16.36: Communication Systems Engineering

Lectures 12/13: Channel Coding

April 2/4, 2002

Eytan Modiano
Channel Coding

- When transmitting over a noisy channel, some of the bits are received with errors

**Example:** Binary Symmetric Channel (BSC)

- **Q:** How can these errors be removed?
- **A:** Coding: the addition of redundant bits that help us determine what was sent with greater accuracy
Example (Repetition code)

Repeat each bit n times (n-odd)

<table>
<thead>
<tr>
<th>Input</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000……..0</td>
</tr>
<tr>
<td>1</td>
<td>11.........1</td>
</tr>
</tbody>
</table>

Decoder:
- If received sequence contains n/2 or more 1’s decode as a 1 and 0 otherwise
  - Max likelihood decoding

\[
P(\text{error} | \text{1 sent}) = P(\text{error} | \text{0 sent}) = P[\text{more than n / 2 bit errors occur}]
\]

\[
= \sum_{i=\lceil n/2 \rceil}^{n} \binom{n}{i} P_e^i (1 - P_e)^{n-i}
\]
Repetition code, cont.

• For \( P_e < 1/2 \), \( P(\text{error}) \) is decreasing in \( n \)
  
  \[ \Rightarrow \text{for any } \varepsilon, \exists n \text{ large enough so that } P(\text{error}) < \varepsilon \]

**Code Rate**: ratio of data bits to transmitted bits

  - For the repetition code \( R = 1/n \)
  
  - To send one data bit, must transmit \( n \) channel bits “bandwidth expansion”

• In general, an \((n,k)\) code uses \( n \) channel bits to transmit \( k \) data bits

  - Code rate \( R = k/n \)

• Goal: for a desired error probability, \( \varepsilon \), find the highest rate code that can achieve \( p(\text{error}) < \varepsilon \)
The capacity of a discrete memoryless channel is given by,

\[ C = \max_{p(x)} I(X;Y) \]

**Example: Binary Symmetric Channel (BSC)**

\[
I(X;Y) = H(Y) - H(Y|X) = H(X) - H(X|Y)
\]

\[
H(X|Y) = H(X|Y=0)*P(Y=0) + H(X|Y=1)*P(Y=1)
\]

\[
H(X|Y=0) = H(X|Y=1) = P_e \log(1/P_e) + (1-P_e)\log(1/(1-P_e)) = H_b(P_e)
\]

\[
H(X) = H_b(P_e) = H(X) - H(X|Y) = H(X) - H_b(P_e)
\]

\[
H(X) = P_0 \log (1/P_0) + (1-P_0) \log (1/(1-P_0)) = H_b(P_0)
\]

\[
I(X;Y) = H_b(P_0) - H_b(P_e)
\]
Capacity of BSC

\[ I(X;Y) = H_b(P_0) - H_b(P_e) \]

- \[ H_b(P) = P \log(1/P) + (1-P) \log(1/(1-P)) \]
  - \[ H_b(P) \leq 1 \text{ with equality if } P=1/2 \]

\[ C = \max_{P_0} \{ I(X;Y) = H_b(P_0) - H_b(P_e) \} = 1 - H_b(P_e) \]

\[ C = 0 \text{ when } P_e = 1/2 \text{ and } C = 1 \text{ when } P_e = 0 \text{ or } P_e = 1 \]
Theorem: For all \( R < C \) and \( \varepsilon > 0 \); there exists a code of rate \( R \) whose error probability < \( \varepsilon \)

- \( \varepsilon \) can be arbitrarily small
- Proof uses large block size \( n \) as \( n \to \infty \) capacity is achieved

• In practice codes that achieve capacity are difficult to find
  - The goal is to find a code that comes as close as possible to achieving capacity

• Converse of Coding Theorem:
  - For all codes of rate \( R > C \), \( \exists \varepsilon_0 > 0 \), such that the probability of error is always greater than \( \varepsilon_0 \)

