Zuverlässigkeitsbasierter Cross-Layer-Entwurf digitaler Übertragungssysteme

Dissertation

zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
(Dr.-Ing.)
der Technischen Fakultät
der Christian-Albrechts-Universität zu Kiel

vorgelegt von

Justus Christian Fricke

Kiel 2008

Tag der Einreichung: 18.09.2008
Tag der Disputation: 06.02.2009

Berichterstatter: Prof. Dr.-Ing. Peter Adam Höher
Prof. Dr.-Ing. Werner Henkel
Dr.-Ing. Ingmar Land

Kurzfassung

Die Funktionen digitaler Übertragungssysteme werden in übereinanderliegende Schichten systematisiert. Da üblicherweise nur benachbarte Schichten Informationen austauschen, können die Schichten unabhängig voneinander entworfen und gepflegt werden. Nachteil dieses schichtweisen Aufbaus ist, dass jede Schicht zusätzliche Verzögerungen und zusätzliches Datenaufkommen durch Protokollinformation verursacht. Häufig sind die einzelnen Schichten auf Grund ihres voneinander unabhängigen Entwurfs schlecht aufeinander abgestimmt. Der stetige Bedarf nach höheren Datenraten führte daher zu der Idee des schichtübergreifenden Systementwurfs (cross-layer design). Ziel des schichtübergreifenden Systementwurfs ist, durch zusätzlichen Informationsaustausch oder gar Auflösung der einzelnen Schichten Vorteile, z. B. höhere Datenraten oder geringere Verzögerungen, zu erzielen.

In dieser Arbeit werden schichtübergreifende Ansätze ausgehend von der untersten, der physikalische Schicht, betrachtet. Dafür werden die weichen Ausgangswerte des üblicherweise als letzte Komponente der physikalischen Schicht angeordneten Kanaldecodierers verwendet, um die Bit- oder die Wortfehlerwahrscheinlichkeit zu berechnen. Mit der Bit- bzw. Wortfehlerwahrscheinlichkeit steht ein Maß zur Schätzung der Übertragungsqualität zur Verfügung, das schichtübergreifend genutzt werden kann. Ein großer Vorteil dieser Vorgehensweise die Fehlerwahrscheinlichkeit zu berechnen ist ihre Unabhängigkeit von Kanal, Modulationsformat und verwendetem Kanalcode.

Im ersten Teil der Arbeit wird nach einer kurzen Einführung der betrachteten Kanalcodes und des verwendeten Systemmodells erläutert, wie mit Hilfe des Decodierers bzw. seiner weichen Ausgangswerte Bit- und Wortfehlerwahrscheinlichkeit berechnet werden können. Ferner werden verschiedene Fehlerquellen und ihre Auswirkung auf die Schätzung untersucht. Der zweite Teil der Arbeit umfasst verschiedene schichtübergreifende Anwendungsmöglichkeiten der Fehlerwahrscheinlichkeiten. Eher anwendungsorientierte Ansätze stellen die Verwendung der Fehlerwahrscheinlichkeiten als Neuanforderungskriterium für ARQ-Verfahren und die güteorientierte Decodierung dar. Die Verwendung als Neuanforderungskriterium ermöglicht einen Abtausch zwischen der für eine Anwendung maximal zulässigen Fehlerrate und der Datenrate. Bei der güteorientierten Decodierung werden die empfangenen Daten nur dann decodiert, wenn die Fehlerwahrscheinlichkeit für die Ansprüche einer gegebene Anwendung zu groß ist. Durch Ablaufplanung und Medium-Access-Control werden die begrenzten Ressourcen eines Kommunikationssystems zwischen den unterschiedlichen Teilnehmern aufgeteilt. Auch hier kann die Kenntnis der Fehlerwahrscheinlichkeiten der einzelnen Teilnehmer dazu genutzt werden, um die Datenrate des Systems zu steigern. Schließlich können die Fehlerwahrscheinlichkeiten auch zur Berech-

nung von Routing-Metriken genutzt werden, um einen möglichst geeigneten Pfad durch ein Netzwerk zu finden. In allen Fällen können durch die schichtübergreifende Kenntnis bzw. Anwendung der aus der Decodierung gewonnenen Fehlerwahrscheinlichkeit positive Effekte erzielt werden.

Stichwörter: schichtübergreifender Systementwurf (cross-layer design), Kanalcodierung, soft-output Decodierung, Wortfehlerwahrscheinlichkeit, Bitfehlerwahrscheinlichkeit, zuverlässigkeitsbasierte Neuanforderungskriterien, güteorientierte Decodierung, hybrid ARQ, zuverlässigkeitsbasiertes Routing, Routing-Metrik

Abstract

The operations of digital transmission systems can be organised in different layers which are arranged in a stack. Usually, only neighbouring layers exchange information between each other. Therefore, the different layers can be designed and modified without taking care of the other (non-neighbouring) layers. A disadvantage of this layered structure is that every layer introduces additional delay and causes additional data in terms of protocol information. Furthermore, the different layers often do not harmonise with each other due to their independent design. The continuous demand for higher data rates led to the idea of cross-layer design. The goal of cross-layer design are improvements such as higher data rates or lower delays obtained by additional information transfer between the layers or even dissolution of the layers.

In this thesis, cross-layer approaches based on the lowest layer of the protocol stack, the physical layer, are considered. Towards that goal, the soft output of the last component of the physical layer, the channel decoder, is used to calculate the bit or word error probability. The bit or word error probability is a measure of the transmission quality which can be applied in a cross-layer design. An advantage of this approach to calculate the error probability is its independence of channel, modulation format, and applied channel code.

The first part of this thesis covers a brief introduction of the system model and channel codes employed. The calculation of bit and word error probability by the channel decoder or its soft output is explained. Furthermore, different sources of error and their impact on the estimation of the error probabilities are investigated. The second part of the thesis covers different cross-layer applications of the error probabilities. Application-dependent approaches are the use of the error probabilities as retransmission criteria for ARQ protocols and for quality-oriented decoding. When used as a retransmission criterion, the error probabilities offer a trade-off between a maximum acceptable error rate of an application and the data rate. In quality-oriented decoding, the received data is only decoded if the error probability is too high for the requirements of a given application. The limited resources of a communication system are distributed among the different subscribers by scheduling and medium access control. The knowledge of the error probabilities of the individual subscribers can be used to increase the data rate of the system. Finally, the error probabilities can be also applied for the calculation of routing metrics to find a suitable path through a network. In all above-mentioned cases, the application of the error probabilities obtained by the decoding process in a cross-layer manner results in positive effects.

Keywords: cross-layer design, channel coding, soft-output decoding, word error probabil-

ity (WEP), bit error probability (BEP), reliability-based retransmission criteria, quality-oriented decoding, hybrid ARQ, reliability-based routing, routing metric

Inhaltsverzeichnis

1 Einleitung **1**
- 1.1 Motivation . . . 1
- 1.2 Aufbau der Arbeit . . . 2

2 Digitale Übertragungssysteme **5**
- 2.1 Schichtmodelle und schichtübergreifender Systementwurf . . . 5
  - 2.1.1 Schichtmodelle . . . 5
  - 2.1.2 Das OSI-Schichtmodell . . . 6
  - 2.1.3 Schichtübergreifender Systementwurf . . . 9
- 2.2 Zeitdiskretes Übertragungsmodell der physikalischen Schicht . . . 12
  - 2.2.1 Sender . . . 14
  - 2.2.2 Zeitdiskreter Ersatzkanal . . . 14
  - 2.2.3 Empfänger . . . 17
- 2.3 Dienstgüte (QoS) . . . 17
  - 2.3.1 QoS-Attribute der physikalischen Schicht . . . 17
- 2.4 Zusammenfassung . . . 21

3 Fehlerkontrolle **23**
- 3.1 Faltungscodes . . . 24
  - 3.1.1 Codierung . . . 24
  - 3.1.2 Decodierung . . . 25
- 3.2 Parallel verkettete Faltungscodes . . . 26
  - 3.2.1 Codierung . . . 26
  - 3.2.2 Decodierung . . . 27
- 3.3 LDPC-Codes . . . 28
  - 3.3.1 Codierung . . . 29
  - 3.3.2 Decodierung . . . 30
- 3.4 Fehlererkennende Codes . . . 31
  - 3.4.1 Codierung . . . 31
  - 3.4.2 Decodierung . . . 32
- 3.5 ARQ-Protokolle . . . 32
- 3.6 Hybride ARQ-Protokolle . . . 35
  - 3.6.1 Diversity-Combining . . . 35
  - 3.6.2 Code-Combining . . . 36
- 3.7 Zusammenfassung . . . 37

4 Zuverlässigkeitsinformation als Qualitätsmaß 39
4.1 Zuverlässigkeitsinformation . . . 39
4.2 Fehler bei der Berechnung der Zuverlässigkeitsinformation . . . 41
4.2.1 Suboptimale Decodierer . . . 41
4.2.2 Quantisierung durch Festkommadarstellung . . . 42
4.2.3 Hard-input soft-output Decodierung . . . 43
4.2.4 Korrelation der weichen Schätzwerte der Informationsbits . . . 44
4.3 Bitfehlerwahrscheinlichkeit . . . 50
4.3.1 Schätzung Bitfehlerwahrscheinlichkeit . . . 50
4.3.2 Fehler Bitfehlerwahrscheinlichkeit . . . 51
4.4 Wortfehlerwahrscheinlichkeit . . . 60
4.4.1 Schätzung der Wortfehlerwahrscheinlichkeit . . . 60
4.4.2 Fehler Wortfehlerwahrscheinlichkeit . . . 61
4.5 Bit- und Wortfehlerwahrscheinlichkeit . . . 73
4.6 Zusammenfassung . . . 76

5 Zuverlässigkeitsbasierte Neuanforderungskriterien 77
5.1 Systemmodelle für ARQ . . . 78
5.1.1 CRC-Code als Neuanforderungskriterium . . . 81
5.1.2 Wortfehlerwahrscheinlichkeit als Neuanforderungskriterium . . . 81
5.1.3 Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium . . . 83
5.2 Durchsatzeffizienz für verschiedene Neuanforderungskriterien . . . 84
5.3 Packet-Combining . . . 87
5.3.1 Diversity-Combining . . . 87
5.3.2 Code-Combining . . . 88
5.4 Zusammenfassung . . . 90

6 Güteorientierte Decodierung 91
6.1 Abbruchbedingungen für iterative Strukturen . . . 91
6.1.1 Turbo-Decodierer . . . 92
6.1.2 BPA für LDPC-Code . . . 95
6.2 Decodierung in Netzwerken mit Weiterleitung . . . 97
6.2.1 Systembeschreibung . . . 97
6.2.2 Güteorientierte Decodierung . . . 100
6.3 Zusammenfassung . . . 102

7 Zuverlässigkeitsbasiertes Routing 103
7.1 Einfluss der physikalischen Schicht auf die effektive Pfadlänge . . . 104
7.1.1 Physikalische Schicht . . . 106
7.1.2 Sicherungsschicht . . . 107
7.1.3 Netzwerkschicht . . . 107
7.2 Routing-Metriken für abschnittsweise Fehlerkontrolle . . . 108
7.3 Routing-Metriken für durchgehende Fehlerkontrolle . . . 111
7.4 Zusammenfassung . . . 111

8 Zuverlässigkeitsbasierte Ablaufplanung 113
8.1 Systemmodell 114
8.2 Nutzermetriken 115
8.3 Fair-opportunistische Ablaufplanung 118
8.4 Zusammenfassung 121

9 Zuverlässigkeitsinformation in IDM 123
9.1 Systemmodell 124
9.2 Layer-weises ARQ in IDM 125
9.3 Zuverlässigkeitsbasierte Layer-Verteilung in IDM 127
9.4 Zusammenfassung 129

10 Zusammenfassung und Ausblick 131

A Notation 135

B Algorithmen zur soft-output Decodierung 141

C Ergänzende Erläuterungen 157

Literaturverzeichnis 163

Stichwortverzeichnis 173

Kapitel 1

Einleitung

1.1 Motivation

Digitale Übertragungssysteme übertragen Binärzeichen (Bits) und sind dadurch flexibel was die Anwendung der übertragenden Daten, z. B. Text, Sprache oder Video, angeht. In den letzten Jahren gab es dabei ein rasantes Wachstum in zwei Bereichen, zum einen im Bereich der verfügbaren Datenraten zum anderen bezüglich der Verbreitung von Anwendungen im digitalen Mobilfunk. Die Zunahme der Datenraten ermöglicht Anwendungen, wie z. B. digitale Videoübertragung für den digitalen Massenmarkt, die vorher nicht denkbar waren. Mit der Zunahme von Mobilfunkkommunikation, insbesondere Mobiltelefonen und drahtlosen Computernetzwerken, stieg dabei auch das Bestreben, alle digitalen Dienste auch „mobil“ zu ermöglichen.

Herkömmlicherweise sind digitale Kommunikationssysteme in Schichten aufgebaut, ein Referenzsystem hierfür bietet das ISO/OSI-Schichtmodell. Der schichtweise Aufbau macht den Systementwurf und Pflege solcher Kommunikationssysteme überschaubar und bietet eine große Flexibilität da Schichten getrennt voneinander ausgetauscht werden können. Die große Menge der unterschiedlichen Anwendungen, die mit Hilfe eines digitalen Kommunikationssystems realisiert werden können wird erst durch den schichtweisen Aufbau ermöglicht.

Andererseits zeigt sich gerade im digitalen Mobilfunk, dass der schichtweise Aufbau auch Nachteile mit sich bringt bzw. eine Auflösung der Schichten Vorteile, wie z. B. höhere Datenraten, ermöglicht. Durch den zunehmenden Druck, Systeme mit höheren Datenraten zu realisieren, fand diese Idee unter dem Stichwort „Cross-Layer Design“ (dt. schichtübergreifender Systementwurf) zunehmende Beachtung.

Die unterste Schicht eines digitalen Übertragungssystems stellt üblicherweise die physikalische Schicht dar, die die Aufgabe hat, die Bits über den physikalischen Kanal zu übertragen. In der überwiegenden Mehrheit der Systeme wird in dieser Schicht Kanalcodierung verwendet. Mit der Verwendung eines Kanalcodes können am Empfänger mit Hilfe eines Kanaldecodierers Übertragungsfehler korrigiert werden um so eine zuverlässige Übertragung zu ermöglichen. Ferner erlaubt die Kanalcodierung die Verwendung von geringeren Sendeleistungen bei vergleichbarer Zuverlässigkeit, aber geringerer Datenrate – also einen Abtausch zwischen Energieverbrauch, Zuverlässigkeit und Datenrate. Diese drei System-

parameter sind zweifellos auch Parameter, die für die Schichten über der physikalischen Schicht von großer Bedeutung sind. Besonders eine (zu) unzuverlässige Datenübertragung verhindert die Funktionsweise eines digitalen Kommunikationssystems in einer Weise, die den Betrieb unmöglich macht.

Auf Grund der guten Decodiereigenschaften und der sich bietenden Möglichkeit iterativer Decodierer erfreuen sich sogenannte *soft-output Decodierer* zunehmender Beliebtheit. Diese Decodierer liefern neben den decodierten und korrigierten Bits am Empfänger Informationen darüber, wie zuverlässig die decodierten Bits sind, also wie groß die Wahrscheinlichkeit darin verbliebener Fehler ist. Aus vielfältigen Gründen ist es in einem digitalen Kommunikationssystem – nicht nur für die physikalische Schicht – von Interesse, wie zuverlässig die Daten übertragen werden bzw. wie viele Fehler bei der Übertragung entstehen. Das Wissen darüber kann in schichtübergreifenden Ansätzen verwendet werden um ein System zu adaptieren bzw. die Funktionen der unterschiedlichen Schichten zu harmonisieren. Ein besonderer Vorteil vom Kanaldecodierer bereitgestellten Zuverlässigkeitsinformation ist, dass die Bits am Ausgang des Kanaldecodierers das Ende der physikalischen Schicht darstellen, also alle Einflüsse der physikalischen Schicht enthalten. Dazu zählen z. B. der Einfluss des Kanals, des verwendeten Modulationsformats und des verwendeten Kanalcodes.

In dieser Arbeit wird die Möglichkeit untersucht, die von Kanaldecodierern zur Verfügung gestellte Zuverlässigkeitsinformation in schichtübergreifenden Ansätzen zu nutzen. Dafür werden verschiedene gängige Kanalcodes und Decodierer in Hinblick auf die Möglichkeit untersucht, aus der von ihnen bereitgestellte Zuverlässigkeitsinformation Wahrscheinlichkeiten für Übertragungsfehler zu berechnen. Die so gewonnenen Fehlerwahrscheinlichkeiten werden dann in verschiedenen schichtübergreifenden Ansätzen genutzt, um, je nach Verwendung, unterschiedliche positive Effekte zu erzielen. Im Vordergrund der Arbeit stehen dabei folgende Fragen:

- Wie können Bit- und Wortfehlerwahrscheinlichkeit aus den Ausgangswerten von Kanaldecodierern geschätzt werden?
- Wie exakt sind diese Schätzwerte (auch unter nicht idealen Umständen)?
- Welche schichtübergreifenden Anwendungsmöglichkeiten gibt es für Bit- und Wortfehlerwahrscheinlichkeit?
- Was wird durch die schichtübergreifende Anwendung gewonnen?

1.2 Aufbau der Arbeit

Die Kapitel 2 und 3 dienen der Einführung der für das Verständnis der Arbeit notwendigen Grundlagen. In Kapitel 2 werden kurz Schichtmodelle und schichtübergreifender Systementwurf dargestellt. Ferner werden Systemmodell der physikalischen Schicht und wesentliche Größen dieses Modells eingeführt. Alle Ansätze dieser Arbeit beruhen auf der physikalischen Schicht, so dass diese detailgetreuer als die anderen Schichten modelliert wird.

Als Teil der physikalischen Schicht spielt die Fehlerkontrolle die wesentlichste Rolle in dieser Arbeit. Daher werden die verwendeten Fehlerkontrollmechanismen in Kapitel 3 in einer Tiefe dargestellt, die es erlaubt, die Ergebnisse der Arbeit nachzuvollziehen.

Kapitel 4 erläutert die Schätzung von Bit- und Wortfehlerwahrscheinlichkeit mit Hilfe von soft-output Decodierern. Neben den Erläuterungen zur Schätzung mit unterschiedlichen Decodierern wird auch untersucht, inwiefern die Qualität der Schätzung durch suboptimale Decodierer beeinflusst wird. Theoretische Überlegungen zur Genauigkeit von Bit- und Wortfehlerschätzung beenden dieses Kapitel, dessen Inhalte teilweise in [FSH06], [FH07] und [FH08b] vorab veröffentlicht wurden.

Kapitel 5 behandelt die Verwendung der in Kapitel 4 eingeführten Schätzwerte im Zusammenhang mit ARQ-Verfahren. Durch die Verwendung von Fehlerwahrscheinlichkeiten als Neuanforderungskriterium sind je nach Anwendung teilweise beachtliche Durchsatzsteigerungen möglich. Die Grundidee, die Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium zu verwenden, wurde in [BFH06] bzw. [FH08a] veröffentlicht.

In Kapitel 6 wird die Idee der güteorientierten Decodierung vorgestellt. Dahinter steht die Idee, dass nur dann decodiert wird, wenn die Anforderungen einer Anwendung sonst nicht erfüllt werden können. Diese Idee wird sowohl für iterative Decodierer als auch Netzwerke mit Relays betrachtet. Der Abschnitt über Relays wurde in [FBH08] vorab veröffentlicht.

Die Verwendung von Zuverlässigkeitsinformation für die Suche nach dem günstigsten Pfad für die Übertragung in Netzwerken wird in Kapitel 7 vorgestellt. Dabei werden zwei unterschiedliche Szenarien der Fehlerkontrolle betrachtet, abschnittsweise und durchgehende Fehlerkontrolle. Die Ergebnisse zur abschnittsweisen Fehlerkontrolle wurden in [FRH07] vorabveröffentlicht.

In Kapitel 8 wird die Zuverlässigkeitsinformation genutzt, um festzustellen, welcher Nutzer geeigneterweise als nächstes Zugriff auf Kanalressourcen erhält. Durch Berücksichtigung der Fehlerwahrscheinlichkeit der einzelnen Nutzer können dabei höhere Systemdatenraten erzielt werden.

Die Ideen der schichtübergreifenden Nutzung von Zuverlässigkeitsinformation entstammen ursprünglich Vorschlägen für auf Interleave-Division-Multiplexing (IDM) bzw. Interleave-Division-Multiple-Access (IDMA) basierenden Systemen [FHS05a], [FHS05b], [FHS05c], [SHF05]. In Kapitel 9 werden einige in [KFH08] veröffentlichte IDM-spezifische Ansätze erläutert. Darin wird der layer-weise Aufbau der Übertragungsstruktur in Verbindung mit Leistungszuweisung und anwendungsspezifischen Dienstgüteanforderungen ausgenutzt. Die Idee des layer-weisen ARQ wurde vorab in [HSF08] veröffentlicht.

Die wesentlichen Ergebnisse werden am Ende der Arbeit zusammenfassend dargestellt. Die Ergebnisse der Arbeit beinhalten die Entwicklung bzw. Weiterentwicklung von Algorithmen zur Decodierung. Ferner haben die zur Decodierung verwendeten Algorithmen einen wesentlichen Einfluss auf die Qualität der Zuverlässigkeitsinformation. Aus diesem Grund sind sämtliche verwendeten Algorithmen in einer einheitlichen Darstellung im Anhang wiedergegeben.

Kapitel 2

Digitale Übertragungssysteme

2.1 Schichtmodelle und schichtübergreifender Systementwurf

2.1.1 Schichtmodelle

Moderne digitale Übertragungssysteme sind äußerst komplexe Gebilde. Ihre Komplexität rührt dabei vor allem aus der Vielzahl der unterschiedlichen Aufgaben, die sie bewältigen müssen. Darunter ist nicht nur die vom Nutzer wahrgenommene Aufgabe, z. B. ein Telefongespräch, sondern auch die zunehmende Integration von unterschiedlichen Diensten. So soll ein Mobiltelefon nicht mehr nur Telefonie, sondern auch SMS, E-Mail, MMS, Faxversand, Navigation und Internetverbindungen ermöglichen. Dabei sind dies nur die vom Nutzer wahrgenommenen Dienste. Gleichzeitig soll eine Internetverbindung oder ein Telefongespräch über unterschiedliche existierende Infrastrukturen wie z. B. GSM, UMTS oder WiFi (IEEE 802.11) möglich sein. Diese hohe Komplexität macht die Systeme sehr unübersichtlich, was wiederum Entwurf und Wartung erschwert.

Um den Aufwand beherrschbar zu gestalten, werden digitale Übertragungssysteme üblicherweise in übereinander liegenden Schichten strukturiert. Dabei wird ein System in so viele Schichten aufgeteilt, dass jede Schicht nur wenige, klar definierte und nach Möglichkeit zusammengehörige Aufgaben für die Kommunikation wahrnimmt. Dies hat den Vorteil, dass einzelne Schichten getrennt voneinander entworfen und auch im Nachhinein geändert werden können, ohne das Gesamtsystem zu berücksichtigen. Benachbarte Schichten tauschen Daten mit Hilfe dieser zur Verfügung gestellten Dienste aus (vertikaler Datenaustausch). Eine Schicht stellt dabei *Dienste* (engl. services) für die darüber liegende Schicht zu Verfügung und nutzt die von der darunter liegenden Schicht angebotenen Dienste. Die Kommunikation innerhalb einer Schicht (zwischen Sender und Empfänger) findet zwischen *Einheiten (engl. entities)* statt (horizontaler Datenaustausch). Dieser symmetrische Austausch (Sender- und Empfängereinheit in einer Schicht müssen sich entsprechen) wird als *Protokoll* bezeichnet.

Als Referenzmodell für die Beschreibung der Schichten in Kommunikationssystemen hat sich das Open System Interconnection (OSI) Modell durchgesetzt. Dieses besteht aus sieben Schichten und wurde von der ISO und IEC standardisiert [ISO94]. Häufig wird auch

von anderen Schichtmodellen gesprochen, wie z. B. dem 4-Schichtmodell [KR05]. In vielen Systemen sind einzelne Schichten allerdings gar nicht klar trennbar. Ferner ist [ISO94] als ein *Referenzmodell* gedacht, um Schichten bzw. deren Aufgaben spezifizieren und benennen zu können. Es stellt ausdrücklich keine Implementierungsanleitung dar. Andererseits ist es möglich, die Schichten des ISO/OSI-Modells *jedem* digitalen Kommunikationssystem zuzuordnen (sogar dem oben genannten 4-Schichtmodell). Als ein solches Referenzmodell soll [ISO94] auch in dieser Arbeit verwendet werden: Um bestimmte Funktionen eines beliebigen Kommunikationssystems zu benennen und dabei die Vergleichbarkeit mit anderen Systemen zu wahren.

2.1.2 Das OSI-Schichtmodell

Im Folgenden soll kurz der Aufbau und die Funktionsweise des OSI-Schichtmodells wiedergegeben werden. Abb. 2.1 zeigt die sieben Schichten des OSI-Schichtmodells.

OSI-Schicht	Name	Beispiele
7	Anwendungsschicht	http, ftp, SMTP, Telefonie
6	Darstellungsschicht	MPEG, ASCII
5	Sitzungsschicht	
4	Transportschicht	TCP
3	Netzwerkschicht	IP, UMTS L3
2	Sicherungsschicht	UMTS L2, IEEE 802.3 MAC
1	Physikalische Schicht	UMTS L1, IEEE 802.3 PHY
—	Übertragungsmedium	Luft, Glasfaserkabel, Kupfer

Abbildung 2.1: Darstellung des OSI-Schichtmodells und einige Beispiele für die einzelnen Schichten.

In der Regel findet ein vertikaler Informationsaustausch nur zwischen benachbarten Schichten statt. Der horizontale Informationsaustausch zwischen *(N)-Schicht-Einheiten* eines Senders und eines Empfängers wird durch schichtspezifische Protokolle geregelt. Benachbarte Schichten tauschen sogenannte *Protocol-Data-Units (PDUs)* aus. Eine PDU der $(N-1)$-ten Schicht ($(N-1)$-PDU) setzt sich dabei aus der *Service-Data-Unit (SDU)* der nächsthöheren (N-ten) Schicht und den von der $(N-1)$-ten Schicht ergänzten *Protokollkontrollinformation* (engl. protocol-control-information, PCI) zusammen. Eine entsprechende Darstellung findet sich in Abb. 2.2. Die PCI dient der vertikalen Kommunikation der Protokolle zwischen Sender und Empfänger. Eine PDU wird im Allgemeinen Sprachgebrauch auch als *Paket* und eine SDU als *Nachricht* bezeichnet. Die Protokollkontrollinformation beinhaltet zusätzliche Daten, die für die Kommunikation zwischen $(N-1)$-Schicht-Einheiten benötigt werden. Eine Nachricht der Anwendungsschicht wächst also durch die PCI der darunterliegenden Schichten immer weiter an, so dass das letztendlich übertragende Paket weit mehr Daten als die ursprüngliche Nachricht enthält.

Die Aufgaben der unterschiedlichen Schichten sollen hier nur kurz und auf die für die Arbeit wichtigsten Mechanismen reduziert dargestellt werden. Eine genauere Beschreibung mit Erläuterungen findet sich in [BB99] bzw. direkt in [ISO94].

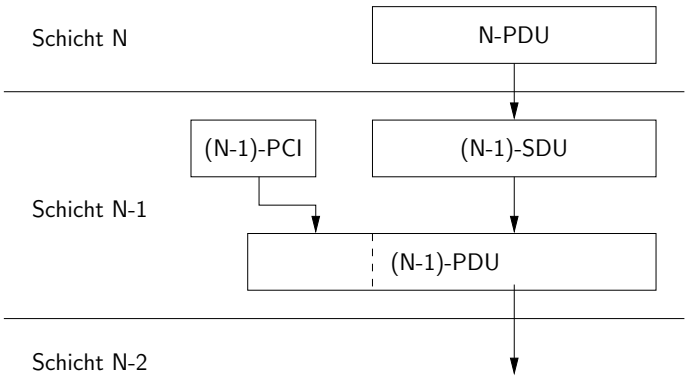

Abbildung 2.2: Darstellung des Datentransports von einer Schicht N zur darunterliegenden Schicht $N-1$.

Physikalische Schicht (physical layer)

Die physikalische Schicht[1] überträgt Bits über das physikalische Medium. Dazu gehört z. B. die entsprechende Darstellung und Anpassung an den Übertragungskanal (Kanalcodierung und Modulation) sowie die empfängerseitige Rückgewinnung der übertragenen Symbole (z. B. Kanalschätzung, Entzerrung, Decodierung). Auch Multiplexing für unterschiedliche, zeitgleiche Übertragung ist Aufgabe der physikalischen Schicht.

Sicherungsschicht (data link layer)

Die Sicherungsschicht stellt die logische Verbindung zwischen zwei Einheiten dar (die physikalische Schicht sendet an alle potentiellen Empfänger). Sie regelt vor allem die Segmentierung von Datenströmen für die physikalische Schicht und verwaltet den Zugriff auf den physikalischen Kanal (medium access). Ebenfalls Aufgabe der Sicherungsschicht ist die Überprüfung auf Übertragungsfehler und eventuelle Gegenmaßnahmen, z. B. durch erneute Übertragung der fehlerhaften Daten (logical link control).

Netzwerkschicht (network layer)

Die Netzschicht verbindet mehrere Einheiten zu einem Netz bestehend aus *Netzknoten*. Die Netzschicht ist dann wichtig, wenn zwei Schicht-2-Einheiten nicht direkt miteinander kommunizieren können. Dies kann z. B. auf Grund großer örtlicher Entfernung der Fall sein. Zwei Netzknoten können Daten austauschen, indem diese über direkte Nachbarn so lange weitergereicht werden, bis sie schließlich den Zielknoten erreichen. Eine wichtige Aufgabe besteht dabei in der *Pfadsuche* (engl. routing). Darunter versteht man das Auffinden eines geeigneten Pfads durch das Netzwerk, um über diesen die 3-PDU zu übertragen. Damit schafft die Netzwerkschicht einen virtuellen Übertragungskanal zwischen beliebigen Netzknoten.

[1] Für die physikalische Schicht ist auch der deutsche Begriff *Bitübertragungsschicht* gebräuchlich. Dieser Begriff ist jedoch irreführend, da die physikalische Schicht häufig die einzige Schicht ist, die *keine* Bits, sondern Signale bzw. Wellen, überträgt.

Transportschicht (transport layer)

Die Transportschicht stellt die Übertragung von Daten zwischen einem Quell- und einem Zielnetzwerkknoten sicher. Dazu gehört auch die Segmentierung von Daten für die Netzwerkschicht und die erneute Übertragung verlorener oder fehlerhaft empfangener PDUs. Damit erfüllt die Transportschicht ähnliche Aufgaben wie die Sicherungsschicht – allerdings auf Netzebene.

Sitzungsschicht (session layer)

Die Sitzungsschicht vereinheitlicht die Kommunikation. Der Transport von Daten über die Netzwerkschicht muss nicht zwingenderweise kontinuierlich oder immer auf demselben Weg stattfinden. Denkbar ist auch ein paralleler Transport auf unterschiedlichen Wegen. Dabei bestimmt die Sitzungsschicht Anfang und Ende der Kommunikation und wer an der Kommunikation beteiligt ist.

Darstellungsschicht (presentation layer)

Die Darstellungsschicht behandelt Syntax und Semantik für die Kommunikation. Dazu gehört die Umwandlung zwischen Bits und anwendungsbezogenen Datenstrukturen. Auch Verschlüsselung und Quellencodierung wird normalerweise dieser Schicht zugeordnet.

Anwendungsschicht (application layer)

Eine Anwendungsschichteinheit stellt jeweils einen dem Nutzer angebotenen Dienst dar. Diese Dienste können sehr unterschiedliche sein, basieren jedoch alle auf Informationsaustausch.

Ein Schichtentwurf, der strikt dem OSI-Schichtmodell folgt, existiert in nahezu keinem Kommunikationssystem. Allerdings werden wiederum in nahezu allen Kommunikationssystemen die oben beschriebenen Aufgaben implementiert. Die Verarbeitung und Übertragung einer Nutzereingabe durch den Protokollstapel erfolgt auch oft in ähnlicher Reihenfolge. Aus diesem Grund werden die im OSI-Schichtmodell definierte Terminologie und Schichten im Weiteren zur Systembeschreibung verwendet.

Abb. 2.3 zeigt eine weitere nützliche Anwendung von Schichtmodellen. Sie enthält eine Darstellung unterschiedlicher Netzkomponenten, die jeweils unterschiedlich viele Schichten des OSI-Schichtmodells verarbeiten. Auf diese Weise ergeben sich unterschiedliche Funktionalitäten. In einem *Repeater* ist nur die physikalische Schicht implementiert – ein empfangenes Signal wird erneut gesendet (wiederholt) und somit verstärkt. Ein *Relay* implementiert außerdem die Datensicherungsschicht. Dadurch wird sichergestellt, dass das gesendete Signal fehlerfrei ist, was natürlich mit einem höheren Aufwand und einer eventuellen erneuten Übertragung der 2-SDU verbunden ist. Der *Router* stellt Verbindungen zu anderen Netzkomponenten auf der Netzwerkschicht (Netzknoten) her, d. h., er kann im Gegensatz zu Repeater und Relay Informationen mit Netzkomponenten austauschen, die nicht direkt benachbart sind. Der Vorteil der Schichtstruktur ist hierbei, dass Relay und Router nicht auf eine bestimmte physikalische Schicht angewiesen sind und somit

z. B. unterschiedliche Übertragungsmedien nutzen können. So kann ein Teil der Übertragungsstrecke zum Ziel über Luft und ein anderer über Kupferleitung zurückgelegt werden. Ähnliches gilt für das Relay, dessen Sicherungsschicht auf unterschiedliche Realisierungen der physikalischen Schichten ausgelegt sein kann.

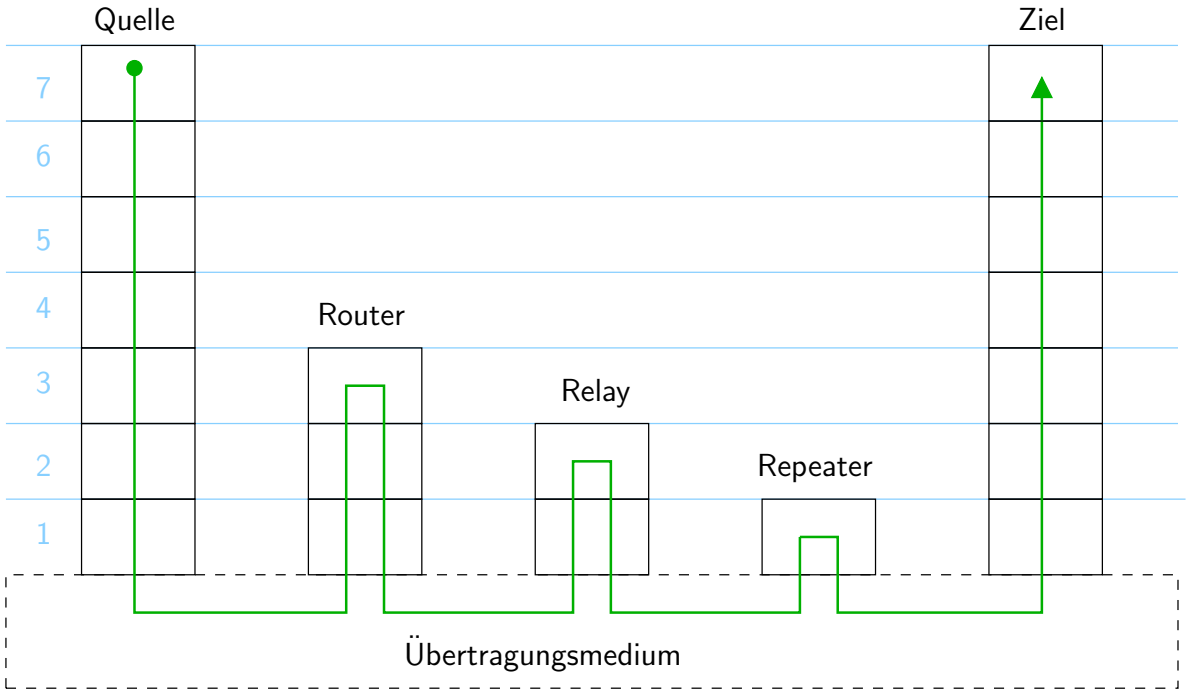

Abbildung 2.3: Netzkomponenten verarbeiten unterschiedlich viele Schichten und erfüllen damit verschiedene Zwecke.

2.1.3 Schichtübergreifender Systementwurf

Schichtmodelle ermöglichen einen vereinfachten und modularen Systementwurf, weisen aber auch Nachteile auf. Ein wesentlicher Nachteil ist die redundante Realisierung bestimmter Funktionen in mehreren Schichten. So weist häufig sowohl die Sicherungs- als auch die Transportschicht ein ARQ-Protokoll zur Korrektur von falsch oder nicht empfangenen PDUs auf. Dies kann bei der Abarbeitung des Protokollstapels zu unnötigen Verzögerungen führen. Generell wird durch jede weitere Schicht eine zusätzliche Verzögerung eingeführt. Selbiges gilt für Daten wie Adressen, Zeitmarken oder Informationen zur Übertragungsqualität, die unter Umständen in mehreren Schichten als Protokollinformation hinzugefügt und dadurch mehrfach übertragen werden. Abgesehen davon fügt jede enthaltene Schicht seine Protokollinformation zu den Nutzdaten hinzu, was die effektiv für die Nutzdaten zur Verfügung stehende Datenrate reduziert. Wenn PDUs einer Schicht eine bestimmte Größe aufweisen und SDUs darüberliegender Schichten inklusive der PCI eine abweichende Größe aufweisen, muss diese angepasst werden:

- Ist die SDU zu klein, so wird der verbleibende Platz für gewöhnlich mit Nullen aufgefüllt, bis die gewünschte Länge erreicht ist (Zero-Padding).

- Ist die SDU zu groß, so wird sie in Segmente der maximal erlaubten Größe aufgeteilt (Segmentation).

Besonders in diesen beiden Fällen wird das Verhältnis zwischen Nutzdaten und Kontrolldaten ungünstig beeinflusst.

Sogar der eigentliche Vorteil des schichtweisen Systementwurfs, die Tatsache, dass die Realisierung der anderen Schicht nicht berücksichtigt wird, kann sich als nachteilig erweisen. Dies ist insbesondere dann der Fall, wenn einzelne Schichten extreme Charakteristiken, wie z.B. relativ hohe Fehlerraten bei der mobilen Funkübertragung oder der Bedarf sehr hoher Datenraten für hochauflösende Videoanwendungen aufweisen. Im Falle der hohen Fehlerraten können schlecht aufeinander abgestimmte Fehlerschutzmaßnahmen zu sehr geringen Datenraten führen, im Falle von hohen Anforderungen an die Nutzdatenrate können sich Kontrollinformationen und durch einen zu großen Protokollstapel hervorgerufene Verzögerungen als unüberwindbare Hindernisse erweisen.

Durch den stetigen Trend zu immer höheren Anforderungen in digitalen Kommunikationssystemen stieß daher die Idee des *schichtübergreifenden Systementwurfs* (engl. cross-layer design) auf größer werdendes Interesse. Verallgemeinert kann das Ziel des schichtübergreifenden Systementwurfs folgendermaßen beschrieben werden:

> Schichtübergreifender Systementwurf schafft eine im klassischen Schichtmodell nicht vorgesehene Verknüpfung von verschiedenen Schichten um einen positiven Effekt zu erzielen.

Dabei sollte nicht vergessen werden, dass durch Auflösung des Schichtmodells auch dessen Vorteile verloren gehen (einfacher Entwurf, einfache Optimierung und Wartung, klare Architektur) [KK05], [SM05b]. Schichtübergreifende Ansätze weisen zwangsläufig einen höheren Implementierungsaufwand auf. Einzelne Funktionen, die sonst getrennt voneinander arbeiten würden, sind nicht mehr einzeln gestalt- und austauschbar. Zwei Quellen üblicher Missverständnisse im Zusammenhang mit schichtübergreifendem Systementwurf sollen im Folgenden ausgeräumt werden.

2.1.3.1 Schichtübergreifender Systementwurf – eine neue Idee?

Schichtübergreifender Systementwurf ist nicht neu. So findet z.B. in UMTS zwischen physikalischer Schicht (L1) und Netzwerkschicht (L3) Datenaustausch statt [ETSe]. Dies ist nach dem OSI-Schichtmodell eigentlich nicht vorgesehen. Ein anderes Beispiel stellen Netze dar, die auf dem asynchronen Transfer-Modus (ATM) beruhen. Diese Netze verfügen über eine Management-Backplane, die als eine vertikale Schicht die anderen Schichten steuert [BB99]. In Grunde genommen beinhaltet jedes System, das den vertikalen Informationsaustausch nicht auf benachbarte und den horizontalen Informationsaustausch nicht auf Einheiten der gleichen Schicht begrenzt, schichtübergreifende Komponenten. Relativ neu hingegen ist der Ansatz schichtübergreifenden Systementwurf als einen allgemeinen Ansatz zur Verbesserung schichtbasierter Systeme zu verwenden. Die Idee vom systematischen schichtübergreifenden Systemdesign geht als „soft layering“ auf [Coo83] zurück. Darin wurde festgestellt, dass der Austausch von Statusinformation zwischen Protokollen

zu Leistungssteigerungen führen kann. Der Begriff „cross-layer“ taucht wahrscheinlich erstmals 1997 im Zusammenhang mit der Adaption von Multimedia-Anwendungen in der Literatur auf [ICPW97].

2.1.3.2 Schichtübergreifende Optimierung

Häufig wird von *schichtübergreifender Optimierung* (engl. cross-layer optimisation) gesprochen (z. B. [VAG05] und Referenzen darin). Da jede Schicht eine Viel zahl von verschiedenen Realisierungen und damit auch unterschiedlichen Konfigurationsmöglichkeiten aufweist, ist eine gemeinsame Optimierung praktisch und theoretisch nicht möglich. Bei solchen Optimierungen handelt es sich daher entweder um theoretische Ansätze, die eine bestimmte Gruppe abstrakter Funktionen optimieren, oder um praktische Ansätze, die den gemeinsamen Einsatz einer kleinen Anzahl Protokolle oder abstrahierter Funktionen optimieren. Ein gängiger Ansatz ist eine Optimierung mit Hilfe von Kostenfunktionen (engl. cost functions) oder Nutzenfunktionen (engl. utility functions), z. B. in [BY04]. Das Problem bei diesen Ansätzen ist, diese Funktionen zu finden, die einerseits einfach genug sind, um eine Optimierung zuzulassen, andererseits aber Zusammenhänge und Auswirkungen in Kommunikationssystemen ausreichend genau wiedergeben.

Aus diesem Grund wird in dieser Arbeit von schichtübergreifendem Systementwurf gesprochen, wobei eine lokale Optimierung bestimmter Parameter nicht ausgeschlossen wird.

2.1.3.3 Arten schichtübergreifenden Systementwurfs

Die Anzahl der möglichen schichtübergreifenden Ansätze ist schier unbegrenzt. Eine Reihe möglicher Ansätze für praktische Systeme findet sich in [RI04]. Eine gute Übersicht über unterschiedliche Ansätze und deren Kategorisierung findet sich in [SM05b], woraus auch Abb. 2.4 in ähnlicher Form entnommen ist. Abb. 2.4 zeigt auf welche Art gegen das OSI-Schichtmodell verstoßen wird, um positive Effekte zu erzielen. Die unterschiedlichen schichtübergreifenden Ansätze werden im Folgenden näher erläutert.

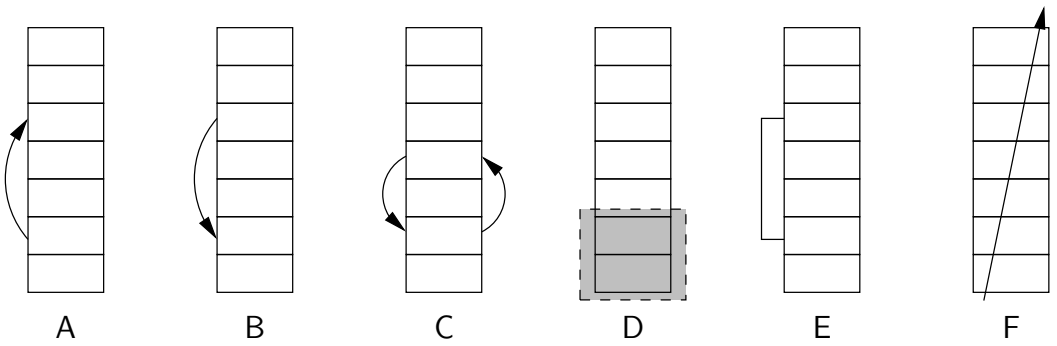

Abbildung 2.4: Darstellung unterschiedlicher Möglichkeiten des schichtübergreifenden Systementwurfs nach [SM05b].

A.–C. Schichtübergreifende Schnittstellen: Die Fälle A–C stellen jeweils Erweiterungen des Schichtstapel um schichtübergreifende Schnittstellen dar. Diese können genutzt werden, um bestimmte Parameter einer anderen Schicht so zu setzen, dass die Funktionen der beiden Schichten möglichst gut aufeinander abgestimmt sind. Dabei werden Schnittstellen für die Kommunikation zweier nicht benachbarter Schichten geschaffen. Im Falle von A liegt ein Informationsfluss von einer tieferen zu einer höheren Schicht, im Falle von B von einer höheren zu einer tieferen Schicht vor. Fall C stellt einen wechselseitigen Informationsfluss dar. Dieser kann z. B. in einer iterativen Art und Weise optimale Parameter für die beiden Schichten bestimmen.

D. Zusammenfassen von Schichten Die Aufgaben aneinandergrenzender Schichten können in einer Schicht zusammengefasst werden. Dies kann z. B. durch gemeinsame PCI zweier Schichten, also einem gemeinsamen Header erreicht werden. So entsteht ein neuer *Super-Layer* in dem die Funktionen der ursprünglichen Schichten nicht mehr von einander getrennt werden können. In diesem Fall (Abb. 2.4 D) wären das physikalische Schicht und Sicherungsschicht. Ein einfacher Repeater wie in Abb. 2.3 ist damit nicht mehr möglich.

E. Ankopplung im Entwurf Ein oder mehrere Schichten können so entworfen werden, dass sie besonders gut miteinander arbeiten. Zum Beispiel kann eine Netzwerkschicht auf eine bestimmte physikalische Schicht ausgerichtet werden (z. B. UMTS). Die Gefahr hierbei ist, dass eine andere Implementierung einer Schicht dann eventuell sehr schlecht oder gar nicht funktioniert.

F. Vertikale Harmonisierung von Schichten Dies ist der in der Literatur am häufigsten anzutreffende Ansatz. Dabei wird versucht, die Parameter der einzelnen Schichten so einzustellen, dass die einzelnen Schichten gut miteinander harmonisieren, also z. B. möglichst geringe Verzögerungen entstehen.

2.2 Zeitdiskretes Übertragungsmodell der physikalischen Schicht

Von den im letzten Abschnitt eingeführten Schichten spielt in dieser Arbeit die physikalische Schicht und in ihr die Kanalcodierung eine wichtige Rolle. Für die Übertragung auf der physikalischen Schicht wird das in Abb. 2.5 dargestellte Basisbandübertragungsmodell angenommen. Die zu übertragenden K (Info-)Bits

$$\boldsymbol{u} = [u_1, ..., u_k, ..., u_K] \text{ mit } u_k \in \mathbb{B} = \{0, 1\} \tag{2.1}$$

stellen dabei die 2-SDU dar, die von der Sicherungsschicht zur Übertragung an die physikalische Schicht weitergereicht wird. Diese wird von einem Kanalcodierer (Vorwärtsfehlerschutz, engl. forward error correction (FEC)) codiert und das resultierende Codewort der Länge N

$$\boldsymbol{c} = [c_1, ..., c_n, ..., c_N] \text{ mit } c_n \in \mathbb{B} = \{0, 1\} \tag{2.2}$$

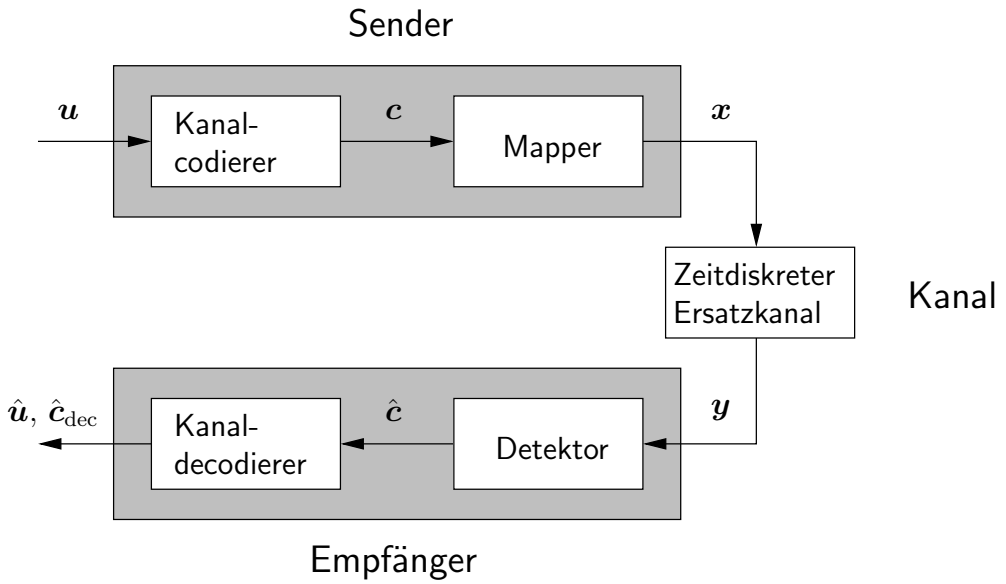

Abbildung 2.5: Zeitdiskretes Modell der physikalischen Übertragungsschicht.

für die Übertragung auf eine Sequenz M diskreter Sendesymbole aus einem Modulationsalphabet $\mathbb{M}$

$$\boldsymbol{x} = [x_1, ..., x_m, ..., x_M] \text{ mit } x_m \in \mathbb{M}, \tag{2.3}$$

abgebildet. Diese werden über einen zeitdiskreten Ersatzkanal übertragen, der am Empfänger die, im Allgemeinen komplexe, Sequenz der Abtastwerte der Länge M

$$\boldsymbol{y} = [y_1, ..., y_m, ..., y_M] \text{ mit } y_m \in \mathbb{C}, \tag{2.4}$$

erzeugt. Der Detektor bestimmt aus der Sequenz der Abtastwerte $\boldsymbol{y}$ Schätzwerte der übertragenen Symbole $\hat{\boldsymbol{x}}$ bzw. bestimmt die Sequenz der geschätzten Codebits $\hat{\boldsymbol{c}}$, die diesen Symbolen entsprechen. Die in der Empfängerstruktur verwendeten Werte hängen von den verwendeteten Methoden ab, die mit weichen oder harten Schätzwerten arbeiten können. Die harten Schätzwerte eine Größe sind mit einem spitzen Dach $\hat{\cdot}$, die weichen Schätzwerte mit einem runden Dach $\breve{\cdot}$ gekennzeichnet. Weiche Schätzwerte enthalten neben dem harten Schätzwert seine Zuverlässigkeit, z. B. die Fehlerwahrscheinlichkeit. Ein weicher Schätzwert für binäre Symbole ist z. B. das *Log-Likelihood-Verhältnis* (engl. log-likelihood ratio, LLR), auch als Log-Likelihood-Wert bezeichnet. Ein LLR eines binären Wertes wird mit einer Tilde $\tilde{\cdot}$ gekennzeichnet. Das Prinzip von Zuverlässigkeitsinformation und Soft-Verarbeitung wird in Abschnitt 3 genauer erläutert. Im einfachen Fall von BPSK-Übertragung gilt $M = N$, wobei die Sequenzen $\hat{\boldsymbol{c}}$ und $\hat{\boldsymbol{x}}$ unter Berücksichtigung des BPSK-Mappings identisch sind. Wenn nicht anders angegeben, gilt im Weiteren $K = 288$.

Ein Decodierer kann harte Eingangswerte $\hat{\boldsymbol{c}}$ (*hard-input*) oder weiche Eingangswerte $\tilde{\boldsymbol{c}}$ (*soft-input*) verwenden. Ebenso können Decodierer harte Schätzwerte $\hat{\boldsymbol{u}}$ (*hard-output*) oder weiche Schätzwerte $\tilde{\boldsymbol{u}}$ (*soft-output*) der Infobits liefern [HROW07]. Die Sequenzen $\hat{\boldsymbol{c}}$ und $\hat{\boldsymbol{c}}_{\text{dec}}$ stellen beide Schätzwerte des übertragenden Codeworts dar. Allerdings ist $\hat{\boldsymbol{c}}$ das geschätzte Codewort vor der Decodierung und $\hat{\boldsymbol{c}}_{\text{dec}}$ das geschätzte Codewort nach der

Decodierung[2], also das dem geschätzten Infowort $\hat{\boldsymbol{u}}$ entsprechende Codewort.

Im Fall von höherstufiger Modulation oder zeitlich korrelierten Fading-Kanälen kann es sinnvoll sein, zwischen Codierer und Mapper einen Interleaver sowie einen entsprechenden Deinterleaver zwischen Detektor und Kanaldecodierer zu verwenden. Dadurch können Bündelfehler am Decodierereingang vermieden werden.

2.2.1 Sender

Der Sender in Abb. 2.5 ist als allgemeine Definition ohne wesentliche Einschränkung zu verstehen. Wichtig ist vor allem, dass er einen Kanalcodierer enthält, dessen Codesymbole auf komplexe Symbole eines Alphabets $\mathbb{M}$ abgebildet werden. Dieses Alphabet ist im einfachsten Fall für $\mathbb{M} = \{+1, -1\}$ *Binary Phase Sift Keying* (BPSK), kann aber bei entsprechender Definition auch kompliziertere Übertragungsverfahren wie Mehrantennenübertragung (MIMO) oder *Orthogonal-Frequency-Division-Multiplex* (OFDM) Übertragung beinhalten.

2.2.2 Zeitdiskreter Ersatzkanal

Der Übertragungskanal in Abb. 2.5 ist ein zeitdiskreter Ersatzkanal wie er in Abb. 2.6 detailliert dargestellt wird. Es existiert eine Vielzahl von Modellen für Übertragungskanäle. In dieser Arbeit finden aus Gründen der Einfachheit nur statistische Kanalmodelle Anwendung. Die in dieser Arbeit verwendeten werden im Folgenden kurz eingeführt.

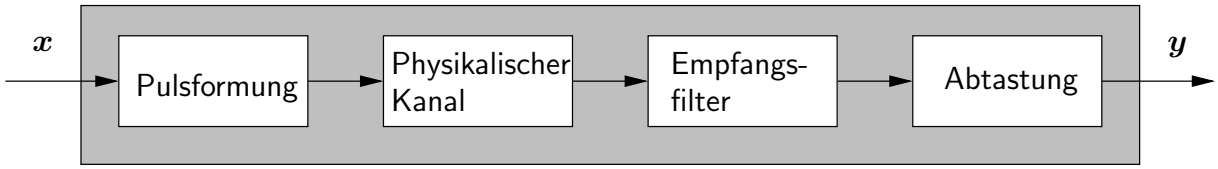

Abbildung 2.6: Zeitdiskreter Ersatzkanal mit Eingangssequenz $\boldsymbol{x}$ und Ausgangssequenz $\boldsymbol{y}$. Der Ersatzkanal modelliert die Effekte von Sendepulsformung, Übertragung über den physikalischen Kanal und Empfangsfilterung mit abschließender Abtastung.

2.2.2.1 AWGN-Kanal

Additives weißes gaußsches Rauschen (AWGN) kann verwendet werden um thermisches Rauschen zu modellieren [Skl01]. Thermisches Rauschen wird durch die thermische Elektronenbewegung in elektrischen Schaltungen verursacht [Joh28]. Es wird modelliert, indem ein Abtastwert des komplexwertigen Rauschprozesses empfängerseitig auf das Sendesignal aufaddiert wird. Die Wahrscheinlichkeitsverteilung des thermischen Rauschens wird als komplexwertige, mittelwertfreie Gaußverteilung

$$p_N(n) = \frac{1}{\pi\sigma_n^2} \exp\left(-\frac{|n|^2}{\sigma_n^2}\right) \tag{2.5}$$

[2]Da noch keine Decodierung stattfand, muss $\hat{\boldsymbol{c}}$ im Gegensatz zu $\hat{\boldsymbol{c}}_{\text{dec}}$ kein gültiges Codewort darstellen.

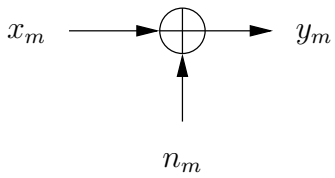

Abbildung 2.7: AWGN-Kanal.

angenommen. Die Rauschvarianz ist $\sigma_n^2 = \mathrm{E}\{|n|^2\}$, die durchschnittliche Energie pro Symbol ist $\sigma_x^2 = \mathrm{E}\{|x|^2\}$. Das Verhältnis daraus ergibt sich zu

$$\gamma_s = \frac{\sigma_x^2}{\sigma_n^2} \,. \tag{2.6}$$

Da die durchschnittliche Symbolenergie in dieser Arbeit zu $\mathrm{E}\{\sigma_x^2\} := 1$ normiert wird, kann die Rauschvarianz in Abhängigkeit von γ_s als

$$\sigma_n^2 = \frac{1}{\gamma_s} \tag{2.7}$$

angegeben werden. Die Rauschvarianz pro Quadraturkomponente ist damit $\sigma_n^2/2$. Für den Vergleich codierter Systeme ist es sinnvoll, statt der Symbolenergie die durschnittliche Infobitenergie

$$\gamma_b = \frac{\gamma_s}{B} \tag{2.8}$$

zu verwenden, wobei B die Anzahl Infobits pro Sendesymbol ist.

2.2.2.2 Fading-Kanäle

Ein entscheidender Effekt in Mobilfunkkanälen ist *Fading* (dt. Schwund). Man unterscheidet zwischen *nicht-frequenzselektivem Fading* und *frequenzselektivem Fading*. Bei nicht-frequenzselektivem Fading wird das AWGN-Kanalmodell um eine multiplikative Komponente erweitert (Abb. 2.8). Dabei ist h_m der Kanalkoeffizient, der den Verstärkungsfaktor für das m-te übertragene Symbol darstellt.

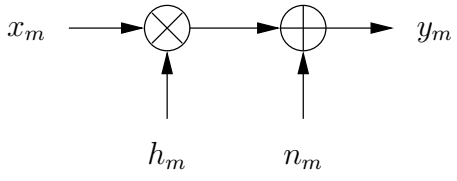

Abbildung 2.8: Nicht-frequenzselektiver Fading-Kanal.

Rayleigh-Fading In Mobilfunkkanälen kommt es häufig zu Mehrwegeausbreitung, d. h., vom Sender findet die abgestrahlte Leistung unterschiedliche Wege zum Empfänger an dem sie sich überlagern. Um den Einfluss von Mehrwegeausbreitung zu modellieren, wird

häufig *Rayleigh-Fading* angenommen [Rap96]. Der Name rührt daher, dass der Betrag $r = |h_m|$ einer Rayleigh-Verteilung [SW02]

$$p_R(r) = \frac{r}{\sigma_r^2} \exp\left(-\frac{r^2}{2\sigma_r^2}\right) \quad \text{für} \quad r \geq 0 \tag{2.9}$$

folgt, wobei σ_r^2 die Leistung des komplexwertigen Prozesses ist. Eine Methode, um Rayleigh-Fading für einen zeitdiskreten Ersatzkanal zu emulieren, ist in [Hoe92] vorgestellt. Dabei dienen die maximale Doppler-Frequenz $f_{D,\max}$ und Symboldauer T_s als deterministische Parameter für das statistische Kanalmodell. Die maximale Doppler-Frequenz ergibt sich in Abhängigkeit von der Geschwindigkeit der Mobilstation v und der Trägerfrequenz f_t zu $f_{D,\max} = v f_t / c_0$. Die Kanalkoeffizienten

$$h_m = \frac{1}{\sqrt{N}} \sum_{n=1}^{N} \exp\left(j\left(\Theta_n + 2\pi f_{D,\max} T_s \cos(\varphi_n) m\right)\right) , \tag{2.10}$$

werden als eine Überlagerung von N Mehrwegekomponenten, jede mit einer eigenen Doppler-Frequenz $f_{D,\max}\cos(\varphi_n)$ und Anfangsphase Θ_n, berechnet. Zu Beginn werden Anfangsphase und Doppler-Frequenz für jede Mehrwegekomponente n zufällig gewählt, wobei Θ_n in $[0, 2\pi)$ und φ_n in $[0, \pi)$ gleichverteilt ist. Das Produkt $f_{D,\max} T_s$ ist die normierte maximale Doppler-Frequenz, je niedriger sie ist, um so langsamer verändert sich der Kanal.

Bei frequenzselektivem Fading ist dieses insofern in dem Kanalmodell enthalten, als dass der Detektor in Abb. 2.6 auch die Funktion eines Entzerrers übernehmen kann.

Log-Distanz Pfaddämpfung Ein anderes, mit Rayleigh-Fading kombinierbares, Fading-Kanalmodell ist *Log-Distanz Pfaddämpfung* (engl. log-distance path loss) [Rap96]. Im Gegensatz zum Rayleigh-Fading, das den Effekt der Mehrwegeausbreitung modelliert, werden hier die Verluste durch durch die Ausbreitung des Signals in Abhängigkeit der Entfernung vom Sender modelliert. Der Verlust kann als Funktion der Entfernung

$$G_{\mathrm{PL}}(d) \propto \left(\frac{d}{d_0}\right)^{n_{\mathrm{PL}}} \tag{2.11}$$

dargestellt werden, wobei d_0 eine Referenzdistanz ist, in der die Sendeleistung gemessen wird. Der Exponent n_{PL} ist abhängig von der Ausbreitungsumgebung. Übliche Werte für den Frequenzbereich zellulärer Mobilfunksysteme liegen je nach Umgebung zwischen 2 (Freiraum) und 6 (Ausbreitung in Gebäuden mit Hindernissen). Bei Verwendung entfernungsabhängiger Kanalmodelle ist es offensichtlich nicht sinnvoll, die Kanalkoeffizienten h_m auf eine Leistung von eins zu normieren. In diesem Fall wird daher die durchschnittliche Symbolenergie am Sender

$$\gamma_s' = \frac{\gamma_s}{\mathrm{E}\left\{|h_m|^2\right\}} \tag{2.12}$$

definiert.

2.2.3 Empfänger

Wie der Sender ist auch der Empfänger in Abb. 2.5 als allgemeine Definition für unterschiedliche Übertragungsverfahren zu verstehen. Je nach Kanal und Modulationsalphabet $\mathbb{M}$ muss der Detektor unterschiedlich realisiert werden. Handelt es sich z. B. um einen *Intersymbolinterferenz (ISI)* erzeugenden Kanal, so muss er auch die Entzerrung realisieren, bei einem Mehrantennensystem die Trennung der unterschiedlichen Antennen. Wesentlich ist für diese Arbeit, dass ein solcher Detektor in der Lage ist, Zuverlässigkeitsinformationen (z. B. in Form von LLRs) für die von ihm gelieferten Schätzwerte zur Verfügung zu stellen, um für den nachfolgenden Decodierer soft-input Decodierung zu ermöglichen. Der Decodierer hängt stark von dem verwendeten Kanalcode ab.

Da die Decodierung einen wesentlichen Aspekt dieser Arbeit darstellt, wird auf sie unter Berücksichtigung unterschiedlicher Verfahren in Kapitel 3 genauer eingegangen.

2.3 Dienstgüte (QoS)

Dienstgüte (engl. quality of service, QoS) beschreibt die Qualität eines Dienstes. Dabei ist Qualität in seiner allgemeinen Bedeutung zu verstehen, d. h., es geht nicht nur darum, wie gut oder schlecht ein Dienst ist, sondern um seine grundsätzliche Beschaffenheit. QoS kann dabei unterschiedlichste Aspekte im Zusammenhang mit der Datenübertragung umfassen und wurde in [Intb] definiert als

> *The collective effect of service performance which determines the degree of satisfaction of a user of the service.*

Für bestimmte Dienste, Protokolle oder Schichten kann QoS allerdings spezieller und durchaus unterschiedlich definiert werden.

In digitalen Kommunikationssystemen sind Dienste unterschiedlicher QoS-Anforderungen häufig in *QoS-Klassen* eingeteilt, wobei Dienste ähnlicher Anforderungen dieselbe QoS-Klasse verwenden. So sind z. B. für UMTS die folgenden QoS-Klassen mit ihren grundlegenden Charakteristiken definiert [ETSd]:

- *conversational*, z. B. Telefonie (geringe Verzögerung, geringe Verzögerungsvariation),
- *streaming*, z. B. Videoübertragung (geringe Verzögerungsvariation),
- *interactive*, z. B. Internet browsen (Fehlerfreiheit, Rückmeldung/Reaktion wird innerhalb bestimmter Zeit erwartet),
- *background*, z. B. Download, E-Mail (Fehlerfreiheit, Daten werden nicht innerhalb kurzer Zeit erwartet).

2.3.1 QoS-Attribute der physikalischen Schicht

Die Anforderungen, die ein Dienst an die Dienstgüte stellt, werden *QoS-Attribute* genannt. Allgemein werden Systemeigenschaften, die zur Erfüllung von QoS-Anforderungen

dienen (z. B. die Datenrate), als *QoS-Parameter* bezeichnet. Dies kann z. B. die maximal zulässige Bitfehlerrate oder Verzögerung sein, aber auch bestimmte Anforderungen an Paketgröße oder Verschlüsselung. In dieser Arbeit sind vor allem die direkt von der physikalischen Schicht abhängigen QoS-Attribute bzw. Parameter von Interesse. Diese sind im Wesentlichen durch die QoS-Parameter Fehlerrate, Datenrate und Verzögerung gegeben.

Fehlerraten

Übertragungsfehler treten bei der Übertragung über den physikalischen Kanal bzw. beim Empfang und der damit verbundenen Schätzung der übertragenden Symbole auf. Höhere Schichten können Übertragungsfehler unter Umständen erkennen und Daten erneut anfordern. Bestimmt wird die Fehlerwahrscheinlichkeit bei der Übertragung über einen bestimmten Kanal aber ausschließlich durch Parameter der physikalischen Schicht wie FEC-Codierung, Modulation und Sendeleistung bzw. der Codierung im Allgemeinen[3]. Für viele FEC-Codes sind lange PDUs der physikalischen Schicht von Vorteil, um kleine Fehlerraten zu erreichen.

Verzögerung

Die Zeit, die benötigt wird, um Daten über einen physikalischen Kanal zu übertragen, ist von der Übertragungsrate der physikalischen Schicht abhängig: zum einen von der Symboldauer der übertragenden Kanalsymbole (je kürzer ein Symbol, desto höher die Übertragungsrate), zum anderen von der verwendeten Codierung (je weniger Infobits pro Symbol, desto geringer die Übertragungsrate). Auch die Länge der PDUs der physikalischen Schicht hat einen Einfluss auf die minimale Verzögerung, da diese komplett empfangen werden muss, also längere PDUs größere Verzögerungen verursachen. Die durch die physikalische Übertragung entstandene Verzögerung kann verständlicherweise durch höhere Schichten nie kleiner, sondern auf Grund von weiteren Verarbeitungsschritten nur größer werden.

Datenrate

Die maximale Datenrate ist wiederum von Codierung und Symboldauer abhängig: Kleine Symboldauern und hohe Coderaten ermöglichen hohe Übertragungsraten. Ähnlich der Verzögerung verringert sich die Nutzdatenrate durch höhere Schichten. Die Unterschiede zwischen der über die physikalische Schicht übertragene Datenrate und der dem Nutzer zur Verfügung stehenden Datenrate können dramatisch sein.

Die obigen QoS-Attribute sind nicht unabhängig voneinander. Mit niederratiger Codierung können niedrige Fehlerraten erreicht werden, die Datenrate wird allerdings verringert. Lange PDUs wirken sich ebenfalls positiv auf die Fehlerraten, aber negativ auf die Verzögerung aus.

[3] Hochstufige Modulationsverfahren, FEC-Codierung, Spreizung u. ä. Verfahren können als Codierung interpretiert werden.

Einen großen Einfluss auf diese Attribute hat auch die nächsthöhere Schicht, die Sicherungsschicht. Auf der Sicherungsschicht werden normalerweise *Automatic-Repeat-Request-Protokolle* (ARQ-Protokolle) zur Fehlerkorrektur eingesetzt. Diese Protokolle fordern gegebenenfalls eine erneute Übertragung vom Sender (häufig durch Senden der Nachrichten *Acknowledged* (ACK) bzw. *Not Acknowledged* (NAK). Dies ist von Bedeutung, da FEC-Codierung der physikalischen Schicht und die ARQ-Protokolle der Sicherungsschicht effizient mit einander kombinierbar sind. Ein erneutes Senden eines Pakets führt zu einer weiteren Verzögerung, da einerseits die Nachricht mit der Anforderung vom Empfänger zum Sender und andererseits die eigentliche Nachricht erneut vom Sender zum Empfänger geschickt wird. Neben der eigentlichen Übertragungszeit fallen dabei auch Verarbeitungszeiten an Sender und Empfänger an.

Die QoS-Parameter der physikalischen Schicht weisen also Abhängigkeiten voneinander auf und können daher nicht gleichzeitig voll erfüllt werden. Diese Tatsache kann in einem „magischen Dreieck“ dargestellt werden, auch *QoS-Dreieck* (Abb. 2.9) genannt. Für Re-

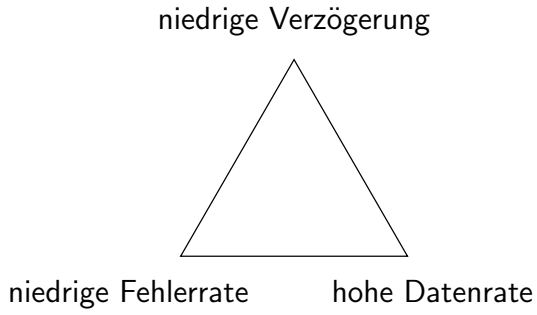

Abbildung 2.9: QoS-Dreieck: Unvereinbarkeit der gleichzeitigen Optimierung der Ziele niedrige Fehlerrate, niedrige Verzögerung und hohe Datenrate.

alisierung, Implementierung und Einstellung der physikalischen Schicht findet sich ein Punkt in der Fläche des QoS-Dreiecks. Idealerweise kann für jeden Dienst ein Arbeitspunkt in der Fläche eingestellt werden, der die QoS-Anforderungen des Dienstes erfüllt. Eine physikalische Schicht ist um so flexibler, je größer der Bereich der Fläche, die durch Wahl von Parametern bzw. Adaption erreichbar ist. Zu beachten ist, dass das QoS-Dreieck für reale Systeme nicht unbedingt gleichseitig ist und die Verbindungen der durch die Extrema definierten Eckpunkte unter Umständen nicht einmal ein Dreieck im geometrischen Sinn ergeben, da seine Seiten keine Strecken sein müssen.

Aufwändigere Algorithmen (für Decodierung im weiteren Sinne) und größere Puffer (für ARQ, Queuing) benötigen größere Rechenleistung und Speicher am Empfänger (und teilweise auch am Sender). Dieser Aufwand kann durchaus als QoS-Parameter gelten, da sich daraus auch direkt den Nutzer betreffende monetäre Kosten ergeben können. Mit Berücksichtigung des Aufwands wird das QoS-Dreieck zu einem *QoS-Tetraeder*[4] Dieser

[4]Die Idee des QoS-Tetraeders wurde am 16.09.2007 in Kiel im Gespräch mit J. Mietzner, K. Scholz und H. Özer entwickelt.

ist in Abb. 2.10 dargestellt – Ziele in der Grundfläche des Tetraeders sind jeweils einfacher zu erfüllen, wenn größerer Aufwand betrieben wird. In Betrachtungen in dieser

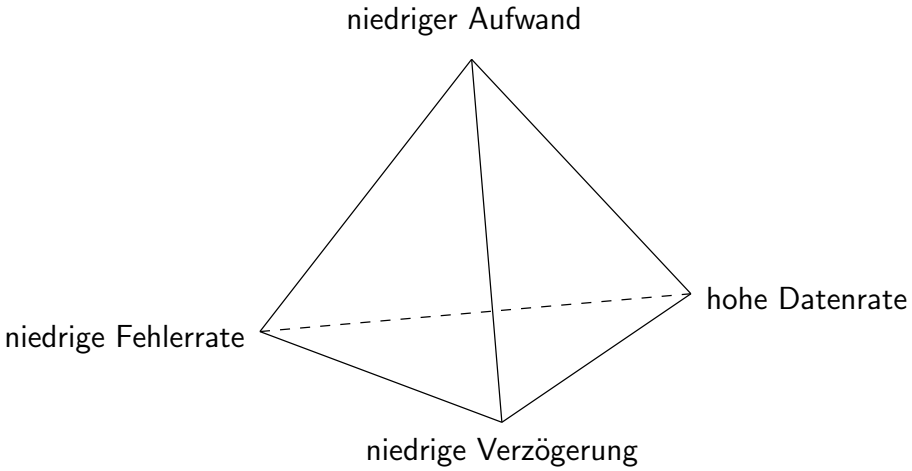

Abbildung 2.10: QoS-Tetraeder: Unvereinbarkeit der gleichzeitigen Optimierung der Ziele niedrige Fehlerrate, niedrige Verzögerung, hohe Datenrate und niedriger Aufwand.

Arbeit wird der Aufwand, der für die Übertragung betrieben wird, weitgehend außer Betracht gelassen, da er für die höheren Schichten von geringer Bedeutung ist. Ziel dieser Arbeit ist die Bestimmung der Fehlerwahrscheinlichkeit und der dadurch ermöglichte Abgleich der drei Parameter des QoS-Dreiecks für von der physikalischen Schicht ausgehende schichtübergreifende Systementwürfe.

