

## Model of information processing

- *Source* (Quelle): creates bit pattern with 0 and 1
- *Source encoder* (Quellencodierer): e.g. can be a compressor or an encryptor (source content is encoded)
- *Channel encoder* (Kanalcodierer): Als Kanalkodierung bezeichnet man das Verfahren, digitale Daten bei der Übertragung über gestörte Kanäle durch Hinzufügen von Redundanz gegen Übertragungsfehler zu schützen.
- *Modulator / Line encoder* (Line encoder): Modulators work on the physical layer and are responsible for adjustment so the signal can be transmitted over a specific medium. Line encoders the transmitted digital signals are in suitable channel code encoded
- *Sender*: e.g. Torax
- *Channel* (Kanal): e.g. MS Teams, Air
- *Receiver*: (Empfänger): e.g. the ears

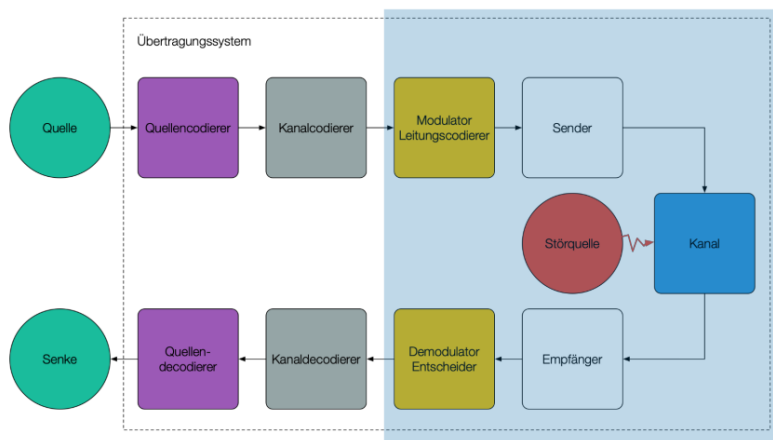


Figure 1: Model of information processing

## Probability TODO!

### Source and Sink

Jeden Quelle hat ein Zeichenvorrat. Nachricht wird kodiert und wird über Kanal übertragen. Senke hat auch ein Zeichenvorrat und muss

das übertragene Wort entscheiden und interpretieren. Dafür müssen die Zeichenvorräte der Quelle und Senke übereinstimmen.

- Wenn Zeichenvorrat ungleich: information irrelevant
- Wenn Zeichenvorrat:
  - wort redundant: vorhersagbar
  - wort nicht redundant: *Information*

Nachricht (Darstellung & Bedeutung)	redundant	nicht-redundant
irrelevant	Zeichenvorrat bei Quelle und Senke verschieden	
relevant	vorhersagbar	Information

Figure 2: Information, Nachricht

## Entscheidungsgehalt

Mass für die minimale Anzahl Entscheidungen die getroffen werden müssen, um ein Zeichen zu erkennen. Dies geschieht unter der Annahme dass alle Zeichen gleich Wahrscheinlich sind. Dies kann in einem Binary Tree abgebildet werden und entspricht der Höhe bzw. dem  $\log_2$  der Anzahl Blätter.

$$H_0 = \log_2(N) [\text{bit}] \quad (1)$$

Zeichenvorrat:  $X = \{a, b, c, d\}$ ,  $N = 4$  verschieden Zeichen. Daher ist der Informationsgehalt  $H_0 = \log_2(4) = 2$ . Das sieht man auch ihm Baum 3. Es müssen zwei Entscheidungen getroffen werden, um zu einem Zeichen zu gelangen.

## Entscheidungsfluss

Der Entscheidungsfluss ist der Entscheidungsgehalt im Verhältnis zu der Übertragungszeit eines Quellenzeichens.

$$H_0^* = \frac{\log_2(N)}{\tau} \left[ \frac{\text{bit}}{s} \right] \quad (2)$$

## Informationsgehalt eines Zeichens

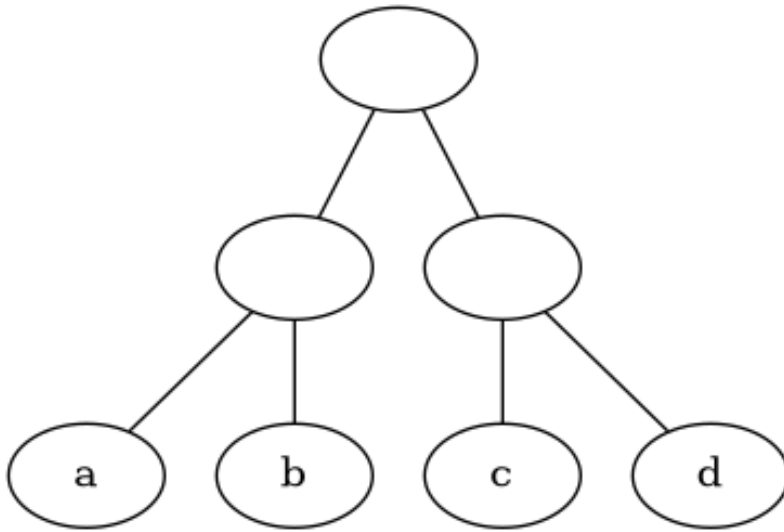


Figure 3: Entscheidungsgehalt

Der Informationsgehalt eines Zeichens sagt aus, wie viele Entscheidungen zu treffen sind, um das zu Zeichen zu bestimmen. Mit anderen Worten, wie viele mal muss ich zwischen Rechts / Links im Baum entscheiden, bis ich das Zeichen erreicht habe.

