Lecture 3

Linear Block Codes

This chapter gives an introduction to linear block codes. The coverage of this chapter includes: (1) fundamental concepts and structures of linear block codes; (2)generation of these codes in terms of their generator

and parity check matrices; and (3) their error correction capabilities; and (4) general decoding of these codes. We will begin our introduction with linear block codes with symbols from the binary field GF(2). Linear block codes over non-binary fields will be given at the end of this chapter.

There are many excellent texts on the subject of error control coding theory [1-10], which have extensive coverage of linear block codes. For in depth study of linear block codes, readers are referred to these texts.

3.1 Introduction to Linear Block Codes

- In block coding, an information sequence is segmented into message blocks of fixed length; each message block consists of k information bits. There are 2^k distinct messages. At the channel encoder, each input message

$$\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$$

of k information bits is encoded into a longer binary sequence

$$\mathbf{v} = (v_0, v_1, \ldots, v_{k-1})$$

of n binary digits with $n > k$, according to certain encoding rules.

- This longer binary sequence $\mathbf{v}$ is called the **code word** of the message $\mathbf{u}$. The binary digits in a code word are called code bits. Since there are 2^k distinct messages, there are 2^k code words, one for each distinct message. This set of 2^k code words is said to form an (n, k) block code.
- For a block code to be useful, the 2^k code words for the 2^k distinct messages must be distinct. The $n - k$ bits added to each input message by the channel encoder are called **redundant** bits. These redundant bits carry no new information and their main function is to provide the code with capability of **detecting** and **correcting** transmission errors caused by the channel noise or interferences.
- How to form these redundant bits such that an (n, k) block code

has good error-correcting capability is a major concern in designing the channel encoder. The ratio $R = k/n$ is called the **code rate**, which is interpreted as the number of information bits carried by each code bit.

- For a block code of length n with 2^k code words, unless it has certain special structural properties, the encoding and decoding apparatus would be prohibitively complex for large k since the encoder has to store 2^k code words of length n in a dictionary and the decoder has to perform a table (with 2^n entries) look-up to determine the transmitted code word. Therefore, we must restrict our attention to block codes that can be implemented in a practical manner. A desirable structure for a block code is the **linearity**.

- **Definition 3.1:** A binary block code of length n with 2^k code words is called an (n, k) **linear block code** if and only if its 2^k code words form a k-dimensional subspace of the vector space $\mathbf{V}_n$ of all the n-tuples over GF(2).

3.2 Generator and Parity-Check Matrices

- Since a binary (n, k) linear block code $\mathcal{C}$ is a k-dimensional subspace of the vector space of all the n-tuples over GF(2), there exist k **linearly independent code words**, $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}$, in $\mathcal{C}$ such that every code word $\mathbf{v}$ in $\mathcal{C}$ is a linear combination of these k linearly independent code words.

- These k linearly independent code words in $\mathcal{C}$ form a basis $\mathcal{B}_c$ of $\mathcal{C}$. Using this basis, encoding can be done as follows. Let $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ be the message to be encoded. The codeword $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ for this message is given by the following linear combination of $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}$, with the k message bits of $\mathbf{u}$ as the coefficients:

$$\mathbf{v} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots + u_{k-1}\mathbf{g}_{k-1} \tag{3.1}$$

- We may arrange the k linearly independent code words, $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}$, in $\mathcal{C}$ as rows of a $k \times n$ matrix over GF(2) as

follows:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{0,0} & g_{0,1} & \cdots & g_{0,n-1} \\ g_{1,0} & g_{1,1} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix}. \quad (3.2)$$

- Then the code word $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ for message $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ given by (3.1) can be expressed as the matrix product of $\mathbf{v}$ and $\mathbf{G}$ as follows:

$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots + u_{k-1}\mathbf{g}_{k-1}. \quad (3.3)$$

Therefore, the code word $\mathbf{v}$ for a message $\mathbf{u}$ is simply a linear combination of the rows of matrix $\mathbf{G}$ with the information bits

in the message $\mathbf{u}$ as the coefficients.

- $\mathbf{G}$ is called a **generator matrix** of the (n, k) linear block code $\mathcal{C}$. Since $\mathcal{C}$ is spanned by the rows of $\mathbf{G}$, it is called the **row space** of $\mathbf{G}$.
- In general, an (n, k) linear block code has more than one basis. Consequently, a generator matrix of a given (n, k) linear block code is not unique.
- Any choice of a basis of $\mathcal{C}$ gives a generator matrix of $\mathcal{C}$. Obviously, the rank of a generator matrix of a linear block code $\mathcal{C}$ is equal to the dimension of $\mathcal{C}$.
- Since a binary (n, k) linear block code $\mathcal{C}$ is a k-dimensional subspace of the vector space $\mathbf{V}_n$ of all the n-tuples over GF(2), its null (or dual) space, denoted $\mathcal{C}_d$, is an $(n - k)$-dimensional

subspace of $\mathbf{V}_n$ that is given by the following set of n-tuples in $\mathbf{V}$:

$$\mathcal{C}_d = \{\mathbf{w} \in V : \langle \mathbf{w}, \mathbf{v} \rangle = 0 \text{ for all } \mathbf{v} \in \mathcal{C}\}, \qquad (3.4)$$

where $\langle \mathbf{w}, \mathbf{v} \rangle$ denotes the inner product of $\mathbf{w}$ and $\mathbf{v}$ (see Section 2.2).

- $\mathcal{C}_d$ may be regarded as a binary (n,$n-k$) linear block code and is called the **dual code** of $\mathcal{C}$. Let $\mathcal{B}_d$ be a basis of $\mathcal{C}_d$. Then $\mathcal{B}_d$ consists of $n-k$ linearly independent code words in $\mathcal{C}_d$.

- Let $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-k-1}$ be the $n-k$ linearly independent code words in $\mathcal{B}_d$. Then every code word in $\mathcal{C}_d$ is a linear combination of these $n-k$ linearly independent code words in $\mathcal{B}_d$. Form the

following $(n-k) \times n$ matrix over GF(2):

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{n-k-1} \end{bmatrix} = \begin{bmatrix} h_{0,0} & h_{0,1} & \cdots & h_{0,n-1} \\ h_{1,0} & h_{1,1} & \cdots & h_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n-k-1,0} & h_{n-k-1,1} & \cdots & h_{n-k-1,n-1} \end{bmatrix}. \tag{3.5}$$

- Then $\mathbf{H}$ is the generator matrix of the dual code $\mathcal{C}_d$ of the binary (n, k) linear block code $\mathcal{C}$. It follows from (3.3) and (3.4) that

$$\mathbf{G} \cdot \mathbf{H}^T = \mathbf{O} \tag{3.6}$$

  where $\mathbf{O}$ is a $k \times (n-k)$ zero matrix.

- Furthermore, $\mathcal{C}$ is also uniquely specified by the $\mathbf{H}$ matrix as

follows: A binary n-tuple $\mathbf{v} \in \mathbf{V}_n$ is a code word in $\mathcal{C}$ if and only if

$$\mathbf{v} \cdot \mathbf{H}^T = \mathbf{0} \text{ the all-zero } (n-k)\text{-tuple} \tag{3.7}$$

i.e.

$$\mathcal{C} = \{\mathbf{v} \in V : \mathbf{v} \cdot \mathbf{H}^T = \mathbf{0}\}. \tag{3.8}$$

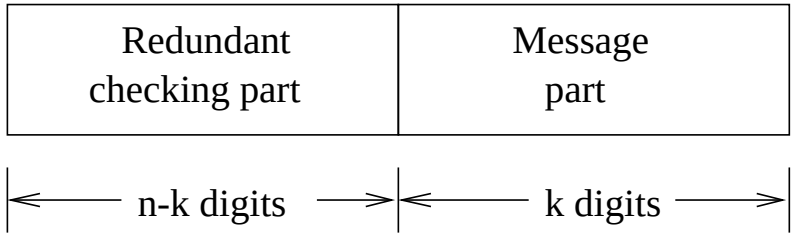
- $\mathbf{H}$ is called a **parity-check matrix** of $\mathcal{C}$ and $\mathcal{C}$ is said to be the **null space** of $\mathbf{H}$.
- Therefore, a linear block code is uniquely specified by two matrices, a generator matrix and a parity-check matrix.
- In general, encoding of a linear block code is based on a generator matrix of the code and decoding is based on a parity-check matrix of the code. Many classes of well known linear block codes are constructed in terms of their parity-check

matrices, such as BCH, RS and LDPC codes.

- A parity-check matrix $\mathbf{H}$ of a linear block code is said to be a **full rank matrix** if its rank is equal to the number of rows of $\mathbf{H}$. However, in many cases, a parity-check matrix of an (n, k) linear block code is not given as a full rank matrix, i.e., the number of its rows is greater than its row rank, $n - k$. In this case, some rows of the given parity-check matrix $\mathbf{H}$ are linear combinations of a set of $n - k$ linearly independent rows. These extra rows are called **redundant rows**.
- LDPC codes are specified by parity-check matrixes that are in general not full rank matrices.

3.3 Linear Systematic Block Codes

- An (n, k) linear block code is said to be **systematic** if it has the following structure: Every code word consists of two parts, the message part and redundant check part as shown in Figures 3.1(a) and 3.1(b), respectively:



(a)

$$(u_0, u_1, \ldots, u_{k-1}) \longleftrightarrow (v_0, v_1, \ldots v_{n-k-1}, u_0, u_1, \ldots, u_{k-1})$$

(b)

Figure 3.1: Systematic format of a code word

- The message part consists of the k unaltered information digits

and redundant check part consists of $n - k$ parity-check digits. A linear block code with this structure is referred to as a **linear systematic code**.

Example 3.1: Let $k = 3$ and $n = 6$. Table 3.1 gives a $(6, 3)$ linear systematic block code.

