Decodierverfahren für negazyklische Codes

vgl. Folie 10, 14.11.2014

Fehlerstellenpolynom: $\sigma(z) = \prod (1 - X_i z)$

Syndrompolynom: $S(z) = \sum_{j=1}^{\infty} S_j z^j$ ← ohne z im Nenner!

$$\Rightarrow \quad \sigma'(z) = -\sum_i X_i \prod_{j \neq i} (1 - X_j z)$$

$$\Rightarrow \quad -\frac{z \cdot \sigma'(z)}{\sigma(z)} = \sum_i \frac{X_i z}{1 - X_i z} = \sum_i \sum_{k=1}^{\infty} (X_i z)^k$$

$$= \sum_{k=1}^{\infty} \left(\sum_i X_i^k \right) z^k = \sum_{k=1}^{\infty} S_k z^k = S(z)$$

$$\Rightarrow \quad \boxed{S(z) \cdot \sigma(z) + z \cdot \sigma'(z) = 0}$$

Unterschied zu BCH-Codes:

$S(z)$ ist nur für ungerade S_j bekannt, $j = 1, 3, \ldots, 2t-1$

$\sigma(z)$ kann mehrfache Nullstellen haben.

kein Fehlerwertpolynom. Fehlerwerte = 1

Minuszeichen durch Exponent von NS.

Lösen von $S \cdot G + z G' = 0$

Bekannt: $S_1, S_3, S_5, \dots, S_{2d-1}$ von $S(z) = S_1 z + S_2 z^2 + \dots$

Aufteilen der Polynome in geraden und ungeraden Anteil:

$$G(z) = 1 + G_1 z + G_2 z^2 + \dots$$

ger. Anteil: $\hat{G}(z) = 1 + G_2 z^2 + G_4 z^4 + \dots$

unger. Anteil: $\overset{s}{G}(z) = G_1 z + G_3 z^3 + \dots$

$$\hat{S}(z) = S_2 z^2 + S_4 z^4 + \dots$$

$$\overset{s}{S}(z) = S_1 z + S_3 z^3 + \dots \quad \leftarrow \text{bekannt (t Werte)}$$

Einsetzen in $S \cdot G + z G' = 0$ und Trennen der geraden und ungeraden Polynome:

$$\left(\hat{S} + \overset{s}{S}\right)\left(\hat{G} + \overset{s}{G}\right) + z\left(\hat{G}' + \overset{s}{G}'\right) = 0$$

$$\Rightarrow \quad \hat{S}\hat{G} + \overset{s}{S}\overset{s}{G} + z\hat{G}' = 0 \qquad / \cdot \overset{s}{G}$$

$$\hat{S}\overset{s}{G} + \overset{s}{S}\hat{G} + z\overset{s}{G}' = 0 \qquad / \cdot \hat{G} \qquad -$$

$$\overset{s}{S}\left(\overset{s}{G}^2 - \hat{G}^2\right) + z\left(\overset{s}{G}\hat{G}' - \hat{G}\overset{s}{G}'\right) = 0$$

$$\Rightarrow \quad \overset{s}{S}\left(\frac{\overset{s}{G}^2}{\hat{G}^2}-1\right) = z\,\frac{\hat{G}\hat{G}' - \overset{s}{G}\hat{G}'}{\hat{G}^2}$$

$$= z\left(\frac{\overset{s}{G}}{\hat{G}}\right)'$$

Generatorfunktion: $U(z) = \frac{\overset{s}{G}(z)}{\hat{G}(z)}$

$$\overset{s}{S}\cdot(U^2-1) = z\cdot U'$$

$$\Rightarrow \quad U = \int \frac{\overset{s}{S}}{z}(-1+U^2)\,dz$$

↑

Sieht kompliziert aus, ist aber einfach zu lösen.

Die Koeffizienten von $U(z)$ lassen sich bis zum Koeffizienten U_{p-2} bestimmen

$$\Rightarrow \quad U_1 = -S_1$$

$$U_3 = \frac{-S_3 + U_1^2 S_1}{3}$$

$$U_5 = \frac{-S_5 + U_1^2 S_3 + 2U_1 U_3 S_1}{5}$$

$\vdots$ $\qquad$ $\vdots$

deshalb NS bis α^{2t-1} $2t-1 < p$

Problem bei Division durch p!

$U(z)$ ist eine ungerade Funktion von z.