- $p(x_k)$  entspricht der Auftrittswahrscheinlichkeit des Zeichens  $x_k$

$$I(x_k) = \log_2\left(\frac{1}{p(x_k)}\right)[bit] \quad (3)$$

$$= -\log_2(p(x_k))[bit] \quad (4)$$

### Entropie

Als Entropie bezeichnet man den mittleren Informationsgehalt einer Quelle. Entropie beschreibt wie viele Entscheidungen die Quelle / Senke im Mittel pro Zeichen treffen muss. Ist also nichts anderes als der Mittelwert über den Informationsgehalt.

- $p(x_k)$ : Die Wahrscheinlichkeit des Zeichens  $x_k$

- $I(x_k)$ : Der Informationsgehalt des Zeichens  $x_k$

$$H(X) = \sum_{k=1}^N p(x_k) \cdot I(x_k) \quad (5)$$

$$= \sum_{k=1}^N p(x_k) \cdot (-1) \log_2(p(x_k)) \quad (6)$$

### Redundanz der Quelle

Wenn die Auftrittswahrscheinlichkeit aller Zeichen gleich ist, ist der Informationsgehalt am grössten, bzw. die Redundanz am kleinsten. Die Redundanz der Quelle ist die Differenz zwischen Entscheidungsgehalt und der Entropie.

- $H_0$  ist der Entscheidungsgehalt
- $H(X)$  ist die Entropie

$$R_Q = H_0 - H(X) \quad (7)$$

### Kleinste Redundanz

*Annahme:* alle Zeichen sind gleich wahrscheinlich (a,b,c,d). Dann würde sich ein Baum wie in 4 aufspannen. Dieser ist der selbe wie im Entscheidungsgehalt. Folglich wird  $R_Q = x - x = 0$  gerechnet.

### Codewortlänge

Die tatsächliche Codewortlänge muss eine Ganzzahl sein, weil man keine Halbenbits übertragen kann. Die Codewortlänge für ein Zeichen lässt sich wie folgt berechnen:

- $I(x_k)$  entspricht dem Informationsgehalt.

$$L(x_k) = \lceil I(x_k) \rceil \quad (8)$$

$$= \lceil \log_2\left(\frac{1}{p(x_k)}\right) \rceil \quad (9)$$

$$= \lceil -\log_2(p(x_k)) \rceil \quad (10)$$

### Entropie eines Codes

Die Entropie des Codes entspricht dem Mittelwert der Codewortlänge:

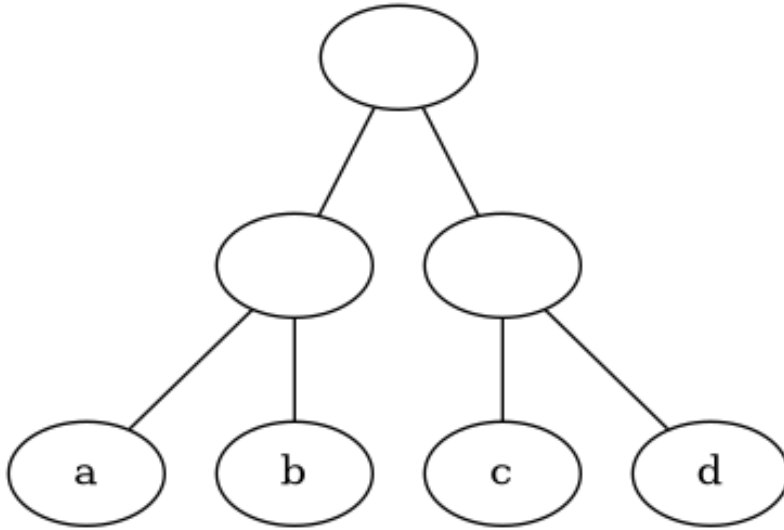


Figure 4: Tree mit gleichmässiger Auftrittswahrscheinlichkeit

- $L(x_k)$  entspricht der Codewortlänge

$$H_C(X) = \sum_{k=1}^N p(x_k) \cdot L(x_k) \left[ \frac{\text{bit}}{\text{Zeichen}} \right] \quad (11)$$

### Shannonsches Codierungstheorem

Für jede beliebige Binärcodierung mit Präfixeigenschaft ist die mittlere Codewortlänge nicht kleiner als die Entropie:

$$H(X) \leq L(X) \quad (12)$$

Für jede beliebige Quelle kann eine Binärcodierung gefunden werden, so dass die folgende Ungleichung gilt:

$$H(X) \leq L(X) \leq H(X) + 1 \quad (13)$$

### Redundanz des Codes

Redunanz des Codes:

- $L(x_k)$  entspricht der Codewortlänge
- $H(X)$  entspricht der Entropie

$$R_C = L - H(X) \quad (14)$$

### Quelle ohne Gedächtnis

Bei einer Quelle ohne Gedächtnis ist die Auftrittswahrscheinlichkeit eines Zeichens unabhängig davon, welche Zeichen zuvor generiert wurde. Die mittlere Entropie einer Quelle ohne Gedächtnis ist stets grösser oder gleich der Entropie einer Quelle mit Gedächtnis.