Häufig wird auch die *Kanalkapazität* [Sha48] herangezogen, um die Übertragungseigenschaften der physikalischen Schicht zu bewerten und schichtübergreifende Anpassungen vorzunehmen. Die Kanalkapazität ist die maximale Rate, für die fehlerfreie Übertragung bei Verwendung eines kapazitätserreichenden Codes möglich ist. Dadurch ist bei bekannter Kanalkapazität, je nach Übertragungsverfahren, eine komfortable Ratenanpassung und -aufteilung möglich. Die Verwendung der Kanalkapazität hat für die Verwendung von schichtübergreifenden Ansätzen allerdings zwei wesentliche Nachteile.

1. Die Kanalkapazität muss relativ genau bekannt bzw. berechenbar sein, um die Coderate entsprechend wählen zu können – dies kann, je nach System, beliebig kompliziert sein. Im Kontext von schichtübergreifenden Ansätzen muss häufig die Systemkapazität (ergibt sich aus der Kanalkapazität unter Berücksichtigung des Übertragungsverfahrens und der verwendeten Protokolle), also nicht nur die Kapazität des physikalischen Kanals, berücksichtigt werden.

2. Bezogen auf das QoS-Dreieck in Abb. 2.9 ist die Fehlerwahrscheinlichkeit bei Erreichen der Kapazität Null (fehlerfreie Übertragung). Damit wären einige in dieser Arbeit verfolgten Ansätze, die kontrolliert Fehler zulassen, um die Datenrate zu steigern, nicht möglich.

2.4 Zusammenfassung

Neben der Einführung des in dieser Arbeit verwendeten zeitdiskreten Systemmodells wurde in diesem Kapitel eine kurze Darstellung vom schichtweisen Aufbau digitaler Übertragungssysteme und den Möglichkeiten schichtübergreifender Ansätze gegeben. Als von der physikalischen Schicht ausgehende Dienstgüteparameter mit großem Einfluss auf darüberliegende Schichten wurden die Parameter Fehlerrate, Datenrate und Verzögerung identifiziert.

Kapitel 3

Fehlerkontrolle

In diesem Kapitel wird auf *Fehlerschutz* bzw. *Fehlerkontrolle* eingegangen, für die auch häufig der Begriff *Kanalcodierung* verwendet wird. Im Wesentlichen existieren zwei Fehlerkontroll- bzw. Fehlerkorrekturverfahren, *forward error correction* (FEC) (dt. Vorwärtsfehlerkontrolle) und *automatic repeat request (ARQ)* (dt. automatische Neuanforderung) [LC04]. Dabei umfasst der Begriff der Kanalcodierung im engeren Sinne die FEC, bei der die zu übertragenden Bits senderseitig codiert werden, um empfängerseitig im Rahmen der Decodierung Übertragungsfehler korrigieren zu können. Bei den ARQ-Verfahren handelt es sich dagegen um *Protokolle* (siehe Abschnitt 2.1.1). Sie bewirken eine erneute Übertragung der (unter Umständen codierten) Daten, wenn diese nicht oder nur unzureichend empfangen werden konnten. Dafür wird in der Regel ein Rückkanal (engl. feedback) vom Empfänger zum Sender benötigt. FEC-Codierung und Decodierung werden üblicherweise der physikalischen Schicht zugeordnet, während ARQ-Protokolle zur Fehlerkontrolle der Sicherungsschicht zugeordnet werden (siehe Abschnitt 2.1.2). Die Kombination von FEC-Codierung und ARQ-Protokollen wird als *hybrid ARQ* (HARQ) bezeichnet. Es existieren eine Vielzahl unterschiedlicher FEC-Codes [LC04], [Wic95]. In dieser Arbeit finden einige der im Bereich des digitalen Mobilfunks gebräuchlichsten FEC-Codes Anwendung: binäre Faltungscodes, parallel verkettete Faltungscodes sowie binäre Low-Density-Parity-Check-Codes (LDPC-Codes). Dieses Codes haben eine hohe praktische Relevanz und finden z. B. in UMTS [ETSc] und DVB-S2 [ETSa] Anwendung. Im Folgenden werden Konzepte zur Codierung und Decodierung dieser Codes kurz eingeführt, um das Verständnis und die Nachvollziehbarkeit der Ergebnisse zu ermöglichen.

Für die Codierung wird jeweils die für die betreffenden Codes gängige Darstellung gewählt; für viele Codes ist dies die Polynomdarstellung von Code- und Infowörtern. Das Infopolynom ergibt sich aus dem Infowort $\boldsymbol{u}$ zu

$$u(D) = u_1 + u_2 D^1 + u_3 D^2 + ... + u_K D^{K-1} \,, \tag{3.1}$$

analog wird das Codepolynom mit den Symbolen aus $\boldsymbol{c}$ zu

$$c(D) = c_1 + c_2 D^1 + c_3 D^2 + ... + c_N D^{N-1} \,, \tag{3.2}$$

definiert. Die Codierung kann mit Hilfe von Generatorpolynomen

$$g(D) = g_0 + g_1 D^1 + ... + g_L D^L \,, \tag{3.3}$$

beschrieben werden. Generatorpolynome sind für gängige Blockcodes in der entsprechenden Literatur tabelliert. Eine wesentliche Eigenschaft eines Codes ist seine *Coderate* $R = K/N$.

3.1 Faltungscodes

Faltungscodes wurden wahrscheinlich erstmals in [Eli55] vorgestellt. Sie lassen sich einteilen in *systematisch* und *nichtsystematisch* sowie *rekursiv* und *nichtrekursiv*. Hier finden nichtsystematische nichtrekursive Faltungscodes (non-recursive non-systematic convolutional codes, NRNSC-Codes) und rekursive systematische Faltungscodes (RSC-Codes) Anwendung. Es ist üblich, Faltungscodes durch die Nennung ihrer Generatorpolynome zur Basis acht (Oktalsystem) zu spezifizieren. Ein $(5_8, 7_8)$ Faltungscode wird durch die Generatorpolynome $G_1 = 5_8 = 101_2$ und $G_2 = 7_8 = 111_2$ bzw. $g_1(D) = 1 + D^2$ und $g_2(D) = 1 + D^1 + D^2$ erzeugt. Da jedes der Generatorpolynome $g_p(D)$ ein Codebit $c_{p,k}$ pro Infobit u_k erzeugt, ergibt sich die Coderate aus der Anzahl der Polynome p zu $R = 1/p$. Ein weiterer wichtiger Parameter für Faltungscodes ist die *Gedächtnislänge* L. Sie ergibt sich aus dem höchstwertigsten Generatorpolynom und ist gleich der Länge des Polynoms in der binären Darstellung minus eins. Unter der *Einflusslänge* (engl. constraint length) versteht man $L + 1$. Für den $(5_8, 7_8)$ Code ergibt sich damit eine Gedächtnislänge von $L = 2$.

3.1.1 Codierung

Die Codierung von Faltungscodes wird üblicherweise in einer Schieberegisterstruktur realisiert. Exemplarisch ist diese in Abb. 3.1 für einen $(5_8, 7_8)$ Faltungscode dargestellt.

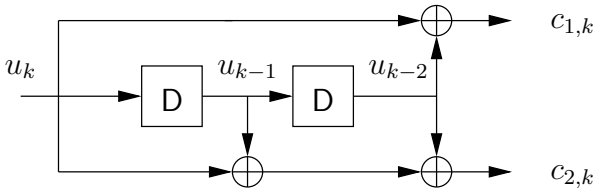

Abbildung 3.1: Schieberegisterstruktur zur Codierung mit einem $(5_8, 7_8)$ Faltungscode.

Die binären Generatorpolynome geben die Abgriffe an den Speicherelementen an, wobei eine 1 für einen Abgriff und eine 0 für keinen Abgriff steht. Aus Abb. 3.1 erklärt sich der Begriff Gedächtnislänge: Ein Faltungscodierer weist L hintereinanderliegende Speicherelemente auf. Ferner wird deutlich, dass ein Codebit c_k aus den Infobits u_{k-L} bis u_k berechnet wird. Die Infobits werden sukzessive in die Schieberegisterstruktur gegeben, so dass sich für eine Infobitfolge $u_1, u_2, ...$ eine Codebitfolge $c_{1,1}, c_{2,1}c_{1,2}, c_{2,2}, ...$ ergibt. Die Speicherelemente werden vorher mit 0 initialisiert ($u_0 = ... = u_{-(L-1)} = 0$).

Bei rekursiven Faltungscodes wird eines der Codebits zurückgeführt und mit dem Infobit kombiniert. In Abb. 3.2 ist dies für das erste Generatorpolynom dargestellt. Damit

ergibt sich ein neues Generatorpolynom $g_2(D) = (1 + D + D^2)/(1 + D^2)$. Systematische Codes enthalten die Infobits explizit im Codewort. Für einen systematischen Faltungscode ist die Schieberegisterstruktur des Codierers in Abb. 3.2 gezeigt. Eines der Codebits – hier $c_{1,k}$ – entspricht dem Infobit u_k. Das entsprechende Generatorpolynom ist also $g_1(D) = 1$. Der rekursive systematische Codierer in Abb. 3.2 erzeugt also einen $(1, 7_8/5_8)$ Faltungscode.

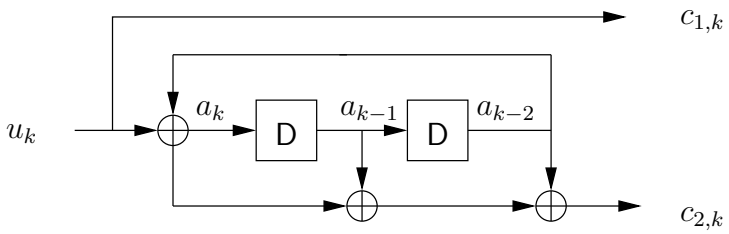

Abbildung 3.2: Schieberegisterstruktur zur Codierung mit einem $(1, 7_8/5_8)$ Faltungscode.

Für die blockweise Übertragung ist es vorteilhaft, den Faltungscode zu terminieren. Darunter versteht man, dass sich die Speicherelemente des Schieberegisters am Ende des Codierprozesses wieder in einem definierten Zustand, z. B. dem Anfangszustand befinden. Dies kann für nicht-rekursive Faltungscodes durch Verlängerung des Infoworts um L Nullen erreicht werden (Zero-Tailing). Für rekursive Faltungscodes wird der Nullzustand erreicht, indem nach Eingang der K Infosymbole L-mal der Speicherzustand a_{k-L} an den Codiereingang zurückgeführt, also als Infobit angenommen, wird. Die Terminierung führt zu einem um LP Bits verlängertes Codewort.

3.1.2 Decodierung

Auf Grund der Speicherzustände, die sich bei der Codierung von Faltungscodes ergeben, bietet sich eine trellis-basierte Decodierung an [LC04]. Das Trellis beschreibt für $k = 1 \ldots K + L$ die möglichen Übergänge der durch die vorher codierten Infobits definierten Speicherzustände $u_{k-1} ... u_{k-L}$. Ein Trellis stellt einen gerichteten Graphen dar, dessen Kanten für Info- bzw. die entsprechenden Codesymbole stehen. Damit entspricht jeder Pfad durch das Trellis einem Codewort. Je größer die Gedächtnislänge L, desto aufwändiger ist in der Regel die Decodierung, da das Trellis 2^L mögliche Zustände beinhaltet. Für den Faltungscode mit den Polynomen $(5_8, 7_8)$ ergibt sich daher die Trellisdarstellung in Abb. 3.3.

In dieser Arbeit werden zwei Gruppen von Algorithmen zur trellis-basierten Decodierung eingesetzt. Die eine basiert auf dem Viterbi-Algorithmus (VA) [Vit67], die zweite auf dem BCJR-Algorithmus [BCJR74]. Der Viterbi-Algorithmus liefert das wahrscheinlichste Infowort (Sequenzschätzung), während der BCJR die wahrscheinlichsten Infosymbole (Symbolschätzung) liefert. Für beide Algorithmen existieren verschiedene Varianten, um Zuverlässigkeitsinformation zu generieren. Beide Gruppen von Algorithmen sind also in der Lage entweder harte Schätzwerte $\hat{u}_k$ oder weiche Schätzwerte $\widehat{u}_k$ für die Infobits zu liefern. Außerdem können beide Algorithmen weiche Eingangsschätzwerte $\widehat{c}_n$ für die Codesymbole verwenden womit sie also soft-input soft-ouput Decodierer darstellen. Da

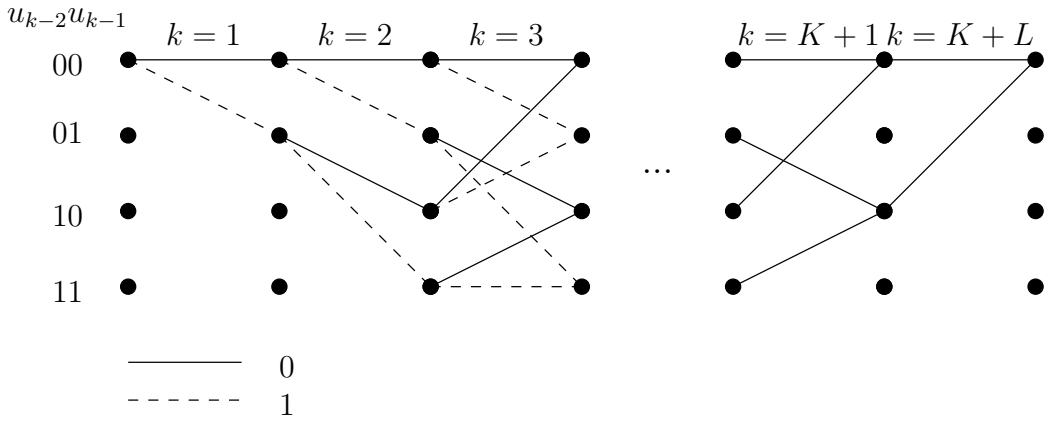

Abbildung 3.3: Trellis eines mit Zero-Tailing terminierten $(5_8, 7_8)$-Faltungscodes.

soft-output Decodierung in dieser Arbeit eine große Rolle spielt, sind die Algorithmen im Detail in Anhang B dargestellt.

3.2 Parallel verkettete Faltungscodes

Eine parallele Verkettung von Codes ermöglicht die Konstruktion eines Codes, der bessere Fehlerkorrektureigenschaften aufweist als die zur Konstruktion verwendeten *Komponentencodes*. Dabei ermöglicht diese Art der Konstruktion iterative Decodierung mit entsprechenden Komponentendecodierern. Da die Komponentencodes im Einzelnen relativ schwache Codes sein können, ermöglicht die Codeverkettung also die Konstruktion starker Codes, die mit relativ einfachen (aufwandsgünstigen) Komponentendecodierern decodiert werden können. Die Idee der parallelen Verkettung von Faltungscodes in Verbindung mit iterativer Decodierung unter Verwendung von soft-output Komponentendecodierern wurde zeitgleich in [LYHH93] und [BGT93] veröffentlicht. Auf Grund der Struktur des iterativen Decodierers werden sie häufig *Turbo-Codes* genannt [BG96]. Turbo-Codes erreichen eine Leistungsfähigkeit nahe der Kanalkapazität. Die Codierung mit Hilfe mehrerer Komponentencodes ermöglicht eine große Flexibilität im Bezug auf unterschiedliche Wortlängen. Parallel verkettete Faltungscodes haben mittlerweile Eingang in verschiedene Anwendungen gefunden. Neben den ursprünglichen Turbo-Codes, die zwei parallel verkettete Faltungscodes darstellen, sind eine Vielzahl anderer Möglichkeiten der Codeverkettung erdacht worden. So können z. B. auch mehr als zwei Codes parallel verkettet werden oder parallele und serielle Codeverkettung kombiniert werden [BGiAR07]. Da sie in dieser Arbeit vor allem als Anwendungsbeispiel für iterative Decodierung dienen, wird sich auf den in [ETSc] standardisierten Codierer beschränkt.

3.2.1 Codierung

Der in [ETSc] standardisierte, parallel verkettete Faltungscodierer (engl. parallel concatenated convolutional code, PCCC) ist in Abb. 3.4 angegeben. Wie im klassischen Codierer

in [BG96] werden auch hier zwei identische rekursive systematische Faltungscodierer als Komponentencodierer verwendet. Beide Codierer codieren dieselben Infobits $\boldsymbol{u}$ mit demselben Code, allerdings wird vor der Codierung mit dem zweiten Komponentencodierer mit Hilfe eines Interleavers Π die Reihenfolge der Infobits $\boldsymbol{u}$ zu $\Pi(\boldsymbol{u})$ verändert. Die Komponentencodierer entsprechen dabei einem $(1, 15_8/13_8)$ Codierer (Abschnitt 3.1), die systematischen Bits des zweiten Komponentencodierers ($c_{4,k}$) werden allerdings alle punktiert um eine Coderate von 1/3 zu erreichen. Weitere Punktierungen sind möglich, wobei sich eine zu starke Punktierung der systematischen Bits negativ auf die Leistung des iterativen Decodierers auswirkt und eine Punktierung der Paritätsbits zu schlechteren Distanzeigenschaften führt [LH00]. Das Codewort $\boldsymbol{c}$ ergibt sich also zu $c_{1,1}, c_{2,1}, c_{3,1}, c_{1,2}, \ldots$ Beide Komponentencodes werden durch Umschalten von Position A nach B in Abb. 3.4 terminiert. Die Terminierungsbits werden nicht punktiert, so dass das Codewort $\boldsymbol{c}$ mit den 12 Terminierungsbits

$$c_{1,K+1}, c_{2,K+1}, c_{1,K+2}, c_{2,K+2}, c_{1,K+3}, c_{2,K+3}, c_{4,K+1}, c_{3,K+1}, c_{4,K+2}, c_{3,K+2}, c_{4,K+3}, c_{3,K+3}$$

abgeschlossen wird.

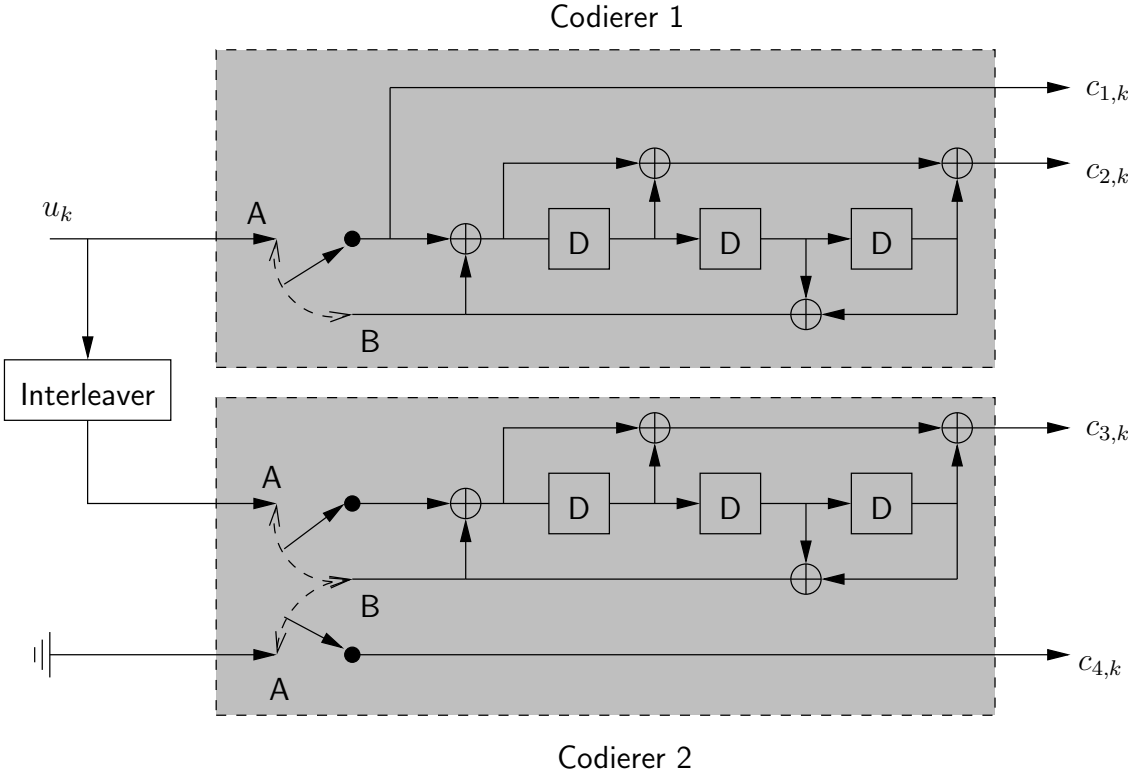

Abbildung 3.4: Codierer für parallel verkettete Faltungscodes nach [ETSc].

3.2.2 Decodierung

Analog zu den zwei Komponentencodierern für die Codierung werden in dem *iterativen Decodierer* zur Decodierung eines parallel verketteten Faltungscodes zwei Komponenten-

decodierer verwendet. Diese Decodierer sind idealerweise optimale Symbolschätzer und decodieren jeweils nur einen der Komponentencodes. Das geschieht dadurch, dass die Infobits des einen Komponentencodes jeweils auch systematische Codebits des anderen Komponentencodes darstellen. Damit können die LLRs der Infobits (siehe Kapitel 4) des einen Komponentendecodierers als Eingangsinformation des anderen verwendet werden. Der erste Decodierer erhält als weiche Eingangswerte LLRs der Paritätsbits, $\tilde{\boldsymbol{c}}_2$, wie auch der systematischen Bits $\tilde{\boldsymbol{c}}_1$ und liefert weiche Schätzwerte der Infobits $\tilde{\boldsymbol{u}}^{(1)}$. Diese dienen als Eingangswerte für die (nicht übertragenen) systematischen Bits des zweiten Komponentencodes. Zusätzlich erhält der zweite Decodierer die Empfangswerte für die Paritätsbits $\tilde{\boldsymbol{c}}_3$ des zweiten Komponentencodes. Die weichen Ausgangswerte des zweiten Decodierers $\tilde{\boldsymbol{u}}^{(2)}$ – wiederum Schätzwerte der Infobits – werden deinterleavt ($\Pi^{-1}(\tilde{\boldsymbol{u}}^{(2)})$), zurückgeführt und zusammen mit den Empfangswerten als Eingangsinformation für den ersten Komponentendecodierer verwendet. Die Nutzung von jeweils einem der Komponentendecodierer wird als eine Halbiteration, die von beiden inklusive des Informationsaustauschs als eine Iteration bezeichnet. Wichtig ist, dass extrinsische Information ausgetauscht wird, also die vom jeweils anderen Decodierer bereitgestellten Schätzwerte vor dem erneuten Informationsaustausch wieder abgezogen werden. Mit zunehmenden Iterationen verbessert sich auf diese Art der Schätzwert des Infoworts. Die Verbesserung des Schätzwertes $\tilde{\boldsymbol{u}}$, also die Anzahl der durchschnittlich korrigierten Fehler pro Iteration, nimmt dabei mit zunehmender Iterationszahl ab. Die Ausgangswerte des iterativen Decodierers $\tilde{\boldsymbol{u}}$ sind die Schätzwerte des zweiten Komponentendecodierers nach der letzten Iteration.

Auch wenn jeder der Komponentendecodierer für sich einen optimalen Symbolschätzer darstellt, ist der Gesamtdecodierer in diesem Sinne suboptimal. Dies liegt daran, dass die Eingangswerte der Komponentendecodierer als statistisch unabhängig angenommen werden, was aber nach spätestens einer Iteration nicht mehr der Fall ist. Dies wird in Abschnitt 4.2.4 visualisiert.

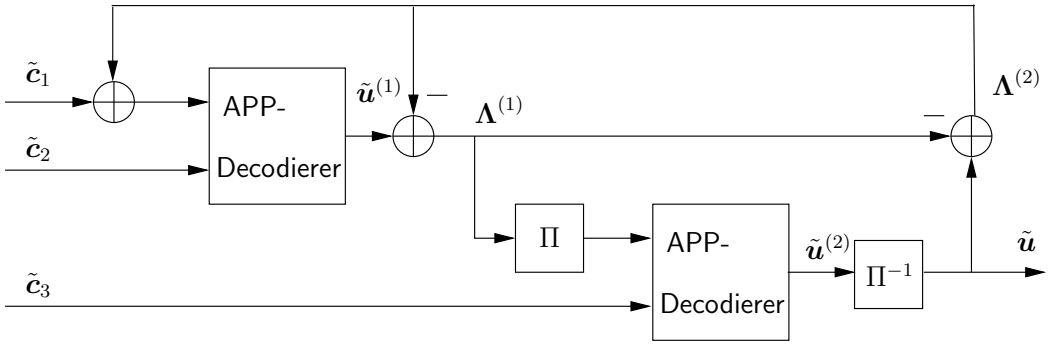

Abbildung 3.5: Iterativer Decodierer für zwei parallel verkettete Faltungscodes nach Abb. 3.4.

3.3 LDPC-Codes

Low-Density Parity-Check Codes (LDPC-Codes) [Gal62] finden zunehmend Verbreitung in der digitalen Nachrichtenübertragung [IEE05], [ETSa]. Ihr Erfolg ist einerseits darauf

zurückzuführen, dass sie ermöglichen nahe der Shannon-Grenze zu operieren [MN97] und sich andererseits gut für iterative Decodierverfahren eignen [Gal62], [Sho03].

LDPC Codes werden durch ihre *Prüfmatrix* (engl. parity check matrix) $\boldsymbol{H}$ mit den Dimensionen $P \times N$ angegeben. Der LDPC-Code besteht dann aus den Wörtern $\boldsymbol{c}$ für die

$$\boldsymbol{c}\boldsymbol{H}^{\mathsf{T}} = \boldsymbol{0} \tag{3.4}$$

erfüllt ist.[1] Die Prüfmatrix ist zeilenweise aus den auf gerader Parität basierenden Paritätsprüfungsgleichungen aufgebaut und ergibt somit mit einem gültigen Codewort multipliziert das Nullwort. Bei LDPC-Codes weist $\boldsymbol{H}$ eine geringe Dichte (engl. low density) von Einsen auf. Dabei gilt für reguläre LDPC-Codes für die Anzahl der Einsen pro Zeile w_z und die Anzahl der Einsen pro Spalte w_s: $w_z \ll N$ und $w_s \ll P$. Die Konstruktionsanweisung der Prüfmatrix für den in dieser Arbeit verwendeten $R = 1/2$ LDPC-Code aus [IEE05] ist in Anhang C angegeben.

3.3.1 Codierung

Da Decodierer für LDPC-Codes normalerweise Codewörter und damit Schätzwerte der Codesymbole liefern, werden häufig systematische Codierer verwendet. Sind die Stellen der systematischen Bits in $\boldsymbol{c}$ bekannt, so kann das Infowort $\boldsymbol{u}$ einfach aus dem Codewortschätzwert $\hat{\boldsymbol{c}}_{\mathrm{dec}}$ gewonnen werden. Für systematische Codes gilt nach (3.4)

$$[\boldsymbol{u}\ \boldsymbol{p}]\boldsymbol{H}^{\mathsf{T}} = \boldsymbol{0} \tag{3.5}$$

wobei $\boldsymbol{u}$ die K systematischen Bits und $\boldsymbol{p}$ die P Paritätsbits darstellt.

Durch Umformungen kann aus (3.5) eine Generatormatrix $\boldsymbol{G}$ für die Paritätsbits berechnet werden. Dafür wird $\boldsymbol{H}$ in eine $(N-P) \times P$ Matrix $\boldsymbol{H_1}$ und eine $P \times P$ Matrix $\boldsymbol{H_2}$ geteilt

$$[\boldsymbol{u}\ \boldsymbol{p}]\,[\boldsymbol{H_1}\boldsymbol{H_2}]^{\mathsf{T}} = \boldsymbol{0}\ , \tag{3.6}$$

so dass

$$\boldsymbol{u}\boldsymbol{H_1}^{\mathsf{T}} + \boldsymbol{p}\boldsymbol{H_2}^{\mathsf{T}} = \boldsymbol{0} \tag{3.7}$$

$$\boldsymbol{p}\boldsymbol{H_2}^{\mathsf{T}} = \boldsymbol{u}\boldsymbol{H_1}^{\mathsf{T}} \tag{3.8}$$

$$\boldsymbol{p} = \boldsymbol{u}\boldsymbol{H_1}^{\mathsf{T}}(\boldsymbol{H_2}^{\mathsf{T}})^{-1}\ , \tag{3.9}$$

und mit $\boldsymbol{G} = \boldsymbol{H_1}^{\mathsf{T}}(\boldsymbol{H_2}^{\mathsf{T}})^{-1}$ eine Generatormatrix für die Paritätsbits $\boldsymbol{p}$ in der Form $\boldsymbol{p} = \boldsymbol{u}\boldsymbol{G}$ ergibt. Damit ergibt sich das Codewort bei systematischer Codierung zu

$$\boldsymbol{c} = [\boldsymbol{u}\ \boldsymbol{u}\boldsymbol{G}]\ . \tag{3.10}$$

Eine aufwandsärmere Codierung ist in [RU01] beschrieben.

[1] Das Codewort $\boldsymbol{c}$ entspricht einem Zeilenvektor.

3.3.2 Decodierung

Für die Decodierung von LDPC-Codes wird häufig der *Belief-Propagation Algorithmus* (BPA) verwendet. Der BPA wurde in [Gal62] vorgeschlagen und kann als eine Form des *Sum-Product-Algorithmus* (SPA) angesehen werden [KFL01]. Er basiert auf der Darstellung eines linearen Codes als bipartiten Graph. Der Graph besteht aus zwei Arten von Knoten, oft *Check-Nodes* und *Variable-Nodes* genannt. Die Check-Nodes stellen dabei die Paritätsprüfungsgleichungen und die Variable-Nodes die Codebits dar. Für eine Prüfmatrix

$$\boldsymbol{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{3.11}$$

ergibt sich damit ein Graph wie in Abb. 3.6. Ein Variable-Node ist mit einem Check-Node verbunden, wenn er Teil der Paritätsprüfgleichung dieses Check-Nodes ist. Diese beiden Gruppen von Knoten tauschen iterativ extrinsische Information in Form von Wahrscheinlichkeiten oder LLRs aus um Schätzwerte der Codebits zu erhalten. Die Funktion der Check-Nodes ist damit vergleichbar mit der Funktion der Komponentendecodierer in Abschnitt 3.2.2. Während die parallel verketteten Codes über die systematischen Bits verknüpft sind, sind es hier die Codebits, die über dieselbe Prüfgleichung miteinander verbunden sind. Damit kann der BPA weiche Schätzwerte der Codebits $\tilde{c}_n$ erzeugen, was bei der Verwendung von systematischen Codes ebenfalls weiche Schätzwerte für die Infobits $\tilde{u}_k$ impliziert. Der Aufwand der Decodierung hängt dabei im Wesentlichen von der Anzahl der Knoten im Graph ab bzw. steigt (bei gleichem Grad der Knoten) linear mit der Anzahl der Codebits (Variable-Nodes). Eine detaillierte Darstellung des BPA findet sich in Anhang B.

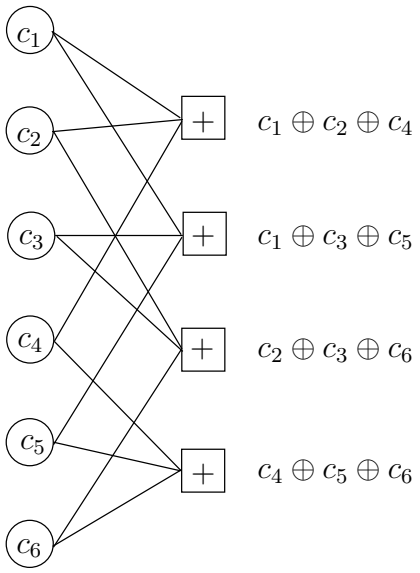

Abbildung 3.6: Graph zur Darstellung der Matrix $\boldsymbol{H}$ nach (3.11).

3.4 Fehlererkennende Codes

Fehlererkennende Codes sind in Übertragungssystemen vor allem von Interesse, um festzustellen, ob eine Übertragung fehlerfrei ist. Im Wesentlichen finden zwei fehlererkennende Codes Anwendung: einfache *Parity-Check-Codes* (PC-Codes) und *Cyclic-Redundancy-Check-Codes* (CRC-Codes). Dabei bestechen PC-Codes durch ihre Einfachheit. CRC-Codes weisen dagegen sehr gute fehlererkennende Eigenschaften auf [Wic95] und werden deswegen in fast allen Übertragungssystemen, häufig in der Sicherungsschicht, eingesetzt. In dieser Arbeit werden sie im Zusammenhang mit ARQ-Protokollen zur Erkennung von Übertragungsfehlern verwendet. Das Generatorpolynom eines CRC-Codes kann in der Form

$$g(D) = (1 + D)p(D) \tag{3.12}$$

dargestellt werden, wobei $p(D)$ ein primitives Polynom ist. Der Grad r von $p(D)$ entspricht der Anzahl der Prüfbits. Es gibt für CRC-Codes eine Vielzahl verschiedener Codierungs- und Decodierungsansätze. Wesentlich ist vor allem, dass sowohl Codierung als auch Decodierung auf sehr effiziente Art und Weise möglich sind [Wil93] [Sar88]. Evaluierungen der Fehlererkennungsraten finden sich unter anderem in [WL85] und [Wic95].

3.4.1 Codierung

In Tabelle 3.4.1 sind die in dieser Arbeit verwendeten Generatorpolynome angegeben. Weitere gebräuchliche Generatorpolynome finden sich in [Wic95] und [Wil93].

Polynom	Prüfbits	Quelle
$G_{\text{CRC8}}(D) = D^8 + D^7 + D^4 + D^3 + D + 1$	8	[ETSc]
$G_{\text{CRC12}}(D) = D^{12} + D^{11} + D^3 + D^2 + D + 1$	12	[ETSc]
$G_{\text{CRC16}}(D) = D^{16} + D^{12} + D^5 + 1$	16	[Inta], [ETSc]
$G_{\text{CRC24}}(D) = D^{24} + D^{23} + D^6 + D^5 + D + 1$	24	[ETSc]
$G_{\text{CRC32}}(D) = D^{32} + D^{26} + D^{23} + D^{22} + D^{16} + D^{12} + D^{11} + D^{10} + D^8 + D^7 + D^5 + D^4 + D^2 + D + 1$	32	[Inta]

Tabelle 3.1: Einige gebräuchliche Generatorpolynome für CRC-Codes mit unterschiedlich vielen Prüfbits.

Für die Codierung existieren unterschiedliche Strategien, wobei eine systematische Codierung, bei der die Prüfbits an das Infowort angehängt werden, weit verbreitet ist. Diese ergeben sich als Rest $R(D)$ der Division des Infopolynoms mit r angehängten Nullen:

$$R(D) = \frac{U(D)D^r}{G(D)}. \tag{3.13}$$

Damit ergibt sich das Codepolynom zu $U(D)D^r + R(D)$ bzw. das Codewort zu $[\boldsymbol{u}\,\boldsymbol{r}]$. Eine Schieberegisterdarstellung zur systematischen Codierung mit einem Generatorpolynom $g(D) = g_r D^r ... g_1 D^1 + g_0$ ist in Abb. 3.7 gegeben [LC04]. Die beiden Schalter befinden sich zunächst in Position A. Nach Eingabe der K Infosymbole enthalten die Speicherelemente

gerade den Rest der Polynomdivision und die beiden Schalter werden auf Position B umgestellt. Dadurch wird die Rekursion unterbrochen, die Speicherelemente mit Nullen aufgefüllt und der Rest der Polynomdivision ausgegeben und an das Infowort angehängt.

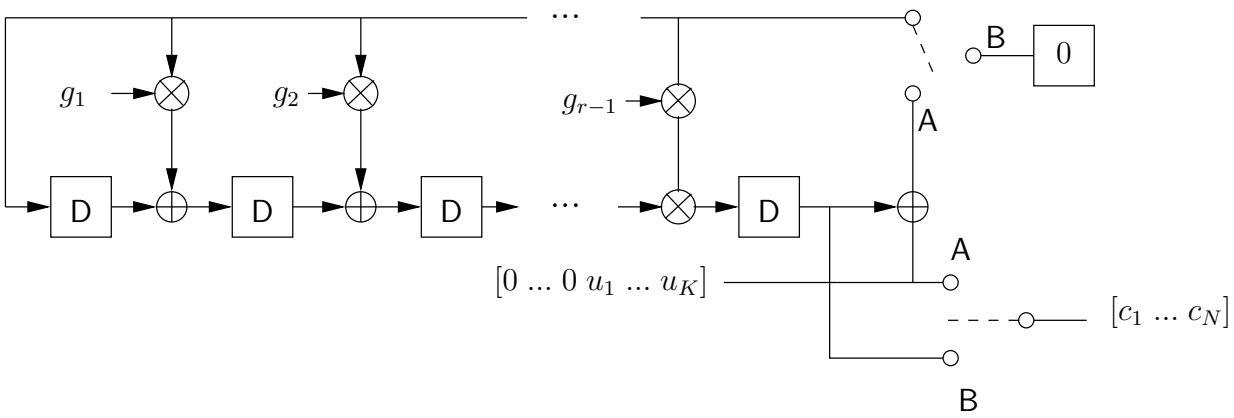

Abbildung 3.7: Schieberegisterstruktur zum systematischen Codieren mit einem CRC-Code.

3.4.2 Decodierung

Die Decodierung von systematischen CRC-Codes ist durch Abschneiden der Prüfstellen möglich.[2] Das Empfangspolynom $X(D)$ geteilt durch das Generatorpolynom ergibt einen Rest von Null, wenn $X(D)$ ein gültiges Codepolynom ist, also insbesondere dann, wenn es keine Fehler enthält:

$$\frac{X(D)}{G(D)} = \frac{U(D)D^r + R(D)}{G(D)} = 0 \ . \tag{3.14}$$

Da es sich wieder um eine Polynomdivision handelt, kann eine ähnliche Struktur wie für die Codierung verwendet werden. Diese ist in Abb. 3.8 dargestellt. Nachdem das Empfangswort $\boldsymbol{x}$ komplett in die zuvor mit Nullen initialisierte Schieberegisterstruktur eingegeben wurde, enthalten die Speicherelemente das *Syndrom*. Dieses ist Null, wenn $\boldsymbol{x}$ ein gültiges Codewort darstellt [LC04].

3.5 ARQ-Protokolle

Automatic-Repeat-Request-Protokolle (ARQ-Protokolle) basieren auf einem Rückkanal zwischen Empfänger und Sender. Konnte eine PDU nicht fehlerfrei empfangenen werden, wird dem Sender signalisiert, diese erneut zu senden. Dies geschieht solange, bis der fehlerfreie Empfang möglich war. Zur Fehlererkennung können beispielsweise CRC-Codes (siehe

[2] Dies ist im eigentlichen Sinne keine Decodierung, sondern nur die Rückgewinnung der systematischen Bits als Schätzwerte der Infobits.

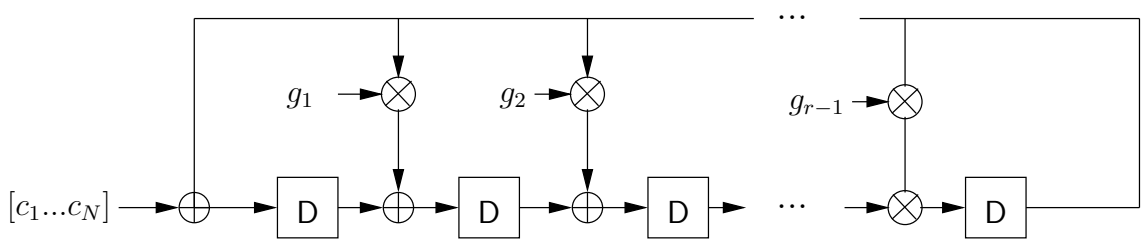

Abbildung 3.8: Schieberegisterstruktur zur Fehlererkennung bei Verwendung eines systematischen CRC-Codes.

Abschnitt 3.4) eingesetzt werden. Ein fehlerfreier Empfang wird vom Empfänger durch das Signal *Acknowledged* (ACK), ein fehlerhafter bzw. ein nicht empfangenes Paket durch *Not Acknowledged* (NAK) quittiert. Für die Verwendung von ARQ-Protokollen wird oft – wie auch hier – davon ausgegangen, dass der Rückkanal fehlerfrei ist. In der Praxis wird dies durch sehr starke Codierung erreicht ($R = 1/36$ in [ETSc]). Der wesentlichste Nachteil von ARQ-Protokollen ist die Verzögerung bis zum fehlerfreien Empfang eines Paketes, die bei der Verwendung von ARQ-Protokollen auftritt. Dabei treten zwei Arten von Verzögerung auf, die Übertragungszeit T_A und die Verarbeitungszeit T_V. Die Übertragungszeit T_A ist die Zeit, die für die Übertragung vom Sender zum Empfänger benötigt wird, während die T_V die Verarbeitung am Empfänger (z. B. Decodierung des Pakets) beinhaltet. Die dadurch erzeugte Verzögerung ist in Abb. 3.9 für ein Stop-and-Wait ARQ-Protokoll (SW-ARQ) illustriert: Ehe der Sender die Rückmeldung über den Empfang eines Pakets empfängt, also wieder ein Paket senden kann, vergeht die Zeitspanne $T_{RT} = 2T_A + T_V$.

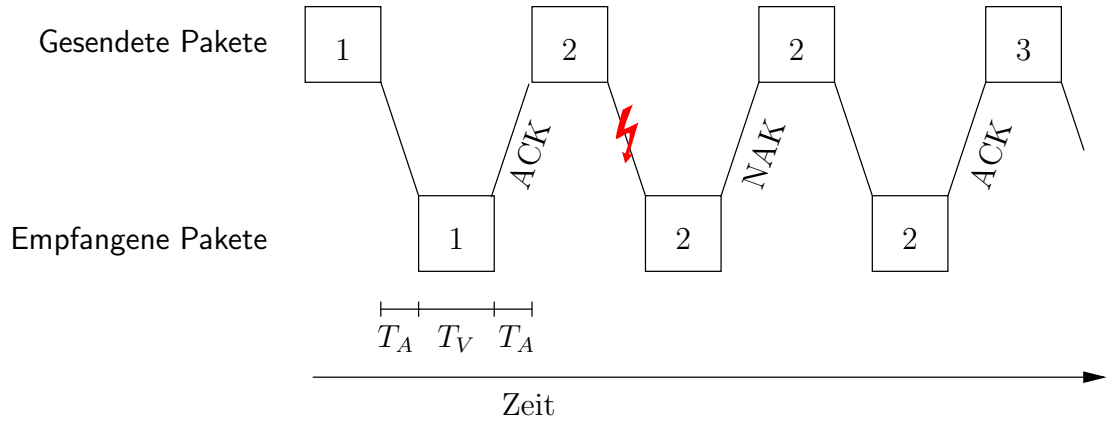

Abbildung 3.9: Funktionsweise eines Stop-and-Wait ARQ-Protkolls.

Diese sogenannte *Round-Trip-Time* T_{RT} ist verlorene Zeit, da in ihr, aus Perspektive des ARQ-Protokolls, keine weiteren Daten gesendet werden können. Bei einer Symboldauer von T_s könnten in dieser Zeit T_{RT}/T_s Symbole gesendet werden. Im Falle eines Übertragungsfehler ist ein Vielfaches von T_{RT} die Zeit in der keine neuen Daten gesendet werden können. Interessant ist also der Einfluss der Wahrscheinlichkeit für Neuanforderungen, P_r, auf die erreichbare Datenrate. Bei einer theoretisch unendlichen Anzahl von Neuan-

forderungen ergibt sich nach [Wic95] die erwartete Anzahl der Übertragungen A für den erfolgreichen Empfang eines Pakets

$$\begin{aligned} \mathrm{E}\{A\} &= (1-P_r)\sum_{a=1}^{\infty} aP_r^{a-1} \\ &= (1-P_r)\frac{\partial}{\partial P_r}\sum_{a=0}^{\infty} P_r^a \ , \end{aligned} \tag{3.15}$$

wobei sich unter Anwendung der geometrische Reihe [BSMM99, S. 18] aus (3.15) ein kompakter Wert

$$\begin{aligned} \mathrm{E}\{A\} &= (1-P_r)\frac{\partial}{\partial P_r}\left(\frac{1}{1-P_r}\right) \\ &= (1-P_r)\frac{1}{(1-P_r)^2} \\ &= \frac{1}{1-P_r} \end{aligned} \tag{3.16}$$

ergibt. Damit ist es nun möglich, die durschnittliche Datenrate bezogen auf die Kanalnutzung, die *Durchsatzeffizienz* (engl. throughput efficiency) η, bei Verwendung dieses ARQ-Protokolls zu berechnen. Im Folgenden wird auf Grund der Kürze häufig nur von Durchsatz gesprochen.[3] Mit jedem erfolgreich empfangenen Paket werden K Infosymbole übertragen. Bei einer Symboldauer T_s könnten in der Zeit T_{RT} T_{RT}/T_s Infosymbole gesendet werden. Damit ergibt sich für die fehlerfreie Übertragung von einem Paket inklusive der folgenden Wartezeit T_{RT} eine Datenrate von

$$\eta_{ARQ} = \frac{K}{K + \frac{T_{RT}}{T_s}} \tag{3.17}$$

pro potentieller Kanalbenutzung. Berücksichtigt man die Wahrscheinlichkeit für Neuanforderungen, so ergibt sich unter Berücksichtigung von (3.16) eine durchschnittliche Datenrate von

$$\begin{aligned} \eta_{ARQ} &= \frac{K}{K + \frac{T_{RT}}{T_s}}(1-P_r) \\ &= K\frac{1-P_r}{1 + \frac{T_{RT}}{KT_s}} \end{aligned} \tag{3.18}$$

pro potentieller Kanalbenutzung. Aus (3.18) ist einfach zu entnehmen, was zu geringen Druchsätzen führt: hohe Wartezeiten T_{RT} bzw. relativ dazu kurze Paketlängen K und hohe Neuanforderungswahrscheinlichkeiten P_r. Die Zeitspanne T_{RT} kann durch die Verwendung aufwändigerer ARQ-Protokolle, z. B. dem *Go-Back-N* (GBN) und dem *Selective-Repeat* (SR) ARQ-Protokoll, besser genutzt werden [BF64], [Wic95], [LC04]. Im weiteren Verlauf dieser Arbeit wird in der Regel $T_{RT} = 0$ angenommen. Damit sind die Durchsätze für die unterschiedlichen Protokolle identisch. Hier liegt der Schwerpunkt der Betrachtungen in der Steigerung des Durchsatzes durch Verringerung von P_r.

[3] Der Durchsatz wird normalerweise in Bit pro Sekunde angegeben. Für die allgemeinen Betrachtungen in dieser Arbeit ist die Betrachtung von Bit pro Symboldauer bzw. Bit pro potentieller Kanalbenutzung sinnvoller. Dieser Wert wird als Durchsatzeffizienz bezeichnet.

3.6 Hybride ARQ-Protokolle

Wie aus (3.18) hervorgeht, sinkt der Durchsatz mit zunehmender Fehlerwahrscheinlichkeit. Daher ist es vorteilhaft, die beiden zuvor vorgestellten Fehlerkorrekturmechanismen, ARQ-Protokoll und FEC-Code, zu kombinieren. Diese *hybride ARQ-Protokolle* (HARQ) genannte Kombination wurde erstmals in [WH61] erwähnt. Die Wahrscheinlichkeit für Neuanforderungen wird durch die fehlerkorrigierenden Eigenschaften des FEC-Codes verringert, idealerweise auf sehr kleine Werte reduziert. Die kleine Restfehlerwahrscheinlichkeit von FEC-Codes kann mit Hilfe von ARQ-Protokollen behoben werden. Andererseits kommt es in realen Systemen immer wieder zu Situationen, in denen der FEC-Code nicht stark genug ist, um die Übertragungsfehler zu korrigieren. Dies geschieht entweder, weil der Übertragungskanal wesentlich schlechter ist als im Systementwurf angenommen, oder, weil die Übertragungsqualität als zu gut eingeschätzt wird und ein System mit adaptiver Modulation und Codierung für die Übertragung ein zu schwaches Modulierungs- und Codierungsschema (MCS) auswählt.

Entsprechend [LC04] wird ein ARQ-Protokoll, das nur einen fehlererkennenden Code (z. B. CRC-Code) verwendet, als reines, also nicht hybrides, ARQ-Protokoll betrachtet. Die Verwendung eines fehlererkennenden Codes senkt zwar genau wie die eines fehlerkorrigierenden Codes den maximal möglichen Durchsatz, da aber die Bezeichnung hybrid auf die Verwendung der Fehler*korrektur*mechanismen zurückgeht, macht die Verwendung eines fehlererkennenden Codes kein hybrides ARQ-Protokoll aus.

Bei Verwendung von HARQ-Protokollen wird pro Kanalbenutzung ein Codesymbol anstatt eines Infosymbols übertragen, pro übertragenem Paket der Länge N werden also nur K Infosymbole übertragen. Der Durchsatz aus (3.18) ändert sich dementsprechend zu

$$\begin{aligned} \eta_{HARQ} &= \frac{K}{N + \frac{T_{RT}}{T_s}}(1 - P_r) \\ &= R\frac{1 - P_r}{1 + \frac{T_{RT}}{NT_s}} \ . \end{aligned} \tag{3.19}$$

Im Gegensatz zu den reinen ARQ-Protokollen kann ein HARQ-Protokoll also selbst bei Ausnutzung der Zeitspanne T_{RT} und $P_r = 0$ keine Effizienz von 1 erreichen, da für FEC-Codes üblicherweise Redundanz mitübertragen wird ($R < 1$).

In [Sin77] wurde erstmals ein HARQ-Verfahren vorgestellt, das mehrere empfangene Pakete kombiniert. HARQ-Protokolle mit einem solchen *Packet Combining* werden als Typ-II HARQ-Protokolle bezeichnet. Im Gegensatz dazu verwerfen Typ-I HARQ-Protokolle fehlerhaft empfangene Pakete. Die Kombinationsmechanismen in Systemen mit Typ-II HARQ-Protokolle lassen sich grob in zwei Gruppen einteilen, *Diversity-Combining* und *Code-Combining*.

3.6.1 Diversity-Combining

Der Name geht darauf zurück, dass unterschiedliche Übertragungen kombiniert werden um einen Diversitätseffekt zu erzielen. Eine solche Idee wurde erstmals in [Sin77] vorgestellt, es existieren aber eine Vielzahl unterschiedlicher Möglichkeiten. Ein einfacher Fall ist in

Abb. 3.10 schematisch dargestellt. Die Neuanforderungen enthalten dieselben Codesymbole wie die ursprüngliche Übertragung. Am Empfänger werden die Empfangswerte beider Pakete geeignet kombiniert (z. B. bei BPSK-Übertragung einfach addiert) und so die Energie beider Übertragungen für den Decodierer nutzbar gemacht. Dieses Verfahren stellt also eine Codeverkettung des inneren FEC-Codes mit einem äußeren Wiederholungscode dar.

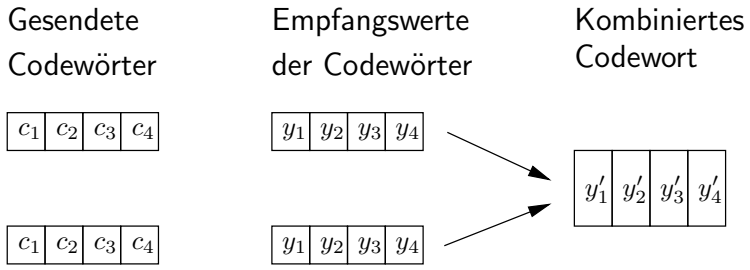

Abbildung 3.10: Diversity-Combining: Die Gesamtenergie aller Übertragungen wird für den Decodierer zugänglich gemacht.

3.6.2 Code-Combining

Der Name Code-Combining geht auf [Cha85] zurück.[4] Die Neuanforderungen stellen hier keine bloße Wiederholung dar, sondern sind so gewählt, dass sie ein Codewort eines stärkeren Codes ergeben, wenn sie kombiniert werden (siehe Abb. 3.11). Dieser Ansatz erzielt für gewöhnlich höhere Durchsätze als Diversity-Combining (ein Wiederholungscode erzielt keinen Codegewinn), stellt aber höhere Anforderungen an das ARQ-Protokoll und speziell an den Decodierer im Empfänger, da der Decodierer in der Lage sein muss, unterschiedliche Codes zu decodieren. Geeignete Codes für diesen Zweck sind z. B. ratenkompatible punktierte Faltungscodes (engl. rate-compatible punctured convolutional codes, RCPC-Codes) [Hag88].

Die beiden obigen Verfahren lassen sich nicht immer klar abgrenzen. Zum Beispiel ist denkbar, dass dasselbe Codewort erneut gesendet wird, aber für die Übertragung eine andere Modulation (z. B. QPSK statt 16-QAM) gewählt wird. Für den Decodierer würde dies – bei getrennter Detektion und Decodierung – Diversity-Combining darstellen, während aber für die Übertragung eine andere Darstellung (also Codierung) gewählt wurde. Bei gemeinsamer Detektion und Decodierung können sich durch geschickte Wahl der Sendesymbole Codiergewinne ergeben (siehe z. B. [SB05]).

In der Literatur wird teilweise zwischen Typ-II und Typ-III ARQ-Protokollen unterschieden. Der Unterschied besteht darin, dass in Typ-III ARQ-Protokollen jede Übertragung für sich, also auch ohne Kombination mit anderen Übertragungen, decodiert werden

[4] Unglücklicherweise stellt das von Chase gewählte Beispiel aus heutiger Sicht gerade kein Code-Combining, sondern Diversity-Combining dar, da die kombinierten Codeworte am Empfänger nur aus Wiederholungen bestehen, die optimal decodiert werden. Seine Idee, die unterschiedlichen Übertragungen mit dem durchschnittlichen SNR des Pakets zu gewichten, um dem Decodierer Zuverlässigkeitsinformation zur Verfügung zu stellen, wird auch *Chase-Combining* genannt.

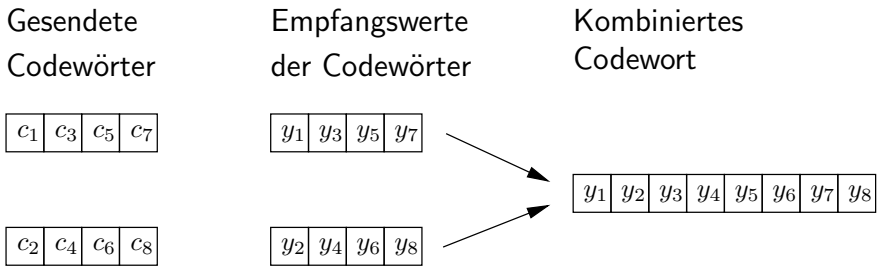

Abbildung 3.11: Code-Combining: die einzelnen Übertragungen ergeben kombiniert ein Codewort eines stärkeren Codes.

kann. Da diese Bedingung zumindest für die erste Übertragung erfüllt sein muss sind die meisten Typ-II ARQ-Protkolls auch Typ-III ARQ-Protokolle. Häufig wird im Zusammenhang mit HARQ-Protokollen auch der Begriff *inkrementelle Redundanz* (IR) verwendet. Dabei handelt es sich ebenfalls um Typ-II HARQ, wobei bei dieser Betrachtung im Vordergrund steht, dass die Redundanz für die Übertragung der Nachricht inkrementell (mit jeder Neuanforderung) erhöht wird.

3.7 Zusammenfassung

In diesem Kapitel wurden die für das Verständnis der folgenden Kapitel wesentlichen Fehlerkorrekturmechanismen eingeführt, soweit sie zum Verständnis der Arbeit erforderlich sind. Neben Faltungscodes, parallel verketteten Faltungscodes und LDPC-Codes wurden ARQ-Protokolle und hybride ARQ-Protokolle erläutert. Bei Verwendung von ARQ-Protokollen sollte die Wahrscheinlichkeit für Neuanforderungen gering sein, um eine hohe Durchsatzeffizienz zu erzielen.

Kapitel 4

Zuverlässigkeitsinformation als Qualitätsmaß

Wie in Abschnitt 2.3 erläutert, ist die Kenntnis der Fehlerwahrscheinlichkeit der physikalischen Schicht eines Übertragungssystems von großem Interesse. Dies gilt einerseits für die Evaluation von Verfahren bzw. unterschiedlicher Systementwürfe, andererseits aber auch zu Zwecken der Adaption und QoS-Kontrolle im laufenden Betrieb. In diesem Kapitel wird die Möglichkeit erläutert, weiche Schätzwerte zur Berechnung von Fehlerwahrscheinlichkeiten zu nutzen. Handelt es sich bei dem Schätzer um einen Decodierer, so kann die Fehlerwahrscheinlichkeit für die decodierten Daten auf Basis der weichen Schätzwerte ermittelt werden. Da der Decodierer die letzte Einheit der physikalischen Schicht darstellt – also das geschätzte Infowort die SDU für die Sicherungsschicht darstellt – sind seine Ausgangswerte entscheidend für die Wahrscheinlichkeit von Übertragungsfehlern. In den Ausgangswerten sind schon alle anderen die Fehlerwahrscheinlichkeit beeinflussenden Effekte enthalten. Dazu gehören z. B. unterschiedliche Kanalcodes und Modulationsformate, diverse Kanaleinflüsse, Entzerrung usw. Neben der Berechnung der Fehlerwahrscheinlichkeiten mit Hilfe von entsprechenden Decodierern wird in diesem Kapitel auch auf Effekte eingegangen, die sich negativ auf die Qualität der Schätzwerte auswirken. Je nach Kontext werden die Begriffe *Schätzer* und *Decodierer* äquivalent verwendet.

4.1 Zuverlässigkeitsinformation und Log-Likelihood Werte

Je nach verwendetem Algorithmus liefert ein Schätzer harte oder weiche Schätzwerte. Bei harten Schätzern ist der Schätzwert $\hat{a}$ üblicherweise aus einer Menge diskreter Elemente $\mathbb{A}$, wobei $\mathbb{A}$ die Menge der Codesymbole oder -wörter darstellt. Für verschiedene Anwendungen ist Information über die Zuverlässigkeit eines Schätzwertes $\hat{a}$ von Interesse. Dies ist die Wahrscheinlichkeit

$$\mathrm{P}(\hat{a} = a|b) \tag{4.1}$$

dafür, dass der geschätzte Wert $\hat{a}$, gegeben die Randbedingung b aus der Menge aller möglichen Randbedingungen $\mathbb{B}$, dem tatsächlichen Wert a entspricht. Von einem *weichen*

Schätzwert $\hat{a}$ wird gesprochen, wenn $\hat{a}$ neben der Information über den *harten Schätzwert* auch Information über die Zuverlässigkeit für die Schätzung $\hat{a}$ enthält – also z. B. (4.1).

Ein möglicher weicher Schätzwert bei bekannten a-posteriori Wahrscheinlichkeiten $\mathrm{P}(a|b)$ und gegebenen b ist z. B. der Mittelwert für $a \in \mathbb{A}$:

$$\hat{a} = \sum_{a' \in \mathbb{A}} a' \, \mathrm{P}(a'|b) \; . \tag{4.2}$$

Komplementär zu (4.1) kann die Wahrscheinlichkeit dafür berechnet werden, dass die harte Schätzung fehlerhaft ist. Für binäre Symbole bietet sich der LLR als weicher Schätzwert an. Der LLR für einen Schätzer wie in (4.1) oben ist

$$\tilde{a} = \ln \frac{\mathrm{P}(a = \hat{a}|b)}{\mathrm{P}(a \neq \hat{a}|b)} \; . \tag{4.3}$$

Für den in dieser Arbeit fast ausnahmslos betrachteten Fall binärer Symbole $a \in \{-1, +1\}$ wird (4.3) zu

$$\tilde{a} = \ln \frac{\mathrm{P}(a = +1|b)}{\mathrm{P}(a = -1|b)} = \ln \frac{\mathrm{P}(a = +1|b)}{1 - \mathrm{P}(a = +1|b)} \; . \tag{4.4}$$

Damit enthält $\tilde{a}$ mit $\mathrm{sgn}(\tilde{a})$ den harten Schätzwert[1] und $|\tilde{a}|$ stellt die Zuverlässigkeit der harten Schätzung dar bzw. enthält die *Zuverlässigkeitsinformation*. Für hochstufige Übertragungssymbole ($\mathbb{M} > 2$) können die LLRs für die auf das nicht-binäre Symbol abgebildeten Bits durch Marginalisierung der Symbolwahrscheinlichkeiten näherungsweise[2] berechnet werden.

Nach dem Modell in Abb. 2.5 gehen verschiedene Variablen in die Berechnung von $\tilde{\boldsymbol{x}}$, $\tilde{\boldsymbol{c}}$ bzw. $\tilde{\boldsymbol{u}}$ ein. Von Bedeutung für die Schätzung der Sequenzen $\boldsymbol{u}$ und $\boldsymbol{c}$ sind verwendeter Code, Decodierer, die Empfangsfolge $\boldsymbol{y}$, der verwendete Detektor bzw. die weichen Schätzwerte der Symbole $\hat{\boldsymbol{x}}$ und, je nach Kanal, die Rauschabtastwerte $\boldsymbol{n}$ (bzw. ihre Varianz σ_n^2) sowie die Kanalkoeffizienten $\boldsymbol{h}$. Vereinfachend wird hier die folgende Notation verwendet. Die LLRs der Codebits vor der Decodierung werden als

$$\tilde{c}_n = \ln \frac{\mathrm{P}(c_n = +1|\boldsymbol{y})}{\mathrm{P}(c_n = -1|\boldsymbol{y})} \tag{4.5}$$

und die LLRs der Codebits nach der Decodierung als

$$\tilde{c}_{\mathrm{dec},n} = \ln \frac{\mathrm{P}(c_{\mathrm{dec},n} = +1|\tilde{\boldsymbol{c}})}{\mathrm{P}(c_{\mathrm{dec},n} = -1|\tilde{\boldsymbol{c}})} \tag{4.6}$$

und die LLRs der Infobits als

$$\tilde{u}_k = \ln \frac{\mathrm{P}(u_k = +1|\tilde{\boldsymbol{c}})}{\mathrm{P}(u_k = -1|\tilde{\boldsymbol{c}})} \tag{4.7}$$

[1] Im Fall $\tilde{a} = 0$ ist die harte Entscheidung zufällig zu treffen.

[2] Durch die Berechnung der Randwahrscheinlichkeiten kommt es zu einem Informationsverlust.

geschätzt werden. Bei Verwendung eines soft-input soft-output Detektors sowie soft-input Decodierers gilt

$$\tilde{u}_k = \ln \frac{\mathrm{P}(u_k = +1|\tilde{\boldsymbol{c}})}{\mathrm{P}(u_k = -1|\tilde{\boldsymbol{c}})} = \ln \frac{\mathrm{P}(u_k = +1|\hat{\boldsymbol{c}})}{\mathrm{P}(u_k = -1|\hat{\boldsymbol{c}})} = \ln \frac{\mathrm{P}(u_k = +1|\boldsymbol{y})}{\mathrm{P}(u_k = -1|\boldsymbol{y})} \,, \tag{4.8}$$

da der Decodierer mit den Eingangs-LLRs $\tilde{\boldsymbol{c}}$ bzw. deren Repräsentation entsprechend einem anderen Symbolalphabet $\hat{\boldsymbol{x}}$ auch die in $\boldsymbol{y}$ enthaltene Information nutzt.

4.2 Fehler bei der Berechnung der Zuverlässigkeitsinformation

4.2.1 Suboptimale Decodierer

Die Zuverlässigkeitsinformation kann nicht von allen Decodierern korrekt berechnet werden. Ein Decodierer, der die tatsächlichen Wahrscheinlichkeiten $P(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})$ bzw. $P(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ bzw. entsprechende LLRs berechnen kann, ist der *a-posteriori probability* (APP) Decodierer. Ein *APP-Sequenzschätzer* liefert als harten Schätzwert die wahrscheinlichste Infobitsequenz

$$\hat{\boldsymbol{u}}_{\mathrm{APP}} = \arg\max_{\boldsymbol{u}'} \mathrm{P}(\boldsymbol{u}'|\tilde{\boldsymbol{c}}) \tag{4.9}$$

und entspricht damit einem *maximum a-posteriori Probability* (MAP) Sequenzschätzer. Dagegen entscheidet ein APP-Symbolschätzer für das an jeder Position k wahrscheinlichste Infobit

$$\hat{u}_{k,\mathrm{APP}} = \arg\max_{u'_k} \mathrm{P}(u'_k|\tilde{\boldsymbol{c}}) \,, \tag{4.10}$$

was dem MAP-Symbolschätzer entspricht. Für die Kriterien in (4.9) und (4.10) werden in einem APP-Schätzer die Wahrscheinlichkeiten $\mathrm{P}(\hat{\boldsymbol{u}}_{\mathrm{APP}}|\tilde{\boldsymbol{c}})$ bzw. $\mathrm{P}(\hat{u}_{k,\mathrm{APP}}|\tilde{\boldsymbol{c}})$ berechnet. Diese entsprechen weichen Schätzwerten, können aber auch für die Berechnung von anderen weichen Schätzwerten, z. B. LLRs, verwendet werden. Für gegebene a-priori Wahrscheinlichkeiten $\mathrm{P}(\boldsymbol{c})$ bzw. $\mathrm{P}(c_k)$ existiert mit der vollständigen Suche über alle Codewörter immer ein möglicher APP-Decodierer. Die vollständige Suche über alle Codewörter ist aber normalerweise zu aufwändig. Werden auf Grund fehlender a-priori Information die Wahrscheinlichkeiten $\mathrm{P}(\boldsymbol{c})$ bzw. $\mathrm{P}(c_k)$ als gleichverteilt angenommen, so entspricht der APP-Schätzer dem *Maximum-Likelihood* (ML) Schätzer.

Für trellis-decodierbare Codes existiert mit dem *reliability-output Viterbi-Algorithm* (ROVA) [RB98] ein APP-Sequenzschätzer[3] und mit dem BCJR-Algorithmus [BCJR74] ein APP-Symbolschätzer. Auf Grund der kostengünstigeren Realisierung werden in der Praxis häufig suboptimale Decodierer – also solche, die nicht (4.9) bzw. (4.10) entsprechen – verwendet. So sind iterative Decodierer im Allgemeinen nicht optimal. Dies gilt für Turbo-Decodierung von seriell oder parallel verketteten Codes [Lan05], aber auch für

[3] Der reliability-output Viterbi-Algorithmus ist nur dann ein APP-Sequenzschätzer, wenn die a-priori Wahrscheinlichkeiten der Codewörter berücksichtigt werden. Dies ist z. B. möglich, wenn die Codewörter alle gleich wahrscheinlich sind oder die Codewortwahrscheinlichkeiten aus statistisch unabhängigen symbolweisen weichen Eingangswerte berechnet werden können.

auf Faktor-Graphen basierende iterative Decodierung [KFL01], sofern der Faktor-Graph Zyklen enthält. Iterative Decodierung ist im Allgemeinen suboptimal, da die LLRs als unabhängig voneinander angenommen werden, diese Annahme aber nicht erfüllt ist (siehe auch 4.2.4).

4.2.2 Quantisierung durch Festkommadarstellung

Die in dieser Arbeit gezeigten Simulationsergebnisse verwenden für die Berechnungen in den Decodierern und die Darstellung von reellen Zahlen üblicherweise Gleitwertarithmetik [ISO89] mit sehr hoher Genauigkeit. Dagegen basieren die Berechnungen in realen Systemen häufig auf Festkommadarstellung und -arithmetik. Die binäre Festkommadarstellung $\mathrm{fp}^{\mathrm{B}}_{n,b}(x)$ (bestehend aus Bits) einer reellen Zahl x besteht aus n Bits von denen b auf die Darstellung des Nachkommateils entfallen und $a = n - b$ den ganzzahligen Teil der Zahl darstellen. Die *Auflösung* der Festkommadarstellung ist 2^{-b} und ist die kleinste Amplitude ungleich Null, die dargestellt werden kann. Der *Bereich* der darstellbaren Zahlen x ist $x_{\min} = -2^n 2^{-b} \leq x \leq (2^n - 1)2^{-b} = x_{\max}$. Zahlen größer als $x_{\max}$ werden auf $x_{\max}$ ab- und Zahlen kleiner als $x_{\min}$ auf $x_{\min}$ aufgerundet. Ein Algorithmus zur Quantisierung einer reellen Zahl auf eine als Festkommazahl darstellbare Zahl $\mathrm{fp}_{n,b}(x)$ ist in Anhang C angegeben. Die sich für $\mathrm{fp}_{4,1}(x)$ ergebene Quantisierungskennlinie ist in Abb. 4.1 dargestellt. Hier wird davon ausgegangen, dass der Decodierer so implementiert ist, dass

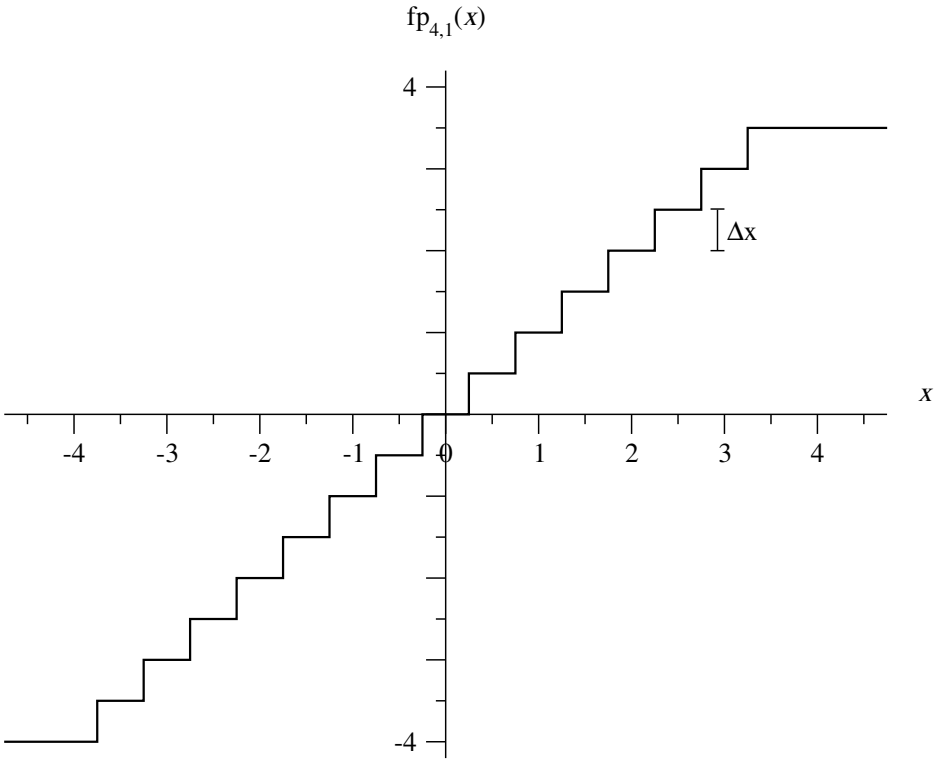

Abbildung 4.1: Quantisierungskennlinie eines 4-Bit Festkomma-Quantisierers mit 3 Bit ganzzahligem Anteil mit 1 Bit Nachkommaanteil (Auflösung $\Delta x = 0.5$).

keine Überläufe vorkommen können. Damit sind mit der Festkommadarstellung noch zwei

Probleme verbunden:

1. Festkommadarstellungen decken nur einen bestimmten Wertebereich ab, größere bzw. kleinere Zahlen werden entsprechend gerundet.

2. Zahlen werden quantisiert, um sie als Festkommazahlen darstellen zu können.

Beides kann großen Einfluss auf die vom Decodierer ausgegebene Zuverlässigkeitsinformation haben und soll darum im Weiteren betrachtet werden. In [MW01], [TOA01] und [RVH95] wurde der Einfluss der Festkommadarstellung auf Turbo-Decodierer bzw. deren (Max-)Log-APP Komponentendecodierer untersucht. Häufig wird zwischen der Quantisierung der Eingangswerte und der inneren Variablen (z. B. zwischen Zweig- und Pfadmetriken) unterschieden. Variablen die akkumulieren, z. B. die Pfadmetrik im Viterbi-Algorithmus, müssen vor allem eine große Dynamik aufweisen. Ansonsten versagen Algorithmen wie VA und BCJR-Algorithmus, da unterschiedliche Zustände durch Rundung dieselbe Metrik aufweisen, das Decodierergebnis also zufällig ist. Die Eingangswerte bzw. daraus direkt abgeleiteten Variablen haben üblicherweise kleinere Beträge (wenn logarithmierte Wahrscheinlichkeiten bzw. LLRs Anwendung finden). Entscheidend ist, dass zuverlässige Werte von weniger zuverlässigen unterschieden werden können, also auch feinere Unterschiede im Betrag erhalten bleiben. Die notwendige Anzahl Bits für die Variablen eines Decodierers hängt stark vom Anwendungsbereich (z. B. Code und Kanal ab). Wird die Anzahl Bits für die Auflösung b für unterschiedliche Variablen unterschiedlich gewählt, so ist vor Rechenoperationen dieser Variablen eine Anpassung notwendig. Selbiges gilt bei unterschiedlicher Skalierung von Werten. Dies ist nach Möglichkeit zu vermeiden, da es einen zusätzlichen Aufwand darstellt.

4.2.3 Hard-input soft-output Decodierung

Wenn die dem Decodierer vorgeschalteten Strukturen, wie z. B. der Detektor (siehe Abb. 2.5), keine weichen Schätzwerte $\tilde{\boldsymbol{c}}$ sondern harte Schätzwerte $\hat{\boldsymbol{c}}$ an den Decodierer weitergeben, so ist eine soft-input soft-output Decodierung im herkömmlichen Sinne nicht möglich. Ein im Sinne des MAP-Kriteriums optimaler hard-input hard-output Decodierer liefert anders als in (4.9) und (4.10) die Schätzwerte

$$\hat{\boldsymbol{u}} = \arg\max_{\boldsymbol{u}'} \mathrm{P}(\boldsymbol{u}'|\hat{\boldsymbol{c}}) \tag{4.11}$$

und

$$\hat{u}_k = \arg\max_{u'_k} \mathrm{P}(u'_k|\hat{\boldsymbol{c}}) \ . \tag{4.12}$$

Im Gegensatz zu den APP-Schätzern wird kein a-priori Wissen über die Zuverlässigkeit der einzelnen Bits an den Decodierer weitergegeben. Dies ist natürlich nicht wünschenswert, aber denkbar, wenn z. B. ein dem Decodierer vorgeschalteter Entzerrer oder Detektor nicht in der Lage ist, weiche Schätzwerte zu erzeugen. Für die in dieser Arbeit untersuchte Idee, die weichen Ausgangswerte des Decodierers zur Schätzung der Übertragungsqualität der physikalischen Schicht zu nutzen, ist es unverzichtbar, einen soft-output Decodierer zu verwenden. Wenn nun ein System, z. B. aus Kostengründen, auf harte Verarbeitung ausgelegt

ist, aber andererseits die Vorzügen des schichtübergreifenden Systementwurfs auf Basis von soft-output Decodierung nutzen soll, so ist der Gedanke eines hard-input soft-output Decodierers als Verbindung zwischen der physikalischen Schicht und der darüberliegenden Schichten naheliegend.