  For code rates greater than capacity, the probability of error is bounded away from 0
Channel Coding

• Block diagram
Approaches to coding

- **Block Codes**
  - Data is broken up into blocks of equal length
  - Each block is “mapped” onto a larger block

**Example:** (6,3) code, n = 6, k = 3, R = 1/2

<table>
<thead>
<tr>
<th>Original</th>
<th>Converted</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>000000</td>
</tr>
<tr>
<td>001</td>
<td>001011</td>
</tr>
<tr>
<td>010</td>
<td>010111</td>
</tr>
<tr>
<td>011</td>
<td>011100</td>
</tr>
<tr>
<td>100</td>
<td>100101</td>
</tr>
<tr>
<td>101</td>
<td>101110</td>
</tr>
<tr>
<td>110</td>
<td>110010</td>
</tr>
<tr>
<td>111</td>
<td>111001</td>
</tr>
</tbody>
</table>

- **An (n,k) binary block code is a collection of** $2^k$ **binary n-tuples (n>k)**
  - n = block length
  - k = number of data bits
  - n-k = number of checked bits
  - $R = k / n$ = code rate
Approaches to coding

- Convolutional Codes
  - The output is provided by looking at a sliding window of input

\[
C_{2K} = U_{2K} \oplus U_{2K-2}, \quad C_{2K+1} = U_{2K+1} \oplus U_{2K} \oplus U_{2K-1}
\]

\( \oplus \) mod(2) addition (1+1=0)
Block Codes

- A block code is systematic if every codeword can be broken into a data part and a redundant part
  - Previous (6,3) code was systematic

Definitions:

- Given \( X \in \{0,1\}^n \), the **Hamming Weight** of \( X \) is the number of 1’s in \( X \)

- Given \( X, Y \in \{0,1\}^n \), the **Hamming Distance** between \( X \) & \( Y \) is the number of places in which they differ,

\[
d_H(X, Y) = \sum_{i=1}^{n} X_i \oplus Y_i = Weight(X + Y)
\]

\[
X + Y = [x_1 \oplus y_1, x_2 \oplus y_2, \ldots, x_n \oplus y_n]
\]

- The **minimum distance** of a code is the Hamming Distance between the two closest codewords:

\[
d_{\text{min}} = \min \{d_H(C_1, C_2)\}
\]

\( C_1, C_2 \in C \)
Decoding

- \( r \) may not equal to \( u \) due to transmission errors
- Given \( r \) how do we know which codeword was sent?

**Maximum likelihood Decoding:**
Map the received n-tuple \( r \) into the codeword \( C \) that maximizes,
\[
P \{ r \mid C \text{ was transmitted} \}
\]

**Minimum Distance Decoding** (nearest neighbor)
Map \( r \) to the codeword \( C \) such that the hamming distance between \( r \) and \( C \) is minimized (i.e., \( \min d_H (r,C) \))

\( \Rightarrow \) For most channels Min Distance Decoding is the same as Max likelihood decoding
Linear Block Codes

- A \((n,k)\) linear block code (LBC) is defined by \(2^k\) codewords of length \(n\)
  
  \[ C = \{ C_1, \ldots, C_m \} \]

- A \((n,k)\) LBC is a \(K\)-dimensional subspace of \(\{0,1\}^n\)
  - \((0\ldots0)\) is always a codeword
  - If \(C_1, C_2 \in C\), \(C_1 + C_2 \in C\)

- **Theorem:** For a LBC the minimum distance is equal to the min weight \((W_{\text{min}})\) of the code
  
  \[ W_{\text{min}} = \min_{\text{over all } C_i} \text{Weight} \left( C_i \right) \]