Die Verbundwahrscheinlichkeit ist für zwei Zeichen  $x_i$  und  $y_k$ :

$$p(x_i, y_k) = p(x_i) \cdot p(y_k) \quad (15)$$

### Gedächtnisbehaftete Quelle

Eine Quelle mit Gedächtnis merkt sich, welches Zeichen bereits generiert wurde. Dadurch ist die Verbundwahrscheinlichkeit wie folgt definiert:

$$p(x_i, y_k) = p(x_i) \cdot p(y_k) \quad (16)$$

Die Verbundentropie ist wie folgt:

$$H(X, Y) = H(X) + H(X|Y) \quad (17)$$

Die mittlere Entropie einer Quelle ohne Gedächtnis ist stets grösser oder gleich der Entropie einer Quelle mit Gedächtnis.

### Verbundentropie

$$H(X, Y) = \sum_{i=1}^N \sum_{k=1}^N p(x_i, y_k) \cdot -\log_2(p(x_i, y_k)) \quad (18)$$

$$H(X, Y) = H(X) - H(Y|X) \quad (19)$$

### Präfixfreier Code

Kein Codewort des Codes ist Präfix eines anderen Codewortes. Anders ausgedrückt darf kein Codewort den Beginn eines anderen Codewortes darstellen. Ein Code zum Beispiel mit den Codewörtern  $\{0, 10, 11\}$  erfüllt die Präfix-Eigenschaft, während hingegen der Code mit den Codewörtern  $\{0, 01, 10\}$  sie nicht erfüllt, da „0“ Präfix von „01“ ist.

– Wikipedia, 2022

## Typen Source Encoder

In the information theory exists two types

- data compressor
  - lossless (verlustfrei)
  - lossy (verlustbehaftet)
- encryption
  - symmetric
  - asymmetric

## Verlustfreie Kompression

The goal of a compression is make the data transfer and storing as efficient as possible. This is achieved with reducing the redundancy and remove not used elements.

A good compression algorithm has a:

- high encode / decode speed
- low requirements for hardware

There exists two kind of lossless compression:

- static procedure (adaptive): character of the data used
- dynamic procedure: character of the data is **not** used. In a worst case scenario the data are bigger with compression

## Huffman Code

Procedure:

1. sort the symbols after possibility of occurrence

2. repeat

- sum up the possibilities of the two least possible symbols
- the less possible symbol got the 1 the other the 0
- this two counts now as one symbol
- repeat this until only every symbols is processed

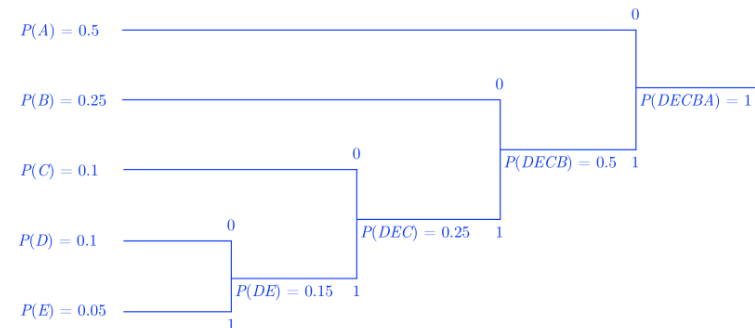


Figure 5: Huffman Example

## Lauf­längen­kodierung

The Run Length Encoding RLE (de. Lauflängenkodierung) is a Lossless compression. If a symbol occurs more than one in a row then the number of repetition is prefix with the character appended.

Quelltext w: Agggbbbehfffggggg => |w| = 15

Codiert w\_c: A3g2beh4f5g => |w\_c| = 11

This compression is very interesting for binary because the transmission is always just a repetition of 0 and 1. First both parts has to agree with witch number the transfer starts. After that only the number of 0 or 1 is transmitted

0000 0111 => 5 3

## Lempel-Ziv Algorithmus

During the encoding (runtime) a code book is created on the fly. If a pattern is already transmitted only the reference to this pattern is transmitted.

Der zu komprimierende Code hat wiederkehrende Mustern oder PHrasen, nur die codierten Phrasen müssen übertragen werden. Während des Durchlaufens der Daten wird ein ständig wachsender Baum erzeugt. Baum dient als Wörterbuch. Knoten dienen als Referenz.

The algorithm needs:

- a search buffer (the already encoded symbols)
- a look-ahead buffer (shows the next symbols to encode)
- if a sequence of symbols in the look-ahead buffer and the search buffer are equal
  - then a code consisting of position and length and stored
  - else the code is stored as in the look-ahead buffer.

search buffer	Look-ahead buffer	coding
	sir_sid_eastman	(0,0,"s")
s	ir_sid_eastman	(0,0,"")
si	r_sid_eastman	(0,0,"r")
sir	_sid_eastman	(0,0,"_")
sir_	sid_eastman	(4,2,"d")
sir_sid	_eastman	(4,1,"e")

Figure 6: Lempel-Ziv Example

### Transposition Cipher

In a transposition cipher every symbol stays the same (no substitution used). Only the order of the symbols is changed. A simple example is shown below:

The clear text "Hello my World" is inserted in a table by row:

H	E	L	L
O	M	Y	W
O	R	L	D

and now read the text by column: HOOEMRLYLLWD

### DES

DES is a symmetric key algorithm and was long the standard for data encryption. Today it is not safe anymore because of its short key length (max. 56 bits) and therefore should not be used anymore.