Als Kriterium für die Schätzung des Codeworts wird bei der hard-input Decodierung häufig die *Hamming-Distanz* zwischen Empfangswort $\hat{\boldsymbol{c}}$ und einem möglichem Codewort $\boldsymbol{c}'$, $d_{\mathrm{H}}(\boldsymbol{c}', \hat{\boldsymbol{c}})$, verwendet. Die Hamming-Distanz ist definiert als die Anzahl Stellen, in denen sich zwei Wörter voneinander unterscheiden. Je kleiner die Distanz, desto größer die Wahrscheinlichkeit, dass $\boldsymbol{c}'$ das übermittelte Codewort ist. Im Falle von trellis-basierter Decodierung wird die Hamming-Distanz als Metrik für den Trellisübergang $\gamma(T_k^{i,j})$ vom Zustand S_{k-1}^i in den Zustand S_k^j verwendet. Dafür wird die Hamming-Distanz zwischen den zu dem Übergang gehörigen Codebithypothesen $c_{k,1}^{i,j} \dots c_{k,P}^{i,j}$ mit den entsprechenden empfangenen Bits $\hat{c}_{(k-1)P+1} \dots \hat{c}_{(k-1)P+P}$ als Metrik genutzt:

$$\gamma(T_k^{i,j}) = d_{\mathrm{H}}([c_{k,1}^{i,j} \dots c_{k,P}^{i,j}], [\hat{c}_{(k-1)P+1} \dots \hat{c}_{(k-1)P+P}]). \tag{4.13}$$

Für die hier üblicherweise genutzte Coderate $R = 1/2$ ist $P = 2$. Die Zweigmetrik $\gamma(T_k^{i,j})$ kann also nur Werte zwischen 0 und 2 annehmen.

Die weichen Schätzwerte $\tilde{\boldsymbol{u}}$ eines hard-input soft-output Decodierers sind unbrauchbar, wenn der Decodiervorgang tatsächlich allein auf den harten Eingangsschätzwerten $\hat{\boldsymbol{c}}$ beruht. Daher sollen nun hard-input soft-output Decodierer betrachtet werden, die zusätzlich zu den harten Schätzwerten der $\hat{\boldsymbol{c}}$ *durchschnittliche* Codebitfehlerwahrscheinlichkeit nutzen. Wenn die Fehlerwahrscheinlichkeit der uncodierten Übertragung, also die Wahrscheinlichkeit $\mathrm{P}(\hat{c}_n \neq c_n)$ bekannt ist, kann statt (4.13) die *durchschnittliche* Wahrscheinlichkeit dafür, dass der Übergang $\gamma(T_k^{i,j})$ richtig ist, berechnet werden:

$$\begin{aligned} \mathrm{P}(T_k^{i,j}) = &\mathrm{P}(\hat{c}_n \neq c_n)^{d_{\mathrm{H}}([c_{k,1}^{i,j} \dots c_{k,P}^{i,j}], [\hat{c}_{(k-1)P+1} \dots \hat{c}_{(k-1)P+P}])} \\ &\cdot \left(1 - \mathrm{P}(\hat{c}_n \neq c_n)^{\left(P - d_{\mathrm{H}}([c_{k,1}^{i,j} \dots c_{k,P}^{i,j}], [\hat{c}_{(k-1)P+1} \dots \hat{c}_{(k-1)P+P}])\right)}\right) \end{aligned} \tag{4.14}$$

$\mathrm{P}(T_k^{i,j})$ ist also die Wahrscheinlichkeit dafür, dass $d_{\mathrm{H}}([c_{k,1}^{i,j} \dots c_{k,P}^{i,j}], [\hat{c}_{(k-1)P+1} \dots \hat{c}_{(k-1)P+P}])$ Codebits fehlerhaft und $P - d_{\mathrm{H}}([c_{k,1}^{i,j} \dots c_{k,P}^{i,j}], [\hat{c}_{(k-1)P+1} \dots \hat{c}_{(k-1)P+P}])$ Codebits richtig übertragen wurden. Durch Logarithmierung von (4.14) erhält man mit

$$\gamma(T_k^{i,j}) = \log(\mathrm{P}(T_k^{i,j})) \tag{4.15}$$

eine Zweigmetrik für trellisbasierte Decodierung (siehe Anhang B). Angaben über die Qualität der Decodierung finden sich in den Abschnitten 4.3.2.3 und 4.4.2.3.

4.2.4 Korrelation der weichen Schätzwerte der Informationsbits

Die LLRs eines Wortes $\tilde{\boldsymbol{u}}$ sind nicht statistisch unabhängig. Dies liegt an den für die Fehlerkorrektur notwendigen Abhängigkeiten zwischen den Codebits. Wie die LLRs korreliert sind wird im Folgenden anhand von Messungen des Autokorrelationskoeffizienten

[Pap91] der Zuverlässigkeitsinformation

$$\rho(|\tilde{u}_k|,|\tilde{u}_{k'}|)=\frac{\mathrm{Cov}\left\{|\tilde{u}_k|,|\tilde{u}_{k'}|\right\}}{\sqrt{\mathrm{Var}\left\{|\tilde{u}_k|\right\}}\sqrt{\mathrm{Var}\left\{|\tilde{u}_{k'}|\right\}}}=\frac{\mathrm{E}\left\{(|\tilde{u}_k|-\mathrm{E}\left\{|\tilde{u}_k|\right\})(|\tilde{u}_{k'}|-\mathrm{E}\left\{|\tilde{u}_{k'}|\right\})\right\}}{\sqrt{\mathrm{Var}\left\{|\tilde{u}_k|\right\}}\sqrt{\mathrm{Var}\left\{|\tilde{u}_{k'}|\right\}}} \tag{4.16}$$

dargestellt. Auf Grund der besseren Übersichtlichkeit ist $\rho(|\tilde{u}_k|,|\tilde{u}_{k'}|)$ nur für den Bereich $k=1\ldots60, k'=1\ldots60$ dargestellt. Dieser Ausschnitt (aus $K=288$) genügt, um die auftretenden Effekte zu erkennen.

In Abb. 4.2 ist $\rho(|\tilde{u}_k|,|\tilde{u}_{k'}|)$ für einen Faltungscode mit den Generatorpolynomen $(23_8,35_8)$ dargestellt. Es ist deutlich zu erkennen, dass $\rho(|\tilde{u}_k|,|\tilde{u}_{k'}|)$ groß ist, wenn $|k-k'|$ klein ist. Mit zunehmendem $|k-k'|$ nimmt die Korrelation ab. Die Korrelation entsteht durch die Faltung der Informationsbits mit den Generatorpolynomen im Codierer und nimmt mit zunehmender Gedächtnislänge L des Faltungscodes und mit sinkender Rauschvarianz σ_n^2 zu.

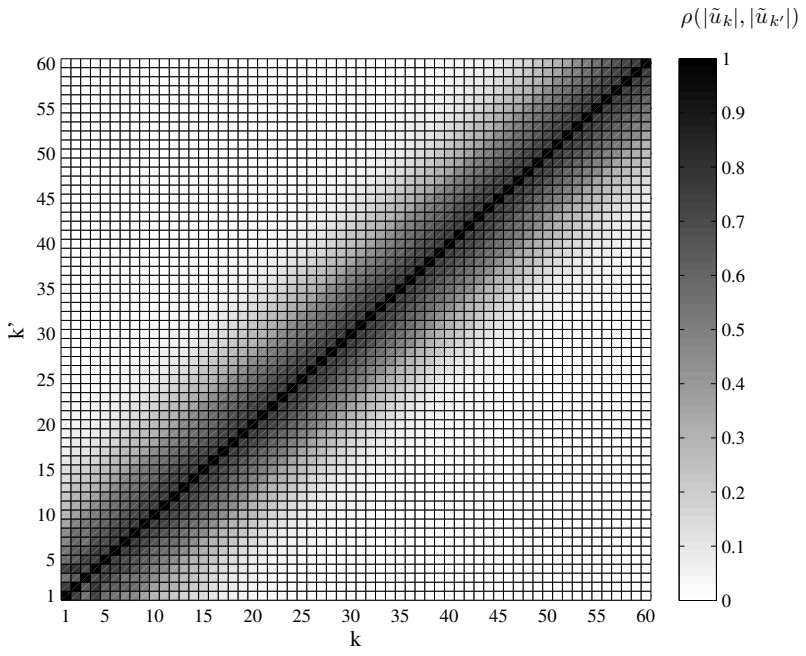

Abbildung 4.2: Korrelationskoeffizient $\rho(|\tilde{u}_k|,|\tilde{u}_{k'}|)$ der Infobit-LLRs $\tilde{u}_k$ für $\gamma_b=4\,\mathrm{dB}$. Faltungscode mit den Generatorpolynomen $(23_8,35_8)$.

In Abb. 4.3 ist der Korrelationskoeffizient für den entsprechenden rekursiven systematischen Faltungscode mit den Generatorpolynomen $(1_8, 35_8/23_8)$ dargestellt. Wieder ist deutlich die größere Korrelation für geringe Abstände $|k - k'|$ zu erkennen. Allerdings mit einem anderen Intensitätsverlauf für größer werdende $|k - k'|$; durch den Einfluss der systematischen Bits nimmt die Korrelation mit größer werdenden $|k - k'|$ schneller ab. Die abweichenden Korrelationskoeffizienten für k nahe dem Codewortanfang sind darauf zurückzuführen, dass am Anfang des Codeworts weniger Zustandsübergänge im Trellis möglich sind.

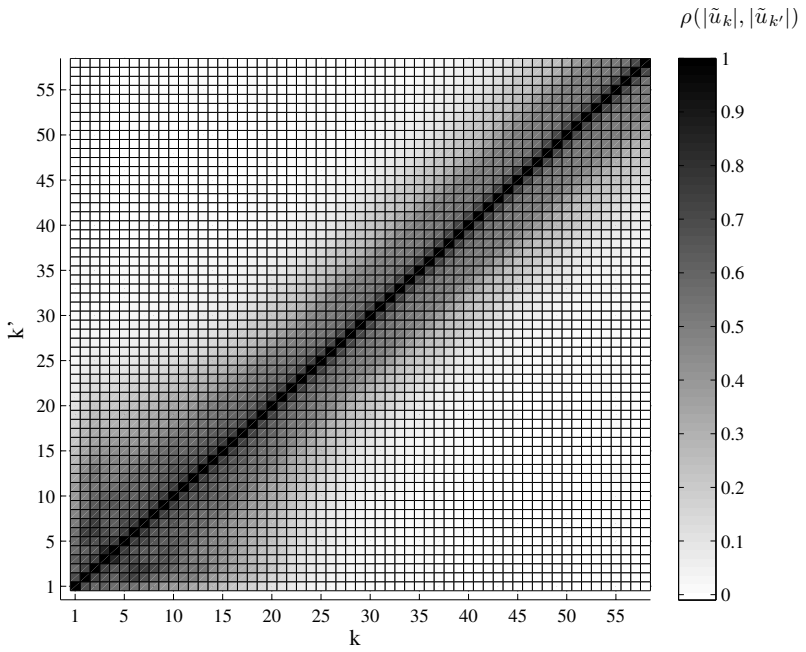

Abbildung 4.3: Korrelationskoeffizient $\rho(|\tilde{u}_k|, |\tilde{u}_{k'}|)$ der Infobit-LLRs $\tilde{u}_k$ für $\gamma_b = 4\,\mathrm{dB}$. Faltungscode mit den Generatorpolynomen $(1_8, 35_8/23_8)$.

In iterativen Decodierern wird häufig statistische Unabhängigkeit (also Unkorreliertheit) zwischen den Schätzwerten der Info- bzw. Codebits angenommen. Die Abbildungen 4.4–4.6 zeigen den Korrelationskoeffizienten $\rho(|\tilde{u}_k|, |\tilde{u}_{k'}|)$ für den IEEE 802.16e LDPC-Code mit $R = 1/2$ [IEE05]. Der Decodierer verwendet den Sum-Product-Algorithmus (Anhang B) ohne Abbruchkriterien. In Abb. 4.4 ist der Korrelationskoeffizient nach der ersten Iteration dargestellt. Abgesehen von einigen Linien ist die Zuverlässigkeitsinformation unkorreliert. Diese Korrelation ergibt sich durch einmalige Auswertung der Prüfgleichungen – nur durch diese verknüpfte Infobits sind korreliert. Mit zunehmenden Iterationen (Abb. 4.5–4.6) nimmt die Korrelation von $\tilde{u}_k$ zu, mehr und mehr Diagonalen bilden sich aus. Schließlich ist die Struktur der Prüfmatrix $\boldsymbol{H}$ zu erkennen (Abb. 4.6); sie wird für die LDPC-Codes aus [IEE05] aus zyklisch periodisch verschobenen Einheits- und Nullmatrizen aufgebaut.

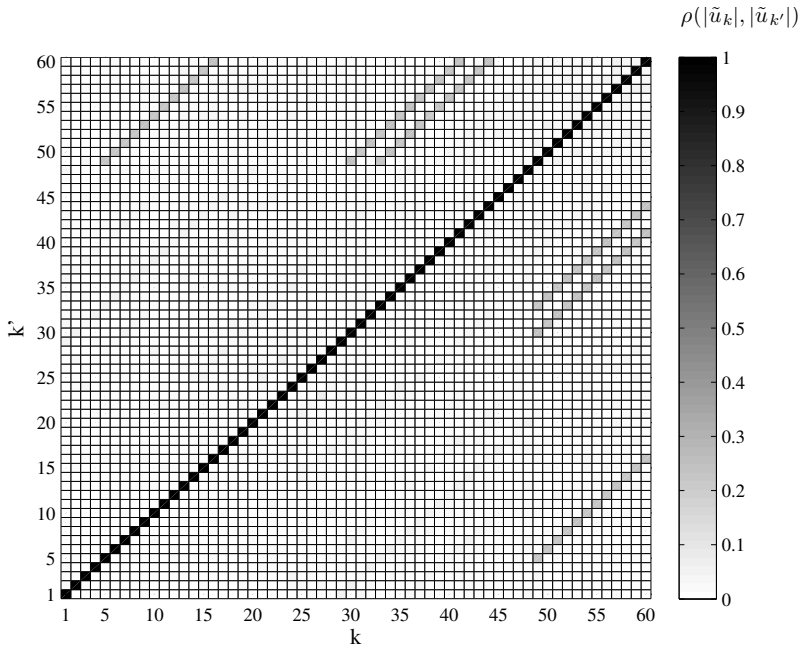

Abbildung 4.4: Korrelationskoeffizient $\rho(\tilde{u}_k, \tilde{u}_{k'})$ der Infobit-LLRs $\tilde{u}_k$ des $R = 1/2$ IEEE 802.16e LDPC-Codes für $\gamma_b = 4$ dB nach einer Iteration des SPAs.

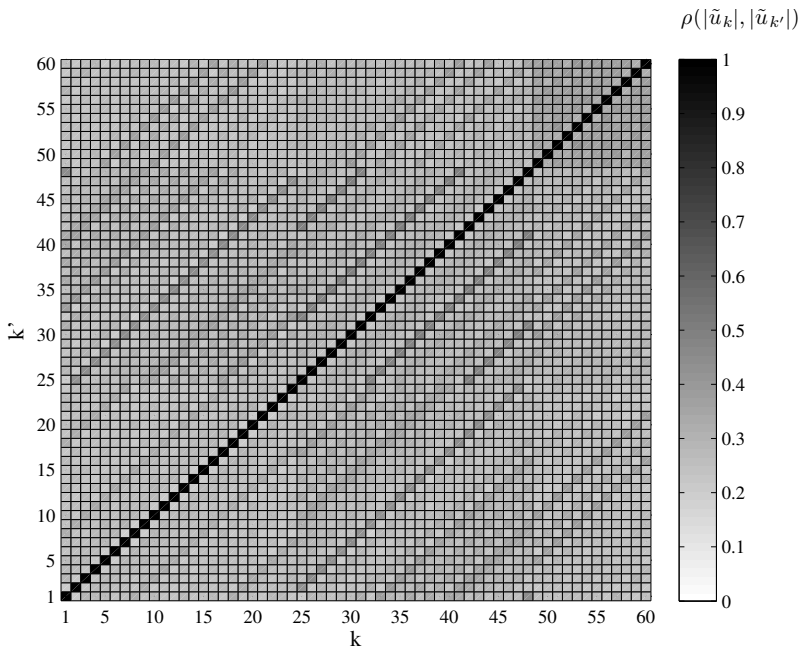

Abbildung 4.5: Korrelationskoeffizient $\rho(\tilde{u}_k, \tilde{u}_{k'})$ der Infobit-LLRs $\tilde{u}_k$ des $R = 1/2$ IEEE 802.16e LDPC-Codes für $\gamma_b = 4$ dB nach 5 Iterationen des SPAs.

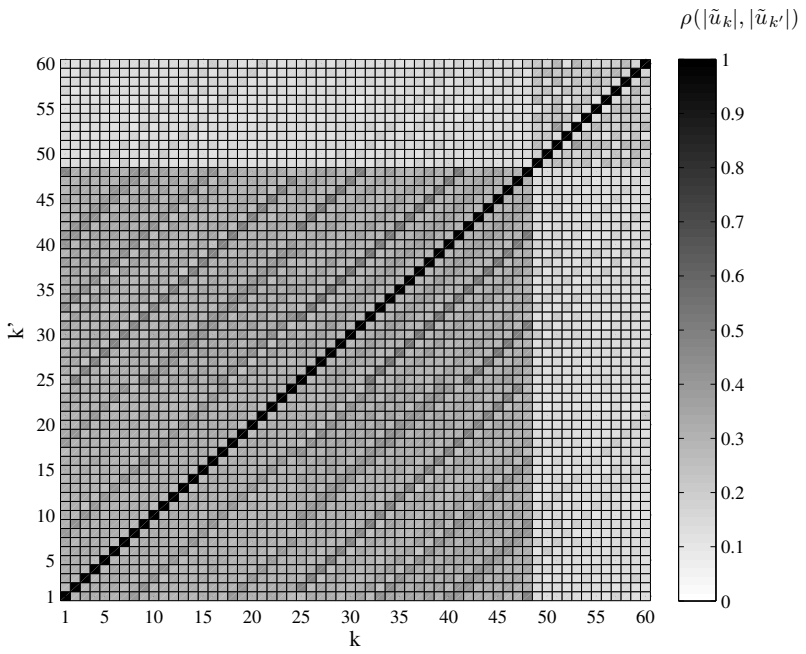

Abbildung 4.6: Korrelationskoeffizient $\rho(\tilde{u}_k, \tilde{u}_{k'})$ der Infobit-LLRs $\tilde{u}_k$ des $R = 1/2$ IEEE 802.16e LDPC-Codes für $\gamma_b = 4$ dB nach 10 Iterationen des SPAs.

Abb. 4.7 und 4.8 zeigen den Korrelationskoeffizient für einen parallel verketteten Code mit iterativem Decodierer nach Abschnitt 3.2. Während die Korrelation nach der ersten Iteration (Abb. 4.7) nur punktuell groß ist, ist sie nach drei Iterationen (Abb. 4.8) im Allgemeinen sehr groß.

Zusammenfassend läßt sich feststellen, dass die Schätzwerte der Infobits $\tilde{u}_k$ innerhalb eines Infoworts statistische Abhängigkeiten aufweisen. Diese Abhängigkeiten sind gewollt und treten für jeden Code (außer dem Wiederholungscode) auf, sind aber unterschiedlich stark bzw. verteilt. Im weiteren Verlauf dieses Kapitels soll untersucht werden, wie stark sich diese Abhängigkeiten auf die Zuverlässigkeit der Schätzwerte auswirken.

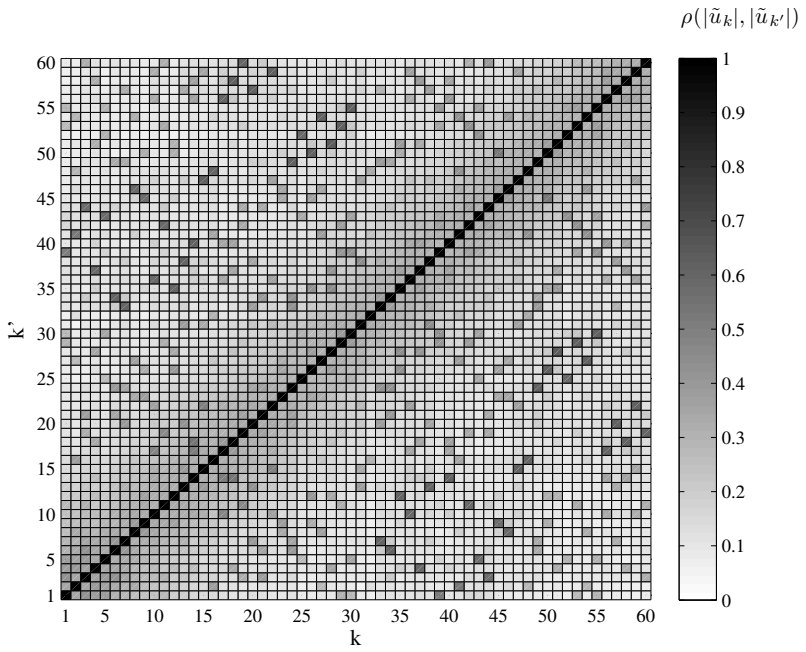

Abbildung 4.7: Korrelationskoeffizient $\rho(\tilde{u}_k, \tilde{u}_{k'})$ der Infobit-LLRs $\tilde{u}_k$ des $R = 1/3$ UMTS Turbo-Codes für $\gamma_b = 1,5$ dB nach einer Iteration des iterativen Decodierers.

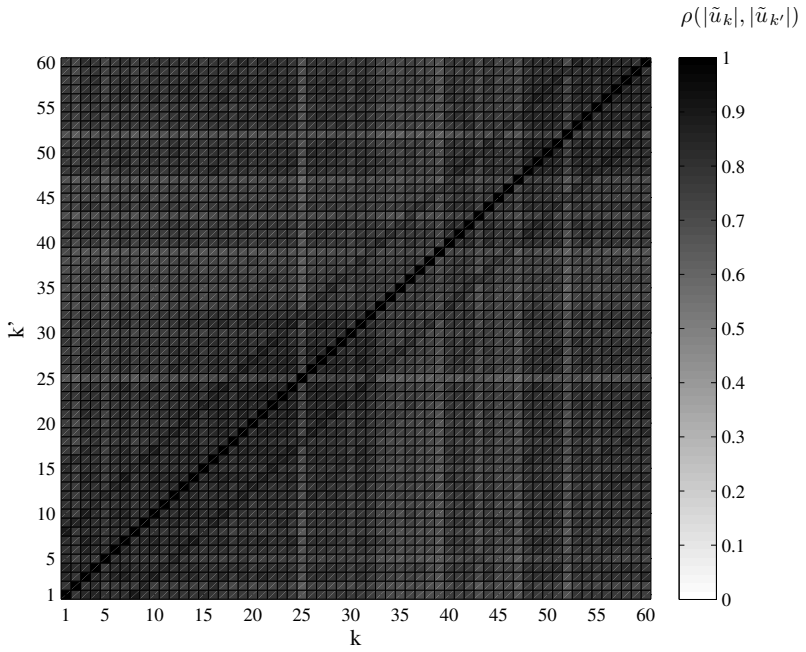

Abbildung 4.8: Korrelationskoeffizient $\rho(\tilde{u}_k, \tilde{u}_{k'})$ der Infobit-LLRs $\tilde{u}_k$ des $R = 1/3$ UMTS Turbo-Codes für $\gamma_b = 1,5$ dB nach drei Iteration des iterativen Decodierers.

4.3 Bitfehlerwahrscheinlichkeit

Eine üblicherweise zur Beurteilung der Übertragungsqualität von digitalen Übertragungssystemen herangezogene Größe ist die durchschnittliche *Bitfehlerwahrscheinlichkeit* (engl. bit error probability, BEP). Sie ist definiert als

$$P_B = \mathrm{P}(\hat{u}_k \neq u_k) \tag{4.17}$$

und ist die *durchschnittliche* Wahrscheinlichkeit dafür, dass ein Informationsbit von einem Übertragungsfehler betroffen ist [Fri96]. In [HSL00] wurde der Zusammenhang zwischen den LLRs $\tilde{u}_k$ und der Fehlerwahrscheinlichkeit des decodierten Infobits

$$P(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}}) = \frac{1}{1+e^{|\tilde{u}_k|}} \tag{4.18}$$

hergeleitet (siehe Anhang C). Mit (4.18) ist die Fehlerwahrscheinlichkeit eines *einzelnen* Infobits berechenbar. Sind die LLRs für das gesamte Infowort $\tilde{\boldsymbol{u}}$ gegeben, so kann durch Mittelwertbildung die durchschnittliche Bitfehlerwahrscheinlichkeit in dem geschätzten Infowort $\hat{\boldsymbol{u}}$,

$$P_b = \frac{1}{K}\sum_{k=1}^{K} P(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}}) = \frac{1}{K}\sum_{k=1}^{K} \frac{1}{1+e^{|\tilde{u}_k|}} \ , \tag{4.19}$$

berechnet werden. Wenn der Decodierer die tatsächlichen Wahrscheinlichkeiten $\mathrm{P}(\hat{u}_k = u_k|\tilde{\boldsymbol{c}})$ bzw. entsprechende LLRs $\tilde{\boldsymbol{u}}$ liefert, ist (4.18) exakt. Ein solcher Decodierer ist ein APP-Symbolschätzer (siehe Abschnitt 4.2.1), der die a-posteriori Wahrscheinlichkeiten für die Infobits $\mathrm{P}(u_k|\tilde{\boldsymbol{c}})$ berechnet.

4.3.1 Schätzung der Bitfehlerwahrscheinlichkeit

Um die Leistungsfähigkeit von Übertragungssystemen zu evaluieren werden üblicherweise Monte Carlo Simulationen durchgeführt. Hierfür werden in einem System wie in Abb. 2.5 A Infoworte übertragen und für das k-te Bit des a-ten übertragenen Infoworts $\boldsymbol{u}_a$ das Bitfehlerereignis gemäß

$$\begin{aligned} u_{a,k} = u_{a,k} &: B_{h,a,k} = 0 \\ u_{a,k} \neq u_{a,k} &: B_{h,a,k} = 1 \end{aligned} \tag{4.20}$$

definiert. Damit läßt sich die *Bitfehlerrate* $\hat{P}_B$ als durchschnittliche Anzahl der Fehlerereignisse bzw. als Mittelwert von $B_{h,a,k}$ berechnen:

$$\hat{P}_B = \frac{1}{AK}\sum_{a=1}^{A}\sum_{k=1}^{K} B_{h,a,k} \ . \tag{4.21}$$

In [Loe94], [HSL00] und [LH03] wurde die Idee der weichen Monte-Carlo-Bitfehlersimulation vorgestellt. Statt des harten Fehlerereignisses in (4.20) wird ein weiches Fehlerereignis, nämlich die Fehlerwahrscheinlichkeit des k-ten Bits des a-ten Infoworts

$$B_{w,a,k} = \mathrm{P}(\hat{u}_{a,k} \neq u_{a,k}|\tilde{\boldsymbol{c}}_a) \tag{4.22}$$

betrachtet und darüber der Mittelwert

$$\widehat{P}_B = \frac{1}{AK} \sum_{a=1}^{A} \sum_{k=1}^{K} B_{w,a,k} = \frac{1}{AK} \sum_{a=1}^{A} \sum_{k=1}^{K} \mathrm{P}(\hat{u}_{a,k} \neq u_{a,k} | \tilde{\boldsymbol{c}}_a) \tag{4.23}$$

gebildet.

Betrachtet man (4.21) und (4.23) als Schätzer, so stellen sie bezüglich der BEP Mittelwertschätzer dar und sind damit erwartungstreu (engl. unbiased) [SW02]. Für $A \longrightarrow \infty$ gilt nach dem Gesetz der großen Zahlen[4] $\hat{P}_B = P_B$ und $\widehat{P}_B = P_B$. Somit gilt also auch

$$\hat{P}_B = \widehat{P}_B \text{ für } A \longrightarrow \infty \, . \tag{4.24}$$

Das weiche Schätzverfahren nach (4.23) hat zwei wesentliche Vorteile. Zum einen wurde in [LH03] gezeigt, dass die Varianz des weichen Schätzers $\sigma^2_{B,w}$ immer kleiner als die Varianz des harten Schätzers $\sigma^2_{B,h}$ ist. Der zweite Vorteil ist die Flexibilität der weichen Schätzung. Im Gegensatz zu dem harten Verfahren ist die Kenntnis des übertragenen Infoworts $\boldsymbol{u}$ für die Schätzung nicht notwendig. Dies ermöglicht die Anwendung dieses Verfahrens zur Schätzung der BEP in realen Systemen, sofern diese über einen geeigneten soft-output Decodierer verfügen. Abbildung 4.9 zeigt harte und weiche Schätzung für die Übertragung über einen AWGN-Kanal bei Verwendung eines BCJR-Decodierers und verschiedenen Faltungscodes. Harte und weiche Schätzung stimmen im Rahmen der Simulationsgenauigkeit überein.

Für begrenzte A kann (4.23) damit auch benutzt werden, um durch Kurzzeitmittelwerte den BEP-Arbeitspunkt von Übertragungssystemen zu ermitteln. So wurde in [Hoe00] adaptive Modulation und Codierung auf Basis der so berechneten BEP vorgestellt. In [SH04] und [Sch08a] wird die weiche BEP-Schätzung zur Adaption der Datenrate und Sendeleistung in einem IDMA-System verwendet.

4.3.2 Fehler bei der Berechnung der Bitfehlerwahrscheinlichkeit

Die Schätzung der BEP nach (4.23) ist nur dann exakt, wenn die LLRs $\tilde{u}_k$ den tatsächlichen a-posteriori Wahrscheinlichkeiten für $\hat{u}_k$ entsprechen. Es gibt verschiedene Gründe, warum dies häufig nicht zutrifft. Der erste Grund ist die Verwendung eines suboptimalen Symbolschätzers als Decodierer (oder eines Nicht-Symbolschätzers, der damit kein optimaler Symbolschätzer ist). Eine weitere Fehlerquelle ist eine in praktischen Systemen vorgenommene Quantisierung der LLRs. Die quantisierten LLRs weisen einen Fehler auf, der sich damit auch auf die Berechnung der BEP auswirkt. Eine große mögliche Fehlerquelle ist die Verwendung eines hard-input Decodierers, der eine schlechte aber günstigere Alternative zu einem soft-input Decodierer darstellt bzw. dann verwendet wird, wenn die dem Decodierer vorgeschalteten Systemkomponenten (z. B. Detektor bzw.

[4] In den üblichen Formulierungen des Gesetzes der großen Zahlen wird statistische Unabhängigkeit zwischen den Zufallsvariablen vorausgesetzt – dies ist für Bitfehler nicht gegeben aber nach anderen Formulierungen auch nicht notwendig. Es genügt, dass die Varianz der Zufallsvariablen o_i, o_j kleiner als eine Konstante ist und für den Korrelationskoeffizient gilt $|\,\mathrm{R}(o_i, o_j)) \leq \phi(|i-j|)$ mit $\phi(n) \to 0$ für $n \to \infty$ [Pro02].

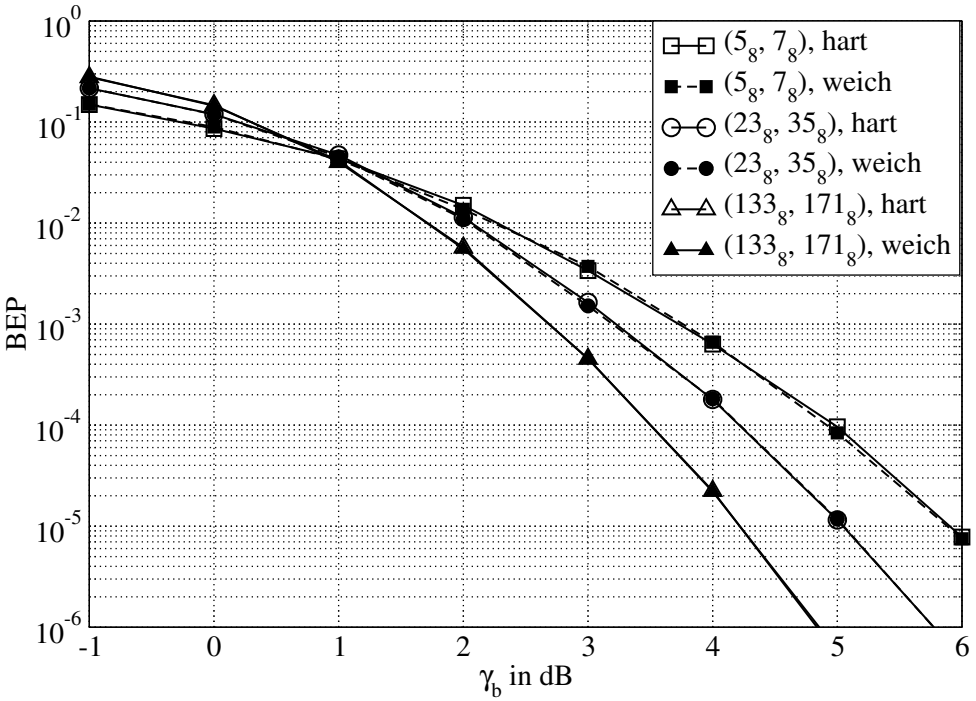

Abbildung 4.9: Harte und weiche BEP-Schätzung für verschiedene Faltungscodes bei Verwendung eines BCJR-Decodierers.

Entzerrer) harte Schätzwerte liefern. In den folgenden Abschnitten werden die Auswirkungen dieser Fehlerquellen untersucht. Eine weitere Fehlerquelle ist ein Detektor, der dem Decodierer falsche bzw. unzuverlässige weiche Eingangswerte liefert (z. B. Matched-Filterung mit falscher SNR-Schätzung, Entzerrung mit falschen Kanalkoeffizienten oder schlechter Detektor [FSMH05]). Die Auswirkungen dieser Fehlerquelle (auch auf die harte Schätzung) sind allerdings nahezu beliebig und werden daher in dieser Arbeit nicht näher betrachtet.

4.3.2.1 Suboptimale Decodierer

Die weichen Schätzwerte von suboptimalen Decodierern, also solchen die keine APP-Decodierer darstellen, können genutzt werden, um nach (4.23) einen Schätzwert der BEP zu erhalten. In Abb. 4.10 sind die harten BEP-Schätzungen $\hat{P}_B$ und weichen BEP-Schätzungen $\widehat{P}_B$ für den $(23_8, 35_8)$ Faltungscode bei Verwendung zweier populärer soft-output Decodierer, dem soft-output Viterbi-Algorithmus (SOVA) [HH89] und der Max-Log-BCJR Algorithmus [KB90], dargestellt. Diese beiden Algorithmen sind im Detail in Anhang B erläutert. Es sei erwähnt, dass es sich bei dem hier verwendeten SOVA *nicht* um den APP-SOVA handelt [JZ99]. Aus den Ergebnissen in Abb. 4.10 wird deutlich, dass die weiche BEP-Schätzung (4.23) für diese Decodierer nicht exakt ist. Die Schätzung ist bei beiden zu optimistisch, d. h., der weiche Schätzwert (4.23) ist kleiner als der harte Schätzwert (4.21). Dies ist darauf zurückzuführen, dass die beiden Algorithmen schon in der Symbolschätzung zu optimistisch sind, also die LLRs $\tilde{u}_k$ einen zu großen Betrag

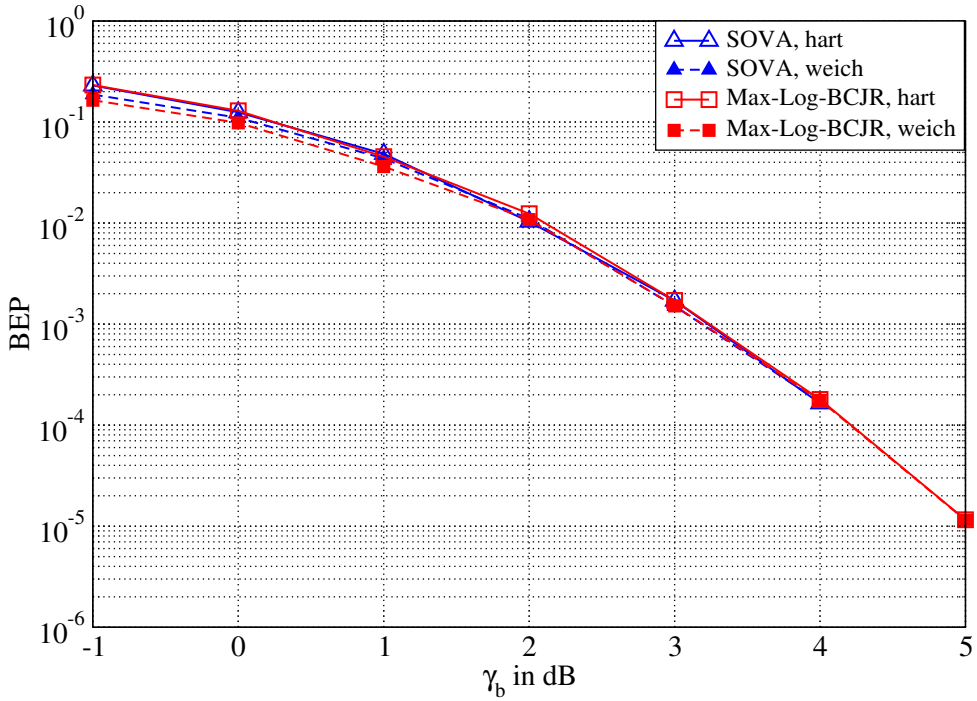

Abbildung 4.10: Weiche und harte Bitfehlerschätzung für suboptimale Decodierer eines $(23_8, 35_8)$ Faltungscode.

aufweisen bzw. die vom Decodierer berechnete Fehlerwahrscheinlichkeit für den Schätzwert $\hat{u}_k$ zu gering ist. Andererseits ist der Fehler der weichen Schätzung relativ gering, insbesondere für den relevanteren Bereich niedriger Bitfehlerwahrscheinlichkeiten (für digitale Übertragungssysteme werden Bitfehlerwahrscheinlichkeiten $< 10^{-2}$ angestrebt).

Eine andere Klasse von Decodierern stellen die graphbasierten Decodierer dar. Im Rahmen dieser Arbeit wird der SPA genutzt, um den $R = 1/2$ IEEE 802.16e LDPC-Code zu decodieren. Deutlich ist zu erkennen, dass sich zwar mit zunehmender Zahl von Iterationen die BEP verringert, jedoch die harte und die weiche Schätzung zunehmend voneinander abweichen. Während die beiden Schätzungen nach der ersten Iteration identisch sind, weist die weiche Schätzung ab der zehnten Iteration eine Abweichung von ca. $-0{,}1$ dB auf. Die, wiederum zu optimistische weiche Schätzung, ist auf Annahmen des SPAs zurückzuführen. Die Schätzwerte der einzelnen Codebits werden vom SPA als statistisch unabhängig angenommen. Wie jedoch Abb. 4.4 bis Abb. 4.6 zeigen, ist diese Annahme mit zunehmender Anzahl von Iterationen nicht mehr erfüllt.

Es ist üblich, für den SPA das Ergebnis der Prüfgleichungen als Abbruchkriterium zu verwenden. Abbildung 4.12 zeigt Ergebnisse der harten und weichen BEP-Schätzung für den SPA, wenn ab der siebten Iteration das geschätzte Codewort anhand der Prüfgleichungen auf Gültigkeit überprüft wird. Sind die Prüfgleichungen erfüllt, werden keine weiteren Iterationen durchgeführt. Diese Mindestanzahl von sieben Iterationen verhindert, dass ein durch die (anfangs vielen) Bitfehler entstandenes, gültiges aber falsches Codewort zu einem Abbruch der Decodierung und weiteren Korrektur von Bitfehlern führt. Durch Einführung dieser Abbruchbedingung wird dieselbe BEP erreicht, harte und

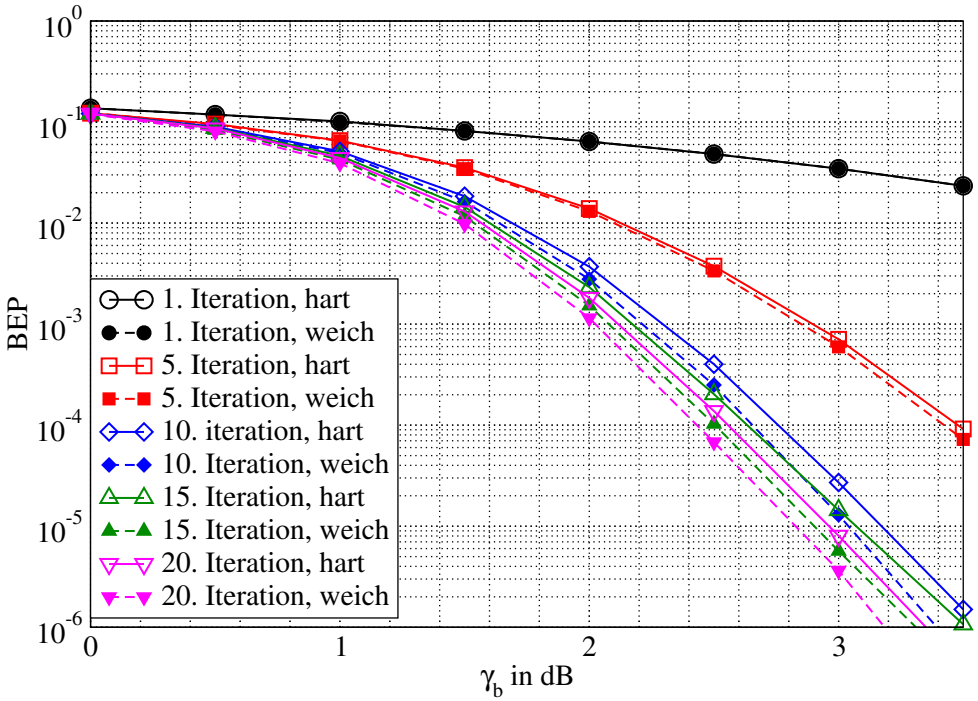

Abbildung 4.11: Weiche und harte Bitfehlerschätzung für den IEEE 802.16e LDPC-Code für verschieden viele Iterationen des SPAs.

weiche Bitfehlerschätzung stimmen aber bedeutend besser überein.

Ein weiterer suboptimaler iterativer Decodierer ist der Turbo-Decodierer zur Decodierung parallel verketteter Codes. Der Vergleich der harten und weichen Schätzung über mehrere Iterationen zeigt auch hier (siehe Abb. 4.13) ein ähnliches Verhalten: Mit zunehmender Anzahl der Iterationen nehmen die statistischen Abhängigkeiten zwischen den Infobits zu, woraus eine zu optimistische Schätzung resultiert.

4.3.2.2 Quantisierung

Aus Abb. 4.9 geht hervor, dass ein APP-Decodierer eine sehr exakte weiche Schätzung der BEP anhand der weichen Ausgangswerte ermöglicht. Da (in Hardware implementierte) Decodierer häufig mit Festkommazahlen arbeiten, stellt sich die Frage, wie groß der Einfluss der daraus resultierenden Quantisierung auf die weiche BEP-Schätzung nach (4.23) ist. Hierzu wird die weiche Schätzung für den Log-BCJR-Algorithmus in Gleitkommaimplementierung mit der eines Log-BCJR-Algorithmus mit Festkommadarstellung nach Abschnitt 4.2.2 verglichen. Dabei werden für Zustandsmetriken (α und β) die Festkommadarstellung $\mathrm{fp}_{16,2}$ gewählt und für Zweigmetriken, Ein- und Ausgangswerte und ähnliche Variablen die Darstellung $\mathrm{fp}_{8,2}$. Die höhere Dynamik von 14 Bit ist notwendig für die großen Beträge, die in der Vor- und der Rückwärtsrekursion entstehen können.[5]

[5]Es existieren eine Vielzahl von Möglichkeiten, die Festkommaimplementierung eines Algorithmus zu verbessern und die Anzahl der zur Darstellung notwendigen Bits zu verringern. Dies würde den Rahmen dieser Arbeit sprengen. Daher werden hier nur ähnliche Betrachtungen wie in [MW01], [TOA01] und [RVH95] durchgeführt.

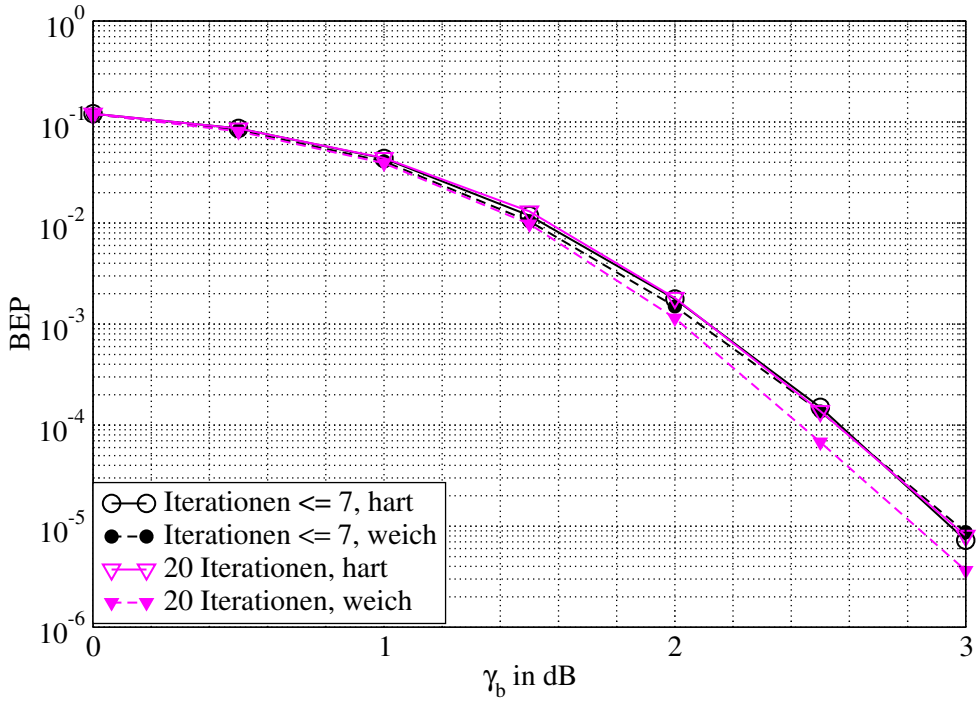

Abbildung 4.12: Weiche und harte Bitfehlerschätzung für mit SPA decodiertem IEEE 802.16e $R = 1/2$ LDPC-Code. Vergleich zwischen SPA mit 20 Iterationen und Abbruch nach sieben oder mehr Iterationen bei erfüllten Prüfgleichungen.

Die Auflösung kann für die Zustandsmetriken geringer gewählt werden. Dadurch würde aber die Notwendigkeit einer Anpassung vor gemeinsamen Rechenoperationen von Zustandsmetriken mit anderen Variablen entstehen. Registerbreiten von 8 und 16 Bit sind in der Praxis übliche Werte. Aus Abb. 4.14 sind zwei Dinge ersichtlich: Erstens wird die Leistungsfähigkeit durch die Quantisierung nicht eingeschränkt – in der quantisierten Variante erreicht der Log-BCJR-Algorithmus die gleiche Bitfehlerwahrscheinlichkeit. Zweitens stimmt auch bei quantisierten Werten die weiche BEP-Schätzung mit der harten überein.

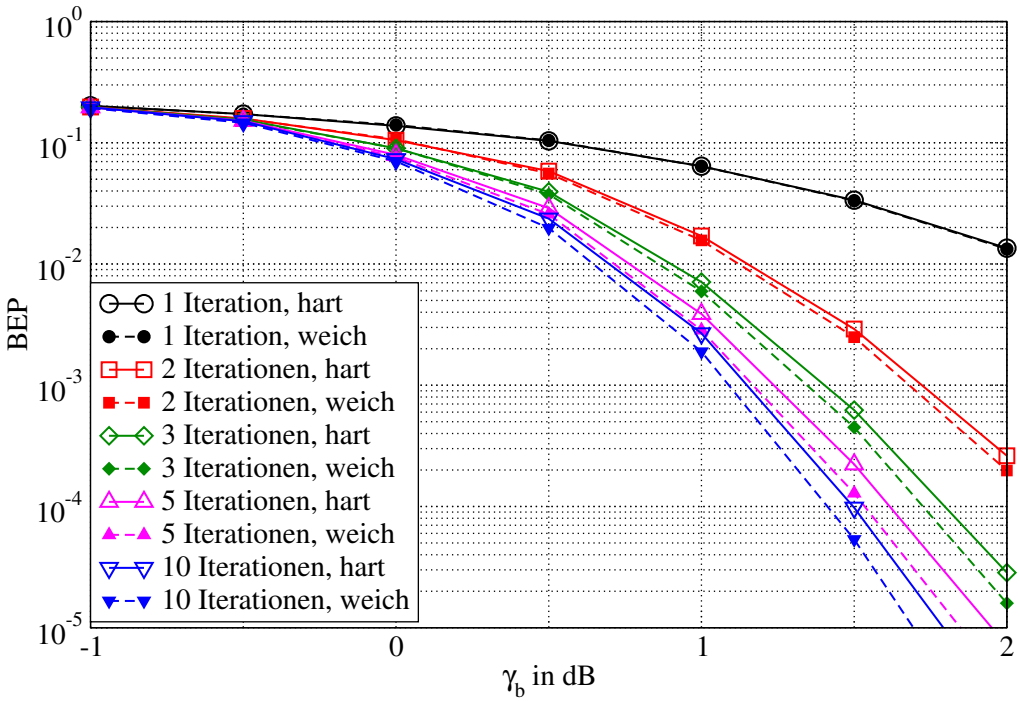

Abbildung 4.13: Weiche und harte Bitfehlerschätzung für verschieden viele Iterationen der Turbo-Decodierung eines parallel verketteten Faltungscodes bei Übertragung über einen AWGN-Kanal.

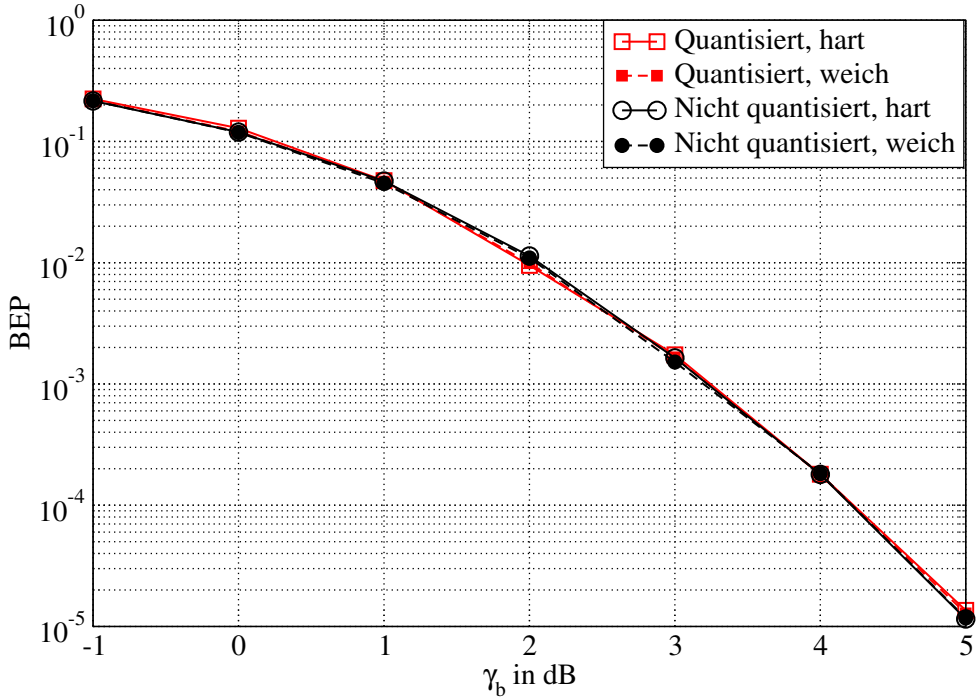

Abbildung 4.14: Weiche und harte Bitfehlerschätzung für einen $(23_8, 35_8)$ Faltungscode bei Verwendung des Log-BCJR-Algorithmus in quantisierter und nicht quantisierter Form. Die Festkommaquantisierung ist $\mathrm{fp}_{16,2}$ für die Zustandsmetriken und $\mathrm{fp}_{8,2}$ für alle anderen Variablen.

4.3.2.3 Hard-input soft-output Decodierung

Die in Abschnitt 4.2.3 vorgestellte Metrik für die hard-input soft-output Decodierung soll hier für die Schätzung der BEP verwendet werden. Für den hier genutzten AWGN-Kanal ergibt sich die analytisch angebbare Bitfehlerwahrscheinlichkeit für BPSK-Übertragung zu

$$\mathrm{P}(c_n \neq \hat{c}_n) = \frac{1}{2}\,\mathrm{erfc}\left(\sqrt{\gamma_s}\right) \ . \tag{4.25}$$

Die Wahrscheinlichkeit aus (4.25) wird nun zusammen mit (4.15) verwendet, um einen hard-input soft-output BCJR-Algorithmus zu implementieren. Die Ergebnisse der harten und weichen Schätzung bei Verwendung des $(23_8, 35_8)$-Faltungscodes sind in Abb. 4.15 dargestellt. Im Gegensatz zu der, ebenfalls in Abb. 4.15 gegebenen, soft-input soft-output Decodierung ist der Codiergewinn um ca. 2 dB geringer. Der geringere Codiergewinn resultiert daraus, das bei der hard-input Decodierung dem Decodierer keine Zuverlässigkeitsinformation über die einzelnen Bits zur Verfügung stehen. Beachtlich ist allerdings, dass auch für hard-input Decodierung harte und weiche Schätzwerte der BEP übereinstimmen. Das bedeutet, dass die in dieser Arbeit vorgestellten Konzepte nicht zwangsläufig soft-

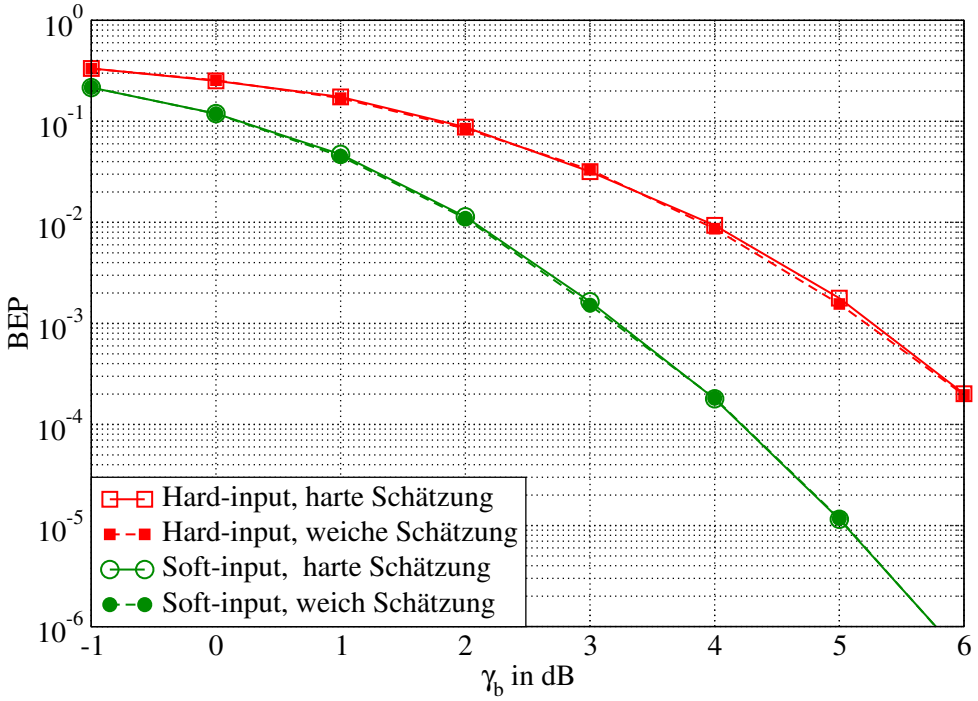

Abbildung 4.15: Harte und weiche Schätzung der BEP bei Verwendung von optimaler soft-input und hard-input Decodierung für Übertragung über einen AWGN-Kanal und Verwendung des $(23_8, 35_8)$-Faltungscodes.

input Decodierer benötigen, zumindest nicht dann, wenn die Fehlerwahrscheinlichkeit des (uncodierten) Übertragungskanals bekannt ist. Auch wenn die Fehlerwahrscheinlichkeit der uncodierten Übertragung in realen Systemen nicht analytisch bestimmt werden kann, so ist es doch weniger aufwändig, sie durch eigens hierfür übermittelte Pakete zu schätzen,

als die Fehlerwahrscheinlichkeit des codierten Systems. In diesem Fall können also kostengünstige hard-output Detektoren mit soft-output Decodierern „veredelt“ werden, um weiche Schätzwerte der BEP zu erzeugen.

Um zu untersuchen, wie empfindlich diese Art der HISO-Decodierung ist, werden nun Fälle betrachtet, in denen die im Decodierer verwendete Fehlerwahrscheinlichkeit $\mathrm{P}(c_n \neq \hat{c}_n)$ nicht der tatsächlichen entspricht, sondern durch einen Faktor f_e verfälscht wird. Die im Decodierer verwendete Fehlerwahrscheinlichkeit

$$\mathrm{P}(c_n \neq \hat{c}_n)' = f_e \, \mathrm{P}(c_n \neq \hat{c}_n) \tag{4.26}$$

ist um den Faktor f_e zu hoch oder zu niedrig. In Abb. 4.16 sind BEP-Schätzungen für den $(23_8, 35_8)$-Faltungscodes für $f_e = 1/5, 1/2, 1, 2$ und 5 angegeben. Der Decodierer ist ein HISO-BCJR-Algorithmus, die Ergebniss für $f_e = 1$ entsprechen denen aus Abb. 4.15. Wie zu erwarten ist die weiche Schätzung für $f_e > 1$ zu hoch und für $f_e < 1$ zu niedrig.

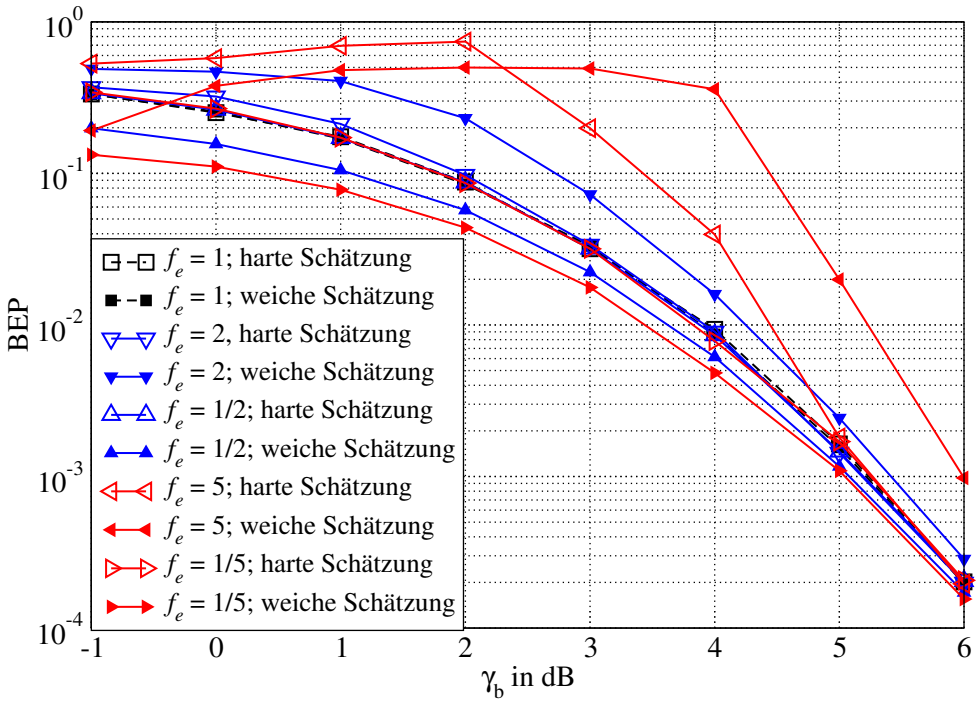

Abbildung 4.16: Harte und weiche Schätzung der BEP bei Verwendung von HISO-Decodierung und um den Faktor f_e verfälschte Bitfehlerwahrscheinlichkeit.

Allerdings verschlechtert sich für $f_e > 1$ auch die harte Schätzung. Für $f_e = 5$ steigt die harte BEP-Schätzung auf 0,7 an, weiche und harte Schätzung liegen weit auseinander – der Decodierer funktioniert zumindest für kleine γ_b nicht mehr. Bei Verwendung dieser Art von HISO-Decodierung ist es also sinnvoll, $\mathrm{P}(c_n \neq \hat{c}_n)'$ lieber zu klein als zu groß anzusetzen.

4.4 Wortfehlerwahrscheinlichkeit

Häufig wird argumentiert, dass die *Wortfehlerwahrscheinlichkeit* (engl. word error probability, WEP) für die Beurteilung eines Übertragungsverfahrens von höherer Bedeutung als die Bitfehlerwahrscheinlichkeit sei, da fehlerbehaftete Pakete sowieso verworfen würden (siehe Abschnitt 3.5). Die Wortfehlerwahrscheinlichkeit ist definiert als

$$P_W = \mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}) \tag{4.27}$$

und ist die *durchschnittliche* Wahrscheinlichkeit dafür, dass ein Infowort nach der Übertragung und Decodierung fehlerhaft ist [Fri96], also mindestens ein fehlerhaftes Bit enthält. Daraus ist sofort ersichtlich, dass die WEP normalerweise (um Größenordnungen) größer als die BEP ist. Im Folgenden wird daher die Idee der weichen Schätzung auch auf die WEP übertragen.

4.4.1 Schätzung der Wortfehlerwahrscheinlichkeit

Die Definition der WEP ist nahezu analog zu der der BEP. Ein Wortfehlerereignis liegt vor, wenn

$$\begin{aligned} \boldsymbol{u}_a = \hat{\boldsymbol{u}}_a &: W_{h,a} = 0 \\ \boldsymbol{u}_a \neq \hat{\boldsymbol{u}}_a &: W_{h,a} = 1 \,, \end{aligned} \tag{4.28}$$

also das a-te geschätzte Infowort $\hat{\boldsymbol{u}}_a$ nicht mit dem übertragenen Infowort $\boldsymbol{u}_a$ übereinstimmt. Damit läßt sich die *Wortfehlerrate* (engl. word error rate, WER) als durchschnittliche Anzahl Fehlerereignisse bzw. Mittelwert von $W_{h,a}$ berechnen:

$$\hat{P}_W = \frac{1}{A} \sum_{a=1}^{A} W_{h,a} \,. \tag{4.29}$$

Die Idee der weichen Monte-Carlo-Wortfehlersimulation wurde in [FSH06] vorgestellt und in [FH08b] im Vergleich mit der Monte-Carlo-Bitfehlersimulation vertieft. Analog zu der Bitfehlerschätzung wird statt des harten Fehlerereignisses in (4.28) ein weiches Fehlerereignis, nämlich die Fehlerwahrscheinlichkeit des a-ten Infoworts

$$W_{w,a} = \mathrm{P}(\hat{\boldsymbol{u}}_a \neq \boldsymbol{u}_a | \tilde{\boldsymbol{c}}_a) \tag{4.30}$$

betrachtet und darüber der Mittelwert

$$\widehat{P}_W = \frac{1}{A} \sum_{a=1}^{A} W_{w,a} = \frac{1}{A} \sum_{a=1}^{A} \mathrm{P}(\hat{\boldsymbol{u}}_a \neq \boldsymbol{u}_a | \tilde{\boldsymbol{c}}_a) \tag{4.31}$$

gebildet. Betrachtet man (4.29) und (4.31) als Schätzer, so stellen sie Mittelwertschätzer für die WEP dar und sind damit erwartungstreu [SW02]. Wie auch für die BEP (Abschnitt 4.3) gilt für $A \longrightarrow \infty$ nach dem Gesetz der großen Zahlen $\hat{P}_W = P_W$ und $\widehat{P}_W = P_W$. Somit gilt also auch

$$\hat{P}_W = \widehat{P}_W \text{ für } A \longrightarrow \infty \,. \tag{4.32}$$

Das weiche Schätzverfahren nach (4.31) weist die höhere Genauigkeit auf, d. h., die Varianz der weichen WEP-Schätzung $\sigma^2_{W,w}$ ist kleiner als die der harten WEP-Schätzung $\sigma^2_{W,h}$. Hierfür wird die in [FH08b] veröffentlichte Argumentation wiedergegeben. Ähnlich ist der Beweis für die höhere Genauigkeit der weichen BEP-Schätzung in [LH03].

Geringere Varianz der weichen WEP-Schätzung: Der Mittelwert der Schätzung nach (4.31) (4.29) ist identisch $\hat{P}_W = \widehat{P}_W = P_W$. Aus der Varianz des weichen

$$\sigma^2_{W,w} = \mathrm{E}\left\{W_w^2\right\} - P_W^2 \tag{4.33}$$

und des harten Schätzers

$$\sigma^2_{W,h} = \mathrm{E}\left\{W_h^2\right\} - P_W^2 \tag{4.34}$$

ergibt sich

$$\sigma^2_{W,w} - \sigma^2_{W,h} = \mathrm{E}\left\{W_w^2\right\} - \mathrm{E}\left\{W_h^2\right\} \ . \tag{4.35}$$

Mit $\mathrm{E}\{W_h^2\} = \mathrm{E}\{W_h\} = P_W = \mathrm{E}\{W_w\}$ wird (4.35) zu

$$\sigma^2_{W,w} - \sigma^2_{W,h} = \mathrm{E}\left\{W_w^2\right\} - \mathrm{E}\left\{W_w\right\} \ . \tag{4.36}$$

Für $W_w \in (0,1)$ ist die rechte Seite in (4.36) immer negativ, da $\mathrm{E}\{W_w^2\} < \mathrm{E}\{W_w\}$ gilt. Die Wahrscheinlichkeitsdichteverteilung der Wortfehlerereignisse, also die Verteilung von W_w, spielt dabei keine Rolle. Also gilt $\sigma^2_{W,w} - \sigma^2_{W,h} < 0$ und daher auch $\sigma^2_{W,w} < \sigma^2_{W,h}$. Für extreme, in der Praxis irrelevante, Verteilungen $W_w \in [0,1]$ gilt $\mathrm{E}\{W_w^2\} \leq \mathrm{E}\{W_w\}$, d. h., die beiden Varianzen können gleich sein.

Durch die geringere Varianz der weichen Schätzung ist es wahrscheinlicher, dass eine weiche WEP-Schätzung basierend auf einer bestimmten Anzahl Messungen ein genaueres Ergebnis liefert als eine harte Schätzung. Wie schon für die weiche BEP-Schätzung (in Abschnitt 4.3) ist auch hier die Kenntnis des gesendeten Infoworts im Gegensatz zur harten WEP-Schätzung nicht notwendig. Interesse an weichen WEP-Schätzungen bestand in der Vergangenheit vor allem im Zusammenhang mit ARQ-Verfahren, um ohne einen fehlererkennenden Code eine Entscheidung über eine Neuanforderungsanfrage zu treffen. Dabei beschränkt sich die WEP-Schätzung auf ein einzelnes Empfangswort ($A = 1$). Näher wird diese Vorgehensweise in Kapitel 5 untersucht. Natürlich können auch wieder Kurzzeitmittelwerte der WEP-Schätzung zur Adaption und allgemeinen Qualitätsbeurteilung einer Übertragung verwendet werden. Die weiche und harte WEP-Schätzung ist für unterschiedliche Faltungscodes in Abb. 4.17 angegeben. Der verwendete Decodierer ist der ROVA. Es wurden ausreichend viele Übertragungen simuliert, damit weiche und harte Schätzung übereinstimmmen.

4.4.2 Fehler bei der weichen Schätzung der Wortfehlerwahrscheinlichkeit

Die weiche Schätzung der WEP nach (4.31) ist nur exakt, wenn der Decodierer die tatsächliche Fehlerwahrscheinlichkeit für das geschätzte Codewort liefert (APP-Sequenzschätzer). Diese Anforderung erfüllt für gleichverteilte Codewortwahrscheinlichkeiten der ROVA.

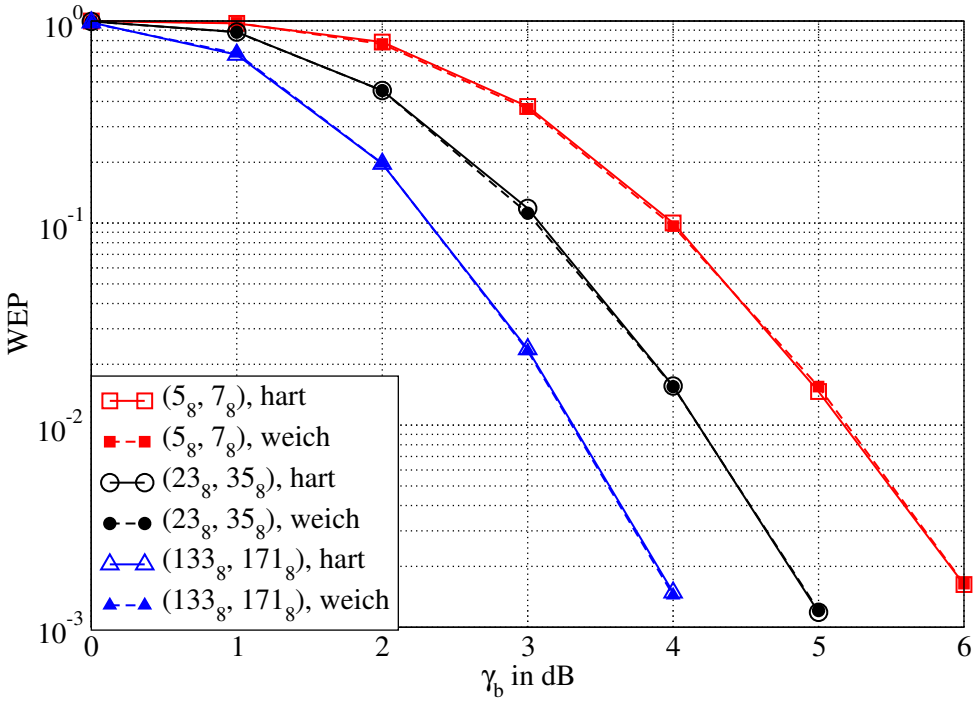

Abbildung 4.17: Weiche nach und harte Wortfehlerschätzung nach (4.31) bzw. (4.29) bei Verwendung des ROVA (optimaler APP-Sequenzschätzer). Hier wurden ausreichend viele Übertragungen simuliert, um die Differenz zwischen weicher und harter Schätzung sehr klein werden zu lassen.

Im Gegensatz zur weichen BEP-Schätzung ist man auf relativ spezielle Decodierer angewiesen – es ist schwierig, gute suboptimale APP-Sequenzschätzer zu finden. Auf Grund des modularen Aufbaus von Empfängerstrukturen ist es häufig interessanter, die Zuverlässigkeit einzelner Symbole zu kennen, als die kompletter, zusammenhängender Folgen von Symbolen. So benötigen z. B. Entzerrung, Kanalschätzung und iterative Verfahren symbolweise Zuverlässigkeiten.

Daher wird neben dem Einfluss der Quantisierung im Nachfolgenden vor allem der Einfluss suboptimaler Decodierer (im Sinne der APP-Sequenzschätzung) betrachtet.

