

Theoretische Grundlagen der Informatik

Informationstheorie

Vorlesungen vom 30. Januar und 1. Februar 2018

INSTITUT FÜR THEORETISCHE INFORMATIK



Thema dieses Kapitels

Informationstheorie hat Anwendungen in

- Quellkodierung
- Kanalkodierung
- Kryptographie

Informationstheorie hat Anwendungen in

- Quellkodierung
 - Reduktion von Redundanz/Irrelevanz am Ausgang einer Informationsquelle
 - Hauptaufgabe: Datenkompression
 - Unterscheidung: Verlustfrei vs. verlustbehaftete Kompression
 - Hohe wirtschaftliche Bedeutung
- Kanalkodierung
- Kryptographie

Informationstheorie hat Anwendungen in

- Quellkodierung
- Kanalkodierung
 - Übertragung von digitalen Daten über gestörte Kanäle
 - Schutz vor Übertragungsfehlern durch Redundanz
 - Fehlerkorrektur
- Kryptographie

Informationstheorie hat Anwendungen in

- Quellkodierung
- Kanalkodierung
- Kryptographie
 - Informationssicherheit:
 - Konzeption, Definition und Konstruktion von Informationssystemen, die widerstandsfähig gegen unbefugtes Lesen und Verändern sind
 - Kryptographie bildet zusammen mit Kryptoanalyse die Kryptologie.

- Vorlesungsfolien
- TGI-Skript von Prof. Müller-Quade aus dem WS 08/09
(auf der TGI-Homepage verlinkt)
- Martin Werner: Information und Codierung, VIEWEG TEUBNER, 2008

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

Beispiel

- Ein idealer Würfel wird durch die Wahrscheinlichkeiten $(\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$ dargestellt.
- Das Ergebnis des idealen Würfels ist schwer vorherzusagen.
- Der Erkenntnisgewinn nach Ausgang des Experiments ist deshalb groß.

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

Beispiel

- Betrachte den gezinkten Würfel mit Wahrscheinlichkeiten $(\frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{2})$.
- Hier ist schon klarer, welche Zahl als nächstes gewürfelt wird.
- Der Erkenntnisgewinn ist also kleiner.

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

Frage

- Wir suchen ein Maß für den Erkenntnisgewinn nach Ausgang k mit Wahrscheinlichkeit p_k .
- Wir bezeichnen diesen Erkenntnisgewinn als **Information** I_{p_k}

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

Wünsche an die Definition von Information

- Information soll nicht negativ sein. In Formeln: $I_{p_i} \geq 0$
- Ein sicheres Ereignis (also $p_i = 1$) soll keine Information liefern.
- Kleine Änderungen an der Wahrscheinlichkeit sollen nur kleine Änderungen an der Information bewirken.
Etwas mathematischer ausgedrückt: Information soll stetig sein.

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

Wünsche an die Definition von Information

- Wunsch: Eine doppelt so lange Zeichenkette soll doppelte Information enthalten können
- Deshalb fordern wir, dass $I_{p_i \cdot p_j} = I_{p_i} + I_{p_j}$
- Dies soll später sicherstellen, dass die Information einer (unabhängigen) Zeichenkette gleich der Summe der Einzelinformationen ist.

- Sei $\Sigma = \{1, \dots, n\}$ eine Menge von Zeichen mit Wahrscheinlichkeiten $\{p_1, \dots, p_n\}$.
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Bemerkung: X wird auch diskrete, endliche Zufallsvariable genannt.

Definition Information

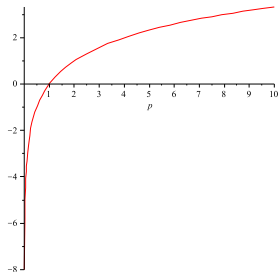
Sei p eine Wahrscheinlichkeit. Die Information von p (zur Basis b) ist

$$I_p = \log_b\left(\frac{1}{p}\right) = -\log_b(p)$$

Im Folgenden verwenden wir immer die Basis $b = 2$.

Wiederholung: Rechenregeln Logarithmus

- $\log_a(x \cdot y) = \log_a(x) + \log_a(y)$
- $\log_a(1/x) = -\log_a(x)$
- Basiswechsel: $\log_a(x) = \frac{\log_b(x)}{\log_b(a)}$



Definition Information

Sei p eine Wahrscheinlichkeit. Die Information von p (zur Basis b) ist

$$I_p = \log_b\left(\frac{1}{p}\right) = -\log_b(p)$$

Im Folgenden verwenden wir immer die Basis $b = 2$.

Beispiel 2:

- Betrachte eine Münze mit Seiten 0, 1 und Wkten $p_0 = p_1 = \frac{1}{2}$.
- Die Information eines Münzwurfs ist $\log(1/\frac{1}{2}) = \log(2) = 1$.
- Werfen wir die Münze k mal, so ist die Wahrscheinlichkeit für einen bestimmten Ausgang gleich $\frac{1}{2} \cdot \dots \cdot \frac{1}{2} = \frac{1}{2^k}$.
- Die Information ist dann $-\log(\frac{1}{2^k}) = \log(2^k) = k$

Anschaulich formuliert

- Entropie ist ein Maß für den mittleren Informationsgehalt pro Zeichen einer Quelle.

Interessante andere Sichtweise

- Entropie eines Strings bezeichnet die Länge unter der ein String nicht komprimiert werden kann.
- Die Kolmogorov-Komplexität eines String ist die Länge des kürzesten Programms, das diesen String ausgibt.
- Damit ist Entropie eine untere Schranke für die Kolmogorov-Komplexität.

Entropie

Die Entropie (zur Basis 2) einer diskreten Zufallsvariable mit Ereignissen (Zeichen) X und Wahrscheinlichkeiten $p(x)$ für $x \in X$, ist definiert durch

$$H(X) = \sum_{x \in X} p(x) \log_2\left(\frac{1}{p(x)}\right)$$

dabei gelten die folgenden Konventionen

$$0 \cdot \log 0 := 0, \quad 0 \cdot \log \frac{0}{0} := 0, \quad a \cdot \log \frac{a}{0} := \infty$$

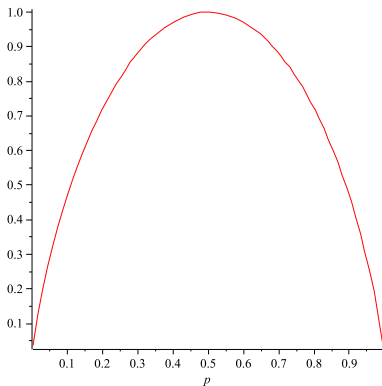
Bemerkung

- Es gilt immer $H(X) \geq 0$.

$$H(X) = \sum_{x \in X} p(x) \log_2 \left(\frac{1}{p(x)} \right)$$

- Die Entropie einer diskreten, endlichen Zufallsvariable mit n Zeichen wird maximal, wenn alle Zeichen gleichwahrscheinlich sind.
- Die maximale Entropie beträgt dann $\log_2(n)$.
- Die Entropie der deutschen Sprache liegt etwa bei 4,1.
- Bei 26 Buchstaben ergibt sich eine maximale Entropie von $\log_2(26) \approx 4,7$.

