EECS 70 Discrete Mathematics and Probability Theory
Fall 2014 Anant Sahai **Discussion 8M**

1. Corruption

Alice wants to send Bob a message of size 3 and guard against 1 corrupt packet. Bob receives from Alice the packets $(0,1),(1,2),(2,3),(3,3),(4,4)$, and he suspects one of the packets was indeed corrupted. Find Alice's original message.

(a) What degree should the encoding polynomial $P(x)$ be? What degree should the error-locating polynomial $E(x)$ be? What degree should $Q(x)$ be, where $Q(x) = P(x)E(x)$?

(b) Using the relation $Q(i) = r_i E(i)$, write a system of linear equations from the received messages.

(c) Solving the system you found in the previous problem, you find that $E(x) = x + 4$ and $Q(x) = 3x^3 + 2x^2 + x + 4$. Which packet contains the error?

(d) What is $P(x)$?

(e) What is Alice's original message?

2. List decoding

(a) Consider an n character message encoded into m characters over the field $GF(p)$ using polynomials. Suppose that one receives $n-1$ of the m packets. Give a method to find a list of size at most p of all possible messages.

(b) Consider an n character message encoded into $m = n+2k$ characters over the field $GF(p)$ using polynomials. Suppose that $k+1$ of the m received packets are corrupted. Give a method to find a list of all possible messages which contain the original message. What is the size of the list for your scheme?

(c) Consider the protocol in (b) where we are working in GP(7). Let the original message have $n = 1$ and $k = 2$, so there are 5 symbols. Now suppose that there are 3 errors, but these three errors all landed on different values. Assume that we received: $0, 0, 1, 2, 3$. How does your list-decoding strategy perform?

3. Linearity

Prove that Reed Solomon codes are *linear*; that is, the sum of two Reed Solomon codewords is also a Reed Solomon codeword. To do this, use coefficient encoding rather than interpolation encoding: If you have a message of length n and you want to send m packets, create a degree $n-1$ polynomial $p(x)$ where your message $(c_0, c_1, \ldots, c_{n-1})$ are the coefficients of $p(x)$, and the codeword is the evaluation of $p(x)$ at $\{0, 1, \ldots, m-1\}$.

4. Hamming Code

Here is an example of a simple single-error correcting code that works without having to invoke polynomials and just deals with bits. In real life, these kinds of codes are often used in conjunction with Reed Solomon codes. In this problem we will see how to send 4 bits (each either 0 or 1) using Hamming codes. This will be accomplished by using 3 parity bits (check bits) so that 1 error in the message can be detected and corrected. To encode a 4-bit message m, we will multiply it by a code-generator matrix G:

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(a) Show how to encode the message $m = 1101$. What is the resulting 7-bit transmitted codeword x?

(b) To detect potential errors, we can use a parity check matrix H:

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

What do you notice about the form of H? If you multiply H by x, what do you get? Will this be true for any message m?

(c) Suppose that there was an error (flipped bit) in the 6th bit of x, yielding x'. To find out where it occurred, we can multiply H by x'. What is the result base 10? How can you correct x?

(d) Now we wish to recover the original message m. To do this, we can multiply the corrected x by a recovery matrix R that pulls out the 4 original bits:

$$R = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Do you get m back?

(e) Do you think that a code like this could exist that could fix a single binary error in a 4-bit message without using at least 7 bits for the codeword? Why or why not?