Error Detection and Correction Part 2

Dr. William H. Robinson
September 8, 2004

http://eecs.vanderbilt.edu/courses/cs231/

Topics

"An error doesn't become a mistake until you refuse to correct it."

– Orlando A. Battista

• General announcements
• Error detecting codes
• Error correcting codes
• Hamming algorithm
• Assignment
  – Continue reading Chapter 2 in Tanenbaum

Guest Lecture

• Dr. Robinson will be on travel
  – Monday, September 13, 2004

• Dr. Larry Dowdy will cover the lecture
  – Lecture topics will remain on schedule
  – Responsible for material presented

Homework Assignment

• Posted on class website schedule page

• Due Wednesday, September 15, 2004

• Remember your TA!!!
  – Jason Tan
  – Office hours: 3pm – 4:30pm, Tuesdays
    4pm – 5:30pm, Thursdays
Where Do Errors Come From?

- Voltage spikes from power supply
- Coupling with nearby signals
- Noise on transmission line
- Radiation
- Manufacturing defect
- Etc.

For Error Detection

- **Hamming Distance**
  - Number of bit positions in which two codewords differ

- **Minimum Distance**
  - Minimum Hamming distance between all distinct pairs of codewords

A code with minimum distance $d_{\text{min}}$ can detect all error patterns of weight less than or equal to $(d_{\text{min}} - 1)$

Parity Check

- For even or odd parity
  - $d_{\text{min}}$ equals 2
  - Detects error patterns $\leq (d_{\text{min}} - 1)$
  - Generates an error flag

- Uses of parity check
  - Memory
  - Disk storage

For Error Correction

- **Hamming Distance**
  - Number of bit positions in which two codewords differ

- **Minimum Distance**
  - Minimum Hamming distance between all distinct pairs of codewords

A code with minimum distance $d_{\text{min}}$ can correct all error patterns of weight less than or equal to $[(d_{\text{min}} - 1)/2]$
Error Correction

- Scenario: transmit a yes/no answer
  - Codeword for “yes” = 1111₂
  - Codeword for “no” = 0000₂

- Minimum distance equals 4
  - Error correction less than or equal to \( \left( \frac{d_{\text{min}}}{} - 1 \right)/2 \)

- Therefore our code corrects all errors ≤ 1.5
  - The 0.5 correction capability isn’t used because errors occur as whole bits
  - However, the 0.5 implies additional detection capability

Hamming Codes

- First major class of binary codes designed for error correction

- Originally used in error control for long-distance telephony

- Encodes parity for groups of bits within data

Error Correcting for 4-Bit Words

Encode 1101₂ with even parity

What about a single error?
Error Correcting for 4-Bit Words

Encode 0100₂ with even parity

```
0 1 0 0 1 1 1
m₃ m₂ m₁ m₀ rₐ rₐ rₐ
```

What about a single error?

```
0 1 0 0 1 0 1
m₃ m₂ m₁ m₀ rₐ rₐ rₐ
```

Boolean Exclusive Sum - XOR Function

• Logic operation
• Output is "true" when either input is "true" but not both
• Corresponds to addition in base 2

```
A B F
0 0 0
0 1 1
1 0 1
1 1 0
```

Method for Parity Check

```
0 1 0 0 1 0 1
m₃ m₂ m₁ m₀ rₐ rₐ rₐ
```

• Parity check for Circle A
  \(- m₃ \oplus m₂ \oplus m₁ \oplus rₐ \)

• Parity check for Circle B
  \(- m₃ \oplus m₂ \oplus m₀ \oplus rₐ \)

• Parity check for Circle C
  \(- m₂ \oplus m₁ \oplus m₀ \oplus rₐ \)

Even parity \(\rightarrow 0\)
Odd parity \(\rightarrow 1\)
Hamming Algorithm

- In a Hamming code
  - \( r \) parity bits added to \( m \)-bit word
  - Forms codeword with length \((m + r)\) bits

- Bit numbering
  - Starts at 1 with leftmost (high-order) bit
  - All powers of 2 are parity bits
  - Remaining bits are for data

Requirements for Single-Bit Error Correction

<table>
<thead>
<tr>
<th>Word size</th>
<th>Check bits</th>
<th>Total size</th>
<th>Percent overhead</th>
</tr>
</thead>
<tbody>
<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>
<tr>
<td>64</td>
<td>7</td>
<td>71</td>
<td>11</td>
</tr>
<tr>
<td>128</td>
<td>8</td>
<td>136</td>
<td>6</td>
</tr>
<tr>
<td>256</td>
<td>9</td>
<td>255</td>
<td>4</td>
</tr>
<tr>
<td>512</td>
<td>10</td>
<td>522</td>
<td>2</td>
</tr>
</tbody>
</table>

Fig. 2-13. Number of check bits for a code that can correct a single error.

Bit Numbering for Hamming Algorithm

Given an 8-bit data word to encode

Parity bit assignment table:

<table>
<thead>
<tr>
<th>Bit No.</th>
<th>Bit No. in Binary</th>
<th>Encoded by</th>
<th>Encodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0001</td>
<td>3, 5, 7, 9, 11</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>3, 6, 7, 10, 11</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0100</td>
<td>2, 1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>5, 6, 7, 12</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>4, 1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>4, 2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>4, 2, 1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>9, 10, 11, 12</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1000</td>
<td>8, 1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1010</td>
<td>8, 2</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1011</td>
<td>8, 2, 1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1100</td>
<td>8, 4</td>
<td></td>
</tr>
</tbody>
</table>
Example

Given an 8-bit data word to encode

- Codeword has 12 bits (8 data, 4 parity)
  - Bit 1 checks: 1, 3, 5, 7, 9, 11
  - Bit 2 checks: 2, 3, 6, 7, 10, 11
  - Bit 4 checks: 4, 5, 6, 7, 12
  - Bit 8 checks: 8, 9, 10, 11, 12

Class Exercise

- Compute the codeword for the 8-bit data:
  - 0x71
  - 0111 0001

  \[
  \begin{array}{cccccccccccccc}
  1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
  0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
  \end{array}
  \]

Class Exercise

- Find the bit that is incorrect in the following codeword. This 12 bit codeword encodes 8 data bits.
  - 0x19D
  - 0001 1001 1101
  - Bit 1 checks: 1, 3, 5, 7, 9, 11
  - Bit 2 checks: 2, 3, 6, 7, 10, 11
  - Bit 4 checks: 4, 5, 6, 7, 12
  - Bit 8 checks: 8, 9, 10, 11, 12

Summary

- Data can be encoded to detect or correct errors
- Hamming codes were the first major class of binary codes designed for error correction
- Minimum Hamming distance determines the error detection/correction capability of a code