Codierungstheorie

Code-Arten und Code-Sicherung

fachhochschule münchen
munich university of applied sciences

Inhaltsübersicht und Literatur

- **Informationstheorie**
  - **Was ist eine Codierung?**
  - **Arten von Codes**
  - **Informationsgehalt und Entropie**
  - **Shannon'sches Codierungstheorem**

- **Fehlererkennende und fehlerkorrigierende Codes**
  - **Die Hammingdistanz**
  - **CRC-Codes**
  - **Prüfzeichenverfahren**
  - **Hamming-Code**

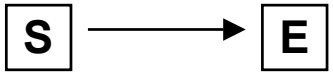
Literatur:

Blieberger et.al.: Informatik (Kap. 3 und 4), Springer-Verlag

R.-H. Schulz: Codierungstheorie, Vieweg

Begriffe der Informationstheorie (1)

- **Information** bedeutet Kenntnis über Sachverhalte oder Vorgänge.
- Informationen werden ausgetauscht mittels **Nachrichten**.
- Nachrichten werden übermittelt von einem Sender zu einem Empfänger:



- Nachrichten werden als **Daten** dargestellt in Form von
  - Zeichen (diskrete Daten, speziell: digitale Daten), z.B.:
    - geschriebene Wörter
    - Gesten der Gehörlosensprache
    - Flaggenalphabet der Marine
    - ASCII-Zeichen
  - kontinuierlichen Funktionen (analoge Daten), z.B.
    - gesprochene Wörter
    - analoges Telefon

Begriffe der Informationstheorie (2)

- Das physikalische Abbild der Daten sind meßbare Signale, gekennzeichnet durch:
  - Signalparameter
  - Wert bzw. Werteverlauf (Zeitfunktion)

- Der Empfänger leitet aus einer Nachricht Information ab: f: N --> I
  - Der Empfänger braucht eine Interpretationsvorschrift.
  - Die Nachricht muß einen Informationsgehalt haben.

- Eine endliche Menge von Zeichen, die für die Darstellung einer Nachricht verwendet werden, heißt Zeichenvorrat. Ist ein Zeichenvorrat geordnet, so heißt er auch Alphabet.

Beispiele:	{A, B, C, ... ,Y, Z}	gewöhnliches Alphabet
	{0, 1, 2, ... , 8, 9}	Dezimalziffern
	{0, 1, 2, ... , 8, 9, A, B, C, D, E, F}	Hexadezimalziffern
	{0, 1}	binäres Alphabet

Informationstheorie

- Die Informationstheorie wurde 1948 von dem amerikanischen Mathematiker C. E. Shannon (1916-2001) begründet.

- Die Informationstheorie untersucht die mathematischen Zusammenhänge der in einer Nachricht enthaltenen Information, insbesondere bei Vorgängen der
  - Speicherung (zeitliche Transformation)
  - Übertragung (räumliche Transformation)

- Grundideen:
  - Die möglichen Zeichen in einer Nachricht treten mit bestimmten Wahrscheinlichkeiten auf.
  - Der Informationsgehalt eines Zeichens (einer Nachricht) ist hoch, wenn die Auftrittswahrscheinlichkeit gering ist.
    "Umso überraschender das Ereignis, desto größer die Information."
    Umgekehrt: Ist die Wahrscheinlichkeit des Auftretens eines Zeichens gleich 1 (damit sicher), so ist der Informationsgehalt 0.

Wörter über einem Alphabet

Definition:

Unter einem **Wort** der Länge n über einem Alphabet A versteht man ein n-Tupel $(a_1, a_2, \ldots, a_n)$ von Elementen a_i aus A. Es wird meist ohne Klammern und Kommata geschrieben: $a_1\, a_2 \ldots a_n$.

Bezeichnungen:

A^n bezeichne die Menge aller Wörter der Länge n über A.

$A^* := \bigcup_{i=0}^{\infty} A^i$ bezeichne die Menge aller Wörter über A von beliebiger (aber endlicher) Länge. Dabei enthält A^0 nur das "leere Wort" $\varnothing$.

Bemerkung: Häufig interessiert man sich nur für eine Teilmenge von A* (siehe auch das Kapitel "Formale Sprachen").

