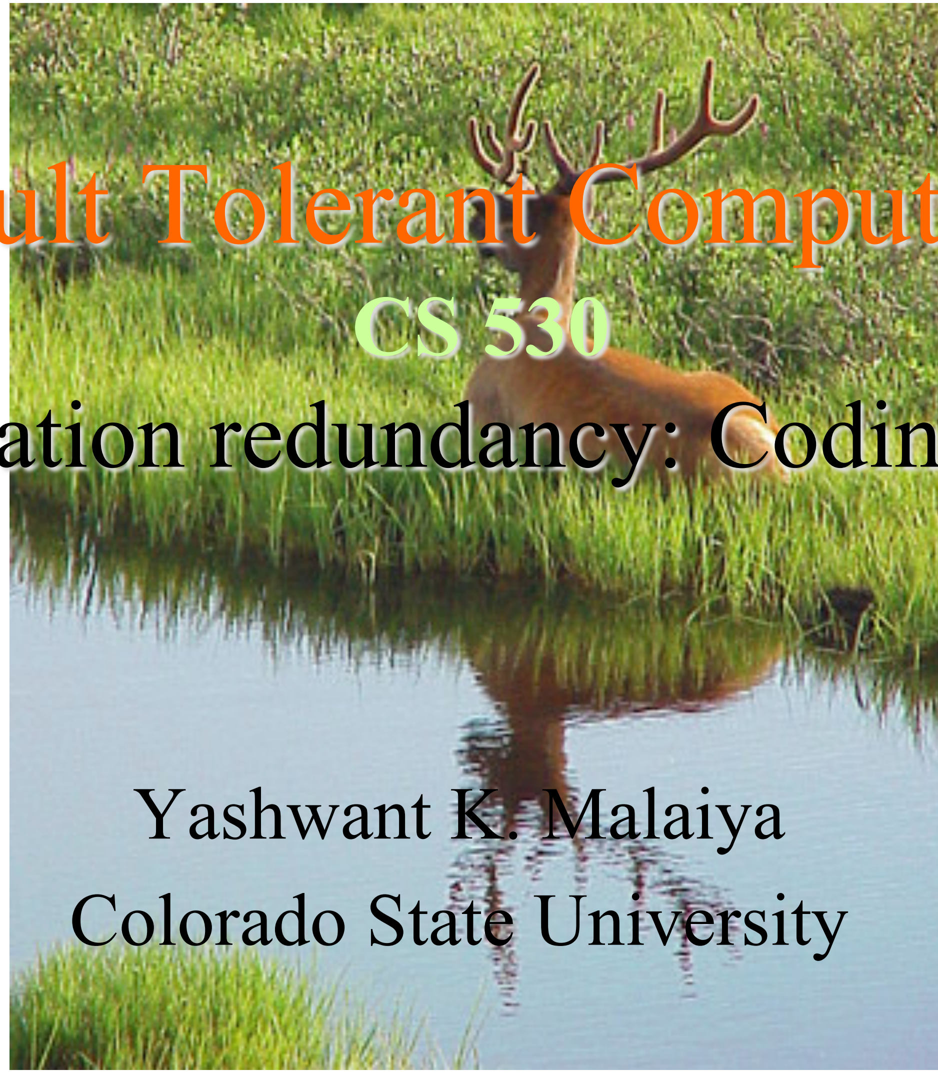


Fault Tolerant Computing

CS 530

Information redundancy: Coding theory

Yashwant K. Malaiya
Colorado State University

Information redundancy: Outline

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

Redundancy at the Bit level

- Errors can bits to be flipped during transmission or storage.
- An extra parity bit can detect if a bit in the word has flipped.
- Some errors an be corrected if there is enough redundancy such that the correct word can be guessed.
- John Tukey (Proncetpn/AT&T): “bit” 1948
- Hamming codes: 1950s
- Teletype, ASCII: 1960: 7+1 Parity bit
- Cyclic Codes: 1960

