

TK 7.1)

Kanalcodierung

Um mit Fehlern bei der Übertragung in einem verrauschten Medium umgehen zu können, wird der serielle Bitstrom in Pakete eingeteilt und jedes Paket erhält zusätzliche Bits (Redundanz) um dem Detektor die Chance zu geben Fehler im Paket zu entdecken oder auch zu korrigieren. Dadurch wird die Informationsrate reduziert.

Fehlererkennung:

Erlaubt es in einem Bitstrom fehlerhafte Blöcke zu erkennen.

Über einen Rückkanal wird der fehlerhafte Block noch einmal angefordert.
D.h. keine konstante Informationsrate

ARQ - Verfahren [automatic repeat request]

1) "Stop and Wait"

→ "Go back N"

2) Kontinuierliches ARQ

→ "Selektive Wiederholung"

Fehlerkorrektionsverfahren:

Erkennt die Position des Fehlers im fehlerhaften Block und korrigiert ihn.

FEC [forward error correction coding]

1) Blockcodes (→ Lineare Gruppencodes → Zyklische Codes)

2) Faltungscodes

TK 7.2)

Güteparameter von Codes:

Hammindistanz:

Anzahl an Stellen an denen sich zwei Codewörter unterscheiden.
⇒ Maß wie leicht ein Codewort in ein anderes übergeht.

Codegewicht:

Wie viele "1"-en in einem binären Codewort enthalten sind.

Systematische Codes:

Wenn die k Informationsbits explizit im Codewort enthalten sind.
(strenge Def.: Codewort explizit enthalten, schwache Def.: Informationsbits explizit)

Coderrate:

$$R = \frac{k}{n} \dots \text{Coderrate}$$

k Informationsbits
 n Gesamtbits im Block

TK 7.2)

Die Wahrscheinlichkeit dass m von n Bits im Codewort fehlerhaft sind entspricht einer binomischen Variable.



$$n=5 \\ m=2$$

$$P(m) = \binom{n}{m} \cdot P_b^m \cdot (1-P_b)^{n-m}$$

$$\binom{n}{m} = \frac{n!}{m! (n-m)!} \quad \dots \dots \text{Binomialkoeffizient}$$

Die Wahrscheinlichkeit dass mehr als t Bits von insgesamt n Bits fehlerhaft sind in einem Codewort:

$$P(m > t) = 1 - P(m \leq t) = 1 - \sum_{m=0}^{t} P(m)$$

Für lange Codewörter und kleine P_b (BEP... Bit Error Probability) gibt es folgende Näherung:

$$P(m) \approx n \cdot P_b$$

Schwelleneffekt der BEP:

Bei sehr schlechtem SNR baut der Decoder mehr Fehler ein als ursprünglich vorhanden waren.

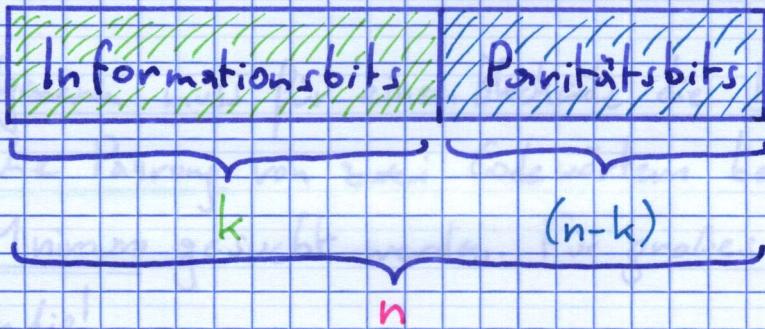
TK 7.3)

Blockcodes

Der einfachste Blockcode ist der Wiederholungscode. Es wird jedes auftretende Bit auf eine ungerade Anzahl gleicher Bits (Codewort) umgesetzt. Blockcodes sind systematische Codes!

$[n, k]$ - Blockcode:

Der Blockcoder fügt zu k Informationsbits, $(n-k)$ Paritätsbits hinzu.



z.B.: Single Parity Check Code (z.B. ASCII-Code: $k=7$, $n=8$)

Geringe Redundanz, kann nur 1 Fehler erkennen (oder ungern)

Produktcode

Werden die Informationsbits in einem 2-dim. Feld angeordnet, so können Single Parity Check Codes für die Zeilen und Spalten eingesetzt werden.

Einzelfehler können korrigiert werden.