Codierung und Codes

- Eine Codierung ist eine Abbildungsvorschrift, die jedem Zeichen eines Zeichenvorrats A (Urbildmenge) eindeutig ein Zeichen oder ein Wort aus einem (meist anderen) Zeichenvorrat B (Bildmenge) zuordnet. Formal:

Definition:

Seien A und B (endliche) Zeichenvorräte. Eine Codierung von A ist eine *injektive* Abbildung

$$c: A \dashrightarrow \bigcup_{i=1}^{N} B^i$$

Das Bild c(A) heißt Code, seine Elemente $b \in c(A)$ Codewörter.

- Eine Codierung c läßt sich fortsetzen zu einer Abbildung

$$c^*: A^* \dashrightarrow B^*, \quad c^*(a_1\, a_2 \ldots a_n) := c(a_1)\, c(a_2) \ldots c(a_n)$$

auf allen Wörtern über A (die ebenfalls Codierung genannt wird). Die fortgesetzte Abbildung c* ist *im allgemeinen nicht mehr injektiv*!

Weitere Begriffe und Beispiele

- Eine Codierung c: A ----> B^n heißt eine Codierung mit fester Wortlänge n (auch Block-Code genannt).

  Jede andere Codierung hat variable Wortlänge.

- Für jede Codierung c : A ----> B^n mit fester Wortlänge ist auch die fortgesetzte Abbildung c*: A* ------> B* injektiv.

- Eine Binärcodierung ist eine Codierung mit der Bildmenge {0,1}:

$$c: A \dashrightarrow \bigcup_{i=1}^{N} \{0,1\}^i$$

Beispiele für Binärcodes auf der Urbildmenge {A,B, ... ,Z}:

- Morsecode (variable Wortlänge)
- ASCII- und EBCDIC-Codes (feste Wortlänge)

