# Informations- und Codierungstheorie

## Zusammenfassung v1.3

**Kälin Thomas, Abteilung I**

**WS 05/06**

# 1. Modell der Informationsverarbeitung

## 1.1. Quelle

Produziert Zeichen, die über einen Kanal (Medium) zu übertragen sind.

## 1.2. Senke

Empfänger, der die Zeichen interpretiert.

## 1.3. Kanal

Kanal ist immer fehlerhaft. Jeder Kanal ist immer auch ein Tiefpass.

## 1.4. Quellencodierer / Quellendecodierer

- Primär den Aufwand zur Darstellung der Nachricht reduzieren
  - Wann Vorwärtskorrektur? Fehler erlaubt? (Todesurteil!)
- Es werden irrelevante & redundante Anteile entfernt
- Bearbeitung der Quellinformation mit dem Ziel: Verschlüsselung, Komprimierung
- Verfahren
  - Huffmann
  - Lempel-Ziv

## 1.5. Kanalcodierer / Kanaldecodierer

Einführung einer Codierung, zur Minimierung der Restfehlerwahrscheinlichkeit. Einzige Aufgabe: Fehlerreduzierung

## 1.6. Modulator / Demodulator

Anpassung der Kodierung an den Kanal -> Leitungskodierung

## 1.7. Sender / Empfänger

Quelle produziert logische Zeichen, darum Umwandlung in physikalische Zeichen notwendig.

# 2. Wahrscheinlichkeitsrechnung

## 2.1. Grundlagen

- Sicheres Ereignis: Wahrscheinlichkeit 1, Unmögliches Ereignis: Wahrscheinlichkeit 0

```
Wahrscheinlichkeit
p = h/n

Gegenwahrscheinlichkeit
q = (n-h)/n = 1-p

Beispiel
-  Würfel mit 6 Seiten (1..6), Wahrscheinlichkeit für eine 5 oder 6?
-  p = 2/6 = 1/3
-  q = 4/6 = 2/3
```

## 2.2. Bedingte Wahrscheinlichkeit

- Wahrscheinlichkeit für E2, unter der Bedingung, dass E1 bereits aufgetreten ist.

```
Bedingte Wahrscheinlichkeit
PR{E2|E1} = PR{E1*E2}/PR{E1}
PR{E1E2} = PR{E1}*PR{E2|E1}

Beispiel
Kongo-Topf: 5 rote Kugeln, 4 schwarze, 2 blaue -> Alle in einem Topf. Wie
hoch ist Wahrscheinlichkeit, 2 blaue zu ziehen (ohne zurücklegen!)?

1. Versuch: p1 = PR{E1} = 2/11
2. Versuch: p2 = PR{E2|E1} = 1/10
Total: pBlau = p1 * p2 = 2/11 * 1/10 = 1/55

Unabhängiges Ereignis
PR{E1E2} = PR{E1}*PR{E2}

Beispiel
Kongo-Topf: 5 rote Kugeln, 4 schwarze, 2 blaue -> Alle in einem Topf. Wie
hoch ist Wahrscheinlichkeit, 2 blaue zu ziehen (mit zurücklegen)?

1. Versuch: p1 = 2/11
2. Versuch: p2 = 2/11
Total: pBlau = p1 * p2 = 2/11 * 2/11 = 4/121
```

## 2.3. Ausschliessliches Ereignis

- Entweder E1 oder E2

```
Ausschliessliches Ereignis
PR{E1E2} = 0 -> Beide zusammen können nicht auftreten
PR{E1+E2} = PR{E1} + PR{E2}

Nicht-Ausschliessliches Ereignis
PR{E1+E2} = PR{E1} + PR{E2} – PR{E1E2}

Beispiel: Jasskarten
Total haben wir 52 Karten.

1. Ausschliesslich: p(Ass oder König) = 4/52 + 4/52 = 2/13
2. N-Ausschliesslich: p(Ass oder Pik) = 4/52 + 13/52 – 1/52 = 4/13
```

# 3. Quellencodierung

## 3.1. Information (17)

- Information ist relevant und nicht-redundant.
- Häufig vorkommende Ereignisse liefern wenig Information, seltene jedoch sehr viel
- Komprimierung: Redundanz herausnehmen

## 3.2. Entscheidungsgehalt / Entscheidungsfluss (18)

- Mass für den Aufwand, der zur Bildung einer Nachricht, bzw. für die Entscheidung einer Nachricht notwendig ist, heisst Entscheidungsgehalt.
- Anzahl der Elementarenscheidungen: 0 oder 1

```
Entscheidungsgehalt
Ho = lb(N) [Bit]

Entscheidungsfluss
Ho* = lb(N)/t [Bit/s]

Beispiel
Zeichenvorrat X={a,b,c,d} => N=4
Ho = lb(4) = 2 Bit
```

## 3.3. Informationsgehalt, Entropie und Redundanz (19, 20)

- Informationsgehalt und Wahrscheinlichkeit stehen in umgekehrtem Verhältnis
- Ist Wahrscheinlichkeit sehr gering, ist der Informationsgehalt sehr hoch
- Entropie und Entscheidungsgehalt sind genau dann gleich gross, wenn alle Zeichen gleich wahrscheinlich sind. In diesem Fall gibt es keine Redundanz.

```
Informationsgehalt
I(xi) = -lb(p(xi)) [Bit]

Entropie: Mittlerer Informationsgehalt der Quelle
H(X) = S(i=1,N,p(xi)*I(xi)) [Bit/Zeichen]
H(X) = -S(i=1,N,p(xi)*lb(p(xi))) [Bit/Zeichen]

Redundanz
Rq = Ho – H(X) [Bit/Zeichen]

Beispiel:
X = {a,b,c,d} => N=4
pa = pb = pc = pd = 0.25

Ho = lb(N) = 2Bit
I(a) = -lb(0.25) = 2 Bit
H(X) = 0.25*2bit + 0.25*2bit + 0.25*2bit + 0.25*2bit = 2 Bit/Zeichen
Rq = Ho-H(X) = 2Bit – 2 Bit/Zeichen = 0 Bit/Zeichen
```

