8. Grundlagen der Informationstheorie

8.1 Informationsgehalt, Entropie, Redundanz

Def.: Sei Σ eine Menge von Zeichen. Die Menge Σ^* aller Zeichenketten (Wörter) über Σ ist die kleinste Menge, für die gilt:

1. die leere Zeichenkette ε ist Element von Σ^*,
2. falls $a \in \Sigma$ und $w \in \Sigma^*$, so ist $aw \in \Sigma^*$.

aw bezeichnet man auch als Konkatenation (Hintereinanderschreibung) von a und w.

Nachricht: Folge von Zeichen, die von Sender (Quelle) an Empfänger (Senke) übermittelt wird. Also: Nachricht Element von Σ^* für ein Alphabet Σ.

Information: intuitiv: was durch eine Nachricht übermittelt wird, Bedeutung der Nachricht

Informationstheorie nach Shannon: Versuch, Begriff der Information rein statistisch zu erfassen

(statistischer) Informationsgehalt I(x) einer Nachricht x (nach Shannon) intuitiv:
Anzahl der Bits, die notwendig sind, um Nachricht in bzgl. Wortlänge optimalem Code binär zu codieren
hängt ab von Auftrittswahrscheinlichkeit der Zeichen
(hier Wahrscheinlichkeit = relative Häufigkeit)

Seien A und B Ereignisse, für die Wahrscheinlichkeit von A und B (w(A) bzw. w(B)) gilt:

1. $0 \leq w(A) \leq 1$
2. $w(A) = 1$ falls A sicher
3. $w(A \text{ oder } B) = w(A) + w(B)$ falls A und B sich ausschließen

daraus ableitbar etwa: $w(\text{nicht } A) = 1 - w(A)$, $w(A \text{ und } B) = w(A)\, w(B)$ falls A und B unabhängig, ...

Forderungen an Informationsgehalt:

1. Je seltener Zeichen auftritt, desto höher soll sein Informationsgehalt sein
2. Der Informationsgehalt einer Zeichenkette soll sich aus Summe der Informationsgehalte der Zeichen ergeben: $I(x1...xn) = I(x1) + ... + I(xn)$
3. Der Informationsgehalt eines absolut sicheren Zeichens ist 0.

Logarithmus einfachste Funktion, durch die diese Bedingungen erfüllt werden können, also definiert man für Zeichen x:

$$I(x) = log_2 \frac{1}{w(x)} [bit]$$

(= $- \log_2 w(x)$) Informationsgehalt einer Nachricht ergibt sich aus 2.

Beispiele:

a) Sendet Quelle immer dasselbe Zeichen x, so ist w(x)=1 und damit I(x) = 0

b) Im binären Fall gilt, unter Voraussetzung w(0) = 0.5 und w(1) = 0.5:
I(0) = I(1) = 1. Informationsgehalt ist die Länge der übertragenen Zeichenkette

c) Gibt es 2^k Symbole gleicher Wahrscheinlichkeit, so ist I(s) = k für jedes Symbol s.

d) In deutschen Texten tritt b mit Wahrscheinlichkeit 0.016 auf, also

$$I(b) = log_2 \frac{1}{0.016} = \frac{\log \frac{1}{0.016}}{\log(2)} = 5.79$$

Bemerkung: $\log_b(x) = \log_{10}(x) / \log_{10}(b)$

Entropie:
mittlerer Informationsgehalt eines Zeichens einer Quelle oder Nachricht
(abstrakt gesehen dasselbe, was man braucht ist Alphabet + Wahrsch. der Symbole)
erhält man durch Aufsummieren aller Informationsgehalte der Zeichen des Alphabets, gewichtet mit der Wahrscheinlichkeit der Zeichen:
(Name wegen Analogie zur Thermodynamik)

Def.: Seien $\Sigma = \{x1,...,xn\}$ ein Alphabet und w eine Wahrscheinlichkeitsverteilung über Σ. Die Entropie H von Σ und w ist wie folgt definiert:

$$H = \sum_{i=1}^{n} w(x_i) ld \frac{1}{w(x_i)} = \sum_{i=1}^{n} w(x_i) I(x_i)$$

Entropie am größten, wenn alle Wahrscheinlichkeiten der Zeichen gleich sind.

