REDUCE ERRORS DURING DATA TRANSMISSION

ALGEBRAIC CODING THEORY

INTRODUCTION TO OUR EXPLORATION

- Goals, Conclusions:
  - explain general ideas behind coding theory
  - explain a couple of simple examples
  - be inspired because we can use Modern Algebra terms to describe real-life problems in CS and EE!

WHY IS THIS NECESSARY?

- when data is transmitted (Digital Signals, usually binary)
- -> noise occurs that disrupts the signal
- “01001” could be received as “01011”
- a single bit error occurred
- we need to be able to detect this
- especially important for encrypted data


interference, noise, corruption

GENERAL IDEA

- add repetition or redundancy to original message (encoding)
- produce a "CODEWORD" to be sent
- receiver receives the codeword and decodes to get original message
- use error vector, e, to define noise corruption

$$\mathbf{m} \to Encode \to \mathbf{c} \to Noise \to \mathbf{c} + \mathbf{e} = \mathbf{r} \to Decode \to \tilde{\mathbf{m}}$$

HAMMING DISTANCE AND HAMMING WEIGHTS

- we will be looking in the Finite Field GF(2) (binary)
- Code = F^n = F + F + F + … + F (n copies)
- HAMMING DISTANCE
  - number of ways in which 2 vectors differ
  - minimum Hamming Distance between any 2 codewords determines error correcting capabilities
- HAMMING WEIGHT
  - number of non-zero elements of the codeword

CORRECTION AND DETECTION OF ERRORS

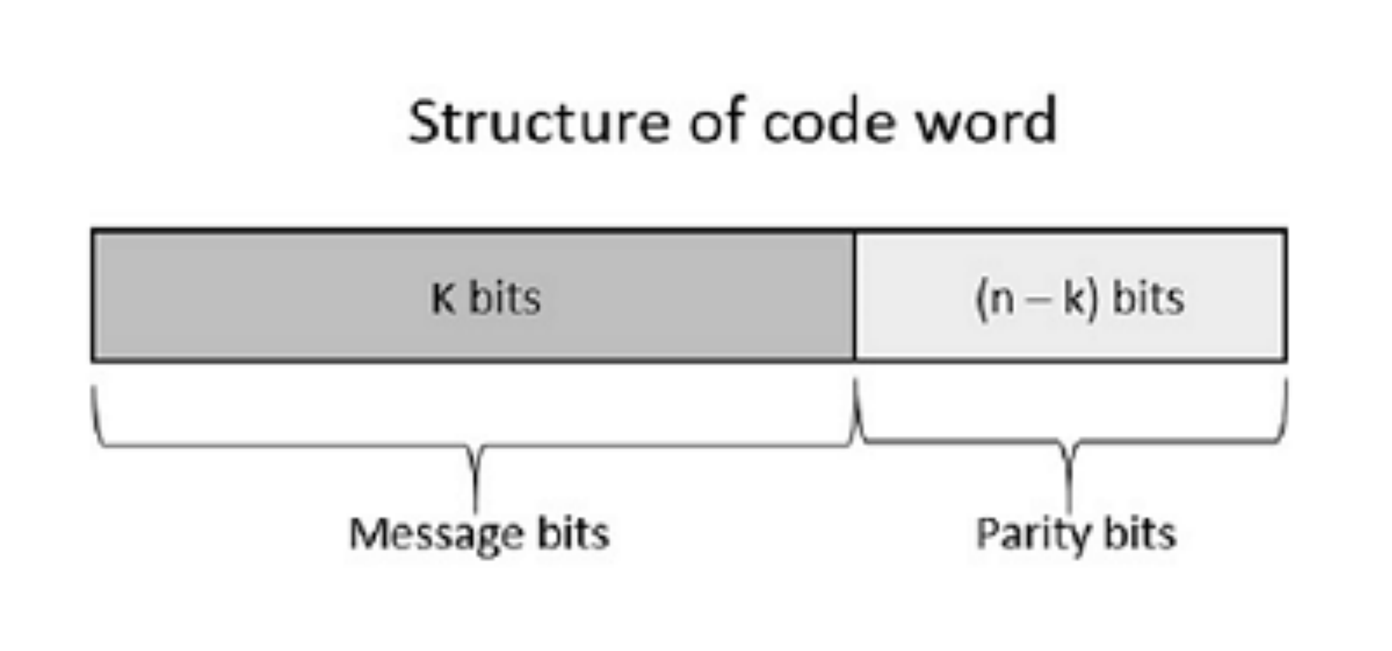
- different codes have different capabilities
- some codes can only detect errors (ISBN #'s)
- some codes are able to detect AND correct up to t errors (2-D Parity Check, Repetition, Nearest Neighbor)
- we will briefly cover these examples later