### Inverse einer Zahl

In math a number  $x$  is called the inverse of  $y$  when  $x \cdot y = 1$ . In general this is the case when  $y = x^{-1}$ . Interesting is when modulo is used. Because now you can multiple  $a$  with  $b \neq a^{-1}$  to get -1

$$a \cdot b \mod c \equiv 1 \quad (20)$$

### Euler Function

The Euler function returns for a given  $n$  the amount of part-unrelated (de: teilerfremd) numbers  $n$ . The symbol for the Euler function is  $\Phi$ . The Euler function is difficult to calculate except for prime numbers: Be  $n$  a prime number

$$\Phi(n) = n - 1 \quad \Phi(5) = 5 - 1 = 4 \quad (21)$$

The result of  $\phi(p \cdot q)$  where  $p$  and  $q$  are prime is:

$$\Phi(p \cdot q) = \Phi(p) \cdot \Phi(q) \quad (22)$$

$$= (p - 1) \cdot (q - 1) \quad (23)$$

### Euler Theorem

For two part-unrelated numbers  $a < b$  applies the following:

$$a^{\Phi(b)} \mod b \equiv 1 \quad (24)$$

$\Phi(b)$  is called Eulers function.

### Euler Function & Theorem

If you combine Euler's Function with his theorem you can say something like this: If  $b = p \cdot q$  where  $p$  and  $q$  are prime then:

$$a^{\Phi(b)} \mod b \equiv 1 \quad (25)$$

$$a^{\Phi(p \cdot q)} \mod (p \cdot q) \equiv 1 \quad (26)$$

$$a^{(p-1) \cdot (q-1)} \mod (p \cdot q) \equiv 1 \quad (27)$$

### RSA Parameters

1. Choose  $p$  and  $q$ 
  - (a)  $n = p_1 \cdot p_2$
  - (b)  $n$  is public,  $p, q$  private
2. Calculate  $\Phi(n)$
3. Choose  $e$ 
  - $e$  has to be part-unrelated to  $\Phi(n)$  and  $1 < e < \Phi(n)$
  - in reality often 3 or 65537
4. Calculate the inverse from  $e$  resulting in  $d$ 
  - For Example with the Extended Euclidean algorithm

Be  $m$  the message. Then to encrypt:  $c = m^e \bmod n$ . Do decrypt  $c$  to  $m$ :  $m = c^d \bmod n$ .

### Channel Matrix

Die Kanalmatrix beschreibt wie Wahrscheinlich es ist, dass man  $y$  erhält, wenn  $x$  gesendet wurde.  $p$  und  $q$  geben an, wie wahrscheinlich es ist, dass ein Fehler auftritt.

$$p(Y|X) = \begin{bmatrix} \sum = 1 \\ \sum = 1 \end{bmatrix} = \begin{bmatrix} p & 1-p \\ 1-q & q \end{bmatrix} = \begin{bmatrix} p(y_1|x_1) & \cdots & p(y_n|x_1) \\ \vdots & \ddots & \vdots \\ p(y_1|x_m) & \cdots & p(y_n|x_m) \end{bmatrix} \quad (28)$$

$$\begin{bmatrix} p(y_1) \\ p(y_2) \end{bmatrix} = \begin{bmatrix} p(x_1) \cdot p(y_1|x_1) + p(x_2) \cdot p(y_1|x_2) \\ p(x_1) \cdot p(y_2|x_1) + p(x_2) \cdot p(y_2|x_2) \end{bmatrix} \quad (29)$$

### Maximum Likelihood

How high is the possibility that I receive  $y_1$  when  $x_1$  is sent. The cell with the highest value in a column (y-value) is maximum likelihood the x-value which was sent.

$$p(Y|X) = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.6 \\ 0.3 & 0.2 & 0.5 \end{bmatrix} \quad (30)$$

$20 \cdot 6 \bmod 43 = 2$   
 $43 = 21 \cdot 2 + 1$   
 $2 = 2 \cdot 1 + 0$   
 $1 = 43 - 21 \cdot 2$   
 $1 = 43 - 21 \cdot (20 - 6 \cdot 43)$   
 $1 = 43 - 21 \cdot 20 + 126 \cdot 43$   
 $1 = 127 \cdot 43 - 21 \cdot 20$   
 $1 = 7 \cdot 43 - 15 \cdot 20$   
 $(7 \cdot 43 - 15 \cdot 20) \bmod 43 = 1$   
 $-15 \cdot 20 \bmod 43 = 1 \Rightarrow -15 + 43 = 28 = d$

Figure 7: RSA nach Rinkel

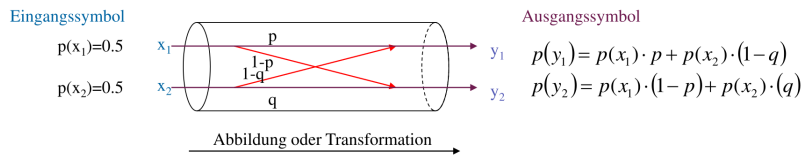


Figure 8: Channel Matrix Model

When I got a  $y_1$  (first column in 30) then it probably  $x_1$  was sent. When I got a  $y_3$  then probably a  $x_2$  was sent. And to cover all possible  $x$  values the  $y_2$  is mapped to  $x_2$ .

