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
  \[ D(x,y) = \sum (x_k \oplus y_k) \]
  - \( D(001,010) = 2 \)
  - \( D(000,111) = 3 \)
- **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

- All single bit errors result in non-code words. Thus all single-bit errors are detectable.
- Error detection capability: min Hamming dist \( d_{\text{min}} \), \( p \): number of errors that can be detected
  \[ p + 1 \leq d_{\text{min}} \text{ or } p_{\text{max}} = d_{\text{min}} - 1 \]

**Ex 1**: consider code:
- 000
- 011
- 101
- 110
Errors Correction Capability

Ex 2: Consider a code

\[
\begin{array}{c}
000 \\
111 \\
\end{array}
\]

- 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:

\[
2t + 1 \leq d_{\text{min}} \quad \text{or} \quad T_{\text{max}} = \left\lfloor \frac{d_{\text{min}} - 1}{2} \right\rfloor
\]

Parity Check Codes

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

\[
i.G = v \quad i: 1 \times k \text{ info, } v : 1 \times n \text{ code word}
\]

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

Ex: \( k=3, r=n-k=2 \)

\[
G = \begin{bmatrix}
1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0
\end{bmatrix}
\]
Parity Check Codes: Code Word Generation

- Ex: info \( i = (1\ 0\ 1) \)
  
  \[
  G = \begin{bmatrix}
  1 & 0 & 0 & 1 \\
  0 & 1 & 0 & 1 \\
  0 & 0 & 1 & 1 \\
  
  \end{bmatrix}
  \]

  then
  
  \[
  v = \begin{bmatrix}
  1 & 0 & 0 & 1 & 1 \\
  0 & 1 & 0 & 1 & 1 \\
  0 & 0 & 1 & 1 & 0 \\
  \end{bmatrix}
  \]

  \[
  v = \begin{bmatrix}
  1 & 0 & 1 \\
  \text{info} \\
  0 & 1 & 1 \\
  \text{check} \\
  \end{bmatrix}
  \]

  Note: Matrix multiplication: (dimensions) \( ab \times bc = abc \)

Parity Check Codes: Parity Check Matrix \( H \)

- If \( v \) is a code word: \( v.H^i = 0 \)
  
  \( H: n \times r, 0: 1 \times r \)

- Corrupted information: \( w = v + e \) all \( 1 \times n \)
  
  \[
  w.H^i = (v+e).H^i = 0 + e. H^i
  \]
  
  \( = s \) syndrome of error

- For t-error correcting code, syndrome is unique for up to \( t \) errors & can be used for correction.

- For systematic codes \( G, H^i = 0, \)
  
  \[
  H = [-P^t \ I_r]
  \]
Hamming Codes

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

<table>
<thead>
<tr>
<th>Info Word Size</th>
<th>Min Check bits</th>
<th>Total bits</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>7</td>
<td>75%</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>12</td>
<td>50</td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td>21</td>
<td>31</td>
</tr>
<tr>
<td>32</td>
<td>6</td>
<td>38</td>
<td>19</td>
</tr>
</tbody>
</table>
Hamming codes: Ex: Non-positioned

\[ G = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
\end{bmatrix} \]

\[ H = \begin{bmatrix}
1 & 1 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
\end{bmatrix} \]

\[ (1110 \ 000) \ \text{H}^T = (000) \]

\[ (0110 \ 000) \ \text{H}^T = (110) \]

\[ (1111 \ 000) \ \text{H}^T = (111) \]

Positioned Hamming Code

**ECC System**

- Ex: Intel, AMD ECC chips. Cascadable 16-64 bits.
- All 1-bit errors corrected.
- Automatic *error scrubbing* using read-modify-write cycle.
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: \( G(x) = x^4+x^3+x^2+1 \) \( (11101) \) \( (7,3) \) cyclic code

<table>
<thead>
<tr>
<th>Message</th>
<th>( v(x) )</th>
<th>codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>000 (0)</td>
<td>0</td>
<td>0000 000</td>
</tr>
<tr>
<td>110 (x^3+x)</td>
<td>( x^6+x^3+x^2+x )</td>
<td>1001 110</td>
</tr>
<tr>
<td>111 (x^2+x+1)</td>
<td>( x^6+x^4+x+1 )</td>
<td>1010 001</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>message</th>
<th>( x^6M(x) )</th>
<th>( C(x) )</th>
<th>codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
</tr>
<tr>
<td>110</td>
<td>( x^6 + x^5 ) ( (1100000) )</td>
<td>( x^2 + 1(1001) )</td>
<td>110 1001</td>
</tr>
<tr>
<td>111</td>
<td>( x^6 + x^5 + x^4 ) ( (1110000) )</td>
<td>( x^2(0100) )</td>
<td>111 0100</td>
</tr>
</tbody>
</table>

- 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 \( x^3 + 1 \).

\[
\begin{array}{cccc}
  x^4 & +x^3 & +x^2 & +1 \\
\hline
  x^6 & +x^3 & +x^4 & +x^2 & +1 \\
  x^6 & +x^3 & +x^4 & +x^2 & +1 \\
  x^4 & +x^3 & +x^2 & +1 \\
  x^3 & +1 & \\
\end{array}
\]

- 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 \( n-k-1=3 \)

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 \( k \to \infty \), \( P \{ \text{an aliasing error} \} = 2^{- (n-k)} \), 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 \( x^k \) is 1.

- Other LFSR implementations: parallel inputs, exors only in the feedback paths.
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](http://www-math.cudenver.edu/~wcherowi/courses/m5410/m5410fsr.html) Linear Feedback Shift Registers, Golomb's Principles
- [http://theory.lcs.mit.edu/~madhu/FT01/](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 \(s\).