GF(2^m) architectures, BCH codes

Coding Technology

Illés Horváth

2025/10/15

Reminder: GF(2^m) arithmetics

Elements of GF(2^m) are binary polynomials of degree $\leq (m-1)$.

Addition is binary polynomial addition.

Multiplication is polynomial multiplication mod the reduction polynomial $p(y)$, where $p(y)$ is irreducible and has degree m.

The primitive element in GF(2^m) is always y.

1	1	y^0
2	y	y^1
3	$y+1$	y^2

GF(4) power table

1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

GF(8) power table

Multiplication with shift registers over GF(8)

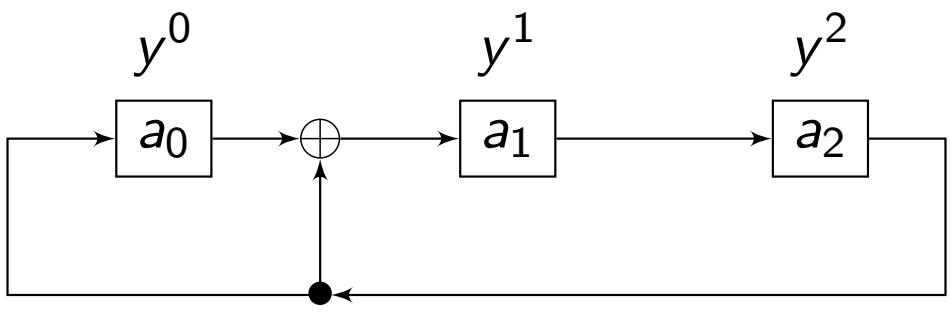
1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

Example. We want to multiply $\alpha(y) = a_0 + a_1 y + a_2 y^2$ by y (a_0, a_1 and a_2 are bits).

Preparation:

$y(a_0 + a_1 y + a_2 y^2) = a_0 y + a_1 y^2 + a_2 y^3 =$

$a_0 y + a_1 y^2 + a_2(y+1) = a_2 + (a_0 + a_2)y + a_1 y^2.$



Multiplication with shift registers over GF(8)

1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

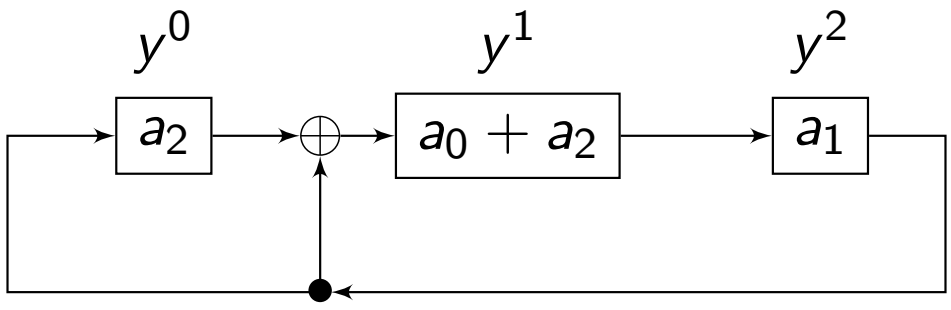
Example. We want to multiply $\alpha(y) = a_0 + a_1 y + a_2 y^2$ by y.

Preparation:

$y(a_0 + a_1 y + a_2 y^2) = a_0 y + a_1 y^2 + a_2 y^3 =$

$a_0 y + a_1 y^2 + a_2(y+1) = a_2 + (a_0 + a_2)y + a_1 y^2.$

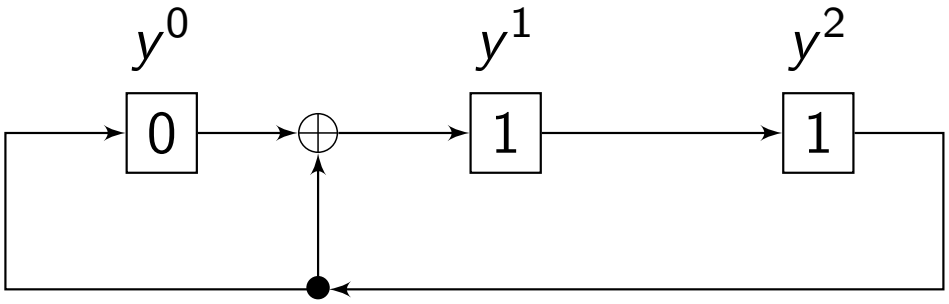
At the next time instance:



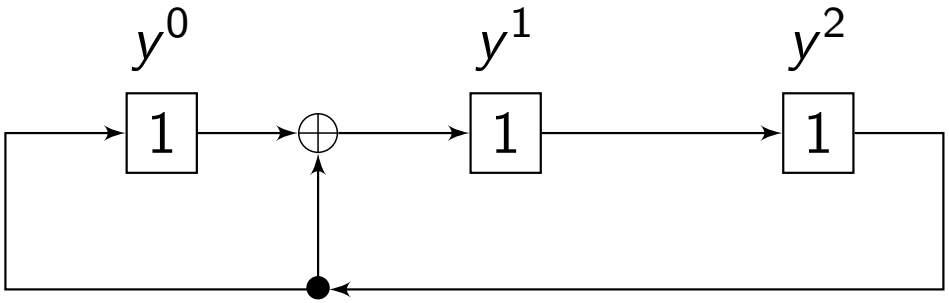
Multiplication with shift registers over GF(8)

1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

Example. We want to multiply $y + y^2$ by y.



At the next time instance:



So $(y + y^2) * y = 1 + y + y^2$.

Multiplication with shift registers over GF(8)

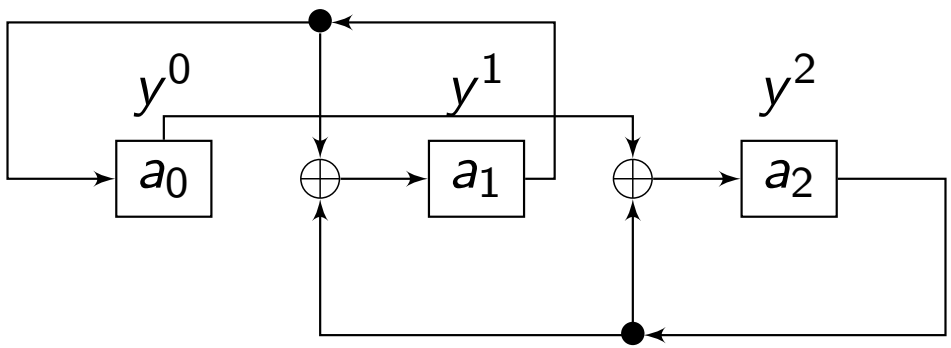
1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

Example. Multiplication by 4. ($4 = y^2$.)

$y^2(a_0 + a_1 y + a_2 y^2) = a_0 y^2 + a_1 y^3 + a_2 y^4 =$

$a_0 y^2 + a_1(y+1) + a_2(y^2+y) =$

$a_1 + (a_1 + a_2)y + (a_0 + a_2)y^2.$



Multiplication with shift registers over GF(8)

1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

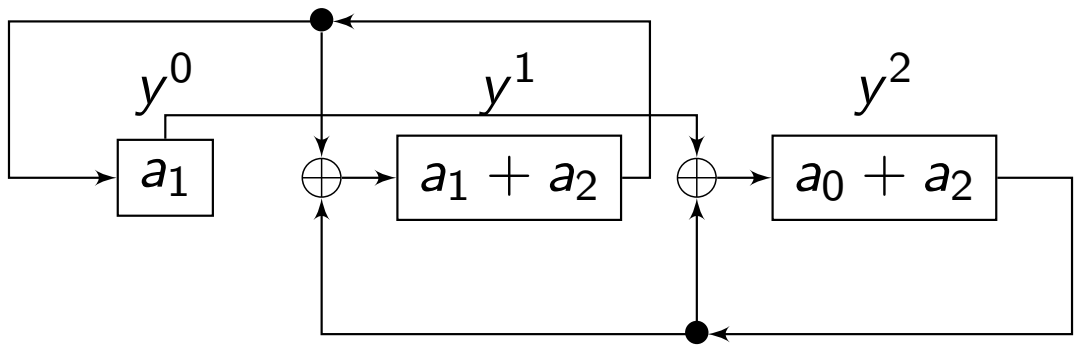
Example. Multiplication by 4. ($4 = y^2$.)