4.4.2.1 Suboptimale Decodierer

Symbolschätzer Symbolschätzer sind gut geeignet für die BEP-Schätzung (siehe Abschnitt 4.3.1) und andererseits in modernen Empfängerstrukturen (iterative Entzerrung/ Decodierung) häufig vorhanden. Schön wäre also die Berechnung der WEP aus den Zuverlässigkeitsinformationen für die einzelnen Bits. Da die Berechnung der Fehlerwahrscheinlichkeiten der Infobits $\mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})$ direkt möglich ist (4.23), scheint es einfach, auch die Infowortfehlerwahrscheinlichkeit $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ daraus zu berechnen. Der hier gezeigte Ansatz wurde in [FSH06] veröffentlicht. Darin wird doppelt ausgenutzt, dass die Wahrscheinlichkeit für fehlerhafte und fehlerfreie Übertragung komplementär sind. So ist die

Infowortfehlerwahrscheinlichkeit gegeben weiche Eingangswerte

$$\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}}) = 1 - \mathrm{P}(\hat{\boldsymbol{u}} = \boldsymbol{u}|\tilde{\boldsymbol{c}}) \;, \tag{4.37}$$

wobei auch

$$\mathrm{P}(\hat{\boldsymbol{u}} = \boldsymbol{u}|\tilde{\boldsymbol{c}}) \approx \prod_{k=1}^{K} (1 - \mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})) \tag{4.38}$$

über die Komplemente der Wahrscheinlichkeit der fehlerfreien Übertragung der K einzelnen Infobits u_k dargestellt werden kann. Die Approximation in (4.38) ergibt sich daraus, dass die Zuverlässigkeit der Informationsbits bzw. ihre Fehlerwahrscheinlichkeiten $\mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})$ nicht statistisch unabhängig sind (siehe Abschnitt 4.2.4). Zusammen mit (4.19) kann $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ somit approximativ aus der Zuverlässigkeitsinformation für die Informationsbits bestimmt werden:

$$\begin{aligned}
\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}}) &\approx 1 - \prod_{k=1}^{K} (1 - \mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})) \\
&= 1 - \prod_{k=1}^{K} \left(1 - \frac{1}{1 + \mathrm{e}^{|\tilde{u}_k|}}\right) \\
&= 1 - \prod_{k=1}^{K} \frac{1}{1 + \mathrm{e}^{-|\tilde{u}_k|}} \;.
\end{aligned} \tag{4.39}$$

Die laut (4.39) benötigten Berechnung wirken auf den ersten Blick aufwändiger als für die BEP in (4.23). Wieder vereinfacht sich die Berechnung, wenn logarithmierte Wahrscheinlichkeiten betrachtet werden. Damit lässt sich sich (4.38) zu

$$\begin{aligned}
\ln \mathrm{P}(\hat{\boldsymbol{u}} = \boldsymbol{u}|\tilde{\boldsymbol{c}}) &\approx \sum_{k=1}^{K} \ln(1) - \ln(1 + \mathrm{e}^{-|\tilde{u}_k|}) \\
&= -\sum_{k=1}^{K} \ln(1 + \mathrm{e}^{-|\tilde{u}_k|})
\end{aligned} \tag{4.40}$$

vereinfachen. Der Term $\ln(1 + \mathrm{e}^{-|\tilde{u}_k|})$ entspricht dem Korrekturterm des Log-BCJR (siehe Anhang B) und kann mit geringem Aufwand über eine Tabelle implementiert werden. Der wesentliche Vorteil dieser Methode zur Schätzung der WEP ist die Einfachheit und Flexibilität: Es spielt keine Rolle woher die Zuverlässigkeitsinformation für die Symbole kommt – solange die Zuverlässigkeitsinformation gut genug ist, um sie zur Berechnung von $\mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})$ nach (4.19) zu nutzen. Die Berechnung ist dabei nicht an den Decodierer selbst gebunden, sondern kann auch nachgelagert werden. Der Nachteil ist die Annahme der statistische Unabhängigkeit der Fehlerwahrscheinlichkeiten der einzelnen Infobits innerhalb eines Infoworts gemäß (4.38). Die Auswirkung dieser Annahme ist anhand der weichen und harten WEP-Schätzung für Codes unterschiedlicher Generatorpolynome in Abb. 4.18 dargestellt. Es ist deutlich zu sehen, dass die Approximation in (4.40) eine Abweichung der weichen WEP-Schätzung von der tatsächlich WEP bzw. harten Schätzung

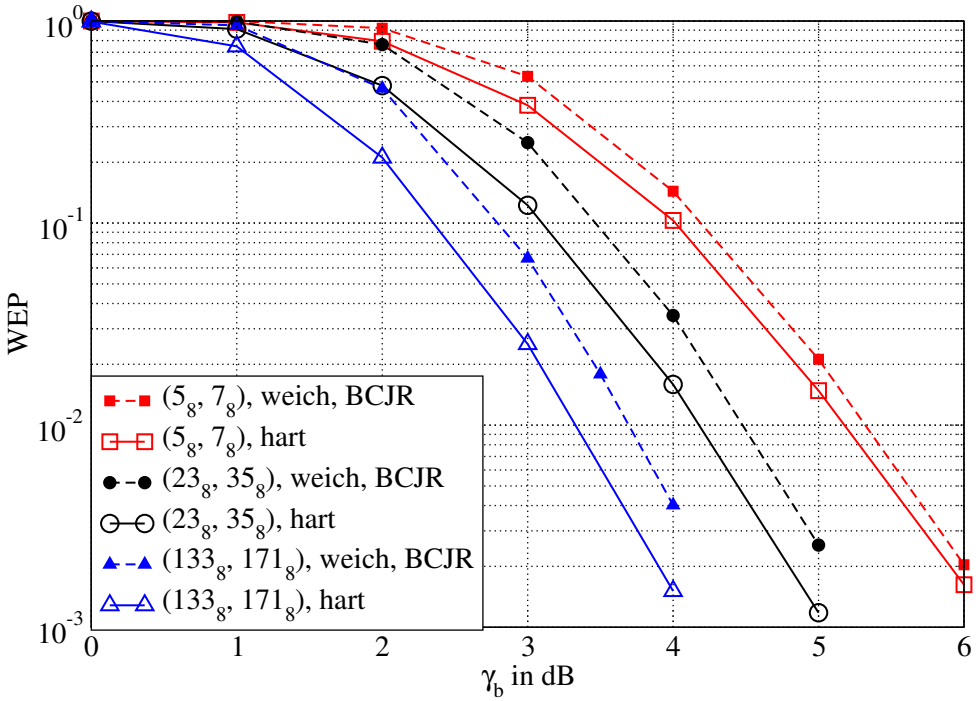

Abbildung 4.18: Weiche Wortfehlerschätzung nach (4.39) im Vergleich zur harten Schätzung.

verursacht. Da in Abschnitt 4.2.4 gezeigt wurde, dass die Zuverlässigkeiten der Infobits nicht statistisch unabhängig sind, kann die Abweichung der weichen Schätzung auf die statistischen Abhängigkeiten der weichen Infobitschätzwerte zurückgeführt werden. Die Korrelation zwischen den Infobits nimmt mit der Gedächtnislänge des Codes zu was sich aus in der Stärke der Abweichung zwischen weicher Schätzung und WEP widerspiegelt - je größer die Gedächtnislänge, desto stärker die statistischen Abhängigkeiten der Infobits, desto schlechter die Schätzung. Andererseits ist die Schätzung relativ gut. Die Kurven zeigen im Bereich relevanter γ_b einen nahezu parallelen Verlauf. Damit ist eine einfache Nachbearbeitung (Verschiebung) der weichen Schätzung möglich, um die tatsächliche WEP zu ermitteln.

Optimaler Subblock-by-Subblock Decodierer Der optimale Subblock-by-Subblock-Decodierer (OBBD) wurde in [Hoe95] vorgeschlagen. Er berechnet die Wahrscheinlichkeiten für $(K+L)/B$ Infowortteilblöcke $\mathrm{P}(u_{qB+1}^{(q+1)B}|\tilde{\boldsymbol{c}})$ der Länge B mit $q = 0, 1, \ldots (K+L)/B-1$. Details zur Realisierung finden sich in Anhang B.

Aus Abb. 4.2 ist ersichtlich, dass die statistischen Abhängigkeiten eines Faltungscodes besonders stark zwischen benachbarten Infobits sind. Wird nun die Verbundwahrscheinlichkeit für diese benachbarten Bits berechnet, so sind die statistischen Abhängigkeiten in dieser schon enthalten. Analog zu (4.39) kann $\mathrm{P}(\hat{\boldsymbol{u}} = \boldsymbol{u}|\tilde{\boldsymbol{c}})$ approximativ berechnet

werden:

$$\begin{aligned} \mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}}) &= 1 - \mathrm{P}(\hat{\boldsymbol{u}} = \boldsymbol{u}|\tilde{\boldsymbol{c}}) \\ &\approx 1 - \prod_{q=1}^{(K+L)/B-1} \mathrm{P}(\hat{u}_{qB+1}^{(q+1)B} = u_{qB+1}^{(q+1)B}|\tilde{\boldsymbol{c}}) \; . \end{aligned} \tag{4.41}$$

Auf Grund der Natur des OBBD ist der BCJR für $B = 1$ als Spezialfall enthalten. Dies gilt sowohl für den Algorithmus wie auch (4.41), da $\mathrm{P}(\hat{u}_{qB+1}^{(q+1)B} = u_{qB+1}^{(q+1)B}|\tilde{\boldsymbol{c}}) = \mathrm{P}(\hat{u}_k = u_k|\tilde{\boldsymbol{c}})$ gilt wenn $B = 1$ ist. Wichtig in Bezug auf den OBBD ist, dass der Aufwand der Decodierung für $B = L$ am geringsten ist. Der Aufwand ist in diesem Fall sogar geringer als für den BCJR-Algorithmus [Hoe95]. Die LLRs für die einzelnen Infobits werden aus den Blockwahrscheinlichkeiten berechnet und entsprechen denen des BCJR-Algorithmus. Daher wurden die Ergebnisse in Abb. 4.19 für eine Blocklänge $B = L$ simuliert. Durch die

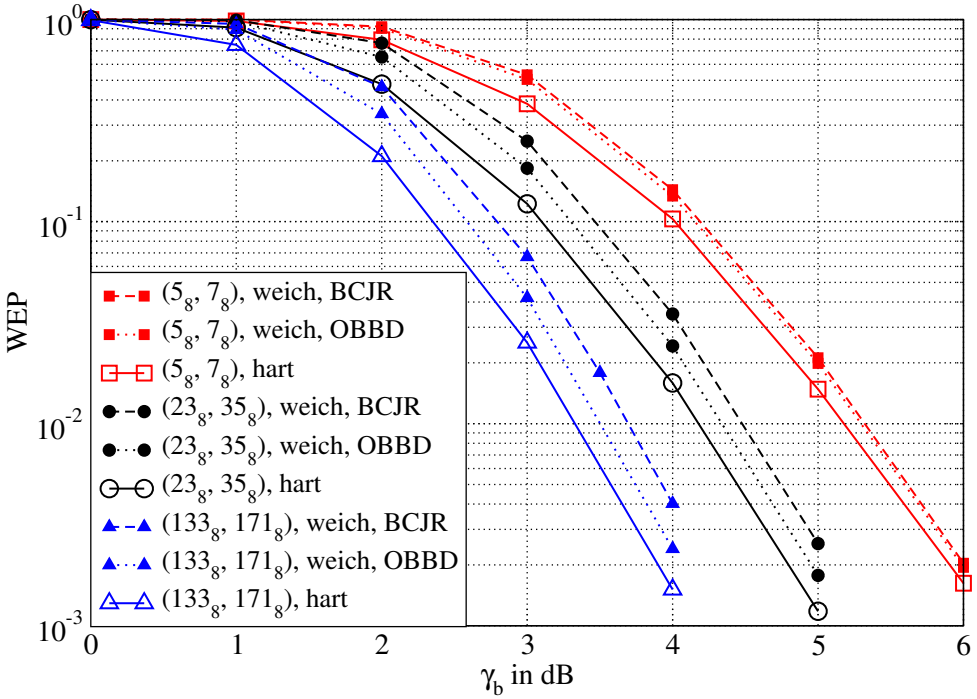

Abbildung 4.19: Weiche Wortfehlerschätzung des OBBD im Vergleich zur weichen Wortschätzung des BCJR-Algorithmus und der harten Schätzung. Die harte Schätzung ist für OBBD und BCJR-Algorithmus identisch.

Verwendung des OBBD wird der Abstand zwischen weicher und harter WEP-Schätzung in dB um ungefähr die Hälfte verringert. Diese Verbesserung ist ohne Aufwandserhöhung möglich. Die Möglichkeit, den OBBD für die Berechnung der Infowortfehlerwahrscheinlichkeit zu verwenden, wurde in [FSH06] veröffentlicht.

Aufwandsreduzierter ROVA Der ROVA berechnet die exakten Infowortfehlerwahrscheinlichkeiten, wodurch auch die weiche WEP-Schätzung exakt ist. Nachteil des ROVAs

ist der im Gegensatz zum VA erhöhte Aufwand durch die Berechnung von jeweils zwei Wahrscheinlichkeiten für alle Zustände: Der Wahrscheinlichkeit dafür, dass der in einem Zustand ausgewählte Pfad richtig ist, $\mathrm{P}(S_k^j)$, und der Wahrscheinlichkeit $\mathrm{P}(\overline{S}_k^j)$ dafür, dass der in einem Zustand ausgewählte Pfad nicht richtig ist. Dies soll an dem Trellisausschnitt in Abb. 4.20 deutlich gemacht werden (Details siehe Anhang B). Zusätzlich zu den Zus-

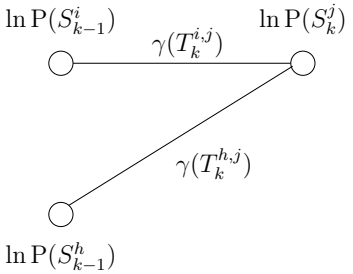

Abbildung 4.20: Trellisausschnitt mit vom ROVA berechneten Wahrscheinlichkeiten.

tandsmetriken des herkömmlichen VA, $\gamma(T_k^{i,j}) = \ln \mathrm{P}(T_k^{i,j})$, berechnet der ROVA in der Vorwärtsrekursion die Wahrscheinlichkeiten

$$\ln \mathrm{P}(S_k^j) = \Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) - \Lambda_k \tag{4.42}$$

und

$$\ln \mathrm{P}(\overline{S}_k^j) = \max{}^\star \left[\max_{\substack{i=1\ldots I \\ i \neq j}}{}^\star \left(\ln \mathrm{P}(S_{k-1}^i) + \gamma(T_k^{i,j}) \right), \max_{i=1\ldots I}{}^\star \left(\ln \mathrm{P}(\overline{S}_{k-1}^i) + \gamma(T_k^{i,j}) \right) \right] - \Lambda_k \ , \tag{4.43}$$

wobei

$$\Lambda_k = \max{}^\star \left[\max_{\substack{i=1\ldots I \\ j=1\ldots J}}{}^\star \left(\ln \mathrm{P}(S_{k-1}^i) + \gamma(T_k^{i,j}) \right), \max_{\substack{i=1\ldots I \\ j=1\ldots J}}{}^\star \left(\ln \mathrm{P}(\overline{S}_{k-1}^i) + \gamma(T_k^{i,j}) \right) \right] \tag{4.44}$$

einen Normalisierungsterm darstellt. Im letzten Zustand des (terminierten) Trellisdiagramms ist $\mathrm{P}(\overline{S}_{K+L}^n)$ die Wahrscheinlichkeit dafür, dass der überlebende Pfad in diesem Zustand nicht korrekt ist. Also ist die Fehlerwahrscheinlichkeit des überlebenden Pfades $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}}) = \mathrm{P}(\overline{S}_{K+L}^j)$. Dabei ist die Summe aller möglichen Ereignisse (die Summe von (4.42) und (4.43)) gleich eins. Damit stellt $\ln \mathrm{P}(S_k^j)$ keine gültige Pfadmetrik für den VA dar. Pro Zustand müssen zwei weitere Werte gespeichert und berechnet werden. Insbesondere für viele Zustände, also große Gedächtnislängen, steigt der Aufwand dadurch stark an.

Mit dem vereinfachten ROVA (engl. simplified ROVA, SROVA) wurde eine Möglichkeit gefunden, den zusätzlichen Aufwand stark zu reduzieren [FH07]. Dies wird erreicht, indem nur eine Näherung von $\mathrm{P}(S_k^j)$ berechnet wird. Diese Näherung erlaubt, dass die notwendigen Berechnungen direkt auf Basis der Pfadmetriken $\Gamma(S_{k-1}^i)$ und $\Gamma(S_{k-1}^h)$ entsprechend Abb. 4.21 berechnet werden können. Sei o. B. d. A. der Übergang $T_k^{i,j}$ der vom VA aus-

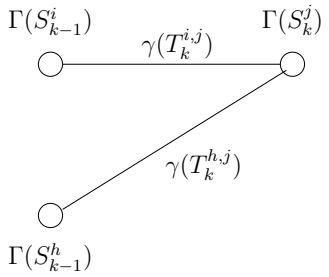

Abbildung 4.21: Trellisausschnitt mit vom SROVA berechneten Zweigmetriken $\gamma(\cdot)$ und partiellen Pfadmetriken $\Gamma(\cdot)$.

gewählte Zweig. Die Wahrscheinlichkeit dafür, dass aus den in einem Zustand endenden Zweigen der richtige ausgewählt wird, ist

$$\ln \mathrm{P}(S_k^j) = \Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) - \Lambda_k \;, \tag{4.45}$$

wobei

$$\Lambda_k = \max_{i=1\ldots I}{}^{\star} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j})\right) \;. \tag{4.46}$$

Für (4.45) wird angenommen, dass einer der vorhergehenden Zustände, entweder S_{k-1}^i oder S_{k-1}^h richtig ist, d. h., die Wahrscheinlichkeiten $\mathrm{P}(\overline{S}_{k-1}^i)$ und $\mathrm{P}(\overline{S}_{k-1}^h)$ werden nicht berücksichtigt. Daher ist (4.45) nicht exakt, benötigt aber die Normalisierung in (4.42) und (4.43) nicht. Ferner sind $\mathrm{P}(S_k^j)$ und $\mathrm{P}(\overline{S}_k^j)$ komplementär, d. h., nur einer der Werte muss berechnet und gespeichert werden. Diese einfachere Berechnung ist vor allem dann bedeutend, wenn für große L die Anzahl der Zustände sehr groß wird.

Die in (4.45) und (4.46) durchgeführten Rechnungen werden auch für die Berechnung der Zuverlässigkeitsinformation im SOVA [HH89] durchgeführt. Da hier der Schwerpunkt auf der Berechnung der Wahrscheinlichkeit der falschen Zweigauswahl, nicht der Bitfehlerwahrscheinlichkeit, liegt, ist keine Aktualisieren dieser Wahrscheinlichkeiten wie im SOVA notwendig. Mit der Rückverfolgung des wahrscheinlichsten Pfades wird näherungsweise

$$\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}}) = 1 - \prod_{k=1}^{K+L} \mathrm{P}(S_k^j) \tag{4.47}$$

berechnet. Damit reduziert sich der zusätzliche Aufwand der Berechnung der Wortfehlerwahrscheinlichkeit auf (4.47), sofern der SOVA ohnehin für die Berechnung von Zuverlässigkeitsinformation für die einzelnen Bits verwendet wird. In (4.47) stellt S_k^j die Zustände dar, die Teil des überlebenden Pfades sind. Die in (4.45) getroffene Annahme ist richtig, wenn der in S_{k-1}^i oder S_{k-1}^h endende Pfad richtig ist. Das ist dann der Fall, wenn der richtige Pfad der ML-Pfad durch das Trellisdiagramm ist, also das richtige Codewort dem ML-Pfad entspricht. Daher ist auch die Wahrscheinlichkeit $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ nach (4.47) exakt, wenn das geschätzte Codewort dem ML-Pfad entspricht. Wenn der ML-Pfad allerdings Fehler enthält, werden in (4.47) approximierte Wahrscheinlichkeiten aufmultipliziert. Je länger der Fehlerpfad ist, d. h., je länger die freie Distanz des Codes ist, um so größer ist der entstehende Fehler durch die Approximation, wenn der ML-Pfad nicht der richtige

Pfad ist. Im Gegensatz dazu betrachtet der ROVA für die Normalisierung alle möglichen Ereignisse zum Zeitpunkt k. Daher berechnet er immer die exakte Wahrscheinlichkeit $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ – sogar wenn der ML-Pfad nicht korrekt ist.

Ein Vergleich zwischen der WEP-Schätzung auf Basis des (für die WEP-Schätzung optimalen) ROVAs und des SROVAs ist in Abb. 4.22 dargestellt. Harte Schätzung des ROVAs wie auch des SROVAs entsprechen der des VAs und sind damit identisch. Die weiche Schätzung des SROVAs weist nur einen geringen Fehler auf, der für große γ_b klein wird.

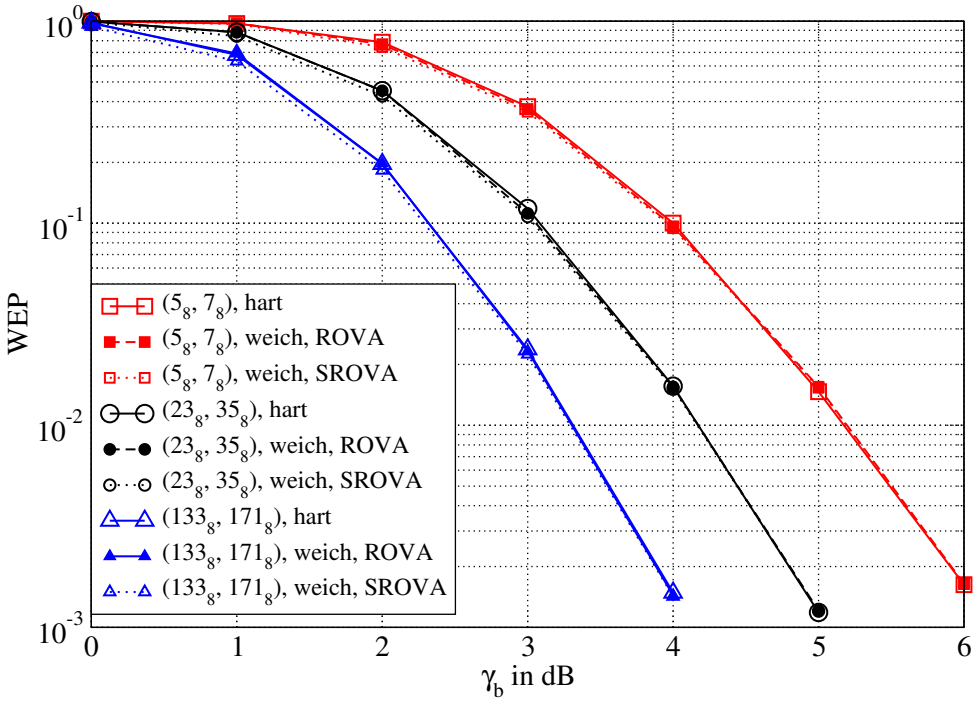

Abbildung 4.22: Weiche Wortfehlerschätzung des ROVAs im Vergleich zu der des SROVAs. Die harte Schätzung ist für beide identisch.

Belief Propagation Algorithmus Ähnlich dem BCJR-Algorithmus liefert der BPA Zuverlässigkeitsinformation für einzelne Infobits. Damit ist (4.39) ebenfalls anwendbar, um eine Approximation von $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ zu berechnen. Die darauf basierende WEP-Schätzung nach unterschiedlich vielen Iterationen ist in Abb. 4.23 dargestellt. Auf den ersten Blick sehen diese Ergebnisse vielversprechend aus: Weiche und harte Schätzung scheinen annähernd übereinzustimmen. Allerdings ist erkennbar, dass es für die weiche und die harte Schätzung Schnittpunkte gibt, die sich mit zunehmender Anzahl Iterationen in den Bereich kleinerer γ_b verschiebt. Der Grund hierfür liegt in der statistischen Abhängigkeit der Infobits, also der Approximation in (4.39), und der Suboptimalität des BPAs auf zyklischen Graphen. Ersteres führt zu einer WEP-Schätzung, welche zu pessimistisch ist, d. h., die weiche WEP-Schätzung ist höher als die harte. Dieser Effekt wurde für die optimale Infobitschätzungen in Abb. 4.18 gezeigt. Wie in Abschnitt 4.2.4 gezeigt

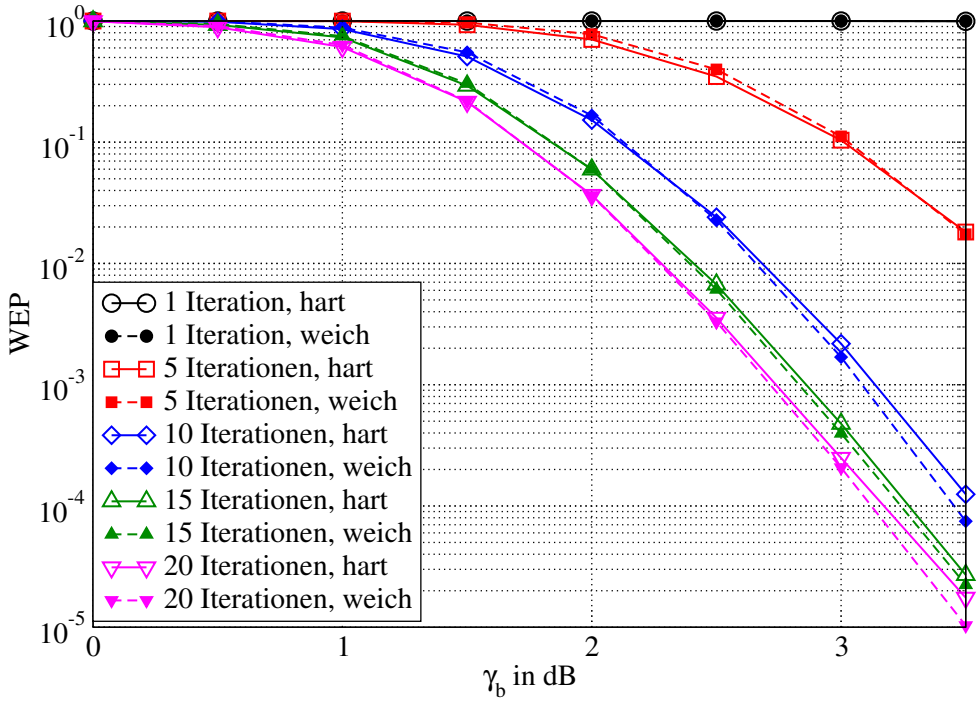

Abbildung 4.23: Weiche und harte WEP-Schätzung bei der Decodierung des $R = 1/2$ Codes aus [IEE05] für Decodierung mit dem BPA für verschieden viele Iterationen.

wurde, nehmen mit zunehmender Anzahl der Iterationen die statistischen Abhängigkeiten zwischen den weichen Infobits zu. Dies führt zu einer zu pessimistischen (zu hohen WEP-Schätzung). Die LLRs im BPA werden von Iteration zu Iteration größer, wenn ein gültiges Codewort gefunden wurde. Nach einigen Iterationen werden die LLRs so groß, dass eine WEP-Schätzung nach (4.39) zu optimistisch wird, also sogar den Effekt der zu pessimistischen Schätzung überkompensiert. Für den hier verwendeten LDPC-Code aus [IEE05] ergibt dies zwar insgesamt gute WEP-Schätzungen (siehe Abb. 4.23), die Schätzung kann aber für andere LDPC-Codes erheblich anders ausfallen.

4.4.2.2 Quantisierung

Mit dem ROVA existiert ein optimaler Sequenzschätzer der eine exakte WEP-Schätzung ermöglicht. Wie schon für die BEP-Schätzung soll hier kurz untersucht werden, ob diese Schätzung anfällig gegen den Einfluss einer Festkommaimplementierung ist.

Hierzu wird die weiche WEP-Schätzung für den ROVA mit Variablen in Festkommadarstellung nach Abschnitt 4.2.2 simuliert. Wie beim BCJR-Algorithmus in 4.4.2.2 ist es auch für den (RO)VA notwendig, dass die Zustandsmetrik ($\Gamma(S_k^i)$) einen größeren Darstellungsbereich aufweist, also mit 16 Bit zu einer Festkommadarstellung $\mathrm{fp}_{16,2}$ quantisiert wird. Alle anderen Variablen, z. B. weiche Eingangswerte, $\ln \mathrm{P}(S_k^i)$ und $\ln \mathrm{P}(\overline{S}_k^i)$, werden zu $\mathrm{fp}_{8,2}$ quantisiert. Entsprechende Ergebnisse sind für die weiche und harte WEP-Schätzung nach Abschnitt 4.4.1 in Abb. 4.24 dargestellt. Aus dem Ergebnissen wird ersichtlich, dass eine solche Quantisierung keinen Einfluss auf die Leistungsfähigkeit hat und

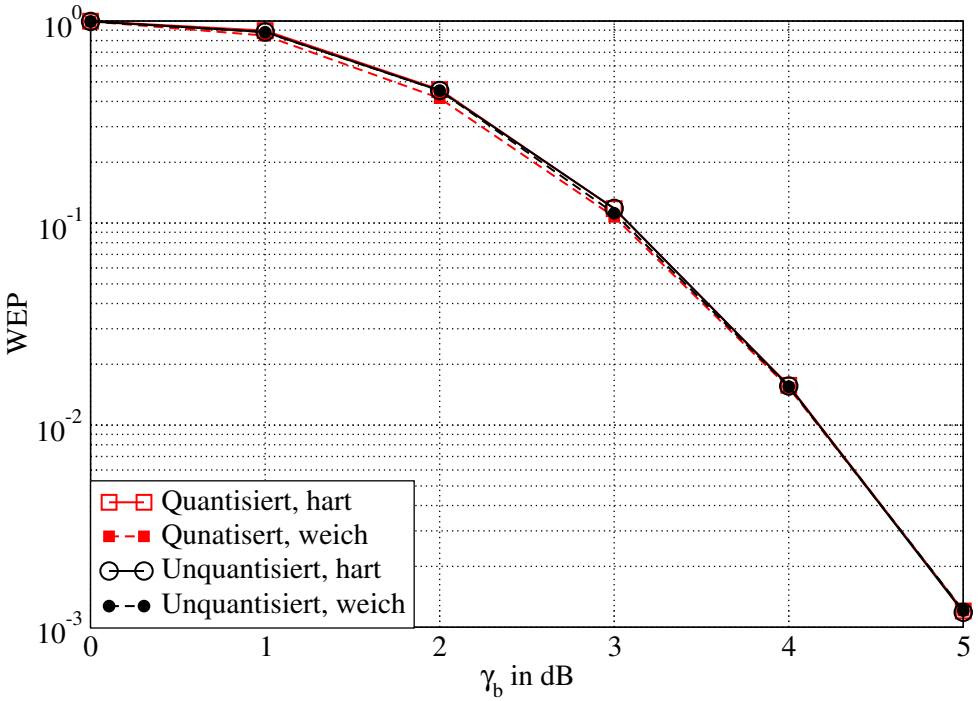

Abbildung 4.24: Weiche und harte Bitfehlerschätzung für einen $(23_8, 35_8)$ Faltungscode bei Verwendung des ROVAs in quantisierter und nicht quantisierter Form. Die Festkommaquantisierung ist $\mathrm{fp}_{16,2}$ für die Zustandsmetrik und $\mathrm{fp}_{8,2}$ für alle anderen Variablen.

nur eine sehr geringe Abweichung der weichen WEP-Schätzung verursacht. Diese Ergebnisse sind auch für andere Codes zu erwarten. Es ist allerdings möglich, dass sich der Bereich auftretender Zustandsmetriken vergrößert, da deren Betrag tendenziell mit Gedächtnislänge und Codewortlänge wächst. Dies würde entweder unterschiedliche Skalierungen für verschiedene k (effektive Emulation von Gleitkommazahlen) oder mehr Bit für die Darstellung als Festkommazahl erfordern. Davon wäre aber nicht nur der ROVA, sondern genauso der herkömmliche soft-input Viterbi-Algorithmus betroffen, da dieser mit denselben Zustandsmetriken arbeitet.

4.4.2.3 Hard-input soft-output Decodierung

Mit Hilfe der in (4.15) definierten Metrik für hard-input soft-ouput Decodierung kann ebenso wie die BEP (in Abschnitt 4.4.2.3) auch die WEP für trellis-decodierbare Codes geschätzt werden. Auch hier wird ein Faltungscode mit den Generatorpolynomen $(23_8, 35_8)$ verwendet, als Decodierer der ROVA.

Die Ergebnisse der harten und weichen Schätzung sind in Abb. 4.25 dargestellt. Wie schon für die BEP-Schätzung zeigt sich auch hier wieder die gute Qualität der weichen Schätzung. Zwar ergibt sich wieder ein um 2 dB geringerer Codiergewinn gegenüber der soft-input Decodierung, die weiche Schätzung der WEP stimmt aber mit der harten überein. Daraus ergeben sich dieselben Vorteile wie für die BEP-Schätzung (Abschnitt 4.3.2.3).

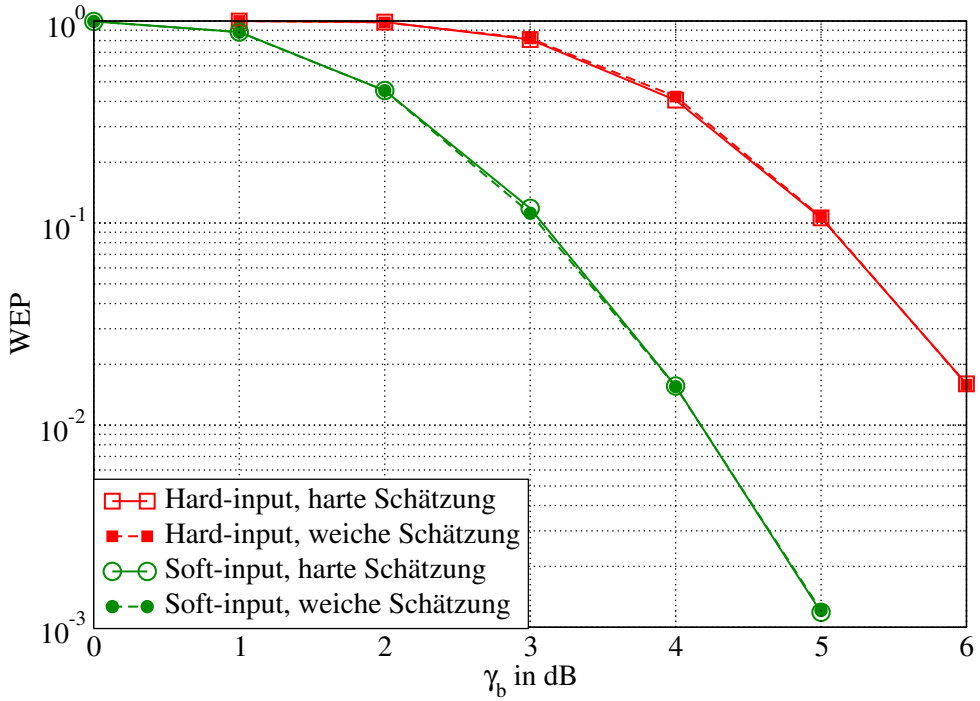

Abbildung 4.25: Harte und weiche Schätzung der WEP bei Verwendung von optimaler soft-input und hard-input Decodierung für Übertragung über einen AWGN-Kanal und Verwendung des $(23_8, 35_8)$-Faltungscodes.

Wie für die BEP-Schätzung in Abschnitt (4.3.2.3) soll untersucht werden, wie sich eine um einen Faktor f_e verfälschte uncodierte Bitfehlerwahrscheinlichkeit (4.26) auf die WEP-Schätzung mit dem ROVA auswirkt. Abb. 4.26 zeigt die Ergebnisse der weichen Schätzung für $f_e = 1/5, 1/2, 1, 2$ und 5. Ähnlich der BEP-Schätzung ist die weiche WEP-Schätzung entsprechend f_e zu hoch oder zu niedrig. Allerdings ist der ROVA robuster gegenüber den verfälschten Wahrscheinlichkeiten: Die harte Schätzung entspricht für alle f_e der unverfälschten, ist also unabhängig von der Größe des durch f_e hervorgerufenen Fehlers. Dies wird durch die Unabhängigkeit der harten Schätzwerte des VA vom SNR erklärt.

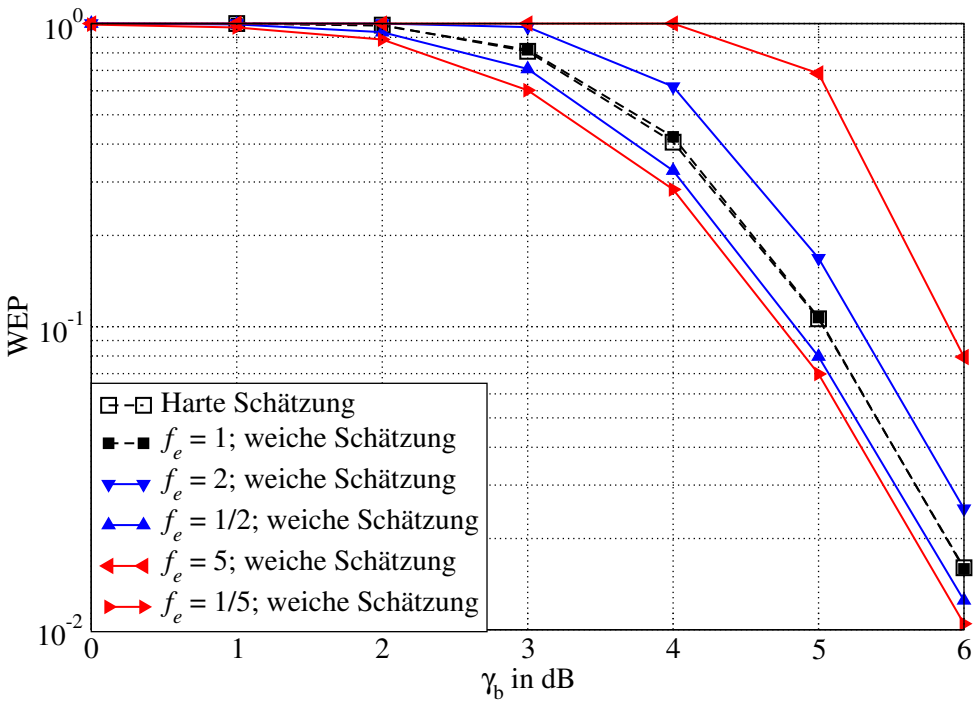

Abbildung 4.26: Harte und weiche Schätzung der WEP bei Verwendung von HISO-Decodierung mit dem ROVA und um den Faktor f_e verfälschte Bitfehlerwahrscheinlichkeit.

4.5 Genauigkeit von Bit- und Wortfehlerwahrscheinlichkeitsschätzung im Vergleich

Ein wesentlicher Punkt für die Überlegung, ob BEP oder WEP das geeignetere Maß zur Beurteilung der Übertragungsqualität eines Kommunikationssystems ist, ist neben der Möglichkeit der Messung auch ihre Genauigkeit. In diesem Abschnitt soll daher die Frage untersucht werden, ob eine der beiden Schätzungen, die BEP-Schätzung (Abschnitt 4.3.1) oder die WEP-Schätzung (Abschnitt 4.4.1), genauer ist.

Die Wortfehlerereignisse eines Übertragungssystems $W_1, \ldots, W_a, \ldots$ können durch eine Folge von Zufallsvariablen dargestellt werden. Entsprechend (4.28) stellt $W_a = 1$ ein Wortfehlerereignis in der a-ten Übertragung dar, während $W_a = 0$ die fehlerfreie Übertragung des a-ten Wortes darstellt. Die Wahrscheinlichkeit eines Wortfehlers ist also $P_W = \mathrm{P}(W_i = 1)$. Die Schätzverfahren für BEP und WEP stellen Mittelwertschätzer dar, deren Stichprobe einen Umfang A, der Anzahl der übertragenden Worte, darstellen. Der Schätzwert des Mittelwerts

$$\hat{\mu}_W = \frac{1}{A} \sum_{a}^{A} W_a \tag{4.48}$$

entspricht also der geschätzten WEP $\hat{P}_W$. Wenn die Zufallsvariablen W_a die gleiche Verteilung und statistische Unabhängigkeit aufweisen (engl. i.i.d.), dann gilt laut dem Gesetz der großen Zahlen [Pro02]

$$\mathrm{P}(|\hat{\mu}_W - \mu_W| > \varepsilon) \to 0 \tag{4.49}$$

für jedes $\varepsilon > 0$ wenn $A \to \infty$ gilt. Wird $\hat{\mu}_w$ selbst als Zufallsvariable angesehen, dann kann eine obere Grenze für die Wahrscheinlichkeit in (4.49) in Abhängigkeit von A anhand der Tschebyschow-Ungleichung (siehe z. B. [SW02]) gefunden werden:

$$\mathrm{P}(|\hat{\mu}_W - \mu_W| > \varepsilon) < \frac{\sigma_W^2}{A\varepsilon^2} \,. \tag{4.50}$$

Das Gesetz der großen Zahlen wird implizit in Monte-Carlo-Simulationen verwendet, um einen Schätzwert für P_W oder P_B zu erhalten. Aus (4.50) ist ersichtlich, dass die Wahrscheinlichkeit großer Schätzfehler für größer werdende A kleiner wird. Üblicherweise wird für die Anwendung des Gesetzes der großen Zahlen für die Zufallsvariablen gleiche Verteilung und statistische Unabhängigkeit gefordert. Dies ist z. B. für uncodierte Übertragung über einen AWGN-Kanal, bei der die Fehlerereignisse unabhängig voneinander und für jede Übertragung mit gleicher Wahrscheinlichkeit auftreten, gültig. Es existieren aber auch andere Formulierungen bzw. Gesetze der großen Zahlen [Pro02]. Eine Version fordert z. B., dass für die Korrelation der Zufallsvariablen gilt: $|\mathrm{R}(W_i, W_j)| \leq \phi(|i-j|)$ mit $a = |i-j| \to \infty$ und $\phi(a) \to 0$. Dies ist für die meisten Fälle von BEP- oder WEP-Schätzungen gegeben, in denen üblicherweise durch Codierung verursachte Korrelation der Fehlerwahrscheinlichkeiten innerhalb eines Codeworts (siehe Abschnitt 4.2.4) oder korrelierte Wortfehlerereignisse auf Grund von Fading-Prozessen bei aufeinanderfolgenden

Übertragungen auftreten. Für beide Fälle nimmt die Korrelation der Fehlerwahrscheinlichkeiten also mit zunehmendem zeitlichen Abstand der übertragenen Codewörter ab. Einige Formulierungen des *schwachen* Gesetzes der großen Zahlen fordern, dass Mittelwert und/oder Varianz begrenzt sind. Dies ist für Fehlerwahrscheinlichkeitsschätzungen gegeben, da $\mu_w \leq 1$ und $\sigma_w^2 \leq 1$ sind. Diese Lockerung ermöglicht auch die Anwendung der vorangegangenen Argumentation auf Bitfehlerwahrscheinlichkeitsschätzungen nach Abschnitt 4.3.1 mit $P_B = \mu_B = \mathrm{E}\{B_{a,k}\}$. Weiterhin ist die obige Argumentation nicht auf herkömmliche Monte-Carlo-Simulationen (harte Schätzung) beschränkt, sondern ist genauso für die weiche Schätzung nach (4.23) und (4.31) gültig, wenn die Variablen $B_{a,k}$ und W_a nicht gleich 0 oder 1 sind sondern den Fehlerwahrscheinlichkeiten der a-ten Übertragung entsprechen.

Für eine bestimmte Anzahl übertragener Codewörter A kann mit Hilfe von (4.50) gezeigt werden, dass eine Schätzung der BEP, $\hat{P}_b$ eine niedrigere obere Schranke für die Wahrscheinlichkeit eines großen Messfehlers aufweist als eine Schätzung der WEP, $\hat{P}_W$. Ein Infowort $\boldsymbol{u}_a$ besteht aus $K \geq 1$ Infobits. Wenn nun Fehlerwahrscheinlichkeiten geschätzt werden, indem fehlerhafte Übertragungen gezählt werden, so trägt die Übertragung eines Wortes nur eine Stichprobe W_a zu der Schätzung der WEP bei, jedoch K Stichproben $B_{a,k}$ zur BEP-Schätzung bei. Für $B_{a,k}, W_a \in \{0,1\}$ gelten

$$\sigma_B^2 = \mathrm{E}\left\{B_{a,k}^2\right\} - \mu_B^2 = P_B(1-P_B) \tag{4.51}$$

und

$$\sigma_W^2 = \mathrm{E}\left\{W_a^2\right\} - \mu_W^2 = P_W(1-P_W)\,. \tag{4.52}$$

Da in jedem nicht-trivialen Fall $P_B \leq P_W$ ist, folgt aus (4.51) und (4.52) $\sigma_B^2 \leq \sigma_W^2$. Wird nun die Anzahl der Stichproben für die Schätzung von $\hat{\mu}_B$ und $\hat{\mu}_W$ für A übertragende Infowörter berücksichtigt, so ergibt sich

$$\frac{\sigma_B^2}{AK\varepsilon^2} < \frac{\sigma_W^2}{A\varepsilon^2}\,. \tag{4.53}$$

Damit wird also $\mathrm{P}(|\hat{\mu}_B - \mu_B| > \varepsilon)$ durch einen kleineren Wert nach oben beschränkt als $\mathrm{P}(|\hat{\mu}_W - \mu_W| > \varepsilon)$. Das heißt, dass die maximale Wahrscheinlichkeit für einen Schätzfehler größer als ε für $\hat{P}_B$ kleiner ist als für $\hat{P}_W$. Dies gilt auch für korrelierte Zufallsvariablen. Die obige Argumentation gilt neben harten auch für weiche Schätzungen, also $\widehat{P}_B$ und $\widehat{P}_W$. Dies folgt direkt daraus, dass (4.51) und (4.52) genauso für die weiche Schätzung gelten.

In Abb. 4.27 ist der Schätzfehler der weichen und der harten BEP-Schätzung ($|\widehat{P}_B - P_B|$ und $|\hat{P}_B - P_B|$) für eine Monte-Carlo Simulation nach verschieden vielen simulierten Übertragungen A dargestellt. Die Infobitenergie beträgt $\gamma_b = 3{,}5\,\mathrm{dB}$ was in $P_B = 5{,}621 \cdot 10^{-4}$ resultiert. Es ist deutlich zu sehen, dass die weiche Messung den durchschnittlich kleineren Schätzfehler aufweist, was seine Ursache in der geringeren Varianz der weichen Schätzung hat. In Abb. 4.28 ist entsprechend der Schätzfehler der WEP-Schätzungen ($|\widehat{P}_W - P_W|$ und $|\hat{P}_W - P_W|$) über die Übertragungen aufgetragen. Hier entspricht die WEP $P_W = 0{,}044$. Wieder zeigt die weiche Schätzung einen schneller kleiner werdenden Schätzfehler als die harte Schätzung. Vergleicht man die Bereiche, in denen sich der Schätzfehler der BEP-Schätzungen bewegt, mit dem der WEP-Schätzungen, so ergibt sich aus den simulierten

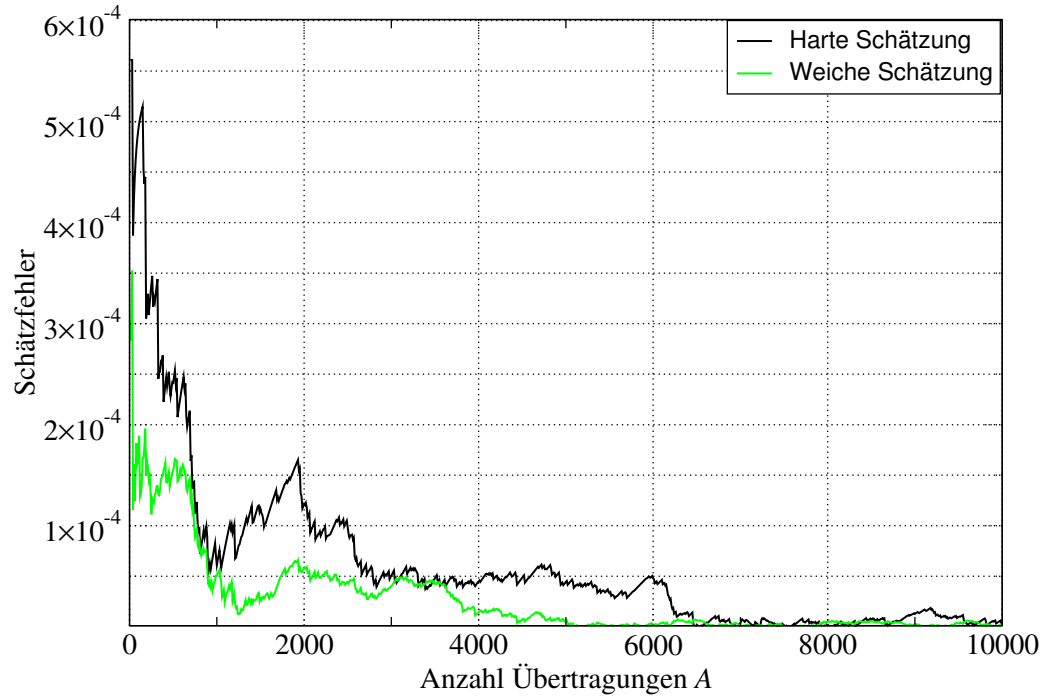

Abbildung 4.27: Schätzfehler für harte und weiche Schätzung der BEP.

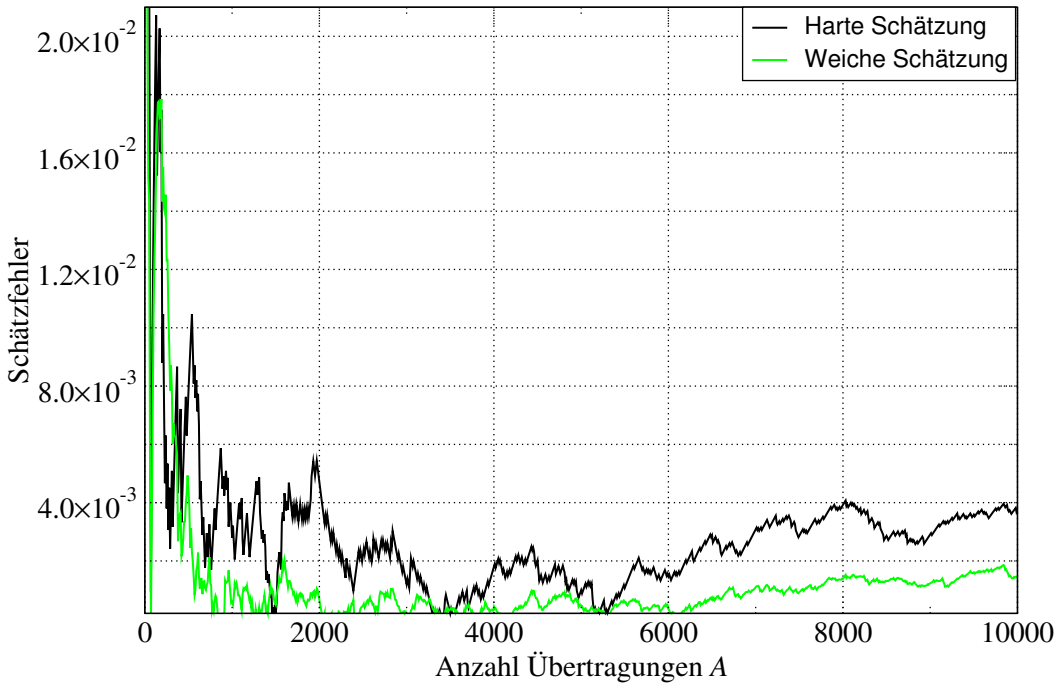

Abbildung 4.28: Schätzfehler für harte und weiche Schätzung der WEP.

Ergebnissen dieselbe Erkenntnis wie vorher aus den theoretischen Ausführungen – die BEP-Schätzungen weisen einen deutlich kleineren Schätzfehler auf.

4.6 Zusammenfassung

In diesem Kapitel wurde erläutert, wie mit Hilfe von geeigneten soft-output Decodierern die Bit- bzw. die Wortfehlerwahrscheinlichkeit eines decodierten Wortes berechnet werden kann. Die daraus abgeleitete weiche (auf Wahrscheinlichkeiten basierende) Schätzung benötigt im Gegensatz zu der harten (auf Fehlerereignissen basierenden) Schätzung keine Kenntnis der übermittelten Daten. Aus symbolweiser Zuverlässigkeitsinformation ist eine weiche Schätzung der Wortfehlerwahrscheinlichkeit im Gegensatz zu der Schätzung der Bitfehlerwahrscheinlichkeit nur approximativ möglich. Die (exakte) Schätzung der Bitfehlerwahrscheinlichkeit bietet also eine größere Flexibilität bei der Auswahl geeigneter Decodierer. Ferner wurde gezeigt, dass es trotz häufig auftretenden suboptimalen Decodierern und verwendeten Approximationen, statistischen Abhängigkeiten und Quantisierung möglich ist, gute Schätzwerte von Bit- und Wortfehlerwahrscheinlichkeit zu erlangen. Die so erlangte weiche Schätzung ist besser als eine harte Schätzung, wobei auf Grund der Statistik die Schätzung der Bitfehlerwahrscheinlichkeit üblicherweise besser als die der Wortfehlerwahrscheinlichkeit ist. Diese Schätzwerte sind ausgezeichnet dafür geeignet die Übertragungsqualität der physikalischen Schicht zu ermitteln.

Kapitel 5

Zuverlässigkeitsbasierte Neuanforderungskriterien für ARQ-Protokolle

Für die Verwendung von ARQ-Protokollen (siehe Abschnitt 3.5) ist es notwendig ein Kriterium für eine Neuanforderung (engl. repeat request) zu definieren. Dies geschieht klassischerweise durch Verwendung fehlererkennender Codes, z. B. CRC-Codes (siehe Abschnitt 3.4). Werden ARQ-Protokolle auf der Sicherungsschicht eingesetzt, so werden die Daten der Sicherungsschicht (2-SDU) üblicherweise mit einem fehlererkennenden Code codiert. Dieses Codewort stellt (mit eventueller zusätzlicher Protokollinformation) die SDU der physikalischen Schicht dar. Im (in der digitalen drahtlosen Nachrichtenübertragung fast ausschließlich vorkommenden) Fall hybrider ARQ-Protokolle findet auf der physikalischen Schicht eine Codierung mit einem FEC-Code statt. Daher wird dieser Ansatz auch als Zwei-Code-System (engl. two-code system) bezeichnet, um eine Unterscheidung von den Fällen zu ermöglichen in denen kein fehlererkennender, sondern nur ein FEC-Code verwendet wird [RW93], [Wic95].[1] Dabei gab es immer wieder Ansätze auf den fehlererkennenden Code zu verzichten und stattdessen das Kriterium für eine Neuanforderung direkt vom FEC-Code bzw. dessen Decodierer abhängig zu machen. Bei LDPC-Codes kann z. B. relativ einfach überprüft werden, ob das Ergebnis der Decodierung ein gültiges Codewort darstellt (siehe Abschnitt 3.3). Vor allem mit der zunehmenden Bedeutung von soft-output Decodierung durch die Entwicklung der Turbo-Codes wurde vielfach versucht, aus der Zuverlässigkeitsinformation, die ein soft-output Decodierer liefert, auf die Zuverlässigkeit eines übertragenden Codeworts zu schließen.

Da ein schichtweise aufgebautes wie in Abschnitt 2.1.2 erläutertes System nur die Nachricht bzw. die der SDU zugehörigen Daten an die oberen Schichten weitergibt, ist eine Verwendung der auf der physikalischen Schicht erzeugten Zuverlässigkeitsinformation einzelner Symbole in der Sicherungsschicht ein schichtübergreifender Ansatz nach Abschnitt 2.1.3. Je nach Realisierung stellt dieser zusätzliche Informationsaustausch einen

[1] Dieser Begriff ist insofern irreführend, als dass der Begriff Zwei-Code-System sich auch auf Systeme beziehen kann, die auf der physikalischen Schicht zwei (z. B. verkettete Codes verwendet werden), also eigentlich ein Drei-Code-System wären.

Ansatz nach A oder C (siehe Abb. 2.4) dar.

Traditionell ist man bei ARQ-Verfahren an Wortfehlern und daher auch Wortfehlerwahrscheinlichkeiten interessiert. Dies liegt an dem klassischen Zuverlässigkeitskriterium, dem CRC-Code. Dieser ist im Allgemeinen nicht in der Lage, zu erkennen wie viele Fehler ein Wort enthält, sondern nur, ob ein Wort Fehler enthält. Mit der Verwendung von Fehlerwahrscheinlichkeiten bzw. Zuverlässigkeiten ist es allerdings möglich, eine durchschnittliche Anzahl Fehler kontrolliert zuzulassen. Es wird also keine fehlerfreie Übertragung, sondern eine Übertragung mit einer maximal zulässigen Anzahl Fehlern angestrebt. In dieser Hinsicht sind Wort- und Bitfehlerwahrscheinlichkeit gleichwertige Kriterien, da für viele Anwendungen die Anzahl fehlerhafter Bits entscheidender ist als die der fehlerhaft übertragenden Worte (auf der physikalischen Schicht).

Die Vorschläge, wie die Zuverlässigkeitsinformation des Decodierers genutzt werden oder die Wortfehlerwahrscheinlichkeit aus symbolweiser Zuverlässigkeitsinformation ermittelt werden kann, sind vielfältig. Diese sollen im kommenden Abschnitt kurz systematisiert werden, ehe verschiedene Kriterien für Neuanforderungen verglichen werden.

5.1 Unterschiedliche Neuanforderungskriterien

Für die Verwendung von ARQ-Protokollen sind, je nach der Art der Zuverlässigkeitsinformation für die decodierten Daten bzw. wie diese gewonnen wird, unterschiedliche Systemvoraussetzungen nötig. Diese werden im Folgenden durch Erweiterungen des Systemmodells in Abb. 2.5 eingeführt.

In Abb. 5.1 ist ein Übertragungssystem mit ARQ-Protokoll unter Verwendung eines fehlererkennenden Codes (hier CRC-Code) angegeben. Das Infowort $\boldsymbol{u}$ wird zur Fehlererken-

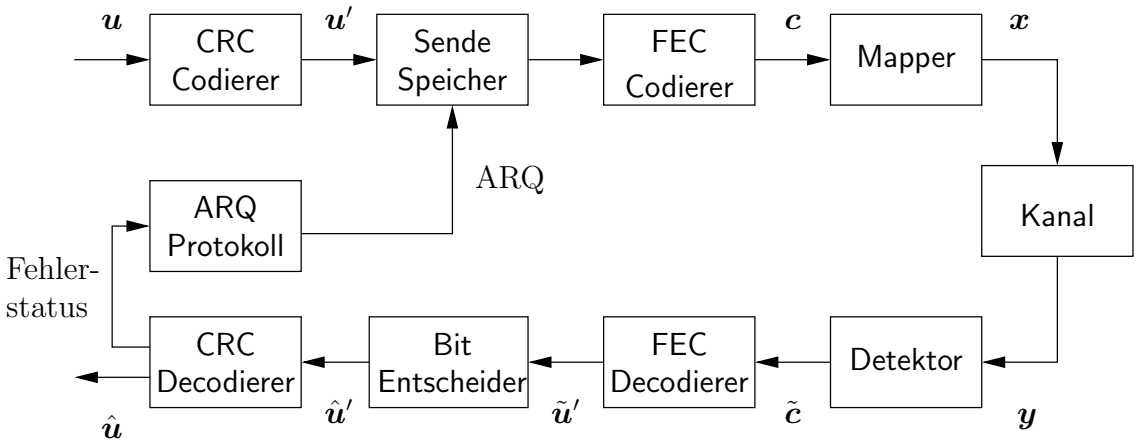

Abbildung 5.1: Übertragungssystem mit CRC-Code nutzendem ARQ-Protokoll.

nung mit einem CRC-Code zu $\boldsymbol{u}'$ codiert bevor es zum Schutz vor Übertragungsfehlern mit einem FEC-Code codiert wird (serielle Codeverkettung). Da der CRC-Codierer das Infowort $\boldsymbol{u}$ um P Prüfbits verlängert, gilt $K' = K + P$. Der CRC-Decodierer decodiert den harten Schätzwert des FEC-Infoworts $\hat{\boldsymbol{u}}'$ und meldet dem ARQ-Protokoll den Fehler-

status (Fehler/kein Fehler) des geschätzten Infoworts. Damit liefert der CRC-Decodierer einen *harten Schätzwert* des Fehlerzustands.

Dagegen basiert ein mit Zuverlässigkeitsinformation arbeitendes Kriterium auf reellen Zahlen, die die Zuverlässigkeit der übertragenden Bits bzw. des übertragenden Wortes repräsentieren. Je nachdem ob die Dienstgüte einen Anspruch an BEP oder WEP stellt, sollte das Neuanforderungskriterium die Bit- oder die Wortfehlerwahrscheinlichkeit sein. Eine Neuanforderungsanfrage wird gesendet, wenn das auf der Zuverlässigkeit basierende Kriterium nicht erfüllt ist, also z. B. das Zuverlässigkeitsmaß unter einer vorher festgelegten Grenze liegt. Diese Methode hat zwei Vorteile: Erstens wird kein (zusätzlicher) fehlererkennender Code benötigt und zweitens ist das Verfahren flexibler als eine einfache richtig oder falsch Entscheidung – es kann einfach an die benötigte Dienstgüte angepasst werden.

Neben den beiden möglichen QoS-Parametern bestehen wenigstens zwei wesentliche Möglichkeiten die Zuverlässigkeitsinformation des decodierten Infoworts $\hat{\boldsymbol{u}}$ zu erlangen. Die erste Möglichkeit stellen Decodierer dar, die – zumeist als Sequenzschätzer – die Zuverlässigkeit des geschätzten Infoworts aus ihrer Natur heraus in dem Decodierungsprozess ermitteln (Abb. 5.2). Zu diesen Decodierern gehört z. B. der in [YI80] veröffentlichte Yamamatoh-Itoh-Algorithmus (YIA). Der YIA ist ein VA der einen überlebenden Pfad als unzuverlässig kennzeichnet, wenn die Differenz zum nicht überlebenden Pfad kleiner als ein bestimmter Wert ist. Korrespondiert das geschätzte Wort mit einem unzuverlässigen Pfad, so wird eine Neuanforderungsanfrage gesendet. Aus den Zuverlässigkeitsdifferenzen können obere Schranken für die Wortfehlerwahrscheinlichkeit berechnet werden [RW94]. Einen anderen Ansatz verfolgt ein System wie in Abb. 5.3 dargestellt. In diesem

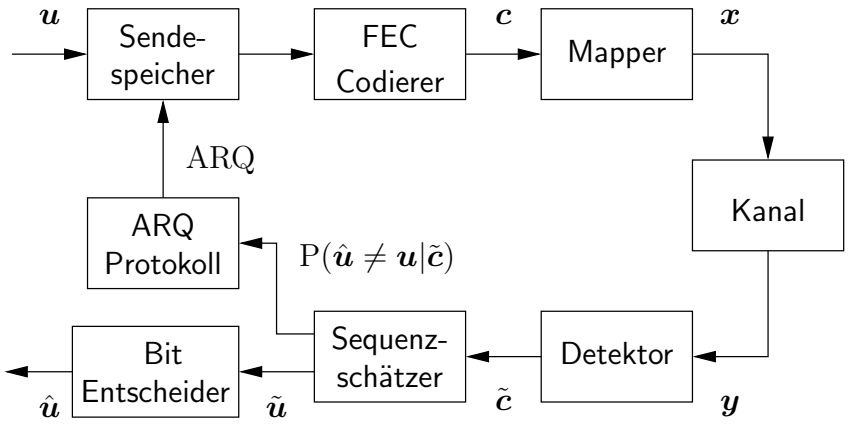

Abbildung 5.2: Übertragungssystem mit einem $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ liefernden Decodierer.

Modell liefert ein Decodierer symbolweise Zuverlässigkeiten. Dabei kann es sich durchaus auch um einen symbolweise Zuverlässigkeitsinformation liefernden Sequenzschätzer wie den SOVA handeln. Wie in Kapitel 4 erläutert, können aus der symbolweisen Zuverlässigkeitsinformation die Fehlerwahrscheinlichkeit eines einzelnen Bits (4.18) und daraus weiter die durchschnittliche Bitfehlerwahrscheinlichkeit eines decodierten Wortes P_b (4.19) bzw. die Wortfehlerwahrscheinlichkeit $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ (4.38) berechnet werden. Diese weiteren Berechnungen werden in Abb. 5.3 in dem Evaluierungsmodul vorgenommen, welches sein

Ergebnis an das ARQ-Protokoll für die Entscheidung einer Neuanforderung weitergibt. Neben den in Kapitel 4 vorgestellten Möglichkeiten wurden im Zusammenhang mit ARQ

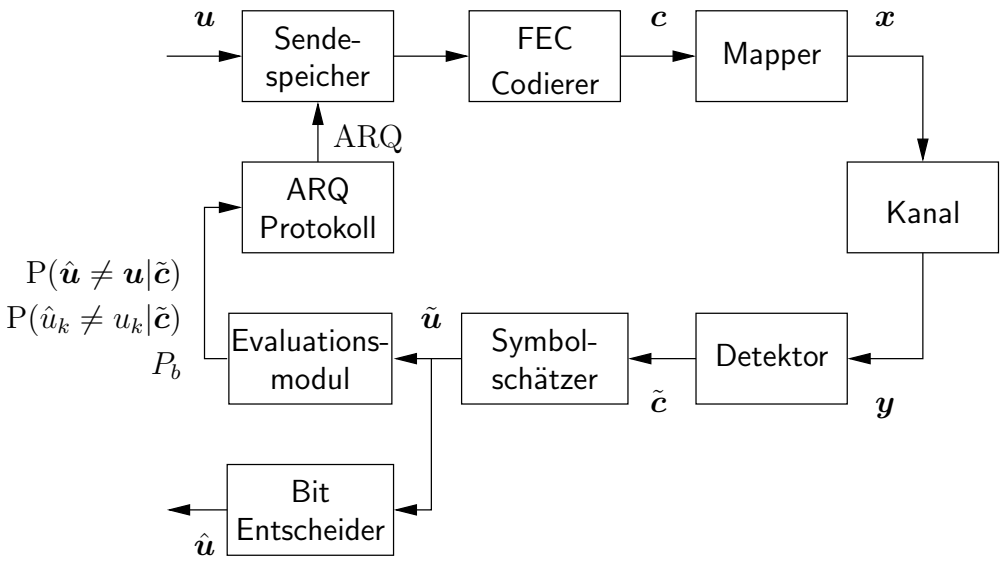

Abbildung 5.3: Übertragungssystem mit symbolweisen Decodierer. Ein Zuverlässigkeitsevaluierungsmodul kann aus $\tilde{u}_k$ verschiedene Maße ermitteln: $\mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})$, P_b oder $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$.

diverse andere Vorschläge zur Ermittlung der Zuverlässigkeit eines decodierten Wortes anhand der symbolweisen Zuverlässigkeitsinformationen gemacht. In [She02] und [RS03] werden die Beträge der LLRs genutzt, um zu entscheiden, welche Teile eines Codeworts neu angefordert werden. Dafür ist hier besonders sinnvoll, dass auch $\mathrm{P}(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}})$ aus der symbolweisen Zuverlässigkeitsinformation berechenbar ist, also die unzuverlässigen Bits eines Worts direkt erkennbar sind. Die durchschnittlichen Beträge der LLRs werden in [CTWH00], [TVPH03] und [VST+05] als Maß für die Zuverlässigkeit eines Wortes genutzt. Schwellenwerte, die eine Neuanforderung auslösen, werden dabei anhand von Simulationen gefunden. In [VST+05] wird dafür eine Abbildung vorgestellt, um die durchschnittlichen Beträge der LLRs auf die WEP abzubilden. Auch die in Kapitel 4 vorgestellten Ansätze P_b und $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ aus $\tilde{\boldsymbol{c}}$ zu berechnen ([FSH06], [BFH06] und [FH08a]) wurden ursprünglich als Zuverlässigkeitsmaß für die Entscheidung für Neuanforderungen entwickelt.

Im Gegensatz zu den anderen oben zitierten Methoden sind die in Kapitel 4 vorgestellten Methoden analytisch begründet und benötigen keine simulativ ermittelten Zusammenhänge, Abbildungen oder Schranken, um auf die jeweils genutzte Fehlerwahrscheinlichkeit zu schließen. Da die ohne Schranken und zusätzliche Abbildungen ermittelten Fehlerwahrscheinlichkeiten keine schlechteren Schätzungen darstellen können, werden im Weiteren dieses Kapitels nur diese verwendet. Der Ansatz in Abb 5.2 fordert spezielle Decodierer (APP-Sequenzschätzer) damit die exakte Wahrscheinlichkeit $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ berechnet werden kann. Der Ansatz in Abb. 5.3 ist flexibler, ermöglicht aber nur die approximative Berechnung der Wahrscheinlichkeit $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$. Ferner ermöglicht ein Decodierer mit symbolweisen weichen Ausgangswerten eine einfache Methode des Diversity-Combining

für Typ-II und III ARQ-Protokolle ([BFH06] bzw. Abschnitt 5.3). Als weiterer Vorteil ermöglicht das Vorhandensein von symbolweiser Zuverlässigkeitsinformation die unzuverlässigen Bits eines Wortes zu identifizieren und gezielt für diese Bits inkrementelle Redundanz anzufordern.

5.1.1 CRC-Code als Neuanforderungskriterium

Der wesentliche Vorteil der Verwendung eines CRC-Codes zur Fehlererkennung bzw. zur Erlangung von Fehlerinformation eines decodierten Wortes nach Abb. 5.1 ist die Einfachheit des Verfahrens. Die Fehlererkennungsleistung von CRC-Codes ist erstaunlich gut und hängt bei geeigneten Generatorpolynomen im Wesentlichen von der Anzahl der Prüfbits ab.[2] Dabei können CRC-Decodierer sehr effizient implementiert werden (siehe Abschnitt 3.4.2). soft-output Decodierung des FEC-Decodierers ist im Gegensatz zu den anderen Verfahren nicht notwendig, das Verfahren kommt also mit kostengünstigen FEC-Decodierern aus.

Der wesentliche Nachteil ist die Verlängerung des ursprünglichen Infoworts $\boldsymbol{u}$ bei der Codierung in ein CRC-Codewort um P Prüfbits. Damit ist die Länge des Wortes $\boldsymbol{u}'$ $K' = K + P$. Die Rate des CRC-Codes ist also

$$R_{\mathrm{CRC}} = \frac{K}{K+P} \,. \tag{5.1}$$

Durch die anschließende Codierung mit einem FEC-Code wird weitere Redundanz hinzugefügt. Das Codewort des FEC-Codierers hat dann die Länge $N = K'/R = (K+P)/R$. Durch diese serielle Codeverkettung ergibt sich die Gesamtrate

$$R = R_{\mathrm{CRC}} R_{\mathrm{FEC}} = \frac{K}{(K+P)N} \,. \tag{5.2}$$

Insbesondere für kurze Infoworte reduziert sich dadurch die Durchsatzeffizienz (3.19). Die Neuanforderungswahrscheinlichkeit ist $P_r = P_W P_d$, wobei P_d die Wahrscheinlichkeit eines erkennbaren Fehlers ist. Wird der CRC-Code ausreichend stark gewählt, so gilt in guter Näherung $P_r \approx P_W$. Somit nimmt die Durchsatzeffizienz aus Abschnitt 3.5 für die Verwendung eines CRC-Codes die Form

$$\eta_{\mathrm{CRC}} = \frac{K}{(K+P)N + \frac{T_{RT}}{T_s}} (1 - P_W) \tag{5.3}$$

an. Aus (5.3) geht hervor, dass – abgesehen von der Wortfehlerwahrscheinlichkeit P_W – die Durchsatzeffizienz wesentlich durch das Verhältnis $K/((K+P)N)$ bestimmt wird.

5.1.2 Wortfehlerwahrscheinlichkeit als Neuanforderungskriterium

Ursprünglich stammt die Idee, die Zuverlässigkeit eines Wortes als Neuanforderungskriterium zu verwenden, aus [YI80]. Diese Zuverlässigkeit sollte idealerweise $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u} | \tilde{\boldsymbol{c}})$, also

[2] Die Leistung von CRC-Codes bei der Fehlererkennung ist, insbesondere bezogen auf die geringe Anzahl Codebits, ausgesprochen gut [Fri96].

die Wahrscheinlichkeit dafür sein, dass der Infowortschätzwert nicht gleich dem gesendeten Infowort ist. Die in Kapitel 4 vorgestellten Decodierer stellen diese Wahrscheinlichkeit bereit bzw. machen es möglich, sie zu berechnen.

Für ein auf Zuverlässigkeitsinformation basierendes Neuanforderungskriterium wird entsprechend der benötigten Dienstgüte ein Schwellenwert $P_{W,t}$ festgelegt. Für eine Fehlerwahrscheinlichkeit

$$\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}}) > P_{W,t} \tag{5.4}$$

wird der Infowortschätzwert als zu unzuverlässig für die definierte Dienstgüte angenommen und eine Neuanforderungsanfrage gesendet. Wenn also $\mathrm{P}(\hat{\boldsymbol{u}} \neq \boldsymbol{u}|\tilde{\boldsymbol{c}})$ nicht approximiert ist, dann ist die resultierende Wortfehlerrate entsprechend kleiner oder gleich $P_{W,t}$. Dies zeigen auch die Ergebnisse der simulierten Wortfehlerwahrscheinlichkeit für verschiedene Schwellenwerte $P_{W,t}$ in Abb. 5.4. Vor Beginn der Kurven in Abb. 5.4 wurden bei der simulierten Anzahl übertragener Pakete keine Wörter mit einer Fehlerwahrscheinlichkeit kleiner oder gleich $P_{W,t}$ decodiert.

Ein entscheidender Nachteil der Wortfehlerwahrscheinlichkeit als Zuverlässigkeitskriterium ist die Tatsache, dass ein Infowort mit hoher Wortfehlerwahrscheinlichkeit nicht zwangsläufig falsch sein muss. Die Wahrscheinlichkeit dafür ist $1 - P_W$, d. h., je niedriger $P_{W,t}$, desto mehr richtige Wörter werden als zu unzuverlässig betrachtet. Die Wahrscheinlichkeit einer Neuanforderung ist daher $P_r = \mathrm{P}(\ \mathrm{P}(\hat{\boldsymbol{u}} = \boldsymbol{u}|\tilde{\boldsymbol{c}}) > P_{W,t})$ - dieser Ausdruck ist nicht analytisch anzugeben, seine Auswirkung aber deutlich an den Monte-Carlo-Simulationen der Durchsatzeffizienz im folgenden Kapitel zu erkennen.

