# Bewertung von Codierverfahren für einen störungssicheren Datentransfer

Dem Fachbereich 11 / Mathematik der
Gerhard-Mercator-Universität - Gesamthochschule - Duisburg
zur Erlangung des akademischen Grades eines

Dr. rer. nat.

eingereichte Dissertation

von

Claudia Osmann

aus

Oberhausen

Tag der mündlichen Prüfung: 1. Oktober 1999
Referent: Prof. Dr. Wolfram Luther
Koreferent: Priv.-Doz. Dr. Gerhard Haßlinger

# Inhaltsverzeichnis

# Kapitel 1

# Einleitung

Gegenstand der vorliegenden Arbeit ist die Modellierung zufällig auftretender Störungen, die während des Transfers von Information über reale Übertragungskanäle eine Verfälschung der zu übermittelnden Information verursachen können. Im Rahmen dieser Arbeit präsentieren wir hierzu ein geeignetes Störungsmodell, mit dem digitale Übertragungskanäle mit Gedächtnis und damit Störungen unterschiedlicher Form und verschiedenen Ausmaßes abgebildet und modelliert werden können. Dabei werden in dem vorgestellten Modell die Vorteile und Stärken bereits bekannter Kanalmodelle mit Gedächtnis miteinander verknüpft und gleichzeitig deren Mängel weitestgehend eliminiert.

Einen weiteren Schwerpunkt der Arbeit bilden die Bewertung und der Vergleich verschiedener Codierverfahren, die zur Absicherung der zu übermittelnden Information gegenüber zufällig auftretenden Störungen in praktischen Anwendungen eingesetzt werden. Zu diesem Zweck führen wir die Restfehlerwahrscheinlichkeit eines Codierverfahrens als Kriterium bzw. Maß für dessen Effizienz ein. Auf der Basis des zuvor vorgestellten Störungsmodells werden neue Algorithmen und Näherungsformeln entwickelt. Anhand derer lassen sich zunächst die Korrekturfähigkeiten unterschiedlicher Codierverfahren mit Blick auf die Korrektur von zufällig verteilten Fehlern sowie von Bündelfehlern erfassen. Ferner läßt sich mittels der entworfenen Algorithmen die Restfehlerwahrscheinlichkeit eines Codierverfahrens ermitteln und so dessen Leistungsfähigkeit beurteilen. Hierbei legen wir besonderen Wert auf eine homogene und in sich geschlossene Darstellung der mathematischen Hintergründe. Auf diese Weise können die Zusammenhänge zwischen den theoretischen Grundlagen und der praktischen Anwendung deutlich herausgestellt und mathematisch korrekt belegt werden. Eine Auswertung der entwickelten Konzepte und Methoden bestätigt schließlich die Vermutung, dass unter gewissen Übertragungsbedingungen der Einsatz bestimmter Codierverfahren wesentlich effizienter und damit zu bevorzugen ist.

Da in der modernen Informationsgesellschaft dem Transfer von Information sowie deren Verarbeitung eine immer größere Bedeutung zukommt, sind in den letzten Jahren die Anforderungen an digitale Datenübertragungssysteme stetig gestiegen. Aus diesem Grund sind immer leistungsfähigere und zuverlässigere digitale Datenübertragungssysteme notwendig. Eine kritische Entwicklung in diesem Zusammenhang sind die rapide steigenden Übertragungsraten sowie die – je nach praktischer Anwendung – völlig unterschiedlichen qualitativen Anforderungen an digitale Übertragungssysteme. Generell gilt dabei für alle Anwendungen, dass in immer kürzerer Zeit immer mehr Information zu erfassen, zu verarbeiten, zu übertragen und gegebenenfalls zu speichern ist. Dies gilt gleichermaßen für die

Satellitenkommunikation sowie für hochratige Datenübertragungen, wie beispielsweise bei Videoübertragungen, also Videokonferenzen oder Video-on-Demand.

Während der Übermittlung von Information über ein physikalisches Übertragungsmedium werden wir stets mit der Problematik konfrontiert, dass die zu übermittelnde Information unter Umständen durch zufällig auftretende Störungen verfälscht werden kann. Ein im Hinblick auf die stetig steigenden Anforderungen an digitale Datenübertragungssysteme wichtiger Aspekt ist daher die Gewährleistung der Zuverlässigkeit und der Qualität der Übertragung. Schließlich können zufällig auftretende Störungen, die durch eine fremde Quelle verursacht werden, bei sensiblen Anwendungen, wie z.B. der Satellitenkommunikation oder bei hochratigen Datenübertragungen, zur Unbrauchbarkeit der Daten und Nicht-Ausführbarkeit von Programmen führen. Sie verursachen dabei entweder einzelne, zufällig verteilte Übertragungsfehler oder Fehler, die in unmittelbarer zeitlicher Nähe auftreten. In welcher Form und in welchem Ausmaß Übertragungsfehler schließlich auftreten, hängt stark vom physikalischen Medium ab.

Um eine möglichst zuverlässige und sichere Übertragung über verschiedenste physikalische Träger zu gewährleisten, ist in den letzten Jahren eine Vielzahl von Codierverfahren entwickelt worden. Dabei verfolgen diese unterschiedliche mathematische Ansätze, um zufällig auftretende Störungen zu erkennen und gegebenenfalls zu korrigieren. Dies führt dazu, dass einige Verfahren mit zufällig verteilten Fehlern sehr gut verfahren können, mit Fehlern in zeitlicher Nähe dagegen weniger gut. Im Rahmen dieser Arbeit werden wir vor dem Hintergrund unterschiedlicher Übertragungsbedingungen eine Bewertung verschiedener Codierverfahren im Hinblick auf ihre Leistungsfähigkeit vornehmen. Hierbei beziehen wir insbesondere die unterschiedlichen mathematischen Ansätze der einzelnen Verfahren und damit ihr unterschiedliches Fehlerkorrekturverhalten mit ein.

Angesichts dieser Zielsetzung ist ein wesentlicher Schwerpunkt der Arbeit die Entwicklung eines Störungsmodells, das den während der Übertragung auftretenden Fehlerprozeß möglichst realistisch nachbildet. Das bedeutet, dass mittels des Kanalmodells das Gedächtnis des Übertragungskanals in angemessener Weise zu modellieren ist. Daher wird in der Arbeit ein endliches Markov Modell als Kanalmodell präsentiert, mit dem einzelne, zufällig verteilte Übertragungsfehler und Übertragungsfehler in zeitlicher Nähe gleichermaßen modelliert werden können. Dieses Kanalmodell verbindet auf diese Weise die Vorteile und Stärken verschiedener bekannter Kanalmodelle mit Gedächtnis und hebt gleichzeitig deren Erneuerungscharakter auf. Folglich erzielen wir derart eine verbesserte und realistischere Modellierung von Bündelfehlern bzw. von sogenannten *bad*-Phasen.

Zur Modellierung nutzen wir Methoden aus der Stochastik, ein endliches Markov Modell bzw. einen sogenannten Markov'schen Hintergrundprozeß mit einem endlichen Zustandsraum. Dieses endliche Markov Modell bildet als Kanalmodell die Grundlage zur Entwicklung von Methoden und Verfahren, mit denen wir schließlich verschiedene zur Fehlerkorrektur eingesetzte Codierverfahren a priori hinsichtlich ihrer Effizienz beurteilen können. Als Kriterium für eine derartige Bewertung dient die Restfehlerwahrscheinlichkeit eines Codierverfahrens, d.h. die Wahrscheinlichkeit, mit der die Information trotz des Einsatzes eines Codierverfahrens verfälscht zum Empfänger gelangt. Basierend auf dem präsentierten Kanalmodell gelingt es, rekursive Gleichungen zur Berechnung der Restfehlerwahrscheinlichkeit eines Codierverfahrens herzuleiten. Hierbei werden die unterschiedlichen mathematischen Ansätze, insbesondere die unterschiedlichen algebraischen Strukturen, verschiedener Codierungsverfahren geeignet berücksichtigt. Dieser in der Arbeit gewählte Ansatz bietet

gegenüber bekannten Konzepten deutliche Vorteile. So werden in den bislang bekannten Ansätzen die unterschiedlichen mathematischen Strukturen verschiedener Codierverfahren im allgemeinen in eine derartige Beurteilung ihrer Leistungsfähigkeit nicht miteinbezogen. Ferner werden Kanalmodelle in der Literatur lediglich dazu genutzt, um mit Hilfe von Simulationen dasjenige Codierverfahren zu bestimmen, das den Anforderungen den gegebenen Übertragungsbedingungen am besten genügt. Diesem mehr experimentellen Ansatz stellen wir in dieser Arbeit demnach eine theoretischere Betrachtungsweise gegenüber.

Die Auswertung des von uns vorgestellten Kanalmodells bzw. der darauf aufbauenden und von uns entworfenen Algorithmen stellt eines der wesentlichen Ergebnisse dieser Arbeit dar. Hierbei ist das von uns entworfene Verfahren zur Beurteilung der Leistungsfähigkeit verschiedener Codierverfahren das einzige uns bekannte, welches die unterschiedlichen mathematischen Ansätze der Codierverfahren berücksichtigt. So eignen sich einige der Verfahren aufgrund ihrer algebraischen Struktur besser zur Korrektur von zufällig verteilten Fehlern, während andere Fehler, die in zeitlicher Nähe auftreten, effizienter korrigieren. Bei dem Vergleich und der Beurteilung von Codes sind diese beiden unterschiedlichen Klassen dementsprechend zu differenzieren. Dies ist mittels des in der Arbeit gewählten Ansatz ohne weiteres möglich, da die mathematischen Strukturen miteinbezogen werden. Entscheidendes Ergebnis bei einem Vergleich ist, dass der Einsatz eines Codierverfahrens zur Fehlerkorrektur, das sich besonders für die Korrektur von Übertragungsfehlern in zeitlicher Nähe eignet, zu bevorzugen ist, falls das Gedächtnis des verwendeten Übertragungskanals hinreichend stark ist. Darüber hinaus lassen sich abhängig von den Übertragungsbedigungen optimale Codierverfahren zur Fehlerkorrektur bestimmen. Dazu geben wir eine tabellarische Übersicht aller faktisch relevanten Parameterkonstellationen an.

Nun ist für eine Bewertung der Leistungsfähigkeit eines Codes eine exakte Auswertung der Restfehlerwahrscheinlichkeit nicht unbedingt notwendig. Daher leiten wir Näherungsformeln zur Berechnung der Restfehlerwahrscheinlichkeit verschiedener Codierverfahren her. Die erzielten Resultate zeigen in diesem Zusammenhang, dass die die erreichte Näherung für unsere Zwecke völlig ausreichend ist.

Darüber hinaus passen wir abschließend das vorgestellte Kanalmodell an einen kontinuierlichen Fehlerprozeß an und setzen somit das Modell auf der Signalebene ein. Diese wird in der Informationstechnik besonders häufig beobachtet und analysiert. Es zeigt sich schließlich, dass auch unter diesen Vorgaben das hergeleitete Schema zur Bewertung verschiedener Codierverfahren anwendbar ist und prinzipiell die erwarteten Ergebnisse zeigt. Auch hier bestätigt sich, dass solche Verfahren, die Fehler in zeitlicher Nähe korrigieren, effektiver unter der Bedingung arbeiten, dass die Korrelation des kontinuierlichen Fehlerprozesses eine gewisse Grenze übersteigt.

Die mögliche Anwendung des vorgestellten Kanalmodells geht weit über die hier skizzierte hinaus. Im Schlußteil dieser Arbeit werden hierzu weitere Einsatzmöglichkeiten aufgezeigt.

Wie bereits zu Beginn erwähnt, legen wir in dieser Arbeit auf eine homogene und in sich geschlossene Darstellung der Thematik viel Wert. Eine Schwierigkeit, die sich diesbezüglich bei der Bearbeitung dieser Themenstellung ergab, ist die bestehende Lücke zwischen der mathematischen Theorie und den Untersuchungen in der praktischen Anwendung im Bereich der Informationstechnik. So werden in der Informationstechnik gezielt konkrete mathematische Ergebnisse genutzt, um eine aus der praktischen Anwendung resultierende Problemstellung zu lösen, ohne dabei jedoch eine systematische Darstellung der kompletten

mathematischen Hintergründe zu präsentieren. Dadurch werden tiefer gehende mathematische Zusammenhänge oftmals nicht ausreichend erschlossen, so dass der Zugang zu den Konzepten der verschiedenen Codierverfahren erschwert wird. Hierbei ist allerdings zu beachten, dass der Schwerpunkt im Bereich der Informationstechnik schlichtweg ein anderer, ein eher anwendungsorientierter ist. Zur Erschließung neuer Konzepte und zur Bestimmung eines optimalen Codierverfahrens für gegebene Übertragungsbedingungen sind die mathematischen Hintergründe jedoch von elementarer Bedeutung und daher ausführlich darzulegen. Daher ist ein weiterer Schwerpunkt der Arbeit die mathematisch exakte Darstellung der Thematik sowie eine systematische Zusammenstellung der mathematischen Hintergründe und Grundlagen. In diesem Zusammenhang hat sich die Arbeit die Aufgabe gestellt neben einer guten Übersichtlichkeit bei der Herleitung und Darstellung der Ergebnisse und der mathematisch homogenen Präsentation der Zusammenhänge, die mathematische Korrektheit der neuen und äußerst effizienten Algorithmen nachzuweisen.

Insgesamt gliedert sich die Arbeit wie folgt. In Kapitel 2 werden zur Motivation der Arbeit ihr Kontext und Hintergrund beschrieben sowie die Aufgabenstellung kurz erläutert. Dazu werden die mit der Übertragung von digitalen Daten einhergehenden Problemstellungen erörtert und die Themenstellung der Arbeit in diesen Rahmen eingegliedert.

Im Anschluß werden in Kapitel 3 zunächst die mathematischen Grundlagen der Stochastik zusammengestellt, die zur Modellierung zufällig aufgetretener Störungen benötigt werden. Der Hauptteil von Kapitel 3 befaßt sich mit der Darstellung von Kanalmodellen, die das Gedächtnis eines Kanals abbilden und aus der Literatur bereits bekannt sind. In diesem Zuge werden insbesondere deren Vor- und Nachteile einander gegenüber gestellt. Abschließend präsentieren wir ein endliches Markov Modell als Störungsmodell zur Abbildung realer Übertragungskanäle mit Gedächtnis.

Im ersten Teil von Kapitel 4 werden die Grundlagen der Codierungstheorie sowie die algebraischen Hintergründe zusammengestellt, die für die Zielsetzung der Arbeit benötigt werden. Im zweiten Teil von Kapitel 4 wird die Restfehlerwahrscheinlichkeit als Maß für die Effizienz eingeführt. Ferner beinhaltet der zweite Teil von Kapitel 4 die Herleitung eines Verfahrens, mit dem die Restfehlerwahrscheinlichkeit verschiedener Codierverfahren berechnet werden kann. Dabei basiert dieses auf dem in Kapitel 3 entwickelten Kanalmodell. Zum Beweis der Korrektheit der hergeleiteten Rekursionsgleichungen greifen wir auf die zuvor dargestellten codierungstheoretischen Grundlagen zurück.

In Kapitel 5 entwickeln und verifizieren wir einen Algorithmus, mit dem wir die Existenz vergleichbarer Codierverfahren nachweisen können. Ferner gelingt es mit einem entsprechenden Algorithmus optimale Parameterkonstellationen für Codierverfahren in Abhängigkeit von den Übertragungsbedingungen, beispielsweise der Bitfehlerrate, zu bestimmen. Abschließend präsentieren wir in Kapitel 5 eine Beurteilung und einen Vergleich verschiedener Codierverfahren in graphischer Form, wobei diese auf der entsprechenden Auswertungen der in den beiden vorherigen Kapiteln entwickelten Methoden beruht.

In Kapitel 6 leiten wir aus den oben genannten Gründen Näherungsformeln zur Berechnung der Restfehlerwahrscheinlichkeit her, wobei auch hier die verschiedenen algebraischen Strukturen der Codierverfahren berücksichtigt werden.

Kapitel 7 befaßt sich zum Abschluß mit einer Erweiterung und Modifizierung des vorgestellten Kanalmodells. Inhalt ist eine Anpassung des Modells an einen kontinuierlichen

Fehlerprozeß, wie er in der Nachrichten- und Informationstechnik häufig Gegenstand der Untersuchungen ist. Eine Anwendung der in der Arbeit präsentierten Methoden zur Bewertung unterschiedlicher Codierverfahren bestätigt die in Kapitel 5 erzielten Resultate unter diesen Vorgaben.

Im Ausblick skizzieren wir weitere Anwendungen für die entwickelten Konzepte sowie zusätzliche Gebiete, in denen Fehlerkorrekturverfahren zur Absicherung von Information bzw. sensiblen Daten eingesetzt werden. Wir präsentieren kurz weitere Forschungsziele, die mit der Erschließung neuer Konzepte bzw. neuer und effizienter Codierverfahren verbunden sind.

An dieser Stelle möchte ich mich bei Herrn Prof. Dr. Wolfram Luther für sein großes Engagement sowie seine zahlreichen kritischen und hilfreichen Kommentare bedanken, die zum Gelingen der Arbeit entscheidend beigetragen haben. Mein besonderer Dank gilt zudem Herrn Priv.-Doz. Dr. Gerhard Haßlinger für die vielen anregenden Diskussionen und konstruktiven Ideen. Ferner möchte ich mich an dieser Stelle bei der Deutschen Telekom Systemlösungen GmbH in Person von Herrn Dr. Kurandt, Herrn Dr. Schröck und Herrn Dr. Klein für die gute Zusammenarbeit bedanken, die diese Arbeit im Rahmen eines Forschungsprojektes gefördert haben. Nicht zuletzt durch die Arbeit im Rahmen dieses Projektes und die Teilnahme an verschiedenen Konferenzen, die durch die Deutsche Telekom Systemlösungen GmbH unterstützt worden ist, ist mein Blick auch für nicht-mathematische Probleme geschärft worden.

Ferner möchte ich mich an dieser Stelle ganz herzlich bei meiner Familie bedanken, die mir dieses Studium erst ermöglicht und mich dabei stets unterstützt hat. Ein besonderer Gruß gilt dabei meiner Schwester Kerstin.
Ferner gilt mein Dank meinen Kollegen und Freunden für ihre moralische Unterstützung.
Zum Schluß danke ich meinem Freund Dr. Oliver van Laak für sein Verständnis und seine Geduld sowie die vielen hilfreichen Korrekturvorschläge.

# Kapitel 2

# Grundzüge der digitalen Datenübertragung

In der modernen Informationsgesellschaft gewinnen der wechselseitige Austausch von Information sowie deren Verarbeitung und Speicherung zunehmend an Bedeutung. Hierbei bezieht sich der Transfer von Information ganz allgemein auf den Austausch von Sprache, Klang, Bildern oder Text. Aufgrund des gestiegenen Stellenwertes erfordern der Austausch von Information sowie deren Verarbeitung und Speicherung ständig effizientere und zuverlässigere digitale Datenübertragungssysteme. Eine problematische und äußerst kritische Entwicklung, die sich in diesem Zusammenhang zur Zeit abzeichnet, stellen die rapide steigenden Übertragungsraten dar sowie die qualitativ sehr unterschiedlichen Anforderungen verschiedener Anwendungen der Praxis. So ist in immer kürzerer Zeit immer mehr Information zu übertragen, zu erfassen, bereitzustellen, zu verarbeiten und gegebenenfalls zu speichern. Dies macht gleichzeitig effizientere Instrumente zur Gewährleistung der Zuverlässigkeit und Qualität der Übertragung sowie neue Übertragungsprinzipien notwendig.

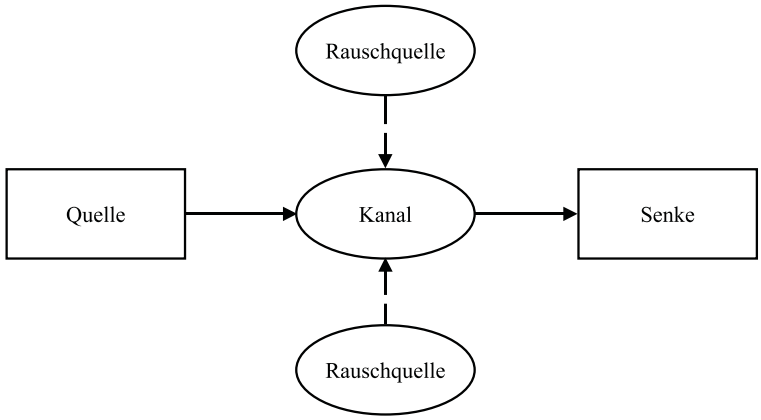


Abbildung 2.1: Konzept der Datenübertragung

Im allgemeinen beruht die Übertragung und damit der Austausch von Information auf dem nachfolgenden Konzept, welches in Abbildung 2.1 anschaulich dargestellt ist. Ausgehend von einer Quelle, die die zu übermittelnde Information generiert, wird diese mittels eines

Übertragungskanals zur Senke übermittelt. Dabei ist unter einem Übertragungskanal ein physikalisches Medium zu verstehen, welches einem gewissen Störeinfluß unterliegt. Dieser bedingt, dass die zu übermittelnde Information unter Umständen während der Übertragung verfälscht wird.

Da die rechnergestützte Datenübertragung in den letzten Jahrzehnten stetig an Bedeutung gewonnen hat, ist von der *International Standards Organization (ISO)* ein Standard erarbeitet worden, der alle für die Kommunikation, also für den Austausch von Information, zwischen Rechnern erforderlichen Abläufe im sogenannten *Open System Interconnection-Referenzmodell (OSI-Modell)* zusammenstellt, diese in 7 Schichten aufteilt und organisiert. Abbildung 2.2 zeigt eine anschauliche Darstellung des OSI-Modells mit seinen 7 Schichten. Hierbei spiegeln insbesondere die beiden letzten Spalten einerseits die Beziehungen zum Rechnernetz und andererseits zum Kommunikationspartner wieder sowie die Abhängigkeiten der Funktionen der einzelnen Schichten. Logisch betrachtet sind die einzelnen Schichten derart aufgebaut, dass Voraussetzung für die Funktionen einer oberen Schicht die darunter liegenden Schichten sind.

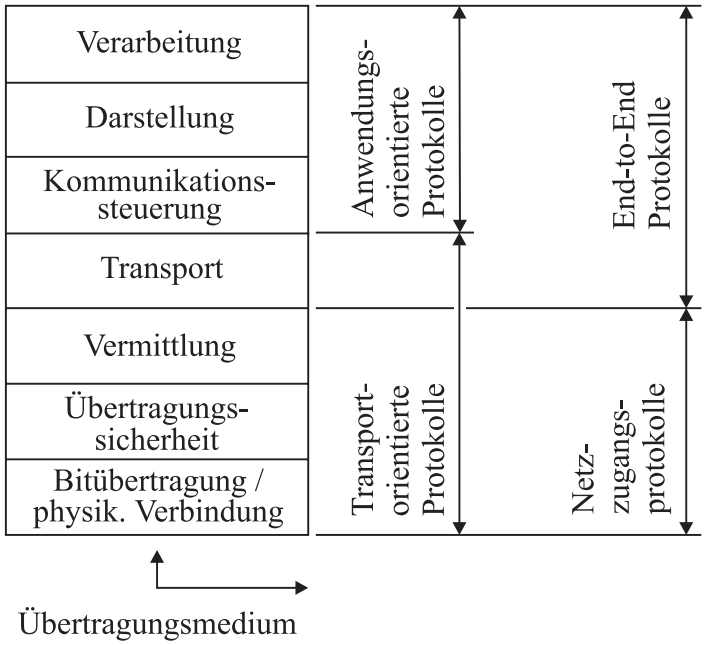


Abbildung 2.2: Das OSI-Referenzmodell

Gemäß dem ausgearbeiteten Standard erfüllen die einzelnen Schichten des OSI-Modells die folgenden Funktionen: Die Bitübertragungsschicht betrifft die Übertragung eines harten Bitstroms über das physikalische Übertragungsmedium, das unterhalb dieser Bitübertragungsschicht liegt. Das Ergebnis der Sicherungsschicht ist die Strukturierung und Blockeinteilung des Bitstroms sowie die Flußsteuerung zum Zweck der Fehlererkennung und -korrektur. Die Vermittlungsschicht dient zum Aufbau der Verbindung über das Netz. Die Schichten 4 bis 7 sind zuständig für die Kommunikation zwischen den Endteilnehmern und betreffen demzufolge also die Ende-zu-Ende-Verbindungen. Daher ist das Rechnernetz auf diesen Ebenen nicht mehr an der Kommunikation beteiligt. Dabei wiederholt sich in den Schichten 4 bis 6 ein Teil der Funktionen, die in den Schichten 1 bis 3 realisiert wer-

den, wie beispielsweise die Fehlererkennung und -korrektur, die Blockeinteilung sowie die Flußsteuerung. Detailliertere Ausführungen zu den Funktionen und Aufgaben der einzelnen Schichten des OSI-Modells finden sich unter anderem in den Arbeiten von [21, 29, 45, 49].

Ein weiteres Modell für die Kommunikation, also den Informationsaustausch, zwischen Rechnern, das wir gegenüber dem OSI-Modell kurz skizzieren, ist das *TCP/IP-Referenzmodell (Transmission Control Protocol/Internet Protocol)*. Dieses kommt im Gegensatz zum OSI-Modell mit nur 4 Schichten aus, ist aber logisch genauso aufgebaut wie das OSI-Modell; d.h. zur Bereitstellung von Diensten und Funktionen auf höheren Schichten sind zunächst die darunter liegenden Schichten abzuarbeiten, wie dies in Abbildung 2.3 illustrativ dargestellt ist.

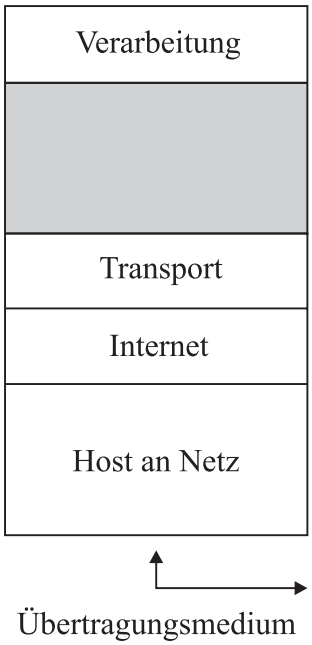


Abbildung 2.3: Das TCP/IP-Referenzmodell

Im TCP/IP-Modell bildet die Internet-Schicht den Kern der gesamten Netzarchitektur. In dieser Schicht wird ein offizielles Paketformat sowie Protokoll definiert, das sogenannte *Internet Protocol (IP)*. Die wesentliche Aufgabe der Internet-Schicht ist das Paket-Routing, d.h. die sogenannten *IP*-Pakete korrekt zuzustellen. Die darüber liegende Transportschicht ermöglicht ähnlich wie im OSI-Modell die Kommunikation zwischen den Endteilnehmern. Ferner gehört die Flußkontrolle, d.h. die Vermeidung von Überlastungen, zu den Funktionen dieser Schicht. An dieser Stelle verweisen wir für eine ausführlichere Darstellung des TCP/IP-Modells auf die Arbeit von [49].

Derzeit existieren für den konventionellen Telefondienst, für das Kabelfernsehen sowie für weitere Datendienste mehrere spezialisierte Netze nebeneinander. Ziel ist es, sämtliche dieser spezialisierten Netze für die verschiedenen Arten des Informationstransfers in einem einzigen Netz zusammenzufassen. Dieses zeichnet sich dann durch eine immense Übertragungs- bzw. Datenrate aus. Beispiele für Anwendungen bzw. Dienste, die über dieses Netz angeboten werden sollen, sind Datenübertragungen mit hohen Datenraten aus den Bereichen der

Industrie und Wissenschaft, interaktives Fernsehen, Musik in CD-Qualität, Videokonferenzen sowie Video-on-Demand. Die sich daraus ergebenden stark veränderten Anforderungen an das Übertragungsverfahren, wie z.B. die hohen Übertragungsraten, machen ein neues Übertragungsprinzip erforderlich, den sogenannten *Asynchronous Transfer Mode (ATM-Technologie)*. Das grundlegende Konzept von ATM besteht darin, alle Arten von Information in kleinen Paketen von 53 byte Länge, den sogenannten *ATM-Zellen*, zu übertragen. Dabei setzt sich eine ATM-Zelle zusammen aus 5 byte, die den sogenannten *Header* angeben, und 48 byte zur Darstellung der Nutzdaten. Die ATM-Technologie führt zu einem Referenzmodell, welches sich in einigen Punkten von den beiden zuvor beschriebenen Modellen unterscheidet, vgl. dazu Abbildung 2.4.

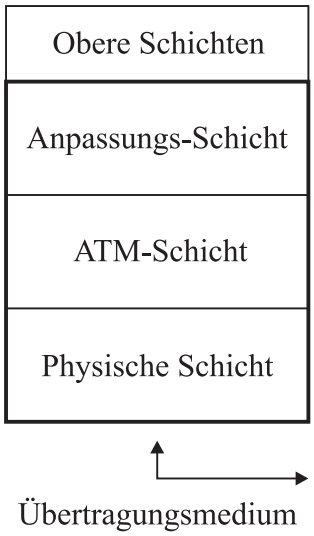


Abbildung 2.4: Das ATM-Referenzmodell

Das ATM-Modell ist durch drei Schichten gekennzeichnet. Die physikalische Schicht bezeichnet ähnlich wie zuvor das darunter liegende physikalische Übertragungsmedium. Die ATM-Schicht definiert den 5 byte langen Header sowie die Bedeutung der übrigen Felder, organisiert den Transport und übernimmt auch die Kontrolle der Überlastung. Einen groben Einblick in das Thema ATM-Netze und die Funktionen der einzelnen Schichten liefern beispielsweise die Arbeiten von [29, 49]. Eine äußerst umfassende und sehr ausgereifte Darstellung der Thematik wird in **??** präsentiert. Ferner sei für detailliertere Fragestellungen auch auf das ATM-Forum und auf die International Telecommunication Union (ITU-T) verwiesen, beides Organisationen, die die zukünftige Richtung der ATM-Technologie vorgeben und an deren Standardisierung arbeiten.

Den vorgestellten Referenzmodellen gemeinsam ist, dass die zu übermittelnde Information über ein physikalisches Übertragungsmedium, d.h. einen materiellen physikalischen Träger, übermittelt wird. Dieser liegt in jedem der Modell unterhalb der untersten Schicht. Als materielle Träger der Information werden in der Regel elektrische oder optische Signale oder auch elektromagnetische Wellen verwendet. Das bedeutet der in Abbildung 2.1 dargestellte Übertragungskanal bzw. das in den Abbildungen 2.2, 2.3 und 2.4 angedeutete physikalische Übertragungsmedium kann in technischer Hinsicht auf ganz unterschiedliche Weise

realisiert werden. So werden beispielsweise Kupfer- oder Koaxialkabel zur Übertragung verwendet, aber auch Lichtwellenleiter, d.h. Glasfaserkabel, oder Funk bzw. Richtfunk werden zur Übermittlung von Information eingesetzt. Eine Gemeinsamkeit der hier aufgelisteten physikalischen Träger ist, dass die Trägersignale während der Übertragung bestimmten Störeinflüssen unterliegen. Im allgemeinen wird hierbei zwischen *Dämpfungen* und *Verzerrungen* sowie dem sogenannten *Rauschen* differenziert. Unter dem Begriff *Dämpfung* wird in diesem Kontext der Energieverlust verstanden, der durch die Verbreitung der Signale entsteht. *Verzerrungen* werden durch unterschiedliche Geschwindigkeiten bei der Ausbreitung der Signale verursacht. Dies führt schließlich zur teilweisen Überlagerung der Signale. Beide Störeinflüsse, Dämpfungen und Verzerrungen, sind abhängig vom gesendeten Signal; d.h. sie treten nur dann auf, wenn auch das Signal auftritt, und können durch den Einsatz entsprechender Verstärker und Entzerrer beinahe vollständig beseitigt werden.

Eine weitere Beeinträchtigung der Übertragung wird durch sogenanntes *Rauschen* verursacht. Dies sind Fremdspannungen, die das gesendete Signal überlagern und im Gegensatz zu Dämpfungen und Verzerrungen unabhängig vom Signal auftreten. Ursachen für Rauschen während der Übertragung über Kupfer- oder Koaxialkabel sind beispielsweise thermisches Rauschen in Widerständen und Transistoren, Nebensprechen sowie Impulsgeräusche, die z.B. durch Spannungsspitzen in der Stromleitung verursacht werden. Bei der Datenübertragung über eine Mobilfunk- oder eine Satellitenübertragungsstrecke bilden atmosphärische Störungen häufig eine Ursache für Rauschen. Diese unvorhersehbaren Störungen in Form von Rauschen können zu einer Verfälschung der zu übermittelnden Information führen. Aufgrund der verschiedenen Ursachen für Rauschen variieren Art und Dauer sowie das Ausmaß der Störungen, je nachdem welches Übertragungsmedium zur Übertragung verwendet wird. So verursachen diese unvorhersehbaren Störungen bei einer Übertragung über Glasfaserkabel im Bitstrom in der Regel äußerst wenige, einzelne, zufällig verteilte Fehler. Bei einer Übertragung über eine Mobilfunk- oder Satellitenübertragungstrecke dagegen verursacht Rauschen, dass Fehler in dem zu übertragenden Bitstrom in zeitlicher Nähe *gebündelt* auftreten. Für eine zuverlässige Kommunikation, d.h. einen verlässlichen und gesicherten Informationsaustausch ist die Kontrolle derartiger unvorhersehbarer Störungen unerlässlich und daher mittlerweile ein fester Bestandteil des Designs und der Gestaltung digitaler Datenübertragungssysteme.

Das in Abbildung 2.1 skizzierte Konzept eines digitalen Übertragungssystems ist daher um die in Abbildung 2.5 dargestellten Komponenten zu erweitern. Generell gehen wir davon aus, dass die von der Quelle generierte Information eine Folge von diskreten Symbolen darstellt, die zur Senke übertragen werden soll. Aufgabe des Quellencodierers ist es, die von der Quelle ausgegebene Information in eine Folge von Bits, d.h. eine binäre *Informationsfolge*, zu transformieren. Dabei ist der Quellencodierer idealerweise derart gestaltet, dass erstens die Anzahl der Bits pro Zeiteinheit, die zur Darstellung der von der Quelle generierten Information erforderlich sind, minimiert wird und zweitens die von der Quelle generierte Information aus der binären Informationsfolge ohne Mehrdeutigkeit rekonstruiert werden kann. Von daher erfolgt im Quellencodierer lediglich eine Komprimierung der erzeugten Information. Der Kanalcodierer versieht die erhaltene binäre Informationsfolge gezielt mit zusätzlicher Redundanz, um auf diese Weise das während der Übertragung auftretende Rauschen möglichst effizient kontrollieren und aufgetretene Fehler korrigieren zu können. Auf diese Weise wird eine zuverlässige und sichere Übertragung gewährleistet. Jedes vom Kanalcodierer ausgegebene Bit wird vom Modulator in ein entsprechendes Signal umgesetzt, welches dann über den phy-

sikalischen Träger des Übertragungkanals übermittelt wird. Dabei wird das Trägersignal, wie erwähnt, durch verschieden starke, vom Übertragungsmedium abhängige, Störungen verfälscht. Es ist die Aufgabe des Demodulators, aus dem empfangenen Trägersignal den Wert des gesendeten Bits zu ermitteln und an den nachfolgenden Kanaldecodierer weiterzugeben. Der Kanaldecodierer speichert die vom Demodulator übergebenen Bits und versucht, trotz der von der Rauschquelle verursachten Störungen die ursprüngliche, vom Kanalcodierer erzeugte, binäre Informationsfolge zu rekonstruieren. Falls der Kanaldecodierer zusätzliche Information verarbeiten kann, ist es prinzipiell von Vorteil, wenn der Demodulator neben dem Wert des empfangenen Bits auch die Wahrscheinlichkeit, mit der das empfangene Bit diesen Wert annimmt, an den Kanaldecodierer weiterleitet, sozusagen als Sicherheit dafür, dass das Bit tatsächlich den ermittelten Wert annimmt. Abschließend transformiert der Quellendecodierer die vom Kanaldecodierer übermittelten und eventuell korrigierten Bits in solche diskreten Symbole zurück, wie sie von der Quelle ursprünglich erzeugt worden sind, und leitet die so erhaltene Information an den Empfänger weiter.

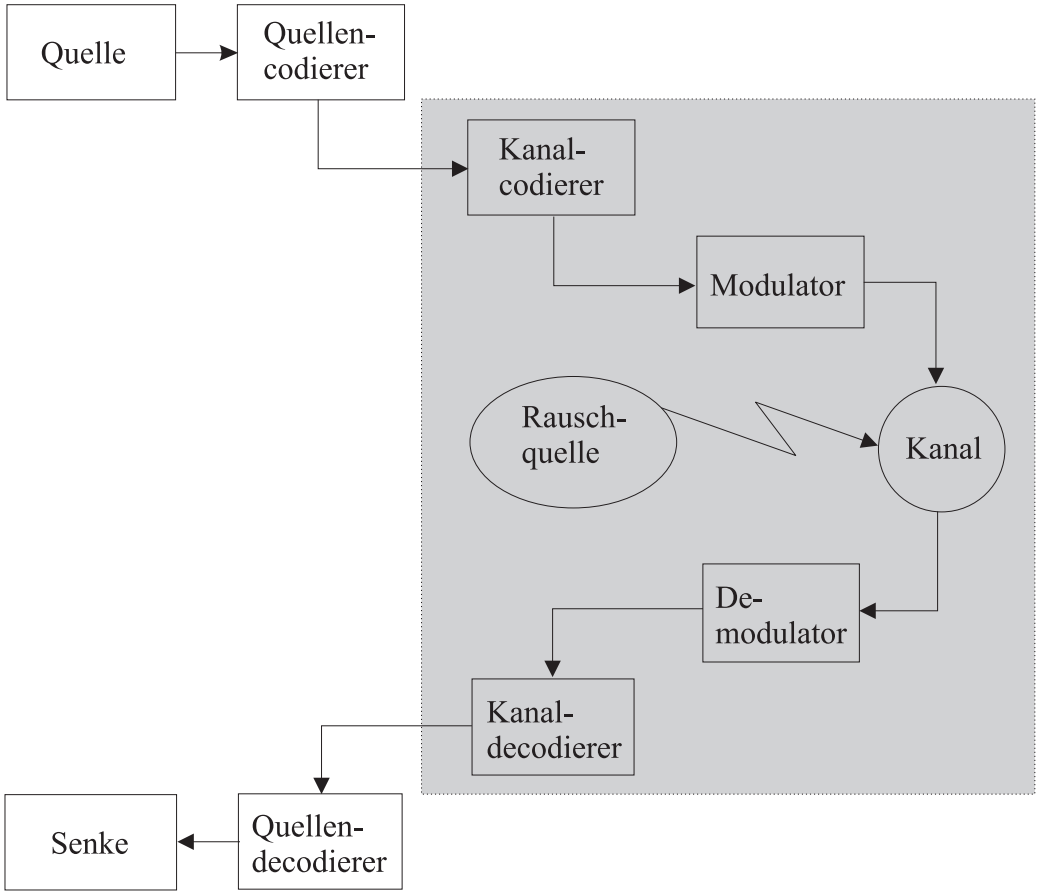


Abbildung 2.5: Digitales Datenübertragungssystem

Da der Austausch und die Verarbeitung von Information in den letzten Jahren immer effizientere und zuverlässigere digitale Datenübertragungssysteme erforderlich gemacht haben, besteht nun eine Hauptaufgabe im Bezug auf die Entwicklung digitaler Datenübertragungssysteme in der Gestaltung einer effizienten Fehlerkontrolle, so dass eine zuverlässige Reproduktion der zu übermittelnden Information möglich wird. Der Teil eines Kommunikations- bzw. Datenübertragungssystems, der unvorhersehbaren Störungen

entgegenwirkt, ist nach den vorangehenden Ausführungen der Teil bestehend aus dem Kanalcodierer und dem Kanaldecodierer. Im Jahre 1948 hat Shannon in seiner Arbeit [48] gezeigt, dass durch eine geeignete *Codierung*, also Absicherung der Information, die Anzahl der Fehler, die durch die Rauschquelle des physikalischen Übertragungskanals verursacht werden, bis auf jeden gewünschten Level reduziert werden kann, ohne dabei die Rate der Informationsübertragung zu beeinträchtigen. Der Einsatz entsprechender Verfahren zur Absicherung der Information gegenüber zufälligen Störungen ist damit zu einem integralen Bestandteil des Designs moderner digitaler Datenübertragungssysteme geworden.

Generell wird zwischen zwei grundlegend verschiedenen Methoden zur Fehlerkontrolle differenziert, der *Forward Error Correction (FEC)* auf der einen Seite und Systemen mit *Automatic Repeat Request (ARQ)* auf der anderen Seite. Für ein ARQ-System ist es notwendig, zwei Kanäle zur Verfügung zu haben, also Information in beiden Richtungen sowohl von der Quelle zur Senke als auch umgekehrt senden zu können. Die Absicherung der Information, d.h. die Fehlerkontrolle, in einem derartigen System wird dann folgendermaßen realisiert. Es wird ein Verfahren zur Fehlererkennung eingesetzt. Bei der Erkennung eines Fehlers wird vom Empfänger beim Sender eine wiederholte Übertragung der verfälschten Information nachgefragt. Ein FEC-System dagegen benötigt nur einen Kanal vom Sender zum Empfänger und verwendet zur Fehlerkontrolle Verfahren, die erkannte Fehler automatisch korrigieren können. Der Hauptvorteil von ARQ gegenüber FEC besteht darin , dass die bloße Fehlererkennung mit einem wesentlich geringeren technischen Aufwand implementiert werden kann. Demgegenüber stehen allerdings entscheidende Nachteile. So existieren viele Systeme, die nur über einen Kanal vom Sender zum Empfänger verfügen. Dies ist beispielsweise der Fall bei Messungen von physikalischen oder chemischen Experimenten. Ein weiterer Nachteil besteht darin, dass eine wiederholte Übertragung der verfälschten Information zu oft nachgefragt werden muß, falls die Störungen eine sehr hohe Fehlerrate auf dem betrachteten Übertragungskanals verursachen. Ferner ist die durch die wiederholte Übertragung der Information verursachte Verzögerung für manche Anwendungen in der Praxis inakzeptabel, so beispielsweise bei Echtzeit-Anwendungen, wie Videokonferenzen und Video-on-Demand, um nur einige Beispiele zu nennen. Aus diesem Grund befassen wir uns im Rahmen dieser Arbeit ausschließlich mit FEC-Systemen. Dabei existieren viele sehr unterschiedliche Verfahren, die eine FEC durchführen; d.h. auf verschiedene Art und Weise wird die zu übermittelnde Information mit zusätzlicher Redundanz versehen und geschützt. Daraus resultiert, dass die verschiedenen Verfahren von ganz unterschiedlicher Qualität und Effizienz im Hinblick auf Störungen unterschiedlicher Art und verschiedenen Ausmaßes sind.

Unser Ziel ist es, im Rahmen dieser Arbeit Störungen unterschiedlicher Art und verschiedenen Ausmaßes in einem mathematischen Modell abzubilden. Dazu interpretieren wir die auf den verschiedenen physikalischen Übertragungsmedien auftretenden Störungen als einen stochastischen Prozess. Wir entwickeln auf der Basis dieser Idee ein mathematisches Modell, welches die unterschiedlichen Eigenschaften der verschiedenen Übertragungsmedien und damit die Übertragungseigenschaften eines Kanals abbildet, vgl. dazu Kapitel 3. Dieses mathematische Modell ermöglicht es uns, die Art und Dauer sowie das Ausmaß der Störungen abzubilden, um dann in Abhängigkeit von diesen die verschiedenen Verfahren zur Fehlerkontrolle einander gegenüber zu stellen und zu bewerten. Dabei beachten wir insbesondere, dass die verschiedenen Verfahren zur Fehlerkorrektur mit unterschiedlichen mathematischen Ansätzen arbeiten. Sie beruhen auf verschiedenen algebraischen Struk-

turen und besitzen daher bezüglich Störungen unterschiedlicher Art und verschiedenen Ausmaßes auch verschiedene Qualitäten. Demzufolge unterscheiden sie sich im Bezug auf ihre Effizienz und Leistungsfähigkeit. Mittels einer derartigen Bewertung verschiedener Fehlerkorrekturverfahren kann schließlich im voraus ein optimales Verfahren zur Fehlerkontrolle bestimmt werden.

Abschließend sei noch bemerkt, dass der Einsatz dieser Verfahren zur FEC nicht nur bei der Übertragung des Bitstroms vorgesehen ist. Ähnliche Verfahren zur Fehlerkontrolle sind im Standard des OSI-Modells, aber auch im TCP/IP-Modell sowie beim Einsatz der ATM-Technologie an weiteren Stellen, d.h. auch auf höheren Ebenen, ebenso vorgesehen. So werden im OSI-Modell auch in den Schichten 2, 3 und 4 Verfahren zur Fehlerkontrolle eingesetzt, vgl. dazu beispielsweise die detaillierteren Ausführungen in [49]. Ähnliches gilt auch für ATM-Netze. Hier ist vorgesehen, den Header einer ATM-Zelle mit einem derartigen Verfahren zur Fehlerkontrolle zu sichern. Es erscheint hier durchaus möglich, das mathematische Modell zur Abbildung zufälliger Störungen auch auf diesen höheren Ebene einzusetzen, um so die dort eingesetzten Verfahren zur Fehlerkorrektur ebenfalls anhand einer Beurteilung ihrer Effizienz miteinander zu vergleichen, d.h. das in Kapitel 3 entwickelte mathematische Modell ist demnächst dahingehend zu erweitern, dass es nicht ausschließlich auf der Bitebene sondern auch auf der Paketebene angewendet werden kann. Im Ausblick der Arbeit finden sich zu diesen zusätzlich denkbaren Anwendungen des entwickelten Modells ausführlichere Überlegungen.

# Kapitel 3

# Modellierung zufälliger Störungen

Der Schwerpunkt dieses Kapitels ist nun die Modellierung zufälliger Störungen, die während der Übertragung digitaler Daten über reale Kanäle, also auf den verschiedenen physikalischen Übertragungsmedien, auftreten und zu einer Verfälschung der zu übermittelnden Information führen können. In Kapitel 2 haben wir ferner darauf hingewiesen, dass zur Korrektur aufgetretener Übertragungsfehler verschiedene Verfahren existieren, die dabei allerdings unterschiedliche mathematische Ansätze verfolgen und demzufolge bei Störungen unterschiedlicher Art und verschiedenen Ausmaßes von unterschiedlicher Qualität sein werden. Bereits in der Einleitung haben wir mit Blick auf die Zielsetzung der Arbeit, nämlich anhand der spezifischen Übertragungseigenschaften des verwendeten Kanals ein optimales Codierverfahren auszuwählen, die Entwicklung eines geeigneten Konzeptes zur Beurteilung verschiedener Codierverfahren motiviert. Eine derartige Beurteilung verschiedener Codierverfahren in Abhängigkeit von den spezifischen Eigenschaften des betrachteten Übertragungskanals setzt allerdings eine entsprechende Beschreibung und Modellierung des digitalen Fehlerprozesses voraus, der durch die auf dem physikalischen Übertragungsmedium auftretenden, zufälligen Störungen verursacht wird. In diesem Kapitel leiten wir hierzu ein geeignetes Störungsmodell her, das reale Kanäle mit Gedächtnis abbildet. Dazu stellen wir im ersten Abschnitt 3.1 die mathematischen Grundlagen und Methoden der Stochastik zusammen, die wir zur Modellierung zufälliger Störungen benötigen. Der zweite Abschnitt 3.2 befasst sich mit der Darstellung des durch zufällige Störungen verursachten digitalen Fehlerprozesses und nutzt dazu die in Abschnitt 3.1 zusammengestellten, mathematischen Grundlagen. In Abschnitt 3.3 präsentieren wir einige bereits bekannte Kanalmodelle ohne bzw. mit Gedächtnis und stellen deren Vor- und Nachteile einander gegenüber. Zum Abschluss entwickeln wir in Abschnitt 3.4 ein weiteres Störungsmodell, das ebenfalls reale Kanäle mit Gedächtnis abbildet, in dem aber die Vorteile der zuvor in Abschnitt 3.3 vorgestellten Modelle miteinander verschmolzen und deren Schwachstellen weitestgehend aufgehoben werden. Neu ist hierbei die in dieser Arbeit präsentierte Auswertung des Modells in diesem Zusammenhang.

## 3.1 Mathematische Grundlagen

Zu Beginn des Kapitels stellen wir in diesem Abschnitt die mathematischen Grundlagen und Werkzeuge zur Verfügung, die wir zur Modellierung eines digitalen Übertragungskanals,

dem dort aufgetretenen, digitalen Fehlerprozeß und damit zur Entwicklung eines geeigneten Kanalmodells benötigen. Zur Modellierung des während der Übertragung aufgetretenen Fehlerprozesses setzen wir grundlegende Konzepte aus der Stochastik ein. Bevor wir die in diesem Zusammenhang verwendeten, stochastischen Methoden näher erläutern, führen wir einige Bezeichnungen ein und orientieren uns bei der Notation in erster Linie an [1, 2, 11, 38] und [20, 44].

Im Rahmen dieser Arbeit bezeichne $\Omega$ im allgemeinen die Menge der Elementarereignisse und $\mathcal{A}$ eine $\sigma$-Algebra über $\Omega$. Ist ein Wahrscheinlichkeitsmaß $p$ auf der $\sigma$-Algebra $\mathcal{A}$ gegeben, so heißt das Tripel $(\Omega, \mathcal{A}, p)$ ein Wahrscheinlichkeitsraum. Ferner bezeichnet $X$ in der Regel eine Zufallsvariable, die auf einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, p)$ definiert ist. Betrachten wir eine Menge von Zufallsvariablen $(X_t)_{t\in\mathcal{T}}$, so interpretieren wir typischerweise den Index $t$ als Zeitparameter und damit die Indexmenge $\mathcal{T}$ als eine Menge von Zeitpunkten, was zur folgenden Definition führt.

**3.1.1 Definition**

*(i) Ein stochastischer Prozeß ist eine Menge von Zufallsvariablen $(X_t)_{t\in\mathcal{T}}$, die auf einem gemeinsamen Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, p)$ definiert sind.*

*(ii) Hinsichtlich des Zeitparameters $t$ klassifizieren wir stochastische Prozesse in*

- *(zeit-)diskrete stochastische Prozesse, falls die Indexmenge $\mathcal{T}$ diskret, also endlich oder abzählbar unendlich ist, und schreiben $(X_t)_{t\in N}$ oder $(X_1, X_2, \ldots)$*
- *(zeit-)kontinuierliche stochastische Prozesse, wenn die Indexmenge $\mathcal{T}$ reell ist, und schreiben dann $(X_t)_{t\geq 0}$.*

*(iii) Die Werte, die die Zufallsvariablen $X_t$, $t \in \mathcal{T}$, annehmen können, heißen die Zustände des stochastischen Prozesses zum Zeitpunkt $t$.*

*(iv) Die Menge aller Zustände bezeichnen wir als Zustandsraum $\mathcal{S}$.*

Nun beschränken wir uns im Rahmen dieser Arbeit ausschließlich auf die Betrachtung (zeit-)diskreter stochastischer Prozesse. Um in Abschnitt 3.3 die Nachteile der bis dahin vorgestellten Kanalmodelle charakterisieren zu können, benötigen wir eine spezielle Klasse von stochastischen Prozessen, die wir in der folgenden Definition typisieren.

**3.1.2 Definition**

*Es sei eine Folge unabhängiger und identisch verteilter Zufallsvariablen $X_i$, mit $i \geq 1$, gegeben. Dann liefert die Bildung von Teilsummen der Form $X_1+X_2+\cdots$ einen sogenannten Erneuerungsprozeß $(Y(t))_{t\in N}$. Jede Zufallsvariable $X_i$ entspricht einem Ereignis und $Y(t)$ ist dann eine Zufallsvariable, die die Anzahl der in einem Zeitintervall $[0, t]$ vervollständigten Ereignisse zählt.*

Zur Modellierung des während der Übertragung aufgetretenen, digitalen Fehlerprozesses bedienen wir uns einer wichtigen und charakteristischen Klasse von diskreten stochastischen

Prozessen, den sogenannten *Markov Ketten*. Jeder diskrete stochastische Prozeß dieser Klasse besitzt die Eigenschaft, dass der Verlauf in der Vergangenheit in die zukünftige Entwicklung miteinbezogen wird; d.h. Markov Ketten besitzen eine einfache Form von Gedächtnis. Ferner besitzen Markov Ketten in der Regel einen diskreten, also endlichen oder abzählbar unendlichen, Zustandsraum $\mathcal{S}$. Ohne Einschränkungen nehmen wir daher nun der Einfachheit halber an, dass der Zustandsraum $\mathcal{S}$ einer Markov Kette $(X_t)_{t\in\mathbb{N}}$ eine Teilmenge von $\mathbb{N}$ ist, also $\mathcal{S} \subset \mathbb{N}$ gilt. Bei der Notation und Bezeichnungsweise orientieren wir uns in der folgenden Definition beispielsweise an [2, 11] und [38].

**3.1.3 Definition**
*Ein diskreter stochastischer Prozeß $(X_t)_{t\in\mathbb{N}}$ heißt eine Markov Kette erster Ordnung mit diskretem Zustandsraum $\mathcal{S} \subset \mathbb{N}$, wenn die Markov Eigenschaft erfüllt ist:*

$$P(X_{t+1} = j \mid X_t = i, X_{t-1} = i_{t-1}, \ldots, X_0 = i_0) = P(X_{t+1} = j \mid X_t = i)$$

*für alle $j, i, i_{t-1}, \ldots, i_0 \in \mathcal{S}$.*

**3.1.4 Bemerkung**
Die Markov Eigenschaft aus Definition 3.1.3 läßt sich folgendermaßen interpretieren: Wenn $t+1$ als ein Zeitpunkt in der Zukunft, $t$ als die Gegenwart und dementsprechend $t-1, t-2, \ldots, 1, 0$ als Zeitpunkte in der Vergangenheit einer Markov Kette interpretiert werden, so hängt die zukünftige Entwicklung einer Markov Kette nur vom gegenwärtigen Zustand der Markov Kette ab, aber nicht davon, wie sich die Markov Kette in der Vergangenheit entwickelt hat. Folglich beinhaltet der Zustand der Markov Kette zum gegenwärtigen Zeitpunkt $t$ sämtliche Information über das Verhalten und den Verlauf der Kette in der Vergangenheit, die benötigt wird, um die Entwicklung der Markov Kette in der Zukunft zu bestimmen; d.h. Markov Ketten besitzen eine einfache Form von Gedächtnis.

Nach Definition 3.1.3 und deren Interpretation wird eine Markov Kette durch ihren Verlauf, also die Übergänge zwischen den Zuständen des Zustandsraumes $\mathcal{S}$ zu bestimmten Zeitpunkten, charakterisiert. Dies führt zur Definition der bedingten Wahrscheinlichkeit, die den Übergang von einem Zustand in einen anderen zu einem gewissen Zeitpunkt $t$ beschreibt.

**3.1.5 Definition**

(i) *Für eine Markov-Kette $(X_t)_{t\in\mathbb{N}}$ definieren wir als Übergangswahrscheinlichkeit vom Zustand $i$ in den Zustand $j$ zum Zeitpunkt $t$ die bedingte Wahrscheinlichkeit*

$$q_{ij}(t) := P(X_{t+1} = j \mid X_t = i) \qquad \text{für alle } i, j \in \mathcal{S}.$$

(ii) *Eine Markov-Kette $(X_t)_{t\in\mathbb{N}}$ heißt homogen, falls für alle $m \geq 0$ und alle $i, j \in \mathcal{S}$ gilt*

$$P(X_{t+1} = j \mid X_t = i) \quad = \quad P(X_{t+1+m} = j \mid X_{t+m} = i)$$

**3.1.6 Definition**
*Da für eine homogene Markov Kette $(X_t)_{t\in\boldsymbol{N}}$ die Übergangswahrscheinlichkeiten von einem Zustand in einen anderen nicht vom Zeitpunkt $t$ abhängig sondern zeitlich unabhängig sind, schreiben wir für die Übergangswahrscheinlichkeit vom Zustand $i$ in den Zustand $j$*

$$q_{ij} := P(X_t = j \,|\, X_{t-1} = i) \qquad \text{für alle } i, j \in \mathcal{S}$$

*und definieren wir die sogenannte Übergangsmatrix*

$$\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}\,.$$

**3.1.7 Bemerkung**
Eine homogene Markov-Kette $(X_t)_{t\in\boldsymbol{N}}$ mit endlichem Zustandsraum $\mathcal{S} = \{1, \ldots, N\}$ heißt eine *endliche Markov-Kette*, folglich besitzt die Übergangsmatrix $\mathbf{Q}$ endliche Dimension

$$\mathbf{Q} = \begin{pmatrix} q_{11} & q_{12} & \cdots & q_{1N} \\ q_{21} & q_{22} & \cdots & q_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N1} & q_{N2} & \cdots & q_{NN} \end{pmatrix}.$$

**3.1.8 Definition**

*(i) Für eine homogene Markov-Kette $(X_t)_{t\in\boldsymbol{N}}$ heißt der Vektor $\boldsymbol{\pi^{(0)}} = (\pi_1^{(0)}, \pi_2^{(0)}, \ldots)$ die Anfangsverteilung der Markov Kette. Dabei entspricht $\pi_i^{(0)}$ gerade der Wahrscheinlichkeitsverteilung*

$$\pi_i^{(0)} = P(X_0 = i) \qquad \text{für alle } i \in \mathcal{S}$$

*und gibt somit die Wahrscheinlichkeit an, dass der Anfangszustand der Markov Kette zum Zeitpunkt 0 gerade der Zustand $i \in \mathcal{S}$ ist.*

*(ii) Für eine homogene Markov-Kette $(X_t)_{t\in\boldsymbol{N}}$ heißen die Komponenten des Vektors $\boldsymbol{\pi^{(m)}} = (\pi_1^{(m)}, \pi_2^{(m)}, \ldots)$ die absoluten Zustandswahrscheinlichkeiten der Markov Kette. Dabei sind die absoluten Zustandswahrscheinlichkeiten $\pi_i^{(m)}$ festgelegt durch die Wahrscheinlichkeitsverteilung*

$$\pi_i^{(m)} = P(X_m = i) \qquad \text{für alle } i \in \mathcal{S}$$

*und geben somit die Wahrscheinlichkeit an, dass sich die Markov Kette nach $m$ Schritten im Zustand $i \in \mathcal{S}$ befindet.*

**3.1.9 Definition**
*Für eine homogene Markov Kette $(X_t)_{t\in\boldsymbol{N}}$ heißt die bedingte Wahrscheinlichkeit*

$$q_{ij}^{(m)} := P(X_{t+m} = j \,|\, X_t = i) \qquad \text{für } m \geq 0 \text{ und alle } i, j \in \mathcal{S}$$

*die $m$-stufige Übergangswahrscheinlichkeit für den Übergang vom Zustand $i$ in den Zustand $j$.*

*Entsprechend ist die $m$-stufige Übergangsmatrix definiert durch*

$$\mathbf{Q}^{(m)} = (q_{ij}^{(m)})_{i,j\in\mathcal{S}} \qquad \text{für } m \geq 0.$$

In den folgenden drei Sätzen werden einige wesentliche Beziehungen zwischen den $m$-stufigen Übergangs- und den absoluten Zustandswahrscheinlichkeiten einer homogenen Markov Kette erläutert. Für einen detaillierten Beweis verweisen wir an dieser Stelle auf einige Werke der Standardliteratur, wie [2, 11, 20, 38], um nur einige wenige zu nennen. Dabei werden bei der Beweisführung die Markov Eigenschaft sowie das Gesetz der totalen Wahrscheinlichkeit ausgenutzt, das Prinzip der vollständigen Induktion eingesetzt und elementare Operationen mit Matrizen durchgeführt.

**3.1.10 Satz**
*Die Formel von Chapman und Kolmogorov stellt zwischen den $m$-stufigen Übergangswahrscheinlichkeiten folgende Beziehung her:*

$$\begin{aligned} q_{ij}^{(m)} &= \sum_{k\in\mathcal{S}} q_{ik}^{(n)} q_{kj}^{(m-n)} \quad \text{für } 0 \le n \le m \text{ und alle } i,j \in \mathcal{S} \\ \mathbf{Q}^{(m)} &= \mathbf{Q}^{(n)}\mathbf{Q}^{(m-n)} \quad \text{für } 0 \le n \le m. \end{aligned}$$

**3.1.11 Satz**
*Ferner lassen sich die nachstehenden Aussagen aus der Formel von Chapman und Kolmogorov für die $m$-stufigen Übergangswahrscheinlichkeiten $q_{ij}^{(m)}$ herleiten:*

$$\begin{aligned} &(i) \quad q_{ij}^{(m)} = \sum_{i_1\in\mathcal{S}} \cdots \sum_{i_{m-1}\in\mathcal{S}} q_{ii_1} q_{i_1 i_2} \cdots q_{i_{m-2} i_{m-1}} q_{i_{m-1} j} \quad \text{für } m \ge 0 \text{ und alle } i,j \in \mathcal{S} \\ &(ii) \quad q_{ij}^{(m)} = \sum_{k\in\mathcal{S}} q_{ik} q_{kj}^{(m-1)} \quad \text{für } m > 0 \text{ und alle } i,j \in \mathcal{S} \\ &(iii) \quad \mathbf{Q}^{(m)} = \mathbf{Q}^m \quad \text{für } m \ge 0 \end{aligned}$$

**3.1.12 Satz**
*Überdies erfüllen die absoluten Zustandswahrscheinlichkeiten $\boldsymbol{\pi}^{(m)} = (\pi_1^{(m)}, \pi_2^{(m)}, \ldots)$ einer homogenen Markov Kette $(X_t)_{t\in\boldsymbol{N}}$ die Gleichungen:*

$$\begin{aligned} &(i) \quad \pi_j^{(m)} = \sum_{i\in\mathcal{S}} \pi_i^{(0)} q_{ij}^{(m)} \quad \text{für } m \ge 0 \text{ und alle } j \in \mathcal{S} \\ &(ii) \quad \boldsymbol{\pi}^{(m)} = \boldsymbol{\pi}^{(0)}\mathbf{Q}^m \quad \text{für } m \ge 0 \end{aligned}$$

Aus den vorangegangenen Sätzen lassen sich die nachfolgenden Hypothesen ableiten:

Im weiteren Verlauf dieses Abschnittes zeigen wir, dass diese beiden Hypothesen für eine spezielle Klasse von Markov Ketten gültig sind. Um diese Markov Ketten näher zu charakterisieren, klassifizieren wir die Zustände von Markov Ketten. Kriterium für eine Klassifizierung der Zustände ist die Wahrscheinlichkeit, dass eine Markov Kette in einem Zustand $i \in \mathcal{S}$ beginnend diesen Zustand $i$ nach endlich vielen oder auch abzählbar unendlich vielen Schritten wieder erreicht.

1. Hypothese: Für wachsendes $m \geq 0$ konvergiert die Folge der Matrizen $(\mathbf{Q}^m)_{m\geq 0}$. Als Grenzwert $\lim_{m\to\infty} \mathbf{Q}^m$ existiert eine Matrix, deren Zeilen identisch sind und jeweils dem Grenzwert $\lim_{m\to\infty} \boldsymbol{\pi}^{(m)}$ entsprechen.

2. Hypothese: Die Wahrscheinlichkeit $\pi_i^{(m)}$, dass sich die Markov Kette nach $m$ Schritten in einem gegebenen Zustand $i \in \mathcal{S}$ befindet, ist für ein $m$, das groß genug ist, unabhängig vom Anfangszustand $\pi_i^{(0)}$ bzw. von der Anfangsverteilung $\boldsymbol{\pi}^{(0)}$. Die Markov Kette verliert demnach ihr Gedächtnis über den Anfangszustand.

**3.1.13 Definition**
*Es sei eine homogene Markov Kette $(X_t)_{t\in N}$ mit diskretem Zustandsraum $\mathcal{S}$ gegeben. Dann lassen sich die Zustände dieser Markov Kette wie folgt klassifizieren:*

(i) *Ein Zustand $i \in \mathcal{S}$ heißt absorbierend, wenn die Übergangswahrscheinlichkeit $q_{ii} = 1$ ist; d.h. nach dem Erreichen des Zustandes $i \in \mathcal{S}$ verbleibt die Markov Kette in diesem Zustand $i$ und verlässt ihn nicht mehr.*

(ii) *Ein Zustand $i \in \mathcal{S}$ heißt transient, wenn unter der Voraussetzung, dass die Markov Kette im Zustand $i \in \mathcal{S}$ beginnt, die Wahrscheinlichkeit, dass die Markov Kette nicht mehr in diesen Zustand $i$ zurückkehrt, echt positiv ist.*

(iii) *Ein Zustand $i \in \mathcal{S}$ heißt rekurrent, wenn unter der Voraussetzung, dass die Markov Kette im Zustand $i \in \mathcal{S}$ beginnt, die Wahrscheinlichkeit, dass die Markov Kette nach einer endlichen oder auch abzählbar unendlichen Anzahl von Schritten in diesen Zustand $i$ zurückkehrt, stets gleich 1 ist.*

*In diesem Zusammenhang unterscheiden wir positiv rekurrente und null rekurrente Zustände. Ein rekurrenter Zustand $i$ heißt positiv rekurrent, wenn die Markov Kette im Mittel nach endlich vielen Schritten wieder in diesen Zustand $i$ zurückkehrt, und null rekurrent, wenn die Markov Kette im Mittel unendlich vielen Schritte für die Rückkehr in den Zustand $i$ benötigt.*

(iv) *Ein Zustand $i \in \mathcal{S}$ heißt periodisch mit der Periode $d$, wenn die Markov Kette ausgehend von diesem Zustand $i \in \mathcal{S}$ den Zustand $i$ nur nach einem Vielfachen von $d$ Schritten wiedererreicht; d.h. für die $m$-stufige Übergangswahrscheinlichkeit $q_{ii}^{(m)}$ gilt:*

$$q_{ii}^{(m)} \begin{cases} > 0 \text{ ; für } \quad m = rd \quad \text{mit } d > 1 \text{ und } r \in I\!N \\ = 0 \text{ ; sonst} \end{cases}$$

(v) *Ein Zustand $i \in \mathcal{S}$ heißt aperiodisch, wenn er nicht periodisch ist.*

(vi) *Ein Zustand $i \in \mathcal{S}$ heißt ergodisch, wenn er positiv rekurrent und aperiodisch ist.*

(vii) *Zwei Zustände $i \in \mathcal{S}$ und $j \in \mathcal{S}$ heißen wechselseitig erreichbar oder auch verbunden, wenn ein $m \geq 0$ und ein $n \geq 0$ derart existieren, dass die $m$-stufigen Übergangswahrscheinlichkeiten $q_{ij}^{(m)}$ und $q_{ji}^{(n)}$ positiv sind, d.h.*

$$\exists_{m,n\geq 0} \quad \text{mit} \quad q_{ij}^{(m)}, q_{ji}^{(n)} > 0.$$

Unter Zuhilfenahme dieser Klassifizierung der Zustände einer Markov Kette charakterisieren wir spezielle Klassen von Markov Ketten.

**3.1.14 Definition**
*Es sei eine homogene Markov Kette $(X_t)_{t\in N}$ mit Zustandsraum $\mathcal{S}$ gegeben.*

*(i) Dann heißt diese Markov Kette irreduzibel, falls jeder Zustand $j \in \mathcal{S}$ mit jedem Zustand $i \in \mathcal{S}$ verbunden ist; d.h.*

$$\forall_{i,j\in\mathcal{S}} \quad \exists_{m\geq 0} \quad mit \quad q_{ij}^{(m)} > 0.$$

*Andernfalls heißt die Markov Kette reduzibel.*

*(ii) Eine Markov Kette $(X_t)_{t\in N}$ mit Zustandsraum $\mathcal{S}$ heißt ergodisch, falls alle Zustände der Markov Kette ergodisch sind.*

Für endliche Markov Ketten $(X_t)_{t\in N}$ gelten insbesondere die folgenden Aussagen, die hier wiederum ohne Beweis angegeben werden. Ein detaillierter Beweis dieser Aussagen findet sich in [11] und [38].

**3.1.15 Satz**
*(i) Eine endliche irreduzible Markov Kette besteht lediglich aus positiv rekurrenten Zuständen.*

*(ii) Eine endliche irreduzible aperiodische Markov Kette ist demnach ergodisch und heißt auch regulär.*

Im Hinblick auf die zuvor aufgestellten Hypothesen führen wir den Begriff der *stationären Zustandswahrscheinlichkeiten* als Grenzwert der absoluten Zustandswahrscheinlichkeiten $\pi_i^{(m)}$ ein. Es ist zu analysieren, unter welchen Bedingungen dieser Grenzwert existiert, und wenn er existiert, ob der Grenzwert dann von der Anfangsverteilung $\boldsymbol{\pi}^{(\mathbf{0})}$ der Markov Kette unabhängig ist.

**3.1.16 Definition**
*Sei eine homogene Markov Kette $(X_t)_{t\in N}$ mit dem Zustandsraum $\mathcal{S}$ und der Anfangsverteilung $\boldsymbol{\pi}^{(\mathbf{0})} = (\pi_1^{(0)}, \pi_2^{(0)}, \ldots)$ gegeben. Falls der Grenzwert*

$$\pi_i := \lim_{m\to\infty} \pi_i^{(m)} \qquad \text{für alle } i \in \mathcal{S}$$

*existiert, heißt der Vektor $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$ die Grenzverteilung der Markov Kette und die Komponenten $\pi_i$ der Grenzverteilung werden als stationäre Zustandswahrscheinlichkeiten $\pi_i$ bezeichnet.*

Der nachstehende Satz zeigt, für welche Klasse von Markov Ketten die stationären Zustandswahrscheinlichkeiten existieren, und dass sie von der Anfangsverteilung der Markov

Kette unabhängig sind. Ferner wird hier eine einfache Möglichkeit dargelegt, wie die stationären Zustandswahrscheinlichkeiten bestimmt werden können. Zum Beweis verweisen wir ein weiteres Mal auf die Standardliteratur [11, 20] oder auch [38].

**3.1.17 Satz**
*Ist eine irreduzible ergodische Markov Kette $(X_t)_{t\in N}$ mit dem diskreten Zustandsraum $\mathcal{S}$ gegeben, dann existieren die stationären Zustandswahrscheinlichkeiten $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$ unabhängig von der Anfangsverteilung $\boldsymbol{\pi}^{(0)}$ der Markov Kette und sind eindeutig festgelegt durch das Gleichungssystem*

$$\pi_j = \sum_{i\in\mathcal{S}} \pi_i q_{ij} \qquad \text{für alle } j \in \mathcal{S}$$

*und die Normalisierungsbedingung*

$$\sum_{i\in\mathcal{S}} \pi_i = 1.$$

*In Matrixschreibweise erhalten wir entsprechend*

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{Q} \qquad \text{und} \qquad |\boldsymbol{\pi}| = 1.$$

Zum Abschluss dieses Abschnittes befassen wir uns mit einer Verallgemeinerung des Begriffes einer Markov Kette und definieren hierzu sogenannte Markov'sche Prozesse. Dabei verstehen wir unter einem Markov Prozeß einen stochastischen Prozeß, der die Markov Eigenschaft aus Definition 3.1.3 erfüllt. Voraussetzung für die Verallgemeinerung des Konzeptes einer Markov Kette sind die in der nachstehenden Definition eingeführten Begriffe, die sogenannte *bedingte* und die *unbedingte Verweildauer* eines Markov Prozesses in einem Zustand $i \in \mathcal{S}$.

**3.1.18 Definition**
*Es sei ein homogener Markov Prozeß $(X_t)_{t\geq 0}$ mit dem diskreten Zustandsraum $\mathcal{S}$ gegeben.*

*(i) Die stetige Zufallsvariable $H_{i,j}$, für $i, j \in \mathcal{S}$, $i \neq j$, gebe die Zeit an, die der Markov Prozeß im Zustand $i \in \mathcal{S}$ verweilt, bevor ein Zustandsübergang vom Zustand $i$ in einen anderen Zustand $j \in \mathcal{S}$ erfolgt. $H_{i,j}$ wird als die bedingte Verweildauer des Markov Prozesses im Zustand $i$ bezeichnet.*

*(ii) Die stetige Zufallsvariable $H_i$, für $i \in \mathcal{S}$, gebe die Zeit an, die der Markov Prozeß im Zustand $i \in \mathcal{S}$ unabhängig vom nächsten Zustandsübergang verweilt. $H_i$ wird als die unbedingte Verweildauer bzw. die Aufenthaltszeit des Markov Prozesses im Zustand $i$ bezeichnet.*

**3.1.19 Bemerkung**
Ist ein homogener Markov Prozeß $(X_t)_{t\geq 0}$ mit dem diskreten Zustandsraum $\mathcal{S}$ und der Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$ gegeben, dann gilt für die Verteilungsfunktion der Aufenthaltszeit und die Verteilungsfunktion der bedingten Verweildauer des Markov Prozesses im

Zustand $i \in \mathcal{S}$ die Beziehung:

$$P(H_i \leq x) = \sum_{\substack{j \in \mathcal{S} \\ i \neq j}} q_{ij} \, P(H_{i,j} \leq x)$$

**3.1.20 Satz**
*Für einen homogenen Markov Prozeß $(X_t)_{t \geq 0}$ mit einem diskreten Zustandsraum $\mathcal{S}$ sind die Aufenthaltszeiten $H_i$ für jeden Zustand $i \in \mathcal{S}$ stets exponentialverteilt.*

**Beweis**
Wir gehen von der Annahme aus, dass sich der Markov Prozeß $(X_t)_{t \geq 0}$ zu einem beliebigen Zeitpunkt $t \in \mathcal{T}$ im Zustand $i \in \mathcal{S}$ befindet. Zur Bestimmung der Wahrscheinlichkeit, dass sich der Markov Prozeß für mindestens weitere $\tau$ Zeiteinheiten im Zustand $i$ befindet, ist die bedingte Wahrscheinlichkeit

$$P(H_i > t + \tau \,|\, H_i > t)$$

zu ermitteln. Die Markov Eigenschaft impliziert, dass diese bedingte Wahrscheinlichkeit unabhängig von der Vergangenheit und damit unabhängig davon ist, wie lange sich der Markov Prozeß bereits im Zustand $i$ befindet. Daher gilt

$$P(H_i > t + \tau \,|\, H_i > t) = P(H_i > \tau).$$

Folglich ist die Aufenthaltszeit $H_i$ eines Markov Prozesses für jeden Zustand $i \in \mathcal{S}$ gedächtnislos. Ein Ergebnis der Stochastik im Hinblick auf stetige Zufallsvariablen und ihre Verteilungen ist, dass eine stetige und gedächtnislose Zufallsvariable stets exponentialverteilt ist, vgl. [32, 38]. Somit impliziert die Markov Eigenschaft, dass die Aufenthaltszeit $H_i$ eines Markov Prozesses für jeden Zustand $i \in \mathcal{S}$ exponentialverteilt ist. □

**3.1.21 Bemerkung**
Für eine homogene Markov Kette $(X_t)_{t \in \boldsymbol{N}}$ gilt insbesondere, dass die Aufenthaltszeit $H_i$, für $i \in \mathcal{S}$, und die bedingte Verweildauer $H_{i,j}$, für $i, j \in \mathcal{S}$, genau einer Zeiteinheit entsprechen und unabhängig vom nächsten Zustandsübergang sind; d.h. es gilt:

$$H_{i,j} = 1 \qquad \text{für alle} \qquad i, j \in \mathcal{S}, i \neq j.$$

Wie in Satz 3.1.20 liefert die Markov Eigenschaft, dass die Aufenthaltszeit $H_i$ einer Markov Kette für jeden Zustand $i \in \mathcal{S}$ gedächtnislos ist. Aus diesem Grund ist die Aufenthaltszeit $H_i$ einer Markov Kette $(X_t)_{t \in \boldsymbol{N}}$ für jeden Zustand $i \in \mathcal{S}$ geometrisch verteilt. Es gilt nämlich:

$$P(H_i > k) = \left( \sum_{\substack{i \in \mathcal{S} \\ i \neq j}} q_{ij} \right)^k = (1 - q_{ii})^k$$

Für einen homogenen Markov Prozeß $(X_t)_{t \geq 0}$ mit diskretem Zustandsraum $\mathcal{S}$ werden die Übergänge zwischen den Zuständen durch die Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j \in \mathcal{S}}$ gesteuert.

Die Aufenthaltszeiten $H_i$ des Prozesses sind unabhängig vom nächsten Zustandsübergang und für jeden Zustand $i \in \mathcal{S}$ exponentialverteilt. Nach der obigen Bemerkung sind die Aufenthaltszeiten $H_i$ einer homogenen Markov Kette entsprechend geometrisch verteilt. Auf eine derartige Markov Kette kommen wir in Abschnitt 3.3.2 zurück und können dort diese Ergebnisse für weitere Analysen nutzen.

Im allgemeinen müssen die Aufenthaltszeiten $H_i$ jedoch nicht unabhängig vom nächsten Zustandsübergang sein. Ferner liegt es nahe, beliebig verteilte Aufenthaltszeiten $H_i$ für jeden Zustand $i \in \mathcal{S}$ zuzulassen, da bei praktischen Anwendungen nicht immer von exponentialverteilten Aufenthaltszeiten ausgegangen werden kann. Dabei erfolgen die Zustandsübergänge weiterhin gemäß der Übergangsmatrix $\mathbf{Q}$. Diese Verallgemeinerungen führen zum Begriff des *semi-Markov Prozesses*.

**3.1.22 Definition**
*Ein semi-Markov Prozeß $(X_t)_{t\geq 0}$ mit diskretem Zustandsraum $\mathcal{S}$ und der Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$ ist zeitlich durch folgende Abläufe charakterisiert:*

*(i) Nach Erreichen des Zustandes $i \in \mathcal{S}$ wählt der semi-Markov Prozeß gemäß der Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$ den nachfolgenden Zustand $j \neq i$ aus.*

*(ii) Ist der Zustand $j$ festgelegt, so bestimmen die Verteilungsfunktionen für die bedingten Verweildauern $H_{i,j}$ die Zeit, die der semi-Markov Prozeß im Zustand $i$ vor dem Zustandsübergang in den Zustand $j$ verweilt.*

**3.1.23 Bemerkung**
Wenn ein homogener semi-Markov Prozeß $(X_t)_{t\geq 0}$ mit diskretem Zustandsraum $\mathcal{S}$ und der Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$ nur zu den Zeitpunkten betrachtet wird, an denen ein Zustandsübergang gemäß der Übergangsmatrix $\mathbf{Q}$ erfolgt, so wird auf diese Weise durch die Sprungpunkte eine Markov Kette mit der Übergangsmatrix

$$\mathbf{Q}^{(e)} = (q_{ij}^{(e)})_{i,j\in\mathcal{S}}$$

festgelegt, deren Einträge gegeben sind durch:

$$q_{ij}^{(e)} = \begin{cases} 0 \text{ ; für } i = j \\ \dfrac{q_{ij}}{1 - q_{ii}} \text{ ; } i \neq j \end{cases}$$

**3.1.24 Definition**
*Eine Markov Kette, die wie in Bemerkung 3.1.23 festgelegt wird und somit das Verhalten eines semi-Markov Prozesses $(X_t)_{t\geq 0}$ an seinen Sprungpunkten beschreibt, heißt die in den semi-Markov Prozeß $(X_t)_{t\geq 0}$ eingebettete Markov Kette.*
*Bei der Notation verwenden wir $^{(e)}$, um zu signalisieren, dass $\mathbf{Q}^{(e)}$ die Übergangsmatrix der eingebetteten Markov Kette ist.*

**3.1.25 Bemerkung**
Bezeichnen wir entsprechend mit $\pi_i^{(e)}$, für $i \in \mathcal{S}$, die stationären Zustandswahrscheinlichkeiten der in einen homogenen semi-Markov Prozeß $(X_t)_{t\geq 0}$ eingebetteten Markov Kette

$(X_t)_{t\in N}$, dann erfüllen die stationären Zustandswahrscheinlichkeiten $\pi_i^{(e)}$ der eingebetteten Markov Kette das Gleichungssystem

$$\pi_j^{(e)} = \sum_{i\neq j} \pi_i^{(e)} q_{ij}^{(e)}.$$

Dabei geben die stationären Zustandswahrscheinlichkeiten $\pi_i^{(e)}$, $i \in \mathcal{S}$, der eingebetteten Markov Kette die Wahrscheinlichkeiten an, dass sich der semi-Markov Prozeß zu den eingebetteten Zeitpunkten im Zustand $i$ befindet. Dies entspricht jedoch nicht den stationären Zustandswahrscheinlichkeiten $\pi_i$ des semi-Markov Prozesses, also den Wahrscheinlichkeiten, dass sich der semi-Markov Prozeß zu einem zufällig ausgewählten Zeitpunkt im Zustand $i$ befindet. Offensichtlich hängen aber die stationären Zustandswahrscheinlichkeiten $\pi_i$ des semi-Markov Prozesses sowohl von den stationären Zustandswahrscheinlichkeiten $\pi_i^{(e)}$ der eingebetteten Markov Kette als auch von den Aufenthaltszeiten $H_i$ der Zustände ab.

Zum Beweis des nachfolgenden Satzes sei auf [38] verwiesen.

**3.1.26 Satz**
*Ein homogener semi-Markov Prozeß $(X_t)_{t\geq 0}$ erfüllt nicht die Markov Eigenschaft.*

**3.1.27 Bemerkung**
Es gibt zwei Ausnahmen, wann ein semi-Markov Prozeß $(X_t)_{t\geq 0}$ die Markov Eigenschaft erfüllt:

(i) Wenn die Aufenthaltszeit $H_i$ und $H_{i,j}$ für jeden Zustand $i, j \in \mathcal{S}$, $i \neq j$, exponentialverteilt ist, liefert die Gedächtnislosigkeit der Exponentialverteilung die Markov Eigenschaft.

(ii) Wenn der Zustandsraum $\mathcal{S}$ des semi-Markov Prozesses $(X_t)_{t\geq 0}$ dahingehend erweitert wird, dass er eine zusätzliche Komponente enthält, die den Zeitaufwand mißt, wie lange der semi-Markov Prozeß bereits im aktuellen Zustand verweilt, so impliziert diese zusätzliche Information die Markov Eigenschaft. Allerdings führt diese Modifikation des Zustandsraumes zu einem kontinuierlichen Zustandsraum $\mathcal{S}$.

**3.1.28 Bemerkung**
Ist ein semi-Markov Prozeß $(X_t)_{t\geq 0}$ mit diskretem Zustandsraum $\mathcal{S}$ gegeben, dessen Aufenthaltszeiten $H_i$ für jeden Zustand $i \in \mathcal{S}$ endlich sind, dann läßt sich der semi-Markov Prozeß als Markov Kette darstellen. Dazu wird der diskrete Zustandsraum $\mathcal{S}$ des semi-Markov Prozesses insofern erweitert, als dass der Zustandsraum der eingebetteten Markov Kette mit einer zweiten Komponente $G_i$ versehen wird. Diese misst die Zeit, die der semi-Markov Prozeß vom Moment des Erreichens an in dem angenommenen Zustand verweilt; d.h. die zweite Komponente $G_i$ wird hochgezählt bzw. bei einem Zustandswechsel zurückgesetzt. Die erste Komponente $F_i$ ändert sich folglich nur, wenn die zweite zurückgesetzt wird. Auf diese Art und Weise können beispielsweise Poisson-verteilte Aufenthaltszeiten $H_i$ in den Zuständen dargestellt werden, worauf wir in Abschnitt 5.2.2 zurückkommen werden.

## 3.2 Der Fehlerprozeß während digitaler Datenübertragung

Nachdem wir in Abschnitt 3.1 die mathematischen Methoden zur Modellierung eines Übertragungskanals und insbesondere des während der Übertragung entstandenen Fehlerprozesses zusammengestellt haben, befassen wir uns in diesem Abschnitt mit einer geeigneten Darstellung der aufgetretenen physikalischen Störungen auf der Bitebene. Eine solche binäre Darstellung läßt sich auf unterschiedliche Art und Weise modellieren und führt daher zu verschiedenen Kanalmodellen, die wir in Abschnitt 3.3 angeben, analysieren und vergleichen.

Aus der in der Einleitung skizzierten Zielsetzung der Arbeit und den Ausführungen zu Beginn dieses Kapitels geht hervor, dass Gegenstand der nun folgenden Analysen und Untersuchungen unterschiedliche Konzepte und Methoden der Kanalcodierung sind. Demnach konzentrieren wir uns auf den in Abbildung 3.1 dargestellten Teil eines Kommunikationssystems. Dieser umfasst neben dem Kanalcodierer und dem Kanaldecodierer den Modulator sowie den Demodulator und den durch eine Rauschquelle beeinträchtigten physikalischen Kanal.

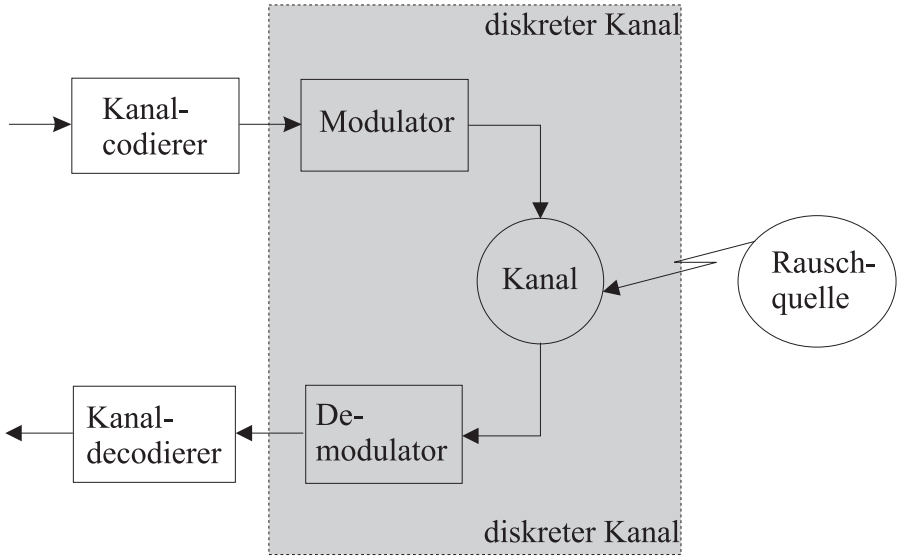


Abbildung 3.1: Kanalcodierer und -decodierer als Teil eines Übertragungssystems

Bislang haben wir wie in Kapitel 2 und Abbildung 3.1 den Begriff des Kanals stets im Sinne eines physikalischen Übertragungskanals verwendet und die Rauschquelle als eine physikalische Störquelle interpretiert. Nun konzentrieren wir uns innerhalb der Arbeit jedoch in erster Linie auf die unterschiedlichen Konzepte und Methoden der Kanalcodierung. Dies erfordert eine geeignete Beschreibung der aufgetretenen physikalischen Störungen auf der Bitebene, also eine passende Darstellung des Störungsprozesses, wie er sich konkret zwischen dem Kanalcodierer und dem Kanaldecodierer auf der Bitebene darstellt. Demzufolge benötigen wir eine zweckmäßige Interpretation der physikalischen Rauschquelle eines Kanals auf der Bitebene. Dazu fassen wir den Modulator, den Demodulator und den verrauschten physikalischen Kanal zu einem sogenannten *diskreten Kanal* zusammen und verwenden den

Begriff des Kanals von nun an im Sinne eines Übertragungskanals auf der Bitebene. In Abbildung 3.1 ist dies symbolisch dargestellt.

**3.2.1 Definition**
*In der Informationstheorie wird ein diskreter Kanal im allgemeinen charakterisiert durch ein $q$-näres Eingangsalphabet $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_q\}$ mit $q \geq 2$ und ein $r$-näres Ausgangsalphabet $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_r\}$ mit $r \geq 2$ sowie durch eine sogenannte Kanalmatrix*

$$\mathbf{P} = (p_{ij})_{i \in \mathcal{A},\, j \in \mathcal{B}} = (P(\beta_j \mid \alpha_i))_{i \in \mathcal{A},\, j \in \mathcal{B}},$$

*deren Einträge die Wahrscheinlichkeit angeben, mit welcher das Ausgabesymbol $\beta_j$ bei der Eingabe von $\alpha_i$ empfangen wird.*

Bei dem Design und der Implementation moderner Informations- und Kommunikationssysteme wird in der Regel auf binäre Zahldarstellungen zurückgegriffen. Auch in der Codierungstheorie werden mittels der binären Betrachtungsweisen die zahlentheoretischen und algebraischen Eigenschaften des Körpers mit zwei Elementen gezielt ausgenutzt.

Im Rahmen der Arbeit lassen wir daher der Einfachheit halber als Eingangs- und Ausgangssymbole für einen diskreten Kanal nur 0 und 1 zu. Gegenstand der Arbeit sind folglich ausschließlich diskrete Kanäle mit binärem Eingangs- und binärem Ausgangsalphabet, obgleich sich alle nachfolgenden Überlegungen analog auch auf $q$-näre Alphabete übertragen lassen. Dies motiviert die folgenden Bezeichnungen.

**3.2.2 Bezeichnungen**
*Entsprechend der Standardliteratur wird mit $GF(2)$ das Galois-Feld von 2, also der Körper $(\{0,1\}, \oplus, \odot)$ mit zwei Elementen bezeichnet, dessen Addition $\oplus$ und Multiplikation $\odot$ festgelegt sind durch die Verknüpfungstafeln:*

| $\oplus$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |

und

| $\odot$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |

**3.2.3 Definition**
*Ein diskreter Kanal mit dem Eingangs- und dem Ausgangsalphabet $GF(2)$ wird auch als ein digitaler Kanal auf Bitebene bezeichnet.*

Indem Modulation und Demodulation zusammen mit dem gestörten physikalischen Kanal als digitaler Kanal über dem Galois-Feld $GF(2)$ interpretiert werden, wird eine abstrakte Betrachtungsweise des Prozesses der Datenübertragung geschaffen. Diese abstrakte Betrachtungsweise ermöglicht es, eine adäquate Darstellung des Prozesses der Verfälschung der Daten und damit eine zweckdienliche Interpretation der physikalischen Rauschquelle auf der Bitebene zu entwickeln.