$y^2(a_0 + a_1 y + a_2 y^2) = a_0 y^2 + a_1 y^3 + a_2 y^4 =$

$a_0 y^2 + a_1(y+1) + a_2(y^2+y) =$

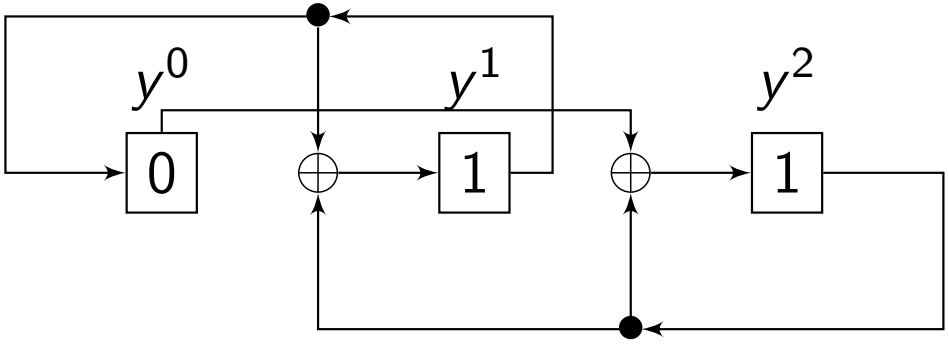
$a_1 + (a_1 + a_2)y + (a_0 + a_2)y^2.$

At the next time instance:

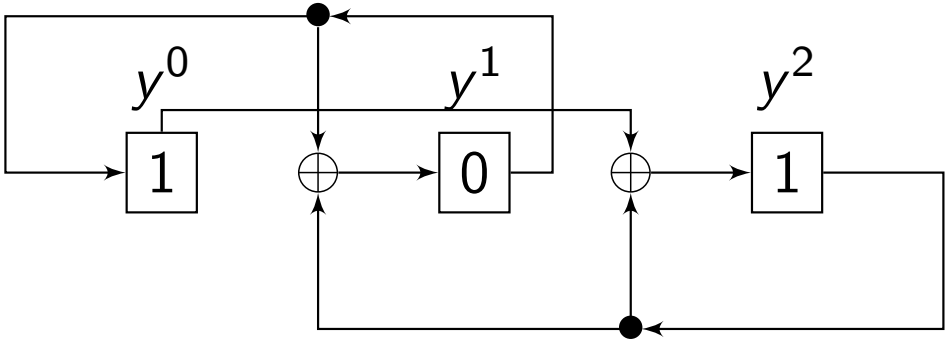


Multiplication with shift registers over GF(8)

Example. We want to compute $6 * 4$.
$(6 = y + y^2,\ 4 = y^2)$



At the next time instance:



So $(y + y^2) * y^2 = 1 + y^2$.

1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

Reed-Solomon codes over GF(2^m)

The Reed-Solomon code over GF(2^m) works exactly the same as over GF(q) when q is a prime. We assume $n = 2^m - 1$. The primitive element used for the RS code is always y.

Generator matrix:

$$G = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & y & y^2 & \dots & y^{n-1} \\ \vdots & & & \ddots & \vdots \\ 1 & y^{k-1} & y^{2(k-1)} & \dots & y^{(n-1)(k-1)} \end{bmatrix}.$$

The C(n, k) RS code over GF(2^m) can

- detect $n - k$ errors, and
- correct $\left\lfloor \frac{n-k}{2} \right\rfloor$ errors.

Reed-Solomon codes over GF(2^m)

1	1	y^0
2	y	y^1
3	$y+1$	y^3
4	y^2	y^2
5	y^2+1	y^6
6	y^2+y	y^4
7	y^2+y+1	y^5

Generator polynomial:

$$g(x) = \prod_{i=1}^{n-k} (x - y^i)$$

Remark. When working over GF(2^m), the variable y will always be used for the auxiliary field variable. Think of polynomials of y as numbers. For example, in GF(4), the numbers are $0, 1, y, y+1$.

