9 Codificação de Canal: Códigos de Bloco Lineares

Em capítulos anteriores estudámos com alguma profundidade dois dos problemas mais importantes associados ao desenho de sistemas de comunicações: o problema da codificação de fonte, e o problema da transmissão de informação através de canais ruidosos. O primeiro destes problemas aborda a questão da representação eficiente dos símbolos (ou de sequências de símbolos) gerados pela fonte, recorrendo a códigos univocamente descodificáveis cujo comprimento médio seja o menor possível. Para o caso de fontes discretas sem memória, verificámos que o comprimento médio mínimo daqueles códigos é determinado pela entropia da fonte, a que corresponde uma representação sem redundância. O problema da transmissão digital através de canais ruidosos foi abordado pela via da definição de técnicas de modulação e do desenho dos respectivos receptores tendo como objectivo a minimização da probabilidade de erro de transmissão. Verificámos também a existência de um compromisso fundamental entre a probabilidade de erro (desempenho do sistema de transmissão) e a largura de banda de transmissão. Este compromisso traduz-se numa medida da capacidade do canal, isto é, o valor máximo da taxa de transmissão de informação que garante, pelo uso do código adequado, uma probabilidade de erro arbitrariamente pequena.

Consideremos a sequência de bits

$$b_0 \quad b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad b_6 \quad b_7 \tag{9.1}$$

com base na qual formamos a palavra binária

$$b_0 \quad b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad b_6 \quad b_7 \mid p \tag{9.2}$$

onde o bit p, designado por bit de paridade,[1]

$$p = b_0 \oplus b_1 \oplus \cdots \oplus b_7 = \begin{cases} 1, & \text{se nº de 1's} = \text{ímpar} \\ 0, & \text{se nº de 1's} = \text{par} \end{cases} \tag{9.3}$$

Suponhamos que a sequência (9.2) é a palavra de código transmitida quando a fonte gera o bloco de 8 bits em (9.1). Tendo em conta (9.3), podemos verificar que todas as palavras de código têm um nº par de 1's, isto é, têm paridade par. Isto quer dizer que qualquer palavra binária de 9 bits com paridade ímpar não faz parte deste código. Assim, se a transmissão de uma palavra de código envolver um nº ímpar de bits errados, então a paridade da palavra recebida é ímpar, e os erros de transmissão são detectados. Este efeito resultou da inclusão do bit de paridade p, o qual, sendo determinado pelos bits $b_0, \ldots, b_7$, não introduz informação adicional. A inclusão nos símbolos transmitidos detes tipo de redundância controlada é a chave para a construção de códigos de canal capazes de detectar e/ou corrigir eventuais erros de transmissão.

Basicamente, existem dois tipos de técnicas de controlo dos erros de transmissão. Uma, baseada em códigos correctores de erro, é designada na literatura por controlo de erro sem retroacção (FEC)[2]. A outra estratégia exige a retransmissão dos blocos de dados onde tenham

[1] O símbolo $\oplus$ designa o operador adição mod 2, o qual, como é sabido, é equivalente ao operador "ou exclusivo".
[2] FEC – Forward Error Control.

sido detectados erro de transmissão através de pedidos destinados ao emissor e gerados automaticamente pelo receptor (ARQ)[3]. Ao contrário das técnicas do tipo FEC, os métodos ARQ, embora recorrendo a códigos menos complexos, envolvem normalmente uma utilização menos eficiente do canal de transmissão.

Os tipos de códigos de canal normalmente usados podem agrupar-se em duas grandes categorias: os códigos de bloco e os códigos convolucionais. Ao contrário dos primeiros, estes últimos são realizados recorrendo a codificadores com memória. Na secção seguinte, iremos introduzir os códigos de bloco lineares.

