COMBINATÓRIA E TEORIA DE CÓDIGOS
Exercise List 1

2/3/2011

Exercises 1.2 - 1.5 (R. Hill)

Problem 1. (Generalization of 1.2) What is the capacity of a code, with minimum distance d, for detecting and correcting errors *simultaneously*? Discuss the cases d odd and even separately. Ilustrate with examples.

Problem 2. (Generalization of 1.5) Find an upper bound (and prove it) for the maximum number of words M in a q-ary (n, M, d) code. Ilustrate with examples.

Problem 3. Discuss the capacity of a code, with minimum distance d, of correcting symbol errors and erasure errors *simultaneously*. State conjectures and theorems, prove them, and ilustrate with examples.

Problem 4. Consider a binary channel with the following error probabilities

$$P(1 \text{ received } |0 \text{ sent}) = 0,3 \ ; \qquad P(0 \text{ sent } |1 \text{ sent}) = 0,2 \ .$$

For the binary code $\{000, 100, 111\}$, use maximum likelihood decoding, to decode the received words

(a) 010 ; (b) 011 ; (c) 001 .

Problem 5. The following binary word

01111000000?001110000?0011001100101011100000000?01110

encodes a date. The encoding method used consisted in writing the date in 6 decimal digits (e.g. 290296 means February 29th, 1996), then converting it to a number in base 2 (e.g. 290296 becomes 1000110110111111000), and enconding the binary number using the rule

$$\begin{aligned} \{0,1\}^2 &\longrightarrow \mathcal{C} \subseteq \{0,1\}^6 \\ 00 &\longmapsto 000000 \\ 01 &\longmapsto 001110 \\ 10 &\longmapsto 111000 \\ 11 &\longmapsto 110011 \end{aligned}$$

The received word contains 3 unknown digits (which were deleted) and it may also contain some switched digits.

(a) Find the deleted bits;

(b) How many, and in which positions, are the wrong bits?

(c) Which date is it?

(d) Repeat the problem switching the bits in positions 15 and 16.

Problem 6. (A HAMMING Code) We encode a *messagge vector* with 4 binary components $\mathbf{m} = m_1m_2m_3m_4$, $m_i \in \{0,1\}$, as a *code word* with 7 binary components $\mathbf{c} = c_1c_2c_3c_4c_5c_6c_7$, $c_j \in \{0,1\}$, defined by

$$c_3 = m_1 \ ; \ c_5 = m_2 \ ; \ c_6 = m_3 \ ; \ c_7 = m4$$

and the other components are chosen so that

$$c_4 : \text{ such that } \alpha = c_4 + c_5 + c_6 + c_7 \text{ is even}$$

$$c_2 : \text{ such that } \beta = c_2 + c_3 + c_6 + c_7 \text{ is even}$$

$$c_1 : \text{ such that } \gamma = c_1 + c_3 + c_5 + c_7 \text{ is even.}$$

Check that with this coding scheme we get a code which corrects an error in any position.

If we receive the vector $\mathbf{x} = x_1x_2x_3x_4x_5x_6x_7$, we compute

$$\left.\begin{aligned} \alpha &= x_4 + x_5 + x_6 + x_7 \\ \beta &= x_2 + x_3 + x_6 + x_7 \\ \gamma &= x_1 + x_3 + x_5 + x_7 \end{aligned}\right\} \text{mod2;}$$

$\alpha\beta\gamma$ is the binary representation of the j component in which the error occured. If $\alpha\beta\gamma = 000$ we assume no error occured.

Study this example carefully.