Institut für Nachrichtenvermittlung und Datenverarbeitung

Universität Stuttgart

Prof. Dr. Ing. A. Lotze

8. Bericht

Statistische Decodierverfahren

mit einer Einführung in die linearen Code

von
JOACHIM BRANDT

1969

A Short Summary

Using the wellknown block-code-methods, redundant data-transmission with error-correction at the receiver but without the possibility of repeating erroneous passages is only possible if the technical limitation does'nt matter: with the well-known block-codes the transmission either needs very much time or a very large technical equipment, otherwise the error-probability will not be very good.

Three new "statistical" coding methods (namely of R. G. GALLAGER[1], J. M. MASSEY[2] and of J. M. WOZENCRAFT and B. REIFFEN[3]) are dealt with in this paper. The original papers are somewhat difficult to read. In the first part of this paper only the function of the three coding-methods will be explained. In the second part the efficiency of the new methods will be described and compared with the coding-methods familiar up to now.

Chapter I introduces theory of coding. Chapter II deals with the general block-coding. Based upon this in chapter III the coding-method of GALLAGER will be discussed. In chapter IV the sequential (or convolutional) coding will be introduced. This coding principle is not yet wellknown and will be presented here in a closed and complete description.

Based on chapter IV the coding-methods of WOZENCRAFT-REIFFEN and MASSEY are discussed in chapter V and chapter VI, respectively.

In chapter VII the attainable error-probabilities in case of redundant coding-methods are discussed, according to FANO[5].
In the last chapter VIII the efficiency of the three "statistical" methods will be compared with those limits obtained in chapter VII.

Kurze Inhaltsangabe

Gesicherte Datenübertragung mit empfängerseitiger Fehlerkorrektur ohne Wiederholungsanforderung leidet bei den bekannten Blocksicherungsverfahren sehr unter technischer Beschränkung: entweder ist der Zeit- oder Geräteaufwand untragbar groß, oder es ist die erreichbare Korrektursicherheit klein.

Hier werden drei neue, "statistische" Decodierverfahren (nämlich von R. G. GALLAGER[3], J. M. WOZENCRAFT und B. REIFFEN[1] sowie von J. L. MASSEY[2]) behandelt. Sie sind in den Originalarbeiten etwas schwer zugänglich. Im ersten Teil der Arbeit wird erklärt, wie diese Verfahren funktionieren und erst im zweiten Teil wird ihre Wirksamkeit miteinander und mit den bisher geläufigen Verfahren verglichen.

Kapitel I stellt eine Einführung in die Codierung dar. Kapitel II behandelt die allgemeine Blockcodierung. Darauf aufbauend kann in Kapitel III das Verfahren von GALLAGER besprochen werden. Kapitel IV bringt eine Abhandlung über sequentielle Codierung. Diese ist noch weitgehend unbekannt und wird hier in einer geschlossenen Darstellung vorgestellt. Mit diesen Grundlagen können jetzt auch die Verfahren von WOZENCRAFT-REIFFEN in Kapitel V und MASSEY in Kapitel VI besprochen werden.

Einer kritischen Untersuchung der erreichbaren Decodiersicherheit bei redundanter Codierung ist Kapitel VII gewidmet, in Anlehnung an FANO[5]. Kapitel VIII vergleicht die Wirksamkeit der drei neuen Decodierverfahren mit den erreichbaren, in Kapitel VII diskutierten Grenzen.

Inhaltsverzeichnis

EINLEITUNG

Aufgabe der vorliegenden Arbeit ist es, drei fehlerkorrigierende Decodierverfahren einführend zu besprechen, die in den Vereinigten Staaten entwickelt worden sind und sich von den bis dahin bekannten Verfahren grundsätzlich unterscheiden. Sie wurden in Arbeiten von WOZENCRAFT und REIFFEN[1], MASSEY[2] und GALLAGER[3] veröffentlicht.

Die herkömmlichen Verfahren[6] lösen das Codierproblem befriedigend; sie haben aber den Nachteil, daß bei fehlerkorrigierender Decodierung der Aufwand der Decodieranlage untragbar groß werden kann. Ziel der neuen Verfahren ist es, außer geringem Codieraufwand auch erträglichen Decodieraufwand zu erreichen.

Die vorliegende Arbeit beschränkt sich auf binäre Übertragung, den wichtigsten Fall. Die Störung wird als "symmetrisch" und statistisch unabhängig vorausgesetzt. Diese Beschränkung wohnt den Verfahren nicht inne; allerdings lassen sich die drei Verfahren mit diesem symmetrischen Binärkanal als Störungskanal gut vergleichen. Es wurde Wert gelegt auf eine geschlossene Darstellung der Grundlagen, um auf diesen in den späteren Kapiteln aufbauen zu können. Deswegen stellen die Kapitel I, II und IV eine allgemeine Einführung in die Codierung dar. In Kapitel I werden einige grundlegende Begriffe der Datenübertragung und Codierung eingeführt. Es wird der von SHANNON 1948 bewiesene, wichtigste Satz der Informationstheorie - spezialisiert auf den symmetrischen Binärkanal - besprochen, und schließlich werden Blockcode und sequentielle Code vorgestellt. Das zweite Kapitel befaßt sich mit allgemeiner Blockcodierung und bringt die geläufigen Codetypen, ohne aber auf spezielle Blockcodeensemble einzugehen. Diese speziellen Blockcode - insbesondere die wichtigen zyklischen Code - behandelt PETERSON[6] geschlossen und ausführlich.

Kapitel III bringt eine Einführung in GALLAGER's Decodierverfahren. Dieses baut auf speziellen Blockcode auf, die GALLAGER "Low-Density-Codes" nennt und die deshalb abkürzend als

LD-Code bezeichnet werden. Kapitel III kann bei erster Lektüre übersprungen werden. Kapitel IV behandelt die sequentielle Codierung. WOZENCRAFT, REIFFEN und MASSEY geben in ihren Arbeiten verschiedene Darstellungen des sequentiellen Codierprozesses. Es wird hier eine einheitliche Darstellung versucht; die Zusammenhänge zwischen den Codierverfahren von WOZENCRAFT-REIFFEN und MASSEY werden aufgezeigt.

Kapitel V behandelt die WR-Decodierung - die Abkürzung soll an WOZENCRAFT und REIFFEN erinnern - und Kapitel VI MASSEY's SW-Decodierung; SW wurde als Abkürzung für Schwellenwert gewählt.

In den Kapiteln III, V und VI werden die drei neuen Decodierverfahren von einem mehr praktischen Gesichtspunkt aus behandelt. Es soll gezeigt werden, wie die Verfahren "funktionieren", ohne die Frage zu beantworten, ob das Arbeiten mit diesen Systemen brauchbare Ergebnisse liefert, d.h. große Decodiergenauigkeit bei vertretbarem Aufwand. Die beiden abschließenden Kapitel VII und VIII gehen auf letztere Frage ein. Dabei können die Antworten - um den Rahmen der Arbeit nicht zu sprengen - nicht immer hergeleitet werden. Die Beweise finden sich in der Originalliteratur. Kapitel VII untersucht zunächst allgemein, welche Grenzen prinzipiell der Decodiergenauigkeit gesetzt sind und lehnt sich bei der Herleitung dieser Grenzen an FANO[5] an. In Kapitel VIII werden dann die Decodiereigenschaften - insbesondere die zu erwartenden Restfehlerwahrscheinlichkeiten - der drei neuen Verfahren besprochen und mit den in Kapitel VII erhaltenen Ergebnissen verglichen.

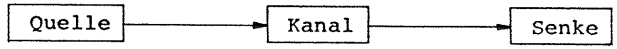
Der Leser, der nur an den Grundgedanken der Decodierverfahren interessiert ist, kann die mathematischen Formulierungen überlesen. Bei den mathematischen Ableitungen werden vorausgesetzt die Grundbegriffe der Matrizenrechnung und der linearen Gleichungssysteme sowie der Wahrscheinlichkeitsrechnung.

Ich bedanke mich bei Herrn Privatdozent Dr. Swoboda, der die Arbeit angeregt hat, für die zahlreichen, fruchtbaren Diskussionen, die der Arbeit entscheidende Impulse und ihre endgültige Form gegeben haben, und danke Herrn Prof. Dr. Lotze, Direktor des Instituts für Nachrichtenvermittlung und Datenübertragung an der Universität Stuttgart, für viele wertvolle Hinweise.

GRUNDBEGRIFFE DER CODIERUNG

1. Der symmetrische Binärkanal

Bei der Übertragung einer Nachricht sind drei Abschnitte zu unterscheiden, die das Blockschaltbild von Figur 1.1 darstellt.



Figur 1.1: Die drei Grundelemente einer Nachrichtenübertragungsanlage

In der Quelle werden die zu übertragenden Nachrichten produziert. Wir wollen vereinfachend annehmen, daß die Quelle ihre Information als eine Folge von Binärzeichen liefert. Dies bedeutet keine prinzipielle Einschränkung, da jedes andere Nachrichtenalphabet auf ein Dualsystem transformiert werden kann. Wir bezeichnen diese Nachrichtenstellen der Reihe nach mit x_1, x_2, x_3, ...; jedes x kann entweder 0 oder 1 sein. Wir nennen diese Binärfolge auch kurz Nachrichtenfolge. Von der Quelle nehmen wir ferner an, daß die Nachrichtenstellen statistisch unabhängig voneinander produziert werden und daß die Quelle im Mittel genau so oft 1 wie 0 sendet, daß also die Wahrscheinlichkeit für 1 und diejenige für 0 gleich groß ist:

$$P(x=1) = P(x=0) \tag{1}$$

Die von der Quelle erzeugten Nachrichten sollen über einen Kanal übertragen werden, damit sie von der Senke verwertet werden können. Statt Quelle und Senke sind auch die Begriffe Sender und Empfänger geläufig.

Von jedem physikalischen Übertragungskanal müssen wir annehmen, daß er nicht störungsfrei arbeitet, sondern einen gewissen Teil der von der Quelle gelieferten Information zerstört, d.h. die Folge y_1, y_2, y_3, ..., die der Kanal der Senke ausliefert - von uns Empfangsfolge genannt - muß nicht mit der gesendeten Folge x_1, x_2, x_3, ... übereinstimmen: es kann vorkommen, daß der Kanal an i-ter Stelle eine 0 liefert, obwohl $x_i = 1$ von der Quelle gesendet wurde, oder umgekehrt. Die

Statistik dieser Störtätigkeit kennzeichnet den Übertragungskanal.

Wir werden uns in dieser Arbeit nur mit dem symmetrischen Binärkanal beschäftigen. Der symmetrische Binärkanal, abgekürzt BSC (von "binary symmetric channel"), ist durch die folgenden Eigenschaften gekennzeichnet:

a) Eingangs- und Ausgangszeichen sind binär. Der Kanal kann nur an Binärquellen angeschlossen werden und liefert der Senke eine Folge von Binärzeichen.

b) Der Kanal hat kein "Gedächtnis"; er wirkt auf die durchlaufenden Binärziffern statistisch unabhängig ein.

c) Die Wahrscheinlichkeit, daß der Kanal richtig überträgt, ist für beide Zeichen gleich; also

$$P(y=1|x=1) = P(y=0|x=0) \qquad (2)$$

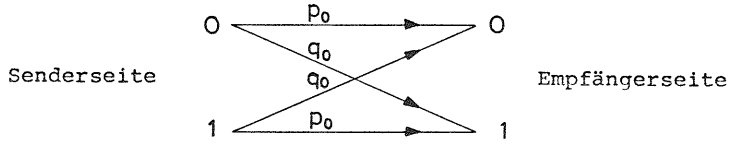
Diese bedingte Wahrscheinlichkeit wollen wir mit q_0 bezeichnen. In Worten heißt die obige Gleichung: die Wahrscheinlichkeit dafür, daß der Kanal eine "1" an die Senke ausliefert, unter der Voraussetzung, daß die Quelle eine "1" produziert hat, ist gleich der Wahrscheinlichkeit, daß der Kanal eine "0" ausliefert, wenn die Quelle eine "0" gesendet hat. Die Wahrscheinlichkeit, gefälscht zu werden, ist damit ebenfalls für beide Zeichen gleich:

$$P(y=1|x=0) = P(y=0|x=1) \qquad (3)$$

Diese Fehlerwahrscheinlichkeit des BSC wollen wir mit p_0 bezeichnen. Es gilt:

$$p_0 + q_0 = 1 \qquad (4)$$

Das mathematische Modell eines BSC wird durch die Darstellung in Figur 1.2 gekennzeichnet.

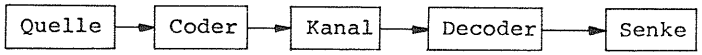


Figur 1.2: Schematische Darstellung eines symmetrisch gestörten Binärkanals

Von der Fehlerwahrscheinlichkeit p_0 setzen wir voraus, daß $p_0 \leq 1/2$ ist. Ein Kanal mit $p_0 > 1/2$ wird auf den Fall $p_0 \leq 1/2$ zurückgeführt, wenn statt der von der Quelle angelieferten Binärfolge die komplementäre übertragen wird, d.h. diejenige Folge, bei der jede 1 durch eine 0 und jede 0 durch eine 1 ersetzt wurde. Da für $p_0 = 1/2$ keine sinnvolle Übertragung möglich ist, können wir $p_0 < 1/2$ voraussetzen.

2. Die Nachrichtenrate

Von der natürlichen Sprache wissen wir, daß auch bei Störgeräusch eine relativ sichere Verständigung möglich ist; die natürliche Sprache ist redundant, weitschweifig. Ziel einer gesicherten Datenübertragung ist es, durch künstlich hinzugefügte Redundanz dem Empfänger die Beseitigung von Übertragungsfehlern zu ermöglichen. Um trotz Kanalstörung eine große Übertragungsgenauigkeit zu erreichen, erweitern wir deshalb unser Übertragungssystem; diese Erweiterung zeigt Figur 1.3.



Figur 1.3: Erweiterte Nachrichtenübertragungsanlage

Im Coder wird die eintreffende Nachrichtenfolge in die Codefolge verschlüsselt und soll damit geschützt werden; z.B. werden jeweils einer Gruppe von m Nachrichtenstellen k Kontrollstellen hinzugefügt. Diese Kontrollstellen sollen es dem Decoder ermöglichen, die vom Kanal angelieferten - möglicherweise gefälschten - Stellen auf ihre Richtigkeit zu untersuchen, also Übertragungsfehler zu erkennen oder zu korrigieren. Dabei soll m an "message", die englische Vokabel für Nachricht, erinnern und k die Beziehung zu Kontrolle bzw. "control" andeuten. Die

Summe $m + k$ bezeichnen wir mit n.

Ziel der Codierung ist es, trotz der Kanalstörung eine sichere Übertragung zu erreichen. Das Verhältnis von Kontrollstellen zu Nachrichtenstellen wird von Bedeutung sein für die erreichbare Decodiergenauigkeit. Die Größe

$$R = \frac{m}{m+k} = \frac{m}{n} \qquad (5)$$

nennen wir Nachrichtenrate oder kurz Rate. Die Rate ist ein Maß für den Nachrichtenbelag, den ein den Coder verlassendes Binärzeichen im Mittel trägt *). Je größer die Zahl k der Kontrollstellen bei festem m, je kleiner also die Rate, desto größer ist die Redundanz - die Weitschweifigkeit - der vom Kanal zu übertragenden Binärfolge.

3. Lineare, sequentielle Netzwerke

Der Coder, aber auch Teile des Decoders, lassen sich als Schaltung durch sog. lineare, sequentielle Schaltnetze verwirklichen.

Ein lineares, sequentielles Netzwerk enthält nur zwei Typen von Schaltelementen; Verzögerungs- bzw. Speicherelemente für eine binäre Größe, die technisch als sog. Flip-Flop realisiert werden, und modulo-2 Addierglieder, die auch als Antivalenzverknüpfer bezeichnet werden. Die Flip-Flop werden in unseren Blockschaltbildern durch Quadrate dargestellt, deren zuführende Leitung durch einen Pfeil gekennzeichnet ist (siehe Figur 1.4a).

Die modulo-2 Addierglieder führen eine Addition modulo-2 aus. In der modulo-2 Rechnung der ganzen Zahlen gibt es nur zwei wesentlich voneinander verschiedene Zeichen: 0 und 1. Statt des normalen Pluszeichens benutzt man in modulo-2 Rechnung gern das Addiersymbol $\oplus$.

*) In der amerikanischen Literatur heißt das Verhältnis $\frac{m}{n}$ "information rate".
Für redundanzfrei vorausgesetzte Nachrichtenstellen entspricht die Nachrichtenrate dem mittleren Informationsbelag eines Symbols beim allgemeinen Übertragungskanal.

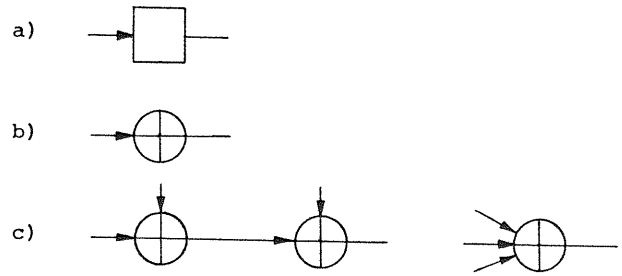
In modulo-2 Rechnung gelten die folgenden Addierregeln:

$$0 \oplus 0 = 0 \quad 0 \oplus 1 = 1 \quad 1 \oplus 0 = 1 \quad 1 \oplus 1 = 0 \qquad (6)$$

und die normalen Multiplikationsregeln:

$$0 \cdot 0 = 0 \quad 0 \cdot 1 = 0 \quad 1 \cdot 0 = 0 \quad 1 \cdot 1 = 1 \qquad (7)$$

Die modulo-2 Addierer werden in den Schaltbildern durch ihr Additionssymbol gekennzeichnet (Figur 1.4b); hintereinandergeschaltete modulo-2 Addierer werden symbolisch zusammengefaßt (wie in Figur 1.4c).



Figur 1.4: Schaltelemente linearer, sequentieller Netzwerke
a) Flip-Flop
b) modulo-2 Addierer
c) Zusammenfassung hintereinandergeschalteter modulo-2 Addierer

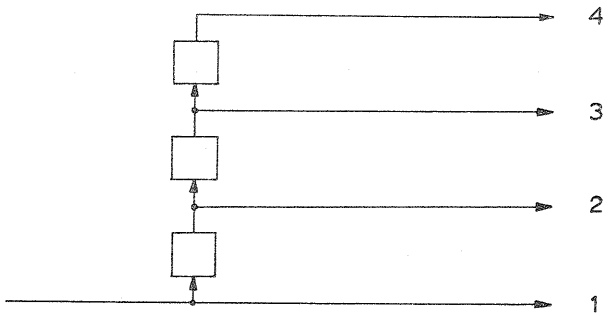
Lineare, sequentielle Netze wurden von HUFFMANN[8] und ELSPAS[9] eingehend untersucht. Für sie gilt das Überlagerungsgesetz für Eingangsfolgen und zugehörige Ausgangsfolgen gemäß der modulo-2 Addition, worauf wir später eingehen werden.

Man unterscheidet Netze mit oder ohne Rückkopplung. Haben wir den Fall eines rückkopplungsfreien Netzwerkes vor uns, so besteht dessen Ausgang durchgehend aus Nullen, wenn nur Nullen eingespeist wurden. Bei einem nicht rückkopplungsfreien Netzwerk ist dies nicht notwendig der Fall.

4. Der Coder

Wenn wir nicht gerade den einfachsten Fall vor uns haben, daß der Coder sich darauf beschränkt, die einlaufenden Nachrichtenstellen zu wiederholen, je öfter, desto sicherer würde die Übertragung, so wird der Ausgang des Coder sinnvollerweise nicht nur von der gerade eingelaufenen Stelle abhängen, sondern auch von weiteren Nachrichtenstellen. Wir wollen zwei Codertypen unterscheiden. Beim Seriencoder treten die Nachrichtenstellen hintereinander einzeln in den Coder ein. Seriencoder sind insbesondere für zyklische Code bekannt und beanspruchen für diese einen minimalen Aufwand[6].

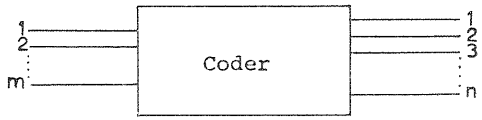
Andere Codierverfahren - speziell auch die hier zu besprechenden - arbeiten als Parallelcoder: bei ihnen wird vom Coder pro Arbeitstakt nicht nur eine einzelne Nachrichtenstelle, sondern es werden m Nachrichtenstellen gleichzeitig verarbeitet. Falls die Nachrichtenstellen seriell anfallen, müssen Gruppen von m Stellen durch Serien-Parallel-Wandlung gebildet werden. Dies läßt sich mit einem Schieberegister von (m-1) Flip-Flop erreichen, wie dies Figur 1.5 für m=4 zeigt.



Figur 1.5: Serien-Parallel-Wandler für m=4

Zu Beginn eines jeden Arbeitstaktes treten in den Parallelcoder gleichzeitig m Nachrichtenstellen ein. Ebenso werden vom Parallelcoder die n = m+k Binärstellen eines Arbeitstaktes gleichzeitig ausgegeben.

Demgemäß hat ein Parallelcoder das Blockschaltbild von Figur 1.6.



Figur 1.6: Der Parallel-Coder

Falls für die Übertragung ein einziger Binärkanal zur Verfügung steht, müssen die n pro Arbeitstakt parallel austretenden Binärzahlen wieder in serielle Form gebracht werden. Diese Rückwandlung führt ein Parallel-Serien-Wandler durch. Er arbeitet entsprechend zu dem in Figur 1.4 gezeigten Serien-Parallel-Wandler.

5. Blockcodierung. Sequentielle Codierung

Wir wollen Blockcode und sequentielle Code unterscheiden. Als Blockcodierung bezeichnet man ein Codierverfahren, bei dem jeweils ein Block von m Nachrichtenstellen für sich behandelt wird. Dem m-stelligen Nachrichtenwort wird durch den Coder ein Block von n>m Stellen zugeordnet, der dann vom Kanal übertragen wird. Dieses n-Tupel nennt man das zum entsprechenden Nachrichtenwort gehörige Codewort; n heißt die Blocklänge des verwendeten Blockcode. Die vom Kanal übertragene Codefolge läßt sich bei Blockcodierung aufteilen in Teilstücke der Länge n - die Codeworte -, die voneinander völlig unabhängig sind.

Beim Arbeiten mit sequentiellen Code *) wollen wir immer einen Parallelcoder voraussetzen. Da der Parallelcoder auch für Blockcode brauchbar ist, wird die Kennzeichnung der sequentiellen Code gegenüber den Blockcode bei diesem Codertyp besonders deutlich: bei Blockcodierung hängen die n pro Arbeitstakt ausgelieferten Binärstellen nur von den gerade anliegenden m Nachrichtenstellen ab. Bei sequentieller Codierung wird dagegen der Coderausgang von r weiteren Nachrichtenworten zu je m Stellen mitbestimmt.

*) In der amerikanischen Literatur entsprechen die sequentiellen Code den "Convolutional Codes", während unter "Sequential Decoding" speziell das Decodierverfahren von WOZENCRAFT und REIFFEN verstanden wird.

Die Größe r wollen wir Rückgriff nennen, die Größe

$$n_0 = (r+1) \cdot n \qquad (8)$$

heiße Codelänge. Die Codelänge ist die maximale Zahl von Stellen in der Codefolge, auf die eine Nachrichtenstelle Einfluß haben kann. Bei Blockcodierung ist r = 0 und damit wird die Codelänge n_0 gleich der Blocklänge n. Bei Blockcodierung werden innerhalb eines Parallelcoders keine Speicherelemente benötigt. Die linearen, sequentiellen Code sind ein besonders wichtiger, da leicht zu instrumentierender Spezialfall der sequentiellen Code. Sie sind dadurch gekennzeichnet, daß ihre Codieranlage durch ein rückkopplungsfreies Netzwerk ausgeführt werden kann. Unter sequentiellen Code seien weiterhin immer lineare sequentielle Code verstanden.

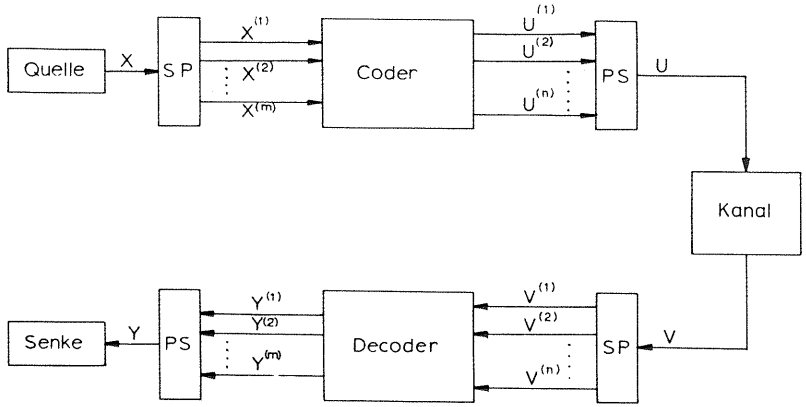
6. Der Decoder

Die dem Übertragungskanal nach der Parallel-Serien-Umwandlung angelieferte Binärfolge nannten wir Codefolge; bei Blockcodierung besteht sie aus den nacheinander folgenden Codeworten. Die den Kanal verlassende Binärfolge - die Empfangsfolge - ist durch Störung aus der Codefolge entstanden. Der Decoder hat die Aufgabe, aus der Empfangsfolge die ursprüngliche Nachrichtenfolge der Quelle möglichst getreu wiederherzustellen.

Bei Blockcodierung arbeitet auch der Decoder blockweise. Aus dem empfangenen n-Tupel, das wegen der Kanalstörungen nicht mit dem gesendeten Codewort übereinstimmen muß, soll der Decoder die m-stellige Nachricht der Quelle bestimmen. Nachdem ein Block bearbeitet ist, wird der Decoder entleert und beginnt erst dann mit der Decodierung des folgenden Blockes.

Ein sequentieller Code muß auch sequentiell decodiert werden. Der Decoder muß den Empfang von r weiteren Gruppen zu je n Stellen abwarten, bevor er aus ihnen die erste Gruppe von m Nachrichtenstellen bestimmen kann. Er arbeitet also gegenüber der Nachrichtenquelle mit einer Verzögerung von r Arbeitstakten.

Die Arbeitsweise der drei zu besprechenden Decodierverfahren im einzelnen wird in den Kapiteln III, IV und VI behandelt. Das für alle drei gemeinsame Blockbild für das Übertragungssystem mit Parallelcoder und Decoder zeigt Figur 1.7.



Figur 1.7: Übertragungssystem eines Parallelcoders
SP: Serien-Parallel-Wandler
PS: Parallel-Serien-Wandler

Sie führt außerdem die in der Arbeit benützte Benennung der Binärzeichen ein, die während des Übertragungsprozesses auftreten. Die großen Buchstaben kennzeichnen eine gesamte Folge von Binärzahlen; einzelne Binärzahlen werden durch kleine Buchstaben bezeichnet: z.B. gehören zur Folge $U^{(\nu)}$ die Binärzahlen $(u_1^{(\nu)}, u_2^{(\nu)}, u_3^{(\nu)}, \ldots)$, und die Codefolge U ist die Gesamtheit der Binärzahlen $(u_1, u_2, u_3, \ldots)$.

Hat der Decoder aus der empfangenen Binärfolge die richtige Nachricht bestimmt, liegt also kein Decodierfehler vor, dann stimmt Y mit X überein, bzw. y_i mit x_i für alle i.

7. Bemerkungen zur Restfehlerwahrscheinlichkeit eines Code

Die Wirksamkeit eines Code wird man nach der Restfehlerwahrscheinlichkeit PR beurteilen; PR ist die Wahrscheinlichkeit dafür, daß eine Nachrichtenstelle vom Decoder nicht richtig reproduziert wird, also PR =P(y≠x).

Über diese Restfehlerwahrscheinlichkeit hat SHANNON 1948 einen grundlegenden Satz bewiesen. FEINSTEIN verfeinerte diesen Satz 1955 für Binärcode. Dieser Satz wird hier in der Formulierung von Robert M. FANO wiedergegeben:
Für jeden stationären Kanal mit endlichem "Gedächtnis" kann eine Kanalkapazität C_0 definiert werden, die folgende Bedeutung besitzt: Für jede binäre Nachrichtenrate R, die kleiner als die Kanalkapazität C_0 ist, kann die Restfehlerwahrscheinlichkeit je Binärstelle durch geeigneten Entwurf von Kanalcoder und -decoder beliebig klein gemacht werden. Umgekehrt kann die Restfehlerwahrscheinlichkeit nicht beliebig klein gemacht werden, wenn R grösser als C_0 ist.

Dadurch ist die Kanalkapazität pro Binärstelle eines Binärkanals definiert. Für den BSC ist sie gegeben durch:

$$C_0 = 1 + p_0 \, ld \, p_0 + (1-p_0) \, ld \, (1-p_0) \qquad (9)$$

Es ist dabei p_0 die in Abschnitt 1 eingeführte Fehlerwahrscheinlichkeit des BSC und ld der Logarithmus zur Basis 2.

Für $R < C_0$ kann die Restfehlerwahrscheinlichkeit zwar beliebig klein gemacht werden; für jedes feste n_0 bleibt jedoch auch bei optimaler Codierung - falls $p_0 > 0$ ist - eine von Null verschiedene Restfehlerwahrscheinlichkeit PR. Es sei vorweggenommen, daß PR_L, die Restfehlerwahrscheinlichkeit, die durch bestmögliche Codierung erreicht werden kann, abhängt von der Codelänge n_0, von der Rate R und der Kanalkapazität C_0. Diese Abhängigkeit hat die folgende Struktur (Kapitel VII, Gleichung 15)

$$PR_L = K \cdot 2^{-n_0 \cdot \alpha_L} \qquad (10)$$

PR_L ist eine untere (englisch "lower") Grenze für die Restfehlerwahrscheinlichkeit PR; daran erinnert der Index L.

Bei Blockcodierung geht n_0 in n über. Es ist K eine langsam veränderliche Funktion von n_0 und R; der für $R < C_0$ positive Exponentialfaktor α_L hängt ab von R und C_0. Er wächst bei fester Rate mit zunehmender Kapazität C_0 und bei festgehaltener Kapazität mit abnehmender Rate; je kleiner die Rate, eine desto geringere Restfehlerwahrscheinlichkeit ist - einleuchtenderweise - also möglich. Für $R = C_0$ ist α_L gleich Null.

Um die Restfehlerwahrscheinlichkeit herabzudrücken, steht uns nach Gleichung (10) natürlich zunächst der Weg offen, den Kanal zu verbessern, d.h. seine Kapazität zu vergrößern. Ist der Kanal aber vorgegeben, so können wir PR verkleinern, indem wir mit einer kleineren Rate senden, also den Kanal relativ schlecht ausnützen.

Halten wir bei gegebenem Kanal eine Rate $R < C_0$ fest, so ermöglicht die Wahl von n_0 immer noch eine beliebig kleine Restfehlerwahrscheinlichkeit, da PR zumindest für große n_0 exponentiell mit n_0 abnimmt (Kapitel VII, 4). Die dadurch gewonnene Decodiergenauigkeit muß zunächst bezahlt werden mit einer längeren Wartezeit zwischen Ankunft und Decodierung einer Stelle, denn der Decoder muß zur Korrektur einer Stelle alle von dieser Stelle beeinflußten (n_0-1) weiteren Stellen abwarten. Außerdem wird der Aufwand von Coder und Decoder mit n_0 wachsen.

In Kapitel II bzw. IV werden wir sehen, daß der Aufwand für den Coder linear mit n_0 wächst, wenn wir lineare Code verwenden. Der Decodieraufwand steigt dagegen bei allen herkömmlichen Verfahren exponentiell mit n_0 an. Ziel der neuen Verfahren war es, ein geringeres Ansteigen des Decodieraufwandes zu erreichen.

BLOCKCODIERUNG

1. Allgemeinste Blockcodierung

Im gesamten Kapitel II wollen wir uns etwas näher mit Blockcodierung beschäftigen. Die Folge der von der Quelle produzierten Nachrichtenstellen wurde bei Blockcodierung in Nachrichtenworte aufgeteilt, die je m Binärzeichen umfassen. Es existieren

$$M = 2^m \qquad (1)$$

verschiedene Nachrichten-m-Tupel; sie seien durchnumeriert und mit $\bar{x}_1$, $\bar{x}_2$ bis $\bar{x}_M$ bezeichnet. Nach den Voraussetzungen über die Quelle (Kapitel I, 1) treten diese M Nachrichten statistisch unabhängig auf und sind gleichwahrscheinlich. Jedes dieser Nachrichtenworte tritt mit der Wahrscheinlichkeit $(\frac{1}{2})^m$ auf. Der Coder ordnet nach vereinbarten Gesetzen jedem Nachrichten-m-Tupel ein wohlbestimmtes n-Tupel als zugehöriges Codewort zu. Es existieren

$$N = 2^n \qquad (2)$$

verschiedene binäre n-Tupel. Eine gewisse Auswahl von 2^m dieser 2^n verschiedenen n-Tupel werden also Codeworte. Diese Codeworte werden dem Kanal anstelle der Nachrichtenworte zur Übermittlung angeliefert. Die Codeworte seien mit $\bar{u}_1$ bis $\bar{u}_M$ bezeichnet. Die Redundanz des Code liegt gerade darin, daß von den 2^n n-Tupeln nur 2^m als Codeworte benutzt werden.

Das Prinzip, nach dem die Zuordnung von Nachrichtenworten zu Codeworten geschieht, ist festgelegt und auch dem Decoder bekannt. Im primitivsten Fall geschieht die Zuordnung aufgrund einer Codeliste, die z.B. für m=2 und n=5 folgendes Aussehen haben kann:

Beispiel 1)

$\bar{x}_1$	00	01101	$\bar{u}_1$
$\bar{x}_2$	01	10010	$\bar{u}_2$
$\bar{x}_3$	10	00010	$\bar{u}_3$
$\bar{x}_4$	11	11110	$\bar{u}_4$

Auf der linken Seite sind alle $2^m = 4$ möglichen Nachrichten-m-Tupel $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$ und $\bar{x}_4$ aufgeführt, auf der rechten Seite die ihnen zugeordneten Codeworte $\bar{u}_1$ bis $\bar{u}_4$.

