[5.1 B]

Digital Communication Systems EES 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

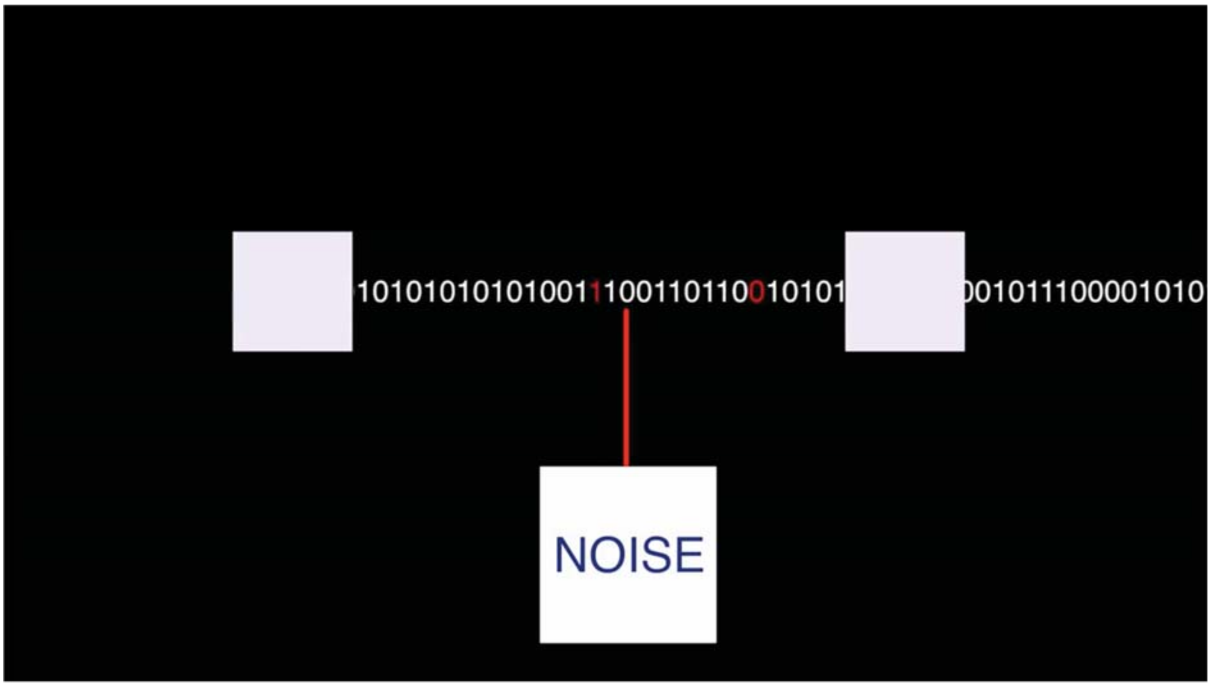
Introduction to Hamming Code

Hamming codes

- One of the earliest codes studied in coding theory.
- Named after Richard W. Hamming
  - The IEEE Richard W. **Hamming Medal**, named after him, is an award given annually by Institute of Electrical and Electronics Engineers (IEEE), for "exceptional contributions to information sciences, systems and technology“.
    - Sponsored by Qualcomm, Inc
    - Some Recipients:
      - 1988 - Richard W. Hamming
      - 1997 - Thomas M. Cover
      - 1999 - David A. Huffman
      - 2011 - Toby Berger

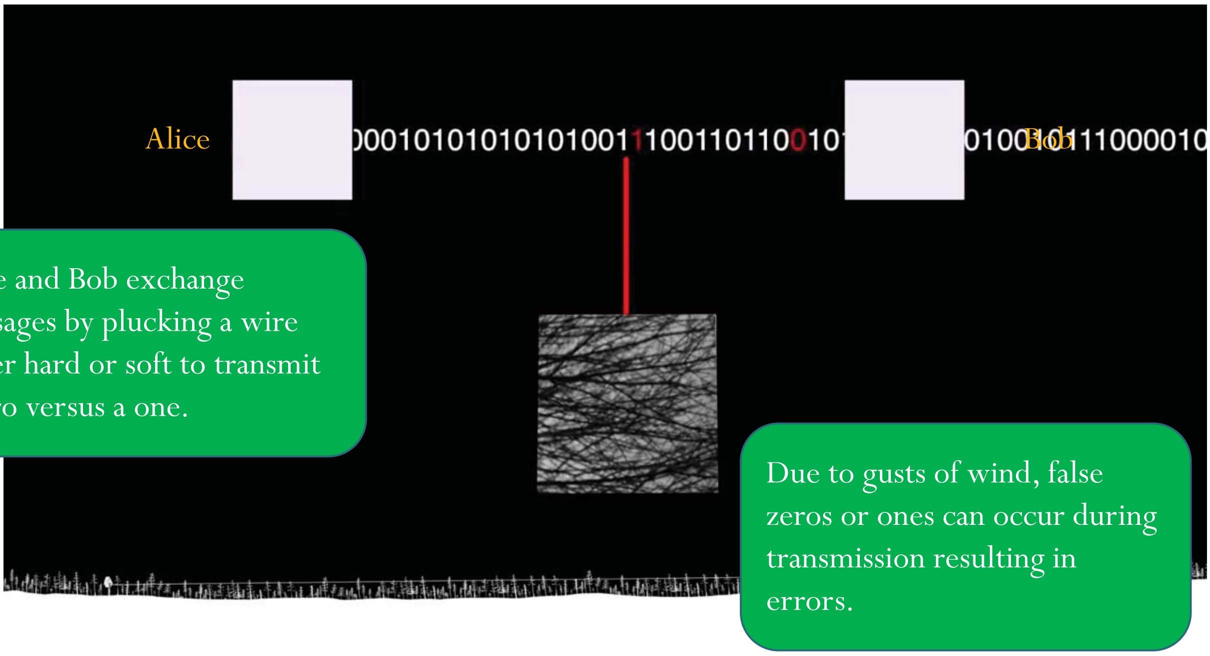
- The simplest of a class of (algebraic) error correcting codes that **can** always ***correct*** **one error in a block of bits**

Hamming codes



[https://www.youtube.com/watch?v=cBBTWcHkVVY]

Hamming codes

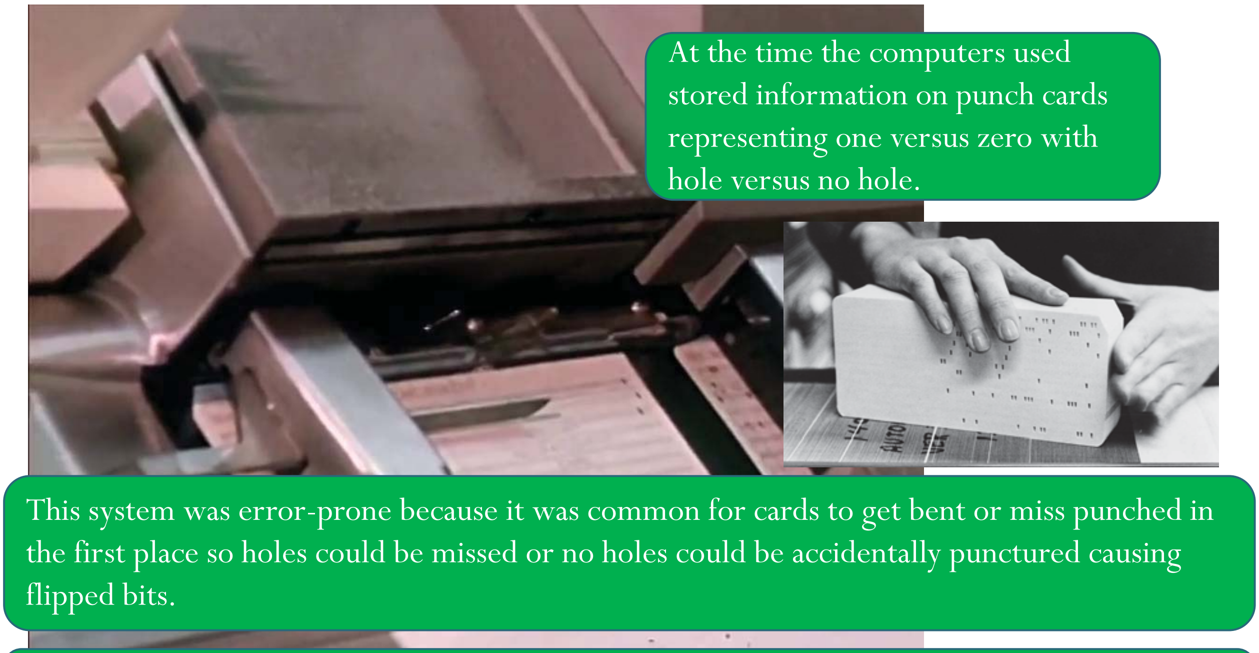


Alice and Bob exchange messages by plucking a wire either hard or soft to transmit a zero versus a one.

