Low-Density-Parity-Check-Codes
— Eine Einführung —

7. Juni 2016

Zusammenfassung

Low-Density-Parity-Check-Codes (LDPC-Codes) sind effiziente Kanalcodierungscodes, welche die Korrektur von Übertragungsfehlern ermöglichen. Sie wurden erstmals 1960 von Gallager in seiner Dissertation beschrieben [Gal63]. Nicht zuletzt wegen des rechentechnischen Aufwandes zur Berechnung dieser Codes konnten sie sich nicht durchsetzen und sind wieder in Vergessenheit geraten. Ausgelöst durch die bahnbrechende Entwicklung der Turbo-Codes von Berrou, Glavieux und Thitimajshima [Ber93], welche eine Codierung nahe der von Claude E. Shannon vorausgesagten Kanalkapazität [Sha48] ermöglichen, wurden die LDPC-Codes wieder entdeckt [Mac96]. Ein wichtiger Baustein für den Erfolg der Turbo-Codes war der Einsatz einer iterativen Decodierung, bei der die Information am Ausgang zweier Decoder in den Eingang des jeweils anderen Decoders rückgekoppelt wird. Diese Art der Decodierung ermöglicht auch bei LDPC-Codes einen leistungsstarken Fehlerschutz.

In diesem Artikel wird anhand eines einfachen Beispiels die grundsätzliche Idee von LDPC-Codes beschrieben. Ein wichtiger Aspekt ist hierbei die Arbeitsweise der iterativen Decodierung. Ein grundlegendes Verständnis für die Kanalcodierung mit linearen Blockcodes wird vorausgesetzt.

1 Allgemeine Beschreibung

Wir betrachten das Beispiel von sechs Datenbits, welche im Block 2×3 angeordnet sind. Sie werden horizontal durch zwei und vertikal durch drei Paritätsbits geschützt

d_1	d_2	d_3	p_1
d_4	d_5	d_6	p_2
p_3	p_4	p_5	

Die Paritätsbits p_1, p_2, p_3, p_4 und p_5 berechnen sich nach folgenden Gleichungen

$$\begin{aligned} d_1 \oplus d_2 \oplus d_3 &= p_1 \\ d_4 \oplus d_5 \oplus d_6 &= p_2 \\ d_1 \oplus d_4 &= p_3 \\ d_2 \oplus d_5 &= p_4 \\ d_3 \oplus d_6 &= p_5 \,. \end{aligned} \tag{1}$$

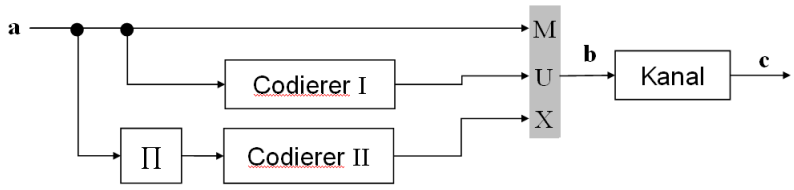

Abbildung 1: Systematischer Encoder mit zwei (eventuell identischen) Codierern. Codierer II verarbeitet die Datenbits in Vektor **a** in permutierter Reihenfolge. Das Kanalcodewort **b** ergibt sich als Verschachtelung der originalen Bits mit den Ausgabe-Bits der beiden Codierer.

Das Zeichen $\oplus$ symbolisiert die Addition modulo 2, d.h., eine XOR-Operation.

Alternativ kann man auch sagen, dass die sechs Datenbits

$$\mathbf{a} = (d_1\ d_2\ d_3\ d_4\ d_5\ d_6)$$

im Codierer I mit zwei Paritätsbits (p_1, p_2) geschützt werden. Parallel dazu werden die Datenbits verschachtelt $(d_1, d_4, d_2, d_5, d_3, d_6)$ und mit Codierer II durch drei Paritätsbits (p_1, p_2, p_3) geschützt (**Abbildung 1**). Als Ergebnis entsteht ein Kanalcodewort

$$\mathbf{b} = (d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ p_1\ p_2\ p_3\ p_4\ p_5)$$

Da die Datenbits direkt aus dem Kanalcodewort abgelesen werden können, heißt dieser Code *systematisch*.

Bei dieser Codierung werden eine konstante Anzahl von l Datenbits algebraisch durch eine konstante Anzahl k von Paritätsbits ergänzt. Es handelt sich um einen linearen Blockcode und die Codierung kann auch mit einer Kontrollmatrix **H** beschrieben werden

$$\mathbf{H} = \left(\mathbf{C}^{\mathrm{T}}|\mathbf{I}\right) = \left(\begin{array}{cccccc|ccccc} d_1 & d_2 & d_3 & d_4 & d_5 & d_6 & p_1 & p_2 & p_3 & p_4 & p_5 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}\right) . \tag{2}$$

Jede Spalte vor der vertikalen Trennlinie entspricht einem Datenbit und jede Spalte hinter der Trennlinie einem Paritätsbit. In diesem Beispiel setzt sich die Kontrollmatrix aus zwei aneinander gereihte Matrizen zusammen, $\mathbf{C}^{\mathrm{T}}$ und eine Einheitsmatrix **I**.

Die Einsen in jeder Zeile aus **H** legen fest, welche Bits an der korrespondierenden Paritätsgleichung aus (1) beteiligt sind.

Das Charakteristische an LDPC-Codes ist nun, dass die Kontrollmatrix **H** sehr wenige Einsen bezogen auf die Gesamtgröße enthält. Man sagt, die Matrix ist dünn besetzt. Daher stammt auch der Name *low density*. Die Paritätsgleichungen (1) enthalten dann im Allgemeinen wenige Elemente und es entsteht ein Vorteil bei der iterativen Decodierung (siehe unten).

Die Beschreibung des Kanalcodes ist auch durch einen zweiteiligen Graphen (*bipartite graph*) möglich, welcher die Datenbits und Paritätsbits miteinander verbindet (**Abbildung 2**). Die oberen Knoten (*variable nodes*) entsprechen den Bits, die übertragen wurden;

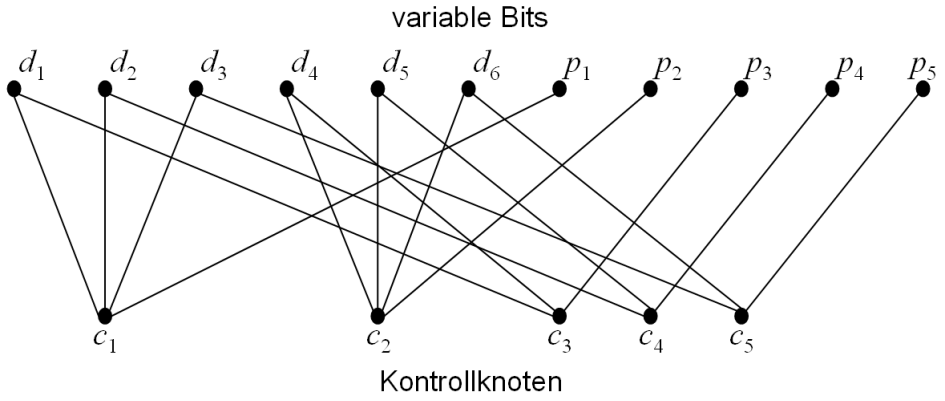

Abbildung 2: Zweiteiliger Graph der Kontrollmatrix $\mathbf{H}$

die unteren Knoten (*check nodes*) fassen die Knoten derjenigen Bits zusammen, deren Summe (modulo 2) Null ergeben müssen. Diese Darstellung wird meist im Zusammenhang mit Low-Density-Parity-Check-Codes (LDPC-Codes) verwendet und Tanner-Graph genannt [Tan81]. Allerdings ist es bei LDPC-Codes üblich, dass jeder Knoten mindestens zwei Kanten hat. In dem oben verwendeten Beispiel haben die Knoten der Paritätsbits nur jeweils eine Kante, weil der Kanalcode systematisch ist.

Im Zusammenhang mit LDPC-Codes macht man meist keine Unterscheidung zwischen Daten- und Paritätsbits. Es muss jedoch sichergestellt sein, dass an jeder Paritätsgleichung auch mindestens ein Paritätsbit beteiligt ist. Um einen Tanner-Graphen mit mindestens zwei Kanten pro Knoten zu generieren, müssen auch die Paritätsbits mindestens zweimal in den Paritätsgleichungen auftreten. In den Gleichungen (1) trifft das jedoch nur auf die Datenbits zu. Man könnte aber jeweils zwei oder mehrere Paritätsgleichungen linear miteinander kombinieren, z.B.

$$\begin{aligned} d_2 \oplus d_3 \oplus d_4 &= p_1 \oplus p_3 \\ d_2 \oplus d_4 \oplus d_6 &= p_2 \oplus p_4 \\ d_1 \oplus d_2 \oplus d_4 \oplus d_5 &= p_3 \oplus p_4 \\ d_2 \oplus d_3 \oplus d_5 \oplus d_6 &= p_4 \oplus p_5 \\ d_1 \oplus d_2 \oplus d_4 \oplus d_5 &= p_1 \oplus p_2 \oplus p_5 \ . \end{aligned} \tag{3}$$

Der entsprechende Graph ist etwas komplexer (**Abbildung 3**) und auch die Kontrollmatrix ist dichter besetzt

$$\mathbf{H}_{\text{mod}} = \left(\begin{array}{cccccc|ccccc} 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right) \ . \tag{4}$$

Aus Sicht des Kanalencodierers sind die Kontrollmatrizen (2) und (4) äquivalent, da beide durch Zeilenkombination in einander überführbar sind. Die Kanalcodewörter können mit derselben Generatormatrix erzeugt werden. Mit Hilfe des Gauß-Jordan-Algorithmus kann (4) wieder in eine Matrix eines systematischen Kanalcodes umgewandelt werden.