Bei den folgenden Überlegungen gehen wir von einem einfachen Modell eines digitalen Kanals aus. Dabei stellt sich der Prozeß der Datenübertragung ganz abstrakt wie in Abbildung 3.2 dar.

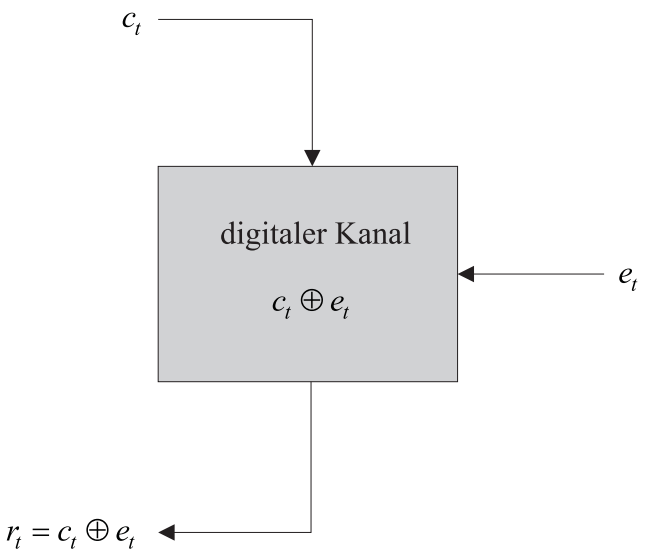


Abbildung 3.2: Modell eines digitalen Kanals

Hier symbolisiert $(c_t)_{t\in\mathbb{N}}$, mit $c_t \in GF(2)$, die Folge der gesendeten Bits, die die zu übertragende Information codiert beinhaltet. Dabei gibt $t \in \mathbb{N}$ den Zeitpunkt der Übertragung an. Eine geeignete Darstellung der zufällig auftretenden physikalischen Störungen auf der Bitebene wird durch die Identifikation der physikalischen Rauschquelle des Kanals mit einer binären Fehlerquelle auf der Bitebene erreicht. Diese produziert eine binäre Folge von Fehlern $(f_t)_{t\in\mathbb{N}}$, mit $f_t \in GF(2)$. Gemäß den Regeln der binären Addition $\oplus$ wird ein Bitfehler $f_t \in GF(2)$ der binären Fehlerquelle im digitalen Kanal zu dem gesendeten Bit $c_t$ hinzuaddiert. Folglich liefert der digitale Kanal am Ausgang eine Folge von Bits $(r_t)_{t\in\mathbb{N}}$, mit $r_t \in GF(2)$, für die gilt:

$$r_t = c_t \oplus f_t \qquad \text{mit } t \in \mathbb{N}.$$

Diese Folge erhält der Kanaldecodierer schließlich als Eingangssequenz.

### 3.2.4 Bemerkung

Abbildung 3.2 veranschaulicht, dass das Modell eines digitalen Kanals eine zweckmäßige Interpretation der physikalischen Rauschquelle auf der Bitebene ermöglicht. So wird die physikalische Rauschquelle eines Kanals mit einer binären Fehlerquelle auf Bitebene identifiziert, so dass die zufällig auftretenden Störungen des physikalischen Übertragungskanals durch eine Folge von Bitfehlern repräsentiert werden können. Auf diese Art und Weise entsteht eine geeignete Darstellung und Beschreibung der zufällig auftretenden Störungen eines physikalischen Kanals in Form eines binären Fehlerprozesses auf Bitebene, welcher im folgenden Gegenstand der Untersuchungen sein wird.

## 3.3 Digitale Kanalmodelle: ein Überblick

Gegenstand dieses Abschnittes ist die Präsentation einiger grundlegender, aus der Literatur bekannter Kanalmodelle, welche den von der physikalischen Rauschquelle induzierten, binären Fehlerprozeß mit geeigneten mathematischen Methoden modellieren. Dazu wird der binäre Fehlerprozeß als stochastischer Prozeß interpretiert. Indem diesem stochastischen Fehlerprozeß weitere zusätzliche Eigenschaften und Merkmale zugewiesen werden, werden unterschiedliche Methoden der Modellierung geschaffen, die dann ihrerseits unterschiedliche Kanalmodelle induzieren. Wir diskutieren in diesem Abschnitt einige Erweiterungen und Verallgemeinerungen der elementaren Modelle. Ferner analysieren wir die Vor- und Nachteile der verschiedenen Kanalmodelle und vergleichen diese im Hinblick auf die Zielsetzungen der Arbeit. Dabei stellen wir insbesondere einen gedächtnislosen Kanal verschiedenen Modellen, die ihrerseits gedächtnisbehaftete Kanäle abbilden, gegenüber.

Bevor wir die verschiedenen digitalen Kanalmodelle im einzelnen vorstellen, geben wir zu Beginn dieses Abschnittes einige allgemeine Grundlagen und Voraussetzungen an, die auf sämtliche nachfolgenden Modelle zutreffen.

### 3.3.1 Bemerkung

(i) Bei der Entwicklung eines Kanalmodells gehen wir davon aus, dass zufällig ausgewählte Bits zur Übertragung über den Kanal gesendet werden. Somit kann ein Bit $c_t \in GF(2)$ der gesendeten Folge von Bits $(c_t)_{t\in\mathbb{N}}$ zu jedem Zeitpunkt $t \in \mathbb{N}$ als Realisierung einer Zufallsvariablen $C_t$, $t \in \mathbb{N}$, angesehen werden. Ferner ist diese Aussage ebenfalls für ein Fehlerbit $f_t \in GF(2)$ des binären Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ gültig, so dass jedes Fehlerbit $f_t \in GF(2)$ zu jedem Zeitpunkt $t \in \mathbb{N}$ als Realisierung einer Zufallsvariablen $E_t$, $t \in \mathbb{N}$, betrachtet werden kann. Folglich stellen das zu übertragende Bit $c_t$ und das Fehlerbit $f_t$ zu jedem Zeitpunkt $t \in \mathbb{N}$ zufällige Größen $C_t$ und $E_t$ dar. Diese können entweder den Wert 0 oder 1 annehmen, also $C_t, E_t \in \{0, 1\}$ für jedes $t \in \mathbb{N}$. Während $E_t = 1$ in diesem Zusammenhang das Auftreten eines Fehlers symbolisiert, stellt $E_t = 0$ die fehlerfreie Übertragung des entsprechenden Bits $C_t$ dar.

(ii) Auch die Folge der empfangenen Bits $(r_t)_{t\in\mathbb{N}}$ kann zu jedem Zeitpunkt $t \in \mathbb{N}$ als Realisierung einer Zufallsvariablen $R_t$ interpretiert werden, denn $r_t$ ergibt sich als binäre Summe aus

$$r_t := c_t \oplus f_t \qquad \text{mit } t \in \mathbb{N}.$$

(iii) Von daher bilden aus mathematischer Sicht die Folgen der gesendeten Bits $(c_t)_{t\in\mathbb{N}}$, der Fehlerbits $(f_t)_{t\in\mathbb{N}}$ und der empfangenen Bits $(r_t)_{t\in\mathbb{N}}$ zufällige, also stochastische Prozesse $(C_t)_{t\in\mathbb{N}}$, $(E_t)_{t\in\mathbb{N}}$ und $(R_t)_{t\in\mathbb{N}}$, für die gilt

$$C_t, E_t, R_t \in GF(2) \qquad \text{für jedes } t \in \mathbb{N}$$

$$\text{und} \qquad R_t := C_t \oplus E_t \qquad \text{für jedes } t \in \mathbb{N}.$$

Mit der mathematischen Interpretation der Folge der gesendeten Bits $(c_t)_{t\in\mathbb{N}}$ und der Folge der Bitfehler $(f_t)_{t\in\mathbb{N}}$ als stochastische Prozesse $(C_t)_{t\in\mathbb{N}}$ und $(E_t)_{t\in\mathbb{N}}$ erhalten wir die

Möglichkeit, Abhängigkeiten zwischen den Mengen der Zufallsvariablen $(C_t)_{t\in\mathbb{N}}$ und $(E_t)_{t\in\mathbb{N}}$ darzustellen. Zum einen lassen sich komplexere Abhängigkeiten zwischen Mengen von Zufallsvariablen $(E_t)_{t\in\mathbb{N}}$ charakterisieren. Zum anderen können wir Abhängigkeiten zwischen den Zufallsvariablen $C_t$ und $E_t$ zu einem beliebigen Zeitpunkt $t \in \mathbb{N}$ betrachten. In diesem Zusammenhang führen wir den Begriff der *Symmetrie* eines digitalen Kanals ein.

**3.3.2 Definition**
*Ein digitaler Kanal heißt symmetrisch, falls der Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ zu jedem beliebigen Zeitpunkt $t \in \mathbb{N}$ unabhängig vom Prozeß der gesendeten Bits $(C_t)_{t\in\mathbb{N}}$ ist.*

Nach Definition 3.3.2 ist der Begriff der Symmetrie eines digitalen Kanals derart zu interpretieren, dass es für das Auftreten eines Fehlers nicht ausschlaggebend ist, ob eine 0 oder eine 1 gesendet worden ist. Für beliebige Zeitpunkte $t_1, t_2 \in \mathbb{N}$ ist $E_{t_1} = 1$ unabhängig davon, ob $C_{t_2} = 0$ oder $C_{t_2} = 1$ über den Kanal gesendet worden ist. Für die bedingte Wahrscheinlichkeit resultiert daraus:

$$P(E_t = 1 \,|\, C_t = 0) = P(E_t = 1 \,|\, C_t = 1).$$

Von daher kann ein digitaler symmetrischer Kanal allein durch das Verhalten seines Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ charakterisiert werden. Aus diesem Grund sind im Rahmen dieser Arbeit ausschließlich digitale symmetrische Kanäle Gegenstand der Untersuchungen. In den folgenden Ausführungen beschränken wir uns daher auf die Beschreibung und Modellierung des während der Übertragung erzeugten Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$.

### 3.3.1 Der symmetrische Binärkanal

In diesem Paragraphen modellieren wir den während der Übertragung aufgetretenen, binären Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ eines digitalen Kanals mit dem einfachsten und wohl auch dem bekanntesten Kanalmodell, dem *gedächtnislosen symmetrischen Binärkanal*.

**3.3.3 Definition**
*Ein digitaler, symmetrischer Kanal heißt gedächtnislos, falls für zwei beliebige Zeitpunkte $t_1, t_2 \in \mathbb{N}$, mit $t_1 \neq t_2$, das Fehlerbit $E_{t_1}$ unabhängig von allen anderen Fehlerbits $E_{t_2}$ vorher und nachher ist.*

*In der Literatur wird dieser Kanal häufig auch kurz als symmetrischer Binärkanal bzw. als binary symmetrical channel (BSC) bezeichnet.*

Laut Definition 3.3.3 impliziert das Kanalmodell des gedächtnislosen symmetrischen Binärkanals (BSC), dass die Zufallsvariablen $E_t$ des binären Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ unabhängig und identisch verteilt sind.

**3.3.4 Definition**
*Für einen symmetrischen Binärkanal (BSC) wird die Wahrscheinlichkeit für die fehlerhafte Übertragung eines beliebigen Bits als Einzelfehlerwahrscheinlichkeit $p_E$ oder auch als*

*Bitfehlerwahrscheinlichkeit $p_E$ bezeichnet und definiert durch*

$$p_E := P(E_t = 1).$$

*Dementsprechend ergibt sich die Wahrscheinlichkeit einer fehlerfreien Übertragung eines beliebigen Bits zu*

$$P(E_t = 0) = 1 - p_E.$$

### 3.3.5 Bemerkung

(i) Aufgrund der Symmetrieeigenschaft des symmetrischen Binärkanals (BSC) gilt, dass die Bitfehlerwahrscheinlichkeit $p_E$ stets von der über den Kanal gesendeten Folge von Bits unabhängig ist, also unabhängig davon, ob eine 0 oder eine 1 gesendet worden ist. Somit gilt stets:

$$\begin{aligned} p_E = P(E_t = 1) &= P(E_t = 1 \,|\, C_t = 0) \\ &= P(E_t = 1 \,|\, C_t = 1). \end{aligned}$$

(ii) Die Gedächtnislosigkeit des symmetrischen Binärkanals (BSC) beschreibt ihrerseits die Beziehung der einzelnen Fehler untereinander. Da der Fehlerprozeß $(E_t)_{t \in \mathbb{N}}$ eines symmetrischen Binärkanals (BSC) gegeben ist durch einen stochastischen Prozeß von unabhängigen und identisch verteilten Zufallsvariablen $E_t$, sind die Einzelfehler, die an verschiedenen Stellen auftreten, unabhängig voneinander. Folglich sind mit dem Kanalmodell des gedächtnislosen symmetrischen Binärkanals (BSC) ausschließlich zufällig verteilte, also voneinander unabhängige Fehler darstellbar, korrelierte Fehler dagegen nicht.

(iii) Ferner ist die Bitfehlerwahrscheinlichkeit $p_E$ aufgrund der Gedächtnislosigkeit unabhängig von den vorherigen Bits des Fehlerprozesses und damit stets konstant.

(iv) Für einen symmetrischen Binärkanal ist die Kanalmatrix aus Definition 3.2.1 gerade gegeben durch:

$$\begin{aligned} \mathbf{P} &= \begin{pmatrix} P(C_t = 0 \,|\, R_t = 0) & P(C_t = 0 \,|\, R_t = 1) \\ P(C_t = 1 \,|\, R_t = 0) & P(C_t = 1 \,|\, R_t = 1) \end{pmatrix} \\ &= \begin{pmatrix} P(E_t = 0) & P(E_t = 1) \\ P(E_t = 1) & P(E_t = 0) \end{pmatrix} = \begin{pmatrix} 1 - p_E & p_E \\ p_E & 1 - p_E \end{pmatrix} \end{aligned}$$

Im Hinblick auf die Zielsetzung der Arbeit, eine Beurteilung verschiedener Codierverfahren in Abhängigkeit von den spezifischen Eigenschaften des Übertragungskanals, diskutieren wir nun abschließend die Vor- und Nachteile, die die Verwendung des symmetrischen Binärkanals als Kanalmodell für einen fehlerbehafteten, digitalen Kanal mit sich bringt.

### 3.3.6 Bemerkung

Das Kanalmodell des BSC besitzt gegenüber anderen Modellen einen wesentlichen Vorteil. Aufgrund der beiden Annahmen, Symmetrie und Gedächtnislosigkeit, wird der symmetrische Binärkanal aus mathematischer Sicht zu einem einfach zu handhabenden Modell für

einen fehlerbehafteten, digitalen Kanal und die Modellparameter sind einfach zu interpretieren, so dass die abstrakte Darstellung des Übertragungsprozesses noch einen gewissen Anschauungswert besitzt. Aus diesem Grund dient der symmetrische Binärkanal in den meisten Lehrbüchern zur Informations- und Codierungstheorie als klassisches Modell für einen fehlerbehafteten, digitalen Kanal, auf dessen Basis eine Bewertung verschiedener Codierverfahren erfolgt, vgl. hierzu [7, 13, 26, 43].

Demgegenüber stehen die folgenden Nachteile. Bereits in der Einleitung haben wir deutlich gemacht, dass die verschiedenen Codierverfahren auf unterschiedlichen algebraischen Strukturen beruhen und daher von unterschiedlicher Qualität mit Blick auf die Fehlerkorrektur sein werden. Aufgrund der Gedächtnislosigkeit des BSC können diese qualitativen Unterschiede bei einer Beurteilung auf der Basis des BSC nicht berücksichtigt werden. Folglich können Codierverfahren, die das Gedächtnis eines digitalen Kanals bei der Fehlerkorrektur ausnutzen, basierend auf dem BSC aus diesem Grund nicht angemessen beurteilt und bewertet werden.

Ferner bildet der symmetrische Binärkanal auch aus dem folgenden Grund keine geeignete Grundlage für eine Bewertung verschiedener Fehlerkorrekturverfahren bzw. für die Auswahl eines optimalen Verfahrens. Reale Kanäle zeichnen sich in der Regel durch eine weniger einfache Struktur als die des symmetrischen Binärkanals aus, vor allem wenn es sich um Funkübertragungskanäle handelt, wie bei Satellitenübertragungsstrecken oder bei Anwendungen aus dem Mobilfunkbereich. Oft betreffen Störungen des Übertragungsweges in diesen Fällen typischerweise mehrere aufeinander folgende Bits. In Kapitel 2 haben wir auf diese Problematik bereits hingewiesen, dass Übertragungsfehler in realen Kanälen häufig in zeitlicher Nähe auftreten. Wenn ein Bit gestört worden ist, ist die Wahrscheinlichkeit, dass ein benachbartes Bit verfälscht wird, wesentlich größer als die Wahrscheinlichkeit, dass ein Nachbarbit eines ungestörten Bits beschädigt wird.

In der Arbeit [24] von Kröll ist ein Auszug aus einer Fehlermessung abgebildet, die vom Deutschen Zentrum für Luft und Raumfahrt (DLR) in Auftrag gegeben worden ist. Hier zeigt sich, dass Übertragungsfehler nicht gleich verteilt sind und häufig direkt hintereinander zum Teil in längeren Ketten mit vereinzelten Lücken auftreten. Offensichtlich sind Übertragungsfehler nicht unabhängig voneinander. Bei realen Anwendungen, wie bei Satellitenübertragungsstrecken oder dem Mobilfunkbereich, handelt es sich offensichtlich um Übertragungskanäle mit Gedächtnis. Entgegen den Annahmen des BSC existieren korrelierte Fehlerpositionen.

### 3.3.2 Das Modell von Gilbert

Um das Gedächtnis eines realen Übertragungskanals und damit auch korrelierte Fehlerpositionen adäquat abbilden zu können, ist die Annahme der Gedächtnislosigkeit fallen zu lassen und eine entsprechende Erweiterung des bisherigen Kanalmodells zu entwickeln. In der Literatur gibt es hierzu in diversen Artikeln eine Vielzahl von Ansätzen. Eine der ersten Arbeiten in diesem Zusammenhang geht auf E. N. Gilbert zurück, der 1960 in seinem Artikel [14] einen einfachen Ansatz zur Modellierung und Darstellung des Gedächtnisses eines fehlerbehafteten, digitalen Kanals präsentiert hat. In dieser Arbeit wird der Fehlerprozeß $(E_t)_{t\in N}$ mittels einer Markov Kette erster Ordnung modelliert. In Abschnitt 3.1 haben wir bereits gezeigt, dass Markov Ketten in sehr einfacher Form ein

Gedächtnis besitzen.

In seinem Modellansatz verwendet Gilbert zur Modellierung des Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ eine Markov Kette $(X_t)_{t\in\mathbb{N}}$ erster Ordnung mit endlichem Zustandsraum $\mathcal{S}$, vgl. [14], die gekennzeichnet ist durch

(i) den Anfangszustand $X_0 = i_0$, mit $i_0 \in \mathcal{S}$

(ii) und ihre Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$, wobei für alle $i, j \in \mathcal{S}$ gilt:

$$\begin{aligned} q_{ij} &= P(X_{t+1} = j \,|\, X_t = i,\, X_{t-1} = i_{t-1}, \ldots, X_0 = i_0) \\ &= P(X_{t+1} = j \,|\, X_t = i)\,. \end{aligned}$$

Insbesondere besteht der Zustandsraum $\mathcal{S}$ der Markov Kette $(X_t)_{t\in\mathbb{N}}$ in [14] aus genau zwei Zuständen, $\mathcal{S} = \{G, B\}$. Dabei bezeichnet $G$ den guten Übertragungszustand *good* und $B$ den schlechten Übertragungszustand *bad* des Kanals.

Im Modell von Gilbert werden die Übergangswahrscheinlichkeiten folgendermaßen gesetzt, wobei die Parameter $p, q \in [0, 1]$ sind:

$$\begin{aligned} q_{GG} &= P(X_{t+1} = G \,|\, X_t = G) = 1 - p, \\ q_{GB} &= P(X_{t+1} = G \,|\, X_t = B) = p, \\ q_{BG} &= P(X_{t+1} = B \,|\, X_t = G) = q \\ \text{und} \quad q_{BB} &= P(X_{t+1} = B \,|\, X_t = B) = 1 - q. \end{aligned}$$

Ferner geht Gilbert für alle weiteren Betrachtungen von der Annahme aus, dass die Markov Kette $(X_t)_{t\in\mathbb{N}}$ stets homogen, irreduzibel und aperiodisch ist. Aus Abschnitt 3.1 Satz 3.1.15 folgt, dass die Markov Kette regulär ist und dann insbesondere die stationären Zustandswahrscheinlichkeiten $\pi_G$ und $\pi_B$ existieren, mit

$$\begin{aligned} \pi_G &= \lim_{m\longrightarrow\infty} \pi_G^{(m)} \\ \text{und} \quad \pi_B &= \lim_{m\longrightarrow\infty} \pi_B^{(m)}. \end{aligned}$$

Diese erfüllen nach Satz 3.1.17 sowohl das Gleichungssystem

$$\begin{aligned} \pi_G &= \pi_G\, q_{GG} + \pi_B\; q_{BG} \\ \wedge \quad \pi_B &= \pi_G\, q_{GB} + \pi_B\; q_{BB} \end{aligned}$$

als auch die Normalisierungsbedingung

$$\pi_G + \pi_B = 1,$$

so dass sich für die stationären Zustandswahrscheinlichkeiten $\pi_G$ und $\pi_B$ die nachfolgende Darstellung ergibt:

$$\begin{aligned} \pi_G &= \frac{q_{BG}}{q_{GB} + q_{BG}} = \frac{q}{p+q} \\ \pi_B &= \frac{q_{GB}}{q_{GB} + q_{BG}} = \frac{p}{p+q} \end{aligned}$$

Nun ist eine zweckmäßige Beziehung zwischen den Zuständen der betrachteten Markov Kette $(X_t)_{t\in\mathbb{N}}$ und dem beobachteten Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ herzustellen. In seiner Arbeit hat Gilbert dazu die Zustände der Markov Kette $(X_t)_{t\in\mathbb{N}}$ nicht direkt mit den beobachteten Fehlerbits des Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ identifiziert sondern mit der jeweiligen Wahrscheinlichkeit, dass ein Übertragungsfehler erfolgt, während die Markov Kette $(X_t)_{t\in\mathbb{N}}$ den betrachteten Zustand annimmt, vgl. [14]. Zur genaueren Erläuterung führen wir zunächst die folgenden Begriffe ein.

**3.3.7 Definition**
*Für den Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ eines fehlerbehafteten, digitalen Kanals und eine homogene Markov Kette $(X_t)_{t\in\mathbb{N}}$ mit endlichem Zustandsraum $\mathcal{S}$ gemäß den obigen Voraussetzungen definieren wir die Wahrscheinlichkeit für das Auftreten eines Fehlers, während sich die betrachtete Markov Kette $(X_t)_{t\in\mathbb{N}}$ in einem beliebigen Zustand $i \in \mathcal{S}$ befindet, als Fehlerwahrscheinlichkeit $e_i$ des betrachteten Zustands $i \in \mathcal{S}$, also*

$$e_i := P(E_t = 1 \mid X_t = i) \quad \textit{und} \quad e_i \in [0,1] \quad \textit{für alle } i \in \mathcal{S}.$$

**3.3.8 Definition**
*Ist eine homogene Markov Kette $(X_t)_{t\in\mathbb{N}}$ im obigen Sinne gegeben und werden die Zustände $i \in \mathcal{S}$ der Markov Kette $(X_t)_{t\in\mathbb{N}}$ mit den jeweiligen Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, identifiziert, so wird eine derartige Markov Kette als Markov'scher Hintergrundprozeß, oder in der Literatur oft auch als hidden Markov process, bezeichnet.*

Indem im Modellansatz nach Gilbert die Zustände einer Markov Kette nicht direkt mit einem Fehlerbit des Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ sondern mit der Wahrscheinlichkeit für die Beobachtung eines Fehlers identifiziert werden, wird ein Markov'scher Hintergrundprozeß induziert, dessen Zustände $i \in \mathcal{S}$ gerade die Fehlerwahrscheinlichkeiten $e_i$ mit $i \in \mathcal{S}$ implizieren. Von daher bestimmt in dem Modell von Gilbert jeweils der gegenwärtige Zustand des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ die Wahrscheinlichkeit $e_i$, $i \in \mathcal{S}$, für das Auftreten eines Übertragungsfehlers. Auf diese Weise wird zwischen dem Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ bzw. dem Übertragungsprozeß des digitalen Kanals und dem beobachtbaren Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ eine Beziehung hergestellt, was anschaulich in Abbildung 3.3 gezeigt wird.

**3.3.9 Bemerkung**
In seiner Arbeit [14] geht Gilbert speziell von einem Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ aus, der einen Zustandsraum $\mathcal{S}$ bestehend aus genau zwei Zuständen $G$ und $B$ besitzt, mit $G$ für den Übertragungszustand *good* und $B$ für den Übertragungszustand *bad*. Dabei beschreibt der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ den Übertragungsprozeß des betrachteten, digitalen Kanals insofern, als dass die Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ den jeweiligen Übertragungszustand des Kanals charakterisieren. So gibt es einen guten Übertragungszustand *good* und einen schlechten Übertragungszustand *bad*, wobei die Übertragungszustände die jeweilige Fehlerwahrscheinlichkeit $e_G$ bzw. $e_B$ implizieren. Im Prinzip stellt der Übertragungszustand *good* daher einen symmetrischen Binärkanal mit der Bitfehlerrate $p_E = e_G$ dar und der schlechte Übertragungszustand *bad* repräsentiert einen symmetrischen Binärkanal mit einer Bitfehlerrate von $p_E = e_B$.

Nun setzt Gilbert in seiner Arbeit die Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ explizit zu $e_G = 0$ und $e_B = 1/2$ und ermöglicht Zustandsübergänge vom Zustand $G$ in den Zustand $B$ und umgekehrt gemäß den gegebenen Übergangswahrscheinlichkeiten des Markov'schen Hintergrundprozesses $(X_t)_{t\in N}$, vgl. [14]. In Abbildung 3.3 ist das Gilbert Modell mit den entsprechenden Größen bzw. Parametern anschaulich dargestellt.

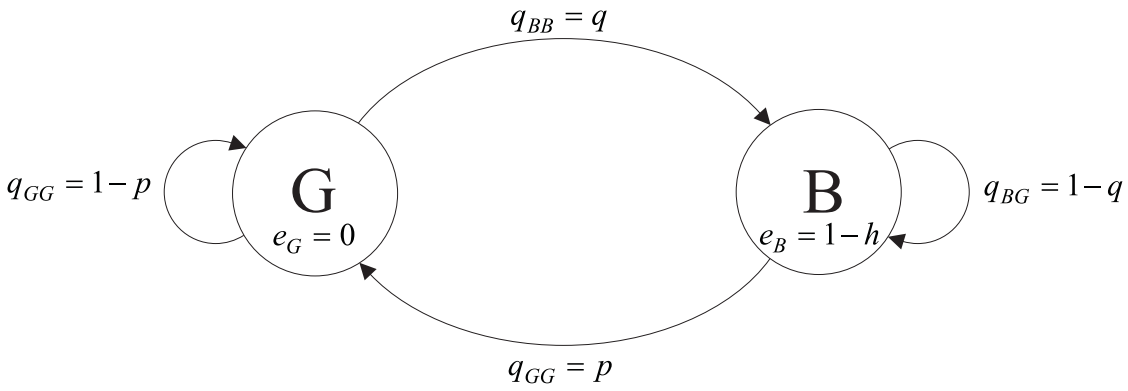


Abbildung 3.3: Das Modell von Gilbert

Nun geben die Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ gemäß Definition 3.3.7 an, wie wahrscheinlich das Auftreten eines Übertragungsfehlers in dem jeweiligen Zustand $G$ bzw. $B$ ist. Demzufolge ist die Wahrscheinlichkeit, dass im guten Übertragungszustand $G$ ein Fehler erzeugt wird, gleich 0. Der betrachtete digitale Kanal erzeugt im Zustand $G$ demnach keine Übertragungsfehler. Im Übertragungszustand *bad* dagegen können sehr wohl Fehler auftreten. Hierbei treten diese dann häufig dicht hintereinander in Form von sogenannten *Burstfehlern* auf, so dass im Zustand $B$ korrelierte Bitfehler zustandekommen können. Dabei muß allerdings nicht zwangsläufig jedes Bit verfälscht werden sondern nur ein Teil. Gilbert setzt in seiner Arbeit dazu explizit $e_B = 1/2$, so dass im Übertragungszustand $B$ ein Bit mit Wahrscheinlichkeit 1/2 verfälscht wird, vgl. [14]. Auf diese Weise liefern die konkreten Annahmen des Gilbert Modells ein realistisches Kanalmodell mit Gedächtnis, mit dem auch korrelierte Fehlerpositionen bzw. Burstfehler modelliert werden können.

Das Kanalmodell von Gilbert wird darüber hinaus durch die Übergangswahrscheinlichkeiten des Markov'schen Hintergrundprozesses $(X_t)_{t\in N}$ bestimmt, vgl. Abbildung 3.3. Gemäß den Übergangswahrscheinlichkeiten $q_{GG}$ und $q_{GB}$ sowie $q_{BG}$ und $q_{BB}$ alternieren Phasen im schlechten Übertragungszustand $B$, mit die sogenannten *Burstfehler* modelliert werden können, mit den fehlerfreien Phasen des guten Übertragungszustandes $G$. Unter der Annahme, dass die physikalischen Ursachen der Fehler in keiner Beziehung zueinander stehen, also voneinander unabhängig sind, können die Aufenthaltszeiten in den beiden Zuständen $G$ und $B$ als diskrete und gedächtnislose Zufallsvariablen betrachtet werden. Nach Abschnitt 3.1 Bemerkung 3.1.21 sind die Aufenthaltszeiten dann geometrisch verteilt. Annahme des Gilbert Modells ist demnach, dass die Länge einer schlechten Übertragungsphase geometrisch mit der mittleren Länge $1/q$ verteilt ist. Entsprechend ist die Länge der fehlerfreien Phasen geometrisch mit der mittleren Länge $1/p$ verteilt.

Da wegen $e_G = 0$ alle Bits im guten Übertragungszustand $G$ fehlerfrei übertragen werden und Fehler daher nur im schlechten Übertragungszustand $B$ auftreten können, können wir für den Markov'schen Hintergrundprozeß $(X_t)_{t\in N}$ von einer 1 im beobachteten Fehlerprozeß

$(E_t)_{t\in\mathbb{N}}$ darauf schließen, dass sich der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ im Zustand $B$ befunden hat. Beobachten wir im Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ jedoch eine 0, so können wir nicht mehr eindeutig darauf zurückschließen, ob sich der Kanal und damit der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ im Zustand $G$ oder $B$ befunden hat, denn wegen $e_B = 1/2$ werden Bits im schlechten Übertragungszustand mit Wahrscheinlichkeit $1/2$ verfälscht und eine fehlerfreie Übertragung ist auch in diesem Zustand $X_t = B$ möglich. Daher kann der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ im Gilbert Modell nicht mehr eindeutig vom beobachteten Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ aus rekonstruiert werden.

### 3.3.10 Bemerkung

Für den trivialen Fall, dass der Zustandsraum $\mathcal{S}$ des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ im Gilbert Modell nur aus genau einem Zustand $B$ bestehen würde, würde das auf diese Weise konstruierte Kanalmodell nur gedächtnislose Fehlerprozesse $(E_t)_{t\in\mathbb{N}}$ mit einer konstanten Bitfehlerrate $p_E = e_B$ abbilden. Das erhaltene Kanalmodell entspräche folglich dem symmetrischen Binärkanal aus Abschnitt 3.3.1.

Mit Blick auf die Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, führen wir den folgenden Begriff ein.

### 3.3.11 Definition

*Die Gesamtwahrscheinlichkeit für das Auftreten eines Fehlers während der Übertragung eines einzelnen Bits wird definiert durch*

$$p_E := P(E_t = 1)$$

*und als Bitfehlerwahrscheinlichkeit oder Bitfehlerrate $p_E$ bezeichnet.*

### 3.3.12 Bemerkung

Für einen Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$, der im Sinne von Definition 3.3.8 durch einen Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ bestimmt wird, dessen Zustände $i \in \mathcal{S}$ die Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, implizieren, kann die Bitfehlerrate $p_E$ insgesamt nach der Formel von Bayes berechnet werden:

$$\begin{aligned} p_E &= \sum_{i=G,B} P(E_t = 1 \,|\, X_t = i)\, P(X_t = i) \\ &= e_G\, \pi_G + e_B\, \pi_B \end{aligned}$$

Insbesondere für das Modell von Gilbert mit $e_G = 0$ und $e_B = 1/2$ bedeutet dies, dass sich die Bitfehlerrate $p_E$ insgesamt ergibt zu

$$\begin{aligned} p_E &= e_G\, \pi_G + e_B\, \pi_B \\ &= 0 \cdot \pi_G + \tfrac{1}{2}\pi_B \quad = \quad \tfrac{1}{2}\pi_B \end{aligned}$$

Mit dieser Arbeit hat Gilbert ein einfaches Konzept präsentiert, um korrelierte Fehlerprozesse abzubilden und damit fehlerbehaftete, digitale Kanäle mit einer einfachen Form von Gedächtnis zu modellieren.

**3.3.13 Bemerkung**
Die Konzeption dieses Modells liefert zwar die Möglichkeit, das Gedächtnis realer Kanäle in einfacher Form darzustellen, aber auch hier sind die Möglichkeiten, reale Kanäle abzubilden, begrenzt. Hauptursache dieser Beschränkung ist der Erneuerungscharakter des Modells. Beobachten wir eine 1 im Fehlerprozeß, so wissen wir, dass diese ausschließlich im Zustand $B$ des Markov'schen Hintergrundprozesses erzeugt werden kann, und können eindeutig auf den Zustand $B$ des Kanals zurückschließen. Der Übertragungsprozeß startet nach dem Auftreten eines Fehlers neu und vergisst die Zustände, die er zuvor durchlaufen hat.

Darüber hinaus ist kritisch zu beleuchten, ob als Verteilung für die Längen der Aufenthaltszeiten im guten bzw. schlechten Übertragungszustand die geometrische Verteilung eine geeignete Verteilung ist und ob nicht auch andere Verteilungen zugelassen und berücksichtigt werden müssen, vgl. hierzu insbesondere Abschnitt 5.2.2.

In den folgenden Paragraphen stellen wir verschiedene Modellerweiterungen des Modells von Gilbert vor. In der Literatur sind diesbezüglich unterschiedliche Ansätze gewählt und verfolgt worden, denn die Modellierung mittels einer Markov Kette ermöglicht eine Vielzahl von Variationen. So reichen die Verallgemeinerungen des Modells von Gilbert ausgehend von einem Modell mit weiteren Modellparametern über Modelle, deren Zustandsübergänge zwischen den Zuständen der Markov Kette mit bestimmten Bedingungen verknüpft sind, bis hin zu Modellen mit einem erweiterten Zustandsraum $\mathcal{S}$ des Markov'schen Hintergrundprozesses. So werden hier endlich viele oder sogar abzählbar unendlich viele Zustände zugelassen. Eine zweckmäßige Charakterisierung eines Übertragungskanals sollte beides berücksichtigen, eine möglichst große Genauigkeit auf der einen und die Einfachheit bzw. Überschaubarkeit des Modells und seiner Parameter auf der anderen Seite.

### 3.3.3 Die Erweiterung nach Elliott

Eine naheliegende Verallgemeinerung des Modells von Gilbert geht zurück auf eine Arbeit von E. O. Elliott aus dem Jahr 1963, siehe [10]. Mit der Einführung zweier weiterer Modellparameter $h$ und $k$ präsentiert Elliott eine simple Modellerweiterung und erhöht so die Flexibilität des Gilbert Modells. Aufgrund der beiden zusätzlichen Parameter können nun auch im guten Übertragungszustand $G$ Übertragungsfehler modelliert werden, so dass der Erneuerungscharakter des Modells von Gilbert hiermit aufgehoben wird.

Die Voraussetzungen, von denen Elliott in seinem Modellansatz ausgeht, sind mit Ausnahme der Werte für die beiden Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ dieselben wie im Modell von Gilbert.

Ein Markov'scher Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ mit endlichem Zustandsraum $\mathcal{S} = \{G, B\}$ bestehend aus einem guten und einem schlechten Übertragungszustand bedingt indirekt den beobachteten Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$. Dabei wird der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ durch die Übergangswahrscheinlichkeiten

$$\begin{aligned} & q_{GG} = 1-p \quad , \quad q_{GB} = p \\ \text{und} \quad & q_{BG} = q \quad , \quad q_{BB} = 1-q \end{aligned}$$

gekennzeichnet, wobei $p, q \in [0,1]$ sind. Ferner sind die stationären Zustandswahrscheinlichkeiten wiederum festgelegt durch die Gleichungen

$$\pi_G = \frac{q}{p+q} \qquad \text{und} \qquad \pi_B = \frac{p}{p+q}.$$

Als wesentlichen Schritt in Richtung einer Erweiterung des Gilbert Modells führt Elliott in seiner Arbeit die beiden zusätzlichen Modellparameter $h$ und $k$ ein, vgl. [10]. Indem Elliott den Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ keine konkreten Werte sondern jeweils einen dieser beiden Parameter zuweist, kann die Flexibilität des ursprünglichen Modells von Gilbert gesteigert werden. Auf diese Weise können nun nicht nur im schlechten Übertragungszustand $B$ sondern auch im guten Zustand $G$ Übertragungsfehler erzeugt werden, wenn auch in geringerem Maße, abhängig von dem Wert des Parameters $k$. Konkret setzt Elliott die Fehlerwahrscheinlichkeiten daher

$$e_G = k \quad \text{und} \quad e_B = 1 - h,$$

wobei $h, k \in ]0,1]$ sind und $k \ll 1$ ist.

### 3.3.14 Bemerkung

In Abbildung 3.4 ist das Modell von Elliott mit den vom Gilbert Modell abweichenden Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ dargestellt. Durch die Erweiterung des Modells um die Parameter $h$ und $k$ gelingt es Elliott, den Erneuerungscharakter des ursprünglichen Modells von Gilbert aufzuheben. Somit können Übergänge zwischen großen Zeiträumen im guten Übertragungszustand mit geringer Fehlerwahrscheinlichkeit $e_G$ und großen Zeiträumen im schlechten Zustand mit hoher Fehlerwahrscheinlichkeit $e_B$ abgebildet werden. Dabei erfolgen die Zustandsübergänge gemäß den Übergangswahrscheinlichkeiten des Markov'schen Hintergrundprozesses $(X_t)_{t \in \mathbb{N}}$. Von daher schafft Elliott mit seinem verallgemeinerten Ansatz die Möglichkeit, Fehlermuster zu modellieren, die sowohl einzelne Übertragungsfehler als auch solche in zeitlicher Nähe, also *Burstfehler* enthalten.

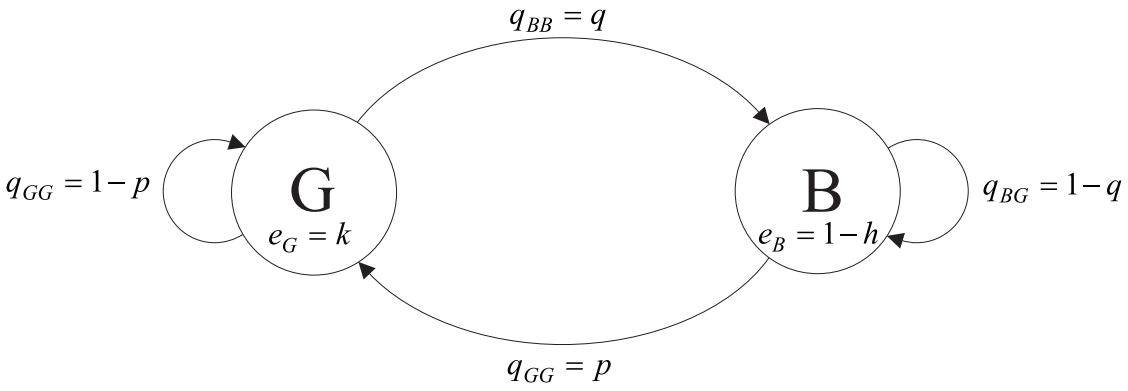


Abbildung 3.4: Die Modellerweiterung von Elliott

Da wir aufgrund der zusätzlichen Modellparameter bei der Beobachtung einer 1 im Fehlerprozeß $(E_t)_{t \in \mathbb{N}}$ nicht mehr entscheiden können, ob sich der Kanal im guten oder im schlechten Übertragungszustand befunden hat, ist der Markov'sche Hintergrundprozeß $(X_t)_{t \in \mathbb{N}}$ vom beobachtbaren Fehlerprozeß $(E_t)_{t \in \mathbb{N}}$ aus nicht mehr rekonstruierbar.

**3.3.15 Bemerkung**
Basierend auf dem Modell von Elliott berechnet sich die Bitfehlerwahrscheinlichkeit $p_E$ daher insgesamt zu:

$$p_E = e_G\,\pi_G + e_B\,\pi_B = k\,\pi_G + (1-h)\pi_B$$

**3.3.16 Bemerkung**
Obgleich das Konzept des Ansatzes von Elliott zumindest den Erneuerungscharakter des Gilbert Modells aufhebt, ist zu untersuchen, ob es reale Kanäle genügend genau abbildet. Dazu ist zu analysieren, ob die geometrische Verteilung zur Darstellung der Aufenthaltszeiten geeignet ist oder ob nicht auch andere Verteilungen modelliert werden müssen. Hier haben Untersuchungen von Berger und Mandelbrot gezeigt, dass die geometrische Verteilung für die Darstellung der Längen der Aufenthaltszeiten oft ungeeignet ist, vgl. [3].

### 3.3.4 Die Modellerweiterung nach McCullough

Um das Phänomen, dass Übertragungsfehler sowohl unabhängig voneinander als auch in zeitlicher Nähe auftreten, adäquat mittels des Kanalmodells reproduzieren zu können, wählt McCullough ein Markov Modell ähnlich dem des Gilbert Modells, vgl. [35]. Allerdings unterscheidet sich das Modell von McCullough in einem wesentlichen Aspekt vom Gilbert Modell. So können im Modell nach McCullough Zustandsübergänge nur nach dem Auftreten eines Übertragungsfehlers stattfinden. Von daher hängen die Übergangswahrscheinlichkeiten zwischen den Zuständen des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ vom Auftreten eines Fehlers ab und sind im Gegensatz zum Gilbert Modell nicht mehr unabhängig davon. Bei der folgenden Präsentation des Ansatzes von McCullough bezeichnen wir die Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ mit 1 und 2, um Verwechselungen mit dem Gilbert Modell zu vermeiden, vgl. [35].

Als Grundlage verwendet McCullough einen Markov'schen Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ mit dem Zustandsraum $\mathcal{S} = \{1, 2\}$, dessen Zustände 1 und 2 jeweils die Fehlerwahrscheinlichkeiten $e_1$ und $e_2$ implizieren. Da $e_1, e_2 \in ]0, 1]$ sind, stellt jeder der beiden Zustände einen symmetrischen Binärkanal mit der Fehlerwahrscheinlichkeit $e_1$ bzw. $e_2$ dar. Demzufolge werden in beiden Zuständen Übertragungsfehler generiert, ähnlich wie im Modell von Elliott. McCullough erlaubt in seinem Modellansatz einen Zustandsübergang vom Zustand 1 in den Zustand 2 bzw. umgekehrt allerdings nur nach dem Auftreten eines Fehlers, so dass die Übergangswahrscheinlichkeiten des Markov'schen Hintergrundprozesses an eine Bedingung geknüpft werden. Hierzu führen wir zunächst die folgenden Bezeichnungen ein.

**3.3.17 Definition**
*Gegeben sei ein Markov'scher Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ mit endlichem Zustandsraum $\mathcal{S} = \{1, 2\}$, dann werden die Übergangswahrscheinlichkeiten des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ mit $q_{ij}^{(0)}$ und $q_{ij}^{(1)}$ für $i, j \in \mathcal{S}$ bezeichnet je nachdem, ob zuvor während der Übertragung ein Fehler erfolgt ist, oder nicht. Für alle $i, j \in \mathcal{S}$ gilt demnach*

$$\begin{aligned} q_{ij}^{(0)} &:= P(X_{t+1} = j \,|\, X_t = i\,,\, E_t = 0) \\ \textit{und} \qquad q_{ij}^{(1)} &:= P(X_{t+1} = j \,|\, X_t = i\,,\, E_t = 1)\,. \end{aligned}$$

**3.3.18 Bemerkung**
In Abbildung 3.5 ist das Zustandsdiagramm für einen Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ mit Übergangswahrscheinlichkeiten im Sinne der vorangehenden Definition 3.3.17 dargestellt. Dabei gilt für die Übergangswahrscheinlichkeiten

$$q_{ij}^{(0)}, q_{ij}^{(1)} \in [0,1] \qquad \text{für alle } i,j \in \mathcal{S}\,.$$

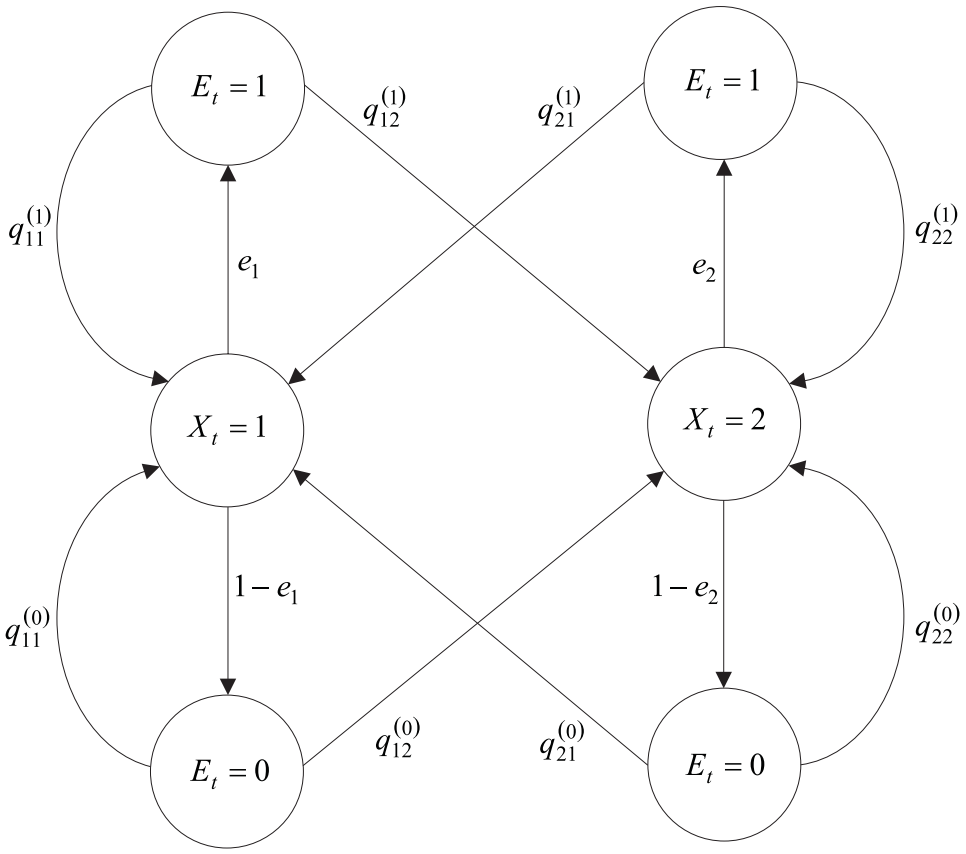


Abbildung 3.5: Das Modell von McCullough mit allgemeinen Parametern ?

Das auf diese Art induzierte Modell beschreibt einen Erneuerungsprozeß, sobald die Übergangswahrscheinlichkeiten $q_{ij}^{(0)}$ und $q_{ij}^{(1)}$, $i,j \in \mathcal{S}$, unabhängig vom Zustand $i$ sind. In diesem Fall ist der nächste Zustand stets unabhängig von dem Zustand, in dem ein Fehler produziert worden ist. Ferner gilt, falls eine der beiden Fehlerwahrscheinlichkeiten $e_1$ bzw. $e_2$ identisch null ist, dass nur in einem Zustand Fehler erzeugt werden und das Modell auf diese Weise Erneuerungscharakter erhält.

**3.3.19 Bemerkung**
Abbildung 3.6 präsentiert schließlich anschaulich das Zustandsdiagramm des Modells von McCullough. Im Gegensatz zu Abbildung 3.5 sind in dieser Abbildung die Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ nicht getrennt vom beobachtbaren Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ dargestellt. Ferner gilt mit Blick auf die Übergangswahrscheinlichkeiten für $i,j \in \mathcal{S}$

$$q_{ij}^{(0)} \;=\; \delta_{i,j}$$

$$\text{und } q_{ij}^{(1)} \quad = \quad q_{ij} \,,$$

wobei $\delta_{i,j}$ das Kronecker-Symbol bezeichnet. Auf diese Weise wird erreicht, dass ein Zustandsübergang ausschließlich nach dem Auftreten eines Fehlers erlaubt ist.

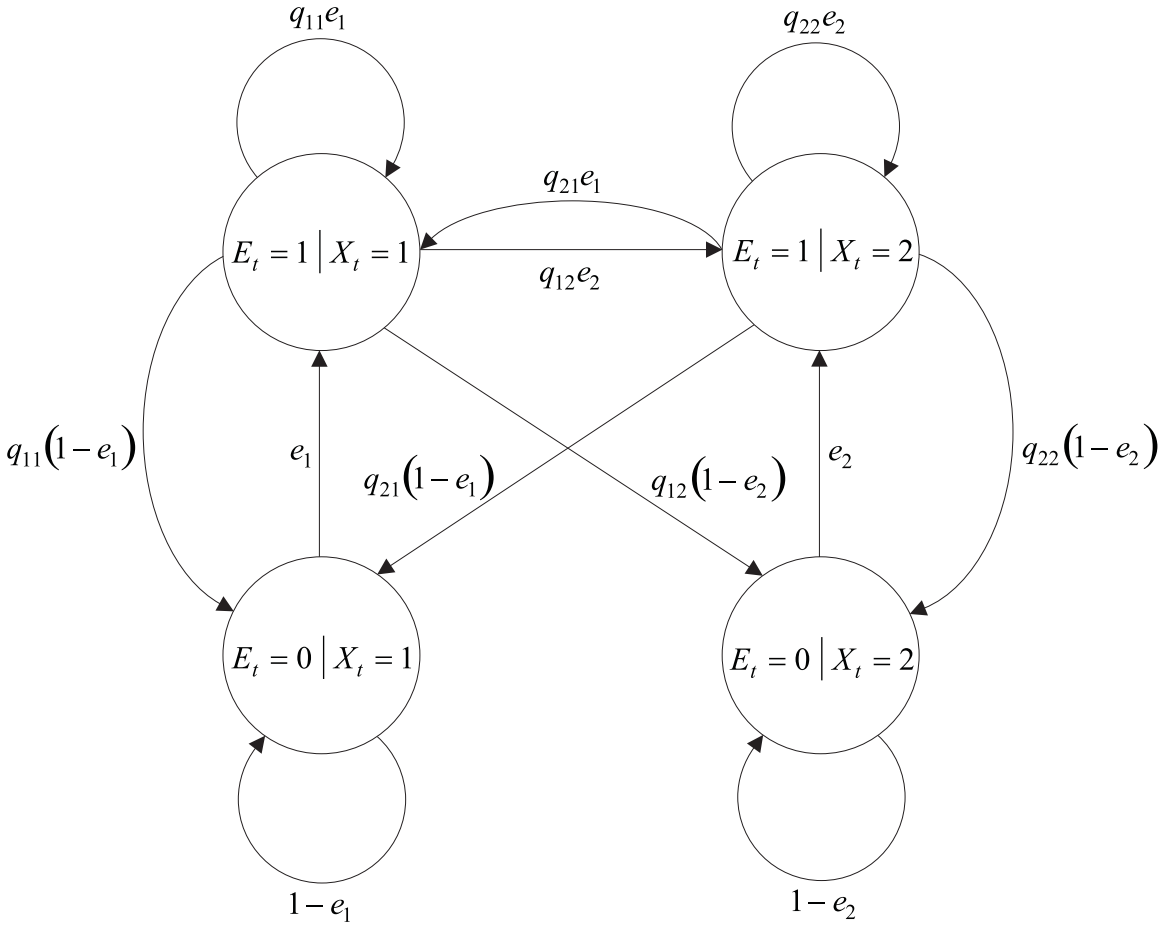


Abbildung 3.6: Das Modell von McCullough

Die Einschränkung, dass ein Zustandsübergang nur nach dem Auftreten eines Fehlers erlaubt ist, erscheint aus zwei Gründen sinnvoll. Erstens können wir nur den Fehlerprozeß $(E_t)_{t\in N}$ nicht aber den Markov'schen Hintergrundprozeß $(X_t)_{t\in N}$ beobachten. Folglich können wir unter dieser Einschränkung davon ausgehen, dass sich zwischen zwei Fehlern kein Zustandswechsel ereignet. Zweitens lassen sich auf diese Art einzelne, zufällig verteilte Fehler und korrelierte Fehler gleichzeitig modellieren. Da in der Regel gilt, dass eine Fehlerwahrscheinlichkeit wesentlich kleiner ist als die andere, beispielsweise

$$e_1 \ll e_2 \approx \frac{1}{2} \,,$$

erzeugt der eine Zustand hauptsächlich zufällig verteilte Fehler und der andere dagegen eher korrelierte Fehler, also *Burstfehler*.

Darüber hinaus beschreibt das Modell von McCullough genau dann einen Erneuerungsprozeß, wenn für die Übergangswahrscheinlichkeiten $q_{11}$ und $q_{21}$ aus Abbildung 3.6 Gleichheit gilt.

**3.3.20 Bemerkung**
Im übrigen sind die Modelle von Gilbert und Elliott Spezialfälle des Modells von McCullough.

- Falls für alle $i, j \in \{1, 2\}$ gilt

$$q_{ij}^{(0)} = q_{ij}^{(1)} \quad \text{und} \quad e_1 \in ]0, 1[\,,\ e_2 = 0$$

  erhalten wir das Modell von Gilbert.

- Für die Einschränkungen

$$q_{ij}^{(0)} = q_{ij}^{(1)} \text{ für alle } i, j \in \{1, 2\} \quad \text{und} \quad e_1 \in ]0, 1[\,,\ e_2 \neq 0$$

  ergibt sich das Modell von Elliott als Spezialfall.

Obgleich die Darstellungen der bislang vorgestellten Modelle in wesentlichen Punkten verschieden sind, sind die mathematischen Berechnungen etwa der stationären Zustandswahrscheinlichkeiten und der Bitfehlerrate $p_E$ vom Prinzip her identisch. Zu berücksichtigen sind dabei jeweils lediglich die modifizierten Übergangswahrscheinlichkeiten $q_{ij}^{(0)}$, $q_{ij}^{(1)}$ für alle $i, j \in \{1, 2\}$ im Modell von McCullough.

### 3.3.5 Eine Verallgemeinerung von Fritchman

Obschon sämtliche der bislang in diesem Abschnitt vorgestellten Kanalmodelle aufgrund der Markov Eigenschaft mathematisch einfach zu handhaben sind und ihre Modellparameter noch einen gewissen Anschauungswert besitzen, sind sie nicht genügend flexibel, um für eine breite Klasse realer Übertragungskanäle eine geeignete und zweckdienliche Form der Modellierung zu liefern. Hierdurch motiviert hat B. D. Fritchman 1966 in seiner Arbeit eine weiterreichende Verallgemeinerung dieser Modelle entwickelt, siehe hierzu [12].

Dabei sind die Voraussetzungen für das Modell von Fritchman ähnlich denen des Gilbert Modells. Als Basis fungiert zur Modellierung des Fehlerprozesses $(E_t)_{t \in \boldsymbol{N}}$ auch hier ein Markov'scher Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$, der gekennzeichnet ist durch seine Übergangswahrscheinlichkeiten

$$q_{ij} = P(X_{t+1} = j \,|\, X_t = i) \quad \text{für alle } i, j \in \mathcal{S}.$$

Die stationären Zustandswahrscheinlichkeiten $\pi_i$ für jedes $i \in \mathcal{S}$ ergeben sich wieder analog aus dem bekannten Gleichungssystem und der Normalisierungsbedingung.

**3.3.21 Bemerkung**
Das Prinzip der Verallgemeinerung nach Fritchman stützt sich auf die folgenden beiden Aspekte, erstens eine Vergrößerung des Zustandsraumes $\mathcal{S}$ und zweitens eine abgewandelte inhaltliche Interpretation der Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t \in \boldsymbol{N}}$. So besitzt der Markov'sche Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ im Fritchman Modell an Stelle eines Zustandsraumes $\mathcal{S}$ bestehend aus nur 2 Zuständen einen Zustandsraum $\mathcal{S} = \{1, \ldots, N\}$

bestehend aus $N$ Zuständen, mit $N \geq 2$. Allerdings werden die Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ nicht mehr mit den jeweiligen Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, identifiziert. Das Konzept des Fritchman Modells beruht dagegen darauf, den Zustandsraum $\mathcal{S} = \{1, \ldots, N\}$ in zwei Mengen aufzuteilen, vgl. [12]. Dabei umfasst die erste Menge $k$ fehlerfreie Zustände und die zweite Menge enthält $N - k$ fehlerhafte Zustände. Demnach beinhaltet die erste Menge diejenigen Zustände, in denen niemals ein Übertragungsfehler auftritt, und die zweite wird ausschließlich aus Zuständen gebildet, die ständig Fehler produzieren. In Abbildung 3.7 ist dies anschaulich dargestellt.

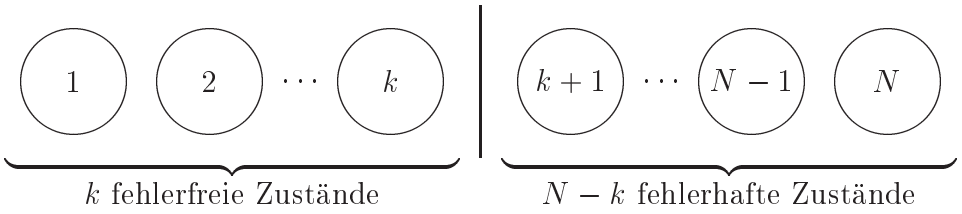


Abbildung 3.7: Das Modell von Fritchman: Aufteilung des Zustandsraumes

Mit Bezug auf das Gilbert Modell können wir dieses Vorgehen wie folgt interpretieren. Da wegen $e_B = 1/2$ ein Bit im Übertragungszustand $B$ mit Wahrscheinlichkeit $1/2$ verfälscht wird, wird dieser Zustand $B$ in einen Zustand $B_0$ und einen Zustand $B_1$ aufgespalten. Der Zustand $B_0$ ist dabei derart charakterisiert, dass in diesem Zustand keine Übertragungsfehler erfolgen. Dagegen ist der Zustand $B_1$ als fehlerhafter Zustand gekennzeichnet, der fortwährend Fehler produziert. Demzufolge erhalten wir ein Markov Modell mit drei Zuständen, wobei zwei davon fehlerfreie Zustände sind und ein Zustand ständig Fehler erzeugt. Diese Modifikation des Gilbert Modells hat Fritchman als Ausgangspunkt und Motivation für die Entwicklung seines Ansatzes in seiner Arbeit [12] gedient.

Ausgehend von der Aufteilung des Zustandsraumes in fehlerfreie und fehlerhafte Zustände und der einhergehenden inhaltlichen Interpretation der Zustände gestaltet sich die Verknüpfung des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ mit dem beobachtbaren Fehlerprozeß $(E_t)_{t\in\boldsymbol{N}}$ wie folgt. Befindet sich der Markov'sche Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ in einem fehlerfreien Zustand aus der Menge $\{1, \ldots, k\}$, so liefert dieser eine Null im Fehlerprozeß $(E_t)_{t\in\boldsymbol{N}}$. Unter der Bedingung, dass

$$X_t = i \quad \text{mit} \quad i \in \{1, \ldots, k\}$$

gilt, besitzt das zugehörige Fehlerbit $E_t$ den Wert 0. Nimmt dagegen der Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ einen Zustand aus der Menge der fehlerhaften Zustände $\{k + 1, \ldots, N\}$ an, so erhalten wir unter der Bedingung, dass

$$X_t = i \quad \text{mit} \quad i \in \{k + 1, \ldots, N\}$$

ist, entsprechend $E_t = 1$.

Gemäß der Notation, die in den vorangegangenen Modellen verwendet worden ist, ist die Fehlerwahrscheinlichkeit $e_i$ für alle Zustände aus der Menge $\{1, \ldots, k\}$ gegeben durch

$$e_i = P(E_t = 1 \,|\, X_t = i) = 0 \qquad \text{für alle } i \in \{1, \ldots, k\}\,.$$

Als Fehlerwahrscheinlichkeit für alle übrigen Zustände erhalten wir

$$e_i = P(E_t = 1 \,|\, X_t = i) = 1 \qquad \text{für alle } i \in \{k+1, \ldots, N\}.$$

Mit der Vergrößerung des Zustandsraumes $\mathcal{S}$ werden die Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, in diesem Zuge auf die Werte 0 und 1 reduziert. Folglich symbolisieren die Zustände im Modellansatz von Fritchman das unmögliche bzw. das sichere Auftreten eines Fehlers, womit die Aufteilung des Zustandsraumes $\mathcal{S}$ in die Menge der fehlerfreien und die Menge der fehlerhaften Zustände einhergeht. Auf diese Weise ist eine breitere Klasse realer Kanäle abbildbar, wobei die mathematische Einfachheit des Modells aufgrund des Markov Charakters erhalten bleibt.

Darüber hinaus liefert die Arbeit von Fritchman Ergebnisse im Hinblick auf die Verteilung der Längen der fehlerfreien und der fehlerhaften Phasen während der Übertragung. In [12] entwickelt Fritchman eine allgemeine Formel für die Verteilung der Längen der fehlerfreien Phasen sowie für die Verteilung der Längen der fehlerhaften Phasen. Zur Herleitung wird der betrachtete Markov'sche Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ insofern modifiziert, als dass Zustandsübergänge nur noch zwischen den fehlerfreien Zuständen der Menge $\{1, \ldots, k\}$ zugelassen sind, wie auch von der Menge $\{1, \ldots, k\}$ der fehlerfreien Zustände in die Menge $\{k+1, \ldots, N\}$ der fehlerhaften Zustände. Zustandsübergänge von der Menge $\{k+1, \ldots, N\}$ der fehlerhaften Zustände in die Menge $\{1, \ldots, k\}$ der fehlerfreien Zustände werden demnach unmöglich. Innerhalb der Menge $\{k+1, \ldots, N\}$ der fehlerhaften Zustände sind lediglich Übergänge von dem erreichten Zustand in sich selbst erlaubt, so dass diese Zustände sämtlich absorbierende Fehlerzustände darstellen, was in Bild 3.8 veranschaulicht ist.

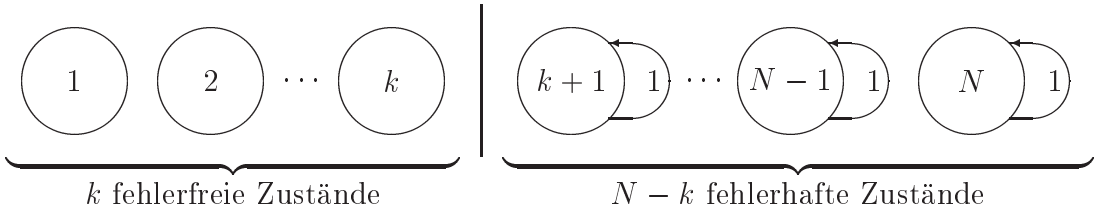


Abbildung 3.8: Das Modell von Fritchman mit absorbierenden Fehlerzuständen

Als Übergangsmatrix dieses modifizierten Prozesses $(X_t)_{t \in \boldsymbol{N}}$ ergibt sich daher

$$\mathbf{Q} = \left(\begin{array}{ccc|ccc} q_{11} & \cdots & q_{1k} & q_{1k+1} & \cdots & q_{1N} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ q_{k1} & \cdots & q_{kk} & q_{kk+1} & \cdots & q_{kN} \\ \hline & & & 1 & & 0 \\ & 0 & & & \ddots & \\ & & & 0 & & 1 \end{array}\right)$$

### 3.3.22 Bemerkung

Aus dem Konzept dieses modifizierten Prozesses ergibt sich ein wesentlicher Nachteil zur

Modellierung von Fehlern in zeitlicher Nähe. Da die fehlerhaften Zustände absorbierende Zustände darstellen, ist eine Terminierung von sogenannten Bündelfehlern nicht möglich. Der Prozeß verbleibt in diesem Zustand. Demzufolge bildet auch dieses Modell keine geeignete Grundlage zur Bewertung von Codierverfahren in unserem Sinne.

Mittels dieses modifizierten Prozesses lassen sich zwar allgemeine Formeln für die Verteilungen der Längen der fehlerfreien und der fehlerhaften Phasen ableiten, aber Fritchman macht in diesem Zusammenhang in seiner Arbeit deutlich, dass dieses allgemeine Kanalmodell zur Bestimmung der freien Parameter zu komplex ist, vgl. [12]. Daneben ist der mangelnde Anschauungswert der Modellparameter problematisch.

Für den Spezialfall, dass das Modell genau einen anstatt mehrere Fehlerzustände besitzt, ist die Verteilung der fehlerfreien Phasen zur Charakterisierung des Übertragungsprozesses ausreichend. Diese Vereinfachung des Fritchman Modells ist in einer Arbeit von S. Tsai wieder aufgenommen worden und auch als das Modell von Tsai bekannt.

In der Arbeit von Fritchman zeigt sich in jedem Fall, dass derartige Modellerweiterungen immer komplexer und mathematisch schwerer handhabbar werden, je mehr ein Kanalmodell einem realen Übertragungskanal entspricht.

### 3.3.6 Die Modellvereinfachung von Tsai

Da die Verteilung der Längen der fehlerfreien und der fehlerhaften Phasen eine äußerst komplexe Funktion ist, hat Fritchman in seiner Arbeit bereits eine Modellvereinfachung erwogen, in der nur ein einziger Fehlerzustand zugelassen ist, siehe [12]. Diese Anregung ist 1969 von S. Tsai wieder aufgegriffen worden und in der Literatur auch als das Modell von Tsai bekannt, vgl. hierzu [50].

In seinem Artikel von 1969 nimmt Tsai dieselben Voraussetzungen wie Fritchman in seinem Modellansatz an, vgl. [50]. So liegt den folgenden Überlegungen ein Markov'scher Hintergrundprozeß $(X_t)_{t\in N}$ zugrunde, der einen endlichen Zustandsraum $\mathcal{S}$ bestehend aus $N$ Zuständen besitzt, $N \geq 2$, und durch die Übergangswahrscheinlichkeiten $q_{ij}$ für alle $i, j \in \mathcal{S}$ gekennzeichnet ist. Dem Modell von Fritchman folgend teilt Tsai den Zustandsraum $\mathcal{S}$ ebenfalls in zwei Mengen auf, wobei die eine die fehlerfreien und die andere Menge die fehlerhaften Zuständen umfasst. Die Zustände sind ebenfalls wieder mit den Fehlerwahrscheinlichkeiten 0 und 1 belegt und symbolisieren somit nur das sichere und das unmögliche Auftreten eines Fehlers. Der wesentliche Unterschied zum Modell von Fritchman besteht in der Vereinfachung, dass Tsai in seiner Arbeit lediglich eine ein-elementige Menge von Fehlerzuständen betrachtet, vgl. [50]. Es existiert also nur ein einziger Zustand, der fortwährend Fehler erzeugt und der in Anlehnung an die Notation von Tsai im folgenden mit 0 bezeichnet wird.

#### 3.3.23 Bemerkung

Ferner ist das Modell von Fritchman derart zu modifizieren, dass Zustandsübergänge zwischen den fehlerfreien Zuständen nicht erlaubt sind. Möglich sind dann ausschließlich Zustandsübergänge von der Menge $\{1, \ldots, N-1\}$ der fehlerfreien Zustände in den Fehlerzustand 0 bzw. umgekehrt, vom Fehlerzustand 0 in einen der fehlerfreien Zustände sowie von

einem fehlerfreien Zustand in sich selbst. Die Übergangsmatrix $\mathbf{Q}$ besitzt dann die folgende einfache Gestalt

$$\mathbf{Q} = \left(\begin{array}{c|ccccc} q_{00} & q_{01} & q_{02} & \cdots & & q_{0(N-1)} \\ \hline q_{10} & q_{11} & & & & 0 \\ q_{20} & & q_{22} & & & \\ \vdots & & & \ddots & & \\ q_{(N-1)0} & 0 & & & & q_{(N-1)(N-1)} \end{array}\right)$$

und das Zustandsdiagramm dieses vereinfachten Kanalmodells nach Tsai kann wie in Abbildung 3.9 veranschaulicht werden.

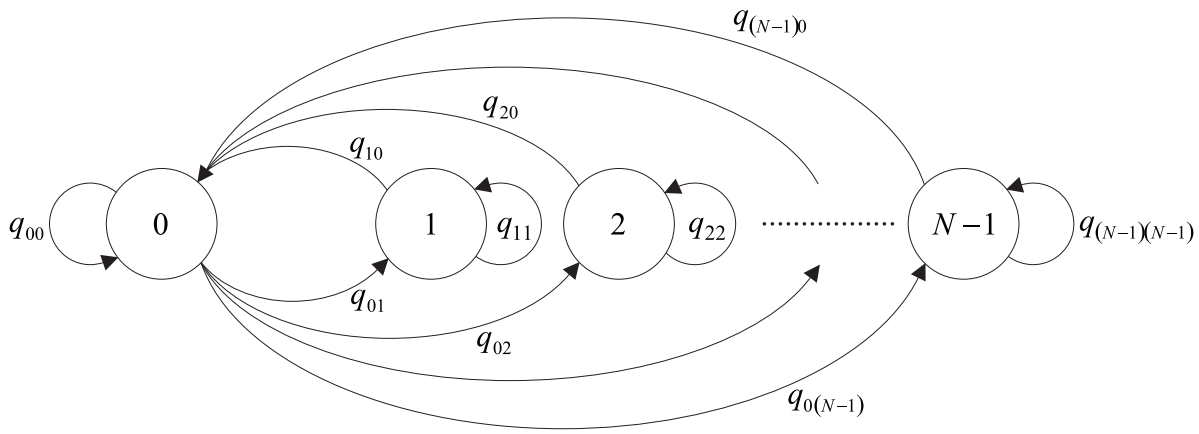


Abbildung 3.9: Das Modell von Tsai

Ein Vorteil dieses vereinfachten Modells ist, dass sich eine geschlossene Darstellung der Verteilung der fehlerfreien Phasen ermitteln läßt. Da das Modell nur einen einzigen Fehlerzustand besitzt, verringert sich die Anzahl der unabhängigen Modellparameter auf $2(N-1)$, zu deren Bestimmung die Verteilung der fehlerfreien Phasen herangezogen werden kann. Dabei legt diese Verteilung die Modellparameter eindeutig fest und läßt sich im übrigen relativ leicht aus beobachtetem Datenmaterial gewinnen. Aufgrund der Einfachheit des Modells gibt Tsai darüber hinaus in seiner Arbeit eine Methode an, mit der nicht nur die Verteilung der fehlerfreien Phasen berechnet werden kann, sondern auch die Verteilung der Burstfehler, also die Verteilung der Länge der fehlerhaften Phasen, in geschlossener Form dargestellt werden kann.

Ein wesentlicher Nachteil des Modells nach Tsai ist, dass wir mit der Reduzierung auf nur einen einzigen Fehlerzustand einen Erneuerungsprozeß erhalten. Der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ liefert in den Zuständen der Menge $\{1,\ldots,N-1\}$ stets eine 0 im beobachtbaren Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$. Demgegenüber liefert der Fehlerzustand 0 des Markov'schen Hintergrundprozesses eine 1 im Fehlerprozeß. Da es in diesem Modellansatz nur genau einen Fehlerzustand gibt, analysieren wir letztendlich ein Modell, für das die fehlerfreien Phasen unabhängig voneinander sind und somit durch gedächtnislose diskrete Zufallsvariablen charakterisiert werden können. Wegen des so entstehenden Erneuerungscharakters repräsentiert auch dieser Modellansatz noch keine genügend gute Beschreibungsform realer Übertragungskanäle.

### 3.3.7 Eine Verallgemeinerung nach Kemeny

Die Vergrößerung des Zustandsraumes $\mathcal{S}$ von zwei auf $N$ Zustände, mit $N \geq 2$, im Modell von Fritchman legt eine weitere Vergrößerung des Zustandsraumes nahe. So beruht eine weitere Verallgemeinerung des Gilbert Modells, die auf J. G. Kemeny zurückgeht, auf der Erweiterung des Zustandsraumes $\mathcal{S}$ auf abzählbar unendlich viele Zustände. Die Erweiterung des Zustandsraumes $\mathcal{S}$ auf abzählbar unendlich viele Zustände bringt einige Schwierigkeiten mit sich. ist die Berechnung grundlegender Wahrscheinlichkeitsgrößen, die notwendig sind, um zu entscheiden, ob eine Markov Kette mit abzählbar unendlichem Zustandsraum null bzw. positiv rekurrent oder transient ist, äußerst komplex. Hierbei ist zu bedenken, dass positive Rekurrenz notwendig für die Existenz der stationären Zustandswahrscheinlichkeiten ist. Im Jahre 1966 sind von J. G. Kemeny einige Arbeiten zur Theorie der abzählbar unendlichen Markov Ketten veröffentlicht worden, vgl. [22], in denen er eine große Klasse von Beispielen angegeben hat sowie einige dieser fundamentalen Größen berechnet hat, vgl. dazu [22]. Die Arbeiten von Kemeny enthalten damit einige wesentliche Ergebnisse, die die Entwicklung eines allgemeinen Markov Modells mit abzählbar unendlichem Zustandsraum erst ermöglichen.

Im Prinzip basiert ein Markov Modell mit einem abzählbar unendlichem Zustandsraum $\mathcal{S}$ auf den gleichen Voraussetzungen wie die Modelle von Fritchman und Tsai. Ein Markov'scher Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$, der durch seine Übergangswahrscheinlichkeiten

$$q_{ij} = P(X_{t+1} = j \mid X_t = i) \quad \text{für alle } i, j \in \mathcal{S}$$

gekennzeichnet ist, bildet die Grundlage des Kanalmodells nach Kemeny. Im Gegensatz zum Modell von Tsai ist der Zustandsraum $\mathcal{S}$ in diesem Fall abzählbar unendlich und kann somit beispielsweise mit den natürlichen Zahlen $\mathbb{N}$ identifiziert werden. Darüber hinaus wird gemäß dem Modell von Tsai vorausgesetzt, dass es genau einen Fehlerzustand gibt, der fortwährend Fehler generiert, wobei dieser Zustand im folgenden wiederum mit 0 bezeichnet wird.

#### 3.3.24 Bemerkung

In seinen Arbeiten befaßt Kemeny sich insbesondere mit sogenannten *sich langsam ausdehnenden Markov Ketten*. Diese sind dadurch gekennzeichnet, dass Zustandsübergänge von einem beliebigen Zustand $i \in \mathcal{S}$ nur in den Zustand $i+1$ oder in einen Zustand $j$ mit $j \in \{1, \ldots, i-1\}$ für jedes $i \in \mathcal{S}$ möglich sind. Im Hinblick auf das Modell von Tsai werden die Zustandsübergänge dahingehend weiter eingeschränkt, als dass von einem fehlerfreien Zustand $i$ nur der Fehlerzustand 0 oder nach der Bedingung von Kemeny der Zustand $i+1$ erreicht werden können. Das zugehörige Zustandsdiagramm eines solchen Markov Modells mit abzählbar unendlichem Zustandsraum $\mathcal{S}$ ist in Abbildung 3.10 dargestellt.

Die Verknüpfung des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ mit dem beobachtbaren Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ erfolgt analog zum Fritchman Modell. Befindet sich der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ in einem fehlerfreien Zustand aus der Menge $\mathbb{N} \setminus \{0\}$, so liefert dieser eine Null im Fehlerprozeß. Unter der Bedingung, dass

$$X_t = i \quad \text{mit} \quad i \in \mathbb{N} \setminus \{0\}$$

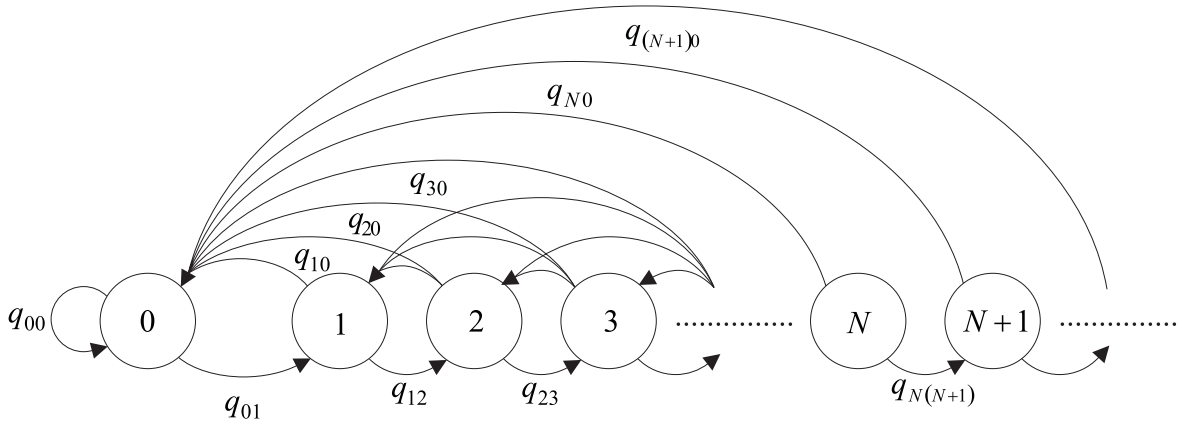


Abbildung 3.10: Das Modell von Kemeny

gilt, ist das zugehörige Fehlerbit $E_t = 0$. Nimmt dagegen der Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ den Zustand 0 an, so erhalten wir entsprechend $E_t = 1$. Die Zustände $i \in \mathbb{N} \setminus \{0\}$ besitzen damit die Fehlerwahrscheinlichkeit $e_i$ mit

$$e_i = P(E_t = 1 \,|\, X_t = i) = 0 \qquad \text{für alle } i \in \mathbb{N} \setminus \{0\}.$$

Demgegenüber ist die Fehlerwahrscheinlichkeit $e_0$ für den Fehlerzustand 0 gegeben durch

$$e_0 = P(E_t = 1 \,|\, X_t = 0) = 1.$$

Gemäß des Modells von Tsai werden die Fehlerwahrscheinlichkeiten auf die Werte 0 und 1 reduziert. Folglich symbolisieren die Zustände auch für diesen Modellansatz das unmögliche oder das sichere Auftreten eines Fehlers. Da auch in diesem Modell nur ein einziger Fehlerzustand zugelassen ist, verliert der Übertragungsprozeß nach jedem Auftreten eines Fehlers sein Gedächtnis. Mit der gleichen Argumentation wie bei dem Modell von Tsai erhalten wir auch bei diesem Modellansatz, dass die Längen der fehlerfreien Phasen unabhängig voneinander sind, was einen Erneuerungsprozeß charakterisiert.

## 3.4 Einführung eines endlichen Markov Modells als Kanalmodell

In den vorangegangenen Abschnitten 3.3.2 bis 3.3.7 haben wir verschiedene, aus der Literatur bekannte Kanalmodelle vorgestellt und analysiert. In diesem Zusammenhang haben wir insbesondere ihre Vor- und Nachteile mit Blick auf die Zielsetzung der Arbeit herausgestellt. Die Hauptursache, weswegen die vorgestellten Modelle zur Modellierung realer Übertragungskanäle mit Gedächtnis nur begrenzt geeignet sind, liegt im Erneuerungscharakter der meisten Modelle begründet, vgl. dazu die Paragraphen 3.3.2, 3.3.6 und 3.3.7.

Eine weitere Ursache haben wir in Paragraph 3.3.3 am Beispiel des Modells von Elliott aufgezeigt. In dem Modell nach Elliott wird zwar durch die Einführung weiterer Modellparameter

der Erneuerungscharakter aufgehoben, die Aufenthaltszeiten in den beiden Zuständen werden aber als geometrisch verteilt angenommen, vgl. hierzu Abschnitt 3.3.3. Arbeiten von Mandelbrot und Berger zeigen jedoch, dass die geometrische Verteilung für die Aufenthaltszeiten oft ungeeignet ist und auch andere Verteilungen modelliert werden müssen, siehe [10] und [3].

Aus diesem Grund ist die Entwicklung eines weiteren Kanalmodells Gegenstand dieses abschließenden Abschnittes. Bei der Konstruktion dieses Modells werden wir versuchen, die Vorteile der in den Abschnitten 3.3.2 bis 3.3.7 vorgestellten Modelle zu kombinieren und dabei gleichzeitig deren Nachteile weitestgehend abzubauen. Das von uns hier vorgestellte Modell beruht auf einem Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ mit einem endlichen Zustandsraum $\mathcal{S}$, dessen Zustände, ähnlich wie im Modell nach Elliott, verschiedene Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, implizieren. Auf diese Weise können wir den Erneuerungscharakter der bisher betrachteten Modelle aufheben. Ferner werden wir, ähnlich wie bei den Modellen nach Fritchman, Tsai bzw. Kemeny, einen endlichen Zustandsraum $\mathcal{S} = \{1, \ldots, N\}$, $n \geq 2$, betrachten, um so als Verteilung für die Längen der Aufenthaltszeiten in den einzelnen Zuständen beliebige Verteilungen modellieren zu können.

Zur Modellierung des binären Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$, mit $E_t \in \{0, 1\}$ für alle $t \in \mathbb{N}$, verwenden wir in unserem Modellansatz einen Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erster Ordnung mit einem endlichen Zustandsraum $\mathcal{S}$ im Sinne von Definition 3.3.8. Dabei sei der endliche Zustandsraum $\mathcal{S}$ gegeben durch die Menge

$$\mathcal{S} = \{1, 2, \ldots, N\}.$$

Nach Definition 3.3.8 ist der Markov'sche Hintergrundprozeß gekennzeichnet durch

(i) seinen Anfangszustand $X_0 = i_0$, mit $i_0 \in \mathcal{S}$,

(ii) und seine Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$, wobei für alle $i, j \in \mathcal{S}$ gilt:

$$\begin{aligned} q_{ij} &= P(X_{t+1} = j \mid X_t = i,\, X_{t-1} = i_{t-1}, \ldots, X_0 = i_0) \\ &= P(X_{t+1} = j \mid X_t = i)\,. \end{aligned}$$

Mit Bezug auf alle weiteren Betrachtungen gehen wir von der Annahme aus, dass der zugrunde liegende Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ stets homogen, irreduzibel und aperiodisch ist. Aus Satz 3.1.15 in Abschnitt 3.1 folgt, dass der Markov'sche Hintergrundprozeß regulär ist und dass insbesondere die stationären Zustandswahrscheinlichkeiten $\pi_i$ für jedes $i \in \mathcal{S}$ existieren, mit

$$\pi_i = \lim_{m\longrightarrow\infty} \pi_i^{(m)} \quad \text{für jedes } i \in \mathcal{S}.$$

Dabei genügen die stationären Zustandswahrscheinlichkeiten $\pi_i$ für jedes $i \in \mathcal{S}$ nach Satz 3.1.17 dem Gleichungssystem und der Normalisierungsbedingung:

$$\begin{aligned} \pi_j &= \sum_{i=1}^{N} q_{ij}\,\pi_i \quad \text{für jedes } i \in \mathcal{S} \\ \text{und} \quad \sum_{i=1}^{N} \pi_i &= 1\,. \end{aligned}$$

Gemäß Definition 3.3.8 implizieren die Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ verschiedene Fehlerwahrscheinlichkeiten $e_i$, $i \in \mathcal{S}$, die im Sinne von Definition 3.3.7 für jeden Zustand $i \in \mathcal{S}$ gegeben sind durch

$$e_i \;=\; P(E_t = 1 \,|\, X_t = i) \quad \text{für jedes } i \in \mathcal{S},$$

wobei für jede Fehlerwahrscheinlichkeit $e_i$ gilt, dass $e_i \in [0,1]$ für jedes $i \in \mathcal{S}$ ist.

Für einen Fehlerprozeß $(E_t)_{t\in\boldsymbol{N}}$, der durch einen Markov'schen Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ im obigen Sinne induziert wird, ergibt sich die Bitfehlerwahrscheinlichkeit $p_E$ insgesamt zu

$$p_E = P(E_t = 1) = \sum_{i=1}^{N} \pi_i \, e_i.$$

Da wir den binären Fehlerprozesses $(E_t)_{t\in\boldsymbol{N}}$ mittels eines Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$, der einen endlichen Zustandsraum $\mathcal{S}$ besitzt, im Sinne von Definition 3.3.8 modellieren, identifizieren wir die Zustände $i \in \mathcal{S}$ der zugrunde liegenden Markov Kette $(X_t)_{t\in\boldsymbol{N}}$ nicht direkt mit den beobachteten Fehlerbits des Fehlerprozesses $(E_t)_{t\in\boldsymbol{N}}$. Stattdessen identifizieren wir die Zustände $i \in \mathcal{S}$ mit der jeweiligen Wahrscheinlichkeit, dass in dem betrachteten Zustand $i \in \mathcal{S}$ ein Fehler generiert wird, also mit der jeweiligen Fehlerwahrscheinlichkeit $e_i$. Von daher bestimmt der aktuelle Zustand des Markov'schen Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ die Wahrscheinlichkeit eines Übertragungsfehlers und damit indirekt den Fehlerprozeß $(E_t)_{t\in\boldsymbol{N}}$.

In Abbildung 3.11 ist das Zustandsdiagramm des von uns hier vorgeschlagenen Kanalmodells mit den möglichen Zustandsübergängen anschaulich dargestellt. Da nach Voraussetzung $q_{ij} \neq 0$ für alle $i, j \in \mathcal{S}$ ist, sind Zustandsübergänge von jedem Zustand $i \in \mathcal{S}$ in jeden anderen Zustand $j \in \mathcal{S}$ gemäß der Übergangswahrscheinlichkeit $q_{ij}$ erlaubt. Darüber hinaus veranschaulicht Abbildung 3.11, dass die Zustände $i \in \mathcal{S}$ mit den jeweiligen Fehlerwahrscheinlichkeiten $e_i$ identifiziert werden.

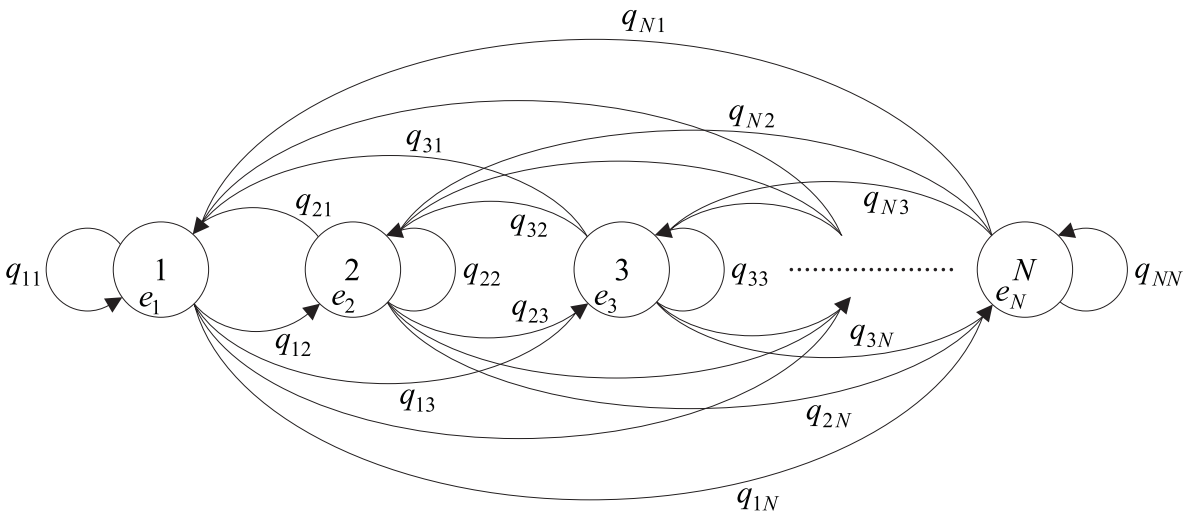


Abbildung 3.11: Markov Modell mit endlichem Zustandsraum

Durch die Identifikation der Zustände der zugrunde liegenden Markov Kette mit den jeweiligen Fehlerwahrscheinlichkeiten $e_i$ für jedes $i \in \mathcal{S}$ wird jedes Fehlerbit $E_t$ nun indirekt

durch den Markov'schen Hintergrundprozeß $(X_t)_{t\in N}$ bestimmt. Da laut Voraussetzung alle Fehlerwahrscheinlichkeiten $e_i$ aus dem Intervall $[0,1]$ sind, können bei entsprechender Wahl der Fehlerwahrscheinlichkeiten $e_i > 0$ in jedem Zustand $i \in \mathcal{S}$ des Markov'schen Hintergrundprozesses Fehler erzeugt werden, vgl. hierzu das Modell von Elliott in Paragraph 3.3.3. Dabei bestimmt die Höhe der Fehlerwahrscheinlichkeit $e_i \in ]0,1[$, wie groß der Anteil der verfälschten Bits in dem betrachteten Übertragungszustand $i \in \mathcal{S}$ ist. Von daher muß je nach Wahl der Fehlerwahrscheinlichkeit $e_i$ nicht zwangsläufig jedes Bit in dem betrachteten Übertragungszustand $i \in \mathcal{S}$ verfälscht werden. Infolgedessen ist der Markov'sche Hintergrundprozeß $(X_t)_{t\in N}$ vom beobachtbaren Fehlerprozeß $(E_t)_{t\in N}$ aus nun nicht mehr zu rekonstruieren.

