Uma generalização dos Códigos Hamming

N. Pedroza[1], **P. E. D. Pinto**[2], **J. L. Szwarcfiter** [1,2]

[1]PESC – Universidade Federal do Rio de Janeiro – RJ – Brasil

[2]IME/DICC – Universidade do Estado do Rio de Janeiro – RJ – Brasil

nataliaps@cos.ufrj.br, pauloedp@ime.uerj.br, jayme@nce.ufrj.br

Abstract. *We present a recursive construction of optimal linear codes of lenght $n \geq 3$ and Hamming distance 3, as well as the coding and decoding processes with efficient performance. In particular, when n has the form $2^r - 1$ we have exactly the binary Hamming codes.*

Resumo. *Apresentamos uma construção recursiva de códigos lineares ótimos de comprimento $n \geq 3$ e distância Hamming 3, bem como os processos de codificação e decodificação com desempenho eficiente. Em particular, para n da forma $2^r - 1$ temos exatamente os códigos Hamming binários.*

1. Introdução

Um código com distância Hamming d é capaz de corrigir $\lfloor \frac{d-1}{2} \rfloor$ erros, [Hamming 1950]. A distância Hamming d de um código linear C é o tamanho do menor conjunto linearmente dependente de colunas de sua matriz de paridade H, [Hartnett 2012]. Para um dado comprimento n e uma distância Hamming 3, o número máximo de palavras de um código é dado por $B(n, 3) = 2^{n-r}$, onde r é o único inteiro que satisfaz $2^{r-1} - 1 < n \leq 2^r - 1$, [MacWilliams and Sloane 1977], de onde podemos concluir que $r = \lceil \log_2(n + 1) \rceil$. Um código com o máximo de palavras possíveis é denominado código *ótimo*.

A seguir, caracterizaremos uma família de códigos lineares binários ótimos, para $d = 3$, denominada $Gham(n)$, onde n indica o comprimento da palavra do código.

2. Construção do código $Gham(n)$

Vamos construir um código de n bits, com $r_n = \lceil \log_2(n+1) \rceil$ bits de paridade e dimensão $k_n = n - \lceil \log_2(n+1) \rceil$, de maneira recursiva, a partir do código $Gham(3) = \{000, 111\}$. Observe que, para este código base, temos $n = 3, r_3 = 2, k_3 = 1$ e $d = 3$. Uma matriz geradora do mesmo é a matriz $G_{1\times 3} = [111]$.

Dado o código $Gham(n-1) = \{c_1^{n-1}, c_2^{n-1}, \cdots, c_M^{n-1}\}$ de tamanho $n-1$ com $M = 2^{k_n}$ palavras e distância Hamming 3, criamos o código $Gham(n) = \{c_1^n, c_2^n, \cdots, c_{M'}^n\}$ da seguinte forma:

1. Se $n \neq 2^m$, para $m \in \mathbb{N}$, então as palavras de $Gham(n)$ são dadas por:

$$c_i^n = \begin{cases} 0||c_i^{n-1}, \text{ para } 1 \leq i \leq M \\ 1||[c_i^{n-1} \oplus bin(n, n-1)], \text{ para } M+1 \leq i \leq 2M, \end{cases}$$

onde $bin(n, n-1)$ denota a representação binária do inteiro n utilizando $n-1$ bits e $a||c_i$ representa a concatenação do bit a com a palavra c_i.
Neste caso, os parâmetros r_n e k_n são dados por: $r_n = r_{n-1}$ e $k_n = k_{n-1} + 1$.

2. Se $n = 2^m$, para $m \in \mathbb{N}$, então as palavras de $Gham(n)$ são dadas por:

$$c_i^n = c_{i_1}^{n-1} \cdots c_{i_k}^{n-1} || 0 || c_{i_{k+1}}^{n-1} \cdots c_{i_{n-1}}^{n-1}, \text{ para } 1 \leq i \leq M,$$

onde $c_{i_j}^{n-1}$ representa o bit j da palavra c_i^{n-1}.
Observe que, neste caso, tomamos o código $Gham(n-1)$ e acrescentamos 0 na posição $k+1$. Observe também que a segunda metade das palavras ainda pode ser escrita como $1||[c_i^{n-1} \oplus bin(n, n-1)]$, para $\frac{M}{2} \leq i \leq M$.
Neste caso, os parâmetros r_n e k_n são dados por: $r_n = r_{n-1} + 1$ e $k_n = k_{n-1}$.

Na Tabela 1 mostramos os códigos para n de 3 a 6. Os códigos para n de 5 e 6 correspondem à primeira forma de criação recursiva, com 3 bits de paridade para todos. O código para $n = 4$ corresponde à segunda forma.