Due to gusts of wind, false zeros or ones can occur during transmission resulting in errors.

In the 1940s, Richard Hamming faced a similar problem while working at Bell Laboratories.

Hamming codes

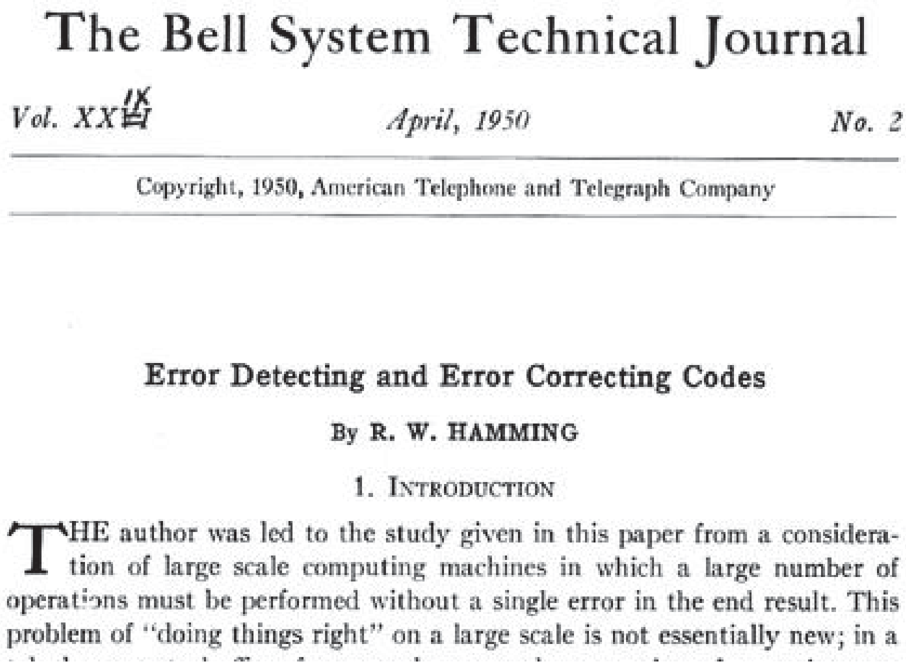


At the time the computers used stored information on punch cards representing one versus zero with hole versus no hole.

This system was error-prone because it was common for cards to get bent or miss punched in the first place so holes could be missed or no holes could be accidentally punctured causing flipped bits.

Hamming took it upon himself to devise a method which could automatically detect and correct single bit errors without interrupting calculations.

Hamming codes



The Bell System Technical Journal

Vol. XXIX *April, 1950* *No. 2*

Copyright, 1950, American Telephone and Telegraph Company

Error Detecting and Error Correcting Codes

By R. W. HAMMING

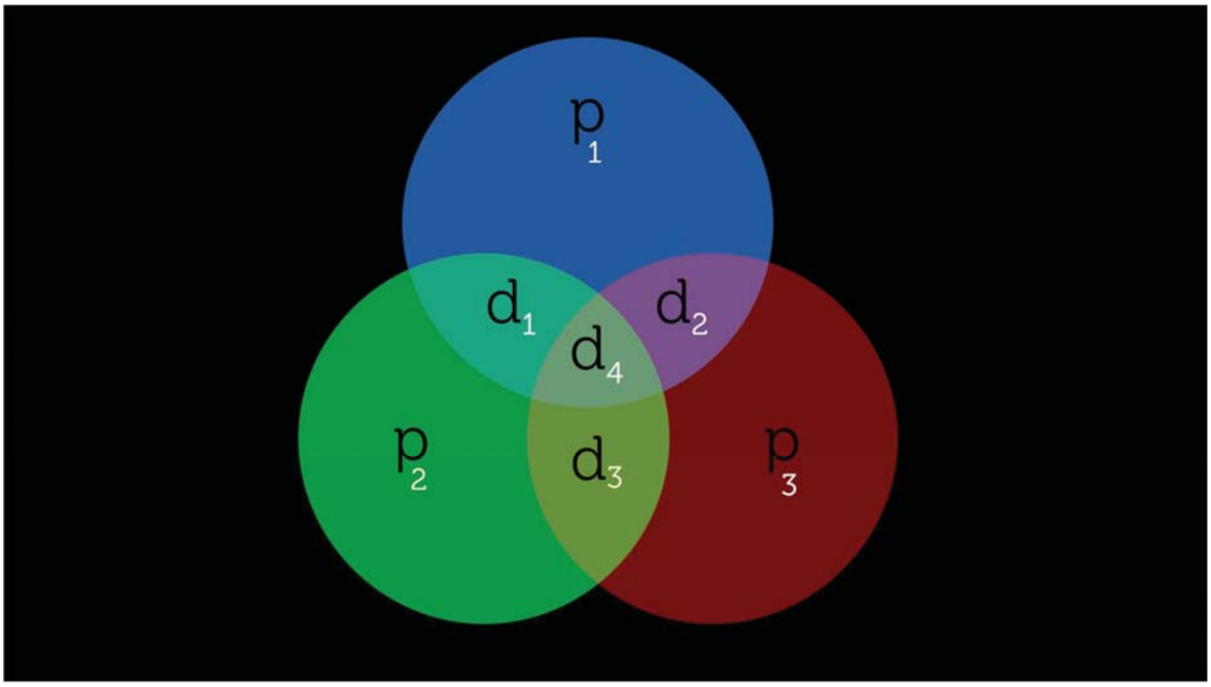
1. Introduction

THE author was led to the study given in this paper from a consideration of large scale computing machines in which a large number of operations must be performed without a single error in the end result. This problem of "doing things right" on a large scale is not essentially new; in a

Hamming codes

[https://www.youtube.com/watch?v=cBBTWcHkVVY]

Hamming codes: Ex. 1



[https://www.youtube.com/watch?v=cBBTWcHkVVY]

Digital Communication Systems
EES 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Code Structure and the Generator Matrix

Review: Even Parity

- A binary vector (or a collection of 1s and 0s) has **even parity** if and only if the number of 1s in there is even.
  - Suppose we are given the values of all the bits except one bit.
    - We can force the vector to have even parity by setting the value of the remaining bit to be the sum of the other bits.

Single-parity-check code

[1 0 1 1 0 _]

Square array

1	0	1	_
0	1	1	_
0	0	1	_
_	_	_	

Review: Even Parity

- A binary vector (or a collection of 1s and 0s) has **even parity** if and only if the number of 1s in there is even.
  - Suppose we are given the values of all the bits except one bit.
    - We can force the vector to have even parity by setting the value of the remaining bit to be the sum of the other bits.