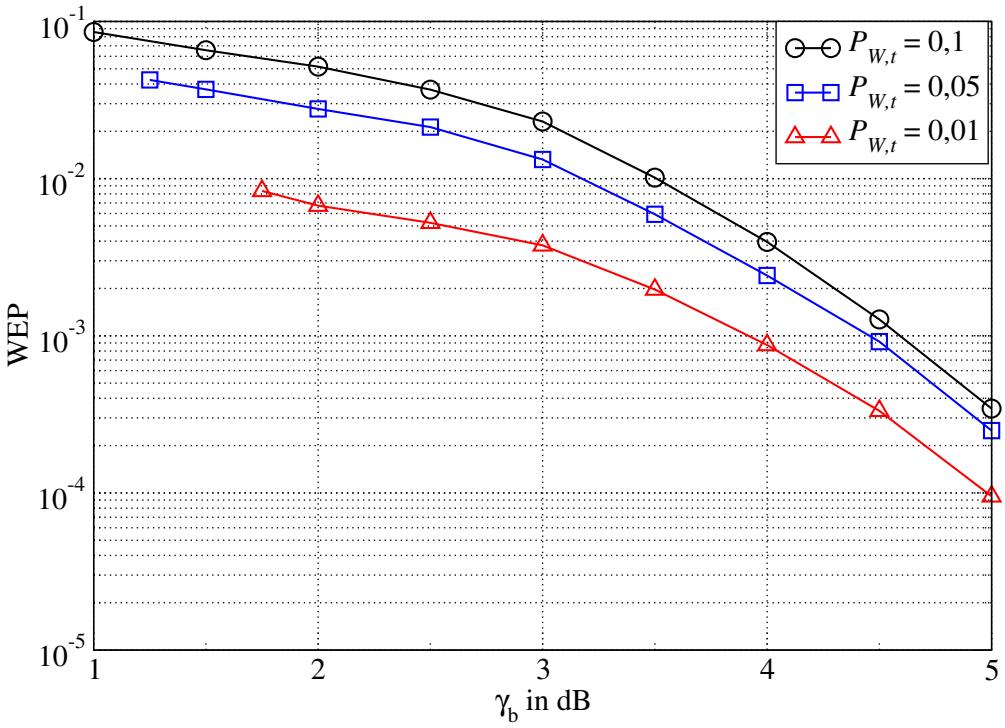

Abbildung 5.4: Verbleibende WEP bei Verwendung der Wortfehlerwahrscheinlichkeit als Neuanforderungskriterium.

5.1.3 Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium

Auch wenn das Interesse im Zusammenhang mit ARQ traditionell in Richtung Wortfehler(-wahrscheinlichkeit) geht, soll hier die Bitfehlerwahrscheinlichkeit eines decodierten Wortes als Zuverlässigkeitskriterium betrachtet werden. Das ursprüngliche Argument, Wortfehler seien auf Grund der Fehlererkennung das wesentliche Maß, ist insofern nicht mehr gültig, als dass bei der Verwendung der Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium eben auch diese relevant für den Durchsatz ist. Dafür wird die durchschnittliche Bitfehlerwahrscheinlichkeit innerhalb eines decodierten Infoworts P_b aus den vom Decodierer bereitgestellten LLRs $\tilde{u}_k$, wie in Abb. 5.3 dargestellt, berechnet. Analog zur Wortfehlerwahrscheinlichkeit wird ein Schwellenwert $P_{b,t}$ entsprechend der geforderten Dienstgüte angegeben. Eine Neuanforderung wird für

$$P_b > P_{b,t} \tag{5.5}$$

gesendet, also für jedes Infowort, dessen Bitfehlerwahrscheinlichkeit größer als die geforderte ist. Abb. 5.5 zeigt, dass der Mittelwert der BEP in den angenommenen Paketen unterhalb der gegebenen maximalen Bitfehlerwahrscheinlichkeit $P_{b,t}$ liegt. Vor Beginn der Kurven konnten in der Simulation keine Pakte erfolgreich übermittelt werden, da keines der decodierten Pakete die Anforderungen an die Bitfehlerwahrscheinlichkeit erfüllte. Wie

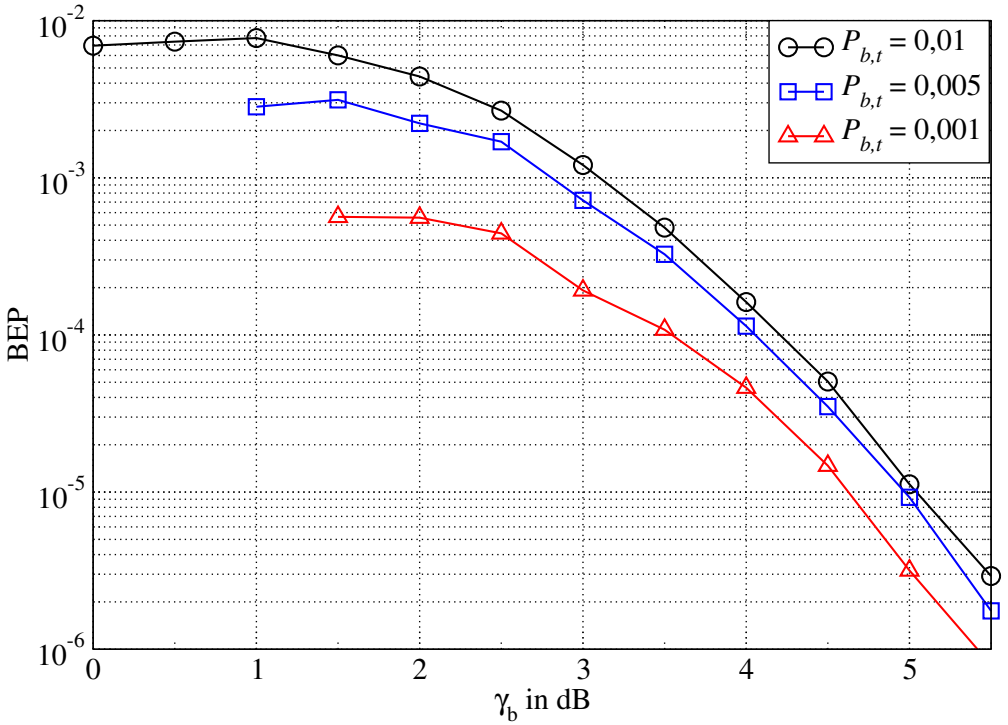

Abbildung 5.5: Verbleibende BEP bei Verwendung der Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium.

bei der Wortfehlerwahrscheinlichkeit leidet dieses Kriterium darunter, dass Bits mit hoher Fehlerwahrscheinlichkeit richtig sein können, jedoch P_b erhöhen. Dieser Effekt wird aber

dadurch kompensiert, dass P_b einen Mittelwert darstellt und (durchschnittlich) ebenso viele falsche Bits mit niedriger Fehlerwahrscheinlichkeit enthält. Daher ergibt sich als resultierende durchschnittliche Bitfehlerrate der erwartete Wert und es werden weniger Worte als bei Verwendung der Wortfehlerwahrscheinlichkeit unbegründet abgelehnt.

5.2 Durchsatzeffizienz für verschiedene Neuanforderungskriterien

Nun soll untersucht werden, welcher messbare Vorteil sich durch die Verwendung zuverlässigkeitsbasierter Neuanforderungskriterien ergibt. Grundsätzlich soll sich die Durchsatzeffizienz erhöhen, wenn in einer Datenübertragung (mehr) Fehler zugelassen werden. Hierfür werden die drei eingeführten Systeme (Abb. 5.1–5.3) verglichen. Der verwendete CRC-Code ist der in Abschnitt 3.4 eingeführte 16-Bit-Code. Dabei ist anzumerken, dass 16 Bit nicht die höchste verwendete Zahl von Prüfbits für blockweise Übertragung in Mobilfunksystemen ist, jedoch in den hier simulierten Ergebnissen alle auftretenden Fehler entdeckt wurden. Im Gegensatz zu den beiden auf Zuverlässigkeitsinformation basierenden Systemen resultiert das Übertragungssystem mit dem CRC-Code also in einer Bit-/Wortfehlerrate von 0. Für die Simulation der Wort- und Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium werden das System in Abb. 5.2 mit einem ROVA als Decodierer und das System in Abb. 5.3 mit einem BCJR-Decodierer verwendet. Die Schätzungen von Wort- und Bitfehlerwahrscheinlichkeit sind also optimal (siehe Abschnitt 4), die durchschnittliche Fehlerrate wird durch die Schwellenwerte $P_{W,t}$ und $P_{b,t}$ begrenzt, liegt also auf jeden Fall unter diesen Schwellenwerten (Abschnitte 5.1.2 und 5.1.3).

Für die zuverlässigkeitsbasierenden ARQ-Protokolle ist die maximale Durchsatzeffizienz (bei dem hier verwendeten terminierten $(23_8, 35_8)$ Faltungscode)

$$\eta_{\max,\mathrm{RB}} = \frac{K}{N} = \frac{288}{(288+4)2} \approx 0,493 \; . \tag{5.6}$$

Die maximal mögliche Durchsatzeffizienz wird erreicht, wenn die Wahrscheinlichkeit für Neuanforderungen P_r Null wird bzw. gegen Null geht (3.19) und entspricht dann der Coderate R. Für das System mit dem CRC-Code ist die maximale Durchsatzeffizienz unter Berücksichtigung der Prüfbits des CRC-Codes

$$\eta_{\max,\mathrm{CRC}} = \frac{K}{N} = \frac{288}{(288+16+4)2} \approx 0,468 \; . \tag{5.7}$$

Die Einsparung der Prüfbits ermöglicht den mit Zuverlässigkeitsinformationen arbeitenden Protokollen also theoretisch einen etwas höheren Durchsatz.

Abbildung 5.6 vergleicht den erreichten Durchsatz bei Verwendung unterschiedlicher Schwellenwerte $P_{W,t}$ und des CRC-Codes. Es ist zu beachten, dass alle verwendeten Neuanforderungskriterien ganze Infowörter – nicht nur Teile oder einzelne Bits – verwerfen. Darin zeigt sich, dass die Verwendung der Wortfehlerwahrscheinlichkeit nur für hohe γ_b ab 3,5 dB einen höheren Durchsatz als der Zwei-Code-Ansatz erreicht, dadurch, dass der maximal mögliche Durchsatz auf Grund der fehlenden CRC-Bits größer ist. Das

eigentlich erwartete Ergebnis – eine Durchsatzsteigerung durch Zulassen von Wortfehlern – tritt nicht ein. Dies liegt daran, dass neben den akzeptierten Paketen mit Fehlern zu viele fehlerfreie Pakete abgelehnt werden, da ihre Fehlerwahrscheinlichkeit trotz Fehlerfreiheit zu groß ist. Dieser Effekt wurde auch für den YIA festgestellt [RW93]. Auf Grund der

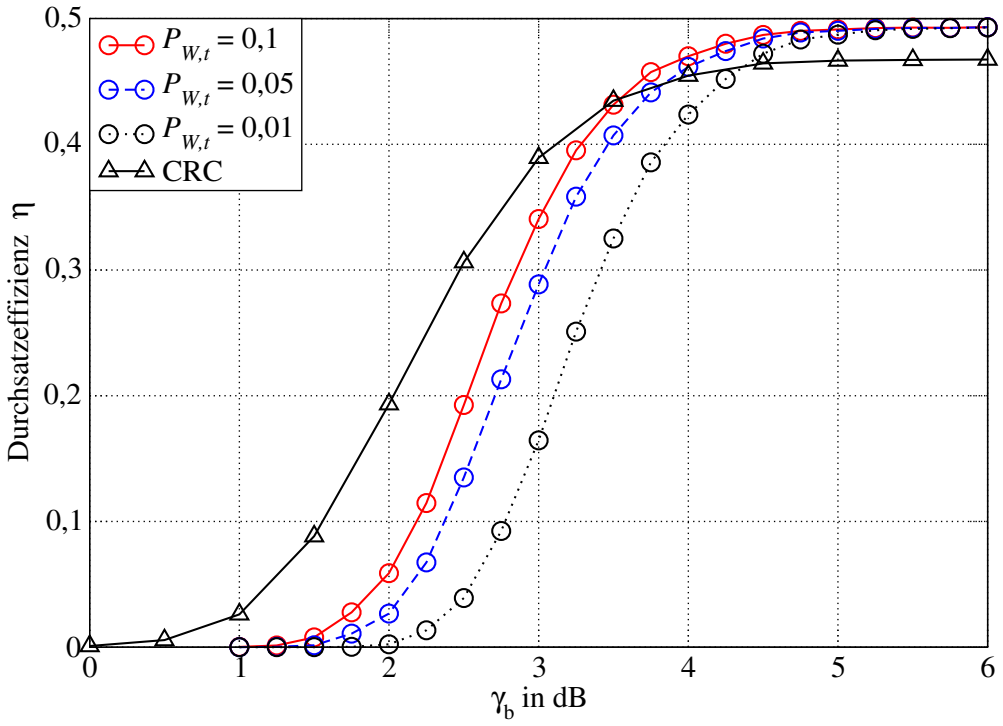

Abbildung 5.6: Durchsatzeffizienz für verschiedene Schwellenwerte bei Verwendung der Wortfehlerwahrscheinlichkeit als Neuanforderungskriterium.

in Abschnitt 5.1.3 beschriebenen Gründe sind bei Verwendung der Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium die in Abb. 5.7 gezeigten Durchsatzsteigerungen möglich. Für höhere $P_{b,t}$ gewinnt das zuverlässigkeitsbasierte Kriterium gegenüber dem CRC-Code.

Für sehr kurze Infowortlängen verschiebt sich das Bild (Abb. 5.8) zu Gunsten der zuverlässigkeitsbasierten Kriterien, beide haben einen ähnlichen Verlauf. Das Verfahren mit CRC-Code leidet bei kurzen Infoworten unter den Durchsatz begrenzenden zusätzlichen CRC-Bits. Für lange Infoworte fallen zwar die zusätzlichen Bits nicht ins Gewicht, allerdings ist die Wahrscheinlichkeit, dass ein langes Wort Fehler enthält, größer. Dadurch wird auch der steile Anstieg der Durchsatzeffizienz für $K = 2880$ erklärt: Tritt noch ein einziger Bitfehler auf, so wird das Paket abgelehnt, bei einem leicht erhöhten γ_b sind dann die meisten Pakete fehlerfrei. Für große Infowortlängen gewinnt daher vor allem die Bitfehlerwahrscheinlichkeit als Zuverlässigkeitskriterium, da bei $K = 2880$ und $P_{b,t} = 0{,}01$ durchschnittlich 29 Bitfehler pro Infowort zugelassen werden. Die Bitfehlerwahrscheinlichkeit ist also eher als die Wortfehlerwahrscheinlichkeit dazu geeignet, durch kontrolliertes Zulassen von Fehlern den Durchsatz zu steigern.

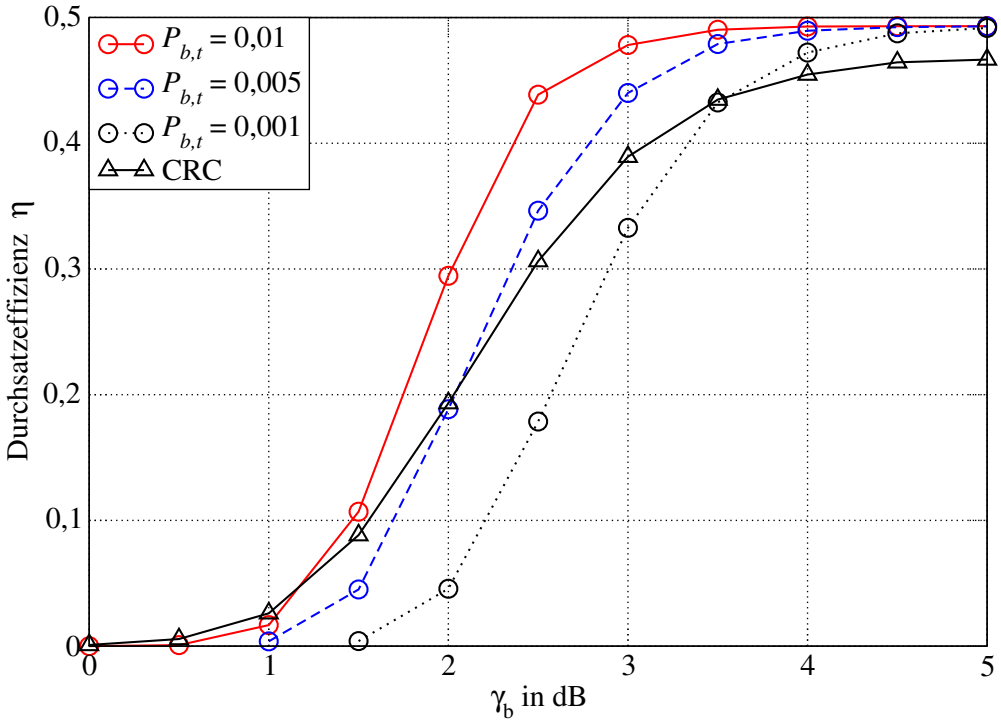

Abbildung 5.7: Durchsatzeffizienz für verschiedene Schwellenwerte bei Verwendung der Bitfehlerwahrscheinlichkeit als Neuanforderungskriterium im Vergleich zum Zwei-Code-Ansatz.

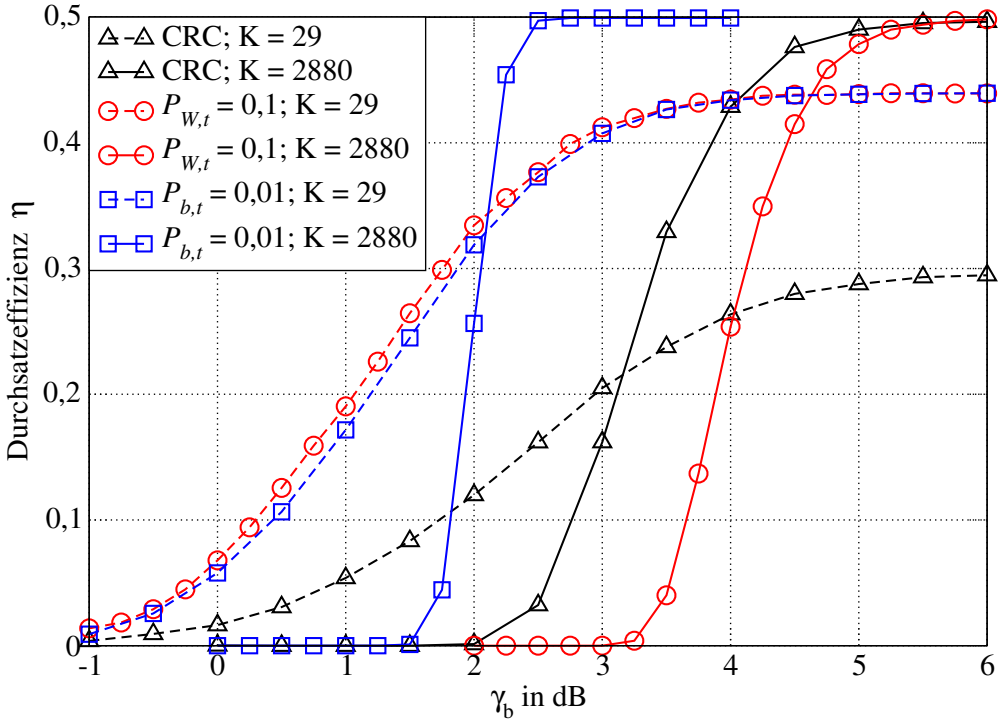

Abbildung 5.8: Auswirkung der Infowortlängen auf die Durchsatzeffizienz verschiedener Neuanforderungskriterien im Vergleich zum Zwei-Code-Ansatz.

5.3 Packet-Combining

In Abschnitt 5.1 wurde gezeigt, dass auf Zuverlässigkeitsinformation basierende Neuanforderungskriterien in einer BEP bzw. WEP resultieren, die unter der Zielfehlerwahrscheinlichkeit liegt. Hybride ARQ-Protokolle mit Packet-Combining-Mechanismen (Typ II und III, siehe Abschnitt 3.6) erzielen für Übertragungskanäle mit mittleren und hohen Fehlerwahrscheinlichkeiten bedeutend höhere Durchsätze. Um diese Verfahren auch für zuverlässigkeitsbasierende Neuanforderungskriterien verwenden zu können, ist es unerlässlich, dass die in Kapitel 4 vorgestellten Schätzmethoden für Fehlerwahrscheinlichkeit bzw. die dafür verwendeten Decodierer mit Packet-Combining-Mechanismen verwendet werden können. In Abb. 5.9 ist die verbleibende BEP bzw. WEP bei Anwendung der zuverlässigkeitsbasierenden Neuanforderungskriterien zusammen mit zwei unterschiedlichen Packet-Combining-Strategien dargestellt.

5.3.1 Diversity-Combining

Um den Einfluss von Diversity-Combining zu illustrieren wird hier die Übertragung unter Verwendung eines $(23_8, 35_8)$-Faltungscode simuliert. Liegt die Fehlerwahrscheinlichkeit des decodierten Wortes am Empfänger über dem Schwellenwert, so wird eine Neuanforderungsanfrage gesendet. Der Sender sendet daraufhin so lange dasselbe Codewort $\boldsymbol{c}$ bis der Empfänger eine positive Rückmeldung sendet. Am Empfänger werden die LLRs der Codebits der A Übertragungen $\tilde{\boldsymbol{c}}^{(1)} \dots \tilde{\boldsymbol{c}}^{(A)}$ als Summe der LLRs

$$\tilde{\boldsymbol{c}} = \sum_{a=1}^{A} \tilde{\boldsymbol{c}}^{(a)} \tag{5.8}$$

kombiniert. Damit entsteht eine Verkettung eines Wiederholungscodes mit einem Faltungscode. Die Rate des Wiederholungscodes, $R = 1/A$, sinkt dabei mit der Anzahl der Übertragungen ab, verbessert also die Fehlerkorrekturmöglichkeiten. Die nachgelagerten Decodierer (ROVA und BCJR-Algorithmus) erhalten als weiche Eingangswerte LLRs $\tilde{\boldsymbol{x}}$. Da die Summenbildung über die LLRs in (5.8) für unterschiedliche, statistisch unabhängige Übertragungen für einen Wiederholungscode den optimalen Decodierer darstellt, sind diese auch für die Fehlerschätzung nach Kapitel 4 geeignet. Die Fehlerwahrscheinlichkeit in Abb. 5.9 weist eine leichte Wellenform mit Maxima bei $\gamma_b = -4\,\mathrm{dB}$, $-1\,\mathrm{dB}$ und $2\,\mathrm{dB}$ auf. Diese stellen keine Simulationsungenauigkeiten dar, sondern resultieren daraus, dass an diesen Punkten durchschnittlich weniger Übertragungen als bei geringerem γ_b genügen, um die vom Schwellenwert geforderte Fehlerwahrscheinlichkeit zu erfüllen. Dies führt zu weniger Übertragungen (höherer Datenrate), die Fehlerwahrscheinlichkeit in den akzeptierten Nachrichten steigt aber an. Mit steigendem γ_b sinkt die Fehlerwahrscheinlichkeit wieder ab, bis sich die Anzahl notwendiger Übertragungen zum Erreichen des Schwellenwerts wieder reduziert, also die Fehlerrate in den angenommenen Paketen wieder ansteigt, da für die Decodierung weniger Redundanz genutzt wird. Die verbleibende Fehlerrate ist dabei immer deutlich unter dem Schwellenwert. Dies erklärt sich daraus, dass die inkrementelle Redundanz in den beiden hier verwendeten Combining-Mechanismen immer die volle Codewortlänge der ersten Übertragung (584 Bits) beträgt,

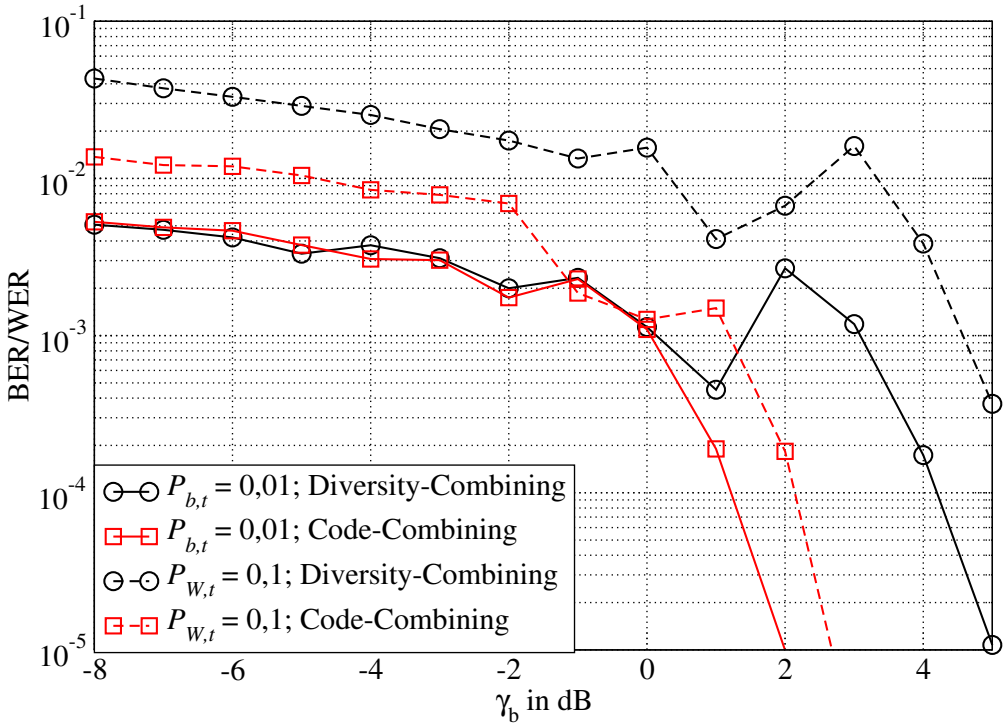

Abbildung 5.9: Verbleibende BER und WER für zuverlässigkeitsbasierende Neuanforderungskriterien und Packet-Combining-Mechanismen.

und die Fehlerwahrscheinlichkeit der kombinierten Wörter daher im Durchschnitt weit unter dem Schwellenwert liegt.
Natürlich ist es möglich, die inkrementelle Redundanz (IR) in kleineren Einheiten zu versenden. Für die hier verwendeten Kriterien eignet sich z. B. gut ein Schema wie in [She02], das kleine Mengen IR speziell für die unzuverlässigen Infobits anfordert. Zu beachten ist allerdings, dass bei der gleichen Fehlerwahrscheinlichkeit und weniger IR pro weiterer Übertragung zwar die Anzahl übertragener Bits pro Kanalbenutzung steigt, da keine unnötige Redundanz übertragen wird, andererseits aber die Anzahl der benötigten Übertragungen größer wird. Dadurch entstehen größere Verzögerungen, die bei Berücksichtigung der RTT zu einer Reduzierung des Durchsatzes führen.

Da für niedrige γ_b, also dem für Packet-Combining-Systeme besonders interessanten Bereich, viele Fehler auftreten, ist auch hier eine Durchsatzsteigerung für zuverlässigkeitsbasierte Neuanforderungskriterien erzielbar. Dies ist in Abb. 5.10 dargestellt.

5.3.2 Code-Combining

Um Code-Combining zu nutzen wird das Schema MCS-4 aus [ETSb] mit einer Infowortlänge von $K = 288$ und den Generatorpolynomen $(25_8, 33_8, 37_8)$ verwendet. Es werden in jeder Übertragung nur 1/3 (also 292) der Codebits übertragen. Bei einer Neuanforderung wird das zweite Drittel der Codebits, bei einer weiteren Neuanforderung das verbliebene Drittel übertragen. Der Decodierer decodiert immer das ganze Codewort und füllt die nicht übertragenden Bits mit LLRs der Wertigkeit Null (vollkommene Unsicherheit über

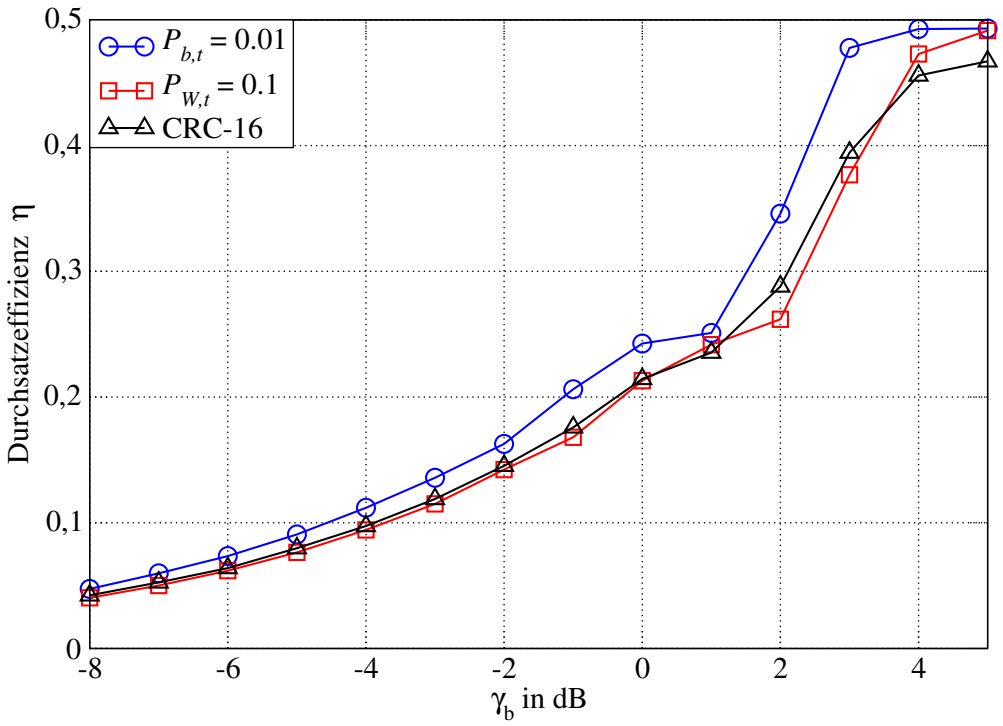

Abbildung 5.10: Durchsatzeffizienz für zuverlässigkeitsbasierende Neuanforderungskriterien zusammen mit Packet-Combining.

das Bit) auf. Treten nach den ersten drei Übertragungen weitere Neuanforderungen auf, so werden die drei Gruppen von Codebits in derselben Reihenfolge erneut übertragen und entsprechend (5.8) kombiniert. Sind die Eingangswerte für den Decodierer mit $\tilde{\boldsymbol{c}}^{(0)} = \mathbf{0}$ initialisiert, so lässt sich Code-Combining für diesen Fall (punktierte Faltungscodes) für $a = 1 \ldots A$ Übertragungen als

$$\tilde{c}_n = \sum_{\substack{a=1 \\ n \mod a=0}}^{A} \tilde{c}_n^{(a)} \quad \text{für } n = 1 \ldots N \tag{5.9}$$

implementieren. Für weniger als drei Übertragungen sind also 1/3 bzw. 2/3 der an den Decodierer übergebenen LLRs $\tilde{c}_n$ Null – über diese Bits herrscht totale Unsicherheit, da sie nicht übertragen wurden.

Code-Combining ist stärker als Diversity-Combining, d. h., für die gleiche Anzahl Übertragungen wird üblicherweise eine geringere Fehlerwahrscheinlichkeit erzielt. Die zeigt sich auch in den Ergebnissen in Abb. 5.9, speziell für $\gamma_b > 1$ dB wird der Unterschied deutlich. Für $P_{b,t} = 0{,}01$ zeigen Code- und Diversity-Combining eine ähnliche Fehlerrate. Dies ist für andere Schwellenwerte und Codes nicht zwangsläufig der Fall - wie sich auch für den Schwellenwert von $P_{W,t} = 0{,}1$ zeigt. Eine allgemeingültige Aussage über die Leistungsfähigkeit von Diversity- im Vergleich zu Packet-Combining ist daher nicht möglich.

5.4 Zusammenfassung

In diesem Kapitel wurde die Idee erörtert, vom Kanaldecodierer bereitgestellte Zuverlässigkeitsinformation als Maß für Neuanforderungsanfragen in ARQ-Protokollen zu verwenden. Zusammen mit dem konventionellen Ansatz eines fehlererkennenden Codes als Neuanforderungskriterium wurden die existierenden Ansätze systematisiert und Wort- und Bitfehlerwahrscheinlichkeit als zuverlässigkeitsbasiertes Kriterium für Neuanforderungen eingeführt. Wörter, deren Fehlerwahrscheinlichkeit über einem Schwellenwert liegt, werden neu angefordert. Damit können abhängig von der geforderten Dienstgüte kontrolliert Fehler bei der Übertragung zugelassen werden, um Neuübertragungen wenn möglich zu vermeiden und einen Abtausch zwischen Datenrate bzw. Verzögerung und Fehlerrate zuzulassen. Beide zuverlässigkeitsbasierenden Ansätze wurden mit dem konventionellen Ansatz verglichen und gezeigt, dass in Abhängigkeit von Wortlänge und Schwellenwert beachtliche Steigerungen der Durchsatzeffizienz möglich sind. Die Bitfehlerwahrscheinlichkeit zeichnet sich gegenüber der Wortfehlerwahrscheinlichkeit durch größeres Potential bei der Durchsatzsteigerung und durch größere Flexibilität bezüglich des verwendeten Decodierers und möglicher weiterer Anwendungen aus.

Kapitel 6

Güteorientierte Decodierung

In Abschnitt 2.3 wurde erläutert, dass die Anforderungen verschiedener Anwendungen an die Dienstgüte unterschiedlich sind. Manche Anwendungen, z. B. temporäre Sprach- oder Videoübertragung zur einmaligen Nutzung (z. B. Rundfunk, Daten werden nach anhören/-schauen gelöscht), haben relativ geringe Anforderungen an die BEP. Die Idee der *güteorientierten Decodierung* (engl. quality-oriented decoding) besteht darin, dass ein System bezüglich der Decodierung adaptiv gestaltet wird. Der Decodierer soll nur dann zum Einsatz kommen, wenn es für die Dienstgüte erforderlich ist. Dabei werden hier zwei Szenarien untersucht:

1. Eine Decodierung ist nicht notwendig, wenn die Daten auch ohne Decodierung in ausreichender Qualität erlangt werden können. Dies ist z. B. in Relays (Decodierung und anschließende Recodierung) oder bei der Verwendung systematischer Codes denkbar.

2. In iterativen Strukturen kann auf weitere Iterationen verzichtet werden, wenn die Fehlerwahrscheinlichkeit ausreichend ist. Es wird nur dann eine weitere Iteration durchgeführt, wenn die Fehlerwahrscheinlichkeit nicht ausreichend ist.

In beiden Fällen kann durch Verzicht der Decodierung bzw. weiterer Iterationen von Decodierern Energie gespart werden, was besonders bei Empfängern mit begrenztem Energievorrat (z. B. Mobiltelefonen) interessant ist. Ein weiterer Vorteil ist die Verringerung von durch Decodierung entstehenden mittleren Verzögerungen. In diesem Kapitel werden diese beiden Szenarien anhand von Anwendungsbeispielen dargestellt. Abschnitt 6.2 wurde in ähnlicher Form in [FBH08] veröffentlicht.

6.1 Abbruchbedingungen für iterative Strukturen

In diesem Abschnitt soll die Möglichkeit dargestellt werden, den in Kapitel 4 eingeführten Schätzwert $\widehat{P}_b$ als Abbruchbedingung für iterative Decodierer-Algorithmen zu verwenden. In [LH01] wurde die die durchschnittliche Zuverlässigkeit der systematischen Codebits in einem Turbo-Decodierer als Abbruchkriterium eingeführt. Es wird keine weitere Iteration

durchgeführt, wenn

$$\Lambda^{(1)} = \frac{1}{K} \sum_{k=1}^{K} |\tilde{u}_k^{(1)} - \Pi^{-1}(\tilde{u}_k^{(2)})| \tag{6.1}$$

bzw.

$$\Lambda^{(2)} = \frac{1}{K} \sum_{k=1}^{K} |\Pi^{-1}(\tilde{u}_k^{(2)}) - \Lambda^{(1)}| \tag{6.2}$$

sich im Gegensatz zur letzten Iteration nicht mehr oder nur wenig verändert haben. Andere Kriterien können auch die wechselseitige Information oder ein äquivalentes SNR sein [SBR06]. Im Gegensatz zu [LH01] und [SBR06] soll hier nicht die Konvergenz bzw. Fehlerfreiheit das Ziel der Decodierung sein, sondern eine ausreichende Güte. Es soll entsprechend den Dienstgüteanforderungen im Decodierungsprozess eine bestimmte maximale Fehlerwahrscheinlichkeit erreicht werden. Dies geschieht auch nicht auf Basis der extrinsischen Information, sondern mit Hilfe der a-posteriori LLRs $\tilde{\boldsymbol{u}}$, mit denen nach (4.19) der weiche Schätzwert der Bitfehlerwahrscheinlichkeit $\widehat{P}_b$ berechnet werden kann. Der iterative Decodierer führt eine weitere Iteration durch, wenn $\widehat{P}_b$ größer als die gegebene maximale Bitfehlerwahrscheinlichkeit $P_{b,t}$ ist. Dies funktioniert natürlich nur insofern, als dass nicht weiter decodiert wird, wenn die Fehlerwahrscheinlichkeit niedrig genug ist. Andererseits ist es auch mit beliebig vielen Iterationen nicht immer möglich, ein Wort fehlerfrei zu decodieren. Die Bitfehlerwahrscheinlichkeit als Abbruchkriterium und die daraus resultierenden Fehlerraten und benötigten Iterationen werden im Folgenden für zwei iterative Decodierer dargestellt, den UMTS PCCC mit entsprechendem Turbo-Decodierer mit Log-BCJRs als Komponentendecodierern (Abschnitt 3.2) und den 802.16e Rate 1/2 LDPC-Code mit dem BPA als Decodierer (Abschnitt 3.3).

6.1.1 Turbo-Decodierer

Abb. 6.1 zeigt die relative Häufigkeit der Iterationen, die der Turbo-Decodierer benötigt, um für die decodierten Wörter jeweils $\widehat{P}_b \leq P_{b,t} = 10^{-3}$ bei einem $\gamma_b = 0,5\,\mathrm{dB}$ zu erreichen. Die maximale Anzahl der Iterationen lag bei 20, d. h., dass die meisten der Wörter für deren Decodierung 20 Iterationen benötigt wurden, die Bedingung $\widehat{P}_b \leq P_{b,t}$ nicht erfüllt haben. Dies ist auch in Abb. 6.2 zu erkennen – die BEP bei 20 Iterationen ist sehr hoch und weit über der vorgegebenen 10^{-3}. Andererseits zeigt Abb. 6.1, dass die Anzahl der korrigierten Fehler nach den ersten zwei Iterationen stetig abnimmt, also die Wörter, die mit 20 Iterationen nicht ausreichend zuverlässig decodiert werden konnten zum größten Teil auch in weiteren Iterationen nicht zuverlässig decodiert werden können. Ferner auffällig ist, dass ab der zehnten Iteration die resultierende BEP leicht über 10^{-3} liegt. Dies ist durch die aus der Verletzung der Unabhängigkeitsannahme resultierende mit Anzahl der Iterationen zunehmend zu optimistische Schätzung des Turbo-Decodierers zu erklären (siehe Kapitel 4). Das Problem kann aber bei bekanntem Code und Decodierer relativ einfach gelöst werden, indem $P_{b,t}$ etwas niedriger als die angestrebte BEP gewählt wird. Damit besteht mit der weichen Schätzung der Bitfehlerwahrscheinlichkeit eine Abbruchbedingung zur Verfügung, mit der eine unnötig hohe Anzahl von Iterationen vermieden werden

und damit durch den Decodierer verursachter Energieverbrauch und Verzögerung minimiert werden können. Im Gegensatz zu anderen Abbruchbedingungen wird hierbei keine Konvergenz angestrebt, sondern nur die Einhaltung einer maximal erlaubten Bitfehlerwahrscheinlichkeit.

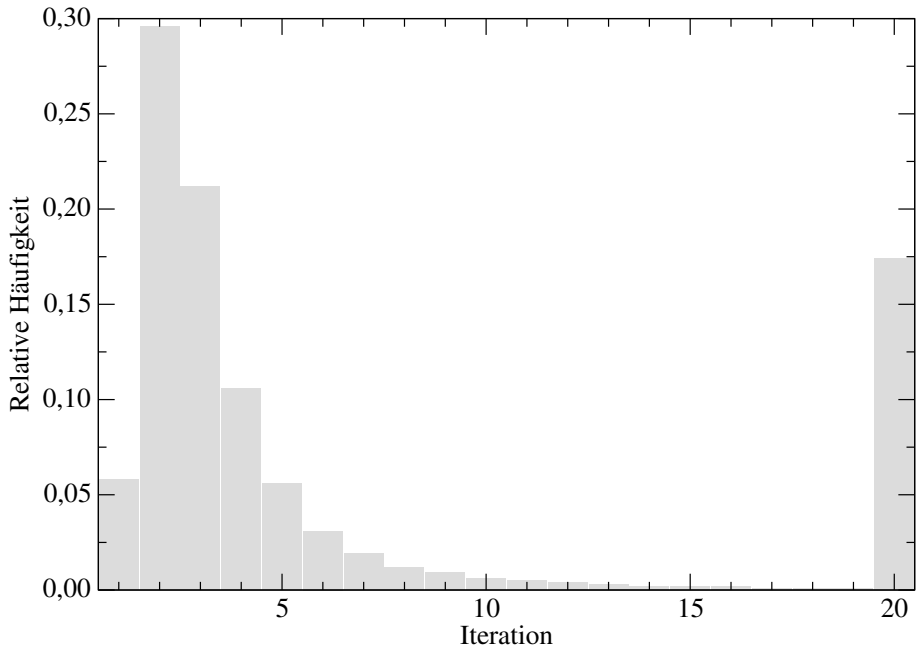

Abbildung 6.1: Relative Häufigkeit der auftretenden Iterationszahl eines Turbo-Decodierers bei güteorientierter Decodierung mit $P_{b,t} = 10^{-3}$ und $\gamma_b = 0{,}5\,\mathrm{dB}$.

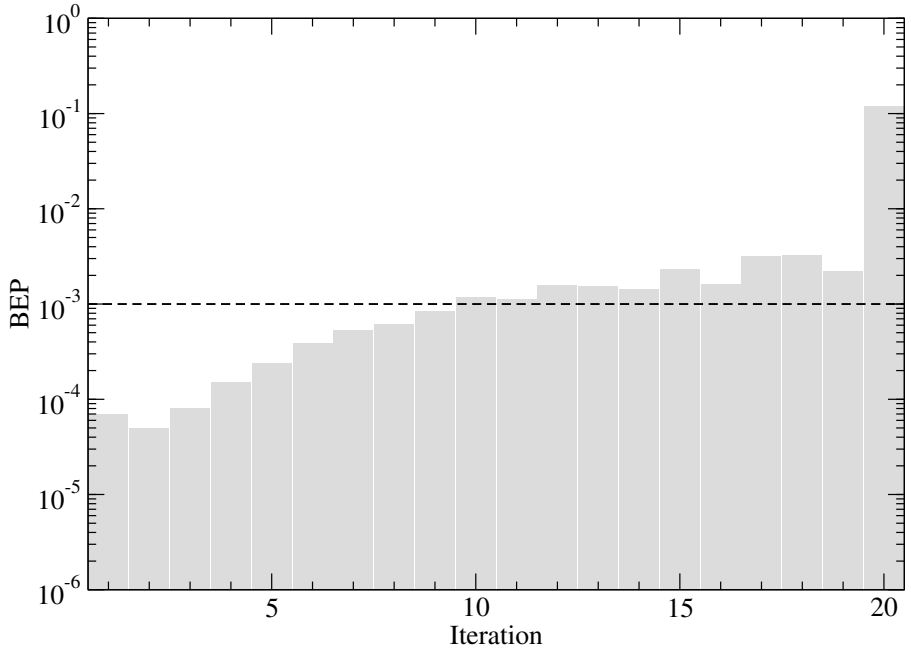

Abbildung 6.2: BEP bei Erfüllung der Abbruchbedingung von $P_{b,t} = 10^{-3}$ für $\gamma_b = 0{,}5\,\mathrm{dB}$.

6.1.2 BPA für LDPC-Code

In Kapitel 3 wurde neben dem PCCC der LDPC-Code als Beispiel für Codes mit iterativen Decodierern eingeführt. Auch hier ist das oben beschriebene Abbruchkriterium möglich. Entsprechende Ergebnisse sind für Decodierung des 802.16e LDPC-Codes in Abb. 6.3 und 6.4 dargestellt.

Im Gegensatz zum Turbo-Decodierer benötigt der LDPC-Decodierer drei Iterationen ehe die ersten Infowörter die Abbruchbedingung von $P_{b,t} = 10^{-3}$ erfüllen. Dies ist verständlich, wenn man bedenkt, dass der BPA mit jeder Iteration keinen vollständigen Informationsaustausch vollführt, sondern nur zwischen miteinander verbundenen Knoten. Es dauert also einige Zeit, bis sich die Information soweit im Graph verteilt hat, dass genügend zuverlässige Infobitschätzwerte entstehen. Ansonsten ist das Verhalten ähnlich dem des Turbo-Decodierers, es bleiben nach 20 Iterationen Worte übrig, die noch immer eine sehr hohe BEP aufweisen und die durch $P_{b,t}$ gegebene Vorgabe nicht erfüllen. Im Gegensatz zu dem Turbo-Decodierer ist die BEP nach Abbruch der Iterationen immer unter dem der Anforderung von $P_{b,t} = 10^{-3}$. Wie in Kapitel 4 gezeigt wurde, leidet der BPA allerdings grundsätzlich auch unter zu optimistischen Schätzungen (zu niedrige $\widehat{P}_b$). Hier macht sich bemerkbar, dass der Decodierer keine weiteren Iterationen durchführt, wenn $P_{b,t}$ erreicht ist. Die Zuverlässigkeit steigt also nicht durch weitere Iterationen an, wodurch die weichen Schätzwerte im Gegensatz zu den Ergebnissen in Kapitel 4 zu pessimistisch sind. Trotzdem ist auch hier zu erkennen, dass die weichen Schätzwerte mit mehr Iterationen optimistischer werden. Mit der Berücksichtigung eines entsprechenden Offsets besteht auch hier die Möglichkeit der güteorientierten Decodierung mit der BEP als Dienstgüte-Parameter.

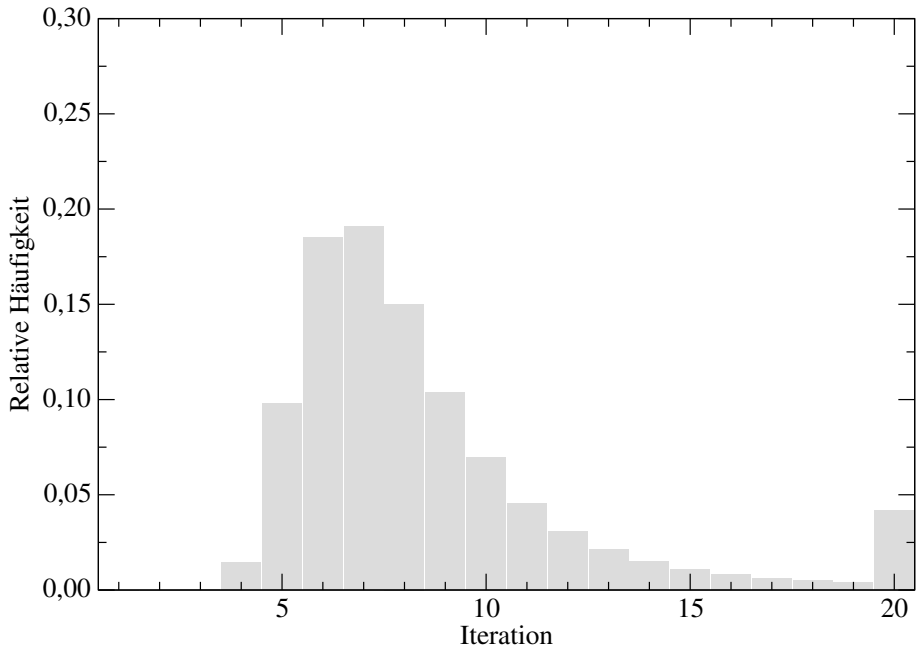

Abbildung 6.3: Relative Häufigkeit der auftretenden Iterationszahl des BPA für die Decodierung eines LDPC-Codes bei güteorientierter Decodierung mit $P_{b,t} = 10^{-3}$ und $\gamma_b = 2{,}0\,\mathrm{dB}$.

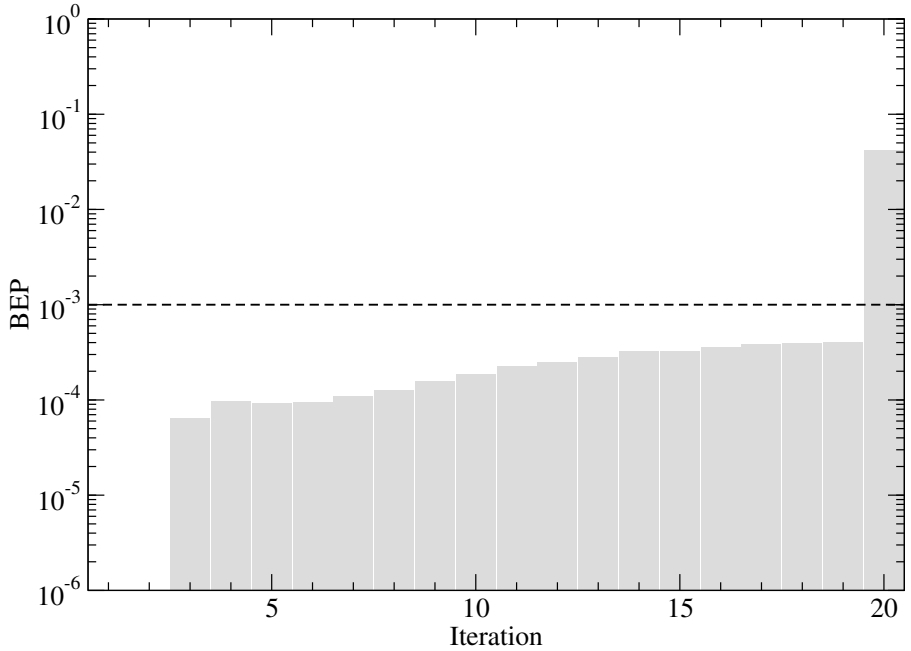

Abbildung 6.4: BEP bei Erfüllung der Abbruchbedingung von $P_{b,t} = 10^{-3}$ für $\gamma_b = 2{,}0\,\mathrm{dB}$ bei Verwendung eines LDPC-Codes mit BPA.

6.2 Decodierung in Netzwerken mit Weiterleitung

Relays werden verwendet, um die Reichweite von Kommunikationssystemen zu erhöhen, indem die Daten von einem oder mehreren Knoten weitergeleitet werden. Durch Relays kann die für die Übermittlung der Nachricht benötigte Signalenergie reduziert werden [HA03], [LVWD06]. Eine Übersicht über die Vielzahl unterschiedlicher Ansätze findet sich in [FK06]. Hier wird nur die Kommunikation über ein einziges Relay betrachtet. Die vorgeschlagene Technik kann aber generell auch für Verkettungen oder mehrere parallel arbeitende Relays verwendet werden.

In Netzwerken mit Relays werden im Wesentlichen drei Strategien verwendet. Die erste, *Amplify and Forward* (AF), detektiert die empfangenen Symbole und sendet sie verstärkt weiter an den nächsten Netzknoten. Durch AF wird der Einfluss von auf der Übertragungsstrecke auftretendem Schwund bzw. Dämpfung kompensiert. Ein wesentlicher Nachteil dieser Strategie ist, dass auch Fehler, die mit der Detektion am Relay aufgetreten sind, verstärkt weitergeleitet werden. Die zweite Strategie, *Decode and Forward* (DF), detektiert die Empfangssymbole ebenfalls, decodiert das empfangene Wort aber zusätzlich. Durch die Decodierung können am Relay aufgetretene Fehler korrigiert werden. Die Weiterleitung von fehlerhaften Symbolen an das nachfolgende Relay wird hiermit unwahrscheinlicher als bei der AF-Strategie. In einer weiteren Strategie, *Decode and Reencode*, wird das Codewort am Relay decodiert und recodiert, aber so, dass ein Codewort ungleich dem ursprünglichen Codewort entsteht. Am Empfänger stellen das ursprünglich übermittelte und das vom Relay übermittelte Codewort einen parallel verknüpften Code dar [ZHF05], [LVWD06], [WWKK08].

Am Relay auftretende Fehler sind deshalb besonders schädlich, weil der nächste Empfänger keine verlässliche Zuverlässigkeitsinformation über die empfangenen Symbole bekommen kann. Ein zuverlässig wirkendes Symbol (hoher LLR oder mit großer Amplitude empfangen) kann ein Symbol sein, das am Relay schon fehlerhaft gesendet wurde. Daher kann es passieren, dass konventionelle soft-input Decodierung in manchen Fällen sogar schlechtere Leistungen erzielt als hard-input Decodierung. Häufig wird daher angenommen, dass die Strecke zwischen Quelle und Relay mit Hilfe von perfekten ARQ-Mechanismen perfekt wiederherstellt wird – also das Relay fehlerfreie Daten weiterleitet. In der jüngeren Vergangenheit wurde die Idee von *Soft-Forwarding*, der Weiterleitung von weichen Schätzwerten vorgestellt. Soft-Forwarding wird im Zusammenhang mit paralleler Codekonstruktion mit Hilfe von Relays [LVWD06], [WWKK08], [SV05]. Ferner kann Soft-Forwarding genutzt werden, um die Diversität von sich durch Relays ergebene unterschiedliche Übermittlungswege zu nutzen [BL05], [GJ06]. Im Zusammenhang mit Cognitive Radio wird die Weiterleitung von weichen Schätzwerten auch in [TL07] genutzt.

6.2.1 Systembeschreibung

Im Rahmen dieser Arbeit wird die Zuverlässigkeitsinformation verwendet, um eine adaptive und autonome Relaying-Strategie zu ermöglichen. Sind die empfangenen Daten zuverlässig genug, so wird auf eine Decodierung verzichtet (AF), da davon ausgegangen werden kann, dass keine oder nur wenige Fehler aufgetreten sind. Im Falle weniger Fehler können diese dann von dem Decodierer im nächsten Empfänger korrigiert werden. Sind

die empfangenen Daten unzuverlässig, so werden sie vor der Weiterleitung decodiert, um Fehler korrigieren zu können (DF). Diese kombinierte AF- and DF-Strategie, im Weiteren ADF genannt, stellt also ebenfalls eine güteorientierte Decodierung dar.

Ein entsprechendes Systemmodell für die Verwendung von nur einem Relay ist in Abb. 6.5 dargestellt. Im Wesentlichen handelt es sich um das Systemmodell aus Kapitel 2, mit dem Unterschied, dass durch das Relay ein weiterer Empfänger/Sender enthalten ist. Eine direkte Verbindung ist hier nicht möglich. Die Informationsbits $\boldsymbol{u}$ werden wie gehabt mit einem $(23_8, 35_8)$ Faltungscode codiert und die Codebits $\boldsymbol{c}$ für die Übertragung auf Symbole $\boldsymbol{x}$ abgebildet. Nach der Übertragung über einen ersten Kanal R werden am Relay

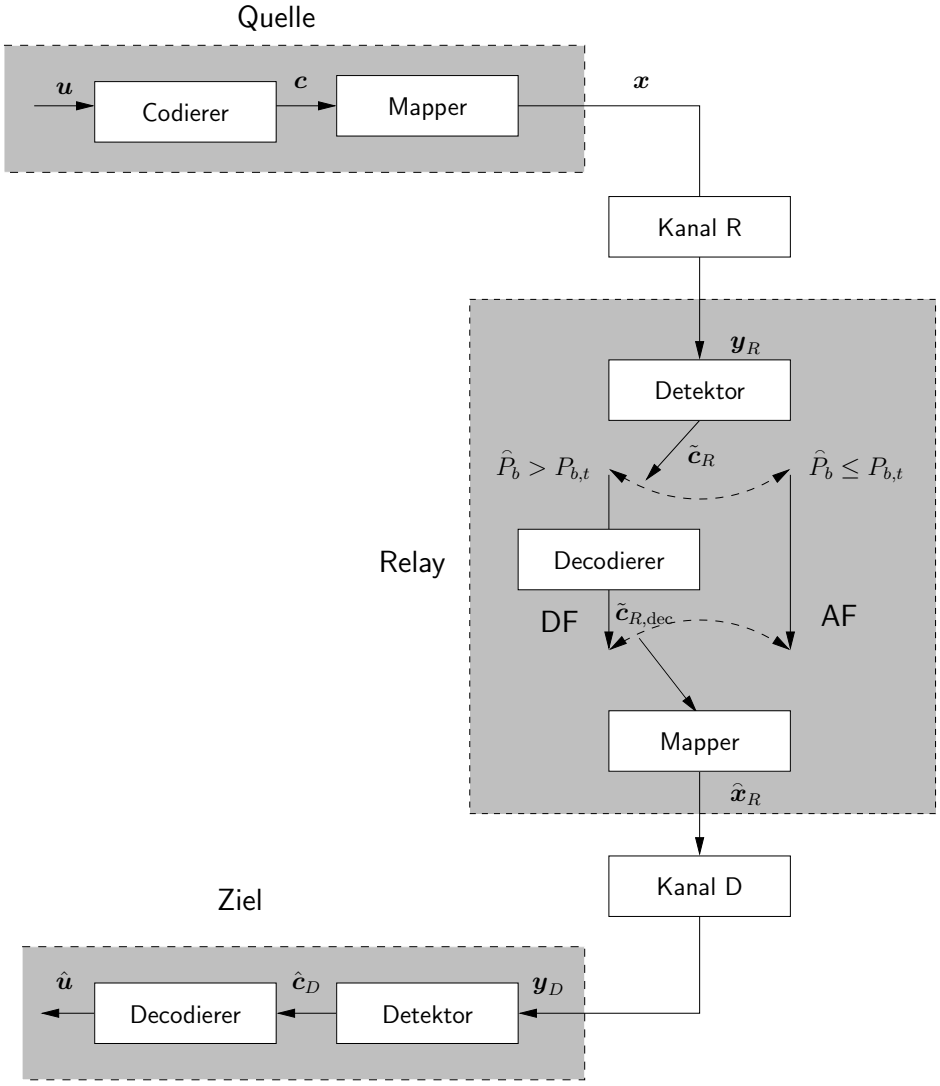

Abbildung 6.5: Systemmodell mit güteorientierter Decodierung im Relay. Im Relay wird entweder die AF- oder die DF-Strategie gewählt.

die Abtastwerte $\boldsymbol{y}_R$ detektiert und LLRs $\tilde{\boldsymbol{c}}$ der übertragenden Codebits bereitgestellt. Verwendet man statt der LLRs der Infobits in (4.19) die vom Detektor gelieferten LLRs der Codebits, so ist es möglich, die durchschnittliche Bitfehlerwahrscheinlichkeit der Bits

im empfangenen Codewort,

$$P_b = \frac{1}{N}\sum_{n=1}^{N} P(\hat{c}_{R,n} \neq c_n|\boldsymbol{y}_R) = \frac{1}{N}\sum_{n=1}^{N} \frac{1}{1+e^{|\tilde{c}_{R,n}|}} \ , \tag{6.3}$$

zu berechnen. Ist P_b größer als ein vorgegebener Schwellenwert $P_{b,t}$, so werden die Schätzwerte am Relay, $\tilde{\boldsymbol{c}}_R$, decodiert, andernfalls nicht. In beiden Fällen erzeugt der Mapper für die Übertragung zum Zielknoten *weiche* Symbole $\widehat{\boldsymbol{x}}_R$. Bei Verwendung von BPSK-Modulation sind diese

$$\widehat{x}_{R,n} = \mathrm{P}(x_n = +1|\tilde{\boldsymbol{c}}_R) - \mathrm{P}(x_n = -1|\tilde{\boldsymbol{c}}_R) \ , \tag{6.4}$$

im AF-Fall, während sie im DF-Fall gemäß

$$\widehat{x}_{R,n} = \mathrm{P}(x_n = +1|\tilde{\boldsymbol{c}}_{R,\mathrm{dec}}) - \mathrm{P}(x_n = -1|\tilde{\boldsymbol{c}}_{R,\mathrm{dec}}) \tag{6.5}$$

berechnet werden, also zusätzlich die Information aus der Decodierung enthalten. Wie in [LVWD06] werden zur Berechnung von weichen Symbolen äquivalente Rauschabtastwerte $\bar{n}_{R,n} = |\hat{x}_{R,n} - \widehat{x}_{R,n}|$ definiert. Der Mittelwert des äquivalenten Rauschprozesses ist

$$\mu_{\bar{n}} = \frac{1}{N}\sum_{n=1}^{N}(1 - \widehat{x}_{R,n}\hat{x}_{R,n}) \tag{6.6}$$

die Varianz

$$\sigma_{\bar{n}}^2 = \frac{1}{N}\sum_{n=1}^{N}(1 - \widehat{x}_{R,n}\hat{x}_{R,n} - \mu_{\bar{n}})^2 \ . \tag{6.7}$$

Vor der Übertragung am Relay werden die weichen Symbole entsprechend des Kanalmodells (Abschnitt 2.2) durch Multiplikation mit

$$\beta = \frac{1}{\sqrt{(1-\mu_{\bar{n}})^2 + \sigma_{\bar{n}}^2}} \tag{6.8}$$

auf eine durchschnittliche Symbolenergie von eins normalisiert. Damit ergeben sich die Abtastwerte am Empfänger zu

$$y_{D,n} = h_{D,n}\beta\widehat{x}_{R,n} + n_{D,n} = h_{D,n}\beta(1-\mu_{\bar{n}})\hat{x}_{R,n} + n_{E,n} \ . \tag{6.9}$$

Aus (6.9) und der Annahme, dass die effektiven Rauschabtastwerte am Empfänger, $\boldsymbol{n}_E$, mit der Varianz $\sigma_E^2 = |\beta h_{D,n}|^2\sigma_{\bar{n}}^2 + \sigma_D^2$ [LVWD06] gaußverteilt sind, können die LLRs für die empfangenen BPSK-Symbole[1] errechnet werden:

$$\begin{aligned}
\tilde{x}_n &= \log\frac{\mathrm{P}(x_n = +1|\boldsymbol{y}_D)}{\mathrm{P}(x_n = -1|\boldsymbol{y}_D)} \\
&= \log\frac{\exp\left(-\dfrac{|y_{D,n} - \beta h_{D,n}(1-\mu_{\bar{n}})|^2)}{\sigma_E^2}\right)}{\exp\left(-\dfrac{|y_{D,n} + \beta h_{D,n}(1-\mu_{\bar{n}})|^2)}{\sigma_E^2}\right)} \\
&= 4\,\mathrm{Re}\left\{\frac{y_{D,n}^*\beta h_{D,n}(1-\mu_{\bar{n}})}{\sigma_E^2}\right\} \ .
\end{aligned} \tag{6.10}$$

[1] Ähnlich lassen sich auch die Wahrscheinlichkeiten für Symbole anderer Modulationsformate und daraus die LLRs der Codebits errechnen.

Die Weiterleitung von weichen Symbolen wird in diesem Szenario aus zwei Gründen benötigt. Erstens wird zwischen Quelle und Relay im Gegensatz zu vielen anderen Ansätzen in der Literatur keine perfekte Fehlerkontrolle vorausgesetzt. Damit ist es in dieser ADF-Strategie möglich, dass fehlerhafte Bits weitergeleitet werden. Damit der nächste Empfänger zuverlässig funktioniert, ist die Zuverlässigkeitsinformation der übertragenen Bits also unabdingbar. Zweitens ist es anders als in dem in Abb. 6.5 dargestellten System natürlich möglich, dass nach dem Relay nicht sofort der Zielknoten folgt, sondern weitere Relays. Sollen diese ebenfalls die adaptive ADF-Strategie verwenden, so benötigen auch sie die Übermittlung weicher Symbole, um die Zuverlässigkeit der empfangenen Daten richtig zu schätzen.

6.2.2 Güteorientierte Decodierung

Je nachdem, wie viele Fehler durch den Kanal zwischen Sender und Relay erzeugt werden, ist AF oder DF die bessere Strategie. Anstatt sich auf eine der beiden Strategien festzulegen, ermöglicht das oben beschriebene System mit Hilfe güteorientierter Decodierung eine adaptive Strategie. Wenn die harten Schätzwerte am Relay $\hat{\boldsymbol{c}}_R$ keine oder genügend wenig Fehler enthalten, werden sie ohne Decodierung weitergeleitet, andernfalls werden sie decodiert, um Fehler zu korrigieren. Dabei sind reines AF und DF als Spezialfälle für die Schwellenwerte $P_{b,t} = 0$ bzw. $P_{b,t} = 0{,}5$ enthalten. Neben der in (6.3) definierten Bitfehlerwahrscheinlichkeit der empfangenen Codebits sind natürlich auch andere Gütekriterien möglich. Naheliegend ist die Wortfehlerwahrscheinlichkeit nach Kapitel 4 oder auch die Symbolfehlerwahrscheinlichkeit der empfangenen Symbole $\hat{\boldsymbol{x}}_R$.[2]

Diese güteorientierte Strategie ermöglicht es, Energie für den Decodierer zu sparen, die andernfalls unnötigerweise für die Decodierung aufgebracht würde, wenn $\hat{\boldsymbol{c}}_R$ keine oder nur wenige Fehler enthält – wenige Fehler werden mit hoher Wahrscheinlichkeit im Empfänger des Zielknotens korrigiert. Alle Relays können den lokalen Parameter $P_{b,t}$ unabhängig voneinander wählen. Wie $P_{b,t}$ gewählt wird hängt von verschieden Parametern wie Kanalcode, Modulationsformat, Kanal, Sendeleistung, Netzplanung und geforderter Dienstgüte ab. Der Schwellenwert könnte z. B. anhand der Zahl der Neuanforderungen adaptiert werden oder aber auch anhand der gemessenen Fehlerwahrscheinlichkeit $\widehat{P}_b$ am folgenden Knoten.

Insbesondere sinnvoll ist eine solche adaptive ADF-Strategie für Netzwerke, die sich selbst organisieren bzw. ohne Netzplanung gebildet werden. Dies passiert z. B. wenn Sensornetzwerke durch Abwurf der Sensorknoten von einem Schiff oder Flugzeug gebildet werden. Damit kann ihre letztendliche Position nicht kontrolliert werden, eine Netzplanung ist nicht möglich. Gerade in solchen Fällen ist auch die Energieversorgung der einzelnen Knoten begrenzt. Mit dem Ausfall eines Knotens fällt unter Umständen nicht nur ein Sensor sondern sogar das ganze Netzwerk aus. Es ist also sinnvoll oder sogar notwendig, diese Möglichkeit der Energieeinsparung zu nutzen.

In Abb. 6.6 ist die BEP für ein Übertragungssystem entsprechend Abb. 6.5 dargestellt. Dabei ist die BEP nach der Decodierung am Zielknoten für unterschiedliche $P_{b,t}$ im Relay

[2] Für die hier angenommene BPSK-Übertragung sind die Wahrscheinlichkeiten von $\boldsymbol{x}_R$ und $\boldsymbol{c}_R$ identisch.

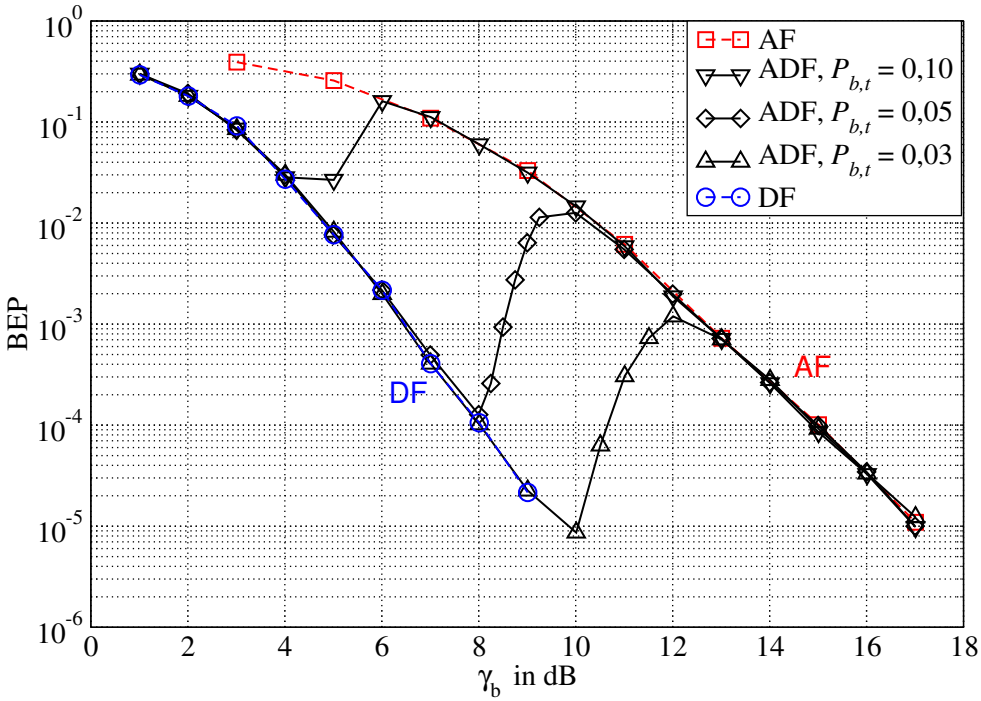

Abbildung 6.6: BEP nach Decodierung am Zielknoten bei Übertragung mit einem Relay.

angegeben. Als Decodierer dient der BCJR-Algorithmus. Die Kanäle R und D stellen frequenzflache Fading-Kanäle nach Abb. 2.8 dar. Die Kanalkoeffizienten $h_{R,m}$ und $h_{D,m}$ sind komplexwertig mit gaußverteilten Amplituden und $\mathrm{E}\{|h_{R,m}|^2\} = \mathrm{E}\{|h_{D,m}|^2\} = 1$. Die Infobitenergie γ_b wird für beide Kanäle gleich gewählt. Gezeigt wird sowohl die BEP für die ADF-Strategie wie auch für die reine AF- bzw. DF-Strategie. Dabei ist die BEP der ADF-Strategie zwischen der der reinen Strategien – was zu erwarten ist, da diese jeweils Spezialfälle für extreme Werte von $P_{b,t}$ darstellen. Dabei zeigt die gemischte Strategie eine Leistung, die für kleine γ_b der DF-Strategie und für große γ_b gleich der AF-Strategie ist. Abhängig von dem gewählten Schwellenwert verlässt die BEP-Kurve der gemischten Strategie die Kurve der DF-Strategie bei einem niedrigerem oder höherem γ_b, da zum Erreichen eines niedrigeren Schwellwerts eine größere Sendeleistung notwendig ist. Nach einem kurzen Anstieg vereinigt sie sich bei sich ca. 2 dB nach dem Verlassen der DF-Kurve mit der AF-Kurve. Je niedriger der Schwellenwert für die Decodierung gewählt wird, um so größer ist γ_b, für das der Anstieg der BEP-Kurven der gemischten Strategie beginnt. Eine analytische Bestimmung dieses Punktes wäre nur bei Kenntnis der durchschnittlichen Fehlerwahrscheinlichkeit unter Berücksichtigung sämtlicher oben genannter Parameter möglich. In Abb. 6.7 ist der relative Anteil der Pakete mit Decodierernutzung im Relay angegeben. Eine Decodierernutzung von $1,0$ entspricht der DF-Strategie, eine von $0,0$ der AF-Strategie. Je niedriger $P_{b,t}$ gewählt wird, um so höher ist γ_b bei der die Decodierernutzung beginnt abzufallen. Vergleicht man Abb. 6.6 mit Abb. 6.7, so erkennt man gut den Zusammenhang zwischen abnehmender Decodierernutzung und ansteigender BER. Dabei ist der Bereich der abnehmenden Decodierernutzung jeweils der Bereich der ansteigenden BER.

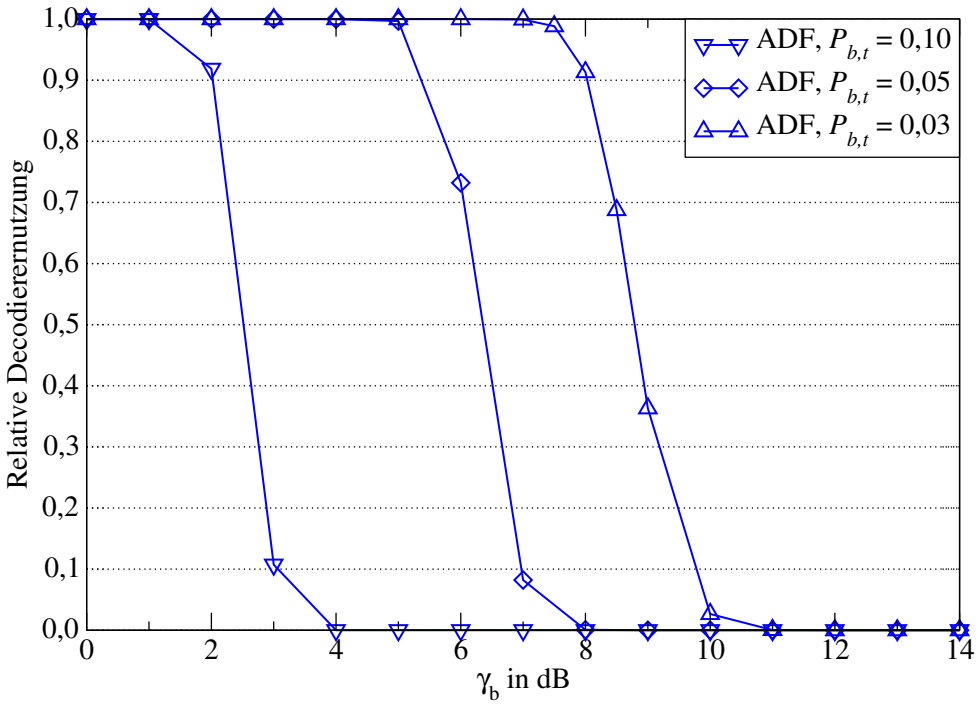

Abbildung 6.7: Relativer Anteil der Übertragungen mit Decodiererverwendung für die ADF-Strategie.

Vergleichbare Ergebnisse ergeben sich (für die Verwendung von nur einem Relay) für Hard-Forwarding, wenn also $\hat{\boldsymbol{x}}_R$ statt $\widehat{\boldsymbol{x}}_R$ gesendet wird. Für Hard-Forwarding sind die Kurven leicht nach rechts verschoben, da durch den Ziel-Decodierer bei harten Eingangswerten ein geringerer Codiergewinn erzielt wird.