$\Rightarrow$ Übergang zu $\quad T(z^2) = \dfrac{1}{1+zU(z)} - 1$

$$\Rightarrow 1 + T(z^2) = \frac{1}{1 + z\frac{\hat{G}^S}{\hat{G}}} = \frac{\hat{G}(z)}{\hat{G} + z\hat{G}^S}$$

Setze: $\quad \omega(z^2) = \hat{G}(z)$

$$\phi(z^2) = \hat{G}(z) + z\hat{G}^S(z)$$

$$\Rightarrow \quad 1 + T(z) = \frac{\omega(z)}{\phi(z)}$$

↑ t Koeffizienten sind bekannt

$$\Rightarrow \quad 1 + T(z) \equiv \frac{\omega(z)}{\phi(z)} \mod z^{t+1}$$

$$\Rightarrow \quad \boxed{\omega(z) \equiv \phi(z) \cdot (1 + T(z)) \mod z^{t+1}} \quad (*)$$

Wenn weniger als t Fehler aufgetreten sind

$$\Rightarrow \text{Grad } \phi \leq \frac{t+1}{2} \; ; \; \text{Grad } \omega \leq \frac{t}{2}$$

und Gleichung $(*)$ kann mit dem Euklidschen Algorithmus gelöst werden. $\Rightarrow$ ω und ϕ $\Rightarrow \hat{G}(z)$

Die Nullstellen von $\sigma(z)$ liefern die Fehlerpositionen.

Fehlerwerte $\neq 1$ oder -1 werden durch mehrfache NS von $\sigma(z)$ erhalten.

z.B. Fehler 2 an Position i mit 2-facher NS von $\sigma(z)$ bei $1/\alpha^i$ oder Fehlerwert -3 durch 3-fache NS von $\sigma(z)$ bei $1/\alpha^{n+i}$.

H.A.: $GF(11)$; $g(x) = (x-2)(x-2^3)$

ergibt einen $[5, 3, 5]$ Code der 2 Fehler in der Lee-Metrik korrigieren kann.

Decodieren Sie: $r(x) = -x + x^3$

$r(x) = 2x^2$

$r(x) = -2x$

Roths Algorithmus zur Decodierung von negazyklischen Codes

- Roths Algorithmus (2006) vereinfacht die Decodierung.
- wird seit WS 2013/14 in Vorlesung behandelt.

ursprüngliche Herleitung mittels alternierender Codes (engl. alternant Codes)
↖ kein Vorlesungsstoff

Jetzt: Herleitung mittels "klassischem Ansatz" Folie 2

Es gilt: $S(z) \cdot \left(\overset{s}{G}(z)^2 - \hat{G}(z)^2 \right) + z \left(\overset{s}{G}(z) \hat{G}'(z) - \hat{G}(z) \overset{s}{G}'(z) \right) = 0$

$$\left.\begin{aligned} G(z) &= \hat{G}(z) + \overset{s}{G}(z) \\ G(-z) &= \hat{G}(z) - \overset{s}{G}(z) \end{aligned}\right\} \Rightarrow \begin{aligned} \hat{G}(z) &= \frac{G(z) + G(-z)}{2} \\ \overset{s}{G}(z) &= \frac{G(z) - G(-z)}{2} \end{aligned}$$

$$\Rightarrow -\overset{s}{S}(z) G(z) G(-z) + z \left(\frac{G(z) - G(-z)}{2} \cdot \frac{G'(z) - G'(-z)}{2} - \frac{G(z) + G(-z)}{2} \cdot \frac{G'(z) + G'(-z)}{2} \right)$$

$$\Rightarrow \overset{s}{S}(z) G(z) G(-z) = \frac{z}{4} \Big(\underline{G(z)G'(z) - G(z)G'(-z) - G(-z)G'(z) + G(-z)G'(-z) = 0} - G(z)G'(z) - G(z)G'(-z) - G(-z)G'(z) - G(-z)G'(-z) \Big)$$

$$\searrow \quad 2 \overset{s}{S}(z) G(z) G(-z) = z \left(-G(z)G'(-z) - G(-z)G'(z) \right)$$

$$\Rightarrow -2 \overset{s}{S}(z) \cdot \frac{G(z)}{G(-z)} = z \cdot \frac{G(z)G'(-z) + G(-z)G'(z)}{G^2(-z)}$$

Generatorfunktion: $V(z) = \frac{G(z)}{G(-z)}$; $V(0) = 1$

$$\Rightarrow V'(z) = \frac{G'(z)G(-z) + G(z)G'(-z)}{G(-z)^2}$$

$$\Rightarrow \underline{-2 \overset{s}{S}(z) \cdot V(z) = z \cdot V'(z)}$$

$V(z)$ kann leicht mod z^{2t+1} bestimmt werden.

formal: $V = \int -\frac{2 \overset{u}{S}(z) V(z)}{z} dz$

Mit $V(z) = \frac{\sigma(z)}{\sigma(-z)}$ folgt neue Schlüsselgleichung:

$$\sigma(z) \equiv \sigma(-z) \cdot V(z) \mod z^{2t+1}$$

↘ kann mit Euklidschem Algorithmus gelöst werden:

Starten des Euklidschen Algorithmus mit z^{2t+1} und $V(z)$ und Durchführen bis Polynom $r_k(z)$ bestimmt ist für das gilt $1 \leq \text{Grad } r_k(z) \leq t$. Dann $\sigma(z) = r_k(z)$.