For the variable of code polynomials, x will be used.

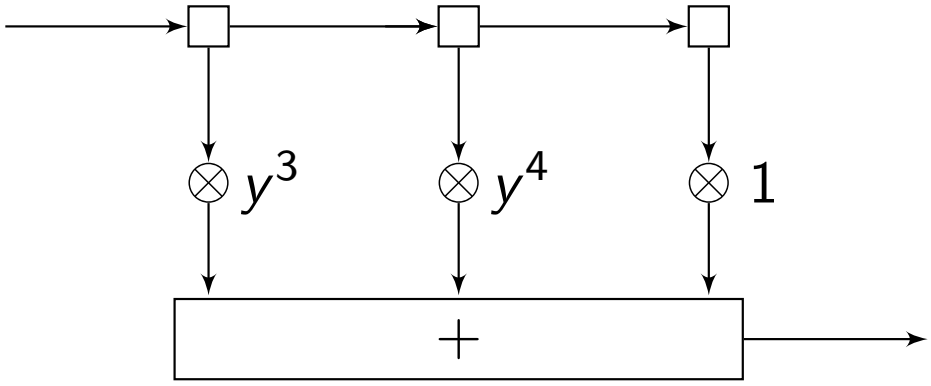
Example. The C(7,5) RS code over GF(8) has generator polynomial

$$g(x) = (x - y)(x - y^2) = (y^3 + y^4 x + x^2)$$

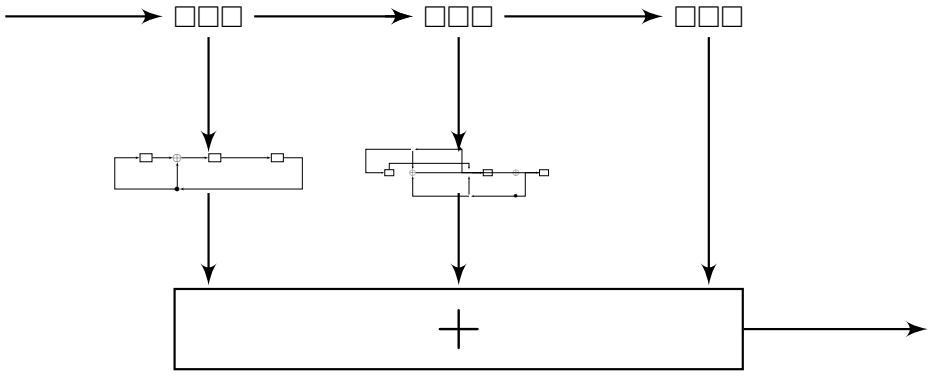
The Error locator algorithm and the Error trapping algorithm also work the same.

Reed-Solomon codes over GF(2^m)

For multiplication by $g(x) = y^3 + y^4x + x^2$ over GF(8), the architecture looks like this:



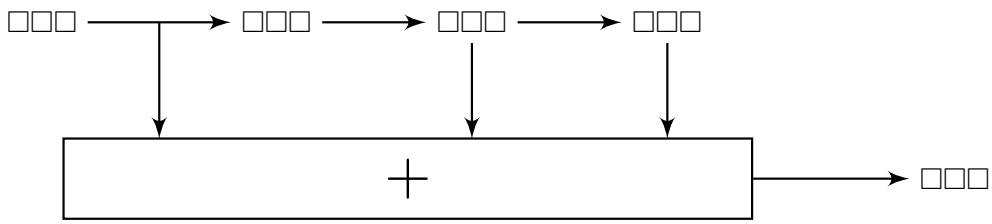
Each register represents a digit; but in GF(8), each digit is represented by 3 bits:



Motivation for BCH codes

What if we had a generator polynomial over GF(8) where all the coefficients were only 0 or 1?

Example. Design the architecture for multiplication by $g(x) = 1 + x^2 + x^3$ over GF(8).



