389.138 Telekommunikation – 2014S

2. Übung – Lösungen

05.05.2014 – 09.05.2014

Einen Überblick über die in der Übung verwendete Notation finden Sie in TUWEL.
Sollten Sie Fehler finden, so wenden Sie sich bitte an tk138@nt.tuwien.ac.at.

Beispiel 1

(a) Die Entropie von S ergibt sich zu (vgl. [1, 9.2.4])

$$H = \sum_{i=1}^{5} p_{\mathsf{S}}(s_i) \log_2\left(\frac{1}{p_{\mathsf{S}}(s_i)}\right) = 1.8987\,\text{bit}.$$

(b) Die maximale Entropie einer Zufallsvariable mit 5 Symbolen beträgt $H_{\max} = \log_2(5)$. Damit ergibt sich die Redundanz zu (Formel [1, (9.8)])

$$R = H_{\max} - H = 0.423\,22\,\text{bit}.$$

(c) Die Konstruktion des Codes ist in Abb. 1 veranschaulicht. Der zugehörige Code ist in Tabelle 1 angegeben.

(d) Die Länge der Codewörter kann aus Tabelle 1 abgelesen werden. Damit ist die durchschnittliche Codewortlänge

$$L = \sum_{i=1}^{5} p_{\mathsf{S}}(s_i) l(s_i) = 1.95$$

und daher die Codeeffizienz

$$\eta_{\text{Code}} = \frac{H}{L} = 0.97369.$$

(e) Ein Vergleich mit der Codierung aus Tabelle 1 ergibt die Symbolfolge $s_2, s_1, s_1, s_5, s_1, s_2$.

(f) Einen möglichen Code finden Sie in Tabelle 2.

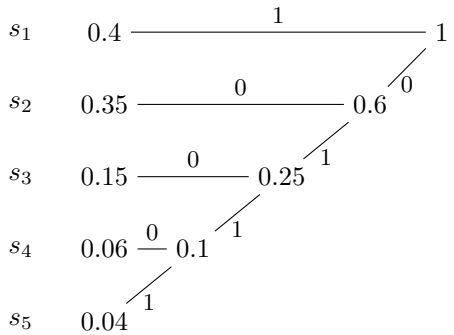


Abbildung 1: Konstruktion des Huffman-Codes (Beispiel 1)

Symbol	Code	Länge, $l(s_i)$
s_1	`1`	1
s_2	`00`	2
s_3	`010`	3
s_4	`0110`	4
s_5	`0111`	4

Tabelle 1: Huffman-Code (Beispiel 1)

Symbol	Code
s_1	0
s_2	00
s_3	000
s_4	0000
s_5	00000

Tabelle 2: Nicht eindeutig dekodierbarer Code (Beispiel 1)

(g) Liest man die Codewörter in Tabelle 1 von Hinten nach Vorne, so ergibt das einen zwar eindeutig, aber nicht fortlaufend dekodierbaren Code, da die resultierende Codierung nicht mehr präfixfrei ist. (`1` ist Präfix von `1110`.)

(h) (Bonus) Der Huffman Code besitzt die Code-Effizienz $\eta_{\text{Code}} = 1$ genau dann, wenn es für jedes Symbol s eine Zahl $k \in \mathbb{N}$ gibt, mit $\mathrm{P}\{s\} = 2^{-k}$.

Beispiel 2

(a) Eine mögliche Strategie ist eine "Binäre Suche"[1]. Für die Beschreibung dieses iterativen Algorithmus verwenden wir i als Laufvariable.

(i) *Initialisierung:* Man setzt $i \triangleq 0$, $M \triangleq \lceil \log_2(N) \rceil$, $A_i \triangleq 0$ und $B_i \triangleq 2^M$.

(ii) Für $i = i + 1$ weiß man dass für die gesuchte Zahl x gilt $A_{i-1} < \mathsf{X} \leq B_{i-1}$. Man stellt fest, ob $A_{i-1} < x \leq \frac{B_{i-1}}{2}$. Dann werden die Intervallgrenzen angepasst:

$$\text{"}x \in \left\{ A_{i-1}+1, A_{i-1}+2, \ldots, \frac{B_{i-1}}{2} \right\}\text{?"} : \quad \begin{cases} \text{Ja} & \Rightarrow A_i \triangleq A_{i-1}, B_i \triangleq \frac{B_{i-1}}{2} \\ \text{Nein} & \Rightarrow A_i \triangleq \frac{B_{i-1}}{2}, B_i \triangleq B_{i-1} \end{cases} \tag{1}$$

(iii) Für $i < M$ gehe zu Punkt (ii). Da das Suchintervall $(A_i, B_i]$ zu Beginn genau 2^M Elemente hat und mit jeder Iteration halbiert wird, folgt für $i = M$, $x = B_M = A_M + 1$.

(b) Die angegebene Strategie ist optimal. Um das zu zeigen nehmen wir an, es gäbe eine bessere Strategie, d.h. eine Strategie mit echt kleinerer maximaler Anzahl an zu stellenden Fragen. Sei L diese maximale Anzahl, dann wissen wir

$$L < \lceil \log_2(N) \rceil \implies 2^L < N. \tag{2}$$

Wir definieren nun die Abbildung $f\colon \{1, 2, \ldots, N\} \to \{J, N\}^L$, die jeder Zahl jene Abfolge von Antworten ("J"/"N") zuweist, die diese Zahl eindeutig identifizieren. Wird eine Zahl bereits mit weniger als L Fragen gefunden, so können die nachfolgenden Antworten beliebig gewählt werden. Nachdem jede Zahl $x \in \{1, 2, \ldots, N\}$ eindeutig durch die L Antworten $f(x)$ bestimmt ist, ist die Funktion f injektiv. Das bedeutet $N = |\{1, 2, \ldots, N\}| \leq \left|\{J, N\}^L\right| = 2^L$, was aber einen Widerspruch zu Gleichung (2) darstellt.

(c) Nachdem die Strategie, unabhängig vom konkreten Wert $x \in \{1, 2, \ldots, N\}$, immer genau $\lceil \log_2(N) \rceil$ Fragen benötigt ist natürlich auch die mittlere Anzahl $\lceil \log_2(N) \rceil$.

[1] `http://de.wikipedia.org/wiki/Binäre_Suche`

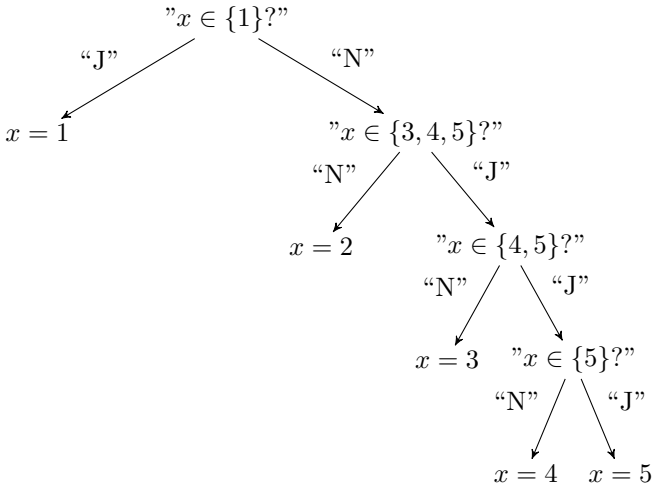


Abbildung 2: Fragestrategie

(d) Man kann den Huffman-Code aus Beispiel 1 verwenden und nacheinander die Werte der Codebits erfragen: Sei $\mathbf{c} = (c_i) \in \{0, 1\}^4$ das zu s_x gehörende Codewort (mit "0" aufgefüllt, wenn nötig). Dann sind die Fragen "$c_i = 1$?", mit nacheinander $i = 1, 2, 3, 4$, optimal, wobei abgebrochen wird, sobald x eindeutig identifiziert ist. Diese Fragestrategie ist in Abb. 2 dargestellt. Beachten sie die Ähnlichkeit zur Konstruktion des Huffman Codes in Abb. 1.

Die mittlere Anzahl an Fragen ist genau die mittlere Codewortlänge des Codes $L = 1.95$. Da jede Fragestrategie über $J : 1$, $N : 0$ einem Code entspricht mit "mittlere Anzahl an Fragen = mittlere Codewortlänge", ist die Strategie ist optimal, da der Huffman-Code optimal ist.

(e) Wie in Punkt (d) erwähnt, entspricht jede Fragestrategie einem Code, wobei die mittlere Anzahl an Fragen gleich der mittleren Codewortlänge ist. Da die Entropie eine untere Schranke für die mittlere Codewortlänge darstellt ist sie somit auch eine untere Schranke für die mittlere Anzahl an Fragen.

