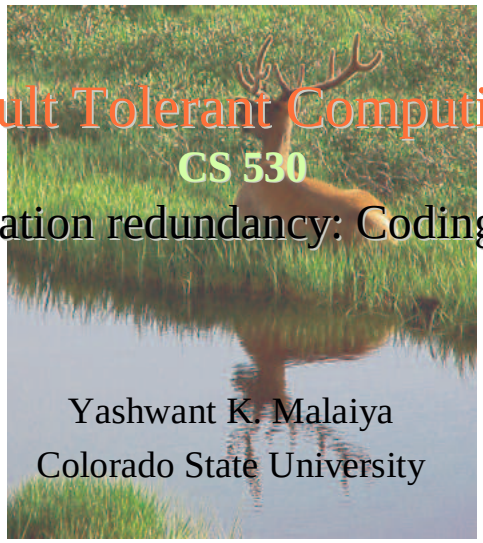


Fault Tolerant Computing

CS 530

Information redundancy: Coding theory

Yashwant K. Malaiya
Colorado State University

Information redundancy: Outline

- Codes & code words
- Hamming distance
  - Error detection capability
  - Error correction capability
- Parity check codes and ECC systems
- Cyclic codes
  - Polynomial division and LFSRs

Information Redundancy: Coding

- **Often applied to**
  - Info transfer: often serial communication thru a channel
  - Info storage
- Hamming distance: error detection & correction capability
- Linear separable codes, hamming codes
- Cyclic codes

Error Detecting/Correcting Codes (EDC/ECC)

- **Code:** subset of all possible vectors
- **Block codes**: all vectors are of the same length
- **Separable (systematic)** codes: check-bits can be separately identified.

  (n,k) code: k info bits, r = n-k check bits
- **Code words**: are legal part of the code.
- ***Linear*** Codes: Check-bits are linear combinations of info bits. Linear combination of code words is a code word.

Hamming Distance

- **Hamming distance** between 2 code words X, Y

  $D(x,y)=\Sigma(x_k \oplus y_k)$

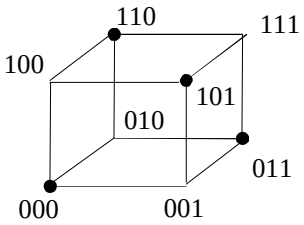
  - D(001,010)=2
  - D(000,111)=3
- **Minimum distance**: min of all hamming distance between all possible pairs of code words.

Ex 1: consider code:

000
011
101
110

Min distance=2

Detection Capability



Ex 1: consider code:

000
011
101
110

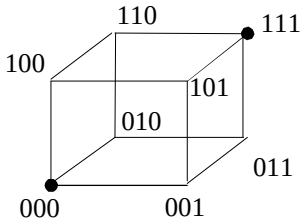
- All single bit errors result in non-code words. Thus all single-bit errors are detectable.
- Error detection capability: min Hamming dist d_{min}, p: number of errors that can be detected

  $p+1 \leq d_{min}$ or $p_{max} = d_{min} - 1$

Errors Correction Capability

Ex 2: Consider a code

000
111



- Assume single-bit errors are more likely than 2-bit errors.
- In Ex 2 all single bit errors can be corrected. All 2 bit errors can be detected.
- Error correction capability: t: number of errors that can be corrected:

$2t+1 \le d_{min}$ or $T_{max}=\lfloor (d_{min}-1)/2 \rfloor$

Proof?

Parity Check Codes

- Are linear block codes
- d_{min} = weight of lightest non-zero code word
- Linear: *addition*: ⊕, *multiplication:* AND
- G_{kxn}: **Generator matrix** of a (n,k) code: rows are a set of basis vectors for the code space.