Entropie einer Münze mit Wkt p für Zahl



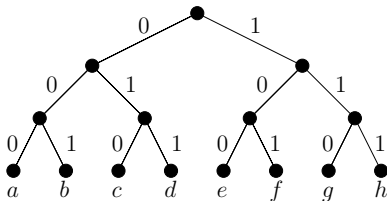
$$H(X) = \sum_{x \in X} p(x) \log_2\left(\frac{1}{p(x)}\right) = -p \log_2(p) - (1-p) \log_2(1-p)$$

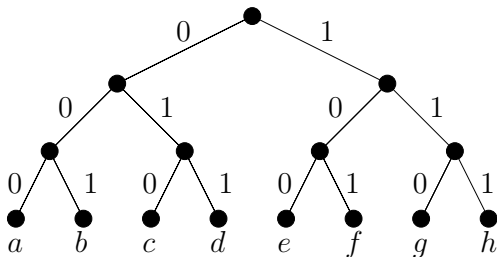
- Wir betrachten eine Informationsquelle X , die Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i liefert.
- Zum Codieren der Zeichen aus Σ haben wir aber nur Zeichenketten aus $\{0, 1\}^*$ zur Verfügung.
- Wie können wir Σ ohne Informationsverlust codieren, dass die erwartete Länge der Ausgabe möglichst klein wird?

Formal

- Wir ordnen jedem Zeichen $i \in \Sigma$ ein Codewort $z_i \in \{0, 1\}^*$ mit n_i Zeichen zu.
- Die mittlere Codewortlänge ist $\bar{n} = \sum_{i \in \Sigma} p_i n_i$.

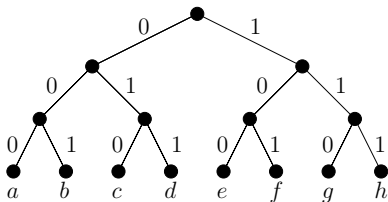
- Wir codieren im Folgenden binär.
- Sei $\Sigma = \{1, \dots, n\}$ ein Alphabet mit Präfix-Code $C = \{c_1, \dots, c_n\}$.
- Der Codierungsbaum T von (Σ, C) ist ein gerichteter, binärer Baum so dass
 - jede Kante mit 0 oder 1 annotiert ist,
 - ausgehend von einem Knoten höchstens eine Kante mit 0 und höchstens eine Kante mit 1 annotiert ist,
 - die Blätter von T genau die Elemente in Σ sind,
 - der Weg von der Wurzel zu $i \in \Sigma$ mit c_i annotiert ist.





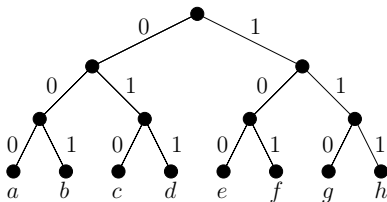
Beispiele

- Zeichen b hat Code 001
- Zeichen e hat Code 100
- Zeichen h hat Code 111



Bemerkungen

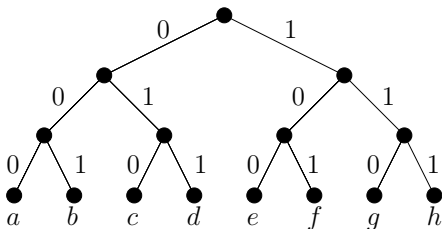
- Es besteht ein direkter Zusammenhang zwischen Codierungen und den zugehörigen Bäumen.
- Die *Tiefe* $d_T(v)$ eines Knotens v in einem Baum T ist die Anzahl der Kanten auf einem kürzesten Weg von der Wurzel zu v .



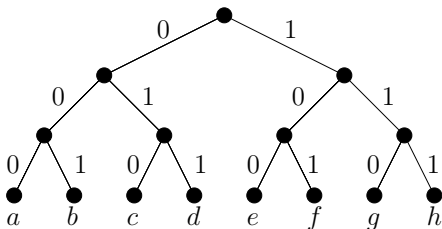
Bemerkungen

- Gegeben sei eine Codierung für Alphabet Σ mit Wahrscheinlichkeit p_i für $i \in \Sigma$, Codewortlänge n_i für $i \in \Sigma$ und zugehörigem Codierungsbaum T .
- Die mittlere Codewortlänge ist $\bar{n} = \sum_{i \in \Sigma} p_i n_i = \sum_{v \in \Sigma} p_v d_T(v)$.

- Betrachte eine Informationsquelle X mit $\Sigma = \{a, b, c, d, e, f, g, h\}$ und Wahrscheinlichkeiten $p_z = 1/8$ für $z \in \Sigma$.



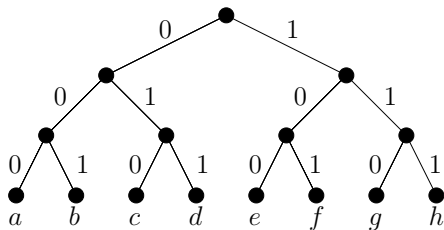
- Betrachte eine Informationsquelle X mit $\Sigma = \{a, b, c, d, e, f, g, h\}$ und Wahrscheinlichkeiten $p_z = 1/8$ für $z \in \Sigma$.



Mittlere Codewortlänge

- Hier haben alle Codes Länge 3.
- Die mittlere Codewortlänge ist also 3.

- Betrachte eine Informationsquelle X mit $\Sigma = \{a, b, c, d, e, f, g, h\}$ und Wahrscheinlichkeiten $p_z = 1/8$ für $z \in \Sigma$.



Anzahl der codierten Zeichen

- Mit jedem zusätzlichen Bit, verdoppelt sich die Größe des darstellbaren Alphabets.
- Um ein Alphabet Σ mit Wörtern gleicher Länge zu kodieren braucht man also $\log_2(|\Sigma|)$ Bits.

- Bei Codes mit variabler Länge muss man wissen, wann ein neues Codewort beginnt.
- Ein **Präfix-Code** ist ein Code, so dass kein Codewort Anfang eines anderen Codeworts ist.
- Für Präfix-Codes benötigt man deswegen keine Trennzeichen.
- Jeder Präfix-Code kann (auf die in der letzten Folie benutzte Art) als Baum dargestellt werden.