(f) (Bonus) Wir berechnen die mittlere Anzahl an Fragen für die Strategie "$x \in \{i\}$?", wobei der Reihe nach $i = 1, 2, 3, \ldots$

$$L = \sum_{x \in \mathcal{X}} l(x) p_{\mathsf{X}}(x) = \sum_{i=1}^{\infty} i 2^{-i} = \sum_{j=1}^{\infty} \sum_{i=j}^{\infty} 2^{-i} = \sum_{j=1}^{\infty} 2^{-j+1} = 2.$$

Diese Strategie ist optimal, da die mittlere Anzahl an Fragen gleich der Entropie ist:

$$H = -\sum_{x \in \mathcal{X}} p_{\mathsf{X}}(x) \log_2 p_{\mathsf{X}}(x) = \sum_{i=1}^{\infty} 2^{-i} i$$

Beispiel 3

(a) Es sei $\mathcal{Y}_{n,k} \subseteq \mathrm{GF}(2)^n$ die Menge aller Vektoren, die sich an k Stellen von einem gegebenen Codewort $\mathbf{x}$ unterscheiden:

$$\mathcal{Y}_{n,k} \triangleq \left\{ \mathbf{y} \middle| \mathrm{d}(\mathbf{x}, \mathbf{y}) = k \right\}$$

Die Wahrscheinlichkeit, dass ein spezifisches $\mathbf{y} \in \mathcal{Y}_{n,k}$ auftritt (z.B. $\mathbf{y} = (\neg x_1, \ldots, \neg x_k, x_{k+1}, \ldots, x_n)$)

ist, unabhängig vom konkreten Vektor $\mathbf{y}$,

$$\mathrm{P}\{\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x}\} = \prod_{i=1}^{n} p_{\mathsf{Y}|\mathsf{X}}(x_i|y_i) = p^k(1-p)^{n-k}$$

für alle $\mathbf{y} \in \mathcal{Y}_{n,k}$. Das Ergebnis entspricht einem Binomial-Experiment (auch Bernoulli Kette genannt) und ist in Grunde das n-fache Ausführen eines Bernoulli Experiments.

(b) Aus kombinatorischen Überlegungen folgt, dass $|\mathcal{Y}_{n,k}| = \binom{n}{k}$. Die Wahrscheinlichkeit, dass ein beliebiges $\mathbf{y} \in \mathcal{Y}_{n,k}$ auftritt ergibt sich somit zu

$$\begin{aligned}\mathrm{P}\{\mathbf{Y}\in\mathcal{Y}_{n,k}|\mathbf{X}=\mathbf{x}\} &= \mathrm{P}\left\{\bigvee_{\mathbf{y}\in\mathcal{Y}_{n,k}} \mathbf{Y}=\mathbf{y}\middle|\mathbf{X}=\mathbf{x}\right\} = \sum_{\mathbf{y}\in\mathcal{Y}_{n,k}} \underbrace{\mathrm{P}\{\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x}\}}_{p^k(1-p)^{n-k}} \\ &= |\mathcal{Y}_{n,k}|\, p^k(1-p)^{n-k} = \binom{n}{k} p^k(1-p)^{n-k}.\end{aligned} \tag{3}$$

Das Ergebnis entspricht der *Binomialverteilung*.

(c) Nein, die Wahrscheinlichkeit, dass k Fehler auftreten ist unabhängig von den konkreten Vektoren $\mathbf{x}$ und $\mathbf{y}$, da die Fehlerwahrscheinlichkeit p sowie die Wahrscheinlichkeit $1-p$ einer korrekten Übertragung unabhängig vom Kanaleingang ist,

$$\begin{aligned}p_{\mathsf{Y}|\mathsf{X}}(1|0) &= p_{\mathsf{Y}|\mathsf{X}}(0|1) = p \\ p_{\mathsf{Y}|\mathsf{X}}(0|0) &= p_{\mathsf{Y}|\mathsf{X}}(1|1) = 1-p\end{aligned}$$

und der Kanal *gedächtnislos* ist. D.h. ob ein Bitfehler an einer Stelle l im Codeblock auftritt ist statistisch unabhängig davon, ob Fehler an den anderen Stellen $\neq l$ auftreten und gleich für alle l. Mathematisch ausgedrückt impliziert dieses gedächtnislose Verhalten den gegebenen Zusammenhang

$$\mathrm{P}\{\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x}\} = \prod_{i=1}^{n} p_{\mathsf{Y}|\mathsf{X}}(x_i|y_i).$$

(d) Ja, denn es gilt $p_{\hat{\mathsf{X}}|\mathsf{X}}(-v|v) = p_{\hat{\mathsf{X}}|\mathsf{X}}(v|-v) = \mathcal{Q}(\frac{v}{\sigma}) = p$ sowie $p_{\hat{\mathsf{X}}|\mathsf{X}}(v|v) = p_{\hat{\mathsf{X}}|\mathsf{X}}(-v|-v) = 1-p$.

(e) Es zeigt sich, dass die Wahrscheinlichkeit (3), dass k Fehler auftreten für gegebenes n und p eine Binomial-verteilte Zufallsvariable K definiert:

$$p_{\mathsf{K}}(k) = \mathrm{P}\{\mathsf{K}=k\} = \binom{n}{k} p^k(1-p)^{n-k}, \tag{4}$$

wobei $\sum_k p_{\mathsf{K}}(k) = 1$ erfüllt ist. Die zugehörige Verteilungsfunktion $P_{\mathsf{K}}(K) = \mathrm{P}\{\mathsf{K}\leq K\}$ ist daher

$$P_{\mathsf{K}}(K) = \mathrm{P}\{\mathsf{K}\leq K\} = \mathrm{P}\{\mathsf{K}=0 \vee \mathsf{K}=1 \cdots \vee \mathsf{K}=K\} = \sum_{k=0}^{K} \mathrm{P}\{\mathsf{K}=k\} = \sum_{k=0}^{K} \binom{n}{k} p^k(1-p)^{n-k}$$

In diesem Zusammenhang sei noch die Wahrscheinlichkeit $\mathrm{P}\{\mathsf{K}>K\}$, dass mehr als K Fehler in n Bits auftreten, erwähnt.

$$\mathrm{P}\{\mathsf{K}>K\} = 1 - \sum_{k=0}^{K} \mathrm{P}\{\mathsf{K}=k\} = 1 - \sum_{k=0}^{K} \binom{n}{k} p^k(1-p)^{n-k}.$$

Damit kann beispielsweise bei gegebener Codewortlänge n, Bitfehlerwahrscheinlichkeit p und Anzahl an korrigierbaren Fehlern t eines Codes $\mathcal{C}$ die Wahrscheinlichkeit eines Blockfehlers bestimmt werden.

Beispiel 4 — Kanalkapazität

(a) In diesem Unterpunkt war der Zusammenhang zwischen der Kapazität des zeitdiskreten AWGN Kanals und des Bandbreite-beschränkten AWGN Kanals herzuleiten. Wie bereits in der Angabe erwähnt, beschäftigt sich die Informationstheorie mit zeitdiskreten Kanälen der Form $f_{\mathsf{Y}|\mathsf{X}}(y|x)$ bzw. $p_{\mathsf{Y}|\mathsf{X}}(y|x)$; die Kapazität für zeitkontinuierliche Kanäle folgt dann aus den zeitdiskreten Kapazitäten. Im Falle der Kapazität des zeitdiskreten AWGN Kanals ist

$$C_{\text{diskret}} = \frac{1}{2}\log_2\left(1+\frac{P}{\sigma^2}\right)$$

Bits pro Benützung des skalaren Kanals $f_{\mathsf{Y}|\mathsf{X}}(y|x)$, wobei $\sigma^2 = N_0 B$ die Rauschleistung pro Sample ist.

Es stellt sich nun die Frage, wie viele Übertragungen pro Sekunde über einen Kanal der Bandbreite B möglich sind, falls die Samples am Kanaleingang jeden beliebigen Wert annehmen dürfen. Das Abtasttheorem besagt, dass jedes mit der Bandbreite B beschränkte Signal durch $R_s = 2B$ Werte (pro Zeiteinheit) exakt bestimmt ist, also (pro Zeiteinheit) nur R_s Freiheitsgrade besitzt (vgl. Bonusbeispiel der 1. Übung). Daraus folgt, dass pro Sekunde R_s beliebige Samples übertragen werden können und, dass das durch Tiefpassfilterung aus den Samples resultierende Signal eine Bandbreite $\leq B$ haben wird. Folglich ist

$$C_{\text{kont}} = \underbrace{R_s}_{\frac{\text{Sample}}{\text{Zeit}}} \cdot \underbrace{C_{\text{diskret}}}_{\frac{\text{Bit}}{\text{Sample}}} = B\log_2\left(1+\frac{P}{N_0 B}\right) \text{ Bit/Sekunde.}$$

(b) Siehe Abb. 3

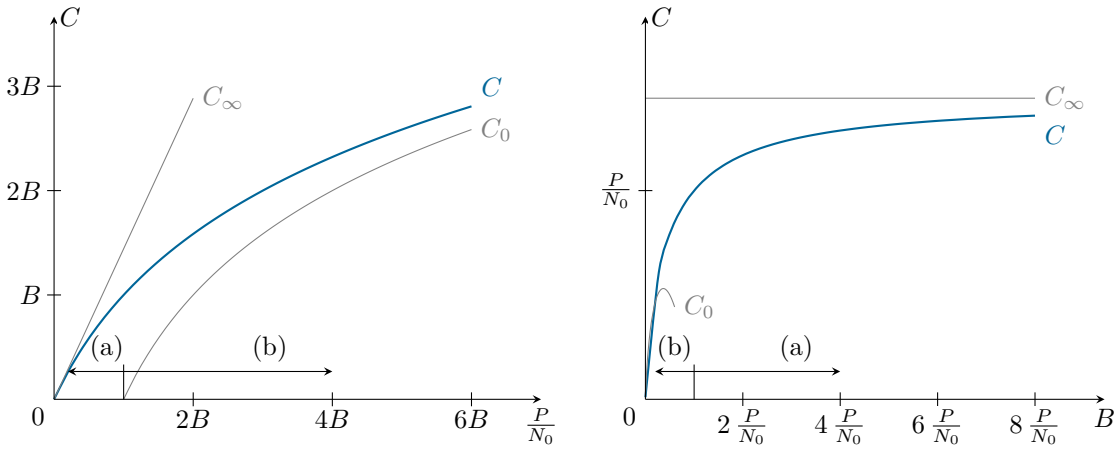


Abbildung 3: Kanalkapazität über SNR und Bandbreite

(c) Wegen der Stetigkeit der Funktion $\log_2(\cdot)$ folgt mit dem Hinweis

$$C_\infty = \lim_{B\to\infty} B\log_2\left(1+\frac{P}{BN_0}\right) = \frac{P}{N_0}\lim_{B\to\infty}\frac{BN_0}{P}\log_2\left(1+\frac{P}{BN_0}\right)$$

$$= \frac{P}{N_0} \log_2 \lim_{B\to\infty} \left(1 + \frac{P}{BN_0}\right)^{\frac{BN_0}{P}} = \frac{P}{N_0} \log_2 \mathrm{e}.$$

