Avaliação Códigos de Bloco

Sistemas de Comunicação II

Gabriel Luiz Espindola Pedro

28 de Outubro de 2024

Sumário

1 Questões

Considere o **código de Hamming estendido** $(8, 4)$, obtido a apartir do código de Hamming $(7, 4)$, adicionando um "bit de paridade global" no final de cada palavra de código. (Dessa forma, todas as palavras-código terão um número par de bits 1.)

a) Determine uma matriz geradora G para o código.

Sabemos que a palavra código do código de Hamming $(7, 4)$ é dada por

$$v = [v_1 \;\; v_2 \;\; v_3 \;\; v_4 \;\; v_5 \;\; v_6 \;\; v_7] = [u_1 \;\; u_2 \;\; u_3 \;\; u_4 \;\; p_1 \;\; p_2 \;\; p_3] \tag{1}$$

Onde $u = [u_1 \;\; u_2 \;\; u_3 \;\; u_4]$ é a palavra de informação e $p = [p_1 \;\; p_2 \;\; p_3]$ é o vetor de paridade. Onde o vetor de paridade é obtido da seguinte maneira:

$$\begin{cases} p_1 = u_1 + u_2 + u_4 \\ p_2 = u_1 + u_3 + u_4 \\ p_3 = u_2 + u_3 + u_4 \end{cases} \tag{2}$$

Sabendo que a matriz geradora G é dada por

$$v = uG \tag{3}$$

Podemos escrever a matriz geradora G do código de Hamming $(7, 4)$ como

$$\begin{aligned} v &= [u_1 \;\; u_2 \;\; u_3 \;\; u_4 \;\; u_1 + u_2 + u_4 \;\; u_1 + u_3 + u_4 \;\; u_2 + u_3 + u_4] = uG \\ &= [u_1 \;\; u_2 \;\; u_3 \;\; u_4] \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = uG \\ \therefore G &= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{aligned} \tag{4}$$

A partir desta análise do código de Hamming $(7, 4)$, devemos alterá-lo de modo a adiconar um bit de paridade global, ou seja:

$$v = [v_1 \;\; v_2 \;\; v_3 \;\; v_4 \;\; v_5 \;\; v_6 \;\; v_7] = [u_1 \;\; u_2 \;\; u_3 \;\; u_4 \;\; p_1 \;\; p_2 \;\; p_3 \;\; p_4] \tag{5}$$

onde

$$\begin{cases} p_1 = u_1 + u_2 + u_4 \\ p_2 = u_1 + u_3 + u_4 \\ p_3 = u_2 + u_3 + u_4 \\ p_4 = v_1 + v_2 + v_3 + v_4 + v_5 + v_6 + v_7 \end{cases} \tag{6}$$

Ou seja

$$
\begin{aligned}
p_4 &= (u_1) + (u_2) + (u_3) + (u_4) \\
&\quad +(u_1 + u_2 + u_4) + (u_1 + u_3 + u_4) + (u_2 + u_3 + u_4) \\
&\quad = u_1 + u_2 + u_3
\end{aligned} \tag{7}
$$

Com a definição de p_4 em termos da palavra de mensagem u, podemos escrever a matriz geradora G do código de Hamming estendido $(8, 4)$ como

$$
\begin{aligned}
v &= [u_1 \;\; u_2 \;\; u_3 \;\; u_4 \;\; u_1 + u_2 + u_4 \;\; u_1 + u_3 + u_4 \;\; u_2 + u_3 + u_4 \;\; u_1 + u_2 + u_3] = uG \\
&= [u_1 \;\; u_2 \;\; u_3 \;\; u_4] \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} = uG
\end{aligned} \tag{8}
$$

$$
\therefore G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}
$$

b) Construa uma tabela mensagem $\rightarrow$ palavra-código

Utilizando a linguagem de programação `Python`, podemos construir uma tabela de mapeamento de mensagens para palavras-código do código de Hamming estendido $(8, 4)$.

u	v
0000	00000000
0001	00011110
0010	00100111
0011	00111001
0100	01001011
0101	01010101
0110	01101100
0111	01110010
1000	10001101
1001	10010011
1010	10101010
1011	10110100
1100	11000110
1101	11011000
1110	11100001
1111	1111111

c) Determine a distância mínima e a distribuição de peso das palavras-código

Sabendo que o código de Hamming estendido $(8, 4)$ é um código linear podemos obter a distância mínima do código a partir da distribuição de pesos das palavras códigos. A distribuição de pesos das palavras códigos é dada por

Palavra-código	Peso
00000000	0
00011110	4
00100111	4
00111001	4
01001011	4
01010101	4
01101100	4
01110010	4
10001101	4
10010011	4
10101010	4
10110100	4
11000110	4
11011000	4
11100001	4
11111111	8

i	0	1	2	3	4	5	6	7	8
A_i	1	0	0	0	14	0	0	0	1

Com a distribuição de peso das palavras códigos, podemos determinar a distância mínima do código de Hamming estendido $(8, 4)$ como sendo $d_{\min} = 4$.

d) Determine a matriz de verificação H para o código.