Bei Durchführen des erweiterten Euklidschen Algorithmus gilt: $P_k(z) = r_k(-z)$ ← siehe Folie 14 vom 15.11.2013

Mit $-2 \overset{\cup}{S}(z) \cdot V(z) = z \cdot V'(z)$ ← $V'(z) = V_1 + 2V_2 z + \ldots$

$\overset{\cup}{S}(z) = S_1 z + S_3 z^3 + \ldots$ $\quad V(z) = 1 + V_1 z + V_2 z^2 + \ldots$

folgen die V_i sukzessive:

$$V_1 = -2S_1$$

$$V_2 = -S_1 V_1 = 2S_1^2$$

$$V_3 = \frac{-2S_1V_2 - 2S_3}{3} = \frac{-4S_1^3 - 2S_3}{3}$$

$$V_4 = \frac{-2S_1V_3 - 2S_3V_1}{4} = \frac{2S_1^4 + 4S_1S_3}{3}$$

$\ddots$ etc.

Beispiel: Code von Aufgabe 24 $[5,3,5]$ Code über $GF(11)$, $t=2$

$S_1 = 6 \qquad S_3 = 9$

$\Rightarrow V_1 = -2S_1 = -12 \equiv 10 \bmod 11$

$V_2 = -S_1 V_1 \equiv 6 \bmod 11$

$V_3 = \frac{-2S_1V_2 - 2S_3}{3} = \frac{-2\cdot 6\cdot 6 - 18}{3} \equiv 3 \bmod 11$

$V_4 = \frac{-2S_1V_3 - 2S_3V_1}{4} = \frac{-2\cdot 6\cdot 3 - 2\cdot 9\cdot 10}{4} \equiv 1 \bmod 11$

$\Rightarrow V(z) = 1 + 10z + 6z^2 + 3z^3 + z^4$

Euklidischer Algorithmus: $V(z)$

$z^5 = (z+8)\cdot(z^4 + 3z^3 + 6z^2 + 10z + 1) + 3z^3 + 8z^2 + 7z + 3$

$V(z) = (4z+5)(3z^3 + 8z^2 + 7z + 3) + \underbrace{4z^2 + 7z + 8}_{\text{Grad} = 2 = t}$

$\Rightarrow G(z) = 4z^2 + 7z + 8$

$\equiv 8\cdot \underbrace{(6z^2 + 5z + 1)}$

↳ siehe $G(z)$ von Aufgabe 24

Weitere Codes für die Lee-Metrik

Roth / Siegel : Decodierverfahren für BCH-Codes (1994)

Prinzip : (stark vereinfacht)

$$-S(z) = \sum_{i=1}^{t} \frac{e_i}{z - \alpha_i} \mod z^s$$

↑ BCH-Code als Goppa Code dargestellt

$$\Rightarrow -S(z) = \sum_{e_i = 1} \frac{1}{z - \alpha_i} - \sum_{e_j = -1} \frac{1}{z - \alpha_j} \mod z^s$$

$$-S(z) = \frac{\sigma_p'}{\sigma_p} - \frac{\sigma_\mu'}{\sigma_\mu}$$

$$-S(z) \underbrace{\frac{\sigma_p}{\sigma_\mu}}_{\varphi(z)} = \underbrace{\frac{\sigma_p' \sigma_\mu - \sigma_\mu' \sigma_p}{\sigma_\mu^2}}_{\varphi'(z)} \mod z^s$$

$\Rightarrow$ Koeffizienten von $\varphi(z)$ folgen

$$\Rightarrow \varphi(z) \cdot \sigma_\mu(z) = \sigma_p(z) \mod z^s$$

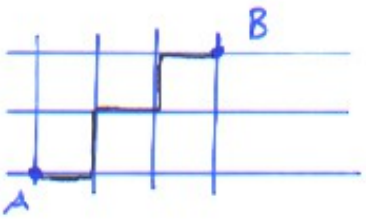
$\Rightarrow$ σ_μ und σ_p mit EEA

Details siehe (Roth/Siegel).

Roth (2006) : Algorithmische Vereinfachung des Decodierverfahrens für Negazyklische Codes.

Mannheim Metrik

= Manhattan- bzw. Taxi metrik modulo eines zweidimensionalen Bereichs.