Abbildung 3 veranschaulicht diese Grenze. Der Grund, warum unendliche Bandbreite nicht zu unendlicher Kapazität führt, liegt im Rauschen: Je größer die Bandbreite B wird, desto stärker wirkt sich auch die Rauschleistung $\sigma^2 = BN_0$ auf die Übertragung aus. Generell können folgende Fälle unterschieden werden

- *Leistungs-beschränktes Regime:* $B \gg \frac{P}{N_0}$ (entsprechend SNR $\ll 0\,\mathrm{dB}$), Abb. 3 (a).

$$C \approx C_\infty = \frac{P}{N_0} \log_2 \mathrm{e}$$

- *Bandbreite-beschränktes Regime:* $B \ll \frac{P}{N_0}$ (entsprechend SNR $\gg 0\,\mathrm{dB}$), Abb. 3 (b).

$$C \approx C_0 = B \log_2 \left(\frac{P}{N_0 B}\right)$$

Beispiel 5 — Binäre, lineare Codes

Ein binärer, linearer Blockcode der Länge n ist eine nichtleere Menge $\mathcal{C} \subseteq \mathrm{GF}(2)^n$ für die gilt

$$\mathbf{c}, \mathbf{c}' \in \mathcal{C} \implies \mathbf{c} + \mathbf{c}' \in \mathcal{C}\,. \tag{5}$$

Im Folgenden verstehen wir $+, \cdot$ die entsprechende Rechenoperation in $\mathrm{GF}(2)^n$.

(a) Man wähle $\mathbf{c} = \mathbf{c}' \in \mathcal{C}$. Dies ist möglich, da $\mathcal{C}$ nach Voraussetzung nicht leer ist. Damit folgt aus Gleichung (5),

$$\mathbf{c} + \mathbf{c}' = \mathbf{c} + \mathbf{c} = \mathbf{0} \in \mathcal{C}.$$

(b) Die Hammingdistanz $\mathrm{d}(\mathbf{c}, \mathbf{c}')$ ist die Anzahl der Bits durch die sich die beiden Vektoren $\mathbf{c}$ und $\mathbf{c}'$ mit den n Elementen c_i bzw. c'_i unterschieden. Durch Vergleich mit der Operations-Tabelle[2] für "+" in GF(2) ist ersichtlich, dass dies für jedes Element c_i und c'_i der Addition $c_i + c'_i$ entspricht. Mit der Summe $\sum_{i=1}^n$ in $\mathbb{Z}$ folgt daher

$$\mathrm{d}(\mathbf{c}, \mathbf{c}') = \sum_{i=1}^{n} \begin{Bmatrix} 1 & c_i \neq c'_i \\ 0 & \text{sonst} \end{Bmatrix} = \sum_{i=1}^{n} c_i + c'_i = \mathrm{w}(\mathbf{c} + \mathbf{c}'). \tag{6}$$

über die Definition des Codegewichts $\mathrm{w}(\mathbf{c}) = \sum_{i=1}^n c_i$.

(c) Wir zeigen zunächst $w_{\min} \geq d_{\min}$. Es existiert mindestens ein Codewort $\mathbf{c}_{\min}$ mit dem minimalen Codegewicht $w_{\min} = \mathrm{w}(\mathbf{c}_{\min})$. Daraus folgt:

$$w_{\min} = \mathrm{w}(\mathbf{c}_{\min}) = \mathrm{w}(\mathbf{c}_{\min} + \mathbf{0}) \overset{(6)}{=} \mathrm{d}(\mathbf{c}, \mathbf{0}) \geq d_{\min} \tag{7}$$

Um $d_{\min} \geq w_{\min}$ zu zeigen beachten wir, dass zwei Codewörter $\mathbf{c}_1$ und $\mathbf{c}_2$ existieren, die den minimalen Hammingabstand besitzen. Es folgt:

$$d_{\min} = \mathrm{d}(\mathbf{c}_1, \mathbf{c}_2) \overset{(6)}{=} \mathrm{w}(\underbrace{\mathbf{c}_1 + \mathbf{c}_2}_{\in \mathcal{C}}) \geq w_{\min} \tag{8}$$

[2] siehe Beiblatt "Notation, Definitionen und Formelsammlung".

Aus (7) und (8) folgt die nützliche Beziehung $w_{\min} = d_{\min}$ für binäre, lineare Blockcodes.

(d) Als linearer Vektorraum besitzt $\mathcal{C}$ immer eine Basis der Form $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k\}$.

Jedes/r Codewort/Vektor in $\mathcal{C}$ hat eine eindeutige Darstellung als Linearkombination der Basisvektoren $\mathbf{b}_i$ mit den Koeffizienten a_i aus dem Skalarkörper GF(2) der Form

$$\mathbf{c} = \sum_{i=1}^{n} a_i \mathbf{b}_i. \tag{9}$$

Gleichung (9) kann als bijektive Abbildung zwischen den Koeffizientenvektor $\mathbf{a} \in \mathrm{GF}(2)^k$ und den Codewörtern $\mathbf{c} \in \mathcal{C}$ verstanden werden. Daher haben beide gleich viele Elemente und es gilt

$$|\mathcal{C}| = \left|\mathrm{GF}(2)^k\right| = |\mathrm{GF}(2)|^k = 2^k.$$

(e) Die Generatormatrix erzeugt die Codewörter $\mathbf{c}$ aus den Datenwörtern $\mathbf{d}$ nach $\mathbf{c}^T = \mathbf{d}^T\mathbf{G}$, bzw. folgt durch Transposition dieser Gleichung

$$\mathbf{G}^T\mathbf{d} = \mathbf{c}.$$

Man kann $\mathbf{c}$ also als Linearkombination der Zeilen von $\mathbf{G}$ mit Koeffizienten in $\mathbf{d}$ auffassen. Im Vergleich mit Gleichung (9) erhalten wir eine Generatormatrix $\mathbf{G}$ gemäß

$$\mathbf{G}^T = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \ldots & \mathbf{b}_k \end{pmatrix} \qquad \text{bzw.} \qquad \mathbf{G} = \begin{pmatrix} \mathbf{b}_1^T \\ \mathbf{b}_2^T \\ \vdots \\ \mathbf{b}_k^T \end{pmatrix}.$$

Beispiel 6 — Blockcode
Gegeben sei der folgende binäre Blockcode $\mathcal{C}$:

$$\mathcal{C} = \left\{ \underbrace{\begin{pmatrix}0\\1\\1\\1\\1\\0\end{pmatrix}}_{\mathbf{c}_1}, \underbrace{\begin{pmatrix}1\\0\\1\\0\\1\\1\end{pmatrix}}_{\mathbf{c}_2}, \underbrace{\begin{pmatrix}1\\1\\0\\1\\0\\1\end{pmatrix}}_{\mathbf{c}_3}, \underbrace{\begin{pmatrix}1\\1\\1\\1\\1\\1\end{pmatrix}}_{\mathbf{c}_4}, \underbrace{\begin{pmatrix}1\\0\\0\\0\\0\\1\end{pmatrix}}_{\mathbf{c}_5}, \underbrace{\begin{pmatrix}0\\1\\0\\1\\0\\0\end{pmatrix}}_{\mathbf{c}_6}, \underbrace{\begin{pmatrix}0\\0\\1\\0\\1\\0\end{pmatrix}}_{\mathbf{c}_7}, \underbrace{\begin{pmatrix}0\\0\\0\\0\\0\\0\end{pmatrix}}_{\mathbf{c}_0} \right\} = \left\{ \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4, \underbrace{\mathbf{c}_5}_{\mathbf{b}_1}, \underbrace{\mathbf{c}_6}_{\mathbf{b}_2}, \underbrace{\mathbf{c}_7}_{\mathbf{b}_3}, \mathbf{c}_0 \right\}$$

(a) (i) Der Code ist *linear*. Es ist zu zeigen, dass $\mathbf{c}, \mathbf{c}' \in \mathcal{C} \implies \mathbf{c} + \mathbf{c}' \in \mathcal{C}$ also $\mathcal{C}$ gegenüber der Addition abgeschlossen ist. Dafür müsste man streng genommen alle $\binom{8}{2} = 28$ Summen zweier Codewörter durchprobieren und überprüfen, ob sie in $\mathcal{C}$ liegen. Einfacher ist, die Codewörter $\{\mathbf{c}_5, \mathbf{c}_6, \mathbf{c}_7\} = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ als Basis $\mathcal{B}$ von $\mathcal{C}$ zu identifizieren. Alle Codewörter können als Linearkombination ($\mathbf{c} = a_1 \cdot \mathbf{b}_1 + a_2 \cdot \mathbf{b}_2 + a_3 \cdot \mathbf{b}_3,\ a_i \in \mathrm{GF}(2)$) dieser Basisvektoren dargestellt werden:

$$\mathbf{c}_1 = \mathbf{b}_2 + \mathbf{b}_3, \mathbf{c}_2 = \mathbf{b}_1 + \mathbf{b}_3, \mathbf{c}_3 = \mathbf{b}_1 + \mathbf{b}_2, \mathbf{c}_4 = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_3$$
$$\mathbf{c}_5 = \mathbf{b}_1, \mathbf{c}_6 = \mathbf{b}_2, \mathbf{c}_7 = \mathbf{b}_3 \text{ und } \mathbf{c}_0 = \mathbf{0}$$