### 3.4.1 Bemerkung

(i) Für den trivialen Fall, dass der Zustandsraum $\mathcal{S}$ des Markov'schen Hintergrundprozesses $(X_t)_{t\in N}$ nur aus genau einem Zustand $b$ besteht, entspricht das so konstruierte Kanalmodell dem gedächtnislosen symmetrischen Binärkanal, siehe Abschnitt 3.3.1.

(ii) Für den Fall, dass der Zustandsraum genau zwei Zustände umfasst, erhalten wir das Modell von Elliott, vgl. Paragraph 3.3.3.

(iii) In dem Spezialfall, dass der Zustandsraum genau zwei Zustände umfasst und zusätzlich eine der beiden Fehlerwahrscheinlichkeiten identisch null ist, stimmt das Kanalmodell mit dem Modell von Gilbert überein, siehe Abschnitt 3.3.2.

(iv) Werden die Fehlerwahrscheinlichkeiten $e_i$ für alle $i \in \mathcal{S}$ entweder gleich 0 oder gleich 1 gewählt, so deckt sich das so entstandene Kanalmodell mit dem Modell von Fritchman, vgl. Paragraph 3.3.5. Für den speziellen Fall, dass nur genau eine Fehlerwahrscheinlichkeit gleich 1 ist und alle übrigen gleich 0 sind, fällt das Modell mit dem Modell von Tsai zusammen, siehe Paragraph 3.3.6.

### 3.4.2 Bemerkung

Ferner beinhaltet das von uns in diesem Abschnitt vorgestellte endliche Markov Modell semi-Markov Prozesse mit endlichem Zustandsraum. Wenn die Aufenthaltszeiten eines semi-Markov Prozesses als endlich angenommen werden, kann der diskrete semi-Markov Prozeß als volle Markov Kette dargestellt werden. Dazu wird der Zustandsraum des semi-Markov Prozesses folgendermaßen gebildet. Die eingebettete Markov Kette wird mit einer zweiten Komponente kombiniert. Diese zählt die Aufenthaltsdauer von dem Moment an, zu dem ein Zustand der eingebetteten Markov Kette erreicht wird. Die zweite Komponente wird hochgezählt oder im folgenden Zustand zurückgesetzt. Die erste Komponente ändert sich nur, wenn die zweite zurückgesetzt wird. So können wir auch Poisson-verteilte Aufenthaltszeiten modellieren, vgl. Abschnitt 5.2.2.

### 3.4.3 Bemerkung

Aus dem von uns hier vorgeschlagenen Ansatz zur Modellierung von Übertragungskanälen mit Gedächtnis ergeben sich die folgenden deutlichen Vorteile gegenüber den bekannten Konzepten:

- Erstens gelingt es den Erneuerungscharakter der bisherigen Modelle aufzuheben, da,

ähnlich wie im Modell von Elliott, durch entsprechende Wahl der Fehlerwahrscheinlichkeiten $e_i$ in jedem Zustand $i \in \mathcal{S}$ Übertragungsfehler generiert werden können.

- Zweitens beinhaltet dieser Ansatz nach der vorherigen Bemerkung auch diskrete semi-Markov Prozesse mit endlichem Zustandsraum. In diesem Rahmen können die Aufenthaltszeiten in den einzelnen Zuständen auch beliebige Verteilungen annehmen, im Gegensatz zum Gilbert Modell beispielsweise. Hier sind die Aufenthaltszeiten geometrisch verteilt, vgl. dazu Abschnitt 3.3.2 sowie Bemerkung 3.1.21 in Abschnitt 3.1. Beispielsweise können mit diesem Modell Poisson-verteilte Aufenthaltszeiten modelliert werden. Diesen Fall analysieren wir genauer in Abschnitt 5.2.2.

- Drittens können mit diesem Ansatz Störungen realistisch nachgebildet werden, d.h. es lassen sich sowohl einzelne Übertragungsfehler als auch solche in zeitlicher Nähe modellieren. Ferner läßt sich auf der Basis dieses Kanalmodells ein Verfahren entwickeln, das Rekursionsgleichungen umfasst und zur Bewertung verschiedener Codierverfahren im Hinblick auf ihre Effizienz bei der Fehlerkorrektur dient. Besonders hervorzuheben in diesem Zusammenhang ist, dass in diesem Verfahren die unterschiedlichen algebraischen Strukturen verschiedener Codierverfahren berücksichtigt werden, was in dieser Form in der Literatur bislang nicht bekannt ist. Das angesprochene, effiziente Berechnungsverfahren zur Beurteilung verschiedener Codierverfahren wird insbesondere im folgenden Kapitel 4 in Abschnitt 4.2 ausgearbeitet und analysiert.

# Kapitel 4

# Grundlagen von Codes und ihre Bewertung

Nachdem wir in Kapitel 3 verschiedene Kanalmodelle zur Modellierung des während der Übertragung aufgetretenen Fehlerprozesses vorgestellt und ihre Vor- und Nachteile gegeneinander abgewogen haben, befassen wir uns in diesem Kapitel mit verschiedenen Verfahren und Mechanismen zur Absicherung der zu übermittelnden Information gegenüber zufällig auftretenden Störungen. Zu Beginn dieses Kapitels stellen wir dazu im ersten Abschnitt elementare mathematische Grundlagen zusammen. Vor diesem Hintergrund werden ferner unterschiedliche Möglichkeiten zur Konstruktion von Codierverfahren präsentiert. Im zweiten Abschnitt dieses Kapitels analysieren wir schließlich das Fehlerkorrekturverhalten der verschiedenen Codierverfahren. Mit Blick auf die Zielsetzung der Arbeit, ein optimales Codierverfahren anhand der Fehlerstatistik des verwendeten Übertragungskanals auszuwählen, entwickeln wir abschließend ein Bewertungsschema, um die Effizienz und Leistungsfähigkeit der verschiedenen Codierverfahren zu beurteilen.

## 4.1 Grundlagen der Codierungstheorie

In diesem Abschnitt befassen wir uns mit den mathematischen Grundbegriffen der Codierungstheorie, die zwei wesentliche, aber völlig unterschiedliche Konzepte zur Fehlererkennung bzw. zur Fehlerkorrektur hervorgebracht hat. So beschäftigen wir uns in diesem Paragraphen ausschließlich mit Codierern, die zur Erzeugung zusätzlicher Redundanz und damit zur Fehlererkennung bzw. -korrektur einen sogenannten *Blockcode* verwenden und analysieren deren spezifische Eigenschaften im Hinblick auf die Fehlerkorrektur. Das Prinzip der *Faltungscodierer* wird im Zuge der Arbeit nur am Rande erwähnt. Eine Erweiterung der in der Arbeit entwickelten Methoden auf das Konzept der Faltungscodierer wird allerdings in Kapitel 8 kurz angedacht.

Zu Beginn dieses Abschnittes 4.1 führen wir die elementaren Begriffe und Bezeichnungen ein und definieren darauf aufbauend lineare sowie zyklische Blockcodes. Dabei orientieren wir uns bezüglich der Notation an Standardwerken der Informations- und Codierungstheorie, wie beispielsweise [4, 7, 13, 18, 26, 34, 43, 47] oder auch [27]. Im wesentlichen konzentrieren wir uns auf zwei ganz spezielle Klassen von Blockcodes: Die linearen sowie die zyklischen

Blockcodes, die wir in Abschnitt 4.1.1 bzw. Abschnitt 4.1.2 einführen. Indem wir die algebraischen Strukturen dieser Codes genau analysieren, können wir erste Schranken für die Fehlerkorrektur angeben. Zum Abschluß dieses Paragraphen befassen wir uns in Abschnitt 4.1.3 mit einigen konkreten Beispielen für zyklische Blockcodes, deren Einsatz in der Praxis äußerst weit verbreitet ist.

**4.1.1 Definition**
*Gegeben sei $\mathcal{A} = \{a_1, \ldots, a_n\}$ eine endliche und nichtleere Menge, die wir im folgenden als Alphabet $\mathcal{A}$ bezeichnen. Jedes Element $a_i \in \mathcal{A}$ heißt dann ein Symbol des Alphabets $\mathcal{A}$. Ein Wort über dem Alphabet $\mathcal{A}$ ist definiert als eine beliebige Folge von Symbolen aus $\mathcal{A}$. Die Menge aller Wörter über dem Alphabet $\mathcal{A}$ wird mit $\mathcal{A}^*$ bezeichnet. Die Länge eines Wortes entspricht der Anzahl der Symbole des Alphabets $\mathcal{A}$, aus denen das betrachtete Wort besteht.*

Als zugrunde liegendes Alphabet $\mathcal{A}$ verwenden wir im Rahmen dieser Arbeit stets den Körper mit zwei Elementen $GF(2)$.

**4.1.2 Definition**
*Gegeben sei $GF(2)$ als zugrunde liegendes Codealphabet.*

*(i) Eine endliche und nichtleere Teilmenge $\mathcal{C}$, $\mathcal{C} \neq \emptyset$, der Menge $GF(2)^*$ aller Wörter über $GF(2)$ heißt ein binärer Code.*

*(ii) Die Elemente eines binären Codes $\mathcal{C}$ werden als Codewörter von $\mathcal{C}$ bezeichnet.*

*(iii) Ein binärer Code $\mathcal{C}$ heißt ein binärer Blockcode der Länge $n$, wenn alle Codewörter dieselbe Länge $n$ besitzen. Wir bezeichnen $n$ dann als die Codelänge von $\mathcal{C}$.*

Aus dieser Definition folgt sofort, dass jede endliche und nichtleere Teilmenge $\mathcal{C} \subset GF(2)^n$, $\mathcal{C} \neq \emptyset$, der Menge aller Binärwörter der Länge $n$ einen binären Blockcode der Länge $n$ bildet, wenn der Körper $GF(2)$ als Codealphabet zugrunde gelegt wird.

**4.1.3 Definition**
*Unter einem systematischen, binären Blockcode $\mathcal{C}$ der Länge $n$ verstehen wir einen binären Blockcode $\mathcal{C}$ der Länge $n$ über dem Codealphabet $GF(2)$ mit der Eigenschaft, dass $k$ Indizes*

$$\{i_0, \ldots, i_{k-1}\} \subset \{0, 1, \ldots, n-1\}, \text{ mit } \quad k < n,$$

*derart existieren, dass sich durch die Einschränkung aller Codewörter $\mathbf{c} \in \mathcal{C}$ auf diese $k$ Indizes alle möglichen Binärwörter der Länge $k$ ergeben; d.h. es gilt:*

$$\{(c_{i_0}, c_{i_1}, \ldots, c_{i_{k-1}}) \; ; \; \mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C}\} = GF(2)^k$$

*Die Indizes $i_0, \ldots, i_{k-1}$ werden als Informationsstellen und die Codesymbole an diesen Stellen als Informationssymbole bezeichnet.*

Die Arbeitsweise eines Kanalcodierers, der zur Erzeugung der Redundanz einen binären oder sogar einen systematischen, binären Blockcode der Länge $n$ verwendet, läßt sich dann wie folgt beschreiben:

Der Kanalcodierer erhält als Eingang eine Folge von binären Informationssymbolen und teilt diese in Blöcke zu jeweils $k$ Informationsstellen ein. Je nachdem welcher binäre Blockcode der Länge $n$ zur Codierung verwendet wird, variiert das Verfahren, mit dem aus den $k$ Informationsbits $n-k$ sogenannte *Kontroll-* oder *Prüfbits* bestimmt werden. Ein Codewort $\mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C}$ setzt sich dann jeweils aus den $k$ Informationsbits sowie weiteren $n-k$ *Kontrollbits* zusammen. Insbesondere werden bei einem systematischen Code $\mathcal{C}$ die $k$ Informationsbits unverändert in die Codewörter übernommen und stehen dabei üblicherweise am Anfang des Codewortes, also

$$\mathbf{c} = (c_0, \ldots, c_{k-1}, c_k, \ldots, c_{n-1}) = (i_0, \ldots, i_{k-1}, c_k, \ldots, c_{n-1}) \in \mathcal{C}.$$

Von daher unterscheiden sich zwei verschiedene Codewörter eines systematischen Codes in den ersten $k$ Informationsstellen.

**4.1.4 Definition**
*Ein binärer Blockcode $\mathcal{C}$ der Länge $n$, der genau $k$ Informationsstellen mit $k < n$ im Sinne der obigen Ausführungen besitzt, wird als binärer $(n,k)$-Blockcode $\mathcal{C}$ bezeichnet. Dabei ist dieser $(n,k)$-Blockcode $\mathcal{C}$ nicht unbedingt ein systematischer Code.*

Ein binärer $(n,k)$-Blockcode $\mathcal{C}$ der Länge $n$ besitzt nach der vorangegangenen Definition genau $k$ Informationsstellen, zu denen sich insgesamt $2^k$ verschiedene Informationsblöcke bilden lassen. Diese werden ihrerseits in die entsprechenden $2^k$ Codewörter umgewandelt. Folglich besitzt jeder binäre $(n,k)$-Blockcode $\mathcal{C}$ insgesamt $2^k$ verschiedene Codewörter.

**4.1.5 Definition**

*(i) Einen binären $(n,k)$-Blockcode $\mathcal{C}$ der Länge $n$ mit genau $2^k$ Codewörtern bezeichnen wir als vollen Code.*

*(ii) Das Verhältnis $k/n$ von der Anzahl $k$ der Informationsstellen zur Codelänge $n$ nennen wir die Coderate von $\mathcal{C}$. In der Literatur wird in diesem Zusammenhang auch der Begriff der Informationsrate verwendet, vgl. dazu [18].*

Nachdem wir die ersten elementaren Grundbegriffe der Codierungstheorie eingeführt haben, geben wir zur Veranschaulichung zwei ganz einfache Beispiele für systematische binäre $(n,k)$-Blockcodes der Länge $n$ mit $2^k$ Codewörtern an.

**4.1.6 Beispiel**
Der *binäre $(n,1)$-Wiederholungscode* $\mathcal{RP}_n$ der Länge $n$, auch *repetition code* oder *$\mathcal{RP}$-Code* genannt, besitzt genau eine Informationsstelle $i_0$ und besteht aus genau zwei Codewörtern, nämlich dem Nullwort und dem Alleinsenwort:

$$\mathcal{RP}_n \;:=\; \{(0,\ldots,0)\,,\,(1,\ldots,1)\}$$

Mit dem binären $(n, 1)$-Wiederholungscode $\mathcal{RP}_n$ kann genau ein Informationsbit $i_0$ codiert werden. Dabei lautet die Codiervorschrift des $(n, 1)$-Wiederholungscodes $\mathcal{RP}_n$, dass das Informationsbit $i_0$ gerade $(n-1)$-mal zu wiederholen ist. Demnach ergibt sich ein Codewort $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ des $(n, 1)$-Wiederholungscodes $\mathcal{RP}_n$ durch:

$$c_0 := c_1 := \cdots := c_{n-1} := i_0$$

In Abbildung 4.1 ist die Bildung eines Codewortes $\mathbf{c} \in \mathcal{RP}_8$ anschaulich für einen $\mathcal{RP}$-Code der Länge 8 dargestellt.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $i_0$ | $i_0$ | $i_0$ | $i_0$ | $i_0$ | $i_0$ | $i_0$ | $i_0$ |

Abbildung 4.1: Codewort des $\mathcal{RP}$-Codes der Länge 8

### 4.1.7 Beispiel

Ein weiteres Beispiel eines systematischen, binären Blockcodes der Länge $n$ ist der sogenannte *Parity-Check-Code* $\mathcal{PC}_n$, der in der Literatur kurz auch als $\mathcal{PC}$*-Code* bezeichnet wird. Ein Parity-Check-Code $\mathcal{PC}_n$ ist ein $(n, n-1)$-Blockcode und die Codiervorschrift dazu lautet:

(i) Setze die ersten $n-1$ Stellen eines Codewortes $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}) \in \mathcal{PC}_n$ gleich den vorgegebenen Informationsbits $i_0, i_1, \ldots, i_{n-2}$, also

$$c_0 := i_0\,,\; c_1 := i_1\,,\ldots,\; c_{n-2} := i_{n-2}$$

(ii) Setze die letzte Stelle $c_{n-1}$ des Codewortes $\mathbf{c} = (i_0, \ldots, i_{n-2}, c_{n-1})$ gleich der binären Summe der Informationsbits $i_0, \ldots, i_{n-2}$, also

$$c_{n-1} := \sum_{j=0}^{n-2} i_j \mod 2$$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $i_0$ | $i_1$ | $i_2$ | $i_3$ | $i_4$ | $i_5$ | $i_6$ | $pc$ |

Abbildung 4.2: Codewort des $\mathcal{PC}$-Codes der Länge 8

Abbildung 4.2 zeigt, wie aus sieben gegebenen Informationsbits ein Codewort des $(8, 7)$-Parity-Check-Codes $\mathcal{PC}_8$ gebildet wird. Dabei stellt $pc$ die binäre Summe der sieben Informationsbits $i_0, i_1, \ldots, i_6$ dar und wird daher auch als *parity check bit* bezeichnet.

### 4.1.1 Lineare Blockcodes

Nun besitzt ein beliebiger systematischer binärer $(n,k)$-Blockcode $\mathcal{C}$ insgesamt $2^k$ Codewörter der Länge $n$. Falls die Parameter $k$ und $n$ sehr groß sind, wird für einen Code, der keine weitere mathematische Struktur besitzt, der Aufbau des Codierers und des Decodierers, deren Implementierung und Organisation zu komplex. Sowohl der Codierer als auch der Decodierer müssen dann für ihre Aufgaben alle $2^k$ Codewörter der Länge $n$ in sogenannten Code-Tabellen abspeichern. Eine wesentliche Vereinfachung der Struktur des Codierers und des Decodierers ist jedoch zu erwarten, wenn der verwendete Code noch mehr mathematische Struktur besitzt. Daher befassen wir uns in diesem Abschnitt ausschließlich mit der Klasse der sogenannten *linearen $(n,k)$-Blockcodes*. Die zusätzliche strukturelle Eigenschaft der Linearität wird wesentlich zur Verringerung der Komplexität von Codierer und Decodierer beitragen.

**4.1.8 Bemerkung**
Wir versehen dazu die Menge $GF(2)^n$ aller Binärwörter der Länge $n$ mit einer Vektorraumstruktur. Die Addition zweier Binärwörter $\mathbf{a},\mathbf{c} \in GF(2)^n$ definieren wir hierzu als komponentenweise binäre Addition $\oplus$ durch:

$$\begin{aligned} + : \left(GF(2)^n, GF(2)^n\right) &\longrightarrow GF(2)^n \\ \mathbf{a} + \mathbf{c} &\longmapsto (a_0 \oplus c_0, a_1 \oplus c_1, \ldots, a_{n-1} \oplus c_{n-1}) \\ &\qquad \text{für alle } \mathbf{a},\mathbf{c} \in GF(2)^n \end{aligned}$$

Dann bildet die Gruppe $\left(GF(2)^n, +\right)$ zusammen mit der skalaren Multiplikation, die als komponentenweise skalare binäre Multiplikation $\odot$ über $GF(2)$ erklärt ist

$$\begin{aligned} \cdot : \left(GF(2), GF(2)^n\right) &\longrightarrow GF(2)^n \\ \lambda \cdot \mathbf{c} &\longmapsto (\lambda \odot c_0, \lambda \odot c_1, \ldots, \lambda \odot c_{n-1}) \\ &\qquad \text{für alle } \mathbf{c} \in GF(2)^n \text{ und alle } \lambda \in GF(2), \end{aligned}$$

einen Vektorraum der Dimension $n$.

**4.1.9 Definition**
*Ein binärer $(n,k)$-Blockcode $\mathcal{C}$, mit $0 < k < n$, heißt ein linearer $(n,k)$-Blockcode, wenn seine $2^k$ Codewörter einen $k$-dimensionalen Untervektorraum des Vektorraumes $GF(2)^n$ aller Binärwörter der Länge $n$ bilden.*

*Die Anzahl $k$ der Informationsbits wird als Dimension des linearen $(n,k)$-Blockcodes $\mathcal{C}$ bezeichnet und $n$ gibt nach wie vor die Codelänge an.*

**4.1.10 Satz**
*Ein binärer $(n,k)$-Blockcode $\mathcal{C}$ ist genau dann ein linearer $(n,k)$-Blockcode, wenn die Summe zweier Codewörter $\mathbf{a},\mathbf{c} \in \mathcal{C}$ wieder ein Codewort liefert und damit die komponentenweise binäre Addition über dem betrachteten Code $\mathcal{C}$ abgeschlossen ist.*

Zum Beweis verweisen wir lediglich auf einige Lehrbücher, wie beispielsweise [16, 18, 26], die eine detaillierte Einführung zum Thema, lineare Codes, geben. Häufig findet sich in

der Literatur im Zusammenhang mit dem Begriff der Linearität auch der Begriff des *Gruppencodes*, vgl. [16, 18], da ein linearer $(n, k)$-Blockcode versehen mit der komponentenweise binären Addition die Gruppenaxiome erfüllt.

Die Definition eines linearen $(n, k)$-Blockcodes $\mathcal{C}$ könnte die Vermutung zu lassen, dass die Theorie der linearen Codes nichts weiter als die Theorie endlich-dimensionaler Vektorräume über endlichen Körpern wäre. Zwei lineare $(n, k)$-Blockcodes über demselben Körper sind als Vektorräume zwar stets isomorph, aber sie können höchst unterschiedliche Fehlerkorrektureigenschaften besitzen, wie wir später sehen werden. Zur Bestimmung der Korrektureigenschaften linearer Codes führen wir noch weitere elementare Grundbegriffe aus der Codierungstheorie ein. Hierzu versehen wir die Menge $GF(2)^n$ aller Binärwörter der Länge $n$ nicht nur mit der obigen Vektorraumstruktur sondern zusätzlich noch mit der Struktur eines metrischen Raumes, mittels der sogenannten *Hamming-Metrik*. Hinsichtlich der Notation orientieren wir uns dabei in erster Linie an [7].

**4.1.11 Definition**
*Gegeben sei der Vektorraum $GF(2)^n$ aller Binärwörter der Länge $n$, dann wird das Hamming-Gewicht* wt *eines Binärwortes $\mathbf{c} \in GF(2)^n$ definiert als die Anzahl der von Null verschiedenen Elemente von $\mathbf{c}$:*

$$\mathrm{wt}(\mathbf{c}) := \sum_{i=0}^{n-1} c_i \quad \textit{für jedes } \mathbf{c} \in GF(2)^n$$

**4.1.12 Definition**
*Gegeben sei der Vektorraum $GF(2)^n$ sämtlicher Binärwörter der Länge $n$, dann ist die Hamming-Distanz* dist *zweier Binärwörter $\mathbf{a}, \mathbf{c} \in GF(2)^n$ definiert als die Anzahl der Elemente, in denen sich die Binärwörter $\mathbf{a}$ und $\mathbf{c}$ unterscheiden:*

$$\mathrm{dist}(\mathbf{a}, \mathbf{c}) := \sum_{i=0}^{n-1} (a_i \oplus c_i) \quad \textit{für alle } \mathbf{a}, \mathbf{c} \in GF(2)^n$$

**4.1.13 Bemerkung**

(i) Offensichtlich erfüllt die Hamming-Distanz auf dem Vektorraum $GF(2)^n$ aller Binärwörter der Länge $n$ die Eigenschaften einer Metrik, denn es gilt:

$$\begin{aligned}
\mathrm{dist}(\mathbf{a}, \mathbf{c}) = 0 &\Leftrightarrow \mathbf{a} = \mathbf{c} && \text{für alle } \mathbf{a}, \mathbf{c} \in GF(2)^n \\
\mathrm{dist}(\mathbf{a}, \mathbf{c}) &= \mathrm{dist}(\mathbf{c}, \mathbf{a}) && \text{für alle } \mathbf{a}, \mathbf{c} \in GF(2)^n \\
\mathrm{dist}(\mathbf{a}, \mathbf{c}) &\leq \mathrm{dist}(\mathbf{a}, \mathbf{b}) + \mathrm{dist}(\mathbf{b}, \mathbf{c}) && \text{für alle } \mathbf{a}, \mathbf{b}, \mathbf{c} \in GF(2)^n
\end{aligned}$$

(ii) Darüber hinaus folgt aus der Definition der Hamming-Distanz, dass die Hamming-Distanz zweier beliebiger Binärwörter $\mathbf{a}, \mathbf{c} \in GF(2)^n$ dem Hamming-Gewicht der Summe dieser beiden Binärwörter entspricht:

$$\mathrm{dist}(\mathbf{a}, \mathbf{c}) = \mathrm{dist}(\mathbf{a} + \mathbf{c}, \mathbf{0}) = \mathrm{wt}(\mathbf{a} + \mathbf{c}) \quad \text{für alle } \mathbf{a}, \mathbf{c} \in GF(2)^n$$

**4.1.14 Definition**
*Als Mindestdistanz $d$ eines beliebigen binären $(n,k)$-Blockcodes $\mathcal{C}$ bezeichnen wir die minimale Distanz zweier unterschiedlicher Codewörter $\mathbf{a},\mathbf{c}\in\mathcal{C}$, $\mathbf{a}\neq\mathbf{c}$. Somit gilt*

$$d := \min\{\mathrm{dist}(\mathbf{a},\mathbf{c});\ \mathbf{a},\mathbf{c}\in\mathcal{C}\ ,\ \mathbf{a}\neq\mathbf{c}\}$$

*Besitzt ein linearer $(n,k)$-Blockcode $\mathcal{C}$ die Mindestdistanz $d$, so bezeichnen wir $\mathcal{C}$ insbesondere als $(n,k,d)$-Blockcode $\mathcal{C}$.*

**4.1.15 Satz**
*Für einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ entspricht die Mindestdistanz $d$ gerade dem minimalen Gewicht der vom Nullwort verschiedenen Codewörter von $\mathcal{C}$.*

**Beweis**
Da nach Satz 4.1.10 für einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ die Summe zweier Codewörter $\mathbf{a},\mathbf{c}\in\mathcal{C}$ wieder ein Codewort $\mathbf{a}+\mathbf{c}\in\mathcal{C}$ liefert, erhalten wir mit Bemerkung 4.1.13 (ii)

$$\begin{aligned} d &:= \min\{\mathrm{dist}(\mathbf{a},\mathbf{c});\ \mathbf{a},\mathbf{c}\in\mathcal{C}\ ,\ \mathbf{a}\neq\mathbf{c}\}\\ &\overset{4.1.10}{=} \min\{\mathrm{dist}(\mathbf{a}+\mathbf{c},\mathbf{0});\ \mathbf{a}+\mathbf{c}\in\mathcal{C}\ ,\ \mathbf{a}+\mathbf{c}\neq\mathbf{0}\}\\ &\overset{4.1.13(ii)}{=} \min\{\mathrm{wt}(\mathbf{a}+\mathbf{c});\ \mathbf{a}+\mathbf{c}\in\mathcal{C}\ ,\ \mathbf{a}+\mathbf{c}\neq\mathbf{0}\}\\ &=: \mathrm{wt}_{min} \end{aligned}$$

Dabei wird $\mathrm{wt}_{min} := \min\{\mathrm{wt}(\mathbf{c})\,;\,\mathbf{c}\in\mathcal{C}\ ,\ \mathbf{c}\neq\mathbf{0}\}$ auch als das *Minimalgewicht* des linearen Codes $\mathcal{C}$ bezeichnet. □

Mit der Mindestdistanz $d$ haben wir einen wichtigen Parameter eingeführt, der die Fehlerkorrektureigenschaften eines linearen $(n,k,d)$-Blockcodes $\mathcal{C}$ der Länge $n$ und der Dimension $k$ entscheidend bestimmt. So lassen sich mit Hilfe der Mindestdistanz $d$ eines linearen $(n,k,d)$-Blockcodes Aussagen darüber machen, wie viele Fehler der betrachtete Code erkennen und wie viele er korrigieren kann. Wir betrachten dazu zunächst den folgenden Satz zur Fehlererkennung. Das anschließende Beispiel dient als Vorbereitung für eine entsprechende Aussage zur Fehlerkorrektur.

**4.1.16 Satz**
*Es sei ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$ der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d$ gegeben. Dann ermöglicht die Mindestdistanz $d$ stets eine Fehlererkennung von bis zu $d-1$ Fehlern.*

**Beweis**
Zum Beweis dieser Aussage betrachten wir ein beliebiges Codewort $\mathbf{c}\in\mathcal{C}$, das über dem gestörten digitalen Kanal verfälscht wird, indem ein Fehlermuster $\mathbf{f}\in GF(2)^n$ der Länge $n$ mit $\mathrm{wt}(\mathbf{f}) =: t$ Fehlern zu dem gesendeten Codewort $\mathbf{c}\in\mathcal{C}$ binär hinzuaddiert wird. Es wird

daher ein Binärwort $\mathbf{r} \in GF(2)^n$ mit $\mathbf{r} = \mathbf{c} + \mathbf{f}$ empfangen, das sich folglich an $t$ Stellen von dem gesendeten Codewort $\mathbf{c}$ unterscheidet. Laut Bemerkung 4.1.13 (ii) gilt für die Distanz von $\mathbf{c}$ und $\mathbf{r}$

$$\mathrm{dist}(\mathbf{c}, \mathbf{r}) = \mathrm{wt}(\mathbf{c} + \mathbf{r}) = \mathrm{wt}(\mathbf{c} + \mathbf{c} + \mathbf{f}) = \mathrm{wt}(\mathbf{f}) = t.$$

Da der $(n, k, d)$-Blockcode $\mathcal{C}$ laut Voraussetzung die Mindestdistanz $d$ besitzt, unterscheiden sich zwei verschiedene, beliebige Codewörter $\mathbf{a}, \mathbf{c} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$, an mindestens $d$ Stellen, d.h. es gilt $\mathrm{dist}(\mathbf{a}, \mathbf{c}) \geq d$. Daher existiert kein Fehlermuster $\mathbf{f} \in GF(2)^n$ mit $t \leq d-1$ Fehlern, das ein Codewort $\mathbf{a}$ in ein anderes Codewort $\mathbf{c}$ verfälschen kann. Somit ist jedes empfangene Binärwort $\mathbf{r} \in GF(2)^n$ mit $t$ Fehlern und $t \leq d-1$ kein Codewort von $\mathcal{C}$. Alle Fehlermuster $\mathbf{f} \in GF(2)^n$ mit $t \leq d-1$ Fehlern werden von einem $(n, k, d)$-Blockcode $\mathcal{C}$ erkannt.

Andererseits existieren aufgrund der Definition der Mindestdistanz mindestens zwei Codewörter $\mathbf{a}, \mathbf{c} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$ und $\mathrm{dist}(\mathbf{a}, \mathbf{c}) = d$, die sich an genau $d$ Stellen unterscheiden. Daneben gibt es ein Fehlermuster $\mathbf{f} \in GF(2)^n$ mit genau $t = d$ Fehlern an exakt den Stellen, an denen sich die Codewörter $\mathbf{a}$ und $\mathbf{c}$ unterscheiden. Somit kann das Codewort $\mathbf{a}$ in das Codewort $\mathbf{c}$ verfälscht werden, bzw. umgekehrt. Daher sind Fehlermuster mit $t \geq d$ Fehlern nicht unbedingt erkennbar.

Ein linearer $(n, k, d)$-Blockcode der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d$ erkennt also stets bis zu $d-1$ Fehler. □

Neben der Fehlererkennung von bis zu $d-1$ Fehlern ist für eine begrenzte Anzahl von Stellen eine Korrektur der erkannten Fehler möglich. Um eine zu Satz 4.1.16 vergleichbare Aussage im Hinblick auf die Fehlerkorrektur formulieren und beweisen zu können, veranschaulichen wir in Abbildung 4.3 und Abbildung 4.4 den Sachverhalt der Fehlerkorrigierbarkeit anhand einer sehr vereinfachten Hilfsdarstellung.

Für einen $(n, k, d)$-Blockcode $\mathcal{C}$ der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d = 5$ bzw. $d = 6$ sind vereinfacht zwei verschiedene Codewörter $\mathbf{a}, \mathbf{c} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$, dargestellt. Dabei entspricht deren Distanz $\mathrm{dist}(\mathbf{a}, \mathbf{c})$ der Mindestdistanz $d$ von $\mathcal{C}$. Die übrigen Punkte, die in Abbildung 4.3 und 4.4 eingezeichnet sind, repräsentieren weitere Binärwörter $\mathbf{r} \in GF(2)^n$. Dabei besitzen zwei benachbarte Punkte jeweils die Distanz 1. Wegen $d = 5$ bzw. $d = 6$ stellen diese aber keine Codewörter dar. Zur Fehlerkorrektur muß jedem empfangenen Binärwort $\mathbf{r} \in GF(2)^n$ mit $\mathbf{r} = \mathbf{c} + \mathbf{f}$ genau ein Codewort $\mathbf{c} \in \mathcal{C}$ eindeutig zugeordnet werden können. Dazu legen wir um jedes Codewort $\mathbf{c} \in \mathcal{C}$ eine Kugel derart, dass in dieser Kugel alle Binärwörter $\mathbf{r}$ mit Distanz $\mathrm{dist}(\mathbf{r}, \mathbf{c}) \leq d/2$ enthalten sind. Die Fehlerkorrektur erfolgt dann in dem Sinne, dass dem empfangenen Wort $\mathbf{r}$ dasjenige Codewort $\mathbf{c} \in \mathcal{C}$ zugeordnet wird, in dessen Kugel das empfangene Binärwort $\mathbf{r}$ liegt. Für eine eindeutige Zuordnung müssen demnach sämtliche Kugeln paarweise disjunkt sein und $GF(2)^n$ überdecken.

Dies ist in Abbildung 4.3 erfüllt, so dass jedem Binärwort $\mathbf{r}$ bei der Fehlerkorrektur genau ein Codewort $\mathbf{c} \in \mathcal{C}$ mit $\mathrm{dist}(\mathbf{r}, \mathbf{c}) \leq d/2$ zugeordnet werden kann.

Anders verhält es sich in Abbildung 4.4. Hier sind die Kugeln mit Radius $d/2$ zwar überdeckend, aber nicht mehr paarweise disjunkt. So kann dem Binärwort $\mathbf{r} \in GF(2)^n$ mit der Distanz $\mathrm{dist}(\mathbf{a}, \mathbf{r}) = \mathrm{dist}(\mathbf{c}, \mathbf{r}) = 3$ sowohl das Codewort $\mathbf{a}$ als auch das Codewort $\mathbf{c}$ zugeordnet werden. Eine Verringerung des Radius auf $< d/2$ ermöglicht wieder eine eindeutige Zuordnung. Allerdings überdecken die Kugeln $GF(2)^n$ dann nicht mehr vollständig.

Bei den vorangegangenen Überlegungen und der Hilfsdarstellung in Abbildung 4.3 bzw. Abbildung 4.4 haben wir vorausgesetzt, dass weniger Fehler wahrscheinlicher sind als viele. Als

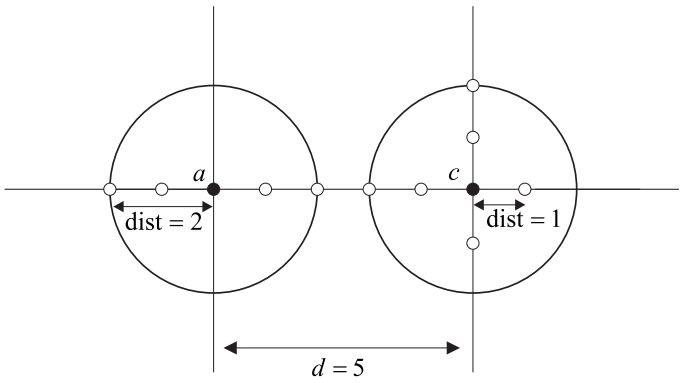


Abbildung 4.3: Korrigierkugeln eines $(n, k, 5)$-Blockcodes

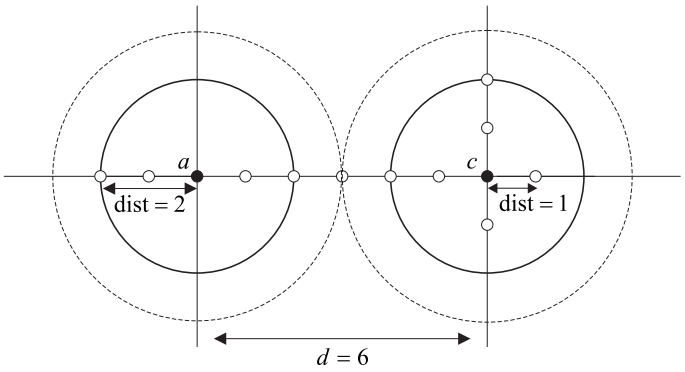


Abbildung 4.4: Korrigierkugeln eines $(n, k, 6)$-Blockcodes

Decodierentscheidung wird daher das Codewort $\mathbf{c} \in \mathcal{C}$ mit der geringsten Distanz zum empfangenen Binärwort $\mathbf{r} \in GF(2)^n$ gewählt. Anhand der Korrigierkugeln und deren Radius lassen sich nun Aussagen zur Fehlerkorrigierbarkeit ableiten.

**4.1.17 Definition**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d$.*

*(i) Die Menge*

$$K^{\varrho}(\mathbf{c}) := \{\mathbf{r} \in GF(2)^n;\, \mathrm{dist}(\mathbf{c}, \mathbf{r}) \leq \varrho\}$$

*bildet eine sogenannte Korrigierkugel mit dem Radius $\varrho$ um das Codewort $\mathbf{c} \in \mathcal{C}$.*

*(ii) Der Überdeckungsradius $\rho$ ist der kleinstmögliche Radius $\varrho$, so dass alle Binärwörter $\mathbf{r} \in GF(2)^n$ in mindestens einer der Korrigierkugeln $K^{\varrho}(\mathbf{c})$ um ein Codewort $\mathbf{c} \in \mathcal{C}$ liegen, also*

$$\rho \;:=\; \min\{\,\varrho;\, \forall_{\mathbf{r}\in GF(2)^n}\, \exists_{\mathbf{c}\in\mathcal{C}}\, \mathrm{dist}(\mathbf{c}, \mathbf{r}) \leq \varrho\} = \max_{\mathbf{r}\in GF(2)^n} \min\{\mathrm{dist}(\mathbf{c}, \mathbf{r})\,;\, \mathbf{c} \in \mathcal{C}\}$$

*Der Überdeckungsradius $\rho$ entspricht damit der sogenannten maximalen Distanz eines Binärwortes zum nächstgelegenen Codewort.*

**4.1.18 Korollar**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d$.*

*(i) Die Größe einer Korrigierkugel $K^{\varrho}(\mathbf{c})$ läßt sich bestimmen als Anzahl aller Binärwörter, deren Distanz zum Codewort $\mathbf{c} \in \mathcal{C}$ kleiner oder gleich $\varrho$ ist, und ergibt sich damit zu*

$$|K^{\varrho}(\mathbf{c})| = \sum_{i=0}^{\varrho} \binom{n}{i}$$

*(ii) Die Definition der Mindestdistanz als minimale Distanz zweier beliebiger unterschiedlicher Codewörter $\mathbf{a}, \mathbf{c} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$, liefert, dass alle Korrigierkugeln $K^{\varrho}(\mathbf{c})$ vom Radius $\varrho < d/2$ paarweise disjunkt sind.*

Verallgemeinern wir die obigen Überlegungen, so bedeutet dies, dass für jedes Fehlermuster $\mathbf{f} \in GF(2)^n$ mit $t < d/2$ Fehlern das ursprünglich gesendete Codewort $\mathbf{c} \in \mathcal{C}$ als das Codewort mit dem kleinstmöglichen Abstand zum empfangenen Binärwort $\mathbf{r} \in GF(2)^n$ mit $\mathbf{r} = \mathbf{c} + \mathbf{f}$, rekonstruiert wird. Demzufolge kann ein empfangenes Binärwort $r$ solange dem gesendeten Codewort $\mathbf{c} \in \mathcal{C}$ eindeutig zugeordnet werden, wie für die Distanz zu einem beliebigen anderen Codewort $\mathbf{a} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$, gilt:

$$\mathrm{dist}(\mathbf{c}, \mathbf{r}) = \mathrm{wt}(\mathbf{c} + \mathbf{r}) = \mathrm{wt}(\mathbf{f}) \quad < \quad \mathrm{wt}(\mathbf{a} + \mathbf{c} + \mathbf{f}) = \mathrm{wt}(\mathbf{a} + \mathbf{r}) = \mathrm{dist}(\mathbf{a}, \mathbf{r})$$

Damit können alle Fehlermuster $\mathbf{f} \in GF(2)^n$ mit maximal $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehlern eindeutig korrigiert werden. Hierbei bezeichnet $\left\lfloor \frac{d-1}{2} \right\rfloor$ die größte ganze Zahl kleiner oder gleich $\frac{d-1}{2}$. Wir formulieren daher den folgenden Satz zur Fehlerkorrigierbarkeit.

**4.1.19 Satz**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d$. Dann korrigiert der lineare $(n, k, d)$-Blockcode $\mathcal{C}$ genau dann $t$ Fehler, wenn für die Mindestdistanz $d$ gilt*

$$d = 2t + 1 \quad \textit{oder} \quad d = 2t + 2.$$

*Der Parameter $t$ wird dabei im allgemeinen auch als die Fehlerkorrekturfähigkeit eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ bezeichnet.*

**Beweis**
$\Longleftarrow$: Das empfangene Binärwort $\mathbf{r} \in GF(2)^n$ unterscheide sich an höchstens $t$ Stellen von dem gesendeten Codewort $\mathbf{c} \in \mathcal{C}$. Dann gilt $\mathrm{dist}(\mathbf{c}, \mathbf{r}) \leq t$. Das empfangene Binärwort $\mathbf{r} \in GF(2)^n$ liegt somit ausschließlich in der Korrigierkugel $K^t(\mathbf{c})$ um das Codewort $\mathbf{c} \in \mathcal{C}$ und in keiner weiteren Korrigierkugel $K^t(\mathbf{a})$ um ein beliebiges anderes Codewort $\mathbf{a} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$. Daher kann das empfangene Wort $\mathbf{r}$ bei der Fehlerkorrektur eindeutig dem Codewort $\mathbf{c} \in \mathcal{C}$ zugeordnet werden. Würde für ein beliebiges anderes Codewort $\mathbf{a} \in \mathcal{C}, \mathbf{a} \neq \mathbf{c}$, gelten $\mathrm{dist}(\mathbf{a}, \mathbf{r}) \leq t$, so würde die Dreiecksungleichung einen Widerspruch liefern:

$$\mathrm{dist}(\mathbf{a}, \mathbf{c}) \leq \mathrm{dist}(\mathbf{a}, \mathbf{r}) + \mathrm{dist}(\mathbf{r}, \mathbf{c}) \leq t + t = 2t < 2t + 1 \leq d.$$

Mit derselben Argumentation wie im Beweis zu Satz 4.1.16 folgt, dass ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ mit der Mindestdistanz $d = 2t + 1$ oder $d = 2t + 2$ Fehlermuster mit $t + 1$ Fehlern nicht mehr korrigieren kann.

$\Longrightarrow$: Aus der Definition der Mindestdistanz resultiert, dass stets zwei verschiedene Codewörter $\mathbf{a}, \mathbf{c} \in \mathcal{C}$, $\mathbf{a} \neq \mathbf{c}$, mit der Distanz $\text{dist}(\mathbf{a}, \mathbf{c}) = d$ existieren. Angenommen, für die Mindestdistanz $d$ gilt $d = \text{dist}(\mathbf{a}, \mathbf{c}) \leq 2t$, dann wäre es möglich, dass ein empfangenes Binärwort $\mathbf{r} \in GF(2)^n$ genau $t$ Fehler enthält und zu dem gesendeten Codewort $\mathbf{c} \in \mathcal{C}$ und zu dem von $\mathbf{c}$ verschiedenen Codewort $\mathbf{a} \in \mathcal{C}$ dieselbe Distanz $\text{dist}(\mathbf{c}, \mathbf{r}) = \text{dist}(\mathbf{a}, \mathbf{r}) = t$ besitzt und somit nicht eindeutig korrigiert werden kann. Dies ist ein Widerspruch zur Voraussetzung, dass der lineare $(n, k, d)$-Blockcode $\mathcal{C}$ bis zu $t$ Fehler korrigieren kann. Daher gilt zumindest $d \geq 2t + 1$.

Nehmen wir an, dass für die Mindestdistanz $d$ gilt

$$d \geq 2t + 3 = 2(t + 1) + 1,$$

dann erhalten wir mittels der obigen Schlußfolgerungen, dass der betrachtete $(n, k, d)$-Blockcode $\mathcal{C}$ sogar $(t + 1)$ Fehler korrigieren kann, im Widerspruch zur Voraussetzung. □

**4.1.20 Korollar**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ der Länge $n$ und der Dimension $k$ mit der Mindestdistanz $d$. Dann korrigiert dieser lineare $(n, k, d)$-Blockcode $\mathcal{C}$ bis zu $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehler.*

**4.1.21 Korollar**
*Für einen linearen $(n, k, d)$-Blockcode $\mathcal{C}$, der bis zu $t$ Fehler, mit $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$, korrigieren kann, sind die Korrigierkugeln $K^t(\mathbf{c})$ vom Radius $t$ paarweise disjunkt. Ferner kann es höchstens $2^n / \sum_{i=0}^{t} \binom{n}{i}$ Codewörter geben. Da die Dimension von $\mathcal{C}$ gerade gleich $k$ ist, ergibt sich daraus für die Anzahl der Codewörter die Hamming-Schranke:*

$$2^k \leq 2^n / \sum_{i=0}^{t} \binom{n}{i} \quad \Longleftrightarrow \quad 2^n \leq 2^k \sum_{i=0}^{t} \binom{n}{i}$$

**4.1.22 Definition**
*Ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ wird ein perfekter Code oder auch dichtgepackt genannt, wenn jedes Binärwort $\mathbf{r} \in GF(2)^n$ in genau einer Korrigierkugel $K^t(\mathbf{c})$ vom Radius $t$ um ein Codewort $\mathbf{c} \in \mathcal{C}$ liegt:*

$$\forall_{\mathbf{r} \in GF(2)^n} \quad \exists_{\mathbf{c} \in \mathcal{C}} \qquad \mathbf{r} \in K^t(\mathbf{c}).$$

**4.1.23 Korollar**
*Für einen perfekten $(n, k, d)$-Blockcode $\mathcal{C}$, der bis zu $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehler korrigieren kann, sind laut Definition die Korrigierkugeln $K^t(\mathbf{c})$ vom Radius $t$ mit $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ sowohl paarweise disjunkt als auch überdeckend. Die Hamming-Schranke aus Korollar 4.1.21 ist mit Gleichheit erfüllt:*

$$2^n = 2^k \sum_{i=0}^{t} \binom{n}{i}$$

*Darüber hinaus besitzt ein perfekter Code, der bis zu $\lfloor \frac{d-1}{2} \rfloor$ Fehler korrigieren kann, ungerade Mindestdistanz $d = 2t + 1$.*

Nachdem wir die Klasse der linearen Blockcodes eingeführt haben, nutzen wir nun die zusätzliche mathematische Struktur sowie Resultate aus der linearen Algebra zur vollständigen Beschreibung eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$. Laut Definition 4.1.9 bilden die $2^k$ Codewörter eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ einen $k$-dimensionalen Untervektorraum des Vektorraumes $GF(2)^n$ aller Binärwörter der Länge $n$. Die Theorie der linearen Algebra liefert, dass $\mathcal{C}$ als Untervektorraum von $GF(2)^n$ eine Basis bestehend aus $k$ linear unabhängigen Codewörtern $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1} \in \mathcal{C}$ besitzt und dass sich alle übrigen Codewörter $\mathbf{c} \in \mathcal{C}$ als Linearkombinationen dieser Basis schreiben lassen.

**4.1.24 Definition**
*Für einen linearen $(n, k, d)$-Blockcode $\mathcal{C}$ schreiben wir $k$ linear unabhängige Codewörter $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1} \in \mathcal{C}$ mit $\mathbf{g}_i = (g_{i0}, \ldots, g_{i(n-1)})$ für alle $0 \leq i \leq k-1$, die eine Basis des Untervektorraumes $\mathcal{C}$ bilden, in eine $(k \times n)$-Matrix*

$$\mathbf{G} = \begin{pmatrix} g_{00} & \cdots & g_{0(n-1)} \\ \vdots & \ddots & \vdots \\ g_{(k-1)0} & \cdots & g_{(k-1)(n-1)} \end{pmatrix} = \begin{pmatrix} \mathbf{g}_0 \\ \vdots \\ \mathbf{g}_{k-1} \end{pmatrix}$$

*Diese wird als Generatormatrix des linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ bezeichnet.*

**4.1.25 Korollar**
*Da die Zeilen der Generatormatrix $\mathbf{G}$ eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ eine Basis von $\mathcal{C}$ bilden, ist jedes Codewort $\mathbf{c} \in \mathcal{C}$ darstellbar als Linearkombination der Zeilen von $\mathbf{G}$*

$$\forall_{\mathbf{c} \in \mathcal{C}} \quad \exists_{\substack{\lambda_i \in GF(2) \\ 0 \leq i \leq k-1}} \quad \mathbf{c} = \lambda_0 \mathbf{g}_0 + \lambda_1 \mathbf{g}_1 + \cdots + \lambda_{k-1} \mathbf{g}_{k-1} \,.$$

Da ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ somit vollständig durch die $k$ Zeilen einer zugehörigen Generatormatrix $\mathbf{G}$ beschrieben wird, muß zur Codierung der Informationsbits im Codierer keine vollständige Code-Tabelle mehr abgespeichert werden sondern lediglich die $k$ Zeilen der Generatormatrix $\mathbf{G}$. Der Speicheraufwand im Codierer sowie dessen Komplexität verringern sich somit ganz erheblich. Die Generatormatrix liefert eine sehr einfache Methode zur Codierung der zu übermittelnden Information. Dazu interpretieren wir die zu übermittelnden Informationsbits $i_0, \ldots, i_{k-1} \in GF(2)$ als ein *Informationswort* $\mathbf{i} := (i_0, \ldots, i_{k-1}) \in GF(2)^k$. Der Codierer bildet dann die entsprechende Linearkombination der $k$ Zeilen der Generatormatrix $\mathbf{G}$ und ermittelt so das zugehörige Codewort $\mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C}$:

$$\mathbf{c} = \mathbf{i}\,\mathbf{G} = (i_0, \ldots, i_{k-1}) \begin{pmatrix} \mathbf{g}_0 \\ \vdots \\ \mathbf{g}_{k-1} \end{pmatrix} = i_0 \mathbf{g}_0 + i_1 \mathbf{g}_1 + \cdots + i_{k-1} \mathbf{g}_{k-1}$$

Gemäß der Theorie der Linearen Algebra ist die Basis eines $k$-dimensionalen Untervektorraumes nicht eindeutig. Daher lassen sich für einen linearen $(n, k, d)$-Blockcode $\mathcal{C}$ mit

der Generatormatrix $\mathbf{G}$ durch elementare Zeilenumformungen weitere Generatormatrizen $\mathbf{G}'$ konstruieren. Diese erzeugen dann denselben linearen $(n, k, d)$-Blockcode $\mathcal{C}$ und stellen lediglich eine andere Basis von $\mathcal{C}$ dar. Eine Vertauschung der Spalten der Generatormatrix $\mathbf{G}$ impliziert jedoch, dass ein von $\mathcal{C}$ verschiedener linearer $(\tilde{n}, \tilde{k}, \tilde{d})$-Blockcode $\tilde{\mathcal{C}}$ mit anderen Codewörtern erzeugt wird.

**4.1.26 Definition**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ mit der Generatormatrix $\mathbf{G}$. Dann entsteht durch Vertauschung der Spalten der Generatormatrix $\mathbf{G}$ eine Generatormatrix $\tilde{\mathbf{G}}$, die einen von $\mathcal{C}$ verschiedenen linearen $(\tilde{n}, \tilde{k}, \tilde{d})$-Blockcode $\tilde{\mathcal{C}}$ erzeugt. Dieser wird als zu $\mathcal{C}$ äquivalenter Code $\tilde{\mathcal{C}}$ bezeichnet und besitzt dieselbe Codelänge $\tilde{n} = n$ und dieselbe Dimension $\tilde{k} = k$. Bezüglich der Mindestdistanz $\tilde{d}$ kann zunächst keine Aussage gemacht werden.*

Detailliertere Ausführungen zur Erzeugung äquivalenter Codes sowie ausführlichere Begründungen für die hier aufgezeigten Zusammenhänge finden sich in einigen Standardlehrbüchern der Informations- bzw. der Codierungstheorie, wie beispielsweise [7, 26, 43, 47]. Für die Zielsetzung dieser Arbeit sind äquivalente Codes lediglich aufgrund des folgenden Satzes von Interesse, den wir hier nur angeben und für dessen Beweis wir auf [26, 43, 47] verweisen.

**4.1.27 Satz**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ mit der Generatormatrix $\mathbf{G}$. Dann existiert ein zu $\mathcal{C}$ äquivalenter linearer $(n, k, \tilde{d})$-Blockcode $\tilde{\mathcal{C}}$ der Länge $n$ und der Dimension $k$, der systematisch ist.*

Gemäß seiner Definition besitzt ein systematischer linearer $(n, k, d)$-Blockcode $\mathcal{C}$ die Eigenschaft, dass alle Codewörter $\mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C}$ in einen Informations- und einen Kontrollteil zerfallen. Dabei besteht der Informationsteil $(c_0, \ldots, c_{k-1})$ aus $k$ unveränderten Informationsbits $i_0, \ldots, i_{k-1} \in GF(2)^n$ und der Kontrollteil aus $n-k$ Kontrollbits $c_k, \ldots, c_{n-1} \in GF(2)^n$, die sich aus den Informationsbits $i_0, \ldots, i_{k-1}$ berechnen lassen. Dies legt die Vermutung nahe, dass die Generatormatrix $\mathbf{G}$ eines systematischen linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ die folgende Gestalt besitzt.

**4.1.28 Satz**
*Für einen systematischen linearen $(n, k, d)$-Blockcode $\mathcal{C}$ existieren $k$ linear unabhängige Codewörter $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1} \in \mathcal{C}$ derart, dass die Generatormatrix $\mathbf{G}$ als linke Teilmatrix die $(k \times k)$-Einheitsmatrix $\mathbf{I}_{k\times k}$ enthält*

$$\mathbf{G} = \left(\begin{array}{ccc|ccc} 1 & \cdots & 0 & \tilde{g}_{0k} & \cdots & \tilde{g}_{0(n-1)} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & \tilde{g}_{(k-1)k} & \cdots & \tilde{g}_{(k-1)(n-1)} \end{array}\right) = \left(\mathbf{I}_{k\times k} \,|\, \tilde{\mathbf{G}}_{k\times(n-k)}\right)$$

*Eine Generatormatrix $\mathbf{G}$ von dieser Form wird als Generatormatrix in Standardform bezeichnet.*

**Beweis**
Um zu überprüfen, ob die angegebene Matrix $\mathbf{G}$ eine Generatormatrix für einen systematischen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ darstellt, bestimmen wir für beliebige Informationsbits $i_0,\ldots,i_{k-1} \in GF(2)$ das zugehörige Codewort $\mathbf{c} \in \mathcal{C}$ mittels der Multiplikation des Informationswortes $\mathbf{i} = (i_0,\ldots,i_{k-1})$ mit der angegebenen Matrix $\mathbf{G}$:

$$\begin{aligned}
\mathbf{c} = (c_0,\ldots,c_{n-1}) &= \mathbf{i}\,\mathbf{G} \\
&= (i_0,\ldots,i_{k-1})\big(\mathbf{I}_{k\times k} \mid \tilde{\mathbf{G}}_{k\times(n-k)}\big) \\
&= (i_0,\ldots,i_{k-1}) \left(\begin{array}{ccc|ccc} 1 & \cdots & 0 & \tilde{g}_{0k} & \cdots & \tilde{g}_{0(n-1)} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & \tilde{g}_{(k-1)k} & \cdots & \tilde{g}_{(k-1)(n-1)} \end{array}\right)
\end{aligned}$$

Für die Komponenten des zugehörigen Codewortes $\mathbf{c} \in \mathcal{C}$ ergeben sich die folgenden Gleichungen:

$$\begin{aligned}
c_j &= i_j && \text{für alle } 0 \le j \le k-1 \\
c_j &= i_0\,\tilde{g}_{0j} + \cdots + i_{k-1}\,\tilde{g}_{(k-1)j} && \text{für alle } k \le j \le n-1
\end{aligned}$$

Demzufolge zerfällt das zugehörige Codewort $\mathbf{c} = (i_0,\ldots,i_{k-1},c_k,\ldots,c_{n-1})$ in einen Informationsteil, der die $k$ Informationsbits unverändert enthält, und einen Kontrollteil bestehend aus $n-k$ Kontrollbits, die sich als Linearkombinationen der Zeilen der Generatormatrix schreiben lassen. □

Aus der Theorie der linearen Algebra können wir für einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ neben der Existenz der Generatormatrix $\mathbf{G}$ die Existenz einer weiteren, in diesem Zusammenhang sehr nützlichen und gebräuchlichen Matrix ableiten. Wir zeigen, dass auch diese Matrix den von $\mathbf{G}$ erzeugten linearen $(n,k,d)$-Blockcode $\mathcal{C}$ vollständig bestimmt und aufgrund dessen mit $\mathcal{C}$ assoziiert werden kann.

**4.1.29 Definition**
*Für einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ mit einer $(k\times n)$-Generatormatrix $\mathbf{G}$ wird eine $((n-k)\times n)$-Matrix $\mathbf{H}$ als Prüf- oder auch Kontrollmatrix des linearen Codes $\mathcal{C}$, bezeichnet, wenn deren $n-k$ Zeilen linear unabhängig sind und es gilt:*

$$\mathbf{G}\mathbf{H}^T = \mathbf{0}$$

*Hierbei symbolisiert $\mathbf{0}$ die Nullmatrix und $\mathbf{H}^T$ die zu $\mathbf{H}$ transponierte Matrix.*

**4.1.30 Korollar**
*Für einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ stellen jeweils die Zeilen der Generatormatrix $\mathbf{G}$ und die Zeilen der Prüfmatrix $\mathbf{H}$ die Basis von zwei zueinander orthogonalen Untervektorräumen von $GF(2)^n$ dar. Daher ist jedes Codewort des von der Generatormatrix $\mathbf{G}$ erzeugten linearen Codes $\mathcal{C}$ orthogonal zu jedem Binärwort, das durch die Zeilen der Prüfmatrix $\mathbf{H}$ erzeugt wird.*

Aufgrund dessen können wir einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$, der durch eine $(k\times n)$-Generatormatrix $\mathbf{G}$ erzeugt wird, alternativ auch folgendermaßen charakterisieren.

**4.1.31 Satz**
*Gegeben sei ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$ mit der Generatormatrix $\mathbf{G}$. Dann ist ein Binärwort $\mathbf{c} \in GF(2)^n$ genau dann ein Codewort $\mathbf{c} \in \mathcal{C}$, wenn gilt*

$$\mathbf{c}\,\mathbf{H}^T = \mathbf{0}.$$

**Beweis**
Die Definition der Prüfmatrix liefert $\mathbf{G}\mathbf{H}^T = \mathbf{0}$. Für jedes Codewort $\mathbf{g}_i \in \mathcal{C}$ mit $0 \leq i \leq k-1$, das als eine Zeile der Generatormatrix $\mathbf{G}$ den linearen Code $\mathcal{C}$ aufspannt, gilt insbesondere $\mathbf{g}_i\,\mathbf{H}^T = \mathbf{0}$. Die $k$ linear unabhängigen Codewörter $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1} \in \mathcal{C}$ bilden somit eine Basis der $k$-dimensionalen Lösungsmenge des linearen Gleichungssystem:

$$\mathbf{x}\,\mathbf{H}^T = \mathbf{0}.$$

Die gesamte Lösungsmenge setzt sich dann aus allen Linearkombinationen der Codewörter $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1}$ und folglich aus allen Codewörtern $\mathbf{c} \in \mathcal{C}$ zusammen. Von daher dient die Prüfmatrix $\mathbf{H}$ somit der Überprüfung der Zugehörigkeit eines beliebigen Binärwortes zu dem betrachteten linearen $(n,k,d)$-Blockcode $\mathcal{C}$. □

**4.1.32 Bemerkung**
Ist die Generatormatrix $\mathbf{G}$ eines systematischen linearen $(n,k,d)$-Blockcodes $\mathcal{C}$ in Standardform $\mathbf{G} = \left(\mathbf{I}_{k\times k} \,|\, \tilde{\mathbf{G}}_{k\times(n-k)}\right)$ gegeben, so existiert auch für die Prüfmatrix $\mathbf{H}$ des systematischen linearen Codes $\mathcal{C}$ eine Standardform. Diese enthält als rechte Teilmatrix die $((n-k)\times(n-k))$-Einheitsmatrix $\mathbf{I}_{(n-k)\times(n-k)}$

$$\mathbf{H} = \left(\begin{array}{ccc|ccc} \tilde{h}_{00} & \cdots & \tilde{h}_{0(k-1)} & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \tilde{h}_{(n-k-1)0} & \cdots & \tilde{h}_{(n-k-1)(k-1)} & 0 & \cdots & 1 \end{array}\right) = \left(\tilde{\mathbf{H}}_{(n-k)\times k} \,|\, \mathbf{I}_{(n-k)\times(n-k)}\right)$$

und die $((n-k)\times k)$-Matrix $\tilde{\mathbf{H}}_{(n-k)\times k}$ wird derart definiert, dass gilt:

$$\tilde{\mathbf{H}}_{(n-k)\times k} = (\tilde{\mathbf{G}}_{k\times(n-k)})^T$$

**Beweis**
Die angegebenen Matrizen $\mathbf{G}$ und $\mathbf{H}$ müssen die Bedingung aus Definition 4.1.29 erfüllen:

$$\begin{aligned} \mathbf{G}\mathbf{H}^T &= \left(\mathbf{I}_{k\times k} \,|\, \tilde{\mathbf{G}}_{k\times(n-k)}\right)\left(\tilde{\mathbf{H}}_{(n-k)\times k} \,|\, \mathbf{I}_{(n-k)\times(n-k)}\right)^T \\ &= \left(\mathbf{I}_{k\times k} \,|\, \tilde{\mathbf{G}}_{k\times(n-k)}\right)\left((\tilde{\mathbf{G}}_{k\times(n-k)})^T \,|\, \mathbf{I}_{(n-k)\times(n-k)}\right)^T = \mathbf{0} \end{aligned}$$

Folglich ist die Matrix $\mathbf{H} = \left(\tilde{H}_{(n-k)\times k} \,|\, \mathbf{I}_{(n-k)\times(n-k)}\right)$ mit $\tilde{\mathbf{H}}_{(n-k)\times k} = (\tilde{\mathbf{G}}_{k\times(n-k)})^T$ eine Prüfmatrix des durch die Generatormatrix $\mathbf{G}$ erzeugten systematischen linearen $(n,k,d)$-Blockcodes $\mathcal{C}$. □

**4.1.33 Bemerkung**
Darüber hinaus kann die Prüfmatrix $\mathbf{H}$ eines systematischen linearen $(n,k,d)$-Blockcodes $\mathcal{C}$ auch zur Berechnung der Kontrollstellen eines Codewortes $\mathbf{c} \in \mathcal{C}$ herangezogen werden. Gegeben seien die zu übermittelnden Informationsbits $i_0, \ldots, i_{k-1} \in GF(2)$. Dann gilt für ein Codewort $\mathbf{c} = (c_0, \ldots, c_{n-1})$ gerade $c_j = i_j$ für alle $0 \leq j \leq k-1$. Da nun $\mathbf{H} = \big((\tilde{\mathbf{G}}_{k\times(n-k)})^T \,|\, \mathbf{I}_{(n-k)\times(n-k)}\big)$ist, lassen sich aus der Bedingung

$$\mathbf{c}\,\mathbf{H}^T = \mathbf{0}$$

entsprechende Gleichungen zur Berechnung der Kontrollstellen $c_k, \ldots, c_{n-1}$ des Codwortes $\mathbf{c} \in \mathcal{C}$ ableiten:

$$c_j \;=\; i_0\,\tilde{g}_{0j} + \cdots + i_{k-1}\,\tilde{g}_{(k-1)j} \quad \text{für alle } k \leq j \leq n-1.$$

Damit haben wir gezeigt, dass ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$ durch seine Generator- und ebenso durch seine Prüfmatrix vollständig beschrieben wird. Ausführlichere Erklärungen und weitergehende Zusammenhänge werden in [7, 16, 26, 47] besprochen. Wir halten zusammenfassend fest:

**4.1.34 Satz**
*Die Parameter eines linearen Blockcodes $\mathcal{C}$, wie beispielsweise die Codelänge, die Dimension und die Mindestdistanz, lassen sich direkt aus der zugehörigen Generator- bzw. Prüfmatrix ableiten.*

**Beweis**

(i) Besitzt die Generatormatrix $\mathbf{G}$ die Dimension $(k \times n)$ und die zugehörige Prüfmatrix $\mathbf{H}$ die Dimension $((n-k)\times n)$, dann entspricht die Codelänge des betrachteten linearen Codes $\mathcal{C}$ der Anzahl der Spalten von $\mathbf{G}$ bzw. $\mathbf{H}$ und ist demnach gleich $n$, vgl. Definition 4.1.24 und Definition 4.1.29.

(ii) Die Anzahl der Zeilen der Generatormatrix $\mathbf{G}$ gibt die Anzahl $k$ der Informationsstellen und somit die Dimension des linearen Codes $\mathcal{C}$ an. Demnach entspricht die Anzahl der Zeilen von $\mathbf{H}$ der Anzahl $n-k$ der Kontrollstellen von $\mathcal{C}$.

(iii) Die Mindestdistanz $d$ des linearen Codes $\mathcal{C}$ läßt sich aus der Prüfmatrix $\mathbf{H}$ ermitteln als die kleinste ganze Zahl $d'$, für die es $d'$ linear abhängige Spalten in $\mathbf{H}$ gibt.

$\Longleftarrow$: Angenommen, $d'$ der Spalten $\mathbf{h}_0, \ldots, \mathbf{h}_{n-1}$ der Prüfmatrix $\mathbf{H}$ sind linear abhängig. Dann existieren Koeffizienten $c_0, \ldots, c_{n-1}$, von denen genau $d'$ ungleich Null sind und für die gilt:

$$\begin{aligned} & c_0\mathbf{h}_0 + c_1\mathbf{h}_1 + \cdots + c_{n-1}\mathbf{h}_{n-1} \;=\; \mathbf{0} \\ \Longleftrightarrow \quad & \mathbf{c}\mathbf{H}^T \;=\; \mathbf{0} \\ \overset{4.1.31}{\Longleftrightarrow} \quad & \mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C} \end{aligned}$$

Da $d'$ die kleinste ganze Zahl ist, für die $d'$ Spalten in $\mathbf{H}$ linear abhängig sind, ist $\mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C}$ ein Codewort mit dem Gewicht $\mathrm{wt}(\mathbf{c}) = d' \leq d$.

$\Longrightarrow$: Umgekehrt gilt, falls $\mathbf{c} \in \mathcal{C}$ ein beliebiges Codewort mit dem Gewicht $\mathrm{wt}(\mathbf{c}) = d' \geq d$ ist, dass $\mathbf{c}\,\mathbf{H}^T = \mathbf{0}$ ist. Damit sind $d$ Spalten von $\mathbf{H}$ linear abhängig. Es folgt, dass die Mindestdistanz $d$ des linearen Codes $\mathcal{C}$ gleich $d'$ und daher gleich der kleinsten ganzen Zahl $d'$ ist, für die $d'$ Spalten von $\mathbf{H}$ linear abhängig sind. □

Mit der Generator- und der Prüfmatrix eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ haben wir zwei Möglichkeiten zur Charakterisierung linearer Codes vorgestellt. Wir befassen uns nun abschließend mit der Konstruktion neuer linearer Codes aus bereits bekannten Codes. Dazu geben wir drei verschiedene Techniken an, die *Erweiterung* und die *Verkürzung* eines gegebenen linearen Codes $\mathcal{C}$ sowie die Bildung des *zu* $\mathcal{C}$ *dualen Codes*. So ist es mit Blick auf die Definition der Prüfmatrix $\mathbf{H}$ eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$ naheliegend, den linearen Untervektorraum des Vektorraumes $GF(2)^n$, der von den $n - k$ linear unabhängigen Zeilen der Prüfmatrix $\mathbf{H}$ aufgespannt wird, als einen linearen Code zu interpretieren.

**4.1.35 Definition**
*Gegeben sei ein linearer $(n, k, d)$-Blockcodes $\mathcal{C}$ mit der $((n-k) \times n)$-Prüfmatrix $\mathbf{H}$. Dann spannen die $n-k$ linear unabhängigen Zeilen der Prüfmatrix $\mathbf{H}$ einen $(n-k)$-dimensionalen Untervektorraum von $GF(2)^n$ auf und bilden somit eine Basis eines linearen Blockcodes. Dieser wird als der zu $\mathcal{C}$ duale Code $\mathcal{C}^\perp$ bezeichnet.*

**4.1.36 Korollar**
*Gegeben sei ein linearer $(n, k, d)$-Blockcodes $\mathcal{C}$ mit der Generatormatrix $\mathbf{G}$ und der Prüfmatrix $\mathbf{H}$. Wird die Generatormatrix des zu $\mathcal{C}$ dualen Codes $\mathcal{C}^\perp$ mit $\mathbf{G}^\perp$ und die Prüfmatrix mit $\mathbf{H}^\perp$ bezeichnet, dann liefert Definition 4.1.35:*

$$\mathbf{G}^\perp = \mathbf{H} \quad \textit{und} \quad \mathbf{H}^\perp = \mathbf{G}.$$

*Demzufolge bildet der zu $\mathcal{C}$ duale Code $\mathcal{C}^\perp$ den Nullraum des von der Generatormatrix $\mathbf{G}$ erzeugten linearen Codes $\mathcal{C}$. Darüber hinaus ist der zu $\mathcal{C}$ duale Code $\mathcal{C}^\perp$ linear und besitzt gemäß Satz 4.1.34 die Parameter:*

$$\begin{aligned} \textit{Codelänge } n^\perp &= n \\ \textit{Dimension } k^\perp &= n - k \\ \textit{Mindestdistanz } d^\perp &= \textit{kleinste ganze Zahl } d', \textit{ für die} \\ &\quad\; d' \textit{ Spalten von } \mathbf{G} \textit{ linear abhängig sind} \end{aligned}$$

Eine weitere Methode, aus einem gegebenen linearen $(n, k, d)$-Blockcode $\mathcal{C}$ einen neuen linearen Code zu konstruieren, ist die *Erweiterung*. Hierzu werden jedem Codewort $\mathbf{c} \in \mathcal{C}$ eine oder mehrere Stellen hinzugefügt. Dabei wird in der Regel bei der Erweiterung um eine Stelle ein sogenanntes *parity check Bit* angehängt.

**4.1.37 Definition**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$. Dann ist der um eine Stelle erweiterte Code $\hat{\mathcal{C}}$ definiert durch*

$$\hat{\mathcal{C}} := \left\{ (c_0, \ldots, c_{n-1}, c_n) \in GF(2)^{n+1} \;;\; (c_0, \ldots, c_{n-1}) \in \mathcal{C} \textit{ und } c_n := \sum_{j=0}^{n-1} c_j \bmod 2 \right\}$$

**4.1.38 Korollar**
*Aus der vorangehenden Definition folgt sofort, dass der um eine Stelle erweiterte Code $\hat{\mathcal{C}}$ linear ist und laut Satz 4.1.34 die folgenden Parameter besitzt:*

$$\begin{aligned} \textit{Codelänge } \hat{n} &= n+1 \\ \textit{Dimension } \hat{k} &= k \\ \textit{Mindestdistanz } \hat{d} &= \begin{cases} d & ; \quad d \textit{ gerade} \\ d+1 & ; \quad d \textit{ ungerade} \end{cases} \end{aligned}$$

Der umgekehrte Prozeß zur Erweiterung eines linearen $(n,k,d)$-Blockcodes $\mathcal{C}$ ist die *Verkürzung* des linearen Codes $\mathcal{C}$ um eine oder mehrere Stellen. Hierbei unterscheiden wir zwei unterschiedliche Verfahren. Auf der einen Seite erreichen wir eine Verkürzung eines gegebenen linearen $(n,k,d)$-Blockcodes $\mathcal{C}$ durch eine sogenannte *Punktierung* der Codewörter, andererseits durch die Streichung einer oder mehrerer Stellen in jedem Codewort $\mathbf{c} \in \mathcal{C}$. Die Punktierung von Codewörtern wird allerdings erst im Zusammenhang mit Faltungscodes interessant, so dass wir uns an dieser Stelle lediglich auf das folgende Verfahren konzentrieren.

**4.1.39 Definition**
*Gegeben sei ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$. Dann erhalten wir einen verkürzten Code $\mathcal{C}^-$, indem wir nur diejenigen Codewörter $\mathbf{c} \in \mathcal{C}$ auswählen, die an einer oder an mehreren fest vorgegebenen Stellen einen vorgegebenen Wert besitzen. Dann streichen wir die entsprechende Stellenzahl in allen Codewörtern. Beispielsweise wählen wir nur diejenigen Codewörter $\mathbf{c} \in \mathcal{C}$ aus, die an den ersten $x$ Stellen gleich Null sind, und streichen genau diese $x$ Stellen, wobei $0 < x < k$ gilt.*

**4.1.40 Korollar**
*Praktisch bedeutet die Verkürzung eines linearen $(n,k,d)$-Blockcodes $\mathcal{C}$, dass ein Teil der Informationsbits zu Null gesetzt wird und daher die entsprechende Anzahl Stellen gestrichen werden kann. Ein verkürzter Blockcode $\mathcal{C}^-$ ist ebenfalls linear und besitzt laut Satz 4.1.34 die folgenden Parameter, sofern die ersten $x$ Stellen gestrichen werden:*

$$\begin{aligned} \textit{Codelänge } n^- &= n-x \\ \textit{Dimension } k^- &= k-x \\ \textit{Mindestdistanz } d^- &= d \end{aligned}$$

## 4.1.2 Zyklische Codes

In diesem Abschnitt befassen wir uns mit einer speziellen und sehr wichtigen Klasse von linearen Blockcodes, den sogenannten *zyklischen Codes*. Die meisten linearen Blockcodes, die in der Praxis zur Fehlerkorrektur eingesetzt werden, besitzen wesentlich mehr mathematische Struktur als nur die eines Untervektorraumes. Neben einem großen theoretischen Interesse, das von der zusätzlichen, reichhaltigen mathematischen Struktur herrührt, kommt der Klasse der zyklischen Codes auch aus einem praktischen Blickwinkel betrachtet eine besondere Bedeutung zu. So kann die Codierung und Decodierung zyklischer Codes mittels sogenannter Schieberegister äußerst effizient implementiert werden. Eine ausführliche und

sehr detaillierte Behandlung linearer zyklischer Blockcodes und deren Eigenschaften findet sich in den Lehrbüchern von [4, 7, 26, 34, 36, 37, 43, 47], auf die wir uns bei den nachstehenden Ausführungen beziehen.

**4.1.41 Definition**
*Ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ heißt ein zyklischer Code, wenn jede zyklische Verschiebung eines Codewortes $\mathbf{c} \in \mathcal{C}$ wieder ein Codewort aus $\mathcal{C}$ ist:*

$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-2}, c_{n-1}) \in \mathcal{C} \quad \Longrightarrow \quad \mathbf{c}_z := (c_{n-1}, c_0, \ldots, c_{n-2}) \in \mathcal{C}$$

*Dabei sind zyklische Verschiebungen eines Codewortes um eine beliebige Anzahl von Stellen in Form einer mehrfachen Verschiebung um eine Stelle mit eingeschlossen.*

Zur Beschreibung zyklischer Codes und der Herleitung ihrer algebraischen Eigenschaften ist es zweckmäßig die aus der Linearen Algebra bekannte *Polynomdarstellung* eines Binärwortes zu verwenden. Das bedeutet, dass die Komponenten eines Binärwortes aus $GF(2)^n$ als Koeffizienten eines Polynoms über $GF(2)$ betrachtet werden. Entsprechend der Bezeichnungsweise in den meisten Lehrbüchern der linearen Algebra verwenden wir dazu die folgende Notation.

**4.1.42 Bezeichnungen**

*(i)* *Mit $GF(2)[z]$ bezeichnen wir den Polynomring über dem Körper $GF(2)$.*

*(ii)* *Für ein normiertes Polynom $p(z) \in GF(2)[z]$ vom Grad $r \geq 1$ zerfällt die Menge aller Polynome über $GF(2)$ bezüglich der Äquivalenzrelation mod $p(z)$ in die Restklassen*

$$\overline{a(z)} := \{x(z) \in GF(2)[z] \; ; \; x(z) = a(z) \bmod p(z)\},$$

*wobei $\overline{a(z)}$ als Repräsentant der entsprechenden Restklasse bezeichnet wird.*

*(iii)* *Werden die einzelnen Restklassen jeweils mit den Repräsentanten kleinsten Grades bezeichnet, so läßt sich die Menge aller Restklassen $GF(2)[z]/p(z)$ wie folgt charakterisieren:*

$$GF(2)[z]\big|_{p(z)} := GF(2)[z]/p(z) = \{\overline{a(z)} \; ; \; a(z) \in GF(2)[z] \, , \, grad(a) < r\}$$

*Die Menge aller Restklassen $GF(2)[z]/p(z)$ sowohl versehen mit der Vektorraum- als auch mit der Ringstruktur heißt die Algebra $GF(2)[z]\big|_{p(z)}$ der Polynome modulo $p(z)$.*

*(iv)* *Für jedes Binärwort $\mathbf{a} = (a_0, \ldots, a_{n-1}) \in GF(2)^n$ existiert eine Polynomdarstellung $a(z)$ vom Grad $n-1$ derart, dass gilt:*

$$a(z) := a_0 + a_1 \, z + \cdots a_{n-1} \, z^{n-1}$$

Zur Charakterisierung zyklischer Blockcodes identifizieren wir den Vektorraum $GF(2)^n$ aller Binärwörter der Länge $n$ im Sinne der vorangegangenen Bezeichnungen mit der *Algebra $GF(2)[z]/(z^n - 1)$ der Polynome über $GF(2)$ modulo $z^n - 1$*.

**4.1.43 Bemerkung**

(i) Jedem Codewort $\mathbf{c} \in \mathcal{C}$ eines zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ entspricht genau ein Polynom $c(z) = c_0 + c_1\,z + \cdots + c_{n-1}\,z^{n-1}$ vom Grad $\leq n-1$, das als *Codewortpolynom* bezeichnet wird. Die Begriffe Codewort und Codewortpolynom sind in diesem Sinne austauschbar, so dass wir der Einfachheit halber auch davon sprechen, dass ein Codewortpolynom $c(z)$ ein Element des zyklischen Codes $\mathcal{C}$ ist, also $c(z) \in \mathcal{C}$.

(ii) Ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$ ist genau dann zyklisch, wenn für jedes Codewortpolynom $c(z) = c_0 + c_1\,z + \cdots + c_{n-1}\,z^{n-1} \in \mathcal{C}$ das Polynom

$$z\,c(z) = c_{n-1} + c_0\,z + c_1\,z^2 + \cdots + c_{n-2}\,z^{n-1} \;\in GF(2)[z]\big|_{z^n-1}$$

wieder ein Codewort ist, d.h. $z\,c(z) \in \mathcal{C}$.

(iii) Mit der Interpretation aus (ii) ist ein Untervektorraum $\mathcal{C}$ der Algebra $GF(2)[z]/(z^n-1)$ der Polynome modulo $z^n-1$ genau dann ein zyklischer Code, wenn $z\,\mathcal{C} = \mathcal{C}$ gilt. Damit stimmt die Menge der zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ mit der Menge der Ideale der Algebra $GF(2)[z]/(z^n-1)$ der Polynome über $GF(2)$ modulo $z^n-1$ überein.

Das folgende Theorem über die sogenannten *Generatorpolynome* zyklischer Blockcodes ist für die Theorie der zyklischen Codes von fundamentaler Bedeutung. Sämtliche algebraischen Eigenschaften zyklischer Blockcodes lassen sich auf dieses grundlegende Theorem zurückführen. Wir geben an dieser Stelle nur eine knappe Beweisskizze an und verweisen für einen ausführlicheren Nachweis auf die Ausführungen in [16, 18, 26].

**4.1.44 Theorem**

*Gegeben sei ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$. Dann existiert genau ein normiertes Codewortpolynom $g(z) \in \mathcal{C}$ vom Grad $n-k$ der Gestalt*

$$g(z) = 1 + g_1 z + \cdots + g_{n-k-1} z^{n-k-1} + z^{n-k},$$

*so dass ein binäres Polynom $c(z) \in GF(2)[z]/(z^n-1)$ genau dann ein Codewort des zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ ist, wenn $c(z)$ ein Vielfaches von $g(z)$ ist:*

$$c(z) \in \mathcal{C} \quad \Longleftrightarrow \quad c(z) \bmod g(z) = 0.$$

**Beweisskizze**

Zum Beweis dieses elementaren Theorems sind die folgenden Teilaussagen zu zeigen:

(i) In einem zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ ist ein vom Nullpolynom verschiedenes Codewortpolynom $g(z) \in \mathcal{C}$, das minimalen Grad besitzt, eindeutig bestimmt.

(ii) In einem zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ sei $g(z) \in \mathcal{C}$ das vom Nullpolynom verschiedene und eindeutig bestimmte Codewortpolynom minimalen Grades, wobei $grad(g) = r < n$ gilt. Dann besitzt das Codewortpolynom $g(z)$ die Gestalt

$$g(z) = 1 + g_1\,z + g_2\,z^2 + \cdots + g_{r-1}\,z^{r-1} + z^r$$

(iii) In einem zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ sei $g(z) \in \mathcal{C}$ das vom Nullpolynom verschiedene und eindeutig bestimmte Codewortpolynom minimalen Grades der Gestalt $g(z) = 1 + g_1\, z + \cdots + g_{r-1}\, z^{r-1} + z^r$ und $grad(g) = r < n$. Dann gilt die folgende Äquivalenzaussage:

Ein Polynom $c(z) \in GF(2)[z]/(z^n - 1)$ vom Grad $< n$ ist genau dann ein Codewortpolynom, wenn $c(z)$ ein Vielfaches von $g(z)$ ist, wenn also gilt:

$$c(z) = a(z)g(z) \quad \text{mit} \quad a(z) \in GF(2)[z]\Big|_{z^n-1}$$

(iv) Da insgesamt $2^{n-r}$ Polynome vom Grad $< n$ existieren, die Vielfache von $g(z)$ sind und die im Sinne von Punkt (iii) jeweils ein Codewortpolynom des zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ darstellen, und da der zyklische $(n,k,d)$-Blockcode $\mathcal{C}$ gerade die Dimension $k$ besitzt, gilt folglich:

$$2^k = 2^{n-r} \quad \Longrightarrow \quad r = n - k$$

und das Polynom $g(z)$ ist von der angegebenen Gestalt

$$g(z) = 1 + g_1\, z + g_2\, z^2 + \cdots + g_{n-k-1}\, z^{n-k-1} + z^{n-k}$$

□

Für einen zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ liefert das Generatorpolynom $g(z)$ eine sehr einfache Methode zur Codierung der zu übermittelnden Information. Sind die Informationsbits $i_0, \ldots, i_{k-1} \in GF(2)$ gegeben, so interpretieren wir diese als Koeffizienten eines Polynoms $i(z) = i_0 + i_1\, z + \cdots + i_{k-1}\, z^{k-1}$. Mittels der Multiplikation von $i(z)$ mit dem Generatorpolynom $g(z)$ erhalten wir das zugehörige Codewortpolynom $c(z) \in \mathcal{C}$ mit

$$c(z) = i(z)\, g(z).$$

**4.1.45 Bemerkung**
Da nach Theorem 4.1.44 jedes Codewortpolynom $c(z)$ eines zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ als Vielfaches des vom Nullpolynom verschiedenen und eindeutig bestimmten Polynoms $g(z)$ minimalen Grades geschrieben werden kann, ist ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ damit vollständig durch das Polynom $g(z) = 1 + g_1\, z + \cdots + g_{n-k-1}\, z^{n-k-1} + z^{n-k}$ bestimmt. Dieses Polynom wird daher als *Generatorpolynom* bezeichnet und legt durch seine Koeffizienten ein sogenanntes *Generatormuster* $\mathbf{g}$ mit

$$\mathbf{g} = (1, g_1, \ldots, g_{n-k-1}, 1, 0, \ldots, 0) \in \mathcal{C}$$

fest. Dieses Generatormuster erzeugt im Sinne von Definition 4.1.41 mit seinen zyklischen Verschiebungen den gesamten zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$. Ferner entspricht die Anzahl $n-k$ der Prüfstellen des zyklischen Codes $\mathcal{C}$ dem Grad des Generatorpolynoms $g(z)$.

Es stellt sich die Frage, ob für jedes $n$ und jedes $k$, mit $k < n$, stets ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ existiert. Die beiden folgenden Theoreme geben an, unter welchen Bedingungen ein zyklischer Code existiert. Wir verzichten auch hier auf die Beweise und verweisen einmal mehr auf die Literatur, [26].

**4.1.46 Theorem**
*Das Generatorpolynom $g(z)$ eines zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ ist stets ein Teiler des Polynoms $z^n-1$, d.h. es gilt:*

$$(z^n-1) \bmod g(z) = 0.$$

**4.1.47 Theorem**
*Es existiert genau dann ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$, wenn das Polynom $z^n-1$ einen echten Teiler $g(z)$ vom Grad $grad(g) = n-k$ besitzt, der als Generatorpolynom dann einen zyklischen Code erzeugt.*

Für einen zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z)$ lassen sich die Generatormatrix $\mathbf{G}$ und die Prüfmatrix $\mathbf{H}$ sehr einfach herleiten.

**4.1.48 Bemerkung**
Gegeben sei nun ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z) = 1+g_1\,z+\cdots+g_{n-k-1}\,z^{n-k-1}+z^{n-k}$. Dann ist durch die Koeffizienten von $g(z)$ laut Bemerkung 4.1.45 das Generatormuster $\mathbf{g} \in \mathcal{C}$ mit $\mathbf{g} = (1, g_1, \ldots, g_{n-k-1}, 1, 0, \ldots, 0)$ festgelegt. Die Polynome

$$z\,g(z), z^2 g(z), \ldots, z^{k-1} g(z) \in GF(2)[z]\big|_{z^n-1}$$

liefern jeweils eine zyklische Verschiebung des Generatormusters $\mathbf{g}$ um eine Stelle. Darüber hinaus sind diese Polynome in der Algebra $GF(2)[z]/(z^n-1)$ der Polynome modulo $z^n-1$ als Polynome paarweise verschiedenen Grades linear unabhängig und bilden, da sie alle vom Grad $< n$ sind, eine Basis des zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$. Von daher erzeugen diese Polynome bzw. die zyklischen Verschiebungen des Generatormusters $\mathbf{g}$ den gesamten zyklischen Code $\mathcal{C}$. Die $(k \times n)$-Generatormatrix $\mathbf{G}$ des zyklischen Codes $\mathcal{C}$ ist dann von der Form

$$\mathbf{G} = \begin{pmatrix} 1 & g_1 & \cdots & g_{n-k-1} & 1 & 0 & \cdots & 0 \\ 0 & 1 & g_1 & \cdots & g_{n-k-1} & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & g_1 & \cdots & g_{n-k-1} & 1 \end{pmatrix}$$

Analog zur Definition der Prüfmatrix $\mathbf{H}$ eines linearen $(n,k,d)$-Blockcodes $\mathcal{C}$ können wir das sogenannte *Prüfpolynom* eines zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ erklären und den zyklischen Code $\mathcal{C}$ entsprechend anhand des Prüfpolynoms charakterisieren.

**4.1.49 Definition**
*Gegeben sei ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z)$. Da das Generatorpolynom $g(z)$ ein Teiler des Polynoms $z^n-1$ ist, existiert auch ein Polynom $h(z) := 1 + h_1\,z + \cdots + h_{k-1}\,z^{k-1} + z^k \in GF(2)^n[z]/(z^n-1)$ vom Grad $k$ mit*

$$h(z)\,g(z) = z^n - 1$$

*Dieses Polynom $h(z)$ vom Grad $k$ heißt das Prüf- oder Kontrollpolynom des zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$.*

**4.1.50 Korollar**
*Ein binäres Polynom $c(z) \in GF(2)[z]/(z^n-1)$ ist genau dann ein Codewort des zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$, wenn das Produkt $c(z)\,h(z)$ verschwindet, wenn also gilt:*

$$c(z)\,h(z) \bmod (z^n-1) = 0.$$

Mittels des Kontrollpolynoms wird demnach überprüft, ob ein beliebiges Polynom vom Grad $< n$ zu dem betrachteten zyklischen Code gehört. Darüber hinaus zeigt das vorangegangene Korollar, dass ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ durch sein Generator- und ebenso durch sein Prüfpolynom vollständig bestimmt ist. Die nachfolgende Bemerkung veranschaulicht, wie aus dem Prüfpolynom $h(z)$ eines zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ die Prüfmatrix $\mathbf{H}$ abgeleitet werden kann.

