Códigos corretores de erros de tamanhos variáveis *

Paulo Eustáquio D. Pinto[1], **Natália Pedroza de Souza**[2], **Jayme L. Szwarcfiter**[2]

[1]Instituto de Matemática e Estatística – UERJ

[2]Programa de Engenharia de Sistemas e Computação – UFRJ

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

Abstract. *In this paper we discuss the construction of Variable Length Error Correcting Codes (VLECC) and we show that the cost of those codes may be lower than the corresponding fixed length ones, even when the frequency distribution of the symbols to be encoded is uniform.*

Resumo. *Neste artigo discutimos a construção de Códigos Corretores de Erro de Tamanho Variável (VLECC) e mostramos que seu custo pode ser menor do que o dos correspondentes de tamanho fixo, mesmo quando a distribuição de frequência dos símbolos a serem codificados é uniforme.*

1. Introdução

Códigos corretores de erros são importantes na segurança de informações transmitidas em diversos contextos, pois permitem detecção e correção de erros introduzidos nos processos de transmissão. Tradicionalmente tem sido usados códigos de tamanhos fixos. Só a partir dos últimos anos é que se passou a considerar os códigos corretores de tamanhos variáveis (VLECC) tais como em [Buttigieg 1995] e [Wu et al. 2011]. A principal vantagem do seu uso é na redução das mensagens enviadas, o que pode ser obtido em situações onde temos diferentes probabilidades de ocorrências dos diversos símbolos codificados. A símbolos muito frequentes são atribuídas codificações pequenas.

Em um VLECC temos que considerar as frequências dos símbolos de um alfabeto e uma quantidade t de erros que ele é capaz de corrigir. De uma forma simplificada, podemos dizer que códigos ótimos seriam aqueles capazes de produzir as menores mensagens codificadas possíveis. Na literatura pesquisada esse problema de otimização encontra-se em aberto. A maioria dos enfoques considera apenas reduções do problema ou a construção de heurísticas para a obtenção de bons códigos. Um exemplo disso é a construção de VLECC através de concatenação de códigos fixos descrita em [Wenisch et al. 2001].

Neste trabalho apresentamos uma possibilidade do uso de VLECC's ainda não comentada pelos diversos autores pesquisados, que é o fato de permitirem economia no tamanho da codificação de mensagens em relação aos códigos de tamanho fixo, mesmo para o caso de frequências constantes. Descrevemos dois processos distintos para obtenção de VLECC's com capacidade de correção de 1 erro e com a característica apontada.

*Parcialmente financiado por FAPERJ, CNPQ, CAPES

2. Definições

Seja $C = \{c_1, c_2, \cdots, c_M\}$ um código binário com M palavras. O tamanho da palavra c_i é definido pelo número de bits de c_i e denotado por $|c_i|$, para $1 \leq i \leq M$. O *custo* de um código, $c(C)$, quando a distribuição das frequências dos símbolos é uniforme, é dado pela soma do número de bits de todas as palavras que formam C. Um código é ótimo quando tem o menor custo possível. A *distância Hamming* d entre duas palavras de mesmo tamanho é definida pelo número de posições em que elas diferem entre si.

Sejam duas palavras c_i e c_j tais que $|c_i| \geq |c_j|$. A *distância divergente* entre c_i e c_j, $d(c_i, c_j)$, é definida como sendo a distância Hamming entre o prefixo de c_i de tamanho $|c_j|$ e c_j. Formalmente, a distância divergente de um código C é definida por:

$$d_d(C) = \min\{d(c_i, c_j), \text{para todo } i \neq j\}.$$

Da teoria de códigos corretores de erros, sabemos que um código com distância Hamming d potencialmente é capaz de corrigir $\lfloor \frac{d-1}{2} \rfloor$ erros. Como queremos corrigir 1 erro, construiremos códigos com distância divergente 3.

Denota-se por $A_q(n, d)$ o maior valor de M tal que existe um código de tamanho fixo q-ário com M palavras de n bits e distância Hamming d. Muitos desses valores ainda encontram-se em aberto. Os valores para $n \leq 15$ são mostrados na Tabela 1 [MacWilliams and Sloane 1977].

Tabela 1. Número máximo de palavras com n bits e distância 3

n	3	4	5	6	7	8	9	10	11	12	13	14	15
$A_2(n,3)$	2	2	4	8	16	20	40	72	144	256	512	1024	2048

A seguir apresentamos uma família infinita de VLECC's com custo inferior ao de códigos de tamanho fixo, mesmo para distribuição uniforme de frequências.

3. Uma família VLECC's com custo inferior ao dos correspondentes códigos corretores de tamanho fixo

Tome um código ótimo C_M com $M = A_2(n, 3)$ palavras. Podemos criar um código C_{M+1} com $M + 1$ palavras, a partir de C_M, da seguinte forma: duplicamos uma palavra qualquer de C_M e, a cada uma destas palavras iguais, adicionamos uma sequência de 3 bits com distância Hamming 3 entre si. Note que C_{M+1} possui uma palavra a mais que C_M e a distância divergente 3 é mantida. De forma mais geral, podemos tomar p palavras de um código C_M para duplicar e, desta forma, constuirmos um código com $M + p$ palavras.

Um código ótimo com $M = A_2(n, 3)$ palavras utiliza no máximo $n \cdot A_2(n, 3)$ bits. Portanto, o custo do código construído através do processo descrito é no máximo:

$$n \cdot A_2(n, 3) + pn + 2p \cdot 3,$$

onde o primeiro termo representa o custo de C_M, o segundo, o número de bits das p palavras duplicadas e terceiro, os 3 bits acrescidos em cada uma das p palavras escolhidas e suas duplicações. Note que este custo pode ser ainda menor, se o código de tamanho variável C_{M-p} é melhor do que o código de tamanho fixo para este valor.