  **Proof:** Suppose \(d_{\text{min}} = d_H (C_i, C_j)\), where \(C_1, C_2 \in C\)

  \[ d_H (C_i, C_j) = \text{Weight} \left( C_i + C_j \right), \]
  
  but since \(C\) is a LBC then \(C_i + C_j\) is also a codeword
Theorem: Any \((n,k)\) LBC can be represented in Systematic form where: data = \(x_1..x_k\), codeword = \(x_1..x_k c_{k+1}..x_n\)

- Hence we will restrict our discussion to systematic codes only

- The codewords corresponding to the information sequences:
  \(e_1 = (1,0,..0)\), \(e_2=(0,1,0..0)\), \(e_k = (0,0,..,1)\) for a basis for the code

  - Clearly, they are linearly independent

  - \(K\) linearly independent \(n\)-tuples completely define the \(K\) dimensional subspace that forms the code

<table>
<thead>
<tr>
<th>Information sequence</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>(e_1 = (1,0,..0))</td>
<td>(g_1 = (1,0,..,0, g_{(1,k+1)} \ldots g_{(1,n)}))</td>
</tr>
<tr>
<td>(e_2=(0,1,0..0))</td>
<td>(g_2 = (0,1,..,0, g_{(2,k+1)} \ldots g_{(2,n)}))</td>
</tr>
<tr>
<td>(e_k = (0,0,..,1))</td>
<td>(g_k = (0,0,..,k, g_{(k,k+1)} \ldots g_{(k,n)}))</td>
</tr>
</tbody>
</table>

- \(g_1, g_2, \ldots, g_k\) form a basis for the code
The Generator Matrix

\[ G = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_k \end{bmatrix} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{bmatrix} \]

- For input sequence \( x = (x_1, \ldots, x_k) \): \( C_x = xG \)
  - Every codeword is a linear combination of the rows of \( G \)
  - The codeword corresponding to every input sequence can be derived from \( G \)
  - Since any input can be represented as a linear combination of the basis \( (e_1, e_2, \ldots, e_k) \), every corresponding codeword can be represented as a linear combination of the corresponding rows of \( G \)

- Note: \( x_1 \leftrightarrow C_1, x_2 \leftrightarrow C_2 \Rightarrow x_1 + x_2 \leftrightarrow C_1 + C_2 \)
Example

- Consider the (6,3) code from earlier:

  \[ 100 \rightarrow 100101; \quad 010 \rightarrow 010111; \quad 001 \rightarrow 001011 \]

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

Codeword for \((1,0,1) = (1,0,1)G = (1,0,1,1,1,0)\)

\[
G = \begin{bmatrix}
I_K & P_{Kx(n-K)}
\end{bmatrix}
\]

\[I_K = K\times K \text{ identity matrix}\]
The parity check matrix

\[ H = \begin{bmatrix} P^T & I_{(n-K)} \end{bmatrix} \]

\[ I_{(n-K)} = (n - K) \times (n - K) \text{ identity matrix} \]

Example:

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

Now, if \( c_i \) is a codeword of \( C \) then, \( c_i H^T = \vec{0} \)

- “\( C \) is in the null space of \( H \)”
- Any codeword in \( C \) is orthogonal to the rows of \( H \)
Decoding

• $v =$ transmitted codeword $= v_1 \ldots v_n$
• $r =$ received codeword $= r_1 \ldots r_n$
• $e =$ error pattern $= e_1 \ldots e_n$
• $r = v + e$

• $S = rH^T =$ Syndrome of $r$
  $= (v+e)H^T = vH^T + eH^T = eH^T$

• $S$ is equal to ‘0’ if and only if $e \in C$
  – I.e., error pattern is a codeword

• $S \neq 0 \Rightarrow$ error detected
• $S = 0 \Rightarrow$ no errors detected (they may have occurred and not detected)

• Suppose $S \neq 0$, how can we know what was the actual transmitted codeword?
Syndrome decoding

- Many error patterns may have created the same syndrome
  For error pattern $e_0 \Rightarrow S_0 = e_0 H^T$

