4.4 CYCLIC CODES

In coding theory, cyclic codes are linear block error-correcting codes that have convenient algebraic structures for efficient error detection and correction.

Let C be a linear code over a finite field $GF(q)^n$ of block length *n*. C is called a cyclic code, if for every codeword $c=(c_1,...,c_n)$ from *C*, the word $(c_n,c_1,...,c_{n-1})$ in $GF(q)^n$ obtained by a cyclic right shift of components is again a codeword. Same goes for left shifts. One right shift is equal to $n-1$ left shifts and vice versa. Therefore the linear code C is cyclic precisely when it is invariant under all cyclic shifts.

Cyclic Codes have some additional structural constraint on the codes. They are based on Galois fields and because of their structural properties they are very useful for error controls. Their structure is strongly related to Galois fields because of which the encoding and decoding algorithms for cyclic codes are computationally efficient.

Cyclic code for correcting error:

a) For correcting single error

The cyclic codes explicitly with error detection and correction. Cyclic codes can be used to correct errors, like Hamming codes as a cyclic codes can be used for correcting single error. Likewise, they are also used to correct double errors and burst errors. All types of error corrections are covered briefly in the further subsections.

The Hamming code has a generator polynomial $g(x)=x^3+x+1$. This polynomial has a zero in Galois extension field GF(8) at the primitive element a, and all codewords satisfy .

C(a)=0 Cyclic codes can also be used to correct double errors over the field GF(2). Blocklength will be n equal to 2^m-1 and primitive elements a and a^3 as zeros in the $GF(2^m)$ because we are considering the case of two errors here, so each will represent one error.The received word is a polynomial of degree n-1 given as

$$v(x) = a(x)g(x) + e(x)$$

where e(x) can have at most two nonzero coefficients corresponding to 2 errors.

Syndrome Polynomial, S(x) as the remainder of polynomial v(x) when divided by the

generator polynomial g(x) i.e.

S(x)=v(x) mod g(x)= (a(x)g(x)+e(x)) mod g(x)=e(x) mod g(x) as (a(x)g(x)) mod g(x) is zero

b) For correcting two errors

Let the field elements $X_1$and X_2 be the two error location numbers. If only one error occurs then

X_2 is equal to zero and if none occurs both are zero.

Let $S_1 = v(\alpha)$ and $S_3 = v(\alpha^3)$.

These field elements are called "syndromes". Now because $g(x)$is zero at primitive elements $\alpha$and α^3, so we can write $S_1 = e(\alpha)$ and $S_3 = e(\alpha^3)$. If say two errors occur, then

$$S_1 = \alpha^i + \alpha^{i'} \text{ and } S_3 = \alpha^{3i} + \alpha^{3i'}.$$

And these two can be considered as two pair of equations in GF(2m) with two unknowns and

hence we can write

$$S_1 = X_1 + X_2 \text{ and } S_3 = (X_1)^3 + (X_2)^3.$$

Hence if the two pair of nonlinear equations can be solved cyclic codes can used to correct two errors.

Syndrome Calculator

We know $r(X) = q(X)g(X) + S(X)$.

We can also write $r(X) = c(X) + e(X)$.

Rewrite the expression to:

$$(X) = c(X) + r(X)$$
$$= c(X) + q(X)g(X) + S(X)$$
$$= (f(X) + q(X))g(X) + S(X)$$

If error polynomial $e(X)$ is divided by generator polynomial $g(X)$, the remainder is the syndrome polynomial $S(X)$.

Error Detection

$$r(X) = c(X) + e(X) = q(X)g(X) + S(X)$$
$$e(X) = c(X) + q(X)g(X) + S(X) = (f(X) + q(X))g(X) + S(X)$$

Investigate error detecting capability of cyclic code:

Assuming $e(X)$ is a burst of length $n - k$ or less, i.e., errors are confined to $n - k$ or fewer consecutive positions;

$e(X)$ can be expressed by $e(X) = XjB(X)$, here $B(X)$ is a polynomial of degree $n – k –1$ or less;

Xj cannot divided by $g(X)$, $B(X)$ cannot divided by $g(X)$ neither, $e(X) = X^{ij}B(X)$ is NOT divisible by $g(X)$;

Thus the syndrome polynomial is not equal to zero.

It means that a cyclic code *Ccyc* (n,k) can detect any error burst of length $n - k$ or less. A cyclic code *Ccyc* $(n; k)$ can also detect all the *end-around* error bursts of length $n - k$ or less.

e = (1 1 0 0 0 0 0 0 1 0 1)

Cyclic Redundancy Check (CRC) codes:

Cyclic redundancy .check codes are extremely well suited for "error detection". The two important reasons for this statement are,

(1) they can be designed to detect many combinations of likely errors.

(2) The implementation of both encoding and error detecting circuits is practical.

