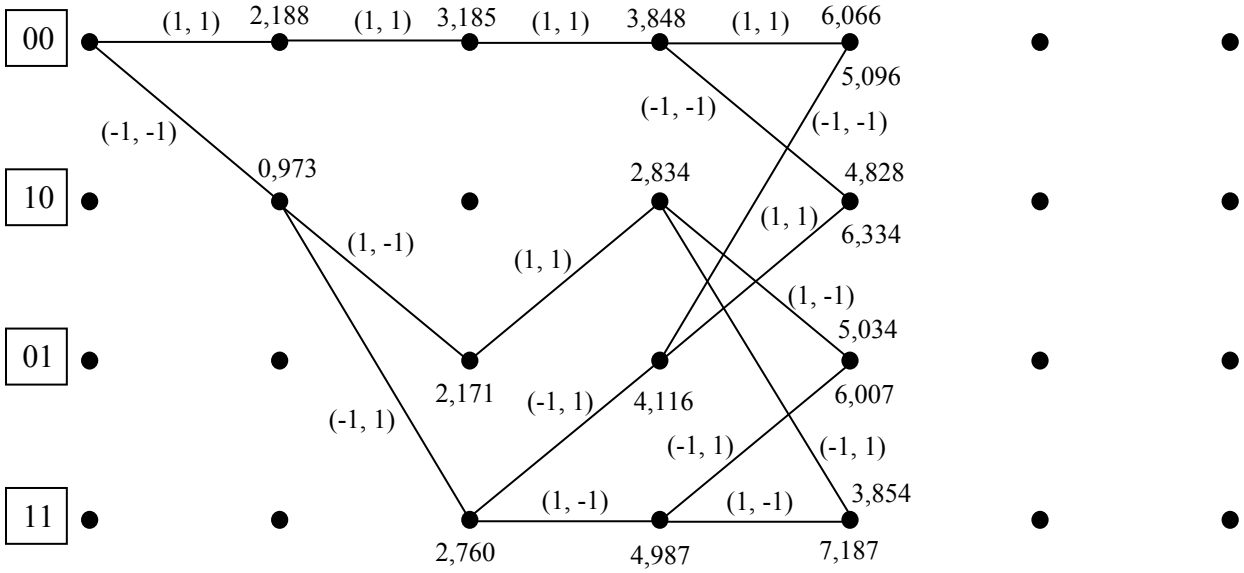


A partir desta etapa deve-se eleger os caminhos sobreviventes para a etapa seguinte.

De $r_4 = (-0{,}97;\ -0{,}02)$ para:

$$S_0 = (1,\ 1) \quad \Rightarrow \quad E_0 = \sqrt{(1+0{,}97)^2 + (1+0{,}02)^2} \qquad E_0 = 2{,}218$$
$$S_3 = (-1,\ -1) \quad \Rightarrow \quad E_3 = \sqrt{(-1+0{,}97)^2 + (-1+0{,}02)^2} \qquad E_3 = 0{,}980$$
$$S_2 = (1,\ -1) \quad \Rightarrow \quad E_2 = \sqrt{(1+0{,}97)^2 + (-1+0{,}02)^2} \qquad E_2 = 2{,}200$$
$$S_1 = (-1,\ 1) \quad \Rightarrow \quad E_1 = \sqrt{(-1+0{,}97)^2 + (1+0{,}02)^2} \qquad E_1 = 1{,}020$$