Beispiele:

1 Zeichen a mit w(a) = 1:	H = 0
2 Zeichen a,b, w(a) = w(b) = 0.5	H = 0.5 * 1 + 0.5 * 1 = 1
4 Zeichen a,b,c,d mit gleicher W. 0.25	H = 2
4 Zeichen a,b,c, d mit w(a) = .5, w(b) = 0.25, w(c) = w(d) = 0.125	H = 1/2 * 1 + 1/4 * 2 + 1/4 * 3 = 1.75

Def.: Seien A, B Alphabete.
Eine Codierung C von A in B ist eine injektive Abbildung C: A -> B*.

Binärcodierung: |B| = 2, meist B = {0,1}

mittlere Wortlänge eines Codes C für A = {a1, ..., an} und Wahrscheinlichkeitsverteilung w für A :

$$L = \sum_{i=1}^{n} w(a_i) l_i$$

wobei l_i Länge von $C(a_i)$.

Shannon'sches Codierungstheorem:
Seien $\Sigma = \{x1,...,xn\}$ ein Alphabet, w eine Wahrscheinlichkeitsverteilung über Σ und H die Entropie von Σ und w. Sei C eine Binärcodierung von Σ und L ihre mittlere Wortlänge. Es gilt:

$$H \leq L$$

(Gleichheit kriegt man hin, wenn Codierungen von Zeichengruppen zugelassen: C: A* -> B*)

intuitiv: man kann für eine Nachricht mit Entropie H keine Binärcodierung finden, die mit weniger als H Bits pro Zeichen (im Schnitt) auskommt.

Code-Redundanz: R = L - H

gesucht oft: Code mit möglichst wenig Redundanz
(andererseits erhöht Redundanz natürlich Übertragungssicherheit)

Beispiel:

s	w(s)	I(s)
a	1/2	1
b	1/4	2
c	1/8	3
d	1/8	3

Entropie: 1/2 * 1 + 1/4 * 2 + 1/8 * 3 + 1/8 * 3 = 1.75

<u>Codierung 1:</u>

a -> 00
b -> 01
c -> 10
d -> 11

Code fester Wortlänge

mittlere Wortlänge: 1/2 * 2 + 1/4 * 2 + 1/8 * 2 + 1/8 * 2 = 2

Redundanz: 2 - 1.75 = 0.25

<u>Codierung 2:</u>

a -> 1
b -> 01
c -> 000
d -> 001

Code variabler Wortlänge
wichtig: Fano-Eigenschaft: kein Codewort ist Anfangsstück eines anderen

mittlere Wortlänge:

1/2 * 1 + 1/4 * 2 + 1/8 * 3 + 1/8 * 3 = 1.75

damit Redundanz 0, also bezogen auf die mittlere Wortlänge optimaler Code

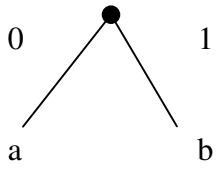
8.2 Berechnung optimaler Binärcodes nach Huffman

Def.: Ein Binärcode *C* für ein Alphabet $\sum$ ist eine injektive Abbildung:

$$C : \Sigma \to \{0,1\}^*$$

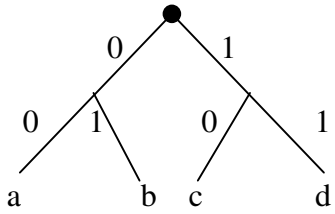
Binärcodes lassen sich durch Code-Bäume repräsentieren:

(a) $\Sigma = \{a,b\}$, $C(a) = 0$, $C(b) = 1$



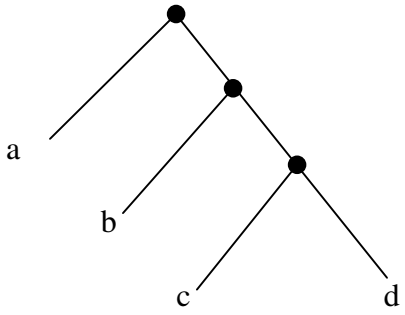
b)

$$\sum = \{a,b,c,d\}, A(a) = 00$$
$$A(b) = 01$$
$$A(c) = 10$$
$$A(d) = 11$$



c)

$$\sum = \{a,b,c,d\}, B(a) = 0$$
$$B(b) = 10$$
$$B(c) = 110$$
$$B(d) = 111$$



<u>Repräsentation:</u> Jeder Binärcode für $\sum$ kann als Binärbaum mit Symbolen aus $\sum$ an den Knoten repräsentiert werden.