Weil mit $\{\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_7\}$ alle acht möglichen Linearkombinationen der drei Basisvektoren in $\mathcal{C}$ liegen gilt

$$\begin{aligned}\mathbf{c} + \mathbf{c}' = \mathbf{c} &= a_1 \cdot \mathbf{b}_1 + a_2 \cdot \mathbf{b}_2 + a_3 + \mathbf{c} + a_1' \cdot \mathbf{b}_1 + a_2' \cdot \mathbf{b}_2 + a_3' \cdot \mathbf{b}_3 \\ &= \underbrace{(a_1 + a_1')}_{a_1^*} \cdot \mathbf{b}_1 + \underbrace{(a_2 + a_2')}_{a_2^*} \cdot \mathbf{b}_2 + \underbrace{(a_3 + a_3')}_{a_3^*} \cdot \mathbf{b}_3 \in \mathcal{C}.\end{aligned}$$

Diese Rechnung zeigt, dass es hinreichend ist eine Basis $\mathcal{B}$ anzugeben, um zu beweisen, dass ein Code $\mathcal{C}$ linear ist.

(ii) Der Code ist *nicht zyklisch*. Wäre er zyklisch müsste das zyklische Verschieben eines Codevektors wieder einen Codevektor ergeben. Doch verschiebt man beispielsweise $\mathbf{c}_1$ zyklisch um eine Stelle, so ist der resultierende Vektor nicht mehr im Code.

(iii) Die Codewörter haben 6 Bits daher ist $n = 6$. Durch die 8 Codewörter können $k = 3$ Nutzbit pro Codewort übertragen werden. Die Coderate ergibt sich demnach zu $R = \frac{k}{n} = \frac{3}{6} = \frac{1}{2}$.

(b) Man nimmt eine beliebige Basis des Codes, beispielsweise $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$, um die Zeilen der Generatormatrix $\mathbf{G}$ zu bilden:

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} \underbrace{1 & 0 & 0}_{\mathbf{I}_3} & \underbrace{0 & 1 & 0}_{\mathbf{P}} \end{pmatrix} \tag{10}$$

Die Checkmatrix

$$\mathbf{H} = \begin{pmatrix} \mathbf{h}_1^T \\ \vdots \\ \mathbf{h}_{n-k}^T \end{pmatrix} \in \mathrm{GF}(2)^{n-k \times n}$$

besteht aus $n - k$ linear unabhängigen Zeilen, sodass für alle Codevektoren $\mathbf{c} \in \mathcal{C}$ und alle $j \in \{1, 2, \ldots, n-k\}$ gilt $\mathbf{h}_j^T \mathbf{c} = 0$, bzw. in Matrixschreibweise $\mathbf{Hc} = \mathbf{0}$. Damit müssen die Vektoren $\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{n-k}\}$ im *orthogonalen Komplement* von $\mathcal{C}$,

$$\mathcal{C}^\perp \triangleq \left\{ \mathbf{h} \in \mathrm{GF}(2)^n \;\middle|\; \mathbf{h}^T \mathbf{c} = 0 \; \forall \mathbf{c} \in \mathcal{C} \right\}$$

liegen. Ähnlich wie bei der Generatormatrix $\mathbf{G}$, kann jede Basis[3] von $\mathcal{C}^\perp$ als Zeilen der Checkmatrix $\mathbf{H}$ verwendet werden. Nachdem die Generatormatrix in Standardform vorliegt kann $\mathbf{H}$ gewählt werden, gemäß

$$\mathbf{H} = \left(\begin{array}{c|c} \mathbf{P}^T & \mathbf{I}_3 \end{array} \right) = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Die Matrizen $\mathbf{G}$ und $\mathbf{H}$ sind jedoch nicht eindeutig, da $\mathcal{C}$ und $\mathcal{C}^\perp$ viele verschiedene Basen besitzen. Man kann z.B. eine Generatormatrix $\mathbf{G}'$ aus der Basis $\{\mathbf{c}_4, \mathbf{c}_1, \mathbf{c}_2\}$ bilden. Um eine weitere Checkmatrix zu erhalten kann man beispielsweise die erste zur dritten Zeile von $\mathbf{H}$

[3] Man beachte, dass $\mathcal{C}^\perp$ ein Vektorraum ist und gemäß dem Dimensionssatz die Dimension $n - k$ besitzt.

addieren. Das ergibt

$$\mathbf{G}' = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \text{ und} \tag{11}$$

$$\mathbf{H}' = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Für jede Generatormatrix $\mathbf{G}$ und Checkmatrix $\mathbf{H}$ gilt jedoch $\mathbf{H}\mathbf{G}^T = \mathbf{0}$. Damit gilt auch $\mathbf{H}\mathbf{G}^T = \mathbf{H}'\mathbf{G}'^T = \mathbf{H}'\mathbf{G}^T = \mathbf{H}\mathbf{G}'^T = \mathbf{0}$. Das kommt daher, dass die Checkmatrix alle Codewörter auf $\mathbf{0}$ abgebildet, was unabhängig von der konkreten Basis ist.

(c) Ob der Code systematisch ist oder nicht hängt von der Wahl von $\mathbf{G}$ ab. Wird $\mathbf{G}$ wie in Gleichung (10) gewählt, handelt es sich um einen systematischen Code, da die Datenbits in den Codewörtern vorkommen (Einheitsmatrix $\mathbf{I}_3$ in $\mathbf{G}$). Die Generatormatrix $\mathbf{G}'$ aus Gleichung (11) erzeugt den Code $\mathcal{C}$ nicht systematisch – die Datenbits kommen nicht direkt in den Codewörtern vor.

(d) Ja. Siehe Gleichung (10).

(e) Da es sich bei $\mathcal{C}$ um einen linearen Code handelt, gilt $d_{\min} = w_{\min} = 2$. Das minimale Gewicht $w_{\min}$ lässt sich leicht aus den Codewörtern ablesen.

(f) Weil $d_{\min} = 2$ ist können $\lfloor \frac{d_{\min}-1}{2} \rfloor = 0$ beliebig angeordnete Bitfehler korrigiert und $d_{\min} - 1 = 1$ Bitfehler erkannt werden.

Da im vorliegenden Fall nicht einmal ein beliebig angeordneter Bitfehler korrigiert werden kann, dekodieren wir nach dem kleinsten Hammingabstand. Der Vektor $\mathbf{r}_1$ hat sowohl von $\mathbf{c}_0$ als auch von $\mathbf{c}_5$ den gleichen Hammingabstand $\mathrm{d} = 1$. Er ist daher nicht eindeutig dekodierbar. Das empfangene Codewort $\mathbf{r}_2$ entspricht dem Codewort $\mathbf{c}_1$ und ist daher als dieses zu dekodieren. Der Empfangsvektor $\mathbf{r}_3$ kann, ähnlich wie $\mathbf{r}_1$, als $\mathbf{c}_1$ oder $\mathbf{c}_7$ interpretiert werden und ist daher ebenfalls nicht eindeutig dekodierbar.

Beispiel 7

(a) Die Checkmatrix $\mathbf{H}$ ist eine $[(n-k) \times n]$-Matrix und hat die Standardform

$$\mathbf{H} = \left(\begin{array}{c|c} \mathbf{P}^T & \mathbf{I}_{n-k} \end{array} \right)$$

Durch einen Vergleich mit der Angabe

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

folgt unmittelbar $k = 4$ und $n = 7$. Die Coderate ergibt sich aus der bekannten Definition $R = \frac{n}{k} = \frac{7}{4}$.

(b) Die Generatormatrix $\mathbf{G}$ ist eine $[n \times k]$-Matrix und in Standardform definiert durch

$$\mathbf{G} = \left(\begin{array}{c|c} \mathbf{I}_k & \mathbf{P} \end{array} \right)$$

Wegen Punkt (a) ergibt sich

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \\ \mathbf{g}_4 \end{pmatrix}.$$

Wenn der (n, k)-Code $\mathcal{C}$ als linearer Vektorraum aufgefasst wird, dann bilden $\mathbf{g}_1$ bis $\mathbf{g}_4$ eine mögliche Basis. Die Checkmatrix bildet jedes Codewort auf $\mathbf{0}$ ab, insbesondere auch die Basisvektoren. Damit gilt

$$\mathbf{H}\mathbf{G}^T = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \mathbf{0}_{n-k,k}$$

(c) Ein Code ist systematisch, wenn die Informationsbits explizit als Klartext im Codewort vorkommen. Der Code in diesem Beispiel ist somit *systematisch*. Ein Code ist zyklisch, wenn jedes zyklische Verschieben eines Codeworts wieder ein Codewort ergibt. Daher ist der Code in diesem Beispiel *nicht zyklisch*.

(d) Da der Code linear ist gilt die Gleichheit zwischen minimaler Hammingdistanz und minimalem Hamminggewicht $d_{\min} = w_{\min}$. In Tabelle 3 sind alle Codewörter mit zugehörigem Hamminggewicht aufgelistet. Es ist unmittelbar erkennbar, dass $d_{\min} = w_{\min} = 3$ gilt.