9.1 Códigos de Bloco Lineares

Seja $\{m_0, m_1, \ldots, m_{k-1}\}$ um bloco arbitrário de k bits gerados pela fonte. Tipicamente, o codificador de bloco usa estes k bits para gerar uma palavra de código com $n > k$ bits, acrescentando n-k bits de controlo. A palavra de código assim construída é constituída pelos símbolos binários

$$x_i = \begin{cases} b_i, & i = 0,1,\ldots,n-k-1 \\ m_i, & i = n-k, n-k+1,\ldots,n-1 \end{cases} \tag{9.4}$$

e tem a estrutura ilustrada na Figura 9.1. Este é um código (n,k) e tem uma taxa de codificação definida por

$$R = \frac{k}{n}. \tag{9.5}$$

$$b_0 \quad b_1 \quad \cdots \quad b_{n-k-1} \mid m_0 \quad m_1 \quad \cdots \quad m_{k-1}$$

Figura 9.1: Palavra de código (n,k)

No caso de um código de bloco linear, os bits de paridade $b_0, b_1, \ldots, b_{n-k-1}$ dependem linearmente (numa aritmética binária mod 2) dos bits da mensagem $m_0, m_1, \ldots, m_{k-1}$. Ou seja, definindo os vectores linha

$$\mathbf{b} = [b_0 \quad b_1 \quad \cdots \quad b_{n-k-1}] \tag{9.6}$$

e

$$\mathbf{m} = [m_0 \quad m_1 \quad \cdots \quad m_{k-1}], \tag{9.7}$$

podemos escrever

$$\mathbf{b} = \mathbf{mP}, \tag{9.8}$$

[3] ARQ – Automatic ReQuest for retransmission.

onde **P** é uma matriz binária $(k \times n-k)$ que determina o código. Portanto, sendo

$$\mathbf{x} = [\mathbf{b} \vdots \mathbf{m}], \tag{9.9}$$

de (9.8) resulta

$$\mathbf{x} = \mathbf{m}[\mathbf{P} \vdots \mathbf{I}_k], \tag{9.10}$$

onde $\mathbf{I}_k$ é a matriz identidade de dimensão $(k \times k)$. Definindo a matriz geradora do código

$$\mathbf{G} = [\mathbf{P} \vdots \mathbf{I}_k], \tag{9.11}$$

de dimensão $(k \times n)$, usando (9.10) temos

$$\mathbf{x} = \mathbf{mG}. \tag{9.12}$$

É fácil verificar que o código formado pelas palavras **x**, geradas pela matriz **G** a partir das 2^k mensagens **m**, é um código linear. Com efeito, sendo $\mathbf{x}_i = \mathbf{m}_i\mathbf{G}$ e $\mathbf{x}_j = \mathbf{m}_j\mathbf{G}$ palavras do código, então $\mathbf{x}_i \oplus \mathbf{x}_j = \mathbf{m}_i\mathbf{G} \oplus \mathbf{m}_j\mathbf{G} = (\mathbf{m}_i \oplus \mathbf{m}_j)\mathbf{G}$; como $\mathbf{m}_i \oplus \mathbf{m}_j$ é necessariamente uma mensagem, $\mathbf{x}_i \oplus \mathbf{x}_j$ é uma palavra do código.

Definindo a matriz de verificação de paridade

$$\mathbf{H} = [\mathbf{I}_{n-k} \vdots \mathbf{P}^T], \tag{9.13}$$

de dimensão $(n-k \times n)$, e usando (9.11), verifica-se que[4]

$$\mathbf{HG}^T = [\mathbf{I}_{n-k} \vdots \mathbf{P}^T]\begin{bmatrix}\mathbf{P}^T \\ \cdots \\ \mathbf{I}_k\end{bmatrix} = \mathbf{P}^T \oplus \mathbf{P}^T = \mathbf{0}. \tag{9.14}$$

Assim sendo, de (9.12) vem

$$\mathbf{x}^T = \mathbf{G}^T\mathbf{m}^T,$$

o que, tendo em conta (9.14), conduz a

$$\mathbf{Hx}^T = \mathbf{0} \Leftrightarrow \mathbf{xH}^T = \mathbf{0}. \tag{9.15}$$