304,805 letters in the Torah:
Count as a signature

Redundancy at the Bit level

Even/odd parity (1)

- Errors can bits to be flipped during transmission/storage.
- Even/odd parity:
  - is basic method for detecting if one bit (or an odd number of bits) has been switched by accident.
- Odd parity:
  - The number of 1-bit must add up to an odd number
- Even parity:
  - The number of 1-bit must add up to an even number

Even/odd parity (2)

- It is known which parity it is being used in the system.
- If it uses an even parity:
  - If the number of of 1-bit add up to an odd number then it knows there was an error:
- If it uses an odd:
  - If the number of of 1-bit add up to an even number then it knows there was an error:
- However, If an even number of 1-bit is flipped the parity will still be the same. But an error occurs
  - The even/parity can't this detect this error:

Even/odd parity (3)

- It is useful when an odd number of 1-bits is flipped.
- Suppose we have an 7-bit binary word (7-digits).
  - Need to add 1 (parity bit) to the binary word.
  - You now have 8 digit word.
  - However, the computer knows that the added bit is a parity bit and therefore ignore it.
- If Pr{1 bit error}=0.01,
  - Pr{2 errors} = 0.01x0.01 = 0.0001 if if errors are statistically independent

Example (1)

- Suppose you receive a binary bit word "0101" and you know you are using an odd parity.
- Is the binary word corrupted?
- The answer is yes:
    - There are 2 1-bit, which is an even number
    - We are using an odd parity
    - So there must have an error.
- Do we know which bit is in error?
    - No, not enough redundancy.
    - Correction not possible

Parity Bit

- A single bit is appended to each data chunk
  - makes the number of 1 bits even/odd
- Example: even parity
  - `1000000(1)`
  - `1111101(0)`
  - `1001001(1)`
- Example: odd parity
  - `1000000(0)`
  - `1111101(1)`
  - `1001001(0)`

Parity Checking

- Assume we are using even parity with 7-bit ASCII.
- The letter V in 7-bit ASCII is encoded as 0110101.
- How will the letter V be transmitted?
  - Because there are four 1s (an even number), parity is set to zero.
  - This would be transmitted as: 00110101.
- If we are using an odd parity:
  - The letter V will be transmitted as 10110101

Formal discussion: Coding Theory

- The following slides discuss coding theory in formal terms.

Coding theory: Overview

- **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
  - ***Linear*** Codes: Check-bits are linear combinations of info bits. Linear combination of code words is a code word.
- **Code words**: are legal part of the code.

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

Hamming distance : number of bits that are different

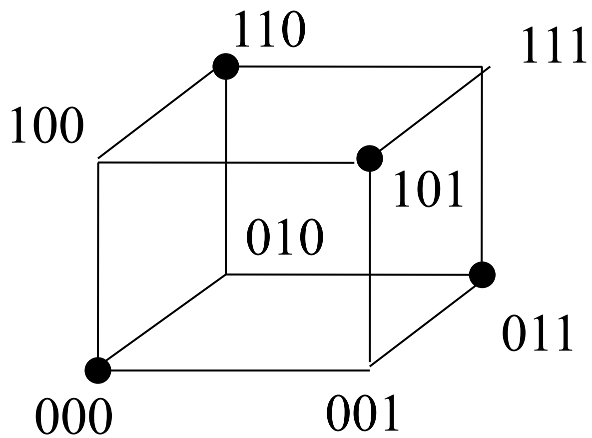
- **Minimum distance**: min of all hamming distance between all possible pairs of code words.