**4.1.51 Bemerkung**
Gegeben sei ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z)$ und dem zugehörigen Prüfpolynom $h(z) = 1 + h_1\,z + \cdots + h_{k-1}\,z^{k-1} + z^k$. Dann gilt für ein Codewortpolynom $c(z) \in \mathcal{C}$ gerade $c(z) = a(z)\,g(z)$ mit $a(z) = a_0 + a_1\,z + \cdots + a_{k-1}\,z^{k-1}$. Da erstens gilt

$$\begin{aligned} c(z)\,h(z) &= a(z)g(z)h(z) \\ &= a(z)(z^n-1) \\ &= a(z)\,z^n - a(z) \end{aligned}$$

und da zweitens $grad(a) < k$ ist, treten in $a(z)\,z^n - a(z)$ die Monome $z^k$, $z^{k+1}, \ldots, z^{n-1}$ nicht auf. Folglich sind in $c(z)\,h(z)$ die Koeffizienten dieser Monome Null und wir erhalten demnach die folgenden $n-k$ Gleichungen

$$\sum_{i=0}^{k} h_i\,c_{n-i-j} = 0 \quad \text{für alle } 1 \le j \le n-k.$$

Betrachten wir das sogenannte *reziproke Polynom* zu $h(z)$, das definiert ist durch

$$z^k\,h(z^{-1}) := h_k + h_{k-1}\,z + \cdots + h_0\,z^k,$$

dann ist das zu $h(z)$ reziproke Polynom ebenfalls ein Teiler von $z^n-1$. Als solcher erzeugt es nach Theorem 4.1.44 einen zyklischen $(n, n-k)$-Blockcode $\mathcal{C}^\perp$ und die $((n-k)\times n)$-Matrix $\mathbf{H}$ mit

$$\mathbf{H} = \begin{pmatrix} h_k & h_{k-1} & \cdots & h_1 & h_0 & 0 & \cdots & 0 \\ 0 & h_k & h_{k-1} & \cdots & h_1 & h_0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_k & h_{k-1} & \cdots & h_1 & h_0 \end{pmatrix}$$

ist eine Generatormatrix von $\mathcal{C}^\perp$. Aufgrund der obigen $n-k$ Gleichungen

$$\sum_{i=0}^{k} h_i\,c_{n-i-j} = 0 \text{ für alle } 1 \le j \le n-k$$

folgt, dass jedes Codewort $\mathbf{c} \in \mathcal{C}$ orthogonal zu jeder Zeile der Matrix $\mathbf{H}$ ist. Somit ist $\mathbf{H}$ eine Prüfmatrix des zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$.

**4.1.52 Korollar**
*Gegeben sei ein zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z)$ und dem zugehörigen Prüfpolynom $h(z)$. Dann ist der zu $\mathcal{C}$ duale Code $\mathcal{C}^\perp$ zyklisch und wird von dem zu $h(z)$ reziproken Polynom $z^k\,h(z^{-1})$ erzeugt.*

Sowohl die Generatormatrix als auch die Prüfmatrix eines systematischen zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$ lassen sich in Standardform angeben.

**4.1.53 Bemerkung**
Um für einen zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z)$ die Standardform der Generatormatrix $\mathbf{G}$ und der Prüfmatrix $\mathbf{H}$ herzuleiten, dividieren wir die Monome $z^{n-k+i}$ für alle $0 \le i \le k-1$ durch das Generatorpolynom $g(z)$ und erhalten

$$z^{n-k+i} = a_i(z)g(z) + \tilde{g}_i(z) \quad \text{für jedes } 0 \le i \le k-1.$$

Dabei bildet $\tilde{g}_i(z)$ den Rest der Division und ist von der Form

$$\tilde{g}_i(z) = \tilde{g}_{i0} + \tilde{g}_{i1}\,z + \cdots + \tilde{g}_{i(n-k-1)}\,z^{n-k-1} \quad \text{für jedes } 0 \le i \le k-1.$$

Da für jedes $0 \le i \le k-1$ die Polynome $\tilde{g}_i(z) + z^{n-k+i}$ Vielfache von $g(z)$ sind, stellen sie jeweils ein Codewortpolynom $\tilde{g}_i(z) + z^{n-k+i} \in \mathcal{C}$ dar und geben, als Zeilen einer $(k \times n)$-Matrix $\mathbf{G}$ angeordnet, eine Generatormatrix von $\mathcal{C}$ in systematischer Form an:

$$\mathbf{G} = \left(\begin{array}{cccc|ccc} \tilde{g}_{00} & \tilde{g}_{01} & \cdots & \tilde{g}_{0(n-k-1)} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \tilde{g}_{(k-1)0} & \tilde{g}_{(k-1)1} & \cdots & \tilde{g}_{(k-1)(n-k-1)} & 0 & \cdots & 1 \end{array}\right)$$

Nach Bemerkung 4.1.32 ergibt sich die entsprechende Prüfmatrix $\mathbf{H}$ als die zu $\mathbf{G}$ transponierte Matrix mit

$$\mathbf{H} = \left(\begin{array}{ccc|cccc} 1 & \cdots & 0 & \tilde{g}_{00} & \tilde{g}_{10} & \cdots & \tilde{g}_{(k-1)0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & \tilde{g}_{0(n-k-1)} & \tilde{g}_{1(n-k-1)} & \cdots & \tilde{g}_{(k-1)(n-k-1)} \end{array}\right)$$

Da $\tilde{g}_i(z)$ für jedes $0 \le i \le k-1$ jeweils den Polynomrest von $z^{n-k+i}$ bei Division durch das Generatorpolynom $g(z)$ darstellt, stimmt die Polynomdarstellung jedes Spaltenvektors $\mathbf{h}_i = (h_{0i}, \ldots, h_{(n-k-1)i})$ mit $0 \le i \le n-1$ der Prüfmatrix $\mathbf{H}$ mit dem Polynomrest der Monome $z^i$ mit $0 \le i \le n-1$ bei Division durch das Generatorpolynom $g(z)$ überein

$$h_i(z) = h_{0i} + h_{1i}\,z + \cdots + h_{(n-k-1)i}\,z^{n-k-1} = z^i \bmod g(z) \quad \text{für alle } 0 \le i \le n-1$$

**4.1.54 Korollar**
*Die vorangegangene Bemerkung zeigt, dass die Reste der Monome $z^i$ mit $0 \le i \le n-1$ hinsichtlich der Polynommultiplikation bei Rechnung* mod $g(z)$ *eine zyklische Gruppe bilden. Nach der Definition einer zyklischen Gruppe entspricht die Codelänge $n$ des von $g(z)$ erzeugten zyklischen Codes $\mathcal{C}$ der Ordnung des Elementes $z$ in dieser zyklischen Gruppe. Betrachten wir insbesondere ein primitives Generatorpolynom $g(z)$ vom Grad $n-k$, so beträgt die Codelänge $n$ gerade $2^{n-k}-1$, denn gemäß der Definition eines primitiven Polynoms besitzt dieses ein primitives Element $\alpha \in GF(2)^{n-k}$ als Wurzel, wobei $\alpha$ gerade $GF(2)^{n-k}$ erzeugt und die Ordnung $2^{n-k}-1$ besitzt. ??? für n-k einen neuen Parameter einführen ???*

Bevor wir im nächsten Abschnitt einige konkrete Beispiel für zyklische Codes angeben, die in der Praxis sehr häufig zur Fehlerkorrektur verwendet werden, stellen wir vor dem Hintergrund der algebraischen Struktur linearer bzw. zyklischer Codes ein einfaches Prinzip zur Decodierung vor, die sogenannte *Syndromdecodierung*. Bei der Einführung der Korrigierkugeln und in den Sätzen 4.1.16 und 4.1.19 haben wir bereits angemerkt, dass für eine korrekte Decodierung eines empfangenen und möglicherweise durch ein Fehlermuster $\mathbf{f} \in GF(2)^n$ verfälschten Binärwortes $\mathbf{r} \in GF(2)^n$ mit $\mathbf{r} = \mathbf{c} + \mathbf{f}$ eine eindeutige Zuordnung für jedes Binärwort $\mathbf{r} \in GF(2)^n$ zu genau einem Codewort $\mathbf{c} \in \mathcal{C}$ gewährleistet sein muß; d.h. die Korrigierkugeln $K^{\varrho}(\mathbf{c})$ müssen paarweise disjunkt sein und $GF(2)^n$ überdecken. Für einen linearen $(n,k,d)$-Blockcode $\mathcal{C}$ müssen daher genau $2^k$ Korrigierkugeln mit jeweils $2^{n-k}$ Elementen existieren. Der Begriff des Quotientenraumes aus der linearen Algebra liefert eine Möglichkeit, die Menge $GF(2)^n$ aller Binärwörter der Länge $n$ in $2^k$ disjunkte Korrigierkugeln aufzuteilen.

**4.1.55 Definition**
*Gegeben sei ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$. Dann heißt die Menge*

$$GF(2)^n/\mathcal{C} := \{\mathbf{a}+\mathcal{C}\ ;\ \mathbf{a} \in GF(2)^n\} \quad mit \quad \mathbf{a}+\mathcal{C} := \{\mathbf{a}+\mathbf{c}\ ;\ \mathbf{c} \in \mathcal{C}\}$$

*der Quotientenraum von $GF(2)^n$ nach $\mathcal{C}$. Die Menge $\mathbf{a}+\mathcal{C}$ wird in der linearen Algebra als Restklasse und in der Codierungstheorie als Coset bezeichnet und anstatt Repräsentant wird $\mathbf{a}$ in diesem Zusammenhang Coset-Leader genannt.*

Aus der linearen Algebra ergeben sich die folgenden Eigenschaften für *Cosets*.

**4.1.56 Satz**
*Gegeben sei ein linearer $(n,k,d)$-Blockcode $\mathcal{C}$.*

*(i) Die Anzahl der Elemente eines Cosets $\mathbf{a}+\mathcal{C}$, mit $\mathbf{a} \in GF(2)^n$ beliebig, entspricht gerade der Anzahl $2^k$ der Codewörter.*

*(ii) Die Vereinigung aller Cosets ergibt den gesamten Raum $GF(2)^n$.*

*(iii) Zwei Cosets sind entweder identisch oder disjunkt; d.h. aus $\mathbf{a}+\mathcal{C} = \mathbf{c}+\mathcal{C}$ folgt $\mathbf{a}+\mathbf{c} \in \mathcal{C}$. Daher existieren genau $2^{n-k}$ disjunkte Cosets.*

Eine elementare Voraussetzung dieser Arbeit besteht darin, dass wir als Decodierprinzip stets eine *Maximum-Likelihood-Decodierung* verwenden. Wir gehen also davon aus, dass weniger Übertragungsfehler wahrscheinlicher sind als viele. Daher wird zu einem empfangenen Binärwort $\mathbf{r} \in GF(2)^n$ als gesendetes Codewort stets dasjenige Codewort $\mathbf{c} \in \mathcal{C}$ decodiert, das minimalen Abstand zu $\mathbf{r}$ besitzt. Für lineare Codes ist dies nach Satz 4.1.15 gleichbedeutend damit, dass die Summe des empfangenen Binärwortes $\mathbf{r}$ und des Codewortes $\mathbf{c} \in \mathcal{C}$ minimales Gewicht besitzt,

$$\begin{aligned} \min\{\mathrm{wt}(\mathbf{r}+\mathbf{c})\ ;\ \mathbf{c} \in \mathcal{C}\} &= \min\{\mathrm{wt}(\mathbf{c}+\mathbf{c}+\mathbf{f})\ ;\ \mathbf{c} \in \mathcal{C},\ \mathbf{f} \in GF(2)^n\} \\ &= \min\{\mathrm{wt}(\mathbf{f})\ ;\ \mathbf{f} \in GF(2)^n\}. \end{aligned}$$

Der Decodierprozeß läßt sich mittels eines sogenannten *standard arrays* beschreiben, wobei wir alle Binärwörter aus $GF(2)^n$ folgendermaßen in einem *Feld* anordnen:

Die erste Zeile besteht aus genau allen Codewörtern $\mathbf{c} \in \mathcal{C}$, wobei das Nullwort der erste Eintrag ganz links oben ist. Um die zweite Zeile zu bilden, wählen wir ein Binärwort $\mathbf{f}_1 \in GF(2)^n \setminus \mathcal{C}$ von minimalem Gewicht aus, schreiben dieses unter das Nullwort und addieren es zu jedem Codewort in der ersten Zeile. Diesen Prozeß führen wir fort, bis alle Elemente aus $GF(2)^n$ in diesem Feld enthalten sind. Jede Zeile des *standard arrays* bildet dann ein *Coset* $\mathbf{f}_i + \mathcal{C}$, mit $1 \leq i \leq 2^{n-k} - 1$, wobei jeder *Coset-Leader* $\mathbf{f}_i$ ein Binärwort minimalen Gewichts im *Coset* $\mathbf{f}_i + \mathcal{C}$ ist.

| $\mathbf{0}$ | $\mathbf{c}_1$ | $\mathbf{c}_2$ | $\cdots$ | $\mathbf{c}_{2^k-1}$ |
|---|---|---|---|---|
| $\mathbf{f}_1$ | $\mathbf{f}_1 + \mathbf{c}_1$ | $\mathbf{f}_1 + \mathbf{c}_2$ | $\cdots$ | $\mathbf{f}_1 + \mathbf{c}_{2^k-1}$ |
| $\mathbf{f}_2$ | $\mathbf{f}_2 + \mathbf{c}_1$ | $\mathbf{f}_2 + \mathbf{c}_2$ | $\cdots$ | $\mathbf{f}_2 + \mathbf{c}_{2^k-1}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | | $\vdots$ |
| $\mathbf{f}_{2^{n-k}-1}$ | $\mathbf{f}_{2^{n-k}-1} + \mathbf{c}_1$ | $\mathbf{f}_{2^{n-k}-1} + \mathbf{c}_2$ | $\cdots$ | $\mathbf{f}_{2^{n-k}-1} + \mathbf{c}_{2^k-1}$ |

Da wir davon ausgehen, dass weniger Fehler wahrscheinlicher sind als viele, erhalten wir das zu decodierende Codewort, indem wir das Binärwort minimalen Gewichts des Cosets $\mathbf{r} + \mathcal{C}$ bestimmen. Gemäß der Konstruktion des *standard arrays* und da $\mathbf{r}$ in der $i$-ten Zeile des *arrays* liegt, ist dies gerade der *Coset-Leader* $\mathbf{f}_i$, so dass wir $\mathbf{r} + \mathbf{f}_i$ decodieren. Ferner bildet jede Spalte eines solchen *standard arrays* gerade eine Korrigierkugel um das Codewort, welches zu Beginn der jeweiligen Spalte steht. Liegt ein empfangenes Binärwort $\mathbf{r} \in GF(2)^n$ in der $j$-ten Spalte des *standard arrays*, so wird $\mathbf{r}$ decodiert als das Codewort $\mathbf{c}_j \in \mathcal{C}$, das zu Beginn der $j$-ten Spalte steht. Stimmt das Fehlermuster, durch das das gesendete Codewort in das Binärwort $\mathbf{r}$ verfälscht worden ist, mit dem *Coset-Leader* $\mathbf{f}_i$ überein, so erfolgt die Decodierung korrekt. Stimmt das Fehlermuster nicht mit $\mathbf{f}_i$ überein, so produziert der Decodierer einen Decodierfehler. Somit stellen die *Coset-Leader* $\mathbf{f}_i$ mit $1 \leq i \leq 2^{n-k} - 1$ alle korrigierbaren Fehlermuster des betrachteten Codes dar. Für einen linearen $(n, k, d)$-Blockcode $\mathcal{C}$ existieren daher genau $2^{n-k}$ korrigierbare Fehlermuster.

Nun können wir die Speicherung eines kompletten *standard arrays* vermeiden und die Decodierung weitergehend vereinfachen. Dazu benötigen wir die Prüfmatrix $\mathbf{H}$ eines linearen $(n, k, d)$-Blockcodes $\mathcal{C}$, mit der wir nach Satz 4.1.31 die Zugehörigkeit eines Binärwortes zu dem linearen $(n, k, d)$-Blockcode $\mathcal{C}$ überprüfen können.

**4.1.57 Definition**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$ und die zugehörige Prüfmatrix $\mathbf{H}$. Dann liefert die Multiplikation der Prüfmatrix $\mathbf{H}$ mit einem empfangenen Binärwort $\mathbf{r} \in GF(2)^n$ ein Binärwort*

$$\mathbf{s} \quad := \quad \mathbf{r}\,\mathbf{H}^T = (s_0, \ldots, s_{n-k-1}) \in GF(2)^{n-k}.$$

*Dieses Binärwort heißt das Fehlersyndrom von $\mathbf{r}$ bzw. wird als das Symptom des Fehlers bezeichnet.*

**4.1.58 Satz**
*Gegeben sei ein linearer $(n, k, d)$-Blockcode $\mathcal{C}$. Dann hängt das Fehlersyndrom $\mathbf{s} \in GF(2)^{n-k}$*

*eines empfangenen Binärwortes* $\mathbf{r} \in GF(2)^n$ *mit* $\mathbf{r} = \mathbf{c} + \mathbf{f}$ *nur von dem jeweiligen Fehlermuster* $\mathbf{f} \in GF(2)^n$ *ab und nicht von dem gesendeten Codewort* $\mathbf{c} \in \mathcal{C}$*. Darüber hinaus ist das Syndrom* $\mathbf{s}$ *genau dann identisch Null, wenn das Fehlermuster* $\mathbf{f}$ *ein Codewort ist:*

$$\mathbf{s} = \mathbf{0} \quad \Longleftrightarrow \quad \mathbf{f} \in \mathcal{C}$$

**Beweis**
Wegen $\mathbf{c} \in \mathcal{C}$ gilt nach Satz 4.1.31 $\mathbf{c}\,\mathbf{H}^T = \mathbf{0}$ und daraus folgt:

$$\mathbf{s} = \mathbf{r}\,\mathbf{H}^T \;=\; (\mathbf{c} + \mathbf{f})\,\mathbf{H}^T = \mathbf{c}\,\mathbf{H}^T + \mathbf{f}\,\mathbf{H}^T = \mathbf{f}\,\mathbf{H}^T,$$

so dass das Syndrom $\mathbf{s}$ nur vom Fehlermuster $\mathbf{f}$ abhängt. Insbesondere folgt daraus:

$$\mathbf{s} = \mathbf{f}\,\mathbf{H}^T = \mathbf{0} \quad \Longleftrightarrow \quad \mathbf{f} \in \mathcal{C}$$

□

**4.1.59 Satz**
*Gegeben sei ein linearer* $(n, k, d)$*-Blockcode* $\mathcal{C}$*. Zwei Binärwörter* $\mathbf{r}_1, \mathbf{r}_2 \in GF(2)^n$ *besitzen genau dann das selbe Syndrom* $\mathbf{s}$*, wenn* $\mathbf{r}_1$ *und* $\mathbf{r}_2$ *in dem selben Coset und damit in derselben Zeile des standard arrays liegen.*

**Beweis**
Es gilt:

$$\begin{aligned} \mathbf{r}_1, \mathbf{r}_2 \in \mathbf{f} + \mathcal{C} \;&\Longleftrightarrow\; \mathbf{r}_1 + \mathbf{r}_2 \in \mathcal{C} \\ &\Longleftrightarrow\; (\mathbf{r}_1 + \mathbf{r}_2)\mathbf{H}^T = \mathbf{0} \Longleftrightarrow \mathbf{r}_1\,\mathbf{H}^T + \mathbf{r}_2\,\mathbf{H}^T = \mathbf{s}_1 + \mathbf{s}_2 = \mathbf{0} \end{aligned}$$

□

Da aufgrund des vorangegangenen Satzes alle Binärwörter der selben Zeile eines *standard arrays* das gleiche Syndrom besitzen, können wir das Decodierschema weiter vereinfachen. Anstatt das komplette *standard array* abzuspeichern, müssen wir lediglich eine Tabelle bestehend aus Syndromen und *Coset-Leadern* ablegen und können dann über den *Coset-Leader* $\mathbf{f}_i$ das entsprechende Codewort ermitteln. Wird das Binärwort $\mathbf{r} \in GF(2)^n$ empfangen, berechnen wir das zugehörige Syndrom $\mathbf{s} = \mathbf{r}\,\mathbf{H}^T = \mathbf{f}\,\mathbf{H}^T \in GF(2)^{n-k}$, bestimmen den zugehörigen *Coset-Leader* $\mathbf{f}_i$, der das gleiche Syndrom $\mathbf{s}$ besitzt, und decodieren das Binärwort $\mathbf{r}$ dann als $\mathbf{r} + \mathbf{f}_i = \mathbf{c}_j$. Diese Form der Decodierung wird als *Syndromdecodierung* bezeichnet und ist sehr einfach zu implementieren. Allerdings wird auch hier, genauso wie bei der *standard-array-Decodierung*, der Speicheraufwand für sehr große Parameter $n$ und $k$ immens groß, vgl. dazu [26].

Für zyklische $(n, k, d)$-Blockcodes $\mathcal{C}$ können wir das Syndrom auch über die Polynomdarstellung charakterisieren.

**4.1.60 Bemerkung**
Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$ mit dem Generatorpolynom $g(z)$.

(i) Dann lassen sich die Syndrome sehr leicht berechnen. Jedes empfangene Binärwort $\mathbf{r} \in GF(2)^n$ ist mit $r(z) := r_0 + r_1\, z + \cdots + r_{n-1}\, z^{n-1}$ als binäres Polynom vom Grad $< n$ darstellbar. Dividieren wir $r(z)$ durch das Generatorpolynom $g(z)$,

$$r(z) = i(z)g(z) + s(z),$$

so erhalten wir als Rest der Polynomdivision ein Polynom

$$s(z) = s_0 + s_1\, z + \cdots + s_{n-k-1}\, z^{n-k-1}$$

über $GF(2)$ vom Grad $< n - k$. Die Koeffizienten dieses Polynoms $s(z)$ bilden dann das Syndrom $\mathbf{s} = (s_0, \ldots, s_{n-k-1}) \in GF(2)^{n-k}$ von $\mathbf{r}$.

(ii) Nach Theorem 4.1.44 ist $s(z) = 0$ genau dann, wenn $r(z)$ ein Codewortpolynom ist.

(iii) Ist $s(z)$ das Syndrom des empfangenen Polynoms $r(z) = r_0 + r_1\, z + \cdots + r_{n-1}\, z^{n-1}$, dann entspricht der Rest der Division von $z\, s(z)$ durch das Generatorpolynom $g(z)$ dem Syndrom einer zyklischen Verschiebung von $r(z)$ um eine Stelle; d.h. es gilt:

$$s(z) = r(z) \bmod g(z) \Longrightarrow z\, r(z) \bmod g(z) = z\, s(z) \bmod g(z) =: \tilde{s}(z)$$

Die vorangegangenen Ausführungen zur *standard-array*- und zur Syndromdecodierung zeigen im Hinblick auf die Fehlerkorrektur, dass ein Fehlermuster $\mathbf{f} \in GF(2)^n$ genau dann korrekt korrigiert werden kann, wenn dieses Fehlermuster einem *Coset-Leader* entspricht.

Für einen linearen $(n, k, d)$-Blockcode $\mathcal{C}$ mit der Mindestdistanz $d$ gilt ferner, dass $\mathcal{C}$ nach Satz 4.1.19 und Korollar 4.1.20 bis zu $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehler korrigieren kann. Nach den obigen Ausführungen müssen dann alle Fehlermuster $\mathbf{f} \in GF(2)^n$ mit Gewicht $\mathrm{wt}(\mathbf{f}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ jeweils einem *Coset-Leader* entsprechen und damit auf unterschiedliche Syndrome führen. Dieses Ergebnis, das einen Zusammenhang zwischen den Syndromen und den Fehlermustern bzw. den *Coset-Leadern* herstellt, benötigen wir später bei den Auswertungen unseres Kanalmodells sowie zum Entwurf eines entsprechenden Algorithmus, vgl. hierzu Abschnitt 5.1.1.

## 4.1.3 Einige Beispiele für zyklische Codes

Zum Abschluß dieses Abschnittes diskutieren wir nun konkrete Beispiele spezieller Codes, die zur Fehlerkontrolle in digitalen Kommunikations- bzw. Datenspeicherungssystemen eingesetzt werden. Die Codes, mit denen wir uns im folgenden auseinandersetzen, werden in der Praxis häufig in der Satellitenkommunikation oder auch in CD-Spielern zum Einsatz gebracht.

Die Familie der *Hamming-Codes* $\mathcal{H}$ ist wohl die bekannteste der Fehler-korrigierenden Codes. Im Jahr 1950 bzw. 1949 sind diese Codes unabhängig voneinander von Richard Hamming und Marcel Golay entdeckt worden. Hamming-Codes sind perfekte, lineare Codes, die aufgrund ihrer algebraischen Struktur sehr leicht zu decodieren sind.

**4.1.61 Definition**
*Für jede Zahl $m \geq 3$ ist ein Hamming-Code $\mathcal{H}$ ein linearer $(n, k, d)$-Blockcode mit den Parametern:*

$$\begin{aligned} \textit{Codelänge } n &= 2^m - 1 \\ \textit{Dimension } k &= 2^m - 1 - m \\ \textit{Mindestdistanz } d &= 3 \end{aligned}$$

*Die Prüfmatrix $\mathbf{H}$ eines Hamming-Codes $\mathcal{H}$ zeichnet sich dadurch aus, dass sie spaltenweise aus allen $2^m - 1$ Binärwörtern der Länge $m$ besteht, die vom Nullwort verschieden sind.*

**4.1.62 Bemerkung**
Da die Prüfmatrix $\mathbf{H}$ eines Hamming-Codes $\mathcal{H}$ laut Definition aus allen $2^m - 1$ vom Nullwort verschiedenen Binärwörtern der Länge $m$ besteht, können wir die Prüfmatrix $\mathbf{H}' = \left(\mathbf{I}_{m\times m} | \tilde{\mathbf{H}}'_{m\times 2^m-1-m}\right)$ eines zu $\mathcal{H}$ äquivalenten systematischen Hamming-Codes $\mathcal{H}'$ sofort angeben. Dabei beinhaltet die Teilmatrix $\tilde{\mathbf{H}}'_{m\times 2^m-1-m}$ folglich alle Binärwörter der Länge $m$ vom Gewicht $\geq 2$.

**4.1.63 Satz**
*Gegeben sei ein Hamming-Code $\mathcal{H}$ der Länge $n = 2^m - 1$ und der Dimension $k = 2^m - 1 - m$. Dann ist die Mindestdistanz $d$ von $\mathcal{H}$ stets genau $d = 3$ und damit unabhängig von der Codelänge $n$ und der Dimension $k$.*

**Beweis**
Da die Prüfmatrix $\mathbf{H}$ eines Hamming-Codes $\mathcal{H}$ nach Definition 4.1.61 aus allen $2^m - 1$ vom Nullwort verschiedenen Binärwörtern der Länge $m$ besteht, sind einerseits jeweils zwei Spalten von $\mathbf{H}$ stets voneinander verschieden, so dass ihre Summe nie das Nullwort ergibt. Andererseits liefert die Summe zweier Spalten $i$ und $j$ von $\mathbf{H}$ stets wieder eine Spalte $l$ von $\mathbf{H}$. Von daher ergibt die Summe dieser drei Spalten $i, j$ und $l$ von $\mathbf{H}$ das Nullwort. Diese drei Spalten von $\mathbf{H}$ sind somit linear abhängig. Nach Satz 4.1.34 besitzt jeder Hamming-Codes $\mathcal{H}$ stets die Mindestdistanz $d = 3$. □

**4.1.64 Satz**
*Gegeben sei ein Hamming-Code $\mathcal{H}$ der Länge $n = 2^m - 1$ und der Dimension $k = 2^m - 1 - m$. Dann korrigiert $\mathcal{H}$ genau einen Fehler und ist ein perfekter Code.*

**Beweis**
Weil die Mindestdistanz für jeden Hamming-Code $\mathcal{H}$ stets $d = 3$ ist, folgt aus Satz 4.1.19 und aus Korollar 4.1.20, dass jeder Hamming-Code $\mathcal{H}$ genau

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor = \left\lfloor \frac{3-1}{2} \right\rfloor = 1$$

Fehler korrigieren kann. Darüber hinaus erfüllt jeder Hamming-Code $\mathcal{H}$ die Hamming-Schranke aus Korollar 4.1.21 mit Gleichheit

$$2^n = 2^k \sum_{i=0}^{1} \binom{n}{i} = 2^k(1+n) \iff 2^m = 2^{n-k} = 1 + n = 1 + 2^m - 1 = 2^m,$$

so dass jeder Hamming-Code $\mathcal{H}$ laut Definition 4.1.22 und Korollar 4.1.23 ein perfekter Code ist. □

Die *Bose-Chaudhuri-Hocquenghem-Codes* sind ein weiteres Beispiel für eine sehr große Klasse effizienter Fehler-korrigierender zyklischer Codes und sind in der Literatur kurz unter dem Namen *BCH-Codes* bekannt. Die binären BCH-Codes, mit denen wir uns im folgenden auseinandersetzen, sind unabhängig im Jahre 1959 von Hocquenghem und im Jahre 1960 von Bose und Chaudhuri entdeckt worden, vgl. dazu [6, 19]. Sie bilden aus verschiedenen Gründen eine äußerst wichtige Klasse der Fehler-korrigierenden Codes. So besitzen sie sehr gute Fehlerkorrektureigenschaften, falls die Codelänge $n$ nicht zu groß ist. BCH-Codes sind aufgrund ihrer algebraischen Struktur sehr leicht zu codieren und zu decodieren, wobei unter allen Decodieralgorithmen derjenige von Berlekamp zu den effektivsten und in diesem Sinne auch zu den schnellsten zählt. Darüber hinaus stellen BCH-Codes eine gute Basis zur Konstruktion weiterer Familien Fehler-korrigierender Codes dar. Die Klasse der BCH-Codes repräsentiert insbesondere eine bemerkenswerte Verallgemeinerung der Hamming-Codes $\mathcal{H}$ im Hinblick auf die Korrektur mehrerer Fehler anstelle nur eines Fehlers.

**4.1.65 Definition**
*Gegeben seien ein primitives Element $\alpha \in GF(2^m)$ mit der Ordnung $ord(\alpha) = 2^m - 1$ sowie ein Polynom $g(z)$, welches das Polynom kleinsten Grades über $GF(2)$ ist, welches $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{d-1}$ für $d-1 < 2^m$ als Wurzeln besitzt, d.h. $g(\alpha_i) = 0$ für alle $1 \leq i \leq d-1$. Dann erzeugt das Polynom $g(z)$ als Generatorpolynom einen linearen zyklischen Code $\mathcal{C}$, der als BCH-Code, oder genauer als primitiver BCH-Code, bezeichnet wird.*

**4.1.66 Satz**
*Ein primitiver BCH-Code $\mathcal{C}$, dessen Generatorpolynom $g(z)$ laut Definition 4.1.65 durch $d-1$ aufeinanderfolgende Wurzeln aus $GF(2^m)$ festgelegt wird, besitzt die folgenden Parameter:*

$$\begin{array}{lcl} \textit{Codelängen} & = & 2^m - 1 \\ \textit{Dimension} k & \geq & 2^m - 1 - m\left\lfloor \frac{d-1}{2} \right\rfloor \end{array}$$

**Beweis**

(i) Da $\alpha \in GF(2^m)$ ein primitives Element und damit von der Ordnung $ord(\alpha) = 2^m - 1$ ist, besitzt ein binärer BCH-Code $\mathcal{C}$ die Codelänge $n = 2^m - 1$.

(ii) Indem wir zeigen, dass das Generatorpolynom $g(z)$ höchstens vom Grad $m\left\lfloor \frac{d-1}{2} \right\rfloor$ ist, erhalten wir, dass die Dimension $k$ eines binären BCH-Codes $\mathcal{C}$ mindestens $2^m - 1 - m\left\lfloor \frac{d-1}{2} \right\rfloor$ beträgt.

Die Methoden der linearen Algebra liefern, dass das Generatorpolynom $g(z)$ die Elemente $\alpha, \alpha^2, \ldots, \alpha^{d-1}$ sowie deren konjugierte Wurzeln als seine Wurzeln besitzt. Wird mit $m_i(z)$ das *Minimalpolynom* jeder Wurzel $\alpha^i$ für $1 \leq i \leq d-1$ bezeichnet,

so entspricht das Generatorpolynom $g(z)$ dem kleinsten gemeinsamen Vielfachen der Minimalpolynome

$$g(z) = \text{kgV}\{m_1(z), m_2(z), \ldots, m_{d-1}(z)\}.$$

Nun kann jede gerade ganze Zahl $i$ mit $1 \leq i \leq d-1$ geschrieben werden als $i = 2^l i'$, wobei $l \geq 1$ und $i' \leq i$ ungerade ist. Daher ist $\alpha^i = (\alpha^{i'})^{2^l}$ eine konjugierte Wurzel von $\alpha^{i'}$, so dass $\alpha^i$ und $\alpha^{i'}$ dasselbe Minimalpolynom $m_{i'}(z) = m_i(z)$ besitzen. Demzufolge besitzt jede gerade Potenz von $\alpha$ in der Folge $\alpha$, $\alpha^2$, $\alpha^3$, $\ldots$, $\alpha^{d-1}$ dasselbe Minimalpolynom wie eine vorangegangene ungerade Potenz von $\alpha$ in dieser Folge. Das Generatorpolynom $g(z)$ eines binären BCH-Codes der Länge $n = 2^m - 1$ ist damit gegeben als kleinstes gemeinsames Vielfaches der Minimalpolynome $m_{i'}(z)$ mit $i'$ ungerade, $1 \leq i' \leq d-2$ und $i' \leq i$:

$$g(z) = \text{kgV}\{m_1(z), m_3(z), \ldots, m_{d-2}(z)\}.$$

Da der Grad jedes dieser Minimalpolynome $m_{i'}(z)$ gerade $grad(m_{i'}) \leq m$ ist, beträgt der Grad des Generatorpolynoms $g(z)$ höchstens $m \left\lfloor \frac{d-1}{2} \right\rfloor$, also $grad(g) \leq m \left\lfloor \frac{d-1}{2} \right\rfloor$. Da der Grad des Generatorpolynoms $g(z)$ der Anzahl $n-k$ der Kontrollstellen eines zyklischen Codes $\mathcal{C}$ entspricht, ist die Dimension $k$ eines durch $g(z)$ erzeugten BCH-Codes $\mathcal{C}$ mindestens von der Größe

$$k \geq 2^m - 1 - m \left\lfloor \frac{d-1}{2} \right\rfloor.$$

□

Um eine Aussage hinsichtlich der Mindestdistanz $d$ eines BCH-Codes $\mathcal{C}$ der Länge $n = 2^m - 1$ beweisen zu können, untersuchen wir gemäß Satz 4.1.34 die Anzahl der linear abhängigen Spalten der Prüfmatrix $\mathbf{H}$ des BCH-Codes $\mathcal{C}$. Dazu zeigen wir zunächst die beiden folgenden Sätze, mit denen BCH-Codes und deren Prüfmatrix $\mathbf{H}$ charakterisiert werden.

**4.1.67 Satz**
*Ein Binärwort $\mathbf{c} \in GF(2)^n$ mit $\mathbf{c} = (c_0, \ldots, c_{n-1})$ ist genau dann ein Codewort eines BCH-Codes $\mathcal{C}$ der Länge $n = 2^m - 1$, wenn das Polynom $c(z)$ mit $c(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1}$ die Elemente $\alpha$, $\alpha^2$, $\ldots$, $\alpha^{d-1}$ als Wurzeln besitzt und somit $c(\alpha^i) = 0$ für jedes $1 \leq i \leq d-1$ gilt.*

**Beweis**
$\Longleftarrow$: Die Elemente $\alpha$, $\alpha^2$, $\ldots$, $\alpha^{d-1}$ seien Wurzeln des Polynoms $c(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1}$. Dann ist nach den Regeln der linearen Algebra das Polynom $c(z)$ teilbar durch die Minimalpolynome $m_1(z), m_2(z), \ldots, m_{d-1}(z)$ der Wurzeln $\alpha$, $\alpha^2$, $\ldots$, $\alpha^{d-1}$. Offensichtlich ist das Polynom $c(z)$ dann auch teilbar durch das kleinste gemeinsame Vielfache der Minimalpolynome, das gemäß Definition 4.1.65 dem Generatorpolynom $g(z)$ des betrachteten BCH-Codes $\mathcal{C}$ entspricht. Als Vielfaches des Generatorpolynoms $g(z)$ ist das Polynom $c(z)$ damit nach Theorem 4.1.44 ein Codewortpolynom des BCH-Codes $\mathcal{C}$.

$\Longrightarrow$: Aus der Definition eines BCH-Codes $\mathcal{C}$ der Länge $n = 2^m - 1$ folgt direkt, dass jedes Codewortpolynom $c(z)$ die Elemente $\alpha$, $\alpha^2$, $\ldots$, $\alpha^{d-1}$ und deren konjugierte Wurzeln als

Nullstellen besitzt. □

Aus dem obigen Satz läßt sich für einen BCH-Code $\mathcal{C}$ der Länge $n = 2^m - 1$ die folgende zweckmäßige Darstellung einer zugehörigen Prüfmatrix $\mathbf{H}$ herleiten:

**4.1.68 Bemerkung**
Das Polynom $c(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1}$ sei ein Codewortpolynom des BCH-Codes $\mathcal{C}$ der Länge $n = 2^m - 1$. Dann ist nach Satz 4.1.67 $\alpha^i$ für jedes $1 \leq i \leq d-1$ eine Wurzel des Polynoms $c(z)$ und es gilt folglich für jedes $1 \leq i \leq d-1$, dass $c(\alpha^i) = 0$ ist. In Matrixschreibweise bedeutet dies

$$(c_0, c_1, \ldots, c_{n-1}) \begin{pmatrix} 1 \\ \alpha^i \\ \alpha^{2i} \\ \vdots \\ \alpha^{(n-1)i} \end{pmatrix} = 0 \quad \text{für alle } 1 \leq i \leq d-1,$$

woraus wir für den gegebenen BCH-Code $\mathcal{C}$ der Länge $n = 2^m - 1$ die folgende Matrix ableiten

$$\mathbf{H}^* = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \cdots & \alpha^{n-1} \\ 1 & (\alpha^2) & (\alpha^2)^2 & (\alpha^2)^3 & \cdots & (\alpha^2)^{n-1} \\ 1 & (\alpha^3) & (\alpha^3)^2 & (\alpha^3)^3 & \cdots & (\alpha^3)^{n-1} \\ \vdots & & & & & \vdots \\ 1 & (\alpha^{d-1}) & (\alpha^{d-1})^2 & (\alpha^{d-1})^3 & \cdots & (\alpha^{d-1})^{n-1} \end{pmatrix}$$

So wie die Matrix $\mathbf{H}^*$ hergeleitet worden ist, folgt direkt, dass für jedes Codewortpolynom $c(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1}$ die Bedingung $\mathbf{c}\,\mathbf{H}^{*T} = \mathbf{0}$ aus Satz 4.1.31 erfüllt ist.

Umgekehrt gilt, wenn die Bedingung $\mathbf{c}\,\mathbf{H}^{*T} = \mathbf{0}$ für ein beliebiges Polynom $c(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1}$ gültig ist, dass dann $\alpha^i$ für jedes $1 \leq i \leq d-1$ eine Wurzel von $c(z)$ ist. Nach Satz 4.1.67 ist $c(z)$ dann ein Codewortpolynom des BCH-Codes.

In Satz 4.1.66 haben wir gezeigt, dass für jede gerade ganze Zahl $i$ mit $1 \leq i \leq d-1$ das Element $\alpha^i$ eine konjugierte Wurzel von $\alpha^{i'}$ ist, mit $i'$ ungerade, $1 \leq i' \leq d-2$ und $i' \leq i$. Dann gilt $c(\alpha^{i'}) = 0 \iff c(\alpha^i) = 0$, so dass eine der entsprechenden Zeilen, entweder die Zeile bezüglich $\alpha^i$ oder diejenige bezüglich $\alpha^{i'}$, in der Matrix $\mathbf{H}^*$ gestrichen werden kann. Von daher ergibt sich die Matrix $\mathbf{H}$ zu:

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \cdots & \alpha^{n-1} \\ 1 & (\alpha^3) & (\alpha^3)^2 & (\alpha^3)^3 & \cdots & (\alpha^3)^{n-1} \\ 1 & (\alpha^5) & (\alpha^5)^2 & (\alpha^5)^3 & \cdots & (\alpha^5)^{n-1} \\ \vdots & & & & & \vdots \\ 1 & (\alpha^{d_3^2}) & (\alpha^{d_3^2})^2 & (\alpha^{d_3^2})^3 & \cdots & (\alpha^{d_3^2})^{n-1} \end{pmatrix}$$

Hierbei gilt:

$$d_3^2 := \begin{cases} d-2; & \text{falls } d \text{ ungerade ist} \\ d-3; & \text{falls } d \text{ gerade ist} \end{cases}$$

Werden letztendlich die Elemente $\alpha^{i'} \in GF(2^m)$ mit $1 \leq i' \leq d-2$ und $i'$ ungerade in dieser Matrix $\mathbf{H}$ durch die entsprechenden Binärwörter aus $GF(2^m)$ ersetzt, repräsentiert $\mathbf{H}$ eine binäre Prüfmatrix des BCH-Codes $\mathcal{C}$.

**4.1.69 Satz**
*Gegeben sei ein BCH-Code $\mathcal{C}$ der Länge $n = 2^m - 1$, dessen Generatorpolynom $g(z)$ gemäß Definition 4.1.65 durch die aufeinanderfolgenden Wurzeln $\alpha, \alpha^2, \ldots, \alpha^{d-1} \in GF(2^m)$ bestimmt wird. Dann beträgt die Mindestdistanz des BCH-Codes $\mathcal{C}$ mindestens $d$.*

**Beweis**
Wir zeigen dazu, dass $d-1$ oder weniger Spalten der in der vorangegangenen Bemerkung aufgestellten Prüfmatrix $\mathbf{H}$ des BCH-Codes $\mathcal{C}$ niemals linear abhängig sind, in Summe also niemals das Nullwort ergeben. Angenommen, es existiert ein Codewort $\mathbf{c} \in \mathcal{C}$, $\mathbf{c} \neq \mathbf{0}$, mit dem Gewicht $\mathrm{wt}(\mathbf{c}) = \delta \leq d-1$. Ohne Einschränkungen seien die Komponenten $c_{i_1}, c_{i_2}, \ldots, c_{i_\delta}$ diejenigen Komponenten in $\mathbf{c}$, die ungleich Null sind. Die vorherige Bemerkung liefert

$$\begin{aligned}
\mathbf{c}\mathbf{H}^T &= (c_{i_1}, c_{i_2}, \ldots, c_{i_\delta}) \begin{pmatrix} \alpha^{i_1} & (\alpha^2)^{i_1} & \cdots & (\alpha^{d-1})^{i_1} \\ \alpha^{i_2} & (\alpha^2)^{i_2} & \cdots & (\alpha^{d-1})^{i_2} \\ \vdots & & & \vdots \\ \alpha^{i_\delta} & (\alpha^2)^{i_\delta} & \cdots & (\alpha^{d-1})^{i_\delta} \end{pmatrix} \\
&= (1, 1, \ldots, 1) \begin{pmatrix} \alpha^{i_1} & (\alpha^{i_1})^2 & \cdots & (\alpha^{i_1})^{d-1} \\ \alpha^{i_2} & (\alpha^{i_2})^2 & \cdots & (\alpha^{i_2})^{d-1} \\ \vdots & & & \vdots \\ \alpha^{i_\delta} & (\alpha^{i_\delta})^2 & \cdots & (\alpha^{i_\delta})^{d-1} \end{pmatrix} \overset{4.1.67}{=} \mathbf{0} \\
\Longrightarrow \qquad & (1, 1, \ldots, 1) \begin{pmatrix} \alpha^{i_1} & (\alpha^{i_1})^2 & \cdots & (\alpha^{i_1})^{\delta} \\ \alpha^{i_2} & (\alpha^{i_2})^2 & \cdots & (\alpha^{i_2})^{\delta} \\ \vdots & & & \vdots \\ \alpha^{i_\delta} & (\alpha^{i_\delta})^2 & \cdots & (\alpha^{i_\delta})^{\delta} \end{pmatrix} = \mathbf{0}
\end{aligned}$$

Um die letzte Gleichung für die angegebene $(\delta \times \delta)$-Matrix zu erfüllen, muß die Determinante dieser $(\delta \times \delta)$-Matrix gleich 0 sein. Klammern wir aus jeder Zeile $i_j$ für $1 \leq j \leq \delta$ den gemeinsamen Faktor $\alpha^{i_j}$ aus, so erhalten wir eine Vandermonde-Determinante, welche stets ungleich 0 ist. Folglich kann das Produkt $\mathbf{c}\mathbf{H}^T$ nicht $\mathbf{0}$ ergeben und $\mathbf{c}$ ist kein Codewort, im Widerspruch zur Annahme. Demnach sind $d-1$ Spalten der Prüfmatrix $\mathbf{H}$ stets linear unabhängig und nach Satz 4.1.34 ist die Mindestdistanz des BCH-Codes $\mathcal{C}$ daher mindestens $d$. □

**4.1.70 Bemerkung**
Der Parameter $d$ wird gewöhnlich auch als *geplante Mindestdistanz* eines BCH-Codes bezeichnet. Die tatsächliche Mindestdistanz eines BCH-Codes muß nicht zwangsläufig der

geplanten Mindestdistanz entsprechen. So gibt es eine Vielzahl von BCH-Codes, deren tatsächliche Mindestdistanz größer als die geplante Mindestdistanz $d$ ist.

**4.1.71 Korollar**
*Ein BCH-Code $\mathcal{C}$ der Länge $n = 2^m - 1$ mit der geplanten Mindestdistanz $d$ kann somit mindestens bis zu $t$ Fehler korrigieren, wobei $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ ist. Umgekehrt besitzt ein BCH-Code $\mathcal{C}$ der Länge $n = 2^m - 1$, der bis zu $t$ Fehler korrigieren kann, die geplante Mindestdistanz $d = 2t + 1$ und die tatsächliche Mindestdistanz ist $\geq 2t + 1$.*

**4.1.72 Korollar**
*Gegeben sei ein BCH-Code $\mathcal{C}$ der Länge $2^m - 1$ mit der tatsächlichen Mindestdistanz $d = 3$. Dann ist dieser BCH-Code $\mathcal{C}$ ein Hamming-Code $\mathcal{H}$.*

**Beweis**
In Satz 4.1.66 haben wir gezeigt, dass das Generatorpolynom $g(z)$ eines BCH-Codes $\mathcal{C}$ das kleinste gemeinsame Vielfache der Minimalpolynome $m_{i'}(z)$ der Wurzeln $\alpha^{i'}$ mit $1 \leq i' \leq d-2$ und $i' \leq i$ ungerade ist. Wegen $d = 3$ wird ein BCH-Code $\mathcal{C}$ mit der tatsächlichen Mindestdistanz 3 durch ein Generatorpolynom $g(z)$ der Form $g(z) = m_1(z)$ erzeugt. Dabei ist $m_1(z)$ das Minimalpolynom der Wurzel $\alpha$. Nach der Definition eines BCH-Codes ist $\alpha \in GF(2^m)$ ein primitives Element und damit ist $m_1(z)$ ein primitives Polynom vom Grad $m$. Daraus folgt, dass der durch das primitive Generatorpolynom $g(z) = m_1(z)$ erzeugte BCH-Code $\mathcal{C}$ ein Hamming-Code $\mathcal{H}$ ist. □

BCH-Codes im Sinne unserer Definition heißen auch *primitive BCH-Codes*, da $\alpha \in GF(2^m)$ ein primitives Element ist. In [26] werden primitive BCH-Codes sehr detailliert diskutiert. Ferner werden Tabellen angegeben, die alle möglichen Parameter primitiver BCH-Codes der Länge $n = 2^m - 1$ mit $m \leq 10$ enthalten. Darüber hinaus sind in [26] Tabellen und Listen aufgeführt, die sämtliche Generatorpolynome aller binären primitiven BCH-Codes der Länge $n = 2^m - 1$ mit $m \leq 10$ beinhalten.

Wenn zur Bestimmung des Generatorpolynoms $g(z)$ im Sinne von Definition 4.1.65 ein Element $\beta \in GF(2^m)$ verwendet wird, das kein primitives Element aus $GF(2^m)$ ist, dann ist die Codelänge $n$ des durch $g(z)$ erzeugten BCH-Codes $\mathcal{C}$ ungleich $2^m - 1$. Der durch ein solches Generatorpolynom $g(z)$ erzeugte Code heißt ein *nicht-primitiver BCH-Code*. Weitergehende Ausführungen zu nicht-primitiven BCH-Codes finden sich in den Lehrbüchern von [7, 47].

In [26, 47] werden zudem sehr ausführlich auch ganz allgemein sogenannte *$q$-näre BCH-Codes* diskutiert. Diese basieren auf einem $q$-nären Codealphabet. Dazu sei $q$ eine beliebige Primzahlpotenz $q = p^m$, mit $p$ prim und $m \geq 1$. Dann existieren Codes mit Codesymbolen aus $GF(q)$, die als $q$-näre Codes bezeichnet werden. Die Konzepte und Eigenschaften, die wir für binäre Codes entwickelt haben, lassen sich mit geringfügigen Modifikationen auf $q$-näre Codes übertragen. Für beliebige Zahlen $l, t \in \mathbb{N}$ und ein primitives Element $\alpha \in GF(q^l)$ erzeugt ein Generatorpolynom $g(z)$, welches das Polynom kleinsten Grades über $GF(q)$ ist und die Elemente $\alpha, \alpha^2, \ldots, \alpha^{2t}$ als Wurzeln besitzt, einen *$q$-nären BCH-Code* der Länge $n = q^l - 1$. Dieser q-näre BCH-Code besitzt höchstens $2lt$ Kontrollstellen und die tatsächliche Mindestdistanz $d \geq 2t + 1$. Folglich korrigiert dieser $q$-näre BCH-Code

mindestens bis zu $t$ Fehler.

Zum Abschluß stellen wir eine sehr bekannte und ganz spezielle Teilmenge der BCH-Codes vor, die sogenannten *Reed-Solomon-Codes*, kurz auch *RS-Codes* genannt. Die Familie der Reed-Solomon-Codes umfaßt alle $q$-nären BCH-Codes, für die der Parameter $l = 1$ und deren Codelänge $n$ somit nach den vorangegangenen Überlegungen gerade $n = q - 1$ ist. Reed-Solomon-Codes werden in der Praxis sehr häufig eingesetzt. So nutzt die US-Raumfahrtbehörde NASA Reed-Solomon-Codes in Verknüpfung mit anderen Fehler-korrigierenden Codes sehr häufig in ihren Forschungsprogrammen. Zum Beispiel sind Reed-Solomon-Codes während der Galileo- und der Ulysses-Mission von der NASA für eine möglichst sichere Übertragung der Bilddaten via Satellit verwendet worden.

**4.1.73 Definition**
*Für eine Primzahlpotenz $q = p^m$, mit $p$ prim und $m \geq 1$, heißt ein $q$-närer BCH-Code der Länge $n = q - 1$ ein $q$-närer Reed-Solomon-Code $RS$.*

Aufgrund ihrer Definition ist die Klasse der $q$-nären Reed-Solomon-Codes eine spezielle Teilmenge der $q$-nären BCH-Codes. Dabei zeichnen sich Reed-Solomon-Codes dadurch aus, dass ihre Länge der Anzahl der Elemente aus $GF(q) \setminus \{\mathbf{0}\}$ entspricht. Da wir uns im Rahmen der Arbeit ausschließlich mit binären Codes befassen, betrachten wir insbesondere Reed-Solomon-Codes der Länge $n = 2^m - 1$ über $GF(2^m)$.

**4.1.74 Satz**
*Gegeben sei ein Reed-Solomon-Code $RS$ der Länge $n = 2^m - 1$ über $GF(2^m)$ mit der geplanten Mindestdistanz $d = 2t + 1$, dann korrigiert der RS-Code bis zu $t$ Fehler und besitzt ein Generatorpolynom $g(z)$ der Gestalt*

$$\begin{aligned} g(z) &= (z-\alpha)(z-\alpha^2)\cdots(z-\alpha^{d-1}) = (z-\alpha)(z-\alpha^2)\cdots(z-\alpha^{2t}) \\ &= g_0 + g_1 z + \cdots + g_{2t} z^{2t} = g_0 + g_1 z + \cdots + g_{d-1} z^{d-1} \end{aligned}$$

**Beweis**
Da die Codelänge $n$ eines Reed-Solomon-Codes $RS$ $n = 2^m - 1$ ist und da die Wurzeln des Polynoms $z^{2^m-1} - 1$ nach den Ergebnissen der linearen Algebra genau alle Elemente aus $GF(2^m) \setminus \{\mathbf{0}\}$ sind, gilt

$$z^n - 1 = z^{2^m-1} - 1 = \prod_{a \in GF(2^m)\setminus\{\mathbf{0}\}} (z - a)$$

Ist $\alpha \in GF(2^m)$ eine primitive Wurzel über $GF(2^m)$, also ein primitives Element aus $GF(2^m)$ mit der Ordnung $ord(\alpha) = 2t = d - 1$, dann sind die Minimalpolynome der Wurzeln $\alpha^i$ von der Form $m_i(z) = z - \alpha^i$ für alle $1 \leq i \leq 2t = d - 1$. Satz 4.1.66 liefert, dass das Generatorpolynom $g(z)$ das kleinste gemeinsame Vielfache der Minimalpolynome $m_i(z) = z - \alpha^i$ der Wurzeln $\alpha^i$ für jedes $1 \leq i \leq d - 1$ ist. Folglich ist das Generatorpolynom $g(z)$ eines Reed-Solomon-Codes $RS$ der Länge $n = 2^m - 1$ von der

angegebenen Gestalt $g(z) = \prod_{i=1}^{d-1} z - \alpha^i$. Ferner folgt aus Satz 4.1.34, dass die geplante Mindestdistanz $d = 2t+1$ ist und ein Reed-Solomon-Code daher bis zu $t$ Fehler korrigiert. □

**4.1.75 Bemerkung**
Nach dem vorangehenden Satz ist das Generatorpolynom $g(z) = \prod_{i=1}^{d-1} z - \alpha^i$ eines Reed-Solomon-Codes $RS$ der Länge $n = 2^m - 1$ mit der geplanten Mindestdistanz $d$ vom Grad $grad(g) = d - 1$. Demzufolge besitzt ein Reed-Solomon-Code mindestens $d - 1$ Kontrollstellen, so dass sich die Dimension $k$ zu $k = n - grad(g) = n - d + 1$ ergibt.

Im Hinblick auf die tatsächliche Mindestdistanz $d'$ gilt einerseits:

$$d' \geq d = n - k + 1$$

Nun kann eine $((n-k) \times n)$-Prüfmatrix $\mathbf{H}$ eines $(n, k)$-Reed-Solomon-Codes maximal $n - k$ linear unabhängige Spalten besitzen, so dass für die tatsächliche Mindestdistanz $d'$ andererseits gilt:

$$d' - 1 \leq n - k$$

Insgesamt folgt daraus für die tatsächliche Mindestdistanz $d'$ eines Reed-Solomon-Codes

$$n - k \geq d' - 1 \geq d - 1 = n - k \quad \Longrightarrow \quad d' = d = n - k + 1.$$

Folglich stimmen die geplante Mindestdistanz $d$ und die tatsächliche Mindestdistanz $d'$ überein. Reed-Solomon-Codes besitzen aus diesem Grund die größtmögliche Mindestdistanz aller $q$-nären Codes der Länge $n = q - 1$ und der Dimension $k = n - d + 1$. Daher heißen Reed-Solomon-Codes auch *maximum distance separable*.

Reed-Solomon-Codes sind erstens aufgrund ihrer außerordentlichen Fehlerkorrektureigenschaften in der Praxis sehr weit verbreitet. Zweitens gibt es vielfältige Möglichkeiten die Parameter, Codelänge, Dimension und Mindestdistanz, zu kombinieren. So sind Reed-Solomon-Codes hervorragend zur Erweiterung im Sinne von Definition 4.1.37 geeignet.

**4.1.76 Bemerkung**
Ein Reed-Solomon-Code $RS$ der Länge $n = 2^m - 1$ kann durch Hinzufügen zweier weiterer Informationsbits verlängert werden zu einem Reed-Solomon-Code $\hat{RS}$ der Länge $\hat{n} = n + 2$. Wir erweitern dazu die gegebene Prüfmatrix $\mathbf{H}$

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^2)^{n-1} \\ \vdots & & & & \vdots \\ 1 & \alpha^{d-1} & (\alpha^{d-1})^2 & \cdots & (\alpha^{d-1})^{n-1} \end{pmatrix},$$

des vorgegebenen RS-Codes, indem wir folgendermaßen zwei Spalten hinzufügen und so die erweiterte Prüfmatrix $\hat{\mathbf{H}}$

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

erhalten. Der um zwei Stellen erweiterte Reed-Solomon-Code $\hat{RS}$ besitzt dann die Parameter:

$$\begin{aligned} \text{Codelänge } \hat{n} &= n+2 \\ \text{Dimension } \hat{k} &= k = n-d+1 \\ \text{Mindestdistanz } \hat{d} &= d+2. \end{aligned}$$

Zum Beweis und für weitere ausführlichere Aussagen zu Reed-Solomon-Codes verweisen wir auf das Lehrbuch von [7]. Hier wird überdies ein anderer Zugang zur Familie der Reed-Solomon-Codes gewählt und eine zu unserer Definition äquivalente Charakterisierung mittels der Transformationstechnik angegeben. In [7] orientiert sich die Einführung der Reed-Solomon-Codes in erster Linie an dem Problem der Decodierung von Codes. Daher ist die Beschreibung und Charakterisierung von Codes in diesem Zusammenhang in der Hauptsache auf praktische Anwendungen und in diesem Zuge auf effiziente Decodierverfahren und -algorithmen ausgerichtet.

**4.1.77 Bemerkung**
Neben diesen drei wichtigen Familien Fehler-korrigierender Codes, Hamming-, BCH- und Reed-Solomon-Codes, gibt es noch weitere Codeklassen mit guten Fehlerkorrektureigenschaften. Diese erwähnen wir an dieser Stelle nur kurz und gehen auf deren genaue Struktur hier nicht näher ein. In [7, 43, 47, 27], beispielsweise, werden darüber hinaus *Golay-Codes* und *Reed-Muller-Codes* $\mathcal{R}(r,m)$ der Ordnung $r$ sowie *Simplex-Codes* $\mathcal{S}(m)$ analysiert. Dabei ist jeder Simplex-Code $\mathcal{S}(m)$ gerade der duale Code $\mathcal{C}^{\perp}$ zu einem Hamming-Code $\mathcal{H}$.

## 4.2 Berechnung der Restfehlerwahrscheinlichkeit

Im vorangegangenen Abschnitt 4.1 haben wir als eine Möglichkeit zur Absicherung von Information gegenüber zufälligen Störungen das Konzept der Blockcodes kennengelernt. Ferner haben wir spezielle Klassen von Blockcodes genauer erörtert sowie einige prominente Beispiele für die Klasse der zyklischen Codes angegeben. In diesem Abschnitt werden wir nun die Fehlerkorrektureigenschaften dieser Codierverfahren detaillierter analysieren. Schließlich zielt die Fehlerkorrektur darauf ab, beim Empfänger eine möglichst geringe Fehlerrate zu erreichen. So ist eine zentrale Fragestellung in dieser Arbeit, in Abhängigkeit vom betrachteten Übertragungskanal dasjenige Codierverfahren zu bestimmen, das eine möglichst geringe Fehlerrate liefert. Dies führt zu der grundlegenden Idee, einen optimalen Code zur Fehlerkorrektur basierend auf den Übertragungseigenschaften des Kanals zu bestimmen. Mit Blick auf diese Zielsetzung sind daher die vorgestellten zyklischen Codes in Abhängigkeit von den Übertragungseigenschaften des Kanals zu bewerten. Hierzu benötigen wir ein geeignetes Kriterium als Maßstab.

Im allgemeinen wird der Erfolg bzw. die Effizienz eines zur Datensicherung eingesetzten zyklischen Codes $\mathcal{C}$ anhand der Restfehlerwahrscheinlichkeit des verwendeten Codes $\mathcal{C}$ gemessen. Die Restfehlerwahrscheinlichkeit entspricht dabei gerade der Wahrscheinlichkeit, dass trotz des Einsatzes eines Codierverfahrens verfälschte Wörter unbemerkt zum Empfänger gelangen. Ziel dieses Abschnittes ist es nun, eine geeignete Basis für eine möglichst realistische Bewertung Fehler-korrigierender zyklischer Blockcodes zu schaffen, die unterschiedliche

Fehlerkorrektureigenschaften besitzen. Dazu konkretisieren wir den Begriff der Restfehlerwahrscheinlichkeit als Bewertungskriterium. Ferner entwickeln wir ein geeignetes Schema zur Berechnung der Restfehlerwahrscheinlichkeit eines zur Fehlerkorrektur eingesetzten zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$.

Hierzu kommen wir zunächst auf die Darstellung einer Folge $(f_t)_{t\in N}$ aus Abschnitt 3.3 zurück. In Bemerkung 3.3.1 haben wir die Folge $(f_t)_{t\in N}$ der aufgetretenen Bitfehler als einen zufälligen Fehlerprozeß und damit als einen stochastischen Prozeß $(E_t)_{t\in N}$ interpretiert. Da stochastische Prozesse laut Definition 3.1.1 die Möglichkeit bieten, komplexere Formen der Abhängigkeit zwischen Mengen von Zufallsvariablen $(E_t)_{t\in N}$ zu charakterisieren, können wir zeitliche Abhängigkeiten zwischen den aufgetretenen Übertragungsfehlern beschreiben und mittels verschiedener Kanalmodelle aus Abschnitt 3.3 abbilden. In Abschnitt 4.1 haben wir uns ausschließlich mit der Verfälschung eines zu übertragenden Codewortes $\mathbf{c} \in \mathcal{C}$ durch ein Fehlermuster $\mathbf{f} \in GF(2)^n$ befaßt, was in Abschnitt 4.1 mittels der binären Addition $\mathbf{c} + \mathbf{f}$ dargestellt worden ist. Eine Verknüpfung dieser Darstellungen aus Abschnitt 3.3 und Abschnitt 4.1 erfolgt folgendermaßen:

Wir interpretieren die Folge der Komponenten eines Codewortes $\mathbf{c} \in \mathcal{C}$ aus Abschnitt 4.1 als die codierte Folge $(c_t)_{t\in N}$ aus Abschnitt 3.2 bzw. 3.3 und damit als stochastischen Prozeß $(C_t)_{t\in N}$ aus Bemerkung 3.3.1. Analog bilden die Komponenten eines beliebigen Fehlermusters $\mathbf{f} \in GF(2)^n$ aus Abschnitt 4.1 eine Folge, die mit der Folge $(f_t)_{t\in N}$ der aufgetretenen Bitfehler und damit mit dem stochastischen Fehlerprozeß $(E_t)_{t\in N}$ identifiziert werden kann.

### 4.2.1 Bemerkung

(i) Für einen zyklischen $(n, k, d)$-Blockcode $\mathcal{C}$ entspricht jede Komponente eines Codewortes $\mathbf{c} = (c_0, \ldots, c_{n-1}) \in \mathcal{C}$ einem Zustand des stochastischen Prozesses $(C_t)_{t\in N}$ zu einem Zeitpunkt $t$.

(ii) Ebenso entspricht jede Komponente eines Fehlermusters $\mathbf{f} = (f_0, \ldots, f_{n-1}) \in GF(2)^n$ einem Zustand des Fehlerprozesses $(E_t)_{t\in N}$ zu einem Zeitpunkt $t$.

Um diese Interpretation zu konkretisieren und darauf aufbauend ein Schema zur Berechnung der Restfehlerwahrscheinlichkeit herleiten zu können, vereinbaren wir die folgende Notation.

### 4.2.2 Bezeichnungen

*Im folgenden bezeichnen wir ein Fehlermuster* $\mathbf{f} \in GF(2)^n$ *der Länge* $n$ *mit*

$$\mathbf{f_n} := (f_0, \ldots, f_{n-1}),$$

*wobei* $f_i \in GF(2)$ *für jedes* $0 \leq i \leq n-1$ *ist.*

Die Restfehlerwahrscheinlichkeit eines zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ ist charakterisiert als diejenige Wahrscheinlichkeit, mit der ein verfälschtes Wort unbemerkt zum Empfänger gelangt. Da die Verfälschung eines Codewortes $\mathbf{c} \in \mathcal{C}$ durch ein Fehlermuster $\mathbf{f_n}$ in Abschnitt 4.1 mittels der binären Addition dargestellt worden ist, folgt aus Bezeichnung 4.2.2, dass die Menge der Fehlermuster $\mathbf{f_n}$ der Länge $n$, die nicht mittels des eingesetzten zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ korrigiert werden können, die Restfehlerwahrscheinlichkeit des betrachteten zyklischen Codes $\mathcal{C}$ bestimmt. Daher definieren wir:

**4.2.3 Definition**
*Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Dann bezeichnen wir*

*(i) die Menge aller durch den Code $\mathcal{C}$ korrigierbaren Fehlermuster $\mathbf{f_n} \in GF(2)^n$ mit*

$$\mathcal{F}_\mathcal{C} := \{\mathbf{f_n} \in GF(2)^n \ ; \ \mathbf{f_n} \text{ korrigierbares Fehlermuster von } \mathcal{C}\}$$

*(ii) und die Wahrscheinlichkeit für ein beliebiges Fehlermuster $\mathbf{f_n} \in GF(2)^n$ mit $P(\mathbf{f_n})$.*

*Die Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}$ des betrachteten zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ ist dann gegeben durch*

$$\mathcal{R}_\mathcal{C} := 1 - \sum_{\mathbf{f_n} \in \mathcal{F}_\mathcal{C}} P(\mathbf{f_n}).$$

Damit haben wir die Berechnung der Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}$ eines zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ darauf reduziert, zunächst die Menge $\mathcal{F}_\mathcal{C}$ der von $\mathcal{C}$ korrigierbaren Fehlermuster $\mathbf{f_n}$ und dann deren Wahrscheinlichkeit $P(\mathbf{f_n})$ zu bestimmen.

**4.2.4 Korollar**
*Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Dann läßt sich die Wahrscheinlichkeit eines beliebigen Fehlermusters $\mathbf{f_n} \in GF(2)^n$ mittels des während der Übertragung aufgetretenen Fehlerprozesses $(E_t)_{t \in N}$ aus der folgenden Beziehung ermitteln:*

$$P(\mathbf{f_n}) = P\big((E_0 = f_0), (E_1 = f_1), \ldots, (E_{n-1} = f_{n-1})\big)$$

**Beweis**
Sei $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ ein beliebiges Fehlermuster der Länge $n$. Dann repräsentiert jede Komponente von $\mathbf{f_n}$ nach Bemerkung 4.2.1 (ii) einen Zustand des Fehlerprozesses $(E_t)_{t \in N}$ zu dem entsprechenden Zeitpunkt. Folglich gilt für jedes $i$ mit $0 \leq i \leq n-1$:

$$f_i = E_i$$

Damit erhalten wir für die Wahrscheinlichkeit eines beliebigen Fehlermusters $\mathbf{f_n}$ die obige Beziehung. Die Berechnung der Wahrscheinlichkeit $P(\mathbf{f_n})$ eines Fehlermusters $\mathbf{f_n}$ hängt offensichtlich von dem während der Übertragung erzeugten Fehlerprozeß $(E_t)_{t \in N}$ und dessen Eigenschaften ab. □

In Abschnitt 3.3 haben wir verschiedene Kanalmodelle kennengelernt und gezeigt, dass damit unterschiedliche Formen der Abhängigkeit zwischen den aufgetretenen Übertragungsfehlern modelliert werden können. Dabei haben wir grob zwischen einem gedächtnislosen und einem gedächtnisbehafteten Übertragungskanal unterschieden. So haben wir in Abschnitt 3.3.1 den symmetrischen Binärkanal (BSC) als einen gedächtnislosen Kanal kennengelernt. Dessen Fehlerprozeß wird durch unabhängige, identisch verteilte Zufallsvariablen $(E_t)_{t \in N}$ beschrieben. Folglich sind Übertragungsfehler im Falle eines

gedächtnislosen Kanals unabhängig voneinander und zufällig verteilt; d.h. korrelierte Fehlerpositionen existieren im Prinzip nicht. Solche Fehler bezeichnen wir im folgenden als *random errors*.

Im Gegensatz zum BSC haben wir in Abschnitt 3.4 zur Modellierung eines gedächtnisbehafteten Kanals ein endliches Markov Modell verwendet. In diesem Fall wird der Fehlerprozeß $(E_t)_{t\in \boldsymbol{N}}$ mittels voneinander abhängiger, korrelierter Zufallsvariablen $(E_t)_{t\in \boldsymbol{N}}$ gekennzeichnet. Folglich treten Übertragungsfehler im Falle eines gedächtnisbehafteten Kanals in zeitlicher Nähe auf und sind voneinander abhängig bzw. korreliert. In diesem Zusammenhang sprechen wir von einem *burst error* oder einem *Bündelfehler*. In diesem Sinne unterscheiden wir im folgenden zwei Fehlertypen, *random errors* und *burst errors*.

Bei Untersuchungen bezüglich der Fehlerkorrigierbarkeit in Abschnitt 4.1.1, insbesondere in Satz 4.1.19 und Korollar 4.1.20, sind wir bislang davon ausgegangen, dass Übertragungsfehler stets zufällig verteilt sind und vornehmlich als *random errors* auftreten. In Kapitel 2 haben wir im Zusammenhang mit der Motivation dieser Arbeit bereits darauf verwiesen, dass in realen Übertragungskanälen Übertragungsfehler zumeist in einem zeitlichen Abhängigkeitsverhältnis stehen. Somit ist die Generierung von Bündelfehlern wahrscheinlicher.

Ein Schwerpunkt dieser Arbeit ist daher, verschiedene Codierverfahren, und insbesondere die in der Arbeit vorgestellten zyklischen Codes, auf ihr Fehlerkorrekturverhalten hin zu analysieren; d.h. es wird überprüft, ob ein Codierverfahren *random* oder *burst errors* effizienter korrigiert. Dazu muß der Umstand, dass während der Übertragung Bündelfehler auftreten können, in die Bewertung verschiedener zyklischer Codes mit einfließen und damit bei der Berechnung der Restfehlerwahrscheinlichkeit beachtet werden. In den folgenden Abschnitten zeigen wir dazu, dass sich alle zyklischen $(n, k, d)$-Blockcodes durch unterschiedliches Fehlerkorrekturverhalten auszeichnen. Vor diesem Hintergrund berechnen wir die Restfehlerwahrscheinlichkeit zyklischer Blockcodes einerseits auf der Grundlage eines gedächtnislosen Kanals und andererseits basierend auf einem gedächtnisbehafteten Kanal. Dabei repräsentiert der letztere Fall eine wesentlich realistischere Basis für einen Vergleich verschiedener zyklischer Blockcodes. Ein derartiger Ansatz ist in dieser Form in der Literatur bislang nicht bekannt. Dort wird in der Regel lediglich der BSC als Grundlage betrachtet. Ferner entwickeln wir ein geeignetes Schema sowie entsprechende Formeln zur Berechnung der Restfehlerwahrscheinlichkeit eines zyklischen Codes $\mathcal{C}$ basierend auf einem endlichen Markov Modell.

### 4.2.1 Restfehlerwahrscheinlichkeit auf der Basis des BSC

In diesem Abschnitt befassen wir uns mit einem gedächtnislosen Übertragungskanal. Wir gehen davon aus, dass während der Übertragung in erster Linie voneinander unabhängige, zufällig verteilte Einzelfehler generiert werden.

**4.2.5 Definition**
*Im folgenden werden wir voneinander unabhängige und zufällig verteilte Einzelfehler als random errors bezeichnen.*

Erzeugt der betrachtete Übertragungskanal fast ausschließlich *random errors* und besteht der Fehlerprozeß $(E_t)_{t\in\mathbb{N}}$ somit aus unabhängigen, identisch verteilten Zufallsvariablen, so ist laut Abschnitt 3.3.1 ein BSC ein zur Modellierung dieses Übertragungskanals geeignetes Kanalmodell. Auf der Basis des BSC berechnen wir nun die Restfehlerwahrscheinlichkeit eines zur Fehlerkorrektur eingesetzten zyklischen $(n,k,d)$-Blockcodes $\mathcal{C}$. Laut Definition 4.2.3 ist die Restfehlerwahrscheinlichkeit von $\mathcal{C}$ gegeben durch die Menge der von $\mathcal{C}$ korrigierbaren Fehlermuster sowie deren Wahrscheinlichkeit:

$$\mathcal{R}_{\mathcal{C}} = 1 - \sum_{\mathbf{f_n}\in\mathcal{F}_{\mathcal{C}}} P(\mathbf{f_n}).$$

**4.2.6 Satz**
*Gegeben sei ein beliebiger zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$, dann umfaßt die Menge $\mathcal{F}_{\mathcal{C}}$ der korrigierbaren Fehlermuster alle Fehlermuster $\mathbf{f_n}$ vom Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor$:*

$$\mathcal{F}_{\mathcal{C}} = \left\{ \mathbf{f_n} \in GF(2)^n \;;\; \mathrm{wt}(\mathbf{f_n}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor \right\}$$

*Dabei legen wir als Decodierprinzip eine Maximum-Likelihood-Decodierung zugrunde.*

**Beweis**
Unter der Annahme, dass das zugrunde liegende Kanalmodell ein BSC ist, die Übertragungsfehler also unabhängig voneinander sind, haben wir in Abschnitt 4.1, Satz 4.1.19 bzw. Korollar 4.1.20, Aussagen zur Fehlerkorrigierbarkeit für einen zyklischen Code $\mathcal{C}$ bewiesen. So korrigiert ein $(n,k,d)$-Blockcode $\mathcal{C}$ unter diesen Voraussetzungen bis zu $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehler. An dieser Stelle sei nochmals darauf hingewiesen, dass eine elementare Voraussetzung der Ausführungen in Abschnitt 4.1 stets die Unabhängigkeit der Übertragungsfehler gewesen ist.

In Abschnitt 4.1 haben wir ferner festgelegt, dass im Rahmen dieser Arbeit stets eine Maximum-Likelihood-Decodierung als Decodierprinzip angenommen wird. Als ein Beispiel für eine Maximum-Likelihood-Decodierung haben wir in diesem Zuge die Decodierung mittels eines *standard arrays* erörtert. Hierbei ist deutlich geworden, dass für einen zyklischen $(n,k,d)$-Blockcode $\mathcal{C}$ genau die $2^{n-k}$ *Coset-Leader* des *standard arrays* die Menge $\mathcal{F}_{\mathcal{C}}$ der korrigierbaren Fehlermuster ausmachen. Mit der Einführung des Syndrombegriffes in Definition 4.1.57 ist darüber hinaus ein simples Verfahren aufgezeigt worden, wie die *Coset-Leader* mittels des Syndroms $\mathbf{s}$ über die Beziehung $\mathbf{s} = \mathbf{f_n}\,\mathbf{H}^T$ zu bestimmen sind.

In diesem Kontext haben wir zudem gezeigt, dass genau diejenigen Fehlermuster $\mathbf{f_n}$ korrigierbar sind, die minimales Gewicht besitzen. Das Prinzip der Maximum-Likelihood-Decodierung besagt nämlich, dass dasjenige Codewort $\mathbf{c} \in \mathcal{C}$ decodiert wird, das zu dem empfangenen Binärwort $\mathbf{r}$ minimalen Abstand besitzt bzw. das durch ein Fehlermuster $\mathbf{f_n}$ minimalen Gewichts in das empfangene Wort $\mathbf{r}$ verfälscht wird, also mit

$$\min_{\mathbf{c}\in\mathcal{C}}\{\mathrm{dist}(\mathbf{c},\mathbf{r})\} = \min_{\mathbf{c}\in\mathcal{C}}\{\mathrm{wt}(\mathbf{c}+\mathbf{r})\} = \min_{\mathbf{c}\in\mathcal{C}}\{\mathrm{wt}(\mathbf{c}+\mathbf{c}+\mathbf{f_n})\} = \min_{\mathbf{f_n}\in GF(2)^n}\{\mathrm{wt}(\mathbf{f_n})\}$$

Demzufolge müssen mit Satz 4.1.19 alle Fehlermuster $\mathbf{f_n}$ mit Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ auf unterschiedliche Syndrome $\mathbf{s} \in GF(2)^{n-k}$ führen und damit korrigierbar sein. Die Menge $\mathcal{F}_{\mathcal{C}}$

aller korrigierbaren Fehlermuster $\mathbf{f_n}$ umfaßt daher diejenigen Fehlermuster, die vom Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ sind:

$$\mathcal{F}_\mathcal{C} = \left\{ \mathbf{f_n} \in GF(2)^n \; ; \; \mathrm{wt}(\mathbf{f_n}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor \right\}$$

□

Für die Wahrscheinlichkeit $P(\mathbf{f_n})$ eines einzelnen beliebigen Fehlermusters $\mathbf{f_n}$ gilt auf der Basis des BSC im allgemeinen die folgende Beziehung:

**4.2.7 Satz**
*Gegeben sei ein beliebiger zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Als Basis der Berechnungen werde der BSC mit der Bitfehlerrate $p_E$ als variablem Parameter zugrunde gelegt. Dann ist die Wahrscheinlichkeit eines einzelnen beliebigen Fehlermusters $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ der Länge $n$, das $i$ Übertragungsfehler mit $1 \leq i \leq n$ beinhaltet, gegeben durch:*

$$P(\mathbf{f_n}) \;\; = \;\; p_E^i (1 - p_E)^{n-i}$$

**Beweis**
Es sei ein beliebiges Fehlermuster $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ gegeben, das genau $i$ Übertragungsfehler mit $1 \leq i \leq n$ enthält. Aus Korollar 4.2.4 folgt, dass sich die Wahrscheinlichkeit des Fehlermusters $\mathbf{f_n}$ aus dem während der Übertragung produzierten Fehlerprozeß $(E_t)_{t \in N}$ über die Gleichung

$$P(\mathbf{f_n}) = P\big((E_0 = f_0), \ldots, (E_{n-1} = f_{n-1})\big)$$

ermitteln läßt.

Da ein BSC die Basis der nachfolgenden Berechnungen darstellt, wird laut Abschnitt 3.3.1 und Definition 3.3.4 jedes einzelne zu übertragende Bit unabhängig von den zuvor aufgetretenen Fehlern mit der gleichen Bitfehlerrate $p_E$ verfälscht. Aufgrund der Gedächtnislosigkeit des BSC bzw. der Unabhängigkeit der Zufallsvariablen $(E_t)_{t \in N}$ besitzen die Positionen der Fehler keinen Einfluß auf die Wahrscheinlichkeit $P(\mathbf{f_n})$ des betrachteten Fehlermusters $\mathbf{f_n}$. Demzufolge ist lediglich die Anzahl $i$ der aufgetretenen Übertragungsfehler ausschlaggebend für die Wahrscheinlichkeit $P(\mathbf{f_n})$ des Fehlermusters $\mathbf{f_n}$. Da die Zufallsvariablen $(E_t)_{t \in N}$ ferner alle identisch mit der Bitfehlerrate $p_E$ verteilt sind, ergibt sich die Wahrscheinlichkeit eines Fehlermusters $\mathbf{f_n}$ zu:

$$\begin{aligned} P(\mathbf{f_n}) &= P\big((E_0 = f_0), \ldots, (E_{n-1} = f_{n-1})\big) \\ &= p_E^i (1 - p_E)^{n-i}, \end{aligned}$$

falls das Fehlermuster $\mathbf{f_n}$ genau $i$ Übertragungsfehler beinhaltet. □

**4.2.8 Beispiel**
Gegeben sei ein beliebiges Fehlermuster $\mathbf{f_n}$ der Länge $n$, das genau $i$ Fehler mit $1 \leq i \leq n$ enthält. Dann sind die folgenden Fehlermuster auf der Basis des BSC gleich wahrscheinlich:

00000...000...000...0001111000...000...000...00000

und

0000...0001000...0001000...0001000...000100...0000

Aus dem Beweis des vorherigen Satzes geht hervor, dass die Unabhängigkeit der Zufallsvariablen $(E_t)_{t\in N}$ impliziert, dass lediglich die Anzahl $i$ der Fehler, nicht aber deren Positionen, für die Wahrscheinlichkeit eines Fehlermusters entscheidend ist. Eventuell bestehende Korrelationen zwischen den Übertragungsfehlern werden somit bei der Berechnung der Wahrscheinlichkeit $P(\mathbf{f_n})$ auf der Grundlage des BSC ignoriert.

**4.2.9 Satz**
*Gegeben sei ein beliebiger zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Dann ergibt sich die Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}$ von $\mathcal{C}$ auf der Basis des BSC zu:*

$$\mathcal{R}_\mathcal{C} = 1 - \sum_{i=0}^{\left\lfloor \frac{d-1}{2} \right\rfloor} p_E^i (1-p_E)^{n-i}.$$

**Beweis**
Nach Definition 4.2.3 ist die Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}$ gegeben durch

$$\mathcal{R}_\mathcal{C} = 1 - \sum_{\mathbf{f_n} \in \mathcal{F}_\mathcal{C}} P(\mathbf{f_n}).$$

In Satz 4.2.6 haben wir die Menge $\mathcal{F}_\mathcal{C}$ aller von $\mathcal{C}$ korrigierbaren Fehlermuster $\mathbf{f_n}$ als die Menge aller Fehlermuster $\mathbf{f_n}$ vom Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ bestimmt. Ferner ist in Satz 4.2.7 die Wahrscheinlichkeit eines Fehlermusters $\mathbf{f_n}$ mit $i$ Fehlern berechnet worden:

$$P(\mathbf{f_n}) = p_E^i (1-p_E)^{n-i}$$

Somit ergibt sich aus den Sätzen 4.2.6 und 4.2.7 die Behauptung. □

Satz 4.2.9 liefert eine einfache geschlossene Formel, um die Restfehlerwahrscheinlichkeit eines beliebigen zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ auf der Basis des BSC zu berechnen. Diese hängt lediglich von der Länge $n$ des Blockcodes, dessen Mindestdistanz $d$ und der Bitfehlerrate $p_E$ des Übertragungskanals als dem einzigen, variablen Kanalparameter ab. In dieser Formel werden die unterschiedlichen algebraischen Eigenschaften, die verschiedene zyklische $(n, k, d)$-Blockcodes besitzen, jedoch nicht berücksichtigt. Die in Abschnitt 4.1 vorgestellten Familien zyklischer Fehler-korrigierender Codes zeichnen sich allerdings durch völlig unterschiedliche algebraische und zahlentheoretische Eigenschaften aus. Diese fließen nun also nicht in die Berechnung der Restfehlerwahrscheinlichkeit und somit auch nicht in die Bewertung verschiedener Fehler-korrigierender Codes mit ein.

In Abschnitt 3.3.1 haben wir den BSC als das bekannteste aber auch einfachste Kanalmodell charakterisiert. In den meisten Lehrbüchern, wie auch in den Standardwerken von

[7, 26, 34, 43, 47], sowie vielen Publikationen zum Thema Codierungstheorie wird der BSC als einziges Kanalmodell detailliert vorgestellt. Dort dient er als klassisches Modell für einen digitalen fehlerbehafteten Kanal. Da der BSC einen gedächtnislosen Fehlerprozeß $(E_t)_{t\in N}$ mit einer konstanten Bitfehlerrate $p_E$ als dem einzigen variablen Kanalparameter darstellt, ist das Modell des BSC mathematisch äußerst einfach zu handhaben. Insbesondere läßt sich die Bitfehlerrate $p_E$ sehr leicht aus vorhandenem Datenmaterial schätzen. Aufgrund der simplen Handhabung des Modells dient der BSC daher in den meisten Lehrbüchern und Veröffentlichungen zum Thema Codierungstheorie als Basis zur Analyse unterschiedlicher zur Fehlerkorrektur eingesetzter Codes. Obgleich die unterschiedlichen algebraischen Eigenschaften verschiedener Codes bei einer Berechnung auf der Basis des BSC nicht angemessen berücksichtigt werden, wird der BSC als Grundlage für die Berechnung der Restfehlerwahrscheinlichkeit verwendet. Zwar werden in vielen Lehrbüchern verschiedene Familien zyklischer Codes mit ihren unterschiedlichen algebraischen Merkmalen ausführlich vorgestellt, aber ihre algebraischen Vorteile können aus den genannten Gründen nicht in entsprechender Weise genutzt werden.

Ferner ist die Annahme, dass Übertragungsfehler unabhängig voneinander auftreten, mit Blick auf reale Anwendungen unzutreffend. Wenn bei einer Glasfaser-Übertragungsstrecke beispielsweise ca. 99,97 % der Bitfehler unabhängig voneinander aufzutreten scheinen, so können die übrigen 0,03 % voneinander abhängigen, korrelierten Fehler zu wesentlichen Abweichungen bezüglich der Restfehlerwahrscheinlichkeit führen. Wie in Abschnitt 3.3.1 angedeutet, haben Messungen der DLR ergeben, dass Übertragungsfehler in realen Kanälen in zeitlicher Nähe auftreten. Ein Auszug dieser Messungen ist in der Arbeit von Kröll, [24], angegeben. Folglich sind reale Kanäle in der Regel Kanäle mit einer weniger einfachen Struktur als der des BSC. Diesem Umstand ist mittels eines geeigneten Kanalmodells, mit dem auch gedächtnisbehaftete Kanäle abgebildet werden können, Rechnung zu tragen. In dem Buch von [52] ist bereits angeregt worden, basierend auf einem Kanalmodell mit Gedächtnis die Restfehlerwahrscheinlichkeit verschiedener Codes zu berechnen, um so eine realistischere Grundlage für eine Bewertung verschiedener Fehler-korrigierender Codes zu schaffen.

### 4.2.2 Restfehlerwahrscheinlichkeit auf der Basis eines gedächtnisbehafteten Kanals

Zum Abschluß des vorherigen Abschnittes haben wir entscheidende Nachteile aufgezeigt, falls der BSC als Grundlage zur Berechnung der Restfehlerwahrscheinlichkeit und damit zur Analyse verschiedener zyklischer Fehler-korrigierender Codes dient. So können erstens mit dem BSC keine korrelierten Fehler modelliert werden und zweitens können die unterschiedlichen algebraischen Strukturen und zahlentheoretischen Eigenschaften verschiedener Fehler-korrigierender Codes nicht in die Berechnung der Restfehlerwahrscheinlichkeit einfließen. Im Gegensatz zu diesem bisher in der Literatur verfolgten Ansatz werden wir in diesem Abschnitt ein Kanalmodell zugrunde legen, mit welchem auch gedächtnisbehaftete Kanäle entsprechend modelliert werden können. Ein solches Kanalmodell bietet schließlich eine realistischere Basis für die Berechnung der Restfehlerwahrscheinlichkeit und Analyse der Fehlerkorrektureigenschaften verschiedener zyklischer Codes.

Nun besitzen reale Kanäle eine weniger einfache Struktur als der BSC. Dies zeigt insbesondere der in der Arbeit von C. Kröll, [24], angegebene Auszug aus einer Fehlermessung für eine

Funkübertragungsstrecke, die im Auftrag der DLR durchgeführt worden ist. Dieser Auszug bestätigt, dass im Falle dieses realen Kanals Übertragungsfehler, entgegen den Annahmen des BSC, nicht zufällig verteilt sind, sondern häufig direkt hintereinander in längeren Ketten mit vereinzelten Lücken auftreten. In realen Kanälen treten Fehler demnach häufig in zeitlicher Nähe auf, in sogenannten *Clustern* oder *Bursts*. Die Positionen der Fehler sind also korreliert. Wir differenzieren daher zwischen den beiden folgenden Fehlertypen.

**4.2.10 Definition**

*(i) Voneinander unabhängige, zufällig verteilte Einzelfehler werden als random errors bezeichnet.*

*(ii) Korrelierte Fehler, die in zeitlicher Nähe auftreten, werden zu einem sogenannten burst error zusammengefaßt, oder auch Bündel- bzw. Burstfehler genannt. Somit repräsentiert ein Bündelfehler einen Vektor der Länge $b$, mit $b \geq 2$, der mit einem Fehler beginnt und endet und ansonsten aus Fehlern oder korrekt übertragenen Bits besteht.*

*(iii) Der Parameter $b$ heißt dann die Länge des Bündelfehlers bzw. des Bursts.*

Aufgrund der Differenzierung von zwischen *random* und *burst errors*, klassifizieren wir nun die zur Fehlerkorrektur eingesetzten zyklischen $(n, k, d)$-Blockcodes im Hinblick auf ihr Korrekturverhalten.

**4.2.11 Definition**

*Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$.*

*(i) Korrigiert $\mathcal{C}$ bis zu $r$ random errors, so heißt $\mathcal{C}$ ein $r$-random-error-correcting-Code.*

*(ii) Korrigiert $\mathcal{C}$ sämtliche burst errors bis zur Länge $b$, so heißt $\mathcal{C}$ ein $b$-burst-error-correcting-Code.*

*(iii) Abkürzend sprechen wir auch von einem $r$-REC-Code bzw. einem $b$-BEC-Code.*

Je nachdem, ob ein zyklischer Code eher *random* oder *burst errors* korrigieren kann, nehmen wir eine Einteilung der zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ in REC- und BEC-Codes vor. Mit Blick auf das Fehlerkorrekturverhalten analysieren wir nun die Schranken der Fehlerkorrigierbarkeit sowohl für REC-Codes als auch für BEC-Codes. Für einen $r$-REC-Code haben wir in Satz 4.1.19 bzw. Korollar 4.1.20 diese Schranke bereits bestimmt.