■	■	■	■	■	■	■	■	■
■	■	■	■	■	■	■	■	■
■	■	■	■	■	■	■	■	■
■	■	■	■	■	■	■	■	■

Doppelfehler in denselben Zeile oder Spalte können nicht korrigiert werden.

TK 7.3)

Lineare Gruppencodes

Die linearen Gruppencodes stellen die größte Gruppe der Blockcodes dar.

⇒ Die Codewörter eines linearen Gruppencodes haben eine direkte Beziehung mit den Elementen einer mathematischen Gruppe.

D.h. durch Modulo-2 Addition zweier Codewörter ergibt sich immer ein anderes Codewort → die Gruppe ist abgeschlossen

Beurteilung der Leistungsfähigkeit

Im Allgemeinen muss für einen Blockcode die Hammingdistanz für jede mögliche Paarung von zwei Codewörtern berechnet werden und davon das Minimum gesucht werden. Für großes n wird dies sehr aufwendig!

Für lineare Gruppencodes vereinfacht sich diese Prozedur, indem nur die Hammingdistanz in Bezug auf das Nullwort berechnet wird und davon das Minimum bestimmt wird. Die Hammingdistanz lässt sich im Fall von linearen Gruppencodes einfach aus dem Codegewicht ablesen.

F35 [5,2]-Blockcode

TK 7.3)

d_{\min} minimale Hammingdistanz

$$t = \frac{d_{\min} - 1}{2} \quad \dots \text{garantierte Anzahl korrigierbarer Fehler im CW}$$

$d_{\min} - 1$ garantierte Anzahl an erkennbaren Fehlern im Codewort

Die Maximum-Likelihood-Entscheidung entscheidet sich für jenes Codewort, welches die geringste Distanz zum empfangenen Codewort hat.

Für lineare Blockcodes gibt es zwei Dekodierstrategien:

(1) Nearest Neighbor [Nächster Nachbar]

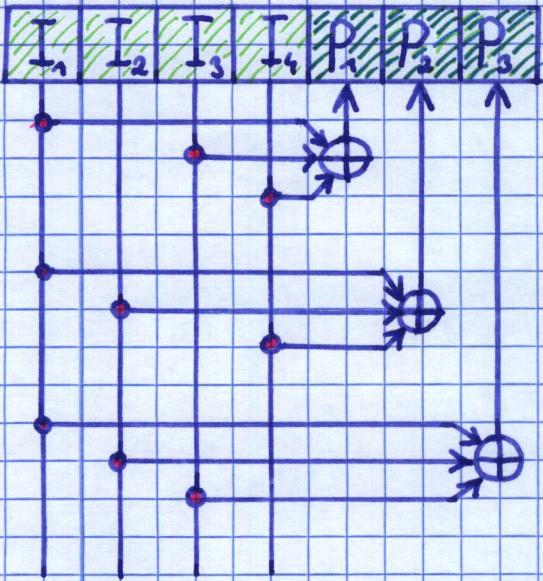
(2) Maximum-Likelihood [Höchst Wahrscheinlichster]

Hamming-Grenze? F 42

TK 7.4)

Dekodierung mit Hilfe des Syndroms

Die schaltungstechnische Realisierung von linearen Blockcodes erfolgt mit einem Schieberegister und EXOR's.



Paritätsgleichungen:

$$P_1 = I_1 \oplus I_3 \oplus I_4$$

$$P_2 = I_1 \oplus I_2 \oplus I_4$$

$$P_3 = I_1 \oplus I_2 \oplus I_3$$

Die Paritätsgleichungen werden in der Prüfmatrix H zusammengefasst:
 Transponierte Paritätsmatrix $[(n-k) \times k]$

$$H = \underbrace{\begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}}_{\text{Koeffizienten zu diesem Prüfbit}} \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \left(P^T : I_{(n-k)} \right)$$

Koeffizienten zu diesem Prüfbit

Die Generatormatrix G erzeugt die Codewörter auf systematische Art:

$$G = \begin{pmatrix} I_k & | & P \end{pmatrix} \Rightarrow c_m = d_m \cdot G$$

\uparrow

\uparrow

[$k \times n$]

Paritätsmatrix $[k \times (n-k)]$

Fehler in
Folien?

IK 7.4)

Erzeugung der Syndromtabelle

Das Problem der Dekodierung mit Hilfe der „Nearest Neighbor Decoding Tabelle“ ist deren Komplexität für großes n .