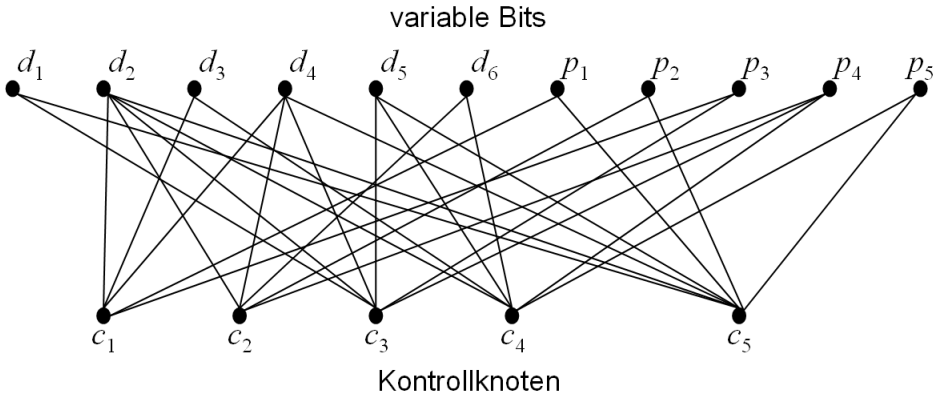

Abbildung 3: Zweiteiliger Graph der Kontrollmatrix $\mathbf{H}_{\text{mod}}$

2 Encodierung

Aus der Kontrollmatrix $\mathbf{H}$ in (2) leitet sich die (kanonische) Generatormatrix

$$\mathbf{G} = (\mathbf{I}|\mathbf{C}) = \left(\begin{array}{cccccc|ccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right) \tag{5}$$

ab.

Der Vektor des Kanalcodewortes $\mathbf{b}$ berechnet sich aus dem Vektor der Datenbits $\mathbf{a}$ und der Generatormatrix $\mathbf{G}$ gemäß

$$\mathbf{b} = \mathbf{a} \otimes \mathbf{G} \; . \tag{6}$$

Das Symbol $\otimes$ entspricht einer (Matrix-)Multiplikation in Modulo-2-Arithmetik.

Bsp.:

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Idealer Weise ist auch die Generatormatrix dünn besetzt und eine Berechnung der Paritätsbits (auch: Kontrollbits) ist unter Umständen durch eine spezielle Arithmetik schneller zu realisieren als mit einer direkten Multiplikation nach (6).

3 Decodierung

3.1 Allgemeine Decodierung mit harter Entscheidung

Am Kanalausgang wird der (kontinuierliche) Signalwert mit einer harten Entscheidung (*hard decision decoding*) durch eine Schwelle als entweder Null-Bit oder Eins-Bit interpretiert und daraus das empfangene Codewort $\mathbf{c}$ zusammengesetzt.

Die Kontrollmatrix $\mathbf{H}$ ist auf der Empfängerseite für das Überprüfen der Fehlerfreiheit dieses empfangenen Codewortes $\mathbf{c}$ erforderlich. Ohne Übertragungsfehler gilt $\mathbf{c} = \mathbf{b}$ und das so genannte Fehlersyndrom

$$\mathbf{s} = \mathbf{H} \otimes \mathbf{c}^{\mathrm{T}} \tag{7}$$

gleicht einem Nullvektor (alle Elemente von $\mathbf{s}$ sind gleich Null). Man könnte auch sagen, dass dann alle Paritätsgleichungen (1) erfüllt sind. Ein oder mehrere Elemente ungleich Null in $\mathbf{s}$ deuten auf mindestens einen Übertragungsfehler hin.

Bei Decodierung mit harter Entscheidung lässt sich mit dem Syndrom $\mathbf{s}$ auf die Position der Bitfehler schließen, vorausgesetzt, die Anzahl der Bitfehler übersteigt nicht die Korrekturkapazität des verwendeten Kanalcodes.

Da $\mathbf{H}_{\mathrm{mod}}$ aus einer Linearkombination der Zeilen aus $\mathbf{H}$ entstanden ist, kann der mit $\mathbf{G}$ erzeugte Kanalcode auch mit $\mathbf{H}_{\mathrm{mod}}$ geprüft werden.

3.2 Iterative Decodierung

Parallele Decodierung

In der gleichen Art und Weise wie die Encodierung im Beispiel aus Abschnitt 1 parallel mit zwei Codieren erfolgte, kann auch die Decodierung als parallel betrachtet werden. Die Paritätsgleichungen werden dementsprechend in zwei Gruppen unterteilt. Die horizontale Linie in der folgenden Beispiel-Kontrollmatrix soll die Aufteilung andeuten

$$\mathbf{H} = \left(\begin{array}{cccccc|ccccc} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}\right) \begin{array}{l} \\ \\ \text{Decodierer I} \\ \\ \\ \text{Decodierer II} \end{array} . \tag{8}$$

Die Information über jedes Datenbit stammt nun aus drei Quellen, wenn es sich um einen systematischen Kanalcode handelt. Die (eventuell verfälschten) Datenbits können aus dem empfangenen Codewort abgelesen werden. Außerdem liefern beide Decoder Hinweise über die gesendeten Datenbits. Es sei an dieser Stelle darauf hingewiesen, dass je nach Anzahl der Einsen in den Spalten der Kontrollmatrix $\mathbf{H}$ auch mehr als zwei Decodierer betrachtet werden können.

Wenn am Kanalausgang keine harte Entscheidung über Null und Eins getroffen wird, sondern die Signalpegel am Ausgang ausgewertet werden, erweist sich eine iterative Decodierung als vorteilhaft, bei der die Information aus diesen drei Quellen ausgetauscht wird.

Nehmen wir das obige Codierungsbeispiel mit konkreten Bit-Werten

$$\left(\begin{array}{ccc|c} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ \hline 1 & 0 & 1 & \end{array}\right) .$$

Bei einer Modulation der Art $x_i = 1 - 2 \cdot d_i$ lautet der Kanalinput $\mathbf{x}$ also

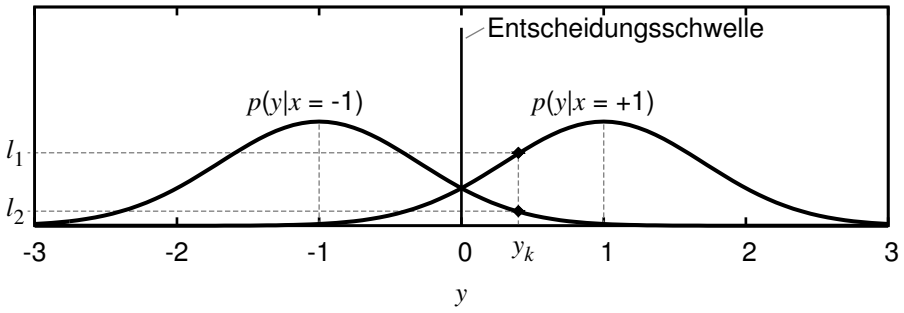

Abbildung 4: Wahrscheinlichkeit eines Wertes y_i am Ausgang eines AWGN-Kanals unter der Bedingung, dass der Signalwert x_i am Kanaleingang entweder gleich -1 oder +1 war.

-1	-1	1	1
1	-1	-1	1
-1	1	-1	

.

Durch Störungen auf dem Kanal erhält man am Kanalausgang Signalwerte $y_i = x_i + \varepsilon$. Für einen AWGN-Kanal (*Additive White Gaussian Noise*) gilt $\varepsilon \sim \mathcal{N}(0;\sigma^2)$. Die Verteilung der Werte am Kanalausgang ist in **Abbildung 4** für $\sigma = 0.7$ dargestellt. Für einen gemessenen Signalwert y_k am Kanalausgang ergeben sich zwei Wahrscheinlichkeiten l_1 und l_2, dass entweder $x_k = +1$ oder $x_k = -1$ am Kanaleingang anlag. Falls gilt $l_1 = p(y|x = +1) > l_2 = p(y|x = -1)$, dann wurde vermutlich $x_k = +1$ gesendet, und umgekehrt. Die Schwelle liegt somit bei $y = 0$ und nur das Vorzeichen des gemessenen Wertes ist entscheidend.

Die empfangenen Werte y_i könnten wie folgt aussehen

-0.8	-1.5	2	0.7
0.7	0.4	-1	2
0.2	0.4	-0.5	

.

Eine harte Entscheidung anhand des Vorzeichens $x'_i = \mathrm{sgn}(y_i)$ und eine Demodulation mit $d'_i = (1 - x'_i)/2$ ergibt

1	1	0	0
0	0	1	0
0	0	1	

.