**4.2.12 Korollar**

*Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Ist der betrachtete Code $\mathcal{C}$ ein $r$-REC-Code, so gilt*

$$r \leq \left\lfloor \frac{d-1}{2} \right\rfloor .$$

**4.2.13 Korollar**
*Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Ist der betrachtete Code $\mathcal{C}$ ein $b$-BEC-Code, so gilt*

$$b \leq \left\lfloor \frac{n-k}{2} \right\rfloor .$$

**Beweis**
Zum Beweis dieser Aussage zeigen wir zunächst, dass ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$ zur Korrektur sämtlicher *burst errors* bis zur Länge $b$ mindestens $2b$ Kontrollstellen benötigt. Folglich muß gelten:

$$n - k \geq 2b.$$

Angenommen, es gäbe zwei Fehlermuster der Länge $n$, die jeweils einen *burst error* der Länge $b$ beinhalten. Ferner ergäbe die Addition dieser beiden Fehlermuster wieder ein Fehlermuster, das seinerseits einen *burst error* der Länge $2b$ enthalte. Würden diese beiden Fehlermuster in dem selben Coset liegen, dann liefert Definition 4.1.55, dass die Summe dieser beiden Fehlermuster ein Codewort ergeben würde. Nach den Ausführungen zum Aufbau eines *standard arrays* in Abschnitt 4.1.2 wären diese beiden Fehlermuster dann nicht korrigierbar. Dies ist ein Widerspruch zur Voraussetzung, dass wir einen $b$-BEC-Code betrachten und daher alle Fehlermuster mit einem *burst error* bis zur Länge $b$ korrigierbar sind.

Demzufolge kann ein Fehlermuster der Länge $n$ mit einem *burst error* der Länge $2b$ kein Codewort sein. Es existiert also kein Codewort, das an $2b$ aufeinanderfolgenden Stellen ungleich Null und sonst gleich Null ist. Daher müssen alle Fehlermuster der Länge $n$, die außer an $2b$ gegebenen Stellen gleich Null sind, in verschiedenen Cosets liegen. Aus diesem Grund müssen mindestens $2^{2b}$ verschiedene Cosets existieren. Da nach Satz 4.1.56 (iii) die Anzahl der verschiedenen Cosets gerade $2^{n-k}$ beträgt, besitzt der betrachtete $b$-BEC-Code $\mathcal{C}$ also mindestens $2b$ Kontrollstellen, so dass gilt:

$$n - k \geq 2b.$$

Als Schranke für die Fehlerkorrigierbarkeit eines $b$-BEC-Codes $\mathcal{C}$ der Länge $n$ mit $n - k$ Kontrollstellen ergibt sich daraus:

$$b \leq \left\lfloor \frac{n-k}{2} \right\rfloor .$$

□

Diese obere Schranke für die Fehlerkorrigierbarkeit eines zyklischen $b$-BEC-Codes der Länge $n$ ist 1960 von S. H. Reiger in seiner Arbeit [46] bewiesen worden. In der Literatur ist diese Schranke daher auch unter dem Namen *Reiger-Schranke* bekannt. Aufgrund eines Druckfehlers in einem Standardwerk ist diese Schranke in einigen Arbeiten unter dem Stichwort „Rieger-Schranke“ zu finden.

**4.2.14 Definition**
*Gegeben sei ein zyklischer $(n, k, d)$-Blockcode $\mathcal{C}$. Ist $\mathcal{C}$ ein $b$-BEC-Code, für den die Reiger-Schranke eine scharfe Schranke ist, so wird $\mathcal{C}$ auch als optimaler Code bezeichnet.*

*In diesem Zuge wird das Verhältnis $2b/(n-k)$ für b-BEC-Codes auch als Maß für deren Effizienz hinsichtlich der Korrektur von Burstfehlern verwendet.*

In Abschnitt 4.1.3 haben wir in den Definitionen 4.1.61, 4.1.65 und 4.1.73 prominente Beispiele für die Klasse der zyklischen $(n, k, d)$-Blockcodes vorgestellt, die Familie der Hamming-Codes, der BCH-Codes und der RS-Codes. Gemäß ihrer Definition sind diese durch unterschiedliche algebraische und zahlentheoretische Merkmale charakterisiert. Dabei bedingt die spezifische algebraische Struktur eines Codes sowie seine zahlentheoretischen Eigenschaften dessen Fehlerkorrektureigenschaften, d.h. ob er sich eher zur Korrektur einer bestimmten Anzahl von *random errors* bzw. zur Korrektur von *burst errors* bis zu einer gewissen Länge eignet, ob er folglich eher als REC-Code oder als BEC-Code charakterisiert werden kann. Konkret ablesbar ist das Fehlerkorrekturverhalten eines betrachteten Codes $\mathcal{C}$ an der Menge $\mathcal{F}_{\mathcal{C}}$ seiner korrigierbaren Fehlermuster, vgl. hierzu Definition 4.2.3. Dabei umfaßt gemäß den Ausführungen im Zusammenhang mit Definition 4.1.57 die Menge $\mathcal{F}_{\mathcal{C}}$ der korrigierbaren Fehlermuster gerade diejenigen Fehlermuster, die anhand ihrer Syndrome $\mathbf{s}$ unterschieden werden können. So haben wir gezeigt, daß bei einer Maximum-Likelihood-Decodierung mittels eines *standard arrays* die *Coset-Leader* des *standard arrays* gerade die korrigierbaren Fehlermuster $\mathbf{f_n}$ festlegen. Nun lassen sich die *Coset-Leader* auf sehr einfache Weise über die Syndrome $\mathbf{s} \in GF(2)^{n-k}$ und die Prüfmatrix $\mathbf{H}$ des betrachteten Codes $\mathcal{C}$ aus der Beziehung

$$\mathbf{s} = \mathbf{f_n}\,\mathbf{H}^T$$

ermitteln, vgl. Definition 4.1.57. Somit beeinflußt offensichtlich die Prüfmatrix $\mathbf{H}$ des betrachteten Codes bzw. dessen algebraische Struktur ganz entscheidend, welche Fehlermuster $\mathbf{f_n}$ in einem *standard array* die *Coset-Leader* ausmachen. Für einen $r$-REC-Code bedeutet dies nach den Ausführungen in Abschnitt 4.1.2, dass alle Fehlermuster $\mathbf{f_n}$ vom Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq r$ auf unterschiedliche Syndrome führen müssen, um als *Coset-Leader* ein korrigierbares Fehlermuster darzustellen. Ferner geht aus den Ausführungen im Anschluß an Definition 4.1.57 ebenfalls hervor, dass im Falle eines $b$-BEC-Codes alle Fehlermuster, die einen *burst error* der Länge $\leq b$ enthalten, auf verschiedene Syndrome führen müssen, um so als *Coset-Leader* zu den korrigierbaren Fehlermustern zu zählen. Folglich besitzen die Prüfmatrix $\mathbf{H}$ und in diesem Zuge die algebraische Struktur sowie die zahlentheoretischen Merkmale eines Codes erheblichen Einfluß darauf, ob die Menge der *Coset-Leader* und damit die Menge der korrigierbaren Fehlermuster vornehmlich aus Fehlermustern besteht, die in der Mehrzahl *random* oder *burst errors* umfassen.

Wie bereits mehrfach in Kapitel 2, Abschnitt 3.3 und Abschnitt 4.2 erwähnt, treten in realen Kanälen in der Regel sowohl *random* als auch *burst errors* auf. Vom Übertragungsmedium des Kanals hängt dabei ab, wie viele Burstfehler auftreten, wie lang diese sind und damit wie stark das Gedächtnis des betrachteten Kanals ist. So treten beispielsweise bei Funkübertragungsstrecken Burstfehler sehr häufig auf, vgl. Kapitel 2. Vom Übertragungsmedium des Kanals hängt damit also ab, ob der Einsatz eines REC-Codes dem eines BEC-Codes vorzuziehen ist. Um nun eine realistische Basis für einen aussagekräftigen Vergleich von REC- und BEC-Codes zu schaffen, ist bei der Berechnung der Restfehlerwahrscheinlichkeit des betrachteten Codes dessen Fehlerkorrekturverhalten zu berücksichtigen; d.h. die algebraischen und zahlentheoretischen Eigenschaften müssen in diese Berechnungen miteinbezogen werden. Dies wird Gegenstand der weiteren Ausführungen in diesem Abschnitt sein. So werden wir im Gegensatz zum vorherigen Abschnitt die Klassifizierung in

REC- und BEC-Codes in die Berechnung der Restfehlerwahrscheinlichkeit mit einbeziehen. Für beide Typen von Codes entwickeln wir dazu ein Berechnungsschema auf der Basis eines geeigneten Kanalmodells, welches das Gedächtnis eines Übertragungskanals in angemessener Weise abbildet. In den Abschnitten 3.3.2 bis 3.3.7 haben wir verschiedene, aus der Literatur bekannte Kanalmodelle vorgestellt, die Kanäle mit Gedächtnis und somit *random* und *burst errors* gleichermaßen abbilden können. Im Hinblick auf die Zielsetzung der Arbeit haben wir ferner deren Vor- und Nachteile analysiert, vgl. Paragraph 3.3. Dabei sind die wesentlichen Schwachstellen der bekannten Kanalmodelle erstens der Erneuerungscharakter der Modelle sowie zweitens die Unabhängigkeit der Zufallsvariablen, die die Aufenthaltszeiten des Prozesses in den einzelnen Zuständen beschreiben, vgl. Definition 3.1.18. Nach Bemerkung 3.1.21 sind diese stets geometrisch verteilt, was ebenfalls auf reale Kanäle nicht zutrifft. In Abschnitt 3.4 haben wir daher ein endliches Markov Modell als Kanalmodell mit Gedächtnis vorgeschlagen, das erstens die Modellierung von *random* und *burst errors* ermöglicht, zweitens mathematisch handhabbar bleibt und drittens darauf abzielt, bei der Berechnung der Restfehlerwahrscheinlichkeit die verschiedenen algebraischen Strukturen unterschiedlicher Codes zu berücksichtigen.

Ziel der Arbeit ist schließlich, basierend auf einem möglichst realistischen Vergleich von REC- gegenüber BEC-Codes in Abhängigkeit von den Eigenschaften des betrachteten Übertragungskanals, also der Stärke des Gedächtnisses eines realen Kanals, einen optimalen Code zur Fehlerkorrektur auszuwählen. Die nun folgenden Ausführungen bilden eine adäquate Grundlage hierzu. So leiten wir auf der Basis des in Abschnitt 3.4 präsentierten Kanalmodells ein geeignetes Schema zur Berechnung der Restfehlerwahrscheinlichkeit zyklischer $(n, k, d)$-Blockcodes $\mathcal{C}$ her. Zur Erinnerung fassen wir die wesentlichen Punkte des von uns in Abschnitt 3.4 vorgeschlagenen Kanalmodells kurz zusammen.

### 4.2.15 Bemerkung

- Zur Modellierung des während der Übertragung aufgetretenen Fehlerprozesses $(E_t)_{t\in\mathbb{N}}$ verwenden wir einen Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ mit einem endlichen Zustandsraum $\mathcal{S} = \{1, \ldots, N\}$.

- Der Markov'sche Hintergrudprozeß $(X_t)_{t\in\mathbb{N}}$ ist gekennzeichnet durch seinen Anfangszustand $X_0 = i_0$ mit $i_0 \in \mathcal{S}$ und die Übergangsmatrix $Q = (q_{ij})_{i,j\in\mathcal{S}}$, wobei für alle $i, j \in \mathcal{S}$ gilt:

$$\begin{aligned} q_{ij} &= P(X_{t+1} = j \mid X_t = i,\, X_{t-1} = i_{t-1}, \ldots, X_0 = i_0) \\ &= P(X_{t+1} = j \mid X_t = i) \end{aligned}$$

- Weiterhin setzen wir voraus, dass der zugrunde liegende Markov'sche Hintergrundprozeß homogen, irreduzibel und aperiodisch ist, so dass die stationären Zustandswahrscheinlichkeiten $\pi_i$ für jedes $i \in \mathcal{S}$ existieren.

- Ferner identifizieren wir die Zustände des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ mit den jeweiligen Fehlerwahrscheinlichkeiten

$$e_i = P(E_t = 1 \mid X_t = i) \quad \text{mit } e_i \in [0, 1] \quad \text{für jedes } i \in \mathcal{S}.$$

Mit diesen Vorgaben umgehen wir die wesentlichen Schwachstellen, die die übrigen in Abschnitt 3.3 vorgestellten Kanalmodelle aufweisen, den Erneuerungscharakter sowie die Unabhängigkeit der Zufallsvariablen, die die Aufenthaltszeiten in den einzelnen Zuständen $i \in \mathcal{S}$ beschreiben. Aus diesem Grund können wir neben der geometrischen Verteilung auch andere Verteilungen, wie z.B. die Poisson Verteilung, modellieren, worauf wir in Abschnitt 5.2.2 zurückkommen.

- Die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}$ eines beliebigen $(n, k, d)$-Blockcodes $\mathcal{C}$ ist laut Definition 4.2.3 gegeben durch

$$\mathcal{R}_{\mathcal{C}} = 1 - \sum_{\mathbf{f_n} \in \mathcal{F}_{\mathcal{C}}} P(\mathbf{f_n}).$$

- Dabei wird die Wahrscheinlichkeit eines beliebigen Fehlermusters $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ der Länge $n$ nach Korollar 4.2.4 bestimmt durch den während der Übertragung aufgetretenen Fehlerprozeß $(E_t)_{t \in \boldsymbol{N}}$; d.h. es gilt:

$$P(\mathbf{f_n}) = P\big((E_0 = f_0), \ldots, (E_{n-1} = f_{n-1})\big).$$

Unter der Annahme, dass die Zufallsvariablen $(E_t)_{t \in \boldsymbol{N}}$ unabhängig und identisch verteilt sind, haben wir in Abschnitt 4.2.1 eine geschlossene Formel für die Wahrscheinlichkeit eines einzelnen beliebigen Fehlermusters $\mathbf{f_n}$ angegeben. Im Gegensatz dazu impliziert die Konstruktion unseres Kanalmodells, dass der während der Übertragung erzeugte Fehlerprozeß $(E_t)_{t \in \boldsymbol{N}}$ stets vom jeweiligen Zustand des Kanals, also dem zugrunde liegenden Markov'schen Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ abhängt. Vor diesem Hintergrund wird die Wahrscheinlichkeit eines einzelnen beliebigen Fehlermusters $\mathbf{f_n}$ nicht nur von dem Fehlerprozeß $(E_t)_{t \in \boldsymbol{N}}$ bestimmt, sondern auch vom zugrunde liegenden Markov'schen Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ beeinflußt. Dies ist bei der Berechnung der Wahrscheinlichkeit $P(\mathbf{f_n})$ eines beliebigen Fehlermusters $\mathbf{f_n}$ entsprechend zu berücksichtigen. Im Hinblick auf die Berechnung der Wahrscheinlichkeit $P(\mathbf{f_n})$ eines beliebigen Fehlermusters $\mathbf{f_n}$ führt das von uns vorgeschlagene Kanalmodell auf ein sehr effizientes rekursives Schema. Hierzu führen wir die folgende Bezeichnung ein.

### 4.2.16 Bezeichnungen

*Gegeben sei ein beliebiges Fehlermuster* $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ *der Länge* $n$ *mit* $f_j \in GF(2)$ *für jedes* $j$ *mit* $0 \leq j \leq n-1$. *Dann bezeichnen wir die Wahrscheinlichkeit, dass der Fehlerprozeß* $(E_t)_{t \in \boldsymbol{N}}$ *genau das Fehlermuster* $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ *generiert, während der zugrunde liegende Markov'sche Hintergrundprozeß* $(X_t)_{t \in \boldsymbol{N}}$ *abschließend den Zustand* $i$ *mit* $i \in \mathcal{S} = \{1, \ldots, N\}$ *erreicht, mit*

$$p_i(\mathbf{f_n}) := P\big((E_0 = f_0), \ldots, (E_{n-1} = f_{n-1}), (X_{n-1} = i)\big).$$

Diese Bezeichnungsweise spiegelt wieder, dass auf der Basis unseres Kanalmodells die Wahrscheinlichkeit $P(\mathbf{f_n})$ eines beliebigen Fehlermusters $\mathbf{f_n}$ nicht nur vom Fehlerprozeß $(E_t)_{t \in \boldsymbol{N}}$ bestimmt wird, sondern auch vom Markov'schen Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ beeinflußt wird.

Unter diesen Vorgaben leiten wir nun rekursive Gleichungen zur Berechnung der Wahrscheinlichkeit $P(\mathbf{f_n})$ einzelner beliebiger vom Fehlerprozeß $(E_t)_{t\in N}$ generierter Fehlermuster $\mathbf{f_n}$ her. Mittels dieser rekursiven Gleichungen werden die Wahrscheinlichkeiten $p_i(\mathbf{f_{n-1}})$ und $p_i(\mathbf{f_n})$ für Fehlermuster der Längen $n-1$ und $n$ zueinander in Beziehung gesetzt. Auf diese Weise liefern sie ein effizientes rekursives Schema zur Berechnung der Wahrscheinlichkeit von Fehlermustern mit wachsender Länge $n$.

**4.2.17 Satz**
*Gegeben sei ein beliebiges Fehlermuster $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ der Länge $n$. Dann errechnet sich die Gesamtwahrscheinlichkeit $P(\mathbf{f_n})$ dieses Fehlermusters $\mathbf{f_n}$ aus den entsprechenden Wahrscheinlichkeiten $p_i(\mathbf{f_n})$ für jedes $i \in \mathcal{S}$ mittels der Gleichung*

$$P(\mathbf{f_n}) = \sum_{i=1}^{N} p_i(\mathbf{f_n}).$$

**Beweis**
Nun bezeichnet $P(\mathbf{f_n})$ die Gesamtwahrscheinlichkeit des Fehlermusters $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ und ist laut Korollar 4.2.4 gegeben durch:

$$P(\mathbf{f_n}) = P\big((E_0 = f_0), \ldots, (E_{n-1} = f_{n-1})\big)$$

Dann liefert die obige Bezeichnung 4.2.16

$$p_i(\mathbf{f_n}) = P\big((E_0 = f_0), \ldots, (E_{n-1} = f_{n-1}), (X_{n-1} = i)\big) \quad \text{mit } i \in \mathcal{S}$$

dass die Summe über alle $i \in \mathcal{S}$ der Wahrscheinlichkeiten $p_i(\mathbf{f_n})$ die Gesamtwahrscheinlichkeit $P(\mathbf{f_n})$ ergibt.

Die Wahrscheinlichkeit $p_i(\mathbf{f_n})$ berechnet sich in diesem Zusammenhang für jedes $i \in \mathcal{S}$ rekursiv aus den folgenden Gleichungen:

Sei dazu ein beliebiges Fehlermuster $\mathbf{f_{n-1}} = (f_0, \ldots, f_{n-2}) \in GF(2)^{n-1}$ der Länge $n-1$ gegeben und bezeichne $p_i(\mathbf{f_{n-1}})$, mit $i \in \mathcal{S}$, die Wahrscheinlichkeit dieses Fehlermusters $\mathbf{f_{n-1}}$ im Sinne von Bezeichnung 4.2.16. Dann ist die Wahrscheinlichkeit $p_i(\mathbf{f_n})$ mit $i \in \mathcal{S}$ für ein Fehlermuster

$$\mathbf{f_n} = (\mathbf{f_{n-1}}, f_{n-1}) = (f_0, \ldots, f_{n-2}, f_{n-1}) \quad \text{mit} \quad f_{n-1} \in GF(2)$$

rekursiv über die folgenden beiden Gleichungen bestimmt

$$\begin{aligned}
(i) \quad p_i(\mathbf{f_n}) &= p_i\big((\mathbf{f_{n-1}}, 0)\big) = \sum_{k=1}^{N} p_k(\mathbf{f_{n-1}})\, q_{ki}\,(1 - e_i), \quad \text{falls } f_{n-1} = 0 \text{ gilt} \\
(ii) \quad p_i(\mathbf{f_n}) &= p_i\big((\mathbf{f_{n-1}}, 1)\big) = \sum_{k=1}^{N} p_k(\mathbf{f_{n-1}})\, q_{ki}\, e_i, \qquad \text{falls } f_{n-1} = 1 \text{ gilt}
\end{aligned}$$

Zur Initialisierung dieser Rekursion wählen wir mit Blick auf die Existenz der stationären Zustandswahrscheinlichkeiten $\pi_i$ für jedes $i \in \mathcal{S}$

$$p_i(\mathbf{f_0}) := P(X_0 = i) = \pi_i \quad \text{für jedes } i \in \mathcal{S} = \{1, \ldots, N\}.$$

Sind sämtliche Wahrscheinlichkeiten $p_i(\mathbf{f_n})$ des Fehlermusters $\mathbf{f_n} = (\mathbf{f_{n-1}}, f_{n-1})$ für jedes $i \in \mathcal{S}$ rekursiv aus den obigen Gleichungen ermittelt, so summiert sich die Gesamtwahrscheinlichkeit eines beliebigen Fehlermusters $\mathbf{f_n}$ der Länge $n$ zu

$$P(\mathbf{f_n}) = \sum_{i=1}^{N} p_i(\mathbf{f_n}).$$

□

**4.2.18 Bemerkung**
Unter der Voraussetzung, dass die stationären Zustandswahrscheinlichkeiten $\pi_i$ existieren und bekannt sind, kann die Wahrscheinlichkeit eines beliebigen Fehlermusters $\mathbf{f_n}$ der Länge $n$ mittels der obigen rekursiven Gleichungen unter einem Gesamtaufwand der Ordnung $\mathcal{O}(nN^2)$ berechnet werden. Die Laufzeit der kompletten Auswertung dieses rekursiven Schemas wächst damit proportional zur Anzahl $\tilde{N}$ der von Null verschiedenen Elemente $q_{ij}$ der Übergangsmatrix $Q$. Falls die Übergangsmatrix $Q$ nur sehr dünn besetzt ist, ist der Aufwand sogar linear zur Anzahl $\tilde{N}$ der von Null verschiedenen Elemente von $Q$ und somit von der Ordnung $\mathcal{O}(n\tilde{N})$.

Wie eingangs erläutert, haben wir zur Ermittlung der Restfehlerwahrscheinlichkeit eines zyklischen $(n, k, d)$-Blockcodes $\mathcal{C}$ die Menge $\mathcal{F}_\mathcal{C}$ der korrigierbaren Fehlermuster $\mathbf{f_n}$ unter Berücksichtigung der Klassifizierung von Codes in REC- und BEC-Codes zu bestimmen. Im Falle eines $r$-REC-Codes müssen gerade alle Fehlermuster $\mathbf{f_n}$ vom Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq r$ unterschiedliche Syndrome besitzen und somit korrigierbar sein. Entsprechend müssen für einen $b$-BEC-Code alle Fehlermuster, die einen *burst error* der Länge $\leq b$ enthalten, auf verschiedene Syndrome führen. Um in die Berechnung der Restfehlerwahrscheinlichkeit das Fehlerkorrekturverhalten des betrachteten Fehler-korrigierenden Codes mit einzubeziehen, leiten wir nun auf der einen Seite eine Darstellung der Restfehlerwahrscheinlichkeit für einen $r$-REC-Code und auf der anderen Seite eine für einen $b$-BEC-Code her. Zu diesem Zweck entwickeln wir sowohl für REC-Codes als auch für BEC-Codes auf der Basis des in Abschnitt 3.4 vorgestellten Kanalmodells ein effizientes Rekursionsschema zur Bestimmung der Restfehlerwahrscheinlichkeit.

In ihrer Arbeit [53] von 1995 haben J. R. Yee und E. J. Weldon auf der Basis eines Markov'schen Hintergrundprozesses mit nur zwei Zuständen ein Rekursionsschema entwickelt, mit dem die Wahrscheinlichkeit ermittelt werden kann, dass $i$ beliebige Bits in einem Block der Länge $l$ fehlerhaft sind, während der Markov'sche Hintergrundprozeß abschließend einen der beiden Zustände erreicht hat. Folglich läßt sich hiermit die Wahrscheinlichkeit eines jeden Fehlermusters $\mathbf{f_n}$ der Länge $n$ errechnen, welches gerade $i$ zufällig verteilte Einzelfehler enthält. Mittels dieser Rekursion kann also die Restfehlerwahrscheinlichkeit eines $r$-REC-Codes bestimmt werden.

Da Yee und Weldon allerdings einen Markov'schen Hintergrundprozeß mit genau zwei Zuständen betrachten, beruht die von ihnen in [53] vorgestellte Rekursion auf dem Gilbert-Elliott-Modell. Dies besitzt laut Abschnitt 3.3.2 jedoch Erneuerungscharakter. Aufgrund dessen werden wir an dieser Stelle in mehrerer Hinsicht eine Erweiterung des von Yee und

Weldon präsentierten rekursiven Schemas entwickeln. Zunächst werden wir das vorgestellte Rekursionsschema für unsere Zwecke dahingehend erweitern, dass wir als Basis das von uns in Abschnitt 3.4 vorgeschlagene Kanalmodell zugrunde legen und somit als Grundlage einen Markov'schen Hintergrundprozeß mit $N$ Zuständen zulassen. Hierauf aufbauend entwickeln wir ein verallgemeinertes Rekursionsschema, mit dem wir die Wahrscheinlichkeit für $i$ zufällig verteilte Einzelfehler in einem Fehlermuster der Länge $n$ berechnen und damit die Restfehlerwahrscheinlichkeit eines $r$-REC-Codes auswerten können. Anschließend verallgemeinern wir dieses rekursive Schema in der Hinsicht, als dass wir entsprechende Rekursionsgleichungen für Fehlermuster ableiten, die Burstfehler bis zur Länge $b$ beinhalten. Zunächst beginnen wir aber mit der Herleitung der Rekursion bezüglich eines $r$-REC-Codes und führen dazu folgende Bezeichnung ein.

**4.2.19 Bezeichnungen**
*Gegeben sei ein beliebiges Fehlermuster* $\mathbf{f_n} = (f_0, \ldots, f_{n-1})$ *der Länge* $n$*, das* $i$ *einzelne Bitfehler enthält, wobei* $0 \leq i \leq n$ *ist. Dann bezeichnen wir die Wahrscheinlichkeit dafür, dass* $\mathbf{f_n}$ *gerade* $i$ *einzelne fehlerhafte Bits beinhaltet, während der Markov'sche Hintergrundprozeß* $(X_t)_{t \in \mathbb{N}}$ *abschließend den Zustand* $j \in \mathcal{S}$ *erreicht hat, mit*

$$P_j(i,n) := P\big((\sum_{k=0}^{n-1} E_k = i), (X_{t+n} = j)\big) \quad \text{für } n \geq 1 \text{ und } t \in \mathbb{N}_0.$$

**4.2.20 Satz**
*Gegeben sei ein beliebiger zyklischer* $(n,k,d)$*-Blockcode* $\mathcal{C}$*. Ist* $\mathcal{C}$ *ein* $r$*-REC-Code, so errechnet sich die Restfehlerwahrscheinlichkeit* $\mathcal{R}_{\mathcal{C}}(r,n)$ *von* $\mathcal{C}$ *aus der Gleichung*

$$\mathcal{R}_{\mathcal{C}}(r,n) = 1 - \sum_{i=0}^{r} \sum_{j=1}^{N} P_j(i,n).$$

**Beweis**
Da nach Voraussetzung der betrachtete zyklische $(n,k,d)$-Blockcode $\mathcal{C}$ ein $r$-REC-Code ist, korrigiert $\mathcal{C}$ alle Fehlermuster der Länge $n$, die bis zu $r$ einzelne Bitfehler enthalten. Zur Berechnung der Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(r,n)$ von $\mathcal{C}$ sind nach der obigen Bezeichnungsweise 4.2.19 die Wahrscheinlichkeiten $P_j(i,n)$ für alle $i$ mit $0 \leq i \leq r$ und alle $j \in \mathcal{S}$ zu ermitteln. Dann ergibt sich die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(r,n)$ von $\mathcal{C}$ folgendermaßen:

$$\mathcal{R}_{\mathcal{C}}(r,n) = 1 - \sum_{i=0}^{r} \sum_{j=1}^{N} P_j(i,n).$$

Dabei berechnet sich die Wahrscheinlichkeit $P_j(i,n)$ für jedes $i$ mit $0 \leq i \leq r$ und jedes $j \in \mathcal{S}$ rekursiv aus der Gleichung, die wir im folgenden herleiten:

Seien $i$ mit $0 \leq i \leq r$ und $j \in \mathcal{S}$ beliebig und bezeichne $P_j(i,l)$ im Sinne von Bezeichnung 4.2.19 die Wahrscheinlichkeit dafür, dass $i$ einzelne Bitfehler in einem Block der Länge $l$ mit $1 \leq l \leq n$ auftreten, während der Markov'sche Hintergrundprozeß zum Abschluß den

Zustand $j \in \mathcal{S}$ erreicht hat. Dann errechnet sich die Wahrscheinlichkeit $P_j(i, l+1)$ für einen Block der Länge $l+1$, der $i$ einzelne Bitfehler beinhaltet, rekursiv aus:

$$\begin{aligned} P_j(i, l+1) &= P\big((\sum_{k=0}^{l} E_k = i), (X_{t+l+1} = j)\big) \\ &= \sum_{k=1}^{N} P_k(i-1, l)\, q_{kj}\, e_j + P_k(i, l)\, q_{kj}\, (1 - e_j) \\ &\quad \text{für } 1 \leq l \leq n-1,\ t \in I\!N_0 \\ &\quad \text{sowie } 0 \leq i \leq r \text{ und } j \in \mathcal{S} \text{ beliebig} \end{aligned}$$

Hierbei setzen wir insbesondere speziell für $i > l$ die Wahrscheinlichkeit $P_j(i, l)$ zu Null, d.h. es gilt

$$P_j(i, l) := 0, \quad \text{falls } i > l \text{ ist.}$$

Sind sämtliche Wahrscheinlichkeiten $P_j(i, n)$ für jedes $i$ mit $0 \leq i \leq r$ und jedes $j \in \mathcal{S}$ unter Zuhilfenahme der obigen Rekursion ermittelt, so ergibt die Summe über alle $j \in \mathcal{S}$ der Wahrscheinlichkeiten $P_j(i, n)$ für ein beliebiges $i$ mit $0 \leq i \leq r$, die Gesamtwahrscheinlichkeit für alle Fehlermuster der Länge $n$, die genau $i$ Fehler mit $0 \leq i \leq r$ enthalten. Da $\mathcal{C}$ ein $r$-REC-Code ist, ergibt sich die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(r, n)$ von $\mathcal{C}$ daher zu

$$\mathcal{R}_{\mathcal{C}}(r, n) = 1 - \sum_{i=0}^{r} \sum_{j=1}^{N} P_j(i, n).$$

□

### 4.2.21 Bemerkung

Für die Wahrscheinlichkeit $P_j(i, n)$ aus Bezeichnung 4.2.19 betrachten wir den Spezialfall $i = 0$; d.h. die Wahrscheinlichkeit einer fehlerfreien Folge der Länge $n$ mit $n \geq 1$, während sich der zugrunde liegende Markov'sche Hintergrundprozeß abschließend im Zustand $j \in \mathcal{S}$ befindet. Im Zusammenhang mit dem Beweis des vorherigen Satzes gilt für die Wahrscheinlichkeit $P_j(0, n)$ insbesondere:

$$\begin{aligned} P_j(0, n) &\overset{4.2.19}{=} P\big((\sum_{k=0}^{n-1} E_k = 0), (X_{t+n} = j)\big) \\ &= P\big((E_0 = 0), \ldots (E_{n-1} = 0), (X_{t+n} = j)\big) \\ &\overset{4.2.20}{=} \sum_{k=1}^{N} P_k(0, n-1)\, q_{kj}\, (1 - e_j) \\ &\quad \text{für } n \geq 1,\ t \in I\!N_0 \text{ und } j \in \mathcal{S} \text{ beliebig.} \end{aligned}$$

Zur Initialisierung dieser Rekursion setzen wir für $n = 0$ mit Blick auf die Existenz der stationären Zustandswahrscheinlichkeiten $\pi_j$ für jedes $j \in \mathcal{S}$

$$P_j(0, 0) := P(X_t = j) = \pi_j \quad \text{für jedes } j \in \mathcal{S} \text{ und mit } t \in I\!N_0.$$

Bezeichnen wir nun mit $P_{kj}(0,n)$ insbesondere die Wahrscheinlichkeit einer fehlerfreien Folge der Länge $n$ mit $n \geq 1$, während der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ vom Zustand $k \in \mathcal{S}$ in den Zustand $j \in \mathcal{S}$ übergeht,

$$P_{kj}(0,n) \;:=\; P\big((E_0=0),\ldots,(E_{n-1}=0),(X_t=k),(X_{t+n}=j)\big) \quad \text{für } t \in \mathbb{N}_0,$$

dann resultiert die Wahrscheinlichkeit $P_j(0,n)$ mit $n \geq 1$ auch aus der folgenden Beziehung:

$$\begin{aligned} P_j(0,n) &\overset{4.2.19}{=} P\big((\sum_{k=0}^{n-1} E_k = 0),(X_{t+n}=j)\big) \\ &= P\big((E_0=0),\ldots(E_{n-1}=0),(X_{t+n}=j)\big) \\ &= \sum_{k=1}^{N} \pi_k\, P_{kj}(0,n) \\ &\qquad \text{für } n \geq 1,\ t \in \mathbb{N}_0 \text{ und } j \in \mathcal{S} \text{ beliebig.} \end{aligned}$$

Dabei berechnet sich die Wahrscheinlichkeit $P_{kj}(0,n)$ für alle $k,j \in \mathcal{S}$ rekursiv aus der Gleichung, die wir im folgenden herleiten:

Seien $k,j \in \mathcal{S}$ beliebig und bezeichne $P_{kj}(0,l)$ im obigen Sinne die Wahrscheinlichkeit einer fehlerfreien Folge der Länge $l$ mit $1 \leq l \leq n$, während der Markov'sche Hintergrundprozeß vom Zustand $k \in \mathcal{S}$ in den Zustand $j \in \mathcal{S}$ übergeht. Dann errechnet sich die Wahrscheinlichkeit $P_{kj}(0,l+1)$ für eine fehlerfreie Folge der Länge $l+1$ rekursiv aus:

$$\begin{aligned} P_{kj}(0,l+1) &= P\big((E_0=0),\ldots,(E_{l-1}=0),(E_l=0),(X_t=k),(X_{t+l+1}=j)\big) \\ &= \sum_{m=1}^{N} P_{km}(0,l)\, q_{mj}\,(1-e_j) \\ &\qquad \text{für } 1 \leq l \leq n-1,\ t \in \mathbb{N}_0 \text{ und } k,j \in \mathcal{S} \text{ beliebig.} \end{aligned}$$

Insbesondere für die Wahrscheinlichkeit eines korrekt übertragenen Bits bei einem gleichzeitigen Zustandswechsel des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ vom Zustand $k \in \mathcal{S}$ in den Zustand $j \in \mathcal{S}$ ergibt sich

$$P_{kj}(0,1) = q_{kj}\,(1-e_j).$$

Auf ganz ähnliche Art können wir ein passendes Rekursionsschema zur Berechnung der Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(b,n)$ eines $b$-BEC-Codes herleiten. Dazu modifizieren wir die obigen Gleichungen in angemessener Weise und vereinbaren dazu die folgende Bezeichnung.

### 4.2.22 Bezeichnungen

*Gegeben sei ein beliebiges Fehlermuster* $\mathbf{f_n} = (f_0,\ldots,f_{n-1})$ *der Länge* $n$*, das einen burst error der Länge* $\leq b$ *enthält. Dann bezeichnen wir die Wahrscheinlichkeit für eine fehlerfreie Folge der Länge* $l$ *mit* $0 \leq l \leq n-b$*, auf die ein beliebiges Muster der Länge* $b$*, beginnend mit einem Bitfehler, folgt, während der zugrunde liegende Markov'sche Hintergrundprozeß* $(X_t)_{t\in\mathbb{N}}$ *abschließend den Zustand* $j \in \mathcal{S}$ *erreicht hat, mit*

$$P_j^b(l,b) := P\big((E_0=0),\ldots,(E_{l-1}=0),(E_l=1),(X_{t+l+b}=j)\big) \quad \textit{mit } t \in \mathbb{N}_0.$$

**4.2.23 Satz**
*Gegeben sei ein beliebiger zyklischer $(n,k,d)$-Blockcode $\mathcal{C}$. Ist $\mathcal{C}$ ein $b$-BEC-Code, so errechnet sich die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(b,n)$ von $\mathcal{C}$ aus der Gleichung*

$$\begin{aligned}\mathcal{R}_{\mathcal{C}}(b,n) &= 1-\sum_{j=1}^{N} P_j(0,n)\\ &\quad -\sum_{l=0}^{n-b}\sum_{j=1}^{N}\sum_{k=1}^{N} P_j^b(l,b)P_{jk}(0,n-l-b)\\ &\quad -\sum_{l=n-b+1}^{n-1}\sum_{j=1}^{N} P_j^b(l,n-l)\end{aligned}$$

**Beweis**
Da nach Voraussetzung der betrachtete zyklische $(n,k,d)$-Blockcode $\mathcal{C}$ ein $b$-BEC-Code ist, korrigiert $\mathcal{C}$ alle Fehlermuster der Länge $n$, die einen *burst error* der Länge $\leq b$ beinhalten. Zur Berechnung der Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(b,n)$ von $\mathcal{C}$ sind mit Bezeichnung 4.2.19 speziell die Wahrscheinlichkeiten $P_j(0,n)$ für alle $j\in\mathcal{S}$ zu bestimmen. Ferner sind mit der obigen Bezeichnung 4.2.22 die Wahrscheinlichkeiten $P_j^b(l,b)$ für alle $l$ mit $0\leq l\leq n-b$ und jedes $j\in\mathcal{S}$ sowie die Wahrscheinlichkeiten $P_j^b(l,n-l)$ im Sinne von Bezeichnung 4.2.22 für jedes $l$ mit $n-b+1\leq l\leq n-1$ und alle $j\in\mathcal{S}$ zu ermitteln. Dann resultiert die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(b,n)$ von $\mathcal{C}$ schließlich in:

$$\begin{aligned}\mathcal{R}_{\mathcal{C}}(b,n) &= 1-\sum_{j=1}^{N} P_j(0,n)\\ &\quad -\sum_{l=0}^{n-b}\sum_{j=1}^{N}\sum_{k=1}^{N} P_j^b(l,b)P_{jk}(0,n-l-b)\\ &\quad -\sum_{l=n-b+1}^{n-1}\sum_{j=1}^{N} P_j^b(l,n-l)\end{aligned}$$

Dabei berechnet sich die Wahrscheinlichkeit $P_j^b(l,b)$ für jedes $l$ mit $1\leq l\leq n-b$ und jedes $j\in\mathcal{S}$ aus der Gleichung, die wir im folgenden herleiten:

$$\begin{aligned}P_j^b(l,b) &= P\big((E_0=0),\ldots,(E_{l-1}=0),(E_l=1),(X_{t+l+b}=j)\big)\\ &= \sum_{k=1}^{N} P_k(0,l)\,P_{kj}(b) \quad \text{für alle } 1\leq l\leq n-b,\ t\in\mathbb{N}_0 \text{ und } j\in\mathcal{S} \text{ beliebig.}\end{aligned}$$

Da wir in Bemerkung 4.2.21 zur Initialisierung die Wahrscheinlichkeit $P_j(0,0):=\pi_j$ für jedes $j\in\mathcal{S}$ gesetzt haben, erhalten wir für den Spezialfall $l=0$ insbesondere:

$$\begin{aligned}P_j(0,b) &= P\big((E_0=1),(X_{t+b}=j)\big)\\ &= \sum_{k=1}^{N} \pi_k\,P_{kj}(b) \quad \text{für } t\in\mathbb{N}_0 \text{ und } j\in\mathcal{S} \text{ beliebig.}\end{aligned}$$

Hierbei bezeichnet $P_{kj}(b)$ insbesondere die Wahrscheinlichkeit dafür, dass der zugrunde liegende Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ vom Zustand $i \in \mathcal{S}$ in den Zustand $j \in \mathcal{S}$ übergeht, wobei der Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ gleichzeitig ein beliebiges Muster der Länge $b$ generiert, das mit einem Bitfehler beginnt, also

$$P_{kj}(b) \;:=\; P\big((X_t = k), (E_0 = 1), (X_{t+b} = j)\big) \quad \text{für } t \in \mathbb{N}_0 \text{ und } k, j \in \mathcal{S} \text{ beliebig.}$$

Dabei berechnet sich die Wahrscheinlichkeit $P_{kj}(b)$ für alle $k, j \in \mathcal{S}$ rekursiv aus der folgenden Gleichung:

Seien $k, j \in \mathcal{S}$ beliebig und bezeichne $P_{kj}(l)$ im obigen Sinne die Wahrscheinlichkeit dafür, dass der zugrunde liegende Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ vom Zustand $i \in \mathcal{S}$ in den Zustand $j \in \mathcal{S}$ übergeht, wobei $(X_t)_{t\in\mathbb{N}}$ gleichzeitig ein beliebiges Muster der Länge $l$ mit $1 \le l \le b$ generiert, das mit einem Bitfehler beginnt, dann errechnet sich die Wahrscheinlichkeit $P_{kj}(l+1)$ rekursiv aus:

$$\begin{aligned} P_{kj}(l+1) &= P\big((X_t = k), (E_0 = 1), (X_{t+l+1} = j)\big) \\ &= \sum_{m=1}^{N} P_{km}(l)\, q_{mj} \quad \text{für } 1 \le l \le b,\ t \in \mathbb{N}_0 \\ &\qquad \text{sowie } k, j \in \mathcal{S} \text{ beliebig.} \end{aligned}$$

Insbesondere für die Wahrscheinlichkeit eines fehlerhaften Bits bei einem gleichzeitigen Zustandswechsel des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ vom Zustand $k \in \mathcal{S}$ in den Zustand $j \in \mathcal{S}$ ergibt sich

$$P_{kj}(1) = q_{kj}\, e_j.$$

Sind sämtliche Wahrscheinlichkeiten $P_j(0,n)$ für jedes $j \in \mathcal{S}$, alle Wahrscheinlichkeiten $P_j^b(l,b)$ für alle $l$ mit $0 \le l \le n-b$ und jedes $j \in \mathcal{S}$ sowie die Wahrscheinlichkeiten $P_j^b(l,n-l)$ für alle $l$ mit $n-b+1 \le l \le n-1$ und alle $j \in \mathcal{S}$ unter Zuhilfenahme der obigen Rekursion ermittelt, so ergibt sich die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}(b,n)$ von $\mathcal{C}$ daher zu

$$\begin{aligned} \mathcal{R}_{\mathcal{C}}(b,n) &= 1 - \sum_{j=1}^{N} P_j(0,n) \\ &\quad - \sum_{l=0}^{n-b} \sum_{j=1}^{N} \sum_{k=1}^{N} P_j^b(l,b) P_{jk}(0, n-l-b) \\ &\quad - \sum_{l=n-b+1}^{n-1} \sum_{j=1}^{N} P_j^b(l, n-l) \end{aligned}$$

□

### 4.2.24 Bemerkung

Bezüglich des Aufwandes einer vollständigen Auswertung dieser Rekursion sei an dieser Stelle kurz bemerkt. Für einen $r$-REC-Code ist der berechenbare Aufwand einer vollständigen

Auswertung des Rekursionsschemas beschränkt durch $\mathcal{O}(n\tilde{N}(N+r))$. Dagegen kann eine vollständige Auswertung für einen $b$-BEC-Code mit einem Gesamtaufwand in einer Größenordnung von $\mathcal{O}(n\tilde{N}N)$ durchgeführt werden. Dabei bezeichnet $\tilde{N}$ die Anzahl der von Null verschiedenen Elemente der Übergangsmatrix $q$. Aufgrund dieser äußerst geringen Komplexität und des niedrigen Aufwandes können wir eine vollständige Auswertung dieser Rekursion auch für Markov'sche Hintergrundprozesse mit einem äußerst großen Zustandsraum $\mathcal{S}$ durchführen. Insbesondere, wenn die Übergangsmatrix $\mathbf{Q}$ nur dünn besetzt ist, bleiben die Berechnungen auch für 1000 Zustände durchführbar.
Allerdings kann diese Näherung keine beliebig kleinen Werte annehmen, da die Berechnungen von Differenzen immer ungenauer werden, wenn das Ergebnis kleiner als die Zahldarstellung des Rechners ist. Unsere Implementierung arbeitet mit doppelter Genauigkeit und wird daher für Werte in einem Bereich von $10^{-12}$ ungeeignet.

Zusammenfassend können wir an dieser Stelle festhalten: Auf der Basis des in Abschnitt 3.4 vorgeschlagenen Kanalmodells ist es uns gelungen, rekursive Gleichungen zur Berechnung der Restfehlerwahrscheinlichkeit jeweils für REC-Codes und für BEC-Codes herzuleiten, vgl. hierzu die Sätze 4.2.20 und 4.2.23. Da wir die Klassifizierung in REC- und BEC-Codes in die Entwicklung der Rekursionen miteinbezogen haben, haben wir das Fehlerkorrekturverhalten und somit die verschiedenen algebraischen Merkmale unterschiedlicher Codes bei der Berechnung der Restfehlerwahrscheinlichkeit berücksichtigt. Folglich haben wir mit den in diesem Abschnitt präsentierten Rekursionen eine realistische Grundlage geschaffen, die einen aussagekräftigen Vergleich von REC- gegenüber BEC-Codes ermöglicht. Im folgenden Kapitel werden wir einen solchen Vergleich durchführen und das Ergebnis zur Auswahl eines optimalen Codes nutzen. Hierbei verstehen wir unter einem optimalen Code denjenigen Code, der unter den gegebenen Bedingungen, d.h. den gegebenen Charakteristika des Übertragungskanals, wie beispielsweise der Bitfehlerrate, am effizientesten in dem Sinne ist, als dass er die niedrigste Fehlerrate beim Empfänger erzielt.

# Kapitel 5

# Analyse und Bewertung verschiedener Codierverfahren

In den Kapiteln 3 und 4 haben wir einen geeigneten Ausgangspunkt für eine Gegenüberstellung von REC- und BEC-Codes geschaffen. So haben wir in Paragraph 3.4 ein endliches Markov Modell als Kanalmodell vorgeschlagen, mit dem sowohl *random* als auch *burst errors* und somit das Gedächtnis des betrachteten Kanals modelliert werden können. In Abschnitt 4.2 haben wir zyklische Fehler-korrigierende $(n, k, d)$-Blockcodes in REC- und BEC-Codes klassifiziert. Ferner haben wir diese Klassifizierung in die Berechnung der Restfehlerwahrscheinlichkeit für einen Fehler-korrigierenden Code bzw. in die Herleitung entsprechender Gleichungen miteinbezogen. Gegenstand dieses Kapitels ist nun die Anwendung der in den Kapiteln 3 und 4 diskutierten Konzepte und entwickelten Methoden.

Wir beginnen mit einer allgemeinen Analyse aller zyklischen Fehler-korrigierenden $(n, k, d)$-Blockcodes im Hinblick auf ihr konkretes Fehlerkorrekturverhalten. Dazu überprüfen wir in Abschnitt 5.1 zunächst, ob unter Vorgabe der Parameter $n$ und $k$ Codes mit unterschiedlichen Fehlerkorrektureigenschaften existieren, ob also REC- und BEC-Codes gleicher Coderate $k/n$ existieren. Erst dann ist eine gleichwertige Ausgangssituation für eine aussagekräftige Bewertung und Gegenüberstellung verschiedener, Fehler-korrigierender Codes sichergestellt. Auf den erzielten Resultaten aufbauend vergleichen wir anschließend in Abschnitt 5.2 Fehler-korrigierende Codes bezüglich ihrer Effizienz und Leistungsfähigkeit. Dieser Vergleich wird unter Vorgabe der Fehlerstatistik des betrachteten Übertragungskanals durchgeführt. Dabei spiegelt die Fehlerstatistik des Kanals wieder, in welchem Maße *random* und *burst errors* in dem betrachteten Übertragungskanal auftreten, anders ausgedrückt, wie stark das Gedächtnis des betrachteten Übertragungskanals ist. So bedingt die Stärke des Gedächtnisses ganz erheblich die Häufigkeit und Länge der aufgetretenen *burst errors*.

Wie schon in der Einleitung und den vorangegangenen Kapiteln 2, 3 und 4 skizziert, besitzen verschiedene Fehler-korrigierende Codes unterschiedliche Fehlerkorrektureigenschaften und ihr Einsatz zielt darauf ab, beim Empfänger eine möglichst niedrige Fehlerrate zu erzielen. Von daher ist es zweckmäßig die Auswahl eines Codierverfahrens zur Fehlerkorrektur an der Fehlerstatistik des betrachteten Übertragungskanals zu orientieren. Eine wesentliche Aufgabenstellung der Arbeit ist also, anhand der Charakteristika des betrachteten Übertragungskanals einen optimalen Fehler-korrigierenden Code zu bestimmen. Hierbei verstehen wir unter einem optimalen Fehler-korrigierenden Code ein Codierverfahren, welches eine möglichst große Anzahl von Informationsbits unter Verwendung einer möglichst

kleinen Anzahl von Kontrollbits möglichst sicher übermittelt, also eine möglichst große Anzahl von Übertragungsfehlern korrigiert. Das bedeutet, dass im Falle eines $r$-REC-Codes das Verhältnis $r/n-k$ und im Falle eines $b$-BEC-Codes das Verhältnis $b/n-k$ möglichst groß werden sollte.

In Abhängigkeit von den Kanalparametern, wie z.B. der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge, ist mit Blick auf die Zielsetzung der Arbeit das effizienteste Fehlerkorrekturverfahren zu bestimmen und zu entscheiden, ob die Verwendung eines REC-Codes lohnenswerter ist als die eines BEC-Codes. Für diese Entscheidung sind entsprechende Codes mittels eines geeigneten Kriteriums miteinander zu vergleichen. Als ein geeignetes Maß, um die Leistungsfähigkeit und Effizienz eines Codes zu beurteilen, verwenden wir die Restfehlerwahrscheinlichkeit im Sinne von Definition 4.2.3. Hierbei beruhen sämtliche unserer weiteren Untersuchungen in diesem Kapitel lediglich auf den Parametern $n$ und $k$, d.h. der Länge und der Dimension eines Codes, sowie auf der Fehlerstatistik des Übertragungskanals. Wir realisieren also für unsere Analysen kein konkretes Generatorpolynom. Aufgrund dieser Vorgehensweise beschränken wir unsere Untersuchungen nicht auf eine spezifische Familie zyklischer Fehler-korrigierender $(n,k,d)$-Blockcodes. Im Gegenteil neben sämtlichen zyklischen Fehler-korrigierenden Codierverfahren berücksichtigen wir auch im Sinne von Definition 4.1.39 verkürzte zyklische Codes.

An dieser Stelle sei darauf verwiesen, dass in der Literatur bereits erste Ansätze für derartige Analysen und Bewertungen unterschiedlicher Fehler-korrigierender Codierverfahren zu finden sind. So werden unter anderem in den Standardwerken von S. Lin und D. J. Costello, [26], sowie S. Roman, [47], verschiedene Codierverfahren mit ihren Fehlerkorrektureigenschaften in tabellarischer Form aufgelistet. Hierbei stammen die betrachteten Fehlerkorrigierenden Codes jedoch stets nur aus genau einer Familie zyklischer Blockcodes. Wir hingegen berücksichtigen bei unseren Untersuchungen alle zyklischen Codierverfahren aus sämtlichen Familien. Ferner bezieht sich die Analyse der Fehlerkorrektureigenschaften in der Standardliteratur lediglich auf Angaben hinsichtlich der Korrektur von *random errors*. Fähigkeiten bezüglich der Korrektur von Bündelfehlern werden in diesen Tabellen nicht angegeben. Im Gegensatz dazu können wir basierend auf den entwickelten Methoden auch Angaben zur Korrigierbarkeit von *burst errors* machen.

Darüber hinaus geschieht die in der Literatur präsentierte vergleichende Gegenüberstellung verschiedener Fehler-korrigierender Codes in der Regel unter Angabe konkreter Generatorpolynome. So gibt beispielsweise S. Roman in seinem Lehrbuch [47] eine systematische Liste irreduzibler Polynome über $GF(2)$ bzw. über $GF(3)$ bis zum Grad 11 bzw. bis zum Grad 7 mit deren Ordnung an. In einer weiteren Tabelle listet Roman zur Konstruktion primitiver BCH-Codes primitive Polynome über $GF(2)$ auf. In dem Standardwerk von S. Lin und D. J. Costello findet sich hierzu eine noch detailliertere Übersicht, vgl. [26]. Im Anhang des Lehrbuches werden Galois Felder $GF(2^m)$ mit $3 \leq m \leq 10$ in tabellarischer Form angegeben. Hierzu wird jedes Element aus $GF(2^m)$ als Potenz eines primitiven Elementes dargestellt. Das Minimalpolynom dieses primitiven Elementes erzeugt dann das Galois Feld. Als Vorbereitung für die Konstruktion primitiver BCH-Codes wird darüber hinaus eine Liste minimaler Polynome von Elementen aus $GF(2^m)$ mit $2 \leq m \leq 10$ aufgestellt. Abschließend werden sämtliche Generatorpolynome aller primitiven BCH-Codes der Länge $n = 2^m - 1$ mit $3 \leq m \leq 10$ in einer detaillierten Tabelle zusammengestellt. Ferner beinhaltet diese Übersicht über alle primitiven BCH-Codes der Länge $n = 2^m - 1$ mit $3 \leq m \leq 10$ neben der Angabe der Codelänge $n$, der Dimension $k$ und dem entsprechenden Generatorpolynom

Angaben zur Fehlerkorrigierbarkeit $t$ des durch das konkret angegebene Generatorpolynom realisierten BCH-Codes. Folglich werden in der Standardliteratur derartige vergleichende Gegenüberstellungen verschiedener Fehler-korrigierender Codes lediglich unter Realisierung konkreter Generatorpolynome durchgeführt. Hierbei stammen die analysierten Codes zudem stets aus der selben Familie zyklischer Codes, der Familie der BCH-Codes. Darüber hinaus wird nur die Fehlerkorrigierbarkeit von *random errors* untersucht und die Korrektur von Bündelfehlern völlig ignoriert. Aus diesen Gründen sind für die Auswahl eines optimalen Codierverfahrens anhand der Fehlerstatistik eines betrachteten Übertragungskanals die in der Literatur zu findenden Tabellen nicht aussagekräftig genug. Ferner finden bei den in der Literatur vorgestellten Analysen verkürzte zyklische Blockcodes im Sinne von Definition 4.1.39 bislang keine Berücksichtigung. Mit den von uns entwickelten Methoden werden wir in diesem Kapitel nun sehr viel allgemeinere Untersuchungen und Analysen durchführen.

## 5.1 Analyse der Fehlerkorrektureigenschaften zyklischer Codes

Für einen aussagekräftigen Vergleich der Leistung eines REC-Codes gegenüber der eines BEC-Codes in Abhängigkeit von den Eigenschaften des Übertragungskanals ist eine für beide Codetypen gleichwertige Ausgangssituation sicherzustellen. Daher weisen wir zu Beginn dieses Kapitels in Abschnitt 5.1.1 sowohl die Existenz von REC- als auch von BEC-Codes nach, die die gleiche Codelänge $n$ sowie die gleiche Dimension $k$ und damit die selbe Coderate $k/n$ besitzen. So ist bei gleicher Coderate $k/n$ die Anzahl $n-k$ der Kontrollstellen, die zur Korrektur von *random* bzw. zur Korrektur von *burst errors* zur Verfügung steht, identisch. Dies liefert eine gleichwertige Grundlage zur Beurteilung, wie effizient die $n-k$ Kontrollstellen jeweils von einem REC-Code bzw. von einem BEC-Code zur Fehlerkorrektur eingesetzt werden. Im Rahmen von Abschnitt 5.1.1 analysieren wir dazu das Fehlerkorrekturverhalten aller zyklischer Fehler-korrigierender Codes. In diesem Zusammenhang weisen wir nach, dass zyklische Blockcodes existieren, die in der Lage sind, sowohl eine gewisse Anzahl *random errors* als auch *burst errors* bis zu einer bestimmten Länge zu korrigieren. Einige interessante Beispiele geben wir dazu in einer Tabelle an. In Abschnitt 5.1.2 befassen wir uns dann eingehender mit BEC-Codes. Unter der Vorgabe der zu korrigierenden Burstlänge $b$ suchen wir nach der optimalen Parameterkonstellation, Codelänge $n$ und Dimension $k$, für $b$-BEC-Codes. Hierbei beschränkt sich die Suche nach einem optimalen BEC-Code nicht ausschließlich auf zyklische Blockcodes, auch verkürzte zyklische Codes werden bei dieser Analyse berücksichtigt. Als Ergebnis erhalten wir schließlich eine Tabelle, in der wir systematisch die optimalen Parameterkonstellationen, Codelänge $n$ und Dimension $k$, für $b$-BEC-Codes auflisten.

### 5.1.1 Existenz von REC- und BEC-Codes gleicher Codelänge

Vor dem Hintergrund, dass wir im folgenden Abschnitt 5.2 verschiedene Fehler-korrigierende Blockcodes hinsichtlich ihrer Effizienz und Leistungsfähigkeit vergleichen und sich dieser Vergleich allgemein auf die gesamte Klasse der zyklischen Blockcodes und nicht nur auf eine spezielle Familie bezieht, weisen wir an dieser Stelle nach, dass REC- und BEC-Codes existieren, die die gleiche Codelänge $n$ sowie die gleiche Dimension $k$ besitzen. Dazu analysieren

wir die gesamte Menge aller zyklischen Blockcodes unter Vorgabe der Parameter $n$ und $k$ im Hinblick auf ihr Fehlerkorrekturverhalten, ohne dabei jedoch konkrete Generatorpolynome zu realisieren; d.h. die Menge aller zyklischen $(n,k)$-Blockcodes $\mathcal{C}$ bzw. sämtliche Generatorpolynome, die zu den fest vorgegebenen Parametern $n$ und $k$ einen zyklischen Blockcode generieren, werden auf ihre Fehlerkorrektureigenschaften hin überprüft. Wir klassifizieren folglich die Menge der zyklischen $(n,k)$-Blockcodes $\mathcal{C}$ gemäß ihrer Fähigkeiten, eine gewisse Anzahl *random errors* bzw. *burst errors* bis zu einer bestimmten Länge zu korrigieren.

Nun wird das Fehlerkorrekturverhalten eines zyklischen $(n,k)$-Blockcodes $\mathcal{C}$ durch die Menge $\mathcal{F}_{\mathcal{C}}$ der korrigierbaren Fehlermuster, genauer deren Anzahl und Gestalt, bestimmt. Nach den Ausführungen im Zusammenhang mit der Einführung des Syndrombegriffes in Definition 4.1.57 und mit dem Beweis von Satz 4.2.6 ist die Menge $\mathcal{F}_{\mathcal{C}}$ der korrigierbaren Fehlermuster $\mathbf{f_n}$ dadurch gekennzeichnet, dass alle Fehlermuster $\mathbf{f_n} \in \mathcal{F}_{\mathcal{C}}$ auf unterschiedliche Syndrome $\mathbf{s} \in GF(2)^{n-k}$ führen. Besitzen zwei Fehlermuster $\mathbf{f_n}$ und $\tilde{\mathbf{f}}_\mathbf{n}$ dasselbe Syndrom $\mathbf{s} \in GF(2)^{n-k}$, mit

$$\mathbf{f_n}\,\mathbf{H}^T = \mathbf{s} = \tilde{\mathbf{f}}_\mathbf{n}\,\mathbf{H}^T,$$

so liegen diese nach Satz 4.1.59 in dem selben Coset und sind daher nicht eindeutig korrigierbar. Besitzt der betrachtete Code $\mathcal{C}$ die Fähigkeit, bis zu $r$ *random errors* zu korrigieren, so müssen sämtliche Fehlermuster $\mathbf{f_n}$, die bis zu $r$ zufällig verteilte Einzelfehler beinhalten, also vom Gewicht $\mathrm{wt}(\mathbf{f_n}) \leq r$ sind, auf unterschiedliche Syndrome führen. Analog gilt, falls der betrachtete Code $\mathcal{C}$, alle Bündelfehler bis zur Länge $b$ korrigiert, so müssen alle Fehlermuster $\mathbf{f_n}$, die einen Bündelfehler der Länge $\leq b$ darstellen, unterschiedliche Syndrome besitzen. Zur Bestimmung der Korrektureigenschaften eines beliebigen zyklischen $(n,k)$-Blockcodes $\mathcal{C}$ sind folglich einerseits alle Fehlermuster $\mathbf{f_n}$, die eine gewisse Anzahl *random errors* beinhalten, und andererseits sämtliche Fehlermuster $\mathbf{f_n}$, die Bündelfehler bis zu einer bestimmten Länge umfassen, zu untersuchen.

Hierzu haben wir einen effizienten Algorithmus entworfen und implementiert, mit dem auf die soeben beschriebene Weise das Fehlerkorrekturverhalten eines beliebigen zyklischen $(n,k)$-Blockcodes $\mathcal{C}$ ermittelt werden kann. Dabei sind die einzigen Eingabeparameter des Algorithmus die Codelänge $n$ sowie die Dimension $k$. Zunächst wird dann für jedes Generatorpolynom $g(z)$, das zu den eingegebenen Parametern $n$ und $k$ aufgestellt werden kann, analysiert, ob dieses tatsächlich einen zyklischen $(n,k)$-Blockcode generiert. Danach werden, wie oben beschrieben, die Fähigkeiten zur Korrektur einer gewissen Anzahl von *random errors* sowie die Möglichkeiten zur Korrektur von *burst errors* bis zu einer bestimmten Länge festgestellt. Dazu wird überprüft, ob die entsprechenden Syndrome alle voneinander verschieden sind oder ob zwei Syndrome miteinander übereinstimmen, also kollidieren. Als Ergebnis dieses Algorithmus erhalten wir bei Eingabe der Parameter $n$ und $k$ sämtliche Generatorpolynome $g(z)$, die einen zyklischen $(n,k)$-Blockcode $\mathcal{C}$ erzeugen, sowie Angaben zu dessen Fehlerkorrekturverhalten bezüglich der Korrigierbarkeit von $r$ *random errors* bzw. der Korrigierbarkeit von Bündelfehlern bis zur Länge $b$. Mit dieser Klassifizierung zyklischer Codes, die die gleiche Codelänge und die selbe Coderate besitzen, in REC- und BEC-Codes haben wir eine geeignete Ausgangsbasis für eine Leistungsbewertung in Abschnitt 5.2 geschaffen.

Bei dem Entwurf dieses Algorithmus haben wir insbesondere die Polynomdarstellung der Codewörter eines zyklischen $(n,k)$-Blockcodes, wie sie in Bezeichnung 4.1.42 bzw. Bemerkung 4.1.43 definiert worden ist, sowie die damit verbundenen algebraischen Eigenschaften ausgenutzt, mit denen wir uns angefangen von Definition 4.1.41 bis hin zu

Korollar 4.1.54 befaßt haben. Dabei spielt insbesondere die Erzeugung des gesamten zyklischen $(n, k)$-Blockcodes durch das zugehörige Generatorpolynom $g(z)$ sowie dessen Eigenschaften, die in Theorem 4.1.44 dargelegt worden sind, eine entscheidende Rolle. In den folgenden Ausführungen skizzieren wir zunächst in groben Zügen den Aufbau des Algorithmus. Anschließend präzisieren wir einige Schritte. Dazu geben wir die wesentlichen Prozeduren der Implementierung im Originalquellcode mit den zugehörigen Kommentaren und Erläuterungen an. Ferner begründen wir die Korrektheit des gesamten Algorithmus basierend auf mathematischen Methoden, indem wir die einzelnen Prozeduren anhand der entsprechenden algebraischen Ergebnisse aus Abschnitt 4.1 verifizieren.

**Algorithmus:**
Bestimmung der Fehlerkorrekturfähigkeiten eines zyklischen $(n, k)$-Blockcodes $\mathcal{C}$

---

**Eingabe:** Codelänge $n$, Dimension. $k$

**Initialisierung:** Setze das erste Generatorpolynom:
$g(z) = z^{n-k} + 1$;

Initialisiere die Parameter für die $r$-REC sowie die $b$-BEC:
$r = 0$;
$b = 1$;

---

**Schritt 1:** Falls sämtliche Generatorpolynome $g(z)$ vom Grad $grad(g) = n-k$ betrachtet worden sind, Programmende.

**Schritt 2:** Überprüfe, ob das betrachtete Generatorpolynom $g(z)$ einen zyklischen Blockcode der vorgegebenen Länge $n$ erzeugt.

**Schritt 3:** Falls $g(z)$ keinen zyklischen Blockcode der vorgegebenen Länge $n$ erzeugt, gehe zu Schritt 6.

**Schritt 4:** Bestimme die REC-Fähigkeit mittels der folgenden Schritte:

**(i):** Erhöhe $r = r + 1$;

**(ii):** Beginne mit dem Fehlermuster $f(z) = z^{r-1} + z^{r-2} + \ldots + z + 1$;

**(iii):** Falls alle $2^r$ Fehlermuster mit $r$ beliebig verteilten Einzelfehlern überprüft worden sind, gehe zu Schritt 4 (i).

**(iv):** Überprüfe, ob die Syndrome $s_j(z) = z^j f(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq n-1$, also für jeden zyklischen Shift des Fehlermusters $f(z)$, von jedem beliebigen Syndrom $\tilde{s}(z)$ verschieden sind.

**(v):** Falls in Schritt (iv) ein Syndrom $s_j(z)$ mit $0 \leq j \leq n-1$ mit einem beliebigen Syndrom $\tilde{s}(z)$ übereinstimmt, gebe zu dem betrachteten Generatorpolynom $g(z)$ den Wert des Parameters $r-1$ für die REC-Fähigkeit aus und gehe zu Schritt 5.

**(vi):** Generiere ein neues Fehlermuster $f(z)$ mit $r$ beliebig verteilten Einzelfehlern und gehe zu Schritt 4 (iii).

**Schritt 5:** Bestimme die BEC-Fähigkeit mittels der folgenden Schritte:

**(i):** Erhöhe $b = b + 1$;

**(ii):** Beginne mit dem Fehlermuster $f(z) = z^{b-1} + 1$;

**(iii):** Falls sämtliche Fehlermuster $f(z)$ vom Grad $grad(f) = b - 1$ betrachtet worden sind, gehe zu Schritt 5 (i).

**(iv):** Überprüfe, ob die Syndrome $s_j(z) = z^j f(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq n-1$, also für jeden zyklischen Shift des Fehlermusters $f(z)$, von jedem beliebigen Syndrom $\tilde{s}(z)$ verschieden sind.

**(v):** Falls in Schritt (iv) ein Syndrom $s_j(z)$ mit $0 \leq j \leq n-1$ mit einem beliebigen Syndrom $\tilde{s}(z)$ übereinstimmt, gebe zu dem betrachteten Generatorpolynom $g(z)$ den Wert des Parameters $b-1$ für die BEC-Fähigkeit aus und gehe zu Schritt 6.

**(vi):** Generiere ein neues Fehlermuster $f(z)$, das einen Bündelfehler der Länge $b$ beinhaltet, und gehe zu Schritt 5 (iii).

**Schritt 6:** Generiere ein neues Generatorpolynom $g(z)$.
Setze die Parameter für die REC und die BEC: $r = 0$; $b = 1$;
und gehe zu Schritt 1.

---

**Ausgabe:** Generatorpolynom $g(z)$, das zu den vorgegebenen Parametern $n$ und $k$ einen zyklischen $(n, k)$-Blockcode $\mathcal{C}$ erzeugt, sowie die Parameter $r - 1$ bzw. $b - 1$ für die REC- bzw. BEC-Fähigkeit.

Tabelle der Form: $n \mid k \mid g(z) \mid r-1 \mid b-1$

---

Nachdem wir den Algorithmus zur Bestimmung der Fehlerkorrekturfähigkeiten eines zyklischen $(n, k)$-Blockcodes $\mathcal{C}$ zu vorgegebenen Parametern $n$ und $k$ hinsichtlich der *random error correction (REC)* und der *burst error correction (BEC)* kurz skizziert haben, konkretisieren wir im folgenden einzelne Schritte des Algorithmus. Wir führen in diesem Zuge näher aus, wann bzw. warum die zugehörigen Abbruchbedingungen erfüllt sind. Hierzu geben wir zu den Schritten 2 und 6 sowie den Schritten 4 Teil (iii) bis (vi) und 5 Teil (iii) bis (vi) den Originalquellcode der entsprechenden Prozeduren an.
Zur Verifikation der einzelnen Prozeduren und damit des gesamten Algorithmus bedienen wir uns der algebraischen Eigenschaften zyklischer Blockcodes, mit denen wir uns in Abschnitt 4.1 auseinander gesetzt haben. Hierbei sind die Polynomdarstellung der Codewörter eines zyklischen Blockcodes sowie die damit verbundenen Ergebnisse aus der Galois-Theorie von entscheidender Bedeutung, vgl. dazu Definition 4.1.41, Bezeichnung 4.1.42 bis hin zu Bemerkung 4.1.43. So stellt nach Bemerkung 4.1.43 $z\,f(z)$ gerade die zyklische Verschiebung des zu $f(z)$ gehörenden Binärmusters um eine Position nach links dar. Dabei wird der Koeffizient $f_{n-1}$ von $f(z)$ fallen gelassen und sozusagen „nach links“ aus dem zugehörigen Binärmuster „hinausgeschoben“. Dies werden wir im folgenden stets mit den Worten „um eine Position zyklisch verschoben“ zusammenfassen. Daher entspricht $z\,f(z) \bmod g(z)$ gerade der binären Addition des um eine Position zyklisch verschobenen Binärmusters des Polynoms $f(z)$ und dem zum Generatorpolynom $g(z)$ gehörenden Generatormuster $\mathbf{g}$. Darüber hinaus nutzen wir zur Verifikation spezifische Eigenschaften des

Generatorpolynoms $g(z)$, die wir in Theorem 4.1.44 zusammengestellt haben. Wesentliche Grundlage für den Nachweis der Korrektheit der Prozeduren ist in diesem Zusammenhang das Ergebnis aus Korollar 4.1.54. So bilden die Monome $z^j$ für jedes $j$ mit $0 \leq j \leq n-1$ bezüglich der Polynommultiplikation bei Rechnung mod $g(z)$ eine zyklische Gruppe. Ferner bestimmt die Ordnung $ord(z)$ in dieser zyklischen Gruppe die Codelänge $n$ des von $g(z)$ erzeugten zyklischen Blockcodes. Somit ist die Berechnung von $z^j \bmod g(z)$ bzw. von $z^j f(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq n$ für die angesprochenen Prozeduren von grundlegender Bedeutung. Daher haben wir die Berechnung von $z\,f(z) \bmod g(z)$ in einer eigenen Prozedur zusammengefaßt, die wir nun als erstes angeben.
Aus programmiertechnischen Gründen sowie zur besseren Lesbarkeit vereinbaren wir für die nachstehenden Ausführungen die folgende Schreibweise. Im C-Quellcode verwenden wir für die Binärdarstellung des Generatorpolynoms $g(z)$ bzw. des Fehlermusters $f(z)$ die Bezeichnung `gpol` bzw. `fehl`. Der C-Quellcode der jeweiligen Prozedur wird in der Schriftart `Typewriter` angegeben, die entsprechenden Kommentare und Begründungen zur mathematischen Korrektheit der einzelnen Prozeduren sowie entsprechende Verweise auf die zugehörigen Ergebnisse in Kapitel 4 dagegen im normalen Schrifttyp.

Prozedur zur Berechnung von $z\,f(z) \bmod g(z)$:

---

```
void shiftadd()
```

Berechnung von $z\,f(z) \bmod g(z)$, wobei die zu $f(z)$ bzw. $g(z)$ gehörenden Binärmuster `fehl` bzw. `gpol` als globale Variablen aus anderen Prozeduren übergeben werden.

```
{
  int koef, mask;

  mask = 1;
  mask = mask << (n-k-1);
  koef = (fehl & mask) >> n-k-1;
```

In der Variablen `koef` wird der Koeffizient bezüglich $z^{n-k-1}$ des Polynoms $f(z)$ gespeichert.

```
  fehl = fehl << 1;
```

Die Berechnung von $z\,f(z)$ entspricht nach Bemerkung 4.1.43 einer zyklischen Verschiebung des Binärmusters `fehl` um eine Position nach links.

```
  if(koef == 1)
```

Falls in dem vorgegebenen Polynom $f(z)$ der Koeffizient bezüglich $z^{n-k-1}$ gleich 1 war,

```
  {
    fehl = (fehl ^ gpol);
```

wird durch binäre Addition $zf(z) \bmod g(z)$ berechnet.

```
  }
}
```

---

In Schritt 2 des Algorithmus wird abgefragt, ob das betrachtete Generatorpolynom $g(z)$ einen zyklischen $(n,k)$-Blockcode $\mathcal{C}$ der vorgegebenen Codelänge $n$ erzeugt. Dabei ist $g(z)$ entweder das zur Initialisierung festgelegte Polynom oder ein in Schritt 6 neu generiertes Polynom vom Grad $grad(g) = n-k$. Aufgabe der folgenden Prozedur ist es zu kontrollieren, ob der von $g(z)$ erzeugte zyklische Blockcode $\mathcal{C}$ die vorgegebene Codelänge $n$ besitzt.

Prozedur zur Überprüfung der Codelänge des von $g(z)$ erzeugten zyklischen Blockcodes:

---

```
int zyklustest()
```

Überprüfung der Codelänge des von $g(z)$ erzeugten zyklischen Blockcodes:

Nach Korollar 4.1.54 bilden die Monome $z^j$ für jedes $j$ mit $0 \leq j \leq n-1$ eine zyklische Gruppe bezüglich der Polynommultiplikation mod $g(z)$, so dass die Ordnung $ord(z)$ in dieser zyklischen Gruppe die Codelänge des von $g(z)$ erzeugten zyklischen Blockcodes angibt. Da nun wegen $grad(g) = n-k$ für jedes $j$ mit $0 \leq j \leq n-k-1$ gilt $z^j \bmod g(z) = z^j$, wird nun für jedes $j$ mit $n-k \leq j \leq n$ kontrolliert, ob $z^j \bmod g(z) = 1$ gilt.

Ergebnis von `zyklustest()` $= \begin{cases} 1 & ; \text{ falls Codelänge } = ord(z) = n \text{ ist.} \\ 0 & ; \text{ falls Codelänge } = ord(z) \neq n \text{ ist.} \end{cases}$

```
{
  int i, j, ok, test, mask, mask1;
  mask = 1;
  mask = mask << n-k;
  fehl = gpol;
  fehl = (fehl & ( ~ mask));
```

Damit gilt $f(z) = g(z) + z^{n-k} = z^{n-k} \bmod g(z)$.

```
  ok = 1;
```

Der Wert der Variablen signalisiert das Ergebnis der Überprüfung bezüglich der Codelänge des von $g(z)$ erzeugten zyklischen Blockcodes $\mathcal{C}$; d.h. also, ob $ord(z) = n$ gilt.

```
  for(j=1;j<=k;j++)
```

Für sämtliche Monome $z^j$ mit $n-k+1 \leq j \leq n$

```
  {
    shiftadd();
```

wird nun $z^j \bmod g(z)$ berechnet.

```
    test = 0;
    mask1 = 1;
    mask1 = mask1 << (n-k-1);
```

```
    for(i=n-k-1;i>=1;i--)
    {
      if(((fehl & mask1) >> i) == 1) test = 1;
```

Anschließend wird für jedes $j$ mit $n-k+1 \leq j \leq n$ überprüft, ob $z^j \bmod g(z) > 1$ ist.

```
      mask1 = mask1 >> 1;
    }
    if((fehl & (mask >> n-k)) == 0) test = 1;
```

Zum Abschluß wird kontrolliert, ob $z^j \bmod g(z) \neq 1$ ist.

```
    if((j < k)&&(test == 0)) ok = 0;
```

Da $ord(z) < n$ ist, ist damit auch die Codelänge ungleich $n$.

```
    if((j == k)&&(test == 1)) ok = 0;
```

Da $ord(z) > n$ ist, ist die Codelänge somit ebenfalls ungleich $n$.

```
  }
  return (ok);
}
```

---

Nachdem in Schritt 2 des Algorithmus sichergestellt ist, dass das betrachtete Generatorpolynom $g(z)$ einen zyklischen $(n,k)$-Blockcode $\mathcal{C}$ der vorgegebenen Länge $n$ generiert, werden in Schritt 4 bzw. in Schritt 5 des Algorithmus die Fähigkeiten zur Korrektur einer gewissen Anzahl *random errors* bzw. zur Korrektur von *burst errors* bis zu einer gewissen Länge bestimmt. In Teilschritt (i) wird der Parameter $r$ für die REC bzw. der Parameter $b$ für die BEC festgelegt. Anschließend ist zu kontrollieren, ob sämtliche Fehlermuster $f(z)$, die $r$ *random errors* bzw. Bündelfehler bis zur Länge $b$ beinhalten, auf unterschiedliche Syndrome führen. Nach Bemerkung 4.1.60 (i) errechnet sich für einen zyklischen Blockcode $\mathcal{C}$ das Syndrom $s(z)$ eines Fehlermusters $f(z)$ aus

$$s(z) = f(z) \bmod g(z).$$

Nach Teil (iii) gilt für das Syndrom einer zyklischen Verschiebung des Fehlermusters $f(z)$ insbesondere

$$\begin{aligned} s_j(z) &= z^j f(z) \bmod g(z) \\ &= z s_{j-1}(z) \bmod g(z) \\ &= z^j s(z) \bmod g(z) \quad \text{für jedes } j \text{ mit } 1 \leq j \leq n-1. \end{aligned}$$

Ferner nutzen wir zur Überprüfung der Syndrome wiederum das Ergebnis aus Korollar 4.1.54. So liefert Korollar 4.1.54, dass die Syndrome für jede zyklische Verschiebung eines vorgegebenen Fehlermusters $f(z)$, also

$$s_j(z) = z^j f(z) \bmod g(z) \quad \text{für jedes } j \text{ mit } 0 \leq j \leq n-1,$$

paarweise verschieden sind. Wird zur Initialisierung, wie in Schritt 4 bzw. in Schritt 5 Teil (ii) angegeben, das Fehlermuster $f(z) = z^{r-1} + \ldots + z + 1$ bzw. das Fehlermuster

$f(z) = z^{b-1} + 1$ festgelegt, so ist zu kontrollieren, ob das Syndrom $s_j(z)$ einer zyklischen Verschiebung des vorgegebenen Fehlermusters $f(z)$, d.h.

$$s_j(z) = z^j f(z) \bmod g(z) \quad \text{für ein } j \text{ mit } 1 \leq j \leq n-1$$

mit dem Syndrom $\tilde{s}(z)$ eines anderen Fehlermusters $\tilde{f}(z)$ übereinstimmt. Dabei ist das Fehlermuster $\tilde{f}(z)$ vom Grad $grad(\tilde{f}) \leq grad(f)$ und der Koeffizient bezüglich $z^0$ ist gerade gleich 1. Die weitere Gestalt von $\tilde{f}(z)$ richtet sich danach, ob Untersuchungen bezüglich der REC oder der BEC durchgeführt werden. Zur Überprüfung, ob das Syndrom $s_j(z)$ einer zyklischen Verschiebung des vorgegebenen Fehlermusters $f(z)$ mit dem Syndrom $\tilde{s}(z)$ eines anderen Fehlermusters $\tilde{f}(z)$ übereinstimmt, kontrollieren wir im Falle der REC, ob $grad(s_j(z)) > r-1$ ist, und im Falle der BEC, ob $grad(s_j(z)) > b-1$ für jedes $1 \leq j \leq n-1$ gilt. Ist dies der Fall, ist das Syndrom $s_j(z)$ für jedes $1 \leq j \leq n-1$ auch von jedem Syndrom $\tilde{s}(z)$ eines anderen Fehlermusters $\tilde{f}(z)$ verschieden. Auf diese Weise arbeiten wir für die Schritte 4 und 5 Teilschritt (iv) ab, was mit der folgenden Prozedur geschieht.

Prozedur zur Bestimmung der Korrekturfähigkeit:

---

`int polcheck()`

Überprüfung, ob die Syndrome sämtlicher Fehlermuster $f(z)$, die eine bestimmte Anzahl *random errors* bzw. *burst errors* bis zu einer gewissen Länge beinhalten, paarweise verschieden sind und nicht miteinander kollidieren.

Nach Korollar 4.1.54 bilden die Monome $z^j$ für jedes $j$ mit $0 \leq j \leq n-1$ eine zyklische Gruppe bezüglich der Polynommultiplikation mod $g(z)$. Die Syndrome für jeden zyklischen Shift eines vorgegebenen Fehlermusters $f(z)$, also $s_j(z) = z^j f(z) \bmod g(z)$ mit $1 \leq j \leq n-1$, sind daher paarweise verschieden. Nun ist zu prüfen, ob es ein Syndrom $s_j(z)$ mit $1 \leq j \leq n-1$ gibt, das mit dem Syndrom $\tilde{s}(z)$ eines anderen Fehlermusters $\tilde{f}(z)$ übereinstimmt. Dabei ist $\tilde{f}(z)$ vom Grad $grad(\tilde{f}) \leq grad(f)$ und beinhaltet entweder dieselbe Anzahl $r$ an *random errors* wie $f(z)$, wobei der Koeffizient bezüglich $z^0$ gleich 1 ist, oder einen Bündelfehler der selben Länge $b$ wie $f(z)$.

Ergebnis: `polcheck()` $= \begin{cases} 1 & ; \text{ falls } s_j(z) = \tilde{s}(z) \text{ für ein } 1 \leq j \leq n-1 \\ 0 & ; \text{ falls } s_j(z) \neq \tilde{s}(z) \text{ für alle } 1 \leq j \leq n-1 \end{cases}$

```
{
  int i, j, koll, mask;
  koll = 0;
```

Der Wert der Variablen signalisiert das Ergebnis der Überprüfung bezüglich der Syndrome $s_j(z)$, ob also für jedes $j$ mit $1 \leq j \leq n-1$ gilt $s_j(z) \neq \tilde{s}(z)$.

```
  while(koll == 0)
  {
```

```
for(j=1;(j<n)&&(koll == 0);j++)
```

Für jeden zyklischen Shift des vorgegebenen Fehlermusters $f(z)$

```
{
  shiftadd();
```

wird das Syndrom $s_j(z)$ mit $s_j(z) = z^j f(z) \bmod g(z)$ über die Beziehung $s_j(z) = z s_{j-1}(z) \bmod g(z) = z^j s(z) \bmod g(z)$ berechnet

```
  koll = 1;
  mask = 1;
  mask = mask << (n-k-1);
  for(i=n-k-1;i>=b;i--)
```

und überprüft,

```
  {
    if(((fehl & mask) >> i) == 1) koll = 0;
```

ob das Syndrom $s_j(z) \neq \tilde{s}(z)$ ist, wobei $\tilde{s}(z)$ das Syndrom eines Fehlermusters $\tilde{f}(z)$ darstellt, das vom Grad $grad(\tilde{f}) \leq grad(f)$ ist und entweder die selbe Anzahl $r$ an *random errors* beinhaltet, wobei der Koeffizient bezüglich $z^0$ gleich 1 ist, oder einen Bündelfehler der selben Länge $b$ umfaßt wie das betrachtete Fehlermuster $f(z)$.

```
    mask = mask >> 1;
  }
}
shiftadd();
```

Wegen $z^n \bmod g(z) = 1$ wird durch diesen $n$-ten Shift das ursprüngliche Fehlermuster $f(z)$ für weitere Prozeduren wiederhergestellt.

```
}
return (koll);
```

Der Wert der Variablen wird an den Hauptteil des Programms zurückgeliefert und bestimmt so den weiteren Programmdurchlauf.

```
}
```

---

Ergibt diese Überprüfung bezüglich aller Syndrome $s_j(z)$ mit $1 \leq j \leq n-1$ eine Kollision eines Syndroms $s_j(z)$ mit einem Syndrom $\tilde{s}(z)$ im obigen Sinne, so sind die beiden zugehörigen Fehlermuster nicht eindeutig korrigierbar. Nach Teilschritt (v) von Schritt 4 bzw. von Schritt 5 ist dann durch den Wert des Parameters $r-1$ bzw. den Wert des Parameters $b-1$ die Fähigkeit zur Korrektur einer bestimmten Anzahl *random errors* bzw. die Fähigkeit zur Korrektur von Bündelfehlern bis zu einer gewissen Länge für den betrachteten zyklischen $(n,k)$-Blockcode $\mathcal{C}$ festgelegt. In Schritt 6 wird anschließend eine neues Generatorpolynom $g(z)$ zu den vorgegebenen Parametern $n$ und $k$ erzeugt und der Algorithmus ab Schritt 1 erneut durchlaufen.

Existiert allerdings keine derartige Übereinstimmung eines Syndroms $s_j(z)$ mit einem Syndrom $\tilde{s}(z)$ im obigen Sinne, so ist ein neues Fehlermuster $f(z)$ zu generieren. Dabei

beinhaltet dieses neu zu generierende Fehlermuster $f(z)$ entweder $r$ *random errors* oder einen *burst error* der Länge $b$. Speziell zur Konstruktion eines neuen Fehlermusters $f(z)$, das einen Bündelfehler der Länge $b$ darstellt, ist die folgende Prozedur geschrieben worden.

Prozedur zur Generierung eines neuen Fehlermusters mit einem Bündelfehler der Länge $b$:

```
void syndromneu()
```

Konstruktion eines neuen Fehlermusters $f(z)$, das einen Bündelfehler der Länge $b$ beinhaltet.

```
{
  int l, mask;
  mask = 2;
  l = 1;
  while(((fehl & mask) >> l) == 1)
```

Falls der Koeffizient bezüglich $z^l$ in dem Fehlermuster $f(z)$ gleich 1 ist,

```
  {
    fehl = (fehl & ( ~ mask));
```

setze diesen Koeffizienten zu Null

```
    l++;
    mask = mask << 1;
  }
  fehl = (fehl | mask);
```

und setze abschließend den Koeffizienten bezüglich $z^l$ für den hochgezählten Parameter $l$ gleich 1.

```
  if(l == b) fehl = (fehl & ( ~ mask));
```

Falls diese Abfrage positiv ist, sind sämtliche Fehlermuster $f(z)$, die einen Bündelfehler der Länge b beinhalten, und vom Grad $grad(f) = b - 1$ sind, generiert worden.

Indem hier als neues Fehlermuster $f(z) = 1$ gesetzt wird, wird die entsprechende Voraussetzung geschaffen, um in Schritt 5 (iii) des Algorithmus die Abbruchbedingung zu erfüllen.

```
}
```

Nachdem wir nun die wesentlichen Prozeduren zur Bestimmung des Fehlerkorrekturverhaltens eines vorgegebenen zyklischen $(n, k)$-Blockcodes $\mathcal{C}$ skizziert haben, betrachten wir zum Abschluß Schritt 6 des Algorithmus genauer. In Schritt 6 wird die Konstruktion eines neuen Generatorpolynoms $g(z)$ vom Grad $grad(g) = n - k$ gefordert, was die Aufgabe der folgenden Prozedur ist. Hierbei wird deutlich, dass ein neues Generatorpolynom $g(z)$ zu vorgegebenen Parametern $n$ und $k$ nach dem gleichen Prinzip erzeugt wird, wie wir in der vorangegangenen Prozedur ein neues Fehlermuster $f(z)$ mit einem Bündelfehler der Länge $b$ generiert haben.

Prozedur zur Generierung eines neuen Generatorpolynoms $g(z)$:

```
void polynomneu()
```

Konstruktion eines neuen Generatorpolynoms $g(z)$ zu den gegebenen Parametern $n$ und $k$.

```
{
  int l, mask;
  mask = 2;
  l = 1;
  while(((gpol & mask) >> l) == 1)
```

Falls der Koeffizient bezüglich $z^l$ in dem Fehlermuster $f(z)$ gleich 1 ist,

```
  {
    gpol = (gpol & ( ~ mask));
```

setze diesen Koeffizienten zu Null

```
    l++;
    mask = mask << 1;
  }
  gpol = (gpol | mask);
```

und setze abschließend den Koeffizienten bezüglich $z^l$ für den hochgezählten Parameter $l$ gleich 1.

```
  if(l == n-k) gpol = (gpol & ( ~ mask));
```

Falls diese Abfrage positiv ist, sind sämtliche Generatorpolynome $g(z)$ vom Grad $grad(g) = n-k$ zu den gegebenen Parametern $n$ und $k$ generiert worden.

Indem hier $g(z) = 1$ als neues Generatorpolynom gesetzt wird, wird die Voraussetzung geschaffen, um in Schritt 1 des Algorithmus die Abbruchbedingung zu erfüllen und das Programm zu beenden.

```
}
```

Mit dem in den vorangehenden Ausführungen skizzierten Algorithmus haben wir ein Instrument entwickelt, mit dem sich unter Vorgabe der Parameter $n$ und $k$ die Menge aller zyklischen $(n, k)$-Blockcodes in REC- und BEC-Codes klassifizieren läßt. Wir können auf diese Weise also die Existenz REC- und BEC-Codes der gleichen Codelänge $n$ und der selben Dimension $k$ nachweisen. Ferner lassen sich auf einfache Weise mittels des entwickelten Algorithmus die entsprechenden Parameter $r$ und $b$ für die REC bzw. BEC berechnen. Einige interessante Ergebnisse aus verschiedenen Programmdurchläufen sind hierzu in Tabelle 5.1 zusammengestellt. Anhand von Tabelle 5.1 können wir schließlich die folgenden Szenarien zum Vergleich des Fehlerkorrekturverhalten verschiedener zyklischer $(n, k)$-Blockcodes durchspielen. So untersuchen wir zunächst, inwiefern sich die Fehlerkorrekturfähigkeit von Codes unterscheidet, die gleich viele Kontrollstellen aber unterschiedliche Codelänge besitzen. Anschließend analysieren wir, ob Codes der selben Codelänge und -rate von unterschiedlichem Fehlerkorrekturverhalten sind.