Nach der Übermittlung durch den Kanal können die Codeworte gefälscht sein. Das Empfangswort $\bar{v}$ braucht nicht mit dem gesendeten Codewort $\bar{u}$ übereinzustimmen. Den Einfluß der Kanalstörung wollen wir durch ein n-stelliges Fehlermuster $\bar{e}$ beschreiben. Dieses n-Tupel soll an denjenigen Stellen eine "1" haben, an denen der Kanal falsch übertragen hat, ansonsten eine "0". Wurde beispielsweise die 1. und 3. Stelle gefälscht, dann hat $\bar{e}$ bei n = 5 das folgende Aussehen: $\bar{e}$ = (10100). Die Zahl der "1" in einem Block von Binärzahlen bezeichnet man als dessen Gewicht. Das Gewicht f der Fehlerfolge ist gleich der Zahl der vom Kanal gefälschten Stellen. Nehmen wir an, daß das Codewort $\bar{u}_1$ = (01101) aus Beispiel 1) gesendet wurde, dann ergibt sich das empfangene n-Tupel $\bar{v}$ folgendermaßen als modulo-2 Summe von Codewort und Fehlerfolge:

$$\begin{array}{lll} & \bar{u}_1 & 0\,1\,1\,0\,1 \\ \oplus & \bar{e} & 1\,0\,1\,0\,0 \\ \hline & \bar{v} & 1\,1\,0\,0\,1 \end{array}$$

Umgekehrt ergibt sich das Fehlermuster $\bar{e}$ als modulo-2 Summe aus gesuchtem Codewort und empfangenem n-Tupel:

$$\begin{array}{lll} & \bar{u}_1 & 0\,1\,1\,0\,1 \\ \oplus & \bar{v} & 1\,1\,0\,0\,1 \\ \hline & \bar{e} & 1\,0\,1\,0\,0 \end{array}$$

2. Decodierung nach maximaler Rückschlußwahrscheinlichkeit. Die Distanzmatrix

Aufgabe des Decoder ist es, das gesendete Codwort aus dem empfangenen, möglicherweise gefälschten n-Tupel $\bar{v}$ zu bestimmen; wir wollen $\bar{v}$ bezeichnen als Empfangswort.
Der Decoder arbeitet dann mit kleinstmöglicher Restfehlerwahrscheinlichkeit, wenn er zum Empfangswort $\bar{v}$ dasjenige Codewort $\bar{u}_1$ sucht, das die größte Rückschlußwahrscheinlichkeit $P(\bar{u}_i|\bar{v})$ hat. $P(\bar{u}_i|\bar{v})$ bedeutet die Wahrscheinlichkeit dafür, daß $\bar{u}_i$ gesendet wurde,

wenn bekannt ist, daß $\bar{v}$ empfangen wurde. Wir nennen dieses Decodierprinzip MRW-Decodierung: der Decoder entscheidet sich nach maximaler Rückschlußwahrscheinlichkeit. Für diese bedingte Wahrscheinlichkeit gilt nach dem Satz von BAYES:

$$P(\bar{u}_i|\bar{v}) = \frac{P(\bar{u}_i)}{P(\bar{v})} \cdot P(\bar{v}|\bar{u}_i) \qquad (3)$$

$P(\bar{u}_i)$ ist für alle i konstant und gleich 2^{-m} (vgl. Kapitel I, 1 und II, 1). Die Wahrscheinlichkeit $P(\bar{v})$ für den Empfang eines beliebigen - aber festgehaltenen - n-Tupels $\bar{v}$ ist berechenbar und konstant, also unabhängig von der gerade gesendeten Nachricht. Das heißt $P(\bar{u}_i|\bar{v})$ ist nach Gleichung (3) proportional zu $P(\bar{v}|\bar{u}_i)$. Letztere Wahrscheinlichkeit, also die Wahrscheinlichkeit dafür, daß $\bar{v}$ empfangen wird unter der Voraussetzung, daß $\bar{u}_i$ gesendet wurde, läßt sich berechnen. Sie hängt ab von der Zahl der unterschiedlichen Stellen, also von der Zahl der Fehler, die der Kanal hervorrufen muß, um $\bar{u}_i$ in $\bar{v}$ überzuführen. Dieser Abstand von $\bar{v}$ und $\bar{u}_i$ wird die Distanz der beiden n-Tupel genannt; die Distanz ist das Gewicht der stellenweisen modulo-2 Summe von $\bar{u}_i$ und $\bar{v}$. Wir nennen diese modulo-2 Summe $\bar{e}_i$: $\bar{e}_i = \bar{u}_i \oplus \bar{v}$. Die Distanz von $\bar{u}_i$ und $\bar{v}$ sei mit f_i bezeichnet. Für die Rückschlußwahrscheinlichkeit gilt dann:

$$P(\bar{v}|\bar{u}_i) = p_0^{f_i} \cdot q_0^{(n-f_i)} \qquad (4)$$

Der Kanal überträgt f_i-mal richtig und $(n-f_i)$-mal falsch. Da wir nach Kapitel I,2 $p_0 < \frac{1}{2}$ voraussetzen können, nimmt $P(\bar{v}|\bar{u}_i)$ mit wachsendem f_i ab.

Bei MRW-Decodierung vergleicht der Decoder deshalb das empfangene $\bar{v}$ mit allen möglichen Codeworten, ermittelt die f_i und entscheidet sich für das Codewort mit dem kleinsten Wert f_i. In unserem Beispiel 1) wäre:

$\bar{v}$:	1 1 0 0 1	
$\bar{e}_1$:	1 0 1 0 0	$f_1 = 2$
$\bar{e}_2$:	0 1 0 1 1	$f_2 = 3$
$\bar{e}_3$:	1 1 0 1 1	$f_3 = 4$
$\bar{e}_4$:	0 0 1 1 1	$f_4 = 3$

Der Decoder würde sich für das Codewort $\bar{u}_1$ entscheiden, also richtig decodieren. Die Zahl der Kanalstörungen kann nun so groß werden, daß das empfangene $\bar{v}$ einem nicht gesendeten Codewort ähnlicher wird als dem gesendeten. Dann fällt der Decoder eine falsche Entscheidung. Die Zahl der Fehlentscheidungen hängt ab von den Distanzen der Codeworte untereinander. Diese Distanzen wollen wir als quadratische Matrix anordnen; dabei soll $d_{i,j}$ die Distanz der beiden Codeworte $\bar{u}_i$ und $\bar{u}_j$ bedeuten. Für alle i, j gilt: $d_{i,j} = d_{j,i}$ und $d_{i,i} = 0$. Deshalb hat die Distanzmatrix das folgende Aussehen:

	$\bar{u}_1$	$\bar{u}_2$	$\bar{u}_3$	...	$\bar{u}_M$
$\bar{u}_1$	0	$d_{1,2}$	$d_{1,3}$	...	$d_{1,M}$
$\bar{u}_2$	$d_{1,2}$	0	$d_{2,3}$	...	$d_{2,M}$
.	.				
.	.				
.	.				
$\bar{u}_M$	$d_{1,M}$	$d_{2,M}$	$d_{3,M}$	...	0

Wir wollen die Zahl der Codeworte mit Abstand f vom Codewort $\bar{u}_i$ bezeichnen mit $N_i(f)$, dabei sei das Codewort $\bar{u}_i$ selbst mitgezählt als Wort mit der Distanz 0. Die Distanzfunktion eines jeden Codewortes $\bar{u}_i$ erfüllt die Gleichung $\sum_{f=0}^{n} N_i(f) = M$. Wir werden sehen, daß die Restfehlerwahrscheinlichkeit eines Code bestimmt ist durch seine Distanzfunktionen.

Die kleinste in der Distanzmatrix auftretende Zahl außerhalb der Hauptdiagonalen wird als HAMMING-Distanz h bezeichnet. Der Decoder entscheidet bei MRW-Decodierung immer richtig, wenn die Zahl der Fehler kleiner ist als die halbe HAMMING-Distanz. Ist die Zahl der Fehler nicht kleiner als $\frac{h}{2}$, entstehen möglicherweise Decodierfehler.

Der Decodiervorgang bei der MRW-Decodierung läßt sich folgendermaßen deuten: die Gesamtheit der Binärfolgen der Länge n wird in 2^m Familien eingeteilt. In jeder Familie liegt genau ein Codewort und außerdem diejenigen n-Tupel, die zum Familiencodewort geringeren Abstand haben als zu anderen Codeworten. Hat ein n-Tupel zu verschiedenen Codeworten gleiche Distanz, so wird es willkürlich einer der Familien zugeordnet. Bei MRW-Decodierung wird nun jedem Familienmitglied das Familiencodewort zugeordnet. Ein Decodierfehler entsteht immer dann, wenn das gesendete Codewort durch Kanalstörung so sehr verfälscht wird, daß das empfangene Wort im Bereich einer anderen Familie liegt.

3. Der Gruppencode

Die Zuteilung der n-Tupel zu den m-stelligen Nachrichtenworten war bisher willkürlich. Eine Möglichkeit zur systematischen Auswahl benutzen die linearen Code , die auch Gruppencode genannt werden. Die Nachrichten sind vereinbarungsgemäß m-stellige Binärtupel. Wir wählen nun m zunächst beliebige "Basiscodeworte" aus der Menge der n-Tupel aus. Im obigen Beispiel 1) war n = 5 und m = 2. Wir wählen etwa die beiden folgenden Basiscodeworte aus für Beispiel 2a):

Nachrichtenstelle	Basiscodeworte
1	1 0 0 1 1
2	1 1 1 1 0

Bei linearen Code erhalten wir die Codeworte als stellenweise modulo-2 Summe einer gewissen, für jede Nachricht charakteristischen Auswahl von Basiscodeworten. Dabei sollen die Nachrichtenstellen anzeigen, welche Basiscodeworte in die Summe eingeschlossen werden.

Ist die i-te Nachrichtenstelle eine "1", so wird das i-te Basiscodewort in die Summe eingeschlossen; ist sie eine "0", so wird es nicht bei der Summenbildung berücksichtigt oder, was dasselbe bedeutet, es wird dann das n-Tupel (0 0 ..... 0) addiert. Damit erhalten wir für Beispiel 2a) die folgende Aufstellung. In der Tafel sind links die Nachrichten $\bar{x}_i$, in der Mitte die entsprechenden

Codeworte $\bar{u}_i$ und rechts neben diesen die Gewichte w_i der Codeworte aufgeführt.

$\bar{x}_1$	0 0	$\bar{u}_1$	0 0 0 0 0	$w_1 = 0$
$\bar{x}_2$	0 1	$\bar{u}_2$	1 0 0 1 1	$w_2 = 3$
$\bar{x}_3$	1 0	$\bar{u}_3$	1 1 1 1 0	$w_3 = 4$
$\bar{x}_4$	1 1	$\bar{u}_4$	0 1 1 0 1	$w_4 = 3$

Wir nennen ein Nachrichtenwort, das nur eine einzige "1" enthält Basisnachrichtenwort. Den Basisnachrichtenworten sind die Basiscodeworte zugeordnet. Wir sehen außerdem, daß der m-stelligen Nachricht (0 0 .... 0) bei einem Gruppencode immer das n-Tupel (0 0 .... 0) als Codewort entspricht. Aufgrund unserer Überlegungen müssen wir bei der Einteilung der Codeworte in Familien vermeiden, daß einer anderen Nachricht ebenfalls das Null-Codewort zugeordnet wird. Das bedeutet,

a) daß das Nullwort kein Basiscodewort sein darf und
b) daß die Basiscodeworte voneinander linear unabhängig sein müssen. Keine Kombination dieser Basisworte darf modulo-2 zusammenaddiert das Nullwort ergeben.

Auf folgende Eigenschaften eines Gruppencode sei noch hingewiesen:

a) Die Summe zweier Codeworte ist wieder ein Codewort, weshalb man statt der gewählten Basiscodeworte auch irgendwelche anderen linear unabhängigen Codeworte dieses Code wählen darf und trotzdem dieselbe Gesamtheit von Codeworten erhält.

b) Die HAMMING-Distanz h eines Gruppencode ist einfach das kleinste auftretende Gewicht w eines vom Nullwort verschiedenen Codewortes. In Beispiel 2a) ist h = 3.

c) Für die Distanzfunktionen $N_i(f)$ gilt bei Gruppencode , daß sie für alle i gleich sind, weshalb wir den Index i wegfallen lassen und sie mit N(f) bezeichnen.

Wir können uns die Codeworte auch durch eine modulo-2-Matrixmultiplikation erzeugt denken:

$$\bar{u} = (\bar{x} \cdot G)_{mod\text{-}2} \qquad (5)$$

dabei werden $\bar{u}$ und $\bar{x}$ als einzeilige Matrizen - also Zeilenvektoren - aufgefaßt:

$$\bar{x} = (x_1, x_2, \ldots, x_m), \qquad \bar{u} = (u_1, u_2, \ldots, u_n).$$

Die Matrix G hat m Zeilen und n Spalten. Ihre m Zeilen sind gerade die Basiscodeworte. Für Beispiel 2a) lautet die Matrix also:

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

4. Der systematische Code und sein Prüfschema

Sind die Basiscodeworte linear unabhängig, dann können wir die Matrix durch lineare Zeilentransformationen und Vertauschen von Spalten so verändern, daß die ersten m Spalten Diagonalform haben. In unserem Beispiel lassen wir die erste Zeile unverändert stehen und addieren diese erste Zeile zur zweiten.
Damit erhalten wir die Matrix unseres Beispiels 2b):

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

Wenn wir diese Matrix G_b statt G_a benutzen, bedeutet dies eine andere Wahl der Basiscodeworte. Wir erhalten aber - wie oben bemerkt - dieselben Codeworte, bloß in anderer Reihenfolge und evtl. mit vertauschter Stellenfolge. Wir haben jetzt erreicht, daß die ersten m Stellen des Codewortes - man wählt meistens die ersten m, könnte aber genauso eine andere Auswahl der n Codewortstellen wählen - mit den Nachrichtenstellen übereinstimmen. Diese Nachrichtenstellen werden im Coder durch k Kontrollstellen ergänzt. Die gewonnene Darstellung nennt man systematisch. Die Codeliste von Beispiel 2b) hat jetzt folgendes Aussehen:

$\bar{x}_1$	0 0	$\bar{u}_1$	0 0	0 0 0
$\bar{x}_2$	0 1	$\bar{u}_2$	0 1	1 0 1
$\bar{x}_3$	1 0	$\bar{u}_3$	1 0	0 1 1
$\bar{x}_4$	1 1	$\bar{u}_4$	1 1	1 1 0

Es ist eine Umordnung der Codewortliste von Beispiel 2a). Systematische Code können wir mit einer anderen Matrix beschreiben. Die alte Darstellung lautete: $\bar{u} = (\bar{x} \cdot G_o)_{mod\text{-}2}$ oder ausführlich

$$(u_1, u_2, \ldots u_n) = (x_1, x_2, \ldots, x_m) \underbrace{\begin{bmatrix} 1 & 0 & 0 & \ldots & 0 & p_{11} & p_{21} & \cdots & p_{k1} \\ 0 & 1 & 0 & \ldots & 0 & p_{12} & p_{22} & \cdots & p_{k2} \\ 0 & 0 & 1 & \ldots & 0 & p_{13} & p_{23} & \cdots & p_{k3} \\ \vdots & & & & & & & & \\ 0 & 0 & 0 & \ldots & 1 & p_{1m} & p_{2m} & \cdots & p_{km} \end{bmatrix}}_{\substack{\longleftarrow n \longrightarrow \\ \longleftarrow m \longrightarrow \quad \longleftarrow k \longrightarrow \\ mod\text{-}2}}$$

Der linke Teil von G_o ist eine m-zeilige Einheitsmatrix. Sie sorgt dafür, daß die Nachrichtenstellen unverändert bleiben, also daß $u_i = x_i$ für $1 \leq i \leq m$ ist. Der rechte Teil bestimmt, wie die Kontrollstellen aus den Nachrichtenstellen berechnet werden. Die (m+1)-te Spalte von G_o bestimmt die 1. Kontrollstelle, die (m+j)-te Spalte die j-te Kontrollstelle. Es gilt aufgrund der Matrixmultiplikation für die 1. Kontrollstelle:

$$u_{m+1} = p_{11} x_1 \oplus p_{12} x_2 \oplus \ldots \oplus p_{1m} x_m = \Big(\sum_{\ell=1}^{m} p_{1\ell}\, x_\ell\Big)_{mod\text{-}2} \qquad (6)$$

oder allgemein für die i-te Kontrollstelle

$$u_{m+i} = p_{i1} x_1 \oplus p_{i2} x_2 \oplus \ldots \oplus p_{im} x_m = \Big(\sum_{\ell=1}^{m} p_{i\ell}\, x_\ell\Big)_{mod\text{-}2} \qquad (7)$$

Zusammenfassend können wir schreiben:

$$u_i = \begin{cases} x_i & \text{für } 1 \leq i \leq m \\ \Big(\sum_{\ell=1}^{m} p_{i\ell}\, x_\ell\Big)_{mod\text{-}2} & m < i = m + k \leq n \end{cases} \qquad (8)$$

Wir wollen die Gleichung für die Kontrollstellen - also für $m < i \leq n$ - noch umschreiben und beachten dabei, daß in der

modulo-2 Rechnung Addition und Subtraktion gleichbedeutend sind (Kapitel I)

$$u_i \oplus \left(\sum_{\kappa=1}^{m} p_{i\kappa} x_\kappa\right)_{\text{mod-2}} = 0 \qquad \text{für } m < i \leq n \qquad (9)$$

Das bedeutet in Worten: die Kontrollstellen sind so zu wählen, daß eine bestimmte, für jede Kontrollstelle charakteristische Auswahl von Nachrichtenstellen zusammen mit dieser Kontrollstelle geradzahliges Gewicht hat, d.h. die modulo-2 Summe dieser Stellen muß 0 sein. Die ausgewählten Nachrichtenstellen werden durch die Kontrollstelle auf eine gerade Zahl von "1" ergänzt, weshalb in der amerikanischen Literatur von "parity-check" gesprochen wird. Besonders deutlich wird diese Interpretation durch das Prüfschema P, das sich direkt aus der Matrix eines systematischen Gruppencode ablesen läßt:

$$P = \underbrace{\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} & 1 & 0 & \cdots & 0 \\ p_{21} & p_{22} & \cdots & p_{2m} & 0 & 1 & \cdots & 0 \\ \vdots & & & & & & & \\ p_{k1} & p_{k2} & \cdots & p_{km} & 0 & 0 & \cdots & 1 \end{bmatrix}}_{n} \Bigg\} k$$

$$\underbrace{\qquad\qquad}_{m} \quad \underbrace{\qquad}_{k}$$

Die i-te Zeile des Prüfschemas gibt an, welche Auswahl der Nachrichtenstellen zusammen mit der i-ten Kontrollstelle geradzahliges Gewicht haben muß. Für Beispiel 2b) lautet das Prüfschema:

$$P = \left[\begin{array}{cc|ccc} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

Es ist demnach die erste Kontrollstelle so zu wählen, daß die modulo-2 Summe aus ihr und der zweiten Nachrichtenstelle gleich Null ist. Entsprechend müssen die zweite Kontrollstelle und die erste Nachrichtenstelle übereinstimmen. Die dritte Kontrollstelle haben wir so zu wählen, daß die modulo-2 Summe von erster und zweiter Nachrichtenstelle und dieser dritten Kontrollstelle gleich Null ist.

5. Der Prüfschemacode

Der systematische Code kann als Spezialfall einer anderen größeren Klasse von Code gesehen werden. Diese Code arbeiten mit einem verallgemeinerten Prüfschema, weshalb wir sie Prüfschemacode nennen wollen. Hier sieht das Prüfschema folgendermaßen aus:

$$P = \underbrace{\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & & & \\ p_{z1} & p_{z2} & \cdots & p_{zn} \end{bmatrix}}_{n} \Bigg\} z$$

Die letzten k Spalten sind hier im allgemeinen keine Einheitsmatrix. Damit sind die z Zeilen des Prüfschemas nicht notwendig voneinander unabhängig, wie es beim systematischen Code durch die Einheitsmatrix in der rechten Hälfte gesichert war. Nehmen wir als Beispiel 3a) folgendes Prüfschema für n=9 und z=6:

$$P = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Wir sehen sofort, daß die Summe aller Prüfzeilen das Null-n-Tupel ergibt: die Zeilen sind linear abhängig. Das bedeutet, wie wir gleich sehen werden, daß wir nicht - wie das Prüfschema andeutet - (n-z)-Nachrichtenstellen zur Verfügung haben, sondern mehr als (n-z).

Ein n-Tupel $\bar{u}$ ist nach Abschnitt 4 genau dann Codewort, wenn seine Stellen u_1 bis u_n die z Gleichungen

$$\begin{aligned} p_{11}\, u_1 \oplus p_{12}\, u_2 \cdots \oplus p_{1n}\, u_n &= 0 \\ p_{21}\, u_1 \oplus p_{22}\, u_2 \cdots \oplus p_{2n}\, u_n &= 0 \\ &\vdots \\ p_{z1}\, u_1 \oplus p_{z2}\, u_2 \cdots \oplus p_{zn}\, u_n &= 0 \end{aligned} \qquad (10)$$

erfüllen.

Die Codeworte stellen die Lösungen dieses Gleichungssystems dar. Die Zahl k der voneinander unabhängigen Prüfschemazeilen, der Rang der Prüfschema-Matrix, diese Zahl k bestimmt die Zahl der nicht frei wählbaren Stellen in der allgemeinen Lösung dieses Gleichungssystems; die Zahl der frei wählbaren Stellen ist (n-k). Diese m=(n-k) frei wählbaren Stellen sind unsere Nachrichtenstellen. Sind sie festgelegt, so lassen sich aus ihnen die übrigen k Stellen, die Kontrollstellen, berechnen. Da das Prüfschema nur eine Kurzschreibweise dieses linearen Gleichungssystems darstellt, ist sofort klar, daß man die Zeilen des Prüfschemas linear transformieren darf, ohne die Codeworte, also die Lösungen des Gleichungssystems, zu ändern. Das bedeutet, daß wir durch Zeilenumformungen das allgemeine Prüfschema in eine systematische Form umwandeln können, aus dem man für gegebene Nachrichtenstellen die k Kontrollstellen direkt ausrechnen kann. Unser Beispiel 3a) liefert durch Zeilenumformung des Prüfschemas die systematische Matrix P_0 für Beispiel 3b):

$$P_0 \quad \begin{bmatrix} 0&0&0&0&0&0&0&0&0 \\ 1&1&1&0&0&0&0&0&0 \\ 0&0&0&1&1&1&0&0&0 \\ 1&0&0&1&0&0&1&0&0 \\ 0&1&0&0&1&0&0&1&0 \\ 1&1&0&1&1&0&0&0&1 \end{bmatrix}$$

Nachrichtenstelle	1	2		3	4			
Kontrollstelle			1			2	3	4	5

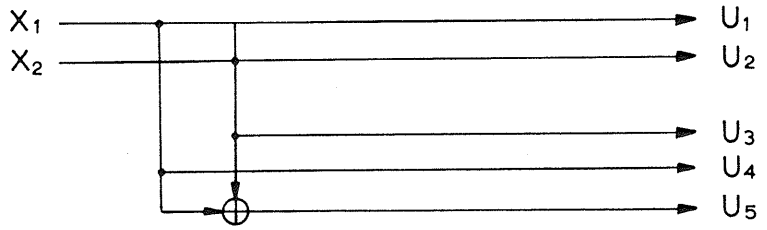
Da die erste Zeile des Prüfschemas P_0 aus lauter Nullen besteht, ist der Rang k der Prüfschema-Matrix nur 5. Wir haben also n-k= 9 - 5 = 4 Nachrichtenstellen zur Verfügung, deren Verteilung innerhalb der n Codewortstellen aus der Matrix leicht zu erkennen ist.

Bei dieser Form der Matrix müssen wir darauf achten, daß die Prüfstellen sich wirklich nur aus den Nachrichtenstellen ergeben. Die Spalten, die den Kontrollstellen entsprechen, dürfen also - wie in unserem Beispiel 3b) - nur eine einzige "1" enthalten. Das ist durch Zeilenumformungen immer zu erreichen. Im Beispiel 3b) muß Kontrollstelle 1 so gewählt werden, daß sie zusammen mit der ersten und zweiten Nachrichtenstelle die modulo-2 Summe 0 hat.

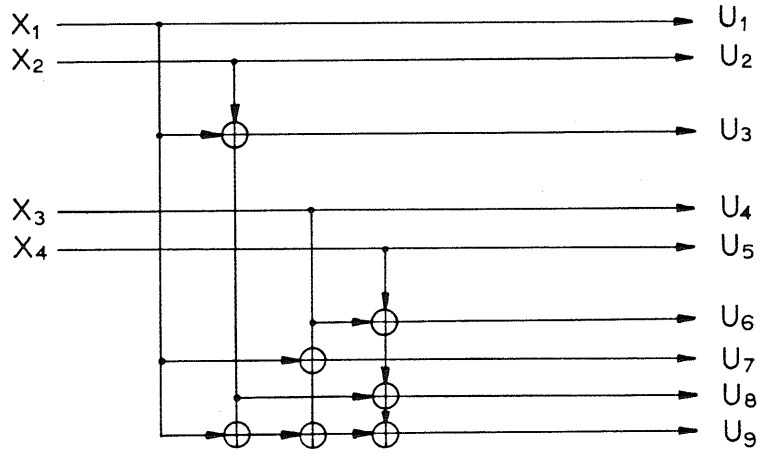
6. Der Coder eines Prüfschemacode

Das Codieren mit Hilfe eines systematischen Prüfschemas läßt sich leicht instrumentieren. Das Schaltbild unseres Beispiels 2b) zeigt Figur 2.1.

Das Schaltbild für den systematischen Code von Beispiel 3b) zeigt Figur 2.2.



Figur 2.1: Schaltbild eines systematischen Code mit den Codegrößen m=2, k=3 nach Beispiel 2b).



Figur 2.2: Schaltbild eines systematischen Code mit den Codegrößen m=4, n=9 nach Beispiel 3b).

Wir sehen, daß zur Instrumentierung eines systematischen Code nur die Speicherelemente benötigt werden, mit deren Hilfe die in Serie angelieferten Nachrichtenstellen auf m Leitungen parallel gewandelt und hinter dem Coder dann wieder auf eine Leitung gegeben werden. Der Codieraufwand wächst also linear mit n. Bei den von PETERSON[6] behandelten zyklischen Code , einer speziellen Klasse von systematischen Code , genügen sogar k Flip-Flop bei Verwendung eines Seriencoder.

Das Codierproblem ist also durch die systematischen Code befriedigend gelöst.

7. Aufwand des Decoder

Wie arbeitet nun der Decoder eines systematischen Code bei MRW-Decodierung?

Der Decoder erhält die Empfangsfolge $\bar{v}$, die sich möglicherweise wegen Übertragungsfehlern vom gesendeten Codewort $\bar{u}$ unterscheidet.

Der Decoder muß aus dem empfangenen n-Tupel $\bar{v} = (v_1, v_2, \ldots v_n)$ das richtige Fehlermuster zu bestimmen versuchen. Er vergleicht dazu $\bar{v}$ - wie wir zu Anfang des Kapitels gesehen haben - mit den 2^m Codeworten und erhält so 2^m mögliche Fehlermuster. Aus diesen 2^m Fehlermustern wählt er dasjenige mit dem geringsten Gewicht aus. Dieser Sachverhalt soll als Abschluß dieses Kapitels noch in Gleichungen gefaßt werden, um zu einer möglichst einfachen Instrumentierung zu kommen.

Der Decoder arbeitet in einem vorbereiteten Schritt zunächst wie ein Coder. Er berechnet die zu den empfangenen m Nachrichtenstellen v_1 bis v_m die gehörenden k Kontrollstellen. Das ergibt für i = m+1, m+2, ..., m+k die Werte

$$t_i = \left(\sum_{\ell=1}^{m} p_{i,\ell}\ v_\ell\right)_{\text{mod-2}} \qquad (11)$$

Diese errechneten Kontrollstellen vergleicht der Decoder mit den tatsächlich eingetroffenen Stellen $v_{m+1}, v_{m+2}, v_{m+3}, \ldots v_n$ mittels eines modulo-2 Addierers. Das ergibt die k Prüfwerte:

$$s_i = t_i \oplus v_i = \left(\sum_{\ell=1}^{m} p_{i,\ell}\ v_\ell\right)_{\text{mod-2}} \oplus v_i \qquad (12)$$

$$i = m+1, m+2, \ldots, n$$

Ist das Empfangswort $\bar{v}$ ein Codewort, was bei fehlerfreier Übertragung bestimmt der Fall ist, dann sind alle Prüfwerte $s_i = 0$.

Um den Einfluß des Fehlermusters zu untersuchen, setzen wir ein

$$v_i = u_i \oplus e_i \qquad i = 1, 2, \ldots, n \qquad (13)$$

$$s_i = \left(\sum_{\ell=1}^{m} p_{i,\ell}\ u_\ell\right)_{\text{mod-2}} \oplus u_i \oplus \left(\sum_{\ell=1}^{m} p_{i,\ell}\ e_\ell\right)_{\text{mod-2}} \oplus e_i \qquad (14)$$

Da $\bar{u}_i$ ein Codewort ist, gilt

$$\left(\sum_{\ell=1}^{m} p_{i,\ell}\ u_\ell\right)_{\text{mod-2}} \oplus u_i = 0 \qquad i = m+1, m+2, \ldots, n \qquad (15)$$

und die Gleichung (14) vereinfacht sich zu

$$s_i = \left(\sum_{\ell=1}^{m} p_{i,\ell}\ e_\ell\right)_{\text{mod-2}} \oplus e_i \qquad i = m+1, \ldots n \qquad (16)$$

Diese k Gleichungen fassen wir auf als lineares Gleichungssystem für die n Unbekannten $e_1, e_2, \ldots, e_n$. Die Lösung des Systems hat n-k = m Parameter. Als diese wählen wir versuchsweise e_1 bis e_m. Für jede bestimmte Wahl dieser Parameter ergeben sich die restlichen Fehlerstellen $e_{m+1}, \ldots, e_n$ aus den Gleichungen:

$$e_i = s_i \oplus \left(\sum_{\ell=1}^{m} p_{i,\ell} \cdot e_\ell\right)_{\text{mod-2}} \qquad i = m+1, m+2, \ldots, n \qquad (17)$$

Will der Decoder eine möglichst gute Entscheidung treffen, dann wählt er aus diesen 2^m möglichen Fehlermustern - sie ergeben sich aus dem Vergleich mit den 2^m Codeworten - dasjenige aus, das geringstes Gewicht hat. Dieses Fehlermuster $\bar{e}^*$ - es stimmt mit $\bar{e}$ dann überein, wenn richtig decodiert wird - addiert der Decoder zu $\bar{v}$ hinzu und erhält $\bar{v}^*$. Die ersten m Stellen von $\bar{v}^*$ sind dann gerade $\bar{y}$, die gesuchte Reproduktion der Nachricht $\bar{x}$. Sind bei systematischen Code nicht die ersten m, sondern eine andere Auswahl von m Stellen identisch mit den Nachrichtenstellen, dann arbeitet der Decoder analog zum eben besprochenen Normalfall.

Ein zweites Arbeitsprinzip für die Decodieranlage ist es, für alle möglichen 2^k Wertekombinationen der k Prüfwerte s_i die Lösungen zu speichern. In beiden Fällen wächst der Aufwand des MRW-Decoder bei konstanter Rate R exponentiell mit n; das ist für große n untragbar.

Bei jedem anderen Decodierverfahren muß ein Kompromiß geschlossen werden zwischen geringem Decodieraufwand und großer Decodiergenauigkeit. Dabei ist ein Decoder wünschenswert, der verhältnismäßig geringen Aufwand an Schaltelementen, Speicherplätzen und Rechenschritten erfordert, selbst wenn dadurch die Restfehlerwahrscheinlichkeit bei fester Rate und fester Codelänge sich etwas vergrößern sollte. Wenn wir eine geringere Restfehlerwahrscheinlichkeit fordern müssen, dann können wir das durch Vergrößerung der Codelänge oder durch Herabsetzung der Nachrichtenrate R erreichen.

LD-CODE

1. Definition der (n, j, l)-Code. Ihre Codierung und Struktur

Low-Density Parity-Chek Code, wir wollen sie LD-Code nennen, sind Prüfschemacode; wir haben also Blockcodierung vor uns. Wie der englische Name andeutet, arbeitet GALLAGER mit einem Prüfschema, das wenig "1"-Stellen enthält. GALLAGER's Decodierprinzip ist zugeschnitten auf den Spezialfall der von ihm so bezeichneten (n, j, l)-Code . Sie haben die Blocklänge n; ihr Prüfschema hat also n Spalten, und es ist so gebaut, daß jede Spalte j "1"-Stellen und jede Zeile l "1"-Stellen enthält, ansonsten nur "0"-Stellen. Bezeichnen wir die Zahl der Zeilen mit z, so muß für die Gesamtzahl der Einsen im Prüfschema gelten: $n \cdot j = z \cdot l$, und damit erhalten wir die Zeilenzahl zu

$$z = \frac{j}{l} \cdot n \qquad (1)$$

In Kapitel II, 5 haben wir gesehen, daß beim allgemeinen Prüfschemacode durchaus die Zeilenzahl z nicht mit der Zahl k der Kontrollstellen übereinstimmen muß. Die Zahl der Kontrollstellen ist gegeben durch den Rang der Prüfschemamatrix. Sind die Prüfschemazeilen voneinander linear abhängig, so gilt $z > k$ oder $m = n-k > n-z$. Führen wir die Nachrichtenrate $R = \frac{m}{n}$ ein, dann gilt für den allgemeinen Prüfschemacode $R \geq 1 - \frac{z}{n}$ oder für einen (n, j, l)-Code

$$R \geq 1 - \frac{j}{l} \qquad (2)$$

Bei der Behandlung der Prüfschemacode in Kapitel II, 5 wurde als Beispiel 3a) bereits ein (n, j, l)-Code verwandt mit n=9, j=2 und l=3. Wir haben gesehen, daß in diesem Beispiel die Informationsrate $R = \frac{m}{n} = \frac{4}{9}$, also größer als $1 - \frac{z}{n} = 1 - \frac{2}{3} = \frac{1}{3}$ war.