Güteorientierte Decodierung ermöglicht hier also ohne zentrale Steuerung des Netzes einen Abgleich zwischen BEP und Energieverbrauch. Damit sind speziell Netzwerke mit begrenzten Energiereserven in den Netzknoten und begrenztem Einfluss auf die Netztopologie eine interessante Anwendung güteorientierter Decodierung.

6.3 Zusammenfassung

In diesem Kapitel wurden zwei Anwendungen der güteorientierten Decodierung vorgestellt. In beiden Fällen wird nur (weiter-)decodiert, wenn es die Anforderung an die Dienstgüte erfordert. Im ersten Fall wird in Relays bei ausreichend kleiner Fehlerwahrscheinlichkeit auf die Decodierung verzichtet, im zweiten Fall auf weitere Iterationen von iterativen Strukturen verzichtet, wenn die Fehlerwahrscheinlichkeit für die Dienstgüteanforderungen ausreichend klein geworden sind. In beiden Fällen können auf diese Weise Energieverbrauch und entstehende Verzögerungen möglichst klein gehalten werden.

Kapitel 7

Zuverlässigkeitsbasiertes Routing

Die Netzwerkschicht verbindet einzelne Netzknoten und ermöglicht die Kommunikation zwischen ihnen selbst dann, wenn sie, z. B. auf Grund großer Entfernung, nicht direkt miteinander kommunizieren können. Die Kommunikation erfolgt dabei über andere Knoten, die zwischen den beiden kommunizierenden Netzknoten liegen. Eine wesentliche Voraussetzung hierfür ist die *Pfadsuche* (engl. routing). Bei der Pfadsuche wird der geeignetste Pfad $\boldsymbol{P}$ (bzw. Route) durch das Netzwerk aus der Menge der möglichen Pfade $\mathbb{P}$ von einem Quellknoten S zu einem Zielknoten D gesucht. Dieser Pfad besteht aus einer Reihe von Netzknoten beginnend mit S und endend mit D, die jeweils die von S nach D übermittelte Nachricht empfangen und zu dem nächsten, auf der Route nach D liegenden, Knoten weiterleiten. Die Verbindung zwischen zwei Knoten wird üblicherweise als *Link*, die Übertragung über eine solche Knoten-zu-Knoten-Verbindung als ein *Hop* bezeichnet. Die Knoten, die diese Route bestimmen, werden auch *Router* genannt. Welcher Pfad durch das Netzwerk der geeignetste ist bzw. ob dieser ausgewählt wird, entscheidet ein Routing-Algorithmus. Entsprechend den Anwendungen und Voraussetzungen existieren eine Vielzahl unterschiedlicher Routing-Algorithmen. Ähnlich verhält es sich mit den Routing-Protokollen (siehe auch Abb. 2.1.1), die den Informationsaustausch zwischen den Routern übernehmen. Um den geeignetsten Weg durch das Netzwerk zu ermitteln, benutzt der Routing-Algorithmus eine sogenannte *Routing-Metrik*. Die Routing-Metrik stellt abstrakte Kosten für die Übermittlung eines Paketes über einen Pfad oder Teilpfad im Netzwerk dar. Die Kosten können Verzögerung, Datenraten, Energieverbrauch oder andere übertragungsrelevante Werte repräsentieren. Verschiedene Routing-Algorithmen und -Protokolle sind in [KR05] angegeben.

Ein weitverbreiteter konventioneller Ansatz um die Pfadkosten darzustellen, stellt der Hop-Count dar [KR05]. Ausgehend von der Annahme, dass Verzögerungen hauptsächlich durch die Verarbeitung in den Netzknoten (Warteschlange, Signalverarbeitung und Abarbeitung von Protokollstapeln) entstehen, ist die Übermittlung einer Nachricht am schnellsten, wenn der gewählte Pfad möglichst wenige Hops enthält. Der Ansatz dieser Routing-Metrik ist also, den Pfad mit der geringsten Anzahl Verbindungen (engl. minimum link count – MLC) zu wählen. In Funknetzwerken ist dieser Ansatz nicht immer der günstigste. Dort stellen Hops größerer Distanz höhere Anforderungen an die Sendeleistung und weisen damit einen höheren Energieverbrauch auf. Andererseits steigt bei gleicher Sendeleistung mit steigender Distanz die Fehlerwahrscheinlichkeit, es kommt zu Neuüber-

tragungen, die Datenrate sinkt. Der höheren Fehlerwahrscheinlichkeit kann zwar wieder mit niedrigeren Coderaten bzw. robusterer Modulation entgegengewirkt werden, jedoch senkt auch dies die Datenrate. Im Grunde genommen werden die abstrakten Übertragungskosten in einem Funknetzwerk also von dem QoS-Dreieck der physikalischen Schicht (siehe Abb. 2.9) dominiert. Besonders schwierig sind diese Effekte zu handhaben, wenn die Qualität der Verbindungen zwischen den einzelnen Netzknoten nicht von vornherein planbar ist und sich die Struktur des Netzwerks spontan ergibt. Gerade für diese (normalerweise funkbasierten) Ad-Hoc-Netzwerke wurden eine große Zahl Algorithmen und Metriken vorgeschlagen; eine kleine, aber relevante Auswahl soll kurz vorgestellt werden. Ein Ansatz ist z. B. *Signal-Stability Based Adaptive Routing* (SSA), in dem die Signalstabilität (Übertragungsstabilität) einer Verbindung als Metrik verwendet wird [DRKT97]. Bei dem Konzept des *power-aware routing* wird der Pfad ausgewählt, der die kleinste Menge Energie pro Paket verbraucht [SWR98]. Eine andere Kategorie von Routing-Metriken basieren auf der Verbindungsqualität. Ein Ansatz ist die Verwendung der Round-Trip-Time einer Verbindung (z. B. [ABP+04]) – je höher diese ist, desto schlechter die Verbindung, da entweder die Warteschlange voll ist, oder die Übertragungsrate niedrig ist. Die *Per-Hop Packet-Pair Delay* Metrik [DPZ04] misst dagegen die Verzögerung zwischen zwei Testpaketen, um die Verbindungsqualität zu messen. Ein anderer Ansatz wird in [SM05a] verfolgt; dort wird statistisches Kanalwissen verwendet, um Routing adaptiv in Abhängigkeit der zu Grunde liegende physikalischen Kanäle zu gestalten. Eine Übersicht mit genaueren Erläuterungen unterschiedlicher güteorientierter Routing-Metriken findet sich in [Kok07].

7.1 Einfluss der physikalischen Schicht auf die effektive Pfadlänge

In einem Funknetzwerk wird der mögliche Durchsatz zu einem großen Teil von der Umgebung des Netzwerks beeinflusst. Die Empfangsleistung wird durch Ausbreitungsdämpfung, Abschattung und Kurzzeitschwund durch Mehrwegeausbreitung und Mobilität beeinflusst. Diese relativ starken Fluktuationen in der empfangenen Symbolenergie führen zu geringen Datenraten oder sogar Verbindungsabbrüchen. Durch Wegfall von Routen können Netzwerk-Explorationsprozesse notwendig werden, deren Kontroll- und Pilotnachrichten dann die Gesamtleistung des Netzwerks verringern. Die meisten existierenden Algorithmen und Metriken ziehen die physikalische Schicht nicht zur Metrikermittlung heran oder basieren auf sehr einfachen Modellen wie z. B. Freiraumausbreitung. Der Einfluss von Kanalcodierung (FEC) wird üblicherweise nicht betrachtet. Andererseits werden in fast allen digitalen Übertragungssystemen Kanalcodes verwendet, um die Übertragungsqualität zu verbessern bzw. zu stabilisieren.

Der Einfluss der physikalischen Schicht einer einzelnen Verbindung zwischen zwei Knoten kann großen Einfluss auf die Dauer, die ein Paket für die Übertragung benötigt, haben. Bei guter Übertragungsqualität wird das Paket des sendenden Knotens mit Hilfe der physikalischen Schicht an den empfangenden Knoten übertragen. Wird in der Sicherungsschicht des Empfangsknotens ein Fehler entdeckt, so wird eine Neuanforderung

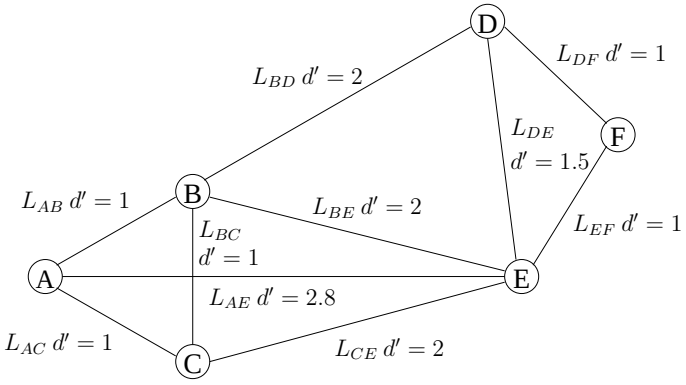

Abbildung 7.1: Aus Netzwerkknoten und Funkverbindungen nach Abb. 7.2 bestehendes Funknetzwerk.

an die Sicherungsschicht des Sendeknotens geschickt – das Paket erreicht aber gar nicht die Netzwerkschicht. Der Empfangsknoten weiß also gar nicht, dass er ein Paket hätte empfangen sollen. Werden auch die neu angeforderten Übertragungen auf der physikalischen Schicht nicht fehlerfrei durchgeführt, so kann das Paket zwischen den beiden Knoten nicht oder nur nach mehrfachen physikalischen Übertragungen übermittelt werden. Für die Verbindung auf der Netzwerkschicht zwischen zwei Netzknoten bedeutet dies, dass die Anzahl Hops, die ein Paket benötigt, nicht unbedingt der Anzahl Knoten-zu-Knoten-Verbindungen entspricht. Die Zahl der Verbindungen eines Pfades M entspricht der Anzahl Knoten des Pfades minus 1. Die Anzahl Hops, also Übertragungen auf der physikalischen Schicht, kann jedoch durch die vom ARQ-Protokoll auf der Sicherungsschicht initiierten Neuübertragungen erheblich größer sein. Dem soll mit dem Begriff *effektive Pfadlänge* Rechnung getragen werden, die der tatsächlich für die Übermittlung von S nach D benötigten Hops entspricht. Konsequenterweise sollte die Routing-Metrik also die Fehlerwahrscheinlichkeit bzw. die Wahrscheinlichkeit von Neuübertragungen der physikalischen bzw. Sicherungsschicht berücksichtigen. Beides hängt von Beschaffenheit des Kanals, Modulationsverfahren, Sendeleistung und Kanalcodierung ab. Damit liegt es in dieser Arbeit nahe, die in Kapitel 4 erläuterte Möglichkeit die vom Decodierer zur Verfügung gestellte Bitfehler- bzw. Wortfehlerwahrscheinlichkeit zu nutzen. Diese wird in einem schichtübergreifendem Ansatz an das Routing-Protokoll weitergegeben, das es wiederum dem Routing-Algorithmus zur Auswahl eines geeigneten Pfades zur Verfügung stellt. Daraus ergeben sich direkt die beiden Ansätze, die Bitfehlerwahrscheinlichkeit und die Wortfehlerwahrscheinlichkeit als Metrik zu nutzen. Eine weitere Möglichkeit ist, aus der Wortfehlerwahrscheinlichkeit die erwartete Anzahl Hops (engl. expected hop count – EHC) zu ermitteln. Die erwartete Anzahl Hops entspricht der effektiven Pfadlänge. Also wird mit dem Pfad, der den kleinsten EHC aufweist, der effektiv kürzeste ausgewählt.

Die Möglichkeit, die Fehlerwahrscheinlichkeiten als Routing-Metriken zu nutzen, soll im Folgenden erörtert werden. Dafür wird ein Modell wie in Abb. 7.2 verwendet, um die Verbindung zwischen zwei Netzknoten zu simulieren. Zur Beurteilung der Leistungsfähigkeit der einzelnen Routing-Metriken werden Übertragungen in einem Netzwerk (Abb. 7.1) simuliert. In regelmäßigen Abständen (fünf Pakete) werden die Routing-Metri-

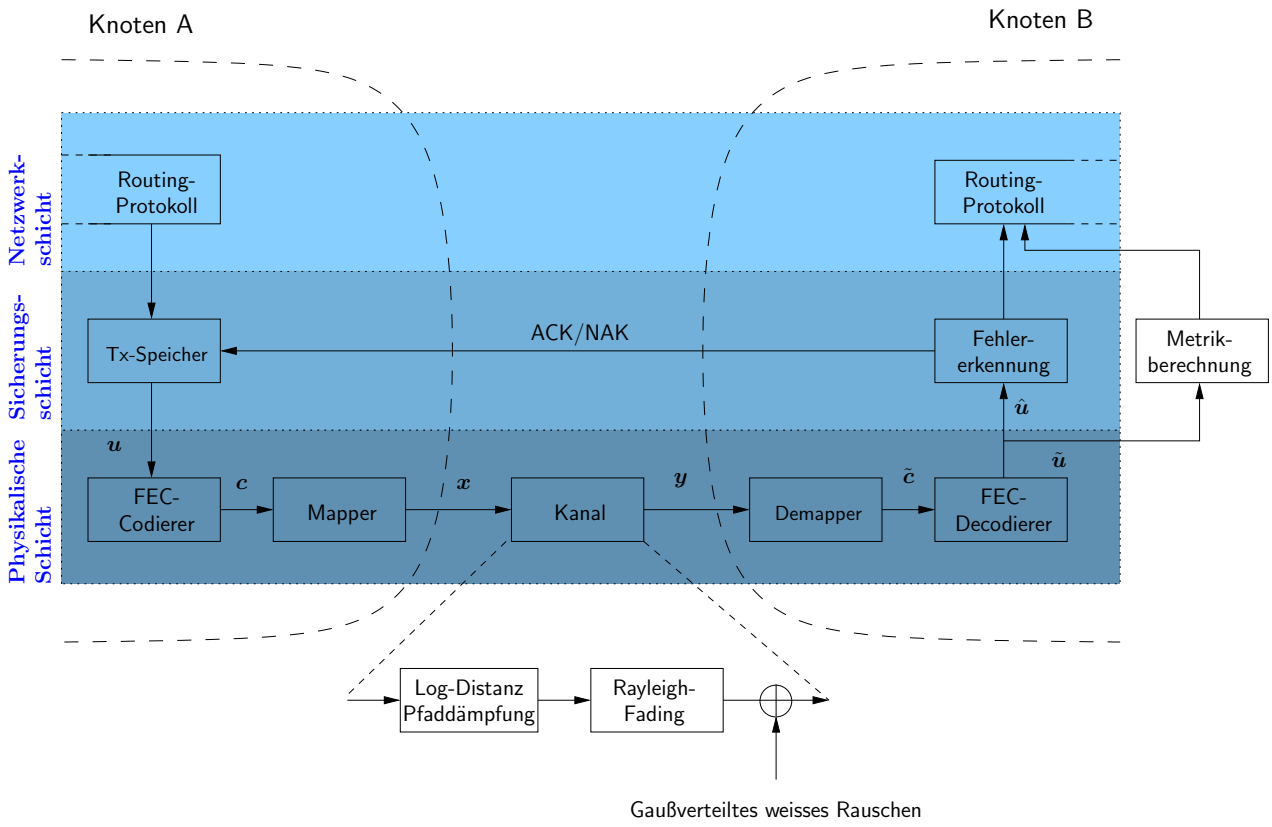

Abbildung 7.2: Physikalische, Sicherungs- und Netzwerkschicht einer Funkverbindung zwischen zwei Netzwerkknoten.

ken aktualisiert. Die Abstände der Aktualisierungen können relativ klein sein, da keine Pilotpakete notwendig sind, sondern die Fehlerwahrscheinlichkeit anhand jedes decodierten Pakets bestimmt werden kann. Andererseits müssen die Routing-Metriken (zumindest bei einem zentralisiertem Algorithmus) auch den anderen Knoten mitgeteilt werden – was zusätzliche Kontrollnachrichten impliziert. Daher werden die Routing-Metriken in den durchgeführten Simulationen nur alle fünf Pakete anhand einer einzelnen Übertragung aktualisiert. In Abhängigkeit von der Sendeleistung der einzelnen Knoten im Netz wird die durchschnittliche Anzahl Hops (Übertragungen von Paketen auf der physikalischen Schicht) ermittelt. Aus Abb. 7.1 ist erkennbar, dass die kürzeste Verbindung zwischen den Knoten A und F aus dem Weg $\boldsymbol{P}_{\mathrm{MLC}} = \{L_{AE}, L_{EF}\}$ besteht. Dieser Pfad $\boldsymbol{P}_{\mathrm{MLC}}$ wird von dem Routing-Algorithmus ausgewählt, wenn die MLC-Metrik verwendet wird. Diese Metrik entspricht der Anzahl Verbindungen, die ein Pfad enthält – resultiert also in einer minimalen Anzahl Übertragungen auf der physikalischen Schicht, wenn diese fehlerfrei überträgt.

7.1.1 Physikalische Schicht

Das Modell der physikalischen Schicht entspricht im Wesentlichen dem in Abschnitt 2.2 vorgestellten. Als Kanalcode wird ein terminierter Faltungscode mit den Generatorpoly-

nomen $(23_8, 35_8)$ und einer Infowortlänge von $K = 288$ verwendet. Das Modulationsverfahren ist der Einfachheit halber konventionelle BPSK-Übertragung, was allerdings keineswegs zwingend ist und die grundsätzliche Aussage der Ergebnisse nicht verändert, solange soft-output Verarbeitung verwendet wird. Der Kanal ist eine Kombination verschiedener Modellierungsansätze. Neben einem AWGN-Anteil wird Log-Distanz Pfad-Dämpfung und Rayleigh-Schwund verwendet, um den Kanal zu modellieren [Rap96]. Mit der distanzabhängigen Pfaddämpfung (Pfaddämpfungsexponent $n_{\mathrm{PL}} = 3$) wird die entfernungsabhängige Dämpfung zwischen zwei Netzknoten modelliert. Der Rayleigh-Schwund mit einer maximalen normalisierten Doppler-Frequenz $f_{D,\max} T_s = 6,6 \cdot 10^{-6}$ (entspricht einer Geschwindigkeit von 3 km/h bei einer Trägerfrequenz von 2,4 GHz und Symboldauer $T_s = 1\,\mu s$). Der Rayleigh-Schwund bewirkt eine langsame Änderung der Übertragungsqualität einer Verbindung, wodurch sich auch die Qualität der einzelnen Pfade mit der Zeit ändert.

7.1.2 Sicherungsschicht

Die Sicherungsschicht realisiert hier nur die ARQ-Funktionalität. Das ARQ-Protokoll fordert Pakete neu an, wenn sie am Empfangsknoten nicht fehlerfrei decodiert werden können. Durch die Neuübertragung auf der physikalischen Schicht entstehen also zusätzliche Verzögerungen, wenn der Kanal zu viele bzw. nicht korrigierbare Fehler erzeugt. Da das Augenmerk hier hauptsächlich auf der Auswirkung schlechter Übertragungsbedingungen auf der physikalischen Schicht und deren Auswirkungen auf die Übertragung auf der Netzschicht liegt, wird das ARQ-Protokoll so einfach wie möglich gehalten. Es wird ein einfaches SW ARQ-Protokoll (Abschnitt 3.5) ohne Packet-Combining verwendet, das ein Infowort $\boldsymbol{u}$ erneut überträgt, wenn die harten Schätzwerte $\hat{\boldsymbol{u}}$ Fehler enthalten. Die Verarbeitungszeit und die Verzögerungszeit durch den Rückkanal werden dabei als Null angenommen.

Im Folgenden werden zwei unterschiedliche Strategien für die Realisierung des ARQ-Protokolls verwendet. Im ersten Fall findet die Fehlerkontrolle abschnittsweise statt (engl. link-wise error control) – dies ist der herkömmliche Ansatz, die Fehlerkontrolle in der Sicherungsschicht anzusiedeln. Im zweiten Fall wird nur im Zielknoten auf Fehler geprüft. Dies wird häufig vorgeschlagen, da bei der Verwendung vieler Protokolle in der Transportschicht (z. B. TCP) eine weitere Fehlerkontrolle durchgeführt wird. Eine der beiden ist also theoretisch überflüssig.

Auf die Modellierung des MAC-Protokolls wird hier verzichtet. Dies kann zwar, insbesondere wenn einzelne Knoten überlastet sind, großen Einfluss auf die Paketübertragungsdauer haben, hat aber auf den Vergleich der hier vorgeschlagenen Metriken keine Auswirkung.

7.1.3 Netzwerkschicht

Der Routing-Algorithmus findet anhand der Routing-Metrik den günstigsten Weg durch das Netzwerk. Dabei wird der Aufbau des Netzwerks insofern als bekannt vorausgesetzt, als dass die einzelnen Verbindungen und ihre Metrik bekannt sind. Der Inhalt der Pakete ist zufällig generiert. Da Netzwerkschicht und Sicherungsschicht in diesen Simulationen keine PCI hinzufügen, stellt die SDU gleichzeitig die PDU dar und ist gleich dem zu

übermittelnden Infowort $\boldsymbol{u}$. Der Quellknoten in den hier angestellten Betrachtungen ist immer A, der Zielknoten F.

7.2 Routing-Metriken für abschnittsweise Fehlerkontrolle

Für den Ansatz der abschnittsweisen Fehlerkontrolle wird das ARQ-Protokoll zwischen zwei Netzwerkknoten implementiert. Tritt bei der Übertragung auf der physikalischen Schicht ein Fehler auf, so wird $\boldsymbol{u}$ so lange übertragen, bis eine fehlerfreie Übertragung erfolgt.

Entsprechend Kapitel 4 kann mit (4.23) ein weicher Schätzwert der Bitfehlerwahrscheinlichkeit $\widehat{P}_{B,l}$ der l-ten Verbindung in einem Pfad $\boldsymbol{P} = [L_1, \ldots, L_l, \ldots, L_L]$ berechnet werden. Daraus kann die Bitfehlerwahrscheinlichkeit für den Pfad

$$\widehat{P}_B = 1 - \prod_{l=1}^{L} (1 - \widehat{P}_{B,l}) \tag{7.1}$$

geschätzt werden.[1] Der Routing-Algorithmus wählt mit Hilfe einer vollständigen Suche den Pfad mit der kleinsten BEP $\widehat{P}_B$, da dort die Wahrscheinlichkeit von Neuanforderungen entsprechend gering ist. Ein ähnlicher Ansatz wird in [FMBT05] beschrieben. Dort wird die Fehlerwahrscheinlichkeit einer Verbindung allerdings analytisch aus den Kanaleigenschaften (Distanz zwischen den Knoten sowie SNR) berechnet, was für kompliziertere Kanäle und Verwendung von Kanalcodierung nicht möglich ist.

Ähnlich kann aus den Wortfehlerwahrscheinlichkeiten der einzelnen Verbindungen $\widehat{P}_{W,l}$ die Wortfehlerwahrscheinlichkeit des gesamten Pfades,

$$\widehat{P}_W = 1 - \prod_{l=1}^{L} (1 - \widehat{P}_{W,l}) \ , \tag{7.2}$$

berechnet werden. Wie bei der BEP-Metrik wählt der Routing-Algorithmus für die Übertragung den Pfad mit der kleinsten Fehlerwahrscheinlichkeit.

Das eigentliche Entscheidungskriterium, welcher Pfad gewählt wird, sollte nicht die Fehlerwahrscheinlichkeit, sondern die Anzahl der daraus resultierenden erwarteten Übertragungen (engl. expected hop count – EHC) sein. Die EHC-Metrik kann aus der Wortfehlerwahrscheinlichkeit berechnet werden. Die erwartete Anzahl Übertragungen der Ver-

[1] Gleichung (7.1) ist exakt. Es handelt sich jedoch um eine Schätzung, da das Ergebnis auf Einzelschätzungen beruht, die von der Anzahl der Stichproben abhängt und nur für unendlich viele Stichproben exakt sind (Abschnitt 4.4.1).

bindung l unter der Annahme unendlich vieler erlaubter Übertragungen ist

$$\begin{aligned}
\mathrm{E}\{N_{H,l}\} &= \sum_{a=1}^{\infty} a(1-\widehat{P}_{W,l})\widehat{P}_{W,l}^{a-1} \\
&= (1-\widehat{P}_{W,l})\sum_{a=1}^{\infty}(1-\widehat{P}_{W,l})a\widehat{P}_{W,l}^{a-1} \\
&= (1-\widehat{P}_{W,l})\frac{\partial}{\partial\widehat{P}_{W,l}}\sum_{a=0}^{\infty}\widehat{P}_{W,l}^{a} \\
&= (1-\widehat{P}_{W,l})\frac{1}{(1-\widehat{P}_{W,l})^2} \\
&= \frac{1}{1-\widehat{P}_{W,l}}\,. \qquad (7.3)
\end{aligned}$$

Da sowohl Rauschen wie auch Fading für die unterschiedlichen Verbindungen unabhängig voneinander sind, kann auch angenommen werden, dass $\mathrm{E}\{N_{H,l}\}$ für $l = 1 \ldots L$ statistisch unabhängig sind. Damit kann die erwartete Anzahl Hops eines Pfades für abschnittsweise Fehlerkontrolle

$$\mathrm{E}\{N_H\} = \sum_{l=1}^{L}\mathrm{E}\{N_{H,l}\} = \sum_{l=1}^{L}\frac{1}{1-\widehat{P}_{W,l}} \qquad (7.4)$$

aus (7.3) berechnet werden. Die EHC-Metrik minimiert nicht nur die Anzahl Hops für die Übertragung, sondern ist bei perfekten Schätzwerten der abschnittsweisen Wortfehlerwahrscheinlichkeiten $\widehat{P}_{W,l} = P_{W,l}$ gleichzeitig energieoptimal in dem Sinne, dass bei Verwendung dieser Metrik die durchschnittlich für die Übermittlung eines Pakets benötigte Energie nach [SWR98] minimiert wird.

Energieoptimalität der EHC-Metrik:
Sei $\boldsymbol{P}_{\mathrm{EHC}} \in \mathbb{P}$ ein Pfad für den gilt: $\mathrm{E}\{N_H|\boldsymbol{P}_{\mathrm{EHC}}\} \leq \mathrm{E}\{N_H|\boldsymbol{P}\}\ \forall\ \boldsymbol{P} \in \mathbb{P}$ (der entsprechend der EHC-Metrik ausgewählte Pfad). Bei konstanter Symbollänge ist die Sendeleistung eines Symbols proportional zur Sendeenergie. Bei konstanter und gleicher Sendeleistung γ_s' der Knoten ist die Energie für die Übertragung eines Paketes zwischen zwei Knoten

$$\gamma_h = M\gamma_s'\,, \qquad (7.5)$$

also für den gesamten Pfad

$$\gamma_h\,\mathrm{E}\{N_H|\boldsymbol{P}\} = M\gamma_s'\,\mathrm{E}\{N_H|\boldsymbol{P}\}\ . \qquad (7.6)$$

Damit ist also

$$M\gamma_s'\,\mathrm{E}\{N_H|\boldsymbol{P}_{\mathrm{EHC}}\} \leq M\gamma_s'\,\mathrm{E}\{N_H|\boldsymbol{P}\}\ \forall\ \boldsymbol{P} \in \mathbb{P}\,, \qquad (7.7)$$

der Energieverbrauch für die Übermittlung eines Pakets über den Pfad $\boldsymbol{P}_{\mathrm{EHC}}$ kleiner oder gleich dem Energieverbrauch bei Nutzung eines anderen Pfades.

Zum Vergleich der unterschiedlichen Metriken wird ein einfaches Netzwerk, wie in Abb. 7.1 dargestellt, simuliert. Für die vom Decodierer gelieferten Wahrscheinlichkeiten werden die in Kapitel 4 vorgestellten jeweils optimalen Decodierer verwendet und die Übermittlung von jeweils 500 000 Paketen simuliert. Die Verwendung der MLC-Metrik resultiert dabei immer in Verwendung des kürzesten Pfades $\boldsymbol{P}_{\mathrm{MLC}} = [L_{AE}, L_{EF}]$. Dieser Pfad wird natürlich auch bei Verwendung der anderen Routing-Metriken ausgewählt, wenn die Metrik ihn als am geeignetsten ausweist. Die durchschnittliche Anzahl Hops bei abschnittsweiser Fehlerkontrolle in Abhängigkeit von γ_s' ist in Abb. 7.3 dargestellt. Für große γ_s' konvergiert die Anzahl Hops für alle Metriken gegen 2 – der Länge von P_{MLC}. Wenn γ_s' groß genug und damit die Fehlerwahrscheinlichkeit klein genug ist, dann wird also auch bei Verwendung der zuverlässigkeitsbasierten Routing-Metriken praktisch immer der kürzeste Pfad gewählt.

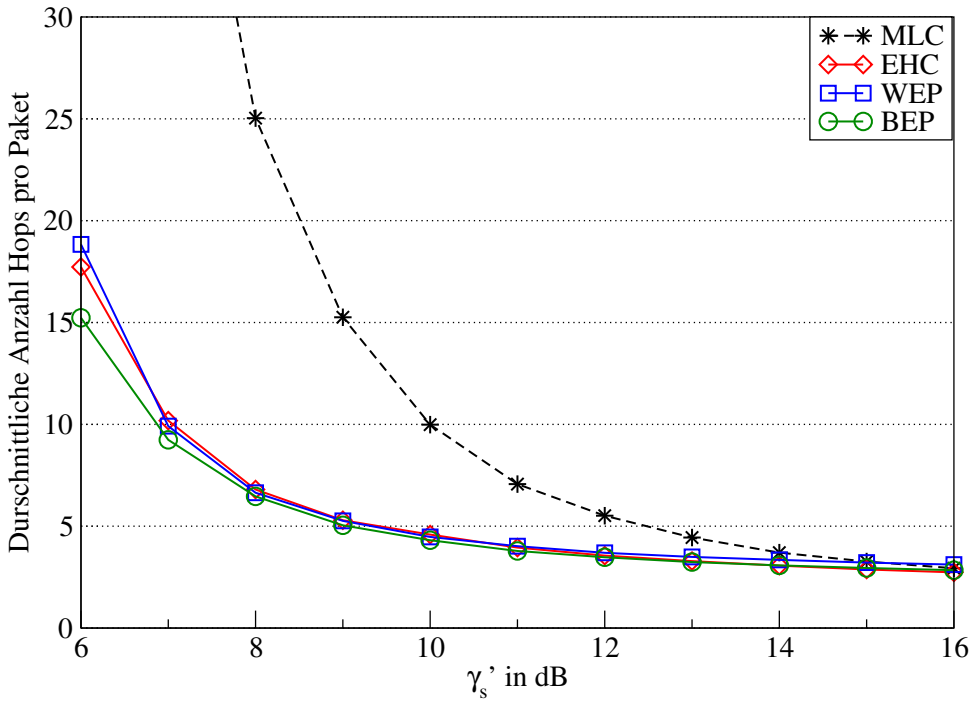

Abbildung 7.3: Durchschnittliche Anzahl Hops (Übertragungen auf der physikalischen Schicht) für die Übermittlung eines Pakets von Knoten A nach Knoten F bei Verwendung von abschnittsweiser Fehlerkontrolle.

Alle der zuverlässigkeitsbasierten Metriken erreichen für kleine und mittlere γ_s' bessere Ergebnisse als die MLC-Metrik. Interessant ist, dass die BEP-Metrik am besten abschneidet und nicht die EHC-Metrik. Dabei scheint die EHC-Metrik, entsprechend dem Modell, am besten geeignet, da sie nach (7.4) gerade den Pfad mit der kleinsten erwarteten Anzahl Hops wählt. Der Grund hierfür liegt in der Art und Weise wie die Fehlerwahrscheinlichkeiten ermittelt werden. In Abschnitt 4.5 wurde gezeigt, dass die Schätzung $\widehat{P}_B$ im Durchschnitt besser ist als die von $\widehat{P}_W$. Auf Grund der weniger zuverlässigen Schätzung erbringt die eigentlich schlechtere BEP-Metrik bessere Ergebnisse als WEP- und die daraus berechnete EHC-Metrik.

7.3 Routing-Metriken für durchgehende Fehlerkontrolle

Im Falle durchgehender Fehlerkontrolle stellt erst der Zielknoten fest, ob das Paket fehlerfrei übertragen wurde. Die Aufgabe der Sicherungsschicht wird also von der Netzwerkschicht oder der darüberliegenden Transportschicht wahrgenommen. Interpretiert man dies als einen schichtübergreifenden Ansatz nach Abschnitt 2.1.3, so stellt dieser Ansatz entweder die Zusammenlegung zweier Schichten (Fall D) oder der Bereitstellung von Informationen der physikalischen Schicht für die Netzwerkschicht unter Beibehaltung einer Transportschicht mit Fehlerkontrolle (Fall A) dar. Allerdings können auf allen Verbindungen des zurückgelegten Pfades Fehler verursacht worden sein. Im Falle eines Fehlers muss das Paket die gesamte Strecke (bestehend aus L Funkverbindungen) erneut übertragen werden, ein Fehler erhöht die Anzahl Hops also um L statt um 1.

Die BEP- und WEP-Metrik berechnen sich für durchgehende ebenso wie für abschnittsweise Fehlerkontrolle in (7.1) bzw. (7.2). Die EHC-Metrik ist jedoch unterschiedlich, da jede Neuanforderung L zusätzliche Hops (die Anzahl Verbindungen des gewählten Pfades) verursacht. Analog zu (7.3) kann mit Hilfe der Pfadwortfehlerwahrscheinlichkeit (7.2) und der Pfadlänge L eines Pfades als konstantem Vorfaktor auch in diesem Fall

$$\mathrm{E}\{N_H\} = \frac{L}{1 - \hat{P}_W} \tag{7.8}$$

berechnet werden. Die Simulation der Metriken für das Netzwerk nach Abb. 7.1 ist in Abb. 7.4 dargestellt. Grundsätzlich benötigen alle Metriken für die durchgehende Fehlerkontrolle höhere Sendeleistungen, um die gleiche durchschnittliche Anzahl Hops zu erreichen. Dies ist ein Anzeichen dafür, dass eine abschnittsweise Fehlerkontrolle für ein Netzwerk mit fehleranfälligen Verbindungen vorzuziehen ist. Wieder erreicht die BEP-Metrik, gefolgt von der WEP-Metrik, die besten Ergebnisse unter den zuverlässigkeitsbasierten Metriken. Alle zuverlässigkeitsbasierten Metriken erreichen für kleine und mittlere γ_s' niedrigere Hop-Counts. Wie erwartet konvergieren für große γ_s' alle Metriken gegen 2, da dies die Länge des kürzesten Pfads ist. Im Vergelich zwischen Abb. 7.3 und 7.4 ist die Überlegenheit der abschnittsweisen Fehlerkontrolle in der Anwesenheit von Übertragungsfehlern (kleine und mittlere γ_s') zu erkennen und zeigt, dass ein ARQ-Protokoll zur Vermeidung von Fehlern besser auf der Sicherungsschicht angesiedelt ist.

7.4 Zusammenfassung

In diesem Kapitel wurde erläutert, wie von einem soft-output Decodierer bereitgestellte Zuverlässigkeitsinformation genutzt werden kann, um Routing-Metriken zur Pfadsuche in der Netzwerkschicht zu berechnen. Zu diesem Zweck wurden drei sogenannte zuverlässigkeitsbasierte Routing-Metriken eingeführt. Mit abschnittsweise realisiertem ARQ-Protokoll zwischen Sender- und Empfänger in der physikalischen Schicht und der durchgehenden Fehlerkontrolle mit ARQ-Protokoll zwischen Quell- und Zielknoten wurden die Routing-Metriken für zwei unterschiedliche schichtübergreifende Ansätze simulativ untersucht. Für niedrige und mittlere Sendeleistungen resultierten die zuverlässigkeitsbasierten

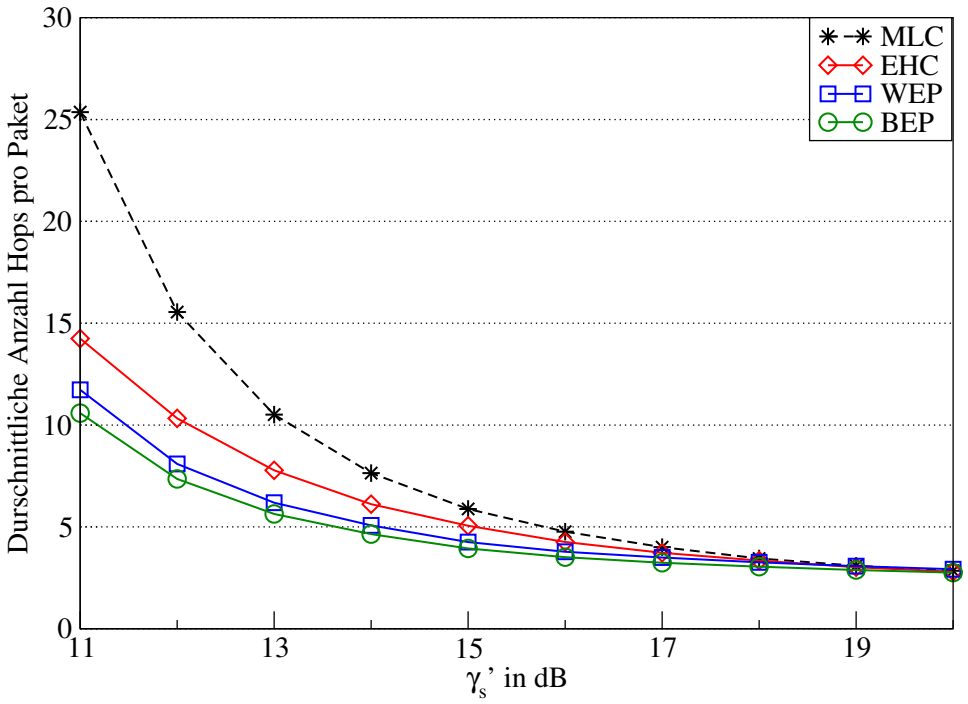

Abbildung 7.4: Durchschnittliche Anzahl Hops (Übertragungen auf der physikalischen Schicht) für die Übermittlung eines Pakets von Knoten A nach Knoten F bei Verwendung von durchgehender Fehlerkontrolle.

Metriken in deutlich weniger Übertragungen der physikalischen Schicht als die konventionelle Wahl des Pfads mit der kürzesten Anzahl Verbindungen zwischen Quell- und Zielknoten. Auf Grund der schon in Kapitel 4 erläuterten Varianz der Schätzung der Fehlerwahrscheinlichkeiten ergibt sich das beste Ergebnis für die BEP als Routing-Metrik.

Kapitel 8

Zuverlässigkeitsbasierte Ablaufplanung

Der Zugriff auf den Kanal wird von Medium-Access-Protokollen (MAC-Protokollen) geregelt. Da dies besonders wichtig ist, wenn mehrere unterschiedliche Nutzer oder Anwendungen auf den Kanal zugreifen wollen, werden in diesem Zusammenhang *Multiple-Access Techniken* (MA-Techniken) verwendet – daher wird MAC auch häufig als Multiple-Access interpretiert. Im Mobilfunk spielen konfliktfreie MAC-Protokolle mit dynamischer Ressourcen-Zuteilung eine wesentliche Rolle. Konfliktfrei bedeutet, dass eine Übertragung nicht durch ungewollte Interferenz zweier unterschiedlicher Nutzer behindert wird (Kollision) [RS90]. Dazu gehören Techniken wie FDMA, TDMA, aber auch CDMA (hier kommt es zwar zu Interferenz, die aber systeminhärent ist). Durch die verwendete MA-Technik wird die Kanalressource definiert. Bei den oben genannten MA-Techniken sind dies Frequenzbereiche, Zeitschlitze, oder Spreizcodes. Das MAC-Protokoll vergibt jede Kanalressource nur an einen Nutzer, so dass Interferenz vermieden wird.

Der Prozess der Vergabe der Kanalressourcen an Nutzer oder Anwendungen nennt sich *Scheduling* (dt. *Ablaufplanung*). Eine einfache Variante ist das sogenannte *Round-Robin-Scheduling* bei dem jeder Nutzer für einen vorher festgelegten, gleichen Zeitraum Zugriff auf eine Ressource, also z. B. einen Zeitschlitz oder einen Frequenzbereich bekommt. In [SRK03] wurde festgestellt, dass Information über den Zustand des Kanals gewinnbringend genutzt werden kann. Wird der Zugriff einzelner Nutzer auf den Kanal mit Berücksichtigung des Kanalzustandes, also z. B. die Übertragungsqualität eines Nutzers zu einem Zeitpunkt, geregelt, so sind stark erhöhte Datenraten möglich [Sch08b], [BY04]. Dieses Prinzip wird *opportunistic Scheduling* genannt und basiert im Wesentlichen darauf, dem Nutzer die Kanalressource zuzuweisen, der den „besten“ Kanal aufweist. So wird die Wahrscheinlichkeit von Übertragungsfehlern und damit auch Neuanforderungen vermieden. Andererseits verhindern die hohen Durchsätze auch Speicherüberläufe auf der Senderseite, da dort eintreffende Daten bis zu ihrer Übertragung (bzw. ihrem erfolgreichem Empfang) gespeichert werden müssen. Die Auswahl des geeigneten Nutzers bzw. der geeigneten Nutzer erfolgt über eine *Nutzermetrik*. Dies kann z. B. die Kanalkapazität der einzelnen Nutzer sein [Sch08b]. In diesem Kapitel sollen für diese Nutzermetrik die vom Kanaldecodierer zur Verfügung gestellte Zuverlässigkeitsinformation verwendet werden, um in einem TDMA-System einen geeigneten Nutzer auszuwählen.

8.1 Systemmodell

Angenommen wird im Weiteren ein Systemmodell wie in Abb. 8.1 dargestellt.

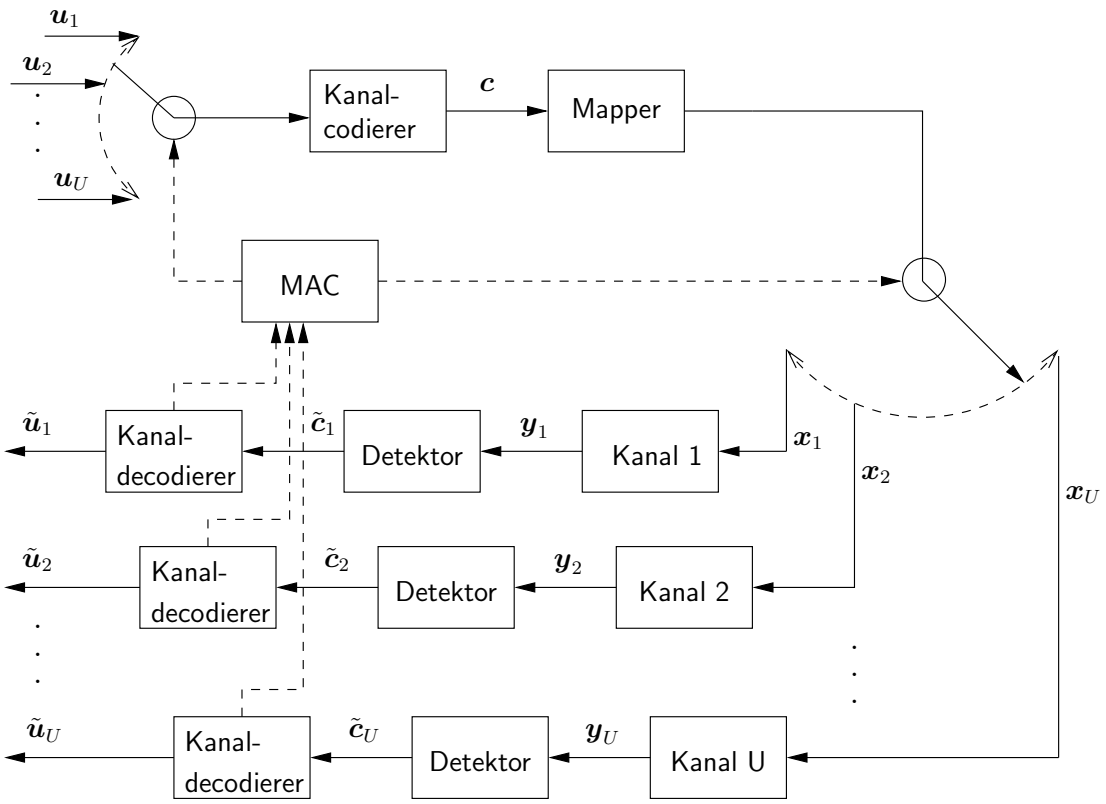

Abbildung 8.1: Systemmodell für ein Mehrnutzersystem mit U Mobilstationen (Nutzer) und Nutzerselektion durch ein Medium-Access-Protokoll an der Basisstation.

Es entspricht einer Erweiterung des Modells in Abb. 2.5 durch eine Parallelisierung von Kanal und Empfänger durch U vorhandene Nutzer. Es wird ein einfaches TDMA-System angenommen, das jeweils nur einen aktiven Nutzer pro Zeitschlitz hat. Dafür wählt das MAC-Protokoll vor jeder Übertragung einen der U Nutzer aus. Ein Anwendungsbeispiel für dieses Modell ist z. B. eine Basisstation, die in jedem Zeitschlitz an eine von mehreren Mobilstationen sendet. Es ist leicht nachvollziehbar, dass die Kanäle für die einzelnen Mobilstationen unterschiedlich sind. Der Kanalcode und die Modulation für die Übertragung zu den unterschiedlichen Mobilstationen können unterschiedlich sein, werden hier der Einfachheit halber aber als gleich angenommen (Faltungscode mit den Polynomen $(23_8, 35_8)$). Für jeden Zeitschlitz wird einer der U Nutzer ausgewählt, seine Infobits $\boldsymbol{u}_u$ codiert, und nach der Abbildung auf Modulationssymbole über den jeweiligen Kanal übertragen. Die Kanäle der Nutzer stellen dabei unabhängige Rayleigh-Fading-Kanäle mit einer maximalen normalisierten Doppler-Frequenz $f_{D,\max} T_s = 3,42 \cdot 10^{-5}$ dar. Dies entspricht einer Geschwindigkeit von 5 km/h bei einer Frequenz von 2 GHz und einer Symboldauer von $3,69\,\mu$s. Bei dieser Doppler-Frequenz verändert sich der Kanal eines Nutzers innerhalb eines Zeitschlitzes (entspricht einer Paketlänge von $M = 584$ Symbolen) nicht stark, aber

wahrnehmbar über mehrere Zeitschlitze. Damit kann der Nutzer mit dem „besten“ Kanal in jedem Zeitschlitz ein anderer sein.

Da immer nur aktuelle Information über den Kanal existiert, der im letzten Zeitschlitz genutzt wurde, wird alle fünf Zeitschlitze ein Paket an alle Nutzer geschickt. Diese Aufgabe kann in einem realen System auch ein Kontrollpaket mit Informationen für alle Mobilstationen übernehmen, so dass keine speziellen Trainingspakete notwendig sind. Nur alle fünf Zeitschlitze ist also die Metrik aller Nutzer aktuell. Trotzdem wird für jeden Zeitschlitz der Nutzer mit der besten Metrik ausgewählt, d. h., die hier ermittelten Ergebnisse beziehen sich auf ein System mit einer realistischen Annahme unvollständiger Kanalkenntnis. Modulation und Codierung eines Nutzers bleiben über die gesamte Simulationsdauer konstant.

8.2 Nutzermetriken

Die Nutzermetrik sollte so beschaffen sein, dass der jeweils „beste“ Nutzer ausgewählt wird. Was besser oder schlechter ist, hängt stark von den Vorgaben des Systems ab. Eine sinnvolle Möglichkeit ist, den Nutzer auszuwählen, der die höchste Datenrate erreicht. So wird der *Systemdurchsatz*, der Durchsatz (siehe (3.18)), der zwischen Sender und den ausgewählten Nutzern erzielt wird, gegenüber anderen Nutzerselektionsmechanismen maximiert.

Mit den in Kapitel 4 eingeführten weichen Schätzwerten $\widehat{P}_B$ und $\widehat{P}_W$ stehen zwei einfache Nutzermetriken zur Verfügung: Es wird der Nutzer mit der geringsten Fehlerwahrscheinlichkeit ausgewählt. Unter der naheliegenden Annahme, dass ein vorhandenes ARQ-Protokoll Pakete mit zu hoher Fehlerrate neu anfordert, maximiert diese Strategie also die Datenrate. Beide Nutzermetriken haben den Vorteil, dass sie die auf Grund von unterschiedliche Modulationsformaten und Kanalcodes unterschiedlichen Fehlerwahrscheinlichkeiten implizit mit berücksichtigen, da sie sich auf das Infowort beziehen.

Abb. 8.2 zeigt die Durchsatzeffizienz bei Verwendung der BEP als Nutzermetrik. Es wird deutlich, dass durch Nutzerselektion große Steigerungen des Durchsatzes möglich sind. Für zehn Nutzer wird die maximale Systemdatenrate (nahezu alle Pakete fehlerfrei decodiert) für $\gamma_s = 4\,\mathrm{dB}$ erreicht, während dies für nur einen Nutzer für $\gamma_s = 10\,\mathrm{dB}$ noch nicht möglich ist.

Eigentliches Kriterium zur Auswahl des Nutzers sollte nicht die fehlerfreie Übertragung sein, sondern die Menge der übertragenden Daten. Es sollte also gegenüber der bloßen Fehlerwahrscheinlichkeit eines Paketes berücksichtigt werden, wie viele Bits mit diesem Paket im fehlerfreien Fall übertragen werden. Um eine entsprechende Nutzermetrik zu ermitteln, kann die Wortfehlerwahrscheinlichkeit des u-ten Nutzers, $\widehat{P}_{W,u}$, mit Hilfe von (3.19) zu einer weiteren Nutzermetrik, der erwarteten Durchsatzeffizienz des Nutzers u

$$\mathrm{E}\left\{\eta_u\right\} = \frac{1}{1-\widehat{P}_{W,u}} R_{\mathrm{FEC}} \log_2 \left|\mathbb{M}_u\right| , \tag{8.1}$$

erweitert werden. Im Gegensatz zu (3.19) findet sich in (8.1) auch die Anzahl Bits pro Modulationssymbol ($\log_2 |\mathbb{M}_u|$) des u-ten Nutzers, so dass das Modulationsalphabet berücksichtigt wird.

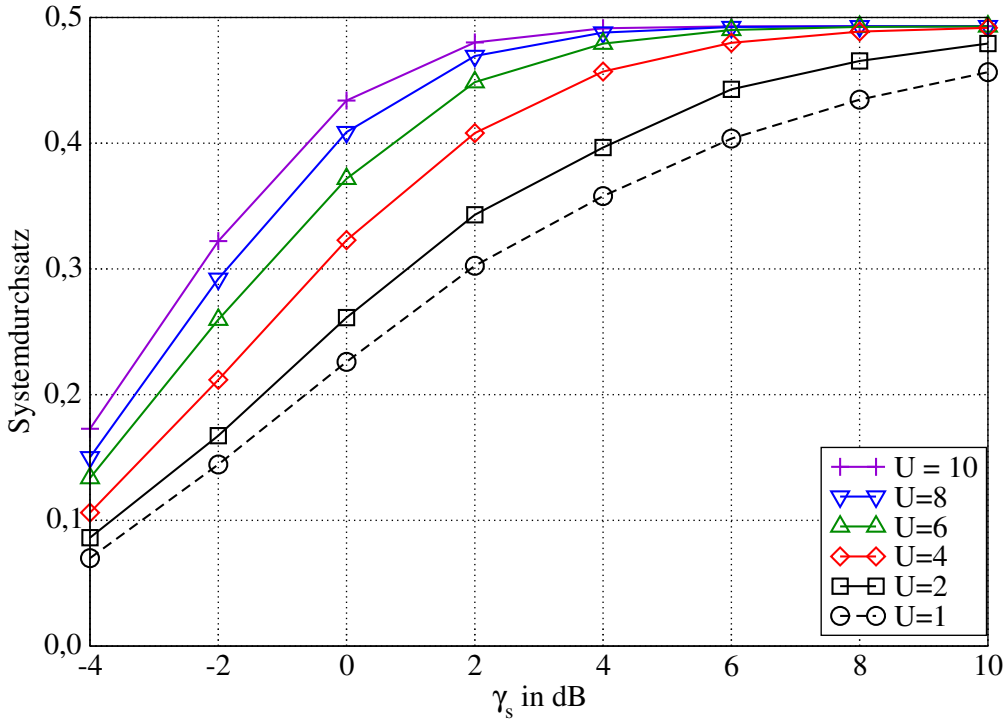

Abbildung 8.2: Einfluss zusätzlicher Nutzer auf den Durchsatz Verwendung von P_B als Nutzermetrik.

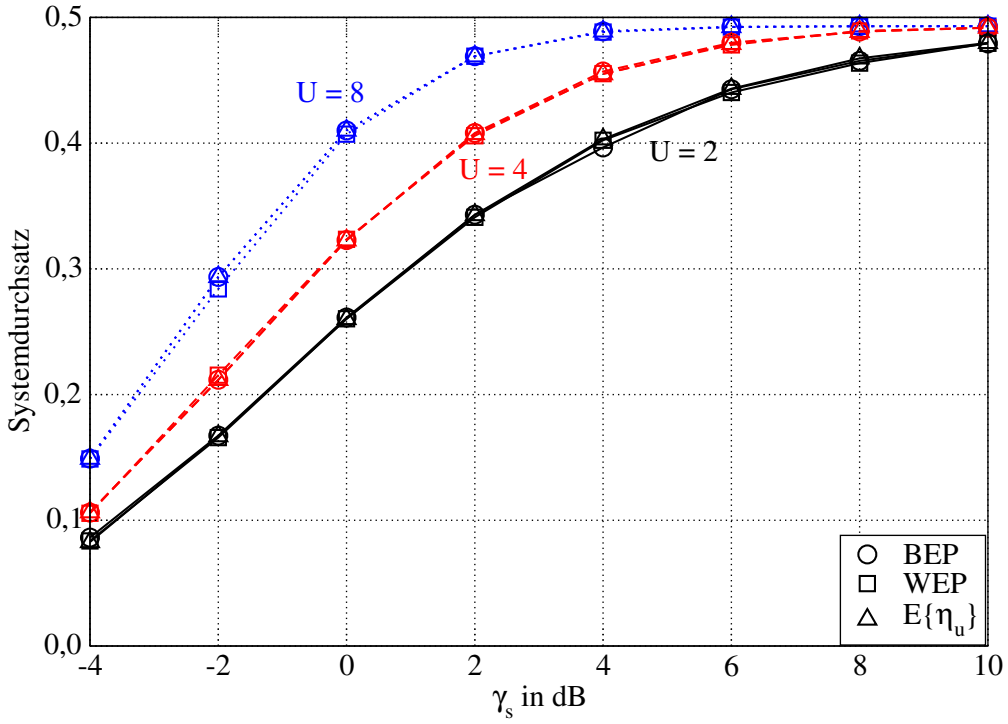

Abbildung 8.3: Durchsatz bei Verwendung der unterschiedlichen Nutzermetriken $P_{B,u}$, $P_{W,u}$ und $\mathrm{E}\{\eta_u\}$ bei BPSK-Übertragung.

In Abb. 8.3 ist die Durchsatzeffizienz für die unterschiedlichen Nutzermetriken dargestellt. Es zeigt sich, dass zwischen den unterschiedlichen Metriken keine signifikanten Unterschiede auftreten. Dies ist zu erwarten, wenn davon ausgegangen wird, dass alle Metriken zuverlässig sind und daher zuverlässig der beste Kanal ausgewählt wird.

Die Vorteile der aufwändigeren Nutzermetrik $\mathrm{E}\{\eta_u\}$ zeigen sich, wenn der Einfluss des Modulationsalphabets $\mathbb{M}$ zum Tragen kommt, also pro erfolgreich übermitteltem Paket je nach Nutzer unterschiedlich viele Informationsbits übertragen werden. In Abb. 8.4 sind die Durchsatzeffizienzen für Systeme mit vier und acht Nutzern angegeben. Dieses Mal verwenden alle ungeraden Nutzer ($u = 1, 3, 5, 7$) 16-QAM-Modulation, alle geraden Nutzer ($u = 2, 4, 6, 8$) BPSK-Modulation. Damit entspricht ein erfolgreich übertragenes Paket für einen Nutzer mit ungerader Nummer $K = 1164$ Infobits, für einen Nutzer mit geradem Index nur $K = 288$ Infobits.

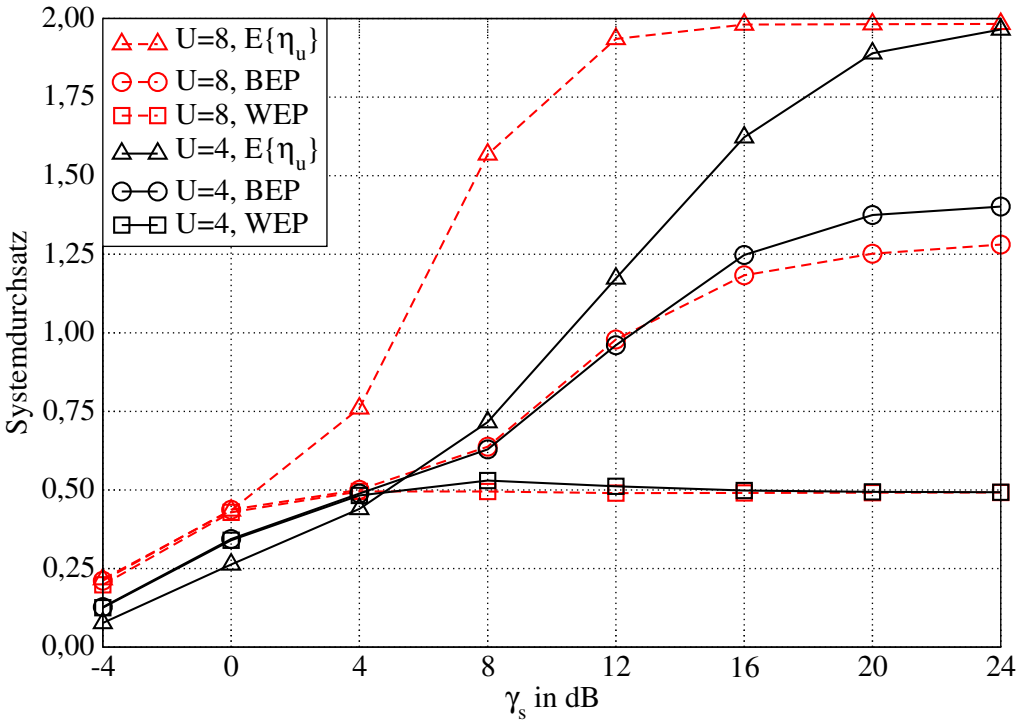

Abbildung 8.4: Durchsatz bei Verwendung der unterschiedlichen Nutzermetriken $P_{B,u}$, $P_{W,u}$ und $\mathrm{E}\{\eta_u\}$ in Verbindung mit BPSK- und 16-QAM-Modulation.

Die Nutzermetriken zeigen in diesem Szenario ein sehr unterschiedliches Verhalten. Zu erwarten ist die Überlegenheit von $\mathrm{E}\{\eta_u\}$, da nicht mehr nur die Wahrscheinlichkeit der Fehlerfreiheit eines Pakets entscheidend ist, sondern die Anzahl Bits, die im fehlerfreien Fall übermittelt werden. Bei Verwendung von $\widehat{P}_{B,u}$ und $\widehat{P}_{W,u}$ fließt diese nicht mit ein.

Da BPSK-Modulation auf Grund der kleineren Infowortlänge eine niedrigere Wortfehlerwahrscheinlichkeit als 16-QAM erreicht, werden bei Nutzung von $\widehat{P}_{W,u}$ als Metrik häufiger die Nutzer mit BPSK-Modulation ausgewählt. Dies geschieht auch dann, wenn für große γ_s die Übertragung für beide Modulationsformate quasi fehlerfrei ist. Damit wird erklärt, warum die Durchsatzeffizienz für diese Nutzermetrik für große γ_s nur bei

ca. $0,5$ liegt – der Durchsatzeffizienz für fehlerfreie BPSK-Übertragung bei einer Coderate von $1/2$. Dies zeigt, dass nur in wenigen Fällen ein Nutzer mit 16-QAM-Modulation ausgewählt wird.

Da die Bitfehlerwahrscheinlichkeit der Durchschnitt aller Bits im Infowort ist, ist dieser Nachteil bei $\widehat{P}_{B,u}$ als Nutzermetrik nicht zu erkennen – der Durchsatz steigt deutlich höher an, da gleichberechtigt zwischen BPSK- und 16-QAM-Nutzern ausgewählt wird.

Den höchsten Durchsatz zeigt die Nutzermetrik $\mathrm{E}\{\eta_u\}$, sobald die Symbolenergie groß genug ist, um die höhere Fehlerwahrscheinlichkeit der 16-QAM-Nutzer durch die höhere Anzahl übertragener Bits auszugleichen, werden verstärkt 16-QAM-Nutzer ausgewählt. Für große γ_s (kleine Fehlerwahrscheinlichkeit) werden schließlich nur noch 16-QAM-Nutzer ausgewählt und damit ein Durchsatz von $2,0$ erreicht. Nachteil dieser Metrik ist, dass die BPSK-Nutzer für große γ_s nicht mehr eingeplant werden.

8.3 Fair-opportunistische Ablaufplanung

In dem Beispiel des letzten Abschnitts sind die unterschiedlichen Nutzermetriken bei gleicher spektraler Effizienz des MCS gleich gut geeignet, um den geeignetsten Nutzer auszuwählen. Im Durchschnitt erreichen dann auch alle Nutzer die gleiche Übertragungsrate da die Kanäle aller Nutzer der gleichen Statistik für $|h_m|^2$ folgen. Bei unterschiedlicher spektraler Effizient der *Modulations- und Codierungsschemata* (MCS) der Nutzer erreicht $\mathrm{E}\{\eta_u\}$ als Nutzermetrik den höchsten Gesamtdurchsatz, allerdings werden schwache Nutzer gar nicht in der Ablaufplanung berücksichtigt. Dabei sind nun zwei Fälle zu unterscheiden:

1. Die Übertragungsqualität des Nutzers ist zu schlecht, Daten können nicht in ausreichender Qualität empfangen werden. In diesem Fall hat es keinen Sinn dem Nutzer Zeitschlitze für die Übertragung zuzuweisen, da in diesen nicht erfolgreich übertragen werden würde. Das MCS sollte zu einem robusteren gewechselt werden, um die Übertragungsqualität zu verbessern. Mit der daraus resultierenden Verbesserung der Übertragungsqualität würde der Nutzer automatisch bei der Ablaufplanung berücksichtigt.

2. Die Übertragungsqualität ist ausreichend, aber andere Nutzer haben eine bessere Übertragungsqualität. In diesem Fall muss der Nutzer – entgegen dem Interesse maximalen Systemdurchsatzes – bei der Ablaufplanung berücksichtigt werden. Dies ist insbesondere dann der Fall, wenn für sehr hohe Sendeleistungen die Fehlerwahrscheinlichkeiten gering sind und daher nur die Nutzer mit hochstufiger Modulation ausgewählt werden, da sie den erwarteten Durchsatz maximieren.

Nur der zweite Fall kann von der Ablaufplanung berücksichtigt werden – und soll mit Hilfe von Abb. 8.5 illustriert werden. Darin ist die Durchsatzeffizienz der Nutzer eins (BPSK) und zwei (16-QAM) für ein System mit $U = 8$ Nutzern dargestellt. Erkennbar ist, dass für niedrige γ_s nur der Nutzer mit BPSK-Modulation und für hohe γ_s nur der Nutzer mit 16-QAM ausgewählt wird. Für die anderen sechs Nutzer sind die Ergebnisse entsprechend.

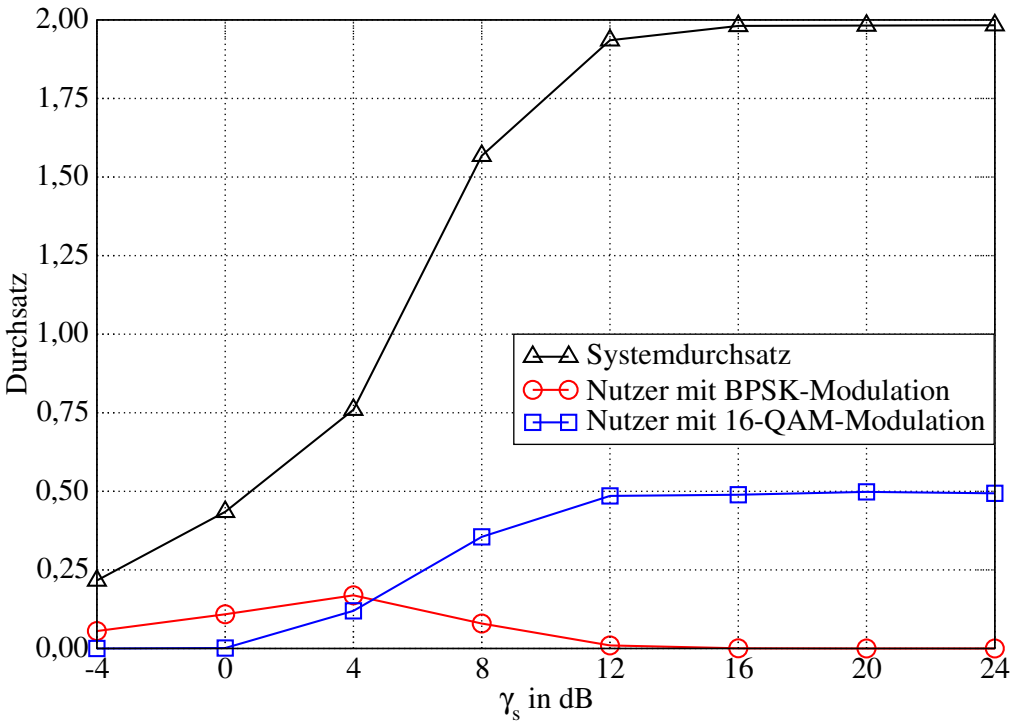

Abbildung 8.5: Nutzerdurchsatz bei Verwendung von $\mathrm{E}\{\eta_u\}$ als Nutzermetrik in Verbindung mit BPSK- und 16-QAM-Modulation.

Es soll angestrebt werden, den Systemdurchsatz fairer unter den Nutzern des Systems aufzuteilen, also auch schlechtere Nutzer einzuplanen. Dementsprechend muss die begrenzte Ressource, hier die Zeitschlitze, verteilt werden. Andererseits gehen die Vorteile von opportunistischer Ablaufplanung verloren, wenn zu viele Zeitschlitze schlechten Nutzern zugeteilt werden. Daher soll ein Nutzer nur eingeplant werden, wenn er einen erwarteten Mindestdurchsatz von $\eta_{\min}$ erreicht. Diese Vorgehensweise wird im Folgenden als *fair-opportunistisch* bezeichnet. Bei einer komplett fairen Ablaufplanung sollte jeder Nutzer einen durchschnittlichen Durchsatz von $\eta_u = \eta/U$ erreichen. Es soll weiter die vorher getroffene Annahme gelten, dass die Qualität der Nutzer alle fünf Zeitschlitze ermittelt wird. Daher soll der Ablaufplan für $T = 5$ Zeitschlitze (ZS) aufgestellt werden. Im Weiteren stellt $\eta_{u,s}$ jeweils den Solldurchsatz der nächsten T Zeitschlitze des Nutzers u und $\eta_{u,p}$ den erwarteten (geplanten) Durchsatz des Nutzers u auf Grund der zugewiesen Zeitschlitze dar. Der Algorithmus erwartet, dass die Nutzer absteigend entsprechend ihrer erwarteten Rate $\mathrm{E}\{\eta_u\}$ vorliegen, und folgt für die Zuteilung der U Nutzer zu den T Zeitschlitzen drei einfachen Überlegungen für die verbliebenen freien Zeitschlitze.

1. Wenn ein Nutzer ein genügend hohen erwarteten Durchsatz $\mathrm{E}\{\eta_u\} \geq \eta_{\min} = 0,1$ hat und sein geplanter Durchsatz von $\eta_{u,p}$ noch nicht den Solldurchsatz von $\eta_{u,s}$ übersteigt, so wird ihm ein Zeitschlitz zugewiesen und mit dem nächsten Nutzer fortgefahren. Der hier angenommenen Wert von $\eta_{\min} = 0{,}1$ entspricht BPSK-Übertragung bei einer Coderate von $R_u = 0{,}5$ und WEP $P_{W,u} = 0{,}8$, d. h. Nutzer mit einer

Fehlerwahrscheinlichkeit von 0,8 werden nicht mehr eingeplant.[1]

2. Sind die Durchsatzanforderungen aller Nutzer mit $\mathrm{E}\{\eta_u\} > \eta_{\min}$ erfüllt, so werden unter diesen weiter Zeitschlitze zugewiesen (`overRate=true`) ohne zu beachten, ob $\eta_{u,s} \leq \eta_{u,p}$ schon erfüllt ist.

3. Erfüllt kein Nutzer die Anforderung $\mathrm{E}\{\eta_u\} > \eta_{\min}$ (`noGoodUsers=true`), so werden die Zeitschlitze beginnend mit dem besten Nutzer vergeben, auch wenn dieser nicht den Mindestdurchsatz erreicht.

Die formelle Darstellung dieses Algorithmus findet sich auf der folgenden Seite. Dieser Algorithmus führt dazu, dass dem besten Nutzer im Gegensatz zur reinen opportunistischen Ablaufplanung oft nur ein Zeitschlitz zugeteilt wird, da ein Zeitschlitz genügt, um den geplanten Durchsatz $\eta_{u,p}$ größer oder gleich dem angestrebten Durchsatz $\eta_{u,s}$ werden zu lassen. Hierdurch werden also für die Ablaufplanung der kommenden T Zeitschlitze die bis zu T besten Nutzer berücksichtigt, sofern ihr erwarteter Durchsatz $\mathrm{E}\{\eta_u\}$ groß genug ist. Ergebnisse der Nutzerdurchsätze η_u sind, wiederum exemplarisch, für die ersten beiden Nutzer, in Abb. 8.6 dargestellt.

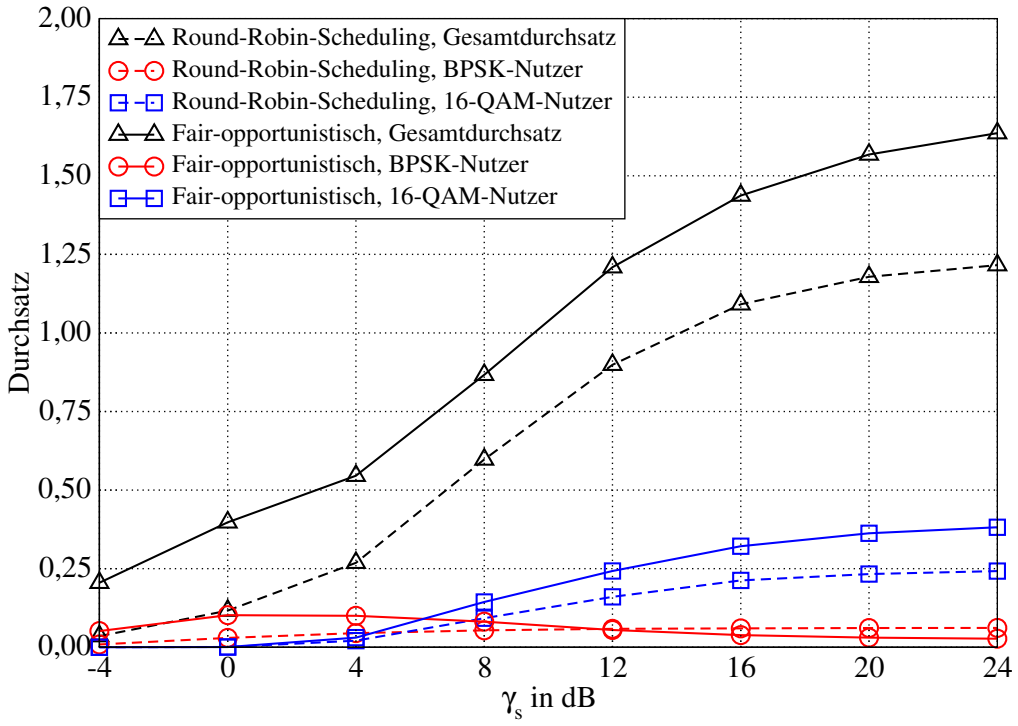

Abbildung 8.6: Durchsatz einzelner Nutzer sowie Systemdurchsatz bei Ablaufplanung mit Berücksichtigung von Fairness als Nutzermetrik in Verbindung mit BPSK- und 16-QAM-Modulation.

[1] Die Werte können je nach System und angestrebter Verteilung des Durchsatzes beliebig gewählt werden. Hier liegt die Überlegung zu Grunde, dass ein Nutzer in dem robustesten zur Verfügung stehenden MCS, z. B. BPSK bei $R_{\mathrm{FEC}} = 1/2$, eingeplant bzw. eine entsprechende Datenrate ereichen soll. Offensichtlich kann dies jedoch nicht für Fehlerraten nahe 1 funktionieren.

Vergleicht man die Ergebnisse aus Abb. 8.5 mit denen aus Abb. 8.6, so stellt man fest, dass der Gesamtdurchsatz zwar für hohe γ_s niedriger ist, dafür aber die BPSK-Nutzer einen Teil des Gesamtdurchsatzes zugeteilt bekommen, der Algorithmus also fairer verteilt. Zum Vergleich ist der Durchsatz bei Verwendung eines anderen fairen Ablaufplaners, dem Round-Robin-Scheduler, angegeben. Da dieser die Zeitschlitze reihum ohne Rücksicht auf die Übertragungsqualität verteilt, erreichen die BPSK-Nutzer für große γ_s einen höheren Durchsatz. Der Gesamtdurchsatz wie auch der Durchsatz der 16-QAM-Nutzer bleibt aber für große Sendeenergien deutlich hinter dem des hier vorgeschlagenen fair-opportunistischen Ablaufplaners zurück.

