

NT - Zusammenfassung

Informationstheorie

Information ist eine Nachricht, die relevant und nicht-redundant ist.

redundant: überflüssig

relevant: Sender/Empfänger müssen gleichen Zeichenvorrat haben

Entscheidungsgehalt: Wie viele Schritte (Bit's) brauche ich, um alle Zeichen binär zu beschreiben?

gleichwahrsch.: $H_0 = \log_2 \left(\frac{1}{p(x_k)} \right)$

$$H_0 = \log_2(N) \text{ Bit}$$

H_0 = Entscheidungsgehalt [Bit]

\log_2 = Logarithmus binär

N = Zeichenvorrat

(bei $\{1, 2, 3, 4\}$ $N=4$)

Binärer Logarithmus: $\log_2(n) = \frac{\log(n)}{\log 2}$

Entscheidungsfluss: $H_0^* = \frac{\log_2(N)}{\tau} \text{ Bit/s}$

H_0^* = Entscheidungsfluss

τ = Übertragungszeit eines Zeichens

Informationsgehalt:
eines Zeichens

$$I(x_k) = \log_2 \left(\frac{1}{p(x_k)} \right) \text{ Bit}$$

$I(x_k)$ = Information des Zeichens

x_k

x_k = Zeichen x_k

$p(x_k)$ = Wahrscheinlichkeit,

dass Zeichen x_k vorkommt

Mittlerer Informationsgehalt
eines Zeichens / Entropie:

$$H(x) = \sum_{k=1}^N p(x_k) \cdot I(x_k)$$

$$H(x) = \sum_{k=1}^N p(x_k) \cdot \log_2 \left(\frac{1}{p(x_k)} \right)$$

Bei Gleichwahrsch.:

$$\log_2(N) = \frac{\log(N)}{\log(2)}$$

$H(x)$ = Entropie / mittlerer

Informationsgehalt

eines Zeichens

Wann ist Entropie maximal:

Bsp.

$$X = \{x_1, x_2\}$$

$$p(x_1) = p, p(x_2) = 1-p$$

$$\Rightarrow H(x) = -p \cdot \log_2(p) - (1-p) \cdot \log_2(1-p)$$

Maximale Entropie, wenn alle Zeichen gleichwahrscheinlich sind. Minimale Entropie, wenn

$H(x)$ gegen H_0 läuft.

Redundanz: "Überflüssigkeit"

$$R_a = H_0 - H(X) \text{ [Bit/Zeichen]}$$

R_a = Redundanz

H_0 = Entscheidungsgehalt

$H(X)$ = Entropie (Mittlerer Informationsgehalt der Quelle)

Mittlere Codewortlänge:

(z.B. beim Morsem)

$$L = \sum_{i=1}^N p(x_i) \cdot L(x_i)$$

→ Script Folie S. 32ff

L = mittlere Codewortlänge

$L(x_i)$ = Codewortlänge eines Zeichen x_i

$p(x_i)$ = Wahrscheinlichkeit v. x_i

Shannon'sches

Codierungstheorem:

→ S. 35

$$H(X) \leq L$$

$$H(X) \leq L \leq H(X) + 1$$

Redundanz des Codes:

$$R_c = L - H(X) \text{ [Bit/Zeichen]}$$

R_c = Redundanz des Codes

R_c soll kleiner sein als R_a !

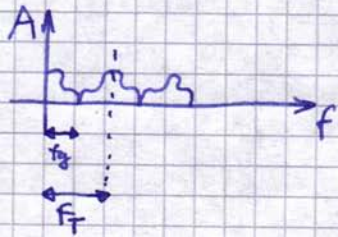
NT-Zusammenfassung

Signalformeln

$$f_T \geq 2 f_g$$

f_T = Trägerfrequenz

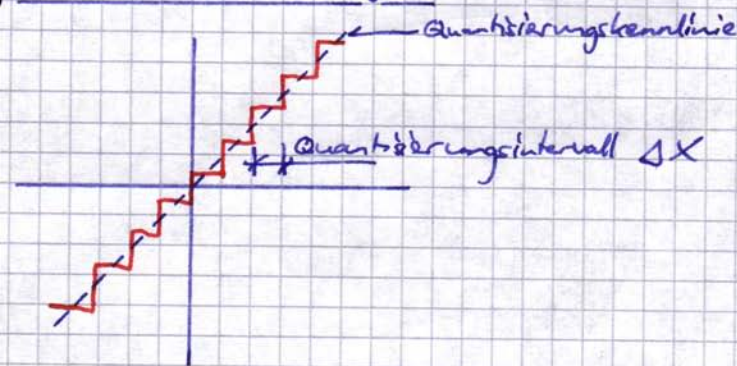
f_g = Grenzfrequenz



Unterabtastung: $f_T < 2 f_g$ (Signale können nicht auseinandergehalten werden)