Esta é uma condição necessária e suficiente para que **x** seja uma palavra do código (n,k) gerado pela matriz **G**. No entanto, a verificação de (9.15) à saída do canal ruidoso não significa necessariamente que não tenham ocorrido erros de transmissão. Com efeito, se for transmitida a palavra de código **x**, então a palavra **y** recebida é, em geral,

$$\mathbf{y} = \mathbf{x} + \mathbf{e}, \tag{9.16}$$

[4] Mais uma vez se chama a atenção para o facto de estarmos a usar aritmética binária mod 2.

onde

$$\mathbf{e} = [e_0 \quad \cdots \quad e_i \quad \cdots \quad e_{n-1}],\ e_i = \begin{cases} 1, & \text{se houver erro no bit i} \\ 0, & \text{caso contrário} \end{cases}, \tag{9.17}$$

é o vector de erro. Se usarmos (9.16) em (9.15), obtemos

$$\mathbf{y}\mathbf{H}^T = \mathbf{x}\mathbf{H}^T + \mathbf{e}\mathbf{H}^T = \mathbf{e}\mathbf{H}^T \begin{cases} = \mathbf{0}, & \text{se } \mathbf{e} \text{ for palavra do código} \\ \neq \mathbf{0}, & \text{caso contrário} \end{cases}$$

o que significa que sendo o vector de erro uma palavra do código, a palavra y, recebida com erros de transmissão, cumpre o teste de verificação de paridade (9.15), e os erros de transmissão não são detectados.

9.1.1 Descodificação pelo Sindroma

Para proceder à descodificação, isto é, à detecção e/ou correcção de erros de transmissão, o descodificador começa por calcular o sindroma da palavra recebida **y**, isto é, o vector binário

$$\mathbf{s} = \mathbf{y}\mathbf{H}^T \tag{9.18}$$

de dimensão $n-k$. Tendo em conta (9.16) e (9.15), verificamos que o sindroma só depende do padrão de erros, ou seja,

$$\mathbf{s} = \mathbf{e}\mathbf{H}^T. \tag{9.19}$$

Por outro lado, todos os padrões de erro que diferem entre si de uma palavra de código têm o mesmo sindroma. Com efeito, dado um vector **e** que verifique (9.19), então todos os vectores de erro

$$\mathbf{e}_i = \mathbf{e} \oplus \mathbf{x}_i,\ i = 0,1,\ldots,2^k - 1, \tag{9.20}$$

onde os $\mathbf{x}_i$, $i = 0,1,\ldots,2^k - 1$, são todas as palavras do código, verificam também (9.19). Assim, define-se o *coset* do padrão de erros **e** como sendo o conjunto dos vectores de erro definido em (9.20) que têm o mesmo sindroma (9.19). Uma vez que um código de bloco linear (n,k) tem 2^k palavras admissíveis, num total de 2^n palavras binárias de comprimento n, conclui-se que existem 2^{n-k} *cosets*, isto é, 2^{n-k} sindromas distintos.

O sindroma contém alguma informação sobre o correspondente padrão de erros, embora geralmente insuficiente para o identificar sem ambiguidade. Se assim fosse, qualquer padrão de erros poderia ser corrigido. De qualquer modo, o conhecimento do sindroma **s** reduz o espaço de busca de uma 2^n dimensão para 2^k. Uma vez calculado **s**, o descodificador deve escolher o elemento do respectivo *coset* que optimize um determinado critério. Por exemplo, a respectiva probabilidade de ocorrência. Para valores relativamente baixos da probabilidade

de ocorrência de erros de transmissão, o padrão de erros mais provável corresponde àquele que tem menos 1's, isto é, aquele cujo peso

$$w(\mathbf{e}) = \sum_{i=0}^{n-1} e_i \tag{9.21}$$

é mínimo.

Algoritmo de descodificação de máxima verosimilhança

Dada a palavra recebida **y**:

1. Calcular o sindroma $\mathbf{s} = \mathbf{y}\mathbf{H}^T$
   a) $\mathbf{s} = \mathbf{0} \Leftrightarrow$ **y** é uma palavra do código $\Rightarrow \mathbf{y}_o = \mathbf{y}$
   b) $\mathbf{s} \neq \mathbf{0} \Rightarrow$ executar passo 2.
2. Calcular o *coset* de **y**, $\{\mathbf{e}_i = \mathbf{y} \oplus \mathbf{x}_i \ , \ i = 0,1,\ldots,2^k-1\}$, escolher o padrão $\mathbf{e}_o$ de menor peso, e executar o passo 3.
3. Construir a palavra corrigida $\mathbf{y}_o = \mathbf{y} \oplus \mathbf{e}_o$.