Definition:

$a_1 \ldots a_n \in \sum^*$ heißt Präfix von $b_1 \ldots b_l \in \sum^*$, falls
$n \le l$ und $b_i = a_i$ für alle $i \le n$.

Definition:

Ein Binärcode $C : \sum \to \{0,1\}^*$ heißt Präfixcode, falls kein $C(a_i)$ Präfix eines $C(a_j)$ mit $i \neq j$ ist.

Vorteil von Präfixcodes: 0-1-Folgen können eindeutig dekodiert werden.
In Beispiel (c) codiert 01100011110 das Wort a c a a d b.
Entsprechen Code-Bäumen, die Symbole aus $\sum$ nur an den Blättern haben!

Was sind "gute" Präfixcodes ?

$(\sum,w)$ sei ein Alphabet mit einer Wahrscheinlichkeitsverteilung w:$\sum \to [0,1]$.

Beispiel:

$$\sum = \{a,b,c,d\} \ , \quad w(a) = 0,5$$
$$w(b) = 0,25$$
$$w(c) = 0,125$$
$$w(d) = 0,125$$

Def.: Sei $(\sum,w)$ gegeben und C ein Binärcode. Die mittlere Wortlänge $L_C(\sum,w)$ des Codes C über $(\sum,w)$ ist

$$L_C(\sum, w) = \sum_{a \in \sum} w(a) \cdot l(C(a)).$$

Beispiele: (w und A, B oben definiert)

(a) $L_A(\sum, w) = 2$

(b) $L_B(\sum, w) = 1,75.$

Ein Präfixcode ist "gut", falls die mittlere Wortlänge klein ist.

Wie bekommt man optimalen Präfixcode (bzgl. Codierung einzelner Zeichen) für $(\sum,w)$?

Algorithmus von Huffman:

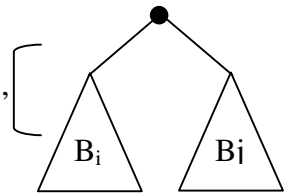
Sei $(\sum, w)$ mit $\sum = \{a_1, \ldots, a_n\}$ gegeben.

Verwendete Datenstuktur: (B, w) mit B Baum und $w \in [0,1]$.

Initialisierung: Eingabe der Liste $(a_1, w(a_1)), (a_2, w(a_2)), \ldots (a_n, w_n(a_n))$