Überabtastung: $f_T > 2 f_g$

Quantisierung:



Quantisierungsrauschen

Klirrfaktor

$$\sqrt{\frac{P_a}{P_s}} = \frac{1}{2^n}$$

n = Codewort-Länge

P_a = Geräuschabstand

P_s = Signalabstand

Anzahl der Quantisierungsschritte: 2^n

$$\log \sqrt{\frac{P_s}{P_a}} = 2n \cdot \log 2 \approx n \cdot 6 \text{ dB}$$

Wird die Stellenzahl des Binärcodes um 1 Bit erhöht, so verbessert sich der Signal-/Geräuschabstand um 6 dB.

Blockcodes / Hamming-Code

Prinzip:

→ Folie 59



Die Kontrollstellen werden anhand eines Algorithmus mit den Nachrichtenstellen erstellt. z.B. $x_1, x_2 =$ Nachrichtenstellen, $k = x_3$

$$x_3 = (x_1 + x_2) \bmod 2$$

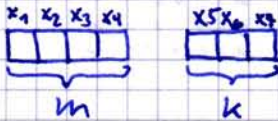
Beim Beispiel ist 1 Fehler erkennbar, da sich x_3 ändert und so ein ungültiges Codewort entsteht.

2 Fehler sind nicht mehr erkennbar, da ein anderes gültiges CW entsteht (aus 101 wird 011 $\Rightarrow 0 + 1 \bmod 2 = 1$)

Der Fehlerort (Nachrichten- oder Kontrollstellen) ist unabhängig von der Fehlererkennungs- bzw. Fehlerkorrekturfähigkeit.

Hamming-Code:

Ein spezieller Blockcode ist der Hamming-Code. Er hat eine



Hammingdistanz $h = 3$. Somit ist

1 Fehler korrigierbar und 2 Fehler erkennbar.

$$n = \text{Nachrichtlänge} = m + k$$

Es können max. 2^m gültige Codeworte erstellt werden.

Algorithmus zur Berechnung der Kontrollstellen:

$$x_5 = (x_1 + x_2 + x_3) \bmod 2$$

$$x_6 = (x_2 + x_3 + x_4) \bmod 2$$

→ Folie 60/61

$$x_7 = (x_1 + x_2 + x_4) \bmod 2$$

Generatormatrix:

Aus ~~folgend~~ diesen 3 Gleichungen kann man eine Generatormatrix aufstellen, indem man für jede zu Addierende Stelle eine '1' und für die nicht addierende Stelle eine '0' einsetzt.

Generatormatrix:

Vektor

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} \equiv (1 \cdot x_1 + 1 \cdot x_2 + 1 \cdot x_3 + 0 \cdot x_4) \pmod 2 = 1 \cdot x_5 + 0 \cdot x_6 + 0 \cdot x_7 \\ \equiv (0 \cdot x_1 + 1 \cdot x_2 + 1 \cdot x_3 + 1 \cdot x_4) \pmod 2 = 0 \cdot x_5 + 1 \cdot x_6 + 0 \cdot x_7 \\ \equiv (1 \cdot x_1 + 1 \cdot x_2 + 0 \cdot x_3 + 1 \cdot x_4) \pmod 2 = 0 \cdot x_5 + 0 \cdot x_6 + 1 \cdot x_7 \end{matrix}$$

↑ ↑ ↑ ↑ ↑ ↑ ↑
x₁ x₂ x₃ x₄ x₅ x₆ x₇

Durch Umsortieren des einzelnen Vektoren entstehen neue gültige Codeworte. → Folie 61/62

Fehlersyndrom:

Falls ein Fehler übertragen wurde markiert die Prüfspalte den Fehlerort:

Generatormatrix:

Nachrichtenstellen				Kontrollstellen		
1	1	1	0	1	0	0
0	1	1	1	0	1	0
1	1	0	1	0	0	1

1 1 0 1 0 0 1	$\begin{matrix} 1+1+0 = 0 \\ 1+0+1 = 0 \\ 1+1+1 = 1 \end{matrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
---------------	---	---

1 0 0 1 0 0 1	$\begin{matrix} 1+0+0 = 1 \\ 0+0+1 = 1 \\ 1+0+1 = 0 \end{matrix}$	$\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$
---------------	---	---

Nullmatrix → alle 0

1 1 0 0 0 0 1	$\begin{matrix} 1+1+0 = 0 \\ 1+0+0 = 1 \\ 1+0+0 = 1 \end{matrix}$	$\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$
---------------	---	---

$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \checkmark$$

$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \checkmark$$

$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

Zyklische Codes

Prinzip: Die Generatormatrix des Blockcodes kann durch ein Generatorpolynom ersetzt werden.

Codebedingung: Das Codewortpolynom ist ohne Rest durch das Generatorpolynom teilbar (in mod-2-Rechnung)

Generatorpolynom: $1 \cdot u^3 + 0 \cdot u^2 + 1 \cdot u^1 + 1 \cdot u^0$

\uparrow \uparrow \uparrow \uparrow
 g_1 g_2 g_3 g_4

u^i definiert die Codewortstelle

Generatorpolynom $G(u) = \sum_{i=0}^k g_i \cdot u^i$

$k = \text{Anz. Prüfstellen}$
 $m = \text{Anz. Nachrichtenstellen}$
 $n = \text{Anz. Codewortstelle}$

Im Beispiel ergibt $G(u) = 1011$

Ein Generatorpolynom ist ein primitives Polynom (wie Primzahl). Einige sind auf Folie 70 abgebildet.

Unterschied $G(u)$ & $X(u)$? Folie 65?

Polynomdivision: Wenn das Codewort durch das Generatorpolynom ohne Rest teilbar ist, ist das Codewort gültig, sonst ungültig.

Mit der Polynomdivision (resp. Mehrfachaddition) kann man für jedes Code Nachricht (m -Stellig) das entsprechend gültige Codewort (n -Stellig) ausrechnen.

Beispiel: Nachricht 1000 ($m = 4$ Stellen) } $n = 7$ Stellen
 ($k = 3$ Stellen)

$$\begin{array}{r}
 1000 : 1011 = 1011 \\
 \underline{1011} \\
 \text{mod 2 Rest } 1100 \\
 \underline{1011} \\
 1111 \\
 \underline{1111} \\
 1011 \\
 \underline{1011} \\
 \boxed{101} \text{ Rest}
 \end{array}$$

Erste Annahme: Alle Kontrollstellen sind '0'. Falls Division kein Rest ergibt ist Annahme richtig, sonst muss der erhaltene Rest dem Codewort hinzugefügt werden, damit ohne Rest geteilt werden kann
 \Rightarrow gültiges CW = 1000101

Mehrfachaddition:

→ Folie 67

Es wird mühsam, für jedes mögliche Nachrichtenwort durch Polynomdivision seine gültigen Kontrollstellen auszurechnen. Einfacher geht dies durch Mehrfachaddition.

Bsp.: Code wort 1000

$$\begin{array}{r}
 1000 \dots \\
 1011 \\
 \hline
 11011 \\
 \hline
 11011 \\
 \hline
 \boxed{101}
 \end{array}$$

Es wird immer bei der vordersten '1' das Generatorpolynom hinzuaddiert und modulo-2 gerechnet. Wenn man am Ende

des Nachrichtenwort angelangt ist, ist die Rechnung fertig.

Die letzten 3 Stellen ergeben die Kontrollstellen.