  $i.G = v$ i: 1×k info, v :1×n code word
- For systematic code: $G=[I_k \; P]$ $I_{k:}$ k×k, P: k×(n-k)

  Ex: k=3, r=n-k=2

$$G = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right]$$

Parity Check Codes: Code Word Generation

- Ex: info i = (1 01)

$$G = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right]$$

then

$$v = (1\ 0\ 1) \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right]$$

$$v = (\underbrace{1\ 0\ 1}_{\text{info}}\ \underbrace{0\ 1}_{\text{check}})$$

Note: Matrix multiplication: (dimensions) a×b. b×c= a×c

Parity Check Codes: Parity Check Matrix H

- If v is a code word: $v.H^t = 0$ H^t: n×r, **0**: 1×r
- Corrupted information: $w=v+e$ all 1×n

  $w.\ H^t = (v+e)\ H^t = 0+e.\ H^t$

  $=s$ *syndrome of error*
- For t-error correcting code, syndrome is unique for up to t errors & can be used for correction.
- For systematic codes $G.\ H^t = 0$,

  $H=[-\ P^t\ I_r]$

Parity Check Matrix: Ex

$$v = (1\ 0\ 1) \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right] \qquad v = (1\ 0\ 1\ 0\ 1)$$

$$H = \left[\begin{array}{ccc|cc} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

$v.H^t$ is $(1\ 0\ 1\ 0\ 1) \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} = (0\ 0)$

0 1

Hamming Codes

- Single error correcting $d_{min} = 3$
- Syndrome : $s = v.H^T$, $1\times r = 1\times n.\ n\times r$
  - s=0 normal, rest 2^r-1 syndromes indicate error. Can correct one error if syndrome is unique.
  - Hamming codes: $n \le 2^r-1$

Info Word Size	Min Check bits	Total bits	Overhead
4	3	7	75%
8	4	12	50
16	5	21	31
32	6	38	19

Hamming codes: Ex: Non-positioned

$$G = \begin{array}{|cccc:ccc|} \multicolumn{4}{c:}{d0\ \ \ d3} & c1 & & c3 \\ \hline 1&0&0&0&1&1&0 \\ 0&1&0&0&1&0&1 \\ 0&0&1&0&0&1&1 \\ 0&0&0&1&1&1&1 \\ \hline \end{array}$$

$$H = \begin{array}{|ccccc:cc|} \hline 1&1&0&1&1&0&0 \\ 1&0&1&1&0&1&0 \\ 0&1&1&1&0&0&1 \\ \hline \end{array}$$

i v

(1110)→(1110 000)

data check

$$H^T = \begin{array}{|ccc|} \hline 1&1&0 \\ 1&0&1 \\ 0&1&1 \\ 1&1&1 \\ 1&0&0 \\ 0&1&0 \\ 0&0&1 \\ \hline \end{array}$$

(1110 000) H^T= (000)

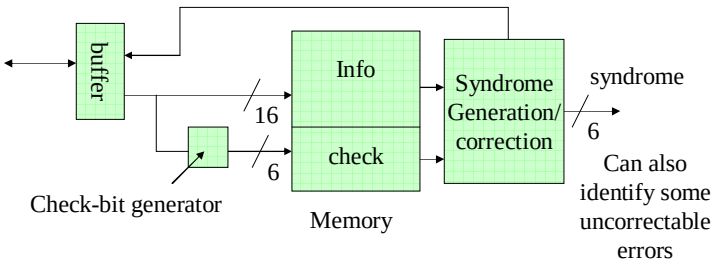
(*0*110 000) H^T= (110)

(111*1* 000) H^T= (111)

Positioned Hamming Code

Error in	d3	d0	d1	c1	d2	c2	c3
syndrome	111	110	101	100	011	010	001

ECC System



- Ex: Intel, AMD ECC chips. Cascadable 16-64 bits.
- All 1-bit errors corrected.
- Automatic *error scrubbing* using read-modify-write cycle.