## Mutual Information / Transinformation

Mutual information describes how much two random variables correlate. The higher the more they correlate (1 total correlation, 0 no correlation).

In the context of information transfer over a channel the mutual information describes also how big the flow is which I can transfer without errors. Transinformation (T) ist der Informationsfluss, den ich fehlerfrei über einen gestörten Kanal übertragen kann.

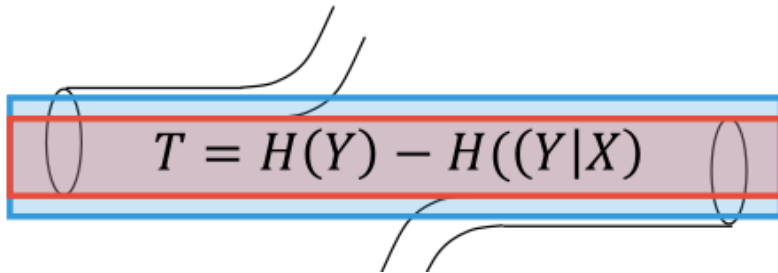


Figure 9: Transinformation / Mutual Information

$$H(Y|X) = - \sum_i^n \sum_j^n p(x_i) \cdot p(y_j|x_i) \cdot \log_2(p(y_j|x_i)) \quad (31)$$

$$T = H(Y) - H(Y|X) = H(X) - H(X|Y) \quad (32)$$

## Wie ändert die Kanalmatrix die Transinformation

Olivier Lischer

Ein nicht gestörter Kanal (Einheitsmatrix) überträgt den mittleren Informationsgehalt. Transinformation wird nur durch die Quelle bestimmt. Verändert sich die Entropie der Quelle, ändert sich die Transinformation. Nimmt die Fehlerwahrscheinlichkeit zu, nimmt die Transinformation ab. Sind in der Kanalmatrix alle Positionen gleich besetzt, wird die Transinformation  $T = 0$ .

## Coderaum

The Code space is a set of code words / code points.

## Hamming Distance

When one code word is given the Hamming Distance ( $h$ ) describes how many positions have to change until a new valid code word is reached.

- Number of reliable detectable errors (33)
- Number of reliable correctable errors (34)

$$e^* = h - 1 \quad (33)$$

$$e = \begin{cases} \frac{h-2}{2} & \text{if } h \text{ is even} \\ \frac{h-1}{2} & \text{if } h \text{ is odd} \end{cases} \quad (34)$$

## Korrigierkugel

The correction sphere has a radius of  $e$ : the number of reliable correctable errors. The center of a correction sphere is always a *valid* code word. If you receive an invalid code word and this code word is inside of the sphere then it will be changed to the center / valid code word.

## Perfekter Code / Dichtgepackter Code

A code space is called a perfect code when every code word (valid or invalid) is in a correction circle.

- $n$ : The dimension of the code (number of all code words (CW) =  $2^n$ )
- $m$ : The dimension of message (number of all valid CW =  $2^m$ )
- $k$ : The dimension of check positions ( $n = m + k$ )

Then applies:

$$2^m \cdot \sum_{w=0}^e \binom{n}{w} \leq 2^n \quad (35)$$

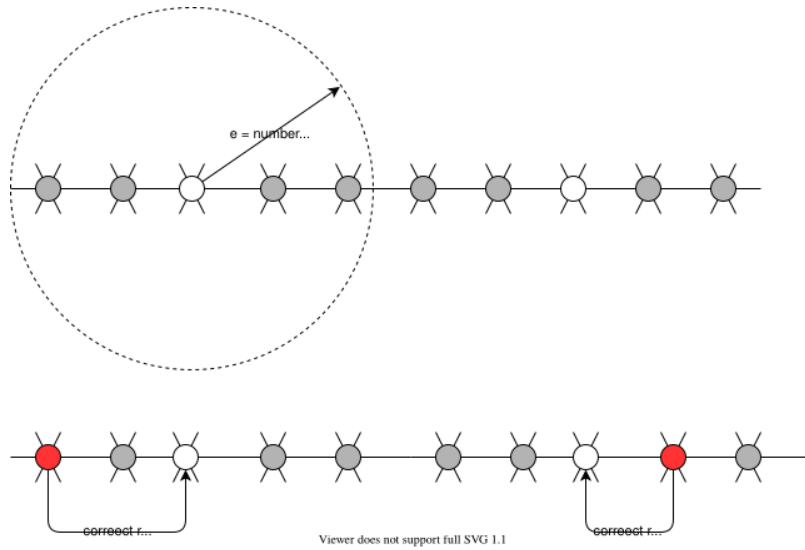


Figure 10: Korrigierkugel / Correction Sphere

- $2^m$ : Number of valid CW (or number correction circles)
- $\sum$ : Number of CW per correction circle
- $2^n$ : Number of all CW

In the following case we call the code perfect:

$$2^m \cdot \sum_{w=0}^e \binom{n}{w} = 2^n \quad (36)$$

### Hamming Codes

Hamming Code is a family of error correcting codes. Every Hamming Code has a Hamming Distance of 3.