Drei Paritäten sind verletzt; eine Korrektur der zwei Bitfehler ist nicht möglich.

Log-Likelihood-Algebra

Wie schon im vorangegangenen Abschnitt angedeutet wurde, erfolgt bei der iterativen Decodierung ein Austausch von Information, oder genauer, eine Fortpflanzung von Wahrscheinlichkeiten. Die dabei erforderlichen multiplikativen Verknüpfungen kann man umgehen mit Hilfe der Log-Likelihood-Algebra [Hag96]. In Abbildung 4 ist ein konkreter Signalwert y_k

eingezeichnet, der zum Zeitpunkt k beobachtet wurde. Die Werte l_1 und l_2 an der Ordinate sind die Wahrscheinlichkeiten dieses Signalwertes unter der Bedingung, dass entweder $x=+1$ oder $x=-1$ gesendet wurde.

Interessanter ist jedoch eine Aussage über die Wahrscheinlichkeit des Signals x am Eingang, wenn der Wert y am Ausgang beobachtet wird. Das ist die A-posteriori-Wahrscheinlichkeit $p(x|y)$. Nach dem Theorem von Bayes gilt

$$p(x|y) = \frac{p(y|x)\cdot p(x)}{p(y)} \propto p(y|x)\cdot p(x)\ . \tag{9}$$

Da $p(y)$ lediglich normierenden Charakter hat, kann diese Variable im Weiteren weggelassen werden.

Die Hypothese $x=+1$ ist richtig, wenn gilt $p(x=+1|y) > p(x=-1|y)$ und umgekehrt. Wir können den Test $p(x=+1|y) \lessgtr p(x=-1|y)$ mit Hilfe von Gleichung (9) umschreiben in

$$\begin{aligned} p(x=+1|y) &\lessgtr p(x=-1|y) \\ p(y|x=+1)\cdot p(x=+1) &\lessgtr p(y|x=-1)\cdot p(x=-1) \\ \frac{p(y|x=+1)\cdot p(x=+1)}{p(y|x=-1)\cdot p(x=-1)} &\lessgtr 1\ . \end{aligned}$$

Unter Verwenden des Logarithmus vereinfacht sich der Zusammenhang wie folgt

$$\begin{aligned} \log\left[\frac{p(y|x=+1)\cdot p(x=+1)}{p(y|x=-1)\cdot p(x=-1)}\right] &\lessgtr \log(1) \\ \log\frac{p(y|x=+1)}{p(y|x=-1)} + \log\frac{p(x=+1)}{p(x=-1)} &\lessgtr 0\ . \end{aligned}$$

oder allgemein

$$L(x|y) = L_\mathrm{c}(y) + L(x) \lessgtr 0\ .$$

Die L-Werte werden auch als LLR-Werte (*Log-Likelihood Ratio*) bezeichnet. Der Term $L_\mathrm{c}(y)$ wird durch die Eigenschaften des Kanals bestimmt; der Term $L(x)$ beschreibt die A-priori-Information und ist zu Beginn der Decodierung zunächst gleich Null, unter der Annahme einer Gleichverteilung der Bits $p(x=+1) = p(x=-1)$. Falls $L(x|y)$ größer als Null ist, deutet das auf einen modulierten Wert von $x_i=+1$ und demzufolge auf ein Datenbit $d_i=0$; wenn gilt $L(x|y)<0$, dann wird $d_i=1$ decodiert.

Jeder Decodierer gibt eine so genannte *extrinsische* Information aus, welche sich auf Basis der Paritätsgleichungen ergibt. Wie bereits oben erwähnt, tauschen die Decodierer diese Information aus. Dadurch wird $L(x)$ präzisiert und die Entscheidung für Null oder Eins verbessert.

Der L-Wert $L_\mathrm{c}(y)$ für einen AWGN-Kanal gilt gemäß Abbildung 4

$$\begin{aligned}
L_\mathrm{c}(y) &= \log\frac{p(y|x=+1)}{p(y|x=-1)} = \log\frac{\exp\left[-\frac{1}{2\sigma^2}\cdot(y-1)^2\right]}{\exp\left[-\frac{1}{2\sigma^2}\cdot(y+1)^2\right]} \\
&= \log\left(\exp\left[-\frac{1}{2\sigma^2}\cdot(y-1)^2+\frac{1}{2\sigma^2}\cdot(y+1)^2\right]\right) \\
&= \frac{1}{2\sigma^2}\cdot\left((y+1)^2-(y-1)^2\right) = \frac{1}{2\sigma^2}\cdot 4y \\
L_\mathrm{c}(y) &= \frac{2}{\sigma^2}\cdot y\;. \qquad (10)
\end{aligned}$$

Für andere Kanalmodelle muss diese Berechnung entsprechend angepasst werden.

Wichtig für den Austausch der extrinsischen Information ist zu wissen, wie sich der L-Wert für die Summe zweier oder mehrerer Bits berechnet. Die A-priori-Information

$$L(x) = \log\left(\frac{p(x=+1)}{p(x=-1)}\right) = \log\left(\frac{p(x=+1)}{1-p(x=+1)}\right)$$

lässt sich nach $p(x=+1)$ umstellen

$$\begin{aligned}
\mathrm{e}^{L(x)} &= \frac{p(x=+1)}{1-p(x=+1)} \\
\mathrm{e}^{L(x)} - \mathrm{e}^{L(x)}\cdot p(x=+1) &= p(x=+1) \\
\mathrm{e}^{L(x)} &= p(x=+1)\cdot(1+\mathrm{e}^{L(x)}) \\
p(x=+1) &= \frac{\mathrm{e}^{L(x)}}{1+\mathrm{e}^{L(x)}}
\end{aligned}$$

Analog gilt

$$p(x=-1) = 1-p(x=+1) = 1-\frac{\mathrm{e}^{L(x)}}{1+\mathrm{e}^{L(x)}} = \frac{1}{1+\mathrm{e}^{L(x)}}$$

Außerdem ist die Wahrscheinlichkeit, dass der BPSK-Wert aus der XOR-Summe zweier Bits gleich Eins ist, abhängig davon, mit welcher Wahrscheinlichkeit beide Bits denselben Wert haben ($0\oplus 0 = 1\oplus 1 = 0$).

$$p(x_1\oplus x_2=+1) = p(x_1=+1)\cdot p(x_2=+1) + p(x_1=-1)\cdot p(x_2=-1) \qquad (11)$$

Für die Summe zweier Bits, unter Voraussetzung ihrer statistischen Unabhängigkeit, gilt

nun

$$
\begin{aligned}
L(x_1 \oplus x_2) &= \log \frac{p(x_1=+1)\cdot p(x_2=+1) + p(x_1=-1)\cdot p(x_2=-1)}{1-[p(x_1=+1)\cdot p(x_2=+1) + p(x_1=-1)\cdot p(x_2=-1)]} \\
&= \log \left(\frac{\frac{\mathrm{e}^{L(x_1)}}{1+\mathrm{e}^{L(x_1)}} \cdot \frac{\mathrm{e}^{L(x_2)}}{1+\mathrm{e}^{L(x_2)}} + \frac{1}{1+\mathrm{e}^{L(x_1)}} \cdot \frac{1}{1+\mathrm{e}^{L(x_2)}}}{1-\left[\frac{\mathrm{e}^{L(x_1)}}{1+\mathrm{e}^{L(x_1)}} \cdot \frac{\mathrm{e}^{L(x_2)}}{1+\mathrm{e}^{L(x_2)}} + \frac{1}{1+\mathrm{e}^{L(x_1)}} \cdot \frac{1}{1+\mathrm{e}^{L(x_2)}}\right]} \right) \\
&= \log \left(\frac{\frac{\mathrm{e}^{L(x_1)}\cdot \mathrm{e}^{L(x_2)}+1}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}}{1-\left[\frac{\mathrm{e}^{L(x_1)}\cdot \mathrm{e}^{L(x_2)}+1}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}\right]} \right) \\
&= \log \left(\frac{\frac{\mathrm{e}^{L(x_1)}\cdot \mathrm{e}^{L(x_2)}+1}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}}{\frac{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]} - \left[\frac{\mathrm{e}^{L(x_1)}\cdot \mathrm{e}^{L(x_2)}+1}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}\right]} \right) \\
&= \log \left(\frac{\frac{\mathrm{e}^{L(x_1)}\cdot \mathrm{e}^{L(x_2)}+1}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}}{\left[\frac{\mathrm{e}^{L(x_1)}+\mathrm{e}^{L(x_2)}}{[1+\mathrm{e}^{L(x_1)}]\cdot[1+\mathrm{e}^{L(x_2)}]}\right]} \right) \\
&= \log \left(\frac{\mathrm{e}^{L(x_1)}\cdot \mathrm{e}^{L(x_2)}+1}{\mathrm{e}^{L(x_1)}+\mathrm{e}^{L(x_2)}} \right) \qquad (12)
\end{aligned}
$$

In [Hag96] wurde folgende Näherung angegeben

$$L(x_1 \oplus x_2) \approx \operatorname{sgn}[L(x_1)] \cdot \operatorname{sgn}[L(x_2)] \cdot \min\{|L(x_1)|, |L(x_1)|\} \ .$$

Von Bedeutung sind demnach nur der betragsmäßig kleinste L-Wert und das Produkt der Vorzeichen. Das lässt sich verallgemeinern zu

$$L\left(\sum_i \oplus\, x_i\right) \approx \left\{\prod_i \operatorname{sgn}[L(x_i)]\right\} \cdot \min_i \{|L(x_i)|\} \ . \qquad (13)$$

Damit hat man nicht nur einen L-Wert für ein einzelnes Bit, sondern auch für die additive Verknüpfung von Bits in Modulo-2-Arithmetik. Dieser Zusammenhang wird bei der iterativen Decodierung zur Auswertung der Paritätsgleichungen und dem Erzeugen der extrinsischen Information benötigt.