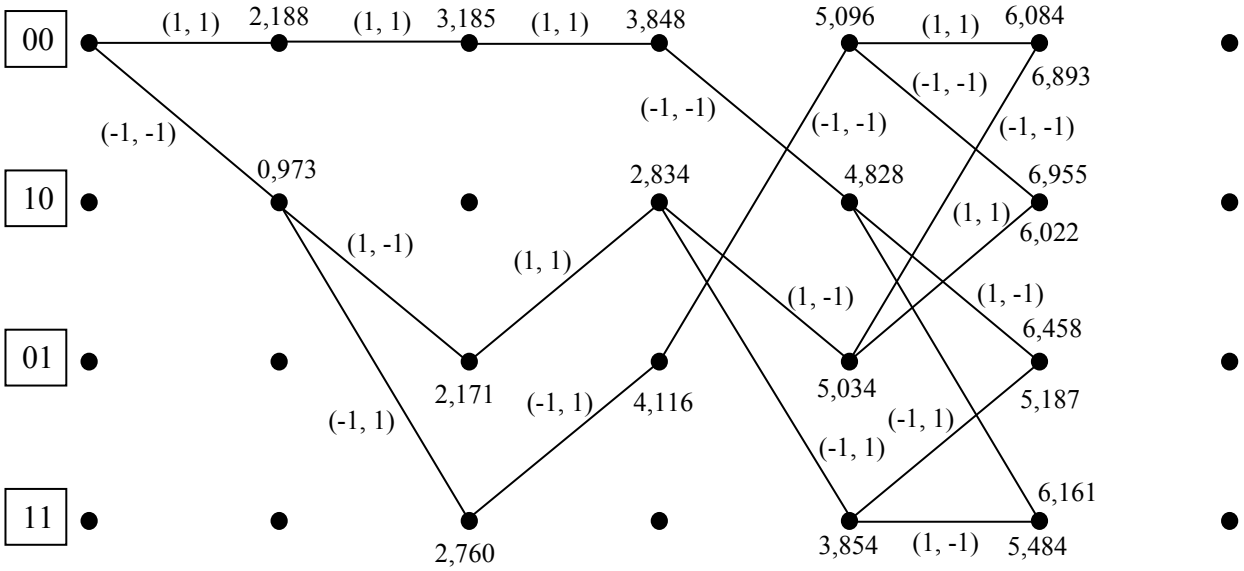
$$\sum_{00-00} = 3{,}848 + 2{,}218 = 6{,}066 \qquad \sum_{01-10} = 4{,}116 + 2{,}218 = 6{,}334$$
$$\sum_{00-10} = 3{,}848 + 0{,}980 = 4{,}828 \qquad \sum_{01-00} = 4{,}116 + 0{,}980 = 5{,}096$$
$$\sum_{10-01} = 2{,}834 + 2{,}200 = 5{,}034 \qquad \sum_{11-11} = 4{,}987 + 2{,}200 = 7{,}187$$
$$\sum_{10-11} = 2{,}834 + 1{,}020 = 3{,}854 \qquad \sum_{11-01} = 4{,}987 + 1{,}020 = 6{,}007$$



De $r_5 = (0{,}20;\ 0{,}42)$ para:

$$S_0 = (1,\ 1) \quad \Rightarrow \quad E_0 = \sqrt{(1-0{,}20)^2 + (1-0{,}42)^2} \qquad E_0 = 0{,}988$$
$$S_3 = (-1,\ -1) \quad \Rightarrow \quad E_3 = \sqrt{(-1-0{,}20)^2 + (-1-0{,}42)^2} \qquad E_3 = 1{,}859$$
$$S_2 = (1,\ -1) \quad \Rightarrow \quad E_2 = \sqrt{(1-0{,}20)^2 + (-1-0{,}42)^2} \qquad E_2 = 1{,}630$$
$$S_1 = (-1,\ 1) \quad \Rightarrow \quad E_1 = \sqrt{(-1-0{,}20)^2 + (1-0{,}42)^2} \qquad E_1 = 1{,}333$$

$$\sum_{00-00} = 5{,}096 + 0{,}988 = 6{,}084 \qquad \sum_{01-10} = 5{,}034 + 0{,}988 = 6{,}022$$
$$\sum_{00-10} = 5{,}096 + 1{,}859 = 6{,}955 \qquad \sum_{01-00} = 5{,}034 + 1{,}859 = 6{,}893$$
$$\sum_{10-01} = 4{,}828 + 1{,}630 = 6{,}458 \qquad \sum_{11-11} = 3{,}854 + 1{,}630 = 5{,}484$$
$$\sum_{10-11} = 4{,}828 + 1{,}333 = 6{,}161 \qquad \sum_{11-01} = 3{,}854 + 1{,}333 = 5{,}187$$



De r_6 = (-0,41; -0,25) para:

$$S_0 = (1,\ 1) \quad \Rightarrow \quad E_0 = \sqrt{(1+0{,}41)^2 + (1+0{,}25)^2} \qquad E_0 = 1{,}884$$

$$S_3 = (-1,\ -1) \quad \Rightarrow \quad E_3 = \sqrt{(-1+0{,}41)^2 + (-1+0{,}25)^2} \qquad E_3 = 0{,}954$$

$$S_2 = (1,\ -1) \quad \Rightarrow \quad E_2 = \sqrt{(1+0{,}41)^2 + (-1+0{,}25)^2} \qquad E_2 = 1{,}597$$