Die Syndromtabelle ist wesentlich kleiner, weil das Syndrom unabhängig vom übertragenen Codewort ist.

$$\text{Codewort: } c = d \cdot G$$

$$\text{weiter gilt: } H \cdot c = H \cdot d \cdot G = 0$$

$$\text{Empfangenes Codewort: } r = c \oplus e$$

$$\begin{aligned} \text{Syndromvektor: } s &= H \cdot r = H \cdot (c \oplus e) = \\ &= H \cdot c \oplus H \cdot e = 0 \oplus H \cdot e = \underline{\underline{H \cdot e}} \end{aligned}$$

\Rightarrow Für fehlerfreie Übertragung entspricht der Syndromvektor dem Nullvektor!

$$s = 0$$

Berechnen der Syndromtabelle:

$$s_n = H \cdot e_n \quad e_n \dots \dots \text{auftretende Fehlermuster}$$

Mit dem Syndrom kann ein Fehlermuster erkannt und damit das richtige Muster rekonstruiert wenn

TK 7.5)

Zyklische Codes

Die zyklischen Codes sind eine Untergruppe der Gruppencodes. Durch die zyklische Eigenschaft kann das Nullwort nicht verarbeitet werden.

Eigenschaften und Vorteile

- [1] Ihre mathematische Struktur erlaubt es Codes zu erzeugen, welche bessere Korrektoreigenschaften besitzen.
- [2] Die Codes können in eine einfache Hardwarestruktur umgesetzt werden, (Schieberegister und EXOR)
- [3] Alle Codewörter werden durch zyklische Verschiebung erzeugt.
- [4] Zyklische Codes werden durch Polynome erzeugt.

Zur Fehlererkennung werden systematisch erzeugte Cyclic Redundancy Checks (CRC) eingesetzt.

Gegeben ist eine Nachricht, bestehend aus k Informationsbits und ein Polynom $M(x)$ der Ordnung $k-1$.

Die Nachricht wird durch das GeneratorkPolynom $P(x)$ getragen. Dies wird durch Multiplikation (oder bitweiser Verschiebung) von $M(x)$ mit $P(x)$ erreicht.

$$m = [m_{k-1}, m_{k-2}, \dots, m_1, m_0] \dots \text{Nachricht}$$

$$M(x) = m_{k-1} \cdot x^{k-1} + \dots + m_1 \cdot x + m_0$$

TK 7.5)

Beispiel "CRC":

Geg.: Nachricht $m = [1001]$, Generatorpolynom $P(x) = 1 + x + x^3$

Gesucht: Codewort?

Aus $P(x) = 1 + x + x^3$ folgt: $[1101]$

$$\begin{array}{r} k \cdot M(x) = 1001000 \\ -1101 \\ \hline 100000 \\ -1101 \\ \hline 10100 \\ -1101 \\ \hline 1110 \\ -1101 \\ \hline 011 \dots \text{Rest} \end{array}$$

Sondersitzige
Codeworterzeugung

Codewort: $c = [1001 \text{ } 011]$

Empfängersitzig wird das Codewort auf Richtigkeit überprüft, indem das empfangene Codewort noch einmal durch das Generatorpolynom $P(x) = 1 + x + x^3$ dividiert wird.

Ist der Divisionsrest dabei Null, ist das Codewort richtig empfangen worden.

TK 7.5)

Interleaving

Codiersysteme können hervorragend mit gleichverteilten isolierten Einzelfehlern umgehen.

In der Praxis (z.B. Mobilfunk in Funkloch) treten aber oft mehrere Fehler hintereinander auf, sogenannte „Bündelfehler“.

Mit Bündelfehlern kann nur der Reed-Solomon Code umgehen.

Man kann jedoch durch Interleaving den Bündelfehler auseinander reißen und wieder zu Einzelfehlern machen.

↓ IN

→	I ₁	I ₅	I ₉	I ₁₃
OUT	I ₂	I ₆	I ₁₀	I ₁₄
	I ₃	I ₇	I ₁₁	I ₁₅
	I ₄	I ₈	I ₁₂	I ₁₆

Interleaver:

Liest den Bitstrom spaltenweise ein und gibt den Bitstrom zeilenweise aus.

Def.: Interleavertiefe Anzahl der Plätze im Feld

⇒ Keine Redundanzerhöhung durch zusätzliche Bandbreite.

⇒ Zeitverzögerung (Delay) entsprechend der Interleavertiefe.
(Vorsicht bei Echtzeitdiensten!)

TK 7.6)

Faltungscodierung $[n, k, v]$:

Der Faltungscode ist durch drei positive ganze Zahlswerte $[n, k, v]$ festgelegt.