Table 1. $Gham(n)$ **para** n **de 3 a 6**

n = 3	n = 4	n = 5	n = 6
000 111	0000 1011	00000 10101	000000 100110
		01011 11110	001011 101101
			010101 110011
			011110 111000

3. Propriedades dos códigos $Gham(n)$

Para uma dada uma palavra c_i^n, denotaremos por u_i^k o string formado pelos primeiros $k = k_n$ bits de c_i^n e, por $p(u_i^k)$ o string formado pelos $r = r_n$ bits finais.

Propriedade 1 *O código $Gham(n)$ é ordenado e os strings u_i^k correspondem à representação binária, usando k bits, de todos os inteiros do intervalo 0 a $2^k - 1$.*

Propriedade 2 *Os códigos $Gham(n)$ são códigos lineares, cujas matrizes geradoras podem ser escritas na forma $G = [I_k | A_{k \times n-k}]$, onde I_k representa a matriz identidade de ordem $k = n - \lceil \log_2(n+1) \rceil$ e as linhas de $A_{k \times n-k}$ são formadas pelas representações binárias dos números de 3 a n que não são potências de 2, em ordem decrescente, usando $r = \lceil \log_2(n+1) \rceil$ bits.*

Prova: Vamos fazer a prova por Indução Matemática. Temos dois casos base, que são $n = 3$ e $n = 4$. É fácil ver que $Gham(3)$ é gerado por $G_3 = [111]$, $k = 3 - \lceil \log_2(3+1) \rceil = 1$ e a única linha de A é o número 3 representado em $3 - 1 = 2$ bits. Já para $Gham(4)$ temos a matriz $G_4 = [1011]$, com $k = 4 - \lceil \log_2(4+1) \rceil = 1$ e a única linha de A é o número 3 representado em $4 - 1 = 3$ bits.

Para o restante da prova consideraremos dois casos conforme n seja potência de 2 ou não. Lembre que todos os vetores u_i^{k+1} de F^{k+1} são da forma $0u_j^k$ ou $1u_j^k$, para $u_j^k \in F^k$. Suponha que $G_n = [I_{k \times k} | A_{k \times n-k}]$ é uma matriz geradora do código $Gham(n)$. Isto é, as palavras de $Gham(n)$ são formadas fazendo-se $c_n = u_i^k \cdot G_n$, para todo $u_i \in F^k$.

1. Se $n + 1 \neq 2^m$, para $m \in \mathbb{N}$, então,

Temos que a matriz G_{n+1} é da forma:

$$G_{n+1} = \left[\begin{array}{c|c} 1 & \text{bin(n+1,n)} \\ \hline 0 & \\ \vdots & G_n \\ 0 & \end{array}\right].$$

onde cada bit de $bin(n+1, n)$ é escrito em uma das n colunas de G_{n+1} ordenadamente.

(a) se $u_i^{k+1} = 0||u_i^k$, então temos que $(0||u_i^k) \cdot G_{n+1} = 0||c_i^n$, pois ao multiplicarmos $0||u_i^k$ pela primeira coluna de G_{n+1}, geramos o primeiro bit 0 no resultado final; e como o primeiro bit de $0||u_i^k$ é 0, multiplicar $0||u_i^k$ pelas n últimas colunas de G_{n+1} é o mesmo que fazer $u_i^k \cdot G_n = c_i^n$.

(b) se $u_i^{k+1} = 1||u_i^k$, então $(1||u_i^k) \cdot G_{n+1} = 1||[c_i^n \oplus bin(n+1, n)]$, pois ao multiplicarmos $1||u_i^k$ pela primeira coluna de G_{n+1}, geramos o primeiro bit 1 no resultado final; e como o primeiro bit de $1||u_i^k$ é 1, multiplicar $1||u_i^k$ pelas n últimas colunas de G_{n+1} é o mesmo que fazer $u_i^k \cdot G_n = c_i^n$ e somar em cada posição, os bits da primeira linha, ou seja, fazer $c_i^n \oplus bin(n+1, n)$.

Assim, todas as palavras de $Gham(n+1)$ são geradas.

2. Se $n + 1 = 2^m$, para $m \in \mathbb{N}$, então,
Neste caso, a matriz geradora G_{n+1} possui uma coluna nula na posição $k + 1$.

$$G_{n+1} = \left[\begin{array}{c|c|c} & 0 & \\ I_k & \vdots & A_{k\times n-k} \\ & 0 & \end{array}\right].$$

Os vetores de informação pertencem a F^k. Logo, temos que $u_i^k \cdot G_{n+1} = c_{i_1}^n \cdots c_{i_k}^n 0 c_{i_{k+1}}^n \cdots c_{i_n}^n = c_i^{n+1}$, neste caso é acrescentado um bit 0 na posição $k+1$ em todas as palavras c_i^n. Isto é, todas as palavras de $Gham(n+1)$ são geradas. □

Na Tabela 2 apresentamos a parte não trivial das matrizes geradoras dos códigos $Gham(n)$ para n de 3 a 9.

