A linear code is said to be cyclic if the $i$th cyclic rotation is also a codeword of the same code. Recalling groups, a group is said to be cyclic if it can be generated by successive powers of a given element.

A cyclic code can be represented as polynomials defined over a Galois field $\mathbb{GF}(2^n)$. A codeword $c = (c_0, c_1, \cdots, c_{n-1})$ is in the form:

$$c(X) = c_0 + c_1 + \cdots + c_{n-1}X^{n-1}, \quad c_i \in \mathbb{GF}(2)$$

Polynomials over $\mathbb{GF}(2)$ can be added (subtracted), multiplied, and divided in the usual way using binary field arithmetic.

A given polynomial $f(X)$ over $\mathbb{GF}(2)$ having an even number of terms, is divisible by $X + 1$. Why?

**Definition 1** A polynomial $f(X)$ over $\mathbb{GF}(2)$ of degree $m$ is said to be irreducible if it is not divisible by any polynomial over $\mathbb{GF}(2)$ of degree less than $m$ but greater than zero.

**Example:** $X^3 + X + 1$ does not have either 0 or 1 as a root and so is not divisible by $X$ or $X + 1$. Since it is not divisible by any polynomial of degree 1, then it is not divisible by a polynomial of degree 2. Why?

**Theorem 1** Any irreducible polynomial over $\mathbb{GF}(2)$ of degree $m$ divides $X^{2^m-1} + 1$.

**Definition 2** An irreducible polynomial $f(X)$ of degree $m$ is said to be primitive if the smallest positive integer $n$ for which $f(X)$ divides $X^n + 1$ is $n = 2^m - 1$. Recall the definition of primitive elements in $\mathbb{GF}(q)$.

**Construction of Galois Fields $\mathbb{GF}(2^n)$**

Given a primitive polynomial of degree $m$ over $\mathbb{GF}(2)$ $p(X)$, let $\alpha$ be a new element such that $p(\alpha) = 0$. Since $p(X)$ divides $X^{2^m-1} + 1$, then,

$$X^{2^m-1} + 1 = q(X)p(X)$$
Replacing:
\[ \alpha^{2^n-1} + 1 = q(\alpha) \ p(\alpha) = 0 \rightarrow \alpha^{2^n-1} = 1 \]

Then, \( \alpha \) is a primitive element of GF(\( 2^n \))

**Example**

Let choose \( p(X) = 1 + X + X^4 \) as the primitive polynomial, we can construct GF(\( 2^4 \)) as powers of \( \alpha \), where \( \alpha \) is a root of \( p(X) \) as follows:

\[ 1 + \alpha + \alpha^4 = 0, \text{ then } \alpha^4 = 1 + \alpha. \]

This relation can be used in order to generate all the elements.

<table>
<thead>
<tr>
<th>Power</th>
<th>Polynomial</th>
<th>4-Tuple</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1000</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>( \alpha )</td>
<td>0100</td>
</tr>
<tr>
<td>( \alpha^2 )</td>
<td>( \alpha^2 )</td>
<td>0010</td>
</tr>
<tr>
<td>( \alpha^3 )</td>
<td>( \alpha^3 )</td>
<td>0001</td>
</tr>
<tr>
<td>( \alpha^4 )</td>
<td>1 + ( \alpha )</td>
<td>1100</td>
</tr>
<tr>
<td>( \alpha^5 )</td>
<td>( \alpha + \alpha^2 )</td>
<td>0110</td>
</tr>
<tr>
<td>( \alpha^6 )</td>
<td>( \alpha^2 + \alpha^3 )</td>
<td>0011</td>
</tr>
<tr>
<td>( \alpha^7 )</td>
<td>1 + ( \alpha + \alpha^3 )</td>
<td>1101</td>
</tr>
<tr>
<td>( \alpha^8 )</td>
<td>1 + ( \alpha^2 )</td>
<td>1010</td>
</tr>
<tr>
<td>( \alpha^9 )</td>
<td>( \alpha + \alpha^3 )</td>
<td>0101</td>
</tr>
<tr>
<td>( \alpha^{10} )</td>
<td>1 + ( \alpha + \alpha^2 )</td>
<td>0010</td>
</tr>
<tr>
<td>( \alpha^{11} )</td>
<td>( \alpha + \alpha^2 + \alpha^3 )</td>
<td>0111</td>
</tr>
<tr>
<td>( \alpha^{12} )</td>
<td>1 + ( \alpha + \alpha^2 + \alpha^3 )</td>
<td>1111</td>
</tr>
<tr>
<td>( \alpha^{13} )</td>
<td>1 + ( \alpha^2 + \alpha^3 )</td>
<td>1011</td>
</tr>
<tr>
<td>( \alpha^{14} )</td>
<td>1 + ( \alpha^3 )</td>
<td>1001</td>
</tr>
</tbody>
</table>

It can be shown (see [2] Chap. 2) that the set \( F^* = \{0, 1, \alpha, \alpha^2, \cdots, \alpha^{2^n-2}\} \) is a commutative group under an addition operation "+" and the non zero elements of \( F^* \) form a commutative group under a multiplication operation ".". Then there is an isomorphism between the set \( F^* \) and the set of polynomials of degree \( n - 1 \).