Integerwert n legt die Anzahl der verschachtelten Codebits oder pro Zeitschritt (Datenbitdurch) erzeugt werden fest. Die Verschachtelung wird von n Modulo-2 Addieren ausgeführt.

Integerwert k legt die Anzahl der in den Encoden übernommenen Datenbits fest.

Integerwert v entspricht der Anzahl der Speicherzellen des Encoders. Damit ist die Anzahl der Zustände des Encoders auf 2^v festgelegt.

Die Komplexität des Encoders ist durch die Einflusslänge L festgelegt. $L = k + v$

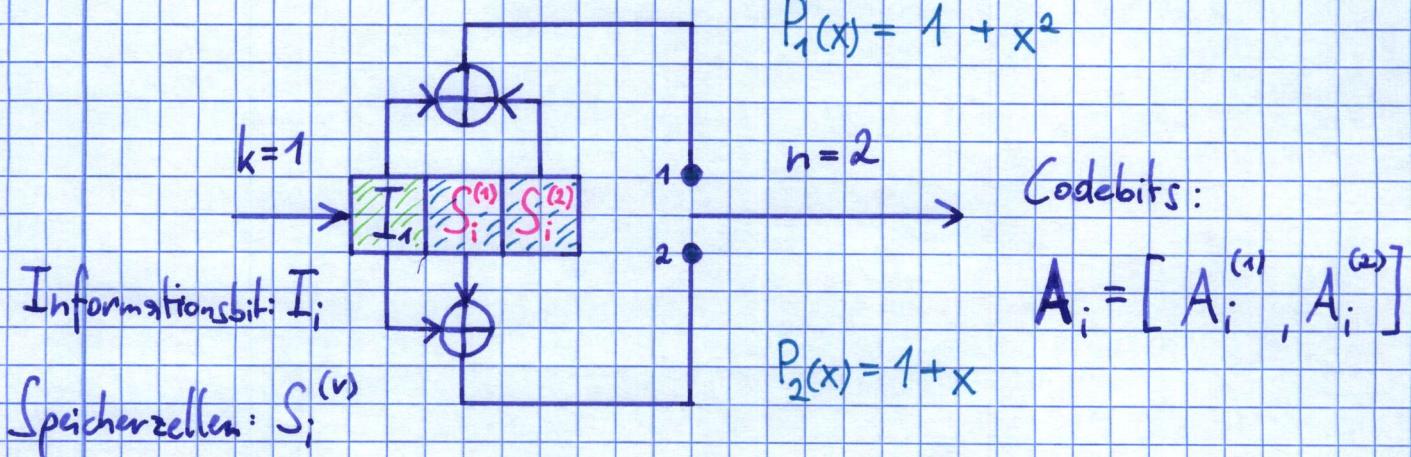
Die Werte für n und k sind in der Regel kleine Werte.

Ein wesentlicher Unterschied zur Blockcodierung liegt in den Speicherzellen. Ein von Faltungscoden erzeugtes n-Tupel ist nicht nur eine Funktion des in den Faltungscoder hineinlaufenden k-Tupels, sondern auch von den vorhergegangenen $(L - 1)$ k-Tupeln. Der Faltungscode gehört daher nicht mehr zur Gruppe der systematischen Codes!

TK 7.6)

Beispiel: $[2, 1, 2]$ -Encoder

Parameter: $n=2$, $k=1$, $v=2$, $\mathbf{g}_1 = [1, 0, 1]$, $\mathbf{g}_2 = [1, 1, 0]$



Einflusslänge: $L = v+k = 2+1=3$

Coderrate: $R = \frac{k}{n} = \frac{1}{2}$

Informationsbitfolge $\mathbf{I} = [I_1, I_2, I_3, I_4] = [1, 1, 0, 1]$

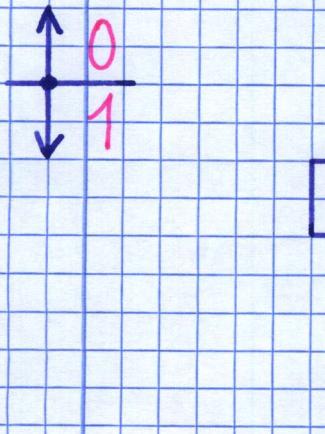
I_i	$S_i^{(1)}$	$S_i^{(2)}$	$A_i^{(1)}$	$A_i^{(2)}$
1	0	0	1	1
1	1	0	1	0
0	1	1	1	1
1	0	1	0	1