(e) Es können $d_{\min} - 1 = 2$ Fehler erkannt und $\left\lfloor \frac{d_{\min}-1}{2} \right\rfloor = 1$ Fehler korrigiert werden.

(f) Die Gleichung $\mathbf{H}\mathbf{c} = \mathbf{0}$ gilt für jeden Codevektor $\mathbf{c} \in \mathcal{C}$. Daher ist die Menge der Empfangswörter $\mathbf{r}$, für die gilt $\mathbf{H}\mathbf{r} = \mathbf{0}$, der Code selbst. Für alle Empfangswörter, für die $\mathbf{H}\mathbf{r} \neq \mathbf{0}$ gilt, wird das Syndrom $\mathbf{s}_n = \mathbf{H}\mathbf{r} = \mathbf{H}(\hat{\mathbf{r}} + \mathbf{e}_n) = \mathbf{H}\mathbf{e}_n$ ermittelt um dann das korrigierte Codewort $\hat{\mathbf{r}} = \mathbf{r} + \mathbf{e}_n$ zu berechnen. Die Symdromtabelle $\mathbf{S}$ erhält man für $t = 1$ gemäß

$$\mathbf{S} = \begin{pmatrix} \mathbf{0}^T \\ \mathbf{H}^T \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{s}_0^T \\ \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \mathbf{s}_3^T \\ \mathbf{s}_4^T \\ \mathbf{s}_5^T \\ \mathbf{s}_6^T \\ \mathbf{s}_7^T \end{pmatrix}. \tag{12}$$

	I_1	I_2	I_3	I_4	P_1	P_2	P_3	$w(c_i)$
c_0	0	0	0	0	0	0	0	0
c_1	0	0	0	1	0	1	1	3
c_2	0	0	1	0	1	0	1	3
c_3	0	0	1	1	1	1	0	4
c_4	0	1	0	0	1	1	0	3
c_5	0	1	0	1	1	0	1	4
c_6	0	1	1	0	0	1	1	4
c_7	0	1	1	1	0	0	0	3
c_8	1	0	0	0	1	1	1	4
c_9	1	0	0	1	1	0	0	3
c_{10}	1	0	1	0	0	1	0	3
c_{11}	1	0	1	1	0	0	1	4
c_{12}	1	1	0	0	0	0	1	3
c_{13}	1	1	0	1	0	1	0	4
c_{14}	1	1	1	0	1	0	0	4
c_{15}	1	1	1	1	1	1	1	7

Tabelle 3: Beispiel 7, Punkt (d)

Das Empfangswort $\mathbf{r}_1 = (1,1,1,1,1,1,1)^T$ ist mit Hilfe von Tabelle 3 direkt identifizierbar als $\mathbf{c}_{15}$ und es gilt $\mathbf{Hr}_1 = \mathbf{0}$. Die Empfangswörter $\mathbf{r}_2 = (0,0,0,0,1,1,0)^T$ und $\mathbf{r}_3 = (1,0,1,0,1,0,1)^T$ werden mittels Checkmatrix $\mathbf{H}$ und Syndromtabelle $\mathbf{S}$ decodiert.

$$\mathbf{Hr}_2 = \begin{pmatrix}1\\1\\0\end{pmatrix} = \mathbf{s}_2 \quad \Longrightarrow \quad \mathbf{e}_2^T = \begin{pmatrix}0 & 1 & 0 & 0 & 0 & 0 & 0\end{pmatrix}$$

$$\mathbf{Hr}_3 = \begin{pmatrix}1\\1\\1\end{pmatrix} = \mathbf{s}_1 \quad \Longrightarrow \quad \mathbf{e}_1^T = \begin{pmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}$$

Somit erhält man folgende Codewörter:

$$\hat{\mathbf{r}}_2 = \mathbf{r}_2 + \mathbf{e}_2 = \begin{pmatrix}0 & 1 & 0 & 0 & 1 & 1 & 0\end{pmatrix}^T = \mathbf{c}_4$$
$$\hat{\mathbf{r}}_3 = \mathbf{r}_3 + \mathbf{e}_1 = \begin{pmatrix}0 & 0 & 1 & 0 & 1 & 0 & 1\end{pmatrix}^T = \mathbf{c}_2.$$

(g) (Bonus) Nein, ein solches Empfangswort kann nicht existieren, da die Syndromtabelle in Gleichung (12) alle möglichen Vektoren der Länge 3 enthält. Daher kann immer ein entsprechender Fehlervektor für jedes mögliche Syndrom gefunden werden.

Beispiel 8

(a) **Code $\mathcal{C}$:** Mit $n = 4$ und $k = 2$ ergibt sich $R = \frac{k}{n} = \frac{1}{2}$. Es existieren $2^k = 4$ Codewörter und diese sind

$$\mathcal{C} = \left\{ \underbrace{\begin{pmatrix}0\\0\\0\\0\end{pmatrix}}_{\mathbf{c}_0}, \underbrace{\begin{pmatrix}1\\0\\1\\1\end{pmatrix}}_{\mathbf{c}_1}, \underbrace{\begin{pmatrix}0\\1\\1\\0\end{pmatrix}}_{\mathbf{c}_2}, \underbrace{\begin{pmatrix}1\\1\\0\\1\end{pmatrix}}_{\mathbf{c}_3} \right\}.$$

Code $\mathcal{C}'$: Mit $n = 6$ und $k = 2$ ergibt sich $R = \frac{k}{n} = \frac{1}{3}$. Es existieren $2^k = 4$ Codewörter und diese sind

$$\mathcal{C} = \left\{ \underbrace{\begin{pmatrix}0\\0\\0\\0\\0\\0\end{pmatrix}}_{\mathbf{c}_0}, \underbrace{\begin{pmatrix}1\\0\\1\\1\\0\\1\end{pmatrix}}_{\mathbf{c}_1}, \underbrace{\begin{pmatrix}0\\1\\1\\0\\1\\1\end{pmatrix}}_{\mathbf{c}_2}, \underbrace{\begin{pmatrix}1\\1\\0\\1\\1\\1\end{pmatrix}}_{\mathbf{c}_3} \right\}.$$

(b) **Code $\mathcal{C}$:** Ja, der Code ist systematisch, da $c_1 = d_1$ und $c_2 = d_2$. Ja, der Code ist linear, da die Addition von 2 beliebigen Codewörtern wieder ein Codewort ergibt, außerdem besitzt der Code die Generatormatrix

$$\mathbf{G} = \begin{pmatrix}1 & 0 & 1 & 1\\ 0 & 1 & 1 & 0\end{pmatrix}$$

Code $\mathcal{C}'$: Ja, der Code ist systematisch, da $c_1 = d_1$ und $c_2 = d_2$. Nein, der Code ist nicht linear, da $\mathbf{c}_1 + \mathbf{c}_2 \notin \mathcal{C}$.

(c) **Code $\mathcal{C}$:** Da der Code nach Punkt (b) linear ist gilt $d_{\min} = w_{\min} = \min(w(\mathbf{c}_1), w(\mathbf{c}_2), w(\mathbf{c}_3)) = \min(3, 2, 3) = 2$.

Code $\mathcal{C}'$: Es gilt $w_{\min} = \min(w(\mathbf{c}_1), w(\mathbf{c}_2), w(\mathbf{c}_3)) = \min(4, 4, 5) = 4$. Jedoch ist der Code nach Punkt (b) nicht linear, weshalb im Allgemeinen $w_{\min} \neq d_{\min}$. Daher wird $d_{\min}$ durch Vergleichen aller Codewörter berechnet:

$$d_{\min} = \min_{i \neq j} d(\mathbf{c}_i, \mathbf{c}_j)$$

In Tabelle 4 sind die Hamming-Distanzen angegeben. Es folgt $d_{\min} = 3$.

$d(\mathbf{c}_i, \mathbf{c}_j)$	$i = 0$	$i = 1$	$i = 2$	$i = 3$
$j = 0$	0	4	4	5
$j = 1$	4	0	4	3
$j = 2$	4	4	0	3
$j = 3$	5	3	3	0

Tabelle 4: Hamming-Distanzen

(d) **Code $\mathcal{C}$:** Es kann $d_{\min} - 1 = 1$ Bitfehler erkannt und $\left\lfloor \frac{d_{\min}-1}{2} \right\rfloor = 0$ Bitfehler korrigiert werden.

Code $\mathcal{C}'$: Es können $d_{\min} - 1 = 2$ Bitfehler erkannt und $\left\lfloor \frac{d_{\min}-1}{2} \right\rfloor = 1$ Bitfehler korrigiert werden.

(e) **Code $\mathcal{C}$:** Es gilt $d(\mathbf{c}_0, \mathbf{r}_1) = 2$, $d(\mathbf{c}_1, \mathbf{r}_1) = 1$, $d(\mathbf{c}_2, \mathbf{r}_1) = 2$, $d(\mathbf{c}_3, \mathbf{r}_1) = 3$. Damit kann dem Empfangswort $\mathbf{r}_1$ eindeutig das Codewort $\mathbf{c}_1$ zugeordnet werden, von dem es minimalen Hammingabstand hat.

Code $\mathcal{C}'$: Es gilt $d(\mathbf{c}_0, \mathbf{r}'_2) = 2$, $d(\mathbf{c}_1, \mathbf{r}'_2) = 4$, $d(\mathbf{c}_2, \mathbf{r}'_2) = 2$, $d(\mathbf{c}_3, \mathbf{r}'_2) = 5$. Die Dekodierung ist somit nicht eindeutig, da sowohl $\mathbf{c}_0$, wie auch $\mathbf{c}_2$ den gleichen, minimalen Hammingabstand von $\mathbf{r}'_2$ haben.