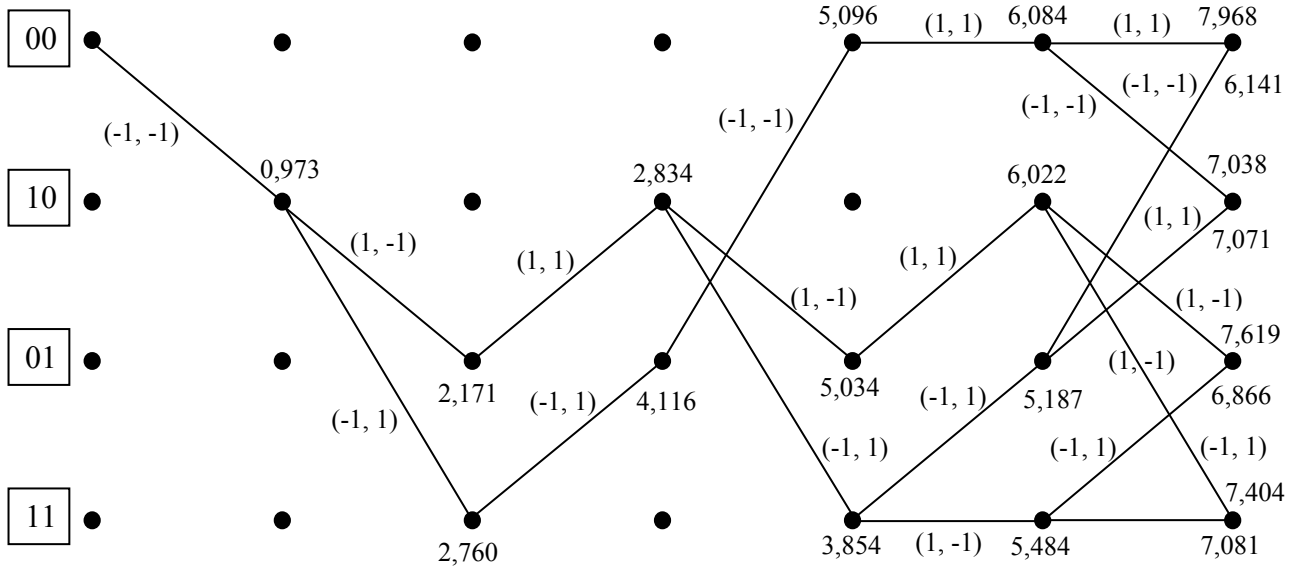
$$S_1 = (-1,\ 1) \quad \Rightarrow \quad E_1 = \sqrt{(-1+0{,}41)^2 + (1+0{,}25)^2} \qquad E_1 = 1{,}382$$

$$\sum\nolimits_{00-00} = 6{,}084 + 1{,}884 = 7{,}968 \qquad \sum\nolimits_{01-10} = 5{,}187 + 1{,}884 = 7{,}071$$

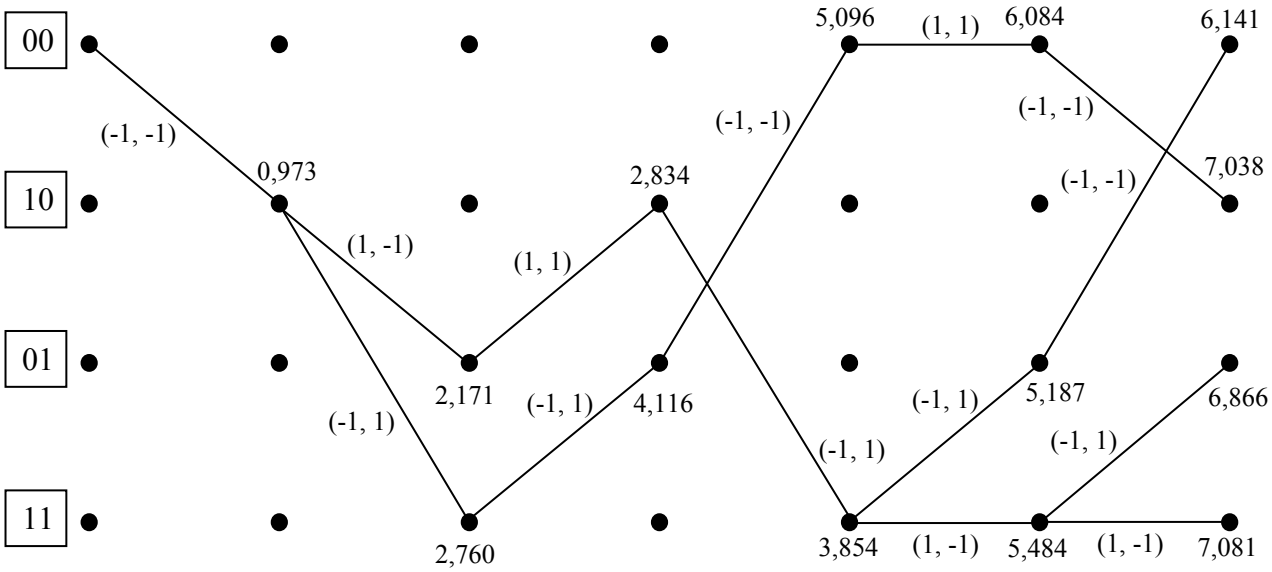
$$\sum\nolimits_{00-10} = 6{,}084 + 0{,}954 = 7{,}038 \qquad \sum\nolimits_{01-00} = 5{,}187 + 0{,}954 = 6{,}141$$