Decodierungsbeispiel

Kehren wir zu dem obigen Beispiel mit den empfangenen Werten y_i

-0.8	-1.5	2	0.7
0.7	0.4	-1	2
0.2	0.4	-0.5	

zurück. Zur Vereinfachung können wir $\sigma^2 = 2$ annehmen und die Zahlenwerte sind nach (10) gleichzeitig die L-Werte des Kanals $L_\mathrm{c}(y)$. Die Information des Kanals lautet also $L_\mathrm{c}(y) = (-0.8\ -1.5\ 2\ 0.7\ 0.4\ -1\ 0.7\ 2\ 0.2\ 0.4\ -0.5)$. In die Struktur der Kontrollmatrix $\mathbf{H}$ aus Gleichung (8) tragen wir diese L-Werte dort ein, wo das entsprechende Bit in eine Paritätsgleichung einbezogen ist. Die Positionen der Null-Bits in der Matrix bleiben leer zur besseren Übersicht. Die Werte sind zunächst unverändert, weil die extrinsische Information $L_{e_1}(x)$ und $L_{e_2}(x)$ der beiden Decodierer noch nicht bekannt ist.

$L_\mathrm{c}(y)$	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
$L_\mathrm{c}(y) + L_{e_2}(x)$	-0.8	-1.5	2.0				0.7				
				0.7	$\underline{0.4}$	-1.0		2.0			
$L_\mathrm{c}(y) + L_{e_1}(x)$	-0.8			0.7					$\underline{0.2}$		
		-1.5			$\underline{0.4}$					0.4	
			2.0			-1.0					-0.5

(14)

Die unterstrichenen L-Werte führen zu Bitfehlern bei einer harten Entscheidung. Die Anzahl der negativen Werte muss in jeder Zeile gerade sein, damit die Paritätsgleichungen erfüllt sind.

Nun werden die extrinsischen Informationen beider Decoder unter Zuhilfenahme der Beziehung aus Gleichung (13) bestimmt. Die Frage lautet: Wie groß ist der L-Wert für ein Bit, wenn man die Information aller anderen Bits auswertet, die über eine Paritätsgleichung mit diesem Bit verknüpft sind?

Nehmen wir zum Beispiel das Bit d_1. Es ist über die Gleichung $d_1 + d_2 + d_3 + p_1 = 0$ mit drei anderen Bits verbunden (Decodierer I). Kennt man die L-Werte von d_2, d_3 und p_1, kann man auf den L-Wert von d_1 schließen. Das Produkt der Vorzeichen dieser drei Werte ist gleich minus Eins und der kleinste Betragswert ist gleich 0.7. Die extrinsische Information aus Decodierer I für das Bit d_1 beträgt also -0.7. Genauso wird für alle anderen Bits verfahren und es ergibt sich folgende extrinsische Information

1. $L_{e_1}(x)$	-0.7	-0.7	0.7				0.8				
				-0.4	-0.7	0.4		-0.4			
$L_{e_2}(x)$	0.2			-0.2					-0.7		
		0.4			-0.4					-0.4	
			0.5			-0.5					-1.0

. (15)

Die extrinsische Information wird nun zwischen den Decodierern ausgetauscht und man erhält

$L_\mathrm{c}(y)$	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
$L_\mathrm{c}(y) + L_{e_2}(x)$	-0.6	-1.1	2.5				0.7				
				0.5	$\underline{0.0}$	-1.5		2.0			
$L_\mathrm{c}(y) + L_{e_1}(x)$	-1.5			0.3					$\underline{0.2}$		
		-2.2			-0.3					0.4	
			2.7			-0.6					-0.5
$L_\mathrm{c} + L_{e_1} + L_{e_2}$	-1.3	-1.8	3.2	0.1	-0.7	-1.1	1.5	1.6	-0.5	$\underline{0.0}$	-1.5

. (16)

Da in den letzten fünf Spalten der Matrix nur jeweils ein Eintrag pro Spalte steht, berechnet zwar jeder Decodierer für seine Bits eine extrinsische Information, es besteht aber

keine Möglichkeit zum Austausch. Das Gesamtergebnis der Decodierung nach dem ersten Durchlauf ergibt sich aus der Summe der L-Werte aus den drei Informationsquellen (Kanal, Decodierer I und II) zu

$$L(\hat{y}) = L_{\mathrm{c}}(y) + L_{e_1}(x) + L_{e_2}(x) \; . \tag{17}$$

Das Vorzeichen von $L(\hat{y})$ liefert den gesendeten Signalwert x, der Betrag von $L(\hat{y})$ gibt Auskunft über die Zuverlässigkeit des decodierten Bits. Es ist zu erkennen, dass durch das Hinzunehmen der extrinsischen Information die beiden ursprünglich fehlerhaften Werte für d_5 und p_3 korrigiert wurden. Dafür ermöglicht der L-Wert für p_4 keine Entscheidung mehr. Ein weiterer Durchlauf ist erforderlich. Die extrinsiche Information wird nun auf Basis der Matrix (16) bestimmt

2.	$L_{e_1}(x)$	−0.7	−0.6	0.6				0.6					
					0.0	−0.5	0.0		0.0				
	$L_{e_2}(x)$	0.2			−0.2					−0.3			
			−0.3			−0.4					0.3		
				0.5			−0.5						−0.6

. (18)

Der Input für die nächste Iteration und das Gesamtergebnis lauten

$L_{\mathrm{c}}(y)$	−0.8	−1.5	2.0	0.7	$\underline{0.4}$	−1.0	0.7	2.0	$\underline{0.2}$	0.4	−0.5
$L_{\mathrm{c}}(y) + L_{e_2}(x)$	−0.6	−1.8	2.5				0.7				
				0.5	$\underline{0.0}$	−1.5		2.0			
$L_{\mathrm{c}}(y) + L_{e_1}(x)$	−1.5			0.7					$\underline{0.2}$		
		−2.1			−0.1					0.4	
			2.6			−1.0					−0.5
$L_{\mathrm{c}} + L_{e_1} + L_{e_2}$	−1.3	−2.4	3.1	0.5	−0.5	−1.5	1.3	2.0	−0.1	0.7	−1.1

. (19)

Damit stimmen alle Vorzeichen, d.h. alle Paritätsgleichungen, die Bits werden ohne Fehler decodiert und die Decodierung wird beendet. Ein weiterer Durchlauf würde nur die Beträge von $L(\hat{y})$ leicht verändern, das Vorzeichen und damit das Decodierungsergebnis jedoch nicht mehr.

Serielle Decodierung mit Decodierungsbeispiel

Das Typische bei der parallelen Verarbeitung ist, dass alle Decoder gleichzeitig arbeiten und anschließend die extrinsische Information austauschen. Bei einer seriellen Verarbeitung arbeitet immer nur ein Decoder. Wenn er fertig ist, wird sein Output als extrinsische Information für den anderen[1] Decoder verwendet. Dieser hat dadurch gleich eine verbesserte Ausgangslage und produziert exaktere Ergebnisse. Sein Output wird dann wiederum vom ersten Decoder mit einbezogen. Die Decoder arbeiten also abwechselnd, was im Allgemeinen zu einer schnelleren Konvergenz führt.

[1] oder den nächsten bei mehr als zwei Decodern

Ausgehend von den empfangenen Signalwerten aus obigem Beispiel

$L_\mathrm{c}(y)$	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
$L_\mathrm{c}(y)+L_{e_2}(x)$	-0.8	-1.5	2.0				0.7				
				0.7	$\underline{0.4}$	-1.0		2.0			
$L_\mathrm{c}(y)+L_{e_1}(x)$	-0.8			0.7					$\underline{0.2}$		
		-1.5			$\underline{0.4}$					0.4	
			2.0			-1.0					-0.5

(20)

wird die serielle Decodierung erläutert. Die extrinsische Information von Decoder I ist zunächst

$1.I$ $L_{e_1}(x)$	-0.7	-0.7	0.7				0.8			
				-0.4	-0.7	0.4		-0.4		

. (21)

Diese Information wird nun von Decoder II verarbeitet

$L_\mathrm{c}(y)$	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
$L_\mathrm{c}(y)+L_{e_1}(x)$	-1.5			0.3					$\underline{0.2}$		
		-2.2			-0.3					0.4	
			2.7			-0.6					-0.5

(22)

und seine extrinsische Information $L_{e_2}(x)$ wird berechnet

$1.II$ $L_{e_2}(x)$	0.2			-0.2					-0.3		
		-0.3			-0.4					0.3	
			0.5			-0.5					-0.6

. (23)

Das Verknüpfen der empfangenen Signalwerte $L_\mathrm{c}(y)$ mit den beiden extrinsischen Informationen $L_{e_1}(x)$ und $L_{e_2}(x)$ ergibt

$L_\mathrm{c}+L_{e_1}+L_{e_2}$	-1.3	-2.5	3.2	0.1	-0.7	-1.1	1.5	1.6	-0.1	0.7	-1.1

. (24)

Alle Paritätsgleichungen sind nach dieser ersten Iteration bereits erfüllt und das Decodieren ist beendet. Anderenfalls müsste die Information $L_{e_2}(x)$ wieder in den Decoder I gespeist werden.