(f) **Code $\mathcal{C}$:** Da 1 Fehler erkannt werden kann (vgl. Punkt (d)) berechnen wir die Wahscheinlichkeit, dass mehr als 1 Fehler auftritt (vgl. Beispiel 3, Punkt (e)):

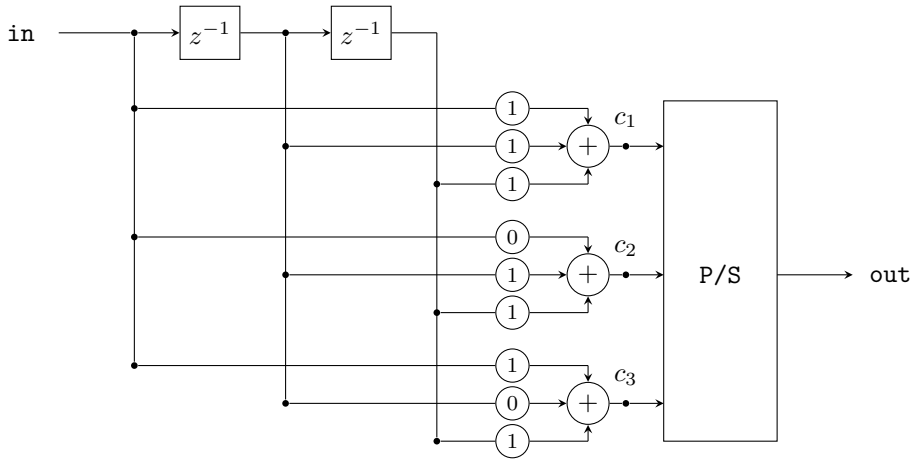
$$P_e = 1 - (1 - P_b)^4 - 4(1 - P_b)^3 P_b = 5.92 \cdot 10^{-4}$$

Code $\mathcal{C}'$: Da 2 Fehler erkannt werden können (vgl. Punkt (d)) berechnen wir die Wahscheinlichkeit, dass mehr als 2 Fehler auftreten (vgl. Beispiel 3, Punkt (e)):

$$P_e = 1 - (1 - P_b)^6 - 6(1 - P_b)^5 P_b - 15(1 - P_b)^4 P_b^2 = 1.955 \cdot 10^{-5}$$

Beispiel 9 — Faltungscode

(a) Die Ordnung der x in p_i bestimmt die Verzögerung. Den Codierer könnte man z. B. so bauen:

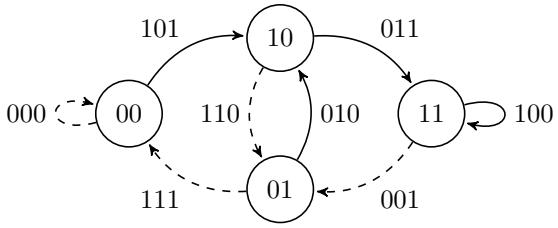


(b) (i) Der Codierer liefert für jedes neue Bit 3 codierte Bits. Die Coderate ist daher $\frac{1}{3}$.

(ii) Der Faltungsencoder besteht aus 3 FIR-Filtern und dem Parallel-Seriell-Wandler. Bei beiden handelt es sich um lineare Systeme, sodass der gesamte Encoder als Zusammenschaltung ebenfalls ein lineares System bildet.

(iii) Der Name Faltungscode kommt daher, dass durch die FIR-Filter die Eingangsbits mit den Polynomen p_i gefaltet werden.

(c) Das Zustandsdiagramm des Codierers könnte so aussehen:



Die durchgezogenen Linien zeigen die Übergänge bei dem Input 1 und die strichlierten Linien Zustandswechsel bei einem 0-Bit.

(d) Mit dem Zustandsdiagramm aus c) lassen sich die beiden Bitfolgen codieren. Abb. 4 zeigt den Encodiervorgang für beide Folgen im Trellisdiagramm. Es ergibt sich:

$$111 \rightarrow 101\,011\,100$$
$$11100 \rightarrow 101\,011\,100\,001\,111$$

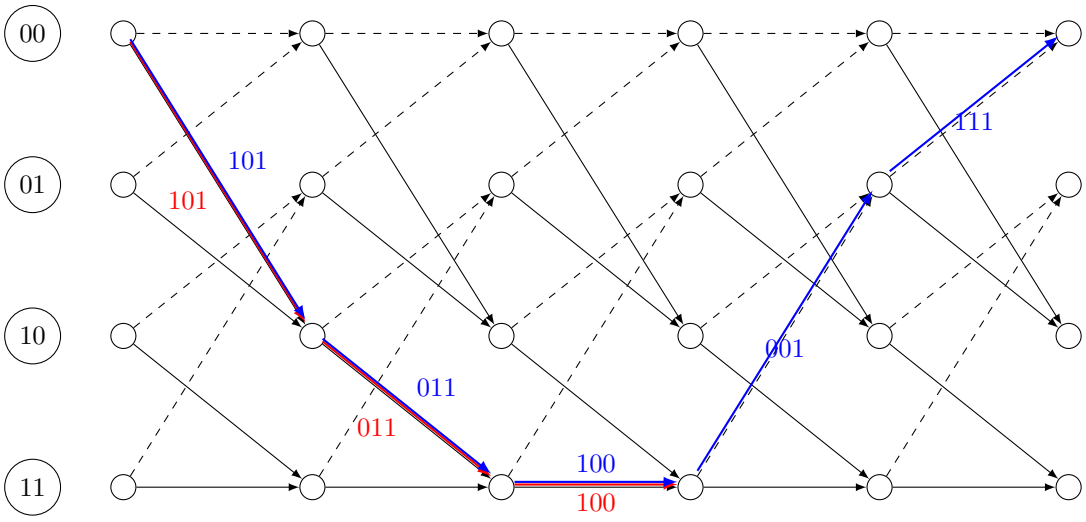


Abbildung 4: Trellisdiagramm des Encodiervorgangs für 9d. Eingangsfolgen: 111 und 11100 ausgehend vom Nullzustand.

(e) Zur Decodierung verwendet man das in Abb. 5 dargestellte Trellisdiagramm. Ausgehend vom Nullzustand werden alle mögliche Wege verfolgt und die Ausgabe bei der jeweiligen Zustandsänderung mit den empfangenen Bits verglichen. Der Hammingabstand dieser beiden Werte wird berechnet und zur bisher geringsten aufsummierten Metrik addiert. Für Abb. 5 stehen an erster Stelle die Metrik für den „oberen“ Pfad an 2. Stelle der des „unteren“ Pfades.

Da angenommen wurde, dass das Codewort mit zwei Nullen erweitert wurde, muss der Endzustand der Nullzustand sein. Von diesem bewegt man sich entlang der geringsten Metrik durch das Trellisdiagramm und erhält so die wahrscheinlichste gesendete Bitfolge: `101 011 100 001 111`. Unter der Annahme dass die wahrscheinlichste Bitfolge die gesendete Bitfolge ist, wurden 4 Fehler, entsprechend der geringsten Metrik, korrigiert.

Über das Zustandsdiagramm erkennt man, dass diese gesendete Bitfolge den uncodierten Bits `11100` entspricht. Die letzten beiden Nullen wurden angefügt, um den Encoder in den Nullzustand zu versetzen, daher sind die eigentlichen Datenbits `111`.

(f) Würde man das Codewort nicht mit zwei Nullen abschließen, wäre der Encoder nach der Übertragung nicht notwendigerweise im Nullzustand. Daher müsste man alle Metriken der letzten Spalte im Trellisdiagramm vergleichen und den Pfad mit der niedrigsten aufsummierten Hammingdistanz wählen. Im gegebenen Beispiel besitzen alle vier Knoten die gleiche Metrik und es wäre von jedem ein Pfad möglich. Es würden sich also neben dem Pfad aus e) drei weitere Pfade ergeben, welche analog zu davor zu dekodieren wären - ein denkbar ungünstiger Fall...

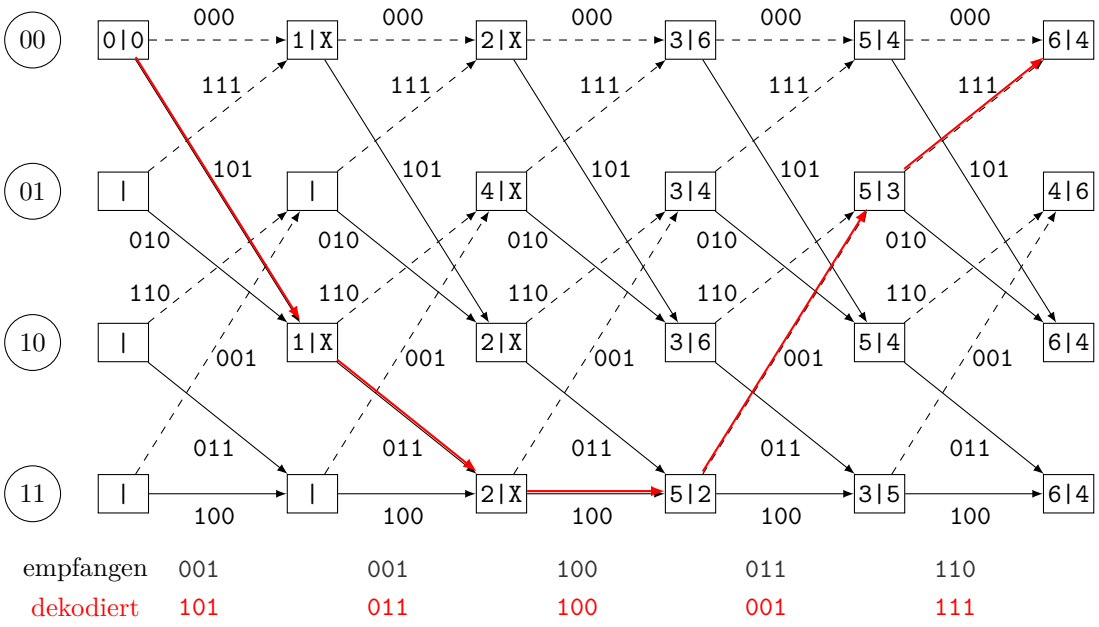


Abbildung 5: Trellisdiagramm des Decodiervorgangs für 9e. Für die Bits `001 001 100 011 110`