Hierzu erfassen wir in Tabelle 5.1 die folgenden Parameter, die Codelänge $n$, die Anzahl $n-k$ der Kontrollstellen und die daraus resultierende Coderate $k/n$. Beispielhaft listen wir ferner zu den vorgegebenen Parametern $n$ und $k$ einige Generatorpolynome $g(z)$ auf, die einen zyklischen $(n,k)$-Blockcode $\mathcal{C}$ mit interessanten Fehlerkorrektureigenschaften erzeugen. In den letzten beiden Spalten von Tabelle 5.1 werden schließlich die Resultate präsentiert, die mittels des vorgestellten Algorithmus mit Blick auf die Fehlerkorrekturfähigkeiten für die angegebenen zyklischen $(n,k)$-Blockcodes $\mathcal{C}$ erzielt worden sind; d.h. hier wird angegeben, wie viele *random errors* maximal vom betrachteten Code korrigiert werden können bzw. bis zu welcher Länge *burst errors* korrigiert werden können.

| Code-länge $n$ | Kontroll-stellen $n-k$ | Code-rate $k/n$ | Generatorpolynom $g(z)$ | $r$-REC | $b$-BEC |
|---|---|---|---|---|---|
| 63 | 12 | 0.81 | 1 001 110 010 101 | 2 | 4 |
| 63 | 12 | 0.81 | 1 000 010 000 111 | 1 | 5 |
| 63 | 13 | 0.79 | 10 010 011 111 111 | 1 | 6 |
| 63 | 15 | 0.75 | 1 000 101 100 011 111 | 2 | 7 |
| 63 | 17 | 0.73 | 111 011 000 101 011 101 | 2 | 8 |
| 63 | 18 | 0.71 | 1 111 001 101 000 001 111 | 3 | 5 |
| 63 | 19 | 0.70 | 10 000 000 010 011 101 011 | 2 | 9 |
| 105 | 12 | 0.89 | 1 011 011 010 001 | 1 | 5 |
| 105 | 18 | 0.84 | 1 000 110 000 110 111 001 | 1 | 8 |
| 105 | 19 | 0.83 | 10 100 010 001 110 000 111 | 2 | 8 |
| 105 | 21 | 0.80 | 1 100 110 000 111 111 011 111 | 1 | 10 |

Tabelle 5.1: Fehlerkorrekturverhalten zyklischer $(n,k)$-Blockcodes hinsichtlich der Korrektur von *random* und *burst errors*

Für den Parameter Codelänge $n$ haben wir in verschiedenen Programmdurchläufen den Wert 63 oder 105 vorgegeben. Motiviert wird die Auswahl dieser Werte durch den Einsatz zyklischer Blockcodes genau dieser Codelängen in der Praxis. So hat es beispielsweise Versuchssysteme zur Spezifikation des GSM-Mobilfunkstandards gegeben, die einen RS-Code der Länge $n = 63$ zur Fehlerkorrektur eingesetzt haben, vgl. [7]. Ferner werden RS-Codes der Länge $n = 63$ zur Fehlerkontrolle häufig auch in Datenspeicherungssystemen verwendet, wie z.B. in dem *Photodigital Mass Storage System* von IBM, das auch unter dem Namen *Digital Cypress* bekannt ist, vgl. dazu [26]. Der Einsatz zyklischer Blockcodes dieser Codelängen ist unter anderem auch in den *Technical Reports*, die schließlich zur Festlegung der ITU- und ETSI-Standards geführt haben, für verschiedenste Anwendungen aus den Bereichen der Satelliten- oder Mobilfunkübertragung analysiert und empfohlen worden. Neben diesen anwendungsbezogenen Gesichtspunkten sind für die Auswahl dieser Werte auch die zahlentheoretischen und algebraischen Zusammenhänge maßgeblich gewesen. So existieren Parameterkonstellationen (Codelänge $n$, Dimension $k$) zu denen nur wenige oder gar keine zyklischen $(n,k)$-Blockcodes generiert werden können. Aufgrund zahlentheoretischer Zusammenhänge existieren zu den Codelängen 63 und 105 zahlreiche Konstellationen für den Parameter $k$, so dass eine Vielzahl zyklischer Blockcodes zu den vorgegebenen Parametern erzeugt werden können. Insbesondere existieren für die Codelänge $n = 63$ bei geeigneten Werten für den Parameter $k$ entsprechende BCH- und RS-Codes. So variiert der Parameter

$k$ in den in Tabelle 5.1 aufgelisteten Beispielen für die Codelänge $n = 63$ zwischen den Werten $k = 51, 50, 48, 46, 45$ und 44. Für die Codelänge $n = 105$ wählen wir für die Dimension $k$ Werte zwischen $k = 93, 87, 86$ und $k = 84$. Dementsprechend variiert die Anzahl $n-k$ der Kontrollstellen in beiden Fällen jeweils zwischen den Werten $n - k = 12, 13, 15, 17, 18, 19$ bis hin zu $n - k = 21$, wie in der zweiten Spalte von Tabelle 5.1 aufgeführt.

Generell können wir Tabelle 5.1 entnehmen, dass zu den vorgegebenen Parametern $n$ und $k$ jeweils zyklische Fehler-korrigierende Codes existieren, die sowohl eine bestimmte Anzahl *random errors* als auch sämtliche *burst errors* bis zu einer bestimmten Länge korrigieren. Hierbei zeigt sich deutlich, dass einige der angegebenen Beispiele eher zur REC und andere wiederum eher zur BEC eingesetzt werden sollten.
Betrachten wir nun Codes mit der gleichen Anzahl Kontrollstellen aber unterschiedlicher Codelänge, so können wir den Einfluß der größeren Codelänge auf das Fehlerkorrekturverhalten analysieren. In Tabelle 5.1 haben wir hierzu einige interessante Beispiele für zyklische Blockcodes der Länge 105 und 63 zusammengestellt. Stellen wir den zweiten $(63, 51)$-Blockcode und den $(105, 93)$-Blockcode aus Tabelle 5.1 einander gegenüber, so besitzen beide Codes die selbe Anzahl Kontrollstellen $n - k = 12$. Eine Analyse der Korrektureigenschaften des $(105, 93)$-Blockcodes ergibt, dass dieser genau wie der $(63, 51)$-Blockcode sämtliche Bündelfehler bis zur Länge 5 korrigiert. Der Parameter $r$ für die REC beträgt für beide Codes nur $r = 1$. Aufgrund der größeren Codelänge überträgt der $(105, 93)$-Blockcode bei gleicher Zuverlässigkeit jedoch mehr Information als der $(63, 51)$-Blockcode.
Ein Vergleich des $(63, 51)$-Blockcodes mit der Coderate $k/n = 0.81$ mit einem längeren zyklischen Code von annähernd der selben Coderate, wie z.B. dem $(105, 84)$-Blockcode mit der Coderate 0.8, liefert, dass der $(105, 84)$-Blockcode wesentlich längere *burst errors* korrigiert. So beträgt die korrigierbare Bündelfehlerlänge für diesen Code 10 im Gegensatz zu der des $(63, 51)$-Blockcodes mit $b = 5$. Aufgrund der größeren Codelänge besitzt der $(105, 84)$-Blockcode schließlich 9 Kontrollstellen mehr, was sich folgerichtig entsprechend im Fehlerkorrekturverhalten wiederspiegelt.

Abschließend betrachten wir nun Codes gleicher Codelänge und -rate. In Tabelle 5.1 ist hierzu ein passendes Beispiel angegeben. So existieren laut Tabelle 5.1 zwei zyklische Blockcodes der Länge 63 und der Dimension 51 mit unterschiedlichen Fehlerkorrektureigenschaften. Der erste in Tabelle 5.1 angegebene $(63, 51)$-Blockcode korrigiert einerseits sämtliche Fehlermuster mit genau 2 *random errors* und andererseits sämtliche Bündelfehler bis zur Länge 4. Demgegenüber zeichnet sich der andere $(63, 51)$-Blockcode durch die Korrektur aller Bündelfehler bis zur Länge 5 aus. Da der Parameter $r$ für die REC für diesen Code jedoch lediglich mit $r = 1$ bestimmt worden ist, besitzt dieser keine herausragenden Fähigkeiten zur REC. Von daher ist der zweite $(63, 51)$-Blockcode in erster Linie für die Korrektur von Bündelfehlern geeignet und kann als 5-BEC-Code charakterisiert werden. Im Gegensatz dazu ist der andere $(63, 51)$-Blockcode für die Korrektur von *random errors* besonders geeignet. So kann dieser als 2-REC-Code charakterisiert werden. Daher haben wir für die vorgegebenen Parameter $n = 63$ und $k = 51$ die Existenz zyklischer $(n, k)$-Blockcodes der gleichen Codelänge und der selben Dimension nachgewiesen, die sich aber durch völlig unterschiedliches Fehlerkorrekturverhalten auszeichnen.
Neben den bereits betrachteten $(63, 51)$-Blockcodes repräsentieren auch der $(63, 45)$- sowie der $(63, 44)$-Blockcode aus Tabelle 5.1 ein weiteres Beispiel für Codes von annähernd gleicher Coderate $k/n = 0.70$ bzw. $k/n = 0.71$ und der selben Codelänge $n = 63$, aber mit unterschiedlichem Fehlerkorrekturverhalten. Für den $(63, 45)$-Blockcode resultiert der Parameter $r$ für die REC in $r = 3$ und der Parameter $b$ für die BEC in $b = 5$. Im Gegensatz dazu kor-

rigiert der $(63, 44)$-Blockcode lediglich bis zu 2 *random errors* aber sämtliche Bündelfehler bis zur Länge 9.
Ferner geht aus Tabelle 5.1 hervor, dass bereits eine zusätzliche Kontrollstelle das Fehlerkorrekturverhalten entscheidend beeinflussen kann. So korrigiert der angegebene $(63, 50)$-Blockcode alle Bündelfehler bis zur Länge 6 im Gegensatz zu dem ersten $(63, 51)$-Blockcode, der nur alle *burst errors* bis zur Länge 4 korrigiert.

Aus diesen wenigen Beispielen in Tabelle 5.1 geht bereits deutlich hervor, dass zu vorgegebenen Parametern $n$ und $k$ zyklische Blockcodes von gleicher Codelänge und annähernd gleicher Coderate $k/n$ aber mit unterschiedlichem Fehlerkorrekturverhalten existieren. Mittels des entworfenen Algorithmus können wir basierend auf den berechneten Parametern $r$ und $b$ für die REC bzw. die BEC eine entsprechende Klassifizierung der Menge aller zyklischen Blockcodes in REC- und BEC-Codes vornehmen. Tabelle 5.1 illustriert insbesondere, dass REC- und BEC-Codes der Länge 63 mit Dimension 51 existieren. Auf diese Weise haben wir eine realistische und gleichwertige Ausgangssituation für die Beurteilung der Leistungsfähigkeit und einen fundierten Vergleich der Effizienz REC- und BEC-Codes in Abhängigkeit von den Kanalparametern in Abschnitt 5.2 geschaffen.

### 5.1.2 Optimale Parameterkonstellationen

In diesem Abschnitt befassen wir uns eingehender mit der BEC, denn bereits in den vorherigen Kapiteln haben wir angemerkt, dass Übertragungsfehler in realen Kanälen häufig in zeitlicher Nähe und damit in Form von *burst errors* auftreten. In diesem Zusammenhang sei noch einmal auf die Arbeit von C. Kröll verwiesen, [24], in der dies mit der Angabe eines Auszuges aus einer Fehlermessung bestätigt wird. Da bei der Übertragung über Mobilfunk- oder Satellitenübertragungsstrecken Bündelfehler relativ häufig auftreten, im Falle eines Glasfaserkabels zwar wesentlich seltener, hier aber dennoch zur Unbrauchbarkeit und Nicht-Ausführbarkeit von Programmen führen können, ist die Korrektur von *burst errors* von entscheidender Bedeutung. Bevor wir in Abschnitt 5.2 REC- und BEC-Codes einander gegenüber stellen, analysieren wir BEC-Codes und deren Parameterkonstellationen genauer. So befassen wir uns im Rahmen dieser detaillierten Analyse mit der Frage nach einem optimalen zyklischen $(n, k)$-Blockcode für die BEC unter Vorgabe der zu korrigierenden Bündelfehlerlänge $b$. Gesucht ist also die optimale Parameterkonstellation (Codelänge $n$, Dimension $k$) eines Fehler-korrigierenden Codes für die BEC aller *burst errors* bis zu einer zuvor festgelegten Bündelfehlerlänge $b$. Hierbei verstehen wir unter einem optimalen zyklischen $(n, k)$-Blockcode $\mathcal{C}$ einen Blockcode, der möglichst viel Information übertragen und gleichzeitig eine möglichst große Anzahl Übertragungsfehler korrigieren kann. Um auf der einen Seite möglichst viel Information übertragen zu können, ist die Coderate $k/n$ und damit die Dimension $k$ des Blockcodes zu maximieren. Um auf der anderen Seite aber auch eine möglichst große Anzahl Übertragungsfehler korrigieren zu können, ist die Zahl der Kontrollstellen, also der Parameter $n-k$, möglichst groß zu wählen. Dies widerspricht der zuvor geforderten Maximierung des Parameters für die Dimension $k$. Aus diesem Grund versuchen wir in diesem Abschnitt, mit einer detaillierten Analyse einen Kompromiß für diese beiden sich widersprechenden Forderungen zu erarbeiten. Dazu gehen wir den folgenden beiden Aspekten nach:

- Wie groß muß die minimale Anzahl $n - k$ der Kontrollstellen sein, um sämtliche Bündelfehler einer vorgegebenen Länge $b$ korrigieren zu können?

- Wie groß ist die maximal mögliche Codelänge $n$ für einen BEC-Code bei Vorgabe der Parameter für die zu korrigierende Bündelfehlerlänge $b$ und die Anzahl $n-k$ der Kontrollstellen?

Schließlich besteht unser Interesse darin, die Coderate $k/n$ für den zur Fehlerkorrektur eingesetzten $b$-BEC-Code zu maximieren, um auf diese Weise so viel Information wie möglich übertragen zu können.

Eine Antwort bezüglich der ersten Frage liefert die sogenannte Reiger-Schranke. So haben wir im Beweis zu Korollar 4.2.13 gezeigt, dass für einen zyklischen $(n,k)$-Blockcode $\mathcal{C}$ gilt:

$$n-k \geq 2b$$

Allerdings läßt Korollar 4.2.13 offen, ob tatsächlich zyklische $(n,k)$-Blockcodes existieren, für die die Reiger-Schranke eine scharfe Schranke ist; d.h. ob zyklische $(n,k)$-Blockcodes $\mathcal{C}$ existieren, für die $n-k=2b$ gilt und die damit nach Definition 4.2.14 als optimale $(n,k)$-Blockcodes charakterisiert werden können.

Zur Beantwortung der zweiten Fragestellung nach der maximal möglichen Codelänge $n$ für einen $b$-BEC-Code bei Vorgabe der Parameter $b$ und $n-k$ stellen wir im folgenden einen weiteren Algorithmus vor, der zu diesem Zweck entwickelt und implementiert worden ist. Dieser bezieht in die Analyse neben zyklischen auch verkürzte zyklische Blockcodes mit ein. In der Literatur werden im Gegensatz zu unserer Untersuchung, die wir gleich im Anschluß präsentieren, zu meist nur zyklische jedoch keine verkürzten zyklischen Blockcodes auf ihre BEC-Fähigkeiten hin geprüft, vgl. [7, 26, 47, 27]. So werden die BEC-Fähigkeiten verkürzter zyklischer Blockcodes in den Standardwerken der Literatur stets unter Realisierung spezieller Generatorpolynome bestimmt, nicht aber allgemein, unabhängig von einer speziellen Familie zyklischer Blockcodes. Bekanntermaßen werden bei der Datenspeicherung auf Compact Disks zwei verkürzte zyklische RS-Code, $(32,28,5)$-RS-Code sowie $(28,24,5)$-RS-Code, eingesetzt, die aus dem $(255,251,5)$-RS-Code konstruiert werden. Diese Untersuchungen sind allerdings mit Hilfe von Versuchssystemen und nicht über allgemeine theoretische Analysen ermittelt worden. Demzufolge existiert es in der Literatur keine allgemeine Übersicht zu den BEC-Fähigkeiten sowohl zyklischer als auch verkürzter zyklischer Blockcodes.

Entgegen den Ansätzen in der Literatur dehnen wir mittels des folgenden Algorithmus unabhängig von einer speziellen Familie zyklischer Blockcodes unsere Untersuchungen auch auf verkürzte zyklische Codes zu den gegebenen Parametern aus. Zur Bestimmung der maximal möglichen Codelänge $n$ wird in Schritt 5 des Algorithmus in der Variablen $shifts_{neu}$ für alle möglichen Bündelfehler der Länge $\leq b$ jeweils die minimale Anzahl der zyklischen Verschiebungen nach links bestimmt, für die es zu einer Kollision zweier Syndrome kommt. Schließlich wird mit der Variablen $shifts_{alt}$ das Maximum sämtlicher dieser Minima bezüglich aller Generatorpolynome $g(z)$ ermittelt. Dieses Maximum gibt dann gerade die maximal mögliche Codelänge $n$ an. Bei der Entwicklung des Algorithmus und zum Nachweis von dessen Korrektheit nutzen wir einmal mehr die Polynomdarstellung der Codewörter eines zyklischen Blockcodes nach Bezeichnung 4.1.42 bzw. Bemerkung 4.1.43, die Charakteristika eines Generatorpolynoms $g(z)$ gemäß Theorem 4.1.44 sowie die damit verbundenen algebraischen Eigenschaften nach Korollar 4.1.54. Im folgenden skizzieren wir kurz den Algorithmus und verweisen zur Präzisierung und Verifikation einzelner Schritte auf die im vorangehenden Abschnitt 5.1.1 vorgestellten Prozeduren.

**Algorithmus:**
Bestimmung der maximal möglichen Codelänge $n$ für einen BEC-Code unter Vorgabe der Parameter $b$ und $n-k$

**Initialisierung:** Setze den Parameter für die Anzahl der Kontrollstellen:
$n-k=6$;

**Schritt 1:** Falls $n-k \leq 25$ ist, setze den Parameter für die zu korrigierende Bündelfehlerlänge $b=2$.
Sonst, Programmende.

**Schritt 2:** Falls $b > 14$ ist, erhöhe $n-k := n-k+1$ und gehe zu Schritt 1.

**Schritt 3:** Setze den Parameter $n_{alt}$ für die maximal mögliche Codelänge $n$ eines $b$-BEC-Codes sowie das erste Generatorpolynom:
$n_{alt} = n-k+1$;
$g(z) = z^{n-k}+1$;

**Schritt 4:** Falls alle Generatorpolynome $g(z)$ vom Grad $grad(g) = n-k$ betrachtet worden sind, gebe den Wert des Parameters $n_{alt}$ als maximal mögliche Codelänge $n$ für einen $b$-BEC-Code aus.
Erhöhe $b := b+1$ und gehe zu Schritt 2.

**Schritt 5:** Bestimme für das betrachtete Generatorpolynom $g(z)$ die kleinste Zahl $shifts_{alt}$, so dass die Syndrome $s_j(z) = z^j f(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq shifts_{alt}-1$ und jedes Fehlermuster $f(z)$ das einen Bündelfehler der Länge $b$ enhält, von jedem beliebigen Syndrom $\tilde{s}(z)$ verschieden sind, mittels der folgenden Schritte:

**(i):** Setze den Parameter $shifts_{alt} = 4095$;

**(ii):** Beginne mit dem Fehlermuster $f(z) = z^{b-1}+1$;

**(iii):** Falls sämtliche Fehlermuster $f(z)$ vom Grad $grad(f) = b-1$ betrachtet worden sind, gehe zu Schritt 6.

**(iv):** Bestimme die kleinste Zahl $shifts_{neu}$ mit $shifts_{neu} \leq shifts_{alt}$, so dass das Syndrom des $shifts_{neu}$-mal zyklisch verschobenen Fehlermusters $f(z)$ mit einem beliebigen Syndrom $\tilde{s}(z)$ übereinstimmt; d.h. dass die Syndrome $s_j(z) = z^j f(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq shifts_{neu}-1$ von jedem beliebigen Syndrom $\tilde{s}(z)$ verschieden sind.

**(v):** Falls $shifts_{neu} \leq n-k$ ist, gehe zu Schritt 9.

**(vi):** Falls $shifts_{neu} < shifts_{alt}$ ist, setze $shifts_{alt} = shifts_{neu}$;

**(vii):** Generiere ein neues Fehlermuster $f(z)$, das einen Bündelfehler der Länge $b$ beinhaltet, und gehe zu Schritt 5 (iii).

**Schritt 6:** Bestimme für das betrachtete Generatorpolynom $g(z)$ die kleinste Zahl $shifts_{alt}$, so dass die Syndrome $s_j(z) = z^j \overline{f}(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq shifts_{alt}-1$ und jedes Fehlermuster $\overline{f}(z)$ das einen Bündelfehler der Länge $< b$ enhält, von jedem beliebigen Syndrom $\tilde{s}(z)$ verschieden sind, mittels der folgenden Schritte:

**(i):** Beginne mit dem Fehlermuster $\overline{f}(z) = 1$;

**(ii):** Falls alle Fehlermuster $\overline{f}(z)$ vom Grad $grad(\overline{f}) < b-1$ betrachtet worden sind, gehe zu Schritt 7.

**(iii):** Bestimme die kleinste Zahl $shifts_{neu}$ mit $shifts_{neu} \leq shifts_{alt}$; d.h. dass das Syndrom des $shifts_{neu}$-mal zyklisch verschobenen Fehlermusters $\overline{f}(z)$ mit einem beliebigen Syndrom $\tilde{s}(z)$ übereinstimmt, so dass also die Syndrome $s_j(z) = z^j\overline{f}(z) \bmod g(z)$ für jedes $j$ mit $0 \leq j \leq shifts_{neu}-1$ von jedem beliebigen Syndrom $\tilde{s}(z)$ verschieden sind.

**(iv):** Falls $shifts_{neu} \leq n-k$ ist, gehe zu Schritt 9.

**(v):** Falls $shifts_{neu} < shifts_{alt}$ ist, setze $shifts_{alt} = shifts_{neu}$;

**(vi):** Generiere ein neues Fehlermuster $\overline{f}(z)$, das einen Bündelfehler der Länge $< b$ beinhaltet, und gehe zu Schritt 6 (ii).

**Schritt 7:** Setze $n_{neu} = shifts_{alt}$;

**Schritt 8:** Falls $n_{neu} > n_{alt}$ ist, dann setze $n_{alt} = n_{neu}$;

**Schritt 9:** Generiere ein neues Generatorpolynom $g(z)$ und gehe zu Schritt 4.

---

**Ausgabe:** Maximal mögliche Codelänge, die zu den vorgegebenen Parametern $b$ und $n-k$ erreicht werden kann, wobei sowohl zyklische als auch verkürzte zyklische Blockcodes in Betracht gezogen werden können

Tabelle der Form:

| $n-k$ | $b$ | $n_{alt}$ |
|---|---|---|

---

Um einzelne Schritte dieses Algorithmus, wie beispielsweise die Schritte 5, 6 und 9, zu konkretisieren, deren Korrektheit nachzuweisen und zu zeigen, wann bzw. warum die zugehörigen Abbruchbedingungen erfüllt sind, geben wir kurz die passenden Prozeduren des vorangegangenen Abschnittes 5.1.1 mit entsprechenden Kommentaren und Erläuterungen an.

Zunächst legen wir zur Initialisierung des Algorithmus die Anzahl der Kontrollstellen mit $n-k = 6$ fest, denn aufgrund der Reiger-Schranke benötigen wir wegen $3 = b \leq \frac{n-k}{2}$ zur Korrektur aller *burst errors* der Länge 3 mindestens 6 Kontrollstellen, vgl. dazu Korollar 4.2.13. Ferner beschränken wir aus rechentechnischen Gründen die Anzahl der Kontrollstellen nach oben. Als obere Schranke legen wir in Schritt 1 $n-k \leq 22$ fest. Somit bleibt die Rechenzeit in einem akzeptablen Rahmen und darüber hinaus werden in der Praxis Codes mit wesentlich mehr als 22 Kontrollstellen ohnehin nur selten eingesetzt.

In Schritt 5 (iv) bzw. in Schritt 6 (iii) wird die Bestimmung des Parameters $shifts_{neu}$ mittels der Prozedur `polcheck()` aus dem vorherigen Abschnitt 5.1.1 durchgeführt. Im Unterschied zum vorherigen Abschnitt ändern wir diese Prozedur lediglich dahingehend, dass wir den Wert des Parameters $shifts_{alt}$ an die Prozedur `polcheck()` übergeben. Somit ist diese Prozedur nicht mehr, wie im vorangehenden Abschnitt 5.1.1, als `int polcheck()` sondern in diesem Fall als `int polcheck(int` $shifts_{alt}$`)` deklariert. Dies bedeutet, dass im Gegensatz zum vorherigen Abschnitt nicht alle Syndrome

$$s_j(z) = z^j f(z) \bmod g(z) \quad \text{für jedes } j \text{ mit } 1 \leq j \leq n-1$$

bezüglich einer Kollision mit einem beliebigen Syndrom $\tilde{s}(z)$ überprüft werden, sondern alle Syndrome

$$s_j(z) = z^j f(z) \bmod g(z) \quad \text{für jedes } j \text{ mit } 1 \leq j \leq shifts_{alt} - 1$$

analysiert werden. Indem wir aus dem Hauptteil des Algorithmus den Parameter $shifts_{alt}$ an die Prozedur `polcheck(`$shifts_{alt}$`)` übergeben, bestimmen wir die kleinste Zahl $shifts_{neu}$, für die $shifts_{neu} \leq shifts_{alt}$ gilt, so dass die Syndrome aller zyklischen Verschiebungen um maximal $shifts_{neu} - 1$ Positionen nach links eines vorgegebenen Fehlermusters $f(z)$ keine Kollision mit einem beliebigen Syndrom $\tilde{s}(z)$ hervorrufen. Nach Korollar 4.1.54 liefert die Zahl $shifts_{neu}$ dann gerade eine obere Schranke für die Länge des von $g(z)$ erzeugten BEC-Codes bzw. genau die Länge des von $g(z)$ erzeugten BEC-Codes, wenn die Syndrome

$$s_j(z) = z^j f(z) \bmod g(z) \quad \text{für jedes } j \text{ mit } 1 \leq j \leq shifts_{neu} - 1$$

für jedes Fehlermuster $f(z)$, das einen Bündelfehler der Länge $\leq b$ enthält, keine Kollision verursachen.

In Schritt 5 (vii) bzw. in Schritt 6 (vi) des Algorithmus erfolgt die Generierung eines neuen Fehlermusters $f(z)$, das einen Bündelfehler der Länge $b$ bzw. $< b$ umfaßt, mittels der bereits im vorherigen Abschnitt vorgestellten Prozedur `syndromneu()`. Ebenso läßt sich, wie in Schritt 9 des Algorithmus skizziert, ein neues Generatorpolynom $g(z)$ mit Hilfe der in Abschnitt 5.1.1 präsentierten Prozedur `polynomneu()` konstruieren.

Die Verifikation einzelner Schritte des Algorithmus läßt sich damit in erster Linie ebenfalls auf das Ergebnis aus Korollar 4.1.54 zurückführen. An dieser Stelle sei ferner darauf hingewiesen, dass sich unsere Untersuchungen mit dem obigen Algorithmus nicht nur auf die Menge der zyklischen Blockcodes sondern auch der verkürzten zyklischen Blockcodes bezieht. Indem die betrachteten Generatorpolynome $g(z)$, die den fest vorgegebenen Grad $grad(g) = n - k$ besitzen, nicht mittels der Prozedur `zyklustest()` daraufhin kontrolliert werden, ob sie einen zyklischen Blockcode erzeugen, werden bei der Analyse auch verkürzte zyklische Blockcodes in Betracht gezogen. Demzufolge wird die gesamte Menge aller zyklischen sowie aller verkürzten zyklischen Blockcodes unter Vorgabe der zu korrigierenden Bündelfehlerlänge $b$ und der Anzahl $n - k$ der Kontrollstellen in Bezug auf ihr Bündelfehlerkorrekturverhalten untersucht. Wir erhalten auf diese Weise eine allgemeine Übersicht über die BEC-Fähigkeiten zyklischer sowie verkürzter zyklischer Blockcodes und eine systematische Auflistung der maximal möglichen Codelänge $n$ für einen $b$-BEC-Code bei Vorgabe der Parameter $b$ und $n-k$. Als Ergebnis unseres Algorithmus präsentieren wir eine solche Übersicht in der folgenden Tabelle 5.2, mit der wir die zu Beginn dieses Abschnittes aufgeworfenen Frage nach der optimalen Parameterkonstellation für einen BEC-Code nun beantworten können.

In Tabelle 5.2 präsentieren wir zu vorgegebenen Parametern $b$ und $n - k$ für die zu korrigierende Bündelfehlerlänge und die zur Verfügung stehende Anzahl Kontrollstellen die maximal mögliche Codelänge $n$ eines BEC-Codes, wobei dieser nicht unbedingt zyklisch ist, sondern auch ein verkürzter zyklischer Blockcode sein kann. Dabei signalisiert ein - Querstrich in einem Feld von Tabelle 5.2, dass zu den vorgegebenen Werten für $b$ und $n - k$ weder ein zyklischer noch ein verkürzter zyklischer Blockcode existiert, der die festgelegte Bündelfehlerlänge $b$ korrigieren kann. Ferner bedeutet $*$ , dass zu diesen Werten $b$ und $n - k$ ein BEC-Code existiert, der sämtliche *burst errors* bis zur Länge $b$ korrigiert

| Bündelfehlerlänge $b$ / Kontrollstellen $n-k$ | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|
| 6 | 15 | - | - | - | - | - | - | - | - |
| 7 | 27 | - | - | - | - | - | - | - | - |
| 8 | 63 | 19 | - | - | - | - | - | - | - |
| 9 | 121 | 35 | - | - | - | - | - | - | - |
| 10 | 255 | 82 | 24 | - | - | - | - | - | - |
| 11 | 487 | 164 | 47 | - | - | - | - | - | - |
| 12 | 1023 | 511 | 127 | 31 | - | - | - | - | - |
| 13 | 1999 | 1023 | 290 | 64 | - | - | - | - | - |
| 14 | $*$ | 1647 | 765 | 165 | 34 | - | - | - | - |
| 15 | $*$ | $*$ | 1365 | 363 | 99 | - | - | - | - |
| 16 | $*$ | $*$ | 2404 | 819 | 197 | 50 | - | - | - |
| 17 | $*$ | $*$ | $*$ | 1785 | 448 | 91 | - | - | - |
| 18 | $*$ | $*$ | $*$ | 3765 | 963 | 255 | 53 | - | - |
| 19 | $*$ | $*$ | $*$ | $*$ | 2690 | 520 | 121 | - | - |
| 20 | $*$ | $*$ | $*$ | $*$ | $*$ | 1105 | 289 | 52 | - |
| 21 | $*$ | $*$ | $*$ | $*$ | $*$ | 3031 | 554 | 118 | - |
| 22 | $*$ | $*$ | $*$ | $*$ | $*$ | $*$ | 1365 | 309 | 65 |

Tabelle 5.2: BEC-Korrekturfähigkeit und maximal mögliche Codelänge $n$

und dessen Länge in die Zehntausende gehen kann. Aus rechentechnischen Gründen und aufgrund der verwendeten Datentypen haben wir in Schritt 5 (i) des Algorithmus eine Abbruchbedingung in Form einer obere Grenze für die zu berechnende maximal mögliche Codelänge $n$ eingeführt. Indem wir den Parameter $shifts_{alt} = 4095$ setzen und in Schritt 5 (iv) an die Prozedur `polcheck(`$shifts_{alt}$`)` übergeben, stellen wir sicher, dass die Prozedur `polcheck(`$shifts_{alt}$`)` maximal 4095-mal durchlaufen wird. So begrenzen wir zwar die maximal mögliche Codelänge $n$ für einen $b$-BEC-Code nach oben durch den Wert 4095, aber gleichzeitig stellen wir sicher, dass die Rechenzeit in einem akzeptablen Rahmen bleibt. Ohne diese Einschränkung würde das Programm die Prozedur `polcheck(`$shifts_{alt}$`)` so lange durchlaufen, bis es im Sinne von Schritt 5 (iv) zu einer Kollision zweier Syndrome kommt. Für kleine Bündelfehlerlängen $b$ und beispielsweise mehr als 14 Kontrollstellen werden hierbei sehr viel größere Werte als 4095 erwartet. Die maximal mögliche Codelänge $n$ kann z.B. für die Parameter $b \leq 4$ und $n - k \geq 14$ unter Umständen in die Zehntausende gehen kann. Dies würde zu inakzeptablen Rechenzeiten führen. Motiviert wird die Wahl des Wertes $shifts_{alt} = 4095$ dadurch, dass erstens tatsächlich Blockcodes der Länge 4095 in der Praxis eingesetzt werden und zweitens zu dieser Codelänge BCH-Codes sowie RS-Codes existieren, für die es sehr effiziente Decodieralgorithmen gibt. Darüber hinaus ist drittens der immer größer werdende Decodieraufwand zu berücksichtigen, der mit einer wachsenden Codelänge $n$ einhergeht. Je länger der verwendete Fehler-korrigierende Blockcode ist, desto höher ist auch der Decodieraufwand. Von daher werden Fehler-korrigierende Blockcodes, deren Codelänge in die Zehntausende geht, aufgrund des immensen Decodieraufwandes in der Praxis nicht zur Fehlerkorrektur eingesetzt. Somit können wir unsere Berechnungen ohne Einschränkung bei einer Länge von 4095 abbrechen.

Ganz allgemein können wir in Tabelle 5.2 die beiden folgenden Entwicklungen ablesen. Erstens wird deutlich, dass bei einer fest vorgegebenen Anzahl Kontrollstellen für eine wachsende zu korrigierende Burstfehlerlänge $b$ die maximal mögliche Codelänge $n$ abnimmt. Beispielsweise geht die maximal mögliche Codelänge bei 18 Kontrollstellen für einen 6-BEC-Code von $n = 3765$ auf $n = 53$ für einen 9-BEC-Code zurück. Um bei der selben Anzahl Kontrollstellen längere Bündelfehler korrigieren zu können, muß sich das Verhältnis $\frac{k}{n-k}$ der Dimension zur Anzahl der Kontrollstellen und damit die Dimension $k$ verringern. Folglich geht die maximal mögliche Codelänge $n$ bei einer fest vorgegebenen Anzahl Kontrollstellen für eine wachsende zu korrigierende Bündelfehlerlänge $b$ zurück.
Zweitens können wir beobachten, dass die maximal mögliche Codelänge $n$ wächst, falls die zu korrigierende Bündelfehlerlänge $b$ fest gehalten und die Anzahl der zur Verfügung stehenden Kontrollstellen gesteigert wird. So können wir Tabelle 5.2 entnehmen, dass ein 9-BEC-Code mit 18 Kontrollstellen und der maximal möglichen Codelänge $n = 53$ existiert. Demgegenüber gibt es aber auch ein 9-BEC-Code mit 19 Kontrollstellen und der maximal möglichen Codelänge $n = 121$. Da 18 Kontrollstellen bereits zur Korrektur der fest vorgegebenen Bündelfehlerlänge $b = 9$ ausreichen, wird die zusätzliche Kontrollstelle nicht mehr zur Fehlerkorrektur benötigt. Daher kann mit jeder zusätzlichen Kontrollstelle das Verhältnis $\frac{k}{n-k}$ der Dimension zur Anzahl der Kontrollstellen und somit auch die Dimension $k$ des $b$-BEC-Codes gesteigert werden. Als Folge dessen wächst die maximal mögliche Codelänge $n$ für eine feste zu korrigierende Bündelfehlerlänge $b$ bei wachsender Anzahl Kontrollstellen.

Tabelle 5.2 liefert ferner Lösungen für die zu Beginn aufgeworfenen Fragestellungen. So wird mit dieser allgemeinen Übersicht und systematischen Auflistung der maximal möglichen Codelänge $n$ für einen BEC-Code zu den vorgegebenen Parametern $b$ und $n - k$ die zweite Frage nach einer optimalen Parameterkonstellation beantwortet. Bezüglich der ersten Fragestellung, ob die Reiger-Schranke eine scharfe Schranke ist, ob also ein Blockcode existiert, für den $n - k = 2b$ gilt und der damit im Sinne von Definition 4.2.14 ein optimaler Code ist, zeigt die systematische Auflistung in Tabelle 5.2, dass für eine gerade Anzahl Kontrollstellen die Reiger-Schranke als scharfe Schranke bestätigt wird. So gibt es laut Tabelle beispielsweise einen Blockcode mit 12 Kontrollstellen, der sämtliche Bündelfehler bis zur Länge 6 korrigiert und eine maximale Codelänge von $n = 31$ besitzt. Ein weiteres Beispiel dafür, dass die Reiger-Schranke für eine gerade Anzahl Kontrollstellen eine scharfe Schranke ist, stellt ein Blockcode mit 18 Kontrollstellen und einer maximalen Codelänge von $n = 53$ dar, der alle Bündelfehler bis zur Länge 9 korrigiert. Auf der anderen Seite zeigt Tabelle 5.2 auch, dass für $n - k$ ungerade nur Blockcodes existieren, die sämtliche Bündelfehler bis zur Länge $\frac{n-k-1}{2}$ korrigieren können. Insgesamt bestätigt Tabelle 5.2 für eine gerade Anzahl Kontrollstellen die Existenz von Blockcodes, für die die Reiger-Schranke eine scharfe Schranke ist. Dies ist für bis zu 22 Kontrollstellen in Tabelle 5.2 nachgewiesen und fest gehalten. Demzufolge existieren für eine geradzahlige Anzahl Kontrollstellen Blockcodes, die wir im Sinne von Definition 4.2.14 als optimale Blockcodes charakterisieren können. Für eine ungerade Anzahl Kontrollstellen existieren solche Blockcodes jedoch nicht existieren.

Zusammenfassend halten wir fest. Tabelle 5.2 gibt unter Vorgabe der Parameter für die zu korrigierende Bündelfehlerlänge $b$ und die zur Verfügung stehende Anzahl Kontrollstellen eine allgemeine und systematische Übersicht zu den Korrekturfähigkeiten zyklischer und verkürzter zyklischer Blockcodes in Bezug auf die BEC an. Dabei beruht entgegen den Ansätzen in der Literatur die Erstellung dieser tabellarischen Übersicht nicht auf der Rea-

lisierung spezieller Generatorpolynome und berücksichtigt neben der Menge der zyklischen Blockcodes auch verkürzte zyklische Blockcodes. Das Ergebnis des entwickelten Algorithmus in Tabelle 5.2 zeigt, dass die Betrachtung einer speziellen Familie von Blockcodes nicht aussagekräftig genug für die Auswahl eines optimalen Fehler-korrigierenden Blockcodes ist, denn in Tabelle 5.2 sind die angegebenen maximal möglichen Codelängen $n$ oftmals Codelängen verkürzter zyklischer Blockcodes. Tabelle 5.2 stellt in diesem Sinne ein geeignetes Werkzeug für die Auswahl eines optimalen Fehler-korrigierenden Blockcodes zur Korrektur aller Bündelfehler einer vorgegebenen Länge $b$ dar.

## 5.2 Vergleich der Effizienz von REC- gegenüber BEC-Codes

Nachdem wir im vorherigen Abschnitt 5.1 die Existenz zyklischer Blockcodes der gleichen Codelänge $n$ und der selben Dimension $k$, die aber unterschiedliche Fehlerkorrektureigenschaften besitzen, nachgewiesen haben, verfügen wir damit nun über eine fundierte Grundlage, um die Leistungsfähigkeit und Effizienz verschiedener Fehler-korrigierender Blockcodes miteinander vergleichen zu können. Von daher befassen wir uns in diesem Abschnitt nun mit der in der Einleitung skizzierten Zielsetzung der Arbeit, anhand der Fehlerstatistik eines betrachteten Übertragungskanals die effektivste Form der Fehlerkorrektur, d.h. einen optimalen Fehler-korrigierenden Blockcode, zu bestimmen. Indem wir REC- und BEC-Codes der gleichen Codelänge $n$ sowie der selben Dimension $k$ in Abhängigkeit von den Übertragungseigenschaften des verwendeten Übertragungskanals im Hinblick auf ihre Leistung und Effizienz bewerten, stellen wir REC-Codes und BEC-Codes einander gegenüber. Auf der Basis eines derartigen Vergleichs können wir somit anhand der Fehlerstatistik des betrachteten Kanals einen optimalen Fehler-korrigierenden Blockcode zur Fehlerkorrektur auswählen.

In diesem Abschnitt führen wir einen derartigen Vergleich von REC- und BEC-Codes in Abhängigkeit von den Übertragungseigenschaften des Kanals beispielhaft für verschiedene zyklische Blockcodes der Codelänge $n = 63$ bzw. $n = 105$ durch. Die Begründung für die Auswahl dieser Werte für den Parameter der Codelänge $n$ geht auf die Ausführungen in Abschnitt 5.1.1 zurück. Dort haben wir die Werte $n = 63$ bzw. $n = 105$ für den Parameter der Codelänge $n$ durch den Einsatz zyklischer Blockcodes genau dieser Codelängen $n = 63, 105$ in der Praxis motiviert. Darüber hinaus sind in Abschnitt 5.1.1 auch zahlentheoretische und algebraische Zusammenhänge als entscheidende Gründe für die Auswahl der Werte $n = 63, 105$ angeführt worden. So existiert für diese Codelängen eine Vielzahl an Parameterkonstellationen $(n, k)$, zu denen eine große Anzahl zyklischer Blockcodes der Länge $n$ und der Dimension $k$ konstruiert werden kann. In Abschnitt 5.1.1 haben wir ferner für die Werte $n = 63$ bzw. $n = 105$ anhand von Tabelle 5.1 die Existenz von REC- und BEC-Codes der gleichen Codelänge $n = 63$ bzw. $n = 105$ und der selben Dimension $k$ bzw. mit einer annähernd gleich großen Coderate $k/n$ nachgewiesen. Zyklische Blockcodes mit den Parametern $n$ und $k$ aus Tabelle 5.1 bilden daher eine geeignete Grundlage für einen Vergleich von REC- und BEC-Codes im Hinblick auf deren Leistungsfähigkeit und Effizienz. Von daher stützen wir unsere Untersuchungen in diesem Abschnitt auf die in Tabelle 5.1 vorgegebenen bzw. ermittelten Werte.

Um REC- und BEC-Codes im Hinblick auf ihre Leistungsfähigkeit vergleichen zu können und auf diese Art schließlich einen optimalen Blockcode zur Fehlerkorrektur auswählen zu können, ist die Effizienz von REC- gegenüber BEC-Codes in Abhängigkeit von den Übertragungseigenschaften des Kanals zu beurteilen. In Abschnitt 4.2 haben wir als Maß für die Effizienz und als Bewertungskriterium für den Erfolg eines Fehler-korrigierenden Blockcodes $\mathcal{C}$ die Restfehlerwahrscheinlichkeit $\mathcal{R}_{\mathcal{C}}$ von $\mathcal{C}$ eingeführt. In Abschnitt 4.2.2 haben wir diesbezüglich zwischen der Restfehlerwahrscheinlichkeit eines REC-Codes und derjenigen eines BEC-Codes differenziert. So haben wir mit Satz 4.2.20 eine Formel zur Berechnung der Restfehlerwahrscheinlichkeit eines $r$-REC-Codes und mit Satz 4.2.23 eine entsprechende Formel zur Bestimmung der Restfehlerwahrscheinlichkeit eines $b$-BEC-Codes entwickelt. Beide Ausdrücke, sowohl die Formel für die Restfehlerwahrscheinlichkeit eines $r$-REC-Codes als auch diejenige für einen $b$-BEC-Code, werden als Funktionen der Übergangswahrscheinlichkeiten $q_{ij}$ mit $i, j \in \mathcal{S}$ und der Fehlerwahrscheinlichkeiten $e_j$ mit $j \in \mathcal{S}$ dargestellt. Damit stehen beide Formeln in keinem Abhängigkeitsverhältnis zu den Kanalparametern des Übertragungskanals. Ein Ziel dieses Abschnittes ist es, die Restfehlerwahrscheinlichkeit sowohl für REC-Codes als auch für BEC-Codes als Funktion der Kanalparameter zu definieren und sie somit in Abhängigkeit von den Kanaleigenschaften darstellen und auswerten zu können. Indem wir die Übergangswahrscheinlichkeiten $q_{ij}$ für $i, j \in \mathcal{S}$ und die Fehlerwahrscheinlichkeiten $e_j$ für $j \in \mathcal{S}$ als Funktionen der Kanalparameter darstellen, können wir auch die Restfehlerwahrscheinlichkeit für einen REC-Code bzw. für einen BEC-Code als eine Funktion der Kanalparameter interpretieren und sie somit in Abhängigkeit von den Übertragungseigenschaften des Kanals auswerten.

Dabei spiegeln die Kanalparameter eines Übertragungskanals dessen Übertragungseigenschaften wieder und zwar in diesem Zusammenhang insbesondere auch das Gedächtnis des betrachteten Kanals. Wie bereits eingangs des Kapitels sowie auch in Kapitel 2 und Kapitel 3 angemerkt, beeinflußt das Gedächtnis bzw. die Stärke des Gedächtnisses das Auftreten von Bündelfehlern während der Übertragung. So hängt vom Gedächtnis des betrachteten Übertragungskanals ganz erheblich ab, in welchem Maße und bis zu welcher Länge Bündelfehler jeweils in einem realen Übertragungskanal auftreten; d.h. die Stärke des Gedächtnisses beeinflußt, inwieweit die Übertragungsfehler gebündelt auftreten, und bedingt auf diese Art, wie häufig und wie lang die aufgetretenen Bündelfehler sind. Aufgrund dessen nehmen die Kanalparameter des betrachteten Übertragungskanals in ganz erheblichem Maße Einfluß darauf, ob der Einsatz eines REC- oder der eines BEC-Codes von Vorteil ist.

Unter dem Begriff, Kanalparameter, fassen wir die folgenden Größen zusammen, die gemeinsam das Fehlerverhalten des betrachteten Übertragungskanals charakterisieren: Die Bitfehlerrate $p_E$ sowie die Verteilung der Zufallsvariablen, die die Aufenthaltszeiten des Markov'schen Hintergrundprozesses $(X_t)_{t \in \mathbb{N}}$ in den einzelnen Zuständen $i \in \mathcal{S}$ beschreiben. Dabei spielt die Verteilung der Zufallsvariablen, die die Aufenthaltszeiten des Markov'schen Hintergrundprozesses $(X_t)_{t \in \mathbb{N}}$ in den einzelnen Zuständen $i \in \mathcal{S}$ angeben, bei der Beschreibung und Modellierung des aufgetretenen Fehlerprozesses $(E_t)_{t \in \mathbb{N}}$ eine entscheidende Rolle. Nach dieser Verteilung richtet sich schließlich, wie lang der Aufenthalt des Markov'schen Hintergrundprozesses $(X_t)_{t \in \mathbb{N}}$ in einem erreichten Zustand $i$ mit $i \in \mathcal{S}$ jeweils ist; d.h. mit dieser Verteilung wird auch die Aufenthaltsdauer in den Zuständen $i$ mit $i \in \mathcal{S}$ festgelegt, in denen häufiger und damit mit einer größeren Wahrscheinlichkeit Übertragungsfehler generiert werden. Wie wahrscheinlich dabei das Auftreten eines Übertragungsfehlers in einem

beliebigen Zustand $i$ mit $i \in \mathcal{S}$ ist, wird durch die in Abschnitt 3.3.2 Definition 3.3.7 erklärten Fehlerwahrscheinlichkeiten $e_i$ mit $i \in \mathcal{S}$ bestimmt. Da die Verteilung der Aufenthaltszeiten gerade die Dauer des Aufenhaltes in einem Zustand $i$ mit $i \in \mathcal{S}$ festlegt, der wiederum durch die Fehlerwahrscheinlichkeit $e_i$ mit $i \in \mathcal{S}$ charakterisiert wird, bedingt die Verteilung der Aufenthaltszeiten, wie wahrscheinlich insgesamt das Auftreten eines Bündelfehlers ist und mit welcher Wahrscheinlichkeit dieser von einer gewissen Länge ist. So ist das Auftreten kürzerer Bündelfehler wahrscheinlicher als das längerer.

### 5.2.1 Bezeichnungen

*Im folgenden bezeichnet der Parameter $E_b$ die durchschnittliche Länge der Bündelfehler, d.h. die mittlere Bündelfehlerlänge.*

Da die Verteilung der Aufenthaltszeiten bedingt, mit welcher Wahrscheinlichkeit Bündelfehler einer gewissen Länge auftreten, bestimmt diese Verteilung also auch die durchschnittliche Länge der Bündelfehler, d.h. die mittlere Bündelfehlerlänge $E_b$. Dabei gilt, je größer die mittlere Bündelfehlerlänge $E_b$ ist, desto wahrscheinlicher wird das Auftreten von Bündelfehlern insgesamt und desto größer wird auch die Wahrscheinlichkeit für das Auftreten längere Bündelfehler sein. Da die mittlere Bündelfehlerlänge $E_b$ nun durch die Verteilung der Aufenthaltszeiten charakterisiert wird, fassen wir der Einfachheit halber für die folgenden Überlegungen unter dem Begriff, Kanalparameter, die Bitfehlerrate $p_E$ sowie die mittlere Bündelfehlerlänge $E_b$ zusammen.

In den beiden folgenden Abschnitten 5.2.1 und 5.2.2 betrachten wir für die Verteilung der Aufenthaltszeiten beispielhaft zwei verschiedene Verteilungen. So gehen wir in Abschnitt 5.2.1 der Einfachheit halber zunächst davon aus, dass die Zufallsvariablen, die die Aufenthaltszeiten in den einzelnen Zuständen beschreiben, geometrisch verteilt sind. Anschließend legen wir in Abschnitt 5.2.2 eine Poisson Verteilung dieser Zufallsvariablen zugrunde. In beiden Abschnitten berechnen wir in Abhängigkeit von diesen beiden verschiedenen Verteilungen sowie der Bitfehlerrate $p_E$ die Restfehlerwahrscheinlichkeiten für REC- und BEC-Codes zwecks des angestrebten Vergleichs zwischen beiden Codetypen. Dazu geben wir sowohl für die geometrische Verteilung in Abschnitt 5.2.1 als auch die Poisson Verteilung in Abschnitt 5.2.2 entsprechende Formeln für die Berechnung der Restfehlerwahrscheinlichkeit an, so dass wir diese dann abhängig von der Bitfehlerrate $p_E$ sowie der mittleren Bündelfehlerlänge $E_b$ auswerten können. Auf dieser Basis können wir die Leistungsfähigkeit von REC-Codes gegenüber BEC-Codes in Abhängigkeit von den Kanalparametern beurteilen und die Effizienz von REC-Codes und BEC-Codes miteinander vergleichen.

## 5.2.1 Geometrische Verteilung

Ziel dieses Abschnittes ist es, die Restfehlerwahrscheinlichkeit für REC- und BEC-Codes abhängig von den Kanalparametern, der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_b$, zu berechnen, um auf diese Weise einen Vergleich von REC- gegenüber BEC-Codes hinsichtlich der Effizienz ermöglichen zu können. Dazu wählen wir in diesem Abschnitt der Einfachheit halber ein simples Beispiel für die Verteilung der Aufenthaltszeiten, die gerade die Darstellung der mittleren Bündelfehlerlänge $E_b$ charakterisiert,

und zwar die geometrische Verteilung. Anhand dieses einfachen Beispiels läßt sich die Beziehung zwischen den Kanalparametern und der Restfehlerwahrscheinlichkeit auf sehr einfache Weise besonders anschaulich darlegen, denn die Auswertung der Restfehlerwahrscheinlichkeit kann basierend auf einem sehr übersichtlichen und einfachen Kanalmodell realisiert werden. Daher läßt sich erstens die mittlere Bündelfehlerlänge $E_b$ sehr leicht als der Erwartungswert der geometrischen Verteilung ermitteln und zweitens läßt sich die Vorgehensweise bei der Auswertung der Restfehlerwahrscheinlichkeit deutlich herausarbeiten. Demzufolge kann für die geometrische Verteilung der Aufenthaltszeiten bereits eine erste einfache Analyse im Hinblick auf die Leistungsfähigkeit von REC- gegenüber BEC-Codes erstellt werden, was einen ersten Überblick in Bezug auf den angestrebten Vergleich liefert. Anhand dieses Beispiels präsentieren wir in diesem Abschnitt also eine übersichtliche Darstellung der Zusammenhänge zwischen den Kanalparametern und der Restfehlerwahrscheinlichkeit, bevor wir im nächsten Abschnitt dann in Gestalt der Poisson Verteilung eine komplexere Verteilung betrachten, für die die Auswertung der Restfehlerwahrscheinlichkeit bzw. der Vergleich von REC- und BEC-Codes auf einem allgemeineren Kanalmodell erfolgen muß.

In diesem Abschnitt verfolgen wir dazu die folgende Vorgehensweise. Zunächst ist für einen Fehler-korrigierenden $(n,k)$-Blockcode $\mathcal{C}$ die Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}$ als Funktion der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_b$ zu interpretieren. In Abschnitt 4.2.2 haben wir sowohl für REC- als auch für BEC-Codes der Länge $n$ jeweils passende Formeln zur Berechnung der Restfehlerwahrscheinlichkeit hergeleitet. So haben wir in Satz 4.2.20 nachgewiesen, dass die Restfehlerwahrscheinlichkeit für einen $r$-REC-Code der Länge $n$ aus der Gleichung

$$\mathcal{R}_\mathcal{C}(r,n) = 1 - \sum_{i=0}^{r} \sum_{j=1}^{N} P_j(i,n)$$

resultiert. Eine entsprechende Darstellung für die Restfehlerwahrscheinlichkeit eines $b$-BEC-Codes der Länge $n$ liefert Satz 4.2.23 mit

$$\mathcal{R}_\mathcal{C}(b,n) \;=\; 1 - \sum_{j=1}^{N} P_j(0,n) - \sum_{l=0}^{n-b} \sum_{j=1}^{N} \sum_{k=1}^{N} P_j^b(l,b) P_{jk}(0,n-l-b) - \sum_{l=n-b+1}^{n-1} \sum_{j=1}^{N} P_j^b(l,n-l).$$

In den Beweisen zu den Sätzen 4.2.20 und 4.2.23 haben wir jeweils rekursive Gleichungen zur Berechnung der Wahrscheinlichkeiten $P_j(i,n)$, $P_j(0,n)$, $P_j(l,b)$ und $P_{jk}(0,n-l-b)$ für alle $0 \leq i \leq r$, für alle $0 \leq l \leq n-b$ und alle $j,k \in \mathcal{S}$ entwickelt. Diese Rekursionen basieren auf Produkten der Übergangswahrscheinlichkeiten $q_{ij}$ mit $i,j \in \mathcal{S}$ und der Fehlerwahrscheinlichkeiten $e_j$ mit $j \in \mathcal{S}$. Indem wir die Übergangswahrscheinlichkeiten $q_{ij}$ mit $i,j \in \mathcal{S}$ sowie die Fehlerwahrscheinlichkeiten $e_j$ mit $j \in \mathcal{S}$ als Funktionen der Kanalparameter, d.h. der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_b$, darstellen, können wir sämtliche der obigen Wahrscheinlichkeiten und demnach auch die Restfehlerwahrscheinlichkeiten für REC- oder BEC-Codes als Funktionen der Kanalparameter $p_E$ und $E_b$ interpretieren. In diesem Zusammenhang wird die Darstellung der Übergangswahrscheinlichkeiten und Fehlerwahrscheinlichkeiten abhängig von den Kanalparametern $p_E$ und $E_b$ durch die obige Voraussetzung, dass die Aufenthaltszeiten geometrisch verteilt sind, immens vereinfacht.

In Abschnitt 4.2.2 haben wir die Rekursionen sowie die Formel zur Berechnung der Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}(r,n)$ eines $r$-REC-Codes bzw. zur Berechnung der Restfehlerwahrscheinlichkeit $\mathcal{R}_\mathcal{C}(b,n)$ eines $b$-BEC-Codes basierend auf dem von uns in Abschnitt 3.4 entwickelten Kanalmodell hergeleitet. Dieses Modell beruht auf einem Markov'schen

Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$, der gekennzeichnet ist durch einen endlichen Zustandsraum $\mathcal{S} = \{1, 2, \ldots, N\}$, den Anfangszustand $X_0 = i_0$, mit $i_0 \in \mathcal{S}$, sowie die Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$. Nun setzen wir in diesem Abschnitt voraus, dass die Aufenthaltszeiten in den einzelnen Zuständen $i \in \mathcal{S}$ geometrisch verteilt sind, so dass nach Bemerkung 3.1.21 in Abschnitt 3.1 die Zufallsvariablen, die die Aufenthaltszeiten von $(X_t)_{t\in\boldsymbol{N}}$ in den einzelnen Zuständen angeben, gedächtnislos sind. Aufgrund dessen können wir der Einfachheit halber den Zustandsraum des zugrunde liegenden Kanalmodells auf zwei Zustände reduzieren, einen guten Übertragungszustand $G$, in dem nur wenige oder gar keine Übertragungsfehler auftreten, und einen schlechten Übertragungszustand $B$, der viele Übertragungsfehler generiert. Aus diesem Grund werten wir die rekursiven Gleichungen aus Abschnitt 4.2.2 zur Berechnung der Wahrscheinlichkeiten $P_j(i, n)$, $P_j(0, n)$, $P_j(l, b)$ und $P_{jk}(0, n-l-b)$ für alle $0 \le i \le r$, für alle $0 \le l \le n-b$ und alle $j, k \in \mathcal{S}$ auf der Basis des Gilbert Modells aus und bestimmen damit auch die Restfehlerwahrscheinlichkeiten für REC- und BEC-Codes auf diesem einfachen Modell. Das bedeutet, dass wir uns auf diese Art zur Auswertung der rekursiven Gleichungen die Vorteile des Modells von Gilbert, d.h. dessen Einfachheit und Übersichtlichkeit, zunutze machen. Aufgrund der guten Übersichtlichkeit des Modells lassen sich die Zusammenhänge zwischen den wesentlichen Größen auf einfache Weise sehr leicht darstellen.

Nach Abschnitt 3.3.2 beruht das Gilbert Modell auf einem Markov'schen Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ mit einem Zustandsraum $\mathcal{S} = \{G, B\}$ bestehend aus zwei Zuständen $G$ und $B$. Dabei ist der Markov'sche Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ charakterisiert durch seinen Anfangszustand $X_0 = i_0$, mit $i_0 \in \{G, B\}$ sowie seine Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\{G,B\}}$, wobei für die Übergangswahrscheinlichkeiten gilt:

$$q_{GG} = 1 - p, \quad q_{GB} = p, \quad q_{BG} = q \quad \text{und} \quad q_{BB} = 1 - q.$$

Gemäß Bemerkung 3.3.12 sind die Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ gegeben durch

$$e_G = 0 \qquad \text{und} \qquad e_B = 1/2.$$

Ferner werden die stationären Zustandswahrscheinlichkeiten $\pi_G$ und $\pi_B$ nach Abschnitt 3.3.2 festgelegt durch die nachfolgenden Gleichungen:

$$\pi_G = \frac{q_{BG}}{q_{GB} + q_{BG}} = \frac{q}{p+q} \quad \text{und} \quad \pi_B = \frac{q_{GB}}{q_{GB} + q_{BG}} = \frac{p}{p+q}.$$

Da nach der Konstruktion des Gilbert Modells Übertragungsfehler nur im Zustand $B$ auftreten können und die Zufallsvariablen, die die Aufenthaltszeiten in den beiden Zuständen $G$ und $B$ angeben, geometrisch verteilt sind, ist folglich in diesem Fall die Dauer, d.h. die Länge der Bündelfehler, geometrisch verteilt. Demnach ist die mittlere Bündelfehlerlänge $E_b$ gegeben als der Erwartungswert dieser geometrischen Verteilung mit

$$E_b = \frac{1}{q}.$$

Mittels der vorangehenden Gleichungen lassen sich die Übergangs- und Fehlerwahrscheinlichkeiten nun auf ganz einfache Art und Weise in Abhängigkeit von den Kanalparametern, der Bitfehlerrate $p_E$ sowie der mittleren Bündelfehlerlänge $E_b$, darstellen. Dabei sind gemäß

den Vorgaben des Gilbert Modells die beiden Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ fest vorgegeben mit den Werten

$$e_G = 0 \quad \text{und} \quad e_B = 1/2.$$

Da die mittlere Bündelfehlerlänge $E_b$ als der Erwartungswert der geometrischen Verteilung der Aufenthaltszeiten durch $E_b = 1/q$ gegeben ist, können wir die Übergangswahrscheinlichkeiten $q_{BG}$ und $q_{BB}$ als Funktionen der mittleren Bündelfehlerlänge $E_b$ kennzeichnen:

$$\begin{aligned} q_{BG} &= q = \frac{1}{E_b} \\ \text{und} \quad q_{BB} &= 1 - q = 1 - \frac{1}{E_b}. \end{aligned}$$

Ferner gilt bezüglich der Bitfehlerrate $p_E$ nach der Formel von Bayes

$$p_E = e_G\,\pi_G + e_B\,\pi_B = \frac{1}{2}\,\pi_B,$$

so dass wir zusammen mit den stationären Zustandswahrscheinlichkeiten $\pi_G$ und $\pi_B$, mit

$$\pi_G = \frac{q}{p+q} \quad \text{und} \quad \pi_B = \frac{p}{p+q},$$

die Übergangswahrscheinlichkeiten $q_{GG}$ und $q_{GB}$ als Funktionen der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_b$ darstellen können:

$$\begin{aligned} q_{GB} &= p = \frac{2\,p_E}{E_b(1-2\,p_E)} \\ \text{und} \quad q_{GG} &= 1 - p = 1 - \frac{2\,p_E}{E_b(1-2\,p_E)}. \end{aligned}$$

Da wir die Übergangs- und Fehlerwahrscheinlichkeiten auf derart einfache Weise als Funktionen der Kanalparameter $p_E$ und $E_b$ angeben können,

$$\begin{aligned} e_G &= 0 && \text{und} \quad e_B = \frac{1}{2} \\ q_{BG} &= \frac{1}{E_b} && \text{und} \quad q_{BB} = 1 - \frac{1}{E_b} \\ q_{GB} &= \frac{2\,p_E}{E_b(1-2\,p_E)} && \text{und} \quad q_{GG} = 1 - \frac{2\,p_E}{E_b(1-2\,p_E)}, \end{aligned}$$

hängt folglich auch die Auswertung der rekursiven Gleichungen aus Abschnitt 4.2.2 von den beiden Parametern $p_E$ und $E_b$ ab. Demnach können wir aufgrund aller bisherigen Ergebnisse aus den vorangegangenen Kapiteln die Restfehlerwahrscheinlichkeiten für REC- und BEC-Codes nun abhängig von der Bitfehlerrate $p_E$ sowie der mittleren Bündelfehlerlänge $E_b$ auswerten und die Leistungsfähigkeit von REC- gegenüber BEC-Codes somit in Abhängigkeit von der Fehlerstatistik des betrachteten Übertragungskanals beurteilen. Da die mittlere Bündelfehlerlänge $E_b$ bedingt, wie wahrscheinlich das Auftreten von Bündelfehlern ist

und mit welcher Wahrscheinlichkeit Bündelfehler einer gewissen Länge auftreten, nimmt die mittlere Bündelfehlerlänge $E_b$ daher großen Einfluß darauf, ob der Einsatz eines REC-Codes dem eines BEC-Codes vorzuziehen ist. Aus diesem Grund ist die Berechnung der Restfehlerwahrscheinlichkeit und damit die Bewertung von REC- gegenüber BEC-Codes gerade in Abhängigkeit von einer variierenden mittleren Bündelfehlerlänge $E_b$ von besonderem Interesse.

Nach sämtlichen bisherigen Ergebnissen hängt die Restfehlerwahrscheinlichkeit eines REC-Codes genauso wie diejenige eines BEC-Codes von drei Parametern ab. So sind die entsprechenden rekursiven Gleichungen aus Abschnitt 4.2.2 zur Bestimmung der Restfehlerwahrscheinlichkeit eines REC- bzw. eines BEC-Codes nun gebunden an die Codelänge $n$ sowie die beiden Kanalparameter, die Bitfehlerrate $p_E$ und die mittlere Bündelfehlerlänge $E_b$. Darüber hinaus hängen die rekursiven Gleichungen bezüglich der REC natürlich auch von dem Parameter $r$ der REC-Korrekturfähigkeit ab und die Rekursionen bezüglich der BEC von dem Parameter $b$ der BEC-Korrekturfähigkeit. Ziel ist es, wie oben erwähnt, die Restfehlerwahrscheinlichkeiten nun jeweils für eine variierende mittlere Bündelfehlerlänge $E_b$ zu berechnen, um dann anhand der erzielten Resultate entscheiden zu können, ob der Einsatz eines REC-Codes dem eines BEC-Codes vorzuziehen ist, und um auf diese Art einen optimalen Blockcode zur Fehlerkorrektur auswählen zu können. Für einen aussagekräftigen Vergleich von REC- gegenüber BEC-Codes sind daher die beiden übrigen Parameter, die Codelänge $n$ und die Bitfehlerrate $p_E$, bei der Auswertung der Rekursionen fest vorzugeben. Da nun für eine derartige Gegenüberstellung von REC- und BEC-Codes eine fundierte Grundlage, also die Existenz REC- und BEC-Codes der gleichen Codelänge $n$ und mit der selben Coderate $k/n$, gewährleistet sein muß, orientieren wir uns bei der Vorgabe der Werte für die Codelänge $n$ an den in Tabelle 5.1 vorgegebenen bzw. ermittelten Werten. In Tabelle 5.1 haben wir schließlich die Existenz von Blockcodes mit unterschiedlichem Fehlerkorrekturverhalten aber der gleichen Codelänge $n$ und der selben Dimension $k$ bzw. einer annähernd gleich großen Coderate $k/n$ nachgewiesen. Bei der Festlegung des zweiten Parameters, der Bitfehlerrate $p_E$, richten wir uns nach den Erfordernissen aus der Praxis und geben somit für die Bitfehlerrate $p_E$ Werte in der Höhe vor, wie sie für bestimmte Anwendungen in der Praxis, wie z.B. bei der Sprach- oder der Datenübertragung, zwingend erforderlich sind.

Bevor wir nun einen Vergleich der Leistungsfähigkeit und Effizienz von REC- gegenüber BEC-Codes in Form einer graphischen Darstellung präsentieren, skizzieren wir kurz die verschiedenen Szenarien, die wir bei der Auswertung der rekursiven Gleichungen zur Berechnung der Restfehlerwahrscheinlichkeiten von REC-Codes und von BEC-Codes realisiert haben.

Für das erste Szenario legen wir den Parameter der Codelänge $n$ in Anlehnung an die Ergebnisse aus Tabelle 5.1 auf den Wert $n = 63$ fest. So hat Tabelle 5.1 gezeigt, dass für die Codelänge $n = 63$ verschiedene zyklische Blockcodes der gleichen Dimension $k = 51$ existieren, wobei einer von beiden als ein 2-REC-Code und der andere als ein 5-BEC-Code charakterisiert werden kann. Von daher repräsentieren diese beiden zyklischen $(63, 51)$-Blockcodes einen geeigneten Bezugspunkt für eine Gegenüberstellung von REC- und BEC-Codes hinsichtlich ihrer Effizienz bei der Fehlerkorrektur. Bei der Höhe des Wertes für die Bitfehlerrate $p_E$ orientieren wir uns an dem Wert, der für eine zuverlässige Sprachübertragung über eine Mobilfunkstrecke zwingend erforderlich ist. Dies ist genau dann der Fall, wenn $p_E < 10^{-4}$ ist. Wir setzen daher die Bitfehlerrate auf $p_E = 10^{-4}$. Der dritte Parameter, die mittlere Bündelfehlerlänge $E_b$, beeinflußt das Auftreten von Bündelfehlern sowie die

Länge der aufgetretenen Bündelfehler wesentlich. Demzufolge wird von dem Wert dieses Parameters abhängen, ob zur Fehlerkorrektur ein REC- oder ein BEC-Code eher geeignet ist. Daher gestalten wir diesen Parameter variabel und berechnen abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ die Restfehlerwahrscheinlichkeiten von REC- und BEC-Codes der Länge $n = 63$ bei einer Bitfehlerrate von $p_E = 10^{-4}$, um REC- und BEC-Codes dann auf der Basis der erzielten Resultate in Bezug auf ihre Effizienz einander gegenüber zu stellen.

Für das zweite Szenario legen wir für die Bitfehlerrate $p_E$ einen verringerten Wert fest, da für viele Anwendungen der Praxis, wie beispielsweise die Datenübertragung über Kabel oder auch über eine Mobilfunkstrecke, eine Bitfehlerrate von $p_E = 10^{-4}$ nicht tragbar ist. So ist im Falle der Datenübertragung beispielsweise erst eine Bitfehlerrate von $p_E < 10^{-10}$ akzeptabel. Wir ermitteln daher die Restfehlerwahrscheinlichkeiten von REC- und BEC-Codes erneut abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ allerdings für eine feste Bitfehlerrate von $p_E = 10^{-10}$ und behalten dabei für den Parameter der Codelänge $n$ den Wert $n = 63$ bei.

Für das dritte Szenario ändern wir abschließend den Wert für den Parameter der Codelänge $n$ ab, denn in Tabelle 5.1 haben wir ebenfalls die Existenz von REC- und BEC-Codes der Länge $n = 105$ mit einer annähernd gleich großen Coderate $k/n$ belegt. Indem wir die Restfehlerwahrscheinlichkeiten von REC- und BEC-Codes wiederum in Abhängigkeit von einer variierenden mittleren Bündelfehlerlänge $E_b$ ermitteln und dabei den Wert für die Bitfehlerrate $p_E$ mit $p_E = 10^{-4}$ wieder an den Erfordernissen für eine zuverlässige Sprachübertragung orientieren, können wir im Vergleich mit dem ersten Szenario darlegen, inwieweit die gesteigerte Codelänge $n$ Einfluß auf die Fehlerkorrekturfähigkeiten der Codes und damit auch auf die Restfehlerwahrscheinlichkeiten nimmt.

Zum Abschluß dieses Abschnittes führen wir nun eine Bewertung von REC- gegenüber BEC-Codes durch und präsentieren einen Vergleich ihrer Leistungsfähigkeit und Effizienz für jedes der drei skizzierten Szenarien in Form einer graphischen Darstellung. Dazu berechnen wir für die drei aufgezeigten Szenarien die Restfehlerwahrscheinlichkeiten von REC- sowie von BEC-Codes mit Hilfe eines C-Programms, welches lediglich die rekursiven Gleichungen aus Abschnitt 4.2.2 zur Berechnung der Restfehlerwahrscheinlichkeiten löst. Die in diesem Rahmen erzielten Resultate setzen wir für jedes der drei Szenarien in einer graphischen Präsentation der ausgewerteten Daten um. Die Abbildungen 5.1 bis 5.4 beinhalten somit eine Bewertung der Leistungsfähigkeit in Bezug auf die REC bzw. die BEC. In jeder dieser Grafiken sind die Restfehlerwahrscheinlichkeiten sowohl nach der Korrektur von *random errors* als auch nach der Korrektur von Bündelfehlern abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ dargestellt, ohne dass zur Bestimmung der Restfehlerwahrscheinlichkeiten jedoch ein spezieller Code realisiert worden ist. Demzufolge lassen sich REC- gegenüber BEC-Codes anhand dieser Grafiken dann bewerten, so dass wir auf diese Art und Weise einen allgemeinen Leistungsvergleich von REC- und BEC-Codes realisieren.

In Abbildung 5.1 präsentieren wir zunächst die Ergebnisse bezüglich des ersten Szenarios. Dieses setzt voraus, dass die Bitfehlerrate $p_E$ mit dem Wert $p_E = 10^{-4}$ fest vorgegeben ist. Wir stellen in dieser Grafik die Restfehlerwahrscheinlichkeiten für REC- und BEC-Codes abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ nach der Korrektur einer gewissen Anzahl *random errors* bzw. nach der Korrektur von Bündelfehlern bis zu einer gewissen Länge dar. In diesem Zusammenhang repräsentieren die durchgezogenen Kurven die Restfehlerwahrscheinlichkeiten für REC-Codes, die 2 bzw. 3 *random errors* korrigieren

können. Im Gegensatz dazu kennzeichnen die gestrichelten Kurven die Restfehlerwahrscheinlichkeiten für die BEC nach der Korrektur aller Bündelfehler bis zur Länge 3, 4, 5 bzw. 6.

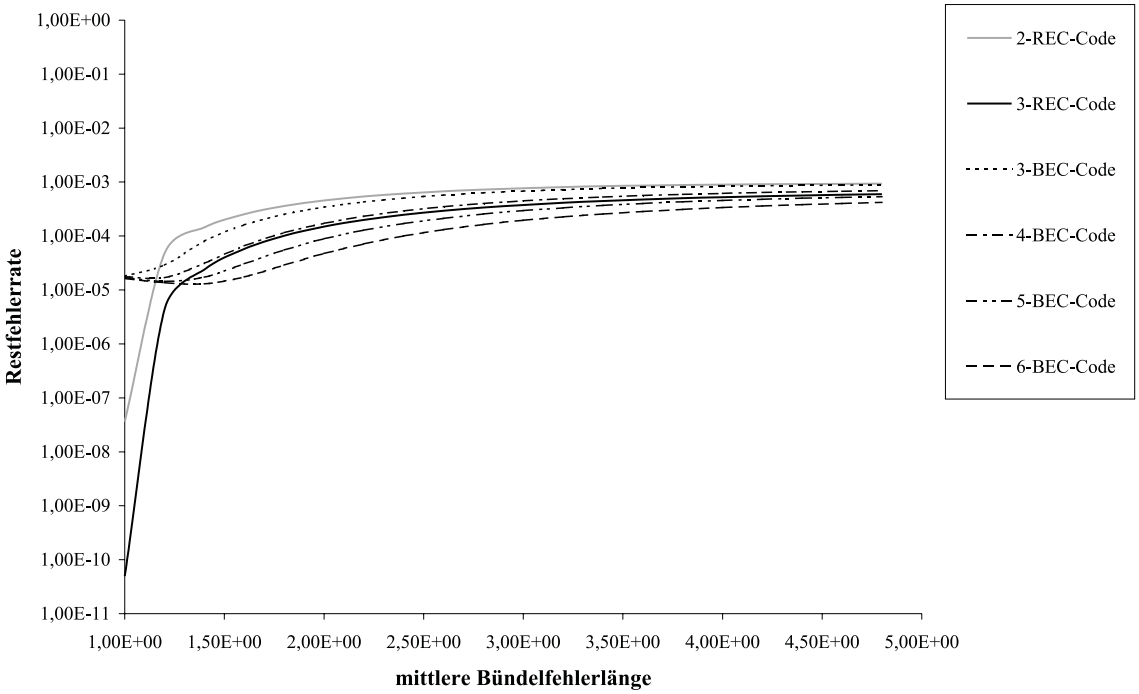


Abbildung 5.1: Vergleich der Restfehlerwahrscheinlichkeiten REC- und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 10^{-4}$

In Grafik 5.1 sind gemäß den Anweisungen des ersten Szenarios die Restfehlerwahrscheinlichkeiten für zyklische Blockcodes der Länge $n = 63$ eingetragen. Zum direkten Leistungsvergleich greifen wir uns nun insbesondere die Restfehlerwahrscheinlichkeit eines 2-REC-Codes gegenüber derjenigen eines 5-BEC-Codes heraus. In Tabelle 5.1 haben wir schließlich nachgewiesen, dass zwei verschiedene $(63, 51)$-Blockcodes mit genau diesen Fehlerkorrektureigenschaften existieren. Der erste in Tabelle 5.1 angegebene $(63, 51)$-Blockcode kann als Beispiel für einen 2-REC-Code betrachtet werden und wird gewöhnlich zur REC in wireless ATM-Netzen eingesetzt, vgl. dazu [28]. Ein Beispiel für einen 5-BEC-Code liefert der zweite in Tabelle 5.1 aufgeführte $(63, 51)$-Blockcode. Da beide Blockcodes von der selben Coderate $k/n$ sind, repräsentieren diese beiden zyklischen $(63, 51)$-Blockcodes daher einen geeigneten Referenzpunkt für einen Vergleich der Effizienz von 2-REC-Codes gegenüber 5-BEC-Codes. Aus diesem Grund greifen wir die Restfehlerwahrscheinlichkeiten für 2-REC-Codes und für 5-BEC-Codes aus Abbildung 5.1 heraus und stellen diese nochmals gesondert in der folgenden Abbildung 5.2 einander gegenüber.

Nachfolgend bewerten wir in Abbildung 5.2 2-REC-Codes gegenüber 5-BEC-Codes und erläutern dazu den Kurvenverlauf der Restfehlerwahrscheinlichkeit für beide Codetypen an markanten Punkten. So ist zu Beginn für eine mittlere Bündelfehlerlänge von $E_b = 1$ der Einsatz eines 2-REC-Codes offensichtlich dem eines 5-BEC-Codes vorzuziehen. Falls die mittlere Bündelfehlerlänge $E_b = 1$ ist, ist auch die Übergangswahrscheinlichkeit $q_{BG} = 1$. Das bedeutet, dass der betrachtete Übertragungskanal keine Bündelfehler generiert und das zugrunde liegende Kanalmodell damit im Prinzip einen symmetrischen Binärkanal darstellt,

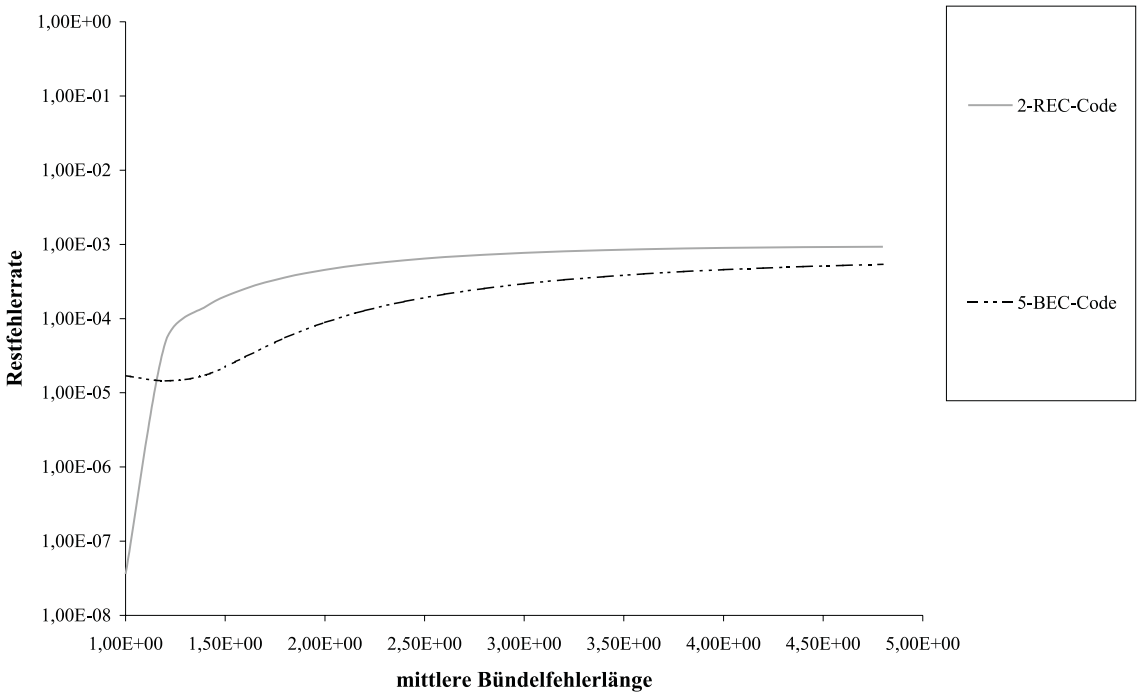


Abbildung 5.2: Vergleich der Restfehlerwahrscheinlichkeiten eines 2-REC-Codes und eines 5-BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 10^{-4}$

vgl. Abschnitt 3.3.1. Für einen symmetrischen Binärkanal, in dem die Übertragungsfehler alle zufällig verteilt und voneinander unabhängig sind und keine Bündelfehler auftreten, ist der Einsatz eines 2-REC-Codes zur Fehlerkorrektur offensichtlich wesentlich günstiger und effizienter, denn aus Abbildung 5.2 geht hervor, dass die Restfehlerwahrscheinlichkeit eines 2-REC-Codes für $E_b = 1$ um den Faktor 400 niedriger liegt als die eines 5-BEC-Codes. Von daher ist der Einsatz eines 2-REC-Codes zur Fehlerkorrektur bei einer mittleren Bündelfehlerlänge von $E_b = 1$ gegenüber dem Einsatz eines 5-BEC-Codes zu favorisieren.

Falls jedoch während der Übertragung bereits wenige Bündelfehler auftreten und die mittlere Bündelfehlerlänge $E_b$ bereits minimal wächst, gewinnt der 5-BEC-Code zunehmend an Effizienz, so dass die Bedeutung der BEC stetig zunimmt. Gleichzeitig sinkt die Effizienz eines 2-REC-Codes, was sich in einer rapide wachsenden Restfehlerwahrscheinlichkeit für einen 2-REC-Code niederschlägt. Für Werte der mittleren Bündelfehlerlänge $E_b$, die nahe bei 1 liegen, ist die Restfehlerwahrscheinlichkeit eines 2-REC-Codes noch immer geringer als die eines 5-BEC-Codes. Die Differenz nimmt jedoch mit wachsender mittlerer Bündelfehlerlänge $E_b$ stetig in großem Maße ab, bis sich die beiden Kurven schließlich bei einer mittleren Bündelfehlerlänge von $E_b = 1.3$ schneiden.

Überschreitet die mittlere Bündelfehlerlänge $E_b$ diesen Wert 1.2, dann ist die Restfehlerwahrscheinlichkeit eines 5-BEC-Codes für den gesamten Bereich $E_b > 1.2$ geringer als die Restfehlerwahrscheinlichkeit eines 2-REC-Codes, so dass der Einsatz eines 5-BEC-Codes für $E_b > 1.2$ zu bevorzugen ist. Die maximale Differenz, d.h. der maximale Vorteil, für einen 5-BEC-Code wird bei einer mittleren Bündelfehlerlänge von $E_b = 1.4$ erreicht. Nach diesem Punkt geht die Differenz zwischen den beiden Restfehlerwahrscheinlichkeiten wieder leicht zurück, da für eine sehr große mittlere Bündelfehlerlänge, wie z.B. $E_b > 4$, die Wahrscheinlichkeit für das Auftreten sehr langer Bündelfehler zunimmt, die dann von einem

5-BEC-Code auch nicht mehr korrigiert werden können. Insgesamt halten wir fest, dass nur für eine sehr kleine mittlere Bündelfehlerlänge $E_b < 1.2$ die Restfehlerwahrscheinlichkeit eines 2-REC-Codes niedriger als diejenige eines 5-BEC-Codes ist und damit ein 2-REC-Code zur Fehlerkorrektur besser geeignet ist.

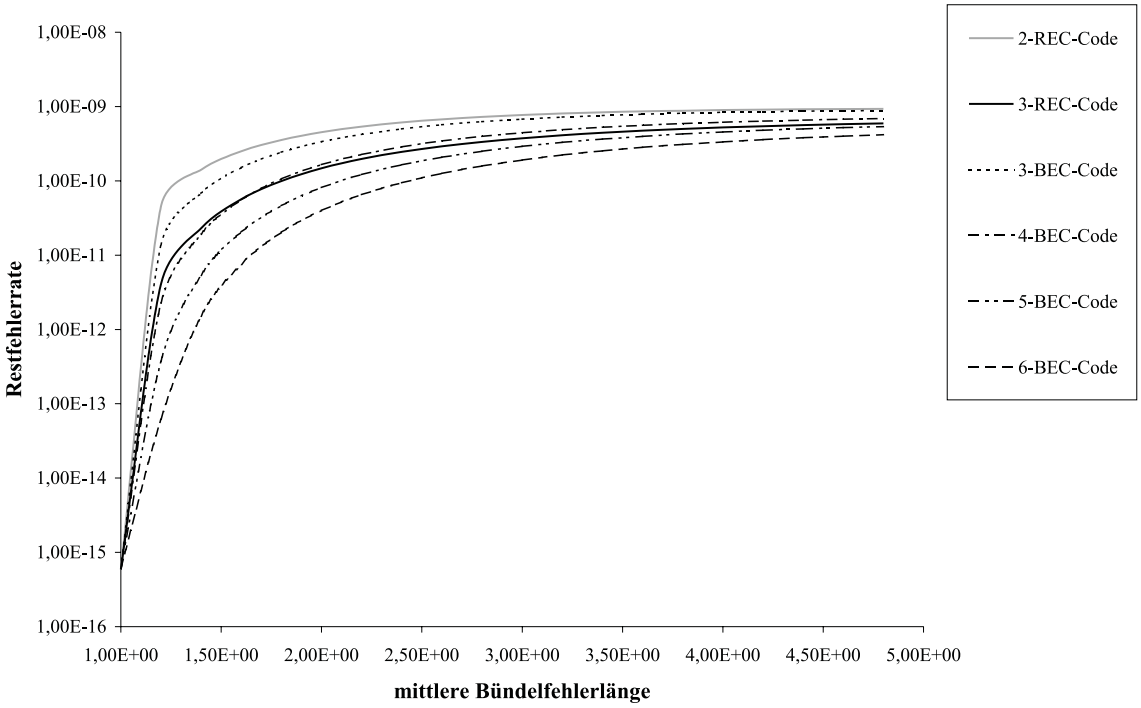


Abbildung 5.3: Vergleich der Restfehlerwahrscheinlichkeiten REC-und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 10^{-10}$

Eine derartige Entwicklung bzw. diese Kurvenverläufe der Restfehlerwahrscheinlichkeiten werden durch die Auswertungen der beiden anderen Szenarien bestätigt. In Abbildung 5.3 präsentieren wir grafisch die Daten, die die Auswertung des zweiten Szenarios ergeben hat. Wiederum stellen wir die Restfehlerwahrscheinlichkeiten nach der Korrektur einer gewissen Anzahl *random errors* sowie nach der Korrektur von Bündelfehlern bis zu einer gewissen Länge abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ für Fehlerkorrigierende Codes der Länge $n = 63$ dar. Allerdings zeigt diese Abbildung die Restfehlerwahrscheinlichkeiten für eine veränderte Bitfehlerrate $p_E$, in diesem Fall ist $p_E$ fest gesetzt mit $p_E = 10^{-10}$ gemäß den erforderlichen Werten für eine zuverlässige Datenübertragung. Wie in den vorangehenden Abbildungen kennzeichnen die durchgezogenen Kurven auch hier die Restfehlerwahrscheinlichkeiten für die REC nach der Korrektur von 2 bzw. 3 *random errors* und die gestrichelten Kurven repräsentieren die Restfehlerwahrscheinlichkeiten für BEC-Codes, die sämtliche Bündelfehler bis zur Länge 3, 4, 5 bzw. 6 korrigieren können.

Da wir im Rahmen des zweiten Szenarios wiederum Fehler-korrigierende Codes der Länge $n = 63$ betrachten, stellen wir wegen Tabelle 5.1 insbesondere wieder den Verlauf der Restfehlerwahrscheinlichkeiten von 2-REC-Codes sowie 5-BEC-Codes einander gegenüber. Mit Abbildung 5.3 können wir die obigen Aussagen zu den Abbildungen 5.1 und 5.2 in Bezug auf den Kurvenverlauf der Restfehlerwahrscheinlichkeiten von REC- und BEC-Codes bestätigen. Für eine mittlere Bündelfehlerlänge von $E_b = 1$ ist die Restfehlerwahrscheinlichkeit eines 2-REC-Codes geringer als die eines 5-BEC-Codes, so dass damit ein 2-REC-Code

zur Fehlerkorrektur geeigneter und dessen Einsatz zur Fehlerkorrektur zu bevorzugen ist. Auch in dieser Grafik zeigt sich, ähnlich wie in Abbildung 5.1, dass bei wachsender mittlerer Bündelfehlerlänge $E_b$ die Restfehlerwahrscheinlichkeit eines 2-REC-Codes rapide ansteigt und der Effizienzvorteil eines 2-REC-Codes gegenüber einem 5-BEC-Codes von daher stetig zurück geht, während gleichzeitig die Bedeutung der BEC zunimmt. So liegt der Schnittpunkt für die beiden Kurven im Rahmen dieses Szenarios bei $E_b = 1.1$. Überschreitet die mittlere Bündelfehlerlänge $E_b$ den Wert 1.1, so ist der 5-BEC-Code in dem gesamten Bereich $E_b > 1.1$ aufgrund einer niedrigeren Restfehlerwahrscheinlichkeit als der günstigere Fehler-korrigierende Code zu charakterisieren. Ähnlich wie bei dem ersten Szenario erreicht der 5-BEC-Code bei einer mittleren Bündelfehlerlänge von $E_b = 1.3$ die maximale Differenz zur Restfehlerwahrscheinlichkeit eines 2-REC-Codes. Für eine mittlere Bündelfehlerlänge $E_b > 1.3$ ist der Vorteil des 5-BEC-Codes wieder leicht rückläufig mit derselben Begründung wie im ersten Szenario. Es treten dann zunehmend Bündelfehler von so großer Länge auf, die von dem 5-BEC-Code nicht mehr korrigiert werden können. Insgesamt bestätigt sich somit das Bild, welches die vorhergehende Abbildung 5.1 bereits geliefert hat. Der einzige Unterschied, der zwischen den beiden Abbildungen besteht, liegt darin begründet, dass durch die variierte Bitfehlerrate $p_E$ von $p_E = 10^{-4}$ auf $p_E = 10^{-10}$ sich der Wertebereich der Restfehlerwahrscheinlichkeiten sowohl für REC- als auch für BEC-Codes verschiebt. Ansonsten bestätigt sich, dass der Einsatz eines BEC-Codes ab einer Grenze, die im Bereich von $E_b \approx 1.2$ liegt, dem eines REC-Codes vorzuziehen ist.