## 3.4. Codewortlänge, Redundanz des Codes (21, 24)

- Bei der Quellencodierung werden die diskreten Zeichen der Quelle auf binäre Codeworte abgebildet
- Warum Unterscheidung zwischen I(xi) und L(xi)? Weil L(xi) keine Kommazahl sein kann, es können ja nicht Viertel-Bits übertragen werden. Die mittlere Codewortlänge kann hingegen durchaus Kommawerte enthalten!
- Feste CW-Länge (ASCII, 8 Bit) und variable CW-Länge (Morse, 1..4 Bit)

```
Codewortlänge
L(xi) = Länge des CW in Bit (keine Kommawerte: 101 -> 3 Bit)

Mittlere Codewortlänge
L = S(i=0,N,p(xi)*L(xi)) [Bit/Zeichen]

Redundanz des Codes
Rc = L – H(X) [Bit/Zeichen]
```

## 3.5. Präfixeigenschaft (S 22)

- Präfixfrei = Keine Trennzeichen vorhanden, gültige Zeichen liegen nur in den Blättern eines Baumes
- Wird auch „kommafreier Code" genannt
- Morse-Code ist nicht Präfixfrei, da Pausen zur Unterscheidung gemacht werden

## 3.6. Quellen mit und ohne Gedächtnis (25, 26, 27)

- Ohne Gedächtnis: Die Auftrittswahrscheinlichkeit eines Zeichens ist unabhängig von dem zuvor emittierten Zeichen, bzw. der zuvor emittierten Zeichenfolge
- Die Entropie einer Quelle ohne Gedächtnis ist stets grösser oder gleich der Entropie einer Quelle mit Gedächtnis
- Beispiel 7 in den Übungsaufgaben

## 3.7. Huffmann-Codierung (29)

- Verfahren zur Entwicklung eines Codes mit minimaler Codewortlänge
- Zeichen, die häufig vorkommen, haben kurze Codewörter
- Es wird ein präfixfreier Binärbaum erzeugt
- Je grösser die Wahrscheinlichkeit für ein Zeichen, desto weiter oben im Baum findet sich dieses

## 3.8. Lempel-Ziv (32, 34)

- Der zu komprimierende Code hat wiederkehrende Muster oder Phrasen
- Es werden längere Zeichenketten codiert
- Anstatt den Code vollständig zu übertragen, werde nur die codierten Phrasen übertragen
- Benötigt einen Phrasenspeicher / Wörterbuch
- Während des Durchlaufens der Daten wird ein ständig wachsender Baum erzeugt
- Problem: Der Text muss Regelmässigkeiten beinhalten. Ansonsten wird der „komprimierte" Text länger als das Ursprungssignal

# 4. Kanalcodierung

## 4.1. Kanalmodell (39, 40, 41)

- Jeder Kanal ist fehlerhaft
- Woher kommt die Fehlerbehaftung?
  - Störeinflüsse, Jede Leitung ist ein Tiefpass, Phasenverschiebung
- Bedeutung: Entropie am Ausgang ist anders als am Eingang

### 4.1.1. Kanalmatrix

- Kanalmatrix wird durch Messungen bestimmt
- Summe jeder Zeile: 1
- Summe der Spalten: BELIEBIG! (Siehe auch Maximum-Likelyhood)

```
+------------------+     Lesebeispiel: p(y2|x1)
|p(y1|x1)  p(y2|x1)|     Wahrscheinlichkeit, dass y2 rauskommt, unter der
|p(y1|x2)  p(y2|x2)|     Voraussetzung, dass x1 anliegt.
+------------------+

p(y1) = p(x1)*(y1|x1)+p(x2)*(y1|x2)
p(y2) = p(x1)*(y2|x1)+p(x2)*(y2|x2)
```

### 4.1.2. Störungsfreier Kanal (kanal-symmetrisch) & Vollständig gestörter Kanal

```
+--------+   +--------+
|1.0  0.0|   |0.5  0.5|
|0.0  1.0|   |0.5  0.5|
+--------+   +--------+
```

### 4.1.3. Verlustfrei, aber rauschbehafteter Kanal

```
+-------------+
|1.0  0.0  0.0|
|0.0  0.6  0.4|
+-------------+
```

## 4.2. Maximum-Likelyhood

- Wie gross ist die Restfehlerwahrscheinlichkeit und wovon hängt sie ab?
  - Wahrscheinlichkeiten am Kanaleingang und Kanalmatrix
- Die Fehlerwahrscheinlichkeit kann unter Einbezug des Informationsgehaltes der einzelnen Zeichen gegebenenfalls reduziert werden.
- Der Entscheider hat Informationen über die Kanalmatrix

```
Beispiel 1       Beispiel 2
+-------------+  +-------------+
|0.6  0.2  0.2|  |0.6  0.2  0.2|
|0.2  0.2  0.6|  |0.2  0.2  0.6|
|0.3  0.2  0.5|  |0.3  0.2  0.5|
+-------------+  +-------------+

Beispiel 1 (Berechnung)
P(x1) = P(x2) = P(x3) = 0.33
Pr = P(y1|x1)*P(x1) + P(y2|x3)*P(x3) + P(y3|x2)*P(x2)
Pr = 0.6*0.33 + 0.2*0.33 + 0.6*0.33 = 0.47
Pf = 1-Pr = 0.53

Beispiel 2 (Berechnung), Eliminierung von x3
P(x1)=0.4, P(x2)=0.3, P(x3)=0.3
Pr = P(y1|x1)*P(x1) + P(y2|x1)*P(x1) + P(y3|x2)*P(x2)
Pr = 0.6*0.4 + 0.2*0.4 + 0.6*0.3 = 0.5
Pf = 1-Pr = 0.5
```