← Manhattan-Distanz von A und B = 5

3 in x- 2 in y-Richt.

Darstellung von endlichen Körpern als Punkte in der Ebene

zunächst $GF(p)$ mit $p \equiv 1 \bmod 4$ → 5, 13, 17, 29, 37, 41, ...

⇒ mit Satz von Fermat: $p = a^2 + b^2$

Statt ganze Zahlen → Gaußsche ganze Zahlen

dies sind komplexe Zahlen, die ganze Zahlen als Real- und Imaginärteil haben.

Beispiel: $p = 5 = 2^2 + 1^2$

$p = 13 = 3^2 + 2^2$

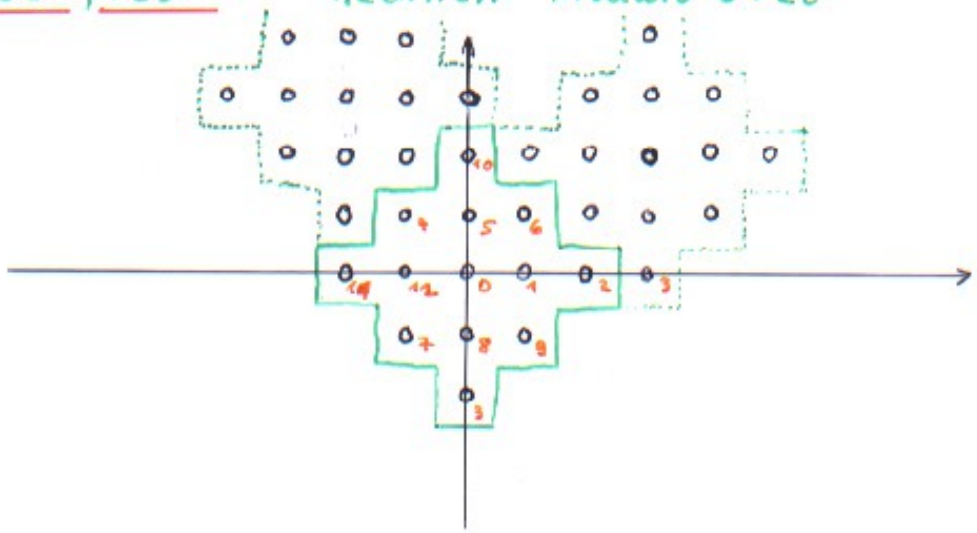
$p = 17 = 4^2 + 1^2$

mit $i^2 = -1 \Rightarrow p = a^2 + b^2 = (a+ib)\cdot(a-ib)$

Statt modulo p zu Rechnen

$\Rightarrow$ modulo $a+ib$ Rechnen!

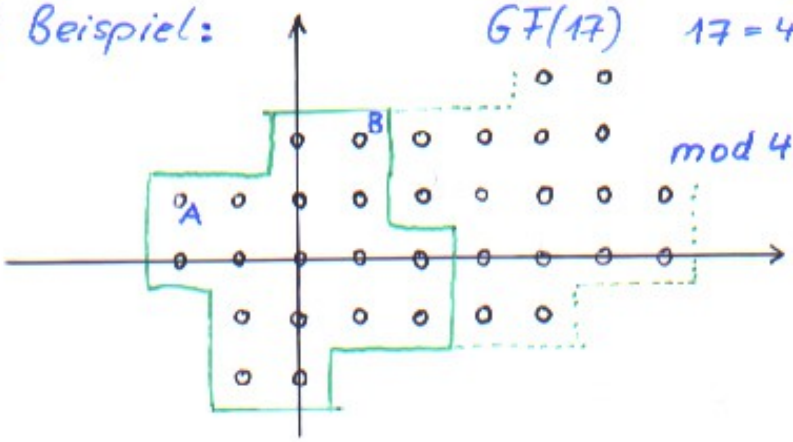
Beispiel: Rechnen modulo $3+2i$



$3 \equiv -2i \mod 3+2i$

→ Mannheim Distanz

anhand Beispiel: $GF(17)$ $17 = 4^2 + 1^2$



Bekannt: Manhattan Distanz zwischen A und B

→ 3 + 1 = 4

Mannheim Distanz: Manhattan Distanz mod 4+i

$A = -2 + i \equiv 2 + 2i$

$B = 1 + 2i$

$A - B \bmod 4+i$

$= 1$

⇒ Mannheimdistanz = 1

Durch das Rechnen modulo $(a+ib)$ ist die Mannheim Metrik algebraisch nutzbar. Die "Rand punkte" spielen keine Sonderrolle.