Beispiel: Morse-Alphabet

- Das Morse-Alphabet hat variable Länge.
- Das Morse-Alphabet ist kein Präfix-Code.
- Zur Unterscheidung von A und ET benötigt man ein Trennzeichen.
- Das Morsealphabet besteht deswegen aus 3 Zeichen.

Buchstabe	Morsezeichen	Buchstabe	Morsezeichen
A	○—	N	—○
B	—○○○	O	— — —
C	—○—○	P	○——○
D	—○○	Q	— — ○—
E	○	R	○—○
F	○○—○	S	○○○
G	— — ○	T	—
H	○○○○	U	○○—
I	○○	V	○○○—
J	○— — —	W	○— —
K	—○—	X	—○○—
L	○—○○	Y	—○— —
M	— —	Z	— — ○○

- Es seien Codes mit variabler Länge erlaubt.
- Es ist dann nützlich, häufige Zeichen mit kurzen Wörtern zu codieren.
- Dies verkleinert die mittlere Codewortlänge.

Satz (Shannon's Quellencodierungstheorem):

Sei X eine diskrete endliche Zufallsvariable mit Entropie $H(X)$. Weiter sei ein Präfix-Code für X mit einem Codealphabet aus D Zeichen und minimaler mittlerer Codewortlänge \bar{n} gegeben. Dann gilt

$$\frac{H(X)}{\log_2 D} \leq \bar{n} < \frac{H(X)}{\log_2 D} + 1 .$$

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	
1	0,1388	
2	0,1125	
3	0,0946	
4	0,0796	
5	0,0669	
6	0,0563	
7	0,0473	
8	0,0398	
9	0,0334	
10	0,0281	
11	0,0237	
12	0,0199	
...

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	0
1	0,1388	0
2	0,1125	0
3	0,0946	0
4	0,0796	1
5	0,0669	1
6	0,0563	1
7	0,0473	1
8	0,0398	1
9	0,0334	1
10	0,0281	1
11	0,0237	1
12	0,0199	1
...

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	00
1	0,1388	00
2	0,1125	01
3	0,0946	01
4	0,0796	1
5	0,0669	1
6	0,0563	1
7	0,0473	1
8	0,0398	1
9	0,0334	1
10	0,0281	1
11	0,0237	1
12	0,0199	1
...

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	000
1	0,1388	001
2	0,1125	01
3	0,0946	01
4	0,0796	1
5	0,0669	1
6	0,0563	1
7	0,0473	1
8	0,0398	1
9	0,0334	1
10	0,0281	1
11	0,0237	1
12	0,0199	1
...

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	000
1	0,1388	001
<hr/> <hr/>		
2	0,1125	010
3	0,0946	011
<hr/> <hr/>		
4	0,0796	1
5	0,0669	1
6	0,0563	1
7	0,0473	1
8	0,0398	1
9	0,0334	1
10	0,0281	1
11	0,0237	1
12	0,0199	1
...

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	000
1	0,1388	001
2	0,1125	010
3	0,0946	011
<hr/>		
4	0,0796	10
5	0,0669	10
6	0,0563	10
7	0,0473	10
<hr/>		
8	0,0398	11
9	0,0334	11
10	0,0281	11
11	0,0237	11
12	0,0199	11
...

Beispiel: Shannon-Fano Codierung

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	000
1	0,1388	001
2	0,1125	010
3	0,0946	011
4	0,0796	100
5	0,0669	100
6	0,0563	101
7	0,0473	101
8	0,0398	11
9	0,0334	11
10	0,0281	11
11	0,0237	11
12	0,0199	11
...

Beispiel: Shannon-Fano Codierung

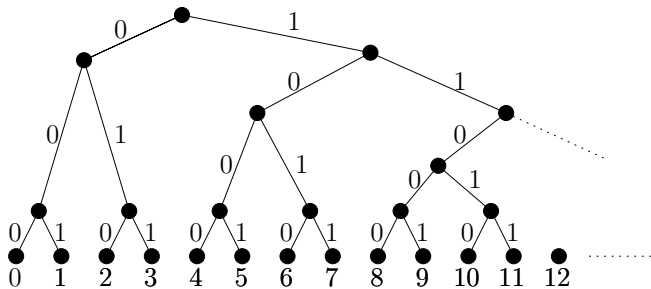
Ein paar Zwischenschritte später ...

Elemente	Wahrscheinlichkeiten	Codewort
0	0,1591	000
1	0,1388	001
2	0,1125	010
3	0,0946	011
4	0,0796	1000
5	0,0669	1001
6	0,0563	1010
7	0,0473	1011
8	0,0398	11000
9	0,0334	11001
10	0,0281	11010
11	0,0237	11011
12	0,0199	111000
...

Funktion $\text{ShannonFano}(Z)$

- **Eingabe:** Zeichenliste $Z = (z_1, \dots, z_k)$ mit Wkten p_1, \dots, p_k
- **Ausgabe:** Shannon-Fano Codierung (c_1, \dots, c_k)
- Wenn $k = 1$
 - return $(c_1 = \epsilon)$ and exit
- Sortiere Zeichen Z absteigend nach Wkt p_i (d.h. $p_1 \geq p_2, \dots \geq p_k$).
- Trenne Z in
 - $Z_1 \leftarrow (z_1, \dots, z_l)$
 - $Z_2 \leftarrow (z_{l+1}, \dots, z_k)$so dass $|\sum_{i=1}^l p_i - \sum_{i=l+1}^k p_i|$ minimal ist.
- $(c_1, \dots, c_l) \leftarrow (0s_1, \dots, 0s_l)$ mit $(s_1, \dots, s_l) \leftarrow \text{ShannonFano}(Z_1)$
- $(c_{l+1}, \dots, c_k) \leftarrow (1s_{l+1}, \dots, 1s_k)$ mit $(s_{l+1}, \dots, s_k) \leftarrow \text{ShannonFano}(Z_2)$
- return (c_1, \dots, c_k)

Codierungsbaum Shannon-Fano



- Die mittlere Codewortlänge der Shannon-Fano Codierung muss nicht optimal sein.
- Sie ist deswegen nicht sehr verbreitet.
- Wir werden sehen, dass die **Huffman-Codierung** optimale mittlere Codewortlänge besitzt.