Table 3.1: A $(6, 3)$ systematic linear block code

Messages (u_0, u_1, u_2)	Code words $(v_0, v_1, v_2, v_3, v_4, v_5, v_6)$
(000)	$(000\mathbf{000})$
(100)	$(011\mathbf{100})$
(010)	$(101\mathbf{010})$
(110)	$(110\mathbf{110})$
(001)	$(110\mathbf{001})$
(101)	$(101\mathbf{101})$
(011)	$(011\mathbf{011})$
(111)	$(000\mathbf{111})$

- A generator matrix for this code is given below,

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

- The code word for the message $\mathbf{u} = (101)$ is

$$\begin{aligned} \mathbf{V} &= \mathbf{u} \cdot \mathbf{G} \\ &= 1 \cdot \mathbf{g}_0 + 0 \cdot \mathbf{g}_1 + 1 \cdot \mathbf{g}_2 \\ &= 1 \cdot (011100) + 0 \cdot (101010) + 1 \cdot (110001) \\ &= (011100) + (000000) + (110001) \\ &= (101101). \end{aligned}$$

- A linear systematic (n, k) block code is completely specified by

a $k \times n$ generator matrix of the following form:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{n-k-1} \end{bmatrix} = \underbrace{\begin{bmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,n-k-1} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,n-k-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} \end{bmatrix}}_{\mathbf{P} \text{ matrix}} \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{k \times k \text{ identity matrix } \mathbf{I}_k} . \tag{3.9}$$

where $p_{i,j} = 0$ or 1.

- The generator matrix $\mathbf{G}$ consists of two submatrices, a $k \times (n-k)$ submatrix $\mathbf{P}$ on the left with entries over GF(2) and a $k \times k$ identity matrix $\mathbf{I}_k$ on the right.

$$\mathbf{G} = \begin{bmatrix} \mathbf{P} & \mathbf{I}_k \end{bmatrix} \tag{3.10}$$

- Let $\mathbf{u} = (u_0, u_1, \cdots, u_{k-1})$ be the message to be encoded.

Taking linear combination of the rows of G given by (3.9) with the information bits in $\mathbf{u}$ as coefficients, we obtain the following corresponding code word:

$$\begin{aligned}\mathbf{v} &= (v_0, v_1, \ldots, v_{n-k-1}, v_{n-k}, \ldots, v_{n-1}) \\ &= (u_0, u_1, \ldots, u_{k-1}) \cdot \mathbf{G} \\ &= u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots + u_{k-1}\mathbf{g}_{k-1}\end{aligned} \tag{3.11}$$

where the n code bits are given by

$$v_{n-k+l} = u_l, \text{ for } l = 0, 1, \ldots, k-1 \tag{3.12}$$

and

$$v_j = u_0p_{0,j} + u_1p_{1,j} + \cdots + u_{k-1}p_{k-1,j}, \text{ for } j = 0, 1, \ldots, n-k-1. \tag{3.13}$$

- Expression (3.12) shows that the rightmost k code bits of code word $\mathbf{v}$ are identical to k information bits $u_0, u_1, \ldots, u_{k-1}$ to be encoded, and (3.13) shows that the left $n-k$ code bits of $\mathbf{v}$ are linear sums of information bits.
- These $n-k$ code bits given by linear sums of information bits are called **parity-check bits** (or simply parity bits) and they are completely specified by the $n-k$ columns of the $\mathbf{P}$ submatrix of $\mathbf{G}$ given by (3.7).
- The $n-k$ equations given by (3.13) are called **parity-check equations** of the code. These parity-check equations completely specified the code.
- The $\mathbf{P}$ submatrix of $\mathbf{G}$ is called the parity submatrix of $\mathbf{G}$. A generator matrix in the form given by (3.9) is said to be in

systematic form.

- Given a $k \times n$ generator matrix $\mathbf{G}$ of an (n, k) linear block code $\mathcal{C}$ not in systematic form, a generator matrix $\mathbf{G}_e$ in systematic form of (3.9) can always be obtained by performing **elementary operations** on the rows of $\mathbf{G}$ and then taking column permutations. The $k \times n$ matrix $\mathbf{G}_e$ is called a **combinatorially equivalent matrix** of $\mathbf{G}$.

- The systematic (n, k) linear block code $\mathcal{C}_e$ generated by $\mathbf{G}_e$ is called a **combinatorially equivalent code** of $\mathcal{C}$. $\mathcal{C}_e$ and $\mathcal{C}$ are only different in the arrangement (or order) of code bits in their code words, i.e., a code word in $\mathcal{C}_e$ is obtained by a fixed permutation of the positions of the code bits in a code word of $\mathcal{C}$, and vise versa. Two combinatorially equivalent (n, k) linear

block codes give the same error performance.

- If a generator matrix of an (n, k) linear block code $\mathcal{C}$ is given in systematic form of (3.9), then its corresponding parity-check matrix in systematic form is given below:

$$\begin{aligned}\mathbf{H} &= \left[\begin{array}{cc} \mathbf{I}_{n-k} & \mathbf{P}^T \end{array}\right] \\ &= \left[\begin{array}{cccccccc} 1 & 0 & \cdots & 0 & p_{0,0} & p_{1,0} & \cdots & h_{k-1,0} \\ 0 & 1 & \cdots & 0 & p_{0,1} & p_{1,1} & \cdots & h_{k-1,1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & p_{0,n-k-1} & p_{1,n-k-1} & \cdots & h_{k-1,n-k-1} \end{array}\right]. \end{aligned} \tag{3.14}$$

It can be easily prove that $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{O}$.

- **Example 3.2:** Consider the $(6, 3)$ linear systematic code given

in Example 3.1 with generator matrix,

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

The parity-check matrix of this code in systematic form is

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

- Let (u_0, u_1, u_2) be the message to be encoded. Then the code

word for this message in systematic form is

$$\mathbf{v} = (v_0, v_1, v_2, v_3, v_4, v_5)$$

where

$$v_5 = u_2, v_4 = u_1, v_3 = u_0,$$
$$v_2 = u_0 + u_1,$$
$$v_1 = u_0 + u_2,$$
$$v_0 = u_1 + u_2.$$

- The parity-check equations actually tell us how to implement the encoder. The encode circuit is shown in Figure 3.2.

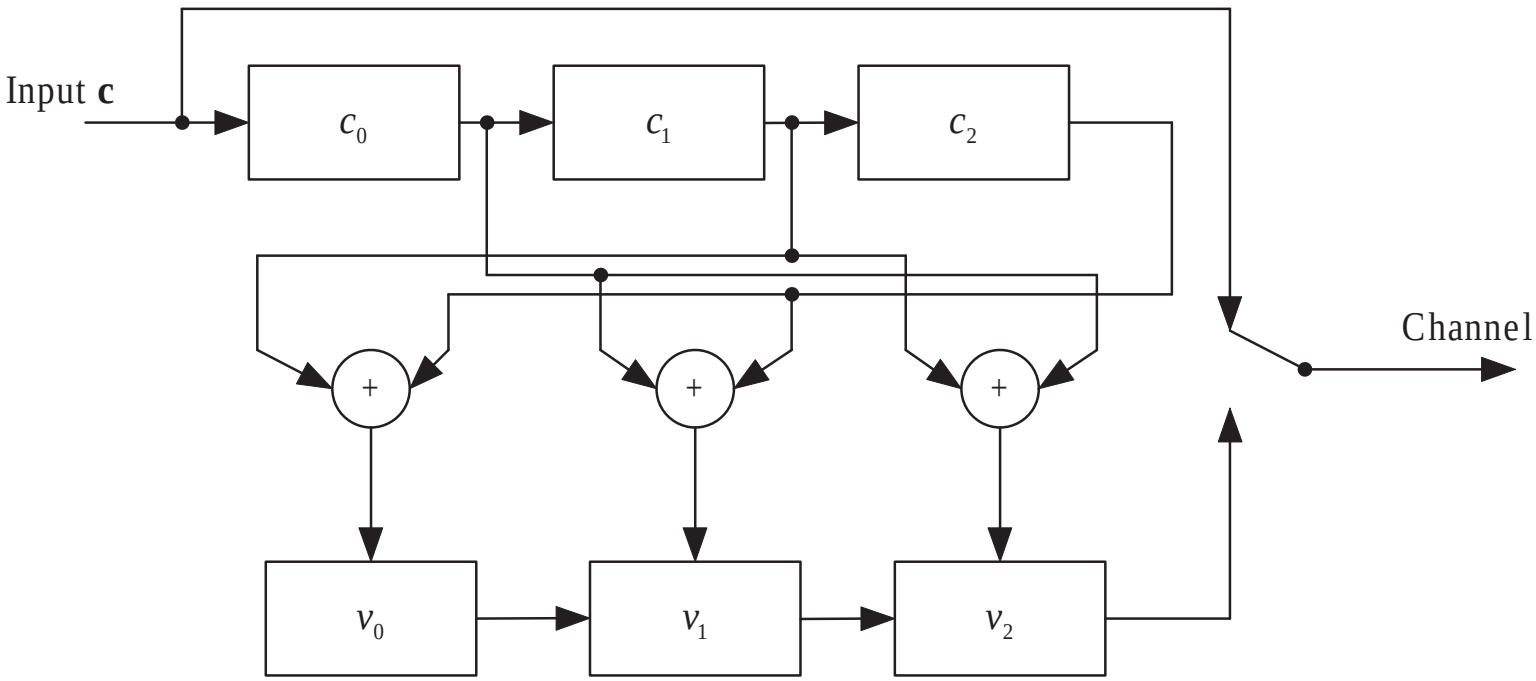


Figure 3.2: An encoder for the $(6, 3)$ systematic code given in Example 3.2.

3.4 Error Detection with Linear Block Codes

- Consider an (n, k) linear block code $\mathcal{C}$ with an $(n-k) \times n$ parity-check matrix $\mathbf{H}$.
- Suppose a code word $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ in $\mathcal{C}$ is transmitted over a BSC (or any binary-input and binary-output channel). Let $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})$ be the corresponding hard-decision received vector (n-tuple or sequence) at the input of the channel decoder. Because of the channel noise, the received vector $\mathbf{r}$ and the transmitted code word $\mathbf{v}$ may differ in some places.