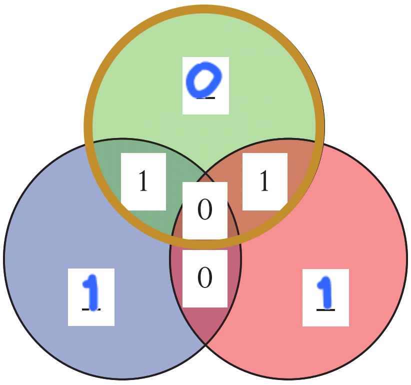
Single-parity-check code

[1 0 1 1 0 _]

Square array

1	0	1	_
0	1	1	_
0	0	1	_
_	_	_	

Hamming code

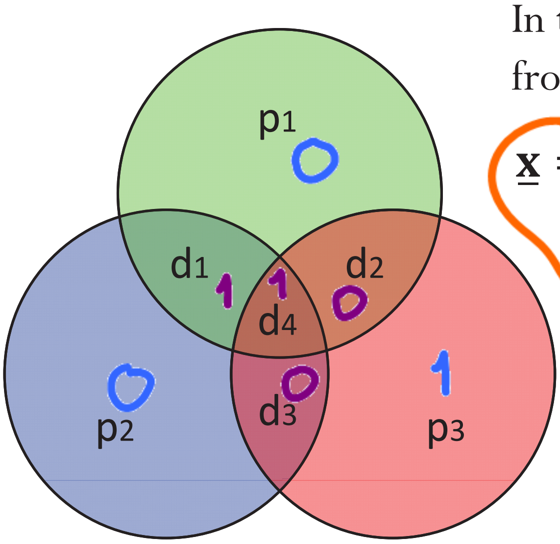


Hamming codes: Ex. 1

$d_1 d_2 d_3 d_4$

Ex. $\underline{d} = [1001]$

This is an example of Hamming $(\overset{n}{7},\overset{k}{4})$ code



In the video, the codeword is constructed from the data by

$$\begin{array}{cccccccc} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ \underline{\mathbf{x}} = [& p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4] \\ & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{array}$$

where

$$p_1 = d_1 \oplus d_2 \oplus d_4$$
$$p_2 = d_1 \oplus d_3 \oplus d_4$$
$$p_3 = d_2 \oplus d_3 \oplus d_4$$

structure of the code

- The message bits are also referred to as the data bits or information bits.
- The non-message bits are also referred to as parity check bits, checksum bits, parity bits, or check bits.

Writing the generator matrix from the code "structure"

Generator matrix: a revisit

- Fact: The 1s and 0s in the j^{th} column of $\mathbf{G}$ tells which positions of the data bits are combined ($\oplus$) to produce the j^{th} bit in the codeword.
- For the Hamming code in the previous slide,

$$\underline{\mathbf{x}} = \begin{bmatrix} p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \end{bmatrix} = \underline{d}\,G$$

$$p_1 = d_1 \oplus d_2 \oplus d_4$$
$$p_2 = d_1 \oplus d_3 \oplus d_4$$
$$p_3 = d_2 \oplus d_3 \oplus d_4$$

$$= \begin{bmatrix} d_1 & d_2 & d_3 & d_4 \end{bmatrix} \underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}}_{\mathbf{G}}$$

"Reading" the generator matrix from the codebook.

	$\underline{\mathbf{d}}$				$\underline{\mathbf{x}}$							
	0	0	0	0	0	0	0	0	0	0	0	
d_4	0	0	0	1	1	0	1	0	1	0	1	$\mathbf{g}^{(4)}$
d_3	0	0	1	0	0	0	1	0	1	1	0	$\mathbf{g}^{(3)}$
	0	0	1	1	1	0	0	0	0	1	1	
d_2	0	1	0	0	1	0	0	1	1	0	0	$\mathbf{g}^{(2)}$
	0	1	0	1	0	0	1	1	0	0	1	
	0	1	1	0	1	0	1	1	0	1	0	
	0	1	1	1	0	0	0	1	1	1	1	
d_1	1	0	0	0	1	1	1	0	0	0	0	$\mathbf{g}^{(1)}$
	1	0	0	1	0	1	0	0	1	0	1	
	1	0	1	0	1	1	0	0	1	1	0	
	1	0	1	1	0	1	1	0	0	1	1	
	1	1	0	0	0	1	1	1	1	0	0	
	1	1	0	1	1	1	0	1	0	0	1	
	1	1	1	0	0	1	0	1	0	1	0	
	1	1	1	1	1	1	1	1	1	1	1	

- One can also "read" $\mathbf{G}$ from the codebook.
- From $\underline{\mathbf{x}} = \underline{\mathbf{d}}\mathbf{G} = \sum_{j=1}^{k} d_j \mathbf{g}^{(j)}$, when we look at the message block with a single 1 at position i, then the corresponding codeword is the same as $\mathbf{g}^{(j)}$.

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

ECS 452: In-Class Exercise # 16

Instructions

1. Working alone is always permitted. However, working in groups is also allowed if social distancing can be used (via, e.g., online group chat/call). For group work, **the group cannot be the same as any of your former group after the midterm.**
2. Only one submission is needed for each group.
3. [**ENRE**] Explanation is not required for this exercise.
4. **Do not panic.**

Date: 31 / 3 / 2020			
Name	ID (last 3 digits)		

1. Consider a linear block code that uses *parity checking on a square array*:

First, we use the provided definition to write down the equations that produce the parity bits. This definition is exactly the same as the one given in lecture when we defined parity checking on a square array

b_1	b_3	p_1
b_2	b_4	p_2
p_3	p_4	

$p_1 = b_1 \oplus b_3$

$p_2 = b_2 \oplus b_4$

$p_3 = b_1 \oplus b_2$

$p_4 = b_3 \oplus b_4$

Each parity bit p_i is calculated such that the corresponding row or column has even parity.
Suppose the following bits arrangement is used in the codeword:

$$\underline{\mathbf{x}} = \left(b_1 \quad b_2 \quad p_1 \quad p_2 \quad b_3 \quad p_3 \quad b_4 \quad p_4\right).$$

a. Find the generator matrix $\mathbf{G}$.

Recall that the 1s and 0s in the j^{th} column of **G** tells which positions of the data bits are combined ($\oplus$) to produce the j^{th} bit in the codeword.

$$\begin{array}{c|cccccccc}
 & b_1 & b_2 & p_1 = b_1 \oplus b_3 & p_2 = b_2 \oplus b_4 & b_3 & p_3 = b_1 \oplus b_2 & b_4 & p_4 = b_3 \oplus b_4 \\
\hline
\mathbf{G} = & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1
\end{array}$$

b. Find the codeword for the message $\underline{\mathbf{b}} = [1 \quad 0 \quad 1 \quad 0]$.

Method 1: First, we fill out the array above with the message. Then, we calculate the parity bits.

1	1	p_1
0	0	p_2
p_3	p_4	

$\Rightarrow$

1	1	0
0	0	0
1	1	

The codeword can be read directly from the array: $\underline{\mathbf{x}} = \left(1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1\right)$.

Method 2: It is still true that $\underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G}$. Therefore, we can still use our old technique: to find $\underline{\mathbf{x}}$ when $\underline{\mathbf{b}} = [1 \quad 0 \quad 1 \quad 0]$, we simply need to add the first and the third rows of $\mathbf{G}$. This also gives $\underline{\mathbf{x}} = \left(1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1\right)$.

c. Find the parity-check matrix **H**.

We look at two parts of **G**: the message part and the parity part.

$$\mathbf{G} = \begin{array}{c} \left(\begin{array}{cccccccc} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{array}\right) \\ \begin{array}{cccccccc} b_1 & b_2 & & & b_3 & & b_4 & \end{array} \end{array}$$

$(\cdot)^T$

put in message part

The parity part (columns) from **G** is transposed and put into the message positions (columns). The remaining columns are filled in by an identity matrix.

$$\mathbf{H} = \begin{array}{c} \begin{array}{cccccccc} b_1 & b_2 & p_1 & p_2 & b_3 & p_3 & b_4 & p_4 \end{array} \\ \left(\begin{array}{cccccccc} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right) \end{array}$$

ECS 452: In-Class Exercise # 17

Instructions

1. Working alone is always permitted. However, working in groups is also allowed if social distancing can be used (via, e.g., online group chat/call). For group work, **the group cannot be the same as any of your former group after the midterm.**
2. Only one submission is needed for each group.
3. **Do not panic.**

Date: 3 / 4 / 2020			
Name	ID (last 3 digits)		

Consider a block code whose generator matrix is

P

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$(\quad)^T$

The crossing here tries to capture the fact that there is a swapping of the positions

a. Find the parity check matrix **H** of this code.

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$$(a_1\ a_2\ a_3)\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = a_1b_1 + a_2b_2 + a_3b_3$$

$y.H^T$

$$[0\,1\,1\,1\,1\,1]\begin{bmatrix} 1\,1\,1 \\ 1\,0\,0 \\ 0\,1\,0 \\ 0\,1\,1 \\ 1\,0\,1 \\ 0\,0\,1 \end{bmatrix}$$

b. Suppose we receive $\underline{\mathbf{y}}$ = 011111.

i. Find the syndrome vector $\underline{\mathbf{s}}$.