Ex 1: consider code:

000
011
101
110

Min distance=2

Detection Capability



Ex 1: consider code:

000
011
101
110

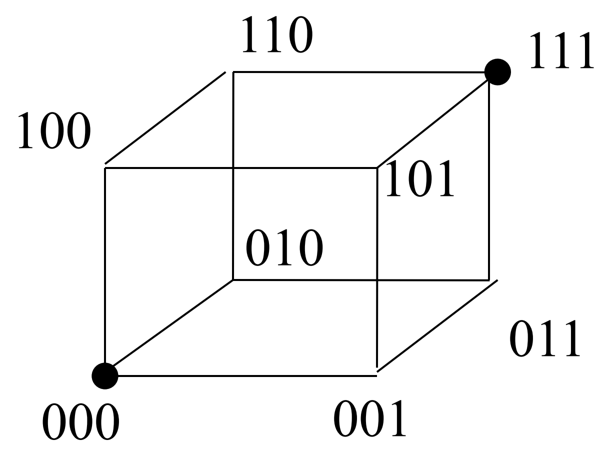
- All single bit errors result in non-code words. Thus all single-bit errors are detectable.
- **Error detection capability**: min Hamming dist d_{min}, p: number of errors that can be detected

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

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:

$$2t+1 \leq d_{min} \qquad \text{or} \qquad t_{max}=\lfloor (d_{min}-1)/2 \rfloor$$

Proof?

Parity Check Codes

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

  $i.G = v$ i: $1 \times k$ info, v : $1 \times n$ code word

- For systematic code: $G=[I_k\ P]$ I_k: $k \times k$, P: $k \times (n-k)$

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

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

Convention:
n: total bits
k: information bits
r = n-k : check bits

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

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

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

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: $n \times r$, $\mathbf{0}$: $1 \times r$
- Corrupted information: $w = v+e$ all $1 \times n$

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

$$= s \quad \textit{syndrome of error}$$

Syndrome is $1xr$
r: check bits

- 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

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

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

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

0   1

Hamming Codes

- Single error correcting requiring $d_{min} = 3$
- 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 for each error.
  - Thus, Hamming codes must have property: $n \leq 2^r - 1$

Info Word Size	Min Check bits	Total bits	Overhead
4	3 (why not 2?)	7	75%
8	4	12	50
16	5	21	31
32	6	38	19

Convention:
n: total bits
k: information bits
r: check bits

Hamming codes: Ex: Non-positioned

$$G = \begin{array}{ccccccc} d0 & & & d3 & c1 & & c3 \\ \end{array}$$

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

$$\begin{array}{c} \mathbf{i} \\ (1110) \end{array} \rightarrow \begin{array}{c} \mathbf{v} \\ (\underbrace{1110}_{\text{data}}\ \underbrace{000}_{\text{check}}) \end{array}$$

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

$$H^T = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$(1110\ 000)\quad H^T = \quad (000)$

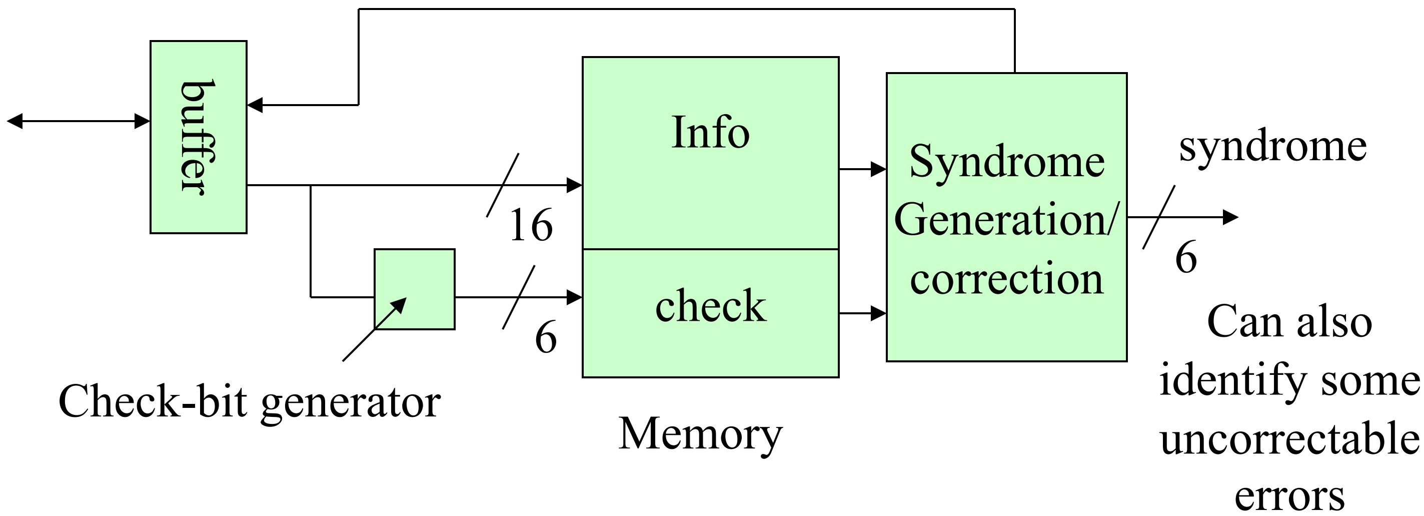
$(\mathit{0}110\ 000)\quad H^T = \quad (110)$