Beispiel: Huffman-Codierung

d

0,4

f

0,2

b

0,15

e

0,15

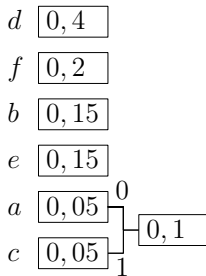
a

0,05

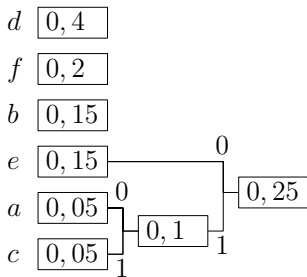
c

0,05

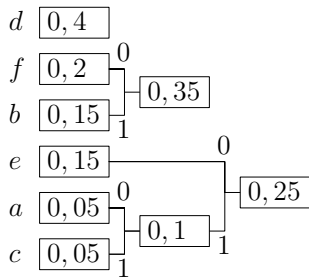
Beispiel: Huffman-Codierung



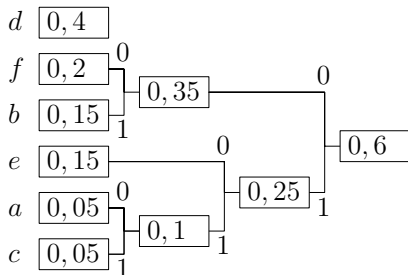
Beispiel: Huffman-Codierung



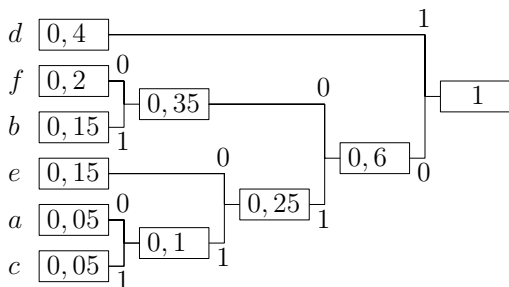
Beispiel: Huffman-Codierung



Beispiel: Huffman-Codierung



Beispiel: Huffman-Codierung



Eingabe: Zeichen $1, \dots, n$ mit Wahrscheinlichkeiten p_1, \dots, p_n

Ausgabe: Baum T des Huffman-Codes

- Menge $Q = \{1, \dots, n\}$
- Füge alle Zeichen aus Q als Blätter in T ein
- Für $i = 1, \dots, n - 1$
 - Erzeuge neuen Knoten z für T
 - $u \leftarrow$ extrahiere Element x aus Q mit p_x minimal
 - Bestimme u als linker Nachfolger von z
 - $v \leftarrow$ extrahiere Element x aus Q mit p_x minimal
 - Bestimme v als rechter Nachfolger von z
 - Wahrscheinlichkeit p_z von z ist $p_u + p_v$
 - Füge z in Q ein
- $r \leftarrow$ extrahiere letztes Element aus Q
- return r (r ist Wurzel von T)

Satz:

Der Huffman-Algorithmus berechnet einen Codierungsbaum mit minimaler mittlerer Codewortlänge.

Satz:

Sei $\Sigma = \{1, \dots, n\}$ ein Alphabet mit Wahrscheinlichkeiten P , wobei $P = \{p_1, \dots, p_n\}$. Seien $x, y \in \Sigma, x \neq y$ eine beliebige Wahl für die zwei unwahrscheinlichsten Zeichen.

Dann gibt es einen Codierungsbaum T für (Σ, P) mit minimaler mittlerer Codewortlänge, so dass x und y den gleichen Elternknoten besitzen.

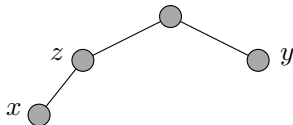
- Sei $\Sigma = \{1, \dots, n\}$ ein Alphabet mit Wkten $P = \{p_1, \dots, p_n\}$.
- Seien $x, y \in \Sigma, x \neq y$ eine beliebige Wahl für die zwei unwahrscheinlichsten Zeichen.

Beweis

- Sei T' ein beliebiger Codierungsbaum für (Σ, P) mit minimaler mittlerer Codewortlänge.
- O.B.d.A gelte für die Tiefe, dass $d'_{T'}(x) \geq d'_{T'}(y)$.
- Sei z der Elternknoten von x in T' .

Fall 1: z hat nur x als Nachkommen

- Dann könnte man z löschen und durch x ersetzen.
- Dieser Baum hätte eine kleinere mittlere Codewortlänge.
- Widerspruch zur Optimalität von T' .



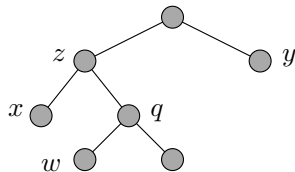
- Sei $\Sigma = \{1, \dots, n\}$ ein Alphabet mit Wkten $P = \{p_1, \dots, p_n\}$.
- Seien $x, y \in \Sigma, x \neq y$ eine beliebige Wahl für die zwei unwahrscheinlichsten Zeichen.

Beweis

- Sei T' ein beliebiger Codierungsbaum für (Σ, P) mit minimaler mittlerer Codewortlänge.
- O.B.d.A gelte für die Tiefe, dass $d'_{T'}(x) \geq d'_{T'}(y)$.
- Sei z der Elternknoten von x in T' .

Fall 2: z hat mehr als 2 Nachkommen

- Sei $w \neq x$ ein Nachfahre von z von maximaler Tiefe.
- Optimalität von T' : $p_w \leq p_x$.
- Wahl von x, y : $p_w = p_x$.
- Tausche x mit w . Weiter mit Fall 3.



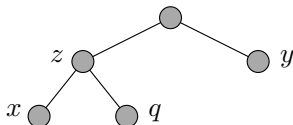
- Sei $\Sigma = \{1, \dots, n\}$ ein Alphabet mit Wkten $P = \{p_1, \dots, p_n\}$.
- Seien $x, y \in \Sigma, x \neq y$ eine beliebige Wahl für die zwei unwahrscheinlichsten Zeichen.

Beweis

- Sei T' ein beliebiger Codierungsbaum für (Σ, P) mit minimaler mittlerer Codewortlänge.
- O.B.d.A gelte für die Tiefe, dass $d'_{T'}(x) \geq d'_{T'}(y)$.
- Sei z der Elternknoten von x in T' .

Fall 3: z hat genau 2 Nachkommen

- Sei $q \neq x$ der andere Nachfahre von z .
Tausche q mit y .
- q und y gleiche Tiefe: Tauschen ok.
- q tiefer als y : Wegen Optimalität
 $p_q = p_y$.



Satz:

Der Huffman-Algorithmus berechnet einen Codierungsbaum mit minimaler mittlerer Codewortlänge.

Beweis

- Wir benutzen Induktion nach der Anzahl der Zeichen $|\Sigma|$ des Alphabets Σ .
- **Induktionsanfang:** Die Aussage ist für ein Zeichen erfüllt.