i-zyklische Codes für die Mannheim Metrik

Ein Fehler-Fall

Blocklänge n : $\rightarrow$ $4n$ mögliche Fehler

$\pm 1, \pm i$ an jeder Stelle $\rightarrow$ $4n+1$ Syndrome

↑ cw

Kugelpackung : $\underbrace{M}_{=p^k \text{ bei linearem Code}} \cdot (4n+1) \leq p^n$

$$n \leq \frac{p^{n-k}-1}{4}$$

Analog Lee-Metrik Codes

$\alpha^j \rightarrow$ Fehler $+1$ an Stelle j $(j=0,1,\dots n-1)$

$i\,\alpha^j \rightarrow$ Fehler $+i$ an Stelle j

$-\alpha^j \rightarrow$ Fehler -1 an Stelle j

$-i\,\alpha^j \rightarrow$ Fehler $-i$ an Stelle j

α : primitives Element in $GF(p^m)$

$\alpha^{p^m-1} = 1$

mit $n = \frac{p^m-1}{4} \Rightarrow \alpha^n = i$

$\alpha^{\frac{p^m-1}{2}} = -1$

$\alpha^{\frac{p^m-1}{4}} = i$

$\Rightarrow$ $i\,\alpha^j = \alpha^{n+j}$

$-\alpha^j = \alpha^{2n+j}$

$-i\,\alpha^j = \alpha^{3n+j}$

1- Fehlerkorrektur

Die Prüfmatrix $\underline{H}$ mit $\underline{H} = (\alpha^0, \alpha^1, \ldots, \alpha^{n-1})$

wobei α ein primitives Element von $GF(p^m)$ ist

und $n = \frac{p^m - 1}{4}$ erzeugt einen $[n, n-m, d_M = 3]$ Code

über $GF(p)$. (d_M: Mannheim dist.)

Beweis: Ein Fehler mit Wert $e \in \{1, i, -1, -i\}$

an der Stelle j erzeugt das Syndrom

$$S = e\,\alpha^j = \begin{cases} \alpha^j & \text{für } e = 1 \\ \alpha^{n+j} & " \quad e = i \\ \alpha^{2n+j} & " \quad e = -1 \\ \alpha^{3n+j} & " \quad e = -i \end{cases}$$

Jeder Fehler erzeugt ein verschiedenes Syndrom.

Die ein-Fehler korrigierenden Codes

für die Mannheim Metrik sind perfekt.

↳ folgt sofort mit Kugelpackungsgleichung.

i-zyklische Codes

Def.:

$$\underline{H} = \begin{pmatrix} \alpha^0 & \alpha^1 & \alpha^2 & \cdots & \alpha^{n-1} \\ \alpha^0 & \alpha^5 & \alpha^{10} & \cdots & \alpha^{(n-1)5} \\ \alpha^0 & \alpha^9 & \alpha^{18} & \cdots & \alpha^{(n-1)9} \\ \vdots & \vdots & & & \vdots \\ \alpha^0 & \alpha^{4t+1} & & \cdots & \alpha^{(n-1)(4t-1)} \end{pmatrix}$$

$\alpha^{4n} = 1$

$\alpha^n = i$

$\Rightarrow$ Codeworte $c(\alpha^i) = 0 \qquad i = 1, 5, 9, \ldots 4t+1$

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

Die Codewörter eines izyklischen Codes sind Vielfache eines Generatorpolynoms $g(x)$ welches das Polynom $(x^n - i)$ teilt.

$\rightarrow$ Analog Lee-Fall: $x^{4n} - 1 = (x^n - 1)(x^n - i)(x^n + 1)(x^n + i)$

$$\underline{c} = (c_0, c_1, c_2, \ldots, c_{n-1}) \in \mathcal{C}$$

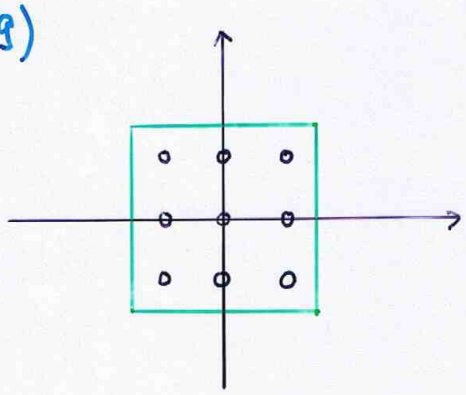
$$\Rightarrow \quad (i \cdot c_{n-1}, c_0, c_1, \ldots, c_{n-2}) \in \mathcal{C}$$

Als Polynomgleichung:

$$\hat{c}(x) = x \cdot c(x) - c_{n-1}(x^n - i)$$

Für $p \equiv 3 \bmod 4$ liegt i im Erweiterungskörper $GF(p^2)$ ⇒ Ergebnisse können übertragen werden.