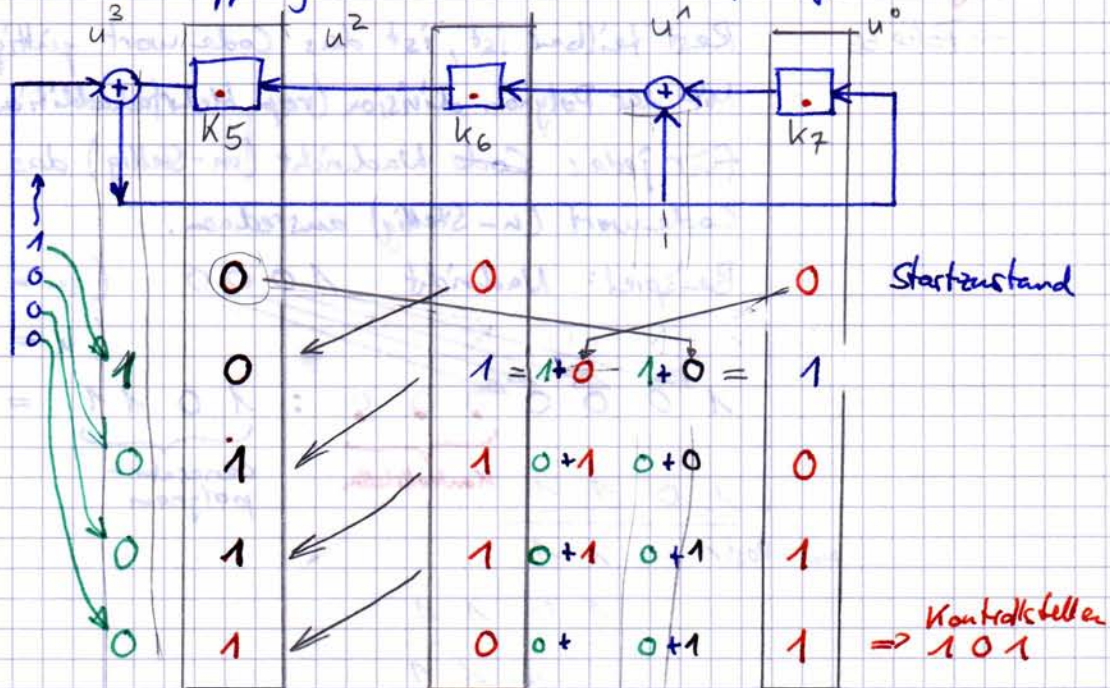
Im Beispiel ist also 1000101 ein gültiges Code wort.

Rückgekoppeltes Schieberegister:

Zur Berechnung der Kontrollstellen (wie Polynomdivision bzw. Mehrfachaddition). Ein Schieberegister ist Hardwaremäßig einfach zu realisieren und deshalb hat diese Methode zur Kontrollstellenberechnung seine Berechtigung.

Wir haben wieder das Generatorpolynom 1011
 $\begin{matrix} \uparrow & \uparrow & \uparrow & \uparrow \\ g_3 & g_2 & g_1 & g_0 \end{matrix}$

Eine Rückkopplung wird überall dort realisiert, wo $g_i=1$



Das Code wort (Nachrichtenstellen) wird Bit für Bit ins Schieberegister geschoben. Die 3 Stellen deuten die Kontrollstelle. Es wird jeweils von links nach rechts gerechnet.

NT-Zusammenfassung Forts. Zyklische Codes

2.2.2003, M. Riegling

Zur Kontrollstellen ermittlung: Zur Berechnung der Kontrollstellen erhält man immer dasselbe Ergebnis, ob man mit Polynomdivision, Mehrfachaddition oder Schieberegister rechnet.
Falls man die zum Generatorpolynom entsprechende Generatormatrix hat, ergibt dessen Algorithmus das selbe Kontrollstellen-Ergebnis!

CRC-/Abramson-Codes:

Diese Codes haben eine Hammingdistanz $h=4$. Dies wird erreicht, indem dem primitiven Polynom (\rightarrow Folie 70) ~~zur~~ der Term $(x+1)$ multipliziert wird.

Generatorpolynom:

Beim ^{Aus dem} primitiven Polynom: $x^3 + x + 1$ wird so
 $(x+1)(x^3+x+1) = x^4 + x^3 + x^2 + 1$ ($x^4 + x^3 + x^2 + 2x + 1$)
 \rightarrow Hammingdistanz = 4

$$n = 2^{k_1} - 1$$

$$k = k_1 + 1$$

n = Anzahl der CW-Stellen

k = Anzahl der Kontrollstellen