$$\sum\nolimits_{10-01} = 6{,}022 + 1{,}597 = 7{,}619 \qquad \sum\nolimits_{11-11} = 5{,}484 + 1{,}597 = 7{,}081$$

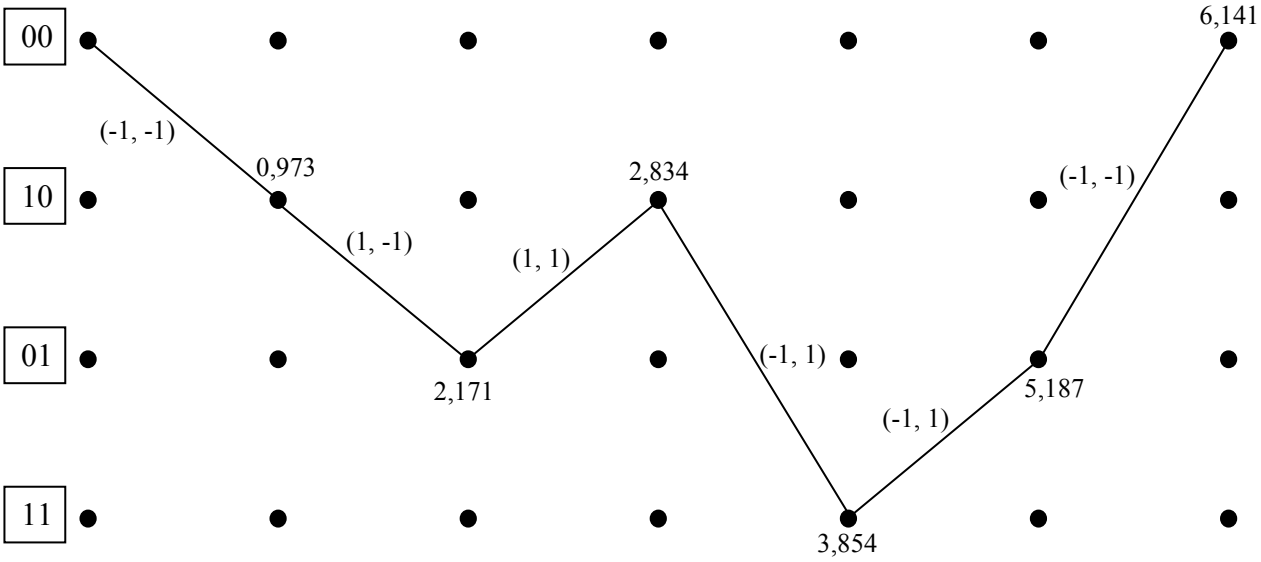
$$\sum\nolimits_{10-11} = 6{,}022 + 1{,}382 = 7{,}404 \qquad \sum\nolimits_{11-01} = 5{,}484 + 1{,}382 = 6{,}866$$



Que resulta nos seguintes caminhos sobreviventes:



Como não há mais pontos a serem decodificados e existe um dos caminhos sobreviventes que retorna ao estado 00 com a menor distância Euclidiana acumulada, pode-se concluir que este é o caminho de máxima verossimilhança entre os sobreviventes, conforme mostrado a seguir.



As coordenadas do caminho de máxima verossimilhança indicam os seguintes pares binários decodificados: 11 10 00 01 01 11.

É importante salientar que a distância Euclidiana não é o único parâmetro utilizado na decodificação suave. O exemplo apresentado é apenas uma das forma de decodificação suave. Qualquer parâmetro que mantenha a proporcionalidade entre as distâncias do símbolo recebido para os símbolos da constelação pode ser utilizado na decodificação suave.

5.4. Capacidade de Correção de Erros dos Códigos Convolucionais [1] [2]

O desempenho de um código convolucional depende do algoritmo de decodificação e das propriedades de distância do código. Neste aspecto a principal característica de um código convolucional é a *distância livre*, denotada por d_{free}. A distância livre de um código convolucional é definida como a distância de Hamming entre duas palavras códigos.