State-Machine mit
zwei Speicherzellen:
 $[00], [01], [10], [11]$
Anfangszustand

Das erste Informationsbit ist nach 3 Zeitschritten durch den Encoder
durch \rightarrow Einflusslänge L

TK 7.6)

Das Verhalten des Faltungscoders kann auch als Codebaum analysiert werden:



a

b

A00
000

a
b

B00
000

000

D00
000

000

H00
000

000

J01
010

111

J01
010

100

E11
010

100

K10
110

110

F01
010

010

k10
001

001

C11
100

100

j11
101

101

G10
011

110

j11
011

011

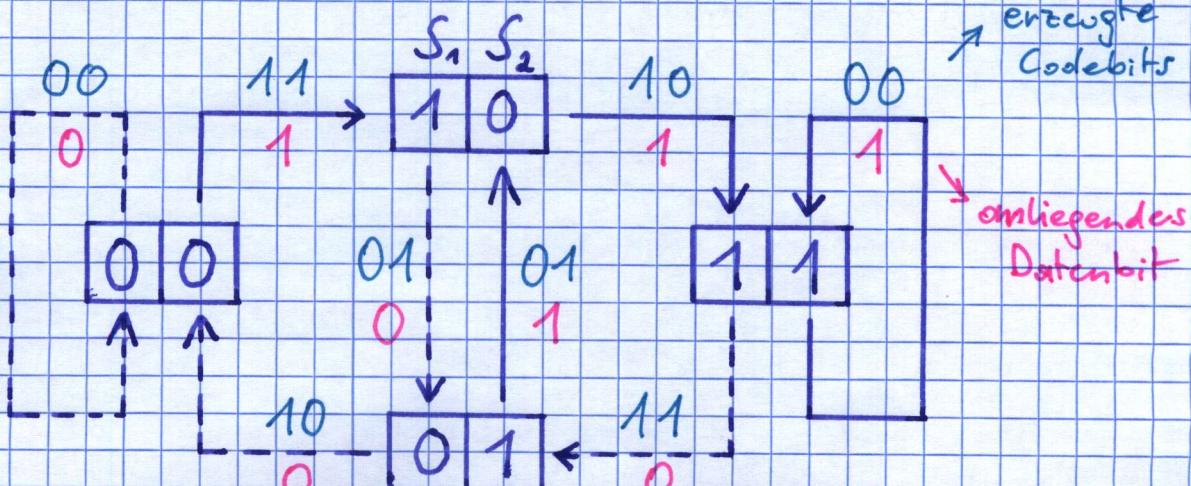
h00
111

111

Man erkennt, nach der Einflusslänge (3 Taktzyklen) sind alle Möglichkeiten aufgetreten.

Der Codebaum gibt keine Auskunft über das zeitliche Verhalten.