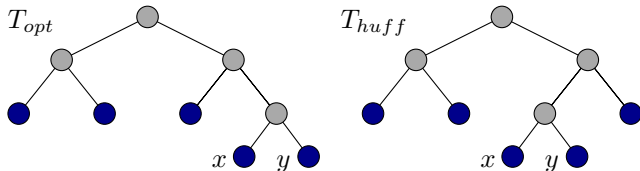
Induktions-Voraussetzung

- Der Huffman-Algorithmus berechnet einen optimalen Codierungsbaum für alle Alphabete Σ mit $|\Sigma| \leq n$ und alle Möglichkeiten für P .

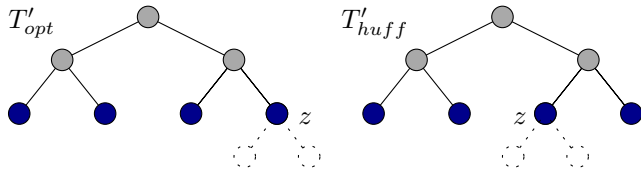
Induktions-Schluss

- Gegeben sei Alphabet $\Sigma = \{1, \dots, n+1\}$ mit Wahrscheinlichkeiten $P = \{p_1, \dots, p_{n+1}\}$.
- Für einen Codierungsbaum T bezeichne $f(T) = \sum_{v \in \Sigma} p_v d_T(v)$ die zugehörige mittlere Wortlänge.
- Wir machen einen Widerspruchsbeweis.
- Bezeichne T_{huff} einen Huffman-Baum für (Σ, P) .
- Sei T_{opt} einen Codierungsbaum für (Σ, P) mit $f(T_{\text{opt}}) < f(T_{\text{huff}})$.

- Seien $x, y \in \Sigma$ die Zeichen, die im Huffman-Algorithmus zuerst zusammengefasst werden.
- Vorbereitendes Lemma: Wir können T_{opt} so wählen, dass x und y den gleichen Elternknoten besitzen.

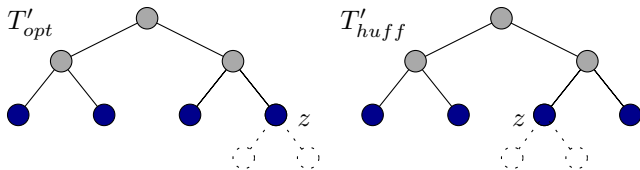


- Seien $x, y \in \Sigma$ die Zeichen, die im Huffman-Algorithmus zuerst zusammengefasst werden.
- Vorbereitendes Lemma: Wir können T_{opt} so wählen, dass x und y den gleichen Elternknoten besitzen.



- Sei $\Sigma' = \Sigma \setminus \{x, y\} \cup \{z\}$ Instanz für neues Zeichen z mit Wkt $p_x + p_y$.
- Seien T'_{opt} und T'_{huff} die Bäume, die sich aus T_{opt} und T_{huff} ergeben, wenn man x, y mit ihrem Elternknoten zu Knoten z verschmilzt.

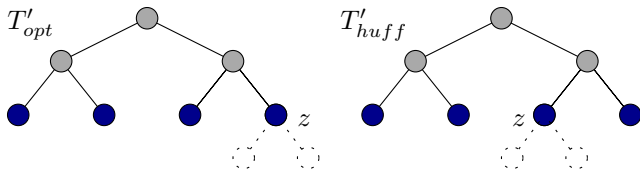
- Sei $\Sigma' = \Sigma \setminus \{x, y\} \cup \{z\}$ Instanz für neues Zeichen z mit Wkt $p_x + p_y$.
- Seien T'_{opt} und T'_{huff} die Bäume, die sich aus T_{opt} und T_{huff} ergeben, wenn man x, y mit ihrem Elternknoten zu Knoten z verschmilzt.



Es gilt:

- T'_{huff} ist ein Huffman-Baum für Instanz Σ' mit den neuen Wkten.
- T'_{opt} ist ein Codierungs-Baum für Instanz Σ' mit den neuen Wkten.

- Sei $\Sigma' = \Sigma \setminus \{x, y\} \cup \{z\}$ Instanz für neues Zeichen z mit Wkt $p_x + p_y$.
- Seien T'_{opt} und T'_{huff} die Bäume, die sich aus T_{opt} und T_{huff} ergeben, wenn man x, y mit ihrem Elternknoten zu Knoten z verschmilzt.



Es gilt:

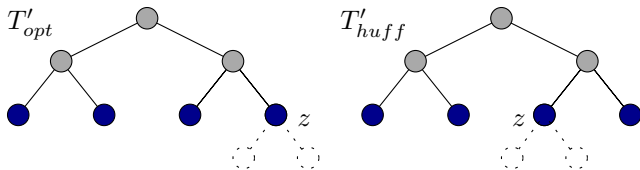
$$f(T'_{huff}) = f(T_{huff}) - d_{T_{huff}}(x)p_x - d_{T_{huff}}(y)p_y + d_{T'_{huff}}(z)p_z$$

$$= f(T_{huff}) - p_x - p_y$$

$$f(T'_{opt}) = f(T_{opt}) - d_{T_{opt}}(x)p_x - d_{T_{opt}}(y)p_y + d_{T'_{opt}}(z)p_z$$

$$= f(T_{opt}) - p_x - p_y$$

- Sei $\Sigma' = \Sigma \setminus \{x, y\} \cup \{z\}$ Instanz für neues Zeichen z mit Wkt $p_x + p_y$.
- Seien T'_{opt} und T'_{huff} die Bäume, die sich aus T_{opt} und T_{huff} ergeben, wenn man x, y mit ihrem Elternknoten zu Knoten z verschmilzt.



- Damit ist T'_{opt} ein besserer Coderierungsbaum für Σ' als T'_{huff} .
- Da $|\Sigma'| = n$ ist dies ein Widerspruch zur Induktionsvoraussetzung.

- Unterschiedliche Codewortlängen führen zu unterschiedlichen Bitraten und Decodierungsverzögerung.
- Datenkompression reduziert die Redundanz und erhöht damit die Fehleranfälligkeit.
- Die Kenntnis der Wahrscheinlichkeiten der Zeichen wird vorausgesetzt.
- Universelle Codierverfahren wie der Lempel-Ziv Algorithmus setzen kein a-priori Wissen an die Statistik der Daten voraus.

- Bei der Faxübertragung wird die Vorlage zeilenweise abgetastet und in weiße (w) und schwarze (s) Bildelemente zerlegt.
- Üblicherweise ist die Zahl der weißen Elemente viel höher, als die der schwarzen.
- Wir nehmen der Einfachheit halber an, dass die Bildpunkte voneinander unabhängig sind.
- Bei 15% Schwärzungsgrad ergibt sich eine Entropie von $H = -0.85 \cdot \log_2(0.85) - 0.15 \cdot \log_2(0.15) \approx 0.61$
- Bei guter Codierung sollte eine entsprechende mittlere Codewortlänge zu erwarten sein.