$GF(9)$



Irred. Polynom: x^2+1 (aber nicht primitiv)

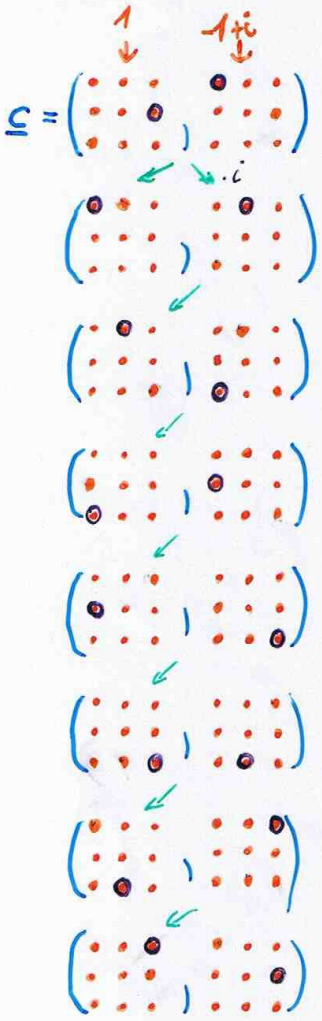
Real- und Imaginärteil mod 3.

$(GF(3) = \{-1, 0, 1\})$

$n = \frac{9-1}{4} = 2$

$[2, 1, d_{\min}]$-Code

zyklischer Code über $GF(3^2)$



$\underline{c} = (\ ,\ ) = g(x)$

jeweils zyklisch verschieben → $\underline{c}$

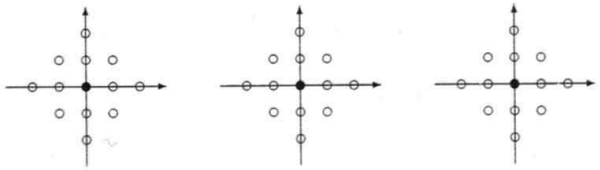
$g(x) = x - \alpha$; $\alpha = 1 - i$ (prim. Element)

k	α^k	$1 \quad i$
0	1	1
1	α	$1 - i$
2	α^2	i
3	α^3	$1 + i$
4	α^4	-1
5	α^5	$-1 + i$
6	α^6	$-i$
7	α^7	$-1 - i$

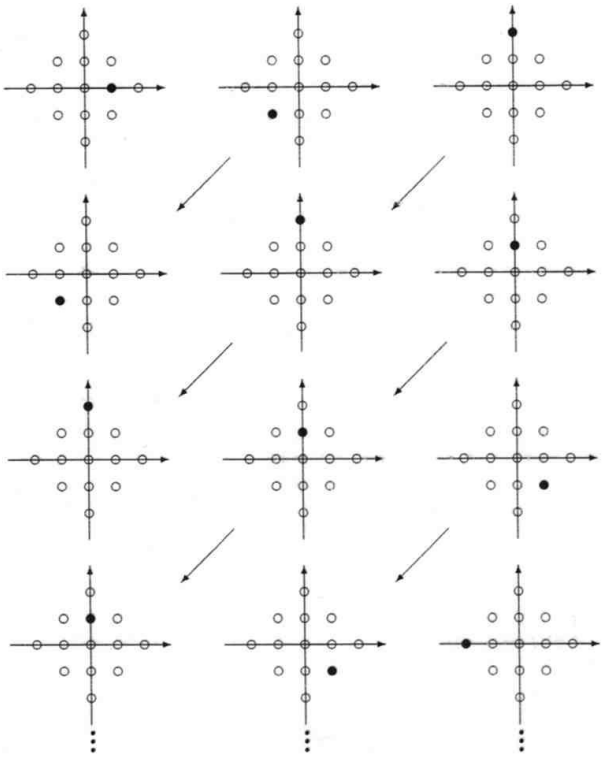
8 CW + Nullwort = 9 CW

$d_{\min} = 3$

Beispiel: izyklischer $[3, 1, 5]$ Code über $GF(13)$
Codeworte sind das Nullwort:



sowie die 12 izyklischen Verschiebungen von:



Decodierverfahren für zyklische Codes

Geht aus (wie negazykl. Dec. Verfahren)

von:

$$S(z) \cdot \sigma(z) + z \cdot \sigma'(z) = 0$$

↖ siehe Folie Decodierverfahren für negazykl. Codes

Unterschied zu negazykl. Codes:

$S(z)$ ist nur für jeden vierten Wert bekannt S_j, $j = 1, 5, 9, \ldots 4t+1$.