Comentário: sendo $\mathbf{e}_0 = \mathbf{y} \oplus \mathbf{x}_o$, com $\mathbf{x}_o\mathbf{H}^T = \mathbf{0}$, tem-se

$$\begin{aligned} \mathbf{y}_o &= \mathbf{y} \oplus \mathbf{e}_o \\ &= \mathbf{y} \oplus \mathbf{y} \oplus \mathbf{x}_o = \mathbf{x}_o \ , \end{aligned} \tag{9.22}$$

isto é, a saída do descodificador é a palavra de código que difere da palavra recebida num número mínimo de posições (correspondentes às posições dos 1's em $\mathbf{e}_o$). Ainda por outras palavras, é a palavra de código para a qual $\mathbf{y} \oplus \mathbf{x}_o$ tem peso mínimo.

9.1.2 Distância de Hamming

Def. 9.1- Distância de Hamming. Sejam $\mathbf{x}_i$ e $\mathbf{x}_j$ duas palavras binárias de comprimento n. A distância de Hamming entre $\mathbf{x}_i$ e $\mathbf{x}_j$,

$$d(\mathbf{x}_i, \mathbf{x}_j) = w(\mathbf{x}_i \oplus \mathbf{x}_j), \tag{9.23}$$

onde w é o peso definido em (9.21), mede o número de bits distintos em $\mathbf{x}_i$ e $\mathbf{x}_j$.

Usando esta definição em (9.22), obtemos

$$d(\mathbf{y}_o, \mathbf{y}) = d(\mathbf{x}_o, \mathbf{y}) = w(\mathbf{e}_o).$$

Portanto, o critério de máxima verosimilhança usado para desenhar o descodificador é equivalente a minimizar a distância de Hamming entre a palavra recebida e cada palavra admissível do código.

Def. 9.2- Distância mínima do código. A distância mínima do código é dada por

$$d_{min} = \min_{i,j} d(\mathbf{x}_i, \mathbf{x}_j),\ i, j = 0,1,\ldots,2^k - 1, \qquad (9.24)$$

ou seja, é o valor mínimo da distância de Hamming entre todos os pares de palavras do código.

Consideremos então um padrão de erros $\mathbf{e}_t$ com peso t correspondendo, portanto, a t bits errados. Sejam $\mathbf{x}_i$ e $\mathbf{x}_j$ duas palavras do código tais que $d(\mathbf{x}_i, \mathbf{x}_j) = d_{min}$. Naturalmente, $\mathbf{x}_i \oplus \mathbf{e}_t$ está mais próximo de $\mathbf{x}_i$ do que de $\mathbf{x}_j$, e $\mathbf{x}_j \oplus \mathbf{e}_t$ está mais próximo de $\mathbf{x}_j$ do que de $\mathbf{x}_i$ se e só se

$$d_{min} \geq 2t + 1. \qquad (9.25)$$

Como se mostra na Figura 9.2, qualquer padrão de erro com peso inferior a t conduz a uma palavra recebida que cai num dos círculos de raio t. Verificando-se (9.25), a palavra recebida só pode resultar da transmissão da palavra de código que está no centro daquele círculo. Portanto, (9.25) constitui uma condição necessária e suficiente para que o código tenha capacidade de corrigir padrões de erro com peso não superior a t.