$(111\mathit{1}\ 000)\quad H^T = \quad (111)$

"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 R.C. Bose, (-1987)
Colorado State

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 $\Rightarrow$ generator polynomial of degree n-k

$v(x)=M(x).G(x)$ degrees $(n-1)=(k-1)(n-k)$

- Ex: $G(x) = x^4+x^3+x^2+1 \Rightarrow (11101)$ degree 4 (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 0011

$(x^2+x)(x^4+x^3+x^2+1\ )$
$=(x^6+0.x^5+0.x^4+x^3+x^2+x)$
$=(x^6+x^3+x^2+x)$

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 to the left
  - Fill vacated bits by remainder
- Polynomial division to get remainder
  - Note computation is *linear*

Convention:
n: total bits
k: information bits
r: check bits

Systematic Cyclic Codes

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

message	$x^4M(x)$	Remainder $C(x)$	codeword
000	0 (0000 000)	0(0000)	000 0000
110	x^6+x^5 (1100000)	X^3+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

Polynomial division

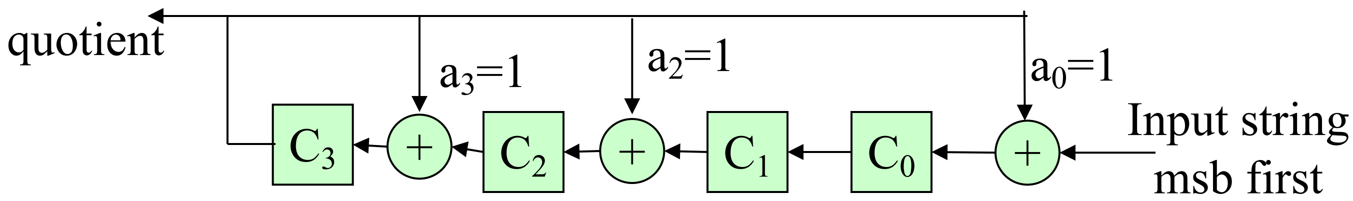
- Ex: $G(x)=x^4+x^3+x^2+1$ $n-k=4$, $n=7$,
  $M=(110)$, $x^4M(x)$ is x^6+x^5, remainder is x^3+1.

$$
\begin{array}{r|llllll}
 & x^2 & & +1 & & & \\
\cline{2-4}
x^4 \quad +x^3 \quad +x^2 \quad +1 & x^6 & +x^5 & & & & \\
 & x^6 & +x^5 & +x^4 & & +x^2 & \\
\cline{2-6}
 & & & x^4 & & +x^2 & \\
 & & & x^4 & +x^3 & +x^2 & +1 \\
\cline{4-7}
 & & & & x^3 & & +1
\end{array}
$$

- Code word then is
  $(110\ \underbrace{1001}_{\text{remainder}})$

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 1980

Not impressive

  - 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.

n-k: number of check bits

Autonomous LFSRs (ALFSR)

- ALFSR: LFSR with input=0.
- If polynomial is *primitive* *(irreducible)*, 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.ucdenver.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.**