- Define the following vector sum of $\mathbf{r}$ and $\mathbf{v}$:

$$\begin{aligned}\mathbf{e} &= \mathbf{r} + \mathbf{v} \\ &= (e_0, e_1, \ldots, e_{n-1}) \\ &= (r_0 + v_0, r_1 + v_1, \ldots, r_{n-1} + v_{n-1}),\end{aligned} \tag{3.15}$$

  where $e_j = r_j + v_j$ for $0 \leq j < n$ and the addition $+$ is the modulo-2 addition.

- Not that $e_j = 1$, for $r_j \neq v_j$ and $e_j = 0$, for $r_j = v_j$.

- Therefore, the positions of the 1-components in the n-tuple $\mathbf{e}$ are the places where $\mathbf{r}$ and $\mathbf{v}$ differ. At these places, transmission errors have occurred.

- Since $\mathbf{e}$ displays the pattern of transmission errors in $\mathbf{r}$, we called e the **error pattern** (or **vector**) during the transmission of

the code word $\mathbf{v}$. The 1-components in $\mathbf{e}$ are called **transmission errors** that are caused by the channel noise.

- For a BSC, an error can occur at any place with the same probability p over a span of n places (the length of the code). There are 2^n possible error patterns.
- It follows from (3.15), we can express the received vector $\mathbf{r}$ as

$$\mathbf{r} = \mathbf{v} + \mathbf{e}. \tag{3.16}$$

- At the receiver, neither the transmitted code word $\mathbf{v}$ nor the error pattern $\mathbf{e}$ is known. Upon receiving $\mathbf{r}$, the decoder must first determine whether there are transmission errors in $\mathbf{r}$. If the presence of errors is detected, then the decoder must estimate the error pattern $\mathbf{e}$ based on the code $\mathcal{C}$ and the channel

information provided to the decoder.

- Let $\mathbf{e}^*$ denote the estimated error pattern. Then the estimated transmitted code word is given by

$$\mathbf{v}^* = \mathbf{r} + \mathbf{e}^*.$$

- To check whether a received vector $\mathbf{r}$ contains transmission errors, we compute the following $(n-k)$-tuples over GF(2),

$$\begin{aligned} \mathbf{s} &= (s_0, s_1, \ldots, s_{n-k-1}) \\ &= \mathbf{r} \cdot \mathbf{H}^T \end{aligned} \tag{3.17}$$

- Note that $\mathbf{r}$ is an n-tuple in the vector space $\mathbf{V}_n$ of all the n-tuples over GF(2). Recall that an n-tuple $\mathbf{r}$ in $\mathbf{V}_n$ is code word in $\mathcal{C}$ if and only if

$$\mathbf{r} \cdot \mathbf{H}^T = \mathbf{0}.$$

- Therefore, if $\mathbf{s} \neq \mathbf{0}$, $\mathbf{r}$ is not a code word in $\mathcal{C}$. In this case, the transmitter transmitted a code word but receiver receives a vector that is not a code word. Hence, the **presence of transmission errors** is being detected.
- If $\mathbf{s} = \mathbf{0}$, then $\mathbf{r}$ is a code word in $\mathcal{C}$. In this case, the channel decoder assumes that $\mathbf{r}$ is **error-free** and accepts $\mathbf{r}$ as the transmitted code word.
- However, in the event that $\mathbf{r}$ is a code word in $\mathcal{C}$ but differs from the transmitted code word $\mathbf{v}$, accepting $\mathbf{r}$ as the transmitted code word, the decoder commits a **decoding error**. This occurs when the error pattern $\mathbf{e}$ caused by the noise changes the transmitted code word $\mathbf{v}$ into another code word in $\mathcal{C}$.
- This happens when and only when the error pattern $\mathbf{e}$ is

identical to a nonzero code word in $\mathcal{C}$.

- An error pattern of this type is called an **undetectable error pattern**. There are $2^k - 1$ such undetectable error patterns.
- Since the $(n-k)$-tuple $\mathbf{s} = (s_0, s_1, \ldots, s_{n-k-1})$ over GF(2) is used for detecting whether the received vector $\mathbf{r}$ contains transmission errors, it is called the **syndrome** of $\mathbf{r}$.
- **Example 3.3:** Consider the $(7, 4)$ linear code with parity check matrix

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

Let $\mathbf{r} = (0100001)$. The syndrome of $\mathbf{r}$ is

$$\mathbf{s} = (s_0, s_1, s_2) = \mathbf{r} \cdot \mathbf{H}^T$$

$$= (0100001) \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$= (111) \neq \mathbf{0}.$$

Hence, $\mathbf{r}$ is not a code word and the presence of errors has been

detected.

- The syndrome circuit for the $(7, 4)$ code given in Example 3.3 can be implemented easily based on the syndrome equations.
- Let $\mathbf{r} = (r_0, r_1, r_2, r_3, r_4, r_5, r_6)$ be the received vector. The

syndrome of $\mathbf{r}$ is

$$\mathbf{s} = (s_0, s_1, s_2) = \mathbf{r} \cdot \mathbf{H}^T$$

$$= (r_0, r_1, r_2, r_3, r_4, r_5, r_6) \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Then

$$s_0 = r_0 + r_3 + r_5 + r_6$$
$$s_1 = r_1 + r_3 + r_4 + r_5$$
$$s_2 = r_2 + r_4 + r_5 + r_6.$$

- The syndrome circuit for the $(7, 3)$ code given in Example 3.3 is shown in Figure 3.3.

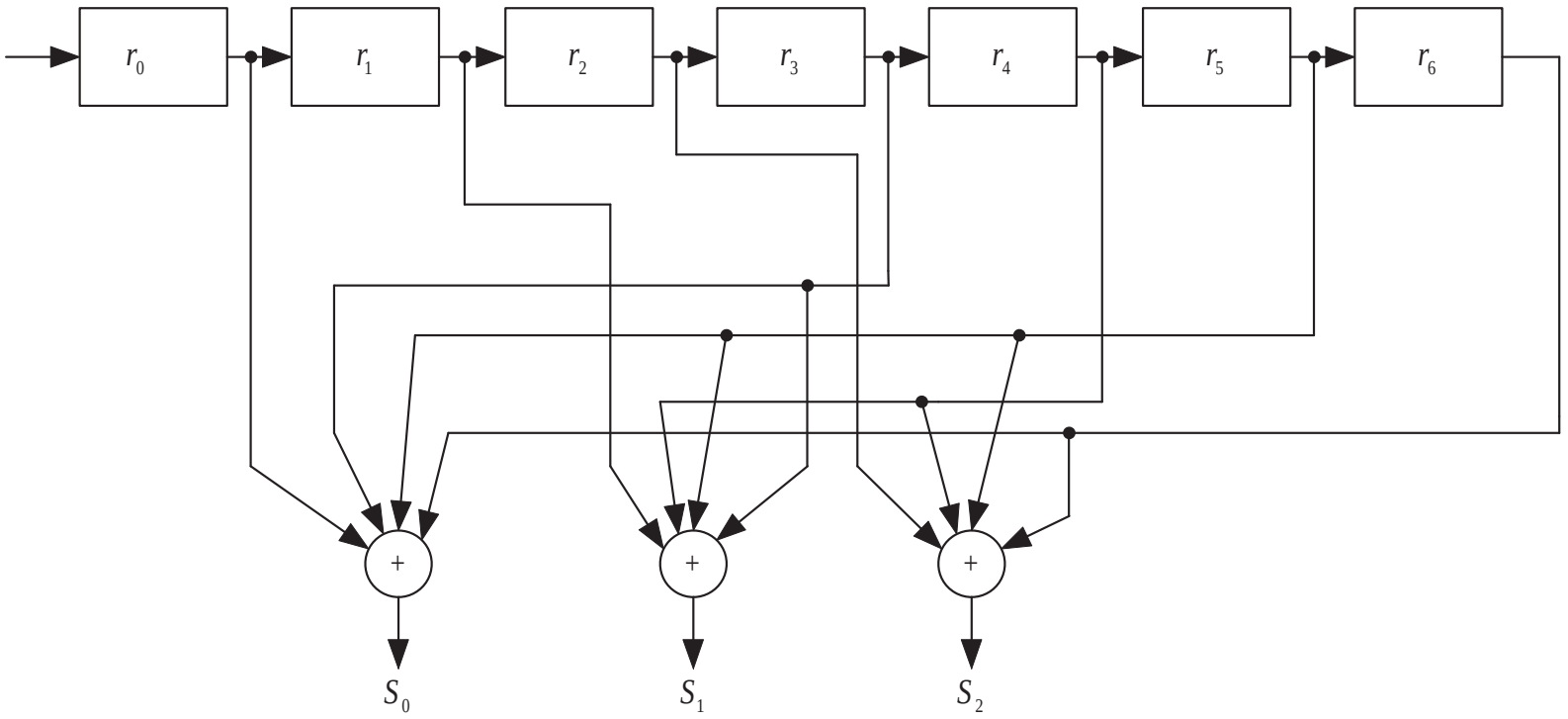


Figure 3.3: A syndrome circuit for the $(7, 4)$ code given in Example 3.3.