Because the 1s in $\underline{\mathbf{y}}$ are in the last five positions, to find the syndrome, we add the last five columns of **H**.

$$\underline{\mathbf{s}} = \underline{\mathbf{y}}\mathbf{H}^T = \left(\begin{pmatrix}1\\0\\0\end{pmatrix}+\begin{pmatrix}0\\1\\0\end{pmatrix}+\begin{pmatrix}0\\1\\1\end{pmatrix}+\begin{pmatrix}1\\0\\1\end{pmatrix}+\begin{pmatrix}0\\0\\1\end{pmatrix}\right)^T = \begin{pmatrix}0\\0\\1\end{pmatrix}^T = \begin{pmatrix}0 & 0 & 1\end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & \boxed{0} \\ 1 & 0 & 1 & 1 & 0 & \boxed{0} \\ 1 & 0 & 0 & 1 & 1 & \boxed{1} \end{pmatrix}$$

ii. Find the decoded codeword $\hat{\underline{\mathbf{x}}}$.

The syndrome $\underline{\mathbf{s}}$ is the same as the *last* column of $\mathbf{H}$.

Therefore, $\hat{\underline{\mathbf{e}}} = \begin{pmatrix}0 & 0 & 0 & 0 & 0 & 1\end{pmatrix}$ and

$$\hat{\underline{\mathbf{x}}} = \underline{\mathbf{y}} - \hat{\underline{\mathbf{e}}} = \underline{\mathbf{y}} \oplus \hat{\underline{\mathbf{e}}} = \begin{pmatrix}0 & 1 & 1 & 1 & 1 & 0\end{pmatrix}.$$

iii. Find the decoded message $\hat{\underline{\mathbf{b}}}$.

From $\mathbf{G}$, we have columns of $\mathbf{I}_3$ in the 1st, 4th, and 5th columns; so, given a codeword $\underline{\mathbf{x}}$, the message $\underline{\mathbf{b}}$ corresponding to this codeword is given by the codeword's 1st, 4th, and 5th bits. Here, the decoded codeword is $\hat{\underline{\mathbf{x}}} = \begin{pmatrix}\underline{0} & 1 & 1 & \underline{1} & \underline{1} & 0\end{pmatrix}$. Therefore, the corresponding decoded message is $\hat{\underline{\mathbf{b}}} = \begin{pmatrix}0 & 1 & 1\end{pmatrix}$.

Checking whether a code is generated by some generator matrix **G**

d				**x**						
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	0	1	0	1
0	0	1	0	0	0	1	0	1	1	0
0	0	1	1	1	0	0	0	0	1	1
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	0	1	1	0	0	1
0	1	1	0	1	0	1	1	0	1	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	1	1	0	0	0	0
1	0	0	1	0	1	0	0	1	0	1
1	0	1	0	1	1	0	0	1	1	0
1	0	1	1	0	1	1	0	0	1	1
1	1	0	0	0	1	1	1	1	0	0
1	1	0	1	1	1	0	1	0	0	1
1	1	1	0	0	1	0	1	0	1	0
1	1	1	1	1	1	1	1	1	1	1

- If a code is generated by a generator matrix, it is automatically a linear code.
- When the codebook is provided, look at each column of the codeword part.
- Write down the equation by reading the structure from appropriate row discussed earlier.
    - For example, here, we read $x_1 = d_1 \oplus d_2 \oplus d_4$.
- Then, we add the corresponding columns of the message part and check whether the sum is the same as the corresponding codeword column.
- So, we need to check n summations.
    - Direct checking that we discussed earlier consider $\binom{M-1}{2}$ summations.

Example:

d				**x**							
0	0	0	0	0	0	0	0	0	0	0	
0	0	0	1	1	0	1	0	1	0	1	$\mathbf{g}^{(4)}$
0	0	1	0	0	0	1	0	1	1	0	$\mathbf{g}^{(3)}$
0	0	1	1	1	0	0	0	0	1	1	
0	1	0	0	1	0	0	1	1	0	0	$\mathbf{g}^{(2)}$
0	1	0	1	0	0	1	1	0	0	1	
0	1	1	0	1	0	1	1	0	1	0	
0	1	1	1	0	0	0	1	1	1	1	
1	0	0	0	1	1	1	0	0	0	0	$\mathbf{g}^{(1)}$
1	0	0	1	0	1	0	0	1	0	1	
1	0	1	0	1	1	0	0	1	1	0	
1	0	1	1	0	1	1	0	0	1	1	
1	1	0	0	1	1	1	1	1	0	0	
1	1	0	1	1	1	0	1	0	0	1	
1	1	1	0	0	1	0	1	0	1	0	
1	1	1	1	1	1	1	1	1	1	1	

- We read $x_1 = d_1 \oplus d_2 \oplus d_4$.
- We add the message columns corresponding to d_1, d_2, d_4,
    - We see that the first bit of the 13th codeword does not conform with the structure above.
- Conclusion: This code is not generated by a generator matrix.

Example:

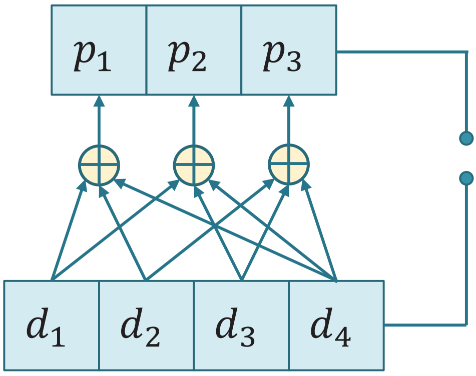
$\underline{\mathbf{d}}$				$\underline{\mathbf{x}}$							
0	0	0	0	0	0	0	0	0	0	0	
0	0	0	1	1	0	1	0	1	0	1	$\underline{\mathbf{g}}^{(4)}$
0	0	1	0	0	0	1	0	1	1	0	$\underline{\mathbf{g}}^{(3)}$
0	0	1	1	1	0	0	0	0	1	1	
0	1	0	0	1	0	0	1	1	0	0	$\underline{\mathbf{g}}^{(2)}$
0	1	0	1	0	0	1	1	0	0	1	
0	1	1	0	1	0	1	1	0	1	0	
0	1	1	1	0	0	0	1	1	1	1	
1	0	0	0	1	1	1	0	0	0	0	$\underline{\mathbf{g}}^{(1)}$
1	0	0	1	0	1	0	0	1	0	1	
1	0	1	0	1	1	0	0	1	1	0	
1	0	1	1	0	1	1	0	0	1	1	
1	1	0	0	1	1	1	1	1	0	0	
1	1	0	1	1	1	0	1	0	0	1	
1	1	1	0	0	1	0	1	0	1	0	
1	1	1	1	1	1	1	1	1	1	1	

- The case found in the previous slide may help with the search to show that the code is not linear.
- The corresponding message is 1100.
  - The codeword corresponding to this message should be $\underline{\mathbf{g}}^{(1)} \oplus \underline{\mathbf{g}}^{(2)}$.
  - If $\underline{\mathbf{g}}^{(1)} \oplus \underline{\mathbf{g}}^{(2)}$, is not a codeword, then we can quickly conclude that the code is not linear:

$\underline{\mathbf{g}}^{(1)}$ and $\underline{\mathbf{g}}^{(2)}$ are codewords but $\underline{\mathbf{g}}^{(1)} \oplus \underline{\mathbf{g}}^{(2)} = 0111100$ is not one of the codewords.