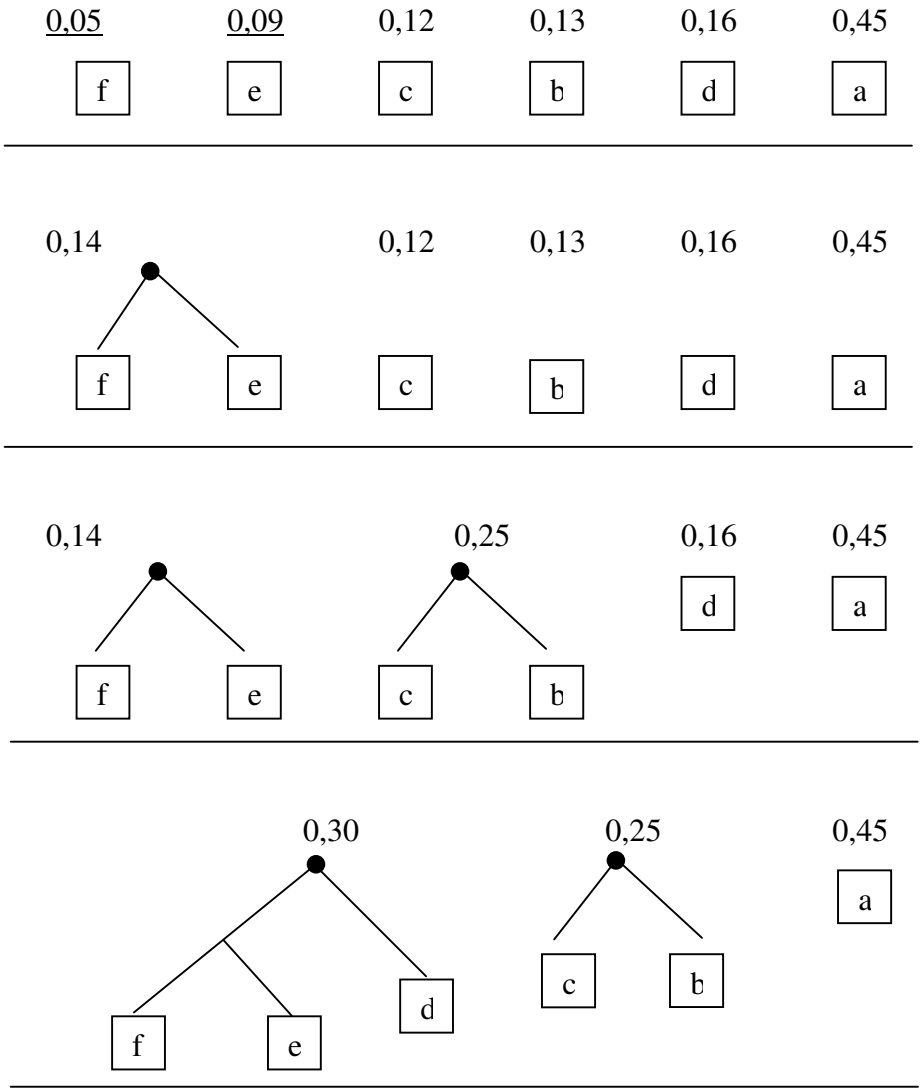
Dann wird n-1 mal folgende Operation durchgeführt:

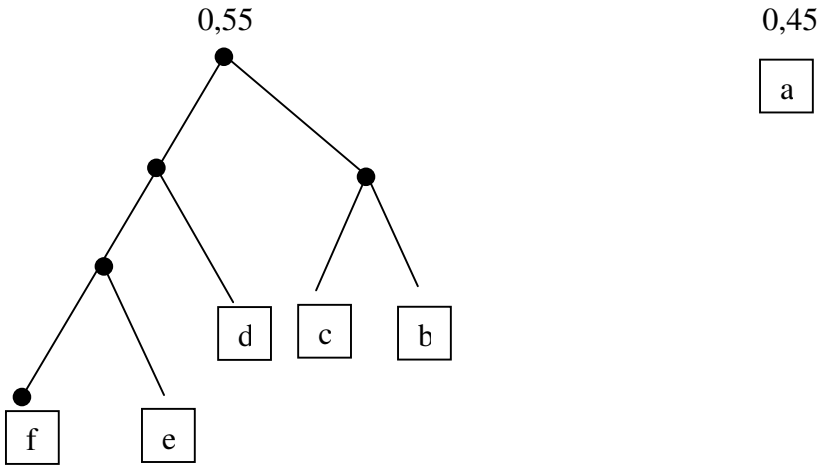
$(B_1, w_1), ..., (B_k, w_k)$ gegeben.
Bestimme w_i, w_j minimal unter $\{w_1, ..., w_k\}$ und berechne die Liste

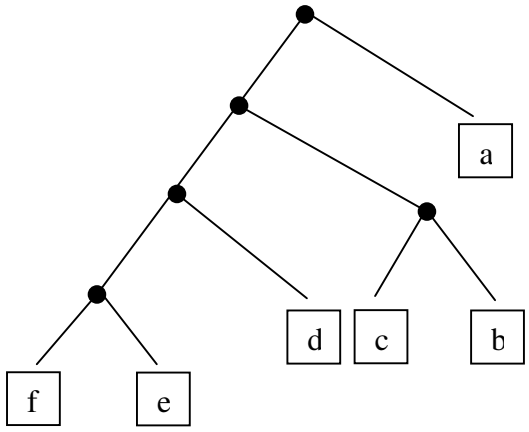
$\left((B_l, w_l), l \notin \{i, j\}\right)$, [ , $w_i + w_j$]

Nach n-1 Iterationen enthält die Liste nur noch 1 Element.
Dieses liefert die Baumrepräsentation *B* des gesuchten Binärcodes.