## 4.3. Transinformation

### 4.3.1. H(X) = Entropie am Kanaleingang

```
H(X) = -S(i=1,N, p(xi)*lb(p(xi)) ) [Bit/Zeichen]
```

### 4.3.2. H(Y) = Entropie am Kanalausgang

- Bei störungsfreiem Kanal ist H(Y) = H(X)

```
H(Y) = -S(i=1,N, p(yi)*lb(p(yi)) ) [Bit/Zeichen]
```

### 4.3.3. H<X,Y> = Verbundentropie

- Gibt Entropie für Zeichenpaare (xi,yi) am Eingang und am Ausgang des Kanals an

```
H<X,Y> = -S(i,N, S(j,N, p(xi,yj)*lb(xi,yj)) ) [Bit/Zeichenpaar]
H<X,Y> = -S(i,N, S(j,N, p(xi)*p(yj|xi)*lb(p(xi)*p(yj|xi))) )

H<X,Y> = H(X) + H<Y|X>
```

- Errechnet sich aus der Wahrscheinlichkeit der Zeichen und der Übergangswahrscheinlichkeit
- Summen der Zeilen und Spalten muss Wahrscheinlichkeit der einzelnen Zeichen ergeben

```
Übergangswahrscheinlichkeit          Wahrscheinlichkeit der Zeichen
     x1   x2   x3
   +--------------
x1 | 0.1  0.5  0.4                   p(x1)=0.276
x2 | 0.4  0.2  0.4                   p(x2)=0.323
x3 | 0.3  0.3  0.4                   p(x3)=0.4

=> Durch Berechnung Verbundwahrscheinlichkeiten
       x1    x2    x3
   +--------------------
x1 |   0.028 0.139 0.111       Beispiel:  p(x1,x1) = p(x1) * px(y1|x1)
x2 |   0.129 0.065 0.129                  p(x1,x1) = 0.276 * 0.1
x3 |   0.12  0.12  0.16                   p(x1,x1) = 0.028
   +--------------------
       0.276 0.323 0.4
```

### 4.3.4. H<X|Y> = Äquivokation, Verlust

- Gibt die Unsicherheit des Entscheiders beim Rückschluss auf das vermutlich gesendete Zeichen
- Benötigt Kanalmatrix

```
H<X|Y> = -S(i,N, S(j,N, p(xi)*p(yj|xi)*lb(p(xi|yj)) [Bit/Zeichen]
```

### 4.3.5. H<Y|X> = Irrelevanz, Rauschen

- Beschreibt, welche Veränderung der Kanal an den Zeichen in Folge eines Fehlers vornimmt.
- Bei Störungsfreiem Kanal ist H<Y|X> = 0
- Benötigt Kanalmatrix

```
H<Y|X> = -S(i,N, S(j,N, p(xi)*p(yj|xi)*lb(p(yj|xi)) [Bit/Zeichen]
```

### 4.3.6. T = Transinformation

- Transinformation ist ein Mass für den tatsächlichen Informationsfluss
- Beim vollständig gestörten Kanal ist diese 0

```
T = H<Y> - H<Y|X> [Bit/Zeichen]
T = H<X> - H<X|Y> [Bit/Zeichen]
```

### 4.3.7. Symbolrate = Tatsächlicher Informationsfluss

- Wie gross ist der Tatsächliche Informationsfluss?

```
S = T * Übertragungsrate [Bit/s]
-> Benötigte Übertragungsrate unter Einbezug der Symbolrate (Beispiel 3)
-> Üb = Übertragungsrate * Übertragungsrate / Symbolrate
```

### 4.3.8. Rechenbeispiel

```
     x1   x2   x3
   +--------------
x1 | 0.1  0.5  0.4       =>    I     |p(x1)= 0.1*p(x1)+0.4*p(x2)+0.3*p(x3)|
x2 | 0.4  0.2  0.4             II    |p(x2)= 0.5*p(x1)+0.2*p(x2)+0.3*p(x3)|
x3 | 0.3  0.3  0.4             III   |1    = p(x1)+p(x2)+p(x3)            |

                         => p(x1)=0.276, p(x2)=0.323, p(x3)=0.4

H(X) = Entropie am Kanaleingang
H(X) = -(p(x1)*lb(p(x1)) + p(x2)*lb(p(x2)) + p(x3)*lb(p(x3)))
H(X) = -(0.276*lb(0.276)) + 0.323*lb(0.323) + 0.4*lb(0.4))
H(X) = 1.5684 Bit/Zeichen

H<X,Y> = Verbundentropie
H<X,Y> =-(p(x1,y1)*lb(p(x1,y1))+p(x1,y2)*lb(p(x1,y2))+p(x1,y3)*lb(p(x1,y3))
        +(p(x2,y1)*lb(p(x2,y1))+p(x2,y2)*lb(p(x2,y2))+p(x2,y3)*lb(p(x2,y3))
        +(p(x3,y1)*lb(p(x3,y1))+p(x3,y2)*lb(p(x3,y2))+p(x3,y3)*lb(p(x3,y3))
H<X,Y> = -(0.276*0.1*lb(0.276*0.1) + 0.276*0.5*lb(0.276*0.5) +
           0.276*0.4*lb(0.276*0.4) + 0.323*0.4*lb(0.323*0.4) +
           0.323*0.2*lb(0.323*0.2) + 0.323*0.4*lb(0.323*0.4) +
           0.4*0.3*lb(0.4*0.3)     + 0.4*0.3*lb(0.4*0.3)     +
           0.4*0.4*lb(0.4*0.4))
H<X,Y> = 3.06 Bit/Zeichenpaar

H<Y|X> = Irrelevanz (Bedingte Entropie)
H<Y|X> = H<X,Y> - H(X)
H<X|X> = 3.06 – 1.5684 = 1.492 Bit/Zeichen
```