A distância livre pode ser obtida de forma bastante simples a partir do diagrama de estados do codificador convolucional. Considere o diagrama de estados da Figura 5.6. Qualquer seqüência código não zero corresponde a um caminho que começa e termina no estado 00. Pode-se dividir o estado 00 em dois de modo a permitir que o diagrama de estado possa ser transformado em um grafo de fluxo de sinal com uma única entrada e uma única saída, conforme mostrado na Figura 5.16. Um grafo de fluxo de sinal consiste de nós e ramos e opera segundo as seguintes regras:

1. Um ramo multiplica o sinal em seu nó de entrada pela função que caracteriza cada ramo.
2. Um nó com ramos de chegada somam os sinais produzidos por todos aqueles ramos.
3. O sinal em cada nó é aplicado igualmente à todos os ramos de saída daquele nó.
4. A função de transferência do grafo é a relação entre o sinal de saída e o sinal de entrada.

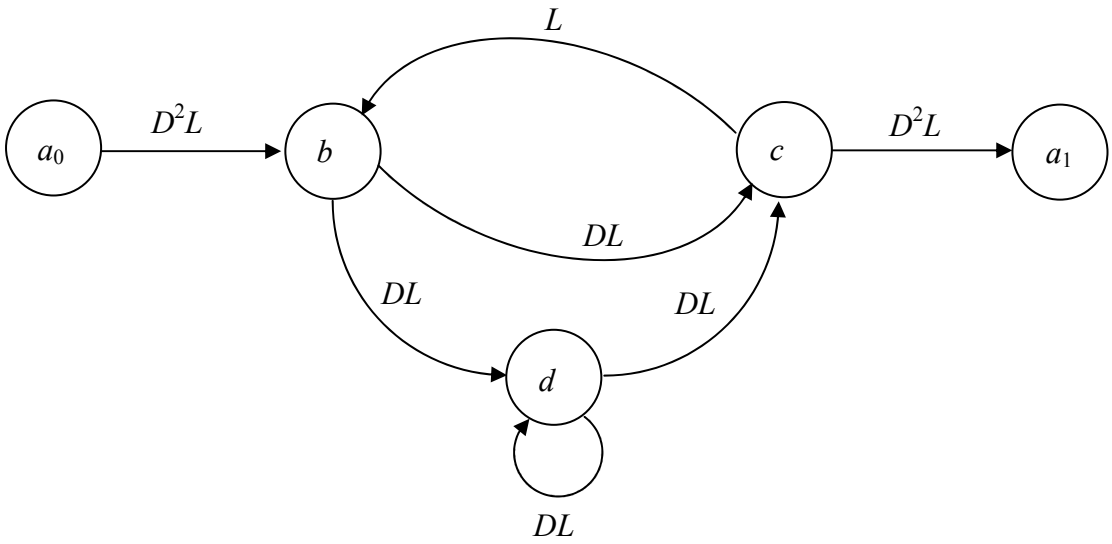


Figura 5.16 – Diagrama de estado modificado.

Retornando à Figura 5.16, note que o expoente de D, sobre cada ramo do grafo, corresponde ao peso de Hamming da saída correspondente ao ramo. O expoente de L é sempre igual a um, uma vez que o comprimento de cada ramo é igual a um. Considere que $T(D, L)$ representa a função de transferência do grafo de fluxo de sinal, com D e L fazendo o papel de variável fantasma.

Segundo as regras 1, 2 e 3, apresentadas acima, para o grafo da Figura 5.16 pode-se obter as seguintes relações entrada-saída:

$$\left.\begin{aligned} b &= D^2La_0 + Lc \\ c &= DLb + DLd \\ d &= DLb + DLd \\ a_1 &= D^2Lc \end{aligned}\right\} \tag{5.2}$$

onde a_0, b, c, d e a_1 são os sinais do nó do grafo. Resolvendo o conjunto de Equações (5.2) para a relação a_1/a_0, encontra-se a função de transferência do grafo dada por

$$T(d,L) = \frac{D^5L^3}{1 - DL(1+L)} \tag{5.3}$$

Usando a expansão binomial, (5.3) torna-se

$$T(d,L) = D^5L^3\sum_{i=0}^{\infty}(DL(1+L))^i \tag{5.4}$$

Fazendo $L = 1$, obtém-se a função de transferência da distância na forma de uma série de potência como

$$T(d, L) = D^5 + 2D^6 + 4D^7 + \cdots \tag{5.5}$$

Uma vez que a distância livre é a mínima distância de Hamming entre duas palavras código quaisquer e a função de transferência $T(D, 1)$ enumera o número de palavras códigos que possuem uma dada distância umas da outras, conclui-se que o expoente do primeiro termo na expansão de $T(D, 1)$ define a distância livre. Conseqüentemente o código convolucional correspondente ao diagrama de estados apresentado na Figura 2.6 tem distância livre $d_{\text{free}} = 5$.