Implementation

- Linear block codes are typically implemented with modulo-2 adders tied to the appropriate stages of a shift register.



$$\underline{\mathbf{x}} = [p_1 \quad d_1 \quad p_2 \quad d_2 \quad p_3 \quad d_3 \quad d_4]$$

where

$$p_1 = d_1 \oplus d_2 \oplus d_4$$
$$p_2 = d_1 \oplus d_3 \oplus d_4$$
$$p_3 = d_2 \oplus d_3 \oplus d_4$$

Digital Communication Systems
EES 452

Asst. Prof. Dr. Prapun Suksompong

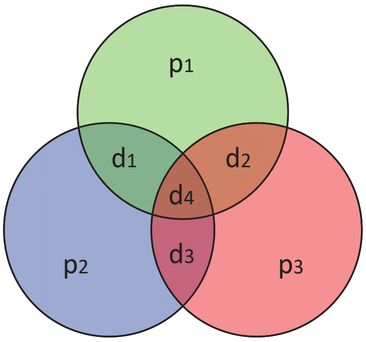
prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Parity-Check Matrix H

Back to

Hamming codes: Ex. 1



$$\begin{array}{cccccccc} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ \underline{\mathbf{x}} = & [p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4] \end{array}$$

Structure in the codewords:

$$\begin{aligned} p_1 &= d_1 \oplus d_2 \oplus d_4 \\ p_2 &= d_1 \oplus d_3 \oplus d_4 \\ p_3 &= d_2 \oplus d_3 \oplus d_4 \end{aligned} \quad \Longleftrightarrow \quad \begin{aligned} p_1 \oplus d_1 \oplus d_2 \oplus d_4 &= 0 \\ p_2 \oplus d_1 \oplus d_3 \oplus d_4 &= 0 \\ p_3 \oplus d_2 \oplus d_3 \oplus d_4 &= 0 \end{aligned}$$

At the receiver, we check whether the received vector $\underline{\mathbf{y}}$ still satisfies these conditions via computing the **syndrome vector**:

$\underline{s} = \underline{y} H^T$

$$\underline{\mathbf{s}} = \begin{array}{ccccccc} [y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7] \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \end{array} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \underline{\mathbf{0}}?$$

If $\underline{e} = \underline{0}$, we should get $\underline{s} = \underline{0}$.

If $\underline{s} \neq \underline{0}$, then $\underline{e} \neq \underline{0}$. ⇒ Error detection.

Parity Check Matrix: Ex 1

- Intuitively, the **parity check matrix** $\mathbf{H}$, as the name suggests, tells which bits in the observed vector $\underline{\mathbf{y}}$ are used to "check" for validity of $\underline{\mathbf{y}}$.
- The number of rows is the same as the number of conditions to check (which is the same as the number of parity check bits).
- For each row, a one indicates that the bits (including the bits in the parity positions) are used in the validity check calculation.

Structure in the codeword:

$$p_1 \oplus d_1 \oplus d_2 \oplus d_4 = 0$$
$$p_2 \oplus d_1 \oplus d_3 \oplus d_4 = 0$$
$$p_3 \oplus d_2 \oplus d_3 \oplus d_4 = 0$$

⟺

$$\mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

Parity Check Matrix: Ex 1

Relationship between $\mathbf{G}$ and $\mathbf{H}$.

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}$$

⟺

$$\mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

Parity Check Matrix: Ex 1

Relationship between **G** and **H**.

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \end{array} \atop \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad \Longleftrightarrow \quad \mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \end{array} \atop \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Parity Check Matrix: Ex 1

Relationship between **G** and **H**.

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \end{array} \atop \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad \Longleftrightarrow \quad \mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \end{array} \atop \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

(columns of) identity matrix in the data positions

(columns of) identity matrix in the parity check positions

Parity Check Matrix: Ex 1

Relationship between **G** and **H**.

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array} \Longleftrightarrow \mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

$(\cdot)^T$

Content of G in the parity part is transposed and put in the message part of H.

Digital Communication Systems
EES 452

Asst. Prof. Dr. Prapun Suksompong
prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

More on H and G, and Systematic Encoding

Summary

Review: Linear Block Codes

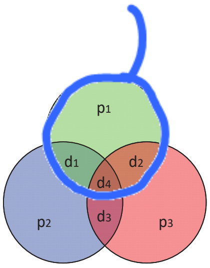
$p_1 \oplus d_1 \oplus d_2 \oplus d_4 = 0$

①

Code structure

$\underline{\mathbf{x}} = [p_1 \quad d_1 \quad p_2 \quad d_2 \quad p_3 \quad d_3 \quad d_4]$

where $p_1 = d_1 \oplus d_2 \oplus d_4$
$p_2 = d_1 \oplus d_3 \oplus d_4$
$p_3 = d_2 \oplus d_3 \oplus d_4$



②

Codebook

$\underline{\mathbf{d}}$				$\underline{\mathbf{x}}$						
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	0	1	0	1
0	0	1	0	0	0	1	0	1	1	0
0	0	1	1	1	0	0	0	0	1	1
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	0	1	1	0	0	1
0	1	1	0	1	0	1	1	0	1	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	1	1	0	0	0	0
1	0	0	1	0	1	0	0	1	0	1
1	0	1	0	1	1	0	0	1	1	0
1	0	1	1	0	1	1	0	0	1	1
1	1	0	0	0	1	1	1	1	0	0
1	1	0	1	1	1	0	1	0	0	1
1	1	1	0	0	1	0	1	0	1	0
1	1	1	1	1	1	1	1	1	1	1

③

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}$$

Generator Matrix

④

$$\mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

Parity Check Matrix

Review: Linear Block Codes

- The code structure is built into each codeword at the encoder (transmitter) via the generator matrix
  - Each codeword is created by $\underline{\mathbf{x}} = \underline{\mathbf{d}}\mathbf{G}$.
- The code structure is checked at the decoder (receiver) via the parity check matrix.
  - A valid codeword must satisfy $\underline{\mathbf{x}}\mathbf{H}^T = \underline{\mathbf{0}}$.

This is why we calculate the syndrome vector $\underline{s} = \underline{y} H^T$ at Rx

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array} \quad \longleftrightarrow \quad \mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

Remark: Location of the Message Bits

$$\mathbf{G} = \begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array} \quad \longleftrightarrow \quad \mathbf{H} = \begin{array}{ccccccc} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ p_1 & d_1 & p_2 & d_2 & p_3 & d_3 & d_4 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}$$

- The "identity-matrix" columns in **G** corresponds to positions of the **message (data) bits** in each codeword.
  - Ex. For this code, codeword $\underline{\mathbf{x}} = [1\ 1\ 0\ 0\ 1\ 1\ 0]$ corresponds to message $\underline{\mathbf{b}} = [1\ 0\ 1\ 0]$.

    d_1 d_2 d_3 d_4
- The "identity-matrix" columns in **H** corresponds to positions of the parity (check) bits in each codeword.

Parity Check Matrix

Key property:

$$\boxed{\mathbf{G}\mathbf{H}^T = \mathbf{0}_{k\times(n-k)}}$$

rows of H are "orthogonal" to rows of G.

Proof:

$HG^T = 0$

- When there is no error $(\underline{\mathbf{e}} = \underline{\mathbf{0}})$, the syndrome vector calculation should give $\underline{\mathbf{s}} = \underline{\mathbf{0}}$.
- By definition,

$$\underline{\mathbf{s}} = \underline{\mathbf{y}}\mathbf{H}^T = (\underline{\mathbf{x}} \oplus \underline{\mathbf{e}})\mathbf{H}^T = \underline{\mathbf{x}}\mathbf{H}^T \oplus \underline{\mathbf{e}}\mathbf{H}^T = \underline{\mathbf{b}}\mathbf{G}\mathbf{H}^T \oplus \underline{\mathbf{e}}\mathbf{H}^T.$$

- Therefore, when $\underline{\mathbf{e}} = \underline{\mathbf{0}}$, we have $\underline{\mathbf{s}} = \underline{\mathbf{b}}\mathbf{G}\mathbf{H}^T$.
- To have $\underline{\mathbf{s}} = \underline{\mathbf{0}}$ for any $\underline{\mathbf{b}}$, we must have $\mathbf{G}\mathbf{H}^T = \mathbf{0}$.