**Theorem 2** The \( 2^n - 1 \) non zero elements of GF(\( 2^n \)) form all the roots of \( X^{2^n-1} + 1 \).

Shifting of codewords can be easily managed in polynomial representation, being \( c(X) = c_0 + c_1 X + \cdots + c_{n-1} X^{n-1} \) a given polynomial, let’s
write $c(X)$ in another way:

$$c(X) = \sum_{l=1}^{n} c_{n-l}X^{n-l}$$

then, the shifted polynomial is (check it out):

$$c^{(i)}(X) = \sum_{l=0}^{n-1} c_{n-i+l}X^{l}$$

multiplying $c(X)$ by $X^i$, we get:

$$X^i \sum_{l=1}^{n} c_{n-l}X^{n-l} = \sum_{l=1}^{n} c_{n-l}X^{n-l+i} = \sum_{l=1}^{i} \left( c_{n-l}X^{i-l}(X^n + 1) + c_{n-l}X^{i-l} \right) + \sum_{l=i+1}^{n} c_{n-i}X^{n-l+i}$$

(1)

Then (verify):

$$X^i\ c(X) = q(X) \ (X^n + 1) + c^{(i)}(X)$$

the shifted codeword can be obtained as:

$$c^{(i)}(X) = X^i\ c(X) \ mod (X^n + 1)$$

Being a cyclic code a linear code, the corresponding $G$ matrix is given by:

$$G = \begin{pmatrix}
g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\
g_0 & g_1 & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k}
\end{pmatrix}$$
Cyclic Redundant Codes

1. In cyclic codes, any codeword is represented by a polynomial, e.g. 1101 is represented as $X^3 + X^2 + 1$. It is used the algebra of polynomials mod $(X^n + 1)$.

2. Every divisor $g(X)$ of $(X^n + 1)$ generates an $(n, k)$ cyclic code, where $r = n - k$ is the degree of $g(X)$.

3. Primitive polynomials can be used as generator polynomials.

4. Every codeword polynomial is a multiple of $g(X)$.

Being $d(X)$ a given message word, the product $d(X) \times g(X)$ generates codewords that are usually nonsystematic. To generate systematic CRC codewords, $d(X) \times X^r$ is divided by $g(X)$, the remainder $r(X)$ is then added to the data part, i.e. $c(X) = d(X) \times X^r + r(X)$. $d(X) \times X^r$ means to append $r$ 0’s to the right of $d(X)$.

Decoding: The syndrome $s(X) = c(X) + e(X)$ is obtained by dividing the received codeword by $g(X)$. All error patterns that do not have $g(X)$ as a factor can be detected.

1. If $e(X) = X^i$, then, any $g(X)$ having 2 or more terms will detect it (Prove).

2. If $e(X) = X^i + X^j = X^i (1 + X^{j-i})$, then, it is sufficient to detect if $(1 + X^{j-i})$ can not be divided by $g(X)$, e.g. $X^{15} + X^{14} + 1$ will not divide $1 + X^k$ for $k$ up to 32768.

3. If $X + 1$ is a factor of $g(X)$, all odd number of bits errors can be detected (Prove).

4. $r$ check bits detect all bursty errors of length $\leq r$. Proof: See below.

Let $e(X) = X^{j+k-1} + \cdots + X^j$, $0 < k \leq r$ be $k$ bursty errors at the $j$th position. Then $c(X) = X^j (X^{k-1} + \cdots + 1)$, if $g_0 = 1$, $X^j$ is not a factor of $g(X)$. Recalling $r$ is the degree of $g(X)$, if $k \leq r$, $(X^{k-1} + \cdots + 1)$ is never divisible by $g(X)$.

Examples of $g(X)$:

<table>
<thead>
<tr>
<th>Code</th>
<th>Polynomial</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC-12</td>
<td>$X^{12} + X^{11} + X^3 + X^2 + X + 1$</td>
</tr>
<tr>
<td>CRC-16</td>
<td>$X^{16} + X^{15} + X^3 + X^2 + X + 1$</td>
</tr>
<tr>
<td>CRC-CCITT</td>
<td>$X^{16} + X^{12} + X^5 + 1$</td>
</tr>
<tr>
<td>CRC Ethernet, see [3]</td>
<td>$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$</td>
</tr>
</tbody>
</table>
Special classes of cyclic codes

**Bose-Chaudhuri-Hocquenghem (BCH)**

For any positive integer \( m \geq 3 \) and \( t < 2^{m-1} \), there exists a binary BCH code \( C_{BCH}(n,k) \) with the following properties:

- **Block length:** \( n = 2^m - 1 \)
- **Number of message bits:** \( k \geq n - mt \)
- **Minimum distance:** \( d_{\text{min}} \geq 2t + 1 \)
- **Error-correction capability:** \( t \) errors in a code vector

**Reed-Solomon (RS)**

Reed-Solomon codes are also called non binary since coefficients of polynomials are elements in \( \text{GF}(2^m) \). These coefficients can be represented as powers of a primitive element \( \alpha \). Recalling table (1), the following message given in non systematic form in a Reed-Solomon Code \( C_{RS}(15,k) \) results (\( k \geq 5 \)):

\[
m(X) = \alpha^4 + \alpha^3 X^2 + \alpha^9 X^4
\]

- **Block length:** \( n = 2^m - 1 \)
- **Parity-Check size:** \( n - k = 2t \)
- **Minimum distance:** \( d_{\text{min}} \geq 2t + 1 \)
- **Error-correction capability:** \( t \) errors in a code vector

Further reading: [1], [3].

**References**