Die (n, j, l)-Code sind spezielle Prüfschemacode, deren allgemeines Prüfschema nach Kap. II, 5 in ein systematisches Prüfschema mit eingestreuten Prüfstellen entwickelt werden kann. Mit dem

systematischen Prüfschema werden die k Kontrollstellen aus den m=n-k Nachrichtenstellen berechnet.

Durchsetzt mit Kontrollstellen, werden die unveränderten Nachrichtenstellen dem Kanal zur Übermittlung angeliefert. Da der Codieraufwand für einen systematischen Code linear mit n steigt (Kapitel II, 7), ist das Codierproblem für GALLAGER's (n, j, l)-Code - auch für große n - befriedigend gelöst. Ein mögliches Schaltbild für den als Beispiel gewählten (9, 2, 3)-Code wurde in Fig. 2.2, Kapitel II, gezeigt.

Für die (n, j, l)-Code werden wir die drei von GALLAGER entwickelten Decodierverfahren kennenlernen.

Der Decoder nach dem dritten Verfahren erfordert zwar den größten Aufwand, dafür ist er wirksamer als die beiden ersten in dem Sinn, daß er die vom Kanal angelieferte Information besser ausnützt, also auch eine größere Decodiergenauigkeit erreicht.

Bevor wir uns den Decodierverfahren zuwenden können, müssen wir uns die Struktur der LD-Code etwas näher ansehen. Dazu wählen wir als Beispiel 1) den bei GALLAGER[3)] angegebenen Code für n=20, j=3 und l=4; sein Prüfschema zeigt Figur 3.1.

Codewortstelle:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
P =	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	g_1
	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	g_2
	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	g_3
	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	g_4
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	g_5
	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	g_6
	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	g_7
	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	g_8
	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	g_9
	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	g_{10}
	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	g_{11}
	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	g_{12}
	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	g_{13}
	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	g_{14}
	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	g_{15}

Figur 3.1: Prüfschema eines (n,j,l)-Code mit den Codegrößen n=20, j=3, l=4

In ihr sind rechts die z Prüfgleichungen g_i durchnumeriert von 1 bis 15 und oben die n Codewortstellen von 1 bis 20. Aus der Bauart des Prüfschemas ist sofort klar, daß jede Stelle genau j Prüfgleichungen beeinflußt. Bei einem einzigen Fehler in den n Stellen werden genau j Prüfgleichungen nicht erfüllt. Wir wollen annehmen, daß Stelle 1 gefälscht wurde, also $e_1 = 1$ ist und alle anderen Stellen korrekt übertragen wurden, also $e_i = 0$ für $i = 2, 3, \ldots, n$. Untersuchen wir nun die sich ergebenden Prüfgleichungen, so sind die Gleichungen g_1, g_6 und g_{11} nicht erfüllt. Wir sehen, daß x_1 von drei nicht erfüllten Prüfgleichungen erfaßt wird, während jede andere Stelle in höchstens einer nicht erfüllten Gleichung auftaucht. Auf dieser Eigenschaft der (n, j, l)-Code bauen sich GALLAGER's Decodierverfahren auf.

2. Das erste Decodierverfahren von GALLAGER

Dem Decoder wird das empfangene n-Tupel $\bar{v}$, das aufgrund der Kanalstörungen möglicherweise nicht mit dem gesendeten Codewort $\bar{u}$ übereinstimmt, angeliefert; der Decoder soll nun dieses Empfangswort aufgrund von festgestellten Unstimmigkeiten so lange verändern, bis es mit einem Codewort übereinstimmt. Dabei wird wie folgt vorgegangen:

Im ersten Schritt stellt der Decoder die Prüfgleichungen auf, führt also in der Matrix von Figur 3.1 die Zeilenprobe durch. Sind alle Prüfgleichungen erfüllt, das Empfangswort ist dann ein Codewort, so leitet er das n-Tupel direkt an die Datensenke weiter. Sind die z Prüfgleichungen nicht alle erfüllt, ist das empfangene n-Tupel also kein Codewort, so muß der Decoder Korrekturen durchführen. Dazu stellt er fest, wie oft jede einzelne Stelle in nicht erfüllten Prüfgleichungen auftaucht. Der Decoder ändert alle diejenigen Stellen ab, die in mehr als b nicht erfüllten Prüfgleichungen vorkommen. Die Schranke b ist dabei vorgegeben. Sollte keine Stelle in mehr als b nicht erfüllten Prüfgleichungen auftreten, obwohl das Empfangswort kein Codewort ist, dann erniedrigt der Decoder die Schranke um 1 und ändert diejenigen Stellen ab, die in (b-1) nicht erfüllten Gleichungen auftauchen. Sollte auch jetzt keine

Stelle korrigiert werden können, dann erniedrigt der Decoder die Schranke um jeweils den Wert 1 so lange, bis er mindestens eine Stelle korrigieren kann.

Durch diese Korrekturen geht das Empfangswort $\bar{v}$ über in das n-Tupel $\bar{v}^{(1)}$. Dieses wird mittels der Prüfgleichungen von neuem untersucht; ist es ein Codewort, dann ist der Decodierprozeß beendet. An dem folgenden Beispiel 2) gemäß dem Code nach Figur 3.1 wird ein solcher Fall demonstriert. Da wir einen Gruppencode vor uns haben, können wir ohne Vernachlässigung der Allgemeinheit immer annehmen, daß das Nullcodewort gesendet wurde.

In der folgenden Aufstellung sind: $\bar{u}$ das gesendete Codewort, $\bar{e}$ das aktuelle Fehlermuster und $\bar{v}$ das aus beiden resultierende n-Tupel, also das Empfangswort. Im n-Tupel $b^{(\nu)} = (b_1^{(\nu)}, b_2^{(\nu)}, \ldots, b_n^{(\nu)})$ geben die Komponenten $b_i^{(\nu)}$ an, von wievielen nicht erfüllten Prüfgleichungen die Stelle i vor dem ν-ten Korrekturschritt erfaßt wird.

Stelle:		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$\bar{u}$	=	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$\bar{e}$	=	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
$\bar{v}$	=	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
$\bar{b}^{(1)}$	=	3	2	1	2	1	1	1	0	2	1	3	2	1	0	1	1	0	1	1	0
$\bar{v}^{(1)}$	=	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Der Vergleich mit dem Prüfschema von Figur 3.1 zeigt, daß $\bar{v}$ die Prüfgleichungen g_1, g_3, g_6, g_9, g_{11} und g_{12} nicht erfüllt. Es sei gegeben b=2; damit werden gerade die Stelle v_1 und v_{11} umgedreht, da $b_1^{(1)}$ und $b_{11}^{(1)}$ die Schranke b=2 übersteigen. Es wurde also richtig decodiert: $\bar{v}^{(1)}$ stimmt mit $\bar{u}$ überein. Die zweite Kontrollrechnung ergibt folglich, daß alle Prüfgleichungen erfüllt sind.

Wäre $\bar{v}^{(1)}$ aber kein Codewort, sind also nicht alle Prüfgleichungen erfüllt, dann dreht der Decoder im notwendig gewordenen zweiten Decodierschritt alle diejenigen Stelle von $\bar{v}^{(1)}$ um, die jetzt in mehr als b nicht erfüllten Gleichungen vorkommen, und erhält dadurch $\bar{v}^{(2)}$. Ist auch dieses n-Tupel noch kein Codewort, so fährt der Decoder in analoger Weise fort und kommt zu $\bar{v}^{(3)}$. Der Decoder

wiederholt dieses Verfahren so lange, bis er ein Codewort erhält. Dieses liefert er an der Datensenke ab.

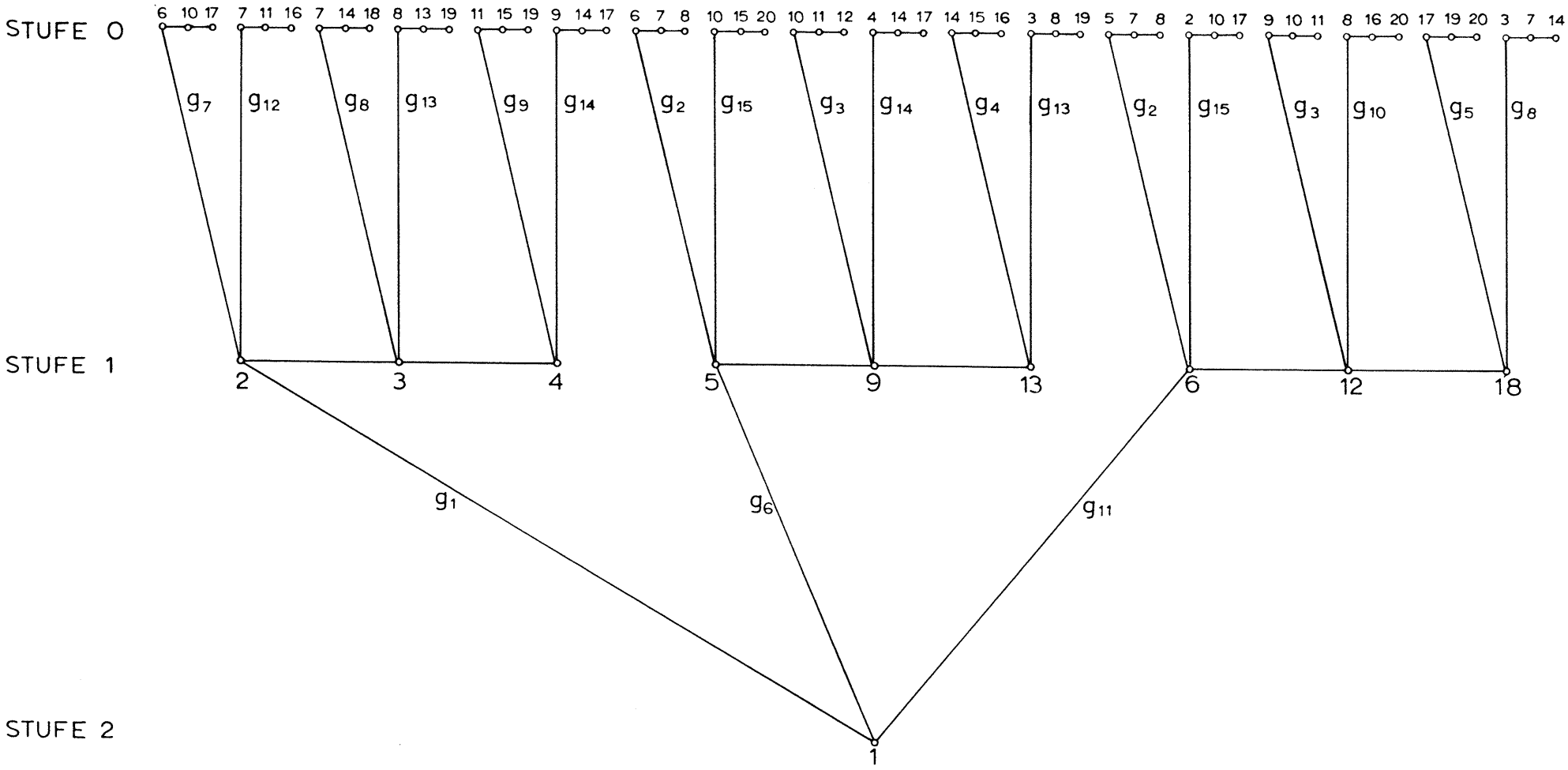
Um die erste Korrektur durchführen zu können, müssen wir - wie oben beschrieben - die Schranke b=2 um 1 erniedrigen, da alle $b^{(1)} < 3$ sind. Im folgenden Beispiel 3) ist die erste Korrektur $\bar{v}^{(1)}$ noch kein Codewort; erst die zweite Korrektur $\bar{v}^{(2)}$ erfüllt alle Prüfgleichungen: $\bar{v}^{(2)}$ ist Codewort. Da $\bar{v}^{(2)}$ mit $\bar{u}$ übereinstimmt, wurde richtig decodiert. Den Korrekturvorgang beschreibt die folgende Aufstellung:

Stelle:		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
$\bar{u}$	=	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
$\bar{e}$	=	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
$\bar{v}$	=	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
$\bar{b}^{(1)}$	=	2	0	0	2	1	1	0	0	2	0	1	1	1	1	1	0	1	1	1	0	
$\bar{v}^{(1)}$	=	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1. Korrektur
$\bar{b}^{(2)}$	=	1	0	0	1	1	0	0	0	3	1	1	1	1	1	0	0	1	0	0	0	
$\bar{b}^{(2)}$	=	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2. Korrektur

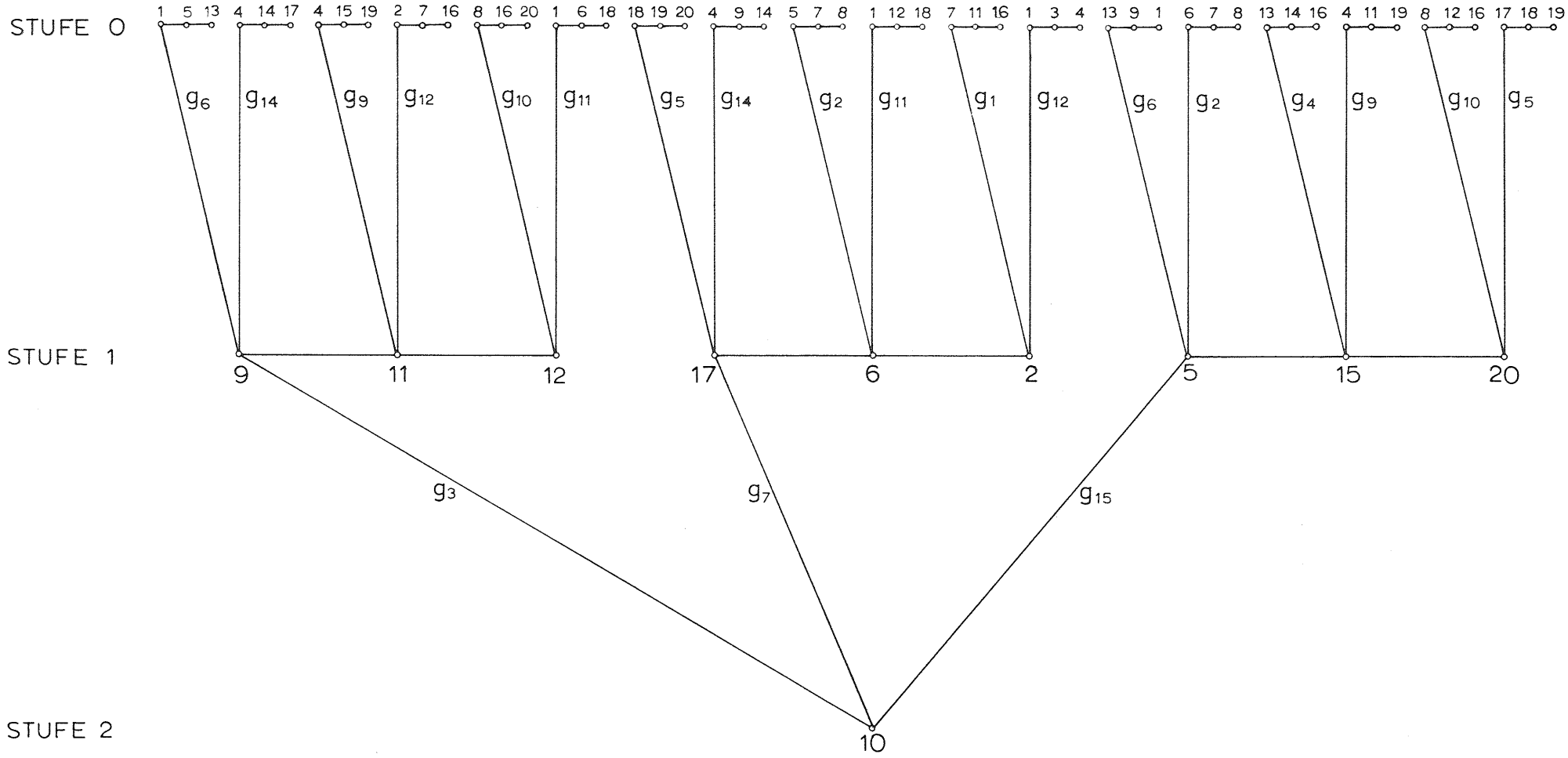
Es wurden falsch übertragen die Stellen 1 und 4. Im ersten Schritt dreht der Decoder zwar diese beiden Stellen um; fälscherlicherweise aber auch Stelle 9. Diese Entscheidung wird im zweiten Schritt aber revidiert. Wir sehen, daß vom Decoder in einem vorbereitenden Schritt vorgenommene Korrekturen durchaus nicht endgültig zu sein brauchen.

3. Darstellung der (n, j, 1)-Code durch einen Codebaum

Den besten Einblick in die Struktur des (n, j, 1)-Code liefert seine Darstellung als Codebaum. In ihm läßt sich der Zusammenhang der Prüfgleichungen und Codewortstellen untereinander leicht überblikken. Für eine jede der n Stellen läßt sich ein Codebaum entwickeln.



Figur 3.2: Codebaum der Stelle 1 für den Blockcode von Figur 3.1



Figur 3.3: Codebaum der Stelle 10 für den Blockcode von Figur 3.1

Wir wollen ihn für die Stellen 1 und 10 aufstellen: den Codebaum für Stelle 1 zeigt Figur 3.2, den Codebaum für Stelle 10 Figur 3.3. Jeder Knoten im Codebaum repräsentiert eine Codewortstelle. Die von diesen Knoten nach oben ausgehenden Linien stellen die Prüfgleichungen dar, welche die (unten stehende) Ausgangsstelle und ferner die Stellen im (sich oben anschliessenden) waagerechten Teil der Linie erfassen. Wir lesen einen Codewortbaum von unten nach oben; zum Beispiel den Codebaum für Stelle 1 in Figur 3.2. Stelle 1 wird kontrolliert durch die Gleichungen g_1, g_6 und g_{11}; deshalb gehen von der Wurzel des Codebaumes j=3 Äste nach oben. Von Prüfgleichung g_1, symbolisiert durch den linken Ast werden außer Stelle 1 erfaßt die (1-1)=3 weiteren Stellen 2, 3 und 4. Das lesen wir ab aus der waagerechten Linie, die sich an den linken Ast anschließt. Analog verfahren wir mit den beiden anderen Prüfgleichungen g_6 und g_{11}. Damit ist eine Stufe des Codebaumes konstruiert. Jetzt wollen wir den Codebaum nach oben fortsetzen. Wir beginnen mit Stelle 2 aus Prüfgleichung g_1;diese wird nicht nur von Prüfgleichung g_1 kontrolliert, sondern auch von den Gleichungen g_7 und g_{12}. Von Stelle 2 müssen also (j-1)=2 Äste nach oben ausgehen. An jeden dieser Äste hat sich eine waagerechte Linie anzuschließen, auf der wiederum diejenigen Stellen markiert sind, die außer Stelle 2 in Prüfgleichungen g_7 bzw. g_{12} auftauchen.

Während von der Wurzel des Codebaumes j Äste ausgehen, gehen von jedem anderen Knoten nur noch (j-1) Äste nach oben aus. Wenn eine begonnene Stufe des Codebaumes vervollständigt ist, können wir ihn weiter nach oben fortsetzen. Wird der Codebaum auf diese Art und Weise auf einige Stufen ausgedehnt, wird die Zahl der Knoten immer größer. Jedem Knoten ist eine Codewortstelle zugeordnet. Es werden deshalb bald Codewortstellen öfter als einmal auftreten. Nehmen wir an, daß - einschließlich der Wurzel - (t+1) Stufen gebildet werden können, ohne daß irgendeine Stelle mehrfach auftritt, dann heißt das: die Gesamtzahl der im Codebaum auftauchenden Knoten ist nicht größer als die Blocklänge n. Wir bezeichnen die oberste Stufe mit 0, die zweitoberste mit 1 usw.; die Stufe oberhalb der Wurzel erhält dann die Nummer (t-1) und die Würzel selbst heißt Stufe t. Nach unseren Feststellungen enthält Stufe t eine einzige Stelle und für $\nu = 0, 1, \ldots, (t-1)$ enthält Stufe ν genau $j \cdot (1-1) \cdot (j-1)^{t-1-\nu}$ Stellen. Damit lautet die Bedingungsgleichung

zwischen n und t:

$$n \geq 1 + j \cdot (1-1) \sum_{\nu=0}^{t-1} (j-1)^{t-1-\nu} \qquad (3)$$

$$n \geq 1 + j \cdot (1-1) \frac{1 - (j-1)^t}{2 - j}$$

Gleichung (3) zeigt, daß bloß sehr wenige Stufen gebildet werden können, wenn jede Stelle nur einmal im Codebaum auftreten darf.

In Stufe (t-1) stehen alle diejenigen Stellen, die direkt mit der Ausgangsstelle zusammenhängen, die also gemeinsam mit dieser Ausgangsstelle in einer Prüfgleichung kontrolliert werden. Nehmen wir nun an, daß Stelle 1 falsch übertragen wurde, alle Stellen in Stufe (t-1) aber korrekt, dann erlaubt - wie wir in Abschnitt 2 gesehen haben - GALLAGER's erstes Decodierverfahren die Korrektur von Stelle 1. Alle j=3 Prüfgleichungen, in denen Stelle 1 auftritt, sind in diesem Fall nicht erfüllt. Treten aber in Stufe (t-1) auch gestörte Stellen auf, kann das zur Folge haben, daß von Stelle 1 ausgehende Prüfgleichungen erfüllt sind, obwohl Stelle 1 falsch übertragen wurde. Es sind dann nicht alle von der Wurzel ausgehenden j Prüfgleichungen nicht erfüllt. Deshalb wird möglicherweise im ersten Schritt noch keine Korrektur der als Wurzel stehenden Stelle 1 erreicht, weil wir eine Stelle nur dann korrigieren, wenn sie in mehr als b nicht erfüllten Prüfgleichungen auftritt. Tritt dieser Fall ein, dann ist es möglich, daß die "vielen" richtigen Stellen in den oberen Stufen des Codebaumes in einem vorbereitenden Korrekturschritt die Fehler der unteren Stufen ausbessern und damit im letzten Schritt die Decodierung der als Wurzel stehenden Stelle erlauben.

4. Das zweite Decodierverfahren von GALLAGER

Das erste Verfahren nutzt die Codebaumstruktur noch nicht systematisch aus. Das erste Verfahren entspricht dem Decodieren der Wurzelstelle eines Codebaumes aufgrund der Stellen in Stufe (t-1). Das erste Verfahren benutzt diesen Übergang mehrmals, ohne die mehrstufigen Abhängigkeiten innerhalb des Codebaumes zu beachten.

Es ist nochmals darauf hinzuweisen, daß das Ziel bisher war, Stelle 1 (aufgrund des Codebaumes 1) zu decodieren. Die Zwischenkorrekturen in Stufe 1 des Codebaumes sind für die Decodierprozedur einer anderen Stelle irrelevant. Vielmehr muß man für die Decodierung dieser Stellen - ausgehend von $\bar{v}$ - die Codebäume dieser Stellen untersuchen.

Schauen wir uns nun an, was im Codebaum für Stelle 10 geschieht, wenn die Stellen 6 und 11 (Beispiel 5) falsch übertragen wurden. Beim ersten Korrekturschritt würden dort Stelle 11 und Stelle 6 in Stufe 1 umgedreht und im zweiten Schritt würde Stelle 10 nicht abgeändert, also richtig decodiert.

Bei der Decodierung des Beispiel 5) wurde im Codebaum 1 Stelle 2 im ersten Schritt umgedreht, dagegen wurde Stelle 2 beim Codebaum 10 im ersten Schritt für richtig befunden. Stelle 2 ist enthalten in den Prüfgleichungen g_1, g_6 und g_{11}. Im Codebaum 1 wurde die Entscheidung über Stelle 2 gefällt aufgrund von Gleichung g_7 und g_{12}; im Codebaum 10 dagegen aufgrund von Gleichung g_1 und g_{12}. Wir sehen, daß die Vorentscheidungen über einzelne Ziffern bei verschiedenen Codebäumen durchaus verschieden sein können. Bei der Untersuchung der verschiedenen Codebäume muß man deshalb jeweils von dem tatsächlich empfangenen n-Tupel $\bar{v}$ ausgehen und darf nicht etwa Vorkorrekturen anderer Codebäume benutzen. Wir sind jetzt imstande, das zweite Decodierverfahren für einen (n, j, 1)-Code zusammenfassend zu formulieren, wenn dieser Code für jede der n Blockstellen einen t-stufigen Codebaum zuläßt. Die eben beschriebene Decodierung der Codebaumwurzeln mußte für jeden Codebaum gesondert ausgeführt werden; dabei treten dieselben Konstellationen in den n Codebäumen öfter auf und müssen jedesmal neu berechnet werden. Das nun folgende Schema erlaubt es, die Codebäume parallel zu durcharbeiten.

1. Korrekturschritt

Er hat das Ziel, die Stellen der Stufe 1 in allen Codebäumen zu korrigieren aufgrund der Stellen von Stufe 0 der jeweiligen Codebäume. Die Stellen in Stufe 0 sind die vom Kanal angelieferten v_ν ($\nu = 1, 2, \ldots, n$). Bei jedem Codebaum gehen von jedem Knoten

der Stufe 1 genau (j-1) Äste - (j-1) Prüfgleichungen repräsentierend - nach oben zu Stufe 0 aus. Die Stellen in Stufe 1 sollen dann umgedreht werden, wenn mehr als b_1 dieser (j-1) nach oben gehenden Prüfgleichungen nicht erfüllt sind. Stelle x_ν wird erfaßt von j Prüfgleichungen; diese seien bezeichnet mit $g_{\nu_1}, g_{\nu_2}, \ldots g_{\nu_j}$. (Natürlich kommen in einem bestimmten - nach Voraussetzung unabhängigen - Codebaum in der zweiten Stufe die Stellen von Stufe 1 nicht vor; da wir aber die Codebäume parallel verarbeiten, müssen trotzdem alle Stellen vorkorrigiert werden). Je nach Codebaum geht eine bestimmte Auswahl von (j-1) dieser j Prüfgleichungen von Stelle ν nach oben zu Stufe 0. Die übrigbleibende Prüfgleichung repräsentiert denjenigen Ast, der im gerade akuten Codebaum von Stelle ν nach unten geht.

Da j solcher verschiedenen Auswahlmöglichkeiten bestehen, gibt es j Korrekturwerte erster Stufe, die aus den obigen j Prüfgleichungen bestimmt werden unter Weglassung je einer dieser Gleichungen. Das Symbol dieser weggelassenen Gleichung wird als Index an die Korrekturwerte angehängt: $v^{(1)}_{\nu,\nu_1}$, $v^{(1)}_{\nu,\nu_2}$, ..., $v^{(1)}_{\nu,\nu_j}$. Es gibt im Gesamten $j \cdot n$ solcher Korrekturwerte $v^{(1)}$. Welcher dieser j Korrekturwerte im zweiten Korrekturschritt benötigt wird, hängt ab vom Codebaum.

2. Korrekturschritt

Die Werte $v^{(1)}_{\nu,\nu_i}$ des ersten Korrekturschrittes sollen nun zur Korrektur der Stellen aus Stufe 2 dienen. Es wird dann eine Stelle von Stufe 2 umgedreht, wenn mehr als b_2 ihrer (j-1) nach oben gehenden Prüfgleichungen nicht erfüllt sind. Von den j Prüfgleichungen führt eine nach unten zu Stufe 3. Dafür gibt es j verschiedene Möglichkeiten und demgemäß für jede Stelle j verschiedene Korrekturwerte zweiter Stufe. Wollen wir in einem beliebigen, aber festen Codebaum die zweiten Korrekturwerte von Stelle μ ermitteln, dann müssen wir jenen Korrekturwert einer Stelle ν der ersten Stufe verwenden, der entstanden ist unter Weglassung jener der j Prüfgleichungen $g_{\nu_1}, g_{\nu_2}, \ldots, g_{\nu_j}$ von Stelle ν, in der Stelle μ auftritt, denn diese Prüfgleichung führt gerade von Stelle μ in der 2. Stufe zu Stelle ν in Stufe 1. Kommt Stelle μ in den j Prüfgleichungen

$g_{\mu_1}, g_{\mu_2}, \ldots, g_{\mu_j}$ vor, dann ergibt der zweite Korrekturschritt die j Werte für Stelle μ: $v^{(2)}_{\mu,\mu_1}, v^{(2)}_{\mu,\mu_2}, \ldots, v^{(2)}_{\mu,\mu_j}$.

t-ter, endgültiger Korrekturschritt

Wir kommen zur endgültigen Decodierung der einzelnen Stellen und dürfen uns dabei auf die Nachrichtenstellen beschränken. Stelle λ sei zu decodieren. Es werden diejenigen Werte $v^{(t-1)}$ der (t-1)-ten Korrektur zur Kontrolle der Prüfgleichungen herangezogen, die unter Weglassung derjenigen Prüfgleichungen entstanden sind, in der Stelle λ vorkommt. Stelle λ wird dann umgedreht, wenn mehr als b_t der - da wir jetzt die Codebaumwurzeln decodieren - j von Stelle λ ausgehenden Prüfgleichungen nicht erfüllt sind. Der decodierte Wert einer Nachrichtenstelle heißt: $v^{(t)}_\lambda = y_\lambda$.

Auf die Wahl der b_i wird in Kapitel VIII eingegangen.

Kehren wir noch einmal zu unserem Beispiel 4) zurück; nehmen wir also an, daß das Nullwort gesendet wurde und daß die Stellen 1, 2 und 5 falsch übertragen wurden. Es sei $b_1 = 1$, $b_2 = 1$. Dann lauten die j=3 ersten Korrekturwerte für Stelle 2 (vgl. Figur 3.1 und Figur 3.2): $v^{(1)}_{2,1} = 0$; $v^{(1)}_{2,7} = 1$; $v^{(1)}_{2,12} = 1$.

Der Korrekturwert $v^{(1)}_{2,1}$ z.B. ergibt sich aus den Prüfgleichungen g_7 und g_{12}. Diese beiden Prüfgleichungen sind wegen der gestörten Stelle 2 nicht erfüllt; die Schranke b_1=1 wird überschritten und daher wird die falsch übertragende Stelle v_2 in Stufe 1 umgedreht. Führt man die Korrektur für Stelle 1 weiter fort, so stellt man fest, daß das Verfahren in diesem Beispiel funktioniert, also $y_1 = x_1 = 0$ decodiert wird.

5. GALLAGER's drittes, endgültiges Decodierverfahren

GALLAGER's drittes Verfahren arbeitet im Prinzip genauso wie sein zweites, nur werden die Stellen nicht in jeder Stufe abgeändert, sondern es wird die Wahrscheinlichkeit p_ν dafür berechnet, daß Stelle ν eine 1 ist. In der 0-ten Stufe gilt:

$$p^{(0)}_\nu = \begin{cases} p & \text{falls } v_\nu = 0 \\ q & \text{falls } v_\nu = 1 \end{cases} \qquad (4)$$

Wurde Stelle ν als 1 empfangen, so ist sie eben mit der Wahrscheinlichkeit 1-p=q auch eine 1.

In der ersten Stufe haben wir wie beim zweiten Verfahren j Wahrscheinlichkeiten $p^{(1)}_{\nu,\nu_1}, p^{(1)}_{\nu,\nu_2}, \ldots, p^{(1)}_{\nu,\nu_j}$ für jede Stelle x_ν (ν = 1, 2, ..., n), die unter Weglassung je einer der j Prüfgleichungen entstanden sind. Zur Vereinheitlichung der Schreibweise wollen wir noch festsetzen, daß $p^{(0)}_{\nu,\nu_i} = p^{(0)}_\nu$ ist für alle ν_i (i = 1, 2, ..., j). Analoges wie in Stufe 1 gilt für die weiteren Stufen. Beim letzten, dem t-ten Korrekturschritt, der zur Decodierung der Stellen führen soll, erhalten wir die Wahrscheinlichkeiten $p^{(t)}_\nu$.

Ist $p^{(t)}_\nu \geq \frac{1}{2}$, dann decodieren wir Stelle ν als 1, wenn $p^{(t)}_\nu < \frac{1}{2}$, dann als 0. Wenn wir erfolgreich decodieren, dann streben die mit einer Stelle ν verbundenen Wahrscheinlichkeiten einem Grenzwert zu, entweder der 1 oder der 0. Die Wahrscheinlichkeiten $p^{(\varphi)}_\nu$ der Stufe φ, also die Wahrscheinlichkeiten, daß Stelle ν eine 1 ist unter Berücksichtigung der Stellen aus den Stufen 0, 1 bis (φ-1), lassen sich berechnen aus den Wahrscheinlichkeiten der Stufe (φ-1).

Ohne auf einen konkreten Vorschlag für eine Decodierschaltung eingehen zu können - sie enthält kleine Rechner, die eine Summation und die Bestimmung von Logarithmen durchführen können - sei noch vermerkt, daß es mittels Rückkopplung der Korrekturwerte der ν-ten Stufe (ν = 1, 2, ..., t-1) möglich ist, mit einem Decodieraufwand auszukommen, der unabhängig ist von der Zahl der Iterationsschritte t. Der Decodieraufwand wächst linear mit n.

Kapitel IV

SEQUENTIELLE CODE

Nach Kapitel I, 5 wollen wir unter sequentiellen Code immer lineare, sequentielle Code verstehen.