A matrix of zeroes

Ex. single-Parity-check code

$\underline{x} = [b_1, b_2, b_3, b_1 \oplus b_2 \oplus b_3]$

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

I

P

P^T

I

$$H = [\,1\ 1\ 1\ \ 1\,]$$

Systematic Encoding

- Code constructed with distinct information bits and check bits in each codeword are called **systematic codes**.
  - **Message bits are directly "visible" in the codeword.**
- Popular forms of **G**:

$$\mathbf{G} = \left[\mathbf{P}_{k\times(n-k)} \mid \mathbf{I}_k\right] \qquad \underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G} = [b_1\ \ b_2\ \ \cdots\ \ b_k]\left[\mathbf{P}_{k\times(n-k)} \mid \mathbf{I}_k\right]$$

$$= \left[x_1\ \ x_2\ \ \cdots\ \ x_{n-k} \mid \underset{x_{n-k+1}}{b_1}\ \ \underset{x_{n-k+2}}{b_2}\ \ \cdots\ \ \underset{x_n}{b_k}\right]$$

$$\mathbf{G} = \left[\mathbf{I}_k \mid \mathbf{P}_{k\times(n-k)}\right] \qquad \underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G} = [b_1\ \ b_2\ \ \cdots\ \ b_k]\left[\mathbf{I}_k \mid \mathbf{P}_{k\times(n-k)}\right]$$

$$= \left[\underset{x_1}{b_1}\ \ \underset{x_2}{b_2}\ \ \cdots\ \ \underset{x_k}{b_k} \mid x_{k+1}\ \ x_{k+2}\ \ \cdots\ \ x_n\right]$$

Parity check matrix

- For the generators matrices we discussed in the previous slide, the corresponding **parity check matrix** can be found easily:

$$\mathbf{G} = \left[\mathbf{P}_{k\times(n-k)} \mid \mathbf{I}_k\right] \Longrightarrow \mathbf{H} = \left[\mathbf{I}_{n-k} \mid -\mathbf{P}^T\right]$$

Check: $\mathbf{G}\mathbf{H}^T = [\mathbf{P} \mid \mathbf{I}]\begin{bmatrix}\mathbf{I} \\ \textbf{-P}\end{bmatrix} = \mathbf{P} \oplus (-\mathbf{P}) = \mathbf{0}_{k\times(n-k)}$

$$\mathbf{G} = \left[\mathbf{I}_k \mid \mathbf{P}_{k\times(n-k)}\right] \Longrightarrow \mathbf{H} = \left[-\mathbf{P}^T \mid \mathbf{I}_{n-k}\right]$$

Hamming codes: Ex. 2

- Systematic (7,4) Hamming Codes

$$\mathbf{G} = \left[\begin{array}{ccc:cccc} \multicolumn{3}{c}{P} & \multicolumn{4}{c}{I} \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$\mathbf{H} = \left[\begin{array}{ccc:cccc} \multicolumn{3}{c}{} & \multicolumn{4}{c}{P^T} \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array}\right]$$

Digital Communication Systems EES 452

Asst. Prof. Dr. Prapun Suksompong
prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Formal Construction of Hamming Code

Hamming codes

Now, we will give a general recipe for constructing Hamming codes.

Parameters:

Example

$m=$	2	3	4
$n=$	3	7	15
$k=$	1	4	11

g, r

- $m = n - k =$ number of parity bits
- $n = 2^m - 1 \in \{3,7,15,31,63,127,\dots\}$
- $k = n - m = 2^m - m - 1$

It can be shown that, for Hamming codes,

- $d_{\min} = 3.$ → double-error detection
- Error correcting capability: $t = 1$ → single error correction

Construction of Hamming Codes

Ex. $m=2 \Rightarrow \cancel{\begin{pmatrix}0\\0\end{pmatrix}}$ $\begin{pmatrix}0\\1\end{pmatrix}$

$n = 2^2-1 = 3$ $\begin{pmatrix}1\\0\end{pmatrix}$ $\begin{pmatrix}1\\1\end{pmatrix}$

- Start with m.

1. Parity check matrix $\mathbf{H}$:
   - Construct a matrix whose columns consist of *all* nonzero binary m-tuples.

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

   - The ordering of the columns is arbitrary. However, next step is easy when the columns are arranged so that $\mathbf{H} = \left[\mathbf{I}_m \;\vdots\; \mathbf{P}\right]$.

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

2. Generator matrix $\mathbf{G}$:
   - When $\mathbf{H} = \left[\mathbf{I}_m \;\vdots\; \mathbf{P}\right]$, we have $\mathbf{G} = \left[-\mathbf{P}^T \;\vdots\; \mathbf{I}_k\right] = \left[\mathbf{P}^T \;\vdots\; \mathbf{I}_k\right]$.

Hamming codes: Ex. 2

- Systematic (7,4) Hamming Codes

$$\mathbf{H} = \left[\begin{array}{ccc|cccc} & & & & & -\mathbf{P}^T & \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array}\right]$$

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

- Columns are all possible <u>nonzero</u> 3-bit vectors
- We arrange the columns so that $\mathbf{I}_3$ is on the left to make the code systematic. (One can also put $\mathbf{I}_3$ on the right.)

- Note that the size of the identity matrices in **G** and **H** are not the same.

Review: Hamming Code Recipe

Here,

$m = 3$

$n = 2^3 - 1$

$= 7$

$k = 4$

$$\mathbf{H} = \left[\begin{array}{ccc|cccc} & \mathbf{I}_m & & & & \mathbf{P}^T & \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array}\right]$$

$$\mathbf{G} = \left[\begin{array}{ccc|cccc} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ & \mathbf{P} & & & & \mathbf{I}_k & \end{array}\right]$$

- Start with the parity-check matrix
- m rows
  - $m = n - k$
- Columns are all possible <u>nonzero</u> m-bit vectors
  - $n = 2^m - 1$ columns
  - Arranged to have $\mathbf{I}_m$ on the left (or on the right).
    - This simplifies conversion to **G.**
- Get **G** from **H**.

$$\boxed{\mathbf{G} = \left[\mathbf{P}_{k\times(n-k)} \;\middle|\; \mathbf{I}_k\right] \Leftrightarrow \mathbf{H} = \left[\mathbf{I}_{n-k} \;\middle|\; -\mathbf{P}^T\right]}$$

- Note that the size of the identity matrices in **G** and **H** can be different.

Digital Communication Systems
EES 452

Asst. Prof. Dr. Prapun Suksompong
prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Properties of Syndrome Vector and Decoding of Linear Codes

Minimum Distance Decoding

- At the decoder, suppose we want to use minimum distance decoding, then
  - The decoder needs to have the list of all the possible codewords so that it can compare their distances to the received vector $\underline{\mathbf{y}}$.
  - There are 2^k codewords each having n bits.
    Therefore, saving these takes $2^k \times n$ bits.
  - Also, we will need to perform the comparison 2^k times.
- Alternatively, we can utilize the syndrome vector.
  - The syndrome vector is computed from the parity-check matrix $\mathbf{H}$.
  - Therefore, saving $\mathbf{H}$ takes $(n-k) \times n$ bits.

Minimum Distance Decoding

- Recall that

$$d(\underline{x},\underline{y}) = w(\underline{x}\oplus\underline{y}) = w(\underline{e})$$

- Therefore, minimizing the distance is the same as minimizing the weight of the error pattern.
- New goal:
  - find the decoded error pattern $\hat{\underline{e}}$ with the minimum weight
  - then, the decoded codeword is $\hat{\underline{x}} = \underline{y} \oplus \hat{\underline{e}}$
- Once we know $\hat{\underline{x}}$, we can directly extract the message part from the decoded codeword if we are using systematic code.
- For example, consider

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

Suppose $\hat{\underline{x}} = 101\underline{1010}$, then we know that the decoded message is $\hat{\underline{b}} = 1010$.

$$[e_1\ e_2 \cdots e_n]\begin{bmatrix} \vec{v}^{(1)} \\ \vdots \\ \vec{v}^{(n)} \end{bmatrix}$$

Properties of Syndrome Vector

- Recall that, from $\mathbf{GH}^T = \mathbf{0}$, we have

$$\underline{\mathbf{s}} = \underline{\mathbf{y}}\mathbf{H}^T \underset{\underline{y}=\underline{x}\oplus\underline{e}}{=} (\underline{\mathbf{x}}\oplus\underline{\mathbf{e}})\mathbf{H}^T \overset{\underline{b}G}{=} (\underline{\mathbf{b}}\mathbf{G}\oplus\underline{\mathbf{e}})\mathbf{H}^T \overset{GH^T=0}{=} \underline{\mathbf{e}}\mathbf{H}^T$$

so, $\underline{y}$ and $\underline{e}$ have the same effect on $\underline{s}$

- Thinking of **H** as a matrix with many columns inside,

$$\mathbf{H} = \begin{bmatrix} \underline{\mathbf{h}}^{(1)} \\ \boxed{\underline{\mathbf{h}}^{(2)}} \\ \vdots \\ \underline{\mathbf{h}}^{(n-k)} \end{bmatrix}_{(n-k)\times n} = \begin{bmatrix} \left(\underline{\mathbf{v}}^{(1)}\right)^T & \boxed{\left(\underline{\mathbf{v}}^{(2)}\right)^T} & \cdots & \left(\underline{\mathbf{v}}^{(n)}\right)^T \end{bmatrix}$$

$$\underline{\mathbf{s}} = \underline{\mathbf{e}}\mathbf{H}^T = \sum_{j=1}^{n} e_j \underline{\mathbf{v}}^{(j)}$$

A "1" in $\underline{e}$ corresponds to a column of H that is in the sum for $\underline{s}$

- Therefore, $\underline{\mathbf{s}}$ is a (linear combination of the columns of **H**)$^{\mathrm{T}}$.