Um código convolucional com distância livre igual a d_{free} pode corrigir t erros de acordo com a relação

$$t = \left\lfloor \frac{d_{free} - 1}{2} \right\rfloor. \tag{5.6}$$

Pode-se concluir que a capacidade de correção de um código cuja $d_{\text{free}} = 5$ é igual a dois ou talvez três erros. Essa conclusão, entretanto leva a uma questão importante: para que comprimento da seqüência recebida pode-se corrigir dois ou talvez três erros? Infelizmente não existe uma resposta exata para essa questão já que a capacidade de correção está associada à forma como os erros estão distribuídos na seqüência (mais próximos ou mais afastados uns dos outros).

Mais uma vez, pode-se dizer que, observações permitem concluir que a capacidade de correção de t erros ocorre, geralmente, dentro de algumas *extensões de influência K*, onde "algumas" aqui significa 3 a 5. Logo, voltando ao código em questão, em que k = 3, pode-se dizer então que 2 ou 3 erros podem ser corrigidos em uma seqüência recebida cujo comprimento pode variar de 9 a 15 bits, dependendo da distribuição dos erros na seqüência.

5.5. CÓDIGOS CONVOLUCIONAIS CATASTRÓFICOS [1] [2]

Quando se usa a função de transferência $T(D, 1)$ para a determinação da distância livre fica implícito que a série de potência que a representa é *convergente*, i.e., a soma possui um valor finito. No caso do exemplo apresentado, pode-se demonstrar que, para o codificador em questão, a série é de fato convergente. Entretanto, para outros codificadores convolucionais, não existe nenhuma garantia de que $T(D, 1)$ é sempre convergente. Quando $T(D, 1)$ é *não-convergente*, um número finito de erros introduzidos na transmissão provoca uma seqüência infinita de erros de decodificação. Isso significa uma propagação catastrófica de erros e, neste caso, o código é chamado de um *código catastrófico*.

A condição para a propagação catastrófica de erros está nos codificadores em que os polinômios geradores possuem um polinômio fator comum de grau 1, pelo menos. Como exemplo, considere o codificador apresentado na Figura 5.17, cujo diagrama de estados modificado é apresentado na Figura 5.18. Este codificador apresenta os seguintes polinômios geradores

$$\mathbf{g}_1(D) = 1 + D$$
$$\mathbf{g}_2(D) = 1 + D^2.$$

Ambos os polinômios possuem um polinômio fator comum que é $1 + D$, pois

$$1 + D^2 = (1 + D)(1 + D).$$

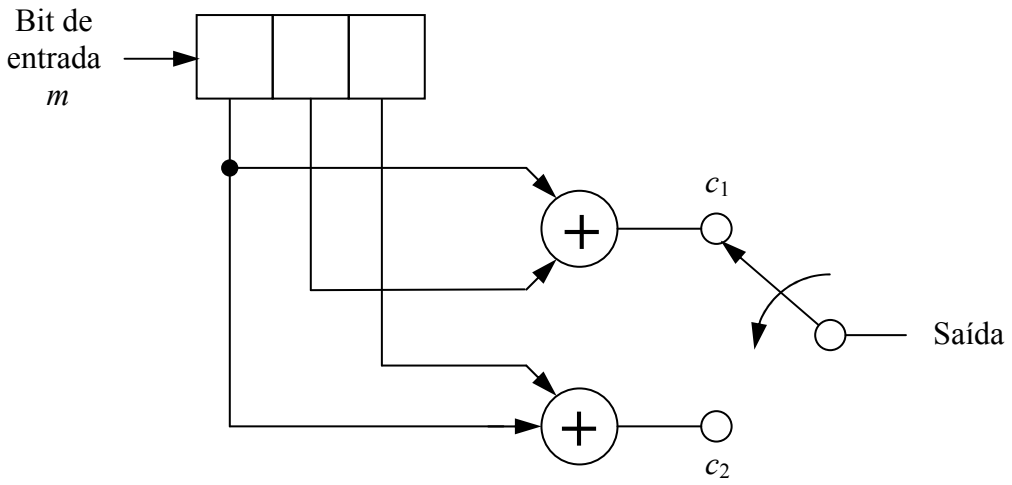


Figura 5.17 – Codificador convolucional catastrófico.