1. Beschreibung der sequentiellen Code als lineares, rückkopplungsfreies, sequentielles Netzwerk

Die allgemeine parallelarbeitende Codieranlage wurde als Blockschaltbild mit Figur 1.6 bereits vorgestellt. Von links treten im ν-ten Arbeitstakt die m Nachrichtenstellen $x_\nu^{(1)}, x_\nu^{(2)}, \ldots, x_\nu^{(m)}$ in den Coder gleichzeitig ein und werden dort verarbeitet. Das Blockschaltbild legt es nahe, diese m gleichzeitig einlaufenden Nachrichtenstellen als Spaltenvektor $\bar{x}_\nu$

$$\bar{x}_\nu = \begin{pmatrix} x_\nu^{(1)} \\ \vdots \\ x_\nu^{(m)} \end{pmatrix}$$

aufzufassen. Der Index ν weist auf die Nummer des Arbeitstaktes hin. Im selben Takt verlassen den Coder nach rechts die n Codewortstellen $u_\nu^{(1)}, u_\nu^{(2)}, \ldots, u_\nu^{(n)}$. Wir wollen sie zu einem Spaltenvektor $\bar{u}_\nu$ zusammenfassen:

$$\bar{u}_\nu = \begin{pmatrix} u_\nu^{(1)} \\ \vdots \\ u_\nu^{(n)} \end{pmatrix}$$

Bei sequentieller Codierung ist der Ausgang $\bar{u}_\nu$ des Coder nach Definition nicht nur abhängig von dem gerade anliegenden Nachrichtenwort $\bar{x}_\nu$, sondern auch von den r vorangegangenen Nachrichtenworten $\bar{x}_{\nu-1}, \bar{x}_{\nu-2}, \ldots, \bar{x}_{\nu-r}$. Eine Nachrichtenstelle kann damit (r+1) aufeinanderfolgende n-Tupel von Coderausgängen beeinflußen. Obwohl es keine Grenze gibt, die Codewortstellen ohne gegenseitigen Einfluß allgemein trennen könnte, haben wir die Codelänge $n_0 = (r+1) \cdot n$ in Kapitel I, 5 definiert als die Maximalzahl von Codestellen, die eine einzelne Nachrichtenstelle beeinflußen kann.

Der sequentielle Coder ist ein rückkopplungsfreies, lineares, sequentielles Netzwerk. Die Ausgangs-n-Tupel werden durch die

Eingangs-m-Tupel eindeutig bestimmt; die Gesetze dieser Zuordnung wollen wir jetzt untersuchen.

In Kapitel I, 6 hatten wir die Nachrichtenfolge, die von der Quelle gesendet wird, mit X bezeichnet; sie besteht aus den Binärzahlen $(x_1, x_2, x_3, \ldots)$ oder in der arbeitstaktbezogenen Schreibweise (die auf die Parallel-Verarbeitung im Coder zugeschnitten ist) $(X = (x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(m)}, x_2^{(1)}, \ldots, x_2^{(m)}, x_3^{(1)}, \ldots)$, dabei gilt $x_\nu^{(i)} = x_{(\nu-1)\cdot m+i}$. Die Binärfolge U, die dem Kanal vom Coder nach der Parallel-Serien-Rückwandlung zur Übertragung angeliefert wird, nannten wir Codefolge: $U = (u_1, u_2, u_3, \ldots)$, oder auch $U = (u_1^{(1)}, u_1^{(2)}, \ldots, u_1^{(n)}, u_2^{(1)}, \ldots, u_2^{(n)}, u_3^{(1)}, \ldots)$ mit $u_\nu^{(i)} = u_{(\nu-1)\cdot m+i}$.

Jeder Nachrichtenfolge X wird durch den Coder eine Codefolge U zugeordnet (analog der obigen Zuordnung von Eingangs-m-Tupeln $\bar{x}$ zu Ausgangs-n-Tupeln $\bar{u}$). Um diese Zuordnung zu untersuchen, werden wir uns zunächst mit besonders einfachen Nachrichtenfolgen beschäftigen, und zwar solchen, die nur eine einzige von Null verschiedene Nachrichtenstelle haben: wir wollen von der Basisnachrichtenfolge $X_\nu^{(i)}$ sprechen, wenn $x_\nu^{(1)}$ die einzige von Null verschiedene Nachrichtenstelle der gesamten Nachrichtenfolge ist. Die Indizes deuten darauf hin, daß im ν-ten Arbeitstakt die i-te Nachrichtenstelle 1 ist. Der Basisnachrichtenfolge $X_\nu^{(i)}$ wird durch den Coder die Basiscodefolge $U_\nu^{(i)}$ zugeordnet.

Wir kehren zu dem parallelarbeitenden Coder von Figur 1.6 zurück.

Bei Beginn der Übertragung sei der Coder ganz geleert, was gleichbedeutend damit ist, daß "genügend" lange Nullen eingespeist wurden. Genügend lange heißt mindestens während der r vorangehenden Takte. Um den Einfluß einer einzigen Nachrichtenstelle zu verfolgen, wollen wir annehmen, daß $x_1^{(1)} = 1$ ist, während alle anderen Nachrichtenstellen Null sein sollen, also insbesondere die Stellen $x_1^{(2)}, x_1^{(3)}, \ldots, x_1^{(m)}$ und die gesamten folgenden Eingänge $\bar{x}_2, \bar{x}_3, \ldots, \bar{x}_{r+1}$. Die Quelle sendet also die Basiscodefolge $X_1^{(1)}$. Der Coderausgang im Arbeitstakt 1 sei für die angenommene Nachrichtenfolge:

$$\bar{u}_1 = \begin{pmatrix} b_1^{(1)} \\ \vdots \\ b_n^{(1)} \end{pmatrix}$$

Der obere Index der Koeffizienten weist darauf hin, daß Nachrichtenstelle 1 als einzige von Null verschieden ist. Nachrichtenstelle $x_1^{(1)}$ beeinflußt auch die folgenden r Ausgänge des Coder ; dagegen ist der (r+1)-te darauffolgende Ausgang $\bar{u}_{r+2}$ von der nach Voraussetzung einzigen von Null verschiedenen Nachrichtenstelle $x_1^{(1)}$ unabhängig, weshalb er nur aus Nullen bestehen kann. Die r weitern Coderausgänge, die von Nachrichtenstelle $x_1^{(1)}$ beeinflußt werden, seien analog zu $\bar{u}_1^{(1)}$ bezeichnet mit:

$$\bar{u}_2^{(1)} = \begin{pmatrix} b_{n+1}^{(1)} \\ \vdots \\ b_{2n}^{(1)} \end{pmatrix}, \ \ldots, \ \bar{u}_{r+1}^{(1)} = \begin{pmatrix} b_{r\cdot n+1}^{(1)} \\ \vdots \\ b_{n_0}^{(1)} \end{pmatrix}$$

Diese (r+1) wesentlichen Ausgangsvektoren $\bar{u}_\nu^{(1)}$ wollen wir nun als Zeilenvektoren umschreiben, also

$$\bar{u}_\nu^{(1)} = (b_{(\nu-1)\cdot n+1}^{(1)}, \ \ldots, \ b_{\nu\cdot n}^{(1)})$$

und dann zu einem n_0-Tupel $\bar{b}^{(1)}$ vereinigen:

$$\bar{b}^{(1)} = (b_1^{(1)}, b_2^{(1)}, \ \ldots, \ b_{n_0}^{(1)})$$

Der Index von $\bar{b}^{(1)}$ deutet darauf hin, daß die erste Nachrichtenstelle 1 ist, während alle anderen Nachrichtenstellen Null sind. Das Umordnen der (r+1) Spalten zu einem einzigen n_0-Tupel $\bar{b}^{(1)}$ entspricht der üblicherweise seriellen Übertragung.

Das n_0-Tupel $\bar{b}^{(1)}$ stellt einen Ausschnitt aus der Basiscodefolge $U_1^{(1)}$ dar; die n_0 Stellen von $\bar{b}^{(1)}$ sind gerade die ersten n_0 Stellen von $U_1^{(1)}$. Alle folgenden Stellen der Basiscodefolge $U_1^{(1)}$ sind Null, da sie nicht mehr von Nachrichtenstelle $x_1^{(1)}$ beeinflußt werden.

Als zweiten Fall nehmen wir bei ebenfalls "leerem" Coder an, daß die zweite Nachrichtenstelle im ersten Arbeitstakt $x_1^{(2)} = 1$ ist und alle anderen Nachrichtenstellen Null sind. Die Quelle soll also die Basiscodefolge $X_1^{(2)}$ senden. Wir erhalten entsprechend zu oben als seriell zusammengefaßte Coderausgänge $\bar{u}_\nu^{(2)}$ das n_0-Tupel $\bar{b}^{(2)} = (b_1^{(2)}, b_2^{(2)}, \ \ldots, \ b_{n_0}^{(2)})$. Analog verfahren wir mit den

folgenden Nachrichtenstellen $x_1^{(i)}$, bis wir schließlich für $x_1^{(m)}=1$, alle anderen Nachrichtenstellen Null - also Basisnachricht $X_1^{(m)}$ - das n_0-Tupel $\bar{b}^{(m)}$ erhalten. Alle diese n_0-Tupel $\bar{b}^{(i)}$ sind für die jeweils gesendete Basisnachricht $X_1^{(i)}$ Ausschnitte (die ersten n_0 Stellen) der Basiscodefolge $U_1^{(i)}$.

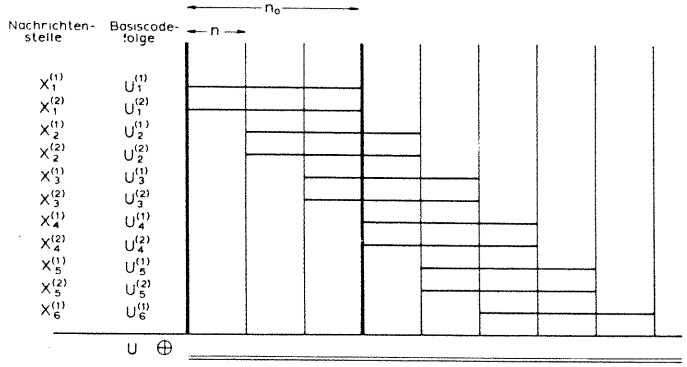
Nun interessiert uns der Fall $x_1^{(1)} = x_1^{(2)} = 1$, d.h. die beiden ersten Eingangsstellen im Arbeitstakt 1 sind gleichzeitig Eins, alle anderen Nachrichtenstellen aber gleich Null. Wegen des linearen Netzwerkes entsprechen jetzt die Ausgangs-n-Tupel $\bar{u}$ des Coder den stellenweisen modulo-2 Summen der obigen Ausgangs-n-Tupel $\bar{u}_\nu^{(1)}$ und $\bar{u}_\nu^{(2)}$; symbolisch schreiben wir dafür $\bar{u}_\nu = \bar{u}_\nu^{(1)} \oplus \bar{u}_\nu^{(2)}$ für alle ν . Die Coderausgänge $\bar{u}_{r+2}$, $\bar{u}_{r+3}$, ... bestehen - genauso wie oben - nur aus Nullen. Das bedeutet: die gesuchte Codefolge U ist gleich der modulo-2 Summe der Basiscodefolgen $U_1^{(1)}$ und $U_1^{(2)}$.

Wir gehen über zu den Basisnachrichtenfolgen $X_\nu^{(i)}$ mit $\nu > 1$, z.B. zu $X_4^{(2)}$. Demgemäß bestehen die Nachrichtenworte $\bar{x}_1$, $\bar{x}_2$ und $\bar{x}_3$ aus lauter Nullen. Im Arbeitstakt 4 ist nur die zweite Nachrichtenstelle von Null verschieden; die Nachrichtenworte $\bar{x}_5$, $\bar{x}_6$, ... bestehen wieder nur aus Nullen. Es liegen jetzt 3 Arbeitstake später genau dieselben Verhältnisse vor wie im Fall der Basisnachrichtenfolge $X_1^{(2)}$:während der ersten drei Takte bestehen die Coderausgänge aus lauter Nullen, da dem Coder in den ersten 3 Takten nur Nullen eingespeist werden. Der Coderausgang $\bar{u}_4$ dagegen stimmt überein mit dem Coderausgang $\bar{u}_1^{(2)}$ bei der Basisnachrichtenfolge $X_1^{(2)}$. Es treten 3 Takte später dieselben Coderausgänge wie bei Basisnachrichtenfolge $X_1^{(2)}$ auf: $\bar{u}_{3+j} = \bar{u}_j^{(2)}$ für $j = 1, 2, 3, \ldots$ Die Basiscodefolge $U_4^{(2)}$ entsteht also aus der Basiscodefolge $U_1^{(2)}$, indem man $U_1^{(2)}$ um $3\cdot n$ Stellen nach rechts verschiebt. Die ersten $3\cdot n$ Stellen von $U_4^{(2)}$ sind Null.

Allgemein erhalten wir die Basiscodefolge $U_\nu^{(1)}$ aus der Basiscodefolge $U_1^{(i)}$, indem wir die Binärzahlen von $U_1^{(i)}$ um $(\nu-1)\cdot n$ Stellen nach rechts verschieben. Die ersten $(\nu-1)\cdot n$ Stellen von $U_\nu^{(i)}$ sind Null. Damit haben wir gefunden, daß die Basiscodefolgen bestimmt sind durch die m n_0-Tupel $\bar{b}^{(1)}$, $\bar{b}^{(2)}$, ..., $\bar{b}^{(m)}$. Wir nennen deshalb diese n_0-Tupel die Generatormuster des sequentiellen Code. Mit diesen Ergebnissen lassen sich die sequentiellen Code als unendliche Gruppencode darstellen.

2. Beschreibung der sequentiellen Code als unendliche Gruppencode mit Hilfe ihrer Generatormuster

An die Stelle der Basiscodeworte bei endlichen Gruppencode treten jetzt die unendlich vielen Basiscodefolgen $U_\nu^{(i)}$, entsprechend den Nachrichtenstellen $x_\nu^{(i)}$. Unter den unendlich vielen Basiscodefolgen sind aber nur die ersten m wesentlich verschieden; alle anderen ergeben sich einfach durch Verschiebung der ersten m Basiscodefolgen $U_1^{(i)}$ um ganze Vielfache von n nach rechts, wobei die frei gewordenen Anfangsstellen durch Nullen zu ersetzen sind. Das Codierprinzip ist analog zum endlichen Gruppencode: wird dem Coder eine beliebige Nachrichtenfolge X von der Quelle angeliefert, so ergibt sich die zugehörige Codefolge U als stellenweise modulo-2 Summe derjenigen Basiscodefolgen $U_\nu^{(i)}$, deren zugeordnete Nachrichtenstellen $x_\nu^{(i)}$ den Wert 1 haben. Symbolisch kann dieser unendliche Gruppencode wie in Figur 4.1 dargestellt werden.



Figur 4.1: Symbolische Darstellung eines unendlichen Gruppencode für m=2 und r=2

Der waagerechte Strich der Länge n_0 zeigt dabei denjenigen Bereich der eigentlich unendlich langen Basiscodefolgen an, der allein interessant ist, da bloß dort von Null verschiedene Binärzahlen auftreten.

Aus Figur 4.1 sehen wir, daß die ersten n Stellen der Codefolge den Coder sofort nach Verarbeitung der ersten m Nachrichtenstellen verlassen können, denn sie werden durch nachfolgende modulo-2 Additionen späterer Basiscodefolgen nicht mehr beeinflußt. Ebenso kann nach Verarbeitung der folgenden m Nachrichtenstellen die nächste Gruppe von n Stellen den Coder verlassen usw.

Das entspricht der Darstellung in Abschnitt 1. Der gesamte Codierprozeß ist bestimmt durch die m Generatormuster $b^{(1)}, b^{(2)}, \ldots, b^{(m)}$, also durch $m \cdot n_0$ Binärzahlen. Eine wichtige Unabhängigkeitsforderung sollten die Generatormuster auf jeden Fall erfüllen: es muß garantiert sein, daß man aus den Codefolgen eindeutig auf die Nachrichtenfolgen rückschließen kann.
Die bei der Übertragung ersten n_0 Stellen, also die ersten n_0 Stellen der Codefolge, seien als Anfangscodewort $\bar{u}$ bezeichnet:
$\bar{u} = (u_1^{(1)}, \ldots, u_1^{(n)}, \ldots, u_{r+1}^{(1)}, \ldots, u_{r+1}^{(n)})$. Im folgenden Beispiel 1a) wollen wir den sequentiellen Codierprozeß für die Codegrößen m=2, k=3, r=1 und damit n_0=10 verdeutlichen. Wir haben m=2 Generatormuster zu wählen:

$$b^{(1)} = (1010100011)$$
$$b^{(2)} = (1111100111)$$

Die schematische Darstellung gemäß Figur 4.1 deutet das folgende Zahlenschema an:

$U_1^{(1)}$	1010100011	$x_1^{(1)} = 0$
$U_1^{(2)}$	1111100111	$x_1^{(2)} = 1$
$U_2^{(1)}$	1010100011	$x_2^{(1)} = 1$
$U_2^{(2)}$	1111100111	$x_2^{(2)} = 1$
$U_3^{(1)}$	1010100011	$x_3^{(1)} = 0$
$U_3^{(2)}$	1111100111	$x_3^{(2)} = 1$
.....		
U	111110110111011.....	

Beginnt die Nachrichtenfolge mit $x_1^{(1)} = 0$, $x_1^{(2)} = 1$, $x_2^{(1)} = 1$ und $x_2^{(2)} = 1$, dann ergibt sich das Anfangscodewort $\bar{u}$ aus der modulo-2 Summe der Basiscodefolgen $U_1^{(2)}$, $U_2^{(1)}$ und $U_2^{(2)}$. Es lautet:
$\bar{u}$ = (1111101101).

3. Das sequentielle Decodierprinzip

Wenden wir uns kurz dem Decoder zu. Im Arbeitstakt 1 liegen die ersten n Stellen $v_1^{(1)}, \ldots, v_1^{(n)}$ vor. Bis der Decoder aber eine Entscheidung über diese Stellen treffen kann, muß er weitere r Takte abwarten, um alle Stellen zur Verfügung zu haben, die mit

$v_1^{(1)}, \ldots, v_1^{(n)}$ zusammenhängen; ihm liegen dann n_0 Stellen vor. Auf diese n_0 Stellen kann sich der Decoder bei der Korrektur der ersten n Stellen auf jeden Fall beschränken. Die danach ankommenden Stellen haben mit den Anfangsstellen $v_1^{(1)}, \ldots, v_1^{(n)}$ keinen Zusammenhang mehr. Die ersten n_0 empfangenen Stellen wollen wir empfangenes Anfangswort $\bar{v}$ nennen; wegen der Kanalstörungen braucht das empfangene Anfangswort nicht mit dem Anfangscodewort übereinstimmen.

Um nun aus diesem Anfangswort nach dem r-ten Takt eine Aussage über die Stellen $u_1^{(1)}, \ldots, u_1^{(n)}$ machen zu können, müssen wir die Gesamtheit der Anfangscodeworte untersuchen. Den Einfluß der Nachrichtenstellen auf die n_0 Stellen des Anfangscodewortes ersehen wir aus Figur 4.1. Er umfaßt gerade den durch die dickausgezogene Linie umrahmten Teil. Die Enden der Basiscodefolgen, die über diesen Rahmen hinausragen, sind für das Anfangscodewort uninteressant. Wir können uns also das Anfangscodewort entstanden denken aus einem endlichen Gruppencode mit folgender Matrix G:

$$\overleftarrow{\qquad n_0=(r+1)\cdot n \qquad}$$

$$G = \begin{pmatrix} C_0 & C_1 & C_2 & \ldots & C_r \\ 0 & C_0 & C_1 & \ldots & C_{r-1} \\ 0 & 0 & C_0 & \ldots & C_{r-2} \\ 0 & 0 & 0 & \ldots & C_0 \end{pmatrix} \Bigg\uparrow \Bigg\downarrow m_0 = (r+1)\cdot m$$

Dabei stellen die C_ν und 0 Matrixteile von je m Zeilen und n Spalten dar. Die Elemente 0 bestehen aus lauter Nullen, die Elemente C haben den folgenden Aufbau:

$$C_\nu = \begin{pmatrix} b^{(1)}_{\nu\cdot n+1} & b^{(1)}_{\nu\cdot n+2} & \ldots & b^{(1)}_{(\nu+1)\cdot n} \\ b^{(2)}_{\nu\cdot n+1} & b^{(2)}_{\nu\cdot n+2} & \ldots & b^{(2)}_{(\nu+1)\cdot n} \\ \vdots & \vdots & & \vdots \\ b^{(m)}_{\nu\cdot n+1} & b^{(m)}_{\nu\cdot n+1} & \ldots & b^{(m)}_{(\nu+1)\cdot n} \end{pmatrix}$$

Die ersten m Zeilen von G sind gerade die Basiscodeworte $\bar{b}^{(1)}, \ldots, \bar{b}^{(m)}$. Das Anfangscodewort $\bar{u} = (u_1^{(1)}, \ldots, u_{r+1}^{(n)})$ ergibt sich als Überlagerung ausgewählter Zeilen der Matrix G oder formal aus einer Matrixmultiplikation: $\bar{u} = (\bar{x}\cdot G)_{\text{mod-2}}$, wobei wir uns die Nachricht $\bar{x}$ als Zeilenvektor $(x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(m_0)})$ zu denken haben.

Die Anfangscodewort-Matrix für das oben angeführte Beispiel hat die folgende Gestalt:

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

Nehmen wir an, daß ein Verfahren zum Decodieren der ersten n Stellen des Anfangswortes zur Verfügung steht.

Hat der Decoder nach diesem Verfahren die ersten n Stellen der Codefolge decodiert, so läßt sich der Decodierprozeß mit Hilfe des folgenden Tricks fortsetzen:

Aus Figur 4.1 sehen wir, daß der Einfluß der ersten m Nachrichtenstellen $x_1^{(1)}, \ldots, x_1^{(m)}$ in der Codefolge dadurch eliminiert werden kann, daß wir diejenigen Basiscodefolgen $U_1^{(\nu)}$, deren zugehörige Nachrichtenstellen $x_1^{(\nu)}$ eine Eins war, modulo-2 von der Codefolge abziehen. Da in modulo-2 Rechnung Addition und Subtraktion identisch sind, hat man die entsprechenden Basiscodefolgen einfach zur Codefolge zu addieren: Der Einfluß der ersten m Nachrichtenstellen ist rückgängig gemacht, d.h. die Codefolge sieht aus, als ob die ersten m Nachrichtenstellen alle Null gewesen seien.

Um im Arbeitstakt (r+2) die zweite Gruppe von n Stellen - nämlich die Stellen $v_2^{(1)}, v_2^{(2)}, \ldots, v_2^{(n)}$ - zu decodieren, geht der Decoder deshalb folgendermaßen vor: er ersetzt zunächst die ersten n Stellen der empfangenen Folge V durch ihre korrigierten Werte $v_1^{(1)*}, v_1^{(2)*}, \ldots, v_1^{(n)*}$ und kommt so zur Folge ${}^{(1)}V^*$. Dieser sind die mutmaßlichen Nachrichtenstellen $y_1^{(1)}, \ldots, y_1^{(m)}$ nach unserer Voraussetzung über die Unabhängigkeit der Generatormuster eindeutig zugeordnet. Der korrigierten Binärfolge ${}^{(1)}V^*$ addiert der Decoder nun diejenigen Basiscodefolgen $U_1^{(i)}$ hinzu, deren zugeordnete Nachrichtenstelle $y_1^{(i)}$ eine Eins ist. Das ergibt die Binärfolge ${}^{(1)}V$. Die ersten n Stellen der Folge ${}^{(1)}V$ sind automatisch Null, was bedeutet, daß dieselbe Situation vorliegt wie zu Beginn des Decodierprozesses.

Nehmen wir nun an, daß der Decoder die richtige Nachricht gefunden hat, daß also $y_1^{(i)} = x_1^{(i)}$ für i=1, ..., m ist, dann ist der Einfluß der ersten Nachrichtenstellen wirklich aufgehoben und wir können genau dasselbe Verfahren, das wir auf das Anfangscodewort angewandt haben, auf die n_0 Stellen ${}^{(1)}v_2^{(1)}, \ldots, {}^{(1)}v_{r+2}^{(n)}$ anwenden und aus ihnen den zweiten Satz von Nachrichtenstellen bestimmen; das ergibt die m Werte: $y_2^{(1)}, y_2^{(2)}, \ldots, y_2^{(m)}$.

Unterlief dem Decoder aber ein Fehler, z.B. $y_1^{(i)} \neq x_1^{(i)}$, dann geht er im nächsten Arbeitstakt natürlich auch von der Voraussetzung aus, daß der Einfluß der vorhergehenden Stellen ausgelöscht worden ist. In Wirklichkeit steckt jetzt ein Teil der Basiscodeworte $U_1^{(1)}, U_1^{(2)}, \ldots, U_1^{(m)}$ aber noch in der vorliegenden Folge ${}^{(1)}V$ drin. Deshalb neigt jedes sequentielle Decodierverfahren dazu, Decodierfehler fortzupflanzen. Die Fehlerfortpflanzung kann durch Resynchronisation eingedämmt werden: in die zu übertragende Nachrichtenfolge wird in vorbestimmten Abständen immer eine Serie von n_0 Nullen eingeschaltet. Damit ist garantiert, daß der Decoder jeweils nach diesen eingeschalteten Nullen wieder fehlerfrei arbeiten kann. Resynchronisation kommt praktisch einer Erniedrigung der Nachrichtenrate gleich.

4. Systematische sequentielle Code. Erste Codierschaltung

Wir haben gesehen, daß der gesamte sequentielle Code bestimmt ist durch die Wahl seiner m Generatormuster. Diese Generatormuster legen die Gruppencode-Matrix des Anfangscodewortes fest und umgekehrt ist ein sequentieller Code bestimmt durch seine Anfangsmatrix G. Aus dieser Matrix können wir ablesen, wie die Generatormuster eines systematischen sequentiellen Code gebaut sein müssen, also eines sequentiellen Code, der die Nachrichtenstellen unverändert überträgt: $u_\nu^{(i)} = x_\nu^{(i)}$ für i=1, ..., m und alle Arbeitstakte ν. Die Nicht-Nachrichtenstellen wollen wir Kontrollstellen nennen.
Ist Stelle i Nachrichtenstelle, dann darf die i-te Spalte von G nur eine einzige 1 enthalten, und zwar an der i-ten Stelle. Folglich hat die Matrix eines systematischen, sequentiellen Code den folgenden Aufbau:

$$\longleftarrow n_0 = (r+1)\cdot n \longrightarrow$$

$$G = \begin{pmatrix} E & B_0 & N & B_1 & N & B_2 & N & \ldots & B_r \\ N & 0 & E & B_0 & N & B_1 & N & \ldots & B_{r-1} \\ N & 0 & N & 0 & E & B_0 & N & \ldots & B_{r-2} \\ \vdots & & & & & & & & \\ N & 0 & N & 0 & N & 0 & N & \ldots & B_0 \end{pmatrix} \Bigg\updownarrow m_0 = (r+1)\cdot m$$

Es bedeutet E die m-zeilige Einheitsmatrix, N die m-zeilige, m-spaltige Nullmatrix, 0 die m-zeilige, k-spaltige Nullmatrix und die B_i sind ebenfalls Matrizenteile mit m Zeilen und k Spalten der folgenden Gestalt:

$$\longleftarrow k \longrightarrow$$

$$B_0 = \begin{pmatrix} b_{m+1}^{(1)} & b_{m+2}^{(1)} & \ldots & b_n^{(1)} \\ b_{m+1}^{(2)} & b_{m+2}^{(2)} & \ldots & b_n^{(2)} \\ \vdots & & & \\ b_{m+1}^{(m)} & b_{m+2}^{(m)} & \ldots & b_n^{(m)} \end{pmatrix} \Bigg\updownarrow m$$

und für beliebiges ν:

$$B_\nu = \begin{pmatrix} b_{\nu\cdot n+m+1}^{(1)} & b_{\nu\cdot n+m+2}^{(1)} & \cdots & b_{(\nu+1)\cdot n}^{(1)} \\ b_{\nu\cdot n+m+1}^{(2)} & b_{\nu\cdot n+m+2}^{(2)} & \cdots & b_{(\nu+1)\cdot n}^{(2)} \\ \vdots & & & \\ b_{\nu\cdot n+m+1}^{(m)} & b_{\nu\cdot n+m+2}^{(m)} & \cdots & b_{(\nu+1)\cdot n}^{(m)} \end{pmatrix}$$

Die Generatormuster eines systematischen Code sind also bestimmt durch $(r+1)\cdot k\cdot m$ Binärzahlen. Sind die Zeilen der allgemeinen Matrix voneinander unabhängig - die oben geforderte Unabhängigkeitsbedingung impliziert dies - dann läßt sich die allgemeine Matrix durch Zeilenumformung auf systematische Form transformieren.
Durch diese Zeilentransformation werden einfach aus der Gesamtheit der Anfangscodeworte andere als die ursprünglichen Basiscodeworte ausgewählt. Die folgende Matrix von Beispiel 1b) zeigt die systematische Form, die man aus Beispiel 1a) entwickeln kann.

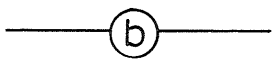
$$G_0 = \begin{pmatrix} 1\,0\,1\,0\,1\,0\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,0\,1\,0 \end{pmatrix}$$

Sie entstand dadurch, daß Zeile 1 zu Zeile 2 und Zeile 3 zu Zeile 4 modulo-2 addiert wurde. Die neuen Generatormuster lauten: $b^{(1)}$ = (1010100011) und $b^{(2)}$ = (0101000100).

Die ersten m Zeilen der Anfangscodewort-Matrix eines systematischen Code sind wieder die - jetzt systematischen - Generatormuster. Für das ν -te Generatormuster $b^{(\nu)}$ eines systematischen Code gilt also:

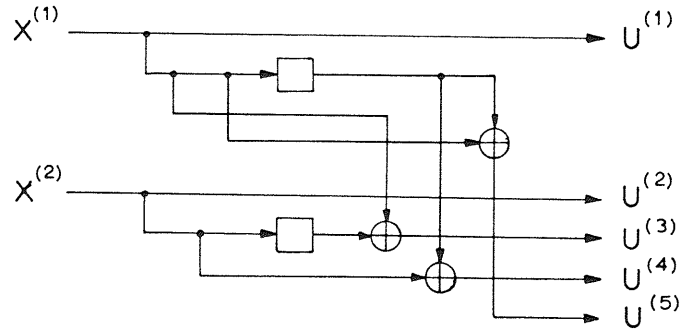
$$b_{\nu}^{(\nu)} = 1$$
$$b_{i}^{(\nu)} = 0 \qquad i=1,\ldots,\nu-1,\nu+1,\ldots m$$
$$b_{j\cdot n+i}^{(\nu)} = 0 \qquad \text{für } i=1,2,\ldots,m \text{ und } j=1,2,\ldots,r$$

Wir kommen jetzt zur Codierschaltung für systematische, sequentielle Code. Dazu sei in Figur 4.2 ein Schaltsymbol vorgestellt, das kennzeichnen soll, ob eine Verbindung durchgeschaltet (b=1) oder unterbrochen (b=0) ist. Mit Hilfe dieses Symbols können wir jetzt die Codierschaltung beschreiben.



Figur 4.2:
Schaltsymbol
b=1 durchgeschaltet
b=0 unterbrochen

Figur 4.3 zeigt das Schaltbild für Beispiel 1b) und Figur 4.4 bringt den allgemeinen Fall.



Figur 4.3: Erste Coderschaltung für einen sequentiellen Code mit m=2, k=3 und r=1 nach Beispiel 1b).

Die Arbeitsweise der Schaltung läßt sich verfolgen, indem - wie in Abschnitt IV, 1 - an einem der m Eingänge eine einzige 1 angelegt wird, während alle anderen Eingänge - auch der folgenden Takte - 0 erhalten. Diese 1 wandert mit jedem Arbeitstakt des Coder in dem Schieberegister des betrachteten Eingangs um eine Stelle weiter nach rechts. Die einzelnen Register bestehen aus r Stellen, so daß der Einfluß über (r+1) Arbeitstakte sich erstrecken kann. Bei einer einzigen 1 an einem der Eingänge erscheinen an den Ausgängen gerade die gewünschten Generatormuster.

5. Darstellung der systematischen, sequentiellen Code durch Prüfmuster. Zweite Codierschaltung

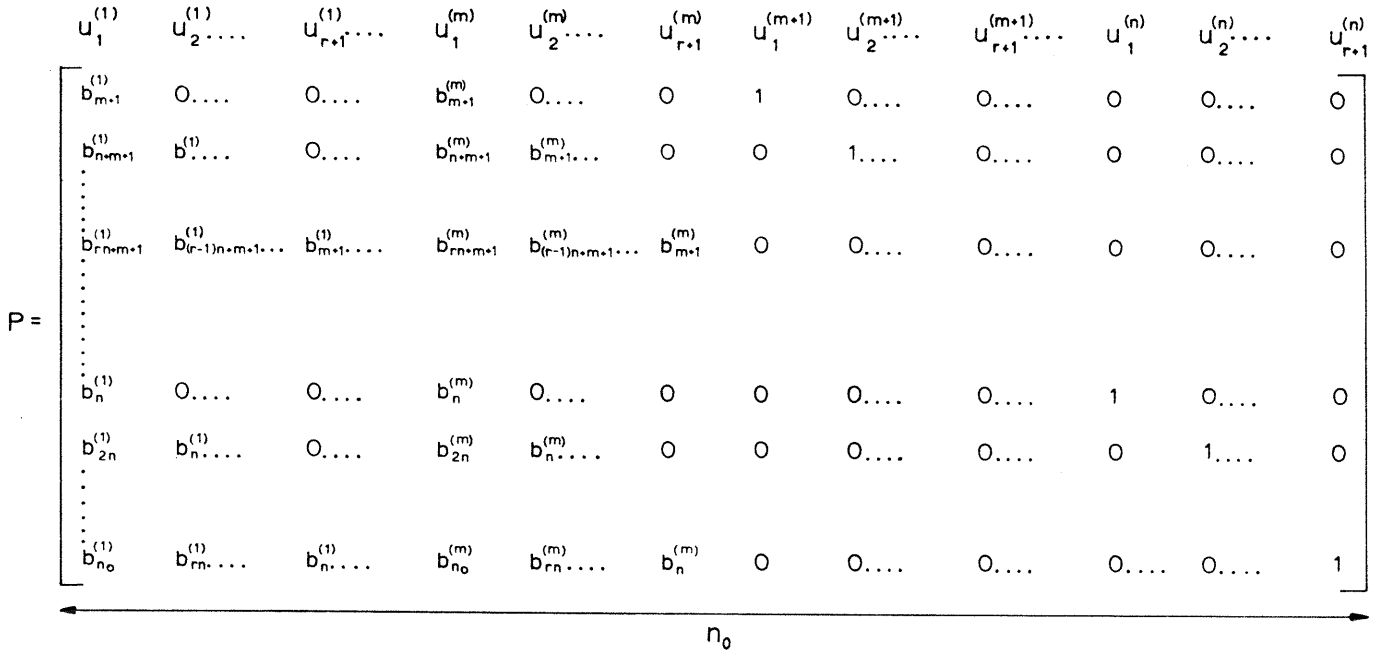
Die Gestalt der Matrix, die im folgenden hergeleitet werden soll, ist speziell zugeschnitten auf die SW-Decodierung nach MASSEY und entpsricht der Prüfschema-Darstellung bei Blockcode. Wir ordnen zunächst die Codewortstellen um. Der obere Index i der Codewortstelle $u^{(i)}$ zeigt an, auf welcher der n Ausgangsleitungen des Codieres die Stelle $u^{(i)}$ übermittelt wird. Wir wollen alle diejenigen Stellen zusammenfassen, die auf derselben Leitung laufen.

