6. Lecture

Mittwoch, 27. September 2023 10:04

$c'(x) = x \cdot c(x) \bmod (x^n - 1)$

Cyclic linear codes $C(n,k)$

If $c(x) \in C$ (code word) $\rightarrow c'(x) = S c(x) = x \cdot c(x) \bmod (x^n-1) \in C$ (also code word)
If $c(x), c'(x) \in C \rightarrow \alpha\, c(x) + \beta\, c'(x) \in C$ (linear combination)
If $g(x)$ generator polynom

$g(x)$
- $\deg(g(x)) = n-k$
- $g_{n-k} = 1$
- $\forall\, c(x): \; c(x) = u(x)\, g(x)$ ← message polynom
- $g(x) \mid x^n - 1$

Encoder: $u(x) \rightarrow$ LFBSR $g(x) \rightarrow c(x)$

$g(x) = b(x)\,(x^n - 1)$

PROOF (not exam)

$a(x)$: $\deg(a(x)) < \deg(c(x)) \;\; \forall\, c(x) \in C \quad \deg(a(x)) = m$

$g(x) = a_m^{-1}\, a(x) = a_m^{-1} a_0 + a_m^{-1} a_1 x + \dots + \underbrace{a_m^{-1} a_m}_{1} x^m \in C$

$\nexists\, g'(x)$: $g(x) - g'(x)$: $\deg(g(x) - g'(x)) \le m$ (smallest above) also impossible

$g(x) - g'(x) \in C$

$u_0 g(x) + u_1 x\, g(x) + \dots + u_{n-m-1} x^{n-m-1} g(x) \in C$

$\left(u_0 + u_1 x + \dots + u_{n-m-1} x^{n-m-1}\right) g(x) = c(x)$

$u(x) \cdot g(x) = c(x)$

$\nexists\, c(x) = u(x)\, g(x) + r(x) \rightarrow c(x) - u(x) g(x) = r(x) \in C$

$c(x) \in C$, $u(x) g(x) \in C$ — linear combination

$\deg(r(x)) < m$ ↯ smallest one

$\forall\, c(x)$: $c(x) = u(x)\, g(x) \qquad n - m - 1 = k - 1$

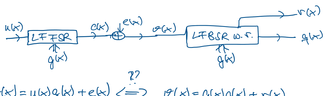
$m = \deg(g(x)) = n - k \;\checkmark$

$g_{n-k} = 1 \;\checkmark$

$\forall\, c(x)$: $c(x) = u(x)\, g(x)$ ⇒ LFBSR $g(x) \rightarrow c(x)$

$g(x) \mid x^n - 1$

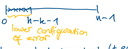
(Background not worthy for midterm, nur Tabelle auf receiver seite)



$v(x) = u(x) g(x) + e(x) \;\Longleftrightarrow\; v(x) = q(x) g(x) + r(x)$??

$v(x)$ observable, $u(x), e(x)$ unknown, $q(x), r(x)$ comp.

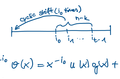
$u(x) = q(x)$, $e(x) = r(x)$ only true if $\deg(r(x)) < n-k$ and $\deg(e(x)) < n-k$



lowest configuration of error!

$e(x) = e_{i_1} x^{i_1} + e_{i_2} x^{i_2} + \dots + e_{i_{t-1}} x^{i_{t-1}}$ (t errors)

$i_{t-1} - i_0 < n-k$

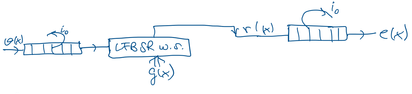


$x^{-i_0} v(x) = x^{-i_0} u(x) g(x) + x^{-i_0} e(x)$

$x^{-i_0} v(x) = q(x) g(x) + r'(x)$

$r'(x) = x^{-i_0} e(x)$
$e(x) = x^{i_0} r'(x)$

⇒



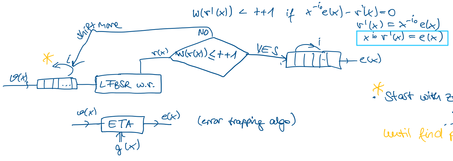
i_0?

$x^{-i_0} e(x) = b'(x) g(x) + r'(x)$ (same)
$x^{-i_0} v(x) = q'(x) g(x) + r'(x)$

$x^{-i_0} e(x) = b'(x) g(x) + r'(x)$
$x^{-i_0} e(x) - r'(x) = b'(x) g(x)$

$w(x^{-i_0} e(x)) \ne t$; $w(r'(x)) \ge t+1$; $\in C$ (code word) ; $w(b'(x) g(x)) \ge 2t+1$

$w(r'(x)) < t+1$ if $x^{-i_0} e(x) - r'(x) = 0$
$r'(x) = x^{-i_0} e(x)$
$x^{i_0} r'(x) = e(x)$



(error trapping algo)

* Start with zero shift
* until find perfect i so class $w(r(x)) \le t+1$

Full implementation of linear cyclic codes

$c(x) = u(x) g(x) \Rightarrow u(x) = \frac{c(x)}{g(x)}$

⇒ shift registers no LUT : real-time ✓ fast polynom complexity

!! Reed-Solomon codes are cyclic !!

MDS codes — Shift register based code

Given any number of errors to be corrected "t" ⇓ real-time

$c(x)\Big|_{x = \alpha^i,\, i = 1, \dots, n-k} = 0$ $\qquad c(x) = \prod_{i=1}^{n-k} (x - \alpha^i)\, u(x)$

$g(x)$

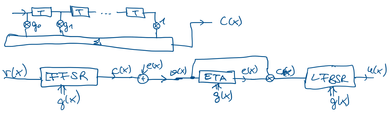
$c(x) = g(x)\, u(x)$

Code design

Given "t" # of errors to be corrected

$n - k = 2t$, $n = q-1$, q - prime $\rightarrow$ $C(n,k)$ over $GF(q)$ $\quad x \in GF(q)$

$g(x) = \prod_{i=1}^{n-k} (x - \alpha^i) = g_0 + g_1 x + \dots + 1 x^{n-k}$

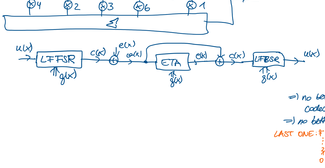


Bsp.: $t = 2 \Rightarrow n - k = 4 \quad n = q-1$, q prime

q	n	k
2	1	—
3	2	—
5	4	0
7	6	2
11	10	6

⇒ $C(6,2)$ over $GF(7)$; primitive element $3 \in GF(7)$ always given in exam

$g(x) = \prod_{i=1}^{n-k} (x - \alpha^i) = \prod_{i=1}^{4} (x - 3^i) = (x-3)(x-2)(x-6)(x-4)$
$= (x+4)(x+5)(x+1)(x+3)$
$= (x^2 + 2x + 6)(x^2 + 4x + 3)$
$= x^4 + 2x^3 + 6x^2 + 4x^3 + 1x^2 + 3x + 3x^2 + 6x + 4$
$= x^4 + 6x^3 + 3x^2 + 2x + 4$



⇒ no Golay codes
⇒ no Golay implementation

LAST ONE !!

for away von Grenzen ⇒ robuste gegen noise

Algebra over $GF(2^n)$??