→ Details: Bei Interesse → pdf-Datei.

Dualer Code

Vertauschung der Rolle von Generator- und Prüfmatrix führt zum Dualen Code.

G: $k \times n$ Matrix

H: $(n-k) \times n$ Matrix

$[n, k, d]$-Code $\mathcal{C}$

$[n, n-k, d^{\perp}]$ - Code $\mathcal{C}^{\perp}$

Beispiel: $[n, 1, n] \leftrightarrow [n, n-1, 2]$

Achtung: Anordnung ist unterschiedlich

H. A.: Zeigen Sie

$$\underline{c}_a \cdot \underline{c}_b^T = 0$$

mit $\underline{c}_a \in \mathcal{C}$ und $\underline{c}_b \in \mathcal{C}^{\perp}$.

Dualer Code bei zyklischen Codes

$c(x) = i(x) \cdot g(x) \qquad g(x) \mid x^n - 1$

$x^n - 1 = g(x) \cdot h(x)$

$\Rightarrow \quad c(x) \cdot h(x) = i(x) \cdot g(x) \cdot h(x) = i(x)(x^n - 1)$

$\Rightarrow \quad \underline{c(x) \cdot h(x) = 0 \mod (x^n - 1)} \qquad (*)$

$\Rightarrow$ Prüfgleichungen für Codewort

H.A.: Stellen Sie (*) in Matrixform dar

$\left(c(x) = \sum_{i=0}^{n-1} c_i x^i \; ; \; h(x) = \sum_{i=0}^{k} h_i \cdot x^i \right).$

Zyklischer Code: mit $g(x)$ erzeugt

Dualer Code: mit $x^k \cdot h\left(\frac{1}{x}\right)$ erzeugt

"umgedrehtes" Polynom

$h(x)$ erzeugt einen äquivalenten Code.

H.A.: $[7, 4, 3]$ binärer Hamming Code mit $g(x) = x^3 + x + 1$. Bestimmen Sie das Generatorpolynom des dualen Codes.

Die MacWilliams Transformation

Gewichtsverteilung eines Codes: $W(z) = \sum_{c \in \mathcal{C}} z^{wt(c)}$

Gibt an wieviele Codewörter ein bestimmtes Gewicht haben.

z.B. binärer $[8, 2, 5]$ Goppa Code:

$$W(z) = 1 + 2z^5 + z^6$$

(← Folie 5 12.12.2014)

1 CW mit Gew. 0
2 CW mit Gew. 5
1 CW mit Gew. 6

MacWilliams Transformation (ohne Beweis)

$A(z)$: Gewichtsverteilung eines $[n, k, d]$ Codes über $GF(q)$

$\Rightarrow$ Die Gewichtsverteilung des dualen $[n, n-k, d^+]$ Codes ist gegeben durch

$$B(z) = \frac{(1+(q-1)z)^n}{q^k} A\left(\frac{1-z}{1+(q-1)z}\right)$$

Beispiel $[8, 2, 5]$ Code ; $q = 2$

$$A(z) = 1 + 2z^5 + z^6$$

$$B(z) = \frac{(1+z)^8}{2^2} \cdot A\left(\frac{1-z}{1+z}\right)$$

$$= \ldots$$ am Besten mit Computeralgebra-Programm

$$B(z) = 1 + 7z^2 + 18z^3 + 15z^4 + 12z^5 + 9z^6 + 2z^7$$

$\Rightarrow$ $[8, 6, 2]$ Code

Vorteil von MacWilliams Transformation:

Die kompelette Gewichtsverteilung von sehr hochratigen Codes kann erhalten werden durch die Gewichtsverteilung eines niederratigen Codes (der sehr viel weniger Codewörter hat).

H.A.: Bestimmen Sie die Gewichtsverteilung des $[7, 4, 3]$ Ham.Codes mit der Verteilung des $[7, 3, 4]$ Codes.

Originalarbeiten zu speziellen Themen der Vorlesung

- V.D.Goppa, "A New Class of Linear Error-Correcting Codes",Probl. Peredach. Inform. Vol.6,No.3, pp. 24-30, 1970.
- E.R.Berlekamp, "Goppa Codes", IEEE Transactions on Information Theory, No.5, pp.590-592, 1973.
- Y.Sugiyama, M.Kasahara, S.Hirasawa, T.Namekawa, "A Method for Solving Key Equation for Decoding Goppa Codes", Information and Control, 27, 87-99, 1975.
- N.J.Patterson, "The Algebraic Decoding of Goppa Codes",IEEE Trans. on Inf. Theory, Vol.IT-21, No.2, March 1975, pp.203-207.
- K.Huber, "Note on Decoding Binary Goppa Codes", Electronics Letters, 18th January 1996, Vol. 32 No.2, pp. 102-103 (sowie verallgemeinert: K.Huber, "Taking pth roots modulo polynomials over finite fields", Designs, Codes and Cryptography, 28, pp. 303-311, 2003).
- R.Roth, "Introduction to Coding Theory", Cambridge University Press, 2006. (Chapter 10: Lee Metric)
- K.Huber, "Codes over Gaussian Integers", IEEE Transactions on Information Theory, Vol.40, No.1, pp. 207-216, 1994.