### 4.3.9. Rechenbeispiel 2

```
+----------+
| 0.9  0.1 | => p(x1) = 0.3
| 0.1  0.9 | => p(x2) = 0.7
+----------+

H(X) = Entropie am Kanaleingang
H(X) = -(0.3*ld(0.3) + 0.7*ld(0.7)) =  0.8812 Bit/Zeichen

H(Y) = Entropie am Kanalausgang
p(y1) = 0.3*0.9 + 0.7*0.1 = 0.34
p(y2) = 0.3*0.1 + 0.7*0.9 = 0.66
H(Y) = -(0.34*ld(0.34) + 0.66*ld(0.66)) = 0.9248 Bit/Zeichen

H<Y|X> = Irrelevanz, Rauschen
H<Y|X> = -(p(x1)*p(y1|x1)*lb(y1|x1) + p(x1)*p(y2|x1)*lb(y2|x1) +
           p(x2)*p(y1|x2)*lb(y1|x2) + p(x2)*p(y2|x2)*lb(y2|x2) )
H<Y|X> = -(0.3*0.9*lb(0.9) + 0.3*0.1*lb(0.1) +
           0.7*0.1*lb(0.1) + 0.7*0.9*lb(0.9)
H<Y|X> = 0.4689 Bit/Zeichen

T = Transinformation
T = H(Y) – H<Y|X> = 0.9248 – 0.4689 = 0.4558 Bit/Zeichen

S = Symbolrate (Tatsächlicher Informationsfluss)
- Die Übertragungsrate ist 1000 Bit/s
- S = T * Übertragungsrate = 0.4558 * 1000Bit/s = 455 Bit/s
```

### 4.3.10. Rechenbeispiel 3

```
+--------------+
| 0.995  0.005 |   => p(x1) = 0.5
| 0.01   0.99  |   => p(x2) = 0.5
+--------------+

H(Y) = Entropie am Kanalausgang
p(y1) = 0.5*0.995 + 0.5*0.01 = 0.5025
p(y2) = 0.5*0.005 + 0.5*0.99 = 0.4975
H(Y) = -(p(y1)*lb(p(y1) + p(y2)*lb(p(y2)))
H(Y) = -(0.5025*lb(0.5025) + 0.4975*lb(0.4975))
H(Y) = 0.999982 Bit/Zeichen

H<Y|X> = Irrelevanz, Rauschen
H<Y|X> = -(p(x1)*p(y1|x1)*lb(y1|x1) + p(x1)*p(y2|x1)*lb(y2|x1) +
           p(x2)*p(y1|x2)*lb(y1|x2) + p(x2)*p(y2|x2)*lb(y2|x2) )
H<Y|X> = -(0.5*0.995*lb(0.995) + 0.5*0.005*lb(0.005) +
           0.5*0.01*lb(0.01) + 0.5*0.99*lb(0.99))
H<Y|X> = 0.0631039 Bit/Zeichen

T = Transinformation
T = H(Y) – H<Y|X> = 0.999982 – 0.0631039 = 0.93688 Bit/Zeichen

S = Symbolrate (Tatsächlicher Informationsfluss)
- Die Übertragungsrate ist 1600Bit/s
S = T * Übertragungsrate = 0.93688 * 1600 Bit/s = 1499Bit/s

- Welche Übertragungsrate ist nötig, um wirklich 1600Bit/s zu erreichen?
Ü = (1600Bit/s) * (1600 Bit/s) / (1499 Bit/s) = 1707 Bit/s
```

## 4.4. Fehlererkennung

### 4.4.1. Hinzufügen von Redundanz

Wir fügen Redundanz hinzu, so dass sich der zur Verfügung stehende Coderaum in gültige und ungültige Codeworte aufteilt.

### 4.4.2. Idee zur Fehlererkennung

Zeichen zu Blöcken fassen. Block bildet gültiges oder ungültiges CW. Zur Erkennung ist Redundanz erforderlich.

### 4.4.3. Fehlerkorrektur

- Retransmission: Fehler erkennen und nachfragen (HDLC -> GSM | ISDN, LLC, TCP -> OSI4)
- Vorwärtskorrektur: Fehler erkennen & wahrscheinlichen Fehlerort bestimmen, anschliessend Korrektur (GSM, UMTS)

## 4.5. Coderaum (48)

### 4.5.1. Hammingdistanz

Die Hammingdistanz gibt den minimalen Abstand zwischen 2 gültigen CW an

```
h = Min(d(xi,xj))
```

### 4.5.2. Sicher erkennbare Fehler

- Achtung: Erkennbar sind viel mehr, aber nicht sicher!

```
e*= h-1
```

### 4.5.3. Sicher korrigierbare Fehler

```
h ist gerade:      e = (h-2) / 2
h ist ungerade:    e = (h-1) / 2
```

### 4.5.4. Beispiel

```
h=3    =>    o  x)(x  o        => e = 1
h=4    =>    o  x) x (x  o     => e = 1
h=5    =>    o  x  x)(x  x  o  => e = 2
```

### 4.5.5. Dichtgepackt

Der Coderaum ist dichtgepackt, wenn sich alle Codewörter (gültige und ungültige) in einer Korrigierkugel befinden.