3.5 Syndrome and Error Pattern

- Let $\mathbf{r} = \mathbf{v} + \mathbf{e}$ be the received vector where $\mathbf{v}$ and $\mathbf{e}$ are the transmitted code word and error pattern, respectively.
- Then the syndrome of $\mathbf{r}$ is

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (\mathbf{v} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{v} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T. \tag{3.18}$$

- Expanding. (3.18), we obtain the following the following $n-k$ linear equations:

$$\begin{aligned} s_0 &= e_0 + e_{n-k}p_{0,0} + e_{n-k+1}p_{1,0} + \ldots + e_{n-1}p_{k-1,0} \\ s_1 &= e_1 + e_{n-k}p_{0,1} + e_{n-k+1}p_{1,1} + \ldots + e_{n-1}p_{k-1,1} \\ &\vdots \\ s_{n-k-1} &= e_{n-k-1} + e_{n-k}p_{0,n-k-1} + e_{n-k+1}p_{1,n-k-1} \\ &\quad + \ldots + e_{n-1}p_{k-1,n-k-1} \end{aligned} \tag{3.19}$$

- The $n-k$ equations given by (3.19) relate the error digits in the error pattern $\mathbf{e}$ to the computed syndrome digits.
- Any methods solving these $n-k$ equations is a decoding method.
- Since there are more unknowns than equations, the equations

given by (3.19) do not have a unique solution. In fact, there are 2^k possible solutions. Each solution gives an error pattern whose digits satisfy the $n-k$ equations given by (3.19). The true error pattern is just one of them.

- To minimize the probability of a decoding error, the most probable error pattern which satisfies the $n-k$ equations given by (3.19) is chosen as the true error pattern.
- **Example 3.4:** Consider the $(7,4)$ code given in Example 3.3 with following parity-check matrix:

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

- Suppose the code word $\mathbf{v} = (1001011)$ is transmitted and $\mathbf{r} = (1001001)$ is received. The syndrome of $\mathbf{r}$ is

$$\mathbf{s} = (s_0, s_1, s_2) = \mathbf{r} \cdot \mathbf{H}^T = (111).$$

- Let $\mathbf{e} = (e_0, e_1, e_2, e_3, e_4, e_5, e_6)$. Since $\mathbf{s} = \mathbf{e} \cdot \mathbf{H}^T$, we have the following three linear equations related the error digits to the computed syndrome digits:

$$1 = e_0 + e_3 + e_5 + e_6$$
$$1 = e_1 + e_3 + e_4 + e_5$$
$$1 = e_2 + e_4 + e_5 + e_6.$$

- There are $2^4 = 16$ solutions for the above 3 linear equations.

They are:

$$\begin{array}{ll} (0000010) & (1010011) \\ (0101010) & (0111011) \\ (0110110) & (1100111) \\ (1011110) & (0001111) \\ (1110000) & (0100001) \\ (0011000) & (1001001) \\ (1000100) & (0010100) \\ (0101100) & (1111101). \end{array}$$

- For the BSC, the most probable error pattern among the above 16 solutions is $\mathbf{e}^* = (0000010)$. Adding this error pattern to the

received vector $\mathbf{r} = (1001001)$, we have

$$\begin{aligned} \mathbf{r} + \mathbf{e}^* &= (1001001) + (0000010) \\ &= (1001011), \end{aligned}$$

which is the transmitted code word. So the decoding is correct.

- Decoding of a linear block code with parity-check matrix $\mathbf{H}$ can be carried out in three steps:

  (1) Compute the syndrome $\mathbf{s}$ of the received vector $\mathbf{r}$.

  (2) Identify the most probable error pattern $\mathbf{e}^*$ that satisfies the equality $\mathbf{s} = \mathbf{e}^* \cdot \mathbf{H}^T$. Take $\mathbf{e}^*$ as the estimated error pattern.

  (3) Add $\mathbf{e}^*$ to the received vector $\mathbf{r}$. Take $\mathbf{v}^* = \mathbf{r} + \mathbf{e}^*$ as the estimated transmitted code word.

3.6 Weight Distribution of A Linear Block Code

- Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be an n-tuple over GF(2). The **Hamming weight** (or simply weight) of $\mathbf{v}$, denoted $w(\mathbf{v})$, is defined as the number of nonzero components in $\mathbf{v}$.

- Consider an (n, k) linear block code $\mathcal{C}$ with code symbols from GF(2). For $0 \leq i \leq n$, let A_i be the number of code words in $\mathcal{C}$ with Hamming weight i. Then the numbers,

$$A_0, A_1, \ldots, A_n,$$

  are called the **weight distribution** of $\mathcal{C}$. It is clear that $A_0 + A_1 + \ldots + A_n = 2^k$.

- Since there is one and only one all-zero code word in a linear block code, $A_0 = 1$.
- The smallest weight of a nonzero code word in $\mathcal{C}$, denoted $w_{\min}(\mathcal{C})$, is called the **minimum weight** of C. Mathematically, the minimum weight of $\mathcal{C}$ is given as follows:

$$w_{\min}(\mathcal{C}) = \min\{w(\mathbf{v}) \in \mathcal{C}, \mathbf{v} \neq \mathbf{0}\} \tag{3.20}$$

- Suppose $\mathcal{C}$ is used for error control over a BSC with transition probability p. Recall that an undetectable error pattern is an error pattern that is identical to a nonzero code word in $\mathcal{C}$. When such an error pattern occurs, the decoder will not be able to detect the presence of transmission errors and hence will commit a decoding error.

- The total probability that i transmission errors take place during the transmission of n code bits is $A_i p^i (1-p)^{n-i}$. Then the probability that the decoder fails to detect the presence of transmission errors,call the **probability of an undetected error**, is equal to

$$P_u(E) = \sum_{i=1}^{n} A_i p^i (1-p)^{n-i}. \tag{3.21}$$

- Therefore, the weight distribution of a linear block code completely determines its probability of an undetected error. It has been proved that in the ensemble of (n, k) linear block codes over GF(2), there exist codes with probability of an

undetected error, $P_u(E)$, upper bounded by $2^{-(n-k)}$, i.e.,

$$P_u(E) \leq 2^{-(n-k)}. \tag{3.22}$$

A code that satisfies the above upper bound is said to be a good error detection code.

3.7 Minimum Distance of A Linear Block Code

- Let $\mathbf{v}$ and $\mathbf{w}$ be two n-tuples over GF(2). The Hamming distance (or simply distance) between $\mathbf{v}$ and $\mathbf{w}$, denoted $d(\mathbf{v}, \mathbf{w})$, is defined as the number of places where $\mathbf{v}$ and $\mathbf{w}$ differ. The Hamming distance is a metric function that satisfies the triangle inequality. Let $\mathbf{v}$, $\mathbf{w}$ and $\mathbf{x}$ be three n-tuples over GF(2). Then

$$d(\mathbf{v}, \mathbf{w}) + d(\mathbf{w}, \mathbf{x}) \geq d(\mathbf{v}, \mathbf{x}). \tag{3.23}$$

- It follows from the definitions of Hamming distance between two n-tuples and Hamming weight of an n-tuple that the Hamming distance between $\mathbf{v}$ and $\mathbf{w}$ is equal to the Hamming weight of the vector sum of $\mathbf{v}$ and $\mathbf{w}$, i.e.,

$$d(\mathbf{v}, \mathbf{w}) = w(\mathbf{v} + \mathbf{w}). \tag{3.24}$$

- The minimum distance of an (n, k) linear block code $\mathcal{C}$, denote $d_{\min}(\mathcal{C})$ is defined as the **smallest** Hamming distance between two different code words in $\mathcal{C}$, i.e.,

$$d_{\min}(\mathcal{C}) = \min\{d(\mathbf{v}, \mathbf{w}) : \mathbf{v}, \mathbf{w} \in \mathcal{C}, \mathbf{v} \neq \mathbf{w}\}. \tag{3.25}$$

- Using the fact that $d(\mathbf{v}, \mathbf{w}) = w(\mathbf{v} + \mathbf{w})$, we can easily prove that the minimum distance $d_{\min}(\mathcal{C})$ of $\mathcal{C}$ is equal to the minimum

weight $w_{\min}(\mathcal{C})$ of $\mathcal{C}$. This follows from (3.25) and (3.24),

$$\begin{aligned} d_{\min}(\mathcal{C}) &= \min\{d(\mathbf{v}, \mathbf{w}) : \mathbf{v}, \mathbf{w} \in \mathcal{C}, \mathbf{v} \neq \mathbf{w}\} \\ &= \min\{w(\mathbf{v} + \mathbf{w}) : \mathbf{v}, \mathbf{w} \in \mathcal{C}, \mathbf{v} \neq \mathbf{w}\} \\ &= \min\{w(\mathbf{x}) : \mathbf{x} \in \mathcal{C}, \mathbf{x} \neq \mathbf{0}\} \\ &= w_{\min}(\mathcal{C}). \end{aligned} \tag{3.26}$$

- Therefore, for a linear block code, to determine its minimum distance is equivalent to determine its minimum weight. The weight structure (or weight distribution) of a linear block code $\mathcal{C}$ is related to a parity-check matrix $\mathbf{H}$ of the code, as given by the following three theorems. We give these theorems without proofs. For proofs, see [xx].
- **Theorem 3.1:** Let $\mathcal{C}$ be an (n, k) linear block code with a

parity-check matrix $\mathbf{H}$. For each code word in $\mathcal{C}$ with weight i, there exist i columns in $\mathbf{H}$ whose vector sum gives a zero (column) vector. Conversely, if there are i columns in $\mathbf{H}$ whose vector sum results in a zero (column) vector, there is a code word in $\mathcal{C}$ with weight i.

- **Theorem 3.2:** The minimum weight (or minimum distance) of an (n, k) linear block code $\mathcal{C}$ with a parity-check matrix $\mathbf{H}$ is equal to the smallest number of columns in $\mathbf{H}$ whose vector sum is a zero vector.

- **Theorem 3.3:** For an (n, k) linear block code $\mathcal{C}$ given by the null space of a parity-check matrix $\mathbf{H}$, if there are no $d - 1$ or few columns in $\mathbf{H}$ that sum to a zero vector, the minimum distance (or weight) of $\mathcal{C}$ is at least d.