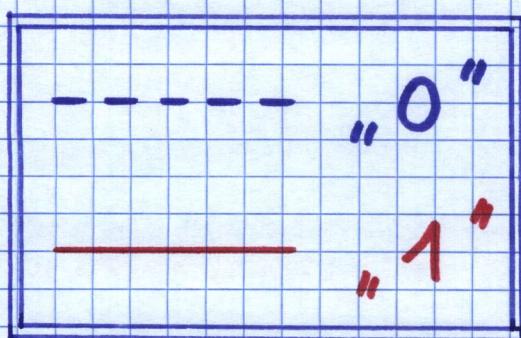
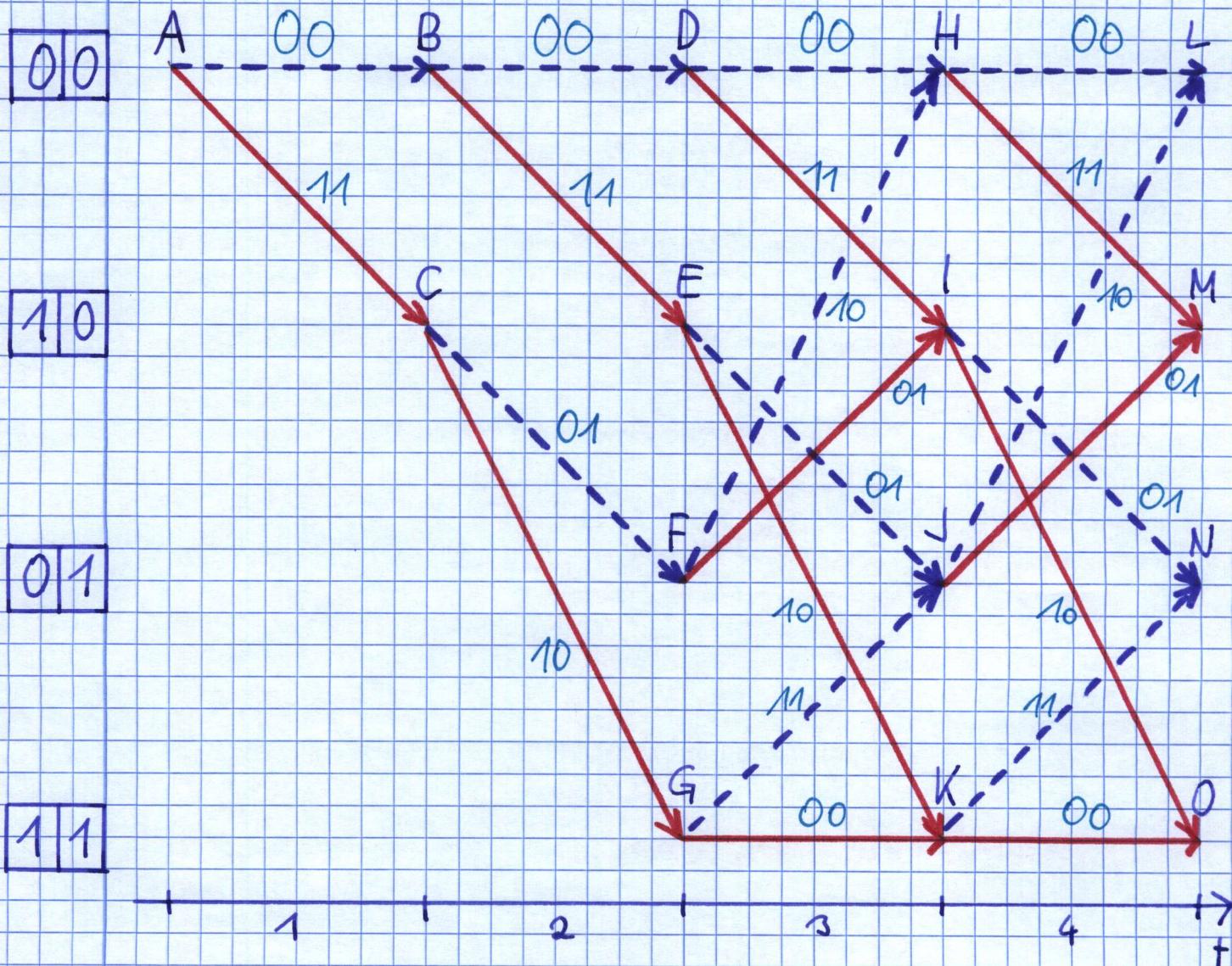
Etwas kompakter werden die Zustände des Encoders mit Hilfe des Zustandsdiagramms angegeben:



TK 7.6)

Ein Diagramm das den zeitlichen Ablauf der Zustände des Encoders enthält ist das Trellisdiagramm.

Man zeichnet dieses Diagramm indem man definiert vom Nullzustand aus geht. (Folgen nur Nulldatenbits bleibt man in der obersten Zeile.)



TK 7.7)

Für Faltungscodes gibt es drei Decodertypen:

[1] Viterbi - Dekodierung (am häufigsten verwendet)

[2] Sequential - Decoding

[3] Threshold - Decoding

{ Komplexität nimmt mit n zu,
Speicherbedarf steigt nur
linear mit n

Die Viterbi - Dekodierung ist ein aufwendiges Dekodierverfahren.

Typische Blocklängen liegen zwischen 500 und 10 000 Bits.

Blockaufbereitung:

[a] Speicher des Encoders wird mit Nullwort initialisiert (def. Beginn)

[b] Informationswortfolge wird codiert

[c] Das Nullwort wird in den Encoder nachgeschoben um diesen wieder in Zustand A zu bringen

Blockdekodierung:

Fensterlänge über Datenbits zur Auswertung
Üblicherweise 5-fache Einflusslänge!

[x] Der Viterbi - Algorithmus benutzt einen „Dekodier - Trellis“

[y] Die unwahrscheinlichsten Pfade werden verworfen. Dies wird über die Hamming - Distanz beurteilt.

[z] Der Viterbi - Algorithmus nimmt an das jener Pfad mit der geringsten Hammingdistanz der wahrscheinlichste ist.

⇒ „Nearest Neighbor Strategie“