Analog zu den Ausführungen über Blockcodes im Kapitel II können wir jetzt aus der umgeordneten Matrix - nach Zeilenvertauschungen, um in den ersten m Spalten die Einheitsmatrix zu haben - das Prüfschema für die k_0 Kontrollstellen $u_1^{(m+1)}, \ldots, u_{r+1}^{(m+1)}, \ldots, u_1^{(n)}, \ldots, u_{r+1}^{(n)}$ ablesen, wie es Figur 4.5 zeigt.

Über den Spalten des Prüfschemas stehen die zugehörigen Nachrichten- bzw. Kontrollstellen.

Die Struktur des Prüfschemas legt eine Umbenennung nahe: jede der sich im Prüfschema automatisch ergebenden k·m Kästchen ist bestimmt durch (r+1) Binärzahlen. Diese (r+1) Binärzahlen wollen wir zusammenfassen und zu einem Prüfmuster vereinigen. Zum Prüfmuster $G^{(r)}$ gehören die (r+1) Binärzahlen: $(g^{(r)}_{,1}, g^{(r)}_{,2}, \ldots, g^{(r)}_{,r+1})$. Das Prüfschema hat mit dieser Bezeichnung die folgende Bauart:

$$P = \begin{pmatrix} G_1^{(m+1)} & G_2^{(m+1)} & \ldots & G_m^{(m+1)} \\ \vdots & & & \\ G_1^{(n)} & G_2^{(n)} & \ldots & G_m^{(n)} \end{pmatrix}$$

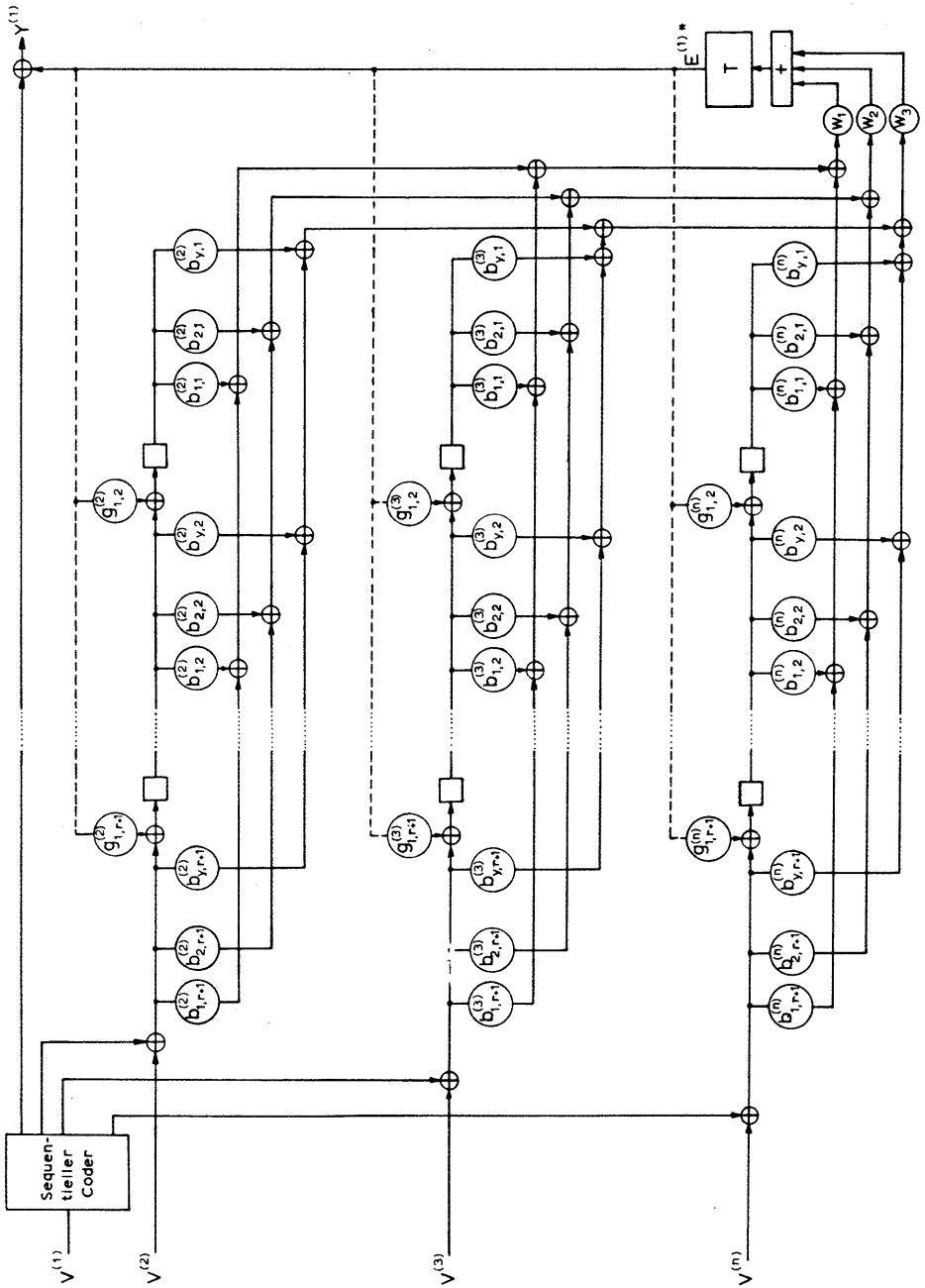


Jeder Teil $G_\nu^{(\mu)}$ des Prüfschemas hat folgenden Aufbau:

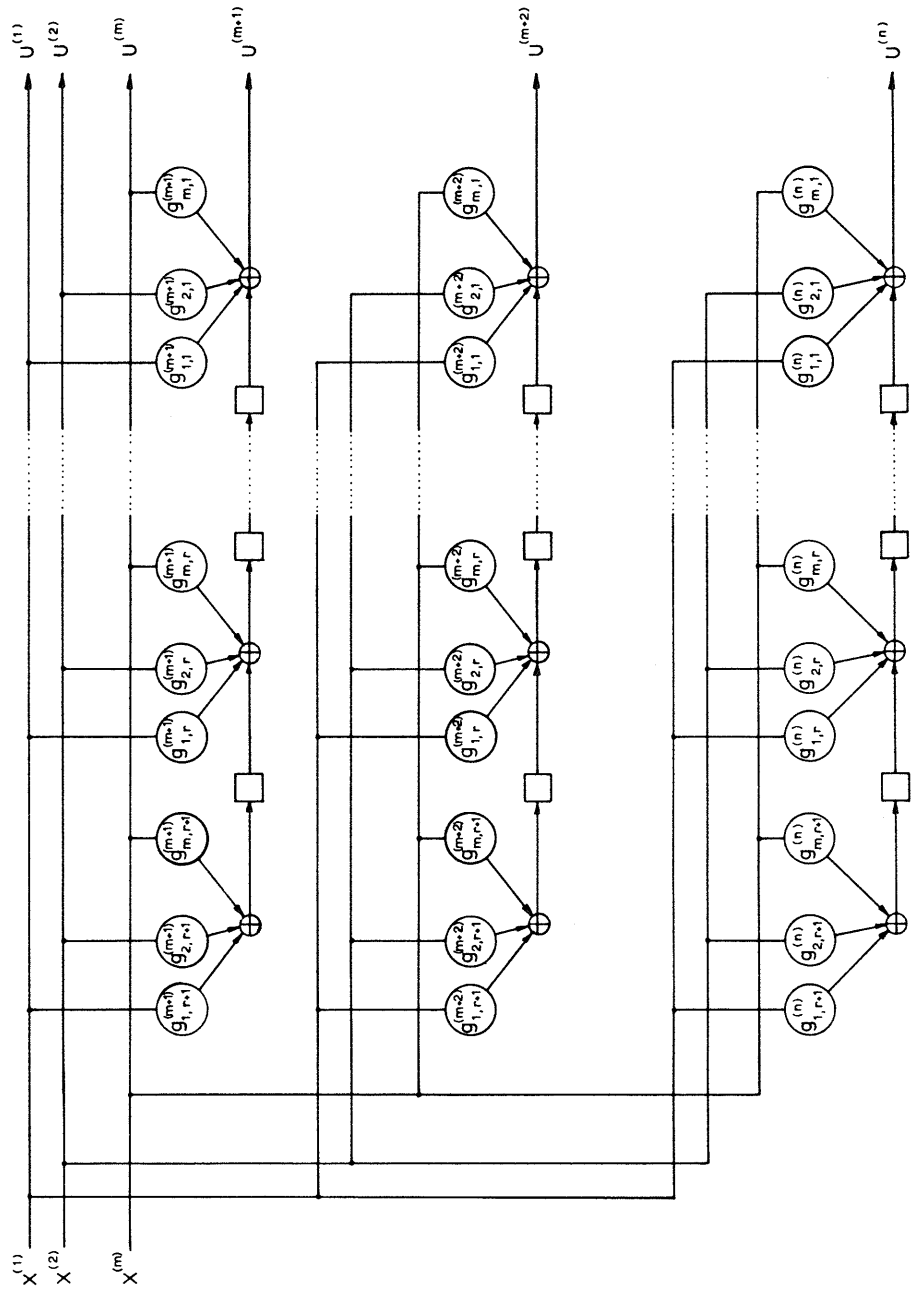
$$G_\nu^{(\mu)} = \begin{pmatrix} g_{\nu,1}^{(\mu)} & 0 & 0 \ldots 0 \\ g_{\nu,2}^{(\mu)} & g_{\nu,1}^{(\mu)} & 0 \ldots 0 \\ g_{\nu,3}^{(\mu)} & g_{\nu,2}^{(\mu)} & g_{\nu,1}^{(\mu)} \ldots 0 \\ \vdots & & \\ g_{\nu,r+1}^{(\mu)} & g_{\nu,r}^{(\mu)} & g_{\nu,r-1}^{(\mu)} \ldots g_{\nu,1}^{(\mu)} \end{pmatrix}$$

Es gilt die Beziehung: $b_{k,n+j}^{(i)} = g_{i,k+1}^{(j)}$

Figur 4.5: Prüfschema für sequentielle Code



Figur 4.4: Erste Codierschaltung für sequentielle Code



Figur 4.6: Zweite Codierschaltung für sequentielle Code

Die Gestalt der Prüfgleichungen für die $(m+\nu)$-te Coderausgangsleitung wird bestimmt durch die Prüfmuster $G_1^{(m+\nu)}$ bis $G_m^{(m+\nu)}$. Darauf bezeiht sich der obere Index. Dagegen kennzeichnet der untere, welche Nachrichtenstellen kontrolliert werden: Prüfmuster $G_i^{(\nu)}$ prüft nur Nachrichtenstellen der i-ten Eingangsleitung. Für die gewonnene Form des Prüfschemas hat MASSEY eine zweite Codierschaltung angegeben, die Figur 4.6 zeigt.

Diese Codierschaltung enthält $r \cdot k = (1-R) \cdot (n_0 - n)$ Speicherelemente, d.h. sie ist für große Informationsraten R günstiger als die oben beschriebene erste Codierschaltung.

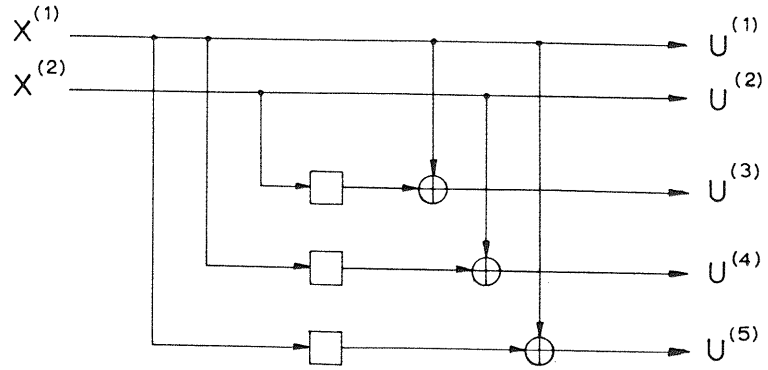
Unser Beispiel 4b) geht zunächst über in die Codematrizen G_1 und G_2; aus G_2 läßt sich dann direkt das gesuchte Prüfschema ablesen. G_1 erhält man durch Spaltenumordnung aus G_0, G_2 erhält man durch Zeilenvertauschungen aus G_1. Die Spalten m_0+1 bis n_0 - also die Spalten 5 bis 10 - der Matrix G_2 ergeben die Zeilen 1 bis k_0 - hier also die Zeilen 1 bis 6 - des Prüfschemas; diese k_0 Zeilen werden durch die k_0-zeilige Einheitsmatrix ergänzt:

$$G_1 = \begin{pmatrix} 1&0&0&0&1&0&0&1&1&1 \\ 0&0&1&0&0&1&1&0&0&0 \\ 0&1&0&0&0&1&0&0&0&1 \\ 0&0&0&1&0&0&0&1&0&0 \end{pmatrix} \qquad G_2 = \begin{pmatrix} 1&0&0&0&1&0&0&1&1&1 \\ 0&1&0&0&0&1&0&0&0&1 \\ 0&0&1&0&0&1&1&0&0&0 \\ 0&0&0&1&0&0&0&1&0&0 \end{pmatrix}$$

$$\overbrace{\qquad\qquad}^{n_0 = 10}$$

$$P = \begin{pmatrix} 1&0&0&0&1&0&0&0&0&0 \\ 0&1&1&0&0&1&0&0&0&0 \\ 0&0&1&0&0&0&1&0&0&0 \\ 1&0&0&1&0&0&0&1&0&0 \\ 1&0&0&0&0&0&0&0&1&0 \\ 1&1&0&0&0&0&0&0&0&1 \end{pmatrix} \quad \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \end{matrix}\right\} k_0 = 6$$

Wir haben damit unser Beispiel 4c) gewonnen; Figur 4.7 zeigt die zugehörige Codierschaltung.



Figur 4.7: Zweite Codierschaltung für einen sequentiellen Code mit m=2, k=3, r=1 und damit n_0=10 nach Beispiel 4c)

In der ersten Coderschaltung (Figur 4.3) werden die $m \cdot r$ Nachrichtenstellen der vorhergegangenen r Arbeitstakte gespeichert; in der zweiten Codierschaltung (Figur 4.6) dagegen die $r \cdot k$ Kontrollstellen der vorhergegangenen Arbeitstakte.

Bis jetzt wurde noch nicht über Verfahren berichtet, die es erlauben, aufgrund einer bestimmten Wahl der Generatormuster aus dem Anfangscodewort die ersten m Nachrichtenstellen zu decodieren. Auf zwei solcher Verfahren werden die beiden folgenden Kapitel eingehen.

Kapitel V

WR - DECODIERUNG

Die Abkürzung "WR-Decodierung" soll erinnern an WOZENCRAFT und REIFFEN, die dieses Decodierverfahren für sequentielle Code entwickelt haben. In den Abschnitten 1 bis 4 dieses Kapitels wird das Decodierverfahren zwar für Blockcode eingeführt und erläutert, aber nur, um es in Abschnitt 5 leicht auf sequentielle Code übertragen zu können, bei welchen es erst voll wirksam werden kann.

1. Das Prinzip der WR-Decodierung

In Kapitel II, 2 wurde gesagt, daß bei MRW-Decodierung, also Decodierung nach maximaler Rückschlußwahrscheinlichkeit, der Decodierprozeß im Prinzip folgendermaßen vor sich geht: um aus dem Empfangswort, das möglicherweise Fehler enthält, das gesendete Codewort zu bestimmen, vergleicht der Empfänger dieses n-Tupel mit der Gesamtheit der möglichen Codeworte und entscheidet sich für dasjenige, das die geringste Distanz vom empfangenen n-Tupel hat. Der Decoder wählt das wahrscheinlichste Codewort $\bar{u}_i$, also dasjenige, das die maximale Rückschlußwahrscheinlichkeit $P(\bar{u}_i|\bar{v})$ aufweist. Der Aufwand, der nötig ist, um diese Entscheidung herbeiführen zu können, ist sehr groß, da die Zahl der Codeworte exponentiell mit der Blocklänge n wächst und man sich oft für ein großes n wird entscheiden müssen, wenn vom Code geringe Restfehlerwahrscheinlichkeit gefordert wird. Beim Vergleichen des empfangenen n-Tupels $\bar{v}$ mit der Codewortliste ist sofort klar, daß eine Vielzahl von Codeworten $\bar{u}_i$ sich von $\bar{v}$ sehr wesentlich unterscheiden wird, da diese Worte von $\bar{v}$ eine sehr große Distanz haben. Die Wahrscheinlichkeit $P(\bar{u}_i|\bar{v})$ ist für eine große Anzahl dieser Codeworte so klein, daß man diese aus der Konkurrenz um das richtige Codewort ausscheiden kann. Bloß eine kleine Untergruppe aller Codeworte wird mit $\bar{v}$ in vielen Positionen übereinstimmen. Da es sehr viel Aufwand kostet, das wahrscheinlichste Codewort zu finden, wird man sich in einem ersten Schritt darauf beschränken, eine Untergruppe von Codeworten zu bestimmen, die sich von $\bar{v}$ in nur relativ wenigen Stellen unterscheiden. Um dahin zu gelangen, wird man die unwahrscheinlichen Codeworte ausscheiden. Dazu hat man eine Ausscheidedistanz a vorgegeben, deren Größe von der Blocklänge n

von der Störwahrscheinlichkeit p_0 und von der geforderten Decodiergenauigkeit abhängen wird. Mit Hilfe dieser Ausscheidedistanz a wird man alle Codeworte als zu unwahrscheinlich verwerfen, die sich von $\bar{v}$ in a oder mehr als a Stellen unterscheiden. Man hat damit die Gesamtheit der Codeworte in zwei Gruppen aufgeteilt. In der einen sind diejenigen Codeworte, die noch als gesendete Worte infrage kommen. In der anderen befinden sich diejenigen, die von dem empfangenen n-Tupel $\bar{v}$ zu große Distanz haben und damit als gesendetes Codewort nicht - bzw. bloß mit sehr geringer Wahrscheinlichkeit - infrage kommen. Aus der ersten Untermenge von Codeworten, dieser guten Untermenge, wird man das am wenigsten unwahrscheinliche aussuchen. Dieses Codewort ist dann das Ergebnis des Decodierprozesses.

2. Das Ausscheidekriterium. Seine optimale Wahl

In Punkt 1 wurde beschrieben, daß der Decoder alle diejenigen Codeworte als unwahrscheinlich verwirft, die sich vom Empfangswort in a oder mehr Stellen unterscheiden. Wird bei diesem Ausscheideprozess das in Wirklichkeit gesendete Codewort verworfen, dann entscheidet sich der Decoder falsch. Wir werden verlangen, daß die Wahrschein lichkeit für eine Fehlentscheidung klein ist. Sie soll den Wert 2^{-A} nicht überschreiten. Die positive Konstante A wird Wahrscheinlichkeitskriterium genannt.

Die Zahl der vom Übertragungskanal hervorgerufenen Fehler bezeichen wir mit f: Das Empfangswort unterscheidet sich vom gesendeten Codewort in f Stellen. Da der Decoder alle diejenigen Codeworte verwirft, die sich vom Empfangswort in a oder mehr Stellen unterscheiden, ist die Wahrscheinlichkeit dafür, daß das gesendete Codewort verworfen wird gleich der Wahrscheinlichkeit, daß der Kanal a oder mehr Fehler hervorruft. Diese Wahrscheinlichkeit nennen wir $P(f \geq a)$. Unsere obige Forderung können wir jetzt folgendermaßen formulieren:

$$P(f \geq a) \leq 2^{-A} \qquad (1)$$

Aus Gleichung (1) läßt sich zu gegebenem Ausscheidekriterium A die Ausscheidedistanz berechnen: a ist die kleinste ganze Zahl, die Gleichung (1) noch erfüllt. Auf die optimale Wahl des Ausscheidekriteriums A wird in Kapitel VIII, 2 eingegangen.

Qualitativ kann man jetzt schon feststellen, daß, je größer das Ausscheidekriterium A gewählt wird, d.h. je sicherer das gesendete Codewort in die gute Untermenge aufgenommen werden soll, desto größer wird auch die Ausscheidedistanz a. Es werden Codeworte $\bar{u}$ erst dann in die gute Untergruppe nicht eingeordnet. wenn sie sich mindestens um a Stellen von dem empfangenen n-Tupel $\bar{v}$ unterscheiden. Daher werden in diese gute Gruppe, die das gesendete Codewort enthalten soll, großzügig sehr viele Codeworte aufgenommen, und die gute Gruppe wird sehr umfangreich, die Gruppe der verworfenen Codeworte sehr klein. Wenn man das Ausscheidekriterium A sehr groß wählt, dann kann die Ausscheidedistanz a so groß werden, daß im Extremfall die schlechte Untermenge völlig leer bleibt. In diesem Fall wäre das Ausscheiden erfolglos. Wählt man dagegen A sehr klein - entsprechend fällt auch a klein aus - dann verwirft man sehr viele Codeworte und es kann im Extremfall passieren, daß die gute Untermenge leer bleibt. Im letzteren Extremfall wäre das Ausscheiden ebenfalls erfolglos. Um das jeweilige Ausscheidekriterium der momentanen Störtätigkeit des Kanals anpassen zu können, muß man sich für eine Folge von Kriterien entscheiden. Das bedeutet, daß die Auswahl der guten Untergruppe evtl. in mehreren Schritten geleistet wird.

Man wählt eine wachsende Folge $A_1 < A_2 < A_3 < \ldots$ von Ausscheidekriterien. Die Vergleichsoperation wird gestartet mit dem kleinsten Wert A_1; damit ist garantiert, daß fast alle Codeworte verworfen werden. Nur diejenigen werden als "gut" angenommen, die in fast allen Stellen mit $\bar{v}$ übereinstimmen. Dieser erste Versuch wird auf jeden Fall dann Erfolg haben, wenn keine Stelle gestört wurde. Hat der Kanal in der untersuchten Periode fehlerfrei gearbeitet, stimmt also das empfangene $\bar{v}$ mit dem gesendeten Codewort $\bar{u}$ überein, dann wird das Codewort $\bar{u}$ naturgemäß nicht verworfen, da es ja mit dem Empfangswort völlig übereinstimmt.

Bei geringer Kanalstörtätigkeit wird man mit A_1 auskommen. War die Störtätigkeit aber stärker und wird das gesendete Codewort an einigen Stellen gefälscht, dann wird $\bar{v}$ bei einem vernünftigen Code - den wir hier voraussetzen - sich nicht nur von dem gesendeten, also richtigen, sondern auch von jedem anderen Codewort in einigen Stellen unterscheiden. Das kann zur Folge haben, daß bei Anwendung des Kriteriums A_1 alle Codeworte als nicht infrage kommend verworfen werden, die gute Untermenge also leer bleibt. Dann muß man zu Kriterium A_2 übergehen.

Es ergibt sich eine größere Ausscheidedistanz a. Es werden - möglicherweise - weniger Worte als unannehmbar ausgeschieden. Wenn die gute Untergruppe jetzt nicht mehr leer bleibt, ist man am Ziel. Ansonsten muß man zu den weiteren Ausscheidekriterien A_3, A_4, ... übergehen.

Es ist nochmals darauf hinzuweisen, daß bei dem ersten Decodierschritt (mit dem Ausscheidekriterium A_1) das gesendete Codewort mit der größten Wahrscheinlichkeit fälschlich verworfen wird. Allerdings muß dies zu keinem endgültigen Decodierfehler führen. Wenn nämlich die anderen Codeworte sich von dem empfangenen n-Tupel $\bar{v}$ noch mehr unterscheiden als das gesendete, so werden die ersteren erst recht ausgeschieden, also ebenfalls in die schlechte Untermenge eingeordnet. Damit bleibt die gute Untermenge leer, und wir müssen zum nächsten Ausscheidekriterium übergehen und mit diesem die Gesamtheit der Codeworte neu durcharbeiten.

3. Das endgültige von WOZENCRAFT und REIFFEN entwickelte Verfahren

Die Schranke 2^{-A} für das irrtümliche Einordnen des gesendeten Codewortes in die schlechte Untermenge sei gegeben; damit ist auch der Wert a der Ausscheidedistanz bestimmt.

Nehmen wir als Beispiel 1) an, daß das folgende n-Tupel $\bar{v}$ empfangen worden ist und es mit dem darunterstehenden Codewort $\bar{u}_i$ verglichen werden soll:

$$\bar{v} = (1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1)$$
$$\bar{u}_i = (0\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 0)$$

Es scheint hier nicht nötig zu sein, den Vergleich - links beginnend - bis zur letzten, der n-ten Stelle durchzuführen, denn man darf möglicherweise schon nach dem Vergleich der ersten 4 oder zumindest 6 Stellen das Codewort $\bar{u}_i$ verwerfen, ohne die gegebene Fehlergrenze zu verletzen.

Um dafür genauere Regeln aufstellen zu können, wird man die Ausscheidedistanz nicht nur für die Gesamtheit aller n Stellen berechnen, sondern auch für die Zwischenwerte i mit $1 \leq i \leq n$ und so die Ausscheidedistanzen a_i erhalten. Wir setzen $a_n = a$.

Um die gegebene Schranke 2^{-A} des Wahrscheinlichkeitskriterium nicht zu verletzen, muß für jede Ausscheidedistanz a_i eine zu Gleichung (1) analoge Bedingung erfüllt sein.
Die Zahl der Fehler in den ersten i Stellen sei mit f_i bezeichnet; es gilt f_n=f. Zu jedem Ausscheidekriterium A hat man eine Folge von n Ausscheidedistanzen a_i, die die Ungleichungen

$$P(f_i \geq a_i) \leq 2^{-A} \quad (i = 1, 2, \ldots, n) \qquad (2)$$

erfüllen; als Ausscheidedistanz a_i bei der Untersuchung der ersten i Stellen ist dabei der kleinste Wert zu wählen, der Ungleichung (2) gerade noch erfüllt.

Liegen die Ausscheidedistanzen a_1 bis a_n vor, so wird die Vergleichsprozedur folgendermaßen durchgeführt: beginnend mit i=1 wird die Zahl der Fehler f_i in den ersten i Stellen verglichen mit der entsprechenden Ausscheidedistanz a_i. Sobald für einen bestimmten Wert i gilt: $f_i > a_i$, wird das gerade untersuchte Codewort verworfen. Nehmen wir einmal an, die Folge der a_i laute für einen Code mit der Blocklänge n=12:

n	1	2	3	4	5	6	7	8	9	10	11	12
a_n	2	3	4	4	4	5	5	5	5	5	6	6

Wenden wir diese Ausscheidedistanzen auf Beispiel 1) an: für dieses ergibt sich das folgende Fehlermuster $\bar{e}$ und die daruntersteheende Fehlerzahlfolge

$\bar{e}$	1	0	1	1	0	1	1	1	0	0	0	1
f_n	1	1	2	3	3	4	5	6	6	6	6	7

Da die Fehlerzahl nicht die Stellenzahl übersteigen kann, besagt die obige Folge (n, a_n), daß man auf jeden Fall erst nach 4 Stellen verwerfen kann, denn für $i < 4$ gilt $f_i \leq i < a_i$. Im vorliegenden Fall kann man sogar erst nach 7 Stellen verwerfen, denn erst dann ist $f_i > a_i$, da $f_i = 5$ und $a_i = 5$ ist. In der Decodieranlage muß also eine Matrix gespeichert sein, bzw. die Größen der Matrix müssen im Bedarfsfall errechnet werden können, die alle

Ausscheidedistanzen zu den Kriterien A_1, A_2, ... enthalten. Wir bezeichnen die Ausscheidedistanzen, die zum Kriterium A_i gehören, mit $a_{i,1}$, $a_{i,2}$, ..., $a_{i,n}$. Mit Hilfe dieser Distanzen kann man dann die gewünschte Unterteilung der Codeworte in jedem Fall ausführen.

4. Die Codebaumstruktur. Seine WR-Decodierung

Noch wirkungsvoller könnte das beschriebene Verfahren angewandt werden, wenn wir mit einer einzigen negativen Entscheidung über ein Codewort zugleich auch dessen benachbarte Codeworte mit verwerfen können. Diese Möglichkeit bietet eine Codewortstruktur, die sich als Codebaum beschreiben läßt. Figur 5.1

	X_1	X_2	X_3	X_4		
0 ↑	000	000	000	000	$\bar{u}_1$	$\bar{x}_1$ = (0000)
				111	$\bar{u}_2$	$\bar{x}_2$ = (0001)
			111	011	$\bar{u}_3$	$\bar{x}_3$ = (0010)
				100	$\bar{u}_4$	$\bar{x}_4$ = (0011)
		111	011	001	$\bar{u}_5$	$\bar{x}_5$ = (0100)
				110	$\bar{u}_6$	$\bar{x}_6$ = (0101)
			100	010	$\bar{u}_7$	$\bar{x}_7$ = (0110)
				101	$\bar{u}_8$	$\bar{x}_8$ = (0111)
1 ↓	111	011	001	011	$\bar{u}_9$	$\bar{x}_9$ = (1000)
				100	$\bar{u}_{10}$	$\bar{x}_{10}$ = (1001)
			110	000	$\bar{u}_{11}$	$\bar{x}_{11}$ = (1010)
				111	$\bar{u}_{12}$	$\bar{x}_{12}$ = (1011)
		100	010	010	$\bar{u}_{13}$	$\bar{x}_{13}$ = (1100)
				101	$\bar{u}_{14}$	$\bar{x}_{14}$ = (1101)
			110	001	$\bar{u}_{15}$	$\bar{x}_{15}$ = (1110)
				110	$\bar{u}_{16}$	$\bar{x}_{16}$ = (1111)

Figur 5.1: Codebaum und zugehörige Nachrichten für einen Code mit den Codegrößen m=1, k=2, r=3

zeigt als Beispiel 2) einen Codebaum für die Codegrößen m=4, k=8, also mit der Blocklänge n = m + k = 12. Die Gesamtzahl der Codeworte ist $2^m = 2^4 = 16$. Die zu jedem Codewort $\bar{u}_i$ des Codebaums gehörige Nachricht ist neben dem Codebaum in Figur 5.1 aufgeführt. Über dem Codebaumschema sind die Stufen des Codebaumes durchnumeriert. Die Nummer i gibt an, welche Nachrichtenstelle die Entscheidung für den oberen (x_i=0) oder unteren(x_i=1) Ast herbeiführt. Da im Beispiel die Nachrichtenrate $R = \frac{m}{n} = \frac{1}{3}$ ist, entsprechen jeder Nachrichtenstelle 3 Codewortstellen.

Man erhält zu einer vorgegebenen Nachricht das gesuchte Codewort, indem man sich in der Tafel von links nach rechts vorarbeitet; bei jedem Schritt, mit dem man in die Tafel tiefer eindringt, hat man eine von zwei Fortsetzungsmöglichkeiten zu wählen. Man wählt die obere der angebotenen Möglichkeiten, wenn die entsprechende Nachrichtenstelle eine 0 ist, andernfalls die untere.

Als Beispiel sei die Nachricht: $\bar{x}$ = (1, 0, 1, 1) zu codieren. Man beginnt bei der Verzweigung auf der ersten vertikalen Linie und erhält als erste 3 Codewortstellen 111, da die erste Nachrichtenstelle 1 ist. Man sieht, daß Codeworte, die einer Nachricht mit gleicher erster Stelle entsprechen, alle dieselben n Anfangsstellen haben. Bei der Codierung der zweiten Stelle steht man jetzt am unteren Knoten in der zweiten vertikalen Linie und hat 011 zu wählen, da die zweite Nachrichtenstelle eine 0 ist. Das Codewort beginnt also mit 111 011 und lautet vollständig: 111 011 110 111.

Hier ist eine Bemerkung einzuflechten, von der in Kapitel VIII, 2 Gebrauch gemacht wird: die Zahl der verschiedenen i-stelligen Codewortanfänge der Länge i gehorcht dem Gesetz

$$S(i) \leq K \cdot 2^{i \cdot R} \qquad (3)$$

wobei K eine Konstante bedeutet.
Im Beispiel gilt

2 mögliche Kombinationen der Länge 1, 2, 3
4 mögliche Kombinationen der Länge 4, 5, 6
8 mögliche Kombinationen der Länge 7, 8, 9

Für K genügt im Beispiel 2) also der Wert 3.
Aus der Codebaumstruktur in Figur 5.1 sehen wir, daß die Gesamtheit der Codeworte in zwei Gruppen unterteilt ist. In der oberen Gruppe sind alle Codeworte, die einer mit 0 beginnenden Nachricht entsprechen; in der unteren Gruppe sind die anderen.

Das Ziel beim Decodieren mit dem WR-Verfahren ist es, festzustellen, ob das empfangene n-Tupel in die untere oder in die obere Untergruppe gehört, also zu entscheiden, ob die Nachricht mit einer 1 oder einer 0 beginnt.
Wir gehen in folgender Weise vor: der Decoder beginnt mit dem kleinsten Kriterium A_1 und den zugehörigen Ausscheidedistanzen $a_{1,1}, a_{1,2}, \ldots, a_{1,n}$. Er erzeugt den ganzen Satz von Codeworten, vergleicht diese mit $\bar{v}$ und scheidet - beginnend mit i=1 - jedes Codewort aus, dessen erste i Stellen sich von den empfangenen ersten i Stellen in $a_{1,i}$ oder mehr Stellen unterscheiden. Wird ein Codewort nach i Stellen ausgeschieden, so sind damit natürlich zugleich alle Codeworte verworfen, die dieselben Anfangsstellen haben. Man beginnt beispielsweise beim obersten Codewort der Liste, dasja der Nachricht $\bar{x}_0$ = (0 0 0 0) entspricht. Kann man dieses erste Codewort nach Vergleich der fünften Stelle verwerfen, so sind damit gleichzeitig die Codeworte $\bar{u}_1$ bis $\bar{u}_4$ verworfen und der Decoder fährt mit der Reproduktion des Codewortes $\bar{u}_5$ fort. Kann er aber Codewort $\bar{u}_0$ schon mit Stelle 3 verwerfen, so sind damit auch die Codeworte 1 bis 8 ausgeschieden und der Decoder fährt mit diesem Verfahren so lange fort, bis er ein Codewort annehmen kann, und entscheidet sich damit für diejenige Gruppe, in der das angenommene Codewort liegt. Mit der Annahme der oberen bzw. unteren Gruppe ist die erste Nachrichtenstelle als 0 bzw. als 1 decodiert.