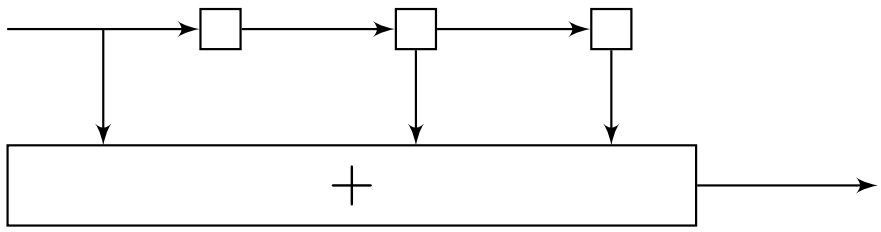
Over GF(8), $g(x) = 1 + x^2 + x^3$ can be used to generate a C(7,4) code. That means 12-bit messages and 21-bit codewords.

Notice that the architecture is doing the same operations on all 3 bits. That means that this C(7,4) code over GF(8) is effectively the same C(7,4) binary code on the 3 bits.

So why don't we just use it as a C(7,4) binary code instead?

Motivation for BCH codes

As a C(7,4) binary code, the generator polynomial is the same $g(x) = 1 + x^2 + x^3$, and the architecture is



This architecture has a very simple structure so it's also faster. This is the main motivation behind BCH codes: design codes with simple architectures, so coding and decoding is fast.

Of course, another the question is how to get good error correction capabilities. That depends on the properties of the original generator polynomial over GF(8) (or GF(2^m) in general).

The starting point will be Reed-Solomon codes.

RS codes over GF(2^m)

A C(n, k) Reed-Solomon code over GF(2^m) has generator polynomial

$$g(x) = \prod_{i=1}^{n-k} (x - y^i)$$

Originally, we designed RS codes with a generator matrix and proved the MDS property from that, but it essentially relied on the fact that the elements $y^1, \ldots, y^{n-k}$ are all roots of $g(x)$.

RS codes over GF(2^m)

Theorem

If $g(x)$ is a polynomial over GF(2^m) such that

$$\prod_{i=1}^{d-1}(x - y^i) \Big| g(x),$$

then the cyclic linear code generated by $g(x)$ has minimal codeword distance at least d.

No proof, but it's essentially the same proof that the RS codes are MDS. □

The theorem says that adding more roots to the generator polynomial $g(x)$ of the RS code will not decrease d_{min}.

RS codes over GF(2^m)

Recall that the C(7,5) RS code over GF(8) has $d_{\min} = n - k + 1 = 5$, so it can correct $\left\lfloor \frac{7-5}{2} \right\rfloor = 1$ error. It has generator polynomial

$$g(x) = (x - y)(x - y^2) = (y^3 + y^4 x + x^2)$$

Then, according to the theorem, each of

$$\begin{aligned} g'(x) &= (x - y)(x - y^2)(x - y^3) \\ g'(x) &= (x - y)(x - y^2)(x - y^4) \\ g'(x) &= (x - y)(x - y^2)(x - y^5) \\ g'(x) &= (x - y)(x - y^2)(x - y^6) \end{aligned}$$

generates a cyclic linear code with $d_{\min} \geq 3$.

RS codes over GF(2^m)

Assume we use $g'(x)$ instead of $g(x)$ to generate a code.

What do we lose?

Due to $\deg g'(x) = 3$, they each generate C(7,4) codes instead of a C(7,5) code, which has a lower code rate.

What do we win?

$$
\begin{aligned}
g'(x) &= (x-y)(x-y^2)(x-y^3) = y^6 + yx + y^6x^2 + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^4) = 1 + x + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^5) = y + y^5x + x^2 + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^6) = y^2 + y^3x^2 + x^3
\end{aligned}
$$

RS codes over GF(2^m)

Assume we use $g'(x)$ instead of $g(x)$ to generate a code.

What do we lose?

Due to $\deg g'(x) = 3$, they each generate C(7,4) codes instead of a C(7,5) code, which has a lower code rate.