A matriz de verificação é definida como

$$Hv^T = 0 \tag{9}$$

Podemos também relacionar a matriz de verificação H com a matriz geradora G através da relação

$$GH^T = 0 \tag{10}$$

Sabendo que o código de Hamming estendido é um código sistemático, ou seja os bits da mensagem podem ser encontrados inalterados na palavra código, e sabendo que podemos separar a matriz geradora G em duas matrizes I e P onde I é a matriz identidade e P é a matriz de paridade, podemos escrever a matriz de verificação H como

$$G = [I_k P]; \tag{11}$$

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \therefore P = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \Rightarrow P^T = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{12}$$

sendo $m = n - k$, logo $m = 4$. Portanto a matriz de verificação é dada por

$$H = [P^T I_m] = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

e) Construa uma tabela de síndrome $\mapsto$ padrão de erro

Para obtermos a tabela de síndrome $\mapsto$ padrão de erro, devemos considerar que a síndrome é dada por

$$s = He^T \tag{14}$$

Onde e é o vetor de erro, obtendo a matriz e podemos obter a síndrome s. Para definir e precisamos montar o arranjo padrão e pegar a primeira coluna dele. Utilizando python obtemos o seguinte arranjo padrão possível:

00000000	00011110	00100111	00111001	01001011	01010101	01101100	01110010	10001101	10010011	10101010	10110100	11000110	11011000	11100001	11111111
10000000	10011110	10100111	10111001	11001011	11010101	11101100	11110010	00001101	00010011	00101010	00110100	01000110	01011000	01100001	01111111
01000000	01011110	01100111	01111001	00001011	00010101	00101100	00110010	11001101	11010011	11101010	11110100	10000110	10011000	10100001	10111111
00100000	00111110	00000111	00011001	01101011	01110101	01001100	01010010	10101101	10110011	10001010	10010100	11100110	11111000	11000001	11011111
00010000	00001110	00110111	00101001	01011011	01000101	01111100	01100010	10011101	10000011	10111010	10100100	11010110	11001000	11110001	11101111
00001000	00010110	00101111	00110001	01000011	01011101	01100100	01111010	10000101	10011011	10100010	10111100	11001110	11010000	11101001	11110111
00000100	00011010	00100011	00111101	01001111	01010001	01101000	01110110	10001001	10010111	10101110	10110000	11000010	11011100	11100101	11111011
00000010	00011100	00100101	00111011	01001001	01010111	01101110	01110000	10001111	10010001	10101000	10110110	11000100	11011010	11100011	11111101
00000001	00011111	00100110	00111000	01001010	01010100	01101101	01110011	10001100	10010010	10101011	10110101	11000111	11011001	11100000	11111110
11000000	11011110	11100111	11111001	10001011	10010101	10101100	10110010	01001101	01010011	01101010	01110100	00000110	00011000	00100001	00111111
10100000	10111110	10000111	10011001	11101011	11110101	11001100	11010010	00101101	00110011	00001010	00010100	01100110	01111000	01000001	01011111
10010000	10001110	10110111	10101001	11011011	11000101	11111100	11100010	00011101	00000011	00111010	00100100	01010110	01001000	01110001	01101111
10001000	10010110	10101111	10110001	11000011	11011101	11100100	11111010	00000101	00011011	00100010	00111100	01001110	01010000	01101001	01110111
10000100	10011010	10100011	10111101	11001111	11010001	11101000	11110110	00001001	00010111	00101110	00110000	01000010	01011100	01100101	01111011
10000010	10011100	10100101	10111011	11001001	11010111	11101110	11110000	00001111	00010001	00101000	00110110	01000100	01011010	01100011	01111101
10000001	10011111	10100110	10111000	11001010	11010100	11101101	11110011	00001100	00010010	00101011	00110101	01000111	01011001	01100000	01111110

Aplicando a equação $s = He^T$ obtemos a tabela de síndrome $\mapsto$ padrão de erro

Síndrome	Padrão de erro
0000	00000000
1101	10000000
1011	01000000
0111	00100000
1110	00010000
1000	00001000
0100	00000100
0010	00000010
0001	00000001
0110	11000000
1010	10100000
0011	10010000
0101	10001000
1001	10000100
1111	10000010

Síndrome	Padrão de erro
1100	10000001

e) Determine a distribuição de peso dos padrões de erro corrigíveis.

Os padrões de erro corrigíveis são os representandes dos cosets do arranjo padrão, ou seja, podemos extrair os padrões de erro corrigíveis a partir da primeira coluna do arranjo padrão. Verificando seus pesos, obtemos a seguinte distribuição de peso dos padrões de erro corrigíveis:

i	0	1	2	3	4	5	6	7	8
ε_i	1	8	7	0	0	0	0	0	0

Escreva um programa que simule o desempenho BER de um sistema de comunicação que utiliza o código de Hamming (8,4) com codificação via síndrome, modulação QPSK (com mapeamento Gray) e canal AWGN. Considere a transmissão de 100000 palavras-código e relação de sinal-ruído de bit (E_b/N_0) variando de -1 a 7 dB. Compare com o caso não codificado.

Utilizando a linguagem de programação Python e a biblioteca `komm` podemos simular o desempenho BER do código de Hamming $(8, 4)$ com codificação via síndrome, modulação QPSK e canal AWGN.

É possível verificar que o desempenho BER do código de Hamming $(8, 4)$ é superior ao não codificado para valores de E_b/N_0 maiores que 4 dB.