Beispiel 10 — Komplexität von Kanalcodierungsverfahren

(a) Für die Berechnung des Syndroms muss $\mathbf{Hr} = \mathbf{s} \in \mathrm{GF}(2)^{n-k}$ ausgewertet werden und anschließend $\mathbf{s}$ in der Syndromtabelle gesucht werden. D.h. für jedes der $(n-k)$ Elemente von $\mathbf{s}$ müssen n Multiplikationen und $n-1$ Additionen durchgeführt werden, insgesamt also $(n-k)n = (1-R)n^2 \propto n^2$ Multiplikationen und $(n-k)(n-1) \propto n^2$ Additionen pro Empfangswort der Länge n.

Mit dem minimalem Abstand $d_{\min}$ können $t = \left\lfloor \frac{d_{\min}-1}{2} \right\rfloor$ Fehler korrigiert werden. Die Syndromtabelle

$$\mathcal{S} = \left\{ \mathbf{He} \;\middle|\; \mathbf{e} \in \mathrm{GF}(2)^n : \mathrm{w}(\mathbf{e}) \leq t \right\}$$

die im Decodierer gespeichert sein muss besteht aus

$$|\mathcal{S}| = \sum_{k=0}^{t} \binom{n}{k}$$

Elementen, da alle Fehlermuster $\mathbf{e}$ mit $\mathrm{w}(\mathbf{e}) \leq t$ abgedeckt werden müssen und es für k Fehler in einem Wort der Länge n genau $\binom{n}{k}$ Fehlermuster gibt.

(b) Da $f(\cdot)$ bijektiv ist folgt $|\mathcal{C}| = M$. Um M Werte darzustellen werden $\log_2 M$ Bits benötigt. Das führt auf die Rate $R = \frac{\log_2 M}{n}$. Für die Dekodierung von $\mathbf{r}$ wird also für jedes der $M = 2^{Rn}$ Codewörter $\mathbf{c} \in \mathcal{C}$ die Hamming Distanz zum Empfangswort $\mathbf{r}$ berechnet ($2n-1$ Additionen pro Codewort), insgesamt also $(2n-1)2^{Rn}$ Additionen/Empfangswort. Bei dieser Art der Decodierung müssen alle $|\mathcal{C}| = 2^{Rn}$ Codewörter im Decodierer gespeichert sein, d.h. sowohl Speicher- als auch Rechenaufwand sind *exponentiell* in n.

$\mathbf{y}$	-3.22		2.001		1.04		2.6	
$\hat{\mathbf{x}}$	-3		3		1		3	
$\hat{\mathbf{b}}$	0	0	1	0	1	1	1	0
$\mathbf{l}$	-177.6	-48.8	80.75	-0.04	41.6	38.4	128	-24

Tabelle 5: Empfangssymbole $\mathbf{y}$ mit Zugehörigen detektierten Sendesymbolen $\hat{\mathbf{x}}$ und detektierten Bits $\hat{\mathbf{b}}$ und LLRs $\mathbf{l}$.

(c) Für (a) ergibt sich mit $n = 1000$ und $R = 0.7$

$(n-k)n \approx (n-k)(n-1) \approx (1-R)n^2 = 30\,000$ Additionen und Multiplikationen pro Empfangswort

und mit $d_{\min} = 3$ (entsprechend $t = 1$) ergibt sich die Anzahl der Syndrome zu

$$|\mathcal{S}| = 1 + n = 1001 \quad \text{d.h.} \quad (n-k)|\mathcal{S}| \approx n^2(1-R) = 30\,000 \text{ Bit}$$

bzw. mit $d_{\min} = 30$ (entsprechend $t = 14$)

$$|\mathcal{S}| = \sum_{k=0}^{14} \binom{n}{k} = 1.0619 \cdot 10^{31} \quad \text{d.h.} \quad (n-k)|\mathcal{S}| = 3.1857 \cdot 10^{33} \text{ Bit} \tag{13}$$

Für (b) ergibt sich mit $n = 1000$, $R = 0.7$

$$(2n-1)2^{Rn} \approx 1.0515 \cdot 10^{214} \text{ Additionen pro Empfangswort}$$

Ausserdem muss der gesamte Code im Speicher der Decodierers zur Verfügung stehen, d.h. der Speicheraufwand beträgt $n\,2^{Rn} \approx 5.2601 \cdot 10^{213}$ Bit $\approx 5.9801 \cdot 10^{200}$ TB.

Wir sehen also, dass das Verfahren aus (b) gänzlich ungeeignet für praktische Anwendungen ist, da hier eine mathematische Struktur, welche effizientes Decodieren ermöglichen würde, fehlt. Der Rechenaufwand für die Syndromdecodierung aus (a) ist quadratisch in n. Der Speicheraufwand ist für $t = 1$ ebenfalls quadratisch in n, für größere $d_{\min}$ kommt es aber wegen der möglichen Kombinationen an Fehlermustern zu einem rasanten Anstieg des Speicherbedarfs, sodass auch dieses Verfahren in der Praxis nicht eingesetzt wird.

Beispiel 11 — Bitmapping und Softdecoding

(a) Mit $\mathbb{E}\big[\mathsf{X}^2\big] = \frac{1}{4}\big((-3)^2 + (-1)^2 + 1^2 + 3^2\big) = 5$ und $\mathbb{E}\big[\mathsf{N}^2\big] = \sigma^2$ folgt aus

$$\text{SNR} = 10\log_{10}\left(\frac{\mathbb{E}\big[\mathsf{X}^2\big]}{\mathbb{E}[\mathsf{N}^2]}\right)\text{dB} \stackrel{!}{=} 20\,\text{dB}$$

$\sigma \approx 0.2236$. Die gewichteten bedingten Dichten sowie die Entscheidungregionen $\mathcal{R}_x$ sind in Abb. 6 skizziert.

(b) Die detektierten Sendesymbole $\hat{\mathbf{x}}$ ergeben sich aus den Empfangssymbolen $\mathbf{y}$ über die Entscheidungsregionen aus Abb. 6. Es sei $f(b_1, b_2) = x$ die in der Angabe gegebene, bijektive Abbildung zweier Bits auf die Sendesymbole x. Die detektierte Bitfolge ergibt sich durch anwenden von $f^{-1}(\cdot)$ auf die Elemente von $\hat{\mathbf{x}}$. Die Ergebnisse sind in Tabelle 5 angeführt.

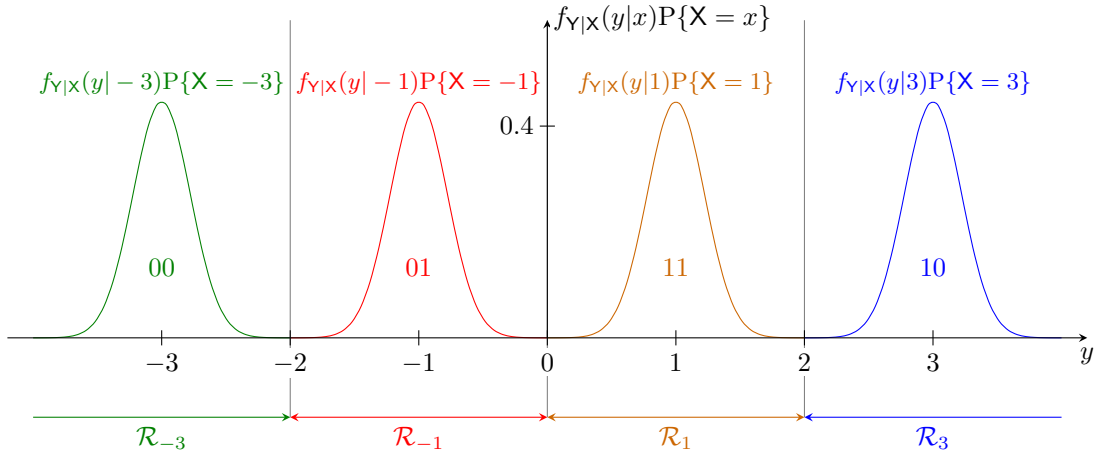


Abbildung 6: Gewichtete bedingte Dichten mit Entscheidungsregionen

(c) Die LLRs sind definiert als

$$L_i^{(k)} \triangleq \log \frac{\mathrm{P}\left\{\mathsf{B}_i^{(k)}=1|y_i\right\}}{\mathrm{P}\left\{\mathsf{B}_i^{(k)}=0|y_i\right\}}$$

Dementsprechend können folgende Extremfälle unterschieden Werden

- $\mathrm{P}\left\{\mathsf{B}_i^{(k)}=1|y_i\right\} \gg \mathrm{P}\left\{\mathsf{B}_i^{(k)}=0|y_i\right\}$, entsprechend einem stark positiven LLR Wert: Bei bekanntem Kanalausgang y_i, ist es sehr viel wahrscheinlicher, dass das Bit $b_i^{(k)}$ Eins war.
- $\mathrm{P}\left\{\mathsf{B}_i^{(k)}=1|y_i\right\} \ll \mathrm{P}\left\{\mathsf{B}_i^{(k)}=0|y_i\right\}$, entsprechend einem stark negativen LLR Wert: Bei bekanntem y_i, ist es sehr viel wahrscheinlicher, dass das Bit $b_i^{(k)}$ Null war.
- $\mathrm{P}\left\{\mathsf{B}_i^{(k)}=1|y_i\right\} \approx \mathrm{P}\left\{\mathsf{B}_i^{(k)}=0|y_i\right\}$, entsprechend einem LLR Wert nahe bei Null: Bei bekanntem y_i, sind die Wahrscheinlichkeiten, dass $b_i^{(k)}$ Null bzw. Eins war ungefähr gleich groß. Dementsprechend kommt für diesen Fall eine Entscheidung zugunsten eines Wertes einem Münzenwurf gleich, eine zuverlässige Detektion ist nicht gewährleistet.