Anmerkung:
Bei LDPC-Codes kann jede Zeile der Kontrollmatrix $\mathbf{H}$ als ein Encoder interpretiert werden. Für jede Zeile wird die extrinsische Information erzeugt und jeweils für alle anderen Zeilen zur verfügung gestellt.

Parallele Decodierung mit modifizierter Kontrollmatrix

Im Gegensatz zu den voran gegangenen Beispielen wird nun mit der Kontrollmatrix aus Gleichung (4) geprüft. Die Anzahl der an den Paritätsgleichungen beteiligten Bits ist dadurch größer. Außerdem sind die Bits auch an mehr Gleichungen beteiligt.

Ausgehend von den empfangenen Signalwerten aus obigem Beispiel wird für jede Zeile die extrinsische Information bestimmt:

L_c	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
L_e		0.20	-0.20	-0.20			-0.20		-0.70		
L_e		-0.40		0.40		-0.40		0.40		0.70	
L_e	-0.20	-0.20		0.20	0.20				0.40	0.20	
L_e		0.40	-0.40		-0.40	0.40				-0.40	0.40
L_e	0.40	0.40		-0.40	-0.50		-0.40	-0.40			0.40
$\sum$	-0.60	-1.10	1.40	0.70	-0.30	-1.00	0.10	2.00	-0.10	0.90	0.30

(25)

Alle Positionen, die nicht an einer Paritätsgleichung beteiligt sind, sind leer dargestellt. Für die Summenzeile

$$\sum = L_c + \sum_i L_e(i) \ .$$

werden alle Paritätsgleichungen geprüft. Da nicht alle erfüllt sind (die beiden letzten Zeilen von $H_\mathrm{mod} \Rightarrow$ Wert in letzter Spalte von $\sum$ sollte negativ sein), werden nun die L_e-Werte ausgetauscht. Für Zeile r gilt

$$L_u(r) = L_c + \sum_{i \neq r} L_e(i)$$

oder unter Verwenden der bereits berechneten Summe

$$L_u(r) = \sum -L_e(r) \ . \tag{26}$$

$L_\mathrm{c}(y)$	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
$L_u(1)$		-1.30	1.60	0.90			0.30		0.60		
$L_u(2)$		-0.70		0.30	$-$	-0.60		1.60		0.20	
$L_u(3)$	-0.40	-0.90		0.50	-0.50				-0.50	0.70	
$L_u(4)$		-1.50	1.80		0.10	-1.40				1.30	-0.10
$L_u(5)$	-1.00	-1.50		1.10	0.20		0.50	2.40			-0.10

(27)

Diese Summenwerte werden als Basis verwendet für die nächste Iteration, d.h. für die nächste Berechnung der extrinsischen Information.

L_c	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
L_e		0.30	-0.30	-0.30			-0.60		-0.30		
L_e		-0.20		0.20		-0.20		0.20		0.30	
L_e	-0.50	-0.40		0.40	-0.40				-0.40	0.40	
L_e		0.10	-0.10		-0.10	0.10				-0.10	0.10
L_e	0.10	0.10		-0.10	-0.10		-0.10	-0.10			0.20
$\sum$	-1.20	-1.60	1.60	0.90	-0.20	-1.10	0.00	2.10	-0.50	1.00	-0.20

(28)

Auch nach dieser Berechnung ist eine Position (0.00) nicht korrekt. Die Summen gemäß

Gleichung (26) ergeben

$L_c(y)$	-0.8	-1.5	2.0	0.7	$\underline{0.4}$	-1.0	0.7	2.0	$\underline{0.2}$	0.4	-0.5
$L_u(1)$		-1.90	1.90	1.20			0.60		-0.20		
$L_u(2)$		-1.40		0.70		-0.90		1.90		0.70	
$L_u(3)$	-0.70	-1.20		0.50	0.20				-0.10	0.60	
$L_u(4)$		-1.70	1.70		-0.10	-1.20				1.10	-0.30
$L_u(5)$	-1.30	-1.70		1.00	-0.10		0.10	2.20			-0.40

(29)

L_e		-0.20	0.20	0.20			0.20		-0.60		
L_e		-0.70		0.70		-0.70		0.70		0.70	
L_e	0.10	0.10		-0.10	-0.10				0.20	-0.10	
L_e		-0.10	0.10		-0.30	-0.10				0.10	-0.10
L_e	-0.10	-0.10		0.10	-0.10		0.10	0.10			-0.10
$\sum$	-0.80	-2.50	2.30	1.60	-0.10	-1.80	1.00	2.80	-0.20	1.10	-0.70

(30)

Jetzt stimmen alle Paritäten für $\sum$ und das Decodieren kann für dieses Codewort beendet werden.

4 Charakteristika von LDPC-Codes

4.1 Parameter

Um ihre Leistungsfähigkeit zu entfalten, haben LDPC-Codes typischer Weise eine Blocklänge n von mehreren hundert oder sogar mehreren tausend Bits. Die Kontrollmatrix $\mathbf{H}$ der Größe $k \times n$ ist dementsprechend riesig. Das obige Beispiel ist im strengen Sinne also kein LDPC-Code. Die Blocklänge n ergibt sich aus der Summe von l Datenbits und k Paritätsbits, d.h. es gilt $n = l + k$.

Wichtige Kennzeichen wind die Anzahl w_r der 1-Bits pro Zeile und die Anzahl w_c der 1-Bits pro Spalte in $\mathbf{H}$, für die gelten sollte

$$w_r \ll n \qquad \text{und} \qquad w_c \ll k \; . \tag{31}$$

Die Anzahl der Einsen in $\mathbf{H}$ ist also insgesamt sehr niedrig, die Matrix ist dünn besetzt. Die Forderung nach einer dünn besetzten Matrix ist aber nicht nur der Leistungsfähigkeit der iterativen Decodierung geschuldet, sondern soll deren Rechenaufwand begrenzen. Ist w_r konstant für alle Zeilen und ist w_c konstant für alle Spalten von $\mathbf{H}$, dann nennt man den Code *regulär*. Sei l die Anzahl der Datenbits und k die Anzahl der Paritätsbits in einem Codewort, dann gilt $w_\mathrm{r} = w_\mathrm{c} \cdot (l + k)/k$ für reguläre Codes.

Variiert die Anzahl der Einsen, handelt es sich um einen *irregulären* Code. Die ursprünglich von Gallager vorgeschlagenen LDPC-Codes waren regulär [Gal63].

Bezogen auf die Visualisierung mit einem zweiteiligen Graphen entspricht w_c der Anzahl der Kanten, welche einen Knoten der variablen Bits verlassen, und w_r ist die Anzahl der Kanten, die einen Kontrollknoten verlassen. Für den Blockcode in Abbildung 2 gelten demnach $w_\mathrm{r} \in \{3, 4\}$ und $w_\mathrm{c} \in \{1, 2\}$. Es handelt sich um einen irregulären Code. Typischer Weise sind irreguläre Codes leistungsfähiger als reguläre.

Weiterhin werden LDPC-Codes durch einen Umfang (*girth*) charakterisiert. Dieser gibt an, wie viele Kanten mindestens erforderlich sind, um einen beliebigen Knoten im zweiteiligen Graphen zu verlassen und wieder zu ihm zurück zu kehren. Aufgrund der Struktur des Graphen ist eine solche Schleifenlänge immer gerade und die kleinstmögliche Schleifenlänge ist gleich vier. In Abbildung 2 hat eine Schleife von und nach Knoten d_1 eine Länge von 8 ($d_1 \rightarrow c_1 \rightarrow d_2 \rightarrow c_4 \rightarrow d_5 \rightarrow c_2 \rightarrow d_4 \rightarrow c_3 \rightarrow d_1$). Das Umwandeln in einen nichtsystematischen Kanalcode (Abbildung 3) reduziert den Umfang deutlich auf eine Schleifenlänge von 4 ($d_1 \rightarrow c_3 \rightarrow d_4 \rightarrow c_5 \rightarrow d_1$).

Wie für andere Blockcodes auch ist der minimale Hamming-Abstand $d_{\min}$, der durch den paarweisen Vergleich von zwei Kanalcodewörtern ermittelt wird, von Bedeutung. Sei

$$d_{\mathrm{H}}(\mathbf{b}_i, \mathbf{b}_j) = \sum_{g=1}^{n} b_i[g] \oplus b_j[g] \qquad \forall\, \mathbf{b}_i, \mathbf{b}_j \in \text{Kanalcode} \tag{32}$$

der Hamming-Abstand zwischen zwei Kanalcodewörten $\mathbf{b}_i$ und $\mathbf{b}_j$ mit den Elementen $b_i[g]$ bzw. $b_j[g]$. Dann ist der minimale Hamming-Abstand dieses Kanalcodes gleich

$$d_{\min} = \min[d_{\mathrm{H}}(\mathbf{b}_i, \mathbf{b}_j)] \qquad \forall (i,j); i \neq j\ . \tag{33}$$

Der minimale Hamming-Abstand bestimmt, wie viele Symbolfehler innerhalb des Codewortes mit Sicherheit erkannt ($d_{\min} - 1$) oder bei harter Entscheidung korrigiert ($\lfloor (d_{\min} - 1)/2 \rfloor$) werden können.