Aufgabe 22: Decodieren von $\underline{r} = (1\,1\,0\,1\,1\,0\,1\,1)$ (r_0 r_1 ...)

des systematisch codierten $[8,2,5]$ bin. Goppa Code.

(Folien 3–5, 12.12.'14)

$G(z) = z^2 + z + 1$

i	0	1	2	3	4	5	6	7
α_i	α	1	0	α^2	α^3	α^4	α^5	α^6
$G(\alpha_i)$	α^5	1	1	α^3	α^5	α^6	α^6	α^3

$\frac{G(z) - G(\alpha_i)}{G(\alpha_i)(z - \alpha_i)}$

$\alpha_0 = \alpha \Rightarrow \frac{z^2 + z + 1 + \alpha^5}{\alpha^5 (z + \alpha)} = \alpha^2 z + \alpha^5$

$GF(2^3)$ Tabelle siehe Folie 3 12.12.'14

$\alpha_1 = 1 \Rightarrow \ldots = z$

$\alpha_3 = \alpha^2 \Rightarrow \ldots = \alpha^4 z + \alpha^3$

$\alpha_4 = \alpha^3 \Rightarrow \ldots = \alpha^2 z + \alpha^3$

$\alpha_6 = \alpha^5 \Rightarrow \ldots = \alpha z + \alpha^5$

$\alpha_7 = \alpha^6 \Rightarrow \ldots = \alpha^4 z + \alpha^6$

$S(z) = \sum \frac{r_i}{z - \alpha_i} = \alpha^2 z + \alpha^5 + z + \alpha^4 z + \alpha^3 + \alpha^2 z + \alpha^3 + \alpha z + \alpha^5 + \alpha^4 z + \alpha^6$

$\Rightarrow S(z) = \alpha^3 z + \alpha^6 \Rightarrow T(z) \equiv S^{-1}(z) \bmod G(z)$

$\Rightarrow T(z) = \alpha^6 z + 1 \Rightarrow T(z) + z = \alpha^2 z + 1 \equiv R(z)^2 \bmod G(z)$

$w(z) = z + 1$, $w^2 = z \bmod G$

$R(z) = \alpha \cdot w(z) + 1$

$R(z) = \alpha \cdot z + \alpha^3 \Rightarrow \underbrace{\alpha \cdot z + \alpha^3}_{a(z)} \equiv \underbrace{1}_{b(z)} \cdot R(z) \bmod G(z)$

$\Rightarrow \sigma(z) = \alpha^2 z^2 + z + \alpha^6$

NS: $1 = \alpha_1 \bmod \alpha^4 = \alpha_5 \Rightarrow \underline{e} = (0\,1\,0\,0\,0\,0\,1\,0\,0)$

$\Rightarrow \underline{c} = (1\,0\,0\,1\,1\,1\,1\,1)$

Aufgabe 23: Speicherplatzbedarf für Mc Eliece Kryptosystem

öffentlicher Schlüssel: $k \cdot n$ Bit für Generatormatrix $\underline{G}'$

Bei $n = 1024$, $k = 614$ $\Rightarrow$ 628736 Bit

$\Rightarrow$ 78592 Byte für $\underline{G}'$

1 Byte für t

78593 Byte

geheimer Schlüssel:

$n \cdot n$ Permutationsmatrix $\underline{P}$ bzw. $\underline{P}^{-1}$: $n \cdot ld\, n$ Bit

$k \cdot k$ Binärmatrix $\underline{S}$ bzw. $\underline{S}^{-1}$: k^2 Bit

Goppa Polynom vom Grad t : $(t+1) \cdot m$ Bit

$\uparrow$ $GF(2^m)$

Bei $n = 1024$, $k = 614$, $m = 10$, $t = 41$

$\Rightarrow$ für $\underline{P}^{-1}$: $1024 \cdot 10$ Bit $= 10240$ Bit $\rightarrow$ 1280 Byte

für $\underline{S}^{-1}$: 618^2 Bit $= 381924$ Bit $\rightarrow$ 47741 Byte

$G(z)$: $42 \cdot 10 = 420$ Bit $\rightarrow$ 53 Byte

49074 Byte