What do we win?

$$
\begin{aligned}
g'(x) &= (x-y)(x-y^2)(x-y^3) = y^6 + yx + y^6x^2 + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^4) = 1 + x + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^5) = y + y^5x + x^2 + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^6) = y^2 + y^3x^2 + x^3
\end{aligned}
$$

RS codes over GF(2^m)

Assume we use $g'(x)$ instead of $g(x)$ to generate a code.

What do we lose?

Due to $\deg g'(x) = 3$, they each generate C(7,4) codes instead of a C(7,5) code, which has a lower code rate.

What do we win?

$$
\begin{aligned}
g'(x) &= (x-y)(x-y^2)(x-y^3) = y^6 + yx + y^6x^2 + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^4) = 1 + x + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^5) = y + y^5x + x^2 + x^3 \\
g'(x) &= (x-y)(x-y^2)(x-y^6) = y^2 + y^3x^2 + x^3
\end{aligned}
$$

How can we find simple $g'(x)$ polynomials in general?

Irreducible polynomials

Is the polynomial $x^2 + 1$ irreducible? (No finite fields involved, just regular real numbers and polynomials.)

That depends. Over the field $\mathbb{R}$ (the real numbers), $x^2 + 1$ is irreducible.

However, over the larger field $\mathbb{C}$ (the complex numbers),

$$x^2 + 1 = (x + i)(x - i).$$

So 'irreducible' is relative to what field we are considering.

(Another example would be $x^2 - 2$ over $\mathbb{Q}$ or $\mathbb{R}$.)

Irreducible polynomials

y^0	1
y^1	y
y^2	$y+1$

Something similar happens for polynomials over GF(2) and GF(2^m).

Example. The polynomial $1 + x + x^2$ is irreducible over GF(2), but over GF(4),

$$1 + x + x^2 = (x - y)(x - y^2).$$

We say that the minimal polynomial of the element $y \in$ GF(4) is $1 + x + x^2$ because it is the minimal degree binary polynomial (so only 0 and 1 coefficients) that y is a root of.

Irreducible polynomials

Reminder: in GF(2^m),

$$a(y)^{2^m-1} = 1$$

for every nonzero element $a(y)$.

This means that the roots of the polynomial $x^{2^m-1} - 1$ are all nonzero elements of GF(2^m), that is,

$$x^{2^m-1} - 1 = (x-1)(x-y)(x-y^2)\dots(x-y^{2^m-2}).$$

But $x^{2^m} - 1$ can also be considered as a polynomial over GF(2) since it has only 0 and 1 coefficients. Over GF(2), it can be decomposed as the product of irreducible binary polynomials:

$$x^n - 1 = p_1(x)p_2(x)\dots p_L(x).$$

Conjugate groups and minimal polynomials

Each $p_\ell(x)$ ($\ell = 1, \ldots, L$) is a polynomial that is irreducible over GF(2), but has roots over GF(2^m).

We group the nonzero elements of GF(2^m) according to the $p_\ell(x)$'s. These groups are called the conjugate groups.

Example. For GF(4), we have

$$x^3 - 1 = (x-1)\underbrace{(1+x+x^2)}_{(x-y)(x-y^2)}$$

So the conjugate groups and corresponding minimal polynomials of GF(4) are

$$\{1\} \to x - 1$$
$$\{y, y^2\} \to 1 + x + x^2$$

Conjugate groups and minimal polynomials

Example. For GF(8), we have

$$x^7 - 1 = (x-1)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) =$$
$$= (x-1) \cdot \underbrace{(x^3 + x + 1)}_{(x-y)(x-y^2)(x-y^4)} \cdot \underbrace{(x^3 + x^2 + 1)}_{(x-y^3)(x-y^5)(x-y^6)}$$

So the conjugate groups and corresponding minimal polynomials of GF(8) are

$$\{1\} \to x - 1$$
$$\{y, y^2, y^4\} \to x^3 + x + 1$$
$$\{y^3, y^5, y^6\} \to x^3 + x^2 + 1$$

Conjugate groups and minimal polynomials

Example. For GF(16) with reduction polynomial $p(y) = y^4 + y + 1$, the conjugate groups and corresponding minimal polynomials are

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

Minimal polynomials

General definition. For any nonzero element $\alpha \in \mathrm{GF}(2^m)$, its minimal polynomial is the binary polynomial $M(x)$ such that $M(\alpha) = 0$.