4.2 Konstruktion

Shannon's Theorie besagt, dass lange und zufällige Codes eine Übertragung an den Grenzen Kanalkapazität ermöglichen. LDPC-Codes liefern hierfür die Lösung. Von Richardson und Urbanke wurde ein LDPC-Code mit einer Coderate von $R = l/n = 1/2$ vorgestellt, der eine Bitfehlerrate (BER) kleiner 10^6 bei einem Abstand von nur 0.13 dB zur theoretischen, von Shannon vorausgesagten Grenze hat [Ric01]. Die Blocklänge n beträgt dafür allerdings 10^6 Bits.

Gute Ergebnisse erreicht man im Allgemeinen, wenn man die Einsen zufällig in die Kontrollmatrix einträgt. Dabei sollten allerdings in jeder Zeile und jeder Spalte mehrere Einsen stehen, damit die oben beschrieben extrinsische Information ausgetauscht werden kann. Diese Eigenschaft ist bei systematischen Codes, wie in Gleichung (2) leider nicht gegeben. Außerdem sollten alle sich daraus ergebenden Paritätsgleichungen linear unabhängig voneinander sein.

Ursprünglich schien es, dass Codes mit einem Umfang (*girth*) von nur vier generell schlecht sind. Solche kurzen Schleifen wurden deshalb aus einem (zufällig) konstruierten Graph wieder entfernt. In [Che04] wurde jedoch gezeigt, dass es auch leistungsstarke Codes mit einem Umfang von nur vier gibt.

Der minimale Hamming-Abstand $d_{\min}$ eines Kanalcodes wird maximiert, wenn man die Ähnlichkeit von Spalten in $\mathbf{H}$ minimiert.

Die folgende Kontrollmatrix mit $\overline{w}_\mathrm{r} = 6.875$ und $\overline{w}_\mathrm{c} = 3.4375$ wurde zufällig generiert

$$\mathbf{H} = \left(\begin{array}{cccccccccccccccc}
|\overline{1} & \overline{0} & \overline{1}| & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
|\underline{1} & \underline{0} & \underline{1}| & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0
\end{array}\right) . \tag{34}$$

Wenn zwei Spalten zwei Einsen an der gleichen Position haben (siehe Markierung in der Matrix), dann wird dadurch eine Schleife der Länge vier im zweiteiligen Graphen generiert.

Das große Problem mit zufälligen Matrizen ist jedoch, dass sowohl die Encodierung als auch die Decodierung rechentechnisch relativ aufwändig sind. Einfache, auf Schieberegistern basierende Schaltungen, wie sie für viele andere Kanalcodierungsverfahren verwendet werden, sind dafür nicht geeignet. Deshalb gibt es ein großes Interesse Kontrollmatrizen zu finden, die zwar einen zufälligen Charakter haben, aber trotzdem eine gewisse innere Struktur aufweisen, welche eine Reduktion des Rechenaufwandes ermöglicht. Dazu gehören quasi-zyklische Codes [Fos04] oder so genannte Protograph-Konstruktionen [Tho04], bei welcher der zweiteilige Graph aus Prototypen kleinerer Graphen zusammengesetzt wird.

Eine ausführlichere Beschreibung findet man auch in [Joh00] oder [Joh02].

4.3 Decodieren von LDPC-Codes

Das Decodieren von LDPC-Codes erfolgt prinzipiell nach dem oben beschriebenen iterativen Verfahren. Es wird auch als „Turbo-Prinzip“ bezeichnet, in Anlehnung an die Funktionsweise eines Turbo-Verbrennungsmotors, bei dem die Abgase wieder in den Kreislauf des Motors eingespeist werden.

Da Wissenschaftler mit unterschiedlichstem fachlichen Hintergrund an der Entwicklung dieser Kanalcodierungstechnik beteiligt waren und sind, gibt es Variationen des iterativen Prinzips, die mit anderen Name versehen wurden, wie zum Beispiel *sum-product algorithm*, *message-passing algorithm* oder *belief propagation*. Letzter Algorithmus hat seinen Ursprung in der Graphentheorie und stützt sich auf auf die Beschreibung von LDPC-Codes mittels zweiteiliger Graphen (siehe Abschnitt 1).

Im Beispiel der Kontrollmatrix $\mathbf{H}$ in Gleichung (2) gibt es für jedes Datenbit genau zwei Einsen pro Spalte und für jedes Paritätsbits eine Eins. Typische LDPC-Matrizen unterscheiden nicht zwischen Daten- und Paritätsbits (zusammen: variable Bits) und haben im Durchschnitt eine größere Anzahl von Einsen pro Spalte (siehe Gleichung 34). Sie sollte mindestens gleich drei sein ($\overline{w}_\mathrm{c} \geq 3$). Jede Paritätsgleichung (jede Zeile in $\mathbf{H}$) kann als ein separater Decoder interpretiert werden, der dann aber nur eine Untermenge der variablen Bits verarbeitet. Das variable Bit in Spalte j mit dem Gewicht $w_{\mathrm{c},j}$ ist also an $w_{\mathrm{c},j}$ Paritätsgleichungen beteiligt und kann mit $w_{\mathrm{c},j} - 1$ extrinsischen Informationen kombiniert werden.

Die Iteration der Decodierung wird beendet, wenn alle Paritätsgleichungen erfüllt sind, d.h., ein gültiges Kanalcodewort gefunden wurde. Weitere Durchläufe verändern das Resultat

im Allgemeinen nicht mehr. Die Anzahl der erforderlichen Durchläufe sinkt mit steigendem Signal-Rausch-Verhältnis (SRV) der Übertragung. Bei gleichem SRV benötigen Codes mit großen Blocklängen weniger Durchläufe als kurze LDPC-Codes.

Da bei zu stark gestörtem Signal die Nachricht eventuell nicht rekonstruiert werden kann, muss die maximale Anzahl der Iterationen begrenzt werden. Das schränkt außerdem den Rechenaufwand und die Zeitverzögerung ein. Diese Arbeitsweise führt allerdings zu einer variablen Decodierungszeit. Die durchschnittliche Arbeitszeit ist aber trotzdem geringer, als wenn die Anzahl von Iterationen konstant und ausreichend hoch sein würde. Ist diese maximale Anzahl an Durchläufen erreicht, signalisiert der Decoder einen nicht korrigierbaren Übertragungsfehler an die weitere Verarbeitungskette.

Die Rate der nicht erkannten Fehler ist sehr gering, ähnlich wie bei Reed-Solomon-Codes. Lediglich, wenn der minimale Hamming-Abstand des Kanalcodes zu klein im Vergleich zu den aufgetretenen Störungen ist, kann es passieren, dass der Decoder ein falsches gültiges Kanalcodeword generiert und der Übertragungsfehler somit unerkannt bleibt. Turbo-Codes hingegen, welche die iterative Decodierung in Verbindung mit rekursiven, systematischen Faltungscodierern ausnutzen, können keine Information über nicht korrigierbaren Übertragungsfehler erzeugen; Decodierfehler bleiben unerkannt, falls nicht zusätzliche Maßnahmen getroffen werden, wie z.B. ein Cyclic-Redundancy-Check-Code (CRC) als äußerer Code.

4.4 Leistungsfähigkeit

Untersuchungen

Im Internet stehen verschieden Programme zur Verfügung, mit denen man eine LDPC-Encodierung und Decodierung durchführen kann. Die hier präsentierten Ergebnisse basieren auf Simulationen mit dem Softwarepaket von Radford Neal (Koautor von [Mac96]). Der Quellcode (ANSI-C, [wwwSC]) kompiliert unter Unix/Linux ohne Probleme. Unter Windows sind leichte Anpassungen erforderlich. Die Software kann beliebige Kontrollmatrizen einlesen und verwenden, sodass Untersuchungen mit selbst konstruierten Matrizen möglich sind.

Das Softwarepaket enthält unter anderem folgende Komponenten

- Erzeugen von Kontroll- und Generatormatrizen,
- Generieren eines zufälligen Bitstroms,
- Encodieren des Bitstroms,
- Simulation eines Kanals,
- Decodieren und
- Überprüfen der decodierten Nachricht.