- Man kann von der Hammingdistanz nicht darauf schliessen, ob der Code dichtgepackt ist, da nur der minimale Abstand angegeben wird.
- Wenn die Hammingdistanz jedoch gerade ist, ist dies hinreichen dafür, dass der Code NICHT dichtgepackt ist!

```
Alle Codeworte     = 2^n
Gültige Codeworte  = 2^m
n = m+k = Nachrichtenstellen + Kontrollstellen
```

## 4.6. Blockcode (Hammingcode)

- Alle Vektoren paarweise verschieden und ungleich 0, da 0 bei einem gültigen CW herauskommen soll
- Hammingdistanz kann ausgelesen werden, in folgendem Beispiel 3 (3 Vektoren nötig, um einen 000-Vektor zu erzeugen)
- 000 heisst alles okay, oder es sind 3 Fehler aufgetreten…
- Nicht optimal, da bei grossen Codes die Generatormatrix riesig wird

### 4.6.1. Generatormatrix (53)

```
  1 0 1 1 0 0 0                   1 0 1 1 0 1 0
+---------------+               +---------------+
| 1 1 1 0 1 0 0 |        0      | 1 1 1 0 1 0 0 |       0
| 0 1 1 1 0 1 0 | =>     0      | 0 1 1 1 0 1 0 | =>    1 = Fehlersyndrom
| 1 1 0 1 0 0 1 |        0      | 1 1 0 1 0 0 1 |       0
+---------------+               +---------------+
```

## 4.7. Zyklische Codes (57)

Idee: Genereratormatrix kann durch Generatorpolynom beschrieben werden

- Ein linearer code (n,k) C heisst zyklischer Code, wenn jede zyklische Verschiebung eines CW aus C wiederum ein CW in C ergibt
- CW-Länge bleibt immer gleich lang

```
n = Anzahl der Stellen eines CW
m = Anzahl der Nachrichtenstellen
k = Anzahl der Kontrollstellen

n = m+k
n = 2^k-1 => -1 zählt den 0-Vektor ab

Generatorpolynom: u³+u+1 => ³ = 3 Kontrollstellen = k
```

### 4.7.1. Polynomdivision (59)

```
Generator: G(u) = u³ + u + 1 => 1011

1000101 : 1011
1011
----
%011
 0000
 ----
 %110
  1011
  ----
  %111
   1011
   ----
   %101 => 101 sind die gesuchten Kontrollstellen
```

### 4.7.2. Mehrfachaddition (60)

```
Generator: G(u) = u³ + u + 1 => 1011

1000...
-------
1011
  1011
   1011
-------
1000101 => 101 sind die gesuchten Kontrollstellen
```

### 4.7.3. Prüfung der Codebedingung (61)

```
Generator: 1011

1000101
1011
  1011
   1011
-------
0000000 => Codebedingung erfüllt
```

### 4.7.4. Fehlersyndrom => Generatormatrix (62)

```
1001101
1011
  1011
-------
0000011 => Fehlersyndrom, Codebedingung nicht erfüllt
```

- Die Generatormatrix kann ermittelt werden, indem man ein gültiges CW nimmt, und schrittweise an jeder Stelle einen Fehler einbaut. Natürlich muss bei jeder Position das Fehlersyndrom errechnet werden. Diese ergeben anschliessend die Generatormatrix

### 4.7.5. Rückgekoppeltes Schieberegister (63)

Siehe Skript (Grafik)

### 4.7.6. Zyklische Hammingcodes (64)

- Hammingdistanz = 3
- Zykluslänge wird bestimmt durch Grad k des Polynoms
- Werden durch primitive Polynome gebildet
  - Nur durch 1 und sich selbst teilbar
  - Werden aus Büchern herausgelesen

### 4.7.7. Zyklische Abramson-Codes (CRC-Codes)

- Hammingdistanz = 4
- Werden gebildet durch die Multiplikation eines primitiven Polynoms mit dem Term (x+1)

## 4.8. Faltungscodes (65)

- Blockcodes wurden schnell zur Sicherung gegen Übertragungsfehler eingesetzt, nicht zuletzt wegen ihrer einfachen Implementierbarkeit
  - Leichte Implementierung von Encode / Decoder durch Schieberegister
  - Hohe Fehlererkennungsmächtigkeit
  - Blockbildung der zu codierenden Daten notwendig => Nicht optimal für Streams, da jeder Block überprüft werden muss.
- Faltungscodes, Einsatz in GMS und UMTS
  - Faltungscodes erlauben die fortlaufende Codierung eineskontinuierlichen Datenstroms, es ist also KEINE Blockbildung erforderlich
  - Prüfbits sind fortschreitend in den Code eingebaut
  - Gesendete Bits haben Einfluss auf nachfolgende Bits

### 4.8.1. Idee der Faltung (66)

```
Generator: 1011
Eingangsstrom: 11001     =>    Gespiegelt: 10011

1) Spalten multiplizieren            (1*1) (1*0)
2) Resultate addieren                  1  +  0  = 1
3) %2 rechnen -> Ergibt Stelle       1 % 2 = 1

10011        10011       10011       10011       10011        10011
    1011         1011       1011       1011       1011        1011
--------     -------     ------      -----       ------      ------
    1            11         111        1110       11100       111000
```

### 4.8.2. (2,1,3)-Encoder (68)

- Schema siehe Skript
- Rechenvorschrift der binären Faltung lässt sich leicht durch ein Schieberegister darstellen
- Der Eingangsstrom bläht sich auf (Nachricht) -> fortlaufende Redundanz einfügen