Similarly, when $\underline{y}$ is known, a "1" in $\underline{y}$ corresponds to a column in H that is in the sum for $\underline{s}$

Hamming Codes: Ex. 2

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

$$\underline{\mathbf{s}} = \underline{\mathbf{e}}\mathbf{H}^T = \underbrace{\sum_{j=1}^{n} e_j \underline{\mathbf{v}}^{(j)}}_{\text{Linear combination of the columns of } \mathbf{H}}$$

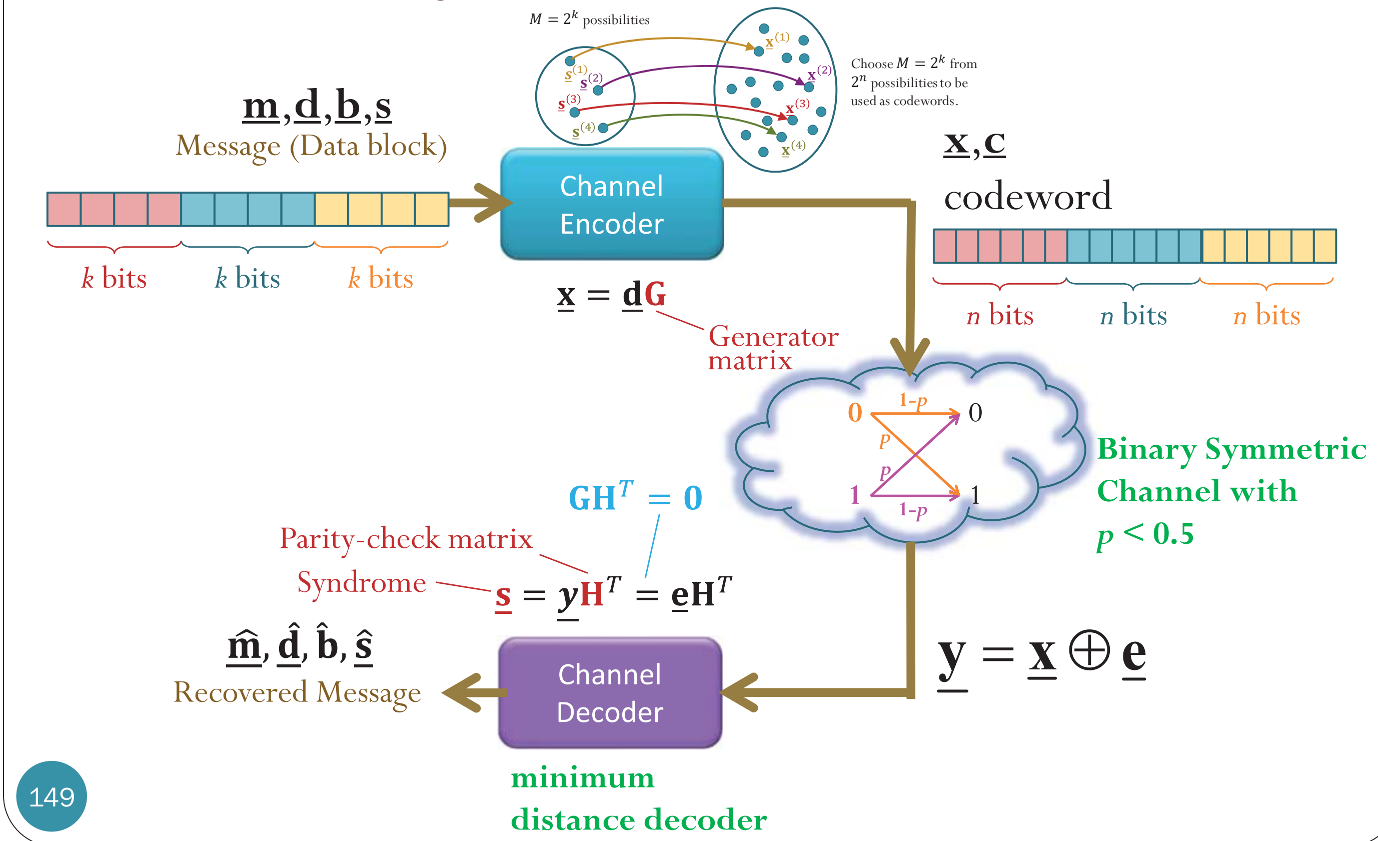
Note that for an error pattern with a single one in the j^{th} coordinate position, the syndrome $\underline{\mathbf{s}} = \underline{\mathbf{y}}\mathbf{H}^T$ is the same as the j^{th} column of $\mathbf{H}$.

Error pattern $\underline{\mathbf{e}}$	Syndrome = $\underline{\mathbf{e}}\mathbf{H}^T$
(0,0,0,0,0,0,0)	(0,0,0)
(0,0,0,0,0,0,1)	(1,1,1)
(0,0,0,0,0,1,0)	(1,1,0)
(0,0,0,0,1,0,0)	(1,0,1)
(0,0,0,1,0,0,0)	(0,1,1)
(0,0,1,0,0,0,0)	(0,0,1)
(0,1,0,0,0,0,0)	(0,1,0)
(1,0,0,0,0,0,0)	(1,0,0)

Decoding Algorithm

- Assumption: the columns of **H** are nonzero and distinct.
- Compute the **syndrome** $\underline{\mathbf{s}} = \underline{\mathbf{y}}\mathbf{H}^T$ for the received vector.
- Case 1: If $\underline{\mathbf{s}} = \underline{\mathbf{0}}$, set $\underline{\hat{\mathbf{x}}} = \underline{\mathbf{y}}$.
- Case 2: If $\underline{\mathbf{s}} \neq \underline{\mathbf{0}}$,
  - determine the position j of the column of **H** that is the same as (the transposition) of the syndrome,
  - set $\underline{\hat{\mathbf{x}}} = \underline{\mathbf{y}}$ but with the j^{th} bit complemented.
- For Hamming codes, because the columns are constructed from all possible non-zero m-tuples, the syndrome vectors must fall into one of the two cases considered above.
- For general linear block codes, the two cases above may not cover every cases.

Summary: Linear Block Code



$\underline{\mathbf{m}},\underline{\mathbf{d}},\underline{\mathbf{b}},\underline{\mathbf{s}}$

Message (Data block)

k bits k bits k bits

Channel Encoder

$M = 2^k$ possibilities

Choose $M = 2^k$ from 2^n possibilities to be used as codewords.

$\underline{\mathbf{x}},\underline{\mathbf{c}}$

codeword

n bits n bits n bits

$\underline{\mathbf{x}} = \underline{\mathbf{d}}\mathbf{G}$

Generator matrix

Binary Symmetric Channel with $p < 0.5$

$\mathbf{G}\mathbf{H}^T = \mathbf{0}$

Parity-check matrix

Syndrome

$\underline{\mathbf{s}} = \underline{y}\mathbf{H}^T = \underline{\mathbf{e}}\mathbf{H}^T$

$\underline{\mathbf{y}} = \underline{\mathbf{x}} \oplus \underline{\mathbf{e}}$

$\hat{\underline{\mathbf{m}}}, \hat{\underline{\mathbf{d}}}, \hat{\mathbf{b}}, \hat{\underline{\mathbf{s}}}$

Recovered Message

Channel Decoder

minimum distance decoder

(decoding algorithm)

$\hat{\underline{x}} \rightarrow \hat{\underline{d}}$

Hamming Codes: Ex. 1

- Consider the Hamming code with

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

- Suppose we observe $\underline{\mathbf{y}} = [0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1]$ at the receiver. Find the decoded codeword $\hat{\underline{x}}$ and the decoded $\hat{\underline{b}}$ message.

$\underline{s} = \underline{y}H^T = (1\,1\,0) \Rightarrow$ same as the $\underline{2^{nd}}$ col. of H.