- Theorem 3.3 gives a lower bound on the minimum distance (or weight) of a linear block code. In general, it is very hard to determine the exact minimum distance (or weight) of a linear block code, however, it is much easier to give a lower bound on its minimum distance (or weight).
- The weight distribution of a linear block code $\mathcal{C}$ actually gives the **distance distribution** of the nonzero code words with respect to the all-zero code word $\mathbf{0}$. For $1 \leq i \leq n$, the number A_i of code word in $\mathcal{C}$ with weight i is simply equal to the number of code words that are at distance i from the all-zero code word $\mathbf{0}$.
- Due to linear structure of $\mathcal{C}$, A_i also gives the number of code words in $\mathcal{C}$ that are at a distance i from any fixed code word $\mathbf{v}$ in

$\mathcal{C}$. Therefore, the weight distribution, $\{A_0, A_1, \leq, A_n\}$, of $\mathcal{C}$ is also the **distance distribution** of $\mathcal{C}$ with respect to any code word in $\mathcal{C}$.

- The capabilities of a linear block code $\mathcal{C}$ for detecting and correcting random errors over a BSC with hard-decision decoding are determined by the minimum distance of $\mathcal{C}$ and its error performance with soft-decision MLD is determined by its distance (or weight) distribution.
- For an (n, k) linear block code C with minimum distance $d_{\min}(\mathcal{C})$, no error pattern with $d_{\min}(\mathcal{C}) - 1$ or fewer errors can change a transmitted code word into another code word in $\mathcal{C}$. Therefore, any error pattern with $d_{\min}(\mathcal{C}) - 1$ or fewer errors will result in a received vector that is not a code word in $\mathcal{C}$ and

hence its syndrome is not equal to zero. Therefore, all the error patterns with $d_{\min}(\mathcal{C}) - 1$ or fewer errors are detectable by the channel decoder.

- However, if a code word $\mathbf{v}$ is transmitted and an error patten with $d_{\min}(\mathcal{C})$ errors that happens to be a code word in $\mathcal{C}$ at a distance $d_{\min}(\mathcal{C})$ from $\mathbf{v}$ occurs, then the received vector $\mathbf{r}$ is a code word and its syndrome is zero. Such an error pattern is an undetectable error.
- This is to say that all the error patterns with $d_{\min}(\mathcal{C}) - 1$ or fewer errors are guaranteed to be detectable, however, detection is not guaranteed for error patterns with $d_{\min}(\mathcal{C})$ or more errors. For this reason, $d_{\min}(\mathcal{C}) - 1$ is called the **error-detecting capability** of the code $\mathcal{C}$. The number of guaranteed detectable error

patterns is

$$\binom{n}{1} + \binom{n}{2} + \ldots + \binom{n}{d_{\min}(\mathcal{C}) - 1}$$

where $\binom{n}{i}$ is a binomial coefficient. For large n, this number is only a small fraction of $2^n - 2^k + 1$ detectable error patterns.

- So far, we have focused on error detection with a linear block code. Decoding and error correcting capability of a linear block code will be discussed in the next section. We will show that a linear block code $\mathcal{C}$ with minimum distance $d_{\min}(\mathcal{C})$ is capable of correcting $\lfloor (d_{\min}(\mathcal{C}) - 1)/2 \rfloor$ or fewer random errors over a span of n transmitted code digits.

3.8 Decoding of Linear Block Codes

- Consider an (n, k) linear block code $\mathcal{C}$ with a parity-check matrix $\mathbf{H}$ and minimum distance $d_{\min}(\mathcal{C})$. Suppose a code word in $\mathcal{C}$ is transmitted and $\mathbf{r}$ is the received vector.
- For maximum-likelihood decoding (MLD) as described in Chapter 1, $\mathbf{r}$ is decoded into a code word $\mathbf{v}$ such that the conditional probability $P(\mathbf{r}|\mathbf{v})$ is maximized.
- For a BSC, this is equivalent to decoding $\mathbf{r}$ into a code word $\mathbf{v}$ such that the Hamming distance $d(\mathbf{r}, \mathbf{v})$ between $\mathbf{r}$ and $\mathbf{v}$ is minimized. This is called **minimum-distance** (or **nearest-neighbor**) decoding.

- With minimum distance decoding, the decoder has to compute the distance between $\mathbf{r}$ and every code word in $\mathcal{C}$ and then choose a code word $\mathbf{v}$ (not necessarily unique) that is closest to $\mathbf{r}$ (i.e., $d(\mathbf{r}, \mathbf{v})$ is the smallest) as the decoded code word.
- This decoding is called a **complete error correction** decoding and requires a total of 2^k computations of distances between $\mathbf{r}$ and the 2^k codewords in $\mathcal{C}$. For large k, implementation of this complete decoder is practically impossible. However, for many linear block codes, efficient algorithms have been developed for incomplete error correction decoding to achieve good error performance with reduced decoding complexity.
- No matter which code word in $\mathcal{C}$ is transmitted over a noisy channel, the received vector $\mathbf{r}$ is one of the 2^n n-tuples over

GF(2). Let $\mathbf{v}_0 = \mathbf{0}, \mathbf{v}_1, \ldots, \mathbf{v}_{2^k-1}$ be the code words in $\mathcal{C}$. Any decoding scheme used at the decoder is a rule to partition $\mathbf{V}$ into 2^k regions; each region contains one and only one code word in $\mathcal{C}$, as shown in Figure 3.5.Decoding is to find the region that contains the received vector $\mathbf{r}$. Then decode $\mathbf{r}$ into the code word $\mathbf{v}$ that is contained in the region.

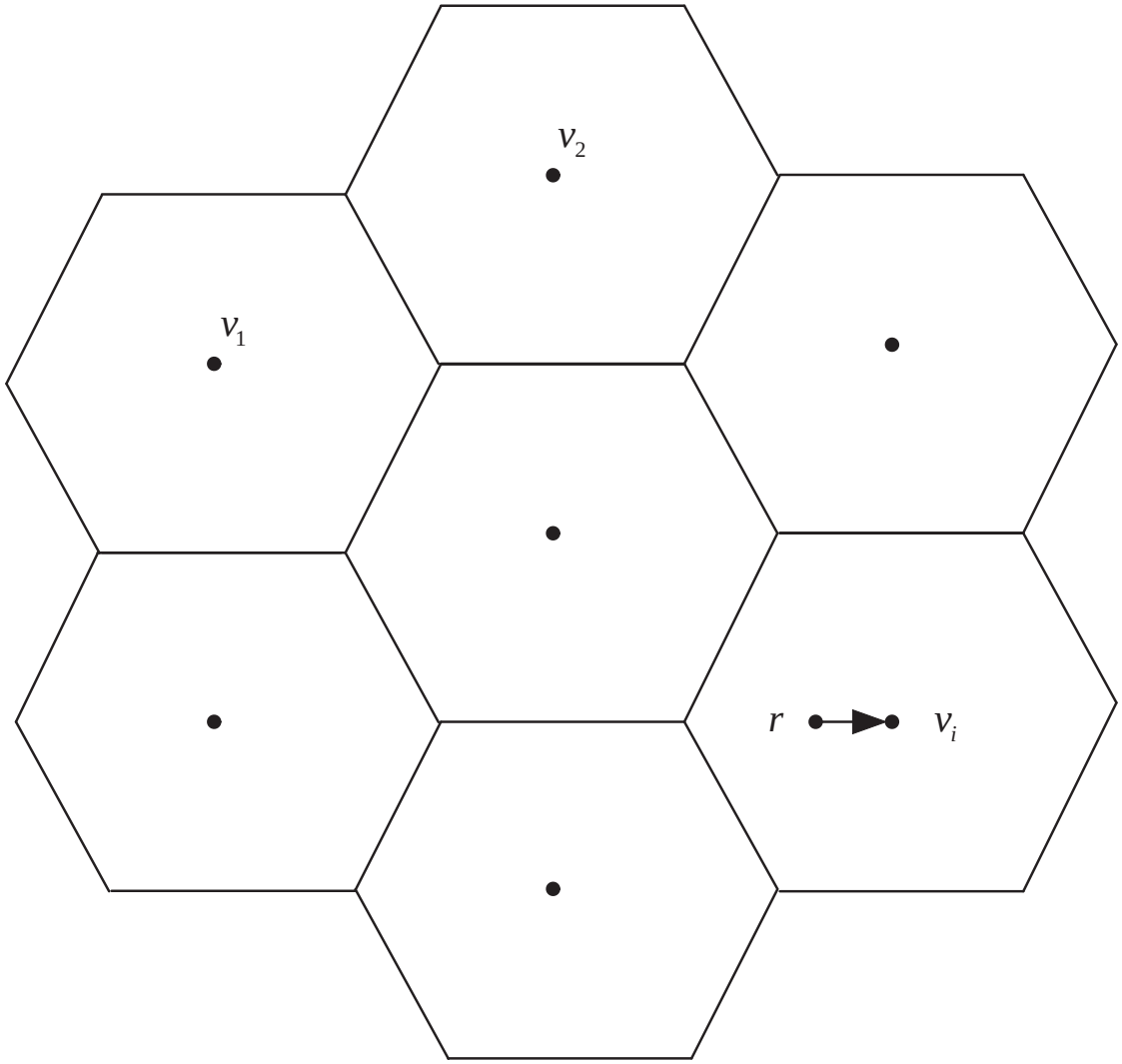


Figure 3.4: Decoding regions.

- These regions are called **decoding regions**.
- For MLD, the decoding regions for the 2^k code words are: for $0 \leq i < 2^k$

$$D(\mathbf{v}_i) = \{\mathbf{r} \in V : P(\mathbf{r}|\mathbf{v}_i) \geq P(\mathbf{r}|\mathbf{v}_j), j \neq i\}. \qquad (3.27)$$

- For minimum-distance decoding, the decoding regions for the 2^k code words are: for $0 \leq i < 2^k$,

$$D(\mathbf{v}_i) = \{\mathbf{r} \in V : d(\mathbf{r}, \mathbf{v}_i) \leq d(\mathbf{r}, \mathbf{v}_j), j \neq i\}. \qquad (3.28)$$

- An algebraic method to partition the 2^n possible received vectors into 2^k decoding regions can be done as follows.
- First, we arrange the 2^k code words in $\mathcal{C}$ as the top row (or 0th row) of a $2^{n-k} \times 2^k$ array with the all-zero code word $\mathbf{v}_0 = \mathbf{0}$ as

the first entry as shown in Figure 3.6.