- Bei der Faxübertragung wird die Vorlage zeilenweise abgetastet und in weiße (w) und schwarze (s) Bildelemente zerlegt.
- Üblicherweise ist die Zahl der weißen Elemente viel höher, als die der schwarzen.
- Wir nehmen der Einfachheit halber an, dass die Bildpunkte voneinander unabhängig sind.
- Bei 15% Schwärzungsgrad ergibt sich eine Entropie von $H = -0.85 \cdot \log_2(0.85) - 0.15 \cdot \log_2(0.15) \approx 0.61$
- Bei guter Codierung sollte eine entsprechende mittlere Codewortlänge zu erwarten sein.

Problem:

Wie ist platzsparende Codierung von einem Alphabet mit zwei Zeichen möglich?

Problem:

Wie ist platzsparende Codierung von einem Alphabet mit zwei Zeichen möglich?

- Möglicher Ansatz: Blockcodes
- Fasse k Zeichen zu Blöcken zusammen und codiere diesen
- Beispiel $k = 2$:
 - Neues Alphabet: ww,ws,sw,ss.
 - Dieses kann platzsparend codiert werden.

Beispiel:

Zeichen	ww	ws	sw	ss
Wkt	$\frac{1}{2}$	$\frac{2}{10}$	$\frac{2}{10}$	$\frac{1}{10}$
Huffman	0	11	100	101

Lauf längencodierung

- Spezielle Zusammenfassung für Bildcodierung bei Fax/Videoanwendungen
- Die Länge der Blöcke ist variabel.
- **Idee:** Codiere nicht die Bildpunkte, sondern den Abstand zwischen zwei schwarzen Bildpunkten.
- Beispiel:

wwwswswsswwwwwswwswwwwwwwswwwwwwws

wird aufgefasst als 3204166.

- Für eine Binärcodierung braucht man noch Codes für die Abstände (also für \mathbb{N}).
- Um dies platzsparend zu machen, benötigt man Wahrscheinlichkeiten für einzelne Abstände.

Lauf längencodierung

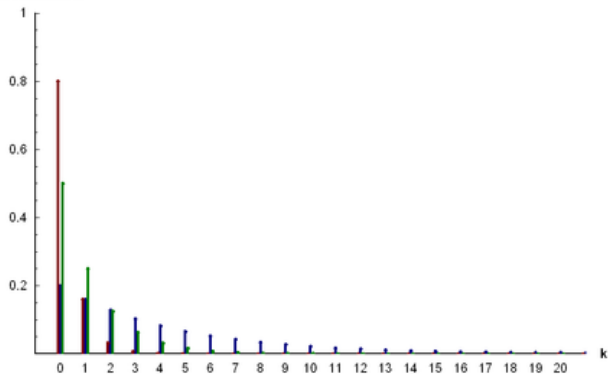
- Wie groß sind die Wkten für die einzelnen Abstände?
- Annahme: Die Bildpunkte sind voneinander unabhängig.
- Sei p_l die Wkt für einen Block aus l aufeinanderfolgenden weißen Bildpunkten mit einem schwarzen Bildpunkten am Schluss

$$p_l = \mathbb{P}(w^l s) = \mathbb{P}^l(w) \cdot \mathbb{P}(s).$$

- Es ergibt sich eine geometrische Verteilung.

Geometrische Verteilung

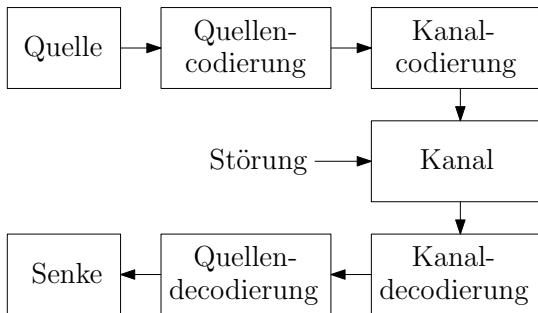
Wahrscheinlichkeit



Quelle: Wikipedia.

- Man kann ein Schwarzweißbild über Angabe der Lauf längen verlustfrei rekonstruieren.
- Sonderbehandlung für letzten Block erforderlich.
- Weiteres Problem: Lauf längen können beliebig groß werden.
- Shannon-Fano-Codierung kann trotzdem einfach angewandt werden.

Abstand	Wkten	Codewort
0	0,1591	000
1	0,1388	001
2	0,1125	010
3	0,0946	011
4	0,0796	1000
5	0,0669	1001
6	0,0563	1010
7	0,0473	1011
8	0,0398	11000
9	0,0334	11001
10	0,0281	11010
11	0,0237	11011
12	0,0199	111000
...



- Qualität einer digitalen Übertragung wird häufig als gemessene Bitfehlerquote bzw. Bitfehlerwahrscheinlichkeit angegeben.
- Beherrschung von Übertragungsfehlern:
 - Fehlerkorrektur (beim Empfänger),
 - Fehlererkennung und Wiederholungsanforderung.
- Tradeoff: Wahrscheinlichkeit unentdeckter Fehler vs. Datendurchsatz.



Quelle: Wikipedia

- Paritätscode der RS232-Schnittstelle.
- Neunpoliges Kabel ermöglicht die parallele Übertragung von 8 Bit.
- Dabei werden nur sieben Bits b_1, \dots, b_7 für die Nachrichtenübertragung genutzt.
- Das achte Bit b_8 wird Paritätsbit genannt.

- Paritätscode der RS232-Schnittstelle.
- Neunpoliges Kabel ermöglicht die parallele Übertragung von 8 Bit.
- Dabei werden nur sieben Bits b_1, \dots, b_7 für die Nachrichtenübertragung genutzt.
- Das achte Bit b_8 wird Paritätsbit genannt.