Table 2. $A_{k\times n-k}$ **dos códigos** $Gham(n)$

n	3	4	5	6	7	8	9
	11	011	101	110	111	0111	1001
			011	101	110	0110	0111
$A_{k\times n-k}$				011	101	0101	0110
					011	0011	0101
							0011

Temos então que a matriz de paridade H do código $Gham(n)$ é formada pelas representações binárias dos números de 1 a n escritos em coluna utilizando $n - k$ bits. E portanto, os códigos de Hamming $Ham(r)$ são um caso particular dos códigos $Gham(n)$.

4. Codificação e Decodificação

Seja uma palavra c_i^n, com $0 \leq i \leq 2^k - 1$, do código $Gham(n)$. Note que pelo processo de construção dos códigos $Gham(n)$ se uma palavra começa com bit 1, então ela foi gerada fazendo-se um ou exclusivo com $bin(n, n-1)$. Assim, tomando $u_i^k = u_{i_k} u_{i_{k-1}} \cdots u_{i_1}$, para calcular $p(u_i^k)$, fazemos a operação de ou exclusivo de $bin(0, r)$ com $A(j)$ para todo j tal que $u_{i_j} = 1$ com $1 \leq j \leq k$, onde $A(j)$ representa o j-ésimo número que não é potência de 2. O processo de decodificação é dado pelo teorema a seguir.

Teorema 3 *Considere que um vetor y de tamanho n é recebido. Vamos denotar os $k = n - \lceil \log_2(n+1) \rceil$ bits inicias de y por u e os r finais por $p'(u)$. O processo de decodificação consiste em calcular $p(u)$ e fazer a operação $p(u) \oplus p'(u)$. Temos que:*

1. *$p(u) \oplus p'(u) = \mathbf{0}$ se, e somente se, não ocorreu erro em y;*
2. *$p(u) \oplus p'(u)$ tem apenas um bit 1, digamos na posição l se, e somente se, ocorreu um erro na posição l de $p'(u)$;*
3. *$p(u) \oplus p'(u)$ é o l-ésimo número em binário que não é potência de 2, com $1 \leq l \leq k$ se, e somente se, ocorreu um erro na posição l (da direita para a esquerda) de u;*
4. *Se $p(u) \oplus p'(u)$ não satisfaz a nenhuma condição anterior, então ocorreu mais de um erro e y não é decodificado.*

Prova:

1. $p(u) \oplus p'(u) = \mathbf{0} \Leftrightarrow p(u) = p'(u) \Leftrightarrow y = u||p(u) \in Gham(n)$. Note que, para todo $u \in F^k$, o vetor $u||p(u)$ de tamanho n é uma palavra do código $Gham(n)$.
2. Se $p(u) \oplus p'(u)$ tem apenas um bit 1, na posição l, então $d(y, u||p(u)) = 1$. O fato de um código C ter distância Hamming 3, garante que um vetor qualquer de F^k tem distância 1 para, no máximo, uma palavra de C. Neste caso, y é decodificado como $u||p(u)$, isto é, ocorreu um erro na posição l de $p'(u)$. A volta é imediata. Nos casos restante, temos que $w(p(u) \oplus p'(u)) > 1$.
3. Vamos mostrar que o erro ocorreu na posição l do vetor u. Para isto, considere o vetor u' obtido a partir de u trocando-se apenas o bit na posição l, para algum $1 \leq l \leq k$. Temos que $p(u') = p(u) \oplus A[l]$ ou podemos escrever $p(u) \oplus p(u') = A[l]$. Se $p(u) \oplus p'(u) = A[l]$, então $p(u') = p'(u)$. Portanto, $d(y, u'||p(u')) = 1$. E o vetor y é decodificado como $u'||p(u')$, tendo ocorrido um erro na posição l de u. Reciprocamente, se ocorreu um erro na posição l de u, então $p(u) = p(u') \oplus A[l]$. Como não ocorreu erro na parte de paridade, temos que $p(u') = p'(u)$. Logo, $p(u) \oplus p'(u) = A[l]$.
4. Se $p(u) \oplus p'(u)$ não satisfaz nenhuma das condições anteriores, considerando as demonstrações dos 3 itens, podemos concluir que houve mais de 1 erro o que impossibilita a correção.

□

Referências

Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160.

Hartnett, W. E. (2012). *Foundations of coding theory*, volume 1. Springer Science & Business Media.

MacWilliams, F. J. and Sloane, N. J. A. (1977). *The theory of error correcting codes*. Elsevier.