Beispiel:







C(f) = 0000
C(e) = 0001
C(d) = 001
C(c) = 010
C(b) = 011
C(a) = 1

8.3 Code-Sicherung

Grundfrage: wie kann man Binärcodierungen so wählen, dass Fehler erkannt, möglicherweise sogar behoben werden können?

→ redundante Codierung:

Maß für Störsicherheit eines Codes: Hamming-Distanz:
minimale Anzahl von Bits, in denen sich die Codes von 2 verschiedenen Zeichen unterscheiden

Beispiel: Ziffern 1 bis 4 in Binärcodierung

Codierung a):

C(1) = 001, C(2) = 010, C(3) = 011, C(4) = 100

Matrix der Distanzen:

	001	010	011	100
001	-			
010	2	-		
011	1	1	-	
100	2	2	3	-

Hamming-Distanz h = 1
kann sein, dass unbemerkt falsche Nachricht ankommt, wenn nur 1 Bit fehlerhaft übertragen wird.

Codierung b): C(1) = 000, C(2) = 011, C(3) = 101, C(4) = 110

Matrix der Distanzen:

	000	011	101	110
000	-			
011	2	-		
101	2	2	-	
110	2	2	2	-

Hamming-Distanz h = 2
wenn 1 Bit fehlerhaft übertragen wird, merkt man, dass etwas faul ist!

Einfache Möglichkeit Hamming-Distanz von mindestens 2 zu erreichen: m-aus-n Codes: Code jedes Zeichens hat Länge n, davon genau m mal 1.

Beispiel:

Zeichen	2-aus-5-Code	1-aus-10-Code
0	00011	0000000001
1	00101	0000000010
2	00110	0000000100
3	01001	0000001000
4	01010	0000010000
5	01100	0000100000
6	10001	0001000000
7	10010	0010000000
8	10100	0100000000
9	11000	1000000000

h mindestens 2

Codes mit Paritätsbits

Häufig verwendete Methode der Fehlererkennung und Korrektur.
Grundidee: zusätzliche Paritäts-Bits geben an, ob Anzahl der Einsen im "eigentlichen" Code eines Zeichens gerade oder ungerade ist (z.B. 0 falls gerade, 1 falls ungerade)

Beispiel (im ASCII-Code):

s	ASCII(s)	Parity Bit
I	100 1001	1
N	100 1110	0
F	100 0110	1
O	100 1111	1
R	101 0010	1
M	100 1101	0
A	100 0001	0
T	101 0100	1
K	100 1011	0

jetzt kann man feststellen, ob 1 Bit fehlerhaft übertragen wurde (aber nicht welches)

Erweiterung zu Rechteck-Codes: nach Übertragung von jeweils k Zeichen wird ein Längsprüfwort gesendet, das angibt, ob die Anzahl der jeweils 1., 2., 3. usw. Bits der gesendeten Wörter gerade ist.

Sender will INFORMATIK übermitteln, Empfänger erhält (Reihenfolge der empfangenen Bits: spaltenweise von oben nach unten, Spalten von links nach rechts).

empfangene Daten	Längsprüf-wort	empfangene Daten	Längsprüf-wort	empfangene Daten	
1 1 1 1	0	1 1 1 1	0	1 1	
0 0 0 0	0	0 0 0 0	0	0 0	
0 0 0 0	0	1 0 0 1	0	0 0	
0 1 0 1	**1**	0 1 0 0	1	1 1	
0 1 1 1	1	0 1 0 **0**	**0**	0 0	
0 1 1 1	1	1 0 0 0	1	0 1	
1 0 0 1	0	0 1 1 0	0	1 1	
1 0 1 1	1	1 0 0 **1**	0	1 0	Paritäts-Bits
ANFO		RMAP		IK	empfangen
INFO		RMAT		IK	korrigiert