- Then we form the rest of the rows of the array one at a time. Suppose we have formed the $(j-1)$th row of the array $1 \leq j \leq 2^{n-k}$. To from the jth row of the array, we choose a vector $\mathbf{e}_j$ in V that is not in the previous $j-1$ rows of the array. Form the jth row of the array by adding $\mathbf{e}_j$ to each code word $\mathbf{v}_i$ in the top row of the array and placing the vector sum $\mathbf{e}_j + \mathbf{v}_i$ under $\mathbf{v}_i$.

- The array is completed when no vector can be chosen from V. This array is called a **standard array** for the code $\mathcal{C}$. Each row of the array is called a **coset** of $\mathcal{C}$. The first element of each coset is called the **coset leader**.

- For $0 \leq j < 2^{n-k}$, the jth coset is given by

$$\mathbf{e}_j + \mathcal{C} = \{\mathbf{e}_j + \mathbf{v}_i : \mathbf{v}_i \in \mathcal{C}, 0 \leq i < 2^k\}, \tag{3.29}$$

  where $\mathbf{e}_j$ is the coset leader.

Cosets	**Coset leaders**					
$\mathcal{C}$	$\mathbf{e}_0 = \mathbf{v}_0 = \mathbf{0}$	$\mathbf{v}_1$	$\cdots$	$\mathbf{v}_i$	$\cdots$	$\mathbf{v}_{2^k-1}$
$\mathbf{e}_1 + \mathcal{C}$	$\mathbf{e}_1$	$\mathbf{e}_1 + \mathbf{v}_1$	$\cdots$	$\mathbf{e}_1 + \mathbf{v}_i$	$\cdots$	$\mathbf{e}_1 + \mathbf{v}_{2^k-1}$
$\mathbf{e}_2 + \mathcal{C}$	$\mathbf{e}_2$	$\mathbf{e}_2 + \mathbf{v}_1$	$\cdots$	$\mathbf{e}_2 + \mathbf{v}_i$	$\cdots$	$\mathbf{e}_2 + \mathbf{v}_{2^k-1}$
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$	$\cdots$	$\vdots$
$\mathbf{e}_{2^{n-k}-1} + \mathcal{C}$	$\mathbf{e}_{2^{n-k}-1}$	$\mathbf{e}_{2^{n-k}-1} + \mathbf{v}_1$	$\cdots$	$\mathbf{e}_{2^{n-k}-1} + \mathbf{v}_i$	$\cdots$	$\mathbf{e}_{2^{n-k}-1} + \mathbf{v}_{2^k-1}$

Figure 3.5: A standard array for an (n,k) linear block code.

- **Example 3.5:** A standard array for the $(6, 3)$ code given in Example 3.1 is shown in Figure 3.7.

Coset Leader							
0 0 0 0 0 0	0 1 1 1 0 0	1 0 1 0 1 0	1 1 0 0 0 1	1 1 0 1 1 0	1 0 1 1 0 1	0 1 1 0 1 1	0 0 0 1 1 1
1 0 0 0 0 0	1 1 1 1 0 0	0 0 1 0 1 0	0 1 0 0 0 1	0 1 0 1 1 0	0 0 1 1 0 1	1 1 1 0 1 1	1 0 0 1 1 1
0 1 0 0 0 0	0 0 1 1 0 0	1 1 1 0 1 0	1 0 0 0 0 1	1 0 0 1 1 0	1 1 1 1 0 1	0 0 1 0 1 1	0 1 0 1 1 1
0 0 1 0 0 0	0 1 0 1 0 0	1 0 0 0 1 0	1 1 1 0 0 1	1 1 1 1 1 0	1 0 0 1 0 1	0 1 0 0 1 1	0 0 1 1 1 1
0 0 0 1 0 0	0 1 1 0 0 0	1 0 1 1 1 0	1 1 0 1 0 1	1 1 0 0 1 0	1 0 1 0 0 1	0 1 1 1 1 1	0 0 0 0 11
0 0 0 0 1 0	0 1 1 1 1 0	1 0 1 0 0 0	1 1 0 0 1 1	1 1 0 1 0 0	1 0 1 1 1 1	0 1 1 0 0 1	0 0 0 1 01
0 0 0 0 0 1	0 1 1 1 0 1	1 0 1 0 1 1	1 1 0 0 0 0	1 1 0 1 1 1	1 0 1 1 1 0	0 1 1 0 1 0	0 0 0 1 1 0
1 0 0 1 0 0	1 1 1 0 0 0	0 0 1 1 1 0	0 1 0 1 0 1	0 1 0 0 1 0	0 0 1 0 0 1	1 1 1 1 1 1	1 0 0 0 11

Figure 3.6: A standard array for the (6,3) code given in Example 3.1

- A standard array for an (n, k) linear block code $\mathcal{C}$ with an $(n-k) \times n$ parity-check matrix $\mathbf{H}$ has the following structural properties that can be easily proved:

  (1) The sum of two vectors in the same coset is a code word in $\mathcal{C}$.

  (2) No two vectors in the same coset or in two different cosets

are the same.

(3) Every n-tuple in the vector space $\mathbf{V}$ of all the n-tuples over GF(2) appears once and only once in the array.

(4) All the vectors in a coset have the same syndrome which is the syndrome of the coset leader, i.e,

$$(\mathbf{e}_j + \mathbf{v}_i) \cdot \mathbf{H}^T = \mathbf{e}_j \cdot \mathbf{H}^T (\text{since } \mathbf{v}_i \cdot \mathbf{H}^T = \mathbf{0}).$$

(5) Different cosets have different syndromes.

- Since there are 2^{n-k} different $(n-k)$-tuple syndromes with respect to the parity-check matrix $\mathbf{H}$ of the code $\mathcal{C}$ and there are 2^{n-k} cosets in a standard array for $\mathcal{C}$, it follows from properties (4) and (5) that there is a **one-to-one correspondence** between a coset leader and an $(n-k)$-tuple syndrome.

- A standard array for an (n, k) linear block code $\mathcal{C}$ consists of 2^k columns; each column contains one and only one code word at the top of the column. For $0 \leq i < 2^k$, the ith column consists the following set of 2^{n-k} n-tuples:

$$D_i = \{\mathbf{v}_i, \mathbf{e}_1 + \mathbf{v}_i, \mathbf{e}_2 + \mathbf{v}_i, \ldots, \mathbf{e}_{2^{n-k}-1} + \mathbf{v}_i\}, \qquad (3.30)$$

  where each element is the vector sum of the ith code word $\mathbf{v}_i$ and a coset leader $\mathbf{e}_j$ (note that $\mathbf{e}_0 = \mathbf{0}$) with $0 \leq j < 2^{n-k}$. We see that the ith code word $\mathbf{v}_i$ is the only code word in D_i.

- The 2^k columns of a standard array for $\mathcal{C}$ form a partition of the vector space V of all the n-tuples over GF(2). These 2^k columns of the array can be used as the regions for decoding $\mathcal{C}$. If the received vector $\mathbf{r}$ is found in ith column D_i, we decode $\mathbf{r}$ into $\mathbf{v}_i$.

- From the structure of a standard array for $\mathcal{C}$, we can easily check that:
  (1) If the ith code word $\mathbf{v}_i$ is transmitted and the error pattern caused by the channel noise is a coset leader $\mathbf{e}_j$, then the received vector
  $$\mathbf{r} = \mathbf{v}_i + \mathbf{e}_j$$
  is in the column D_i which contains $\mathbf{v}_i$.
  (2) If $\mathbf{v}_i$ is transmitted but the error pattern is not a coset leader, then the received vector $\mathbf{r}$ is not in column D_i.

- Therefore, using the columns of a standard array of an (n,k) linear block code $\mathcal{C}$ as decoding regions, decoding is correct (i.e., $\mathbf{r}$ is decoded into the transmitted code word) if and only if

a code word $\mathbf{v}_i$ is transmitted and the error pattern caused by the channel noise is identical to a coset leader.

- This is to say that the $2^{n-k} - 1$ nonzero coset leaders of a standard array are all the **correctable error patterns** (i.e., they result in correct decoding). To minimize the probability of a decoding error, the error patterns that are **most likely** to occur for a given channel should be chosen as the coset leaders.
- For a BSC, an error pattern of smaller weight (or smaller number of errors) is more probable than an error pattern with larger weight (or larger number of errors). So when a standard array for a linear block code $\mathcal{C}$ is formed, each coset leader should be chosen to be an n-tuple of the least weight from the remaining available n-tuples in V. Choosing coset leaders in

this manner, each coset leader has minimum weight in each coset. In this case, decoding based on a standard array for $\mathcal{C}$ is the minimum-distance decoding (or MLD for a BSC). A standard array formed this way is called an **optimal standard array** for $\mathcal{C}$.

- The minimum distance $d_{min}(\mathcal{C})$ of a linear block code $\mathcal{C}$ is either odd or even.
- Let $t = \lfloor (d_{min}(\mathcal{C}) - 1)/2 \rfloor$ where $\lfloor x \rfloor$ denote the integer part of x (or the largest integer equal to or less than x). Then

$$2t + 1 \leq d_{min}(\mathcal{C}) \leq 2t + 2.$$

- It can be shown that all the n-tuples over GF(2) of weight t or less can be used as coset leaders in an optimal standard array for

$\mathcal{C}$ (see Problem 3.10). Also, it can be shown that there is at least one n-tuple of weight $t+1$ cannot be used as a coset leader (see Problem 3.10). When this error pattern occurs, received vector $\mathbf{r}$ will be decoded incorrectly.