Die beispielhaften Untersuchungen wurden ausschließlich mit einer Coderate von $R = 1/2$ und einem AWGN-Kanal durchgeführt. Die Bitfehlerrate (BER), aufgetragen über dem Signal-Rausch-Verhältnis E_b/N_0, ist das Gütekriterium eines Kanalcodes. Die Bitfehlerrate ist die Wahrscheinlichkeit, dass ein Bits nicht korrigiert werden kann. E_b ist die Energie pro Informationsbit und N_0 die einseitige Rauschleistungsdichte. Das Verhältnis berechnet

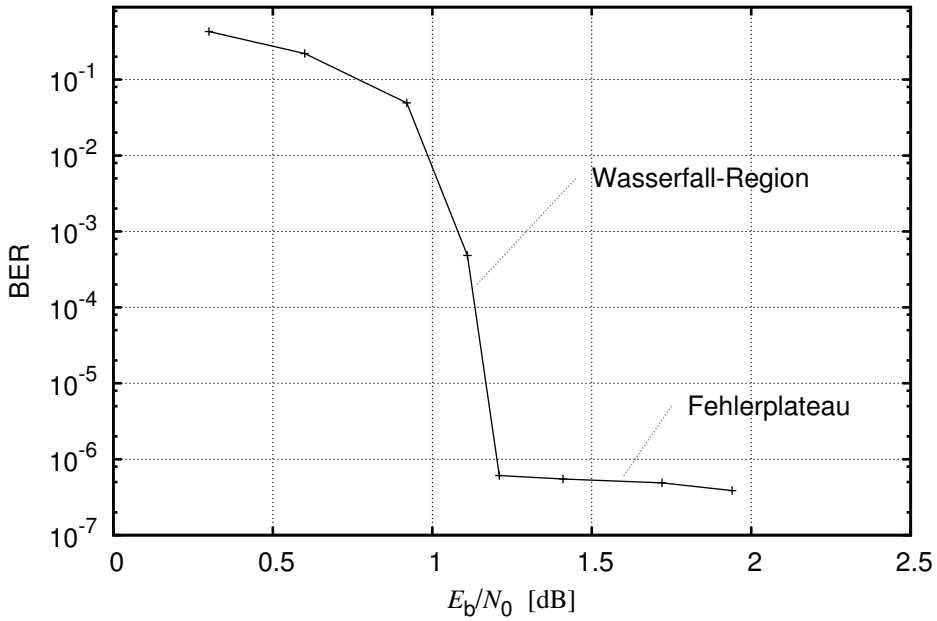

Abbildung 5: Charakteristische Kurve für Kanalcodierungsverfahren mit iterativer Soft-Input/Soft-Output-Decodierung

sich in Abhängigkeit von der Varianz des Rauschens σ_z^2 und der Coderate R nach folgender Gleichung [Mac96]

$$\frac{E_\mathrm{b}}{N_0}\ [\mathrm{dB}] = 10 \cdot \log_{10}\left(\frac{a^2}{2 \cdot R \cdot \sigma_\mathrm{z}^2}\right)\ , \tag{35}$$

wenn die gesendeten Signalwerte $\pm a$ betragen.

Die Leistungsfähigkeit eines Kanalcodes zeigt sich darin, wie stark er die Bitfehlerwahrscheinlichkeit bei einem bestimmten Signal-Rausch-Verhältnis senkt. **Abbildung 5** zeigt eine typische Kurve. Ab einem bestimmten Signal-Rausch-Verhältnis sinkt die Bitfehlerrate deutlich. Diese Region wird als *Wasserfall-Region* bezeichnet. Es kann jedoch passieren, dass der drastische Abfall der Kurve sich wieder stark verringert und ein deutlich höheres Signal-Rausch-Verhältnis erforderlich wird, um die Bitfehlerrate weiter zu senken. Diese Region nennt man Fehlerplateau (*error floor*). Die Ursache liegt meist bei Bits, die nur in eine Paritätsgleichung eingebunden sind, weil diese dann eine erhöhte Wahrscheinlichkeit haben, nicht korrigiert zu werden.

Abbildung 6 vergleicht reguläre LDPC-Codes mit unterschiedlichen Blocklängen. Alle wurden mit der Software von Neal konstruiert. Die Einsen sind zufällig in der Kontrollmatrix $\mathbf{H}$ verteilt und es gelten $w_\mathrm{r} = 6$ und $w_\mathrm{c} = 3$. Deutlich zu erkennen ist, dass die erforderliche Energie pro Bit sinkt, je länger die verwendeten Codes sind.

Wenn die Anzahl der Einsen pro Spalte oder Zeile nicht konstant ist, nennt man, wie bereits oben erwähnt, den LDPC-Code irregulär. Interessanter Weise ist die Leistungsfähigkeit der irregulären Codes stark von der Verteilung der Einsen abhängig.

Abbildung 7 vergleicht irreguläre LDPC-Codes mit unterschiedlichen Verteilungen für

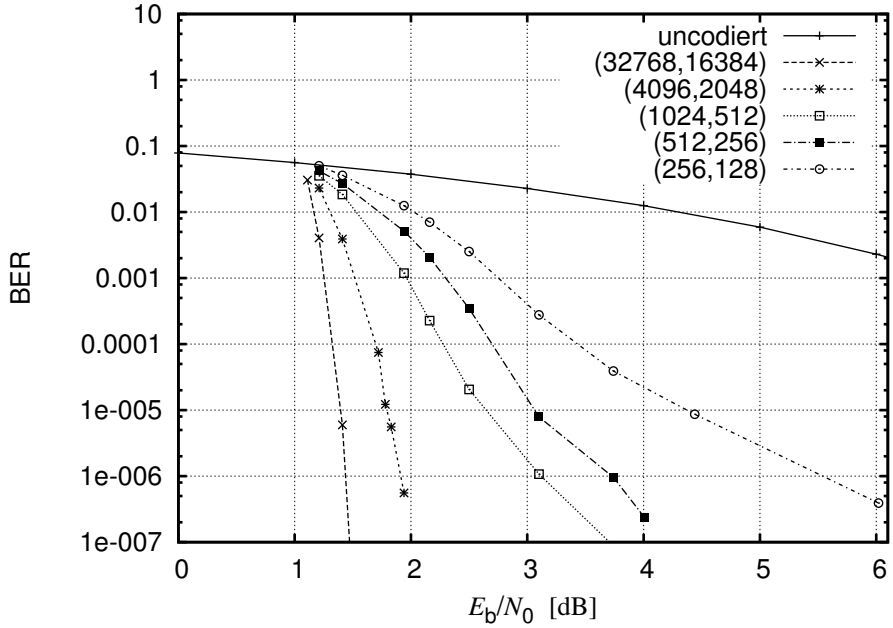

Abbildung 6: Bitfehlerraten (BER) für reguläre LDPC-Codes mit $w_\mathrm{c} = 3$, $w_\mathrm{r} = 6$ im Vergleich

w_c. Zum Vergleich wurde die Kurve des regulären Codes gleicher Länge mit eingezeichnet. Die Verteilungen wurden willkürlich wie folgt gewählt:

	w_c						
Code	2	3	4	5	6	7	$\overline{w}_\mathrm{c}$
regulär	0	1	0	0	0	0	3.00
1	1/8	5/8	1/8	0	0	1/8	3.11
2	2/9	5/9	0	0	0	2/9	3.67
3	3/13	7/13	2/13	1/13	0	0	3.07
4	3/11	7/11	0	0	1/11	0	3.00

.

Offensichtlich gibt es einen Kompromiss zwischen Position der Wasserfall-Region und der Höhe des Fehlerplateaus.

Shannon-Grenze

Wo liegt nun die so genannte Shannon-Grenze? Grundsätzlich hängt diese Grenze von der Art der Modulation und der Coderate ab. Für einen AWGN-Kanals mit binärem Input und wertkontinuierlichem Output ist diese Grenze in **Abbildung 8** als Funktion der Coderate R dargestellt [Lin04]. Dabei ist eine BPSK-Modulation[2] angenommen. Die Kurve zeigt an, welches Signal-Rausch-Verhältnis theoretisch mindestens nötig ist, um bei einer bestimmten Coderate eine fehlerfreie Übertragung zu ermöglichen. Strebt die Coderate gegen Null, d.h.,

[2]Binary Phase Shift Keying

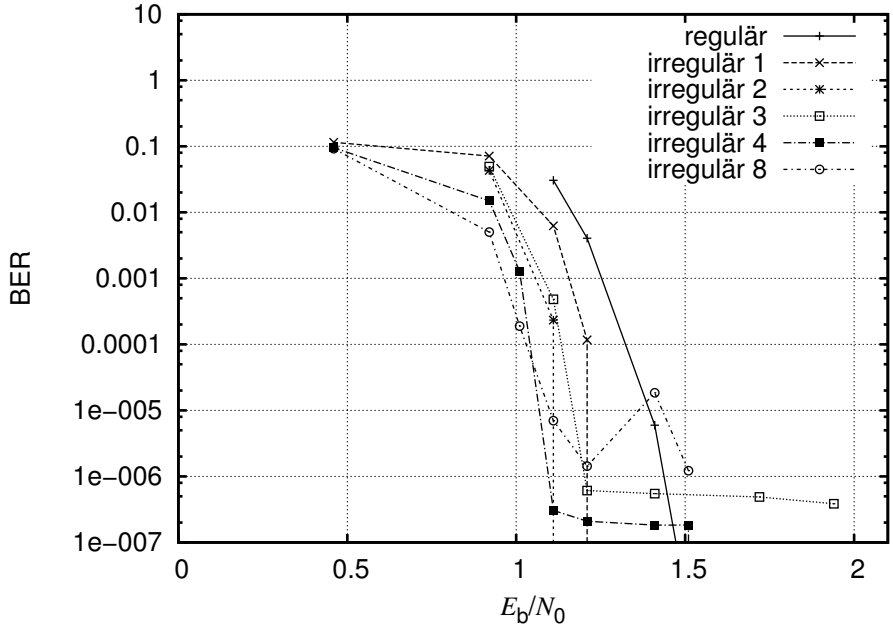

Abbildung 7: Bitfehlerraten (BER) für irreguläre LDPC-Codes mit $l = 16384$ und $R = 1/2$, siehe Text für Details