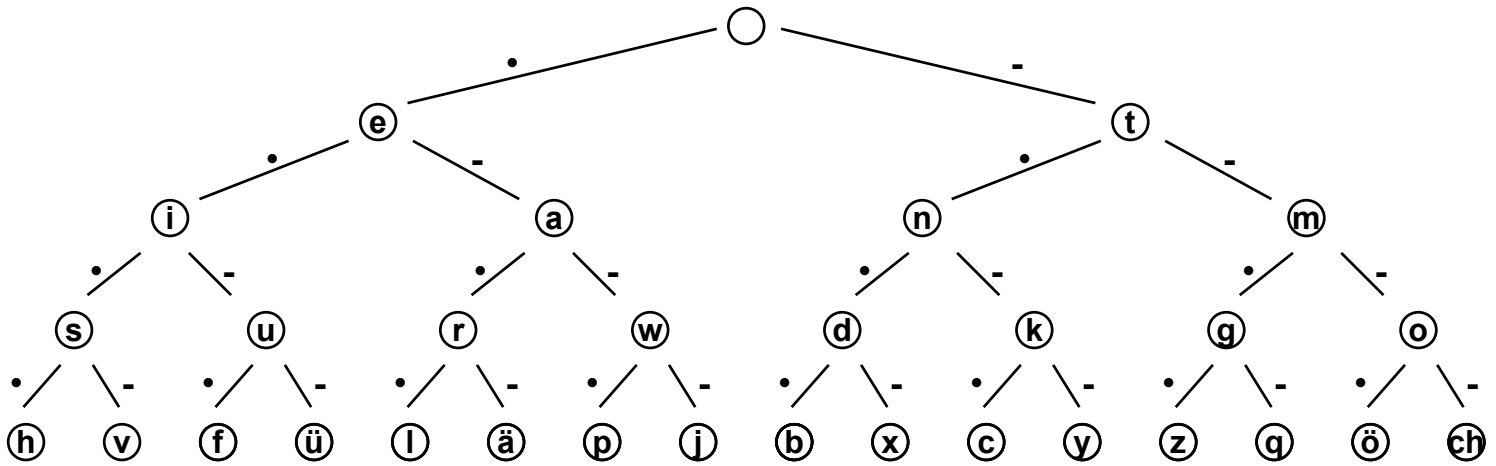
Ein Code mit variabler Wortlänge

Morse-Code: **Zeichenvorrat:** { ·, - , ‾‾‾ } **Pause**

a	· -	d	- · ·	g	- - ·
b	- · · ·	e	·	h	· · · ·
c	- · - ·	f	· · - ·	i	· ·

Signalfolgen: ee ⎍⎍⎍⎍ i ⎍⎍⎍⎍

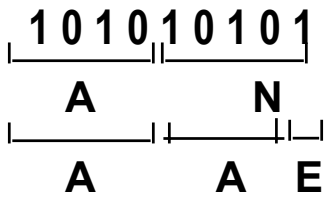
Codebaum



Präfix-Codes

Bei Codes variabler Wortlänge ohne Pausen kann die fortgesetzte Abbildung c *: A* ------> B* mehrdeutig, d.h. nicht injektiv, sein.

Beispiel: $E \rightarrow 1, \quad T \rightarrow 110, \quad A \rightarrow 1010, \quad N \rightarrow 10101$



nicht eindeutig!

Definition: Ein Code heißt **Präfix-Code**, falls kein Codewort Anfang eines anderen Codewortes ist.

Beispiel: $E \rightarrow 1, \quad T \rightarrow 011, \quad A \rightarrow 0101, \quad N \rightarrow 0100$

Bemerkung: Jeder Code mit fester Wortlänge ist ein Präfix-Code.

Satz: Bei Präfixcodes ist auch die fortgesetzte Abbildung c *: A* ------> B* injektiv.

Codes fester Wortlänge für Dezimalziffern

- **BCD-Code:** n --> Binärdarstellung von n (in 4 bit)
  - Ein Stellenwertcode, und zwar 8 - 4 - 2 - 1
- **Exzeß-3-Code:** n --> Binärdarstellung von n+3 (in 4 bit)
  - Ein Komplementärcode, d.h. für jedes Codewort ist auch dessen Einerkomplement ein Codewort.
- **Aiken-Code:** 4-bit-Code
  - Ein Stellenwertcode, und zwar 2 - 4 - 2 - 1
  - Ein Komplementärcode
- **Gray-Code:** 4-bit-Code
  - Die Codewörter aufeinanderfolgender Dezimalziffern unterscheiden sich jeweils nur an einer Bit-Stelle (einschrittiger Code).
- **CCITT2-Code:** 5-bit-Code (wurde für Lochstreifen verwendet)
- **Biquinär-Code:** 6-bit-Code
- **1-aus-10-Code:** 10-bit-Code
- **ASCII-Code:** 7- oder 8-bit-Code
- **EBCDIC-Code:** 8-bit-Code

ASCII- und EBCDIC-Code

- Codierungen von
  - Buchstaben (große und kleine)
  - Dezimalziffern
  - Sonderzeichen (z.B. ?, @, (, % etc.)
  - Steuerzeichen (z.B. LF=Line Feed, BS=Backspace etc.)

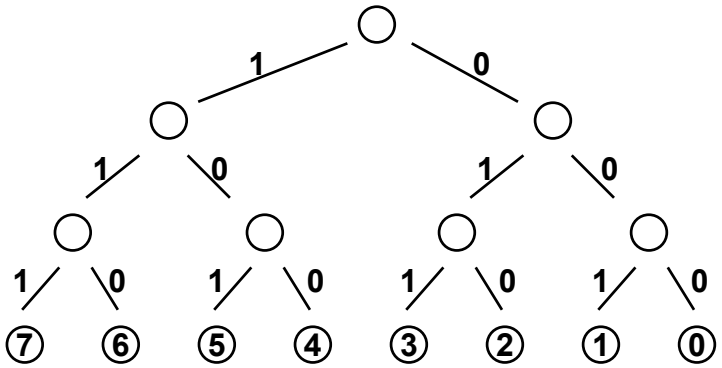
- EBCDIC-Code
  - Ein 8-bit-Code
  - In der IBM-Welt gebräuchlich

- ASCII-Code
  - Ein 7-bit-Code
  - Diverse 8-bit-Erweiterungen zur Darstellung länderspezifischer Zeichen (z.B. ä, ö, ü, ß oder é, è, î etc.)

Binärer Codebaum und Decodierung

- Jeder binäre Code kann als binärer Codebaum (auch binärer Entscheidungsbaum genannt) dargestellt werden.