Ähnliche Aussagen gelten auch für das dritte Szenario, dessen Ergebnisse in Abbildung 5.4 zusammengestellt sind. Dort präsentieren wir einmal mehr die Restfehlerwahrscheinlichkeiten nach der Fehlerkorrektur einer gewissen Anzahl *random errors* bzw. nach der Korrektur von *burst errors* bis zu einer gewissen Länge abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ und für eine feste Bitfehlerrate $p_E = 10^{-4}$. Nach den Vorgaben des dritten Szenarios wird die Codelänge $n$ variiert und auf $n = 105$ gesetzt. In Tabelle 5.1 haben wir auch für diese Codelänge die Existenz von REC- und BEC-Codes mit annähernd gleich großer Coderate $k/n$ nachgewiesen. So existieren nach den Angaben in der Tabelle beispielsweise ein 3-REC-Code sowie ein 10-BEC-Code, deren Restfehlerwahrscheinlichkeiten wir nun in Abbildung 5.4 bewerten und vergleichen. Dazu stellen die durchgezogenen Kurven wiederum die Restfehlerwahrscheinlichkeiten der REC-Codes, die 2, 3 bzw. 3 *random errors* korrigieren können, dar und die gestrichelten Kurven repräsentieren die Restfehlerwahrscheinlichkeiten für die BEC nach der Korrektur aller Bündelfehler bis zur Länge 4, 6, 8 bzw. 10. Für eine bessere Übersichtlichkeit der Grafik beschränken wir uns in Abbildung 5.4 bezüglich der BEC auf die Angabe der Restfehlerwahrscheinlichkeiten für die BEC-Codes, die alle Bündelfehler bis zur Länge 4, 6, 8 bzw. 10 korrigieren.

Für einen direkten Vergleich der Leistungsfähigkeit von REC- gegenüber BEC-Codes greifen wir im Rahmen dieses Szenarios die Restfehlerwahrscheinlichkeit eines 3-REC-Codes und diejenige eines 10-BEC-Codes heraus und stellen diese einander gegenüber. Für eine mittlere Bündelfehlerlänge von $E_b = 1$, also für einen symmetrischen Binärkanal, ist der Einsatz eines 3-REC-Codes zur Fehlerkorrektur offensichtlich dem eines 10-BEC-Codes vorzuziehen, da die Restfehlerwahrscheinlichkeit für einen 3-REC-Code um den Faktor $10^{-5}$ geringer ist als die Restfehlerwahrscheinlichkeit eines 9-BEC-Codes. Dies bestätigt genau die Entwicklung in den vorherigen Abbildungen, wo für eine mittlere Bündelfehlerlänge von $E_b = 1$ ebenfalls REC-Codes zur Fehlerkorrektur wesentlich besser geeignet sind. Falls jedoch die mittlere Bündelfehlerlänge $E_b$ leicht ansteigt und bereits wenige Bündelfehler während der Übertragung auftreten, gewinnen BEC-Codes zunehmend an Effizienz, während gleichzeitig

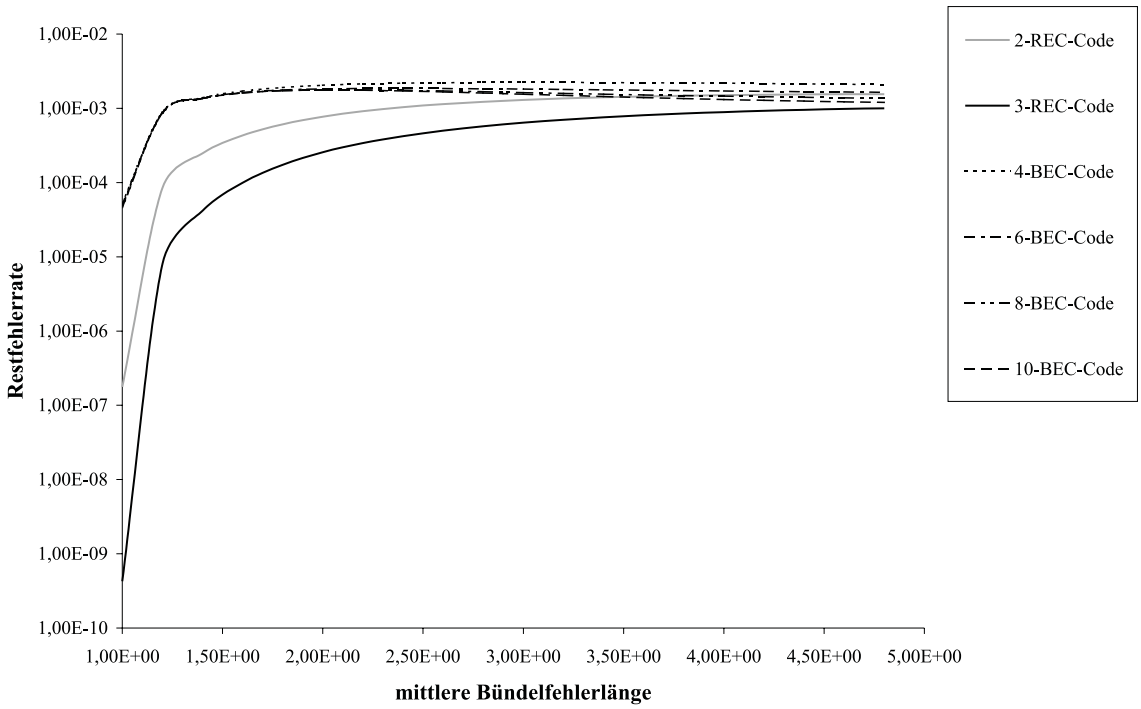


Abbildung 5.4: Vergleich der Restfehlerwahrscheinlichkeiten REC- und BEC-Codes: Codelänge $n = 105$, Bitfehlerrate $p_E = 10^{-4}$

REC-Codes erheblich an Effizienz verlieren, was sich in einer rapide wachsenden Restfehlerwahrscheinlichkeit niederschlägt. Allerdings bleibt die Restfehlerwahrscheinlichkeit eines 3-REC-Codes stets geringer als diejenige eines 10-BEC-Codes. Von daher bestätigt Abbildung 5.4 im Prinzip den Verlauf bzw. das Verhalten der Restfehlerwahrscheinlichkeiten bei variierender mittlerer Bündelfehlerlänge $E_b$ aus den vorangehenden Abbildungen 5.1 und 5.3 mit dem einzigen Unterschied, dass der Wertebereich der Restfehlerwahrscheinlichkeiten aufgrund der abgeänderten Codelänge $n$ im Vergleich zu Abbildung 5.1 verschoben ist. Wegen der vergrößerten Codelänge $n$ ergeben sich weitreichendere Möglichkeiten zur Fehlerkorrektur, so dass die Restfehlerwahrscheinlichkeiten sowohl für REC- als auch für BEC-Codes geringere Werte annehmen. Insgesamt gilt auch hier, dass der Einsatz eines BEC-Codes vor dem eines REC-Codes zu bevorzugen ist, falls die mittlere Bündelfehlerlänge $E_b$ eine bestimmte Grenze im Bereich von 1.3 überschreitet und somit bereits eine geringe Anzahl von Bündelfehlern während der Übertragung auftritt.

In den Abbildungen 5.1 bis 5.4 sind die Restfehlerwahrscheinlichkeiten sowohl für REC- als auch für BEC-Codes abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ dargestellt, ohne dass zur Berechnung der entsprechenden Daten spezielle Codes bzw. konkrete Generatorpolynome realisiert worden sind. Dabei ergibt sich für alle drei Szenarien ein ähnlicher Kurvenverlauf der Restfehlerwahrscheinlichkeiten für REC- und BEC-Codes. Für einen symmetrischen Binärkanal mit $E_b = 1$ sind REC-Codes stets besser zur Fehlerkorrektur geeignet. Überschreitet die mittlere Bündelfehlerlänge $E_b$ allerdings eine gewisse Grenze, die in einem Interval von $[1.1, 1.2]$ liegt, dann ist der Einsatz eines BEC-Codes gegenüber dem Einsatz eines REC-Codes zu favorisieren. Anhand dieser Grafiken können wir somit ganz allgemein abhängig von der mittleren Bündelfehlerlänge $E_b$ einen optimalen Blockcode zu Fehlerkorrektur bestimmen.

### 5.2.2 Poisson Verteilung

Im vorangegangenen Abschnitt 5.2.1 haben wir REC- gegenüber BEC-Codes bewertet und ihre Leistungsfähigkeit und Effizienz verglichen, indem wir die Restfehlerwahrscheinlichkeiten für REC- und BEC-Codes abhängig von der mittleren Bündelfehlerlänge $E_b$ beispielhaft basierend auf einem Kanalmodell mit nur zwei Zuständen berechnet haben. Dazu sind wir der Einfachheit halber von der geometrischen Verteilung als Verteilung der Aufenthaltszeiten ausgegangen. Anhand dieses einfachen Beispiels der geometrischen Verteilung haben wir in Abschnitt 5.2.1 eine übersichtliche Darstellung der Beziehung zwischen den Kanalparametern und der Restfehlerwahrscheinlichkeit präsentieren können und darüber hinaus das zugrunde liegende Kanalmodell auf zwei Zustände reduzieren können.

Nun ermöglicht das von uns in Abschnitt 3.4 entwickelte Kanalmodell die Betrachtung endlich vieler Zustände, denn es basiert auf einem Markov'schen Hintergrundprozeß mit einem endlichen Zustandsraum $\mathcal{S} = \{1, \ldots, N\}$. Ferner haben wir in diesem Zusammenhang in Abschnitt 3.4 erläutert, inwieweit das vorgestellte Kanalmodell auch semi-Markov Prozesse bzw. semi-Markov Modelle mit endlichem Zustandsraum umfaßt, und darauf verwiesen, dass es hierzu effiziente Anpassungsschemata für die Angleichung an korrelierte stochastische Prozesse gibt, vgl. dazu [16]. Da darüber hinaus die in Abschnitt 4.2.2 hergeleiteten rekursiven Gleichungen zur Berechnung der Restfehlerwahrscheinlichkeit von REC- bzw. BEC-Codes auf einem Kanalmodell mit $N$ Zuständen beruhen, beinhaltet eine Analyse der Restfehlerwahrscheinlichkeiten unter Zuhilfenahme dieser rekursiven Gleichungen auch eine Untersuchung der Restfehlerwahrscheinlichkeiten auf der Basis beliebiger semi-Markov Modelle. Von daher können wir die Zustände des Markov'schen Hintergrundprozesses bzw. des semi-Markov Prozesses mit beliebigen Verteilungen in Bezug auf ihre Aufenthaltszeiten versehen und auf diese Art auch komplexere Verteilungen der Aufenthaltszeiten mit dem in Abschnitt 3.4 entwickelten Kanalmodell umsetzen. In diesem Abschnitt betrachten wir daher die Dauer bzw. die Länge der Bündelfehler beispielhaft als Poisson-verteilt, im Gegensatz zum vorherigen Abschnitt, wo wir eine geometrische Verteilung angenommen haben.

Im Rahmen dieses Abschnittes setzen wir nun voraus, dass die Länge der Bündelfehler Poisson-verteilt ist. Wir skizzieren im folgenden kurz, welche Auswirkungen dies auf das zugrunde liegende Kanalmodell bzw. den Markov'schen Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ besitzt. Dazu setzen wir zunächst wiederum bei einem Kanalmodell mit einem guten und einem schlechten Übertragungszustand $B$ an. Wenn der Markov'sche Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ nun den schlechten Übertragungszustand $B$ erreicht, in dem Übertragungsfehler generiert werden, dann bedingt die Poisson-Verteilung der Bündelfehlerlänge, dass der Prozeß $(X_t)_{t \in \boldsymbol{N}}$ in diesem Zustand $B$ während der nachfolgenden $i$ Zustandsübergänge mit der Wahrscheinlichkeit

$$\pi(i) := \frac{m^i \, e^{-m}}{i!}$$

verweilt und den Zustand $B$ dann anschließend verläßt.

Dieses Verhalten können wir allerdings auch folgendermaßen interpretieren. Wir splitten dazu den schlechten Übertragungszustand $B$ in $M+1$ schlechte Übertragungszustände $B_0, B_1, \ldots, B_M$ auf, in denen dann während der Übertragung Fehler auftreten werden können. Dabei signalisiert ein Übertragungszustand $B_i$ mit $0 \leq i \leq M$ in diesem Zusammenhang ferner, dass der Markov'sche Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ im schlechten

Übertragungszustand $B$ für weitere $i$ Zustandsübergänge verbleibt. Der Parameter $M$, mit $M \in \mathbb{N}$, ist hierbei groß genug zu wählen, so dass längere Aufenthaltszeiten im schlechten Übertragungszustand $B$ vernachlässigbar werden; d.h. wir kappen die Poisson-Verteilung für unsere Berechnungen und Analysen entsprechend, so dass gilt

$$\sum_{i>M} \pi(i) < 10^{-10}.$$

Folglich erhalten wir hieraus einen Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ mit einem endlichen Zustandsraum $\mathcal{S}$ bestehend aus insgesamt $M+2$ Zuständen, d.h. mit

$$\mathcal{S} = \{G, B_0, B_1, \ldots, B_M\}.$$

Dabei wird dieser Markov'sche Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ gekennzeichnet durch eine Übergangsmatrix $\mathbf{Q} = (q_{ij})_{i,j\in\mathcal{S}}$, die die folgenden $2M+3$ von Null verschiedenen Übergangswahrscheinlichkeiten enthält:

$$\begin{aligned} q_{GG} &= 1 - \sum_{i=0}^{M} \pi(i) \\ q_{GB_i} &= \pi(i) && \text{für } 0 \le i \le M \\ q_{B_{i+1}B_i} &= 1 && \text{für } 0 \le i \le M-1 \\ \text{sowie} \quad q_{GB_0} &= 1 && . \end{aligned}$$

In Abbildung 5.5 ist das Kanalmodell, welches aus dieser Interpretation resultiert, mit den $M+2$ Zuständen $G, B_0, \ldots B_M$ und den entsprechenden $2M+3$ Übergangswahrscheinlichkeiten anschaulich dargestellt.

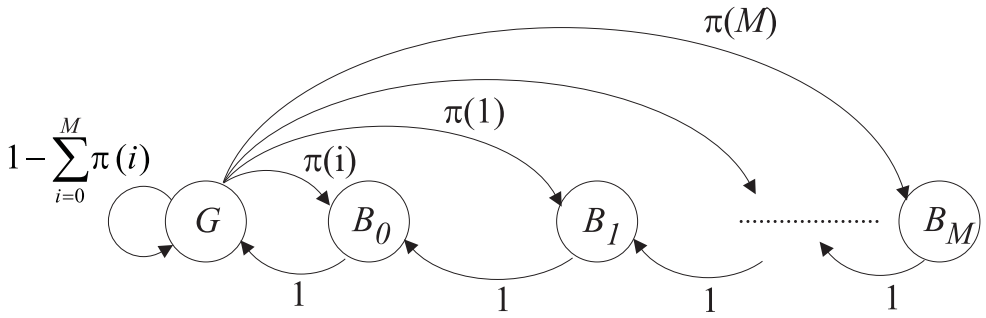


Abbildung 5.5: Kanalmodell bei einer Poisson-verteilten Bündelfehlerlänge $E_b$

Nun gehen wir in diesem Abschnitt davon aus, dass die Dauer bzw. die Länge der Bündelfehler Poisson-verteilt ist. In Bezug auf die mittlere Bündelfehlerlänge $E_b$, die gerade als der Erwartungswert der Verteilung der Bündelfehlerlängen, also als der Erwartungswert der Poisson-Verteilung, charakterisiert werden kann, bedeutet dies, dass der Parameter der mittleren Bündelfehlerlänge $E_b$ in diesem Fall gegeben ist durch

$$E_b = m + 1,$$

wobei der Parameter $m$ gegeben ist durch die Poisson-Verteilung $\pi(i) := \frac{m^i e^{-m}}{i!}$.

Nach den obigen Ausführungen sind die rekursiven Gleichungen aus Abschnitt 4.2.2 zur Berechnung der Restfehlerwahrscheinlichkeiten auf der Basis des in Abbildung 5.5 anschaulich dargestellten Kanalmodells auszuwerten. Dies impliziert, dass die Restfehlerwahrscheinlichkeit eines REC-Codes genauso wie diejenige eines BEC-Codes bzw. die Rekursionen aus Abschnitt 4.2.2 ebenso wie im vorhergehenden Abschnitt 5.2.1, von den beiden Kanalparametern, der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_b$, sowie der Codelänge $n$ abhängen. Der einzige Unterschied besteht darin, dass die mittlere Bündelfehlerlänge als der Erwartungswert der Poisson-verteilten Bündelfehlerlängen in diesem Abschnitt gegeben ist durch $E_b = m + 1$. Ferner sind die rekursiven Gleichungen bezüglich der REC natürlich auch von dem Parameter $r$ der REC-Korrekturfähigkeit abhängig und die Rekursionen bezüglich der BEC von dem Parameter $b$ der BEC-Korrekturfähigkeit. Ziel ist es, wie schon in Abschnitt 5.2.1, die Restfehlerwahrscheinlichkeiten nun jeweils für eine variierende mittlere Bündelfehlerlänge $E_b$ zu berechnen, um dann anhand der erzielten Resultate entscheiden zu können, ob der Einsatz eines REC-Codes dem eines BEC-Codes vorzuziehen ist, und um auf diese Art einen optimalen Blockcode zur Fehlerkorrektur auswählen zu können. Dazu sind wie im vorangehenden Abschnitt die beiden übrigen Parameter, die Codelänge $n$ sowie die Bitfehlerrate $p_E$, fest vorzugeben. Auch in diesem Abschnitt realisieren wir hierzu die in Abschnitt 5.2.1 skizzierten Szenarien. So legen wir für das erste und zweite Szenario den Parameter für die Codelänge $n$ in Anlehnung an Tabelle 5.1 auf $n = 63$ fest. Gemäß den Anforderungen für eine zuverlässige Sprachübertragung setzen wir im ersten Szenario die Bitfehlerrate $p_E$ auf $p_E = 10^{-4}$. Für das zweite Szenario variieren wir lediglich den Parameter für die Bitfehlerrate $p_E$. Im Hinblick auf eine sichere Datenübertragung beträgt die Bitfehlerrate hier $p_E = 10^{-10}$.

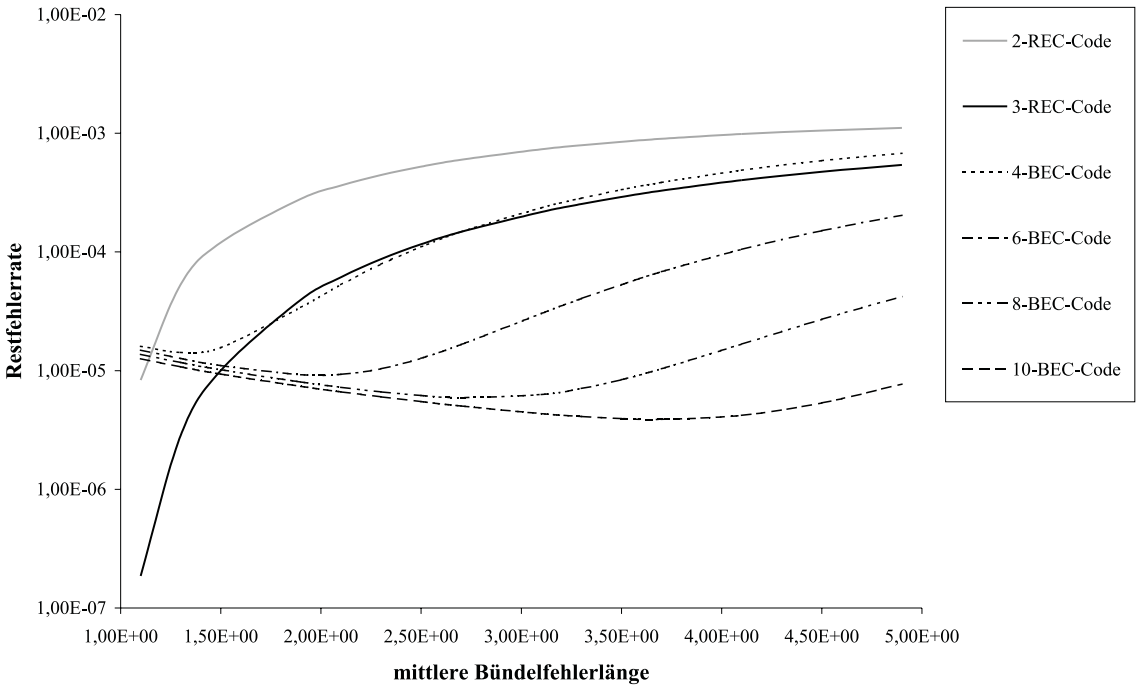


Abbildung 5.6: Vergleich der Restfehlerwahrscheinlichkeiten REC-und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 10^{-4}$

In beiden folgenden Grafiken sind die Restfehlerwahrscheinlichkeiten für REC- und BEC-

Codes abhängig von einer variierenden mittleren Bündelfehlerlänge $E_b$ nach der Fehlerkorrektur dargestellt. Dabei kennzeichnen die durchgezogenen Kurven wiederum die Restfehlerwahrscheinlichkeiten für REC-Codes, die 2 bzw. 3 *random errors* korrigieren. Die gestrichelten Kurven dagegen repräsentieren die Restfehlerwahrscheinlichkeiten nach der Korrektur aller Bündelfehler bis zur Länge 4, 6, 8 bzw. 10.

In Abbildung 5.6 werden die Ergebnisse für das erste Szenario präsentiert. Stellen wir zum direkten Leistungsvergleich die Restfehlerwahrscheinlichkeiten eines 2-REC-Codes und eines 6-BEC-Codes einander gegenüber, so zeigt sich, dass sich die beiden Kurven bei einer mittleren Bündelfehlerlänge von $E_b \approx 1.15$ schneiden. Für Werte von $E_b$, die sehr nahe bei 1 liegen, ist der Einsatz eines 2-REC-Codes zur Fehlerkorrektur nach wie vor vorzuziehen. Für den gesamten Bereich $E_b > 1.15$ ist die Restfehlerwahrscheinlichkeit eines 6-BEC-Codes stets geringer als diejenige eines 2-REC-Codes. Dabei wird die maximale Differenz bei einer mittleren Bündelfehlerlänge von $E_b \approx 2$ erreicht.
Die Resultate für das zweite Szenario werden in Abbildung 5.7 grafisch dargestellt. Hier fällt der Vorteil der BEC noch wesentlich größer aus als im ersten Szenario. So ist die Restfehlerwahrscheinlichkeit eines 6-BEC-Codes sogar für eine mittlere Bündelfehlerlänge $E_b$, die fast gleich 1 ist, wesentlich geringer als diejenige eines 2-REC-Codes.

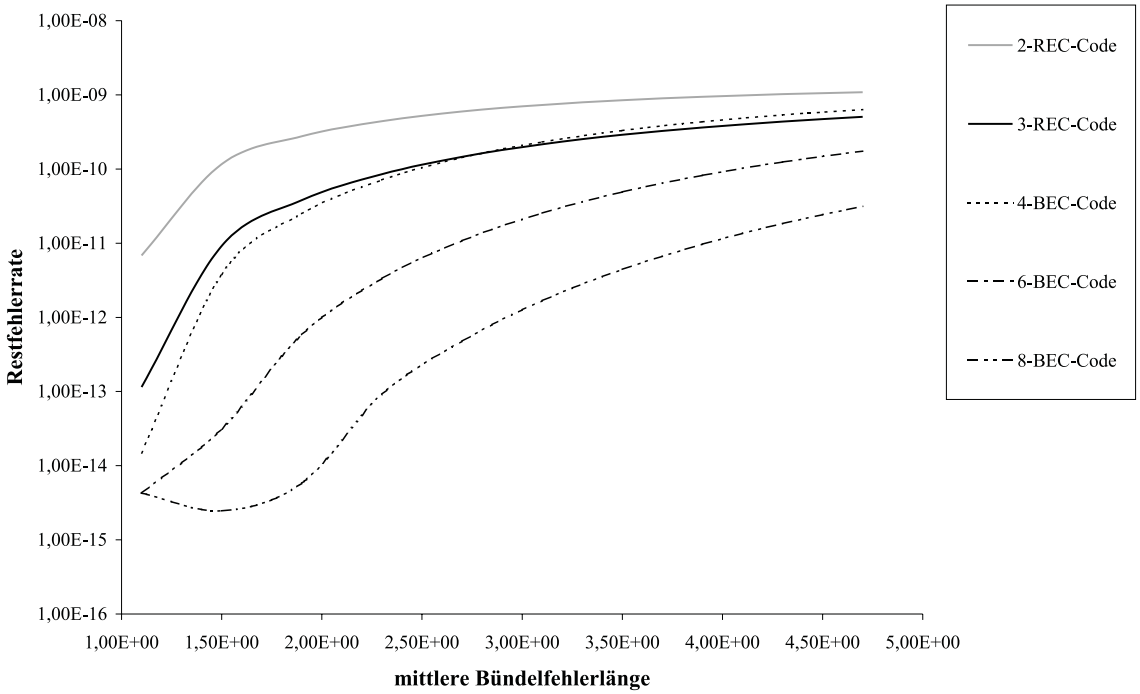


Abbildung 5.7: Vergleich der Restfehlerwahrscheinlichkeiten REC- und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 10^{-10}$

Insgesamt halten wir fest, dass wir mit den Abbildungen 5.6 und 5.7 die Ergebnisse aus Abschnitt 5.2.1 bestätigen können. Sehr deutlich wird in diesem Zusammenhang allerdings, dass die Poisson Verteilung der Bündelfehlerlänge den Vorteil der BEC noch verstärkt. Aufgrund der Poisson Verteilung der Bündelfehlerlänge sind die aufgetretenen Übertragungsfehler noch wesentlich stärker in Bündelfehlern angeordnet als bei einer geometrischen Verteilung. Demzufolge ist der Einsatz eines BEC-Code gegenüber dem eines REC-Codes deutlich zu bevorzugen.

# Kapitel 6

# Approximationsformeln zur Berechnung der Restfehlerwahrscheinlichkeit

Ein Ziel dieser Arbeit besteht darin, abhängig von den Charakteristika eines digitalen Übertragungskanals einen optimalen Code zur Fehlerkorrektur auszuwählen. In diesem Zusammenhang verstehen wir unter einem optimalen Code einen Blockcode mit möglichst großer Coderate $k/n$, um auf diese Weise möglichst viel Information übertragen zu können. Des weiteren besitze dieser Code eine möglichst kleine Codelänge $n$ für schnelles, einfaches und effizientes Decodieren sowie eine möglichst große Fehlerkorrekturfähigkeit für ein hohes Maß an Informationssicherheit. Um die Effektivität von Codes mit unterschiedlichen Fehlerkorrektureigenschaften beurteilen zu können, haben wir ihre Leistungsfähigkeit in Abhängigkeit von der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_B$ bestimmt. Zur Bewertung, inwieweit ein Code zur Fehlerkorrektur geeignet ist, haben wir die Restfehlerwahrscheinlichkeit dieses Codes als Maß für seine Effizienz berechnet. Diese Berechnung ist auf der Grundlage eines Kanalmodells erfolgt, das den durch zufällige Störungen verursachten Fehlerprozeß beschreibt. Das einfachste, in diesem Zusammenhang verwendete, Kanalmodell ist das Gilbert-Elliot Modell. Dieses kann zu einem endlichen Markov Modell mit $N$ Zuständen verallgemeinert werden. In der Arbeit ist ein rekursives Schema vorgestellt worden, das auf einem endlichen Markov Modell mit $N$ Zuständen beruht und mit dem sich die Restfehlerwahrscheinlichkeit eines Codes exakt auswerten läßt. Auf der Basis eines solchen endlichen Markov Modells liegt der Aufwand für eine exakte Bestimmung der Restfehlerwahrscheinlichkeit für einen BEC-Code in der Größenordnung von $\mathcal{O}(nN^2)$. Im Vergleich dazu ist der Aufwand für einen $r$-REC Code in der Größenordnung von $\mathcal{O}(nN(N+r))$ anzusiedeln. Der hohe Aufwand für eine vollständige Auswertung der rekursiven Gleichungen legt daher die Entwicklung von Approximationsformeln zur Berechnung der Restfehlerwahrscheinlichkeiten nahe. Zudem ist eine exakte Bestimmung der Restfehlerwahrscheinlichkeit nicht notwendig, um zu entscheiden, ob der Einsatz eines BEC-Codes dem eines REC-Codes vorzuziehen ist.

In diesem Kapitel beschäftigen wir uns mit der Herleitung entsprechender Näherungsformeln zur Berechnung der Restfehlerwahrscheinlichkeit. Dabei differenzieren wir zwischen Näherungsformeln für REC-Codes und solchen für BEC-Codes. Zunächst analysieren wir, durch welche Faktoren die Restfehlerwahrscheinlichkeit eines Codes überhaupt beeinflußt

wird. In den Abschnitten 6.1.1 und 6.1.2 stellen wir den Einfluß dieser Faktoren in geschlossenen Formeln dar. Abschließend nutzen wir diese geschlossenen Darstellungen, um Näherungsformeln für die Restfehlerwahrscheinlichkeit von REC-Codes bzw. BEC-Codes herzuleiten, vgl. Abschnitt 6.2.

Zu Beginn klären wir noch die Voraussetzungen und Grundlagen dieses Kapitels. Als Basis für die Herleitung der Näherungsformeln zur Berechnung der Restfehlerwahrscheinlichkeit wählen wir der Einfachheit halber ein Kanalmodell mit nur zwei Zuständen, $G$ und $B$. Zur Beschreibung und Charakterisierung des Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ verwenden wir daher in diesem Kapitel das Modell von Gilbert. Zur Erinnerung seien hier noch einmal kurz die Grundlagen des Modells von Gilbert angegeben.

(i) Die Übergangswahrscheinlichkeiten der Übergänge des Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ zwischen den beiden Zuständen $G$ und $B$ bezeichnen wir wie in Abschnitt 3.3.2 mit

$$q_{GG}, q_{GB}, q_{BG} \quad \text{und} \quad q_{BB}.$$

(ii) Die stationären Zustandswahrscheinlichkeiten $\pi_G$ und $\pi_B$ sind ebenfalls wie in Abschnitt 3.3.2 gegeben durch:

$$\pi_G = \frac{q_{BG}}{q_{GB} + q_{BG}} \quad \text{und} \quad \pi_B = \frac{q_{GB}}{q_{GB} + q_{BG}}.$$

(iii) Die Zustände des Hintergrundprozesses $(X_t)_{t\in\boldsymbol{N}}$ werden wiederum mit der Wahrscheinlichkeit identifiziert, mit welcher in diesem Zustand ein Fehler auftritt. Somit stellen sie zwei verschiedene Fehlerwahrscheinlichkeiten $e_G$ und $e_B$ dar:

$$e_G = P(E_t = 1 \,|\, X_t = G) \quad \text{und} \quad e_B = P(E_t = 1 \,|\, X_t = B).$$

Gemäß des Modells von Gilbert setzen wir die Fehlerwahrscheinlichkeiten

$$e_G = 0 \quad \text{und} \quad e_B = 1/2, \text{ vgl. Paragraph 3.3.2.}$$

Diese Gegebenheiten des Modells von Gilbert liefern, dass sich der Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ während der Übertragung jedes einzelnen Bits eines Wortes der Länge $n$ entweder im Zustand $G$ oder im Zustand $B$ befindet. Demzufolge produziert der Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ während der Übermittlung eines Wortes der Länge $n$ ein Muster $M_n$ der Länge $n$, für das gilt

$$M_n = (m_1, \ldots, m_n) \quad \text{mit} \quad m_j \in \{G, B\} \quad \text{für} \quad 1 \leq j \leq n.$$

Dies führt zu den folgenden grundlegenden Definitionen dieses Kapitels, die wir zur Herleitung der Näherungsformeln benötigen.

**6.0.2 Definition**
*Mit $\mathcal{M}$ bezeichnen wir die Menge aller Muster $M_n = (m_1, \ldots, m_n)$ der Länge $n$, die der Hintergrundprozeß $(X_t)_{t\in N}$ während der Übertragung eines Wortes der Länge $n$ erzeugen kann; d.h.*

$$\mathcal{M} := \{M_n = (m_1, \ldots, m_n)\,; X_j = m_j \text{ mit } m_j \in \{G, B\}, 1 \leq j \leq n\}.$$

**6.0.3 Definition**

(i) *Als bad-Phase bezeichnen wir das Ereignis, dass sich der Hintergrundprozeß $(X_t)_{t\in N}$ während der Übertragung eines Wortes der Länge $n$ an $x$ aufeinander folgenden Zeitpunkten $t$ ausschließlich im Zustand $B$ befindet.*
*Für das von $(X_t)_{t\in N}$ erzeugte Muster $M_n \in \mathcal{M}$ gilt:*

$$\begin{aligned} M_n = (m_1, \ldots, m_n) \quad & \text{mit} \quad m_j = m_{j+1} = \cdots = m_{j+x-1} = B \\ & \text{und} \quad m_{j-1} = m_{j+x} = G \\ & \text{für} \quad 1 \leq j \leq n - x + 1 \text{ beliebig.} \end{aligned}$$

(ii) *Die Länge $l$ einer bad-Phase definieren wir als die Anzahl $x$ der aufeinander folgenden Komponenten $m_j$ von $M_n \in \mathcal{M}$ mit $m_j = B$, also $l := x$.*

(iii) *Ganz analog sind auch die Begriffe good-Phase und Länge einer good-Phase definiert.*

**6.0.4 Beispiel**
Angenommen, der Hintergrundprozeß $(X_t)_{t\in N}$ hat bei der Übertragung eines Wortes der Länge $n$ das folgende Muster $M_n \in \mathcal{M}$ erzeugt,

$$M_n = (\text{ G G } \ldots \text{ G } \underline{\text{B B B B B}} \text{ G } \ldots \text{ G } \underline{\text{B B B}} \text{ G G } \ldots \text{ G G }),$$

dann stellen die beiden unterstrichenen Bereiche jeweils eine *bad*-Phase der Länge 5 bzw. eine *bad*-Phase der Länge 3 dar.

**6.0.5 Bemerkung**

(i) Aufgrund der Vorgaben des Modells von Gilbert, insbesondere wegen $e_G = 0$, entstehen im Zustand $G$, also während einer *good*-Phase, überhaupt keine Übertragungsfehler.

(ii) Wegen $e_B = 1/2$ treten Fehler mit Wahrscheinlichkeit $1/2$ ausschließlich dann auf, wenn sich der Hintergrundprozeß $(X_t)_{t\in N}$ im Zustand $B$ befindet, also nur innerhalb einer *bad*-Phase.

(iii) Hierbei können die in einer *bad*-Phase aufgetretenen Übertragungsfehler sowohl korreliert als auch zufällig auf beliebige Positionen verteilt sein. Auf welche Art und Weise die Fehler vorwiegend verteilt sind, hängt dabei entscheidend vom Wert der durchschnittlichen Bündelfehlerlänge $E_B$ ab, einem der wesentlichen Parameter, die den Übertragungskanal charakterisieren, vgl. dazu die Ausführungen in Abschnitt 4.2 und Kapitel 5.

**6.0.6 Bemerkung**
Zusammenfassend halten wir fest:
Aufgrund dieser Voraussetzungen impliziert der Modellansatz nach Gilbert, dass Übertragungsfehler mit Wahrscheinlichkeit 1/2 ausschließlich innerhalb von *bad*-Phasen entstehen. Folglich wird die Restfehlerwahrscheinlichkeit eines Codes im wesentlichen durch zwei Faktoren beeinflußt:

(i) durch die Wahrscheinlichkeit, dass der Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ während der Übertragung eines Wortes der Länge $n$ eine oder mehrere *bad*-Phasen in einem Muster $M_n \in \mathcal{M}$ erzeugt, und

(ii) durch die Wahrscheinlichkeit, dass innerhalb der aufgetretenen *bad*-Phasen Fehler der Gestalt vorhanden sind, die mittels des verwendeten Codes nicht bzw. nicht korrekt korrigiert werden können.

In den Abschnitten 6.1.1 und 6.1.2 werden wir diese beiden Faktoren genauer analysieren. Wir entwickeln zunächst eine allgemeine Formel für die Wahrscheinlichkeit, dass der Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ während der Übertragung eines Wortes der Länge $n$ in einem Muster $M_n \in \mathcal{M}$ eine bestimmte Anzahl von *bad*-Phasen erzeugt. Anschließend leiten wir in Abschnitt 6.1.2 einen geschlossenen Ausdruck für die Wahrscheinlichkeit her, dass Fehler, die in den vorhandenen *bad*-Phasen aufgetreten sind, vom eingesetzten Code nicht bzw. nicht korrekt korrigiert werden können. Mittels der Ergebnisse aus den Abschnitten 6.1.1 und 6.1.2 leiten wir im letzten Paragraphen dieses Kapitels Approximationsformeln her, mit denen wir die Restfehlerwahrscheinlichkeit für REC-Codes bzw. für BEC-Codes näherungsweise bestimmen können.

## 6.1 Vorbereitungen

Im ersten Teil dieses Abschnittes beschäftigen wir uns mit einer allgemeinen Darstellung der Wahrscheinlichkeit, dass der Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ während der Übertragung eines Wortes der Länge $n$ eine gewisse Anzahl von *bad*-Phasen in einem Muster $M_n \in \mathcal{M}$ erzeugt. Hierzu leiten wir in Abschnitt 6.1.1 eine geschlossene Formel her. In Abschnitt 6.1.2 geben wir schließlich eine Darstellung für die Wahrscheinlichkeit an, dass innerhalb der aufgetretenen *bad*-Phase Übertragungsfehlern derart generiert werden, dass diese nicht mehr vom verwendeten Fehler-korrigierenden Code korrigiert werden können.

### 6.1.1 Wahrscheinlichkeit für *bad*-Phasen

Zur Herleitung einer allgemeinen Formel für die Wahrscheinlichkeit, dass während der Übertragung eines Wortes der Länge $n$ in einem Muster $M_n \in \mathcal{M}$ eine gewisse Anzahl von *bad*-Phasen vom Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugt wird, benötigen wir die folgenden grundlegenden Begriffe und Bezeichnungen.

### 6.1.1 Bezeichnungen

*Die diskrete Zufallsvariable $Z_{anz}$ gebe die Anzahl der bad-Phasen an, die in einem vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugten Muster $M_n \in \mathcal{M}$ enthalten sind.*

### 6.1.2 Bemerkung

Da in einem Muster $M_n \in \mathcal{M}$ der Länge $n$ maximal $\lceil \frac{n}{2} \rceil$ *bad*-Phasen auftreten können, ist der Wertebereich der diskreten Zufallsvariable $Z_{anz}$ gegeben durch:

$$Z_{anz} \in \left\{0, 1, \ldots, \left\lceil \frac{n}{2} \right\rceil \right\}.$$

### 6.1.3 Definition

*Mit $\mathcal{M}(i)$ bezeichnen wir die Menge aller Muster $M_n \in \mathcal{M}$, für die $Z_{anz} = i$ gilt, die also genau $Z_{anz} = i$ bad-Phasen enthalten; d.h.*

$$\mathcal{M}(i) := \{M_n \in \mathcal{M}; M_n \textit{ enthält genau } Z_{anz} = i \textit{ bad-Phasen}\}$$

### 6.1.4 Bemerkung

Mit diesen Bezeichnungen gilt offensichtlich

$$\mathcal{M} = \bigcup_{i=0}^{\lceil \frac{n}{2} \rceil} \mathcal{M}(i).$$

### 6.1.5 Definition

*(i) $P_{M_n}$ bezeichne für ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugtes aber festes Muster $M_n \in \mathcal{M}$ die Wahrscheinlichkeit dieses Musters $M_n$.*

*(ii) $P_{M_n}(Z_{anz} = i)$ bezeichne für ein beliebiges aber fest vorgegebenes $i$ mit $0 \leq i \leq \lceil \frac{n}{2} \rceil$ die Wahrscheinlichkeit, dass in einem vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugten Muster $M_n \in \mathcal{M}$ genau $Z_{anz} = i$ bad-Phasen enthalten sind, dass also $M_n \in \mathcal{M}(i)$ gilt.*

Ziel dieses Abschnittes ist es demnach, eine geschlossene Formel für die Wahrscheinlichkeit

$$P_{M_n}(Z_{anz} = i),$$

zu ermitteln, wobei $0 \leq i \leq \lceil \frac{n}{2} \rceil$ gilt.

### 6.1.6 Bemerkung

Für die Wahrscheinlichkeit $P_{M_n}$ eines festen aber beliebigen Musters $M_n \in \mathcal{M}$ kann eine geschlossene Formel angegeben werden. Nach Definition 6.0.2 ist ein vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugtes Muster $M_n \in \mathcal{M}$ festgelegt durch

$$M_n = (m_1, \ldots, m_n) \quad \text{mit} \quad X_j = m_j \text{ und } m_j \in \{G, B\} \quad \text{für } 1 \leq j \leq n.$$

Nach den Vorgaben des Modells von Gilbert sind die Übergangs- und die stationären Wahrscheinlichkeiten gegeben durch:

$$\begin{aligned} q_{GG} &= P(X_{t+1} = G \,|\, X_t = G) \quad \text{und} \quad q_{GB} = P(X_{t+1} = B \,|\, X_t = G), \\ q_{BG} &= P(X_{t+1} = G \,|\, X_t = B) \quad \text{und} \quad q_{BB} = P(X_{t+1} = B \,|\, X_t = B), \\ \pi_G &= \frac{q_{BG}}{q_{GB} + q_{BG}} \quad \text{und} \quad \pi_B = \frac{q_{GB}}{q_{GB} + q_{BG}}. \end{aligned}$$

Folglich ergibt sich eine geschlossene Formeldarstellung der Wahrscheinlichkeit eines festen aber beliebigen Musters $M_n \in \mathcal{M}$,

$$P_{M_n} = P\big((X_1 = m_1), \ldots, (X_n = m_n)\big),$$

aus dem Produkt der stationären Zustandswahrscheinlichkeit, $\pi_G$ bzw. $\pi_B$, und den entsprechenden Übergangswahrscheinlichkeiten

$$P(X_{j+1} = m_{j+1} \,|\, X_j = m_j) \quad \text{mit} \quad m_{j+1},\, m_j \in \{G, B\} \quad \text{für } 1 \leq j \leq n-1.$$

Welche stationäre Wahrscheinlichkeit, $\pi_G$ oder $\pi_B$, dabei in das Produkt eingeht, richtet sich danach, ob $X_1 = m_1 = G$ oder $X_1 = m_1 = B$ gilt.

Ist also die Anzahl der in einem festen aber beliebigen Muster $M_n \in \mathcal{M}$ enthaltenen *bad*-Phasen bekannt und sind auch deren Längen sowie deren Anfangsstellen fest vorgegeben, dann ist die Wahrscheinlichkeit $P_{M_n}$ dieses Musters $M_n$ als Produkt der entsprechenden Übergangswahrscheinlichkeiten und stationären Zustandswahrscheinlichkeiten darstellbar.

### 6.1.7 Bezeichnungen

*Für ein beliebiges vom Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ erzeugtes Muster $M_n \in \mathcal{M}$ sei*

*(i) jede Komponenten $Z_j$, mit $1 \leq j \leq \lceil \frac{n}{2} \rceil$, des Zufallsvektors*

$$\mathbf{Z} = (Z_1, \ldots, Z_{\lceil \frac{n}{2} \rceil})$$

*eine diskrete Zufallsvariable, die die Länge $l_j$ der $j$-ten in $M_n$ enthaltenen bad-Phase angebe, und*

*(ii) jede Komponente $W_j$, mit $1 \leq j \leq \lceil \frac{n}{2} \rceil$, des Zufallsvektors*

$$\mathbf{W} = (W_1, \ldots, W_{\lceil \frac{n}{2} \rceil})$$

*sei eine diskrete Zufallsvariable, die die Anfangsstelle $k_j$ der $j$-ten in $M_n$ enthaltenen bad-Phase bezeichne.*

Die Wertebereiche der diskreten Zufallsvariablen $W_j$ und $Z_j$, für $1 \leq j \leq \lceil \frac{n}{2} \rceil$, unterliegen gewissen Einschränkungen. Diese stellen erstens sicher, dass die in einem beliebigen Muster $M_n \in \mathcal{M}$ enthaltenen *bad*-Phasen durch *good*-Phasen mindestens der Länge 1 voneinander getrennt sind. Zweitens wird mittels dieser Bedingungen gewährleistet, dass die Summe der Längen aller *bad*- und aller *good*-Phasen die Länge $n$ des Musters $M_n$ nicht überschreitet.

**6.1.8 Bemerkung**
Sei ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugtes Muster $M_n \in \mathcal{M}(i)$ gegeben, wobei $0 \leq i \leq \lceil \frac{n}{2} \rceil$ fest aber beliebig ist, dann unterliegen die Wertebereiche der diskreten Zufallsvariablen $W_j$, mit $1 \leq j \leq \lceil \frac{n}{2} \rceil$, des Zufallsvektors $\mathbf{W}$ den folgenden Bedingungen:

(i) Da $M_n \in \mathcal{M}(i)$ ist und damit genau $Z_{anz} = i$ *bad*-Phasen enthält, gilt für die Komponenten $W_j$ mit $i+1 \leq j \leq \lceil \frac{n}{2} \rceil$ von $\mathbf{W}$:

$$W_{i+1}, W_{i+2}, \ldots, W_{\lceil \frac{n}{2} \rceil} \in \{0\} \tag{6.1}$$

(ii) Für die Komponenten $W_j$ mit $1 \leq j \leq i$ von $\mathbf{W}$ gilt dagegen:

$$W_1 \in \{1, \ldots, n - \sum_{j=1}^{i} Z_j - i + 2\}, \tag{6.2}$$

$$W_2 \in \{W_1 + Z_1 + 1, \ldots, n - \sum_{j=2}^{i} Z_j - i + 3\}, \tag{6.3}$$

$$W_3 \in \{W_2 + Z_2 + 1, \ldots, n - \sum_{j=3}^{i} Z_j - i + 4\}, \tag{6.4}$$

$$\vdots$$

$$\text{sowie} \quad W_i \in \{W_{i-1} + Z_{i-1} + 1, \ldots, n - Z_i + 1\} \tag{6.5}$$

Falls ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugtes Muster $M_n \in \mathcal{M}(i)$, mit $0 \leq i \leq \lceil \frac{n}{2} \rceil$ fest aber beliebig, gegeben ist und wir nun die Wertebereiche der diskreten Zufallsvariablen $Z_j$, mit $1 \leq j \leq \lceil \frac{n}{2} \rceil$, des Zufallsvektors $\mathbf{Z}$ angeben, so haben wir dabei bezüglich der Zufallsvariablen $W_1$ und $W_i$ verschiedene Fälle zu unterscheiden, denn die maximal möglichen Längen der aufgetretenen *bad*-Phasen variieren, je nachdem an welcher Stelle die erste *bad*-Phase beginnt bzw. die letzte *bad*-Phase endet. Daher differenzieren wir von nun an bei allen weiteren Betrachtungen zwischen den folgenden vier Fällen:

**6.1.9 Bemerkung**
Sei ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugtes Muster $M_n \in \mathcal{M}(i)$ gegeben, wobei $0 \leq i \leq \lceil \frac{n}{2} \rceil$ fest aber beliebig ist, so müssen die Wertebereiche der diskreten Zufallsvariablen $Z_j$, mit $1 \leq j \leq \lceil \frac{n}{2} \rceil$, des Zufallsvektors $\mathbf{Z}$ die folgenden Forderungen erfüllen:

1. Fall: Die erste *bad*-Phase beginnt am Wortanfang und die letzte *bad*-Phase endet am Wortende.
2. Fall: Die erste *bad*-Phase beginnt am Wortanfang und die letzte *bad*-Phase endet in der Wortmitte.
3. Fall: Die erste *bad*-Phase beginnt in der Wortmitte und die letzte *bad*-Phase endet am Wortende.
4. Fall: Alle *bad*-Phasen beginnen und enden in der Wortmitte.

(i) Da $M_n \in \mathcal{M}(i)$ ist und damit $Z_{anz} = i$ *bad*-Phasen enthält, gilt für die Komponenten $Z_j$ mit $i+1 \leq j \leq \lceil \frac{n}{2} \rceil$ von $\mathbf{Z}$:

$$Z_{i+1}, \ldots, Z_{\lceil \frac{n}{2} \rceil} \in \{0\} \tag{6.6}$$

(ii) Für die Komponenten $Z_j$ mit $1 \leq j \leq i$ von $\mathbf{Z}$ gilt dagegen:

1. Fall: $W_1 \in \{1\}$ und $W_i \in \{n - Z_i + 1\}$,
dann genügen die diskreten Komponenten $Z_j$ von $\mathbf{Z}$ den Bedingungen:

$$Z_1, \ldots, Z_i \in \{1, \ldots, n - 2i + 2\} \tag{6.7}$$

$$\sum_{j=1}^{i} Z_j \leq n - i + 1 \tag{6.8}$$

2. Fall: $W_1 \in \{1\}$ und $W_i \in \{W_{i-1} + Z_{i-1} + 1, \ldots, n - Z_i\}$,
dann ergeben sich für die diskreten Komponenten $Z_j$ von $\mathbf{Z}$ die Einschränkungen:

$$Z_1, \ldots, Z_i \in \{1, \ldots, n - 2i + 1\} \tag{6.9}$$

$$\sum_{j=1}^{i} Z_j \leq n - i \tag{6.10}$$

3. Fall: $W_1 \in \{2, \ldots, n - \sum_{j=1}^{i} Z_j - (i-1)\}$ und $W_i \in \{n - Z_i + 1\}$,
so unterliegen die diskreten Zufallsvariablen $Z_j$ von $\mathbf{Z}$ den Bedingungen:

$$Z_1, \ldots, Z_i \in \{1, \ldots, n - 2i + 1\} \tag{6.11}$$

$$\sum_{j=1}^{i} Z_j \leq n - i \tag{6.12}$$

4. Fall: $W_1 \in \{2, \ldots, n - \sum_{j=1}^{i} Z_j - (i-1)\}$ und $W_i \in \{W_{i-1} + Z_{i-1} + 1, \ldots, n - Z_i\}$, dann erfüllen die diskreten Zufallsvariablen $Z_j$ von $\mathbf{Z}$ die Forderungen:

$$Z_1, \ldots, Z_i \in \{1, \ldots, n - 2i\} \tag{6.13}$$

$$\sum_{j=1}^{i} Z_j \leq n - i - 1 \tag{6.14}$$

**6.1.10 Beispiel**
Ist ein beliebiges vom Hintergrundprozeß $(X_t)_{t \in \boldsymbol{N}}$ erzeugtes Muster $M_n \in \mathcal{M}\left(\left\lceil \frac{n}{2} \right\rceil\right)$ gegeben, so enthält dieses Muster $M_n$ genau $\left\lceil \frac{n}{2} \right\rceil$ *bad*-Phasen, die dann alle die Länge 1 besitzen. In diesem Fall sind die Zufallsvektoren $\mathbf{W}$ und $\mathbf{Z}$ konkret festgelegt durch:

$$\mathbf{W} = (W_1, \ldots, W_{\lceil \frac{n}{2} \rceil}) = \begin{cases} (1, 3, 5, \ldots, n) \text{ ; } n \text{ ungerade} \\ (1, 3, 5, \ldots, n-1) \\ \text{oder } (2, 4, 6, \ldots, n) \end{cases} \text{ ; } n \text{ gerade}$$

$$\text{und} \quad \mathbf{Z} = (Z_1, \ldots, Z_{\lceil \frac{n}{2} \rceil}) = (1, \ldots, 1)$$

**6.1.11 Bemerkung**
Für alle weiteren Ausführungen in diesem Kapitel, die sich auf die Betrachtung eines beliebigen Musters $M_n \in \mathcal{M}(i)$ beziehen, das genau $Z_{anz} = i$ *bad*-Phasen, mit $0 \leq i \leq \left\lceil \frac{n}{2} \right\rceil$, beinhaltet, halten wir an dieser Stelle fest, dass die Längen $l_1, \ldots, l_i$ der in dem Muster $M_n$ erzeugten *bad*-Phasen sowie deren Anfangsstellen $k_1, \ldots, k_i$ stets den Bedingungen aus den Bemerkungen 6.1.9 und 6.1.8 genügen.

**6.1.12 Definition**
*Sei $i$ mit $0 \leq i \leq \left\lceil \frac{n}{2} \right\rceil$ fest aber beliebig vorgegeben, dann definieren wir $\mathcal{M}_i(l_1, \ldots, l_i)$ als die Menge aller Muster $M_n \in \mathcal{M}(i)$, deren $Z_{anz} = i$ bad-Phasen genau die Längen $l_1, \ldots, l_i$ besitzen; d.h.*

$$\mathcal{M}_i(l_1, \ldots, l_i) := \{M_n \in \mathcal{M}(i); M_n \textit{ enthält } Z_{anz} = i \textit{ bad-Phasen der Längen } l_1, \ldots, l_i\}$$

**6.1.13 Bemerkung**
Mit der Bezeichnung aus Definition 6.1.12 folgt für ein festes aber beliebiges $i$, mit $0 \leq i \leq \left\lceil \frac{n}{2} \right\rceil$:

$$\mathcal{M}(i) = \bigcup_{l_1, \ldots, l_i} \mathcal{M}_i(l_1, \ldots, l_i)$$

wobei die Vereinigung über alle nach Bemerkung 6.1.9 möglichen Längen $l_1, \ldots, l_i$ zu bilden ist.

**6.1.14 Definition**
*Seien ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugtes Muster $M_n \in \mathcal{M}$ und ein $i$, mit $0 \le i \le \lceil \frac{n}{2} \rceil$, gegeben, dann bezeichne*

*(i)* $P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i)\big)$ *die Wahrscheinlichkeit, dass in dem vom Hintergrundprozeß erzeugten Muster $M_n$ genau $Z_{anz} = i$ bad-Phasen der Längen $l_1,\ldots,l_i$ auftreten, und*

*(ii)* $P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i), \mathbf{W} = (k_1,\ldots,k_i)\big)$ *die Wahrscheinlichkeit, dass in dem vom Hintergrundprozeß erzeugten Muster $M_n$ genau $Z_{anz} = i$ bad-Phasen der Längen $l_1,\ldots,l_i$ auftreten, die an den Stellen $k_1,\ldots,k_i$ beginnen.*

**6.1.15 Lemma**
*Seien ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugtes Muster $M_n \in \mathcal{M}$ und ein $i$, mit $0 \le i \le \lceil \frac{n}{2} \rceil$, gegeben. Dann läßt sich die Wahrscheinlichkeit, dass in dem vom Hintergrundprozeß erzeugten Muster $M_n \in \mathcal{M}$ genau $Z_{anz} = i$ bad-Phasen der Längen $l_1,\ldots,l_i$ an den Stellen $k_1,\ldots,k_i$ auftreten, in der folgenden geschlossenen Formel angegeben:*

$$P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i), \mathbf{W} = (k_1,\ldots,k_i)\big)$$

$$= \begin{cases} \dfrac{(q_{GB})^i\,(q_{BG})^{i-1}}{q_{GB}+q_{BG}}\,(q_{GG})^{n-\sum_{j=1}^{i} l_j - i + 1}\,(q_{BB})^{\sum_{j=1}^{i} l_j - i} & ; (k_1 = 1 \wedge k_i = n - l_i + 1) \\ \dfrac{(q_{GB})^i\,(q_{BG})^{i}}{q_{GB}+q_{BG}}\,(q_{GG})^{n-\sum_{j=1}^{i} l_j - i}\,(q_{BB})^{\sum_{j=1}^{i} l_j - i} & ; \begin{array}{l}(k_1 > 1 \wedge k_i = n - l_i + 1)\\ \text{oder}\\ (k_1 = 1 \wedge k_i < n - l_i + 1)\end{array} \\ \dfrac{(q_{GB})^i\,(q_{BG})^{i+1}}{q_{GB}+q_{BG}}\,(q_{GG})^{n-\sum_{j=1}^{i} l_j - i - 1}\,(q_{BB})^{\sum_{j=1}^{i} l_j - i} & ; (k_1 > 1 \wedge k_i < n - l_i + 1) \end{cases}$$

**Beweis**
Bei der Einführung der Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i), \mathbf{W} = (k_1,\ldots,k_i)\big)$ in Definition 6.1.14 wird vorausgesetzt, dass sowohl die Anzahl der *bad*-Phasen, deren Längen als auch deren Anfangsstellen bekannt sind. So wird mit den Vorgaben $Z_{anz} = i$, $\mathbf{Z} = (l_1,\ldots,l_i)$ und $\mathbf{W} = (k_1,\ldots,k_i)$ ein Muster $M_n \in \mathcal{M}$ konkret charakterisiert. Daher entspricht die obige Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i), \mathbf{W} = (k_1,\ldots,k_i)\big)$ der Wahrscheinlichkeit $P_{M_n}$ eines konkreten Musters $M_n \in \mathcal{M}$ im Sinne von Bemerkung 6.1.6 (ii):

$$\begin{aligned} P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i), \mathbf{W} = (k_1,\ldots,k_i)\big) &= P_{M_n} \\ &= P\big((X_1 = m_1),\ldots,(X_n = m_n)\big). \end{aligned}$$

Nach Bemerkung 6.1.6 (ii) läßt sich die Wahrscheinlichkeit $P_{M_n}$ eines konkreten Musters $M_n \in \mathcal{M}$ und damit auch die Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1,\ldots,l_i), \mathbf{W} = (k_1,\ldots,k_i)\big)$ aus dem Produkt der entsprechenden Übergangswahrscheinlichkeiten

$$P(X_{j+1} = m_{j+1} \,|\, X_j = m_j) \quad \text{mit} \quad m_{j+1}, m_j \in \{G,B\} \quad \text{für } 1 \le j < n$$

und der entsprechenden stationären Zustandswahrscheinlichkeit $\pi_G$ bzw. $\pi_B$ ermitteln. Folglich können wir eine geschlossene Darstellung der betrachteten Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$ in Form eines solchen Produkts angeben.

Bei der Bildung dieses Produkts ist allerdings zu beachten, dass die Anzahl der Zustandsübergänge $G \longrightarrow B$ bzw. $B \longrightarrow G$ variiert, je nachdem ob die erste *bad*-Phase am Wortanfang oder in der Wortmitte beginnt bzw. ob die letzte *bad*-Phase genau am Wortende oder in der Wortmitte endet; d.h., dass die Anzahl der Übergangswahrscheinlichkeiten $q_{GB}$ bzw. $q_{BG}$ innerhalb des Produkts variiert. Daher differenzieren wir bezüglich der Anfangsstellen der ersten und der letzten *bad*-Phase:

1. Fall: $k_1 = 1$ und $k_i = n - l_i + 1$, dann gilt

$$\begin{aligned} & P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (1, k_2, \ldots, k_{i-1}, n - l_i + 1)\big) \\ = \;& \pi_B \left(q_{BB}\right)^{\sum_{j=1}^{i} l_j - i} \left(q_{BG}\right)^{i-1} \left(q_{GG}\right)^{n - \sum_{j=1}^{i} l_j - i + 1} \left(q_{GB}\right)^{i-1} \end{aligned}$$

2. Fall: $k_1 = 1$ und $k_i < n - l_i + 1$, daher folgt

$$\begin{aligned} & P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (1, k_2, \ldots, k_i)\big) \\ = \;& \pi_B \left(q_{BB}\right)^{\sum_{j=1}^{i} l_j - i} \left(q_{BG}\right)^{i} \left(q_{GG}\right)^{n - \sum_{j=1}^{i} l_j - i} \left(q_{GB}\right)^{i-1} \end{aligned}$$

3. Fall: $k_1 > 1$ und $k_i = n - l_i + 1$, somit gilt

$$\begin{aligned} & P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_{i-1}, n - l_i + 1)\big) \\ = \;& \pi_G \left(q_{GG}\right)^{n - \sum_{j=1}^{i} l_j - i} \left(q_{GB}\right)^{i} \left(q_{BB}\right)^{\sum_{j=1}^{i} l_j - i} \left(q_{BG}\right)^{i-1} \end{aligned}$$

4. Fall: $k_1 > 1$ und $k_i < n - l_i + 1$, es ergibt sich

$$\begin{aligned} & P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big) \\ = \;& \pi_G \left(q_{GG}\right)^{n - \sum_{j=1}^{i} l_j - i - 1} \left(q_{GB}\right)^{i} \left(q_{BB}\right)^{\sum_{j=1}^{i} l_j - i} \left(q_{BG}\right)^{i} \end{aligned}$$

Setzen wir abschließend in die obigen Gleichungen die entsprechenden stationären Zustandswahrscheinlichkeiten

$$\pi_G = \frac{q_{BG}}{q_{GB} + q_{BG}} \qquad \text{bzw.} \qquad \pi_B = \frac{q_{GB}}{q_{GB} + q_{BG}}$$

ein, dann liefern die Fälle 2 und 3 das gleiche Ergebnis und können zusammengefaßt werden, so dass wir die Behauptung erhalten. □

**6.1.16 Satz**
*Seien ein beliebiges vom Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugtes Muster $M_n \in \mathcal{M}$ und ein $i$, mit $0 \leq i \leq \lceil \frac{n}{2} \rceil$, gegeben, dann läßt sich die Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$, dass in dem vom Hintergrundprozeß erzeugten Muster $M_n \in \mathcal{M}$ genau $Z_{anz} = i$ bad-Phasen der Längen $l_1, \ldots, l_i$ auftreten, wie folgt darstellen:*

$$
\begin{aligned}
P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big) &= \binom{n - \sum_{j=1}^{i} l_j - 1}{i-2} \frac{(q_{GB})^i \, (q_{BG})^{i-1}}{q_{GB} + q_{BG}} (q_{GG})^{n - \sum_{j=1}^{i} l_j - i + 1} (q_{BB})^{\sum_{j=1}^{i} l_j - i} \\
&+ 2 \binom{n - \sum_{j=1}^{i} l_j - 1}{i-1} \frac{(q_{GB})^i \, (q_{BG})^{i}}{q_{GB} + q_{BG}} (q_{GG})^{n - \sum_{j=1}^{i} l_j - i} (q_{BB})^{\sum_{j=1}^{i} l_j - i} \\
&+ \binom{n - \sum_{j=1}^{i} l_j - 1}{i} \frac{(q_{GB})^i \, (q_{BG})^{i+1}}{q_{GB} + q_{BG}} (q_{GG})^{n - \sum_{j=1}^{i} l_j - i - 1} (q_{BB})^{\sum_{j=1}^{i} l_j - i}
\end{aligned}
$$

**Beweis**
Bei der Einführung der Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ in Definition 6.1.14 wird vorausgesetzt, dass sowohl die Anzahl der aufgetretenen *bad*-Phasen als auch deren Längen bekannt sind. Wir betrachten nun diejenigen Muster $M_n \in \mathcal{M}$, die genau diese fest vorgegebene Anzahl $Z_{anz} = i$ von *bad*-Phasen mit den fest vorgegebenen Längen $l_1, \ldots, l_i$ besitzen, wobei die Anfangsstellen $k_1, \ldots, k_i$ der $Z_{anz} = i$ *bad*-Phasen unter Berücksichtigung der Bedingungen aus Bemerkung 6.1.8 beliebig sind; d.h. alle $M_n \in \mathcal{M}_i(l_1, \ldots, l_i)$.

Bilden wir die Summe der Wahrscheinlichkeiten $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$ aller Muster $M_n \in \mathcal{M}$ mit $Z_{anz} = i$ *bad*-Phasen der Längen $l_1, \ldots, l_i$ über alle möglichen Anfangsstellen $k_1, \ldots, k_i$, so liefern kombinatorische Methoden, dass diese Summe die zu ermittelnde Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ ergibt. Aufgrund dieser kombinatorischen Überlegungen kann die Berechnung der Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ zurückgeführt werden auf die Bestimmung der Wahrscheinlichkeiten $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$ für diejenigen Muster $M_n \in \mathcal{M}$, die gerade die vorgegebene Anzahl $Z_{anz} = i$ von *bad*-Phasen mit den vorgegebenen Längen $l_1, \ldots, l_i$ beinhalten. Dabei können die Anfangsstellen $k_1, \ldots, k_i$ der *bad*-Phasen unter Berücksichtigung von Bemerkung 6.1.8 alle möglichen Werte annehmen.

In Lemma 6.1.15 haben wir für die Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$ eines konkreten Musters $M_n \in \mathcal{M}$ geschlossene Formeln hergeleitet. Diese hängen nicht von den Anfangsstellen $k_1, \ldots, k_i$ der $Z_{anz} = i$ *bad*-Phasen ab; d.h. bei der Summation der Wahrscheinlichkeiten $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$ über alle möglichen Anfangsstellen $k_1, \ldots, k_i$ hängen die Summanden nicht von den Summationsindizes $k_1, \ldots, k_i$ ab. Daher reduziert sich die Summation auf die Fragestellung, wie viele verschiedene Muster $M_n \in \mathcal{M}$ es mit $Z_{anz} = i$ *bad*-Phasen der Längen $l_1, \ldots, l_i$ gibt, wie viele verschiedene Elemente also die Menge $\mathcal{M}_i(l_1, \ldots, l_i)$ aus Definition 6.1.12 umfaßt. Zu bestimmen ist demnach

$$|\mathcal{M}_i(l_1, \ldots, l_i)|.$$

Zur Lösung dieser kombinatorischen Fragestellung überlegen wir uns zunächst, wie viele *freie Stellen* es in einem Muster $M_n \in \mathcal{M}$ der Länge $n$ gibt, die quasi als mögliche Anfangsstellen der $Z_{anz} = i$ *bad*-Phasen in Frage kommen, wenn $M_n$ genau $Z_{anz} = i$ *bad*-Phasen der Längen $l_1, \ldots, l_i$ enthält. Dazu bezeichnen wir mit

$$M_{i,(l_1,\ldots,l_i)}(k_1, \ldots, k_i)$$

die Anzahl der freien Stellen in einem Muster $M_n \in \mathcal{M}$ der Länge $n$ mit $Z_{anz} = i$ *bad*-Phasen der Längen $l_1, \ldots, l_i$. Zur der Bestimmung von $M_{i,(l_1,\ldots,l_i)}(k_1, \ldots, k_i)$ haben wir die Fallunterscheidung bezüglich der Anfangsstellen der ersten und der letzen *bad*-Phase, $k_1$ und $k_i$, aus Lemma 6.1.15 zu beachten.

1. Fall: $k_1 > 1$ und $k_i < n - l_i + 1$

   Da das Muster $M_n \in \mathcal{M}$ die Länge $n$ besitzt, stehen zuächst insgesamt $n$ Stellen zur Verfügung. Nun umfaßt das Muster $M_n \in \mathcal{M}$ $Z_{anz} = i$ *bad*-Phasen der Längen $l_1, \ldots, l_i$, so dass von diesen $n$ Stellen $\sum_{j=1}^{i}(l_j - 1)$ Stellen zu subtrahieren sind sowie weitere $i$ Stellen, da die $Z_{anz} = i$ *bad*-Phasen durch *good*-Phasen mindestens der Länge 1 voneinander getrennt werden. Noch eine weitere Stelle ist abzuziehen, da die erste *bad*-Phase nicht am Wortanfang sondern in der Wortmitte beginnt. Daher erhalten wir:

$$\begin{aligned} M_{i,(l_1,\ldots,l_i)}(k_1, \ldots, k_i) &= n - \sum_{j=1}^{i}(l_j - 1) - i - 1 \\ &= n - \sum_{j=1}^{i} l_1 + i - i - 1 = n - \sum_{j=1}^{i} l_1 - 1 \end{aligned}$$

   Mit kombinatorischen Methoden ergibt sich, dass die Anzahl der Möglichkeiten, $i$ verschiedene Anfangsstellen aus $n - \sum_{j=1}^{i} l_1 - 1$ *freien Stellen* auszuwählen, und damit die Anzahl $|\mathcal{M}_i(l_1, \ldots, l_i)|$ der Elemente der Menge $\mathcal{M}_i(l_1, \ldots, l_i)$ dem folgenden Binomialkoeffizienten entspricht:

$$|\mathcal{M}_i(l_1, \ldots, l_i)| = \binom{n - \sum_{j=1}^{i} l_j - 1}{i}$$

2. Fall: ($k_1 = 1$ und $k_i < n - l_i + 1$) bzw. ($k_1 > 1$ und $k_i = n - l_i + 1$)

   Da das Muster $M_n$ die Länge $n$ besitzt und die erste *bad*-Phase am Wortanfang beginnt, stehen zunächst $n - l_1$ freie Stellen zur Verfügung. Davon sind wegen der übrigen $i-1$ *bad*-Phasen $\sum_{j=2}^{i}(l_j - 1)$ Stellen zu subtrahieren und weitere $i - 1$ Stellen, da die $Z_{anz} = i$ *bad*-Phasen durch *good*-Phasen mindestens der Länge 1 voneinander getrennt werden. Noch eine weitere Stelle ist abzuziehen, da die letzte *bad*-Phase nicht am Wortende sondern in der Wortmitte endet. Daher erhalten wir:

$$\begin{aligned} M_{i,(l_1,\ldots,l_i)}(1, k_2, \ldots, k_i) &= M_{i,(l_1,\ldots,l_i)}(k_1, \ldots, k_{i-1}, n - l_i + 1) \\ &= n - l_1 - \sum_{j=2}^{i}(l_j - 1) - (i - 1) - 1 = n - \sum_{j=1}^{i} l_1 - 1 \end{aligned}$$

Analog zu Fall 1 ergibt sich mit kombinatorischen Methoden, dass die Anzahl der Möglichkeiten, $i-1$ verschiedene Anfangsstellen aus $n - \sum_{j=1}^{i} l_1 - 1$ *freien Stellen* auszuwählen, und damit die Anzahl $|\mathcal{M}_i(l_1, \ldots, l_i)|$ dem nachstehenden Binomialkoeffizienten entspricht:

$$|\mathcal{M}_i(l_1, \ldots, l_i)| = \binom{n - \sum_{j=1}^{i} l_j - 1}{i-1}$$

3. Fall: $k_1 = 1$ und $k_i = n - l_i + 1$

Da das Muster $M_n$ die Länge $n$ besitzt und die erste *bad*-Phase am Wortanfang beginnt und die letzte am Wortende endet, stehen zunächst $n - l_1 - l_i$ freie Stellen zur Verfügung. Davon sind wegen der übrigen $i-2$ *bad*-Phasen $\sum_{j=2}^{i-1}(l_j - 1)$ Stellen zu subtrahieren und weitere $i-1$ Stellen, da die $Z_{anz} = i$ *bad*-Phasen durch *good*-Phasen mindestens der Länge 1 voneinander getrennt werden. Daher erhalten wir:

$$\begin{aligned} M_{i,(l_1,\ldots,l_i)}(1, k_2, \ldots, k_{i-1}, n - l_i + 1) &= n - l_1 - l_i - \sum_{j=2}^{i-1}(l_j - 1) - (i-1) \\ &= n - \sum_{j=1}^{i} l_1 - 1 \end{aligned}$$

Analog zu den beiden anderen Fällen ergibt sich mit kombinatorischen Methoden, dass die Anzahl der Möglichkeiten, $i-2$ verschiedene Anfangsstellen aus $n - \sum_{j=1}^{i} l_1 - 1$ *freien Stellen* auszuwählen, und damit die Anzahl $|\mathcal{M}_i(l_1, \ldots, l_i)|$ dem nachfolgenden Binomialkoeffizienten entspricht:

$$|\mathcal{M}_i(l_1, \ldots, l_i)| = \binom{n - \sum_{j=1}^{i} l_j - 1}{i-2}$$

Folglich erhalten wir zusammen mit den geschlossenen Formeln bezüglich der Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$ aus Lemma 6.1.15 die Darstellung für die Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ wie oben behauptet. □

### 6.1.17 Bemerkung

Die geschlossene Darstellung aus Satz 6.1.16 für die Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ ist insbesondere auch für den Fall $i = 0$ gültig, d.h. für den Fall, dass während der Übertragung keine *bad*-Phase auftritt:

$$\begin{aligned} P_{M_n}\big(\mathbf{Z} = \mathbf{0}\big) = \pi_G \left(q_{GG}\right)^{n-1} &= \frac{\left(q_{BG}\right)\left(q_{GG}\right)^{n-1}}{q_{GB} + q_{BG}} \\ &= \frac{\left(q_{GB}\right)^0\left(q_{BG}\right)}{q_{GB} + q_{BG}} (q_{GG})^{n-1} (q_{BB})^0 \end{aligned}$$

denn per Definition sind Binomialkoeffizienten über negativen Werten gerade Null, so dass die ersten beiden Summanden aus Satz 6.1.16 Null sind.

Bevor wir zum Abschluß dieses Abschnittes nun eine Formel für die Wahrscheinlichkeit $P_{M_n}(Z_{anz} = i)$, dass der Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ in einem Muster $M_n \in \mathcal{M}$ genau $Z_{anz} = i$ *bad*-Phasen erzeugt, wobei $0 \leq i \leq \lceil \frac{n}{2} \rceil$ gilt, angeben können, sind für die Herleitung einer solchen Formel noch einige Vorüberlegungen anzustellen. Dazu führen wir weitere wichtige Begriffe und Bezeichnungen ein, die wir dann zur Darstellung der Wahrscheinlichkeit $P_{M_n}(Z_{anz} = i)$ benötigen.

**6.1.18 Definition**

*(i) Für $j \in \mathbb{N}$ sind das obere und das untere Pochhammer Symbol $a^{\overline{j}}$ und $a^{\underline{j}}$ folgendermaßen definiert:*

$$a^{\overline{j}} := \underbrace{a\,(a+1)\ldots(a+j-1)}_{\text{insgesamt } j \text{ Faktoren}}$$

$$\text{und} \quad a^{\underline{j}} := \underbrace{a\,(a-1)\ldots(a-j+1)}_{\text{insgesamt } j \text{ Faktoren}}$$

*(ii) Eine allgemeine hypergeometrische Reihe ist in Anlehnung an [23] eine unendliche Potenzreihe in $x$ mit $u+v$ Parametern, die mittels der oberen Pochhammer Symbole definiert wird:*

$$F\left(\left.\begin{matrix} a_1,\ldots,a_u \\ b_1,\ldots,b_v \end{matrix}\right| x\right) := \sum_{j\geq 0} \frac{a_1^{\overline{j}}\ldots a_u^{\overline{j}}}{b_1^{\overline{j}}\ldots b_v^{\overline{j}}}\,\frac{x^j}{j!}$$

*Dabei sind $a_1,\ldots,a_u \in \mathbb{R}$ und $b_1,\ldots,b_v \in \mathbb{R}^{>0}$.*

**6.1.19 Bemerkung**

(i) Für $j = 0$ sind die Pochhammer Symbole definitionsgemäß gleich 1:

$$a^{\overline{0}} := 1 \quad \text{und} \quad a^{\underline{0}} := 1$$

(ii) Die folgenden Rechenregeln stellen einen Zusammenhang zwischen den oberen und den unteren Pochhammer Symbolen her:

$$a^{\underline{j}} = (-1)^j \cdot a^{\overline{j}}$$

$$\text{und} \quad a^{\overline{j}} = (-1)^j \cdot a^{\underline{j}}$$

**6.1.20 Lemma**

*Sei $i$ mit $1 \leq i \leq \lceil \frac{n}{2} \rceil$ fest vorgegeben, dann gelten für ein beliebiges $p \in \mathbb{R}$ die folgenden drei Aussagen:*

$$(i)\quad \sum_{\substack{1\le l_1,\dots,l_i\le n-2i+2\\ \sum_{j=1}^{i} l_j\le n-i+1}} \binom{n-\sum_{j=1}^{i} l_j-1}{i-2} p^{\sum_{j=1}^{i} l_j} = p^i \binom{n-i-1}{i-2} F\left(\begin{matrix} i,\ -n+2i-1 \\ -n+i+1 \end{matrix}\middle| p\right)$$

$$(ii)\quad \sum_{\substack{1\le l_1,\dots,l_i\le n-2i+1\\ \sum_{j=1}^{i} l_j\le n-i}} \binom{n-\sum_{j=1}^{i} l_j-1}{i-1} p^{\sum_{j=1}^{i} l_j} = p^i \binom{n-i-1}{i-1} F\left(\begin{matrix} i,\ -n+2i \\ -n+i+1 \end{matrix}\middle| p\right)$$

$$(iii)\quad \sum_{\substack{1\le l_1,\dots,l_i\le n-2i\\ \sum_{j=1}^{i} l_j\le n-i-1}} \binom{n-\sum_{j=1}^{i} l_j-1}{i} p^{\sum_{j=1}^{i} l_j} = p^i \binom{n-i-1}{i} F\left(\begin{matrix} i,\ -n+2i+1 \\ -n+i+1 \end{matrix}\middle| p\right)$$

**Beweis**
Mittels kombinatorischer Methoden zeigen wir die Behauptungen (i) - (iii). Dazu überlegen wir uns zunächst, wie groß die Anzahl der Möglichkeiten ist, einen festen Wert $l$ als Summe der Summationsindizes $l_1,\dots,l_i$, also in der Form

$$l=\sum_{j=1}^{i} l_j,$$

darzustellen. Dabei unterliegen $l_1,\dots,l_i$ sowie $l$ jeweils den im Summationsindex angegebenen Bedingungen der Aussagen (i)-(iii). Zu diesem Zweck bezeichne $S_l(i)$ nun die Anzahl der Möglichkeiten, den Wert $l$ darzustellen als $l=\sum_{j=1}^{i} l_j$.
Mit vollständiger Induktion nach $i$ beweisen wir nun, dass gilt:

$$S_l(i)=\binom{l-1}{i-1}$$

Induktionsanfang: $i=1$

$$S_l(1)=1=\binom{l-1}{0}$$

Induktionsvoraussetzung:

$$S_l(i)=\binom{l-1}{i-1}$$

Induktionsschritt: $i\longrightarrow i+1$:

$$S_l(i+1)=\sum_{j=i}^{l-1} S_j(i) \overset{\text{IV}}{=} \sum_{j=i}^{l-1}\binom{j-1}{i-1}$$

$$= \sum_{j=0}^{l-i-1}\binom{i+j-1}{i-1}=\binom{l-1}{i}$$

Zur Vereinfachung setzen wir für alle weiteren Überlegungen $l := \sum_{j=1}^{i} l_j$, dann erhalten wir bezüglich der Aussage (iii):

$$
\begin{aligned}
& \sum_{\substack{1 \le l_1,\ldots,l_i \le n-2i \\ \sum_{j=1}^{i} l_j \le n-i-1}} \binom{n - \sum_{j=1}^{i} l_j - 1}{i} p^{\sum_{j=1}^{i} l_j} \\
= & \sum_{\substack{1 \le l_1,\ldots,l_i \le n-2i \\ l \le n-i-1}} \binom{n-l-1}{i} p^l \\
= & \sum_{l=i}^{n-i-1} \binom{l-1}{i-1} \binom{n-l-1}{i} p^l = \sum_{l=i}^{\infty} \binom{l-1}{i-1} \binom{n-l-1}{i} p^l \\
= & \sum_{l=0}^{\infty} \binom{l+i-1}{i-1} \binom{n-l-i-1}{i} p^{l+i} \\
= & \sum_{l=0}^{\infty} \frac{(l+i-1)!}{(i-1)!\,l!} \frac{(n-i-1)!\,(n-2i-1)!}{(n-i-1)!\,(n-2i-1)!} \frac{(n-l-i-1)!}{i!\,(n-l-2i-1)!} p^{l+i} \\
= & \; p^i \binom{n-i-1}{i} \sum_{l=0}^{\infty} \frac{i^{\overline{l}}(-n+2i+1)^{\overline{l}}}{(-n+i+1)^{\overline{l}}} \frac{p^l}{l!} \\
= & \; p^i \binom{n-i-1}{i} F\left( \left. \begin{matrix} i, & -n+2i+1 \\ & -n+i+1 \end{matrix} \right| p \right)
\end{aligned}
$$

Die Aussagen (i) und (ii) lassen sich mit $l := \sum_{j=1}^{i} l_j$ ganz analog beweisen. □

**6.1.21 Satz**
*Sei $i$, mit $0 \le i \le \lceil \frac{n}{2} \rceil$, gegeben, dann läßt sich die Wahrscheinlichkeit $P_{M_n}(Z_{anz} = i)$, dass in einem vom Hintergrundprozeß $(X_t)_{t \in \mathbb{N}}$ erzeugten Muster $M_n \in \mathcal{M}$ genau $Z_{anz} = i$ bad-Phasen auftreten, mit der folgenden Formel bestimmen:*

$$
\begin{aligned}
P_{M_n}(Z_{anz} = i) = & \frac{(q_{GB})^i\,(q_{BG})^{i-1}}{q_{GB}+q_{BG}} (q_{GG})^{n-2i-1} \\
& \Bigg[ (q_{GG})^2 \binom{n-i-1}{i-2} F\left( \left. \begin{matrix} i, & -n+2i-1 \\ & -n+i+1 \end{matrix} \right| \frac{q_{BB}}{q_{GG}} \right) \\
& + 2\,q_{BG}\,q_{GG} \binom{n-i-1}{i-1} F\left( \left. \begin{matrix} i, & -n+2i \\ & -n+i+1 \end{matrix} \right| \frac{q_{BB}}{q_{GG}} \right) \\
& + (q_{BG})^2 \binom{n-i-1}{i} F\left( \left. \begin{matrix} i, & -n+2i+1 \\ & -n+i+1 \end{matrix} \right| \frac{q_{BB}}{q_{GG}} \right) \Bigg]
\end{aligned}
$$

**Beweis**
Bei der Einführung der Wahrscheinlichkeit $P_{M_n}(Z_{anz} = i)$ in Definition 6.1.5 wird vorausgesetzt, dass die Anzahl der aufgetretenen *bad*-Phasen bekannt ist. Wir betrachten nun diejenigen Muster $M_n \in \mathcal{M}$, die genau diese fest vorgegebene Anzahl $Z_{anz} = i$ von *bad*-Phasen beinhalten, also alle $M_n \in \mathcal{M}(i)$. Dabei sind die Längen $l_1, \ldots, l_i$ sowie die Anfangsstellen $k_1, \ldots, k_i$ dieser *bad*-Phasen unter Berücksichtigung der Bedingungen aus den Bemerkungen 6.1.9 und 6.1.8 beliebig.

Bilden wir nun die Summe der Wahrscheinlichkeiten $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ aller Muster $M_n \in \mathcal{M}$ mit $Z_{anz} = i$ *bad*-Phasen der Längen $l_1, \ldots, l_i$ über alle möglichen Längen $l_1, \ldots, l_i$, so liefern kombinatorische Methoden, dass diese Summe die zu ermittelnde Wahrscheinlichkeit $P_{M_n}(Z_{anz} = i)$ ergibt. Aufgrund dieser kombinatorischen Überlegungen kann die Berechnung der Wahrscheinlichkeit $P_{M_n}(Z_{anz} = i)$ zurückgeführt werden auf die Bestimmung der Wahrscheinlichkeiten $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$, wobei die Anzahl der aufgetretenen *bad*-Phasen den vorgegebenen Wert $Z_{anz} = i$ annimmt. Die Längen $l_1, \ldots, l_i$ können dabei unter Berücksichtigung von Bemerkung 6.1.9 alle möglichen Werte annehmen. Von daher ist die folgende Summe zu bilden:

$$P_{M_n}(Z_{anz} = i) \quad = \sum_{\substack{1 \le l_1, \ldots, l_i \le n-2i+2 \\ \sum_{j=1}^{i} l_j \le n-i+1}} P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$$

Bei dieser Summation über alle möglichen Längen $l_1, \ldots, l_i$ ist allerdings erstens zu beachten, dass nach Bemerkung 6.1.9 die maximal möglichen Längen der aufgetretenen *bad*-Phasen variieren, je nachdem an welcher Stelle die erste *bad*-Phase beginnt und die letzte *bad*-Phase endet. Daher müssen wir bei dieser Summation eine Fallunterscheidung bezüglich der Anfangsstelle der ersten und der letzen *bad*-Phase berücksichtigen.

Zweitens haben wir uns in Satz 6.1.16 bei der Herleitung einer Formel für die Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ beschränkt auf die Bestimmung der Wahrscheinlichkeiten $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i), \mathbf{W} = (k_1, \ldots, k_i)\big)$. In diesem Zusammenhang ist in die Berechnung der Wahrscheinlichkeit $P_{M_n}\big(\mathbf{Z} = (l_1, \ldots, l_i)\big)$ die Fallunterscheidung bezüglich der Anfangsstellen der ersten und der letzen *bad*-Phase, $k_1$ und $k_i$, aus Lemma 6.1.15 eingegangen.

Beachten wir diese Differenzierung nun bei der Summation über alle möglichen Längen $l_1, \ldots, l_i$, und setzen wir das Ergebnis aus Satz 6.1.16 ein, so erhalten wir schließlich:

$$\begin{aligned}
&P_{M_n}(Z_{anz} = i) \\
&= \sum_{\substack{1 \le l_1, \ldots, l_i \le n-2i+2 \\ \sum_{j=1}^{i} l_j \le n-i+1}} \binom{n - \sum_{j=1}^{i} l_j - 1}{i-2} \frac{(q_{GB})^i \, (q_{BG})^{i-1}}{q_{GB} + q_{BG}} (q_{GG})^{n - \sum_{j=1}^{i} l_j - i + 1} (q_{BB})^{\sum_{j=1}^{i} l_j - i} \\
&\quad + \sum_{\substack{1 \le l_1, \ldots, l_i \le n-2i+1 \\ \sum_{j=1}^{i} l_j \le n-i}} \binom{n - \sum_{j=1}^{i} l_j - 1}{i-1} \frac{(q_{GB})^i \, (q_{BG})^{i}}{q_{GB} + q_{BG}} (q_{GG})^{n - \sum_{j=1}^{i} l_j - i} (q_{BB})^{\sum_{j=1}^{i} l_j - i}
\end{aligned}$$

$$+ \sum_{\substack{1\le l_1,\ldots,l_i\le n-2i \\ \sum_{j=1}^{i} l_j \le n-i-1}} \binom{n-\sum_{j=1}^{i} l_j - 1}{i} \frac{(q_{GB})^i\,(q_{BG})^{i+1}}{q_{GB}+q_{BG}} (q_{GG})^{n-\sum_{j=1}^{i} l_j - i - 1} (q_{BB})^{\sum_{j=1}^{i} l_j - i}$$

und setzen wir $p := \dfrac{q_{BB}}{q_{GG}}$, dann ergibt sich:

$$= \frac{(q_{GB})^i\,(q_{BG})^{i-1}}{q_{GB}+q_{BG}} (q_{GG})^{n-i-1} (q_{BB})^{-i}$$

$$\left[ (q_{GG})^2 \sum_{\substack{1\le l_1,\ldots,l_i\le n-2i+2 \\ \sum_{j=1}^{i} l_j \le n-i+1}} \binom{n-\sum_{j=1}^{i} l_j - 1}{i-2} p^{\sum_{j=1}^{i} l_j} \right.$$

$$+ 2\,q_{BG}\,q_{GG} \sum_{\substack{1\le l_1,\ldots,l_i\le n-2i+1 \\ \sum_{j=1}^{i} l_j \le n-i}} \binom{n-\sum_{j=1}^{i} l_j - 1}{i-1} p^{\sum_{j=1}^{i} l_j}$$

$$\left. + (q_{BG})^2 \sum_{\substack{1\le l_1,\ldots,l_i\le n-2i \\ \sum_{j=1}^{i} l_j \le n-i-1}} \binom{n-\sum_{j=1}^{i} l_j - 1}{i} p^{\sum_{j=1}^{i} l_j} \right]$$

Mit den Aussagen (i) -(iii) aus dem vorangegangenen Lemma 6.1.20 erhalten wir dann sofort die Behauptung. □

Da die Wahrscheinlichkeit, dass während der Übertragung eines Wortes der Länge $n$ der Hintergrundprozeß eine gewisse Anzahl von *bad*-Phasen in einem Muster $M_n \in \mathcal{M}$ erzeugt, die Restfehlerwahrscheinlichkeit eines Codes maßgeblich beeinflußt, ist eine detailliertere Untersuchung dieser Wahrscheinlichkeit in Abhängigkeit von den Kanalparametern in Zukunft wünschenswert. Mit mathematischen Werkzeugen, wie *Maple*, können wir die obige geschlossene Formel als eine Funktion der Kanalparameter, d.h. der Bitfehlerrate $p_E$ und der mittleren Bündelfehlerlänge $E_b$, darstellen und schließlich die Ableitungen bezüglich dieser beiden Parameter berechnen. Aus den Ableitungen können wir dann beispielsweise Rückschlüsse auf das Monotonieverhalten dieser Wahrscheinlichkeit ziehen.

### 6.1.2 Wahrscheinlichkeiten für Fehler in einer *bad*-Phase

Der zweite Faktor, der die Restfehlerwahrscheinlichkeit eines zur Fehlerkorrektur eingesetzten Blockcodes maßgeblich beeinflußt, ist die Wahrscheinlichkeit, dass innerhalb der in einem Muster $M_n \in \mathcal{M}$ aufgetretenen *bad*-Phasen Fehler in einer Form bzw. Anzahl erzeugt werden, die vom verwendeten Fehler-korrigierenden Blockcode nicht bzw. nicht korrekt korrigiert werden können. Bei der Analyse dieses zweiten Faktors, also bei der Bestimmung dieser Wahrscheinlichkeiten, ist zu differenzieren, ob ein REC-Code oder ein BEC-Code zur Fehlerkorrektur eingesetzt wird. So ist für einen $r$-REC-Code die Wahrscheinlichkeit zu ermitteln, dass innerhalb der aufgetretenen *bad*-Phasen mehr als $r$ beliebig verteilte *random errors* generiert werden. Für einen $b$-BEC-Code dagegen ist die Wahrscheinlichkeit zu berechnen, dass die innerhalb der aufgetretenen *bad*-Phasen erzeugten Fehler einen Bündelfehler der Länge $> b$ bilden. Von daher geht das Fehlerkorrekturverhalten des zur Fehlerkorrektur verwendeten Blockcodes an dieser Stelle in die Herleitung einer entsprechenden Näherungsformel ein, so dass wir im nachfolgenden Abschnitt 6.2 Näherungsformeln jeweils für die Restfehlerwahrscheinlichkeit von REC- und BEC-Codes zu entwickeln haben.