Gelingt es beim ersten Kriterium nicht, ein Codewort bis zur Blocklänge n anzunehmen, wird also der gesamte Satz der Codeworte ausgeschieden, so muß der Decoder die Prozedur mit dem zweiten Kriterium wiederholen. Er durchläuft die Folge der Ausscheidekriterien A_j so lange, bis er zum ersten Mal ein Codewort annehmen kann. Man sieht, daß die Zahl der Rechenschritte, die nötig ist, um eine Stelle zu decodieren, eine Zufallsvariable ist. Die Zahl der Schritte wird dann klein sein, wenn nur wenige Fehler aufgetreten sind, und der Decoder mit einem kleinen Kriterium A_i auskommt. Sie wird stark anwachsen, wenn viele Stellen gefälscht wurden.

5. Anwendung der WR-Decodierung auf sequentielle Code

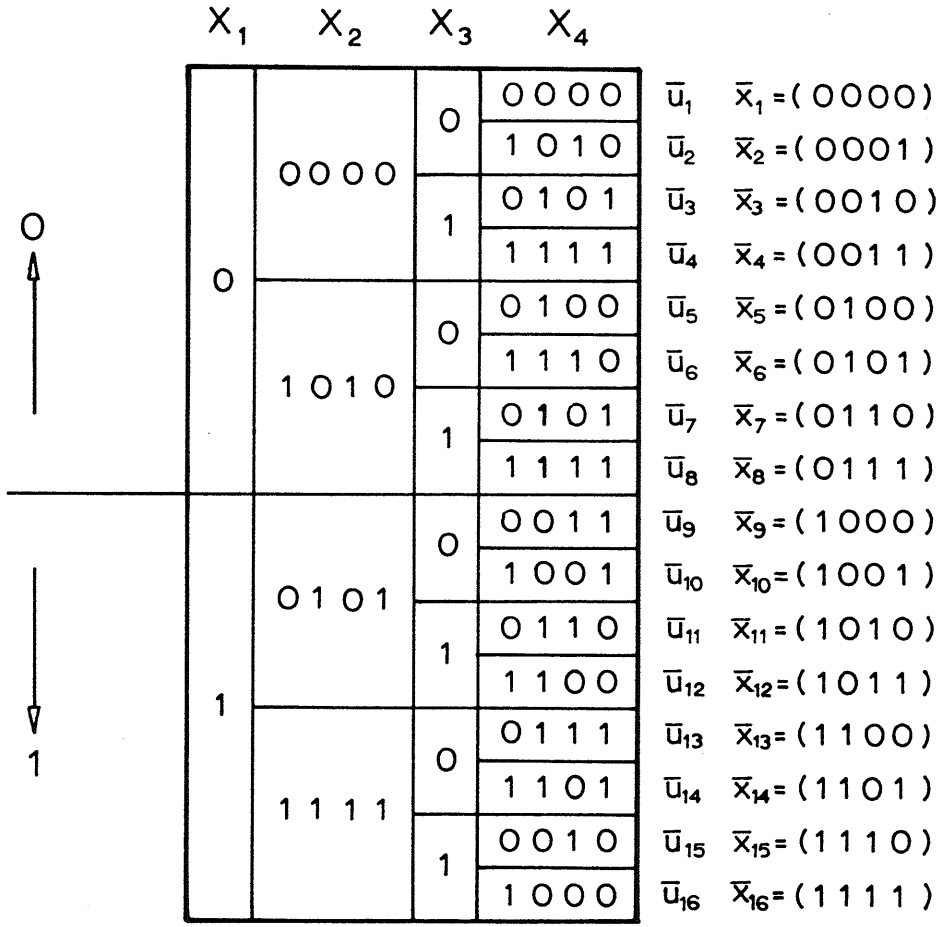
Die in Kapitel IV behandelten linearen, sequentiellen Code besitzen die beschriebene Codebaumstruktur. Nach den Bemerkungen von Kapitel IV, 3 genügt es, einen Decodierprozess für die ersten n Stellen des n_0-stelligen Anfangscodewortes zu finden. Beschränken wir uns auf das Anfangscodewort, dann haben wir einen endlichen Gruppencode vor uns, dessen Codebaum sich auf direktem Wege aufschreiben läßt. Für m=1, also für eine Nachrichtenrate der Form $R = \frac{1}{n}$, hat der Codebaum eines sequentiellen Code eine besonders homogene Gestalt. Wählen wir m=1, k=2 und r=3 und als Generatormuster $b^{(1)}$ = (111 011 001 011), so erhalten wir gerade den Codebaum von Beispiel 2), den Figur 5.1 zeigt.

Wir wissen, daß bei beliebiger Rate $R = \frac{m}{n}$ gerade m Generatormuster zu wählen sind. Der entsprechende Codebaum ist nicht mehr so homogen aufgebaut wie im Fall m=1. Der Codebaum, in dem alle Anfangscodeworte mit ihren n_0 Stellen aufgeführt sind, ist in (r+1) Stufen mit je n Stellen aufgeteilt. Für m=1 entspricht jede Stufe einer der (r+1) Nachrichtenstellen der Anfangscodeworte. Im allgemeinen Fall (m>1) wird jeder der (r+1) Stufen des Codebaumes noch m-fach in Unterstufen unterteilt, durchaus aber nicht notwendig in gleich große Abschnitte. Die Gesamtzahl der Stufen beträgt in jedem Fall $m_0 = (r+1)\,m$. Nehmen wir einen systematischen Code, so bestehen die ersten (m-1) Unterstufen aus je einer Binärzahl, die den Wert der jeweiligen Nachrichtenstelle hat. Dagegen besteht die m-te Unterstufe einer jeden Stufe aus (k+1) Binärzahlen, von denen die erste den Wert der m-ten Nachrichtenstelle hat, während die anderen k Stellen Kontrollstellen sind. Den Aufbau eines allgemeinen Codebaumes soll Beispiel 1b) aus Kapitel IV für einen Code mit m=2, k=3 und r=1 verdeutlichen. Als Generatormuster wurden gewählt:

$$b^{(1)} = (1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 1)$$
$$b^{(2)} = (0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0)$$

Den Codebaum von Beispiel 1b) Kapitel IV zeigt Figur 5.2.

Wir sehen jetzt auch, wie die Binärstellen eines Generatormusters zu wählen sind, wenn wir WR-decodieren wollen: zunächst ist es natürlich erstrebenswert, daß die Gesamthammingdistanz der Anfangscodeworte groß ist.



Figur 5.2: Codebaum und zugehörige Nachrichten für einen Code mit den Codegrößen m=2, k=3, r=3

Darüber hinaus ist es aber Ziel der WR-Decodierung, die nicht infrage kommenden Codeworte schon möglichst früh zu verwerfen, also schon an einer Stelle i mit $i < n_0$. Das kann dadurch erleichtert werden, daß die "aktuelle" Hammingdistanz der ersten i Stellen für $i = 1, 2, \ldots, n_0$ ebenfalls möglichst groß ist. Bei der Wahl der Generatorstellen ist demgemäß schrittweise vorzugehen und an jeder Stelle i ist die anstehende Gruppe von m Binärzahlen (eine für jedes der m Generatormuster) so zu wählen, daß die "aktuelle" Hammingdistanz möglichst groß wird. Für die optimale Wahl der Generatormuster erfordert dies die Beobachtung aller möglichen Codewortanfänge, was ab $i \approx 5$ zweckmäßigerweise auf einer elektronischen Rechenanlage ausgeführt wird. Auf dem Rechner TR 4 der Deutschen Forschungsgemeinschaft im Rechenzentrum Stuttgart wurden die optimalen Generatormuster für geläufige Nachrichtenraten zum Teil für recht große Codelängen berechnet (z.B. für $R = \frac{1}{3}$ bis $n_0 = 51$) und liegen beim Verfasser auf. Soll mit einer Codelänge übertragen werden, die größer als die Länge der berechneten Optimalgeneratormuster ist, dann müssen die zusätzlichen Generatormusterstellen zufällig ausgewürfelt werden.

Die dieses Kapitel abschließende Bemerkung bezieht sich auf eine Modifizierung des Decodierprinzips oder - genauer gesagt - des Absuchprozesses. Bis jetzt wurde der Codebaum nach einem Codewort, das man aufgrund eines Ausscheidekriteriums bis zur Verflechtungslänge n_0 nicht ausscheiden kann, abgesucht, indem man in jedem Decodiertakt etwa vom Nullcodewort ausging. Hat man bei einem bestimmten Kriterium ein annehmbares Wort gefunden, dann ist dadurch im Codebaum ein bestimmter Pfad vorgezeichnet. Im Falle nicht zu großer Störtätigkeit des Übertragungskanals ist zu erwarten, daß dieser Pfad der "richtige" ist. Der Decoder kommt dann im nächsten Schritt am schnellsten zum Ziel, wenn er die Vergleichsprozedur mit diesem Codewort beginnt - unter Weglassen der abge arbeiteten n Stellen. Man verringert die Zahl der Rechenschritte, wenn das Absuchen des Codebaumes auf dem vorgezeichneten Pfad anfängt.

Hinzuweisen ist noch auf einen verfeinerten Absuchalgorithmus, der von FANO[7] entwickelt worden ist. Wir haben festgestellt, daß die Zahl der Rechenschritte eine Zufalls variable ist. Die Zahl der Rechenschritte bleibt dann gering, wenn wir in einer störungsarmen Periode sind; sie steigt mit wachsender Störtätigkeit an. Unterläuft dem Decoder ein Decodierfehler, so haben wir in Kapitel IV, 3 gesehen, daß ein sequentielles Verfahren dazu neigt, Decodierfehler fortzupflanzen. Bei WR-Decodierung äußert sich das in einem rapiden Ansteigen der Zahl der Rechenschritte. Dieser Effekt kann nach FANO folgendermaßen genutzt werden: Der Decoder macht in diesem Fall eine gewisse Anzahl von Entscheidungen rückgängig und versucht einen anderen Pfad durch den Codebaum zu finden. Falls dies der richtige Pfad ist, wird er sich durch einen weitaus geringeren Rechenaufwand auszeichnen. Der FANO-Algorythmus für WR-Decodierung zeigt einen Weg, der Fehlerfortpflanzung bei sequentiellen Verfahren entgegenzuwirken.

Kapitel VI

SW-DECODIERUNG

1. Einführung

Dieses Kapitel ist J.L. MASSEY's "Threshold Decoding" gewidmet. Die englische Vokabel "threshold" bedeutet soviel wie Schwellenwert; daran soll die gewählte Abkürzung "SW-Decodierung" erinnern.

SW-Decodierung ist ein Verfahren, das auf systematische Code angewandt wird, also auf Code, bei denen die Nachrichtenstellen der Quelle unverändert übernommen werden. Der Decoder kann sich darauf beschränken,nicht alle empfangenen Stellen zu decodieren, sondern nur die Nachrichtenstellen.

SW-Decodierung kann vorteilhaft sowohl bei bestimmten Blockcode - insbesondere bei zyklischen Code - als auch bei sequentiellen Code angewandt werden, wobei in dieser Arbeit allerdings das Gewicht auf sequentieller SW-Decodierung liegt und SW-Blockdecodierung nur prinzipiell besprochen wird.

Es sei daran erinnert (Kapitel IV, 3), daß zur Decodierung sequentieller Code ein Verfahren gefunden werden muß, das es erlaubt, die ersten m Nachrichtenstellen mit Hilfe des n_0-stelligen Anfangscodewortes zu decodieren. Es muß also ein spezielles Blockdecodierverfahren gefunden werden.

Eine bestimmte Nachrichtenstelle x_ρ beeinflußt höchstens (n_0-1) weitere Stellen - bei Blockcodierung geht die Codelänge n_0 in die Blocklänge n über. Aus den aktuellen Werten dieser n_0 Stellen errechnet der Decoder einen Entscheidungswert für die zu decodierende Stelle v_ρ und vergleicht diesen mit einer Schranke T, dem Schwellenwert. Er ändert genau dann die empfangene Stelle v_ρ ab, wenn der Entscheidungswert größer als der feste Schwellenwert T ist.

MASSEY's SW-Verfahren beruht auf der folgenden Idee:

Um eine bestimmte Nachrichtenstelle x_ρ aus v_ρ zu decodieren, wählt der Decoder eine für diese Nachrichtenstelle charakteristische Gruppe $\{A_j\}^{(\rho)}$ von Prüfgleichungen A_j aus der Gesamtheit aller Prüfgleichungen aus. Dabei sollen die ausgewählten Prüfgleichungen alle von Nachrichtenstelle x_ϱ beeinflußt werden, während jede andere Stelle höchstens von einer der ausgewählten Prüfgleichungen erfaßt werden darf. Wird nun Nachrichtenstelle x_ρ gestört, dann sind alle Prüfgleichungen A_j aus der Gruppe $\{A_j\}^{(\rho)}$ nicht erfüllt - zumindest dann, wenn die vom Gleichungssatz $\{A_j\}^{(\rho)}$ erfaßten anderen Stellen richtig übertragen wurden. Ist aber irgendeine andere Stelle gestört, x_ϱ aber richtig übertragen worden, so ist bloß eine einzige Prüfgleichung A_j nicht erfüllt. Die Zahl der nicht erfüllten Prüfgleichungen des Satzes $\{A_j\}^{(\rho)}$ erlaubt also Rückschlüsse auf die Stelle x_ρ. Um solche gewünschten Prüfgleichungen gezielt konstruieren zu können, führt MASSEY sog. "zusammengesetzte Prüfgleichungen" ein.

MASSEY hat zwei SW-Verfahren hergeleitet. Das erste ist ein algebraisches Verfahren, das zweite ein statistisches. Diskutiert werden die beiden Begriffe algebraisch und statistisch in Kapitel VII, 8.

2. Zusammengesetzte Prüfgleichungen. Orthogonale Prüfgleichungssysteme

Bevor die beiden grundlegenden Sätze des MASSEY'schen Decodierverfahrens formuliert werden können, müssen wir noch eine Verallgemeinerung der Prüfgleichungen bei systematischen Blockcode besprechen. Im Kapitel II über Blockcodierung hatten wir in Teil 7 die Prüfwerte s_μ definiert:

$$s_\mu = t_\mu \oplus v_\mu = \left(\sum_{\varkappa=1}^{m} p_{\mu,\varkappa}\ v_\varkappa\right)_{\mathrm{mod}\text{-}2} + v_\mu \qquad \mu = m+1, \ldots, n \qquad (1)$$

Dabei sind die t_μ die Werte der Kontrollstellen, die sich aufgrund des Prüfschemas aus den empfangenen, möglicherweise falsch übertragenen Nachrichtenstellen ergeben haben, während die v_μ die Werte der empfangenen, auch möglicherweise gefälschten Kontrollstellen sind.

Die Prüfwerte s_μ geben also an, ob die empfangenen Kontrollstellen v_μ ($\mu = m+1, m+2, \ldots, n$) mit den berechneten Kontrollstellen t_μ übereinstimmen ($s_\mu=0$) oder nicht ($s_\mu=1$).

Die $p_{\mu,\varkappa}$ sind die Elemente des Prüfschemas. Wenn alle k Prüfgleichungen erfüllt werden, also alle k Prüfwerte $s_\mu = 0$, dann ist das empfangene n-Tupel $(v_1, v_2, \ldots, v_n)$ ein Codewort. Wir hatten auch gezeigt, daß die s_μ durch die Fehlermusterstellen darstellbar sind:

$$s_\mu = \Big(\sum_{\varkappa=1}^{m} p_{\mu,\varkappa}\ e_\varkappa\Big)_{\text{mod-2}} + e_\mu, \qquad \mu = m+1, \ldots, n \tag{2}$$

wobei $\bar{e} = (e_1, e_2, \ldots, e_n)$ das Fehlermuster bedeutet. Als zusammengesetzte Prüfgleichungen bezeichnen wir nun Linearkombinationen der Prüfgleichungen. Die zusammengesetzten Prüfwerte A_j ergeben sich als Linearkombination der Prüfwerte s_μ:

$$A_j = \sum_{\mu=m+1}^{n} b_{j,\mu}\ s_\mu \tag{3}$$

auch hier sind die Koeffizienten $b_{j,\mu}$ Binärzahlen. Die Größe $b_{j,\mu}$ gibt an, ob Prüfgleichungen μ von der zusammengesetzten Prüfgleichung A_j erfaßt wird ($b_{j,\mu} = 1$) oder nicht ($b_{j,\mu} = 0$). Da die s_μ ihrerseits Linearkombinationen der Fehlerstellen $(e_1, e_2, \ldots, e_n)$ sind, lassen sich die zusammengesetzten Prüfwerte A_j auch unmittelbar darstellen:

$$A_j = \sum_{\varkappa=1}^{n} a_{j,\varkappa}\ e_\varkappa \tag{4}$$

Die einfachen Prüfgleichungen sind ein Sonderfall der zusammengesetzten Prüfgleichungen. Unter Prüfgleichungen wollen wir in diesem Kapitel sowohl einfache als auch zusammengesetzte Prüfgleichungen verstehen.

Für die Decodierung einer Stelle x_ρ bzw. e_ρ sollen alle Prüfgleichungen eines Satzes $\{A_j\}$ von J Prüfgleichungen diese Stelle erfassen, während alle anderen Stellen höchstens einmal erfaßt werden dürfen. MASSEY bezeichnet diesen Satz von J Prüfgleichungen

orthogonal auf der Stelle ρ, was durch den Index ρ angedeutet wird: $\{A_j\}^{(\rho)}$. Für J=3 und m=5 könnte die Koeffizientenmatrix der A_j wie im folgenden Beispiel 1) aussehen:

$$\begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Der Satz dieser drei zusammengesetzten Prüfgleichungen A_j ist orthogonal auf e_1: $\{A_j\}^{(1)}$. Die Koeffizientenmatrix hat hier n=5 Spalten. Sie kann wieder als Prüfschema aufgefaßt werden und entsteht aus einem ursprünglichen Prüfschema durch Überlagerung von Prüfzeilen. Jedes A_j kontrolliert also die Stelle 1, während alle anderen Stellen von höchstens einer Prüfgleichung A_j erfaßt werden.

Die Zahl der von A_j - außer Stelle ρ - geprüften Stellen sei n_j genannt. Die Gesamtzahl der vom Satz $\{A_j\}$ geprüften Stellen ist damit

$$n_E = 1 + \sum_{j=1}^{J} n_j \tag{5}$$

Bei einem sequentiellen Code empfiehlt sich für die Koeffizienten der zusammengesetzten Prüfgleichungen eine etwas andere Benennung, die sich an das in Kapitel IV, 5 abgeleitete Prüfschema anschließt. Dieses Prüfschema für das Anfangscodewort eines systematischen, sequentiellen Code wird bestimmt durch m·k Prüfmuster $G_i^{(\mu)}$ mit ihren Koeffizienten $g_{i,1}^{(\mu)} \ldots, g_{i,r+1}^{(\mu)}$ ($i = 1, 2, \ldots, m$; $\mu = m+1, \ldots, n$). Die Prüfgleichungen für die Kontrollstellen der μ-ten Leitung: $x_1^{(\mu)}, \ldots, x_{r+1}^{(\mu)}$ ($\mu = m+1, m+2, \ldots, n$) werden bestimmt durch die Prüfmuster $G_1^{(\mu)}, G_2^{(\mu)}, \ldots, G_m^{(\mu)}$. Die Prüfmuster $G_1^{(m+1)}, G_1^{(m+2)}, \ldots, G_1^{(n)}$ bestimmen, wie die Nachrichtenstellen $x_1^{(i)}, x_2^{(i)}, \ldots, x_{r+1}^{(i)}$ der i-ten Eingangsleitung in das Prüfschema eingehen. Aus diesem Prüfschema können wir die

$k_0 = (r+1) \cdot k$ Prüfwerte

$s_1^{(m+1)}, \ldots, s_{r+1}^{(m+1)}, s_1^{(m+2)}, \ldots, s_{r+1}^{(m+2)}, \ldots, s_1^{(n)}, \ldots, s_{r+1}^{(n)}$

ablesen;

dabei gehört Prüfwert $s_\nu^{(\mu)}$ zu Kontrollstelle $u_\nu^{(\mu)}$.

$$s^{(\mu)} = v^{(\mu)} \oplus \Big(\sum_{\kappa=1}^{m} \sum_{\rho=1}^{\nu} g^{(\mu)}_{\kappa,\,\nu-\rho+1}\, v^{(\kappa)}_{\rho}\Big)_{\mathrm{mod}\text{-}2} \qquad (6)$$

$\nu = 1, 2, \ldots, r+1; \qquad \mu = m+1, m+2, \ldots, n$

Mit Hilfe des Fehlermusters $(e^{(1)}_1, e^{(1)}_2, \ldots, e^{(1)}_{r+1})$ ergibt sich wie in IV, 3:

$$s^{(\mu)} = e^{(\mu)} \oplus \Big(\sum_{\kappa=1}^{m} \sum_{\rho=1}^{\nu} g^{(\rho)}_{\kappa,\,\nu-\rho+1}\, e^{(\kappa)}_{\rho}\Big)_{\mathrm{mod}\text{-}2} \qquad (7)$$

Die Binärzahl $b^{(\mu)}_{j,\nu}$ soll nun angeben, ob Prüfgleichung $s^{(\mu)}_{\nu}$ in der zusammengesetzten Prüfgleichung A_j erfaßt wird ($b^{(\mu)}_{j,\nu} = 1$) oder nicht ($b^{(\mu)}_{j,\nu} = 0$). Mit diesen Koeffizienten $b^{(\rho)}_{j,\nu}$ ergeben sich die Prüfwerte A_j zu:

$$A_j = \sum_{\mu=m+1}^{n} \sum_{\nu=1}^{r+1} b^{(\mu)}_{j,\nu} \cdot s^{(\mu)}_{\nu} \qquad (8)$$

Diese Darstellung der zusammengesetzten Prüfwerte wird nützlich sein, um in Teil 4 die Decodierschaltung direkt angeben zu können.

3. Die beiden grundlegenden Sätze der SW-Decodierung

Wir wollen aufgrund der J Prüfgleichungen $\{A_j\}^{(\rho)}$ eine Entscheidung über Ziffer e_ρ herbeiführen. Wir werden zwei Verfahren von MASSEY dazu kennenlernen. Das erste, algebraische Verfahren beruht auf folgendem Satz:
Es sei vorausgesetzt, daß höchstens $\left\lfloor \frac{J}{2} \right\rfloor$ der vom Satz $\{A_j\}^{(\rho)}$ geprüften Stellen falsch übertragen wurden. Wir wählen genau dann $e^*_\rho = 1$, wenn mehr als $\left\lceil \frac{J}{2} \right\rceil$ der A_j nicht erfüllt sind, andernfalls wählen wir $e^*_\rho = 0$. Das bedeutet: genau dann, wenn die normale Summe der - als Dezimalzahlen genommenen - A_j den Wert $\left\lceil \frac{J}{2} \right\rceil$ überschreitet, nehmen wir einen Übertragungsfehler an. Dabei bedeutet $\left\lceil \frac{J}{2} \right\rceil$ den aufgerundeten Wert $\frac{J}{2}$ und $\left\lfloor \frac{J}{2} \right\rfloor$ den abgerundeten Wert $\frac{J}{2}$.

Der Beweis ist sehr einfach: nehmen wir zunächst einmal an, daß $e_\rho = 0$ ist und nach Voraussetzung höchstens $\left\lfloor \frac{J}{2} \right\rfloor$ Fehler insgesamt passiert sind. Damit werden höchstens $\left\lfloor \frac{J}{2} \right\rfloor$ Gleichungen nicht erfüllt: es wird $e^*_\rho = 0$ gewählt - also richtig decodiert. Ist aber $e_\rho = 1$, dann können in den übrigen Gleichungen noch höchstens $(\left\lfloor \frac{J}{2} \right\rfloor - 1)$ Fehler passiert sein. Damit bleiben aber mindestens $J - (\left\lfloor \frac{J}{2} \right\rfloor - 1) = \left\lceil \frac{J}{2} \right\rceil + 1$ Gleichungen nicht erfüllt: es wird $e^*_\rho = 1$ gewählt und die ρ-te Stelle korrekt abgeändert.

Das zweite Decodierverfahren beachtet die Kanalstatistik, also beim BSC die Fehlerwahrscheinlichkeit p_0. Es soll eine Entscheidung über die Nachrichtenstelle u_ρ getroffen werden, aufgrund der Information, die der Satz $\{A_j\}^{(\rho)}$ von Prüfgleichungen liefert. Man trifft sicher dann die beste Entscheidung, wenn e^*_ρ denjenigen Wert Y erhält, für den $P(e_\rho = Y | \{A\}^{(\rho)})$ am größten ist. Nach der Regel von BAYES gilt:

$$P(e_\rho = Y | \{A_j\}^{(\rho)}) = \frac{P(\{A_j\}^{(\rho)} | e_\mu = Y) \cdot P(e_\rho = Y)}{P(\{A\}^{(\rho)})} \qquad (9)$$

Wegen der Orthogonalität der A_j auf e_ρ und der statistischen Unabhängigkeit der Kanaleinwirkungen gilt:

$$P(\{A_j\}^{(\rho)} | e_\rho = Y) = \prod_{j=1}^{J} P(A_j | e_\rho = Y) \qquad (10)$$

Da der Nenner in Gleichung (9) unabhängig von e_ρ ist, muß man Y so wählen, daß $P(e_\rho = Y) \cdot \prod_{j=1}^{J} P(A_j | e_\rho = Y)$ ein Maximum wird.

Oder genau dann ist Y=1 zu wählen, wenn

$$P(e_\rho = 1) \cdot \prod_{j=1}^{J} P(A_j | e_\rho = 1) > P(e_\rho = 0) \cdot \prod_{j=1}^{J} (A_j | e_\rho = 0) \qquad (11)$$

Diese Bedingung wird logarithmiert und noch umgeformt zu:

$$\ln P(e_\rho = 1) + \sum_{j=1}^{J} \ln P(A_j | e_\rho = 1) > \ln P(e_\rho = 0) + \sum_{j=1}^{J} \ln P(A_j | e_\rho = 0) \qquad (12)$$

$$\sum_{j=1}^{J} \left[\ln P(A_j | e_\rho = 1) - \ln P(A_j | e_\rho = 0) \right] > \ln \frac{q_0}{p_0} \qquad (13)$$

oder auch

$$\sum_{j=1}^{J} \ln \frac{P(A_j | e_\rho = 1)}{P(A_j | e_\rho = 0)} > \ln \frac{q_0}{p_0} \qquad (14)$$

Die Wahrscheinlichkeit, daß unter den n_ν Stellen, die von A_ν zusätzlich zur Stelle ρ geprüft werden, eine ungerade Zahl von Fehlern auftritt, wird mit p_ν bezeichnet. Es gilt:

$$P(A_j = 0 | e_\rho = 1) = P(A_j = 1 | e_\rho = 0) = p_j \qquad (15)$$

$$P(A_j = 1 | e_\rho = 1) = P(A_j = 0 | e_\rho = 0) = q_j = 1 - p_j \qquad (16)$$

Wenn wir im folgenden die A_j als natürliche Zahlen behandeln, dann können wir (14) mittels (15) und (16) umschreiben:

$$\sum_{j=1}^{J} (2 A_j - 1) \ln \frac{q_j}{p_j} > \ln \frac{q_0}{p_0} \qquad (17)$$

$$\sum_{j=1}^{J} A_j \left(2 \ln \frac{q_j}{p_j}\right) < \sum_{j=0}^{J} \ln \frac{q_j}{p_j} \qquad (18)$$

Um die Wahrscheinlichkeiten p_j zu berechnen für $j = 1, 2, \ldots, J$ benötigen wir noch folgenden Satz, dessen Beweis z.B. bei MASSEY[2] angegeben ist: $e_1, e_2, \ldots, e_{n_j}$ sei eine Folge von n_j statistisch unabhängigen Binärzahlen mit den einzelnen Wahrscheinlichkeiten $P(e_i=1) = 1 - P(e_i=0) = r_i$; dann ist die Wahrscheinlichkeit p_j dafür, daß die Zahl der Einsen unter diesen n_j Ziffern ungerade ist:

$$p_j = \frac{1}{2}\left[1 - \prod_{i=1}^{n_j} (1 - 2 r_i)\right] \qquad (19)$$

Bei uns sind alle $r_i = p_0$, deshalb wird:

$$p_j = \frac{1}{2}\left[1 - (1 - 2 p_0)^{n_j}\right] \qquad (20)$$

Wir haben also genau dann $e_\rho^* = 1$ zu wählen, wenn

$$\sum_{j=1}^{J} A_j \; 2 \ln \frac{1 + (1 - 2 p_0)^{n_j}}{1 - (1 - 2 p_0)^{n_j}} > \sum_{j=0}^{J} \ln \frac{q_j}{p_j} \qquad (21)$$

4. Die SW-Decodieranlage sequentieller Code

Da die Decodieranlage für den Fall m=1 besonders übersichtlich ist, wollen wir uns zunächst auf diesen Fall beschränken. Wenn m=1 ist, benötigen wir einen einzigen Satz von Prüfgleichungen, der orthogonal auf $e_0^{(1)}$ ist. Die Decodierregel lautet in diesem Fall:
Genau dann ist $e_0^{(1)*} = 1$ zu wählen, wenn $\sum_{j=1}^{J} W_j A_j > T$ ist. (22)

Dabei sind die A_j als natürliche Zahlen zu behandeln. Die Gewichtsfaktoren W_j sind:

$$W_j = 1 \text{ für das erste Verfahren} \qquad (23)$$

$$W_j = 2 \ln \frac{q_j}{p_j} \text{ für das zweite Verfahren} \qquad (24)$$

Die Schranke T lautet für:

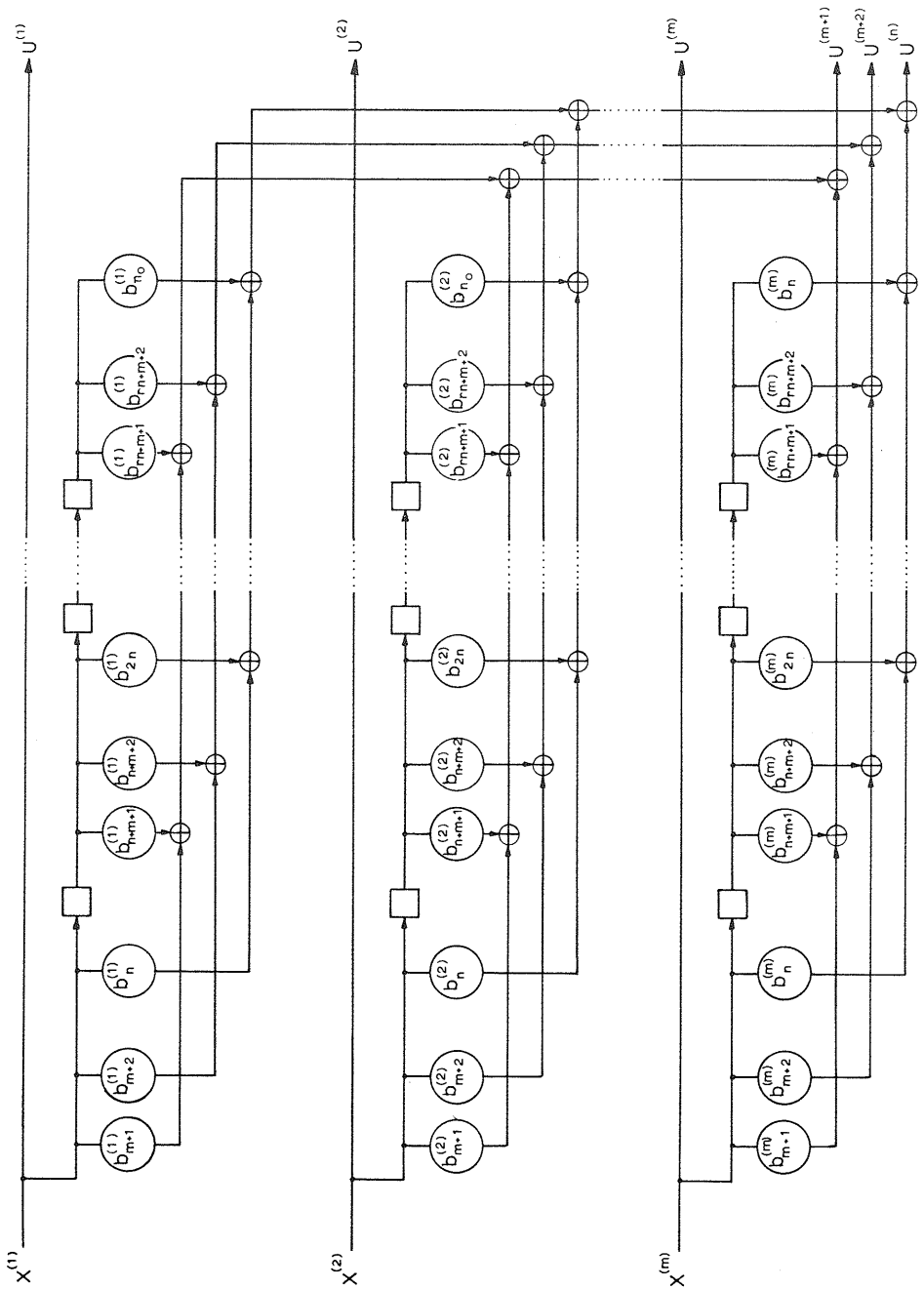
$$\text{Verfahren 1:} \quad T = \left\lceil \frac{J}{2} \right\rceil \qquad (25)$$

$$\text{Verfahren 2:} \quad T = \frac{1}{2} \sum_{j=0}^{J} W_j \qquad (26)$$

Dabei wurde $W_0 = 2 \ln \frac{q_0}{p_0}$ gesetzt. (27)

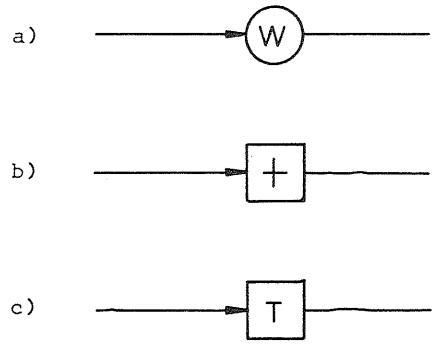
Der Decoder für den Fall m=1 kann wie in Figur 6.1 verwirklicht werden. In Arbeitstakt $(r+\nu)$ wird die Nachrichtenstelle $x_\nu^{(1)}$ decodiert; die Entscheidung über Stelle $x_1^{(1)}$ wird also im Takt (r+1) getroffen.