Accordingly, all error detecting codes used in practice, virtually, are of theCRC - type. In ann-bit received word if a contiguous sequence of _b-bits‘ in which the first and the last bits and any number of intermediate bits are received in error, then we say aCRC "error burst' of length has occurred. Such an error burst may also include an end-shifted version of the contiguous sequence.

In any event, Binary (n, k)CRC codes are capable of detecting the following error patterns:

1. All CRC error bursts of length (n-k) or less.
2. A fraction of (1 - 2 (n –k - 1)) of CRC error bursts of length (n – k + 1).
3. A fraction (1-2(n – k)) of CRC error bursts of length greater than (n – k + 1).
4. All combinations of (d min – 1) or fewer errors.
5. All error patterns with an odd number of errors if the generator polynomial g (X) has an even number of non zero coefficients.

Generator polynomials of three CRC codes, internationally accepted as standards are listed below.

All three contain (1 +X) as a prime factor. TheCRC-12 code is used when the character lengths is
6- bits. The others are used for 8-bit characters.

* CRC-12 code: g (X) = 1 + X + X^2+X^3+ X^{11} + X^{12}*

*CRC-16 code: g (X) = 1 +X^2+ X^{15} + X^{16}

*CRC-CCITT code: g (X) = 1 + X^5+ x^{12}+ X^{26} .

Definition 1 An (n, k) linear block code C is said to be **cyclic** if for every code word $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ in C, there is also a code word $\mathbf{c}' = (c_{n-1}, c_0, \ldots, c_{n-2})$ that is also in C. ($\mathbf{c}'$ is a cyclic shift of $\mathbf{c}$.) □

It will be convenient to represent our codewords as *polynomials*. The codeword

$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$$

is represented by the polynomial

$$c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$$

using the obvious one-to-one correspondence. A cyclic shift can therefore be represented as follows. Observe that

$$xc(x) = c_0 x + c_1 x + \cdots + c_{n-1} x^n.$$

If we not take this product modulo $x^n - 1$ we get

$$xc(x) \pmod{x^n - 1} = c_{n-1} + c_0 x + \cdots + x_{n-2} x^{n-1}.$$

So multiplication by x in the ring $GF(q)[x]/(x^n-1)$ corresponds to a cyclic shift. Furthermore, any power of x times a codeword yields a codeword (apply the definition recursively), so that, for example,

$$\begin{aligned}(c_{n-1}, c_0, c_1, \ldots, c_{n-2}) &\leftrightarrow xc(x)\\ (c_{n-2}, c_{n-1}, c_0, \ldots, c_{n-3}) &\leftrightarrow x^2c(x)\\ &\vdots\\ (c_1, c_2, \ldots, c_{n-1}, c_0) &\leftrightarrow x^{n-1}c(x)\end{aligned}$$

where the arithmetic is done in the ring $GF(q)[x]/(x^n-1)$. Now observe that if we take an polynomial $a(x) \in GF(q)[x]$ of the form

$$a(x) = a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$$

then

$$c(x)a(a)$$

is simply a linear combination of cyclic shifts of $c(x)$ and hence, must also be a codeword. Hence: **a cyclic code is an ideal** in $GF(q)[x]/(x^n-1)$. Because of what we know about ideals in $GF(q)[x]/(x^n-1)$ we can immediately make some observations about cyclic codes:

- A cyclic code has a generator polynomial $g(x)$, which is the generator of the ideal. Let the degree of g be r, where $r < n$.

- Every code polynomial in the code can be expressed as a multiple of the generator.

  $$c(x) = m(x)g(x),$$

  where $m(x)$ is the *message polynomial*. The degree of m is less than $n-r$.

- The generator is a factor of x^n-1 in $GF(q)[x]$.

Example 1 We will consider cyclic codes of length 15 with binary coefficients. We need to find the factors of x^n-1 in some field. Observe that

$$15|2^4-1,$$

so we are dealing in the field $GF(16)$. The conjugacy classes in $GF(16)$ are

$$\begin{aligned}\{1\} &\leftrightarrow x+1\\ \{\alpha, \alpha^2, \alpha^4, \alpha^8\} &\leftrightarrow 1+x+x^4\\ \{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\} &\leftrightarrow 1+x+x^2+x^3+x^4\\ \{\alpha^5, \alpha^{10}\} &\leftrightarrow 1 \mid x \mid x^2\\ \{\alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11}\} &\leftrightarrow 1+x^3+x^4\end{aligned}$$