Wiederholung XOR-Verknüpfung \oplus

\oplus	0	1
0	0	1
1	1	0

- Es wird b_8 so gesendet, dass

$$b_8 = b_1 \oplus b_2 \oplus b_3 \oplus b_4 \oplus b_5 \oplus b_6 \oplus b_7.$$

Wiederholung XOR-Verknüpfung \oplus

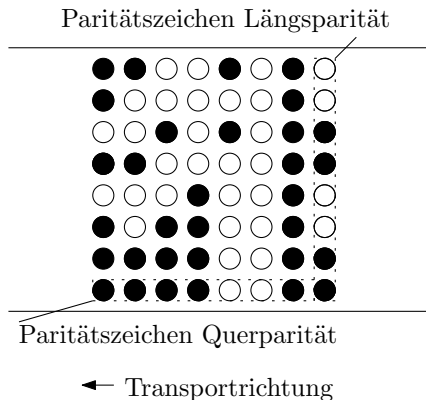
\oplus	0	1
0	0	1
1	1	0

- Es wird b_8 so gesendet, dass

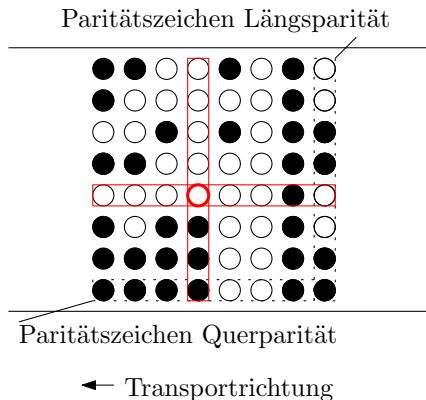
$$b_8 = b_1 \oplus b_2 \oplus b_3 \oplus b_4 \oplus b_5 \oplus b_6 \oplus b_7.$$

- Mit dem Paritätscode werden einfache Fehler im Codewort erkannt.
- Gleichzeitiger Übertragungsfehler von 2 Fehlern werden nicht erkannt.
- Falls ein Fehler erkannt wird, kann die ursprüngliche Nachricht nicht rekonstruiert werden: Die Fehlerstelle ist unbekannt.

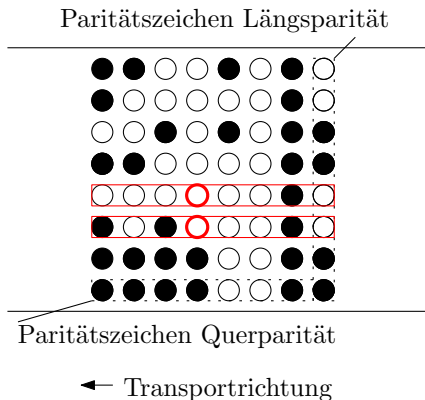
- Dient zum Schutz gegen Doppelfehler
- Erklärung am Beispiel Lochkarte



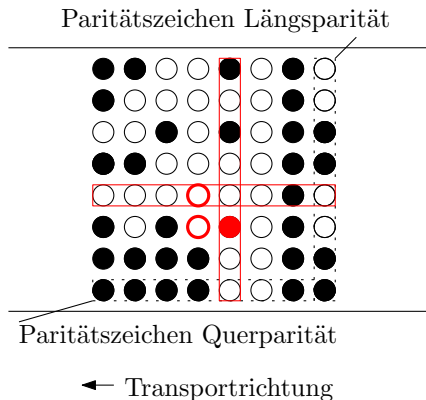
- Alle 1,2,3 fachen Fehler sind erkennbar
- Ab 4 Fehlern nicht zwingend erkennbar



- Alle 1,2,3 fachen Fehler sind erkennbar
- Ab 4 Fehlern nicht zwingend erkennbar



- Alle 1,2,3 fachen Fehler sind erkennbar
- Ab 4 Fehlern nicht zwingend erkennbar



Paritätscodes

- Gegeben ein Alphabet $\Sigma = \{1, 2, \dots, q - 1\}$.
- Ein Code der Länge n zur Basis q sei eine Menge von Folgen mit Elementen aus Σ .
- Einzelne Folgen werden Codewörter genannt.

Ein Paritätscode liegt vor, wenn für jedes Codewort a_1, a_2, \dots, a_n

$$(a_1 + a_2 + \dots + a_n) \bmod q = 0$$

gilt.

Satz:

Jeder Paritätscode erkennt Einzelfehler.

- Sei a_1, \dots, a_n das ursprüngliche Codewort.
- Annahme: Das i -te Element wurde fehlerhaft als \tilde{a}_i übertragen.
- Ein Übertragungsfehler liegt vor, wenn

$$(a_1 + a_2 + \dots + \tilde{a}_i + \dots + a_n) \bmod q = 0.$$

- Für das ursprüngliche Codewort gilt:

$$(a_1 + a_2 + \dots + a_n) \bmod q = 0.$$

Also

$$\begin{aligned} 0 &= (a_1 + a_2 + \dots + \tilde{a}_i + \dots + a_n) \bmod q \\ &= (a_1 + a_2 + \dots + a_n) \bmod q \end{aligned}$$

$$\begin{aligned} 0 &= (a_1 + a_2 + \dots + \tilde{a}_i + \dots + a_n) \pmod{q} \\ &= (a_1 + a_2 + \dots + a_n) \pmod{q} \end{aligned}$$

Damit

$$\begin{aligned} 0 &= (a_1 + a_2 + \dots + \tilde{a}_i + \dots + a_n) \pmod{q} - \\ &\quad (a_1 + a_2 + \dots + a_n) \pmod{q} \\ &= (\tilde{a}_i - a_i) \pmod{q} \end{aligned}$$

Um dies zu erfüllen muss $(\tilde{a}_i - a_i)$ durch q teilbar sein. Weil aber

$$0 \leq a_i < q$$

folgt

$$0 \leq |\tilde{a}_i - a_i| < q .$$

- Häufige Fehlerart bei manueller Eingabe: Vertauschungsfehler.
- Diese werden von gewöhnlichen Paritätscodes nicht erkannt.
- Sei wieder a_1, \dots, a_n ein Codewort.
- **Paritätscode mit Gewichten:** Wir führen zusätzlich ganzzahlige Gewichte w_1, \dots, w_{n-1} ein, so dass

$$(w_1 \cdot a_1 + w_2 \cdot a_2 + \dots + w_{n-1} a_{n-1} + a_n) \pmod q = 0$$

gilt.

- Zusatzbedingung: Alle Gewichte w_i müssen teilerfremd zu q sein.

- **Paritätscode mit Gewichten:** Wir führen zusätzlich ganzzahlige Gewichte w_1, \dots, w_{n-1} ein, so dass

$$(w_1 \cdot a_1 + w_2 \cdot a_2 + \dots + w_{n-1} a_{n-1} + a_n) \pmod q = 0$$

gilt.

- Zusatzbedingung: Alle Gewichte w_i müssen teilerfremd zu q sein.

Satz:

Jeder Paritätscode mit Gewichten erkennt Einzelfehler.

Beweis: Analog zu normalen Paritätscodes kann gezeigt werden, dass Einzelfehler nicht erkannt werden, wenn

$$w_j \cdot (\tilde{a}_j - a_j) \pmod q = 0 .$$

- **Paritätscode mit Gewichten:** Wir führen zusätzlich ganzzahlige Gewichte w_1, \dots, w_{n-1} ein, so dass

$$(w_1 \cdot a_1 + w_2 \cdot a_2 + \dots + w_{n-1} a_{n-1} + a_n) \mod q = 0$$

gilt.