In Figur 6.1 werden links die empfangenen n-Tupel $\bar{v}$ in den Arbeitstakten $\nu = 1, 2, \ldots$ angeboten. Der links oben angedeutete sequentielle Coder ermittelt in Arbeitstakt ν aus den in den r vorhergehenden Takten empfangenen und gespeicherten Nachrichtenstellen $v_{\nu-1}^{(1)}, v_{\nu-2}^{(1)}, \ldots, v_{\nu-r}^{(1)}$ und der gerade anliegenden



Figur 6.1: SW-Decodieranlage für einen sequentiellen Code mit m=1

Nachrichtenstelle $v_\nu^{(1)}$ die zugehörigen k Kontrollstellen für den Takt ν; zu Beginn der Übertragung ist der Coder entleert, d.h. seine Speicher enthalten nur Nullen. Die aus dem empfangenen (möglicherweise gefälschten) Nachrichtenstellen berechneten Kontrollstellen werden in modulo-2 Addiergliedern verglichen mit den im selben Takt empfangenen (und auch möglicherweise gefälschten) Kontrollstellen $v_\nu^{(2)}$, $v_\nu^{(3)}$, ..., $v_\nu^{(n)}$. Die k Vergleichswerte (s=0 bei Übereinstimmung, sonst s=1) $s_\nu^{(2)}$, $s_\nu^{(3)}$, ..., $s_\nu^{(n)}$ werden in den anschließenden Schieberegistern für die nachfolgenden r Takte gespeichert. Damit stehen im Arbeitstakt $(r+\nu)$ alle interessierenden k_0 Prüfwerte zur Verfügung, aus denen Nachrichtenstelle $v_\nu^{(1)}$ decodiert werden kann. Zunächst werden die zusammengesetzten Prüfwerte $A_1, A_2, \ldots, A_J$ nach Gleichung (8) entsprechend den Auswahlkoeffizienten b_j mittels modulo-2 Addiergliedern erzeugt. Der Wert A_j wird durch die in Figur 6.2a dargestellten Schaltungen mit dem Wert W_j multipliziert. Die Addierschaltung von Figur 6.2b bildet die gewöhnliche Summe gemäß Gleichung (22). Die Schwellwertschaltung nach Figur 6.2c ergibt dann den Korrekturwert $e_\nu^{(1)*} = 1$, wenn die Summe größer als der Schwellwert T war, ansonsten den Wert $e_\nu^{(1)*} = 0$. Der Korrekturwert $e_\nu^{(1)*}$ wird zu der r Takte vorher empfangenen Nachrichtenstelle $v_\nu^{(1)}$ hinzuaddiert und ergibt die decodierte Nachrichtenstelle $y_\nu^{(1)}$.



Figur 6.2: Schaltelement für SW-Decoder

Falls die Nachrichtenstelle $v_\nu^{(1)}$ bei der Übertragung gefälscht wurde und die Korrektur mit $e_\nu^{(1)*} = 1$ in erwünschter Weise erfolgte, stehen in den Registern für die Korrektur der folgenden Stellen aber Vergleichswerte $s_\nu^{(\mu)}$, die aus der gefälschten Nachrichtenstelle $v_\nu^{(1)}$ gewonnen wurden. Dieser Einfluß läßt sich in Ordnung bringen mittels der gestrichelten Rückkopplungsleitung in Figur 6.1, die bei $e_\nu^{(1)*} = 1$ wirksam wird.

Die Schaltung für den allgemeinen Fall m>1 kann aus Figur 6.1 durch Erweiterung entwickelt werden. Wir benötigen für jede der m Nachrichtenstellen einen Satz von J orthogonalen Prüfgleichungen und dementsprechend m Schwellwertelemente T.

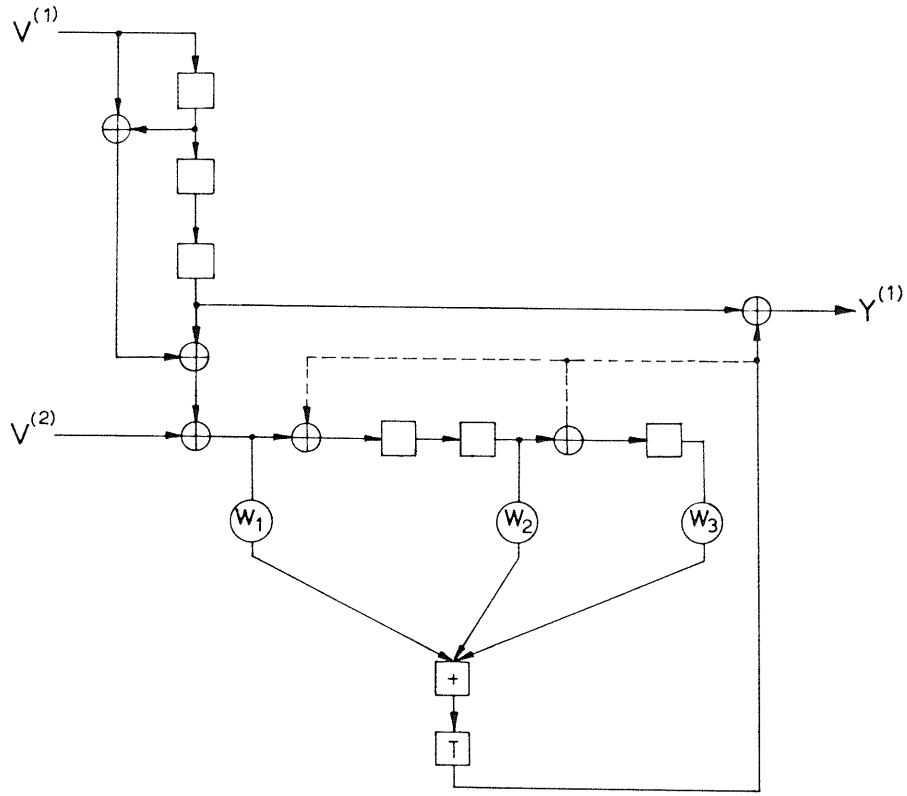
Der Aufwand der Decodieranlage wächst linear mit n_0: wir haben in Kapitel IV gesehen, daß der vorgeschaltete Coder r·m oder r·k Speicherplätze benötigt. Im eigentlichen Decoder werden k·r Speicher gebraucht.

In der Abhandlung von MASSEY werden einige Wege gezeigt, um zu orthogonalen einfachen oder zusammengesetzten Prüfgleichungen zu kommen. Besonders einsichtig sind dabei die selbstorthogonalen Code, d.h. Code, bei denen J der ursprünglichen Prüfgleichungen schon von selbst orthogonal sind. Beispiel 2): für R = 1/2 haben wir nur ein einziges Prüfmuster $G_1^{(2)}$ zu wählen. Die Koeffizienten des Generatormusters sind so zu wählen, daß alle Gleichungen unseres Prüfschemas, die Stelle 1 mitkontrollieren, auf $x_0^{(1)}$ orthogonal sind, d.h. wir dürfen nur dann $g_1^{(2)} = 1$ setzen, wenn dadurch die Orthogonalität auf Stelle 1 nicht gestört wird. Wir erhalten folgendes Schema für r=3:

$$P = \begin{pmatrix} 1\,0\,0\,0\,1\,0\,0\,0 \\ 1\,1\,0\,0\,0\,1\,0\,0 \\ 0\,1\,1\,0\,0\,0\,1\,0 \\ 1\,0\,1\,1\,0\,0\,0\,1 \end{pmatrix}$$

Dem entspricht das Prüfmuster $G_1^{(2)} = (1, 1, 0, 1)$. Die kennzeichnenden Größen dieses Codes sind: m=1, k=1, R=1/2, r=3.

Für Beispiel 2) ist die vollständige Decodieranlage in Figur 6.3 angegeben.



Figur 6.3: Vollständige SW-Decodieranlage für einen sequentiellen Code mit m=1, k=1, r=3 und damit n_0=8

Kapitel VII

RESTFEHLERWAHRSCHEINLICHKEITEN BEI BINÄRCODE

1. Einführung

Die letzten beiden Kapitel sollen sich mit den Restfehlerwahrscheinlichkeiten und den damit zusammenhängenden Fragen befassen. Wir haben die Restfehlerwahrscheinlichkeit kennengelernt als die Wahrscheinlichkeit dafür, daß eine Stelle falsch decodiert wird.

Es empfiehlt sich, diesen Begriff etwas zu modifizieren. Bei Blockcode ist es sinnvoller, die Wahrscheinlichkeit, daß ein übermitteltes Codewort - also ein Block von n Stellen - nicht korrekt decodiert wird, als Restfehlerwahrscheinlichkeit PR zu bezeichnen, während wir bei einem sequentiellen Code der Nachrichtenrate $R = \frac{m}{n}$ und der Codelänge n_0 die Restfehlerwahrscheinlichkeit PR_1 einführen wollen. Es soll die Wahrscheinlichkeit dafür sein, daß der erste Satz von Nachrichtenstellen $x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(m)}$ nicht richtig decodiert wird.

Bevor wir uns den Restfehlerwahrscheinlichkeiten der drei besprochenen Verfahren zuwenden, wollen wir uns die Decodiergenauigkeit bei allgemeiner Blockcodierung erarbeiten, um dann an ihr die Güte der neuen Verfahren messen zu können.

Wollen wir die Restfehlerwahrscheinlichkeit eines bestimmten Code bei gegebener Kanalkapazität C_0 und fester Nachrichtenrate R bestimmen, so stehen uns zwei Wege offen:
Einmal die mathematische Bestimmung von PR aufgrund der Struktur des verwendeten Code. Obwohl die Berechnung für einen speziellen Code im Prinzip ziemlich einfach ist, stößt sie doch in der Praxis - abgesehen von besonders einfachen Fällen - auf unüberwindliche Schwierigkeiten.

Der zweite Weg führt über die Simulation des Übertragungsprozesses auf einer Rechenanlage. Bei genügend großer Übertragungszeit strebt das Verhältnis der falsch übertragenen Stellen bzw. Blöcke zur Gesamtzahl der übertragenen gegen die Restfehlerwahrscheinlichkeit.

2. Restfehlerwahrscheinlichkeit eines Blockcode bei MRW-Decodierung

Die Restfehlerwahrscheinlichkeit PR hängt natürlich nicht nur vom benutzten Blockcode ab, sondern auch sehr wesentlich vom verwendeten Decodierverfahren. Da uns zunächst die größtmögliche Übertragungsgenauigkeit, die ein bestimmter Code zuläßt, interessieren soll, möge die Decodieranlage nach maximaler Rückschlußwahrscheinlichkeit arbeiten, also bei vorgegebenem Code das optimale Decodierverfahren benutzen. Wir hatten diese Decodierung in Kapitel II, 2 als MRW-Decodierung bezeichnet. Empfangen wir das n-Tupel $\bar{v}$, so entscheidet sich - wie wir in Kapitel II, 2 gezeigt haben - der MRW-Decoder für dasjenige Codewort $\bar{u}_\nu$, für das die Wahrscheinlichkeit $P(\bar{v}|\bar{u}_\nu)$ am größten ist.

Wir haben im zweiten Kapitel gesehen, daß die MRW-Decodierung einer Einteilung der 2^n Binärworte der Länge n in $M=2^m$ Familien F_ν entspricht. Ist das Empfangswort $\bar{v}$ ein Mitglied der Familie F_ν, so gilt: $P(\bar{v}|\bar{u}_\nu) \geq P(\bar{v}|\bar{u}_\mu)$ für alle Werte μ. Die Familieneinteilung umfaßt alle 2^n n-Tupel und jedes n-Tupel gehört in genau eine Familie. Hat ein n-Tupel $\bar{v}$, das nicht Codewort ist, von zwei Codeworten $\bar{u}_\nu$ und $\bar{u}_\mu$ die gleiche Distanz, was bedeutet, daß $P(\bar{v}|\bar{u}_\nu) = P(\bar{v}|\bar{u}_\mu)$ ist, dann wird $\bar{v}$ willkürlich einer der beiden Familien F_ν oder F_μ zugeordnet.

Untersuchen wir zunächst die Wahrscheinlichkeit für einen Decodierfehler unter der Voraussetzung, daß das Codewort $\bar{u}_\nu$ gesendet wurde, also die Wahrscheinlichkeit $P(\bar{v} \notin F_\nu|\bar{u}_\nu)$. Der Decoder arbeitet genau dann richtig, wenn der vom Kanal angelieferte Block $\bar{v}$ der Familie F_ν angehört. Also gilt für die bedingte Wahrscheinlichkeit, richtig zu decodieren:

$$1 - P(\bar{v} \notin F_\nu|\bar{u}_\nu) = P(\bar{v} \notin F_\nu|\bar{u}_\nu) = \sum_{\bar{v} \in F} P(\bar{v}|\bar{u}_\nu) \qquad (1)$$

Die Summation ist über alle diejenigen Binärfolgen $\bar{v}$ zu erstrecken, die der Familie F_ν angehören. Wollen wir die mittlere Decodiergenauigkeit eines Code - also den Wert (1-PR) - berechnen, dann müssen wir über alle Codeworte mitteln und erhalten:

$$1 - PR = \sum_{\nu=1}^{M} P(\bar{u}_\nu) \cdot \left[1 - P(\bar{v} \notin F_\nu | \bar{u}_\nu)\right] = \sum_{\nu=1}^{M} P(\bar{u}_\nu) \cdot P(\bar{v} \in F_\nu | \bar{u}_\nu) \tag{2}$$

Falls - wie wir in dieser Arbeit voraussetzen - alle Codeworte gleich wahrscheinlich sind, gilt $P(\bar{u}_\nu) = \frac{1}{M} = 2^{-m}$ für alle ν und damit

$$1 - PR = 2^{-m} \sum_{\nu=1}^{M} P(\bar{v} \in F_\nu | \bar{u}_\nu) = 2^{-m} \sum_{\nu=1}^{M} \sum_{\bar{v} \in F} P(\bar{v} | \bar{u}_\nu) \tag{3}$$

oder

$$PR = 1 - 2^{-m} \sum_{\nu=1}^{M} \sum_{\bar{v} \in F} P(\bar{v} | \bar{u}_\nu) \tag{4}$$

In der Doppelsumme $\sum_{\nu=1}^{M} \sum_{\bar{v} \in F}$ treten alle 2^n n-Tupel auf, weshalb wir auch schreiben können:

$$1 - PR = 2^{-m} \sum_{\text{alle } \bar{v}} P(\bar{v} | \bar{u}) \tag{5}$$

Dabei ist für ein bestimmtes $\bar{v}$ einzusetzen die Wahrscheinlichkeit dafür, daß das zu $\bar{v}$ gehörige Familiencodewort $\bar{u}$ durch Kanalstörung in dieses $\bar{v}$ übergeht. Haben $\bar{u}$ und $\bar{v}$ die Distanz f, so ist beim BSC

$$P(\bar{v} | \bar{u}) = p_0^f \, (1-p_0)^{n-f} \tag{6}$$

Da wir nach Kapitel I, 1 $p_0 < \frac{1}{2}$ voraussetzen können, nimmt $P(\bar{v}|\bar{u})$ monoton ab mit wachsendem f. Daraus folgt, daß ein Code dann eine große Übertragungssicherheit ermöglicht, wenn in der Summe $\sum_{\text{alle } \bar{v}} P(\bar{v}|\bar{u})$ viele Summanden mit kleinem f auftreten, d.h. möglichst viele n-Tupel von ihrem Familiencodewort kleine Distanz haben. Die Berechnung der Doppelsumme in Gleichung (4) bzw. (5) wird schon bei relativ kleinen n wegen des exponentiellen Anwachsens der Summandenzahl nicht mehr ausführbar. Deshalb müssen wir uns damit begnügen, eine untere Schranke PR_L und eine obere Schranke PR_u für die erreichbare Restfehlerwahrscheinlichkeit bei Blockcodierung herzuleiten. Die Indizes sind der amerikanischen Literatur entnommen; es sind die Anfangsbuchstaben von "lower" bzw. "upper". Die Herleitung dieser beiden Grenzen lehnt sich an Robert M. FANO's "Transmission of Information"[5] an.

3. Herleitung einer unteren Grenze PR_L für die Restfehlerwahrscheinlichkeit PR bei Blockcodierung

Aus Gleichung (5) und den anschließenden Bemerkungen folgt, daß bei MRW-Decodierung die Wahrscheinlichkeit für richtiges Decodieren bei fester Blocklänge n und fester Rate R dann ein Maximum erreicht, wenn die 2^m Familien F_ν möglichst gleich mächtig sind. Enthalten einzelne Familien besonders viele n-Tupel, dann geben die weitentfernten Familienmitglieder nur einen sehr geringen Beitrag zur Decodiergenauigkeit in Gleichung (5). Gleichmäßiger verteilt wird nach Gleichung (5) in Verbindung mit Gleichung (6) ein größerer Beitrag für korrektes Decodieren ermöglicht. Die n-Tupel einer Familie sollen sich bei gegebenen Codegrößen m und n innerhalb eines möglichst kleinen Distanzradius um ihr Familiencodewort scharen. MRW-Decodierung impliziert für einen vorgegebenen Code eine bestimmte Einteilung in M Familien F_ν. Die maximale Distanz, die bei dieser Familieneinteilung ein Familienmitglied von seinem Familiencodewort hat, bezeichnen wir als Codedistanz r des verwendeten Code.

Es gibt genau $\frac{n!}{f!\,(n-f)!} = \binom{n}{f}$ n-Tupel, die von einem festen n-stelligen Codewort die Distanz f haben. In einer Familie gibt es deshalb höchstens

$$\sum_{f=0}^{r} \binom{n}{f}$$

Plätze. Da alle 2^n n-Tupel in den Familien Platz haben, gilt für die Codedistanz r die folgende Ungleichung:

$$2^n \leq 2^m \cdot \sum_{f=0}^{r} \binom{n}{f} \quad \text{oder auch} \quad 2^{n-m} \leq \sum_{f=0}^{r} \binom{n}{f} \tag{7}$$

Für eine bestimmte Wahl von n und m kennzeichnen wir das nach Gleichung (7) kleinstmögliche r mit r_0; r_0 ist also definiert durch die folgende Ungleichung:

$$\sum_{f=0}^{r_0-1} \binom{n}{f} < 2^{n-m} \leq \sum_{f=0}^{r_0} \binom{n}{f} \tag{8}$$

Für vorgegebene Codegrößen m und n liegt durch Gleichung (8) die optimale Codedistanz r_0 fest. Mit der in Kapitel II, 2 eingeführten Hammingdistanz h gilt die Ungleichung

$$\left\lfloor \frac{h-1}{2} \right\rfloor \leq r_0 \leq r$$

Ein Code heißt perfekt oder auch dichtgepackt, wenn jede seiner 2^m Codewortfamilien alle n-Tupel enthält, die eine Distanz $f \leq r$ vom Familiencodewort haben; r stimmt dann automatisch mit r_0 überein und erfüllt Gleichung (7) exakt. Die Hammingdistanz h (vgl. Kapitel II, 2) eines perfekten Code ist $h = 2\,r_0 + 1$. Wir sehen aus Gleichung (7), daß perfekte Code nur für ganz bestimmte Werte von n existieren können, wenn wir von dem trivialen Fall absehen, daß wir als einzige Codeworte das Nullwort und das Einswort - oder zwei sonstige komplementäre n-Tupel - wählen. Denn ein solcher Code ist für alle ungeraden n perfekt und für alle geraden n quasi-perfekt. Quasi-perfekt heißt ein Code, wenn jede Codewortfamilie alle n-Tupel enthält, die eine Distanz $f < r$ vom Familiencodewort haben, wenn die Familien aber nicht alle oder zumindest nicht immer alle n-Tupel mit Distanz $f = r$ enthalten. Ein nur quasi-perfekter Code hat die Hammingdistanz $h = 2\,r_0$. Für perfekte Code gilt $r = r_0 = \frac{h-1}{2}$, für quasi-perfekte $r = r_0 = \frac{h}{2}$.

Unter einem optimalen Code für festgelegte Codegrößen m und n verstehen wir einen Code, von dem wir wissen, daß es für diese Codegrößen keinen besseren, d.h. bei MRW-Decodierung wirksameren Code gibt. Die perfekten und quasi-perfekten Code sind optimale Code.

Nehmen wir an, daß wir einen optimalen Code haben, dann stellt seine Decodiergenauigkeit sicher eine obere Schranke für die Decodiergenauigkeit eines jeden anderen Code dar, der dieselben Codegrößen m und n hat. Kennen wir die Codedistanz r eines Code, dann gilt nach Gleichung (7) die folgende Abschätzung:

$$1 - PR \leq 2^{-m} \sum_{f=0}^{r} 2^m \binom{n}{f} p_0^f (1-p_0)^{n-f} = \sum_{f=0}^{r} \binom{n}{f} p_0^f (1-p_0)^{n-f} = P(f \leq r) \tag{9}$$

$$1 - PR \leq P(f \leq r) \tag{10}$$

$$PR \geq 1 - P(f \leq r) = P(f>r) \tag{11}$$

Die Decodiergenauigkeit ist höchstens gleich der Wahrscheinlichkeit, daß der Übertragungskanal r oder weniger Fehler hervorruft.

Die Restfehlerwahrscheinlichkeit eines perfekten Code erfüllt Gleichung (10) sogar mit dem Gleichheitszeichen.

Als Ergebnis erhalten wir, daß die Restfehlerwahrscheinlichkeit eines jeden Blockcode mit den Codegrößen m und n der Gleichung (11) genügt:

$$PR \geq P(f>r) = \sum_{f=r+1}^{n} \binom{n}{f} p_0^f (1-p_0)^{n-f} \tag{12}$$

wobei r die Gleichung (7) erfüllt. Je kleiner r innerhalb dieser Bedingung gewählt wird, eine desto bessere untere Schranke erhalten wir; für $r = r_0$ die beste Schranke.

Zwei Schranken der kleinstmöglichen Codedistanz r_0 wollen wir uns überlegen, zunächst eine obere Schranke. Für den trivial optimalen Code mit zwei Codeworten, also m=1, gilt $r_0 \leq \frac{n}{2}$. Dieses Maximal kann r_0 nicht überschreiten, also $r_0 \leq \frac{n}{2}$. Eine untere Schranke für r_0 erhalten wir durch folgende Überlegung: Produziert der Kanal mehr als r_0 Fehler, so arbeitet der Decoder bei einem perfekten Code sicher falsch. Der Kanal verursacht aber durchschnittlich $n \cdot p_0$ Fehler pro Codewort, weshalb nur dann eine gute Übertragung möglich sein kann, wenn $r_0 > n \cdot p_0$ ist. Für r_0 muß also gelten $n \cdot p_0 < r_0 \leq \frac{n}{2}$ oder mit $\rho = \frac{r}{n}$, der relativen Codedistanz:

$$p_0 < \rho_0 \leq \frac{1}{2} \tag{13}$$

Führen wir in die Ungleichung (7) $2^{n-m} \leq \sum_{f=0}^{r} \binom{n}{f}$ die Nachrichtenrate R ein, so können wir auch sagen, daß r die Ungleichung:

$$R \geq 1 - \frac{1}{n} \operatorname{ld} \sum_{f=0}^{r} \binom{n}{f} \tag{14}$$

erfüllt.

Mit Hilfe der STIRLING'schen Formel läßt sich nun aus der in Gleichung (12) bewiesenen Schranke für PR folgender Satz herleiten, der hier in der Formulierung von FANO wiedergegeben ist:

Mit $p_0 < \rho = \frac{r}{n} \leq \frac{1}{2}$ gilt für die Restfehlerwahrscheinlichkeit PR

$$PR > PR_L = \frac{p_0(1-\rho)}{(1-p_0)(\rho+\frac{1}{n})} \frac{\exp-(\frac{1}{12\, n\rho(1-\rho)})}{\sqrt{2\,\pi\, n\, \rho(1-\rho)}}\, 2^{-n\,\alpha_L} \tag{15}$$

wenn ρ so gewählt ist, daß die Nachrichtenrate die Ungleichung

$$R \geq R_L = \frac{1}{n} \frac{\text{ld } e}{12\, n\, \rho(1-\rho)} + \text{ld } \sqrt{2\, \pi\, n\, \rho(1-\rho)} + C(\rho) \qquad (16)$$

erfüllt. Diese Ungleichung folgt aus Gleichung (14).
Dabei ist

$$\alpha_L = \rho \text{ ld } \frac{\rho}{p_0} + (1-\rho) \text{ ld } \frac{1-\rho}{1-p_0} \qquad (17)$$

und $C(\rho)$ wird definiert durch:

$$C(\rho) = 1 + \rho \text{ ld } \rho + (1-\rho) \text{ ld } (1-\rho) \qquad (18)$$

$C(\rho)$ ist also die Kanalkapazität eines Kanals der Fehlerwahrscheinlichkeit ρ (vgl. Kapitel I, 1).
Für große n vereinfachen sich die beiden Schranken PR_L und R_L; denn es gilt:

$$\lim_{n \to \infty} R_L = C(\rho) \qquad (19)$$

$$\lim_{n \to \infty} \left[- \frac{1}{n} \text{ ld } PR_L(e) \right. = \alpha_L \qquad (20)$$

Gleichung (20) bedeutet einfach, daß für große n das Verhalten der Schranke PR_L im wesentlichen bestimmt ist durch den Exponentialkoeffizienten α_L.

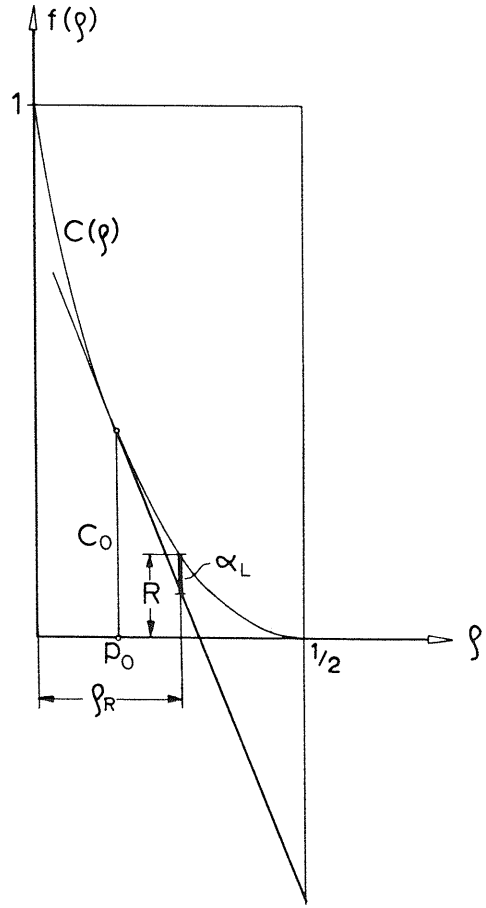
Dieser Exponentialkoeffizient α_L läßt eine einfache geometrische Deutung zu. Die Tangente an die Kurve $C(\rho)$ im Punkte p_0 sei mit $T_0(\rho)$ bezeichnet. Da $T_0(\rho) = 1 + \rho \text{ ld } p_0 + (1-\rho) \text{ ld } (1-p_0)$ (21) ist, gilt einfach:

$$\alpha_L = C(\rho) - T_0(\rho) \qquad (22)$$

Diese geometrische Deutung verdeutlicht Figur 7.1.

Wir erhalten den Exponentialfaktor α_L einer bestimmten Nachrichtenrate R, indem wir R zwischen der Kurve $C(\rho)$ und der ρ-Achse einpassen und an der so gefundenen Stelle ρ_R die Differenz von $C(\rho)$ und $T_0(\rho)$ feststellen.

Wir sehen, daß α_L für $\rho = p_0$ Null ist und für $\rho > p_0$ von Null



Figur 7.1: Graphische Deutung des Exponentialfaktors α_L

ausgehend mit ρ wächst. Dies bedeutet: für eine gegebene Fehlerwahrscheinlichkeit p_0 des Übertragungskanals mit der zugehörigen Kanalkapazität C_0 wird der Exponentialfaktor α_L um so größer, je kleiner die Übertragungsrate R gewählt wird. Mit wachsendem α_L aber verkleinert sich exponentiell nach Gleichung (15) die untere Grenze der Restfehlerwahrscheinlichkeit PR_L und die Übertragung kann zunehmend sicherer werden.

Bei der graphischen Ermittlung von α_L wurden R und R_L gleichgesetzt. Für große n ist dies erlaubt und bedeutet, daß wir für die untere Grenze PR_L einen möglichst großen Wert erhalten. Eine gute untere Schranke soll aber möglichst groß sein. Die Restfehlerwahrscheinlichkeit eines Blockcode genügt der Ungleichung:

$$PR > PR_L = K \cdot 2^{-n \cdot \alpha_L} \qquad (23)$$

K ist eine langsam veränderliche Funktion von n, R und p_0 und α_L hängt von diesen Größen gemäß Figur 7.1 ab.

Die nächsten Überlegungen heben darauf ab, eine untere Schranke für die Decodiergenauigkeit bzw. eine obere Schranke für die Restfehlerwahrscheinlichkeit PR_u (Index von englisch "upper") zu finden.

4. Herleitung einer oberen Grenze PR_u für die Restfehlerwahrscheinlichkeit PR bei Blockcodierung

Wir untersuchen dazu Ensembles von Zufallscode . Ein Zufallscode entsteht dadurch, daß wir die den Code kennzeichnenden Binärzahlen zufällig wählen, wobei $P(0) = P(1) = \frac{1}{2}$ sein soll. Für den allgemeinen Blockcode bedeutet dies, daß wir die $M = 2^m$ n-Tupel, die den M Nachrichten als Codeworte zugeordnet sind, zufällig bestimmen. Die Wahrscheinlichkeit, daß einer Nachricht ein bestimmtes Codewort zugeordnet wird, ist $(\frac{1}{2})^n$ und die Wahrscheinlichkeit für einen ganz bestimmten Code deshalb $(\frac{1}{2^n})^M$, wobei jeder einzelne

Code gleich wahrscheinlich ist. In diesem Ensemble kommen natürlich auch alle schlechten Code vor, wie z.B. derjenige Code, bei dem alle Codeworte nur als Nullen bestehen. Wir wollen nun die durchschnittliche Fehlerwahrscheinlichkeit $\overline{PR}$ dieses Ensembles berechnen. $\overline{PR}$ wird eine obere Grenze PR_u für PR sein, denn wir müssen von einem gute Code erwarten, daß er zumindest die Durchschnittsdecodiergenauigkeit für Zufallscodierung erreicht.

Wir legen wieder Decodierung nach maximaler Rückschlußwahrscheinlichkeit zugrunde, so daß gilt

$$P\{\bar{v} \text{ wird richtig decodiert}\} \geq \sum_{f=0}^{n} P\{f \text{ Fehler passieren}\} \cdot P \qquad (24)$$
$$\{\text{alle nicht gesendeten Codeworte unterscheiden sich von } \bar{v} \text{ in mehr als f Stellen.}\}$$

Das Größer-Zeichen bezieht sich darauf, daß wir annehmen, immer dann eine Fehlentscheidung zu treffen, wenn $\bar{v}$ von zwei Codeworten denselben Abstand hat.