  Consider error pattern $e_0 + c_i$ ($c_i \in C$)
  $$S'_0 = (e_0 + c_i) H^T = e_0 H^T + c_i H^T = e_0 H^T = S_0$$

- So, for a given error pattern, $e_0$, all other error patterns that can be expressed as $e_0 + c_i$ for some $c_i \in C$ are also error patterns with the same syndrome

- For a given syndrome, we can not tell which error pattern actually occurred, but the most likely is the one with minimum weight
  - Minimum distance decoding

- For a given syndrome, find the error pattern of minimum weight ($e_{\text{min}}$) that gives this syndrome and decode: $r' = r + e_{\text{min}}$
Standard Array

\[
\begin{array}{cccccc}
C_1 & C_2 & \ldots & C_M & \text{Syndrome} \\
e_1 & e_1 + C_2 & e_1 + C_M & S_1 \\
M & e_2 + C_2 & e_2 + C_M & S_2 \\
e_{2^{(n-K)}-1} & & & S_{2^{(n-K)}-1} \\
\end{array}
\]

- Row 1 consists of all M codewords
- Row 2 \( e_1 = \text{min weight } n\text{-tuple not in the array} \)
  - i.e., the minimum weight error pattern
- Row i, \( e_i = \text{min weight } n\text{-tuple not in the array} \)
- All elements of any row have the same syndrome
  - Elements of a row are called “co-sets”
- The first element of each row is the minimum weight error pattern with that syndrome
  - Called “co-set leader”
Decoding algorithm

- Receive vector \( r \)

1) Find \( S = rH^T \) = syndrome of \( r \)

2) Find the co-set leader \( e \), corresponding to \( S \)

3) Decode: \( C = r + e \)

- “Minimum distance decoding”
  - Decode into the codeword that is closest to the received sequence
Example (syndrome decoding)

• Simple (4,2) code

Data  | codeword
---|---
00   | 0000
01   | 0101
10   | 1010
11   | 1111

Standard array

<table>
<thead>
<tr>
<th></th>
<th>0000</th>
<th>0101</th>
<th>1010</th>
<th>1111</th>
<th>Syndrome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>1101</td>
<td>0010</td>
<td>0111</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>0100</td>
<td>0001</td>
<td>1110</td>
<td>1011</td>
<td>01</td>
<td></td>
</tr>
<tr>
<td>1100</td>
<td>1001</td>
<td>0110</td>
<td>0011</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

Suppose 0111 is received, S = 10, co-set leader = 1000

Decode: \( C = 0111 + 1000 = 1111 \)
Minimum distance decoding

- Minimum distance decoding maps a received sequence onto the nearest codeword.
- If an error pattern maps the sent codeword onto another valid codeword, that error will be undetected (e.g., e3).
  - Any error pattern that is equal to a codeword will result in undetected errors.
- If an error pattern maps the sent sequence onto the sphere of another codeword, it will be incorrectly decoded (e.g., e2).
Performance of Block Codes

• Error detection: Compute syndrome, $S \neq 0 \Rightarrow$ error detected
  – Request retransmission
  – Used in packet networks

• A linear block code will detect all error patterns that are not codewords

• Error correction: Syndrome decoding
  – All error patterns of weight $< d_{\text{min}}/2$ will be correctly decoded
  – This is why it is important to design codes with large minimum distance ($d_{\text{min}}$)
  – The larger the minimum distance the smaller the probability of incorrect decoding
Hamming Codes

- Linear block code capable of correcting single errors
  - $n = 2^m - 1$, $k = 2^m - 1 - m$
    (e.g., (3,1), (7,4), (15,11)…)
  - $R = 1 - m/(2^m - 1)$ => very high rate
  - $d_{\text{min}} = 3$ => single error correction

- Construction of Hamming codes
  - Parity check matrix ($H$) consists of all non-zero binary $m$-tuples

Example: (7,4) hamming code ($m=3$)

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

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