Beispiel: Code-Baum für einen 3-bit-Teilcode des BCD-Codes



1: nach links
0: nach rechts

- Unter Decodierung versteht man die Zuordnung des zu einem Codewort gehörenden semantischen Zeichens aus der Urbildmenge, also die Anwendung der inversen Abbildungsvorschrift eines Codes.

  Die Decodierung kann leicht mit Hilfe des binären Entscheidungsbaums erfolgen.

Effizienz bzw. Redundanz von Codes

- Oft ist es wünschenswert, daß die codierten Daten möglichst wenig Platz einnehmen, d.h. gesucht ist ein Code mit hoher Effizienz (Stichwort: Datenkompression).

- Eine Redundanz des Codes (d.h. der Code enthält zusätzliche Bits, die keine Information vermitteln) kann aber auch wünschenswert sein und genutzt werden, um Fehler bei der Codierung und Speicherung bzw. Übertragung von Daten zu erkennen oder zu beheben.

Beispiel: Natürliche Sprachen haben eine hohe Redundanz. Ein Satz wird auch dann noch verstanden, wenn er viele Fehler enthält:

"Ds st n Stz, dr knn Vkl nthlt"

- Sicherung gegen Fehler kann erreicht werden durch
  - fehlererkennende Codes (error detecting codes)
  - fehlerkorrigierende Codes (error correcting codes)

Code-Redundanz

Definition: Sei ZV ein Zeichenvorrat mit der Mächtigkeit |ZV|. Der Entscheidungsgehalt von ZV ist definiert als

$$EG = \log_2 |ZV|$$

- Redundanz tritt auf, wenn der physikalische Zeichenvorrat größer ist als der semantische.
- Sei |ZVS| die Mächtigkeit des semantischen Zeichenvorrats, |ZVP| die des physikalischen. Es muß gelten:

  $|ZVP| \geq |ZVS|$ (Gleichheit bedeutet: keine Redundanz)

Definition: Sei EGP der Entscheidungsgehalt des physikalischen, EGS der des semantischen Zeichenvorrats eines Codes.

Die (absolute) Redundanz des Codes ist definiert als

$$R = EGP - EGS = \log_2 |ZVP| - \log_2 |ZVS| = \log_2 (|ZVP| / |ZVS|) \quad [bit]$$

Die relative Redundanz ist definiert als R / EGP.

Beispiel: Redundanz des BCD-Codes

- Ein Zeichenvorrat ZV aus 2^n Zeichen hat den Entscheidungsgehalt
  $EG = \log_2 2^n = n$ [bit]. Bei einer Binärcodierung von ZV ist dies
  - die Anzahl n der benötigten Bits,
  - die Anzahl der bei der Decodierung im binären Entscheidungsbaum notwendigen Entscheidungen.

Beispiel: BCD-Code

$EGS = \log_2 10 = 3{,}32$ [bit]

$EGP = \log_2 24 = 4$ [bit]

Die absolute Redundanz ist: $R = EGP - EGS = 0{,}68$ [bit]

Die relative Redundanz ist: $r = (EGP - EGS)/EGP = 0{,}68 / 4 = 17$ [%]

Dieser Wert kann als nicht ausgenutzte Codekapazität interpretiert werden (unbenutzte Codewörter für die Pseudotetraden).

Statistische Codierung

- Häufig treten die verschiedenen Zeichen einer Quelle mit unterschiedlicher Wahrscheinlichkeit in einer Nachricht auf.

Beispiel: In jeder natürlichen Sprache treten die einzelnen Buchstaben des Alphabets verschieden häufig auf.

Bei einer **statistischen Codierung** werden die Auftretungswahrscheinlichkeiten der Quellenzeichen berücksichtigt, um einen möglichst effizienten Code zu finden.

Prinzip: Häufig zu erwartende Zeichen werden durch kurze Codewörter, seltenere Zeichen durch längere Codewörter beschrieben.

- Bei **adaptiven Codes** wird die aktuelle bisherige Auftretungswahrscheinlichkeit der Zeichen in einer Nachricht verwendet, um den Code dynamisch zu verändern.
- Bei der Bestimmung der Effizienz eines Codes müssen die Auftretungswahrscheinlichkeiten sinnvoll berücksichtigt werden.