Theorem (Properties of minimal polynomials)

(a) *For each $\alpha \in GF(2^m)$, $M(x)$ is unique (and denoted by $M_\alpha(x)$).*

(b) *$M_\alpha(x)$ is an irreducible binary polynomial.*

(c) $M_\alpha(x) | x^{2^m-1} - 1$

(d) $\deg M_\alpha(x) \leq m$ *(so the size of each conjugate group is $\leq m$)*

(e) *α is a primitive element* $\iff$ $\deg M_\alpha(x) = m$.

(f) *α and α^2 have the same minimal polynomial.*

No proof. (But they are not difficult.)

BCH codes

A C(n, k) BCH code is a binary linear cyclic code with generator polynomial $g(x)$ that has the following properties:

- $n = 2^m - 1$
- The roots of $g(x)$ over GF(2^m) include $y^1, y^2, \ldots, y^{2t}$.
- The roots of $g(x)$ contain entire conjugate groups; $g(x)$ is the product of the corresponding minimal polynomials.
- k is not specified in advance, and depends on t.
- The code can correct t errors.

BCH codes

Example. We want a BCH code that can correct $t = 1$ error. Start from the conjugate groups and corresponding minimal polynomials of GF(4):

$$\{1\} \to x - 1$$
$$\{y, y^2\} \to 1 + x + x^2$$

$y^1, \ldots, y^{2t}$ all need to be included among the roots of $g(x)$.

For $t = 1$, this means we need y^1 and y^2. They are in the same conjugate group with minimal polynomial, so

$$g(x) = 1 + x + x^2.$$

We started out from GF(4), so $n = 4 - 1 = 3$.
$n - k = \deg g(x) = 2$, so $k = 1$.

This is a C(3,1) binary linear cyclic code that can correct 1 error. Do we know it from before?

BCH codes

Example. Starting from GF(8), we can obtain C(n, k) codes with $n = 8 - 1 = 7$. Design a code that can correct 1 error.

$$\{1\} \to x - 1$$
$$\{y, y^2, y^4\} \to x^3 + x + 1$$
$$\{y^3, y^5, y^6\} \to x^3 + x^2 + 1$$

We need to include $y^1, \ldots, y^{2t}$ among the roots, so that's y^1 and y^2 in this case.

y^1 and y^2 are in the same group with minimal polynomial $x^3 + x + 1$, so

$$g(x) = x^3 + x + 1.$$

Finally, $n - k = \deg g(x) = 3$, so $k = 4$. This is a C(7,4) binary code that can correct $t = 1$ error.

Have we seen this code before?

BCH codes

Theorem

For every $m \geq 2$, the BCH code obtained from GF(2^m) that can correct $t = 1$ error is equivalent to the C($2^m - 1, 2^m - m - 1$) Hamming code.

No proof.

BCH codes are a very broad class; they include the Hamming codes, but also other codes with lower code rate but better error correction capability.

Compared to RS codes, BCH codes can be coded and decoded faster (no need for sub shift register architectures for digits of GF(2^m)), at the cost of a lower code rate.

For decoding, the Error locator algorithm (discussed for RS codes earlier) can be adapted - details omitted.

BCH codes

Example. Starting from GF(16), obtain a code that can correct 3 errors.

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

$y^1, \ldots, y^6$ have to be included, so

$$\begin{aligned}
g(x) &= (x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1) = \\
&= x^{10}+x^8+x^5+x^4+x^2+x+1
\end{aligned}$$

$n = 16-1 = 15$, and $n-k = \deg(g(x)) = 10$, so $k = 5$.

This C(15,5) BCH code is used in the format part of the QR code.

BCH codes summary

BCH codes are a broad class of binary linear cyclic codes.

Their main property is that they can be coded and decoded efficiently, which makes them relevant in many industrial applications where the main bottleneck is not channel capacity, but computational complexity of coding and decoding.

The error correction capability of BCH codes is scalable; BCH codes can be designed to correct a desired amount of errors.

BCH codes are sometimes punctured to fit codeword length to specific applications; we do not pursue this direction.