### 4.8.3. Zustandsdarstellung (69)

- Der Zustandsraum kann als Automat dargestellt werden, siehe Schema im Skript

### 4.8.4. Netz- oder Trellis-Diagramm (70)

- Mit Helfe des Netzdiagrammes kann eine Eingangsfolge codiert werden
- Schema siehe Skript

### 4.8.5. Fundamentalweg und Gewichte (71)

- Gewicht des Codes ist die Anzahl von Bitstellen eines CW, die von 0 verschieden sind (Anzahl 1)
- Fundamentalweg: Beginnt bei S0 und endet wieder bei S0
- Metrik
  - I: Bezeichnet den Zustandsübergang, der durch eine 1 ausgelöst wird (Hamminggewicht der Nachrichtenfolge)
  - D: Bezeichnet die Anzahl der 1-Bitstellen, die beim Übergang hinzugekommen sind (Hamminggewicht der Codefolge)
  - J: Einfache Zählvariable, wird immer um 1 erhöht

### 4.8.6. Generatorpolynome für optimale Faltungscodes (72)

- Liste siehe Skript
- Gute Faltungscodes werden durch Computersimulation gefunden, kein analytisches Verfahren vorhanden

### 4.8.7. Codierung des Faltungscodes (73)

- Fundamentalwege: Teilfolgen, die von der Nullfolge verschieden sind. Fangen bei S0 an und gehen zu S0 zurück.
- Jede Folge beginnt und endet in seiner Nullfolge. Wegen den Tailbits enden wir immer in einer Null.
- Jeder Code hat einen / mehrere Fundamentalwege
- Um eine Korrektur vorzunehmen, müssen die Fundamentalwege möglichst unterschiedlich sein.
- Durch diese Unterschiedlichkeit wird die Hammingdistanz festgelegt

### 4.8.8. Decodierung des Faltungscodes / Viterbj-Algorithmus (74)

- Zur Decodierung des Faltungscodes wird üblicherweise der Viterbj-Algorithmus eingesetzt

### 4.8.9. Formeln zur Faltung

```
(n,k,m) - Encoder
 | | |
 | | +-- m = Anzahl der Zustandsspeicher
 | +---- k = Anzahl der Eingänge
 +------ n = Anzahl der Ausgänge

Coderate R
R = k/n = Eingänge / Ausgänge

Blockcoderate („Effizienz“): Tailbits einbeziehen
BR = k*Bits / n*(Bits + Tailbits)

Tailbits
Anzahl Zustandspeicher = Anzahl Tailbits

Encodergedächtnis
Anzahl Zustandspeicher = m

Einflusslänge nc / Wirkungstiefe
Die Einflusslänge gibt die maximale Spannweite der von einem Eingangsbit beinflussten Codebits an.
nc = m+1 = Anzahl Speicher + 1

Grösse des Zustandsraumes
Zustandraum = 2^m = 2^Anzahl Zustandsspeicher
```

```
Beispiel
(3,1,2)- Faltungscode
 | | m = 2 Speicher
 | k = 1 Eingang
 n = 3 Ausgänge

Coderate R = k/n = 1/3
BR = 1*100 Bits / 3*(100Bits+2Tailbits) = 0,326
Einflusslänge nc = m+1 = 2+1 = 3

Mit diesen Werten können nun aus der Tabelle im Skript (72) die optimalen Generatorpolynome bestimmt werden. Achtung: Oktale Schreibweise!

g1 = 05 = 101
g2 = 07 = 111
g3 = 07 = 111
```

# 5. Signale

## 5.1. Grundsätzliches (79 / 80)

- Audio / Video: kontinuierliche Signale
  - Haben einen unendlichen Wertebereich bezüglich der Signalstärke (Amplitude)
  - Haben einen unendlichen Wertebereich bezüglich der Frequenz, selbst dann, wenn der Frequenzbereich eingeschränkt ist
- Unendliche Wertebereiche können wir mit einem Computer nicht abspeichern. Wir brauchen einen diskreten Wertebereich.

## 5.2. Klassifikation von Signalen (80)

- Wertdiskret
  - Einfügen diskreter Wertstufen
  - Approximation des kont. Signals an die verfügbaren Werte
  - Treppenbildung, Verlust
  - Wie viele Wertstufen sind nötig? -> Ist abhängig von Hörer!
- Zeitdiskret
  - Signal, welches sich kontinuierlich über die Zeit ändert, muss in äquidistanten Zeitabständen abgetastet werden.
  - Zeitdiskret: Kann fehlerfrei rekapituliert werden
  - Wie oft muss ein Signal abgetastet werden? – Doppelte der höchsten benötigten Frequenz. In der Praxis immer ein wenig mehr (Überabtastung), da ideale Bauteile (Tiefpass) nicht existieren.

## 5.3. Signalformen

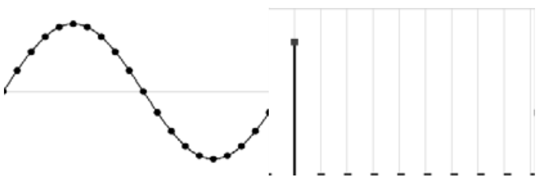
- Die Signalform hat Einfluss auf die erlebte Höhe (Ton) des Signals.
- Jedes Signal besteht aus einer Vielzahl von harmonischen Schwingungen (Sin / Cos)

### 5.3.1. Zeitbereich / Frequenzbereich

- Beide Darstellungen sind gleichwertig

### 5.3.2. Sinus

- Besitzt genau einen Peak

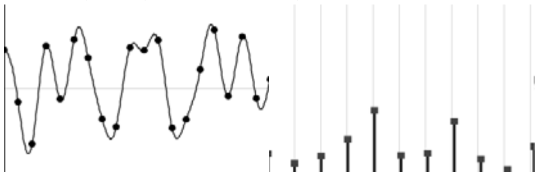


### 5.3.3. Rechteck

- Setzt sich aus mehreren harmonischen Signalen (sin/cos) zusammen
- Je mehr einzelne Sinus zusammen kommen, desto sauberer ist das Signal
  - Theoretisch sind unendlich Sinusse für eine sauberes Rechtecksignal nötig