So, we need to correct the 2^{nd} bit of $\underline{y}$

$\Rightarrow \hat{\underline{x}} = 0\,0\,0\,\underline{1}\,1\,\underline{1}\,\underline{1} \Rightarrow \hat{\underline{b}} = 0111$

[To be explored in the HW]

Hamming Codes: The original method

- Encoding
  - The bit positions that are powers of 2 (1, 2, 4, 8, 16, etc.) are check bits.
  - The rest (3, 5, 6, 7, 9, etc.) are filled up with the *k* data bits.
  - Each check bit forces the parity of some collection of bits, including itself, to be even (or odd).
    - To see which check bits the data bit in position *i* contributes to, rewrite *i* as a sum of powers of 2. A bit is checked by just those check bits occurring in its expansion
- Decoding
  - When a codeword arrives, the receiver initializes a counter to zero. It then examines each check bit at position *i* ($i = 1, 2, 4, 8, \ldots$) to see if it has the correct parity.
  - If not, the receiver adds *i* to the counter. If the counter is zero after all the check bits have been examined (i.e., if they were all correct), the codeword is accepted as valid. If the counter is nonzero, it contains the position of the incorrect bit.

Digital Communication Systems EES 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Proof of the Decoding Algorithm

Proof of the Decoding Algorithm

- We will assume that the columns of $\mathbf{H}$ are nonzero and distinct.
  - This is automatically satisfied for Hamming codes constructed from our recipe.
- Case 1: When $\underline{\mathbf{e}} = \underline{\mathbf{0}}$, we have $\underline{\mathbf{s}} = \underline{\mathbf{0}}$.
  - When $\underline{\mathbf{s}} = \underline{\mathbf{0}}$, we can conclude that $\hat{\underline{\mathbf{e}}} = \underline{\mathbf{0}}$.
    - There can also be $\underline{\mathbf{e}} \neq \underline{\mathbf{0}}$ that gives $\underline{\mathbf{s}} = \underline{\mathbf{0}}$.
      - For example, any nonzero $\tilde{\underline{\mathbf{e}}} \in \mathcal{C}$, will also give $\underline{\mathbf{s}} = \underline{\mathbf{0}}$.
      - However, they have larger weight than $\underline{\mathbf{e}} = \underline{\mathbf{0}}$.
    - The decoded codeword is the same as the received vector.
- Case 2: When, $e_i = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$ (a pattern with a single one in the j^{th} position)

  we have $\underline{\mathbf{s}} = \underline{\mathbf{v}}^{(j)} = (\text{the } j^{\text{th}} \text{ column of } \mathbf{H})^{\mathrm{T}}$
  - When $\underline{\mathbf{s}} = (\text{ the } j^{\text{th}} \text{ column of } \mathbf{H}\ )^{\mathrm{T}}$, we can conclude that $\hat{e}_i = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$
    - There can also be other $\underline{\mathbf{e}}$ that give $\underline{\mathbf{s}} = \underline{\mathbf{v}}^{(j)}$. However, their weights
      - can not be 0 (because, if so, we would have $\underline{\mathbf{s}} = \underline{\mathbf{0}}$ but the columns of $\mathbf{H}$ are nonzero)
      - nor 1 (because the columns of $\mathbf{H}$ are distinct).
    - We flip the j^{th} bit of the received vector to get the decoded codeword.

Digital Communication Systems EES 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Interleaving

Interleaving

- Conventional error-control methods such as parity checking are designed for errors that are isolated or statistically independent events.
- Some errors occur in bursts that span several successive bits.
  - Errors tend to group together in bursts. Thus, errors are no longer independent
  - Examples
    - impulse noise produced by lightning and switching transients
    - fading in wireless systems
    - channel with memory
- Such multiple errors wreak havoc on the performance of conventional codes and must be combated by special techniques.
- One solution is to spread out the transmitted codewords.
- We consider a type of interleaving called **block interleaving**.

Interleave as a verb

- To interleave = to combine different things so that parts of one thing are put between parts of another thing
- Ex. To interleave two books together:

Interleaving: Example

Consider a sequence of ℓ blocks of coded data:

$$\left(x_1^{(1)} x_2^{(1)} \cdots x_n^{(1)}\right)\left(x_1^{(2)} x_2^{(2)} \cdots x_n^{(2)}\right) \cdots \left(x_1^{(\ell)} x_2^{(\ell)} \cdots x_n^{(\ell)}\right)$$

$$\downarrow$$

$$\begin{array}{cccc} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(\ell)} & x_2^{(\ell)} & \cdots & x_n^{(\ell)} \end{array}$$

- Arrange these blocks as rows of a table.
- Normally, we get the bit sequence simply by reading the table by rows.
- With interleaving (by an interleaver), transmission is accomplished by reading out of this table by columns.
- Here, ℓ blocks each of length n are interleaved to form a sequence of length ℓn.

$$\downarrow$$

$$\left(x_1^{(1)} x_1^{(2)} \cdots x_1^{(\ell)}\right)\left(x_2^{(1)} x_2^{(2)} \cdots x_2^{(\ell)}\right) \cdots \left(x_n^{(1)} x_n^{(2)} \cdots x_n^{(\ell)}\right)$$

The received symbols must be deinterleaved (by a deinterleaver) prior to decoding.

Interleaving: Advantage

- Consider the case of a system that can only correct single errors.
- If an error burst happens to the original bit sequence, the system would be overwhelmed and unable to correct the problem.

original bit sequence $\left(x_1^{(1)} x_2^{(1)} \cdots x_n^{(1)}\right)\left(x_1^{(2)} x_2^{(2)} \cdots x_n^{(2)}\right) \cdots \left(x_1^{(\ell)} x_2^{(\ell)} \cdots x_n^{(\ell)}\right)$

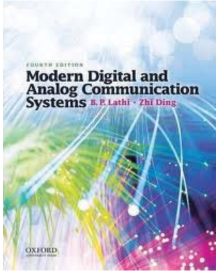
interleaved transmission $\left(x_1^{(1)} x_1^{(2)} \cdots x_1^{(\ell)}\right)\left(x_2^{(1)} x_2^{(2)} \cdots x_2^{(\ell)}\right) \cdots \left(x_n^{(1)} x_n^{(2)} \cdots x_n^{(\ell)}\right)$

- However, in the interleaved transmission,
  - successive bits which come from ***different*** original blocks have been corrupted
  - when received, the bit sequence is reordered to its original form and then the FEC can correct the faulty bits
  - Therefore, single error-correction system is able to fix several errors.

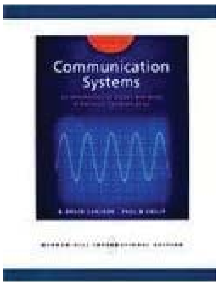
Interleaving: Advantage

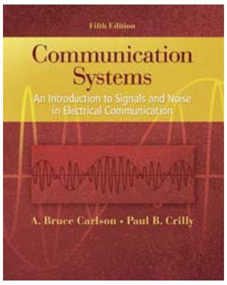
- If a burst of errors affects at most ℓ consecutive bits, then each original block will have at most one error.
- If a burst of errors affects at most $r\ell$ consecutive bits (assume $r < n$),
  then each original block will have at most r errors.
- Assume that there are no other errors in the transmitted stream of ℓn bits.
  - A single error-correcting code can be used to correct a single burst spanning upto ℓ symbols.
  - A double error-correcting code can be used to correct a single burst spanning upto 2ℓ symbols.

References: Linear Codes

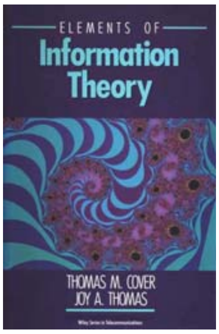


- Lathi and Ding, ***Modern Digital and Analog Communication Systems***, 2009
  - [TK5101 L333 2009]
  - Chapter 15 p. 907-918




Fifth Edition
Communication Systems
An Introduction to Signals and Noise in Electrical Communication
A. Bruce Carlson · Paul B. Crilly

- Carlson and Crilly, ***Communication Systems: An Introduction to Signals and Noise in Electrical Communication***, 2010
  - [TK5102.5 C3 2010]
  - Chapter 13 p. 591-597, 604-611



- Cover and Thomas, ***Elements of Information Theory***, 2006
  - 1991 Version: [Q360 C68 1991]
  - Section 7.11 p. 210-215

- Sklar and F. J. Harris, ***The ABCs of linear block codes***, IEEE Signal Process. Mag., 21(4), (2004).
  - p. 14–24