Der hier vorgeschlagene Algorithmus für fair-opportunistische Ablaufplanung schafft keine komplette Raten-Fairness, d. h. die Nutzer erreichen nicht alle die gleiche Durchsatzeffizienz, und auch keine faire Verteilung der Zeitschlitze wie der Round-Robin-Scheduler, sondern die Nutzer mit 16-QAM-Modulation erreichen durchschnittlich einen höheren Durchsatz. Dieses Problem kann umgangen werden, indem die Anzahl Zeitschlitze, für die die Ablaufplanung vorgenommen wird, vergrößert wird. Dann würden bei dem vorliegendem Algorithmus bei erfüllter Durchsatzanforderung der 16-QAM-Nutzer vermehrt BPSK-Nutzern mehrere Zeitschlitz zugewiesen bekommen und dadurch ihr geplanter Durchsatz $\eta_{u,p}$ bis auf den Solldurchsatz $\eta_{u,s}$ erhöht werden. Über den Solldurchsatz ist auch eine unterschiedliche Verteilung des Gesamtdurchsatzes zwischen den Nutzern möglich, wenn z. B. Nutzer unterschiedlicher Tarife unterschiedliche Datenraten erhalten sollen.

8.4 Zusammenfassung

In diesem Kapitel wurde erläutert, wie die vom Kanaldecodierer bereitgestellte Zuverlässigkeitsinformation einzelner Nutzer in einem Mehrnutzersystem zum Zwecke der Ablaufplanung genutzt werden. Da die Zuverlässigkeitsinformation genutzt werden kann, um die Übertragungsqualität des jeweiligen Nutzers zu schätzen, ist mit den aus ihr berechneten Nutzermetriken eine Maximierung der Durchsatzeffizienz mit Hilfe von opportunistischer Ablaufplanung möglich. Da opportunistische Ablaufplanung dazu führt, dass stets nur der beste Nutzer ausgewählt wird, wurde ein Algorithmus für fair-opportunistische Ablaufplanung vorgeschlagen. Dieser verteilt die zur Verfügung stehenden Kanalressource gleichmäßiger, ohne die Idee der opportunistischen Ablaufplanung mit ihren großen Durchsatzsteigerungen komplett aufzugeben.

Die hier durchgeführten Betrachtungen und Simulationsergebnisse beziehen sich auf ein TDMA-System, sind aber auf jedes System übertragbar, bei der eine Kanalressource zu einem Zeitpunkt einem Nutzer allein zugewiesen wird.

Algorithmus für fair-opportunistische Vergabe von T Zeitschlitzen bei U Nutzern

Sortieren der Reihenfolge der Nutzer, so dass $u = 1...U$ der Größe von $\mathrm{E}\{\eta_u\}$ entspricht
$\eta_{u,p} := 0 \quad \forall \quad u$
$u = 1$
$t = 1$
overRate=false
noGoodUsers=false
while $t < T$
    if (overRate == false)
      if ($\mathrm{E}\{\eta_u\} \geq \eta_{\min}$ and $\eta_{u,p} < \eta_{u,s}$)
        $ZS[t] := u$
        $t := t + 1$
        $\eta_{u,p} := \eta_{u,p} + \mathrm{E}\{\eta_u\}$
      else
        $u := u + 1$
        if ($u > U$)
          $u := 1$
          overRate:= true
    else
      if (noGoodUser == false)
        if ($\mathrm{E}\{\eta_u\} \geq \eta_{\min}$)
          $ZS[t] := u$
          $t := t + 1$
          $\eta_{u,p} := \eta_{u,p} + \mathrm{E}\{\eta_u\}$
        else
          $u := u + 1$
          if ($u > U$)
            $u := 1$
            noGoodUsers:= true
      else
        $ZS[t] := u$
        $t := t + 1$
        $\eta_{u,p} := \eta_{u,p} + \mathrm{E}\{\eta_u\}$

Kapitel 9

Zuverlässigkeitsinformation in IDM

Interleave-Division Multiplexing (IDM) ist eine spezielle Form von Code-Division Multiplexing (CDM). Einzeln codierte Datenströme $\boldsymbol{u}_d$ werden darin durch unterschiedliche Interleaver getrennt. Die Codierung spielt für die Trennung der Datenströme eine geringere Rolle als z. B. in DS-CDM Systemen. Das Unterscheidungsmerkmal der D parallelen Datenströme sind die unterschiedliche Interleaver, daher können die Codes der einzelnen Datenströme gleich sein. In Interleave-Division Multiple Access (IDMA) Systemen, also in Mehrnutzersystemen, kam dieses Prinzip erstmals als Chip-Interleaving für CDMA zur Anwendung [CT97]. Der Name IDMA stammt aus [LWLL02], worin es als eigenständiges CDMA-Verfahren systematisiert wurde.

Im Unterschied zu vielen anderen CDM-Verfahren wird bei IDM nicht mehr zwischen Codierung zur Nutzertrennung und Codierung zur Fehlerkorrektur unterschieden – der Interleaver ist nach FEC-Codierung und Spreizung angeordnet, nicht wie sonst häufig dazwischen. FEC-Codierung und Spreizung kann also auch zu einem Codierer zusammengefasst werden. Die codierten (bzw. gespreizten) Datenströme stellen Codeworte dar, werden aber häufig als *Layer* bezeichnet, die Codesymbole als *Chips*. Für die in diesem Kapitel folgenden Betrachtungen ist es ohne Bedeutung, ob es sich um ein Ein- oder Mehrnutzersystem handelt – es wird im Allgemeinen von IDM gesprochen.

Auf Grund einiger Eigenschaften eignen sich auf IDM basierende Systeme gut für schichtübergreifende Ansätze. Die Ideen hierzu sind im Zusammenhang mit Systemvorschlägen (z. B. in [FHS05a] und [FHS05b]) vorgestellt worden. Vor allem die niederratige Codierung in IDM-Systemen sowie die verwendeten iterativen Empfänger sind vorteilhaft für schichtübergreifende Ansätze.

- Durch Verwendung niederratiger Codes mit chip-weisem Interleaving ergeben sich trotz langer Codewörter/Interleaver kurze Infowortlängen. Kurze Infowortlängen bringen eine hohe Flexibilität im Zeitbereich mit sich, ermöglichen schnelle Adaption an Kanalbedingungen und Einsatz von differenzierten ARQ-Verfahren.

- Die Verwendung unterschiedlicher Layer mit verschieden hohen Fehlerwahrscheinlichkeiten ermöglicht ungleichen Fehlerschutz für unterschiedliche Nutzer und/oder Anwendungen.

- Durch niederratigere Codes ergibt sich eine feinere Granularität, d. h., es ergeben

sich mehr Datenströme mit geringerer Übertragungsrate. Dies ermöglicht eine feine Adaption an den Kanal [Sch08a] sowie eine flexible Ressourcenverteilung.

- Die verwendeten iterativen Empfänger liefern gute Zuverlässigkeitsinformation, die die Schätzung von BEP und WEP nach Kapitel 4 ermöglicht.

Diese IDM-spezifischen Eigenschaften sollen in diesem Kapitel für einen zweistufigen schichtübergreifenden Ansatz auf Basis eines IDM-Systems genutzt werden. Die Ergebnisse dieses Kapitels wurden in [HSF08] und [KFH08] vorab veröffentlicht.

9.1 Systemmodell

Ein Übertragungsmodell für die Verwendung von IDM ist in Abb. 9.1 dargestellt. Die D parallelen Datenströme $\boldsymbol{u}_d$ werden codiert und interleavt. Für den üblicherweise verwendeten iterativen Empfänger ergibt sich eine bessere Leistungsfähigkeit, wenn die codierten Bits gescrambelt werden, d. h., jedes zweite Bit invertiert wird. Als Modulationsformat wird BPSK verwendet, so dass sich aus den Codebtis $\boldsymbol{c}_d$ durch BPSK-Mapping die Symbole $\boldsymbol{z}_d$ ergeben. Diese werden zur besseren spektralen Effizienz mit layer-spezifischen Amplituden a_d und Phasendrehungen φ_d versehen. Die Phasendrehungen sind hier jeweils gleichverteilt in $[-\pi/2, +\pi/2)$. Anschließend werden die so entstandenen Symbole $x_{d,m}$ für die Übertragung über den Kanal zu dem Signal

$$x_m = \sum_{d=1}^{D} z_{d,m} a_d \, \mathrm{e}^{j\varphi_d} \tag{9.1}$$

überlagert. Am Empfänger können die Infobits der einzelnen Datenströme mit Hilfe eines iterativen Empfängers [LL04] (dargestellt in Anhang B) aus den Empfangsabtastwerten $\boldsymbol{y}$ gewonnen werden. Abb. 9.1 stellt im Wesentlichen ein Modell nach Abb 2.5 dar, allerdings sind Codierung und Decodierung parallelisiert.

Üblicherweise ist die Coderate in IDM-Systemen niedrig. Dies ist z. B. durch serielle Verkettung eines Faltungscodes mit einem niederratigem Wiederholungscode zu erreichen [HSF08]. Durch die niederratigen Codes wird eine Verteilung der zur Verfügung stehenden Gesamtdatenrate mit feiner Granularität ermöglicht. Dies ist vorteilhaft bei allen die Ressourcen-Verteilung betreffenden Problemen. Werden die verschiedenen Layer unterschiedlichen Nutzern zugeteilt, so wird aus IDM Interleave-Division Multiple Access (IDMA). Ein weitere Variante, bei der einzelne Nutzer verschieden viele Layer verwenden, ist Multi-Layer IDMA (ML-IDMA). Im Weiteren wird ein IDM bzw. ML-IDMA System betrachtet (für Übertragung über einen AWGN-Kanal ergibt sich dabei kein Unterschied). Die Länge der uncodierten Datenblöcke beträgt $K = 320$. Der verwendete Code ist eine Verkettung eines terminierten, rekursiven systematischen Faltungscodes mit den Polynomen $(1_8, 7_8/5_8)$ ($R_{\mathrm{FEC}} \approx 1/2$) und einem anschließendem Wiederholungscode der Rate $R_{SPR} = 1/4$. Die Gesamtcoderate eines Layers ergibt sich also zu

$$R = R_{\mathrm{FEC}} R_{\mathrm{SPR}} = \frac{1}{2} \cdot \frac{1}{4} = \frac{1}{8} \,, \tag{9.2}$$

die Codewortlänge bzw. Anzahl Chips pro Layer ist $N = M = 2560$. Die verwendeten Interleaver sind Pseudo-Zufallsinterleaver.

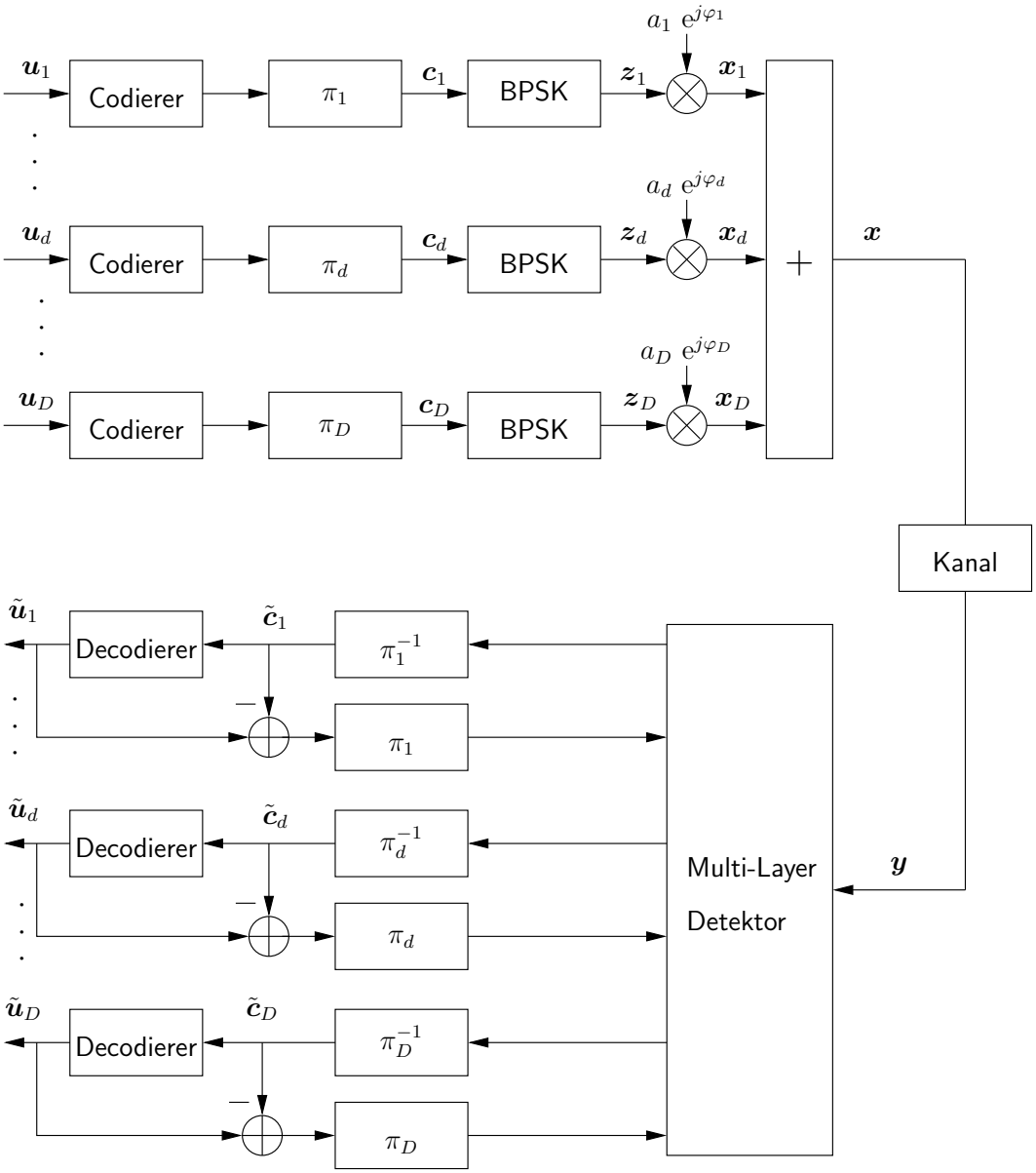

Abbildung 9.1: Auf IDM basierendes Übertragungssystem mit iterativem Empfänger.

9.2 Layer-weises ARQ in IDM

Werden ARQ-Verfahren mit inkrementeller Redundanz verwendet, so ist häufig von Interesse, die unzuverlässigen bzw. fehlerverursachenden Abschnitte eines Wortes zu kennen. Bei der Verwendung von IDM ist durch die D unterschiedlichen Layer eine Segmentierung in einzelne Codeworte vorgegeben. Diese haben durch unterschiedliche Leistungszuweisung (oder im Falle von IDMA unterschiedlichen Kanälen) unterschiedliche Fehlerwahrscheinlichkeiten. Aber auch bei gleicher Leistung und nur einem Nutzer ist es möglich, dass einer der Layer von Bitfehlern betroffen ist, andere aber nicht. Insbesondere bei der Anwendung von ungleichem Fehlerschutz für die verschiedenen Layer [HW07] sind deren

Fehlerwahrscheinlichkeiten unterschiedlich. Sinnvollerweise sollten diese Layer also von einem ARQ-Protokoll unterschiedlich behandelt werden. Einerseits sollten nur die Layer neu angefordert werden, die die erforderliche Dienstgüte nicht erreichen. Andererseits sollte der ungleiche Fehlerschutz der Layer berücksichtigt werden, was z. B. durch zuverlässigkeitsbasierte Neuanforderungskriterien erreicht werden kann. Durch layerweise ARQ-Strategien ergeben sich vier Vorteile

- *Layer-spezifische Zuverlässigkeiten*: Jeder Layer kann ein eigenes Neuanforderungskriterium (zuverlässigkeitsbasiertes oder codebasiertes Neuanforderungskriterium) haben. Dies ermöglicht erst die sinnvolle Nutzung von ungleichem Fehlerschutz.

- *Höhere Datenraten*: Wenn Layer mit geringer Zuverlässigkeit bzw. mit Fehlern akzeptiert werden, wird die Anzahl der Neuanforderungen reduziert und daher die effektive Datenrate/der Durchsatz erhöht.

- *Selektive Neuanforderungen*: Selbst wenn alle Layer dieselben Zuverlässigkeitsanforderungen erfüllen müssen, wird layer-weises ARQ einen erhöhten Gesamtdurchsatz erreichen. Anstatt alle Daten neu zu übertragen, wird nur Redundanz für den Layer angefordert, der die Anforderungen nicht erfüllt.

- *Layer-weise inkrementelle Redundanz*: Die Art der inkrementellen Redundanz (z. B. Diversity-Combining oder Packet-Combining) kann für die Layer unterschiedlich sein, je nachdem welches Ziel erreicht werden soll.

Die obigen Effekte auf die Durchsatzeffizienz sind in Abb. 9.2 dargestellt. Die Simulationsergebnisse zeigen die Durchsatzeffizienz

$$\eta_{\mathrm{IDM}} = \frac{\text{empfangene Infobits}}{\text{übertragende Chips}} \tag{9.3}$$

eines IDM-Systems, die im Wesentlichen (3.19) für $T_{RT} = 0$ entspricht. Die Anzahl der übertragenden Bits hängt allerdings von der Anzahl der überlagerten Layer ab, während die Anzahl der Chips (Codebits) gleich bleibt. Die maximal erreichbare Durchsatzeffizienz ist also $DR = 0{,}375$, bei Verwendung eines CRC Codes werden P Bits der Infobits für die Prüfbits aufgewendet.

Als konventioneller Ansatz in Abb. 9.2 wird das Verfahren bezeichnet, bei dem der gesamte Bitstrom gemeinsam mit dem CRC-Code codiert und anschließend auf die einzelnen Layer verteilt wird. Dieser Ansatz ist natürlich nur möglich, wenn alle beteiligten Layer einem Nutzer zugeordnet sind. Bei dem layer-weisen Ansatz wird jeder Layer einzeln codiert. Obwohl damit DP statt nur P CRC-Bits aufgewendet werden, ergibt sich eine höhere Durchsatzeffizienz, da jeweils nur die von Fehlern betroffenen Layer neu angefordert werden. Für $\gamma_s > 4\,\mathrm{dB}$ machen sich diese zusätzlichen CRC-Bits bemerkbar: während der konventionelle Ansatz eine Durchsatzeffizienz von $R(DK-P)/K \approx 0{,}366$ erreicht, ist der Durchsatz für den layer-weisen Ansatz auf $RD(K-P)/K \approx 0{,}347$ begrenzt. Wie sich die CRC-Bits auf die Durchsatzeffizienz auswirken, hängt natürlich stark von der Anzahl der überlagerten Layer D und deren Infobits K ab.

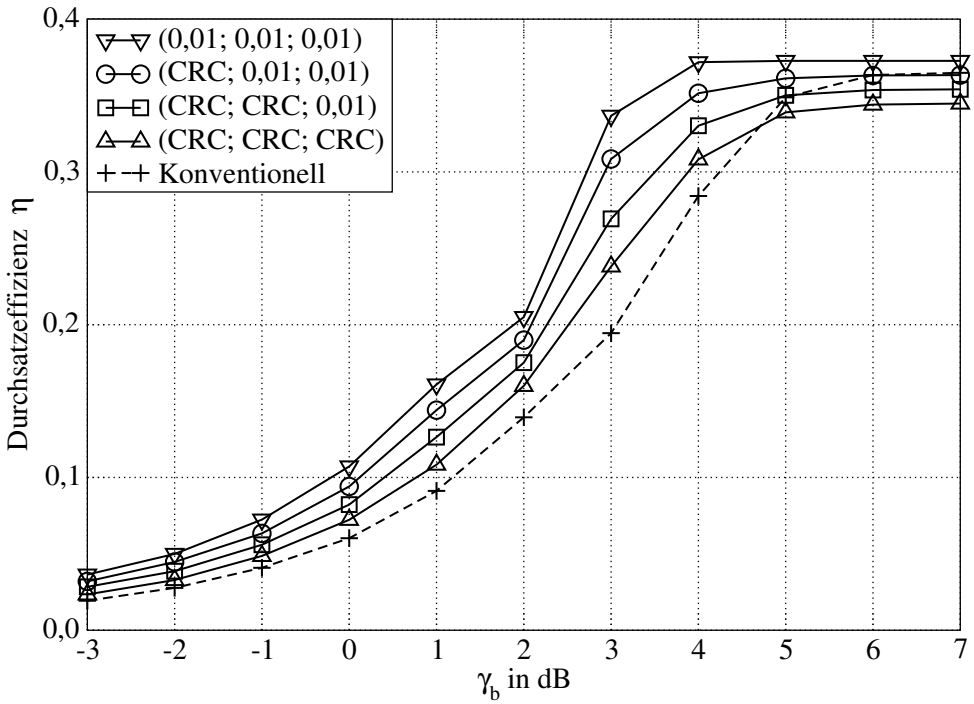

Abbildung 9.2: Layer-weises ARQ für ein IDM-System mit $D = 3$ Layern und unterschiedlichen Neuanforderungskriterien.

Bezieht man nun das Prinzip von zuverlässigkeitsbasierten Neuanforderungskriterien mit ein, so können die zusätzlichen CRC-Bits bei layer-weisem ARQ durch zuverlässigkeitsbasierte Neuanforderungskriterien (Kapitel 5) vermieden werden. Voraussetzung ist ungleicher Fehlerschutz bzw. Anwendungen, deren Dienstgüteanforderung nicht-fehlerfreie Decodierung zulässt. Ebenfalls in Abb. 9.2 ist daher die Durchsatzeffizienz für layerweises ARQ dargestellt, wenn ein bis drei der drei Layer eine maximale Bitfehlerwahrscheinlichkeit von $P_{b,t} = 0{,}01$ erlauben (siehe Abschnitt 5.1.3 – natürlich kann die Bitfehlerwahrscheinlichkeit auch für jeden Layer unterschiedlich festgelegt werden). Durch die Verwendung des zuverlässigkeitsbasierten Neuanforderungskriteriums erhöht sich der Durchsatz für alle γ_b. Gegenüber dem konventionellen Ansatz sind in diesem Beispiel Durchsatzsteigerungen von bis zu 70 % (für $\gamma_b = 3\,\mathrm{dB}$) möglich.

9.3 Zuverlässigkeitsbasierte Layer-Verteilung in IDM

Die für IDM verwendeten iterativen Empfänger funktionieren besser, wenn den unterschiedlichen überlagerten Layern über ihre Amplitude a_d unterschiedliche Leistungen zugewiesen bekommen, [HSF08], [PLL06], [WK07] und [Sch08a]. Interessant im Zusammenhang mit schichtübergreifenden Ansätzen ist die Tatsache, dass die Layer mit geringerer Leistung dann in höheren Fehlerwahrscheinlichkeiten resultieren als die mit höherer Leistung. In [WK07] wird eine Optimierungsstrategie für die Leistungsverteilung unter der Nebenbedingung maximaler BEPs für jeweils Gruppen von Layern vorgeschlagen. Eine andere, einfachere Strategie wird in [HW07] verfolgt. Darin wird die Gesamtheit der Layer

zum Zwecke von ungleichem Fehlerschutz in zwei Gruppen aufgeteilt, jeweils eine Gruppe mit größerer und einer mit kleinerer Amplitude.

Hier soll dieser gruppenweise, ungleiche Fehlerschutz in einem schichtübergreifenden Ansatz genutzt werden, um einerseits die Vorteile der ungleichen Leistungsverteilung für den Empfänger zu nutzen und andererseits Nachteile für die Anwendungen zu vermeiden, die über die schwächeren Layer kommunizieren. Das Übertragungssystem entspricht dabei dem in Abb. 9.1 dargestellten. Die D Layer werden in zwei Gruppen gleicher Größe $D_1 = D_2 = D/2$ eingeteilt. Wie in [HW07] wird den beiden Gruppen entsprechend einem Parameter ρ über die Amplituden $a_1 = 10^{-\frac{\rho}{20}}$ bzw. $a_2 = \sqrt{2 - a_1^2} \geq a_1$ eine Leistung zugewiesen. In Anlehnung an [HW07] wird ρ in dB angegeben. Die Layer der zweiten Gruppe (mit der größeren Amplitude a_2) erreichen eine niedrigere BEP als die der Gruppe mit der geringeren Leistung. Gleichverteilte Leistung wird durch $\rho = 0\,\mathrm{dB}$, also $a_1 = a_2$ erreicht.

Es wird nun angenommen, dass Anwendungen zweier unterschiedlicher Dienstgüteklassen unterstützt werden sollen. Eine der beiden toleriert eine BEP von bis zu 10^{-2} (z. B. Sprachübertragung oder Videoübertragung), die andere strebt quasi-fehlerfreie Übertragung an (z. B. Datenübertragung). Dementsprechend werden der ersten Anwendung die Layer mit der geringen Leistung zugewiesen und der zweiten die Layer hoher Leistung. Entsprechend den Dienstgüteanforderungen werden die einzelnen Layer durch ARQ-Protokolle (layer-weises ARQ, Abschnitt 9.2) geschützt. Die Layer der ersten Gruppe verwenden ein zuverlässigkeitsbasiertes Neuanforderungskriterium mit einem Schwellenwert $P_{b,t} = 10^{-2}$ während die der zweiten Gruppe einen 24-Bit CRC-Code zur Fehlererkennung verwenden. Dieser Punkt ist wesentlich, da die Layer mit geringer Leistung auf Grund der hohen BEP mit einem fehlererkennenden Code als Neuanforderungskriterium nur einen sehr geringen Durchsatz erreichen würden – ungleicher Fehlerschutz ist nur dann sinnvoll, wenn alle beteiligten Schichten entsprechend darauf eingestellt sind.

Die sich ergebende BEP für $\rho = 0{,}8\,\mathrm{dB}$ (ohne Verwendung von ARQ) ist in Abb. 9.3 dargestellt. Für höhere γ_b ist die BEP auch für die Layer niedriger Leistung deutlich unter 10^{-2}. Es wird also für die Layer der ersten Gruppe Leistung „verschwendet“, die besser dafür genutzt werde sollte, die Layer der zweiten Gruppe mit weniger Fehlern zu übertragen. Als weitere Anpassungsmaßnahme soll daher abhängig von der Gesamtleistung der Parameter ρ angeglichen werden. Unter der Annahme, dass für die Layer mit geringer Leistung eine Leistung ausreichend ist, die eine durchschnittliche BEP von weniger als 10^{-2} erreicht, werden die folgenden einfachen Regeln für die Leistungsverteilung angewendet:

- Der Parameter ρ wird so gewählt, dass die Layer niedriger Leistung in einer BEP kleiner als 10^{-2} resultieren.
- Der restliche Anteil der Leistung wird der zweiten Gruppe von Layern zugeordnet, so dass diese eine niedrigere BEP erreichen.
- Ist es für die Layer niedriger Leistung für $\rho > 0$ nicht möglich, eine BEP kleiner als 10^{-2} zu erreichen, so wird ρ zu 0 gesetzt, also eine gleichverteilte Leistung verwendet.

Abb. 9.4 zeigt die Durchsatzeffizienz dreier unterschiedlicher Strategien: gleichverteilte Leistungsverteilung, ungleiche Leistungsverteilung und ungleiche Leistungsverteilung mit

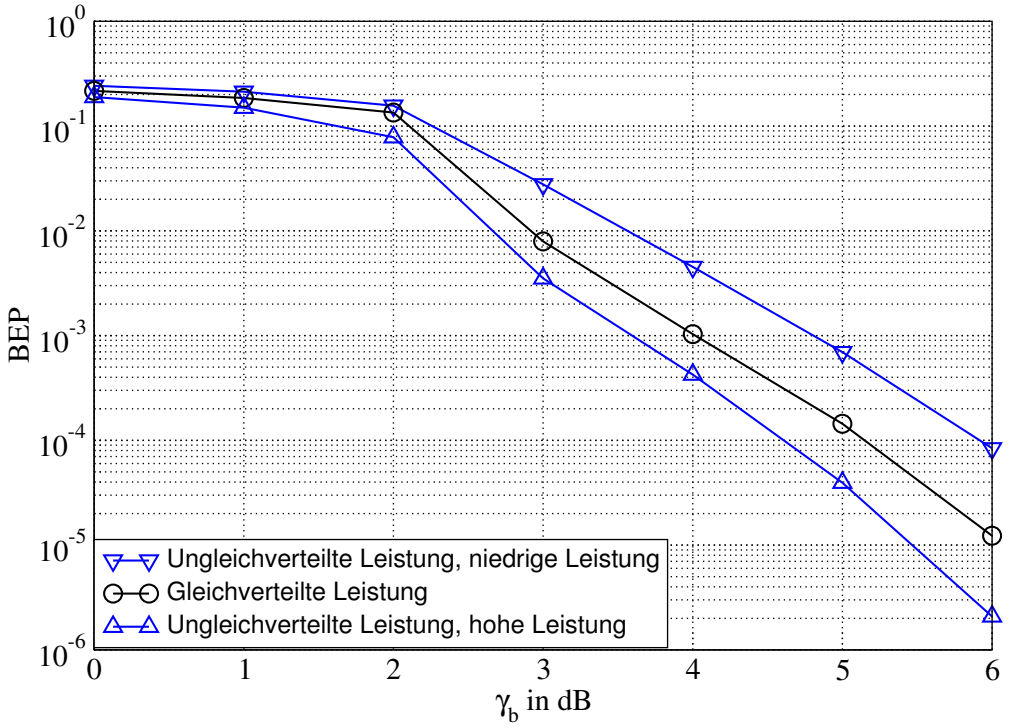

Abbildung 9.3: BEP für layer-weise Leistungsverteilung ($\rho = 0{,}8\,\mathrm{dB}$) im Vergleich zu gleichverteilter Leistung.

zuverlässigkeitsbasiertem Neuanforderungskriterium für die Layer niedriger Leistung. Simuliert wurden $D = 14$ Layer, bei Verwendung ungleicher Leistungsverteilung jeweils mit den angegebenen Werten für ρ. Zu erkennen ist, dass eine ungleiche Leistungsverteilung ohne entsprechende ARQ-Strategie einen *geringeren* Durchsatz als gleichverteilte Leistung erbringt. Der Grund hierfür ist die hohe BEP und die damit einhergehenden Neuanforderungen für die Layer niedriger Leistung. Die oben beschriebene ARQ-Strategie mit gemischten Neuanforderungskriterien erzielt dagegen eine Durchsatzsteigerung und erreicht schon für $\gamma_b = 5\,\mathrm{dB}$ den maximal möglichen Durchsatz von $\eta_{\max} = (7 \times 296 + 7 \times 320)/2560 \approx 1{,}68$. Die relative Leistungssteigerung ist besonders groß für kleine γ_b, erstreckt sich aber über den gesamten Simulationsbereich.

9.4 Zusammenfassung

In diesem Kapitel wurden zuverlässigkeitsbasierte, schichtübergreifende Ansätze beschrieben, die speziell auf die Charakteristika von IDM(A)-Systemen ausgelegt sind. Es wurde ausgenutzt, dass die einzelnen Layer eines IDM-Systems bzgl. Leistungsverteilung, Fehlerwahrscheinlichkeit, Neuanforderung und Zuordnung zu Anwendungen unabhängig voneinander gehandhabt werden können. Durch layer-weise ARQ-Protokolle können große Durchsatzsteigerungen erzielt werden.

Ferner können die negativen Konsequenzen einer ungleichen Leistungsverteilung (niedrigerer Durchsatz) durch die schichtübergreifende Koordination von Dienstgüteanforde-

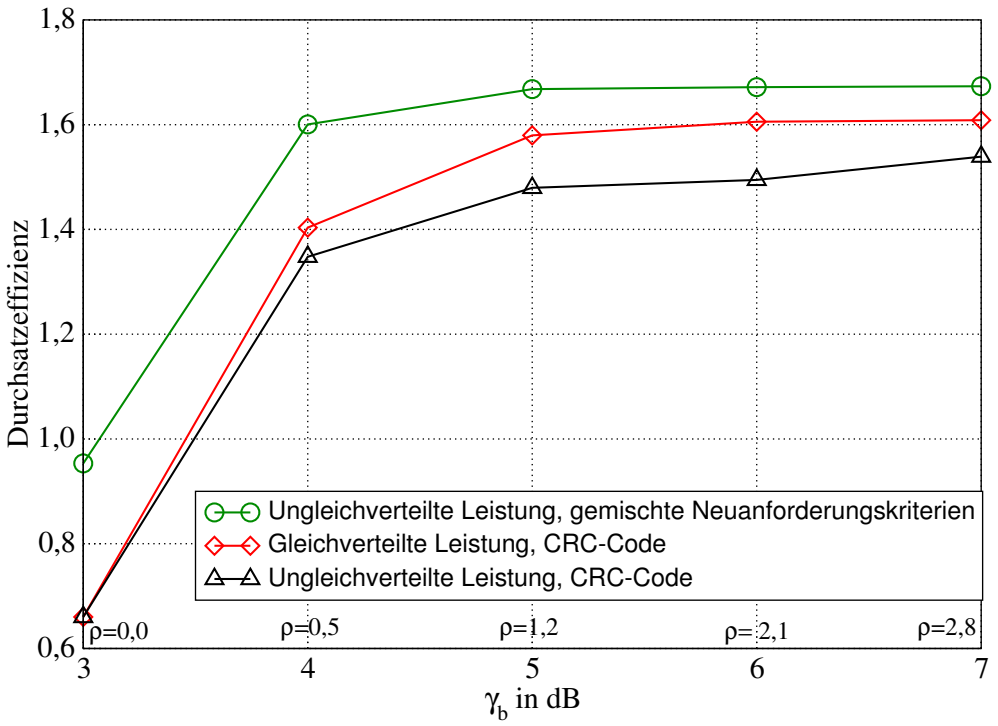

Abbildung 9.4: Durchsatzeffizienz für konventionelles IDM (gleichverteilte Leistung, CRC-Code), IDM mit ungleichverteilter Leistung und IDM mit ungleichverteilter Leistung sowie leistungsabhängigem Neuanforderungskriterium.

rungen neutralisiert werden. Ein relativ einfacher schichtübergreifende Ansatz verbindet die Anforderungen der Anwendung an die BEP mit layer-weisem ARQ und einer auf iterative Empfänger zugeschnittenen Leistungsverteilung.

Die vorgestellten Strategien können bei Verwendung geeigneter Empfänger auf DS-CDM(A)-Systeme übertragen werden.

Kapitel 10

Zusammenfassung und Ausblick

Zusammenfassung

In dieser Arbeit wurde untersucht, wie die von soft-output Decodierern zur Verfügung gestellte Zuverlässigkeitsinformation im schichtübergreifenden Systementwurf genutzt werden kann. Das vorgestellte Systemmodell umfasst eine große Menge an denkbaren Übertragungstechniken. Die einzige Voraussetzung ist das Vorhandensein von soft-output Decodierung am Empfänger. Ein großer Vorteil dieser Strategie ist daher vor allem seine weite Anwendbarkeit. Da die Zuverlässigkeitsinformation am Decodiererausgang bereits sämtliche Einflüsse der physikalischen Schicht mit Auswirkung auf die Fehlerwahrscheinlichkeit enthält, ist diese Strategie unabhängig von Parametern wie z. B. Modulationsformat, Kanaleinflüssen und Kanalcodierung.

Ein wesentlicher Aspekt ist die Art der Nutzung der Zuverlässigkeitsinformation: Nicht die Zuverlässigkeitsinformation selbst wird genutzt, sondern zwei daraus berechnete, für digitale Übertragungssysteme ausgesprochen relevante Kennzahlen, die Bitfehlerwahrscheinlichkeit und die Wortfehlerwahrscheinlichkeit. Durch die Verwendung einer einzelnen Kennzahl ist die Menge der zusätzlich zwischen den Schichten ausgetauschten Daten relativ gering. Für verschiedene, häufig verwendete Familien von Codes wurde in Kapitel 4 erläutert, wie Bit- und Wortfehlerwahrscheinlichkeit eines decodierten Wortes berechnet werden können und welche Vor- und Nachteile daraus entstehen. Neben der Vorstellung der für diesen Zweck idealen (und im Sinne der Fehlerminimierung) optimalen Decodierer wurden approximative Verfahren zur Berechnung der Wortfehlerwahrscheinlichkeit hergeleitet. Damit wurde ein einfaches und vielfältiges Modell der physikalischen Schicht für den schichtübergreifenden Systementwurf geschaffen, das es ermöglicht, die für die höheren Schichten wesentlichen Einflüsse auf die Dienstgüte zu schätzen und zu beeinflussen. Ferner wurde die Idee der weichen Monte-Carlo-Simulationen auf die Schätzung von Wortfehlerwahrscheinlichkeiten erweitert und die Überlegenheit der weichen gegenüber der harten Simulation für diesen Fall nachgewiesen.

Die in den Kapiteln 5–9 vorgestellten schichtübergreifenden Ansätze lassen sich in zwei unterschiedliche Gruppen unterteilen. Erstere basiert darauf, dass die Fehlerwahrscheinlichkeit eines decodierten Wortes gegeben das Empfangswort nach der Decodierung bekannt ist. Anwendungen finden sich dort, wo die Fehlerwahrscheinlichkeit eines einzelnen

Wortes oder der Bits darin eine Rolle spielen. Die Fehlerwahrscheinlichkeit kann entsprechend Kapitel 5 als Neuanforderungskriterium verwendet werden, so dass die Fehlerwahrscheinlichkeit nach Decodierung und ARQ-Protokoll unter dem vorgegebenen Schwellenwert für Neuanforderungen liegt. Dadurch wird in Abhängigkeit von der Anwendung ein Abtausch zwischen Datenrate und Verzögerung einerseits und verbleibenden Fehlern nach Ende der Fehlerkontrolle andererseits erreicht. In Abhängigkeit von der Infowortlänge und der maximal zulässigen Fehlerwahrscheinlichkeit sind dadurch große Steigerungen der Datenrate möglich. Die Ergebnisse zeigen, dass die Bitfehlerwahrscheinlichkeit das geeignetere Neuanforderungskriterium ist. In Kapitel 6 dient die Kenntnis der Fehlerwahrscheinlichkeit eines empfangenen oder decodierten Infoworts dazu, auf eine Decodierung in einem Relay bzw. weitere Iteration einer iterativen Struktur zu verzichten, wenn die Fehlerwahrscheinlichkeit niedrig genug ist. Durch diese güteorientierte Decodierung, die den Verzicht auf unnötige Decodierernutzung ermöglicht, können Energieverbrauch und Verzögerung vermindert werden.

Die zweite Gruppe von Anwendungen interpretiert die vom Decodierer gegebene Fehlerwahrscheinlichkeit als einen Kurzzeitmittelwert (bestehend aus einer oder wenigen Stichproben) im Sinne der weichen Schätzung der Fehlerwahrscheinlichkeit. Damit kann die Übertragungsqualität der Verbindung zwischen einem Sender und einem Empfänger aus der Decodierung gewonnen werden, wenn Kanal, Modulation und Codierung sich nicht ändern bzw. lange genug konstant oder ähnlich bleiben. In Kapitel 7 wurden mit Hilfe der so ermittelten Fehlerwahrscheinlichkeit Routing-Metriken berechnet und so der Pfad mit den wenigsten Übertragungen für die Übermittlung eines Pakets durch ein Netzwerk gefunden. Gegenüber der Wahl des kürzesten Weges ist damit in Funknetzwerken eine Reduzierung der Sendeleistung bzw. Verminderung der Verzögerung möglich. Für die Entscheidung, an welche Mobilstationen eine Basisstation als nächstes sendet, werden die Fehlerwahrscheinlichkeiten in Kapitel 8 verwendet. In Abhängigkeit der Übertragungsqualität wird der Nutzer ausgewählt, der die geringste Fehlerwahrscheinlichkeit aufweist. Damit kann die Durchsatzeffizienz des Systems, auch unter der Annahme unvollständiger Kanalkenntnis, stark gesteigert werden. Aus der Fehlerwahrscheinlichkeit kann auch die erwartete Durchsatzeffizienz berechnet werden, um Nutzer mit unterschiedlichen Coderaten und Modulationsformaten vergleichen zu können. Die erwartete Durchsatzeffizienz ist immer dann der Bit- bzw. Wortfehlerwahrscheinlichkeit als Nutzermetrik überlegen, wenn die spektrale Effizienz der einzelnen Nutzer unterschiedlich ist. Neben der Nutzung für opportunistische Nutzerselektion bietet die aus der Zuverlässigkeitsinformation berechnete Durchsatzeffizienz eines Nutzers auch die Möglichkeit eine Mischung aus fairer und rein opportunistischer Ablaufplanung zu realisieren.

In Kapitel 9 werden exemplarisch für ein IDM-basiertes System verschiedene die Zuverlässigkeitsinformation verwendende Ansätze kombiniert, um IDM-spezifische Eigenschaften gezielt zu nutzen. Wieder wurden höhere Datenraten erzielt als ohne den schichtübergreifenden Ansatz.

Zusammenfassend lässt sich sagen, dass durch die Kenntnis der vom Decodierer bereitgestellten Fehlerwahrscheinlichkeiten auf höheren Schichten fast durchgängig positive Effekte erzielt werden konnten.

Aus symbolweiser Zuverlässigkeitsinformation kann die Wortfehlerwahrscheinlichkeit

nur approximativ berechnet werden, die Bitfehlerwahrscheinlichkeit jedoch exakt. Da soft-output Decodierer häufig, z. B. für iterative Decodierung, darauf ausgelegt sind, symbolweise Zuverlässigkeitsinformation auszugeben, ist die Bitfehlerwahrscheinlichkeit bezüglich der Auswahl der Decodierer flexibler. Ferner weist die Mittelwertschätzung der Bitfehlerwahrscheinlichkeit eine geringere Varianz als die der Wortfehlerwahrscheinlichkeit auf, ist also genauer. Da sich die Bitfehlerwahrscheinlichkeit auch bei der Verwendung als Neuanforderungskriterium als geeigneter erwies, kann es als die geeignetere Kennzahl zum schichtübergreifenden Systementwurf betrachtet werden, sofern eine Anwendung nicht explizit die Wortfehlerwahrscheinlichkeit benötigt wie z. B. die Berechnung der erwarteten Durchsatzeffizienz in Kapitel 8.

Ausblick

Grundsätzlich gibt es sehr viel mehr Anwendungsmöglichkeiten für die Schätzung der Übertragungsparameter als die in dieser Arbeit vorgestellten. Immer wenn die Fehlerwahrscheinlichkeit der Übertragung für ein Protokoll, einen Dienst oder eine Anwendung von Bedeutung ist, gibt es die Möglichkeit, mit den hier vorgestellten Ansätzen Einfluss zu nehmen.

Auch ein radikalerer schichtübergreifender Ansatz ist denkbar. Statt auf den höheren Schichten nur Kennzahlen und harte Schätzwerte zu verwenden, könnten die weichen Schätzwerte direkt Anwendung finden. Damit ist z. B. iterative Quellen- und Kanalcodierung denkbar [Tho07], bei der die Zuverlässigkeitsinformation der Infobits auf der Anwendungsschicht genutzt wird. Nachteil einer solchen Strategie ist allerdings, dass im gesamten Protokollstapel die Verarbeitung von weichen Schätzwerten im Gegensatz zu harten Schätzungen der Bits unterstützt werden muss. Damit wären einerseits ein höherer Speicher und Übertragungsaufwand und andererseits stark modifizierte Protokolle und Dienste notwendig. Diese wären mit den jetzt üblichen überwiegend unvereinbar. Der Ansatz dieser Arbeit bietet durch die Nutzung der Zuverlässigkeitsinformation in kompakten Kennzahlen einen hinsichtlich dieser Problematik realisierbaren Kompromiss. Die unterschiedlichen Anwendungsbeispiele, z. B. zuverlässigkeitsbasiertes Routing und zuverlässigkeitsbasierte Neuanforderungskriterien, können kombiniert werden, um noch höhere Gewinne zu erzielen.

Anhang A

Notation

Übersetzungen englischsprachiger Fachbegriffe

abschnittsweise Fehlerkontrolle	link-wise error control
automatische Neuanforderung	automatic repeat request
Dienst	service
Dienstgüte	quality of service (QoS)
Durchsatzeffizienz	throughput efficiency
erwartungstreu	unbiased
Einheit (im OSI-Schichmodell)	entity
Funknetzwerk	wireless network
unabhängig und identisch verteilt	independent and identically distributed (i. i. d.)
güteorientierte Decodierung	quality-oriented decoding
Pfaddämpungsexponent	path loss exponent
Pfadsuche	routing
Leistungsverteilung	power allocation
schichtübergreifend	cross-layer
schichtübergreifender Entwurf	cross-layer design
Schicht	layer
Vorwärtsfehlerschutz	forward error correction (FEC)
Wiederholungsanfrage	automatic repeat request
Neuanforderungsanfrage	retransmission request

Verwendung von Schriften

x	skalare Variable
$\boldsymbol{x}$	vektorielle Variable
$\boldsymbol{X}$	Matrix-Variable
$\mathbb{X}$	Menge
$\mathrm{x}(\cdot)$	Funktion mit Argument $\cdot$

x	Konstante

Konstanten

e	Eulersche Zahl ($\approx 2,71828$)
j	imaginäre Einheit ($j^2 = -1$)
π	Kreiszahl ($\approx 3,14159$)

Operatoren

$(\bullet)^{-1}$	Matrixinversion
$(\bullet)^{\mathsf{T}}$	Transponierte von $\bullet$

Formelzeichen

a_d	Amplitude des d-ten Layers in IDM-System
A	Anzahl Übertragungen für ARQ
B	Anzahl Infobits pro Sendesymbol
c_n	n-tes Codesymbol eines Codeworts
$\boldsymbol{c}$	Codewort
$c_{k,p}^{i,j}$	p-te Codebithypothese des Trellisübergangs von S_{k-1}^i nach S_k^j
$c(D)$	Codewortpolynom
f_D	Dopplerfrequenz
$f_{D,\max}$	maximale Dopplerfrequenz
f_{D_n}	Doppplerfrequenz der n-ten Mehrwegekomponente
f_e	Fehlerfaktor
$\mathbb{F}_n$	Menge aller mit Variable-Node n verbundenen Check-Nodes
$g(D)$	Generatorpolynom
$\boldsymbol{G}$	Generatormatrix
h_m	Kanalkoeffizient des m-ten übertragenen Symbols
$\mathbf{h}$	Kanalkoeffizienten
K	Infowortlänge
L	Gedächtnislänge von Faltungscodes bzw. Pfadlänge im Routing
M	Länge eines Übertragungsblocks
n_m	Rauschabstastwert zum Zeitpunkt m
$\boldsymbol{n}$	Sequenz der Rauschabstastwerte eines Übertragungsblocks
n_{PL}	Pfaddämpfungsexponent
N	Codewortlänge
P_B	Infobitfehlerwahrscheinlichkeit
$\hat{P}_B$	harter Schätzert der Infobitfehlerwahrscheinlichkeit
$\hat{P}_B$	weicher Schätzert der Infobitfehlerwahrscheinlichkeit

$\widehat{P}_b$	weicher Schätzert der Infobitfehlerwahrscheinlichkeit eines Wortes
P_W	Infowortfehlerwahrscheinlichkeit
$\hat{P}_W$	harter Schätzwert der Infowortfehlerwahrscheinlichkeit
$\widehat{P}_W$	weicher Schätzwert der Infowortfehlerwahrscheinlichkeit
$\boldsymbol{P}$	Pfad durch ein Netzwerk
$\boldsymbol{P}_{\mathrm{MLC}}$	Pfad mit der kleinsten Anzahl Links
R	Coderate
R_{FEC}	Coderate des FEC-Codes
R_{SPR}	Coderate des Spreiz-/Wiederholungscodes
S_k^i	i-ter Trelliszustand zum Zeitpunkt k
u_k	k-tes Infobit
$u_k^{i,j}$	Infobithypothese des Trellisübergangs von S_{k-1}^i nach S_k^j
$u(D)$	Infowortpolynom
$\boldsymbol{u}$	Infowort
$\boldsymbol{u}_d$	Infowort des d-ten Layers in IDM-System
T_A	Übertragungszeit eines Pakets über den Funkkanal
$T_k^{i,j}$	Übergang von Zustand S_{k-1}^i auf Zustand S_k^j in einem Trellisdiagramm
T_{RT}	Round-Trip-Time
T_s	Dauer eines Kanalsymbols
T_V	Verarbeitungszeit eines Pakets am Empfänger
$\mathbb{V}_p$	Menge aller mit Check-Node p verbundenen Variable-Nodes
$\boldsymbol{x}$	Sendesymbole
$\boldsymbol{x}_d$	d-ter Layer in IDM-System
x_m	m-tes Symbol in $\boldsymbol{x}$
$\boldsymbol{y}$	Empfangsabtastwerte für $\boldsymbol{x}$
y_m	m-ter Abtastwert der Empfangsfolge
$\alpha(S_k^i)$	Zustandsmetrik von S_k^i aus Vorwärtsrekursion
$\beta(S_k^i)$	Zustandsmetrik von S_k^i aus Rückwärtsrekursion
$\gamma(T_k^{i,j})$	Zweigmetrik des Übergangs $T_k^{i,j}$
γ_s	Durchschnittliches Symbol- zu Rauschleistungsverhältnis pro gesendetem Symbol
γ_s'	Durchschnittliches Symbol- zu Rauschleistungsverhältnis erhöht um Kanalverlust
γ_b	Durchschnittliches Symbol- zu Rauschleistungsverhältnis pro Infobit
η	Durchsatzeffizienz
$\eta_{\min}$	Mindestdurchsatzeffizienz
η_u	Durchsatzeffizienz des u-ten Nutzers
$\eta_{u,p}$	geplante Durchsatzeffizienz des u-ten Nutzers
$\eta_{u,s}$	Soll-Durchsatzeffizienz des u-ten Nutzers
Θ_n	Phase der n-ten Mehrwegekomponente
$\boldsymbol{\Lambda}$	extrinsische Information
μ_B	Mittelwert der Bitfehlerschätzung
μ_W	Mittelwert der Wortfehlerschätzung

$\mu^i_{v_n \to f_p}$	Nachrichten des n-ten Variable-Nodes an den p-ten Check-Node
$\mu^i_{f_p \to v_n}$	Nachricht des p-ten der Check-Nodes an den n-ten Variable-Node
π_d	Interleaver des d-ten Layers in IDM
φ_d	Phasendrehung des d-ten Layers in IDM-System
σ_n^2	Varianz des AWGN
ω_s	Anzahl Einsen pro Spalte von $\boldsymbol{H}$
ω_z	Anzahl Einsen pro Zeile von $\boldsymbol{H}$

Funktionen und funktionelle Operatoren

$\oplus$	Addition mod 2 bzw. Addition in $\mathbb{F}_2$
$\mathrm{Cov}\{\cdot\}$	Kovarianz von $\cdot$
$\mathrm{erfc}(x)$	komplementäre Fehlerfunktion von x
$\mathrm{fp}_{n,b}(x)$	Realwertinterpretaion der Festkommadarstellung von $x \in \mathbb{R}$
$\mathrm{fp}^{\mathrm{B}}_{n,b}(x)$	Binärdarstellung der Festkommadarstellung von $x \in \mathbb{R}$
$\mathrm{fp}^{\mathrm{Z}}_{n,b}(x)$	Ganzzahlinterpretation der Festkommadarstellung von $x \in \mathbb{R}$
$\mathrm{Im}(x)$	Imaginärteil von x
$\ln(x)$	Natürlicher Logarithmus von x ($\log_e(x)$)
$\max(\cdot)$	Maximum von $\cdot$
$\max^\star(\cdot)$	siehe (C.20)
$\mathrm{P}(\cdot)$	Wahrscheinlichkeit des Ereignisses $\cdot$
$\mathrm{sgn}(\cdot)$	Signum-Funktion
$\mathrm{Re}(x)$	Realteil von x
$\mathrm{R}\{x, y\}$	Korrelation zwischen x und y
$\Pi(\cdot)$	Permutation von $\cdot$ (Interleaver)
$\rho(x, y)$	Korrelationskoeffizient zwischen x und y
$\mathrm{Var}\{\cdot\}$	Varianz von $\cdot$

Mathematische Akzente

$\hat{\cdot}$	Harter Schätzwert von $\cdot$
$\tilde{\cdot}$	LLR von $\cdot$
$\hat{\cdot}$	Weicher Schätzwert von $\cdot$

Abkürzungsverzeichnis

ADF	adaptive decode and forward
AF	amplify and forward
APP	a-posteriori probability
ARQ	automatic repeat request
AWGN	additive white Gaussian noise
BCJR	Bahl-Cocke-Jelinek-Raviv
BEP	bit error probability
BER	bit error rate
Bit	Binärzeichen
BPSK	Binary Phase Shift Keying
CDM	code-divison multiplexing
CDMA	code-division multiple access
DF	decode and forward
EHC	expected hop count

FDMA	frequency-division multiple access
FEC	forward error correction
HISO	hard-input soft-output
IDM	interleave-division multiplexing
IDMA	interleave-division multiple access
ISI	Intersymbolinterferenz
ISO	International Organization for Standarization
IEC	International Electrotechnical Commission
IR	inkrementelle Redundanz
LDPC	Low-Density-Parity-Check
LDPCC	Low-Density-Parity-Check-Code
LLR	log-likelihood Ratio
Log-APP	logarithmic a-posteriori probability
MA	multiple-access
MAC	Medium-Access-Control
MAP	maximum a-posteriori
Max-Log-APP	maximum logarithmic a-posteriori probability
MCS	modulation and coding scheme
ML	multi-layer
MLC	minimum link count
PCCC	parallel concatenated convolutional code
PCI	Protocol-Control-Information
PDU	Protocol-Data-Unit
ML	maximum-likelihood
QoS	quality of service
ROVA	reliability-output Viterbi-Algorithmus
RTT	Round-Trip-Time
SDU	Service-Data-Unit
SNR	signal-to-noise ratio
SPA	Sum-Product-Algorithmus
SISO	soft-input soft-output
SOVA	soft-output Viterbi-Algorithmus
SROVA	simplified reliability-output Viterbi-Algorithmus
VA	Viterbi-Algorithmus
WEP	word error probability
ZS	Zeitschlitz(e)

Anhang B

Algorithmen zur soft-output Decodierung

Alle hier beschriebenen Algorithmen haben gemeinsam, dass als weiche Eingangswerte die LLRs $\tilde{\boldsymbol{c}}$ dienen. Ausgegeben werden normalerweise die den (teilweise approximierten) a-posteriori Wahrscheinlichkeiten entsprechenden LLRs der Infobits $\tilde{\boldsymbol{u}}$ oder der decodierten Codebits $\tilde{\boldsymbol{c}}_{\mathrm{dec}}$.

Trellisbasierte Decodierung

Für die Beschreibung der auf einem Trellisdiagramm arbeitenden Algorithmen wie dem Viterbi- und dem BCJR-Algorithmus wird ein Trellisdiagramm wie in Abb. B.1 angenommen. Das Trellisdiagramm besteht aus $I = 2^L$ möglichen Zuständen zu jedem Zeitpunkt S_k^i und möglichen Übergängen zwischen den Zuständen aufeinander folgender Zeitpunkte $k-1$ und k. Der Übergang $T_k^{i,j}$ zwischen einem Zustand S_{k-1}^i und dem Folgezustand S_k^j repräsentiert dabei die Infobithypothese $u_k^{i,j}$ die diesen Übergang verursacht. Entsprechend können einem Übergang $T_k^{i,j}$ die Codebithypothesen $c_{k,1}^{i,j}...c_{k,P}^{i,j}$ zugeordnet werden. Damit stellt das Trellisdiagramm alle möglichen Code- bzw. Infowörter dar. Das Trellisdiagramm besteht aus $K+L$ (Länge des terminierten Infoworts $\boldsymbol{u}$) Trellissegmenten, wobei zu jedem Zeitpunkt $I = 2^L$ Zustände möglich sind. Wichtig für die hier verwendeten Algorithmen ist, dass das Trellisdiagramm an Anfang und Ende nur einen möglichen Zustand, üblicherweise den nur aus Nullen bestehenden Zustand, besitzt. Die Wahrscheinlichkeit für den Übergang eines Zustands S_{k-1}^i auf einen Folgezustand S_k^j, die Übergangswahrscheinlichkeit $\mathrm{P}(T_k^{i,j})$, wird durch weiche Eingangsschätzwerte $\tilde{\boldsymbol{c}}$ gegeben. Es wird angenommen, dass einem Infobit u_k für $k = 1...K$ die Codebits $c_{n=(k-1)P+1}$ ... $c_{n=(k-1)P+P}$ mit $n = 1...N$ und $p = 1...P$ entsprechen. Für unabhängige Eingangs-LLRs[1] $\tilde{c}_n$, $n = 1...N$, gilt

$$\mathrm{P}(T_k^{i,j}) = \prod_{p=1}^{P} \frac{e^{\frac{1}{2}\,\mathrm{sgn}(c_{k,p}^{i,j})\tilde{c}_{(k-1)P+p}}}{e^{-\frac{1}{2}\tilde{c}_{(k-1)P+p}} + e^{\frac{1}{2}\tilde{c}_{(k-1)P+p}}}\ , \tag{B.1}$$

[1] Dies ist der Fall für BPSK-Übertragung über einen AWGN-Kanal. Für frequenzselektive Kanäle und/oder höherstufige Modulation kann die Unabhängigkeit durch einen Interleaver nach dem Kanalcodierer erreicht werden.

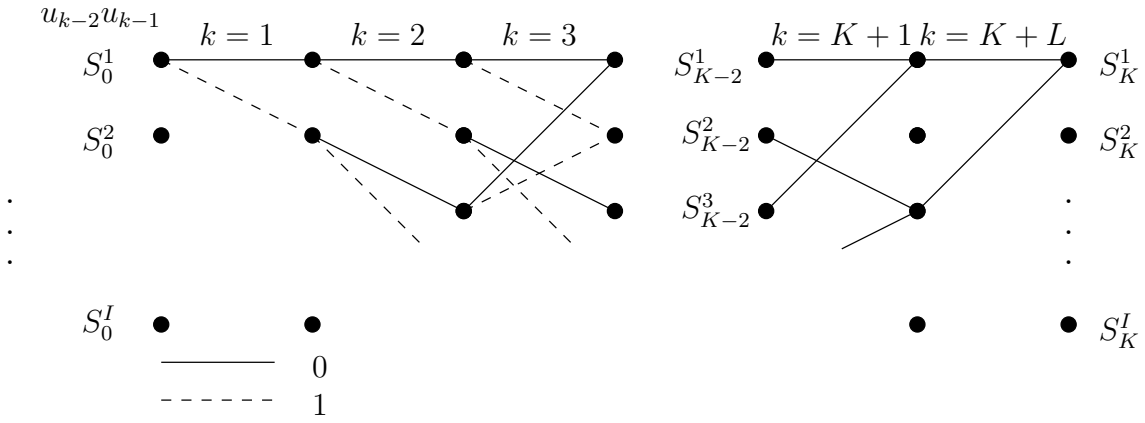

Abbildung B.1: Trellisdiagramm bestehend aus I möglichen Zuständen.

oder – bei Verwendung von logarithmierten Wahrscheinlichkeiten als Metriken –

$$\ln \mathrm{P}(T_k^{i,j}) = \sum_{p=1}^{P} \left(\frac{1}{2} \operatorname{sgn}(c_{k,p}^{i,j}) \tilde{c}_{(k-1)P+p} - \ln(e^{-\frac{1}{2}\tilde{c}_{(k-1)P+p}} + e^{\frac{1}{2}\tilde{c}_{(k-1)P+p}}) \right) . \tag{B.2}$$

Der Term $\ln(e^{-\frac{1}{2}\tilde{c}_{(k-1)P+p}} + e^{\frac{1}{2}\tilde{c}_{(k-1)P+p}})$ ist für alle Übergänge $T_k^{i,j}$ des k-ten Trellissegments identisch und kann daher vernachlässigt werden wenn zwei logarithmierte Wahrscheinlichkeiten eines Zeitpunkts k eine Differenz bilden.

Log-BCJR-Algorithmus

Der Log-BCJR-Algorithmus [RVH95], [RHV97] ist äquivalent zu dem herkömmlichen BCJR-Algorithmus [BCJR74] und stellt damit einen APP-Decodierer für Symbolschätzung dar. Im Gegensatz zum BCJR-Algorithmus arbeitet er nicht mit Wahrscheinlichkeiten sondern mit logarithmierten Wahrscheinlichkeiten. Dies ermöglicht eine effizientere Implementierung und eine geeignetere Darstellung zur Verwendung von Festkommazahlen. Ferner ist die Verwendung logarithmierter Wahrscheinlichkeiten wesentlich für die Anwendung der Max-Log-Approximation [KB90]. Die wichtigsten Größen für die Verwendung des Log-BCJR sind in Abb. B.2 dargestellt. Die LLRs $\tilde{\boldsymbol{c}}$ dienen als Eingangswerte des Algorithmus. Die logarithmierten Wahrscheinlichkeiten entsprechend (B.2) stellen die Zweigmetriken $\gamma(T_k^{i,j})$ für den Übergang von Zustand S_{k-1}^i auf Zustand S_k^j dar. Die Zustandsmetriken $\alpha(S_k^i)$ und $\beta(S_k^i)$ entsprechen der aus der Vorwärts- bzw. Rückwärtsrekursion berechneten Wahrscheinlichkeit des Zustands S_k^i. Die Operation

$$\max^\star(a,b) := \ln(e^a + e^b) = \max(a,b) + \ln(1 + e^{-|a-b|}) \tag{B.3}$$

wird verwendet, um den Logarithmus einer Summe von Exponentialtermen auszudrücken. Für mehr als zwei Argumente kann die max*-Operation rekursiv durchgeführt werden:

$$\max_{i=1\ldots I}^\star(a_i) = \max^\star(a_I, \max^\star(a_{I-1}, \max^\star(\ldots \max^\star(a_2, a_1)\ldots))) . \tag{B.4}$$

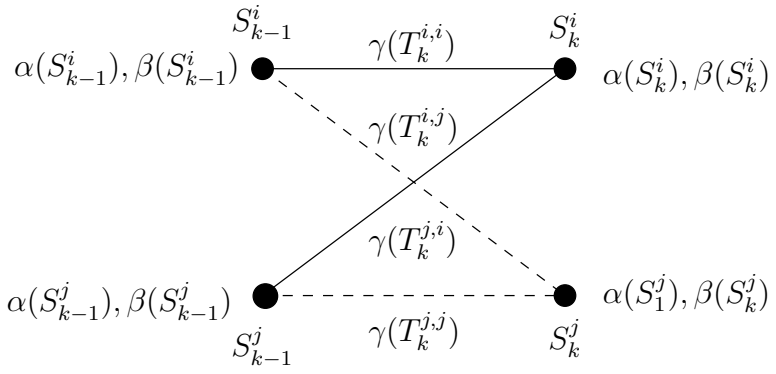

Abbildung B.2: Metriken bei der Verwendung des Log-BCJR-Algorithmus.

Die Berechnung der a-posteriori LLRs der Infobits erfordert die im Folgenden erläuterten vier Schritte.

1. **Berechnung der Zweigmetriken**:
   Die Zweigmetriken
   $$\gamma(T_k^{i,j}) = \sum_{p=1}^{P} \frac{1}{2} \operatorname{sgn}(c_{k,p}^{i,j}) \tilde{c}_{(k-1)P+p} \tag{B.5}$$
   werden für alle k, i und j, also alle Übergänge im Trellis, berechnet.

2. **Vorwärtsrekursion**:
   Die Zustandsmetriken α_k^j werden beginnend mit $k = 0$
   $$\alpha(S_0^j) = \begin{cases} 0 & \text{für } j = 1 \\ -\infty & \text{für } j \neq 1 \end{cases} \tag{B.6}$$
   segmentweise rekursiv
   $$\alpha(S_k^j) = \max_{i=1\ldots I}^{\star} \left(\alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \tag{B.7}$$
   für alles Zustände im Trellisdiagramm berechnet.

3. **Rückwärtsrekursion**:
   Die Zustandsmetriken $\beta(S_{k-1}^i)$ werden beginnend mit $k = K + L$ und
   $$\beta(S_{K+L}^i) = \begin{cases} 0 & \text{für } i = 1 \\ -\infty & \text{für } i \neq 1 \end{cases} \tag{B.8}$$
   segmentweise rekursiv
   $$\beta(S_{k-1}^i) = \max_{j=1\ldots J}^{\star} \left(\beta(S_k^j) + \gamma(T_k^{i,j}) \right) \tag{B.9}$$
   für alle Zustände im Trellis berechnet.

4. **Infobit-LLRs**:
   Die a-posteriori LLRs der Informationsbits, $\tilde{u}_k$, $k = 1...K$, werden als Differenz der Summen der Zweigmetriken der entsprechenden Übergänge berechnet:

$$\tilde{u}_k = \max_{(i,j):u_k^{i,j}=0}{}^{\star} \alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) + \beta(S_k^j) \quad - \max_{(i,j):u_k^{i,j}=1}{}^{\star} \alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) + \beta(S_k^j) \; . \tag{B.10}$$

Eine detaillierte Erläuterung der Äquivalenz zwischen Log-BCJR-Algorithmus und BCJR-Algorithmus findet sich in [RHV97].

Max-Log-BCJR-Algorithmus

Der Max-Log-BCJR-Algorithmus [KB90] geht aus dem Log-BCJR hervor, wenn (B.4) als

$$\max{}^{\star}(a, b) \approx \max(a, b) \tag{B.11}$$

approximiert, also

$$\ln(e^a + e^b) \approx \max(a, b) \tag{B.12}$$

angenommen wird (siehe Anhang C). Der Max-Log-BCJR-Algorithmus verwendet dort wo im Log-BCJR eine Summe über Exponential-Terme auftritt, also in (B.6), (B.8) und (B.10), die Approximation (B.11). Damit ergeben sich die Schritte 2-4 analog zum Log-BCJR-Algorithmus.

2. **Vorwärtsrekursion**:

$$\alpha(S_0^j) = \begin{cases} 0 & \text{für } j = 1 \\ -\infty & \text{für } j \neq 1 \end{cases} \tag{B.13}$$

$$\alpha(S_k^j) = \max_{i=1...I} \left(\alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \tag{B.14}$$

3. **Rückwärtsrekursion**:

$$\beta(S_{K+L}^i) = \begin{cases} 0 & \text{für } i = 1 \\ -\infty & \text{für } i \neq 1 \end{cases} \tag{B.15}$$

$$\beta(S_{k-1}^i) = \max_{j=1...J} \left(\beta(S_k^j) + \gamma(T_k^{i,j}) \right) \tag{B.16}$$

4. **Infobit-LLRs**:

$$\tilde{u}_k = \max_{(i,j):u_k^{i,j}=0} \alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) + \beta(S_k^j) \quad - \max_{(i,j):u_k^{i,j}=1} \alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) + \beta(S_k^j) \tag{B.17}$$

Optimaler Subblock-by-Subblock Decodierer

Der Subblock-by-Subblock APP, auch Optimal Subblock-by-Subblock-Decodierer (OBBD) genannt, ist eine Verallgemeinerung des BCJR-Algorithmus die statt den a-posteriori Wahrscheinlichkeiten der Symbole u_k die Wahrscheinlichkeit für Blöcke aus B Symbolen $u_{qB+1}^{(q+1)B} = u_{qB+1}, \ldots, u_{(q+1)B}$ mit $q = 0, 1, \ldots (K+L)/B - 1$ berechnet [Hoe95]. Damit wird die Infobitsequenz in $Q = (K+L)/B$ Blöcke aufgeteilt.

Bei der Durchführung des OBBD werden zwei Fälle unterschieden: Blocklängen kleiner oder gleich der Gedächtnislänge des Codes ($B \leq L$) und Blocklängen größer der Gedächtnislänge ($B > L$). Beide Varianten sind hier für logarithmierte Wahrscheinlichkeiten dargestellt.

Blocklänge $B \leq L$

Vorwärts- und Rückwärtsrekursion entsprechen dem BCJR-Algorithmus. Die Wahrscheinlichkeiten der Blockhypothesen werden (für nicht-rekursive Codes) aus den Eingangswerten berechnet.

1. **Berechnung der Zweigmetriken**:
   Die Zweigmetriken
   $$\gamma(T_k^{i,j}) = \sum_{p=1}^{P} \frac{1}{2} \operatorname{sgn}(c_{k,p}^{i,j}) \tilde{c}_{(k-1)P+p} \tag{B.18}$$
   werden für alle k, i und j, also alle Übergänge im Trellis berechnet.

2. **Vorwärtsrekursion**:
   Die Zustandsmetriken α_k^j werden beginnend mit $k = 0$
   $$\alpha(S_0^j) = \begin{cases} 0 & \text{für } j = 1 \\ -\infty & \text{für } j \neq 1 \end{cases} \tag{B.19}$$
   segmentweise rekursiv
   $$\alpha(S_k^j) = \max_{i=1\ldots I}^{\star} \left(\alpha(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \tag{B.20}$$
   für alles Zustände im Trellisdiagramm berechnet.

3. **Rückwärtsrekursion**:
   Die Zustandsmetriken $\beta(S_{k-1}^i)$ werden beginnend mit $k = K + L$ und
   $$\beta(S_{K+L}^i) = \begin{cases} 0 & \text{für } i = 1 \\ -\infty & \text{für } i \neq 1 \end{cases} \tag{B.21}$$
   segmentweise rekursiv
   $$\beta(S_{k-1}^i) = \max_{j=1\ldots J}^{\star} \left(\beta(S_k^j) + \gamma(T_k^{i,j}) \right) \tag{B.22}$$

für alle Zustände im Trellis berechnet. In dem Zustand S_k^i sind jeweils die letzten L Hypothesen $u_k^{i,j}$ gespeichert. Die Wahrscheinlichkeit für die Blockhypothese u_{k-B+1}^k ist daher die Wahrscheinlichkeit über die Zustände deren B letzten Bits $u_{k-B+1}^{i,j} \dots u_k^{i,j}$ gleich der Blockhypothese sind. Damit ist

$$\ln \mathrm{P}(u_{k-B+1}^k|\tilde{\boldsymbol{c}}) = \max_{\substack{i=1\dots I \\ S_k^i : u_{k-B+1}^k}}^{\star} (\beta(S_k^i) + \alpha(S_k^i)) \tag{B.23}$$

die logarithmierte Wahrscheinlichkeit für die Infobitblockhypothese $u_{k-B+1}^{i,j} \dots u_k^{i,j}$.

Der Aufwand für den OBBD ist für $1 < B \leq L$ sogar geringer als die des BCJR. Für $B = 1$ entspricht der OBBD dem BCJR [Hoe95].

Blocklänge $B > L$

In diesem Fall arbeitet der Algorithmus nicht mehr auf dem ursprünglichen Trellisdiagramm, sondern auf einer Art *Hypertrellisdiagramm* in dem Übergänge nur noch zwischen Zuständen im Abstand von B Infobits möglich sind. Das Hypertrellisdiagramm besteht also aus Segmenten die jeweils aus B herkömmlichen Segmenten bestehen – siehe Abb. B.3.

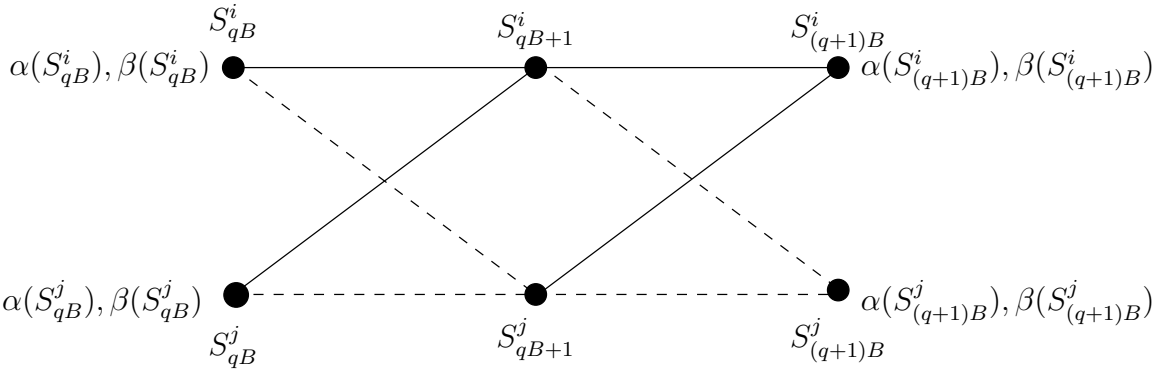

Abbildung B.3: Metriken bei der Verwendung des Log-OBBD-Algorithmus bei einer Blocklänge von $B = 2 > L = 1$.

Die Berechnung der a-posteriori Wahrscheinlichkeiten der Infobitblöcke $u_{qB+1}^{(q+1)B}$ erfordert im Wesentlichen dieselben Schritte wie auch der BCJR.

1. **Berechnung der Zweigmetriken**:
   Die Zweigmetriken für die 2^B möglichen Übergänge $T_q^{i,j}$ vom Zustand S_{qB}^i auf den Zustand $S_{qB+B}^j = S_{(q+1)B}^j$

$$\gamma(T_q^{i,j}) = \sum_{k=lB}^{(l+1)B-1} \sum_{p=1}^{P} \frac{1}{2} \operatorname{sgn}(c_{k,p}^{i,j}) \tilde{c}_{(k-1)P+p} \tag{B.24}$$

   werden für alle Blöcke $q = 0, 1, ...(K+1)/B - 1$, i und j, also alle Übergänge des Hypertrellis berechnet.

2. **Vorwärtsrekursion**:
   Die Zustandsmetriken $\alpha(S_{qB}^{j})$ werden beginnend mit $q = 0$ und

$$\alpha(S_0^j) = \begin{cases} 0 & \text{für } j = 1 \\ -\infty & \text{für } j \neq 1 \end{cases} \tag{B.25}$$

   segmentweise rekursiv

$$\alpha(S_{(q+1)B}^{j}) = \max_{i=1\ldots I}^{\star} \left(\alpha(S_{qB}^{i}) + \gamma(T_l^{i,j})\right) \tag{B.26}$$

   für alles Zustände im Hypertrellisdiagramm berechnet.