Zusammenfassend lässt sich also sagen, dass das Vorzeichen der LLRs den Wert der Bits widerspiegeln, während der Betrag die Verlässlichkeit der Entscheidung beschreibt.

(d) Zunächst leiten wir der Vollständigkeit halber den gegebenen Ausdruck

$$\log \frac{\mathrm{P}\left\{\mathsf{B}_i^{(k)}=1|y_i\right\}}{\mathrm{P}\left\{\mathsf{B}_i^{(k)}=0|y_i\right\}} = \log \frac{\sum_{x:b_k=1} f_{\mathsf{Y}|\mathsf{X}}(y_i|x)}{\sum_{x:b_k=0} f_{\mathsf{Y}|\mathsf{X}}(y_i|x)}$$

her.

Schreiben wir $\mathcal{X}_{k,b} = \{x_1^{(k,b)}, \ldots, x_{|\mathcal{X}_{k,b}|}^{(k,b)}\}$ für die Menge aller Symbole, für die dass $k$te Bit gleich b ist, so haben wir die folgenden äquivalenten Ereignisse:

$$\mathsf{B}^{(k)} = b \iff \mathsf{X} \in \mathcal{X}_{k,b} \iff (\mathsf{X} = x_1^{(k,b)}) \vee (\mathsf{X} = x_2^{(k,b)}) \cdots \vee (\mathsf{X} = x_{|\mathcal{X}_{k,b}|}^{(k,b)}).$$

Daraus folgt, dass

$$\frac{\mathrm{P}\left\{\mathsf{B}_i^{(k)} = 1|y_i\right\}}{\mathrm{P}\left\{\mathsf{B}_i^{(k)} = 0|y_i\right\}} = \frac{\mathrm{P}\left\{\bigvee_{x\in\mathcal{X}_{k,1}} \mathsf{X} = x \,\middle|\, y_i\right\}}{\mathrm{P}\left\{\bigvee_{x\in\mathcal{X}_{k,0}} \mathsf{X} = x \,\middle|\, y_i\right\}} = \frac{\sum_{x\in\mathcal{X}_{k,1}} \mathrm{P}\{\mathsf{X} = x|y_i\}}{\sum_{x\in\mathcal{X}_{k,0}} \mathrm{P}\{\mathsf{X} = x|y_i\}}.$$

Ähnlich dem Gesetz von Bayes

$$\mathrm{P}\{\mathcal{X}|\mathcal{Y}\} = \frac{\mathrm{P}\{\mathcal{Y}|\mathcal{X}\}\,\mathrm{P}\{\mathcal{X}\}}{\mathrm{P}\{\mathcal{Y}\}},$$

gilt

$$\mathrm{P}\{\mathsf{X} = x|y\} = p_{\mathsf{X}|\mathsf{Y}}(x|y) = \frac{f_{\mathsf{Y}|\mathsf{X}}(y|x)\,p_{\mathsf{X}}(x)}{f_{\mathsf{Y}}(y)},$$

womit unter der Annahme gleichwahrscheinlicher Sendesymbole $p_{\mathsf{X}}(x) = \frac{1}{|\mathcal{X}|}$ der gegebene Ausdruck zur Berechnung der LLRs folgt:

$$L_i^{(k)} = \log\frac{\mathrm{P}\left\{\mathsf{B}_i^{(k)} = 1|y_i\right\}}{\mathrm{P}\left\{\mathsf{B}_i^{(k)} = 0|y_i\right\}} = \log\frac{\sum_{x\in\mathcal{X}_{k,1}} \mathrm{P}\{\mathsf{X} = x|y_i\}}{\sum_{x\in\mathcal{X}_{k,0}} \mathrm{P}\{\mathsf{X} = x|y_i\}} = \log\frac{\sum_{x\in\mathcal{X}_{k,1}} f_{\mathsf{Y}|\mathsf{X}}(y_i|x)}{\sum_{x\in\mathcal{X}_{k,0}} f_{\mathsf{Y}|\mathsf{X}}(y_i|x)}.$$

Konkret ergibt sich für das gegebene Bitmapping mit $\mathcal{X} = \{-3, -1, 1, 3\}$ (vgl. Abb. 6)

$$\begin{aligned}
&\text{1. Bit des Symbols:} \quad \mathcal{X}_{1,0} = \{-3, -1\} \quad \mathcal{X}_{1,1} = \{1, 3\} \\
&\text{2. Bit des Symbols:} \quad \mathcal{X}_{2,0} = \{-3, 3\} \quad\;\; \mathcal{X}_{2,1} = \{-1, 1\}
\end{aligned}$$

und weiter

$$\begin{aligned}
L_i^{(k)} &= \log\frac{\sum_{x:b_k=1} f_{\mathsf{Y}|\mathsf{X}}(y_i|x)}{\sum_{x:b_k=0} f_{\mathsf{Y}|\mathsf{X}}(y_i|x)}, \\
&\Longrightarrow L_i^{(1)} = \log\Big(f_{\mathsf{Y}|\mathsf{X}}(y_i|1) + f_{\mathsf{Y}|\mathsf{X}}(y_i|3)\Big) - \log\Big(f_{\mathsf{Y}|\mathsf{X}}(y_i|-3) + f_{\mathsf{Y}|\mathsf{X}}(y_i|-1)\Big), \\
&\Longrightarrow L_i^{(2)} = \log\Big(f_{\mathsf{Y}|\mathsf{X}}(y_i|-1) + f_{\mathsf{Y}|\mathsf{X}}(y_i|1)\Big) - \log\Big(f_{\mathsf{Y}|\mathsf{X}}(y_i|-3) + f_{\mathsf{Y}|\mathsf{X}}(y_i|3)\Big).
\end{aligned}$$

Tabelle 5 enthält die numerischen Werte. Exemplarisch soll hier die Berechnung der LLRs des Empfangssymboles $y = 2.001$ durchgeführt werden. Ein Vergleich mit Abb. 6 zeigt, dass dieser Wert sehr nahe an der Entscheidungsschwelle zwischen den Symbolen 1 und 3 liegt. Für $L^{(1)}$ erhalten wir

$$L^{(1)} = \log\Big(f_{\mathsf{Y}|\mathsf{X}}(2.001|1) + f_{\mathsf{Y}|\mathsf{X}}(2.001|3)\Big) - \log\Big(f_{\mathsf{Y}|\mathsf{X}}(2.001|-3) + f_{\mathsf{Y}|\mathsf{X}}(2.001|-1)\Big) \approx 80.75$$

sowie für $L^{(2)}$

$$L^{(2)} = \log\Big(f_{\mathsf{Y}|\mathsf{X}}(2.001|-1) + f_{\mathsf{Y}|\mathsf{X}}(2.001|1)\Big) - \log\Big(f_{\mathsf{Y}|\mathsf{X}}(2.001|-3) + f_{\mathsf{Y}|\mathsf{X}}(2.001|3)\Big) \approx -0.04.$$

Interessant ist hier folgendes: während für das erste Bit mit $L^{(1)} \approx 80.75$ relativ sicher auf Eins entschieden werden kann, ist für das zweite Bit mit $L^{(2)} \approx -0.04$ nicht eindeutig, ob es Null oder Eins war. Das kommt daher, dass das erste Bit sowohl für das Symbol 1 als auch 3 gleich Eins

ist (vgl. Abb. 6). Quantitativ lässt sich dieser Effekt über

$$f_{\mathsf{Y}|\mathsf{X}}(2.001|1) + f_{\mathsf{Y}|\mathsf{X}}(2.001|3) \gg f_{\mathsf{Y}|\mathsf{X}}(2.001|-3) + f_{\mathsf{Y}|\mathsf{X}}(2.001|3),$$
$$f_{\mathsf{Y}|\mathsf{X}}(2.001|-1) + f_{\mathsf{Y}|\mathsf{X}}(2.001|1) \approx f_{\mathsf{Y}|\mathsf{X}}(2.001|-3) + f_{\mathsf{Y}|\mathsf{X}}(2.001|3),$$

ausdrücken.

Wie bereits in Punkt (c) erklärt, beinhalten die LLRs nicht nur Information über die detektierten Bits sondern auch über die Verlässlichkeit der Entscheidungen. Diese Information kann an einen nachgeschalteten Decoder weitergegeben werden, und führt zu einer Verbesserung der Decodierung.

Literatur

[1] Ian Glover and Peter Mitchell Grant. *Digital Communications.* Pearson Education Canada, 3rd edition, 2009.