Die erste Wahrscheinlichkeit hängt nur von den Eigenschaften des Übertragungskanals ab:

$$P\{f \text{ Fehler passieren}\} = \binom{n}{f} p_0^{\,f} (1-p_0)^{n-f} \qquad (25)$$

Die zweite Wahrscheinlichkeit ist bestimmt durch die Codestruktur. Für einen Zufallscode ist die Wahrscheinlichkeit, daß ein bestimmtes Codewort sich von $\bar{v}$ in mehr als f Stellen unterscheidet, gleich

$$(\tfrac{1}{2})^n \sum_{i=f+1}^{n} \binom{n}{i} = 2^{-n} \left[\sum_{i=0}^{n} \binom{n}{i} - \sum_{i=0}^{f} \right] = 1 - 2^{-n} \sum_{i=0}^{f} \binom{n}{i} \qquad (26)$$

Da die Wahl der Codeworte unabhängig voneinander erfolgte, erhalten wir die Wahrscheinlichkeit, daß alle M-1 nicht gesendeten Codeworte sich von $\bar{v}$ in mehr als f Stellen unterscheiden zu

$$1 - 2^{-n} \sum_{i=0}^{f} \binom{n}{i}^{M-1} \qquad (27)$$

Setzen wir diese Ergebnisse in Gleichung (24) ein, so erhalten wir: die durchschnittliche Wahrscheinlichkeit für richtiges Decodieren ($1-\overline{PR}$) erfüllt die Ungleichung

$$1 - \overline{PR} \geq \sum_{f=0}^{n} \left\{ \binom{n}{f} p_0^{\,f} (1-p_0)^{n-f} \left[1 - 2^{-n} \sum_{i=0}^{f} \binom{n}{i} \right]^{M-1} \right\} \tag{28}$$

und wegen

$$\sum_{f=0}^{n} \binom{n}{f} p_0^{\,f} (1-p_0)^{n-f} = 1 \tag{29}$$

gilt:

$$\overline{PR} \leq \sum_{f=0}^{n} \binom{n}{f} p_0^{\,f} (1-p_0)^{n-f} \left\{ 1 - \left[1 - 2^{-n} \sum_{i=0}^{f} \binom{n}{i} \right]^{M-1} \right. \tag{30}$$

Mit Hilfe eines von FANO[5] ausführlich behandelten Verfahrens der "Verschiebung von Wahrscheinlichkeitsverteilungen" und unter Benutzung der STIRLING'schen Formel läßt sich die erhaltene Grenze für $\overline{PR}$ in eine Form bringen, die - besonders für große n - übersichtlicher ist. Bevor dieser Satz - wiederum in der Formulierung von FANO - zitiert wird, sei noch die kritische Rate R_c mit ihrem Parameter ρ_c eingeführt:

$$R_c = C(\rho_c) \text{ mit} \tag{31}$$

$$\rho_c = \frac{\sqrt{p_0}}{\sqrt{p_0} + \sqrt{1-p_0}} \tag{32}$$

Damit lautet der Satz: Falls $0 < R < C_0$ ist, dann genügt die mittlere Fehlerwahrscheinlichkeit für Zufallsblockcodierung der Gleichung

$$\overline{PR} < K \cdot 2^{-n \cdot \alpha_u} \tag{33}$$

Bei der Festsetzung der Konstanten haben wir zwei Bereiche zu unterscheiden:

a) $C_0 > R > R_c$

$$K = \frac{p_0}{\sqrt{2\pi n}} \sqrt{\frac{1-\rho}{\rho}} \left[\frac{1}{\rho - p_0} + \frac{1}{\sqrt{2\pi n}} \sqrt{\frac{1-\rho}{\rho}} \frac{1-\rho}{(1-2\rho)\, p_0\, (1-\rho)^2 - \rho^2\, (1-p_0)} \right] \tag{34}$$

$$\alpha_u = \rho \operatorname{ld} \frac{\rho}{p_0} + (1-\rho) \operatorname{ld} \frac{1-\rho}{1-p_0} \tag{35}$$

In diesem Bereich wird ρ bestimmt aus der Gleichung

$$R = 1 + \rho \operatorname{ld} \rho + (1-\rho) \operatorname{ld} (1-\rho) = C(\rho) \tag{36}$$

b) $R_c \geq R > 0$

$$K = 1 \tag{37}$$

$$\alpha_u = 1 - 2 \operatorname{ld} (\sqrt{p_0} + \sqrt{1-p_0}) - R = \rho \operatorname{ld} \frac{\rho_c}{p_0} + (1-\rho) \operatorname{ld} \frac{1-\rho_c}{1-p_0} \tag{38}$$

wobei ρ in diesem Bereich gegeben ist durch die Gleichung

$$R = 1 + \rho \operatorname{ld} \rho_c + (1-\rho) \operatorname{ld} (1-\rho_c) \tag{39}$$

Bevor wir uns nach Figur 7.2 den Exponentialfaktor geometrisch verdeutlichen, wollen wir die Größe der kritischen Rate R_c untersuchen: die Steigung der Kurve $C(\rho)$ im Punkte ρ ist gegeben durch

$$t(\rho) = \operatorname{ld} \frac{\rho}{1-\rho} \tag{40}$$

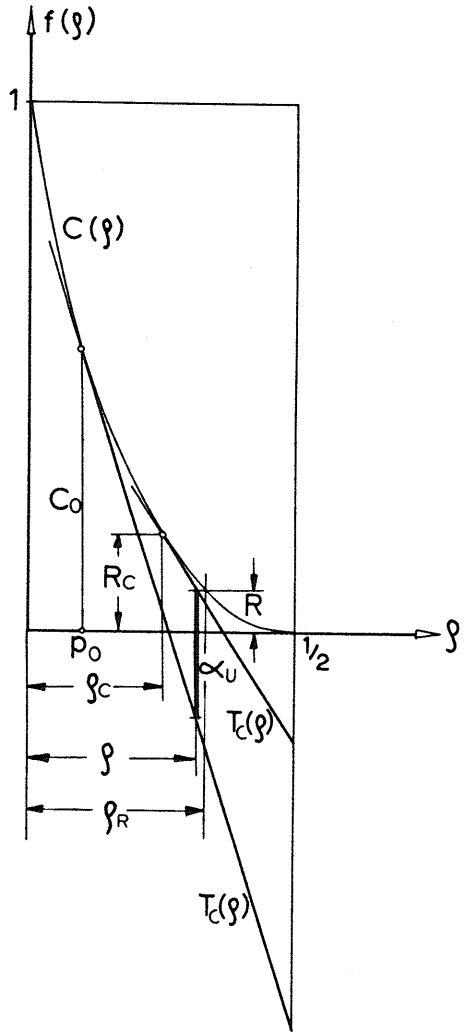
Da nach der Definitionsgleichung für ρ_c gilt:

$$\left(\frac{\rho_c}{1-\rho_c}\right) = \sqrt{\frac{p_0}{1-p_0}} \tag{41}$$

haben wir:

$$t(\rho_c) = \operatorname{ld} \frac{\rho_c}{1-\rho_c} = \frac{1}{2} \operatorname{ld} \frac{p_0}{1-p_0} = \frac{1}{2}\, t(p_0) \tag{42}$$

ρ_c ist also gerade derjenige Punkt der Kurve $C(\rho)$, dessen Steigung halb so groß ist wie die Steigung im Punkte p_0; den Wert $C(\rho_c)$ hatten wir R_c genannt. Die Tangente in diesem Punkt bezeichnen wir mit $T_c(\rho)$. Nach Gleichung (36) bzw. (39) lautet der



Figur 7.2: Graphische Deutung des Exponentialfaktors α_U

Zusammenhang zwischen Nachrichtenrate R und Parameter ρ in den beiden Bereichen:

$$a)\ C_0 > R > R_c \qquad R = C(\rho) \tag{43}$$

$$b)\ R_c > R > 0 \qquad R = T_c(\rho) \tag{44}$$

Den Exponentialfaktor α_u erhalten wir nun einfach als Differenz:

$$\alpha_u = R - T_0(\rho) \tag{45}$$

Wir finden den Wert α_u zu einer bestimmten Nachrichtenrate R, indem wir in Figur 7.2 R einpassen zwischen der ρ-Achse und der Funktion $C(\rho)$, falls $C_0 > R > R_c$ bzw. zwischen ρ-Achse und Tangente $T_c(\rho)$, falls $0 \leq R \leq R_c$. An der gefundenen Stelle ρ lesen wir nun den Wert α_u ab als Differenz von $C(\rho)$ und $T_0(\rho)$, falls $C_0 > R > R_c$ bzw. als Differenz zwischen $T_c(\rho)$ und $T_0(\rho)$, falls $0 \leq R \leq R_c$.

Wir sehen, daß α_u für $\rho > p_0$ positiv ist, daß also für jede Nachrichtenrate $R < C_0$ die mittlere Fehlerwahrscheinlichkeit $\overline{PR}$ beliebig klein wird, wenn nur n hinreichend groß gewählt ist. Der Exponentialfaktor α_u stimmt im Bereich $C_0 < R \leq R_c$ sogar überein mit dem Exponentialfaktor α_L der unteren Grenze von PR_L, wenn wir große n voraussetzen; denn für große n strebt R_L gegen $C(\rho)$. Die Übereinstimmung der Exponentialfaktoren α_u und α_L im Bereich $C_0 < R \leq R_c$ bedeutet soviel, daß man bei großer Blocklänge n ruhig einen Code zufällig auswählen kann. Stellt sich heraus, daß der gewählte Code schlecht ist, dann verwirft man ihn und wählt ein zweites Mal aus. Die Wahrscheinlichkeit, einen Code zu wählen mit $P(PR > A \cdot \overline{PR})$ ist nämlich $= \frac{1}{A}$ (MARKOFF'sche Ungleichung).

5. Sequentielle Decodierung

Die Restfehlerwahrscheinlichkeit PR_1 wird definiert als die Wahrscheinlichkeit, daß dem Decoder bei der Korrektur des ersten Nachrichtensatzes - aufgrund der ersten n_0 empfangenen Stellen - ein Fehler unterläuft. PR_1 genügt denselben Abschätzungen wie PR bei Blockcodierung, wenn in diesen n durch n_0 ersetzt wird.

Wir erreichen also bei gleicher Rate $R = \frac{m}{n}$ eine wesentlich kleinere Restfehlerwahrscheinlichkeit als bei Blockcodierung. Nahm dort PR exponentiell mit n ab, so nimmt bei sequentieller Decodierung PR_1 exponentiell mit $n_0 = (r+1) \cdot n$ ab.

In Kauf nehmen müssen wir dabei, daß der Empfänger länger warten muß, bis er eine Nachrichtenstelle vom Decoder erhält. Dieser kann eine Stelle v erst dann decodieren, wenn alle n_0 mit v zusammenhängenden Ziffern vorliegen.

Der zweite Nachteil sequentieller Decodierverfahren ist ihre Neigung zur Fehlerfortpflanzung. Ein einmal passierter Decodierfehler beeinflußt - wie wir in Kapitel IV,3 gesehen haben - auch die folgenden Arbeitstakte und kann dadurch Folgefehler hervorrufen. Diese Folgefehler bewirken, daß die Restfehlerwahrscheinlichkeiten PR_ν der ν-ten Gruppe von Nachrichtenstellen größer ist als PR_1.

6. Distanzfunktion für das Ensemble der Prüfschemacode

Eine zweite Vergleichsmöglichkeit bietet die Untersuchung der Distanzfunktion für ein Codeensemble. Die Distanzfunktion N(f) für das Codewort $\bar{u}$ gibt für jeden Wert f die Anzahl der Codeworte an, die von $\bar{u}$ die Distanz f haben. Bei Gruppencode - also erst recht bei Prüfschemacode - ist N(f) für alle Codeworte gleich. GALLAGER[3] zeigt in seiner Arbeit, daß für diese Distanzfunktion gemittelt über das Ensemble der Prüfschemacode gilt:

$$\bar{N}(f) = \frac{2^{n\,[R-C(\rho)\,]}}{\sqrt{2\pi n\rho(1-\rho)}} \quad \text{mit} \quad \rho = \frac{f}{n},\ f > 0 \qquad (46)$$

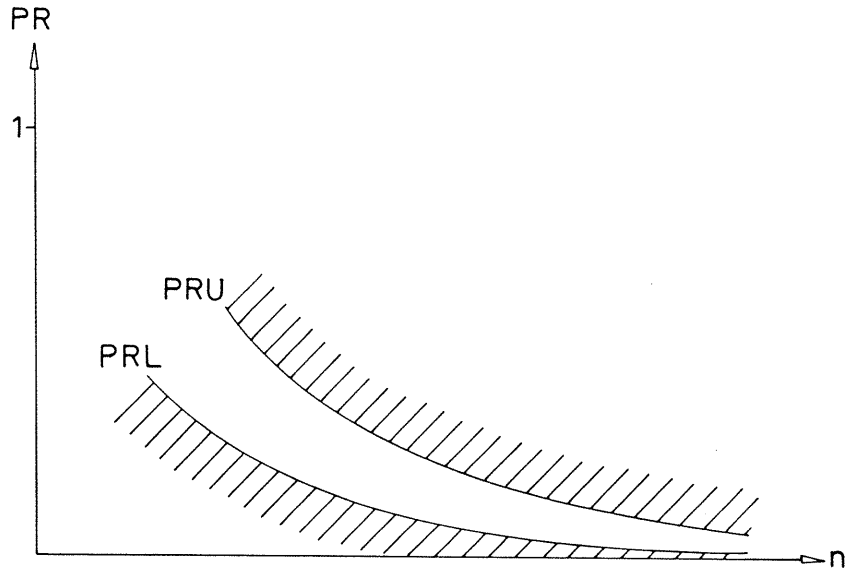
d.h. die Wahrscheinlichkeit für ein Codewort, das ein Gewicht f mit $0 < f < n \cdot \rho_R$ hat, geht mit wachsendem n gegen 0, wenn ρ_R definiert wird durch $R = C(\rho_R)$. Das bedeutet, daß für genügend großes n "fast alle" Prüfschemacode eine Hammingdistanz $h > n \cdot \rho_R$ haben. Exakt gilt für die Wahrscheinlichkeit, daß die

Hammingdistanz h kleiner als $n \cdot \rho_R$ ist:

$$P(h < n\,\rho_R) = \frac{1}{1-2\rho_R} \sqrt{\frac{1-\rho_R}{2\pi n \rho_R}}\; 2^{n \cdot [R\ -\ C(\rho)]} \qquad (47)$$

7. Bemerkungen zur Untersuchung spezieller Codeensemble und Decodierverfahren

Wir haben in Abschnitt 3 und 4 zwei Schranken für PR gefunden, die von der Codelänge n abhängen. Diese Abhängigkeit von n deutet Figur 7.3 an



Figur 7.3: Obere und untere Grenze für die Restfehlerwahrscheinlichkeit PR in Abhängigkeit von der Blocklänge n

Wir wissen, daß jede Restfehlerwahrscheinlichkeit $PR > PR_L$ ist. Wollen wir ein spezielles Codeensemble auf seine Decodierwirksamkeit untersuchen, so darf die durchschnittliche Restfehlerwahrscheinlichkeit dieses Ensemble bei MRW-Decodierung höchstens gleich PR_u sein, denn sonst enthält das zu untersuchende Ensemble

sehr viele "schlechte" Code.

Wenn wir nun eine Aussage über ein gewisses Codeensemble mit einem speziellen Decodierverfahren machen wollen, dann müssen wir zweierlei untersuchen: einmal das verwendete Codeensemble, dann das verwendete Decodierverfahren. Es ist zunächst festzustellen, ob wir ein "gutes" oder zumindest - gemessen an den Zufallscode - ein durchschnittliches Ensemble vor uns haben. Eine Aussage darüber macht die mittlere Restfehlerwahrscheinlichkeit dieses Ensemble , die errechnet wird unter der Voraussetzung, daß MRW-Decodierung verwendet wird.

Bei MRW-Decodierung steigt der Decodieraufwand exponentiell mit n an, was für große n untragbar ist. Andererseits wird man sich oft für große n entscheiden müssen, wenn große Decodiergenauigkeit verlangt ist.

Jedes Decodierverfahren, das auch für große n anwendbar sein soll, muß einen Kompromiß schließen: es verzichtet auf die maximal erreichbare Decodiergenauigkeit, die der verwendete Code eigentlich zuließe, arbeitet dafür aber mit erträglichem Decodieraufwand. Soll eine höhere Decodiergenauigkeit erreicht werden, so bleibt immer noch der Weg, eine niedrigere Nachrichtenrate R zu verwenden oder die Codelänge n zu erhöhen. Diese Nachteile im Vergleich zur optimalen Decodierung werden aber bei weitem aufgewogen durch erträglichen Decodieraufwand.

8. Algebraische Decodierverfahren - Statistische Decodierverfahren

Wir haben gesehen, daß perfekte und quasi-perfekte Code so aufgebaut sind, daß sie bei Decodierung nach maximaler Rückschlußwahrscheinlichkeit eine Familieneinteilung implizieren, bei der die Familienmitglieder vom Familiencodewort eine Distanz $f \leq r_0$ haben und r_0 die kleinste Zahl r ist, die die Gleichung (7) noch erfüllt. Die Hammingdistanz ist $h = 2\,r_0 + 1$, falls er perfekt ist, oder $h = 2\,r_0$ falls er quasi-perfekt ist.

Statistische Verfahren sind nicht nur für den BSC, den symmetrischen Binärkanal, sondern auch speziell für solche Übertragungskanäle geeignet, deren Schrittfehlerwahrscheinlichkeit zwar nicht konstant ist, deren jeweilige Größe aber beim Empfänger bekannt ist. Sie nützen die am Empfänger vorliegende Information im Falle nicht optimaler Code besser aus als rein algebraische Verfahren und treffen über eine zu decodierende Stelle immer, d.h. im Gegensatz zu algebraischen Verfahren auch, falls $f \geq \left\lfloor \frac{h-1}{2} \right\rfloor$ ist, eine gezielte Entscheidung.

Kapitel VIII

DIE RESTFEHLERWAHRSCHEINLICHKEIT DER DREI STATISTISCHEN VERFAHREN

1. LD-Code nach R.G. GALLAGER

R.G. GALLAGER hat in seiner Arbeit gezeigt, daß das Ensemble der LD-Code mindestens so gute Decodierergebnisse erreicht wie ein Optimalcode einer etwas höheren Nachrichtenrate. Wir wissen aus Kapitel VII,4 , daß die durchschnittliche Restfehlerwahrscheinlichkeit $\overline{PR}$ für die Gesamtheit aller Gruppencode bei fester Kanalkapazität C_0 und fester Nachrichtenrate R mit wachsender Blocklänge n beliebig klein wird, falls nur $R < C_0$ ist. Die (n, j, 1)-Code genügen dieser Forderung nicht; es muß vielmehr $R < R_1$ sein, um exponentielle Abnahme der durchschnittlichen Restfehlerwahrscheinlichkeit mit n zu garantieren. Definieren wir ρ_1 durch

$$\rho_1 = \frac{1 + (1 - 2\,p_0)^1}{2} \tag{1}$$

dann ist $R_1 = 1 - \frac{H(p_0)}{H(\rho_1)}$ (2)

Dabei ist $H(\rho)$ die Entropie. Sie hängt mit der Kanalkapazität $C(\rho)$ zusammen:

$$H(\rho) + C(\rho) = 1 \tag{3}$$

Ersetzen wir in Gleichung (2) die Entropie H durch die Kapazität C, so können wir aus

$$R_1 = C_0 - \frac{C_1(1-C_0)}{1-C_1} \tag{4}$$

ablesen, daß R_1 kleiner als C_0 ist. Es ist $R_1 = 0$; R_1 wächst mit l. Für großes l nähert sich R_1 der Kanalkapazität C_0:

$$\lim_{l \to \infty} R_1 = C_0 \tag{5}$$

Je größer l, desto unwesentlicher wird also die Begrenzung für die Nachrichtenrate R.

Eine obere Schranke der Restfehlerwahrscheinlichkeit für das endgültige Verfahren von GALLAGER liefert bestimmt die Restfehlerwahrscheinlichkeit für das zweite Decodierverfahren von GALLAGER. Beide Verfahren decodieren nämlich eine bestimmte Stelle nur aufgrund derjenigen empfangenen Binärzahlen, die im Codebaum dieser Stelle auftauchen. Das dritte Verfahren fällt dabei immer diejenige Entscheidung, die aufgrund der erhaltenen Information die wahrscheinlichste ist; also muß gelten: P (Stelle wird falsch decodiert bei 2. Verfahren) ≥ P(Stelle wird falsch decodiert bei 3. Verfahren).

Die Restfehlerwahrscheinlichkeit beider Verfahren hängt wesentlich ab von der Zahl t der voneinander unabhängigen Iterationsschritte, die ein LD-Code zuläßt. GALLAGER gibt in seiner Arbeit ein Konstruktionsprinzip für LD-Code an, das eine Mindestzahl von unabhängigen Codebaumstufen garantiert. Daraus resultiert die linke Seite der Abschätzung (6) für t. Die rechte Seite ergibt sich auf direktem Wege aus der in Kapitel III, 3 hergeleiteten oberen Schranke für t. Die Ungleichung für t lautet:

$$\frac{\ln\left(\frac{n}{2l} - \frac{n}{2\,j(l-1)}\right)}{2\,\ln(l-1)(j-1)} \leq t \leq \frac{\ln n}{\ln\,(j-1)(l-1)} \tag{6}$$

Es ist n die Blocklänge, l die Zahl der "1" in einer Zeile des Prüfschemas und j die Zahl der "1" in einer Spalte des Prüfschemas. GALLAGER's zweites Decodierverfahren funktioniert - wie besprochen - so, daß im ν-ten Iterationsschritt dann eine Stelle umgedreht wird, wenn b_ν oder mehr Prüfgleichungen, die von dieser Stelle im Codebaum nach oben ausgehen, nicht erfüllt sind. Wir bezeichnen die durchschnittliche Restfehlerwahrscheinlichkeit für eine Stelle nach dem ν-ten Iterationsschritt mit p_ν ; es ist p_0 gerade die Schrittfehlerwahrscheinlichkeit des Kanals. Für p_ν hat GALLAGER folgende Rekursionsformel hergeleitet:

$$p_{\nu+1} = p_0 - \sum_{i=b}^{j-1} \binom{j-1}{i} \left\{ p_0 \left[\frac{1 + (1 - 2\,p)^{l-1}}{2}\right]^i \left[\frac{1 - (1 - 2\,p)^{l-1}}{2}\right]^{j-1-i} - (1 - p_0)\left[\frac{1 - (1-2\,p)^{l-1}}{2}\right]^i \left[\frac{1 + (1-2\,p)^{l-1}}{2}\right]^{j-1-i} \right\} \tag{7}$$

Aus Gleichung (7) läßt sich direkt ableiten, daß diese Restfehlerwahrscheinlichkeit minimisiert werden kann, wenn wir als b_ν die kleinsten Zahlen wählen, die für $\nu = 0, 1, \ldots, l$ gerade noch die Ungleichungen

$$\left[\frac{1 + (1 - 2\,p_\nu)^{l-1}}{1 - (1 - 2\,p_\nu)^{l-1}}\right]^{2\,b_\nu} \geq \left[\frac{1 - p_0}{p_0}\right]^{j-1} \qquad (8)$$

erfüllen.

Nehmen wir an, daß wir einen LD-Code vor uns haben, der t voneinander unabhängige Iterationsschritte zuläßt, wobei t der Ungleichung (6) genüge. Ferner seien die b_ν auf die eben erläuterte Weise bestimmt. Unter diesen Voraussetzungen hat GALLAGER aus seiner Rekursionsformel folgende Grenze für p_t, die Restfehlerwahrscheinlichkeit nach dem letzten, dem t-ten Iterationsschritt des zweiten Verfahrens, hergeleitet:

$$PR = p_t = \exp\left\{-c_{jl} \cdot n^\alpha \left[\frac{1}{2\,l} - \frac{1}{2\,j(1-1)}\right]^\alpha\right\} \qquad (9)$$

c_{jl} ist eine von j und l abhängige Funktion und die Größe α wird definiert durch

$$\alpha = \frac{\ln\left\lfloor\frac{j}{2}\right\rfloor}{2 \ln (j-1)(1-1)} \qquad (10)$$

wobei $\left\lfloor\frac{j}{2}\right\rfloor$ den abgerundeten Wert $\frac{j}{2}$ darstellt. Wir sehen, daß für $j \geq 3$ $\alpha > 0$ ist; immer gilt $\alpha < 1$. GALLAGER vermutet, daß bei Anwendung des dritten Verfahrens ein exponentielles Abnehmen der Restfehlerwahrscheinlichkeit mit n dann stattfindet, wenn wir die Iteration fortsetzen, auch dann, wenn die Unabhängigkeit nicht mehr gewahrt sein sollte.

2. WR-Decodierung nach WOZENCRAFT-REIFFEN

Sowohl MASSEY als auch WOZENCRAFT und REIFFEN beschränken sich in ihren genannten Arbeiten auf Untersuchung der Restfehlerwahrscheinlichkeiten für den Spezialfall $R = \frac{1}{n}$.

Perfekte oder zumindest quasi-perfekte Code sind konstruierbar für m=1 (trivial) und für die Korrektur eines Fehlers (Hamming-Code). Es ist leider kein Weg bekannt, der die Konstruktion perfekter oder zumindest optimaler Code für große n ermöglicht - von diesen einfachen Fällen abgesehen. Bis heute sind nur sehr wenige optimale Code bekannt. Sie sind aufgeführt in PETERSON's "Error-Correcting Codes"[6]; n=25 ist die größte auftretende Blocklänge und die Nachrichtenraten liegen fast alle in der Nähe von $\frac{1}{2}$.

Es ist noch nicht einmal allgemein festzustellen, ob ein perfekter oder quasi-perfekter Code für eine bestimmte Wahl von Nachrichtenstellenzahl m und Blocklänge n überhaupt existiert. Alle bisher bekannten Decodierverfahren sind algebraisch. Sie bauen sich auf einer algebraischen Struktur des verwendeten Code auf. Ihre Fehlerkorrektureigenschaften hängen allein von der garantierten Hammingdistanz h_g des verwendeten Code ab, wobei z.B. für BOSE-CHANDHURI-Code die wirkliche Hammingdistanz h größer sein kann als die garantierte. Sie korrigieren alle Fehlermuster mit Fehlerzahl $f = \left\lfloor\frac{h_g-1}{2}\right\rfloor$; $\left\lfloor\frac{h_g-1}{2}\right\rfloor$ ist der abgerundete Wert $\frac{h_g-1}{2}$. Werden bei einem optimalen Code alle Fehlermuster der Fehlerzahl $f \leq \left\lfloor\frac{h-1}{2}\right\rfloor$ korrigiert, dann kann es kein besseres als ein algebraisches Verfahren geben. Da uns aber so wenige optimale Code bekannt sind, bleibt uns besonders für große n nichts anderes übrig, als mit Code zu arbeiten, die sich ganz beträchtlich von optimalen Code unterscheiden können. Bei solchen Code kann ein algebraisches Verfahren völlig versagen, obwohl die Restfehlerwahrscheinlichkeit des Code bei MRW-Decodierung befriedigend gut wäre.

Nehmen wir z.B. an, daß zwar einige wenige Codeworte eine kleine Distanz h untereinander aufweisen, während fast alle Codeworte eine viel größere Distanz $D \gg h$ besitzen, dann ist für einen solchen Code ein algebraisches Verfahren ungeeignet, denn es korrigiert nur eine Zahl f von Fehlern mit $f \leq \frac{h-1}{2}$. Statistische Verfahren liefern hier bessere Ergebnisse.

Ihren Namen haben statistische Verfahren von ihrer Eigenschaft, die Statistik des Übertragungskanals zu berücksichtigen. Die drei besprochenen Verfahren sind statistisch.

Bei WR-Decodierung, also dem von WOZENCRAFT und REIFFEN entwickelten Verfahren, ist die Nachrichtenrate R einer weitergehenden Einschränkung unterworfen als $R < C_0$. Nur dann erhält man mit WR-Decodierung gute Übertragungsergebnisse, d.h. exponentielles Abnehmen der Restfehlerwahrscheinlichkeit mit der Codelänge n_0, wenn für die Nachrichtenrate R gilt: $R < R_b$. R_b ist bestimmt durch die Störwahrscheinlichkeit p_0; es gilt:

$$R_b = 1 - ld\left[1 + 2\sqrt{p_0(1-p_0)}\right] \qquad (11)$$

R_b können wir in die uns bekannten Größen umrechnen: es ist

$$R_b = 2\,R_c - T_0(\rho_c) = 2\,C(p_0) - T_0(\rho_c) \qquad (12)$$

dabei war ρ_c gerade jene Abszisse, für die die Steigung der C-Kurve halb so groß ist wie für die Abszisse p_0. Es ist $R_c = C(\rho_c)$. T_0 ist die Tangente an $C(\rho)$ im Punkte p_0. R_b erfüllt die Ungleichung: $C_0 < R_b < R_c$. Das Verhältnis von R zu R_b wollen wir mit B bezeichnen:

$$B = \frac{R}{R_b} \qquad (13)$$

die Bedingung $R < R_c$ bedeutet also: $B < 1$. Für das Verhältnis von R_b zu C_0 gelten die beiden Grenzwerte

$$\lim_{p_0 \to \frac{1}{2}} \frac{R_b}{C_0} = \frac{1}{2} \quad \text{und} \quad \lim_{p_0 \to 0} \frac{R_b}{C_0} = 1 \qquad (14)$$

also gilt im Bereich $0 < p_0 < \frac{1}{2}$ die Ungleichung

$$\frac{1}{2}\,C_0 < R_b < C_0$$

Die Zahl Z der Rechenschritte zur Decodierung eines Satzes von n Codewortstellen ist beim WR-Verfahren eine Zufallsvariable, deren Erwartungswert insbesondere von der Wahl der Ausscheidekriterien A_j abhängt. Die geringste mittlere Zahl von Rechenschritten erhält man für die folgende Wahl der A_j:

$$A_j = ld\ n_0 + j \cdot \Delta A \text{ mit } \Delta A = \frac{ld\ B}{B - 1} \qquad (15)$$

Unter diesen Bedingungen gilt, wie WOZENCRAFT und REIFFEN gezeigt haben, daß die durchschnittliche Zahl von Rechenschritten zur Decodierung der ersten Nachrichtenstelle die Ungleichung

$$\bar{Z} < \frac{K \cdot e}{(1-B)^2}\, n_0^{\,B} \left[\frac{1}{1 - 2^{-R}} + \frac{1}{1 - 2^{-R_b}}\right] \qquad (16)$$

erfüllt, falls $B<1$ ist. Die mittlere Zahl von Rechenschritten wächst also weniger als linear mit der Codelänge n_0. K ist die in Kapitel V, 4 eingeführte Konstante und e die Basis der natürlichen Logarithmen. Unter denselben Voraussetzungen für die Kriterien A_j und die Nachrichtenrate R haben WOZENCRAFT und REIFFEN für die Restfehlerwahrscheinlichkeit PR_1 die folgende Grenze hergeleitet:

$$PR_1 = \frac{K\, n_0^{\,2}}{B^{\frac{1}{1-B}}}\; 2^{-n_0 \cdot \alpha_u} \qquad (17)$$

wenn $B<1$ ist. α_u ist der in Kapitel VII eingeführte Exponentialkoeffizient α_u der oberen Grenze PR_u für die Restfehlerwahrscheinlichkeit. Falls wir mit WR-Decodierung im Bereich $R<R_b$ arbeiten, erfüllt sie also das geforderte exponentielle Abnehmen der Restfehlerwahrscheinlichkeit mit n_0.

3. SW-Decodierung nach J.L. MASSEY

Im Kapitel VI haben wir MASSEY's beide Decodierverfahren kennengelernt. Zur Decodierung einer Stelle v_ρ liegen J - möglicherweise zusammengesetzte - Prüfgleichungen $\{A_j\}^{(\rho)}$ vor, die auf v_ρ orthogonal sind. Beide Verfahren fällen nun ihre Entscheidung über Stelle v_ρ aufgrund derjenigen n_E schon vorliegenden Stellen, die vom Prüfsatz $\{A_j\}^{(\rho)}$ erfaßt werden. Die restlichen n_0-n_E Stellen werden bei der Entscheidung über v_ρ nicht berücksichtigt. Je größer das Verhältnis n_E/n_0 bei fester Codelänge n_0 und fester Anzahl von J ist, desto besser wird die im Decoder vorliegende Information zur Korrektur von Stelle v_ρ ausgenützt. Für das algebraische Decodierverfahren werden aus dem empfangenen n-Tupel mit dem Satz von

Prüfgleichungen $\{A_j\}^{(\rho)}$ die Prüfwerte $s_j^{(\rho)}$ berechnet (vgl. Kapitel II, 7 und VI, 2). Das erste Decodierverfahren von MASSEY sucht unter den Fehlermustern, welche die gleichen Prüfwerte $s_j^{(\rho)}$ ergeben, dasjenige aus, das das geringste Gewicht hat, also die geringste Zahl von "1". Decodierung nach maximaler Rückschlußwahrscheinlichkeit dagegen würde bei ihrer Entscheidung weniger Fehlermuster berücksichtigen. Sie müßten nicht nur die J Prüfwerte $s_j^{(\rho)}$, sondern alle k Prüfwerte s_ν ergeben. MRW-Decodierung benützt also bloß einen Teil derjenigen Fehlermuster, die beim ersten SW-Verfahren zugelassen sind. Aus diesem Grunde können sich die beiden Entscheidungen durchaus unterscheiden, obwohl beide Verfahren nach einem Fehlermuster mit geringstem Gewicht suchen.

Das zweite SW-Verfahren ist statistischer Natur; es decodiert eine Stelle zwar aufgrund derselben Auswahl von Stellen. Seine Entscheidung ist aber immer die wahrscheinlichste unter Vorgabe einer bestimmten Übergangswahrscheinlichkeit p_0.

Natürlich gibt es nicht für jede beliebige Wahl von n, r, m und J Code, die die Konstruktion von J Prüfgleichungen für jede der m Nachrichtenstellen des ersten Prüfsatzes erlauben. MASSEY zeigt aber einen Weg zur Konstruktion solcher Code, falls die Zahl n_E der vom Prüfsatz $\{A_1\}$ erfaßten Stellen einer Beschränkung unterliegt. Es muß sein:

$$n_E \leq \frac{1}{2}\frac{R}{1-R} J^2 + \frac{1}{2} m \cdot J + 1 + \frac{1}{2} r \left[m - r \frac{R}{1-R} \right] \qquad (18)$$

dabei ist r = J mod(n-m). Das bedeutet, gleich dem Rest, der bei Division von J durch (n-m) bleibt. Die Restfehlerwahrscheinlichkeit PR dieser Code bleibt aber - unabhängig davon, wie groß n_0 gewählt ist - über einer festen Schranke:

$$PR > \frac{1}{2} \frac{2\, p_0 + n_0 - 1}{2\, p_0} \left(\frac{p_0}{q_0}\right) \qquad (19)$$

Diesem negativen Ergebnis steht aber gegenüber, daß SW-Decodierung besonders gute Decodierergebnisse auch bei kleiner Codelänge zeigt. Die Nachrichtenrate R ist - im Gegensatz zu WR-Decodierung und den LD-Code - keiner weitergehenden Einschränkung unterworfen als $R < C_0$.

Literaturverzeichnis

1) WOZENCRAFT, J.M. REIFFEN, B. — Sequential decoding, John Wiley and Sons, New York 1961

2) MASSEY, J.L. — Threshold decoding, MIT-Press, Cambridge (Mass.) 1963

3) GALLAGER, R.G. — Low-density parity-check codes, MIT-Press, Cambridge (Mass.) 1963

4) FANO, R.M. — A heuristic discussion of probabilistic decoding, IEEE Transact. on Information Theory IT--9, April 1963, Nr. 2, S. 64-74

5) FANO, R.M. — Transmission of information, John Wiley and Sons, New York 1961

6) PETERSON, W.W. — Error-correcting codes, John Wiley and Sons, New York 1961

7) WOZENCRAFT, J.M. JACOBS, I.M. — Principles of communication engineering, John Wiley and Sons, New York 1965

8) ELSPAS, B. — The theory of autonomous linear sequential networks, IRE Transact. CT-6 (1959) S. 45-60

9) HUFFMANN, D.A. — A linear circuit viewpoint of error correcting codes, Transact. IRE, IT-2 (1956), S.20-28