Hamming Codes can be built with either a generator matrix or with a Generator Polynomial.

### Generator Matrix Hamming Code

The generator matrix consists of two parts. The first indicates which you need to pick for your calculation (this with 1). And the second part is the unit matrix which marks the control positions.

A generator matrix has two conditions:

1. every vector of the message part has to be different than the others
2. the equation 37 has to be fullfield

$$\sum_i x_i \cdot \vec{P}_i \equiv \vec{0} \mod 2 \quad (37)$$

An example of a valid generator matrix is shown in 38

$$\left[ \begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array} \right] \quad (38)$$

If only one bit is changed during transmission then the calculation of 37 is wrong. The result is the vector which has changed.

### Generator Polynomial Hamming Code

The generator matrix can be replaced with a polynomial. This has the benefits that it is easier to calculate the control positions.

- $k$ : Number of control positions
- $n$ : Dimension of the code

Where  $g_i = 0, 1$

$$G(u) = \sum_{i=0}^k g_i \cdot u^i \quad (39)$$

$$X(u) = \sum_{i=0}^n g_i \cdot u^i \quad (40)$$

If the code word polynomial (40) can be divided by the generator polynomial (39) mod 2 it is a valid code:

$$Q(u) = X(u)/G(u) \quad (41)$$

$$X(u)/G(u) \equiv Q(u) \mod 2 \equiv 0 \quad (42)$$

To verify if received message is correct divide the message through the generator polynomial. If this is not 0 then an error is occurred.

$$\begin{cases} \text{No errors} & \text{if Message}/G(u) = 0 \\ \text{Error(s) occurred} & \text{if Message}/G(u) \neq 0 \end{cases} \quad (43)$$



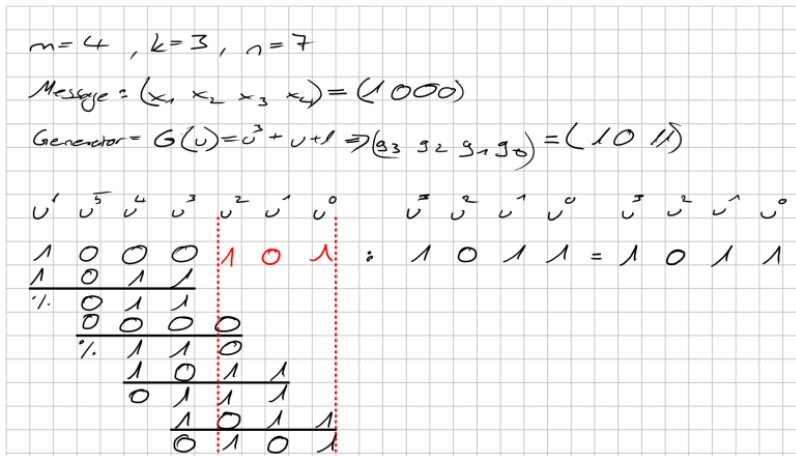


Figure 11: Generator Polynom Hamming Code

### Was ist ein Blockcode

Your data is grouped in blocks and then each block is encoded separately. Block codes are easy to implement and have a high error detection. But you have to build blocks. Therefore, streams are not possible to encode.

### Faltungscodes / Convolutional Codes

With Convolutional codes streaming data (code block length not known) can be encoded. This kind of code is often used in the mobile network. Convolutional codes do not need any block synchronization.

### How does Convolutional Codes work

A Convolutional codes has storage cells for the previous send bits. The highets bits in the generator polynom tells you how many Storage Cells you have. The bit from the past is *xor-ed* with the current one.

In the figure 12 the upper way is the  $g_1$  and the lower  $g_2$ . If a bit was used it shift one further to the storage cell.

The Generator Polynom can be read from the diagram:

- $g_1(x) = 1 + x^2$ : The upper way
- $g_2(x) = 1 + x + x^2$ : The bottom way

The number of tails bits are the same as the number of storage cells. The encoder in figure 12 has two storage cells. Therefore two tailbits

are needed.

Table 1: Output for input 100

$x_i$	$g_1(x_i)$	$g_2(x_i)$	output
1	1	1	11
0	0	1	01
0	1	1	11
0	0	0	00
0	0	0	00

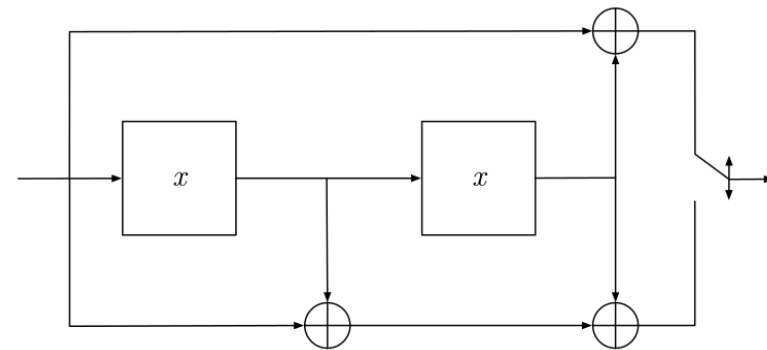


Figure 12: Faltungscode Encoder

### State Machine for Convolutional Codes

A Convolutional Code can also be represented as a State Machine Diagram. For an example see figure 13. The Generator Polynom  $g_1(x)$  and  $g_2(x)$  are as follows:

- $g_1(x) = 1 + x^2$
- $g_2(x) = 1 + x + x^2$

1. Find the numbers of Storage cells  $n$
2. Calculate the number of state  $m = 2^n$
3. Build the table (2)
4. Build from table the State Machine Diagram / DFA

Table 2: Table of States for Code

State	Input	Cells	Output	Next State
$S_0$	0	0 0	0 0	$S_0$
$S_0$	1	0 0	1 1	$S_2$
$S_1$	0	0 1	1 1	$S_0$
$S_1$	1	0 1	0 0	$S_2$
$S_2$	0	1 0	0 1	$S_1$
$S_2$	1	1 0	1 0	$S_3$
$S_3$	0	1 1	1 0	$S_1$
$S_3$	1	1 1	0 1	$S_3$

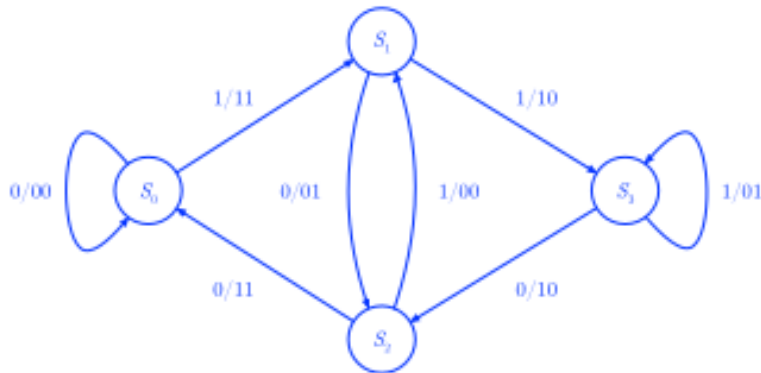


Figure 13: Convolutional Code as State Machine Diagram

### Trellis Diagram

To visualize how the convolutional code is encoded over time the trellis diagram is suitable. Each state of the state machine is only used once per CPU clock. And the transition goes to the next state in the next CPU clock.

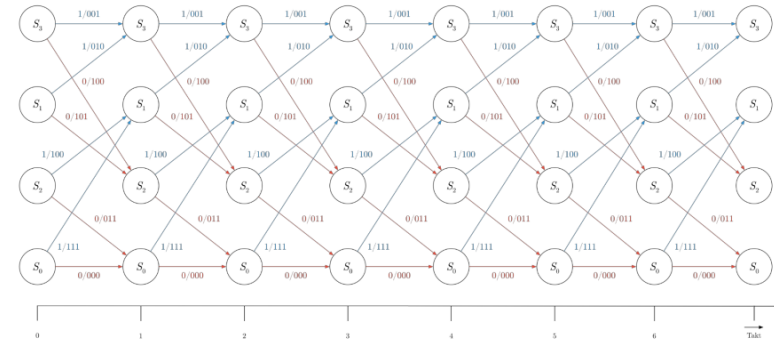


Figure 14: Trellis diagram

### Decode Convolutional Codes

To decode a Convolutional Code you try to find the cheapest way through the trellis diagram. You start at the Starting State ( $S_0$ ). You compare each code word part with the possible transitions. The transition which needs the fewest changes is used.

### Block Code Rate

- $m$ : Encoding Storage
- $R$ : Block code rate
- $x$ : Encoded Bits

$$R = \frac{\text{encodededbits}}{\text{sentbits}} \quad (44)$$

$$= \frac{x}{2 \cdot (x + m)} \quad (45)$$

In 45 the 2 is needed because for each input bit two output bits are generated. The  $m$  is required because you need to send  $m$  tail bits to clear the storage.

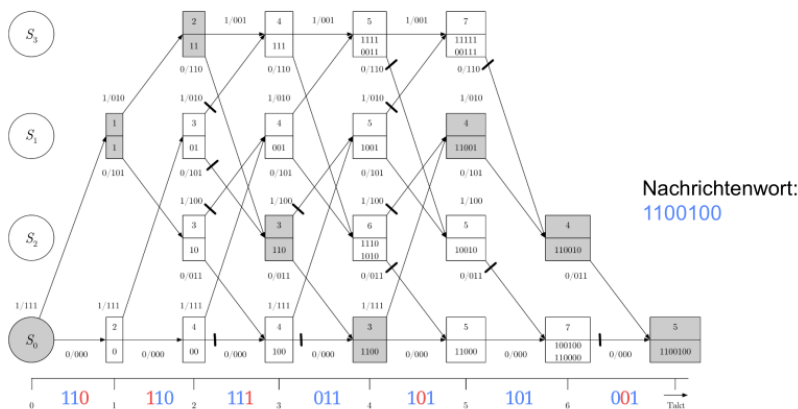


Figure 15: Decoding a Convolutional Code

### Impulse Response

The Impulse Response is the output of a system (for example an encoder) with a brief input signal. In numbers this would be 1000...