- Therefore, for a linear code $\mathcal{C}$ with minimum distance $d_{min}(\mathcal{C})$, any error pattern with $t = \lfloor (d_{min}(\mathcal{C}) - 1)/2 \rfloor$ or fewer errors is guaranteed correctable (i.e., resulting in correct decoding), but not all the error patterns with $t+1$ or more errors.
- The parameter $t = \lfloor (d_{min}(\mathcal{C}) - 1)/2 \rfloor$ is called the **error-correction capability** of $\mathcal{C}$. We say that $\mathcal{C}$ is capable of correcting t or fewer random errors and is called a t-error-correction code.
- Decoding of an (n,k) linear block code $\mathcal{C}$ based on an optimal

standard array for the code requires a memory to store 2^n n-tuples. For large n, the size of the memory will be prohibitively large and implementation of a standard-array-based decoding becomes practically impossible.

- However, the decoding can be significantly simplified by using the facts:

  (1) The coset leaders form all the correctable error patterns;

  (2) there is a one-to-one correspondence between an $(n-k)$-tuple syndrome and a coset leader.

- Based on these two facts, we form a table with only two columns that consists of 2^{n-k} coset leaders (correctable error patterns) in one column and their corresponding syndromes in another column as shown in Figure 3.7.

- Decoding of a received vector $\mathbf{r}$ is carried out in three steps:
  1. Compute the syndrome of $\mathbf{r}$, $\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T$.
  2. Find the coset leader $\mathbf{e}$ in the table whose syndrome is equal to $\mathbf{s}$. Then $\mathbf{e}$ is assumed to be the error pattern caused by the channel noise.
  3. Decode $\mathbf{r}$ into the code word $\mathbf{v} = \mathbf{r} + \mathbf{e}$.
- The above decoding is called **syndrome decoding** or **table look-up decoding**. With this decoding, the decoder complexity is drastically reduced compared to the standard-array-based decoding.

Syndromes	**Correctable error patterns**
$\mathbf{0}$	$\mathbf{e}_0 = \mathbf{0}$
$\mathbf{s}_1$	$\mathbf{e}_1$
$\mathbf{s}_2$	$\mathbf{e}_2$
$\ldots$	$\ldots$
$\mathbf{s}_{2^{n-k}-1}$	$\mathbf{e}_{2^{n-k}-1}$

Figure 3.7: A look-up decoding table

- **Example 3.6:** Consider the standard array for the $(6, 3)$ code given in Example 3.1, (see Figure 3.7). The look-up decoding table is shown in Figure 3.8.

Syndromes	**Correctable error patterns**
$s_0 s_1 s_2$	$e_0 e_1 e_2 e_3 e_4 e_5$
000	000000
100	100000
010	010000
001	001000
011	000100
101	000010
110	100100

Figure 3.8: A look-up decoding table for the $(6, 3)$ code given in Examples 3.1 and 3.5.

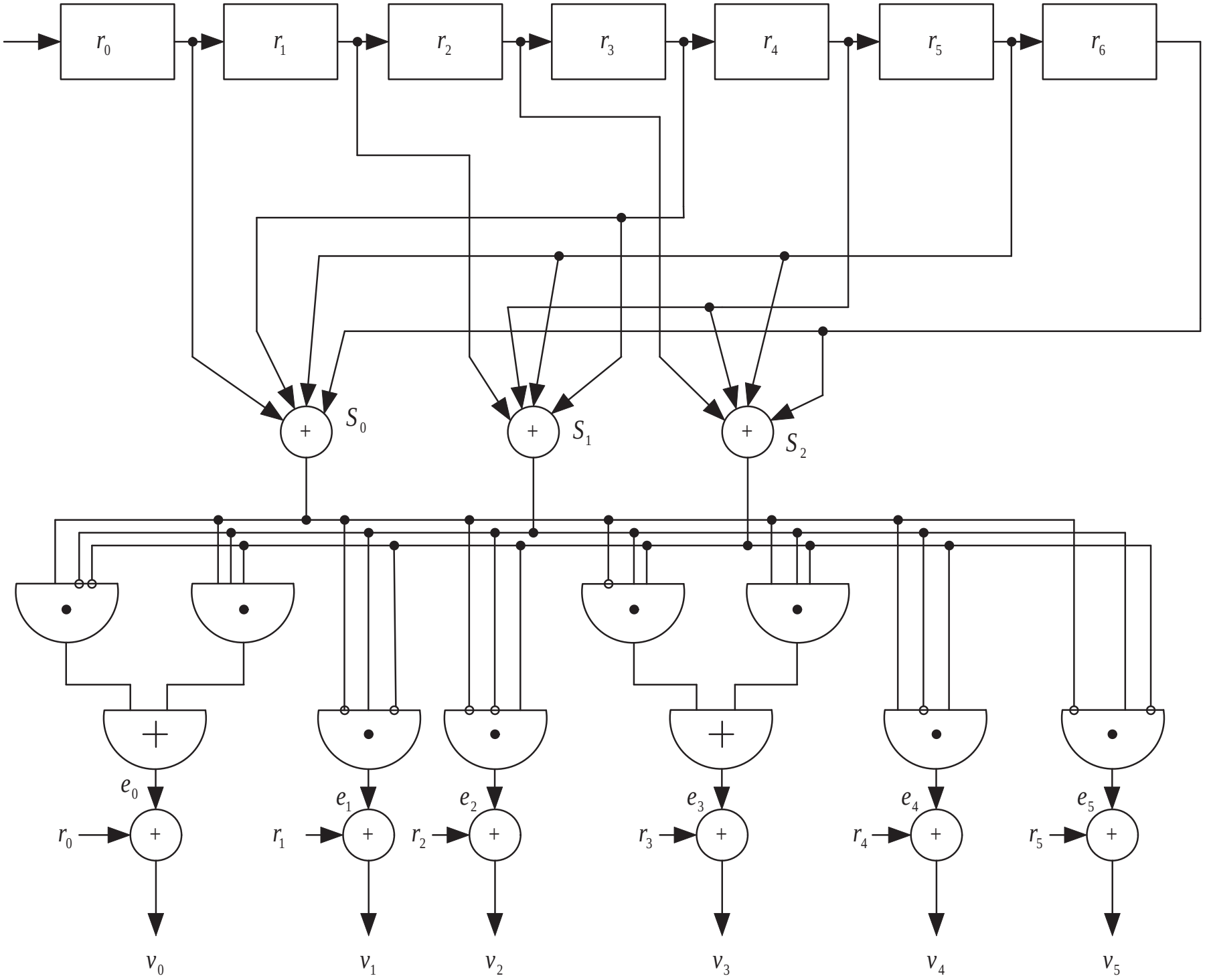
- A combinational logic realization of a decoder for the $(6, 3)$ code based on the look-up table given Figure 3.7 is show in Figure 3.8 with logic expressions for the error digits given below,

$$e_0 = s_0 \cap \overline{s_1} \cap \overline{s_2} + s_0 \cap s_1 \cap s_2$$

$$e_1 = \overline{s_0} \cap s_1 \cap \overline{s_2} \quad e_2 = \overline{s_0} \cap \overline{s_1} \cap s_2$$

$$e_3 = \overline{s_0} \cap s_1 \cap s_2 + s_0 \cap s_1 \cap s_2$$

$$e_4 = s_0 \cap \overline{s_1} \cap s_2 \quad e_5 = s_0 \cap s_1 \cap \overline{s_2}$$


r_0
r_1
r_2
r_3
r_4
r_5
r_6
+
S_0
+
S_1
+
S_2
e_0
e_1
e_2
e_3
e_4
e_5
r_0
r_1
r_2
r_3
r_4
r_5
+
v_0
v_1
v_2
v_3
v_4
v_5

- For a long code with large $n-k$, a complete table look-up decoder is still very complex, requiring a very large memory to store the look-up table. If we limit ourself to correct the error patterns guaranteed by the error correcting capability $t = \lfloor (d_{min}(\mathcal{C}) - 1)/2 \rfloor$ of the code, then the size of the look-up table can be further reduced. The new table consists of only

$$N_t = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}$$

  correctable error patterns guaranteed by the minimum distance $d_{min}(\mathcal{C})$ of the code.

- Based on this new table, decoding of a received vector $\mathbf{r}$ consists of the following four steps:

  1. Compute the syndrome $\mathbf{s}$ of $\mathbf{r}$.

2. If the syndrome $\mathbf{s}$ corresponds to a correctable error pattern $\mathbf{e}$ listed in the table, go to step (3), otherwise go to step (4).
3. Decode $\mathbf{r}$ into the code word $\mathbf{v} = \mathbf{r} + \mathbf{e}$.
4. Declare a **decoding failure**. In this case, the presence of errors is detected but the decoder fail to correct the errors.

- With the above partial table look-up decoding, the number of errors to be corrected is bounded by the error-correction capability

$$t = \lfloor (d_{min}(\mathcal{C}) - 1)/2 \rfloor$$

of the code. It is called **bound-distance decoding**.

- Many classes of linear block codes with good error-correction capabilities have been constructed over the years. Efficient algorithms to carry out the bound-distance decoding of these

classes of codes have been devised.

Weight Structure and Parity-Check Matrix

- The weight structure of a linear block code is related to its parity-check matrix.
- Consider an (n,k) linear code with a parity-check matrix:

$$\mathbf{H} = [\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-1}],$$

  where $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-1}$ are the columns of $\mathbf{H}$.
- Let $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})$ be a codeword of weight l. Let

$$v_{i_1}, v_{i_2}, \ldots, v_{i_l}$$

  be the l nonzero components of $\mathbf{v}$ where $0 \leq i_1 < i_2, \cdots < i_l < n$. Then

$$v_{i_1} = v_{i_2} = \cdots = v_{i_l} = 1.$$

- Since $\mathbf{v}$ is a codeword,

$$\mathbf{v} \cdot \mathbf{H}^T = \mathbf{0}.$$

This says that

$$
\begin{aligned}
(v_0, v_1, \cdots, v_{n-1}) \begin{bmatrix} \mathbf{h}_0^T \\ \mathbf{h}_1^T \\ \vdots \\ \mathbf{h}_{n-1}^T \end{bmatrix} &= \mathbf{0} \\
v_0\mathbf{h}_0^T + v_1\mathbf{h}_1^T + \cdots + v_{n-1}\mathbf{h}_{n-1}^T &= \mathbf{0} \\
v_{i_1}\mathbf{h}_{i_1}^T + v_{i_2}\mathbf{h}_{i_2}^T + \cdots + v_{i_l}\mathbf{h}_{i_l}^T &= \mathbf{0} \\
\mathbf{h}_{i_1}^T + \mathbf{h}_{i_2}^T + \cdots + \mathbf{h}_{i_l}^T &= \mathbf{0}
\end{aligned}
$$

- This says that for every codeword of weight l, there exist l columns in $\mathbf{H}$ which sum to $\mathbf{0}$.
- Next we show that for every l columns in $\mathbf{H}$ which sum to $\mathbf{0}$, there exists a codeword with weight l.