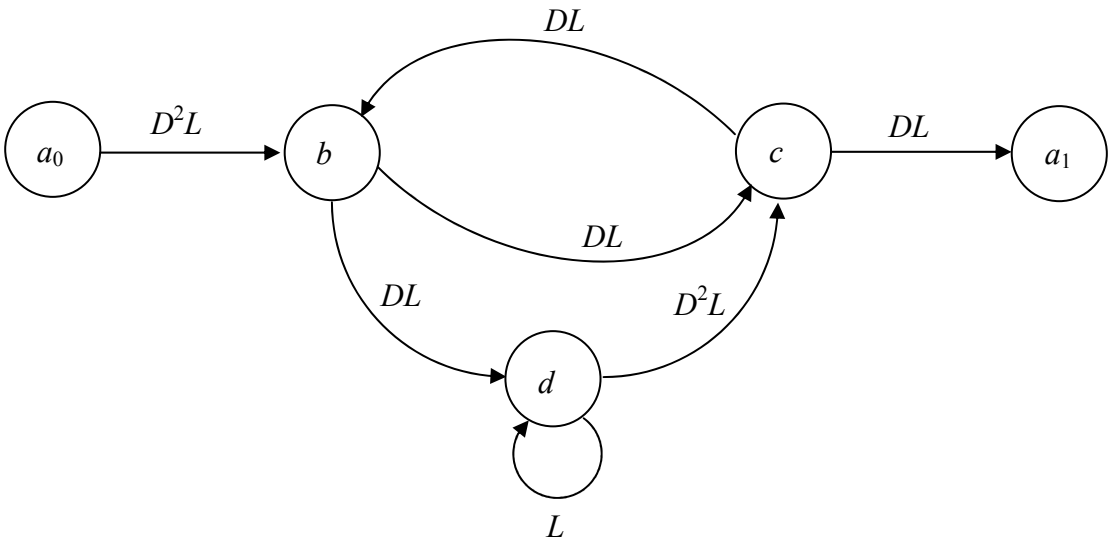


Figura 5.18 – Diagrama de estado modificado para o codificador da Figura 5.17.

Em termos de diagrama de estados para códigos com qualquer taxa, erros catastróficos podem ocorrer se, e somente se, todos os laços fechados do diagrama possuem peso zero.

5.6. CÓDIGOS CONVOLUCIONAIS SISTEMÁTICOS [1][2]

Uma garantia de que um código convolucional é não-catastrófico se dá quando o codificador gera um *código convolucional sistemático.* Um exemplo de codificador sistemático está mostrado na Figura 5.19.

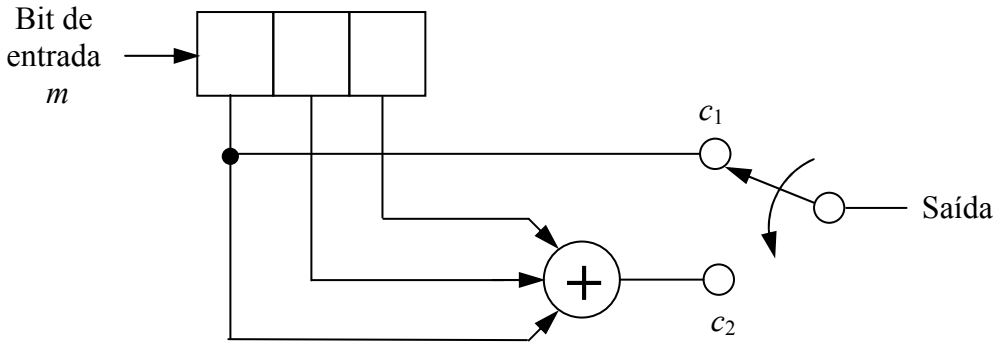


Figura 5.19 – Codificador convolucional sistemático.

Infelizmente, para códigos com a mesma *extensão de influência K* e mesma taxa os valores de distância livre obtidos são, geralmente, menores do que aqueles obtidos pelos *códigos convolucionais não-sistemáticos*, como mostra a Tabela 5.7.

Tabela 5.7 – Máximas distâncias livres obtidas com códigos convolucionais sistemáticos e não-sistemáticos de taxa 1/2.