É válido aplicarmos este processo apenas nos casos em que o custo do código resultante é menor do que o do código de tamanho fixo, ou seja, quando:

$$n \cdot A_2(n,3) + pn + 2p \cdot 3 < (A_2(n,3) + p)(n+1).$$

Ou ainda, quando temos: $p < \dfrac{A_2(n,3)}{5}$.

Observe que este é um processo de concatenação de códigos. O que se faz aqui é escolher p' palavras para serem prefixos no código resultante C_{M+p}. Cada prefixo dá origem a p_i palavras de modo que $\sum_{i=1}^{p'} p_i = p$. Deste modo, um prefixo contribui em 0 com a distância, logo, os bits adicionais devem por si só formar um código com distância divergente de ao menos 3.

A quantidade de bits adicionados é

$$pn + \sum_{i=1}^{p'} c(C_{p_i}).$$

onde $c(C_{p_i})$ representa o número de bits utilizados pelo código C_{p_i}. Como o código que utiliza o menor número de bits por palavra é o código C_2, concluímos que

$$p \cdot c(C_2) = 6p \leq \sum_{i=1}^{p'} c(C_{p_i}).$$

Logo, a alternativa que adiciona o menor número de bits possível é a utilizada no processo descrito: transformar cada um dos p prefixos em duas palavras.

Na próxima seção apresentamos alguns VLECC's especiais, fora da família apresentada e que também são vantajosos em relação aos códigos de tamanho fixo.

4. Casos especiais

Para os valores de M iguais a 3, 5, 6 e 10, o método ilustrado na Seção 3 não se aplica. Porém, através de uma busca exaustiva encontramos VLECC's com custo menor que os códigos de tamanho fixo. E, ainda, com esta busca, para $M = 9$, melhoramos o custo do VLECC. A técnica exaustiva utilizada é descrita a seguir.

Para formar um código com M palavras, construimos uma árvore com M folhas. Às arestas serão associadas sequências de bits. Assim, cada palavra será formada concatenando-se os bits associados às arestas do caminho percorrido da raiz até uma folha.

A construção da árvore é feita nível a nível. A cada nível precisamos formar ao menos uma palavra que deve ter $d_d \geq 3$ para todos os prefixos formados até então. Analisamos todas as possibilidades de escolhas de k bits para associar às arestas de cada nível. O processo limita-se a construir árvores onde o total dos tamanhos das M palavras seja inferior ao do correspondente código de tamanho fixo.

Note que, para o primeiro nível, $k \geq 3$, pois com menos do que isso, não é possível criar uma palavra com $d_d \geq 3$ para os outros prefixos.

Vamos exemplicar o método para $M = 3$. Consideremos, s.p.g., a primeira escolha de bits para associar a uma aresta sendo 000. Neste caso, só existe um segmento com 3 bits e $d_d \geq 3$ para 000, o segmento 111, que é então associado a uma segunda aresta neste primeiro nível. Assim, 000 forma uma palavra do código e 111 é o prefixo das outras duas palavras. No segundo nível, devemos criar dois filhos para o nó associado a aresta 111. O número mínimo de bits que precisamos associar a cada aresta deste nível é 3, já que as palavras possuirão o mesmo prefixo, 111, e devem ter $d_3 = 3$. Isto pode ser feito associando-se os segmentos 000 e 111. Assim criamos o código $C = \{000, 111000, 111111\}$.

Os códigos encontrados através deste processo são apresentados a seguir.

C_3 =	C_5 =	C_6 =	C_{10} =
0000	0000	00000	000000
01110	011100	11011	011110
10111	101101	011011	101011
	110110	011100	110101
	111011	101010	0100111
		101101	0110010
			1001101
			1011000
			1100100
			1110001

5. Conclusão

Mostramos que, para certas quantidades M de símbolos a serem codificados (algumas ilustradas na Tabela 2) é mais vantajosa a utilização de codificações com tamanhos variáveis do que tamanhos fixos para frequências constantes. Salientamos que, para frequências distintas dos símbolos, a utilização de códigos de tamanhos variáveis vem a ser ainda melhor do que de códigos de tamanho fixo. Fizemos o estudo para códigos com distância divergente 3, porém a técnica proposta pode ser estendida para qualquer d_d.

Tabela 2. Média de bits para códigos de tamanho fixo e variável e $d = 3$

M	3	5	6	9	10	11	12	17	18	19	21	22	23	41
Fixo	5	6	6	7	7	7	7	8	8	8	9	9	9	10
Variável	4,67	5,6	5,66	6,44	6,6	6,63	6,66	7,35	7,67	7,95	8,29	8,55	8,78	9,15

Referências

Buttigieg, V. (1995). *Variable-Length Error-Correcting Codes*. PhD thesis, Department of Electrical Engineering, University of Manchester, England.

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

Wenisch, T., Swaszek, P. F., and Uht, A. K. (2001). Combined error correcting and compressing codes. In *Information Theory, 2001. Proceedings. 2001 IEEE International Symposium on*, page 238. IEEE.

Wu, T.-Y., Chen, P.-N., Alajaji, F., and Han, Y. S. (2011). On the construction and map decoding of optimal variable-length error-correcting codes. In *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, pages 2223–2227. IEEE.