Thus

$$x^{15}-1=(x+1)(1+x+x^4)(1+x+x^2+x^3+x^4)(1+x+x^2)(1+x^3+x^4)$$

So: degrees 1,2,4,4,4. If we want a generator of, say, degree 9, we could take

$$g(x)=(x+1)(1+x+x^4)(1+x+x^2+x^3+x^4)$$

If we want a generator of degree 5 we could take

$$g(x)=(x+1)(1+x+x^4)$$

or

$$g(x)=(x+1)(1+x+x^2+x^3+x^4)$$

In fact, in this case, we can get generator polynomials of any degree from 1 to 15. So we have codes

$$(15,0)(15,1),\ldots,(15,15)$$

□

A message sequence $(m_0,m_1,\ldots,m_{k-1})$ (where $k=n-r$) corresponds to a message polynomial

$$m(x)=m_0+\cdots+m_{k-1}x^{k-1}.$$

Then the message polynomial corresponding to m is

$$c_m(x)=m(x)g(x).$$

We can write this as

$$c_m(x)=[m_0,m_1,\ldots,m_{k-1}]\begin{bmatrix} g(x) \\ xg(x) \\ \vdots \\ x^{k-1}g(x)\end{bmatrix}$$

Taking the next step, we can go back to a matrix representation,

$$\mathbf{c}_m=[m_0,m_1,\ldots,m_{k-1}]\begin{bmatrix} g_0 & g_1 & \cdots & g_r & & & & \\ & g_0 & g_1 & \cdots & g_r & & & \\ & & g_0 & g_1 & \cdots & g_r & & \\ & & & \ddots & \ddots & & \ddots & \\ & & & & g_0 & g_1 & \cdots & g_r \\ & & & & & g_0 & g_1 & \cdots & g_r \end{bmatrix}=\mathbf{m}G$$

So we have a linear code, and can write the generator matrix corresponding to it.
Note: G is $k\times n$.

Let $h(x)$ be **parity check polynomial**, that is a polynomial such that

$$x^n-1=g(x)h(x).$$

Since codewords are multiples of $g(x)$, then for a codeword,

$$c(x)h(x)=m(x)g(x)h(x)=m(x)(x^n-1)\equiv 0 \pmod{x^n-1}.$$

We let

$$s(x)=c(x)h(x) \pmod{x^n-1}.$$

be the **syndrome polynomial**. If $s(x)$ is identically zero, then $c(x)$ is a codeword.
Now let's put this in matrix form.

$$s(x)=c(x)h(x)=\sum_{i=0}^{n-1}c_ix^i\sum_{j=0}^{n-1}h_jx^j \pmod{x^n-1}.$$

Performing the multiplication,

$$s_k = \sum_{i=0}^{n-1} c_i h_{((k-i))_n} \qquad k = 0, 1, \ldots, n-1.$$

Writing the last $n-k$ of these out, we have

$$[s_k, s_{k+1}, \ldots, s_{n-1}] = [c_0, c_1, \ldots, c_{n-1}] \begin{bmatrix} h_k & h_{k-1} & \cdots & h_1 & h_0 & & & & & \\ & h_k & h_{k-1} & \cdots & h_1 & h_0 & & & & \\ & & \ddots & \ddots & & & & & & \\ & & & & h_k & h_{k-1} & \cdots & h_1 & h_0 & \\ & & & & & h_k & h_{k-1} & \cdots & h_1 & h_0 \end{bmatrix}^T = \mathbf{c}H^T.$$

LINEAR BLOCK CODES.

Error-Control Coding

Error-control coding techniques are used to detect and/or correct errors that occur in the message transmission in a digital communication system. The transmitting side of the error-control coding adds redundant bits or symbols to the original information signal sequence. The receiving side of the error-control coding uses these redundant bits or symbols to detect and/or correct the errors that occurred during transmission. The transmission coding process is known as *encoding*, and the receiving coding process is known as *decoding*.
There are two major classes in error-control code: block and convolutional. In block coding, successive blocks of *K* information (message) symbols are formed.

The coding algorithm then transforms each block into a codeword consisting of *n* symbols where *n>k*. This structure is called an (*n,k*) code. The ratio*k/n* is called the code rate. A key point is that each codeword is formed independently from other codewords.

An error-control code is a *linear code* if the transmitted signals are a linear function of the information symbols. The code is called a *systematic code* if the information symbols are transmitted without being altered. Most block codes are systematic, whereas most convolutional codes are nonsystematic.

Almost all codes used for error control are linear. The symbols in a code can be either binary or non-binary. Binary symbols are the familiar '0' and '1'.

Linear Block Codes

Linear block coding is a generic coding method. Other coding methods, such as Hamming and BCH codes, are special cases of linear block coding. The codeword

vector of a linear block code is a linear mapping of the message vector. The codeword ***x*** and the message ***m*** have the relationship
x =**mG**

where **G** is a *K*-by-*N* matrix and is known as the *generator matrix*.

Linear block code is called a systematic linear code if the generator matrix has the form

G =[**P** **I***k*]

where **P** is an *(n-k)-by-k* matrix and **I***k* is a *k-by-k* identity matrix. A systematic linear code renders a length *k* message into a length *n* codeword where the last *k* bits are exactly the original message and the first *(n-k)* bits are redundant. These redundant bits serve as parity-check digits.