Prof Bose, CSU

BCH Cyclic Codes

- Cyclic Codes: parity check codes such that cyclic shift of a code word is also a code word.
- *Polynomial*: to represent bit positions

  (n,k) cyclic code⇒generator polynomial of degree n-k

  $v(x)=M(x).G(x)$ degrees (n-1)=(k-1)(n-k)
- Ex: $G(x) = x^4+x^3+x^2+1 \Rightarrow (11101)$ (7,3) cyclic code

Message	Corres. v(x)	codeword
000 (0)	0	0000 000
110 (x^2+x)	$x^6+x^3+x^2+x$	1001 110
111 (x^2+x+1)	x^6+x^4+x+1	1010 001

Systematic Cyclic Codes

- Consider $x^{n-k}M(x) = Q(x)G(x)+ C(x)$

  Quotient Q(x): degree k-1, remainder C(x):degree n-k-1
- Then $x^{n-k}M(x)-C(x) = Q(x)G(x)$,
  thus $x^{n-k}M(x)-C(x)$ is a code word.
  - Shift message (n-k) positions
  - Fill vacated bits by remainder
- Polynomial division to get remainder
  - Note computation is linear

Systematic Cyclic Codes

- Ex: $G(x)=x^4+x^3+x^2+1$ n-k=4, n=7

message	$x^4M(x)$	C(x)	codeword
000	0(00 000)	0(0000)	000 0000
110	x^6+x^5 (1100000)	X^2+1(1001)	110 1001
111	$x^6+x^5+x^4$(1110000)	x^2(0100)	111 0100

- An error-free codeword divided by generator polynomial will give remainder 0.

Polynomial division

- Ex: $G(x)=x^4+x^3+x^2+1$ n-k=4, n=7,
  M=(110), $x^4M(x)$ is x^6+x^5, remainder is x^3+1.

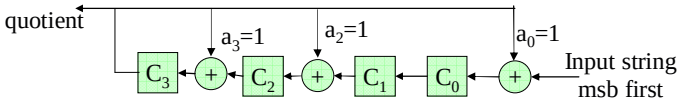
$$
\begin{array}{r|l}
 & x^2 \quad +1 \\
\hline
x^4 \; +x^3 \; +x^2 \; +1 & x^6 \; +x^5 \\
 & x^6 \; +x^5 \; +x^4 \quad +x^2 \\
\hline
 & x^4 \quad +x^2 \\
 & x^4 \; +x^3 \; +x^2 \quad +1 \\
\hline
 & x^3 \quad +1
\end{array}
$$

- Code word then is (110 1001)

  remainder

LFSR: Poly. Div. Circuit

- Ex: $G(x)=x^4+x^3+x^2+1$ n-k=4, C(x) of degree n-k-1=3



1. Clear shift register.
2. Shift (n-k) message bits in.
3. K shift lefts (hence shift out k bits of quotient)
4. Disable feedback, shift out (n-k) bit remainder.

- *Linear feedback shift Register* used for both encoding and checking.

LFSRs

- Remainder is a *signature*. If good and faulty message have same signature, there is an *aliasing error.*
- Error detection properties: Smith
  - For $k\rightarrow\infty$, P{an aliasing error} is $2^{-(n-k)}$, provided all error patterns are equally likely.
  - All single errors are detectable, if poly has 2 or more non-zero coefficients.
  - All (n-k) bit burst errors are detected, if coefficient of x^0 is 1.
- Other LFSR implementations: parallel inputs, exors only in the feedback paths.

Not impressive

Autonomous LFSRs (ALFSR)

- ALFSR: LFSR with input=0.
- If polynomial is *primitive*, its state will cycle through all (2^{n-k-1}-1) combinations, except (0,0,..0,0).
- A list of polynomials of various degrees is available.
- Alternatives to ALFSR:
  - GLFSR
  - Antirandom

Some resources

- http://www-math.cudenver.edu/~wcherowi/courses/m5410/m5410fsr.html
  Linear Feedback Shift Registers, Golomb's Principles
- **http://theory.lcs.mit.edu/~madhu/FT01/**
  Algorithmic Introduction to Coding Theory

An interesting property:

- **Theorem 1 : Let H be a parity-check matrix for a linear (n,k)-code C defined over F. Then every set of s-1 columns of H are linearly independent if and only if C has minimum distance at least s.**