- Suppose $\mathbf{h}_{i_1}, \mathbf{h}_{i_2}, \cdots, \mathbf{h}_{i_l}$ are l columns in $\mathbf{H}$ such that

$$\mathbf{h}_{i_1} + \mathbf{h}_{i_2} + \cdots + \mathbf{h}_{i_l} = \mathbf{0}.$$

We construct an n-tuple $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})$ whose nonzero components are $v_{i_1}, v_{i_2}, \cdots, v_{i_l}$, i.e., $v_{i_1} = v_{i_2} = \cdots = v_{i_l} = 1$. Then

$$\begin{aligned} \mathbf{v} \cdot \mathbf{H}^T &= v_0\mathbf{h}_0 + v_1\mathbf{h}_1 + \cdots + v_{n-1}\mathbf{h}_{n-1} \\ &= v_{i_1}\mathbf{h}_{i_1} + v_{i_2}\mathbf{h}_{i_2} + \cdots + v_{i_l}\mathbf{h}_{i_l} \\ &= \mathbf{h}_{i_1} + \mathbf{h}_{i_2} + \cdots + \mathbf{h}_{i_l} = \mathbf{0}. \end{aligned}$$

- This says that $\mathbf{v}$ is a codeword of weight l.

- Summarizing the above results, we have the following theorem.

  Theorem: Let C be a linear code with parity-check matrix $\mathbf{H}$. For each codeword of weight l, there exist l columns in $\mathbf{H}$ that sum to $\mathbf{0}$. Conversely, if there exist l columns in $\mathbf{H}$ that sum to $\mathbf{0}$, there exists a codeword of weight l.

- This theorem can be used to determine the weight structure of a linear code, especially the minimum distance of the code.

- This theorem is also used for code construction.

- Two direct results are:

  Corollary 1: If no $d-1$ of fewer columns of $\mathbf{H}$ sum to $\mathbf{0}$, the code with $\mathbf{H}$ as the parity-check matrix has minimum distance at least d.

  Corollary 2: The minimum weight of a linear code with parity-check matrix $\mathbf{H}$ is equal to the smallest number of columns in $\mathbf{H}$ sum to $\mathbf{0}$.

Error Correction Capability

- Let C be an (n,k) linear block code with minimum Hamming distance d_{min} (also minimum Hamming weight).

  Suppose d_{min} is odd. Let

$$t = \lfloor \frac{d_{min} - 1}{2} \rfloor$$

- *Theorem*: No two vectors with weight t or less can be in the same coset.

  Proof: Let $\mathbf{e}_1$ and $\mathbf{e}_2$ be two vectors for which $w(\mathbf{e}_1) \leq t$ and $w(\mathbf{e}_2) \leq t$. Suppose they are in the same coset. Then $\mathbf{e}_1 + \mathbf{e}_2$ must be a codeword, $\mathbf{v}$, i.e.,

$$\mathbf{e}_1 + \mathbf{e}_2 = \mathbf{v}.$$

  Then $w(\mathbf{e}_1 + \mathbf{e}_2) = w(\mathbf{v}) \leq w(\mathbf{e}_1) + w(\mathbf{e}_2) \leq 2t < d_{min}$. This contradicts to the fact that the minimum weight of C is d_{min}. Therefore $\mathbf{e}_1$ and $\mathbf{e}_2$ can not be in the same coset.

- *Corollary*: All the vectors of weight t or less are in different cosets and hence can be used as coset leaders.

- There are

$$1 + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{t}$$

  such vectors and they are correctable error patterns.

- *Theorem*: For a code with $d_{min} = 2t + 1$, there is at least one error pattern with weight $t + 1$ is not correctable.

  Proof: Let $\mathbf{v}$ be a codeword with $w(\mathbf{v}) = d_{min} = 2t + 1$. Let $\mathbf{e}_1$ and $\mathbf{e}_2$ be two vectors such that:

  (1) $\mathbf{e}_1$ and $\mathbf{e}_2$ do not have common nonzero components.

  (2) $w(\mathbf{e}_1) = t$ and $w(\mathbf{e}_2) = t + 1$.

  (3) $\mathbf{e}_1 + \mathbf{e}_2 = \mathbf{v}$.

  Suppose $\mathbf{e}_1$ is used as a coset leader, then

  $$\mathbf{e}_2 = \mathbf{e}_1 + \mathbf{v}$$

  is in the coset with $\mathbf{e}_1$ as coset leader. Therefore $\mathbf{e}_2$ can not be used as a coset leader and not correctable.

- Consider the case $d_{min} = 2t + 2$. The above theorems are also true and can be proved in the same manner.

- *Theorem*: If a linear block code is capable of correcting all the error patterns of t or fewer errors, the minimum distance of the code is at least $2t + 1$, i.e.

  $$d_{min} \geq 2t + 1.$$

Structural Properties of A Standard Array

1. The sum of two vectors in the same coset is a codeword in C.

   Two vectors in the same coset must be the forms: $\mathbf{e}_i + \mathbf{v}_j$ and $\mathbf{e}_i + \mathbf{v}_l$. Their sum is $(\mathbf{e}_i + \mathbf{v}_j) + (\mathbf{e}_i + \mathbf{v}_l) = \mathbf{v}_j + \mathbf{v}_l$ which is a codeword.

2. All the vectors in the same coset are distinct.

   Suppose two vectors, $\mathbf{e}_i + \mathbf{v}_j$ and $\mathbf{e}_i + \mathbf{v}_l$ are the same. Then $\mathbf{e}_i + \mathbf{v}_j = \mathbf{e}_i + \mathbf{v}_l$. This implies that $\mathbf{v}_j = \mathbf{v}_l$. This is a contradiction to the fact that all the codewords of an (n,k) linear block code are distinct.

3. No two vectors in two different cosets are the same.

   Suppose two vectors, $\mathbf{e}_i + \mathbf{v}_l$ and $\mathbf{e}_j + \mathbf{v}_m$, are the same, where $i < j$. Then

   $$\mathbf{e}_i + \mathbf{v}_l = \mathbf{e}_j + \mathbf{v}_m.$$

   and $\mathbf{e}_j = \mathbf{e}_i + \mathbf{v}_l + \mathbf{v}_m = \mathbf{e}_i + \mathbf{v}_t$. This says that the coset leader $\mathbf{e}_j$ appears in the i-th coset. This contradicts to the construction of a standard array that every coset leader should not be used before.

4. Every n-tuple in V_n appears one and only once in a standard array.

5. Different cosets have different syndromes.

   Suppose

$$\mathbf{e}_i \cdot \mathbf{H}^T = \mathbf{e}_j \cdot \mathbf{H}^T.$$

   Then

$$(\mathbf{e}_i + \mathbf{e}_j) \cdot \mathbf{H}^T = 0.$$

   This implies that $\mathbf{e}_i + \mathbf{e}_j$ is a codeword, say

$$\mathbf{e}_i + \mathbf{e}_j = \mathbf{v}_k.$$

   Then

$$\mathbf{e}_j = \mathbf{e}_i + \mathbf{v}_k.$$

   This is a contradiction to the fact that $\mathbf{e}_i$ and $\mathbf{e}_j$ are in different cosets.

Incorrect Decoding

- If the error pattern $\mathbf{e}$ is not a coset leader, it is not correctable and decoding will be incorrect.
- Suppose $\mathbf{v}_j$ is transmitted and the error pattern $\mathbf{e}$ caused by channel noise is not a coset leader. Then $\mathbf{e}$ is vector in some coset and under some codeword, say

$$\mathbf{e} = \mathbf{e}_i + \mathbf{v}_k.$$

Then

$$\begin{aligned} \mathbf{r} &= \mathbf{v}_j + \mathbf{e} \\ &= \mathbf{v}_j + \mathbf{e}_i + \mathbf{v}_k \\ &= \mathbf{e}_i + (\mathbf{v}_j + \mathbf{v}_k) \\ &= \mathbf{e}_i + \mathbf{v}_l \end{aligned}$$

This says that $\mathbf{r}$ is under the codeword $\mathbf{v}_l$ other than the transmitted codeword $\mathbf{v}_j$. Consequently, $\mathbf{r}$ is decoded into an incorrect codeword $\mathbf{v}_l$ and the decoder commits a decoding error.

- Let $\mathbf{r}$ be the received sequence.

  Suppose $\mathbf{r}$ is found in the i-th coset and under the codeword $\mathbf{v}_j$. Then $\mathbf{r} = \mathbf{e}_i + \mathbf{v}_j$ and

$$d(\mathbf{r}, \mathbf{v}_j) = d(\mathbf{e}_i + \mathbf{v}_j, \mathbf{v}_j) = w(\mathbf{e}_i)$$

- Consider

$$\begin{aligned} d(\mathbf{r}, \mathbf{v}_l) &= w(\mathbf{e}_i + \mathbf{v}_j + \mathbf{v}_l) \\ &= w(\mathbf{e}_i + \mathbf{v}_k) \\ &\geq w(\mathbf{e}_i) = d(\mathbf{r}, \mathbf{v}_j) \end{aligned}$$

- This says that the syndrome decoding with the vector of smallest weight as the coset leader is minimum distance decoding (Hard-decision MLD).