Extensão de influência *K*	Sistemático	Não-sistemático
2	3	3
3	4	5
4	4	6
5	5	7
6	6	8
7	6	10
8	7	10

Entretanto é importante ressaltar que apenas uma pequena fração dos códigos não sistemáticos (excluindo aqueles onde todos os somadores possuem um número par de conexões) são catastróficos.

5.7. MELHORES CÓDIGOS CONVOLUCIONAIS CONHECIDOS [1]

Conforme apresentado, códigos convolucionais devem apresentar boas características de distância livre para uma dada taxa de codificação e *extensão de influência K*. Uma lista dos melhores códigos convolucionais de taxa 1/2, K = 3 até 9 e taxa 1/3, K = 3 até 8, são apresentados na Tabela 5.8 e 5.9 respectivamente.

Tabela 5.8 – Códigos convolucionais de taxa 1/2, K = 3 até 9.

Taxa	Extensão de influência K	Distância livre	Vetores conexão
1/2	3	5	111 101
	4	6	1111 1011
	5	7	10111 11001
	6	8	101111 110101
	7	10	1001111 1101101
	8	10	10011111 11100101
	9	12	110101111 100011101

Tabela 5.9 – Códigos convolucionais de taxa 1/3, K = 3 até 9.

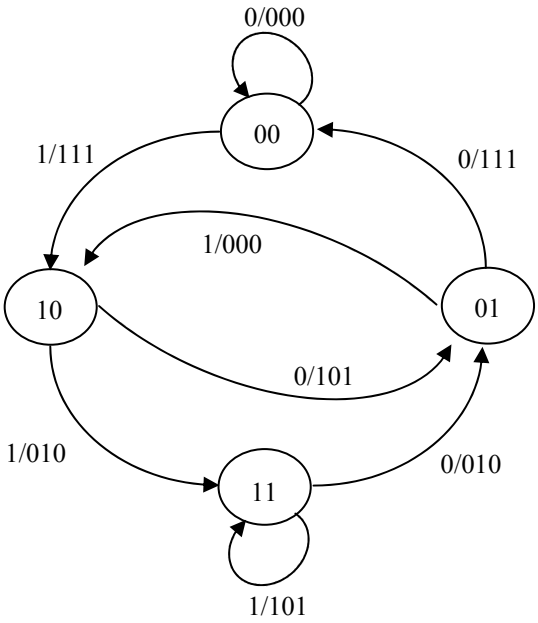
Taxa	Extensão de influência K	Distância livre	Vetores conexão
1/3	3	8	111 111 101
	4	10	1111 1011 1101
	5	12	11111 11011 10101
	6	13	101110 110101 111001
	7	15	1001111 1010111 1101101
	8	16	11101111 10011011 10101001

5.7. EXERCÍCIOS

1. Considere o codificador convolucional representado pelos vetores conexão 1101 e 1111. Pede-se:

   a) Codificar a mensagem 10101.
   b) Determinar o seu diagrama de estados.
   c) Determinar a capacidade de correção de erros deste código.

2. Considere o diagrama de estados de um codificador convolucional apresentado abaixo.



   a) Desenhe o codificador.
   b) Determine a capacidade de correção de erros deste código.
   c) Codifique a mensagem 10101.
   d) Decodifique a seqüência recebida 10100100011010101010101 utilizando o algoritmo de Viterbi.

3. Considere o esquema de codificação e de modulação apresentado na Figura 5.11. Admita que os seguintes pares de coordenadas tenham sido recebidos: (-0,92; -0,89), (0,71; -0,12), (-0,13; -0,02), (0,01; 0,11), (-0,95; 0,85) e (-0,88; -0,99). Admita ainda que o processo de codificação foi iniciado no estado "00" e terminou no estado "00". Fazendo uso do algoritmo de Viterbi pede-se:

   a) A decisão e decodificação abrupta da seqüência recebida.
   b) A decodificação suave da seqüência recebida.
   c) Comparar os dois resultados, escolher a seqüência decodificada e justificar a escolha.

* * *

5.8. REFERÊNCIAS BIBLIOGRÁFICAS

[1] SKLAR, Bernard. Channel coding. In: *Digital communications*: fundamentals and applications. 2. ed. Upper Saddle River, New Jersey: Prentice Hall, 2001. p. 304-429. ISBN 0130847887.

[2] HAYKIN, Simon. Error control coding. In: *Systems communications*: 4. ed. New York: John Wiley & Sons, 2001. p. 626-696. ISBN 0471178691.

[3] LIN, S.; COSTELLO, D. J. *Error control coding*: fundamentals and applications. Englewood Cliffs, New Jersey: Prentice Hall, 1983. ISBN 013283796X.