Zur näherungsweisen Bestimmung der Restfehlerwahrscheinlichkeit eines REC- bzw. BEC-Codes in Abschnitt 6.2 berücksichtigen wir allerdings nur diejenigen Muster $M_n \in \mathcal{M}$, die genau eine *bad*-Phase umfassen, also $M_n \in \mathcal{M}(1)$, und lassen die übrigen Muster $M_n \in \mathcal{M} \setminus \mathcal{M}(1)$ außer acht. Die Berechnung der Wahrscheinlichkeit, dass insgesamt innerhalb aller in einem Muster $M_n \in \mathcal{M}$ aufgetretenen *bad*-Phasen Fehler generiert werden, die nicht mehr korrigierbar sind, reduziert sich aus diesem Grund auf die Ermittlung der Wahrscheinlichkeit, dass innerhalb einer einzigen *bad*-Phase Fehler derart erzeugt werden, dass diese nicht mehr vom eingesetzten Code korrigiert werden können, vgl. Abschnitt 6.2. Für einen $r$-REC-Code bestimmen wir also die Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase mehr als $r$ *random errors* erzeugt werden und entsprechend für einen $b$-BEC-Code die Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase Bündelfehler der Länge $> b$ generiert werden, vgl. Abschnitt 6.2. Als Vorbereitung für die Berechnung dieser Wahrscheinlichkeiten und damit als Vorbereitung für die Herleitung von Näherungsformeln zur Berechnung der Restfehlerwahrscheinlichkeit eines REC- bzw. BEC-Codes im nächsten Abschnitt 6.2 geben wir in diesem Abschnitt eine geschlossene Darstellung der Wahrscheinlichkeit an, dass innerhalb einer *bad*-Phase einer fest vorgegebenen Länge Fehler derart generiert werden, dass diese vom verwendeten Code nicht mehr korrigiert werden können.

Dazu gehen wir in diesem Abschnitt von der Voraussetzung aus, dass ein während der Übertragung vom Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugtes Muster $M_n \in \mathcal{M}$ nur eine *bad*-Phase umfaßt; d.h. $M_n \in \mathcal{M}(1)$. Der Zufallsvektor $\mathbf{Z}$ aus Bezeichnung 6.1.7 gibt dabei gerade die Länge $\mathbf{Z} = l$ der aufgetretenen *bad*-Phase an, wobei die Länge den Beschränkungen aus Bemerkung 6.1.9 unterliegt.

Für alle weiteren Ausführungen in diesem Kapitel differenzieren wir nun zwischen $r$-REC-Codes und $b$-BEC-Codes. Bevor wir nun für einen $r$-REC-Code bzw. einen $b$-BEC-Code eine geschlossene Formel herleiten, die die bedingte Wahrscheinlichkeit angibt, dass innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$ mehr als $r$ *random errors* bzw. Bündelfehler der Länge $> b$ erzeugt werden, führen wir die folgenden Begriffe und Bezeichnungen ein.

### 6.1.22 Bezeichnungen

*(i) Die diskrete Zufallsvariable $U$ gebe die Anzahl der random errors an, die während der Übertragung innerhalb genau einer bad-Phase generiert werden.*

*(ii) Die diskrete Zufallsvariable $V$ gebe die Länge eines Bündelfehlers an, der während der Übertragung innerhalb genau einer bad-Phase generiert wird.*

### 6.1.23 Bemerkung

Der Wertebereich der beiden diskreten Zufallsvariablen $U$ und $V$ hängt dabei ganz offensichtlich von der Länge $\mathbf{Z} = l$ der in einem Muster $M_n \in \mathcal{M}(1)$ aufgetretenen *bad*-Phase ab und ist somit gegeben durch

$$U, V \in \{0, 1, \ldots, l\}.$$

Zunächst bestimmen wir nun für einen $r$-REC-Code die bedingte Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$ mehr als $r$ *random errors* generiert werden, um dann anschließend entsprechend für einen $b$-BEC-Code die bedingte Wahrscheinlichkeit zu ermitteln, dass innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$ Bündelfehler der Länge $> b$ entstehen können.

### 6.1.24 Satz

*Für einen $r$-REC-Code beträgt die bedingte Wahrscheinlichkeit, dass innerhalb einer einzigen bad-Phase der Länge $\mathbf{Z} = l$ mehr als $r$ random errors generiert werden:*

$$P(U > r \,|\, \mathbf{Z} = l) = 1 - \sum_{k=0}^{r} \frac{1}{2^l} \binom{l}{k}$$

*für alle $r \in \{1, 2, \ldots l\}$.*

**Beweis**

Nach den Voraussetzungen des Modells von Gilbert wird jedes Bit innerhalb einer *bad*-Phase mit der Fehlerwahrscheinlichkeit $e_B = P(X_t = B \,|\, E_t = 1) = 1/2$ verfälscht, vgl. dazu Bemerkung 6.0.5. Demzufolge wird die bedingte Wahrscheinlichkeit, dass genau $r$ Fehler innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$ erzeugt werden, durch die Anzahl aller möglichen Kombinationen, $r$ Fehler auf den $\mathbf{Z} = l$ Positionen der *bad*-Phase zu verteilen, sowie durch die Fehlerwahrscheinlichkeit $e_B = 1/2$ bedingt; d.h. es gilt:

$$P(U = r \,|\, \mathbf{Z} = l) = \frac{1}{2^l} \binom{l}{r}.$$

Folglich ist die Wahrscheinlichkeit für das Auftreten von weniger oder genau $r$ Fehlern bzw. von mehr als $r$ Fehlern innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$ gegeben durch

$$P(U \leq r \,|\, \mathbf{Z} = l) = \sum_{k=0}^{r} \frac{1}{2^l} \binom{l}{k} \quad \text{bzw.} \quad P(U > r \,|\, \mathbf{Z} = l) = 1 - \sum_{k=0}^{r} \frac{1}{2^l} \binom{l}{k}.$$

□

**6.1.25 Satz**
*Für einen b-BEC-Code beträgt die bedingte Wahrscheinlichkeit, dass innerhalb einer einzigen bad-Phase der Länge* $\mathbf{Z} = l$ *ein Bündelfehler der Länge* $> b$ *erzeugt wird:*

$$P(V > b \,|\, \mathbf{Z} = l) = 1 - \frac{l - b + 2}{2^{l-b+1}}$$

*für alle* $b \in \{2, \ldots, l\}$.

**Beweis**
Jedes Bit innerhalb einer *bad*-Phase wird laut den Vorgaben des zugrunde gelegten Modells von Gilbert mit der Fehlerwahrscheinlichkeit $e_B = P(E_t = 1 \,|\, X_t = B) = 1/2$ fehlerhaft übertragen, vgl. Bemerkung 6.0.5. Die bedingte Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase der vorgegebenen Länge $\mathbf{Z} = l$ ein Bündelfehler genau der Länge $b$ erzeugt wird, wird beeinflußt durch die Fehlerwahrscheinlichkeit $e_B$, die Anzahl der verschiedenen Bündelfehler der Länge $b$ sowie durch die Anzahl der unterschiedlichen Anfangsstellen für einen Bündelfehler der Länge $b$ innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$. Demzufolge erhalten wir:

$$P(V = b \,|\, \mathbf{Z} = l) = \frac{1}{2^l}\, 2^{b-2}\, (l - b + 1) = \frac{1}{2^{l-b+2}}\, (l - b + 1)$$

Zu beachten sind die Sonderfälle für Bündelfehler der Länge $b = 0$ bzw. $b = 1$:

$$P(V = 0 \,|\, \mathbf{Z} = l) = \frac{1}{2^l} \quad \text{und} \quad P(V = 1 \,|\, \mathbf{Z} = l) = l\, \frac{1}{2^l}.$$

Dann summiert sich die Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase der Länge $\mathbf{Z} = l$ ein Bündelfehler der Länge $> b$ mit $1 \leq b \leq l - 1$ auftritt, zu:

$$\begin{aligned}
P(V > b \,|\, \mathbf{Z} = l) &= 1 - P(V \leq b \,|\, \mathbf{Z} = l) \\
&= 1 - P(V = 0 \,|\, \mathbf{Z} = l) - P(V = 1 \,|\, \mathbf{Z} = l) - \sum_{k=2}^{b} P(V = k \,|\, \mathbf{Z} = l) \\
&= 1 - \frac{1}{2^l} - l\, \frac{1}{2^l} - \sum_{k=2}^{b} \frac{1}{2^{l-k+2}}\, (l - k + 1) \\
&= 1 - \frac{l+1}{2^l} - \sum_{k=0}^{b-2} \frac{1}{2^{l-k}}\, (l - k - 1) \\
&= 1 - \frac{l - b + 2}{2^{l-b+1}}.
\end{aligned}$$

Speziell, für den Sonderfall $b = 0$ erhalten wir:

$$P(V > 0 \,|\, \mathbf{Z} = l) = 1 - \frac{1}{2^l}.$$

□

## 6.2 Näherungsformeln für die Restfehlerwahrscheinlichkeit

Nach den vorangehenden Ausführungen wird die Restfehlerwahrscheinlichkeit eines zur Fehlerkorrektur eingesetzten Blockcodes beeinflußt durch die Wahrscheinlichkeit für das Auftreten einer gewissen Anzahl von *bad*-Phasen in einem vom Markov'schen Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugten Muster $M_n \in \mathcal{M}$ und durch die Wahrscheinlichkeit, dass in den aufgetretenen *bad*-Phasen insgesamt entweder eine so große Anzahl von Fehlern oder Fehler in der Form generiert werden, so dass diese nicht mehr vom verwendeten REC- bzw. BEC-Code korrigiert werden können. Aus den Ergebnissen der beiden vorherigen Abschnitte 6.1.1 und 6.1.2 lassen sich nun Näherungsformeln zur Berechnung der Restfehlerwahrscheinlichkeit für einen $r$-REC-Code bzw. für einen $b$-BEC-Code ableiten.

### 6.2.1 Approximationsformeln für einen $r$-REC-Code

Für eine näherungsweise Bestimmung der Restfehlerwahrscheinlichkeit eines $r$-REC-Codes berechnen wir, wie im vorangehenden Abschnitt erwähnt, die Wahrscheinlichkeit, dass während der Übertragung mehr als $r$ *random errors* unter der Bedingung generiert werden, dass der Markov'sche Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ in einem Muster $M_n \in \mathcal{M}$ genau eine *bad*-Phase erzeugt. Mit Bezeichnung 6.1.22 ist also die Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase mehr als $r$ *random errors* erzeugt werden, zu bestimmen:

$$P(U > r)$$

Von daher wird vorausgesetzt, dass genau eine *bad*-Phase, also $Z_{anz} = 1$, in einem vom Markov'schen Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugten Muster $M_n \in \mathcal{M}$ enthalten ist und damit $M_n \in \mathcal{M}(1)$ ist. Nach Bezeichnung 6.1.7 gibt der Zufallsvektor $\mathbf{Z}$ gerade die Länge $\mathbf{Z} = l$ der aufgetretenen *bad*-Phase an und diese genügt laut Bemerkung 6.1.9 der Bedingung $1 \le l \le n$.

In Abschnitt 6.1.2, Satz 6.1.24 haben wir die bedingte Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase der Länge $l$ mehr als $r$ Fehler generiert werden, angegeben mit:

$$P(U > r \,|\, \mathbf{Z} = l) \;=\; 1 - \sum_{k=0}^{r} \frac{1}{2^l} \binom{l}{k}.$$

Die Wahrscheinlichkeit für das Auftreten einer *bad*-Phase der Länge $l$ in einem Muster $M_n \in \mathcal{M}(1)$ haben wir in Satz 6.1.16, Abschnitt 6.1.1, ermittelt:

$$P_{M_n}(\mathbf{Z} = l) \;=\; (q_{GG})^{n-3}\, \frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}} \left(\frac{q_{BB}}{q_{GG}}\right)^{l-1} (2\,q_{GG} + (n-l-1)q_{BG}).$$

Von daher ergibt sich schließlich die Wahrscheinlichkeit $P(U > r)$ aus der Summe der Produkte $P_{M_n}(\mathbf{Z} = l)\, P(U > r \,|\, \mathbf{Z} = l)$ über alle $l$, mit $1 \le l \le n$, und wir erhalten in diesem

Zuge eine erste Näherungsformel für die Restfehlerwahrscheinlichkeit eines $r$-REC-Codes bezüglich der Menge $\mathcal{M}(1)$:

$$\begin{aligned}
\tilde{\mathcal{R}}_{REC}(r,n) &= P(U>r) \\
&= P_{M_n}(\mathbf{Z}=n)\,P(U>r\,|\,\mathbf{Z}=n) + \sum_{l=r+1}^{n-1} P_{M_n}(\mathbf{Z}=l)\,P(U>r\,|\,\mathbf{Z}=l) \\
&= \frac{q_{GB}}{q_{GB}+q_{BG}}(q_{BB})^{n-1}\left(1-\sum_{k=0}^{r}\frac{1}{2^n}\binom{n}{k}\right) \\
&\quad + \sum_{l=r+1}^{n-1}(q_{GG})^{n-3}\,\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{l-1}(2\,q_{GG}+(n-l-1)q_{BG}) \\
&\quad \left(1-\sum_{k=0}^{r}\frac{1}{2^l}\binom{l}{k}\right).
\end{aligned}$$

Für einen 2-REC-Code, beispielsweise, wie wir ihn in den Kapiteln 4 und 5 bereits exemplarisch betrachtet haben, erhalten wir insbesondere die nachfolgende geschlossene Darstellung einer Nährungsformel für die Restfehlerwahrscheinlichkeit dieses 2-REC-Codes:

$$\begin{aligned}
\tilde{\mathcal{R}}_{REC}(2,n) &= P(U>2) \\
&= P_{M_n}(\mathbf{Z}=n)\,P(U>2\,|\,\mathbf{Z}=n) + \sum_{l=3}^{n-1} P_{M_n}(\mathbf{Z}=l)\,P(U>2\,|\,\mathbf{Z}=l) \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right) \\
&\quad + \sum_{l=3}^{n-1}(q_{GG})^{n-3}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{l-1}(2\,q_{GG}+q_{BG}(n-l-1)) \\
&\quad \left(1-\frac{l^2+l+2}{2^{l+1}}\right) \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right) \\
&\quad + (q_{GG})^{n-3}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{2} \\
&\quad \left[\sum_{l=0}^{n-4}(2\,q_{GG}+(n-4)\,q_{BG}-q_{BG}\,l)\left(1-\frac{l^2+7l+14}{2^{l+4}}\right)\left(\frac{q_{BB}}{q_{GG}}\right)^{l}\right]
\end{aligned}$$

$$
\begin{aligned}
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right)\\
&\quad+\frac{(q_{GG})^{n-3}}{16}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^2\\
&\quad\left[\sum_{l=0}^{n-4}(2\,q_{GG}+(n-4)\,q_{BG}-q_{BG}\,l)\left(16-\frac{l^2+7l+14}{2^l}\right)\left(\frac{q_{BB}}{q_{GG}}\right)^l\right]
\end{aligned}
$$

Zur Vereinfachung setzen wir

$$
\begin{aligned}
A &:= 2\,q_{GG}+(n-4)\,q_{BG}\\
C &:= q_{BB}/q_{GG}\\
D &:= q_{BG}
\end{aligned}
$$

und berechnen die endliche geometrische Reihe, um eine geschlossene Darstellung der Näherungsformel angeben zu können

$$
\begin{aligned}
\tilde{\mathcal{R}}_{REC}(2,n) &= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right)\\
&\quad+\frac{(q_{GG})^{n-3}}{16}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^2\left[\sum_{l=0}^{n-4}(A-D\,l)\left(16-\frac{l^2+7l+14}{2^l}\right)C^l\right]\\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right)\\
&\quad+\frac{(q_{GG})^{n-3}}{16}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^2\\
&\quad\left[16A\sum_{l=0}^{n-4}C^l-16D\sum_{l=0}^{n-4}l\,C^l-14A\sum_{l=0}^{n-4}\left(\frac{C}{2}\right)^l+14D\sum_{l=0}^{n-4}l\left(\frac{C}{2}\right)^l\right.\\
&\quad\left.-7A\sum_{l=0}^{n-4}l\left(\frac{C}{2}\right)^l+7D\sum_{l=0}^{n-4}l^2\left(\frac{C}{2}\right)^l-A\sum_{l=0}^{n-4}l^2\left(\frac{C}{2}\right)^l+D\sum_{l=0}^{n-4}l^3\left(\frac{C}{2}\right)^l\right]\\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right)\\
&\quad+\frac{(q_{GG})^{n-3}}{16}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^2
\end{aligned}
$$

$$
\begin{aligned}
&\left[ 16A\,\frac{1-C^{n-3}}{1-C} - 14A\,\frac{2}{2-C}\left(1-\left(\frac{C}{2}\right)^{n-3}\right)\right.\\
&-16D\,\frac{C}{(1-C)^2}\left(1-(n-3)C^{n-4}+(n-4)C^{n-3}\right)\\
&+(14D-7A)\frac{2C}{(2-C)^2}\left(1-(n-3)\left(\frac{C}{2}\right)^{n-4}+(n-4)\left(\frac{C}{2}\right)^{n-3}\right)\\
&+(7D-A)\frac{4C}{(2-C)^3}\\
&\quad\left(1+\frac{C}{2}-\left(\frac{C}{2}\right)^{n-4}\left(1+(n-4)\left(1-\frac{C}{2}\right)\right)^2-\left(\frac{C}{2}\right)^{n-3}\right)\\
&+D\,\frac{8C}{(2-C)^4}\left(1+4\frac{C}{2}+\left(\frac{C}{2}\right)^2\right.\\
&\qquad-\left(\frac{C}{2}\right)^{n-4}\left(\left(1+(n-4)\left(1-\frac{C}{2}\right)\right)^3+3(n-4)\frac{C}{2}\left(1-\frac{C}{2}\right)\right)\\
&\qquad\left.\left.-4\left(\frac{C}{2}\right)^{n-3}-4\left(\frac{C}{2}\right)^{n-3}\right)\right]
\end{aligned}
$$

$$
\begin{aligned}
=\;& (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n^2+n+2}{2^{n+1}}\right)\\
&+\frac{(q_{GG})^{n-3}}{16}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^2\\
&\left[A\left(\frac{16}{1-C}-\frac{28}{2-C}-\frac{14C}{(2-C)^2}-\frac{4C-2C^2}{(2-C)^3}\right)\right.\\
&+D\left(\frac{-16}{1-C}+\frac{28C}{(2-C)^2}+\frac{14C(2+C)}{(2-C)^3}+\frac{C(8+16C+2C^2)}{(2-C)^4}\right)\\
&+C^{n-3}\left(\frac{-16A}{1-C}+\frac{16D(n-3-C(n-4))}{(1-C)^2}\right)\\
&+\left(\frac{C}{2}\right)^{n-3}\left(\frac{28A}{2-C}+\frac{(14D-7A)(-4(n-3)+2C(n-4))}{(2-C)^2}\right.
\end{aligned}
$$

$$-\frac{7D-A}{(2-c)^3}\left(2\,(2+(n-4)(2-C))^2+4C\right)$$

$$-\frac{D}{(2-C)^4}\left(2\,(2+(n-4)(2-C))^3\right.$$

$$\left.+12(n-4)C(2-C)+16C(2+C))\right)\Bigg].$$

### 6.2.2 Approximationsformeln für einen $b$-BEC-Code

Im vorangehenden Abschnitt haben wir bereits erläutert, dass wir für eine näherungsweise Bestimmung der Restfehlerwahrscheinlichkeit eines $b$-BEC-Codes die Wahrscheinlichkeit berechnen, dass während der Übertragung ein Bündelfehler der Länge $> b$ unter der Bedingung generiert wird, dass der Markov'sche Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ in einem Muster $M_n \in \mathcal{M}$ genau eine *bad*-Phase erzeugt. Von daher haben wir mit Bezeichnung 6.1.22 die Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase ein Bündelfehler der Länge $> b$ erzeugt wird, zu bestimmen:

$$P(V > b)$$

Von daher wird wiederum vorausgesetzt, dass genau $Z_{anz} = 1$ *bad*-Phase in einem vom Markov'schen Hintergrundprozeß $(X_t)_{t\in\boldsymbol{N}}$ erzeugten Muster $M_n \in \mathcal{M}$ enthalten ist und damit $M_n \in \mathcal{M}(1)$ ist. Nach Bezeichnung 6.1.7 gibt der Zufallsvektor $\mathbf{Z}$ gerade die Länge $\mathbf{Z} = l$ der aufgetretenen *bad*-Phase an und diese genügt laut Bemerkung 6.1.9 der Bedingung $1 \le l \le n$.

Die bedingte Wahrscheinlichkeit, dass innerhalb einer *bad*-Phase der Länge $l$ ein Bündelfehler der Länge $> b$ generiert wird, haben wir in Abschnitt 6.1.2, Satz 6.1.25 ermittelt:

$$P(V > b \,|\, \mathbf{Z} = l) \;=\; 1 - \frac{l-b+2}{2^{l-b+1}}.$$

In Satz 6.1.16, Abschnitt 6.1.1, haben wir die Wahrscheinlichkeit für das Auftreten einer *bad*-Phase der Länge $l$ in einem Muster $M_n \in \mathcal{M}(1)$ angegeben:

$$P_{M_n}(\mathbf{Z} = l) \;=\; (q_{GG})^{n-3}\,\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{l-1}(2\,q_{GG}+(n-l-1)q_{BG}).$$

Von daher ergibt sich schließlich die Wahrscheinlichkeit $P(V > b)$ aus der Summe der Produkte $P_{M_n}(\mathbf{Z} = l)\,P(V > b \,|\, \mathbf{Z} = l)$ über alle $l$, mit $1 \le l \le n$, und wir erhalten in diesem Zuge eine erste Näherungsformel für die Restfehlerwahrscheinlichkeit eines $b$-BEC-Codes bezüglich der Menge $\mathcal{M}(1)$:

$$
\begin{aligned}
\tilde{\mathcal{R}}_{BEC}(b,n) &= P(V>b) \\
&= P_{M_n}(\mathbf{Z}=n)\,P(V>b\,|\,\mathbf{Z}=n) + \sum_{l=b+1}^{n-1} P_{M_n}(\mathbf{Z}=l)\,P(V>b\,|\,\mathbf{Z}=l) \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right) \\
&\quad + \sum_{l=b+1}^{n-1}(q_{GG})^{n-3}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{l-1}(2\,q_{GG}+q_{BG}(n-l-1)) \\
&\quad \left(1-\frac{l-b+2}{2^{l-b+1}}\right) \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right) \\
&\quad + (q_{GG})^{n-3}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{b} \\
&\quad \left[\sum_{l=0}^{n-b-2}(2\,q_{GG}+q_{BG}(n-l-b-2))\left(1-\frac{l+3}{2^{l+2}}\right)\left(\frac{q_{BB}}{q_{GG}}\right)^{l}\right] \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right) \\
&\quad + \frac{(q_{GG})^{n-3}}{4}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{b} \\
&\quad \left[\sum_{l=0}^{n-b-2}(2\,q_{GG}+q_{BG}(n-b-2)-q_{BG}\,l)\left(4-\frac{3+l}{2^{l}}\right)\left(\frac{q_{BB}}{q_{GG}}\right)^{l}\right].
\end{aligned}
$$

Da die Komplexität der Darstellung in diesem Fall geringer ist als für einen $r$-REC-Code berechnen wir mit den folgenden Vereinfachungen

$$
\begin{aligned}
A &:= 2\,q_{GG}+q_{BG}(n-b-2) \\
C &:= q_{BB}/q_{GG} \\
D &:= q_{BG}
\end{aligned}
$$

die endliche geometrische Reihe mit dem Ziel, eine geschlossene Darstellung der Näherungsformel für einen $b$-BEC-Code anzugeben:

$$
\begin{aligned}
\tilde{\mathcal{R}}_{BEC}(b,n) &= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right) \\
&\quad + \frac{(q_{GG})^{n-3}}{4}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{b}\left[\sum_{l=0}^{n-b-2}(A-D\,l)\left(4-\frac{3}{2^l}-\frac{l}{2^l}\right)C^l\right] \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right) \\
&\quad + \frac{(q_{GG})^{n-3}}{4}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{b} \\
&\quad \left[4A\sum_{l=0}^{n-b-2}C^l+3A\sum_{l=0}^{n-b-2}\left(\frac{C}{2}\right)^l-A\sum_{l=0}^{n-b-2}l\left(\frac{C}{2}\right)^l\right. \\
&\quad \left.-4D\sum_{l=0}^{n-b-2}lC^l+3D\sum_{l=0}^{n-b-2}l\left(\frac{C}{2}\right)^l+D\sum_{l=0}^{n-b-2}l^2\left(\frac{C}{2}\right)^l\right] \\
&= (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right) \\
&\quad + \frac{(q_{GG})^{n-3}}{4}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^{b} \\
&\quad \left[4A\,\frac{1}{1-C}\left(1-C^{n-b-1}\right)-3A\,\frac{2}{2-C}\left(1-\left(\frac{C}{2}\right)^{n-b-1}\right)\right. \\
&\quad -A\,\frac{4}{(2-C)^2}\,\frac{C}{2} \\
&\qquad \left.\left(1-(n-b-1)\left(\frac{C}{2}\right)^{n-b-2}+(n-b-2)\left(\frac{C}{2}\right)^{n-b-1}\right)\right. \\
&\quad -4D\,\frac{1}{(1-C)^2}\,C\,\left(1-(n-b-1)C^{n-b-2}+(n-b-2)C^{n-b-1}\right) \\
&\quad +3D\,\frac{4}{(2-C)^2}\,\frac{C}{2} \\
&\qquad \left.\left(1-(n-b-1)\left(\frac{C}{2}\right)^{n-b-2}+(n-b-2)\left(\frac{C}{2}\right)^{n-b-1}\right)\right.
\end{aligned}
$$

$$
\begin{aligned}
&+ D\,\frac{4C}{(2-C)^3}\,\left(1+\frac{C}{2}\right.\\
&\qquad\left.-\left(\frac{C}{2}\right)^{n-b-2}\left(1+(n-b-2)\left(1-\frac{C}{2}\right)\right)^2-\left(\frac{C}{2}\right)^{n-b-1}\right)\Bigg]
\end{aligned}
$$

$$
\begin{aligned}
= \;& (q_{BB})^{n-1}\frac{q_{GB}}{q_{GB}+q_{BG}}\left(1-\frac{n-b+2}{2^{n-b+1}}\right)\\
&+\frac{(q_{GG})^{n-3}}{4}\frac{q_{GB}q_{BG}}{q_{GB}+q_{BG}}\left(\frac{q_{BB}}{q_{GG}}\right)^b\\
&\Bigg[A\left(\frac{4}{1-C}+\frac{2C-6}{(2-C)^2}\right)\\
&+D\left(\frac{16C-4C^2}{(2-C)^3}-\frac{4C}{(1-C)^2}\right)\\
&+C^{n-b}\left(\frac{4D\,(n-b-1-C(n-b-2))}{C(1-C)^2}-\frac{4A}{(1-C)C}\right)\\
&+\left(\frac{C}{2}\right)^{n-b}\Bigg(\frac{12A}{(2-C)C}+\frac{4(A-3D)(2n-2b-2-C(n-b-2))}{(2-C)^2C}\\
&-\frac{4D\,((2+(n-b-2)(2-C))^2+C)}{(2-C)^3C}\Bigg)\Bigg]
\end{aligned}
$$

Für eine konkrete Auswertung der oben entwickelten Näherungsformeln für REC- und BEC-Codes betrachten wir einmal mehr exemplarisch einen $(63, 51)$-Blockcode, vgl. Kapitel 4 und Kapitel 5.

| mittlere Bündelfehlerlänge $E_b$ | exakte Restfehlerwahrscheinlichkeit | Näherung |
|---|---|---|
| 1,200020E+00 | 4,816451E-05 | 4.560830E-05 |
| 1,400020E+00 | 1,441523E-04 | 1.418930E-04 |
| 1,600020E+00 | 2,527816E-04 | 2.534673E-04 |
| 1,800020E+00 | 3,584184E-04 | 3.638801E-04 |
| 2,000020E+00 | 4,545973E-04 | 4.659085E-04 |
| 2,200020E+00 | 5,391897E-04 | 5.568815E-04 |
| 2,400020E+00 | 6,120919E-04 | 6.363407E-04 |

Tabelle 6.1: Näherung der Restfehlerwahrscheinlichkeit eines 2-REC-Codes für eine *bad*-Phase

In beiden Tabellen 6.1 und 6.2 ist jeweils die exakte Restfehlerwahrscheinlichkeit nach der Korrektur von 2 *random errors* bzw. nach der Korrektur aller Bündelfehler bis zur Länge 5 und eine Näherung mittels der oben angegebenen Approximationsformeln für variierende mittlere Bündelfehlerlänge $E_b$ aufgelistet. Anhand der Tabellen können wir die Genauigkeit der Näherung beispielhaft belegen. In Tabelle 6.1 erkennen wir, dass die Abweichung der Näherungsformel für einen 2-REC-Code bei ca. 10% liegt und in Tabelle 6.2 bestätigen sich die Ergebnisse aus Tabelle 6.1.

| mittlere Bündelfehlerlänge $E_b$ | exakte Restfehlerwahrscheinlichkeit | Näherung |
|---|---|---|
| 1,200020E+00 | 1,448275E-05 | 3.667719E-07 |
| 1,400020E+00 | 1,730060E-05 | 5.333682E-06 |
| 1,600020E+00 | 3,057582E-05 | 2.027374E-05 |
| 1,800020E+00 | 5,507892E-05 | 4.609413E-05 |
| 2,000020E+00 | 8,851053E-05 | 8.059029E-05 |
| 2,200020E+00 | 1,277259E-04 | 1.206826E-04 |
| 2,400020E+00 | 1,698725E-04 | 1.635633E-04 |

Tabelle 6.2: Näherung der Restfehlerwahrscheinlichkeit eines 5-BEC-Codes für eine *bad*-Phase

Tabelle 6.3 beinhaltet eine genauere Näherung der Restfehlerwahrscheinlichkeit. Diese Näherungswerte erreichen wir, wenn wir bei der Approximation der Restfehlerwahrscheinlichkeit auch vom Markov'schen Hintergrundprozeß $(X_t)_{t\in\mathbb{N}}$ erzeugte Muster $M_n \in \mathcal{M}(2)$ berücksichtigen, die zwei *bad*-Phasen umfassen. Dazu ist dann die Wahrscheinlichkeit für zwei *bad*-Phasen der Längen $\mathbf{Z} = (l_1, l_2)$ mit Satz 6.1.16 zu berechnen. Ferner haben wir die Wahrscheinlichkeit zu bestimmen, dass innerhalb dieser zwei *bad*-Phasen Bündelfehler der Länge $\leq 5$ generiert werden. Auf diese Art ist es möglich, die Abweichung von rund 10% auf ca. 1-2% zu senken. Eine solche Näherung ist für unsere Zwecke zur Bestimmung eines optimalen Fehler-korrigierenden Blockcodes ausreichend.

| mittlere Bündelfehlerlänge $E_b$ | exakte Restfehlerwahrscheinlichkeit | Näherung |
|---|---|---|
| 1,200020E+00 | 1,448275E-05 | 1.576035E-05 |
| 1,400020E+00 | 1,730060E-05 | 1.824357E-05 |
| 1,600020E+00 | 3,057582E-05 | 3.127592E-05 |
| 1,800020E+00 | 5,507892E-05 | 5.559367E-05 |
| 2,000020E+00 | 8,851053E-05 | 8.887902E-05 |
| 2,200020E+00 | 1,277259E-04 | 1.279778E-04 |
| 2,400020E+00 | 1,698725E-04 | 1.700317E-04 |

Tabelle 6.3: Näherung der Restfehlerwahrscheinlichkeit eines 5-BEC-Codes für zwei *bad*-Phasen

# Kapitel 7

# Markov Modelle für autoregressive Prozesse auf Signalebene

In Kapitel 2 haben wir die einzelnen Komponenten eines digitalen Datenübertragungssystems vorgestellt und uns dabei in erster Linie auf die Aufgabe des Kanalcodierers, die Fehlerkorrektur, konzentriert. In diesem Zusammenhang haben wir allerdings bereits erläutert, dass der Bitstrom, wie wir ihn bislang in den vorherigen Kapiteln analysiert haben, mit einem geeigneten Modulationsverfahren in solche Signale umgewandelt wird, wie sie das zur Übertragung ausgewählte physikalische Medium weiterleiten kann. Aus diesem Grund werden die Störungen in der Nachrichtentechnik häufig direkt auf der Signalebene beobachtet und analysiert. Dazu werden die von einer physikalischen Rauschquelle zufällig verursachten Störungen als ein kontinuierlicher Prozeß auf der Signalebene betrachtet. In der Nachrichtentechnik sind zur Modellierung dieses Störungsprozesses auf der Signalebene verschiedene Modelle entwickelt worden. An dieser Stelle seien beispielhaft der *Additiv White Gaussian Noise-Channel* (AWGN-Kanal) und der Rayleigh-Kanal genannt, die das Rauschen auf der Signalebene auf unterschiedliche Art darstellen.

Das AWGN-Kanalmodell wird in der Nachrichtentechnik häufig zum Vergleich verschiedener Codierverfahren genutzt. Dabei dient es lediglich zur Beschreibung kontinuierlicher zeitinvarianter Kanäle; d.h., dass die Störung eines derartigen Kanals lediglich aus der additiven Überlagerung von mittelwertfreiem, weißem, Gaußschem Rauschen besteht und die Störungen dabei von Signal zu Signal unkorreliert sind. In der Regel sind reale Kanäle, wie beispielsweise Mobilfunkkanäle, jedoch zeitvariante Kanäle, deren Eigenschaften sich abhängig von der Zeit während der Übertragung verändern. Bei der Modellierung derartiger Kanäle wird stark vereinfacht, indem lediglich die elementaren Parameter, wie die Verteilung der Amplitude und deren zeitliche Korrelation und damit das Gedächtnis des betrachteten Kanals, in Betracht zieht. Der Rayleigh-Kanal ist ein geeignetes Beispiel für einen AWGN-Kanal, der eine zeitlich variierende Amplitude besitzt. Hier kommt zu der additiven Störung des AWGN-Kanals noch eine multiplikative, Rayleigh-verteilte Störung hinzu, so dass die Störungen aufeinander folgender Kanalamplituden korreliert sind. Zum Vergleich verschiedener Codierverfahren ist der AWGN-Kanal keine geeignete Basis, da er das Gedächtnis des betrachteten Kanals nicht mitberücksichtigt, und der Rayleigh-Kanal ist zu komplex, da ein weiterer zufällig verteilter Parameter nichtlinear auftritt.

In diesem Kapitel werden wir einen anderen Ansatz zur Modellierung der Störungen auf der Signalebene vorstellen, um eine Bewertung verschiedener Codierverfahren in Abhängigkeit

von der Korrelation der Amplituden durchführen und die Vereinfachungen des Rayleigh-Kanals vermeiden zu können. Daher betrachten und analysieren wir den Fehlerprozeß bzw. die zufälligen Störungen in diesem Kapitel auf der Signalebene. In Abschnitt 7.1 stellen wir die zufälligen Störungen auf der Signalebene als einen autoregressiven Prozeß dar und werden dann vor diesem Hintergrund eine Möglichkeit zur Bewertung verschiedener Codierverfahren eröffnen. Abschließend präsentieren wir in Abschnitt 7.3 einen Vergleich von verschiedenen Fehler-korrigierenden Blockcodes und werden damit die Ergebnisse der vorherigen Kapitel bestätigen.

## 7.1 Darstellung von autoregressiven Prozessen durch endliche Markov-Modelle

In diesem Abschnitt nutzen wir für die mathematische Beschreibung der Amplitude des Kanalrauschens auf der Signalebene autoregressive Prozesse. Ferner zeigen wir, inwieweit dieser Ansatz z.B. AWGN-Kanäle beinhaltet und somit eine Erweiterung des AWGN-Kanals auf korrelierte Signale darstellt. Mit Hilfe einer Diskretisierung der Verteilung der Amplitude des Kanalrauschens schaffen wir eine geeignete Voraussetzung, das von uns in Abschnitt 3.4 entwickelte Kanalmodell in diesem Kontext anzuwenden; d.h. am Beispiel dieser autoregressiven Prozesse verdeutlichen wir im folgenden, wie die Anpassung von semi-Markov Prozessen bzw. die Anpassung eines endlichen Markov Modells an derartige Prozesse erfolgt. Schließlich ist autoregressiven Prozessen erster Ordnung und semi-Markov Prozessen gemeinsam, dass die vollständige Information über das vergangene Geschehen, die zur Bestimmung der zukünftigen Entwicklung des Prozesses notwendig ist, im gegenwärtigen Zustand des Prozesses zusammengefaßt ist. Auf der Basis dieses Ansatzes ermöglichen wir die Berechnung der Restfehlerwahrscheinlichkeit verschiedener Fehler-korrigierender Blockcodes anhand der rekursiven Gleichungen, die wir in Abschnitt 4.2.2 hergeleitet haben, wie wir dies bereits in unserer Arbeit [42] präsentiert haben.

### 7.1.1 Definition

*Ein autoregressiver Prozeß erster Ordnung wird durch einen reellwertigen zufälligen Prozeß $Y_t$, $t \in \mathbb{N}_0$, beschrieben. Dieser wird definiert durch*

$$Y_{t+1} = \kappa Y_t + \sqrt{1-\kappa^2}\,\sigma\epsilon_t + \mu(1-\kappa)\,,$$

*wobei $Y_0 \sim \mathcal{N}(\mu, \sigma^2)$ Gauß-verteilt mit Erwartungswert $\mu \in \mathbb{R}$ und der Varianz $\sigma \in \mathbb{R}^+$ und $\epsilon_t \sim \mathcal{N}(0,1)$ normalverteilt ist. Es wird ferner angenommen, dass alle Zufallsvariablen $\epsilon_t$ unabhängig voneinander sind.*

### 7.1.2 Bemerkung

Aus der obigen Definition ergibt sich, dass die Zufallsvariablen $Y_t$ zu jedem Zeitpunkt $t \in \mathbb{N}$ stets Gauß-verteilt sind, d.h. $Y_t \sim \mathcal{N}(\mu, \sigma^2)$, und die Autokorrelationsfunktion $\mathcal{K}_n$ des Prozesses $Y_t$ geometrisch fallend ist. Sie ist bestimmt durch

$$\mathcal{K}_n := \frac{E\Big((Y_t-\mu)(Y_{t+n}-\mu)\Big)}{\sigma^2} = \kappa^n \qquad \forall\, n \in \mathbb{N}\,.$$

Die Autokorrelation $\kappa$ des Prozesses reicht daher auf der einen Seite vom gedächtnislosen Fall, der durch $\kappa = 0$ beschrieben wird, bis zum stark korrelierten Fall auf der anderen Seite, der für $\kappa \to 1$ angenommen wird. Dabei stellt der gedächtnislose Fall für $\kappa = 0$ gerade einen AWGN-Kanal dar.

Bei der Beschreibung des Kanalrauschens durch einen solchen autoregressiven Prozeß interpretieren wir die Zufallsvariable $Y_t$ als die Amplitude des Kanalrauschens. Diese wird in dB auf einer logarithmischen Skala gemessen. Das Signal, welches die Information trägt, besitze eine konstante Amplitude $S$. Dann wird der Fehlerprozeß $E_t$ festgelegt durch

$$P(E_t = 1 \,|\, Y_t < S) = 0 \text{ und } P(E_t = 1 \,|\, Y_t \geq S) = 1/2\,.$$

Von daher treten Fehler mit Wahrscheinlichkeit 1/2 auf, wenn die konstante Amplitude $S$ des Signals kleiner als die Amplitude des Kanalrauschens ist. Dabei ist klar, dass der Modellbildung eine Vereinfachung zugrunde liegt, da die Fehlerwahrscheinlichkeit in der Realität keinen Sprung machen wird, wie er im Modell an der Stelle auftritt, an der das Signal/Rausch-Verhältnis gleich 1 ist.

### 7.1.3 Bemerkung

Unter den obigen Annahmen gilt für die Bitfehlerrate $p_E$ des Fehlerprozesses $E_t$

$$p_E = \frac{1}{2} P(Y_t \geq S)\,.$$

Damit ist sie gleich der halben Wahrscheinlichkeit, dass eine $\mathcal{N}(0,1)$-verteilte Zufallsvariable den Wert $(S - \mu)/\sigma$ übersteigt.

Autoregressive Prozesse stellen somit ein Modell für zeitvariante Kanäle dar und bilden als kontinuierlicher Prozeß das Kanalrauschen auf der Signalebene ab. Eine Parametrisierung eines derartigen autoregressiven Fehlerprozesses auf der Grundlage von Messdaten wird in [24] beschrieben. Dort wird auch ein analytischer Ansatz zur Bestimmung der Restfehlerwahrscheinlichkeit im Hinblick auf einen derartigen autoregressiven Fehlerprozeß erläutert, der sich allerdings für die Anwendung, d.h. für eine Auswertung der Restfehlerwahrscheinlichkeiten verschiedener Fehler-korrigierender Codes, als zu komplex erwies. Daher wird auch in [24] eine Simulation zur Bewertung verschiedener Codes anhand deren Restfehlerwahrscheinlichkeit verwendet. Die im folgenden beschriebene Approximation eines derartigen autoregressiven Fehlerprozesses durch einen semi-Markov Prozeß mit endlichem Zustandsraum bzw. durch ein endliches Markov Modell führt zu einer berechenbaren Abschätzung der Restfehlerwahrscheinlichkeit. Zudem wird durch eine Erweiterung des Zustandsraumes eine merkliche Verbesserung der Abschätzung erreicht werden.

Wie eingangs bereits erwähnt, beinhaltet $Y_t$ für autoregressive Prozesse alle Information über die Vergangenheit des Prozesses. Zur Approximation eines solchen Prozesses wählen wir die $N$ Zustände eines Markov'schen Hintergrundprozesses mit dem endlichen Zustandsraum $\mathcal{S}$ aus Abschnitt 4.2.2. Zu diesem Zweck partitionieren wir die Verteilung von $Y_t$ in $N$ Intervalle der Länge $\gamma = 2y\sigma/N$ und identifizieren diese mit den $N$ Zuständen des Markov'schen Hintergrundprozesses. Dabei beschränken wir uns auf solche Werte, für die gilt:

$$\mu - y\sigma \;\leq\; Y_t \;\leq\; \mu + y\sigma\,.$$

Dabei ist der Parameter $y$ so zu bestimmen, dass die abgeschnittenen Teile der Gaußverteilung von $Y_t$ vernachlässigt werden können.

Ein Zustand $i \in \mathcal{S}$ des Markov'schen Hintergrundprozesses $(X_t)_{t\in\mathbb{N}}$ wird dann gegeben durch

$$X_t = i \;\Leftrightarrow\; Y_t - \mu + y\sigma \in [(i-1)\delta, i\delta)\,.$$

Die stationären Zustandswahrscheinlichkeiten ergeben sich dann zu

$$\pi_i = \phi(i\delta) - \phi((i-1)\delta) \quad \text{für alle } i \in \mathcal{S}.$$

Dabei ist die Abbildung $\phi$ definiert durch

$$\phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}}\, e^{-\frac{1}{2}x^2} dx\,.$$

Die Genauigkeit der Approximation ist von der Anzahl $N$ der Zustände des Markov'schen Hintergrundprozesses abhängig. Um die Übergangsmatrix $\mathbf{Q} = (q_{ij})$ des zugrunde liegenden Markov'schen Hintergrundprozesses zu bestimmen, ermitteln wir zunächst die bedingten Wahrscheinlichkeiten der Zustandsübergänge abhängig von $Y_t$:

$$\begin{aligned} p_i(x) &:= P(X_{t+1} = i \,|\, Y_t = x) \\ &= \phi\left(\frac{i\gamma - y\sigma - \kappa(x-\mu)}{\sqrt{1-\kappa^2}\sigma}\right) - \phi\left(\frac{(i-1)\gamma - y\sigma - \kappa(x-\mu)}{\sqrt{1-\kappa^2}\sigma}\right) \end{aligned}$$

Daraus ergibt sich für die Einträge der Übergangsmatrix $\mathbf{Q}$:

$$\begin{aligned} q_{ij} &:= P(X_{t+1} = i \,|\, X_t = i) \\ &= \frac{\displaystyle\int_{\mu-y\sigma+(i-1)\delta}^{\mu-y\sigma+i\delta} g(x)p_j(x)\,dx}{\phi\left(\frac{i\delta}{\sigma} - y\right) - \phi\left(\frac{(1-i)\delta}{\sigma} - y\right)} \end{aligned}$$

Dabei ist $g(x) := 1/\sqrt{2\pi\sigma}\, e^{-(x-\mu)^2/(2\sigma)^2}$.

Auf diese Weise haben wir eine geeignete Anpassung eines endlichen Markov Modells an einen autoregressiven Fehlerprozeß entwickelt.

## 7.2 Interleaving

Da wir auch in diesem Kapitel das Ziel verfolgen, verschiedene Codierverfahren mit unterschiedlichem Fehlerkorrekturverhalten, also REC- und BEC-Codes, bezüglich ihrer Leistungsfähigkeit und Effizienz miteinander zu vergleichen, untersuchen wir in diesem Kontext die Auswirkungen des *Interleavings* auf die Restfehlerwahrscheinlichkeit. Hierbei stellt *Interleaving* eine beliebte Methode dar, um während der Übertragung aufgetretene Bündelfehler künstlich in *random errors* umzuwandeln, und wird zur Fehlerkorrektur häufig in Verbindung mit Faltungscodes zum Einsatz gebracht. Die Codewörter werden dabei nicht sequentiell

sondern praktisch parallel übertragen. Dazu werden sie der Reihe nach zeilenweise in eine Matrix mit $d$ Spalten geschrieben und dann zur Übertragung spaltenweise wieder ausgelesen. Auf diese Weise werden die einzelnen Bits eines Codewortes mit dem Abstand $d$ voneinander übertragen und damit auch nur jedes $d$-te Bit eines Codewortes während der Übertragung verfälscht. Hierbei ist der Abstand $d$ durch die technische Umsetzung begrenzt, denn erstens verursacht Interleaving eine zeitliche Verzögerung der Übertragung und zweitens steigt bei wachsendem Abstand $d$ der Speicherbedarf des Empfängers.

Der Einfluß des *Interleavings* auf die Restfehlerwahrscheinlichkeit eines Fehlerkorrigierenden Codes kann berechnet werden, indem die Übergangsmatrix $\mathbf{Q}$ durch die Matrix $\mathbf{Q}^d = (q_{ij})^{(d)}$ ersetzt wird. Diese besitzt die Einträge

$$q_{ij}^{(d)} = P(X_{t+d} = j \,|\, X_t = i)\,.$$

Die Matrix $\mathbf{Q}^d$ kann durch eine einfache Matrixmultiplikation aus $\mathbf{Q}$ erzeugt werden. Zum Beweis dieser Aussage verweisen wir auf eine Arbeit von [50].

Nun befassen wir uns in diesem Kapitel mit einem autoregressiven Kanalmodell, so dass in diesem Zusammenhang zu untersuchen ist, ob nach der Anwendung eines *Interleavers* aus dem ursprünglichen autoregressiven Fehlerprozeß $Y_t$ wiederum ein autoregressiver Prozeß entsteht.

### 7.2.1 Satz

*Das vorgestellte autoregressive Kanalmodell besitzt die Eigenschaft, dass Interleaving erneut einen autoregressiven Prozeß $\tilde{Y}_t = Y_{td}$ verursacht. Dieser besitzt dann einen verkleinerten Korrelationsparameter $\tilde{\kappa} = \kappa^d$.*

**Beweis**

Wir erhalten für den Prozeß $\tilde{Y}_t$:

$$\begin{aligned}
\tilde{Y}_t &= Y_{dt} \\
&= \kappa Y_{dt-1} + \sqrt{1-\kappa^2}\,\sigma\epsilon_{dt} + \mu(1-\kappa) \\
&= \kappa\left(\kappa Y_{dt-2} + \sqrt{1-\kappa^2}\,\sigma\epsilon_{dt-1} + \mu(1-\kappa)\right) + \sqrt{1-\kappa^2}\,\sigma\epsilon_{dt} + \mu(1-\kappa) \\
&= \ldots \\
&= \kappa^d Y_{d(t-1)} + \sum_{n=0}^{d-1} \kappa^n \left(\sqrt{1-\kappa^2}\,\sigma\epsilon_{dt-n} + \mu(1-\kappa)\right) \\
&= \kappa^d \tilde{Y}_{t-1} + \sqrt{1-\kappa^{2d}}\,\sigma\tilde{\epsilon}_t + \mu(1-\kappa^d)\,.
\end{aligned}$$

Dabei ist

$$\tilde{\epsilon}_t := \sum_{n=0}^{d-1} \kappa^n \sqrt{\frac{1-\kappa^2}{1-\kappa^{2d}}}\,\epsilon_{dt-n}\,.$$

Da alle Zufallsvariablen $\epsilon_t$ unabhängig und $\mathcal{N}(0,1)$-verteilt sind, folgt:

$$\tilde{\epsilon}_t \sim \mathcal{N}\left(0, \sum_{n=0}^{d-1}\left(\kappa^n\sqrt{\frac{1-\kappa^2}{1-\kappa^{2d}}}\right)^2\right) \quad \Rightarrow \quad \tilde{\epsilon}_t \sim \mathcal{N}(0,1)\,.$$

Zusammenfassend läßt sich feststellen, dass durch Interleaving eine Verminderung $\kappa \longrightarrow \kappa^d$ des Autokorrelationskoeffizienten $\kappa$ des autoregressiven Kanalmodells hervorgerufen wird. □

## 7.3 Anpassung des Modells und Bewertung von Codierverfahren

In Abschnitt 5.1.1 haben wir anhand von Tabelle 5.1 beispielhaft die Existenz von REC- und BEC-Codes der gleichen Codelänge $n = 63$ und der selben Dimension $k = 51$ nachgewiesen. In diesem Abschnitt werden wir daher nun exemplarisch für diese Fehler-korrigierenden Codes die Restfehlerwahrscheinlichkeiten auf der Basis dieses autoregressiven Kanalmodells bestimmen, welches wir auf die oben beschriebene Weise durch ein endliches Markov Modell approximieren, wie wir dies in ähnlicher Form bereits in [42] präsentiert haben.

Die Amplitude des Kanalrauschens wird durch diskrete Markov-Modelle mit $N = 24\,, 48$ oder 96 Zuständen beschrieben. In unserem Beispiel wird die Amplitude des Kanalrauschens in dem Intervall

$$[\mu - y\sigma, \mu + y\sigma]$$

dargestellt, wobei der Faktor $y$ gerade zu $y = 5$ gewählt ist. Dieses Intervall wird nun entsprechend der Anzahl der Zustände des Markov Modells in $N$ äquidistante Intervalle zerlegt und jedes dieser Teilintervalle wird mit jeweils einem Zustand eines Markov Modells identifiziert.

Sowohl die Amplitude des Signals als auch diejenige des Kanalrauschens wird in dB gemessen und auf einer logarithmischen Skala abgetragen. Nun gehen wir von der Annahme aus, dass die Amplitude $S$ des Signals konstant ist. Dann wird der Fehlerprozeß bestimmt durch

$$P(E_t = 1 \,|\, Y_t < S) = 0 \text{ und } P(E_t = 1 \,|\, Y_t \geq S) = 1/2\,.$$

Demzufolge erzeugen genau die Zustände des Markov'schen Hintergrundprozesses, die gerade das Intervall $[S, \mu + 5\sigma]$ darstellen, Fehler mit Wahrscheinlichkeit $1/2$, während demgegenüber die übrigen Zustände fehlerfreie Zustände sind. Aus diesem Grund sollte die Amplitude $S$ des Signals so gewählt sein, dass sie auf einer der Grenzen des Intervalls zwischen den Teilintervallen liegt, die gerade mit einem Zustand des Markov'schen Hintergrundprozesses identifiziert werden. Wir erreichen dies, indem wir den Faktor $y$ zur Skalierung geeignet wählen.

Nach Bemerkung 7.1.3 ist die Bitfehlerrate $p_E$ des autoregressiven Prozesses gerade gegeben durch

$$p_E = \frac{1}{2}\, P(Y_t \geq S)$$

und ist damit gleich der halben Wahrscheinlichkeit, dass eine normalverteilte Zufallsvariable die Grenze $(S - \mu)/\sigma$ überschreitet. In unserem Fall erhalten wir für die Markov Modelle mit $N = 24\,, 48$ bzw. 96 Zuständen gerade die Fehlerraten $p_E = 2.2 \cdot 10^{-4}$, $4.4 \cdot 10^{-5}$ und $8.3 \cdot 10^{-6}$ für die Werte $(S - \mu)/\sigma = 3.33$, 3.75 und 4.17.

In Tabelle 7.1 wird die Wahrscheinlichkeit des Auftretens eines Fehlers in einem empfangenen Wort in Abhängigkeit von der Bitfehlerrate $p_E$ und dem Autokorrelationskoeffizienten $\kappa$ präsentiert. Eine Erhöhung des Autokorrelationskoeffizienten führt dazu, dass die aufgetretenen Übertragungsfehler in stärkerem Maße in Bündelfehlern auftreten; d.h. dies führt zu einer Vergrößerung der Bündelfehlerlänge. Daher werden bei einer konstanten Bitfehlerrate $p_E$ und wachsender Korrelation weniger Codewörter gestört. Dies zeichnet sich deutlich in den Spalten der Tabelle 7.1 ab.

| Bitfehlerrate<br>Autokorrelation | $8.3 \cdot 10^{-6}$ | $4.4 \cdot 10^{-5}$ | $2.2 \cdot 10^{-4}$ |
|---|---|---|---|
| $\kappa$=0.5 | 5.099 E-04 | 2.770 E-03 | 1.316 E-02 |
| $\kappa$=0.6 | 5.043 E-04 | 2.727 E-03 | 1.284 E-02 |
| $\kappa$=0.7 | 4.917 E-04 | 2.640 E-03 | 1.230 E-02 |
| $\kappa$=0.8 | 4.624 E-04 | 2.456 E-03 | 1.125 E-02 |
| $\kappa$=0.9 | 3.868 E-04 | 2.012 E-03 | 8.932 E-03 |
| $\kappa$=0.95 | 2.965 E-04 | 1.512 E-03 | 6.541 E-03 |
| $\kappa$=0.99 | 1.356 E-04 | 6.810 E-04 | 2.912 E-03 |

Tabelle 7.1: Wahrscheinlichkeit eines Fehlers in einem empfangenen Wort

In Tabelle 7.2 wird die Relation zwischen der Güte des Markov-Modells und der Anzahl der Zustände des Markov'schen Hintergrundprozesses verdeutlicht. Hierzu wird die Wahrscheinlichkeit für das Auftreten eines Fehlers in einem empfangenen Wort einmal mehr für eine Bitfehlerrate von $p_E = 4.4 \cdot 10^{-5}$ berechnet, und zwar für Markov Modelle mit $N = 24$, 48 oder 96 Zuständen. Es zeigt sich, dass eine Erhöhung der Anzahl der Zustände von $N = 24$ auf $N = 96$ zwar zu einer Verbesserung der Güte des Markov-Modells führt, die relativen Näherungen liegen bei einer Erhöhung von $N$ allerdings höchstens in einer Größenordnung von ca. 4%. Allerdings wachsen die Abweichungen mit einem steigenden Korrelationsfaktor. Diese Beobachtung bestätigt sich qualitativ auch bei Berechnungen mit $N = 192$ und $N = 384$ Zuständen.

| Anzahl der Zustände<br>Autokorrelation | N=24 | N=48 | N=96 |
|---|---|---|---|
| $\kappa$ =0.5 | 2.758 E-03 | 2.770 E-03 | 2.773 E-03 |
| $\kappa$ =0.6 | 2.715 E-03 | 2.727 E-03 | 2.731 E-03 |
| $\kappa$ =0.7 | 2.626 E-03 | 2.640 E-03 | 2.644 E-03 |
| $\kappa$ =0.8 | 2.442 E-03 | 2.456 E-03 | 2.460 E-03 |
| $\kappa$ =0.9 | 2.011 E-03 | 2.012 E-03 | 2.011 E-03 |
| $\kappa$ =0.95 | 1.548 E-03 | 1.512 E-03 | 1.499 E-03 |

Tabelle 7.2: Genauigkeit der Approximation

Ein ähnlicher Grad an Genauigkeit der Approximation kann bei der Berechnung der Restfehlerwahrscheinlichkeit beobachtet werden. Daher kann davon ausgegangen werden, dass ein Markov-Modell mit $N = 48$ Zuständen eine hinreichend genaue Auswertung der Restfehlerwahrscheinlichkeit zuläßt. Bei den Berechnungen zu Tabelle 7.2 und den Abbildungen 7.1 bis 7.3 ist ein solches Markov-Modell zugrunde gelegt worden.

In Abbildung 7.1 bis 7.3 wird ein Vergleich zwischen der Leistung eines 2-REC-Codes und derjenigen eines 5-BEC-Codes für unterschiedliche Korrelationsfaktoren präsentiert. Dabei ist die Bitfehlerrate $p_E$ fest vorgegeben jeweils mit $p_E = 8.3 \cdot 10^{-6}$, $4.4 \cdot 10^{-5}$ bzw. $2.2 \cdot 10^{-4}$.

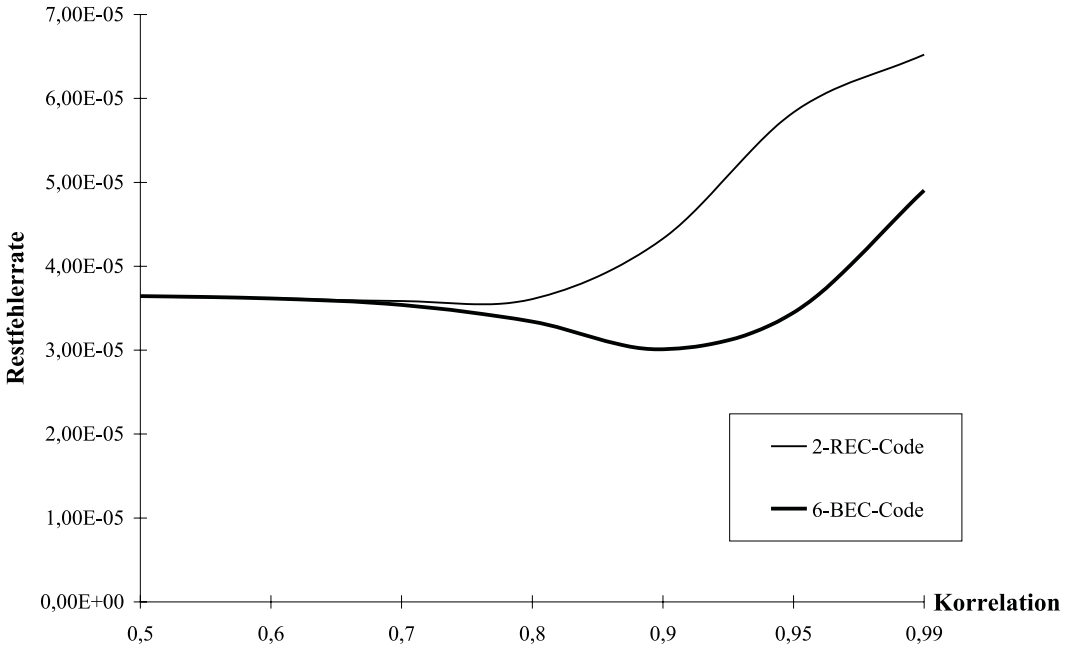


Abbildung 7.1: Vergleich der Restfehlerwahrscheinlichkeit für REC- und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 8.3 \cdot 10^{-6}$

Betrachten wir beispielsweise Abbildung 7.2, so zeigt sich, dass in dem Bereich $\kappa \in [0.5, 0.65]$ die Qualität von 2-REC-Codes und 5-BEC-Codes nahezu identisch ist. Bei einer stärkeren Korrelation ist die Restfehlerwahrscheinlichkeit eines 5-BEC-Codes annähernd halb so groß wie diejenige eines 2-REC-Codes. Der Einsatz eines 5-BEC-Codes ist wesentlich effizienter, sobald $\kappa > 0.65$ gilt. Für eine stärkere Korrelation zeigen sich hier die deutlichen Vorteile für einen 5-BEC-Code. Betrachten wir den unkorrelierten Fall für $\kappa = 0$, der gerade den AWGN-Kanal darstellt, so erhalten wir wesentlich bessere Resultate für den REC-Code.
Die Abbildungen 7.1 und 7.3 bestätigen diese Resultate und zeigen für die Restfehlerwahrscheinlichkeiten von 2-REC- und 5-BEC-Codes einen ähnlichen Verlauf.

Ferner unterstützen diese Ergebnisse unsere Resultate aus den vorangehenden Kapiteln. In Kapitel 5 haben wir gezeigt, dass der Einsatz eines BEC-Codes dem eines REC-Codes vorzuziehen ist, falls die mittlere Bündelfehlerlänge $E_b$ eine Grenze von ungefähr 1.3 überschreitet. Übertragen auf dieses Kapitel halten wir fest, dass BEC-Codes beim Einsatz zur Fehlerkorrektur zu bevorzugen sind, falls der Korrelationskoeffizient $\kappa$ einen Wert von ca. 0.65 überschreitet. Indem wir sämtliche Fehlerzustände eines in diesem Kapitel zur Approximation verwendeten Markov Modells zu einem Fehlerzustand zusammenfassen und ebenso mit den fehlerfreien Zuständen verfahren, ermöglichen wir einen direkten Vergleich der Ergebnisse auf der Basis des Modells von Gilbert in Abschnitt 5.2.1. Auf diese Weise zeigt sich eine direkte Beziehung zwischen den Ergebnissen dieses Abschnittes und denen aus Abschnitt 5.2.1. Es wird deutlich, dass es eine direkte Beziehung zwischen dem Korrelationskoeffizienten und der mittleren Bündelfehlerlänge $E_b$ gibt, was sich an den Grenzwerten $E_b = 1.3$ und $\kappa = 0.65$ manifestiert, für die die Effizienz eines BEC-Codes größer ist. Diesen

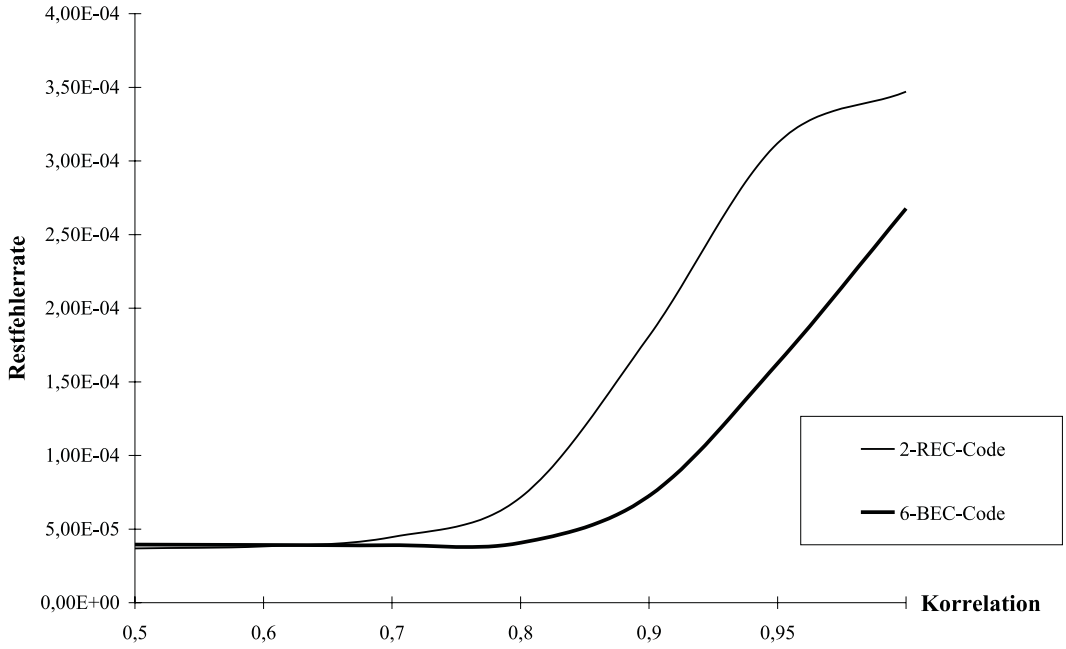


Abbildung 7.2: Vergleich der Restfehlerwahrscheinlichkeit für REC- und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 4.4 \cdot 10^{-5}$

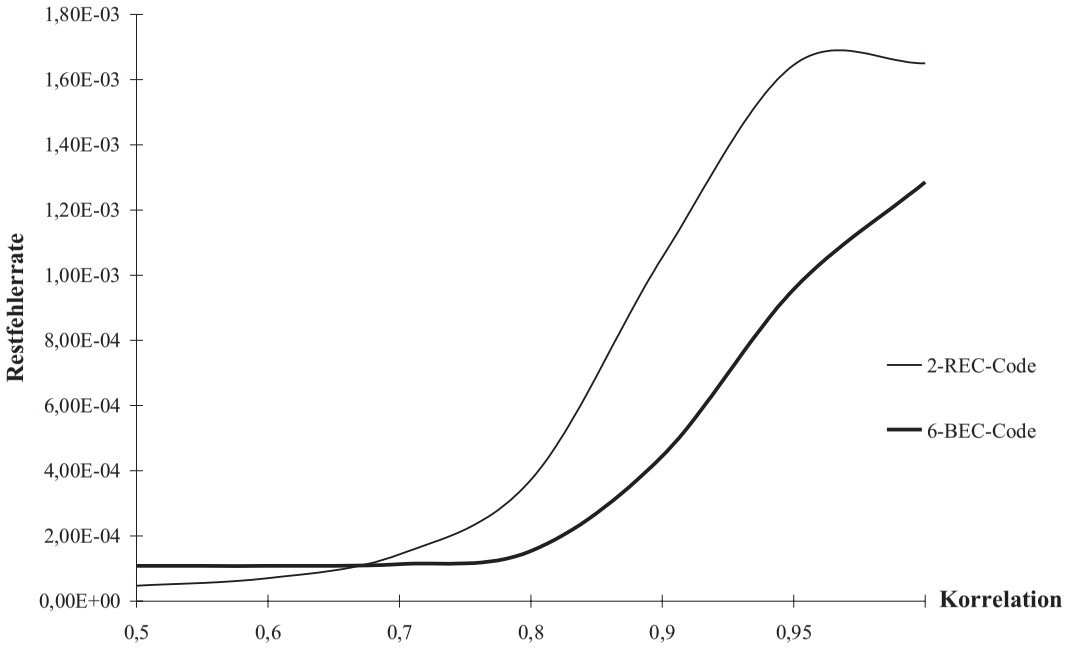


Abbildung 7.3: Vergleich der Restfehlerwahrscheinlichkeit für REC- und BEC-Codes: Codelänge $n = 63$, Bitfehlerrate $p_E = 2.2 \cdot 10^{-4}$

funktionalen Zusammenhang zwischen dem Korrelationskoeffizienten $\kappa$ und der mittleren Bündelfehlerlänge $E_b$ nicht nur auf einer empirischen Ebene aufzudecken, sondern auch auf einer theoretischen Ebene nachzuweisen, ist eine äußerst interessante Fragestellung, die es in der Zukunft zu untersuchen gilt, vgl. hierzu auch Kapitel 8.

# Kapitel 8

# Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit sind zur Modellierung diskreter Kanäle mit Gedächtnis und zur Beurteilung verschiedener Fehler-korrigierender Codierverfahren geeignete Konzepte und Methoden entwickelt worden. So ist in Kapitel 3 Abschnitt 3.4 ein Modell präsentiert worden, mit welchem diskrete Kanäle mit Gedächtnis und damit Übertragungsfehler unterschiedlicher Form und verschiedenen Ausmaßes, also sowohl *random* als auch *burst errors* in entsprechender Weise abgebildet werden können. Dabei werden in dem in Abschnitt 3.4 vorgestellten Kanalmodell die Stärken verschiedener, bereits in der Literatur bekannter Kanalmodelle miteinander verknüpft. Gleichzeitig werden dabei deren Schwachstellen weitgehend eliminiert. Da das entworfene Konzept die Möglichkeit bietet, zeitliche Abhängigkeiten zwischen aufgetretenen Übertragungsfehlern in angemessener Weise zu modellieren, konnten in Kapitel 4 Methoden entwickelt werden, so dass bei der Gegenüberstellung verschiedener Fehler-korrigierender Codes deren unterschiedliches Fehlerkorrekturverhalten in angemessener Form in die Beurteilung miteinbezogen werden kann. Zwecks der Beurteilung verschiedener Fehler-korrigierender Verfahren ist in Kapitel 4 die Restfehlerwahrscheinlichkeit eines Codes als ein geeignetes Maß eingeführt worden. Basierend auf dem in Abschnitt 3.4 vorgestellten Modell ist in Abschnitt 4.2 schließlich ein Verfahren zur Berechnung der Restfehlerwahrscheinlichkeit eines Codes in Form rekursiver Gleichungen entworfen worden. Bei der Herleitung dieser rekursiven Gleichungen sind die unterschiedlichen Korrektureigenschaften verschiedener Codes berücksichtigt worden, so dass zwischen REC- und BEC-Codes differenziert worden ist. In der Literatur werden zwar ebenfalls unterschiedliche Codierverfahren vorgestellt, doch eine angemessene Differenzierung in REC- und BEC-Codes wird hier nicht vorgenommen. Folglich können die Vorteile und Stärken, die die BEC unter gewissen Übertragungsbedingungen besitzt, nie angemessen herausgestellt und beurteilt werden. Entgegen den Ansätzen in der Literatur gelingt es, auf der Basis der in Kapitel 3 und 4 entwickelten Konzepte und Methoden REC- und BEC-Codes einander gegenüber zu stellen, bezüglich ihrer Effizienz zu bewerten und schließlich einen für die gegebenen Bedingungen optimalen Fehler-korrigierenden Code zu bestimmen. Bei den Auswertungen in Kapitel 5 bestätigt sich die Vermutung, dass der Einsatz eines BEC-Codes zur Fehlerkorrektur dem eines REC-Codes vorzuziehen ist, sobald das Gedächtnis des betrachteten Übertragungskanals eine gewisse Grenze übersteigt. Auf diese Weise können in Kapitel 5 auf der Grundlage der Fehlerstatistik eines betrachteten Kanals optimale Fehler-korrigierende Codes mittels der entworfenen Methoden und Konzepte bestimmt werden. Aufgrund der in der Arbeit erzielten Resultate erscheint eine konkrete Anwendung des entwickelten Schemas zur Bewertung verschiedener Fehlerkorrekturverfahren erfolgversprechend. Hieraus ergeben

sich die folgenden interessanten Aspekte für weitere Untersuchungen und Analysen.

Als erstes schließt sich hier die Frage nach der Validierung des Modells und damit nach der Anpassung der Modellparameter an geeignete reale Messdaten an. Bislang hat sich im Rahmen des gemeinsamen Projektes mit der DeTeSystem keine Möglichkeit ergeben, derartige Daten zu erheben, obschon diese Fragestellung mit verschiedenen Unternehmen aus dem Bereich der Telekommunikation sowie mit anderen Wissenschaftlern erörtert worden ist, die auf diesem bzw. einem angrenzenden Gebiet arbeiten. Im Verlauf dieser Diskussionen haben sich die damit verbundenen Schwierigkeiten stärker herauskristallisiert. So ist es nicht möglich, den harten Bitstrom, den sogenannten *Hard-Output*, aus einer Übertragung über einen realen Kanal abzugreifen. Ferner sind die in Kapitel 2 dargestellten Komponenten eines digitalen Datenübertragungssystems, der Demodulator und der Kanaldecodierer, in der Realität miteinander verschmolzen und stellen nach außen eine Art *black box* dar. Aus dieser kann der harte Bitstrom bisher noch nicht ausgelesen werden. Was dagegen während der Übertragung ausgelesen werden kann, ist der sogenannte *Soft-Output*. Hier ist das übertragene Bit mit einer sogenannten Zuverlässigkeitsinformation versehen, genauer mit der Wahrscheinlichkeit, mit der das empfangene Bit den angegebenen Wert annimmt. Daraus ergeben sich die folgenden interessanten Aufgabenstellungen für die Zukunft. Einerseits ist zu analysieren, inwiefern es technisch möglich ist, diese *black box* aufzuspalten und den harten Bitstrom abzugreifen. Andererseits ist zu untersuchen, inwieweit das vorgestellte Kanalmodell an den *Soft-Output* angepasst und entsprechend erweitert und modifiziert werden kann, so dass anhand des *Soft-Outputs* eine Anpassung der Modellparameter vorgenommen werden kann. Erste Lösungsansätze hierzu beinhaltet Kapitel 7. Hier ist ein autoregressiver Fehlerprozeß mittels des in Abschnitt 3.4 präsentierten endlichen Markov Modells approximiert worden. Auf ähnliche Art und Weise könnte auch die Behandlung von *Soft-Ouputs* möglich sein.

Als ein weiterer Aspekt für künftige Arbeiten erscheint die konkrete Anwendung der vorgestellten Konzepte und Verfahren auf unterschiedlichen Netzwerkebenen vielversprechend. So haben wir zum Abschluß von Kapitel 2 bereits darauf verwiesen, dass das in Abschnitt 3.4 vorgeschlagene Kanalmodell auch auf einer anderen Ebene in einem digitalen Datenübertragungssystem Verwendung finden kann. Schließlich führen die entwickelten Verfahren zu einem allgemeinen Modell inklusive Fehlererkennung und Fehlerkorrektur, mit dem verschiedene Schichten bzw. Ebenen eines Netzwerkes, so z.B. die *Paket-* bzw. *Frame-Ebene*, betrachtet werden können. Die folgenden allgemeinen theoretischen Überlegungen sind im Falle einer konkreten Anwendung im Bereich der Videoübertragung von Interesse. Viele Anwendungen und Dienste, die über Netzwerke übertragen bzw. angeboten werden, wie beispielsweise Videoübertragungen oder Multimedia-Kommunikation, lassen bei aufgetretenen Störungen aufgrund von Echtzeit-Bedingungen eine wiederholte Übertragung der gesendeten Daten nicht zu. Daher wird auf verschiedenen Netzwerkebenen zur Gewährleistung von *Quality of Service (QoS)* auf unterschiedliche Fehlerkorrekturverfahren zurückgegriffen. So können Ergebnisse, die beispielsweise die Modellierung eines ATM-Netzes mittels eines endlichen Markov Modells liefert, zur Bestimmung der Ende-zu-Ende QoS-Parameter, zum Design von effizienten Transportprotokollen und Überlaufmechanismen sowie zum *traffic management* genutzt werden. In erster Linie kann mit dem in dieser Arbeit entworfenen Schema eine Beurteilung der bislang eingesetzten Codierverfahren auf den verschiedenen Ebenen eines Netzwerkes erzielt werden. Da Sprach- und Videodaten im allgemeinen über unterschiedliche Medien übertragen werden, läßt sich das vorgestellte endliche Markov Modell und das darauf beruhende Verfahren zur Bewertung

verschiedener Fehlerkorrekturverfahren im Hinblick auf die verwendete Netzwerktechnologie problemlos anwenden. Denkbar sind in diesem Zusammenhang Datenübertragungen über ATM-Netze, insbesondere über *wireless* ATM-Netze. Aber auch im TCP/IP-Bereich oder im Bereich der *High Speed Networks*, wie beispielsweise Datenübertragungen über Kupferkabel mittels *Asymmetric Digital Subsriber Line (ADSL-Technologie)*, sowie Gigabit-Ethernet, ist eine Bewertung der verwendeten Codierverfahren interessant.
Das konkrete Anwendungsbeispiel, die Übertragung von Videodaten, beinhaltet weitere interessante Aspekt. So läßt sich der Verlauf von Zellverlusten bei Videoübertragungen mittels der Verkehrsmodellierung unter Zuhilfenahme endlicher Markov Modelle beschreiben. Der Videoverkehr wird als variabler Bitratenverkehr mit *burstiness* interpretiert. Werden beispielsweise Videokonferenzen betrachtet, dann läßt sich der Verkehr mit endlichen Markov Modellen relativ leicht abbilden. Videoclips dagegen bestehen aus vielen Schnitten und damit mehreren Szenen sehr unterschiedlicher Szenenlänge, so dass der Verkehr als *self-similar process* charakterisiert werden kann. Da jedoch Puffer, die in Vermittlungssystemen oder auch in Endgeräten eingesetzt werden, nur Datenmengen von wenigen Sekunden aufnehmen können und sich zudem für viele Anwendungen zu große Verzögerungszeiten ergeben würden, ist eine kurzfristige Betrachtung des Verkehrs angemessen. Folglich ist es ausreichend, bei der Verkehrsmodellierung Daten innerhalb einer Szene zu betrachten. Daher kann der Verkehr mit Markov Modellen oder besser noch unter Zuhilfenahme von autoregressiven Prozessen abgebildet werden. Das in der Dissertation entworfene Störungsmodell kann somit zur Verkehrsmodelierung verwendet werden. Als Parameter werden dazu für zukünftige Untersuchungen die Zellverlustwahrscheinlichkeit und die Muster der Verluste benötigt.

Als ein weiteres Forschungsziel bietet sich auch die Entwicklung einer Methode bzw. eines Verfahrens an, das Quellen- und Kanalcodierung miteinander verknüpft. In Kapitel 2 ist kurz skizziert worden, dass es Aufgabe des Quellencodierers ist, die zu übertragenden Daten geeignet zu komprimieren. Ein wichtiger und überaus interessanter zu analysierender Punkt ist hierbei, die Möglichkeiten zur Entwicklung eines Verfahrens zu prüfen, das Algorithmen zur Komprimierung von Daten geeignet den Algorithmen eines Kanalcodierers verbindet.
Als konkretes Anwendungsbeispiel erscheint die Übertragung von Videodaten neben der Bewertung von Flußkontrollverfahren auch unter dem folgenden Aspekt besonders erfolgversprechend. Bei der Absicherung von Videodaten gegenüber zufälligen Störungen liegen Daten mit unterschiedlicher Priorität vor, die bei der Sicherung der Daten nach Möglichkeit berücksichtigt werden sollte. So bestehen Videodaten einerseits aus Strukturdaten, die den Aufbau des Bildes organisieren, und aus Farbdaten, die den Farbgehalt der durch die Strukturdaten vorgegebenen Fläche angeben. Da die Strukturdaten den Bildaufbau organisieren, sind diese mit einer höheren Priorität versehen als die Farbdaten und daher bei der Fehlerkorrektur stärker abzusichern. Die existierenden Verfahren, die ein an die Priorität der Daten angepasstes Codierschema verwenden, können mit dem in der Arbeit vorgestellten Bewertungsschema analysiert werden. Dazu ist das entworfene Störungsmodell an die Situation, dass Daten unterschiedlicher Priorität vorliegen, anzupassen. Ebenfalls zu modifizieren ist das zur Bewertung der betrachteten Codierverfahren entworfene Bewertungskonzept. Anschließend ist insbesondere die Frage zu klären, ob sich auf der Grundlage der Fehlerstatistik eines betrachteten Übertragungskanals im Sinne des Bewertungsverfahrens ein optimaler Fehler-korrigierender Code entwickeln läßt, der die unterschiedliche Gewichtung der Daten berücksichtigt.
In diesem Zusammenhang wirft die Verknüpfung von Quellen- und Kanalcodierung,

wie z.B. bei der MPEG-Komprimierung, weitere interessante Fragestellungen auf. Zu analysieren ist, ob eine Verknüpfung von Quellen- und Kanalcodierung ein im Sinne der Kanalcodierung effizientes und sicheres Fehlerkorrekturverfahren mit geringem Aufwand liefern kann, das die Redundanz der Quellencodierung zur Fehlerkorrektur nutzt; d.h. zu untersuchen ist, ob sich ein Codierverfahren entwickeln läßt, das eine hierarchische Datenstruktur der Quellencodierung zur Kanalcodierung unter der Bedingung nutzen kann, dass die Strukturdaten der Quellencodierung stärker geschützt werden als die Nutzdaten. Ein erster Ansatz für eine solche Entwicklung findet sich in [8].
Im Bezug auf Videodaten läßt sich diese Fragestellung konkret an den folgenden Aspekten erörtern und analysieren. Im Bereich der Bildverarbeitung liefern verschiedene Methoden der Quellencodierung, wie z.B. Octrees und Quadtrees oder auch Huffman-Codes, sogenannte Codebäume, also eine hierarchische Baumstruktur. Eine Bildcodierung mit Hilfe eines Octrees bzw. Quadtrees und die anschließende Übertragung der Bilddaten über einen gestörten Kanal beinhalten die gleiche Problematik, wie sie oben bei der Übertragung der Videodaten erläutert worden ist. Wenn während der Übertragung ein Fehler die Baumstruktur des Octrees verfälscht, wird dadurch die Struktur des übertragenen Bildes verändert, so dass das Bild nicht korrekt rekonstruiert werden kann. Bei einer Verfälschung der Nutzdaten, also der Farbinformationen des Bildes, ist das Bild in seiner Struktur korrekt wiederherstellbar, lediglich die Farbtöne können von den ursprünglichen abweichen. Hieraus ergibt sich als vorrangiges Ziel, eine stärkere Absicherung der Strukturinformation gegenüber der Farbinformation zu erreichen. Mit den in dieser Arbeit vorgestellten Konzepten und Methoden kann der so konstruierte Code hinsichtlich seiner Effizienz und Burstfehlerkorrekturfähigkeit bewertet werden. Im Anschluß an die oben genannten Untersuchungen sind dann Aussagen über die Effizienz dieser verknüpften Quellen- und Kanalcodierung möglich. Unter Vorgabe der Kanalparameter, Bitfehlerrate und mittlere Burstlänge, können Angaben zur Korrekturfähigkeit gemacht und bewertet werden, inwieweit die gefundene Codierung Burstfehler korrigieren kann.

Insgesamt ergibt sich so ein weites Spektrum erfolgversprechender Anwendungsmöglichkeiten für die Methoden und Konzepte dieser Arbeit.

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Literaturverzeichnis


[1] H. Bauer, *Wahrscheinlichkeitstheorie*, 4. Auflage, de Gruyter (1990)

[2] F. Beichelt, *Stochastische Prozesse für Ingenieure*, Teubner (1997)

[3] J.M. Berger, B. Mandelbrot, A new model for error clustering in telephone circuits, *IBM J. Res. Dev.*, 7 (1963) 224-236

[4] E. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill (1968)

[5] G. Birkhoff und S. MacLane, *A survey of modern algebra*, Macmillan (1965)

[6] R.C. Bose und D.K. Ray-Chaudhuri, On a class of error correcting binary group codes, *Information und Control*, 3 (1960) 68-79

[7] M. Bossert, *Kanalcodierung*, B.G. Teubner (1998)

[8] X. Changsong, P. Eck, R. Matzner, K. Tröndle, Syntax-orientierte Codierung für syntaktisch strukturierte Quellen, *Proceedings der ersten Konferenz Codierung für Quelle,Kanal und Übertragung* ITG-Fachtagung, VDE-Verlag (1998) 47-52

[9] C.C. Chao and Y.L. Yao, Hidden Markov Models for the Burst Error Statistics of Viterbi Decoding *IEEE Trans. Comm.* 44 (1996) 1620-1622

[10] E.O. Elliot, Estimates of error rates for codes on burst-noise channels, *Bell System Tech. Journal* 44 (1965) 89-109

[11] W. Feller, *An Introduction to Probability Theory and Its Applications* Vol I & II, John Wiley & Sons, Inc. (1971)

[12] B.D. Fritchman, A binary channel characterization using partitioned Markov chains, *IEEE Trans. Info. Theory*, IT-13 (1967) 221-236

[13] R.G. Gallager, *Information Theory and Reliable Communication*, John Wiley and Son, Inc. (1968)

[14] E.N. Gilbert, Capacity of a Burst-Noise Channel *Bell Syst. Tech. J.* 39 (1960) 1253-1266

[15] A.J. Goldsmith and P.P. Varaiya, Capacity, Mutual Information and Coding for Finite-State Markov Channels *IEEE Trans. Info. Theory* 42 (1996) 868-886

[16] G. Haßlinger, Semi-Markovian Modelling and Performance Analysis of Variable Rate Traffic in ATM Networks *Telecommunication Systems, selected papers of the 3. INFORMS Telecommunication Conference*, Boca Raton (1997)

[17] G. Haßlinger and M. Adam, Semi-Markovian Traffic Models Including Autocorrelation, *Proc. IEEE Infocom'96 Conference, San Francisco* (1996) 1460-1467

[18] W. Heise und P. Quattrocchi, *Informations- und Codierungstheorie*, zweite Auflage, Springer-Verlag (1989)

[19] A. Hocquenghem, Codes correcteurs d'erreurs, *Chiffre* 2 (1959) 147-156

[20] J.J. Hunter, *Mathematical Techniques of Applied Probability*, Ac. Press (1983)

[21] K.D. Kammeyer, *Nachrichtenübertragung*, B.G. Teubner (1996)

[22] J.G. Kemeny, *Slowly spreading chains of the first kind,* J. Math. Analysis and Appl. (1966) 295-310

[23] R.L. Graham, D.E. Knuth, O. Patashnik, *Concrete Mathematics,* Addison-Wesley (1994)

[24] C. Kröll, *Digital Symmetrical Channels with Memory (in german)* Dissertation, Darmstadt University of Technology (1992)

[25] A. Kuznetsov, F. Swarts, A.J. van Vinck and H.C. Ferreira, *On the Undetected Error Probability of Linear Block Codes on Channels with Memory* IEEE Trans. Info. Theory 42 (1996) 303-30

[26] S. Lin and D. Costello, *Error Control Coding*, Prentice Hall (1983)

[27] J.H. van Lint, *Introduction to Coding Theory*, second edition, Springer-Verlag (1992)

[28] Z. Liu, M.J. Karol, M. El Zarki and K.Y. Eng, *Channel Access and Interference Issues in Multi-Mode DS-CDMA wireless packet (ATM) networks* Wireless Networks 2 (1996) 163-171

[29] D. Lochmann, *Digitale Nachrichtentechnik : Signale, Codierung, Übertragungssysteme, Netze*, Verl. Technik (1997)

[30] J.L. Massey, *Threshold Decoding*, M. I. T. Press (1963)

[31] J.L. Massey, Foundations and methods of channel coding, *Proc. Intern. Conf. on Inform. Theory and Systems*, NTG-Fachberichte 65 (1978) 148-157

[32] R. Mathar, D. Pfeiffer, *Stochastik für Informatiker,* B. G. Teubner (1990)

[33] D.E. MacDysan, D.L. Spohn, *ATM theory and applications*, McGraw-Hill (1999)

[34] F. MacWilliams, N. Sloane, *The Theory of Error-Correcting Codes*, North-Holland (1977)

[35] R.H. McCullough, The binary regenerative channel, *Bell System Tech. Journal* 47 (1968) 1713-1735

[36] R. McEliece, *The Theory of Information and Coding*, Addison-Wesley (1977)

[37] J.E. Meggitt, Error correcting codes and their implementation, *IRE Trans. on Inform. Theory* 7 (1961) 232-244

[38] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory*, Springer-Verlag (1995)

[39] C. Osmann, G. Haßlinger, Performance of Burst versus Random Error Codes, *Proceedings* 2nd *Europ. Personal Mobile Communication Conf.* ITG-Fachbericht, VDE-Verlag (1997) 441-448

[40] C. Osmann, G. Haßlinger, Modeling of Error Patterns in Communication Channels and Implications for Error-Correction, *Proceedings of the 4th INFORMS Telecommunications Conference*, (1998) 58-69

[41] C. Osmann, G. Haßlinger, Modeling and Analysis of Channel Coding in the Presence of Error Bursts, *Proceedings of the 2nd Symposium on Semi-Markov Models: Theory and Applications* (1998) 231-238

[42] C. Osmann, G. Haßlinger, Residual Error Rates for Coding Analysed by Markovian Modeling of Transmission Channels, *Proceedings of the 3rd European Personal Mobile Communications Conference* (1999) 301-307

[43] W. Peterson und E. Weldon, *Error-Correcting Codes*, MIT-Press (1972)

[44] J. Pfanzagl, *Elemantare Wahrscheinlichkeitsrechnung*, 2. Auflage, de Gruyter (1991)

[45] J.G. Proakis, D.G. Manolakis, *Digital Signal Processing*, thrid edition, Prentice Hall (1996)

[46] S.H. Reiger, Codes for the correction of clustered errors, *IRE Trans. on Inform. Theory* 6 (1960) 16-21

[47] S. Roman, *Coding and Information Theory*, Springer-Verlag (1992)

[48] C. Shannon, A Mathematical Theory of Communication, *Bell Systems Techn. Journal*, 27, 379-423 und 623-656 (1948)

[49] A.S. Tanenbaum, *Computernetzwerke*, 3., revidierte Auflage, Prentice Hall (1998)

[50] S. Tsai, Markov Characterization of the HF Channel, *IEEE Trans. Commu.* 17 (1972) 24-32

[51] S. Tsai, P.S. Schmied, Interleaving and Error-Burst Distributions, *IEEE Trans. Comm.* 20 (1972) 291 -296

[52] H. Tzschach und G. Haßlinger, *Codes für den störungssicheren Datentransfer*, Oldenburg-Verlag (1993)

[53] J.R. Yee and E.J. Weldon, Evaluation of the Performance of Error-Correcting Codes on a Gilbert Channel *IEEE Trans. Comm.* 43 (1995) 2316-2323