### 5.3.4. Impuls

- Hat in einem begrenzten Bereich unendlich viele Peaks



#### 5.3.5. Vermischtes

- Was passiert mit der Frequenz, wenn die Amplitude vergrössert wird, jedoch per Obergrenze eingeschränkt wird?
  - Die Signalform ändert sich. Aus der harmonischen Sinus-Schwingung wird eine Rechteckschwingung. Es entsteht mehrere neue Frequenzen.

#### 5.3.6. Formeln

```
Frequenz
f = 1/T = 1 / Periodendauer
T = 1/f = 1 / Frequenz

Merke: Je grösser die Periodendauer, desto kleiner die Frequenz.
```

### 5.4. Tiefpass / Hochpass / Bandpass

- Ideale Pässe existieren nicht (Schema: Grenzfrequenz senkrecht nach unten)

#### 5.4.1. Tiefpass

- Lässt nur tiefe Frequenzen passieren
- Grundsatz: Jede Leitung ist ein Tiefpass
- Verursacht immer eine Verzögerung

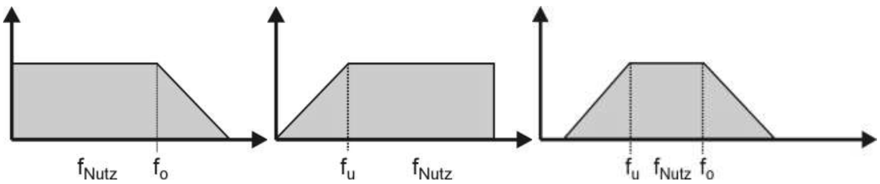
#### 5.4.2. Hochpass

- Lasst nur hohe Frequenzen passieren

#### 5.4.3. Bandpass

- Lässt Frequenzen in einem bestimmten Bereich passieren

#### 5.4.4. Grafiken




#### 5.4.5. Vermischtes

- Wenn wir bei einem TP eine eckige Frequenz eingeben, dann gehen einige Frequenzen im oberen Bereich verloren. Wenn wir nun die Frequenz dieses eckigen Signals erhöhen, dann nähert sich das Ausgangssignal (hinter dem TP) einer einzigen Sinus-Kurve, da alle anderen Signale gedämpft / gefiltert werden. Wenn wir mit der Frequenz noch höher gehen, geht auch die letzte Sinus-Kurve verloren.

### 5.5. Fourier

#### 5.5.1. Fourier-Analyse (84)

- Für periodische Signale
- Unendlich grosses Spektrum, aber diskret (einzelne Peaks)

```
Beispiel: Fourieranalyse eines Sägezahns

+++    ++++        G  = Generator, T=1mS, Signalform = Sägezahn /|/|/|
+G+----+TP+---- ?  TP = Tiefpass, fo = 3000Hz
+++    ++++

1) Frequenz des Generators = f = 1/T = 1/1000ms = 1kHz
2) Für das Sägezahnsignal lässt sich auf dem Formelblatt folgendes finden:
       y    = 2(sin(x)/1   - sin(2x)/2  + sin(3x)/3  - ...)
       y(t) = 2(sin(1*t)/1 - sin(2*t)/2 + sin(3*t)/3 - sin(4*t)/4)
3)Daraus kann jetzt der Frequenzbereich ermittelt werden, es gibt Peaks bei folgenden Werten: 1kHz = 2, 2kHz = -1, 3kHz = 2/3, …
4) Wegen TP wird alles über 3kHz abgeschnitten!
5) Aus den übrig gebliebenen Sinussen könnte jetzt wieder ein Signal gezeichnet werden. Ergibt eine sehr unsauberen Sägezahn.
```

### 5.5.2. Fourier-Transformation

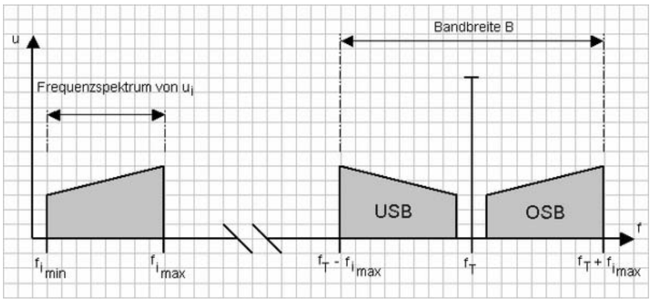
- Für nicht-periodische Signale
- Kontinuierliches Spektrum

## 5.6. Multiplikation von Signalen

- Nutzsignale: Träger von Informationen
- Hilfssignale: Träger der Nutzsignale
- Verschieben der Signale im Frequenzbereich -> Modulation (transponiert)
- Wiederherstellung der Nutzsignale: Multiplikation mit Trägersignal (Hilfssignal) und Tiefpass
- Frequenzmultiplexing: Mehrere Signale gleichzeitig in verschiedenen Frequenzbereichen. Wird beim Radio (AM = Amplitudenmodulation) verwendet. Niederfrequentes Nachrichtensignal wird auf einen Frequenzbereich gebracht, der sich für eine Abstrahlung eignet.

```
Nachrichtensignal * Trägersignal
cos(wn*t) * cos(wt*t)
0.5*(cos(wt-wn)*t + cos(wt+wn)*t)

Achtung: Unbedingt das 0.5 beachten! Amplitude ist nur noch halb so hoch!
```