ISBN 978-1-61364-960-2

9 781613 649602

CORRECTION AND DETECTION (WHAT MAKES A CODE "GOOD"?)

- Criteria to determine functionality of a code:
  - length of a code, n
  - total # of possible codewords, M
  - distance between codewords, $d(C) = \text{minimum } \{d(v, u) = w(v - u) \text{ for all } v, u \text{ in } C\}$
  - efficiency, message size / codeword size


Structure of code word
K bits
(n – k) bits
Message bits
Parity bits

BACK TO ERROR CORRECTING CAPABILITIES

- ______________________________

Note: we are going to ignore probabilities
because that would take up time (and is not as fun).

Just assume the probability that a symbol comes across with an error is the same for each symbol ($p :=\ < 50\%$). and assume that the probability of not receiving an error is $1-p$

okay, moving on

CORRECTION AND DETECTION OF ERRORS

- A code, C, can detect up to s errors in any codeword if d(C) >= s + 1
- A code, C, can correct up to t errors in any codeword if d(C) >= 2t + 1
- EXAMPLES: binary repetitio[n] ... [leng]th 8
- "1" encoded as "11111111" ... [...]11101"
- Can the received message be properly corrected?

this makes sense, because if the number of errors is greater than the minimum distance, we might be looking at another codeword (other than the one originally sent)

YES

- Repetition code of length 8 can correct up to 3 errors
- although it can detect up to 7 errors, anything beyond 3 error could possibly be corrected to the wrong code.
- (this is why probability is kind of important)
- (also why simple repetition codes like this aren't used all that much)

ISBN NUMBERS: IN GF(11)

- Can only detect errors
- first 9 Digits contain information about the book
- last digit is a check digit that is calculated s.t.

$$\sum_{i=1}^{10} i x_i \equiv 0 \pmod{11}.$$

- d(C) = 2. changing a single digit of an ISBN code will fail but changing any 2 digits could result in another valid code!
- many states use similar codes to validate DL #'s
- (usually combined with hash of person's information)
- Book author, Gallian, actually figured out several state's DL codes

ERROR CORRECTION WITH 2-D PARITY MATRIX

Assume initial message = {1001100111}

① PUT INTO MATRIX FORM ⟶ ② ADD PARITY BITS

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} = M \qquad C = \left[\begin{array}{ccccc|c} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 \end{array}\right]$$

③ SEND CODEWORD

④ RECIEVED WORD

{NOISE!}

error

$$C = \{100111001111101000\} \longrightarrow R = \{100111011111101000\}$$

⑤ DECODE

$$\begin{array}{c} \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \leftarrow \text{odd} \\ \uparrow \\ \text{odd} \end{array}$$

⑥ FIX ERRORS

- switch bit in 2nd row x 2nd column.

$$M' = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} = \{1001100111\}$$

ERROR CORRECTION WITH 2-D PARITY MATRIX

- 2-D Parity can only detect single-bit errors
- in general, an n-dimensional Parity scheme can only correct up to n/2 errors
- this is because d(C) = n +1 (where n is the number of dimensions)

RESOURCES:

- Contemporary Abstract Algebra, Joseph Gallian, Chapter 30
- Introduction to Algebraic Coding Theory, Sarah Adams, Chapter 1