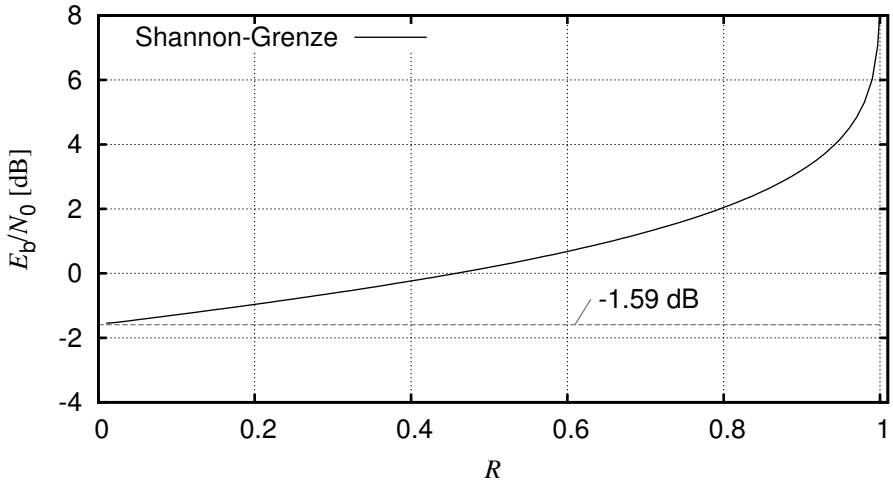

Abbildung 8: Minimales Signal-Rausch-Verhältnis, das bei einer gegebenen Coderate R theoretisch eine fehlerfreie Übertragung ermöglicht (AWGN-Kanal mit BPSK-Signalisierung und kontinuierlichem Output)

der Aufwand zur Kanalcodierung wird unendlich groß, dann sind mindestens -1.59 dB erforderlich. Für eine Coderate von $R = 1/2$, wie sie bei den obigen Untersuchungen verwendet wurde, gilt $E_\mathrm{b}/N_0 \geq 0.188$ dB.

5 Anwendungen und Ausblick

Low-Density-Parity-Check-Codes haben inzwischen in vielen Anwendungen ihren Platz gefunden. Insbesondere in der Raumfahrt spielt die Energieeffizienz eine außerordentlich große Rolle. Es wird solchen Kanalcodes der Vorrang gegeben, die dicht an der Grenze der Kanalkapazität arbeiten können [And07]. Turbo-Codes [Ber93] haben gegenüber LDPC-Codes einen gewissen Vorteil bei sehr kleinen Coderaten ($R < 1/2$). Für das Constellation-Programm der NASA (bemannte Flüge, z.B zum Mond oder Mars) und das Mars-Science-Lab (MSL) wurden LDPC-Codes ausgewählt. Allerdings wurde das Constellation-Programm Anfang 2010 aus finanziellen Gründen zurückgestellt.

In vielen Aktualisierungen von Standards finden die LDPC-Codes ebenfalls Einzug. Das digitale Fernsehen via Satellit (DVB-S2 ... Digital Video Broadcast via Satellit, 2. Generation, 2003) verwendet LDPC-Codes in Verbindung mit einem äußeren Bose-Chaudhuri-Hocquenghem-(BCH)-Code hoher Rate. Letzterer soll das Fehlerplateau absenken und sehr kleine Bitfehlerraten ermöglichen [Mor05]. Diese Kombination setzte sich bei der Standardisierung gegen andere Vorschläge aufgrund ihrer Leistungsfähig bei gegebener maximale Hardware-Komplexität durch. Die Blocklänge beträgt je nach Anwendung 64800 oder 16200 Bits. Es gibt eine Vielfalt von Coderaten ($1/4 \ldots 8/9$). Die Auswahl erfolgt je nach Modulationsart und Anwendung.

Der Standard IEEE 802.16e (Wi-MAX) spezifiziert quasi-zyklische LDPC-Codes mit Blockgrößen von $n = 576$ bis $n = 2304$ bei Coderaten von $R = 1/2$, $R = 2/3$, $R = 3/4$ oder $R = 5/6$. Ein äußerer Kanalcode zum Absenken der Bitfehlerrate ist hier nicht erforderlich, weil bei der Konstruktion der Kontrollmatrix darauf geachtet wurde, dass Bits an kritischen Positionen der quasi-zyklischen Struktur an mindestens zwei Paritätsgleichungen beteiligt sind. Der Standard IEEE 802.11n (Wi-Fi) verwendet ebenfalls quasi-zyklische Codes mit Blockgrößen von $n = 648$, $n = 1296$ und $n = 1944$ und vier verschiedenen Coderaten wie bei Wi-MAX.

Weitere Verbesserungen der Leistungsfähigkeit dieser Art von Kanalcodierung sind kaum zu erwarten, da es heute bereits möglich ist, einen Abstand zur Shannon-Grenze von weniger als einem Dezibel zu erreichen.

Der rechentechnische Aufwand ist im Allgemeinen jedoch noch groß. Hier gibt es noch Raum für Weiterentwicklungen. Die Anzahl von guten LDPC-Codes mit unterschiedlichen Coderaten ist bereits hoch und steigt noch weiter. Mit der wachsenden Popularität werden sie herkömmliche Kanalcodierungsmethoden aus vielen Anwendungen und Standards verdrängen. Ein weitestgehend offenes Feld bieten auch Anwendungen mit unterschiedlichem System-Design, wie zum Beispiel Multiple-Input-Multiple-Output(MIMO)-Systeme.

Literatur

[And07] Andrews, K.S.; Divsalar, D.; Dolinar, S.; Hamkins, J.; Jones, C.R.; Pollara, F.: The Development of Turbo and LDPC Codes for Deep-Space Applications. *Proc. of IEEE*, Vol.95, No.11, Nov. 2007, 2142–2156

[Ber93] Berrou, C.; Glavieux, A.; Thitimajshima, P.: Near Shannon limit error-corecting coding and decoding. *Proc. IEEE Intern. Conf. on Communications*, Geneva, Switzerland, May 1993, 1064–1070

[Che04] Chen, L.; Xu, J.; Djurdjevic, I.; Lin, S.: Near-Shannon-Limit Quasi-Cyclic Low-Density Parity-Check Codes. *IEEE Trans. on Communications*, Vol.52, No.7, 2004, 1038–1042

[Fos04] Fossier, M.P.C.: Quasi-Cyclic Low-Density Parity-Check Codes From Circulant Permutation Matrices. *IEEE Trans. on Information Theory*, Vol.50, No.8, 2004, 1788–1793

[Gal63] Gallager, R.G.: Low-Density Parity-Check Codes. Cambridge, MA: MIT Press, 1963

[Hag96] Hagenauer, J.; Offer, E.; Papje, L.: Iterative Decoding of Binary Block and Convolutional Codes. *IEEE Trans. on Information Theory*, Vol.42, No.2, 1996, 429–445

[Joh00] Johnson, S.J.: *Introducing Low-density Parity-check codes.* Published Internal Technical Report, Department of Electrical and Computer Engineering, University of Newcastle, Australia 2000

[Joh02] Johnson, S.J.; Weller, S.R.: Low-density parity-check codes: Design and decoding. Chapter in Wiley *Encyclopedia of Telecommunications*, 2003

[Lin04] Lin, S.; Costello, D.J.: *Error Control Coding.*, Prentice Hall, 2nd edition, 2004

[Mac96] MacKay, D.J.C.; Neal, R.M.: Near Shannon limit performance of low density parity check codes. *Electron. Letters*, Vol.32, August 1996, 1645-1646 (reprinted with printing errors corrected in Vol.33, 457–458)

[Mor05] Morello, A.; Mignone, V.: DVB-S2: The Second Generation Standard for Satellite Broad-band Services. *Proceedings of IEEE*, Vol.94, No.1, January 2005, 210–227

[Ric01] Richardson, T.J.; Shokrollahi, A.; Urbanke, R.L.: Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes. *IEEE Trans. on Information Theory*, Vol.47, No.2, 2001, 619–637

[Sha48] Shannon, C.E.: A Mathematical Theory of Communication. *Bell System Technical Journal*, Vol.27, 1948

[Tan81] Tanner, R.M.: A recursive approach to low complexity codes. *IEEE Trans. on Information Theory*, Vol.27, No.5, 1981, 533–547

[Tho04] Thorpe, J.; Andrews, K.; Dolinar, S.: Methodologies for Designing LDPC Codes Using Protographs and Circulants. *Proc. of IEEE Int. Symp. Inf. Theory (ISIT)*, Chicago, USA, 27 June - 2 July, 2004

[wwwSC] http://www.cs.toronto.edu/∼radford/ldpc.software.html, besichtigt am 22.Juni 2010