3. **Rückwärtsrekursion**:
   Die Zustandsmetriken $\beta(S_{qB}^{i})$ werden beginnend mit $q = (K + L)/B - 1 = Q$ und

$$\beta(S_Q^i) = \begin{cases} 0 & \text{für } i = 1 \\ -\infty & \text{für } i \neq 1 \end{cases} \tag{B.27}$$

   segmentweise rekursiv

$$\beta(S_{qB}^{i}) = \max_{j=1\ldots J}^{\star} \left(\beta(S_{(q+1)B}^{j}) + \gamma(T_l^{i,j})\right) \tag{B.28}$$

   für alle Zustände im Trellis berechnet. Die Wahrscheinlichkeiten der Blockhypothesen sind

$$\ln \mathrm{P}(u_{qB+1}^{(q+1)B} = \hat{u}_{qB+1}^{(q+1)B} | \tilde{\boldsymbol{c}}) = \max_{(i,j):u_{qB}^{(q+1)B}}^{\star} \alpha(S_{qB}^{i}) + \gamma(T_q^{i,j}) + \beta(S_{(q+1)B}^{j}) \,, \tag{B.29}$$

   also die Summe der Wahrscheinlichkeiten der Übergänge die der Hypothese entsprechen.

LLR-Berechnung

Die Berechnung der Infobit-LLRs geschieht für beide Fälle auf die gleiche Art und Weise. Pro Block werden 2^{B-1} LLRs aus den Wahrscheinlichkeiten der Blockhypothesen berechnet:

$$\tilde{u}_k = \max_{u_{qB+1}^{(q+1)B}:u_k=0}^{\star} \ln \mathrm{P}(u_{qB+1}^{(q+1)B} | \tilde{\boldsymbol{c}}) \quad - \max_{u_{qB+1}^{(q+1)B}:u_k=1}^{\star} \ln \mathrm{P}(u_{qB+1}^{(q+1)B} | \tilde{\boldsymbol{c}}) \,. \tag{B.30}$$

Viterbi-Algorithmus

Der Viterbi-Algorithmus (VA) [Vit67] ist ein maximum-likelihood Sequenzschätzer, liefert also als Schätzwert das wahrscheinlichste Codewort. Wie für den BCJR wird für die Beschreibung des VA ebenfalls die auf logarithmierten Wahrscheinlichkeiten basierende Variante gewählt. Ausgehend von einem Trellisdiagramm wie in Abb. B.1 sind die wesentlichen Größen des VA die *Zweigmetriken* $\gamma(T_k^{i,j})$ und die *Pfadmetriken* $\Gamma(S_k^i)$ (dargestellt in Abb. B.4). Die Zweigmetriken $\gamma(T_k^{i,j})$ entsprechen denen des Log-BCJR-Algorithmus.

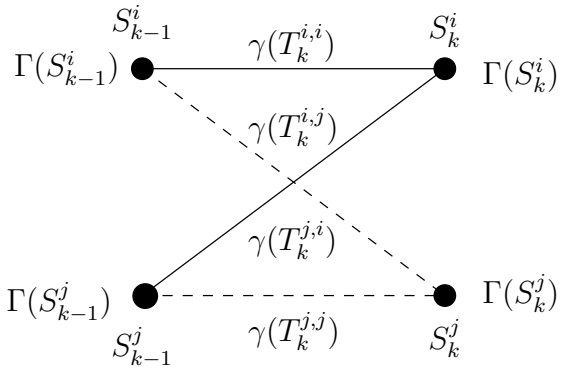

Abbildung B.4: Metriken bei der Verwendung des Log-Viterbi-Algorithmus.

1. **Berechnung der Zweigmetriken**:
   Die Zweigmetriken
   $$\gamma(T_k^{i,j}) = \sum_{p=1}^{P} \frac{1}{2} \operatorname{sgn}(c_{k,p}^{i,j}) \tilde{c}_{(k-1)P+p} \tag{B.31}$$
   für alle Übergänge $T_k^{i,j}$ des Trellisdiagramm werden berechnet.

2. **Vorwärtsrekursion**:

   (a) Die Pfadmetriken für $k = 0$
   $$\Gamma(S_0^j) = \begin{cases} 0 & \text{für } j = 1 \\ -\infty & \text{für } j \neq 1 \end{cases} \tag{B.32}$$
   werden initialisiert.

   (b) Die Pfadmetriken
   $$\Gamma(S_k^j) = \max_{i=1...I} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \tag{B.33}$$
   werden rekursiv trellissegmentweise für $k = 1...K+L$ berechnet. Der Übergang $T_k^{i_{\max},j}$, der (B.33) maximiert, wird gespeichert.

3. **Rückverfolgung**:
   In dem terminierten Trellisdiagramm endet der Übergang $T_{K+L}^{i_{\max},j}$ im Zustand S_{K+L}^1. Folgt man von S_K^1 jeweils den Übergängen $T_k^{i_{\max},j}$ auf die vorangegangenen Zustände S_{k-1}^i bis S_0^1, so ergibt sich der *Maximum-Likelihood-Pfad*, der wahrscheinlichste Pfad durch das Trellisdiagramm. Die den Übergängen $T_k^{i_{\max},j}$ entsprechenden Infobithypothesen $u_k^{i,j}$ ergeben die ML-Schätzung für $\boldsymbol{u}$.

Soft-output Viterbi-Algorithmus

Der soft-output Viterbi-Algorithmus (SOVA) [HH89] ist ein Viterbi-Algorithmus, der erweitert wurde, um weiche Ausgangsschätzwerte zu erzeugen. Dafür wird (hier in der Version aus [FBSH98]) während der Vorwärtsrekursion ein weiterer Schritt zur Erzeugung und zum Aktualisieren von Zuverlässigkeitsinformation eingeführt. Für jeden überlebenden Pfad werden Zuverlässigkeitsinformationen $\check{u}_k$ für die Schätzwerte $\hat{u}_k$ erzeugt, gespeichert und aktualisiert. Die Initialisierung (Schritte 1 und 2(a)) sowie die Rückverfolgung (3) sind identisch mit dem Viterbi-Algorithmus.

2. **Vorwärtsrekursion**:
   Segmentweise werden für $k = 1...K$ die Schritte (b) und (c) durchgeführt.

   (b) Die Pfadmetriken werden wie im VA berechnet:

$$\Gamma(S_k^j) = \max_{i=1...I} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \ . \tag{B.34}$$

   Der Übergang $T_k^{i_{\max},j}$, der laut (B.34) maximiert, wird gespeichert.

   (c) **Berechnung der Zuverlässigkeitsinformation**:

   i. Zusätzlich wird die Differenz Δ zwischen der Pfadmetrik des besten (Pfad 1) und des zweitbesten im Zustand S_k^j endenden Pfades (Pfad 2) gespeichert. Für binäre Symbole sind nur zwei Zustandsübergänge möglich, so dass Pfad 2 der Pfad mit der minimalen Pfadmetrik ist

$$\Delta = \max_{i=1...I} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) - \min_{i=1...I} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \ . \tag{B.35}$$

   Die Zuverlässigkeitsinformation Δ ist immer positiv und entspricht der Amplitude eines LLRs.

   ii. Die Zuverlässigkeitsinformation für das k-te Infobit wird zu $\check{u}_k = \infty$ initialisiert

   iii. Für den überlebenden Pfad (Pfad 1) wird die Zuverlässigkeitswerte der vorangegangenen Entscheidungen mit Hilfe des nicht ausgewählten Pfades (Pfad 2) aktualisiert. Hierfür wird die Zuverlässigkeitsinformation für $b = k...1$ entsprechend

$$\hat{u}_b^{(1)} \neq \hat{u}_b^{(2)} : \qquad \check{u}_b^{(1)} = \min(\Delta, \check{u}_b^{(2)}) \tag{B.36}$$

$$\hat{u}_b^{(1)} = \hat{u}_b^{(2)} : \qquad \check{u}_b^{(1)} = \min(\Delta + \check{u}_b^{(2)}, \check{u}_b^{(1)}) \tag{B.37}$$

   geändert.[2] Die Zuverlässigkeitsinformation $\check{u}_k$ ist ein Schätzwert für den Betrag von $\tilde{u}_k$.

[2] Für praktische Anwendungen sollten die Zuverlässigkeitswerte nur in einem Fenster $b = k, \ldots, k - \delta$ aktualisiert werden [HH89].

Reliability-output Viterbi-Algorithmus

Der reliability-output Viterbi-Algorithmus (ROVA) [RB98] berechnet im Gegensatz zum SOVA *Zuverlässigkeitsinformation für die geschätzten Sequenzen*. Der ROVA liefert die exakte Wahrscheinlichkeit dafür, dass der gefundene Pfad (also Code- bzw Infowortschätzwert) richtig ist. Die komplementäre Wahrscheinlichkeit ist die Fehlerwahrscheinlichkeit.

Die Wahrscheinlichkeit, dass der in Zustand S_k^i endende überlebende Pfad richtig bzw. nicht richtig ist, kann zusammen mit der Vorwärtsrekursion berechnet werden. Dafür wird in der Vorwärtsrekursion für jeden Zustand die Wahrscheinlichkeit $\mathrm{P}(S_k^j)$ berechnet, dass der in ihm überlebende Pfad richtig ist sowie die Wahrscheinlichkeit $\mathrm{P}(\overline{S}_k^j)$, dass der in S_k^j überlebende Pfad nicht richtig ist. Hier wird der ROVA im Gegensatz zu [RB98] in der logarithmischen Form angegeben.

2. **Vorwärtsrekursion**:
   Segmentweise werden für $k = 1...K+L$ die Schritte (b) und (c) für alle Zustände $j = 1 \ldots J$ durchgeführt.

   (b) Die Pfadmetriken werden wie im VA berechnet:

   $$\Gamma(S_k^j) = \max_{i=1...I} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \ . \tag{B.38}$$

   (c) **Berechnung der Wahrscheinlichkeiten** $\mathrm{P}(S_k^j)$ **und** $\mathrm{P}(\overline{S}_k^j)$:
   Sei o. B. d. A. der Übergang von S_{k-1}^j nach S_k^j der Übergang des überlebenden Pfades. Dann werden die Wahrscheinlichkeiten

   $$\ln \mathrm{P}(S_k^j) = \ln \mathrm{P}(S_{k-1}^i) + \gamma(T_k^{i,j}) - \Lambda \tag{B.39}$$

   und

   $$\begin{aligned} \ln \mathrm{P}(\overline{S}_k^j) = \max{}^\star \Bigg[& \max_{\substack{i=1...I \\ i \neq j}}{}^\star \left(\ln \mathrm{P}(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) , \\ & \max_{i=1...I}{}^\star \left(\ln \mathrm{P}(\overline{S}_{k-1}^i) + \gamma(T_k^{i,j}) \right) \Bigg] - \Lambda \end{aligned} \tag{B.40}$$

   berechnet, wobei

   $$\begin{aligned} \Lambda = \max{}^\star \Bigg[& \max_{\substack{i=1...I \\ j=1...J}}{}^\star \left(\ln \mathrm{P}(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) , \\ & \max_{\substack{i=1...I \\ j=1...J}}{}^\star \left(\ln \mathrm{P}(\overline{S}_{k-1}^i) + \gamma(T_k^{i,j}) \right) \Bigg] \end{aligned} \tag{B.41}$$

   einen Normalisierungsterm darstellt.

In dem letzten Zustand (des terminierten) Trellisdiagramms S_{K+L}^1 entspricht die Wahrscheinlichkeit $\mathrm{P}(S_{K+L}^1)$ der Wahrscheinlichkeit dafür, dass die geschätzte Sequenz richtig ist. Dagegen ist $\mathrm{P}(\overline{S}_{K+L}^1)$ die Wahrscheinlichkeit, dass ein nicht überlebender Pfad richtig gewesen wäre, also die geschätzte Sequenz falsch ist.

Vereinfachter reliability-output Viterbi-Algorithmus

Der *vereinfachte* (engl. simplified) ROVA [FH07] hat ebenfalls die Berechnung der Fehlerwahrscheinlichkeit der geschätzten Sequenz zum Ziel. Im Gegensatz zum ROVA werden die Wahrscheinlichkeiten $\mathrm{P}(S_k^j)$ und $\mathrm{P}(\overline{S}_k^j)$ jedoch approximativ berechnet was sich in geringerem Aufwand an Rechenoperationen und Speicherbedarf niederschlägt.

Wie der ROVA beinhaltet auch der vereinfachte ROVA die wesentlichen Schritte des Viterbi-Algorithmus. In der Vorwärtsrekursion werden ähnliche zusätzliche Berechnungen durchgeführt die sich im wesentlichen dadurch unterscheiden, dass für den vereinfachten ROVA nur die (approximierte) Wahrscheinlichkeit $\mathrm{P}(S_k^j)$ dafür, dass der überlebende Pfad in Zustand S_k^j richtig ist.

2. **Vorwärtsrekursion**:
   Segmentweise werden für $k = 1...K$ die Schritte (b) und (c) für alle Zustände $j = 1 \dots J$ durchgeführt.

   (b) Die Pfadmetriken werden wie im VA berechnet:

   $$\Gamma(S_k^j) = \max_{i=1...I} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \ . \tag{B.42}$$

   (c) **Berechnung der Wahrscheinlichkeiten** $\mathrm{P}(S_k^j)$ **und** $\mathrm{P}(\overline{S}_k^j)$:
   Sei o. B. d. A. der Übergang von S_{k-1}^j nach S_k^j der Übergang des überlebenden Pfades. Damit werden die Wahrscheinlichkeiten

   $$\ln \mathrm{P}(S_k^j) = \Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) - \Lambda \tag{B.43}$$

   berechnet, wobei

   $$\Lambda = \max_{i=1...I}^{\star} \left(\Gamma(S_{k-1}^i) + \gamma(T_k^{i,j}) \right) \tag{B.44}$$

   einen Normalisierungsterm darstellt.

Aus (B.42)–(B.44) ist ersichtlich, dass die Berechnungen von $\mathrm{P}(S_k^j)$ des vereinfachten ROVAs direkt auf den Pfadmetriken beruhen und nicht wie die des ROVA auf den Wahrscheinlichkeiten der vorangehenden Zustände. Dies erspart die Verwaltung weiterer Variablen wie im ROVA, der einerseits die Pfadmetriken für jeden Zustand und andererseits die Wahrscheinlichkeiten des Zustands speichern muss. Auch die Berechnung des Normalisierungsterms (B.44) ist weit weniger aufwändig als im ROVA (B.41), da nicht berücksichtigt wird, dass der vorangehende Zustand des ausgewählten bzw. nicht ausgewählten Pfades falsch sein könnte. Für binäre Infosymbole ($I = 2$) können (B.44) und (B.43) umgeschrieben werden:

$$\begin{aligned} \ln \mathrm{P}(S_k^j) &= \ln \frac{e^{\Gamma(S_{k-1}^{i\max}) + \gamma(T_k^{i\max,j})}}{e^{\Gamma(S_{k-1}^{i\max}) + \gamma(T_k^{i\max,j})} + e^{\Gamma(S_{k-1}^{i\neq i\max}) + \gamma(T_k^{i\neq i\max,j})}} \\ &= \ln \frac{1}{1 + e^{\Gamma(S_{k-1}^{i\neq i\max}) + \gamma(T_k^{i\neq i\max,j}) - \Gamma(S_{k-1}^{i\max}) - \gamma(T_k^{i\max,j})}} \ . \end{aligned} \tag{B.45}$$

Der Exponent in (B.45) entspricht der vom SOVA berechneten Zuverlässigkeitsinformation (ohne Update) Δ (B.35). Das bedeutet, dass der Mehraufwand für einen vereinfachten ROVA bei Verwendung des SOVA minimal ist [FH07].

Graphenbasierte Decodierung von LDPC-Codes mit dem Sum-Product-Algorithmus

Die Algorithmen für die Decodierung von LDPC-Codes basieren häufig auf der Darstellung des Codes als Faktorgraph [KFL01]. Für LDPC-Codes besteht dieser Graph aus zwei Arten von Knoten, den N Variable-Nodes $\boldsymbol{v}$ genannten Knoten, die die Codebits repräsentieren, und den P (Parity-)Check-Nodes $\boldsymbol{f}$ genannten Knoten, die die Prüfgleichungen darstellen. Jeder Knoten ist immer nur mit Knoten der jeweils anderen Art verbunden, der Graph kann daher wie in Abb. B.5 angegeben werden.

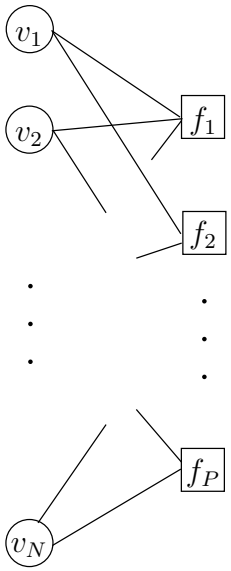

Abbildung B.5: Faktorgraph zur Decodierung von LDPCC-Codes mit N Variable-Nodes und P (Parity-)Check Nodes.

Die Menge der mit dem p-ten Check-Node verbundenen Variable-Nodes sei im Folgenden mit $\mathbb{V}_p = \{v_1^{(p)}, \ldots, v_{|\mathbb{V}_p|}^{(p)}\}$ und die Menge der mit dem n-ten Variable-Node verbundenen Check-Nodes mit $\mathbb{F}_n = \{f_1^{(n)}, \ldots, f_{|\mathbb{F}_n|}^{(n)}\}$ bezeichnet.

Der Sum-Product-Algorithm (oder auch Belief-Propagation Algorithm (BPA)) [Gal62] tauscht iterativ Informationen zwischen den Variable- und den Check-Nodes aus. Da die Variable-Nodes den Codebits entsprechen, verändert sich die über die Codebits vorhandene Information von Iteration zu Iteration. Eine Iteration i umfasst dabei die beiden in Abb. B.6 und B.7 Abläufe. Zuerst (Abb. B.6) wird die Information der Variable-Nodes (also der Codebits) an die Check-Nodes gesendet wo eine weiche Auswertung der Prüfgleichungen stattfindet.
Im zweiten Schritt (B.7) wird die Information über die Auswertung der Prüfgleichungen an die Variable-Nodes geschickt, um die weichen Schätzwerte der Codebits zu verbessern. Die von dem n-ten Variable-Node an die p-ten Check-Node gesendete Nachricht wird mit $\mu^i_{v_n \to f_p}$ bezeichnet, umgekehrt die Nachricht vom p-ten Check-Node an den n-ten Variable-Node mit $\mu^i_{f_p \to v_n}$.

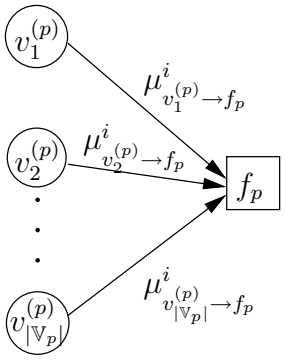

Abbildung B.6: Senden von Nachrichten an den Check-Node p.

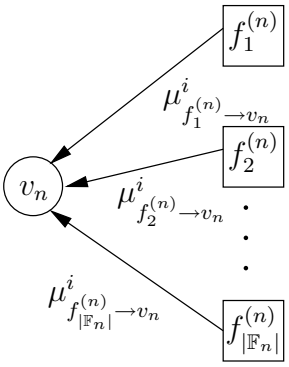

Abbildung B.7: Senden von Nachrichten an den Variable-Node n.

In der hier angegebenen Version wird Information zwischen den Knoten in Form von LLRs ausgetauscht (in Anlehnung an [Sho03], siehe darin für die Herleitung). Eine Version des SPA basierend auf Wahrscheinlichkeiten findet sich in [KFL01]. Die Nachrichten der Variable-Nodes an die Check-Nodes werden zu

$$\mu^i_{v_n\to f_p} = \begin{cases} \tilde{c}_n & i = 1 \\ \tilde{c}_n + \sum\limits_{f'\in\mathbb{F}_n\setminus f_p} \mu^{i-1}_{f'\to v_n} & i > 1 \end{cases} \tag{B.46}$$

berechnet. Dies entspricht dem Eingangs-LLR für das n-te Codebit plus die Information der anderen über die Prüfgleichung mit diesem Codebits verknüpften Bits. Entsprechend der Prüfgleichungen senden alle mit dem Knoten p verbundenen Variable-Nodes $\mathbb{V}_p$ eine Nachricht an den Knoten f_p. Zu beachten ist, dass extrinsische Information ausgetauscht wird, in der Summe in (B.46) wird die vorherige Nachricht von Knoten f_p an den Variable-Node n nicht mit aufsummiert.

Die (wiederum extrinsische Information enthaltenen) Nachrichten der Check-Nodes an die Variable-Nodes entsprechen den Wahrscheinlichkeiten einer erfolgreichen Auswertung

der Prüfgleichung gegeben die Information aus den Variable-Nodes

$$\mu^i_{f_p \to v_n} = \ln \frac{1 + \prod\limits_{v' \in \mathbb{V}_p \setminus v_n} \tanh\left(\frac{\mu^i_{v' \to f_p}}{2}\right)}{1 - \prod\limits_{v' \in \mathbb{V}_p \setminus v_n} \tanh\left(\frac{\mu^i_{v' \to f_p}}{2}\right)} , \tag{B.47}$$

wobei die Information wieder in Form von LLRs übermittelt wird. Nach I Iterationen werden die Nachrichten der Variable-Nodes als LLRs der Codebits angenommen:

$$\tilde{c}_n := \mu^I_{v_n \to f_p} . \tag{B.48}$$

Sowohl (B.46) als auch (B.47) nehmen statistische Unabhängigkeit zwischen den eingehenden Nachrichten an. Die statistische Unabhängigkeit ist aber selbst bei statistisch unabhängigen Eingangswerten $\tilde{\boldsymbol{c}}$ spätestens nach einigen Iterationen nicht mehr gegeben. Daher ist dieser Algorithmus nicht optimal.

Iterativer multi-user Detektor für IDM

Hier soll der in Kapitel 9 verwendete multi-user Detektor für IDM-Systeme angegeben werden. Die hier angegebene Version ist ähnlich der in [LLL03], jedoch für komplexe Kanalkoeffizienten h_m modifiziert. Der Erwartungswert und die Varianz des Empfangssignals y_n zum Symboltakt n können als

$$\mathrm{E}\{y_n\} = \sum_{d=1}^{D} h_{d,n} \, \mathrm{E}\{z_{d,n}\} \tag{B.49}$$

bzw.

$$\mathrm{Var}\{y_n\} = \sum_{d=1}^{D} |h_{d,n}|^2 \, \mathrm{Var}\{z_{d,n}\} + \sigma_n^2 \tag{B.50}$$

berechnet werden, wobei $z_{d,n}$ das BPSK-Mapping der interleavten Codebits darstellt. Layer-spezifische Phasendrehung φ_d sowie Amplitude a_d sind in den Kanalkoeffizienten $\boldsymbol{h}_d$ enthalten. Eine wesentliche in diesem Detektor zur Anwendung kommende Idee ist die, Interferenz und Rauschen für den d-ten Layer

$$\xi_{d,n} = \sum_{\substack{d'=1 \\ d' \neq d}}^{D} h_{d,n} z_{d,n} + n_n \tag{B.51}$$

als gaußverteilt anzunehmen. Der Erwartungswert von (B.51) ist

$$\mathrm{E}\{\xi_{d,n}\} = \sum_{\substack{d'=1 \\ d' \neq d}}^{D} h_{d',n} \, \mathrm{E}\{x_{d',n}\} = \mathrm{E}\{y_n\} - h_{d,n} \, \mathrm{E}\{z_{d,n}\} , \tag{B.52}$$

die Varianz

$$\mathrm{Var}\left\{\xi_{d,n}\right\} = \sum_{\substack{d'=1 \\ d'\neq d}}^{D} h_{d',n}\,\mathrm{Var}\left\{z_{d',n}\right\} = \mathrm{Var}\left\{y_n\right\} - |h_{d,n}|^2\,\mathrm{Var}\left\{z_{d,n}\right\}\ . \tag{B.53}$$

Unter der Annahme der Gaußverteilung kann mit (B.52) und (B.53) die Wahrscheinlichkeitsverteilung für $z_{d,n}$ berechnet werden:

$$\mathrm{P}(z_{d,n} = z'_{d,n}|y_n) = \frac{1}{\mathrm{Var}\left\{\xi_d\right\}\pi}\exp\left(-\frac{|y_n - z'_{d,n} - \mathrm{E}\left\{\xi_{d,n}\right\}|^2}{\mathrm{Var}\left\{\xi_{d,n}\right\}}\right)\ . \tag{B.54}$$

Der LLR

$$\begin{aligned}
\tilde{z}_{d,n} &= \ln\frac{\mathrm{P}(z_{d,n}=+1|y_n)}{\mathrm{P}(z_{d,n}=-1|y_n)} \\
&= \ln\frac{1}{\mathrm{Var}\left\{\xi_d\right\}\pi}\exp\left(-\frac{|y_n - h_{d,n} - \mathrm{E}\left\{\xi_{d,n}\right\}|^2}{\mathrm{Var}\left\{\xi_{d,n}\right\}}\right) \\
&\quad - \ln\frac{1}{\mathrm{Var}\left\{\xi_d\right\}\pi}\exp\left(-\frac{|y_n + h_{d,n} - \mathrm{E}\left\{\xi_{d,n}\right\}|^2}{\mathrm{Var}\left\{\xi_{d,n}\right\}}\right) \\
&= \frac{2\left(y_n h^*_{d,n} - \mathrm{E}\left\{\xi_{d,n}\right\}h^*_{d,n} - h_{d,n}\,\mathrm{E}\left\{\xi_{d,n}\right\}^* + \mathrm{E}\left\{\xi_{d,n}\right\}y^*_n\right)}{\mathrm{Var}\left\{\xi_{d,n}\right\}} \\
&= 4\,\mathrm{Re}\left(\frac{(y_n - \mathrm{E}\left\{\xi_{d,n}\right\})h^*_{d,n}}{\mathrm{Var}\left\{\xi_{d,n}\right\}}\right)
\end{aligned} \tag{B.55}$$

kann mit (B.49) und (B.50) als

$$\tilde{z}_{d,n} = 4\,\mathrm{Re}\left(\frac{h^*_{d,n}\left(y_n - \mathrm{E}\left\{y_n\right\} + h_{d,n}\,\mathrm{E}\left\{z_{d,n}\right\}\right)}{\mathrm{Var}\left\{y_n\right\} - |h_{d,n}|^2\,\mathrm{Var}\left\{z_{d,n}\right\}}\right) \tag{B.56}$$

dargestellt werden. In (B.56) muss die Summe in (B.52) und (B.53) nicht für jedes d neu berechnet werden.

Der Detektor arbeitet iterativ, in jeder Iteration $i > 1$ erhält er LLRs $\tilde{\boldsymbol{z}}_d$ jedes Layers $d = 1 \ldots D$. Für die erste Iteration gilt

$$\tilde{z}_{d,n} = 0\ , \tag{B.57}$$

sofern keine a-priori Information für die Codebits zur Verfügung steht. In jede Iteration werden die folgenden Operationen durchgeführt:

1.
$$\mathrm{E}\left\{z_{d,n}\right\} = \tanh\left(\frac{\tilde{z}_{d,n}}{2}\right) \quad \forall\, d,n$$

2.
$$\mathrm{Var}\left\{z_{d,n}\right\} = 1 - \mathrm{E}\left\{z_{d,n}\right\} \quad \forall\, d,n$$

3.
$$\mathrm{E}\{y_m\} = \sum_{d=1}^{D} h_{d,m}\,\mathrm{E}\{z_{d,n}\} \quad \forall n$$

4.
$$\mathrm{Var}\{y_n\} = \sum_{d=1}^{D} |h_{d,n}|^2\,\mathrm{Var}\{z_{d,n}\} + \sigma_n^2 \quad \forall n$$

5.
$$\tilde{z}_{d,n} = 4\,\mathrm{Re}\left(\frac{h_{d,n}^*\,(y_n - \mathrm{E}\{y_n\} + h_{d,n}\,\mathrm{E}\{z_{d,n}\})}{\mathrm{Var}\{y_n\} - |h_{d,n}|^2\,\mathrm{Var}\{z_{d,n}\}}\right) \quad \forall\, d, n$$

Anschließend werden die LLRs der einzelnen Layer $\tilde{z}_{d,n}$ deinterleavt und an den Decodierer weitergegeben (siehe Abb. 9.1). Dessen weichen Ausgangswerten dienen in der folgenden Iteration zur Berechnung von 1–5.

Anhang C

Ergänzende Erläuterungen

Zusammenhang zwischen LLRs und Bitfehlerwahrscheinlichkeit

Gegeben den LLR für ein Infobit $\tilde{u}_k$ und der Beobachtung $\tilde{\boldsymbol{c}}$ gilt

$$\tilde{u}_k = \log \frac{P(u_k = +1|\tilde{\boldsymbol{c}})}{P(u_k = -1|\tilde{\boldsymbol{c}})} = \log \frac{P(u_k = +1|\tilde{\boldsymbol{c}})}{1 - P(u_k = +1|\tilde{\boldsymbol{c}})} \; . \tag{C.1}$$

Durch Umformung von (C.1)

$$e^{\tilde{u}_k}(1 - P(u_k = +1|\tilde{\boldsymbol{c}})) = P(u_k = +1|\tilde{\boldsymbol{c}}) \tag{C.2}$$

$$e^{\tilde{u}_k} - e^{\tilde{u}_k} P(u_k = +1|\tilde{\boldsymbol{c}}) = P(u_k = +1|\tilde{\boldsymbol{c}}) \tag{C.3}$$

$$e^{\tilde{u}_k} = P(u_k = +1|\boldsymbol{c}) + e^{\tilde{u}_k} P(u_k = +1|\tilde{\boldsymbol{c}}) \tag{C.4}$$

erhält man schließlich

$$P(u_k = +1|\tilde{\boldsymbol{c}}) = \frac{e^{\tilde{u}_k}}{1 + e^{\tilde{u}_k}} = \frac{1}{1 + e^{-\tilde{u}_k}} \; . \tag{C.5}$$

Da $P(u_k = +1|\tilde{\boldsymbol{c}})$ und $P(u_k = -1|\tilde{\boldsymbol{c}})$ komplementär sind, kann aus (C.5)

$$P(u_k = -1|\tilde{\boldsymbol{c}}) = 1 - \frac{e^{\tilde{u}_k}}{1 + e^{\tilde{u}_k}} = \frac{1}{1 + e^{\tilde{u}_k}} \; . \tag{C.6}$$

berechnet werden. Die Wahrscheinlichkeit für ein Fehlerereignis ergibt sich für negative bzw. positive $\tilde{u}_k$ durch eine Fallunterscheidung:

$$P(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}}) = \begin{cases} \tilde{u}_k > 0, \hat{u}_k = 0 : P(u_k = 1|\tilde{\boldsymbol{c}}) = \frac{1}{1+e^{\tilde{u}_k}} \\ \tilde{u}_k < 0, \hat{u}_k = 1 : P(u_k = 0|\tilde{\boldsymbol{c}}) = \frac{1}{1+e^{-\tilde{u}_k}} \; . \end{cases} \tag{C.7}$$

Durch Betragsbildung von $\tilde{u}_k$ können die beiden Fälle in (C.7) zu

$$P(\hat{u}_k \neq u_k|\tilde{\boldsymbol{c}}) = \frac{1}{1 + e^{|\tilde{u}_k|}} \tag{C.8}$$

zusammengefasst werden.

Umwandlung einer reellen Zahl in eine Festkommazahl

Die Menge $\mathbb{FP}_{n,b}$ umfasst die Untermenge der reellen Zahlen, die als Festkommazahl mit n Bit bei einer Auflösung von b Bit darstellbar sind. Im Folgenden wird beschrieben, wie die Funktion $\mathrm{fp}_{n,b}(x)$ eine Zahl $x \in \mathbb{R}$ in $\mathbb{FP}_{n,b}$ abbildet. Dabei soll $\mathrm{fp}^{\mathrm{Z}}_{n,b}(x)$ das Bild von x in $\mathbb{FP}_{n,b}$ in binärer Darstellung angeben und $\mathrm{fp}^{\mathrm{Z}}_{n,b}(x)$ die Interpretation der Festkommazahl als vorzeichenbehafteter Integerwert sein.

Die Mächtigkeit von $\mathbb{FP}_{n,b}$ ist 2^n. Bei vorzeichenbehafteter Darstellung gibt das erste Bit das Vorzeichen an, so dass 2^{n-1} negative und $2^{n-1}-1$ positive Zahlen, einschließlich 0, darstellbar sind. Der kleinste möglich Abstand zweier als Festkommazahl darstellbarer Zahlen $\mathrm{fp}_{n,b}(x)$ in $\mathbb{FP}_{n,b}$ ist durch die Auflösung 2^{-b} und damit der Anzahl Bits b die für die Darstellung des Nachkommaanteils verwendet werden festgelegt. Die kleinste darstellbare Zahl ist damit $x_{\min} = -2^n 2^{-b}$ und die größte $x_{\max} = (2^n - 1)2^{-b}$. Zwischen $x_{\min}$ und $x_{\max}$ ist $x \in \mathbb{FP}_{n,b}$ in Schritten von 2^{-b} quantisiert.

Die Umwandlung einer Zahl $x \in \mathbb{R}$ in eine als Festkommazahl $\mathrm{fp}^{i}_{n,b}(x) \in \mathbb{FP}_{n,b}$ erfolgt in zwei Schritten, *Quantisierung* und *Rundung*. Zuerst wird x als

$$x' = \lfloor 2^b x + 0.5 \rfloor \tag{C.9}$$

quantisiert und anschließend auf den Darstellungsbereich gerundet:

$$x'' = \min(x', 2^{n-1} - 1) \tag{C.10}$$

$$\mathrm{fp}^{\mathrm{Z}}_{n,b}(x) = \max(-2^{n-1}, x'') \; . \tag{C.11}$$

Die Realwertdarstellung ergibt sich als $\mathrm{fp}_{n,b}(x) = \mathrm{fp}^{\mathrm{Z}}_{n,b}(x) \cdot 2^{-b}$.

Zur Verdeutlichung soll die Zahl $x = \pi$ als Festkommazahl $\mathrm{fp}_{4,1}(\pi)$ dargestellt werden. Es ergibt laut (C.9) sich $x' = \lfloor 2\pi + 0.5 \rfloor = 6$. Nach der, hier nicht relevanten, Rundung laut (C.11) ist also $\mathrm{fp}^{\mathrm{Z}}_{n,b}(\pi) = 6$. Es ergeben sich damit für π die folgenden unterschiedlichen Darstellungen:

$$\mathrm{fp}^{\mathrm{Z}}_{n,b}(\pi) = 6 \tag{C.12}$$

$$\mathrm{fp}^{\mathrm{B}}_{n,b}(\pi) = 0110_2 \text{ als Integer bzw. als Festkommazahl } 011_2, 0_2 \; ; \tag{C.13}$$

$$\mathrm{fp}_{n,b}(\pi) = 6 \cdot 0,5 = 3,0 \approx \pi \; . \tag{C.14}$$

Damit ergibt sich bei dem Umwandeln von reellen in Festkommazahlen ein Quantisierungseffekt. Für das obige Beispiel für die Umwandlung in $\mathbb{FP}_{n,b}$ folgt diese der Quantisierungskennlinie in Abb. C.1.

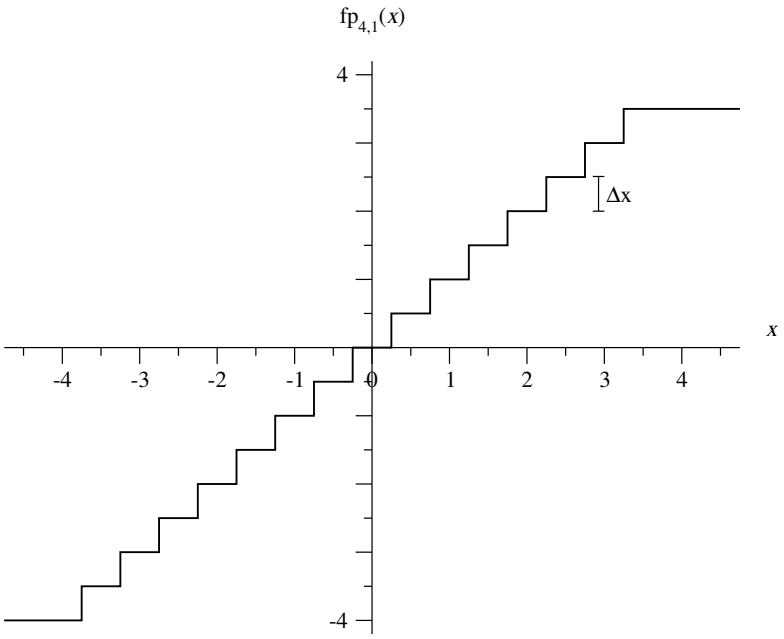

Abbildung C.1: Quantisierungskennlinie eines 4-Bit Festkomma-Quantisierers mit 3 Bit ganzzahligem Anteil mit 1 Bit Nachkommaanteil (Auflösung $\Delta x = 0.5$).

Jacobi-Logarithmus und die Max-Log-Approximation

Der Term $\ln(e^a + e^b)$ kann umgestellt

$$\begin{aligned} \ln(e^a + e^b) &= \ln\left(e^a[1 + e^{(b-a)}]\right) \\ &= \ln(e^a) + \ln\left(1 + e^{-(a-b)}\right) \\ &= a + \ln\left(1 + e^{-(a-b)}\right) \ , \end{aligned} \tag{C.15}$$

werden um eine günstige Approximation zu ermöglichen. Die Reihenentwicklung von $\ln(1 + e^{-(a-b)})$ liefert

$$\ln(1 + e^{-(a-b)}) = e^{-(a-b)} - \frac{e^{-2(a-b)}}{2} + \ldots + (-1)^{n+1}\frac{e^{-n(a-b)}}{n} + \ldots \tag{C.16}$$

Ein Abbruch der Reihe in (C.16) bietet sich an, wenn $(a - b)$ klein ist, also $a > b$ gilt. Selbiges gilt für $\ln(e^a + e^b) = b + \ln\left(1 + e^{-(b-a)}\right)$ und $b > a$. Damit ergibt sich die bekannte Max-Log-Approximation [KB90]

$$\begin{aligned} \ln(e^a + e^b) &= \max(a, b) + \ln(1 + e^{-|a-b|}) \\ &\approx \max(a, b) \ . \end{aligned} \tag{C.17}$$

Ähnlich läßt sich zeigen, dass für $a > b$ gilt:

$$\begin{aligned} \ln(e^a - e^b) &= \max(a, b) - \ln(1 - e^{-|a-b|}) \\ &\approx \max(a, b) \ . \end{aligned} \tag{C.18}$$

In Anlehnung an (C.17) und (C.18) wird die Operation

$$\max^{\star}(a,b) := \ln(e^a + e^b) = \max(a,b) + \ln(1 + e^{-|a-b|}) \tag{C.19}$$

definiert. Die beiden Operationen können mit

$$\max_{i=1\ldots I}^{\star}(a_i) = \max^{\star}(a_I, \max^{\star}(a_{I-1}, \max^{\star}(\ldots \max^{\star}(a_1, a_2)\ldots))) \; . \tag{C.20}$$

auch rekursiv angewandt werden, womit sie auf Summen von Exponentialtermen anwendbar sind, und

$$\max_{i=1\ldots I}(a_i) \approx \max_{i=1\ldots I}^{\star}(a_i) = \ln \sum_{i=1}^{I} e^{a_i} \tag{C.21}$$

gilt.

Summe von statistisch unabhängigen LLRs

Sei a eine binäre Zufallsvariable mit $a \in \{0,1\}$ und $b_1, \ldots, b_i, \ldots, b_I$ statistisch unabhängige Zufallsvariablen bei gegebenem a. Dann gilt

$$\mathrm{P}(b_1, \ldots, b_i, \ldots, b_I | a) = \mathrm{P}(b_1|a) \cdot \ldots \cdot \mathrm{P}(b_i|a) \cdot \ldots \cdot \mathrm{P}(b_I|a) \; . \tag{C.22}$$

Durch Anwendung des Satz von Bayes kann die a-posteriori Wahrscheinlichkeit für a als

$$\mathrm{P}(a|b_1, \ldots, b_i, \ldots, b_I) = \frac{\mathrm{P}(b_1|a) \cdot \ldots \cdot \mathrm{P}(b_i|a) \cdot \ldots \cdot \mathrm{P}(b_I|a) \cdot \mathrm{P}(a)}{\sum\limits_{a=0,a=1} \mathrm{P}(b_1|a) \cdot \ldots \cdot \mathrm{P}(b_i|a) \cdot \ldots \cdot \mathrm{P}(b_I|a) \cdot \mathrm{P}(a)} \tag{C.23}$$

angegeben werden. Damit ist auch der a-posteriori LLR von a als Summe von LLRs darzustellen:

$$\ln \frac{\mathrm{P}(a=0|b_1, \ldots, b_i, \ldots, b_I)}{\mathrm{P}(a=1|b_1, \ldots, b_i, \ldots, b_I)} \tag{C.24}$$

$$= \ln \frac{\mathrm{P}(b_1|a=0) \cdot \ldots \cdot \mathrm{P}(b_i|a=0) \cdot \ldots \cdot \mathrm{P}(b_I|a=0) \cdot \mathrm{P}(a=0)}{\mathrm{P}(b_1|a=1) \cdot \ldots \cdot \mathrm{P}(b_i|a=1) \cdot \ldots \cdot \mathrm{P}(b_I|a=1) \cdot \mathrm{P}(a=1)} \tag{C.25}$$

$$= \ln \frac{\mathrm{P}(b_1|a=0)}{\mathrm{P}(b_1|a=1)} + \ldots + \ln \frac{\mathrm{P}(b_i|a=0)}{\mathrm{P}(b_i|a=1)} + \ldots + \ln \frac{\mathrm{P}(b_I|a=0)}{\mathrm{P}(b_I|a=1)} + \ldots + \ln \frac{\mathrm{P}(a=0)}{\mathrm{P}(a=1)} \; . \tag{C.26}$$

In (C.26) ist als letzter Term auch die a-priori Wahrscheinlichkeit enthalten. Also sind i statistisch unabhängige LLRs für eine Zufallsvariable, inklusive der der LLR-Darstellung der a-priori Wahrscheinlichkeit, durch Addition zu einem einzigen LLR kombinierbar.

Definition der komplementären Fehlerfunktion

$$\mathrm{erfc}(x) = 1 - \mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-n^2} dn \tag{C.27}$$

Prüfmatrix des $R = 1/2$ LDPC-Codes nach IEEE 802.16e

Hier wird die Konstruktion des LDPC-Codes mit $R = 1/2$ aus [IEE05] für die in dieser Arbeit verwendete Infowortlänge von $K = 288$ angegeben. Für alle Infowortlängen dient eine sogenannte *Modellmatrix* als Basis. Diese ist in (C.28) angegeben. Aus $\boldsymbol{H}_{\mathrm{bm}}$ wird die Prüfmatrix $\boldsymbol{H}$ konstruiert.

Die Matrix $\boldsymbol{H}$ wird aus $\boldsymbol{H}_{\mathrm{bm}}$ gewonnen, indem deren Einträge durch Matrizen $\boldsymbol{P}_{i,j}$ der Dimension $z_f \times z_f$ ersetzt werden. Für $K = 288$ bzw. $N = 576$ ist $z_f = 24$. Abhängig von dem Wert $H_{\mathrm{bm},i,j}$ ist die Matrix $\boldsymbol{P}_{i,j}$ eine Nullmatrix oder eine um $H_{\mathrm{bm},i,j}$ zyklisch nach rechts verschobene Einheitsmatrix:

$$\boldsymbol{P}_{i,j} = \begin{cases} H_{\mathrm{bm},i,j} \geq 0 & : \text{ um } \lfloor \frac{H_{\mathrm{bm},i,j} z_f}{2304} \rfloor \text{ zyklisch nach rechts verschobene} \\ & \quad z_f \times z_f \text{ Einheitsmatrix} \\ H_{\mathrm{bm},i,j} < 0 & : z_f \times z_f \text{ Nullmatrix} \end{cases}$$

Die Dimensionen der Matrix $\boldsymbol{H}$ sind also um den Faktor z_f größer als die von $\boldsymbol{H}_{\mathrm{bm}}$. Für $K = 288$ und $R = 1/2$ ergibt sich daher eine Prüfmatrix der Dimensionen $576 \times 288 = N \times (N - P)$.

$$\boldsymbol{H}_{\mathrm{bm}} = \left[\begin{array}{cccccccccccccccccccccccc}
-1 & 94 & 73 & -1 & -1 & -1 & -1 & -1 & 55 & 83 & -1 & -1 & 7 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & 27 & -1 & -1 & -1 & 22 & 79 & 9 & -1 & -1 & -1 & 12 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & 24 & 22 & 81 & -1 & 33 & -1 & -1 & -1 & 0 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
61 & -1 & 47 & -1 & -1 & -1 & -1 & -1 & 65 & 25 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & 39 & -1 & -1 & -1 & 84 & -1 & -1 & 41 & 72 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & 46 & 40 & -1 & 82 & -1 & -1 & -1 & 79 & 0 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & 95 & 53 & -1 & -1 & -1 & -1 & -1 & 14 & 18 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 \\
-1 & 11 & 73 & -1 & -1 & -1 & 2 & -1 & -1 & 47 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\
12 & -1 & -1 & -1 & 83 & 24 & -1 & 43 & -1 & -1 & -1 & 51 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 \\
-1 & -1 & -1 & -1 & -1 & 94 & -1 & 59 & -1 & -1 & 70 & 72 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 \\
-1 & -1 & 7 & 65 & -1 & -1 & -1 & -1 & 39 & 49 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 \\
43 & -1 & -1 & -1 & -1 & 66 & -1 & 41 & -1 & -1 & -1 & 26 & 7 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0
\end{array}\right] \tag{C.28}$$

Literaturverzeichnis

[ABP+04] A. Ayda, P. Bahl, J. Padhye, A. Wolman und Lidong Zhou, "A multi-radio unification protocol for IEEE 802.11 wireless networks," in *Proc. Int. Conf. on Broadband Networks (BroadNets)*, San José, CA, USA, Okt. 2004, S. 344–254.

[BB99] M. Bossert und M. Breitbach, *Digitale Netze*. B. G. Teubner, 1999.

[BCJR74] L. R. Bahl, J. Cocke, F. Jelinek und J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, Bd. 20, S. 284–287, März 1974.

[BF64] R. J. Benice und A. H. Frey, Jr., "An analysis of retransmission systems," *IEEE Transactions on Communication Technology*, Bd. 12, Nr. 4, S. 135–145, Dez. 1964.

[BFH06] M. M. Butt, J. Ch. Fricke und P. A. Hoeher, "Reliability-based packet combining with application to interleave-division multiple access," in *Proc. 4th International Symposium on Turbo Codes & Related Topics (ISTC) in conjunction with International ITG-Conference on Source and Channel Coding (SCC)*, Munich, Germany, Apr. 2006.

[BG96] C. Berrou und A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Transactions on Communications*, Bd. 44, S. 1261–1271, Okt. 1996.

[BGiAR07] F. Brännström, A. Graell i Amat und L. K. Rasmussen, "A general structure for rate-compatible concatenated codes," *IEEE Communications Letters*, Bd. 11, S. 437–439, Mai 2007.

[BGT93] C. Berrou, A. Glavieux und P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE International Conference on Communications (ICC)*, Geneva, Switzerland, Mai 1993, S. 1064–1070.

[BL05] X. Bao und J. Li, "Decode-amplify-forward (DAF): A new class of forwarding strategy for wireless relay channels," in *Proc. IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, New York, NY, USA, Jun. 2005, S. 816–820.

[BSMM99] I. N. Bronstein, K. A. Semendjajew, G. Musiol und H. Mühlig, *Taschenbuch der Mathematik*, 4th Ausg. Verlag Harri Deutsch, 1999.

[BY04] R. A. Berry und E. M. Yeh, "Cross-layer wireless resource allocation," *IEEE Signal Processing Magazine*, Bd. 21, S. 59–68, Sep. 2004.

[Cha85] D. Chase, "Code combining–a maximum likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Transactions on Communications*, Bd. 33, S. 385–393, 1985.

[Coo83] G. H. Cooper, "An argument for soft layering of protocols," Massachusetts Institute of Technology, Techn. Ber., Mai 1983.

[CT97] H. A. Cirpan und M. K. Tsatsanis, "Chip interleaving in direct sequence CDMA systems," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Munich, Germany, Apr. 1997, S. 3877–3880.

[CTWH00] P. Coulton, C. Tanriover, B. Wright und B. Honary, "Simple hybrid type II ARQ technique using soft output information," *Electronics Letters*, Bd. 36, Nr. 20, S. 1716–1717, Sep. 2000.

[DPZ04] R. Draves, J. Padhye und B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," in *Proc. ACM SIGCOMM*, Portland, Oregon, USA, Aug.–Sep. 2004, S. 133–144.

[DRKT97] R. Dube, C. D. Rais, Kuang-Yeh Wang und S. K. Tripathi, "Signal stability-based adaptive routing (SSA) for adhoc mobile networks," *IEEE Personal Communications Magazine*, Bd. 4, S. 36–45, 1997.

[Eli55] P. Elias, "Coding for noisy channels," *IRE International Convention Record*, S. 37–47, 1955.

[ETSa] ETSI, "Digital video broadcasting (DVB); second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications," (ETSI EN 302 307 V1.1.1).

[ETSb] ETSI 3rd Generation Partnership Project (3GPP), "Technical specification group GSM/EDGE radio access network; Channel coding," 3GPP TS 45.003 version 6.9.0 Release 6.

[ETSc] ——, "Universal mobile telecommunications system (UMTS); Multiplexing and channel coding (TDD)," (3GPP TS 25.222 version 6.1.0 Release 6).

[ETSd] ——, "Universal mobile telecommunications system (UMTS); Quality of service (QoS) concept and architecture," 3GPP TS 23.107 version 5.8.0 Release 5.

[ETSe] ——, "Universal mobile telecommunications system (UMTS); Radio interface protocol architecture," 3GPP TS 25.301 version 7.3.0 Release 7.

[FBH08] J. Ch. Fricke, M. M. Butt und P. A. Hoeher, "Quality-oriented adaptive forwarding for wireless relaying," *IEEE Communications Letters*, Bd. 12, Nr. 3, S. 200–202, März 2008.

[FBSH98] M. P. C. Fossorier, F. Burkert, Shu Lin und J. Hagenauer, "On the equivalence between SOVA and max-log-MAP decodings," *IEEE Communications Letters*, Bd. 2, S. 137–139, Mai 1998.

[FH07] J. Ch. Fricke und P. A. Hoeher, "Word error probability estimation by means of a modified Viterbi decoder," in *Proc. IEEE Vehicular Technology Conference (VTC-Fall)*, Baltimore, MD, USA, Sep.–Okt. 2007, S. 1113–1116.

[FH08a] ——, "Reliability-based retransmission criteria for hybrid ARQ," 2008, angenommen zur Veröffentlichung in *IEEE Transactions on Communications*.

[FH08b] ——, "Why it is better to measure bit instead of word error probabilities," in *Proc. 7th International ITG-Conference on Source and Channel Coding (SCC)*, Ulm, Germany, Jan. 2008.

[FHS05a] J. Ch. Fricke, P. A. Hoeher und H. Schoeneich, "An interleave-division multiple access based system proposal for the 4G uplink," in *Proc. 14th IST Mobile & Wireless Communications Summit (IST Summit)*, Dresden, Germany, Jun. 2005, paper no. 234.

[FHS05b] ——, "A system proposal for the 4G uplink based on interleave-division multiple access," in *Proc. 2005 4GMF Annual Conference (4GMF2005)*, San Diego, California, USA, Jul. 2005, S. 50–55.

[FHS05c] ——, "An uplink proposal based on interleave-division multiple access," in *Proc. Wireless World Research Forum 14th Meeting (WWRF14)*, San Diego, California, USA, Jul. 2005, paper no. WG4-16.

[FK06] F. H. Fitzek und M. D. Katz, Eds., *Cooperation in Wireless Networks: Principles and Applications*. Springer, 2006.

[FMBT05] G. Ferrari, S. A. Malvassori, M. Bragalini und O. K. Tonguz, "Physical layer-constrained routing in ad-hoc wireless networks: A modified AODV protocol with power control," in *Proc. Int. Workshop on Wireless Ad-hoc Networks 2005 (IWWAN 2005)*, London, UK, Mai 2005.

[FRH07] J. Ch. Fricke, M. I. Rafique und P. A. Hoeher, "Reliability-based routing metrics," in *Proc. IEEE Vehicular Technology Conference (VTC-Fall)*, Baltimore, MD, USA, Sep.–Okt. 2007, S. 1057–1061.

[Fri96] B. Friedrichs, *Kanalcodierung: Grundlagen und Anwendungen in modernen Kommunikationssystemen*. Springer-Verlag, 1996.

[FSH06] J. Ch. Fricke, H. Schoeneich und P. A. Hoeher, "Reliability-based HARQ using word error probabilities," in *Proc. NEWCOM-ACoRN Joint Workshop (NAW)*, Vienna, Austria, Sep. 2006.

[FSMH05] J. Ch. Fricke, M. Sandell, J. Mietzner und P. A. Hoeher, "Impact of the Gaussian approximation on the performance of the probabilistic data association MIMO decoder," *EURASIP Journal on Wireless Communications and Networking*, Bd. 2005, Nr. 5, S. 796–800, Dez. 2005.

[Gal62] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, Bd. IT-8, S. 21–28, Jan. 1962.

[GJ06] K. S. Gomadam und S. A. Jafar, "Optimizing soft information in relay networks," in *Proc. 40th Asilomar Conference on Signals, Systems and Computers (ACSSC)*, Pacific Grove, CA, USA, Okt. 2006, S. 18–22.

[HA03] M. O. Hasna und M.-S. Alouini, "End-to-end performance of transmission systems with relays over Rayleigh-fading channels," *IEEE Transactions on Wireless Communications*, Bd. 2, Nr. 6, S. 1126–1131, Nov. 2003.

[Hag88] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Transactions on Communications*, Bd. 36, S. 389–400, Apr. 1988.

[HH89] J. Hagenauer und P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Nov. 1989, S. 1680–1686.

[Hoe92] P. Hoeher, "A statistical discrete-time model for the WSSUS multipath channel," *IEEE Transactions on Vehicular Technology*, Bd. 41, S. 461–468, Nov. 1992.

[Hoe95] ——, "Optimal subblock-by-subblock detection," *IEEE Transactions on Communications*, Bd. 43, S. 714–717, Feb./Mar./Apr. 1995.

[Hoe00] ——, "Adaptive modulation and channel coding using reliability information," in *Proc. International OFDM-Workshop (InOWo)*, Hamburg, Germany, Sep. 2000, S. 14.1–14.4.

[HROW07] P. A. Hoeher, P. Robertson, E. Offer und T. Woerz, "The soft-output principle–reminiscences and new developments," *European Transactions on Telecommunications*, Bd. 18, S. 829–835, Dez. 2007.

[HSF08] P. A. Hoeher, H. Schoeneich und J. Ch. Fricke, "Multi-layer IDMA: Theory and practice," *European Transactions on Telecommunications*, Bd. 19, Nr. 5, S. 523–536, Aug. 2008.

[HSL00] P. Hoeher, U. Sorger und I. Land, "Log-likelihood values and Monte Carlo simulation – some fundamental results," in *Proc. 2nd International Symposium on Turbo Codes & Related Topics (ISTC)*, Brest, France, Sep. 2000, S. 43–46.

[HW07] P. A. Hoeher und Wen Xu, "Multi-layer interleave-division multiple access for 3GPP long term evolution," in *Proc. IEEE International Conference on Communications (ICC)*, Glasgow, United Kingdom, Jun. 2007, S. 5508–5513.

[ICPW97] J. Inouye, S. Cen, C. Pu und J. Walpole, "System support for mobile multimedia applications," in *Proc. 7th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, St. Louis, MO, USA, Mai 1997, S. 143–154.

[IEE05] IEEE, "IEEE standard for local and metropolitan area networks – Part 16: Air interface for fixed and mobile broadband wireless access systems; Amendment 2: Physical and medium access control layers for combined fixed and mobile operation in licensed bands and corrigendum 1," Feb. 2005, IEEE Std 802.16e-2005 and IEEE Std 802.16-2004/Cor1-2005.

[Inta] International Telecommunication Union, "Error-correcting procedures for DCEs using asynchronous-to-synchronous conversion," Rec. ITU-T V.42 (03/2003).

[Intb] ——, "Terms and definitions related to quality of service and network performance including dependability," Rec. ITU-T E.800.

[ISO89] ISO/IEC Joint Technnical Committee for Information Technology, "Binary floating-point arithmetic for microprocessor systems," Jan. 1989, ISO/IEC 60559 - Ed. 2.0.

[ISO94] ISO/IEC, "Information technology - open systems interconnection - basic reference model: The basic model," November 1994, ISO/IEC 7598-1:1994(E).

[Joh28] J. B. Johnson, "Thermal agitation of electricity in conductors," *Physical Review*, Bd. 32, S. 97–109, Jul. 1928.

[JZ99] R. Johannesson und K. S. Zigangirov, *Fundamentals of Convolutional Coding*. Piscataway, N. J.: IEEE Press, 1999.

[KB90] W. Koch und A. Baier, "Optimum and sub-optimun detection of coded data disturbed by time-varying intersymbol interference," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, USA, Dez. 1990, S. 1679–1684.

[KFH08] A. Khan, J. Ch. Fricke und P. A. Hoeher, "Throughput improvement by reliability-dependent layer allocation in IDM," *Elecß-tronics Letters*, Bd. 44, Nr. 16, S. 958–960, Jul. 2008.

[KFL01] F. R. Kschischang, B. J. Frey und H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, Bd. 47, S. 498–519, Feb. 2001.

[KK05] V. Kawadia und P. R. Kumar, "A cautionary perspective on cross-layer design," *IEEE Wireless Communications Magazine*, Bd. 12, S. 3–11, Feb. 2005.

[Kok07] C. E. Koksal, *Wireless Mesh Networks*. Springer, 2007, Kap. 9, S. 227–243.

[KR05] J. F. Kurose und K. W. Ross, *Computer Networking–A Top-Down Approach Featuring the Internet*, 3rd Ausg. Pearson Education, 2005.

[Lan05] I. Land, "Reliability information in channel decoding: Practical aspects and information theoretical bounds," Dissertation, Christian-Abrechts-Universität zu Kiel, 2005.

[LC04] S. Lin und D. J. Costello, Jr., *Error Control Coding*, 2nd Ausg. Pearson Prentice Hall, 2004.

[LH00] I. Land und P. Hoeher, "Partially systematic rate 1/2 turbo codes," in *Proc. International Symposium on Turbo Codes & Related Topics (ISTC)*, Brest, France, Sep. 2000, S. 287–290.

[LH01] I. Land und P. A. Hoeher, "Using the mean reliability as a design and stopping criterion for turbo codes," in *Proc. IEEE Information Theory Workshop (ITW)*, Cairns, Australia, Sep. 2001, S. 27–29.

[LH03] ——, "New results on Monte Carlo bit error simulation based on the a posteriori log-likelihood ratio," in *Proc. 3rd International Symposium on Turbo Codes & Related Topics (ISTC)*, Brest, France, Sep. 2003, S. 531–534.

[LL04] Lihai Liu und Li Ping, "Iterative detection of chip interleaved CDMA systems in multipath channels," *Electronics Letters*, Bd. 40, Nr. 14, S. 884–885, Jul. 2004.

[LLL03] Li Ping, Lihai Liu und W. K. Leung, "A simple approach to near-optimal multiuser detection: Interleave-division multiple-access," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, Bd. 1, März 2003, S. 391–396.

[Loe94] H.-A. Loeliger, "A posteriori probabilities and performance evaluation of trellis codes," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Trondheim, Norway, Jun.–Jul. 1994, S. 335.

[LVWD06] Y. Li, B. Vucetic, T. F. Wong und M. Dohler, "Distributed turbo coding with soft information relaying in multi-hop relay networks," *IEEE Journal on Selected Areas in Communications*, Bd. 24, S. 2040–2050, Nov. 2006.

[LWLL02] Li Ping, K. Y. Wu, Lihai Liu und W. K. Leung, "A simple, unified approach to nearly optimal multiuser detection and space-time coding," in *Proc. IEEE Information Theory Workshop (ITW)*, Bangalore, India, Okt. 2002, S. 53–56.

[LYHH93] J. Lodge, R. Young, P. Hoeher und J. Hagenauer, "Separable MAP 'filters' for the decoding of product and concatenated codes," in *Proc. IEEE International Conference on Communications (ICC)*, Mai 1993, S. 1740–1745.

[MN97] D. J. MacKay und R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, Bd. 33, S. 457–458, März 1997.

[MW01] H. Michel und N. Wehn, "Turbo-decoder quantization for UMTS," *IEEE Communications Letters*, Bd. 5, Nr. 2, S. 55–57, Feb. 2001.

[Pap91] A. Papoulis, *Probability, Random Variables and Stochastic Processes*, 3rd Ausg. McGraw-Hill, Inc., 1991.

[PLL06] Peng Wang, Li Ping und Lihai Liu, "Power allocation for multiple access systems with practical coding and iterative multi-user detection," in *Proc. IEEE International Conference on Communications (ICC)*, Istanbul, Turkey, Jun. 2006, S. 4971–4976.

[Pro02] Y. V. Prohorov, *Encyclopaedia of Mathematics*, M. Hazewinkel, Ed. Springer-Verlag Berlin Heidelberg New York, 2002.

[Rap96] T. S. Rappaport, *Wireless Communications - Principles and Practice*. Prentice Hall PTR, 1996.

[RB98] A. R. Raghavan und C. W. Baum, "A reliability output Viterbi algorithm with applications to hybrid ARQ," *IEEE Transactions on Information Theory*, Bd. 44, S. 1214–1216, Mai 1998.

[RHV97] P. Robertson, P. Hoeher und E. Villebrun, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding," *European Transactions on Telecommunications*, Bd. 8, S. 119–125, 1997.

[RI04] V. T. Raisinghani und S. Iyer, "Cross-layer design optimizations in wireless protocol stacks," *Computer Communications*, Bd. 27, Nr. 8, S. 720–724, Mai 2004.

[RS90] R. Rom und M. Sidi, *Multiple Access Protocols: Performance and Analysis*. Springer-Verlag, 1990.

[RS03] A. Roongta und J. M. Shea, "Reliability-based hybrid ARQ using convolutional codes," in *Proc. IEEE International Conference on Communications (ICC)*, Anchorage, Alaska, Mai 2003, S. 2889–2893.

[RU01] T. J. Richardson und R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Transactions on Information Theory*, Bd. 47, S. 638–656, Feb. 2001.

[RVH95] P. Robertson, E. Villebrun und P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. IEEE International Conference on Communications (ICC)*, Bd. 2. IEEE, Jun. 1995, S. 1009–1013.

[RW93] L. K. Rasmussen und S. B. Wicker, "Comparison of two retransmission request mechanisms for trellis coded modulation," *Electronics Letters*, Bd. 29, S. 2162–2163, Dez. 1993.

[RW94] ——, "The performance of type-I trellis coded hybrid-ARQ protocols over AWGN and slowly fading channels," *IEEE Transactions on Information Theory*, Bd. 40, S. 418–428, März 1994.

[Sar88] D. V. Sarwate, "Computation of cyclic redundancy checks via table look-up," *Communications of the ACM*, Bd. 31, Nr. 8, S. 1008–1013, Aug. 1988.

[SB05] L. Snzecinski und M. Bacic, "Constellations design for multiple transmissions maximizing the minimum squared Euclidean distance," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, März 2005, S. 1066–1071.

[SBR06] D. P. Shepherd, F. Brännström und M. C. Reed, "Fidelity charts and stopping/termination criteria for iterative multiuser detection," in *Proc. 4th International Symposium on Turbo Codes & Related Topics (ISTC) in conjunction with International ITG-Conference on Source and Channel Coding (SCC)*, Munich, Germany, Apr. 2006.

[Sch08a] H. Schöneich, "Adaptiver Interleave-Divison Mehrfachzugriff (IDMA) mit Anwendung in der Mobilfunkkommunikation," Dissertation, Christian-Albrechts-Universität zu Kiel, 2008.

[Sch08b] N. Schrammar, "User selection in multi-user-MIMO systems," Diplomarbeit, Christian-Albrechts-Universität zu Kiel, Kiel, Jun. 2008.

[SH04] H. Schoeneich und P. A. Hoeher, "Adaptive interleave-division multiple access - a potential air interface for 4G bearer services and wireless LANs," in *Proc. 1st IEEE and IFIP Int. Conf. on Wireless and Optical Communications and Networks (WOCN 2004)*, Muscat, Oman, Jun. 2004, S. 179–182.

[Sha48] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, Bd. 27, S. 379–423 and 623–656, Jul. and Okt. 1948.

[She02] J. M. Shea, "Reliability-based hybrid ARQ," *Electronics Letters*, Bd. 38, S. 644–645, Jun. 2002.

[SHF05] H. Schoeneich, P. A. Hoeher und J. Ch. Fricke, "Adaptive 4G uplink proposal based on interleave-division multiple access," in *Proc. Proc. 28th General Assembly of International Union of Radio Science (URSI GA)*, New Delhi, India, Okt. 2005, paper no. C03.5.

[Sho03] A. Shokrollahi, "LDPC codes: An introduction," Apr. 2003, Digital Fountain, Inc.

[Sin77] P. S. Sindhu, "Retransmission error control with memory," *IEEE Transactions on Communications*, Bd. 25, S. 473–479, Mai 1977.

[Skl01] B. Sklar, *Digital Communications: Fundametals and Applications*, 2nd, Ed. Prentice Hall PTR, 2001.

[SM05a] M. R. Souryal und N. Moayeri, "Channel-adaptive relaying in mobile ad hoc networks with fading," in *Proc. IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, Santa Clara, CA, USA, Sep. 2005, S. 142–152.

[SM05b] V. Srivastava und M. Motani, "Cross-layer design: A survey and the road ahead," *IEEE Communications Magazine*, Bd. 43, S. 112–119, Dez. 2005.

[SRK03] S. Shakkottai, T. S. Rappaport und P. C. Karlsson, "Cross-layer design for wireless networks," *IEEE Communications Magazine*, Bd. 41, Nr. 10, S. 74–80, Okt. 2003.

[SV05] H. H. Sneessens und L. Vandendorpe, "Soft decode and forward improves cooperative communications," in *Proc. 6th IEE Int. Conf. on 3G and Beyond*, Washington, DC, USA, Nov. 2005, S. 1–4.

[SW02] H. Stark und J. W. Woods, *Probability and Random Processes with Applications to Signal Processing*, 3rd Ausg. Uppder Saddle River, New Jersey 07458: Prentice-Hall, 2002.

[SWR98] S. Singh, M. Woo und C. S. Raghavendra, "Power-aware routing in mobile ad hoc networks," in *Proc. International Conference on Mobile Computing and Networking (MobiCom)*, Dallas, Texas, USA, Okt. 1998, S. 181–190.

[Tho07] R. Thobaben, "Iterative Quellen- und Kanalcodierung für Codes variabler Länge," Dissertation, Christian-Albrechts-Universität zu Kiel, Kiel, 2007.

[TL07] R. Thobaben und E. G. Larsson, "Sensor-network-aided cognitive radio: On the optimal receiver for estimate-and-forward protocols applied to the relay channel," in *Proc. 41st Asilomar Conference on Signals, Systems and Computers (ACSSC)*, Pacific Grove, CA, USA, Nov. 2007.

[TOA01] K. Thorén, V. Öwall und J. B. Anderson, "Precision issues in the implementation of BCJR decoders," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Washington, DC, USA, Jun. 2001, S. 289.

[TVPH03] V. Tripathi, E. Visotsky, R. Peterson und M. L. Honig, "Reliability-based type II hybrid ARQ schemes," in *Proc. IEEE International Conference on Communications (ICC)*, Anchorage, Alaska, USA, Mai 2003.

[VAG05] C. Verikoukis, L. Alonso und T. Giamalis, "Cross-layer optimization for wireless systems: A European research key challenge," *IEEE Communications Magazine*, Bd. 43, S. 1–3, Jul. 2005.

[Vit67] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, Bd. 13, S. 260–269, Apr. 1967.

[VST+05] E. Visotsky, Y. Sun, V. Tripathi, M. L. Honig und R. Peterson, "Reliability-based incremental redundancy with convolutional codes," *IEEE Transactions on Communications*, Bd. 53, S. 987–997, Jun. 2005.

[WH61] J. M. Wozencraft und M. Horstein, "Coding for two-way channels," Research laboratory of Electronics, Massachusetts Institute of Technology, Techn. Ber. 383, Jan. 1961.

[Wic95] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Prentice Hall, 1995.

[Wil93] R. Williams, "A painless guide to CRC error detection algorithms," Aug. 1993, zuletzt abgerufen: 23.11.2007. [Online]. Available: http://www.ross.net/crc/crcpaper.html

[WK07] P. Weitkemper und K.-D. Kammeyer, "Power optimization of IDMA systems with different target BER constraints," in *Proc. IEEE Vehicular Technology Conference (VTC-Spring)*, Dublin, Ireland, Apr. 2007, S. 2812–2816.

[WL85] K. A. Witzke und C. Leung, "A comparison of some error detecting CRC code standards," *IEEE Transactions on Communications*, Bd. 33, S. 996–998, Sep. 1985.

[WWKK08] P. Weitkemper, D. Wübben, V. Kühn und K.-D. Kammeyer, "Soft information relaying for wireless networks with error-prone source-relay link," in *Proc. International ITG-Conference on Source and Channel Coding (SCC)*, Ulm, Germany, Jan. 2008.

[YI80] H. Yamamoto und K. Itoh, "Viterbi decoding algorithm for convolutional codes with repeat request," *IEEE Transactions on Information Theory*, Bd. 26, S. 540–547, Sep. 1980.

[ZHF05] E. Zimmermann, P. Herhold und G. Fettweis, "On the performance of cooperative relaying protocols in wireless networks," *European Transactions on Telecommunications*, Bd. 16, S. 5–16, 2005.

Stichwortverzeichnis

a-posteriori probability (APP), 41
Ablaufplanung, 113–121
ARQ-Protokoll, *siehe* automatic repeat request (ARQ)
automatic repeat request (ARQ), 19, 23, 32–34
    Go-Back-N (GBN), 34
    Hybrid –, *siehe* Hybrid ARQ (HARQ)
    Selective-Repeat (SR), 34
AWGN, 14

BCJR-Algorithmus, 25
    Log-, 142
    Max-Log-, 144, 159
Belief-Propagation-Algorithmus (BPA), 152
Binary Phase Shift Keying (BPSK), 14
bit error probability (BEP), 50–59
Bitfehlerrate (BER), 50
Bitfehlerwahrscheinlichkeit, *siehe* bit error probability (BEP)

Chip, 123
Code-Combining, 36
Codepolynom, 23
Coderate, 24
CRC-Codes, 31
cross-layer design, *siehe* schichtübergreifender Systementwurf
Cyclic-Redundancy-Check-Codes, *siehe* CRC-Codes

Decodierer, 39
    -iterativer, 27
Dienst (service), 5
Diversity-Combining, 35
Durchsatzeffizienz, 34, 115

Einflusslänge, 24
Empfänger, 17
expected hop count (EHC), 105, 108

Fading, 15
    frequenzselektives, 15, 16
    nicht-frequenzselektives, 15
    Rayleigh-, 15
Faktorgraph, 152
Faltungscodes, 24–26
    -parallel verkettete, 26–28
    -rekursive, 24
    -sytematische, 24
Festkommadarstellung, 42, 158
forward error correction (FEC), 23

güteorientierte Decodierung, 91–102
Gedächtnislänge, 24
Generatorpolynom, 24
    CRC-Code, 31
    Faltungscode, 24

Hamming-Distanz, 44
Hop, 103
Hybrid ARQ (HARQ), 23, 35–37
    Typ-I –, 35
    Typ-II –, 35
    Typ-III –, 36

Infobitenergie, 15
Infopolynom, 23
Inkrementelle Redundanz (IR), 37, 88
Intersymbolinterferenz (ISI), 17

Kanalkapazität, 20
Kanalmodell, 14
Komponentencodes, 26

Layer, 123
LDPC-Codes, 23, 28–30

Link, 103
Log-Distanz Pfaddämpfung, 16
Log-Likelihood Ratio (LLR), 13, 40–41
Log-Likelihood Verhältnis,
siehe Log-Likelihood Ratio (LLR)
Log-Likelihood Wert,
siehe Log-Likelihood Ratio (LLR)
Low-Density Parity-Check Codes,
siehe LDPC-Codes

maximum a-posteriori probability (MAP), 41
Maximum-Likelihood-Pfad, 148
minimum link count (MLC), 103
Mittelwertschätzer, 51, 60
Modulierungs- und Codierungsschema (MCS), 35
Multiple-Input Multiple-Output (MIMO), 14

Nachricht, 6
Nutzermetrik, 113, 115

Orthogonal-Frequency-Division-Multiplex (OFDM), 14
OSI-Schichtmodell, 6
OSSD, 145–147

Paket, 6
Pfadmetrik, 147
Pfadsuche, *siehe* Routing
Prüfmatrix, 161
Protocol-Data-Unit (PDU), 6
Protokoll, 5
Protokollkontrolinformation (PCI), 6

QoS, 17
-Attribut, 17
-Dreieck, 19
-Klasse, 17
-Parameter, 18
-Tetraeder, 19

Rückkanal, 33
Rauschvarianz, 15
Relay, 8
Repeater, 8
Round-Trip-Time (RTT), 33, 104
Router, 8, 103
Routing, 7, 103–112
-Algorithmus, 103
-Metrik, 103, 108–111
-Protokoll, 103
ROVA, 41, 150
-vereinfachter, 66, 151

Schätzer, 39
Scheduling, *siehe* Ablaufplanung
schichtübergreifender Systementwurf, 9
Schichtmodell, 5
OSI-, 6
Sender, 14
Service-Data-Unit (SDU), 6
SOVA, 149
Sum-Product-Algorithmus (SPA), 30, 152
Symbolenergie, 15
Systemdurchsatz, 115

throughput efficiency, *siehe* Durchsatzeffizienz
Trellisdiagramm, 25, 141
Tschebyschow-Ungleichung, 73

Viterbi-Algorithmus, 25
Log-, 147
Vorwärtsfehlerschutz, forward error correction (FEC), 12

word error probability (WEP), 60–73
Wortfehlerrate (WER), 60
Wortfehlerwahrscheinlichkeit,
siehe word error probability (WEP)

Zuverlässigkeitsinformation, 39, 40
Zweigmetrik, 147