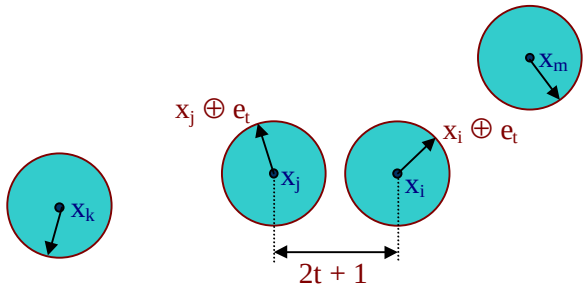

Figura 9.2: Distância mínima e capacidade correctora do código

Seguindo a mesma linha de raciocínio, podemos concluir que o código é capaz de detectar todos os padrões de erro com peso não superior a t_0 se e só se

$$d_{min} \geq t_0 + 1. \qquad (9.26)$$

9.2 Códigos de Hamming

Os códigos de Hamming constituem uma família de códigos lineares (n, k) que verificam

$$\text{para } m \geq 3: \quad n = 2^m - 1, \quad k = n - m. \qquad (9.27)$$

No caso em que $m = 3$ obtém-se um código $(7,4)$. Os códigos de Hamming são desenhados por forma a garantir que, independentemente do valor de m, $d_{min} = 3$. Portanto, e tendo em conta as

condições (9.25) e (9.26), estes códigos têm capacidade para corrigir até um bit errado por bloco $(t=1)$ e de detectar até dois bits errados por bloco $(t_0=2)$.

Uma vez que as matrizes geradora **G** e de verificação de paridade **H** são directamente relacionáveis, é indiferente definir o código pela especificação de **G** ou de **H**. Por outro lado, qualquer que seja a palavra **x** do código, verifica-se

$$\mathbf{x}\mathbf{H}^{\mathrm{T}} = \mathbf{x}\left[\mathbf{I}_{n-k} \;\vdots\; \mathbf{P}^{\mathrm{T}}\right]^{\mathrm{T}} = \mathbf{0}\,. \qquad (9.28)$$

De acordo com a Def. 9.2, eq. (9.24), d_{min} pode ser interpretada como o valor mínimo dos pesos de todas as palavras do código, excepto $w(\mathbf{x}=\mathbf{0})=0$[5]. À luz da condição (9.28), concluímos que d_{min} coincide com o número mínimo de colunas de **H** cuja soma mod 2 dá um vector nulo. Para que tal se verifique, dada a estrutura da matriz **H**, basta impor que qualquer coluna de $\mathbf{P}^{\mathrm{T}}$ tenha dois ou mais 1's. Por exemplo, para um código de Hamming $(7,4)$

$$\mathbf{H} = \left[\begin{array}{ccc:cccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}\right]. \qquad (9.29)$$

A estrutura dos códigos de Hamming pode ainda ser explorada para levar à prática um método de descodificação bastante eficiente. Com efeito, e como se ilustra a seguir,

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

$$1 \quad 2 \quad 4 \quad 3 \quad 6 \quad 7 \quad 5$$

as colunas de **H** codificam em bínário natural os dígitos de 1 a 7. Podemos assim construir a seguinte tabela

dígito	1	2	3	4	5	6	7
coluna de **H**	1	2	4	3	7	5	6

Tabela 9.1: Colunas de **H** onde estão codificados os dígitos de 1 a 7

Suponhamos que a palavra transmitida foi $\mathbf{x}=[0\;\;1\;\;1\,|\,1\;\;0\;\;0\;\;1]$, onde o bloco da direita corresponde à mensagem $\mathbf{m}=[1\;\;0\;\;0\;\;1]$, e que $\mathbf{y}=[0\;\;1\;\;1\,|\,1\;\;1\;\;0\;\;1]$ é a palavra recebida. O vector de erro é $\mathbf{e}=[0\;\;0\;\;0\,|\,0\;\;1\;\;0\;\;0]$ e o sindroma resultante é o vector $\mathbf{s}=[0\;\;1\;\;1]$ (seja qual for o padrão de erro singular com 1 no bit i, o sindroma é sempre a coluna i da matriz **H**). O conteúdo deste vector constitui o código binário natural para o dígito 6. Se consultarmos a Tabela 9.1, verificamos que este dígito identifica a coluna 5 da matriz **H**. Daqui se conclui que basta trocar o 5º bit de **y** para recuperar **x** e identificar a mensagem **m**.

[5] Faz-se notar que a palavra $\mathbf{x}=\mathbf{0}$ é sempre uma palavra admissível de qualquer código de bloco linear.