### Detect Number of Storage Cells

Using a Impulse Response you can detect how many storage cells are used in the convolutional code. The 1 which is send must be cleared from all storage cells. Therefore, the same number of 0 are sent as storage cells.

### Was ist das Basisband

Baseband is the range of frequencies occupied by a signal that has not been modulated to other frequencies. The frequency is unchanged.

### Line Coding TODO!!

### Requirements Line Coding

- Synchronisation of the clock / phase at the receiver After a long period of 1 you have to know how many 1 are sent
- Avoiding direct current components (Gleichstromkomponenten)
- Resistance to interference

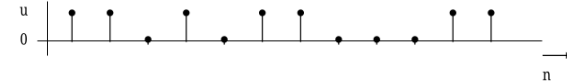
### Classification Line Coding

Line codes can be classified using two properties:

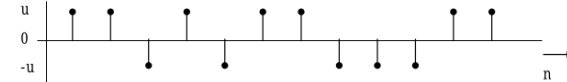
1. polarity (0 to  $u$  /  $-u$  to  $u$ )
2. impuls form (Return to Zero / No Return to Zero)

#### 1. Polarität

unipolares Signalv

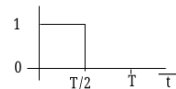


bipolares Signal



#### 2. Impulsform

RZ-Implus (Return to Zero)



NRZ-Implus (no return to zero)

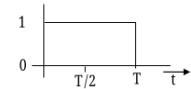


Figure 16: Line Code Classification

### Manchester Code

The Manchester Code (see figure 17) has no direct current (Gleichstrom) but requires the double amount of bandwidth. The Manchester Code by IEEE has the following convention:

- the 0 is represented as a high-low signal sequence
- the 1 is represented as a low-high signal sequence

### AMI Code

The AMI Code uses the following convention:

- the 0 is represented as a 0 level
- the 1 is represented as an alternating level change of pos. and neg.

After a long sequence of 0 it is difficult for the receiver to know how many zeros actually were send.

### Parameter of a signal

A signal consists of two parameters:

1. amplitude

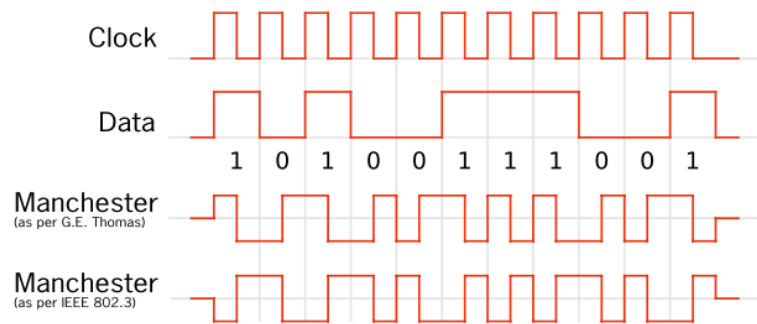


Figure 17: Machester Code

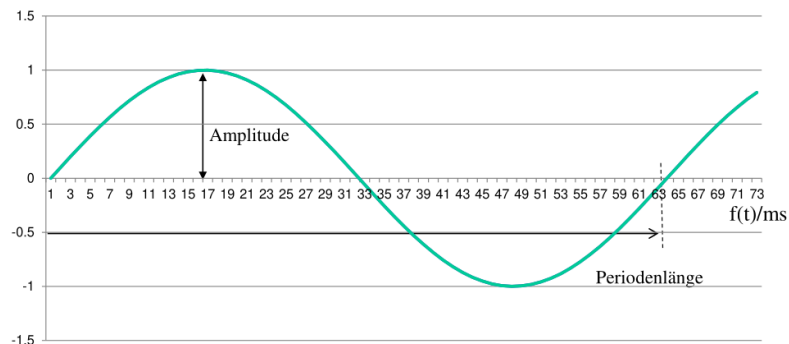


Figure 18: A Signal

## 2. period length / frequency (Period Length vs. Frequency)

### What are dB?

Bel describes the ratio between two values. For example the strength of an output and input signal.

$$S_{Bel} = \log_{10}\left(\frac{S_{out}}{S_{in}}\right) \quad (46)$$

The dezibel is more common than bel.

$$S_{dezibel} = 10 \cdot \log_{10}\left(\frac{S_{out}}{S_{in}}\right) \quad (47)$$

*Example*

$$s_2 = 20, s_1 = 10 \quad (48)$$

$$\log_{10}\left(\frac{s_2}{s_1}\right) \quad (49)$$

$$= \log_{10}\left(\frac{20}{10}\right) \quad (50)$$

$$= 0.301 \text{ Bel} \approx 3 \text{ dB} \quad (51)$$

### Harmonische Schwingungen / Harmonic Oscillations

Harmonic Oscillations are oscillations which does not have any sudden changes. For example sine / cosine are harmonic.

In our world, all things consist of sine and cosine waves.

### Week 09 / 10 Signal Abtasten

Je öfter abgetaste, desto genauer ist das Signal:

- um es fehlerfrei zu erhalte, muss man mit min. der doppelten Frequenz als das signal abgetastet werden
- Mensch hört ca. 20 - 22kHz

Wie viele bits verwenden wir für die quantisierung des signals? Je mehr bits desto genauer. Aber das signal wird **nie** rund. Es ist immer kantig