- OSB = Oberes Seitenband = Regellage
- USB = Unteres Seitenband = Kehrlage
- OSB + USB = Bandbreite

## 5.7. Abtastung (87)

- In der Praxis wird immer überabgetastet, da der ideale TP nicht existiert.

```
Ideale Abtastung
ft = 2*fg = 2 * Obere Grenzfrequenz
```

```
Rechenbeispiel
Abtastung eines Signals: Die Quelle liefert einen Spektralbereich bis
30kHz. Relevant ist das Spektrum bis 20kHz. Entwerfen Sie das
Blockschaltbild der Abtastung mit TP, Abtaster, Quelle und Senke.

+++    ++++    ++++    ++++    +++
+Q+----+TP+----+AT+----+TP+----+S+
+++    ++++    ++++    ++++    +++

Q: 30kHz -> TP: 20kHz -> AT: 40kHz -> TP: 20kHz -> S: 20kHz

- Alle Werte gelten für ideale Bauteile.
- Der erst Tiefpass wird auch Alising-Filter genannt. Er schneidet den
unwichtigen Bereich (im Bsp.: über 20kHz) ab
```

## 5.8. Klirrfaktor (88)

### 5.8.1. Oberwellen

- Durch die Ecken, welche bei der Quantisierung entstehen, produzieren wir Oberwellen. Daraus entsteht ein grösseres Frequenzspektrum.

### 5.8.2. Klirrfaktor

```
Klirrfaktor = Signale / Noise |dB = 2^2w

w = Wortlänge, ACM: 12bit, CD: 16bit
=> Mit jedem Bit Halbierung des Rauschens
```

### 5.8.3. Quantisierungsrauschen

- Nur Abhängig von den Quantisierungsstufen

## 5.9. Kompandierung (90)

- Im höheren Leistungsbereich des Gehörs mehr Abstufungen
- Bei den lauten Tönten ist der Fehler grösser
- Unterschied zwischen Huffmann / LZ und Quantisierung?
  - Bei Huffmann / LZ gehen KEINE Informationen verloren
  - Bei der Quantisierung gehen „Informationen" verloren. Diese Informationen können jedoch Irrelevant sein, das hängt von den Anforderungen ab. Beispiel: Signale über 20kHz sind für Menschen sowieso unhörbar, also irrelevant. Diese werden durch einen beabsichtigten Fehler entfernt.

```
Beispiel: ISDN-Telefon (64kBit/s)

64kBit/s = 2*fg*Anzahl Quantisierungsstufen
64kBit/s = 2*4000Hz*8Bit

Beispiel 2: Stufenbreite berechnen
Abtastfrequenz: 100Hz
Amplitudenbereich: 0 <= U <= 1V

=> 1200Bit/s = 2*100*X -> X = 6 Bit

Quantisierungsstufen = 2^6 = 64 Stufen / Abtastung
Stufenbreite: 1V / 64 = 15.6mV / Stufe
```

# 6. Leitungscodierung

## 6.1. Anforderungen / Probleme (96)

- Welche Bitfolge wird erkannt?
  - Abhängig vom Takt. Ausserdem kann das Signal nur dann richtig zurückgewonnen werden, wenn der Takt PHASENRICHTIG beim Empfänger zugefügt wird.
- Es sollte ein hoher Gleichstromanteil vermieden werden, da diese schlecht für die elektronischen Bauteile (Spulen) sind: Bipolare Signale verwenden!
- Synchronisations-Bits: Zu Beginn eine Bitfolge senden, in welcher die Frequenz eingebaut ist. Dient zur Synchronisation von Frequenz und Phase.
  - Warum? – Nicht jede Leitung hat dieselbe Phasenverschiebung

## 6.2. Klassifizierung binärer Leitungscodes (97)

### 6.2.1. Polarität

```
Unipolares Signal        Bipolares Signal
| o o o     o            | o o     o
| | | | o o |            | | |     |
+-------------->         +-------------->
|                        |     | |
|                        |     o o
```

### 6.2.2. Impulsform

- RZ-Impuls = Return to Zero: Vor jedem Signal auf 0 zurück
- NRZ-Impuls = No Return to Zero: Kein Rücksprung auf 0

## 6.3. Beispiele

### 6.3.1. Unipolare RZ (98)

### 6.3.2. Bipolare NRZ (98)

### 6.3.3. NRZ-L (99)

- L = Level

### 6.3.4. NRZ-M (99)

- M = Mark, eine 1 ändert die Polarität

### 6.3.5. NRZ-S (99)

- S = Spache, eine 0 ändert die Polarität

### 6.3.6. Mancheste-Code (100)

- Codierung durch die Flanke

### 6.3.7. AMI-Code (101)

- Kein Gleichstromanteil
- Erlaubt Taktrückgewinnung
- Problem: Lange 0-Folgen
- pseudoternärer Code: Sieht 3-wertig aus, ist aber in Realität nur 2-wertig.

## 6.4. GFSK: Gaussian Frequence Shift Keying

- Die Daten werden als Frequenzwechsel in einem Trägersignal modelliert
- 2-Level GFSK: 2 Frequenzen zur Kodierung
- 4-Level GFSK: 4 Frequenzen zur Kodierung
- Beide Enden kennen die unterschiedlichen Frequenzen der Codierung

## 6.5. DPSK: Differential Phase Shift Keying

- Die Daten werden als Phasenwechsel, bzw. Phasensprung in einem Trägersignal modelliert
- Prinzip 2-DPSK:
  - 0 -> kein Phase-Shift
  - 1 -> Phase-Shift (Poduziert Oberwellen, welche als Impuls erkannt werden).