- Zusatzbedingung: Alle Gewichte w_i müssen teilerfremd zu q sein.

Satz:

Ein Paritätscode mit Gewichten erkennt die Vertauschung an den Stellen i und j , falls die Zahl $w_i - w_j$ teilerfremd zu q ist.

Beweis: Analog zu oben: Vertauschungsfehler wird nicht erkannt, falls

$$[(w_i a_i + w_j a_j) - (w_j a_i + w_i a_j)] \mod q = [(w_i - w_j)(a_i - a_j)] \mod q = 0 .$$

Bsp: ISBN-10



- ISBN: International Standard Book Number (ISO Standard 2108)
- ISBN-10 (Code oben im letzten Beispiel) war bis 2006 übliche Codierung
- Seit 2007 EAN-13 (European Article Number)

Beschreibung ISBN-10

- Alphabet $\Sigma = \{0, 1, \dots, 9\}$,
- Basis $q = 11$ (Primzahl!),
- Länge $n = 10$,
- Paritätscode,
- Für Code a_1, \dots, a_{10} berechnet sich die Prüfziffer a_{10} aus

$$(10a_1 + 9a_2 + 8a_3 + \dots + 2a_9 + a_{10}) \bmod 11 = 0 .$$

- Man unterscheidet verschiedene Arten von Kanal-Codes.
- **Block-Codes:** Hier betrachtet man Codeworte fester Länge. Aufeinanderfolgende Blöcke werden unabhängig voneinander kodiert.
- **Faltungs-Codes:** Codeworte können beliebig lang sein. Die Zeichen sind vom Vorgeschehen abhängig.
- In TGI befassen wir uns ausschließlich mit Block-Codes.

Hamming-Distanz

Für $x, y \in \{0, 1\}^n$ ist

$$d(x, y) := \#\{i \mid i = 1, \dots, n, x_i \neq y_i\}$$

die Hamming-Distanz zwischen x und y .

Anschaulich: Die Hamming-Distanz zwischen x und y ist die Anzahl der Zeichen in x , die sich von denen in y unterscheiden.

Es sei $B_\rho(x)$ die Menge aller Worte y mit $d(x, y) \leq \rho$.

Anschaulich: B_ρ ist eine Kugel (die Hamming-Kugel) um x mit Radius ρ .

Maximum-Likelihood-Decoding

Sei eine Kodierung C gegeben und y ein empfangenes Wort. Decodiere y als dasjenige Codewort $x \in C$, für das $d(x, y)$ minimal wird.

Shannon's Theorem

- Betrachte einen Code C mit M Worten der Länge n .
- Seien x_1, \dots, x_M die Codeworte.
- Benutze Maximum-Likelihood-Decoding.
- Sei P_i die Wahrscheinlichkeit dafür, dass falsch dekodiert wird, wenn das Wort x_i gesendet wurde.
- Die durchschnittliche Wahrscheinlichkeit einer falschen Dekodierung ist

$$P_C = \frac{1}{M} \sum_{i=1}^M P_i .$$

Block-Code

- Gegeben ist ein endliches Alphabet Σ .
- Ein Block-Code ist eine Teilmenge $C \subseteq \Sigma^n$ für ein $n \in \mathbb{N}$.
- Falls $\#C = 1$, so heißt C trivial, da es nur ein Codewort gibt.

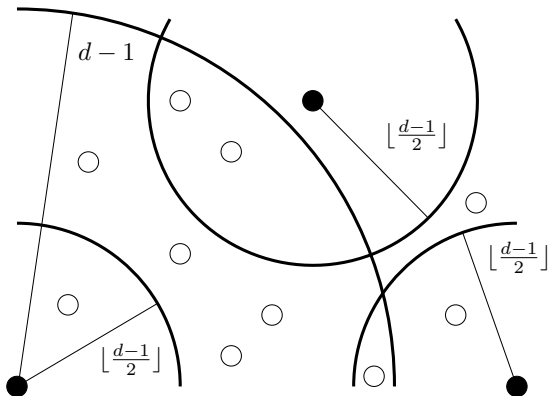
Minimaldistanz

Die Minimaldistanz eines nichttrivialen Block-Codes C ist

$$m(C) := \min_{c_1, c_2 \in C, c_1 \neq c_2} d(c_1, c_2) .$$

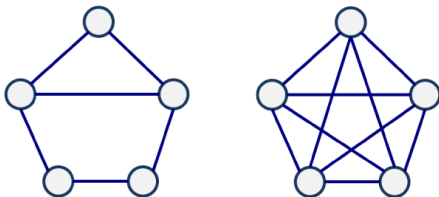
Satz:

Ein Block-Code C mit Minimaldistanz $m(C) = d$ kann entweder bis zu $d - 1$ Fehler erkennen oder bis zu $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehler korrigieren.



- Vorlesung Planare Graphen
- Proseminar Algorithmen für NP-schwere Probleme
- ICPC Praktikum

Ein *planarer* Graph ist ein Graph, der in der Ebene gezeichnet werden, ohne dass die Kanten sich kreuzen.



- besonders einfache, schnelle und schöne Algorithmen
- manche Probleme, die auf allgemeinen Graphen (NP-) schwer sind, können auf planaren Graphen sehr effizient gelöst werden
- 5 ECTS-Punkte

Proseminar Algorithmen für NP-schwere Probleme

- Algorithmen für NP-schwere Probleme mit *beweisbaren* Garantien
- Vertieft und erweitert Themen dieser Vorlesung
 - Approximationsalgorithmen
 - ILPs
 - Parameterisierte Komplexität
 - Effizientere exponentielle Algorithmen
- Prüfbar im Bachelorstudium
- 3 ECTS-Punkte
- Anmeldung bei sascha.gritzbach@kit.edu

ACM International Collegiate Programming Contest (ICPC)

- kleine algorithmische Aufgaben (< 60 Min)
 - Aufgabe verstehen
 - Algorithmus entwerfen
 - Algorithmus implementieren (C++ oder Java)
- Nur die Funktionalität des Codes wird bewertet
 - Kompiliert & übersteht Testdaten \implies Aufgabe gelöst

Vorbereitungspraktikum im Sommersemester

- 4 ECTS-Punkte
- Interessant für Leute, die:
 - beim ICPC mitmachen wollen
 - C++ üben wollen
 - Spaß an kleinen algorithmischen Aufgaben haben
- Details: <http://icpc.iti.kit.edu> **wenig Plätze, schnell anmelden!**