Shannon'sche Nachrichtenquelle

Definition: Eine **Shannon'sche Nachrichtenquelle** (Nachrichtenquelle ohne Gedächtnis) ist ein Zeichenvorrat ZV mit folgenden Eigenschaften:

- Jedes Element $a_i \in ZV$ hat eine zeitlich konstante Auftretungswahrscheinlichkeit p_i mit $0 < p_i \leq 1$, $\sum_i p_i = 1$.
- Die Wahrscheinlichkeit des Auftretens der Zeichenfolge $a_i\, a_k$ ist das Produkt der Wahrscheinlichkeiten $p_i\, p_k$ der einzelnen Zeichen (Unabhängigkeit des Auftretens eines Zeichens von den vorhergehenden Zeichen).

Beispiel:

ZV =	{ a,	b,	c,	d,	e,	f,	g }
p_i =	1/4	1/8	1/8	1/16	1/4	1/8	1/16

Eine mögliche statistische Codierung ist:

a --> 00, e --> 01
b --> 100, c --> 110, f --> 111
d --> 1010, g --> 1011

(Diese Codierung ergibt einen Präfix-Code.)

Informationsgehalt und Entropie

Definition: Der **Informationsgehalt** eines Zeichens mit Wahrscheinlichkeit p_i ist definiert als $K_i = \log2 (1/ p_i)$.

Bemerkung: Der Informationsgehalt einer Zeichenfolge $a_i\, a_k$ ist $K_i + K_k$.

Definition: Die **Entropie** (oder der mittlere Informationsgehalt) einer Nachrichtenquelle ist definiert als

$$H = \sum_i p_i\ K_i = \sum_i p_i \log_2 (1/p_i) \quad [\text{bit}]$$

Bemerkung: Für eine Nachrichtenquelle mit n Zeichen gilt für beliebige Werte der Wahrscheinlichkeiten p_i :

$$H = \sum_i p_i \ \log2 (1/p_i) \ \leq \ \log_2 n$$

Gleichheit gilt, wenn alle Zeichen die gleiche Wahrscheinlichkeit 1/n haben.

Shannon'sches Codierungstheorem

Definition: Für eine Codierung einer Shannon'schen Nachrichtenquelle sei l_i die Länge des Codewortes für das i-te Zeichen. Die mittlere Wortlänge der Codierung ist definiert als

$$L = \sum_i p_i l_i$$

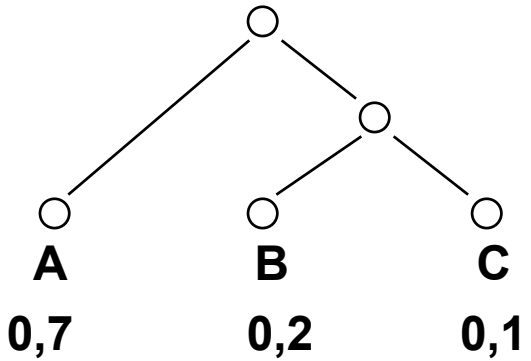
Die Redundanz des Codes ist definiert als $R = L - H$.

Shannon'sches Codierungstheorem

1. Für jede Codierung einer Shannon'schen Nachrichtenquelle ist $H \leq L$.
2. Jede Shannon'sche Nachrichtenquelle kann so codiert werden, daß die Differenz $L - H$ beliebig klein wird.

Bemerkung: Wenn in jedem Knoten des Entscheidungsbaums die Verzweigungen gleich wahrscheinlich sind, gilt: $H = L = \log_2 n$.

Ein Beispiel



$$H = \sum_{i=1}^{3} p_i \log_2 (1/p_i) = 1{,}157$$

$$L = \sum_{i=1}^{3} p_i l_i = 0{,}7 \cdot 1 + 0{,}2 \cdot 2 + 0{,}1 \cdot 2 = 1{,}3$$

Die Coderedundanz (das Maß für nutzlos vorhandene Bits) beträgt:

$$R = L - H = 1{,}3 - 1{,}157 = 0{,}143 \text{ [bit]} \quad (11\ \% \text{ relativ})$$

Huffman-Code

- **Algorithmus von Huffman**

Gegeben sei eine Shannon'sche Nachrichtenquelle E mit Elementen e_i und Wahrscheinlichkeiten $p(e_i)$.

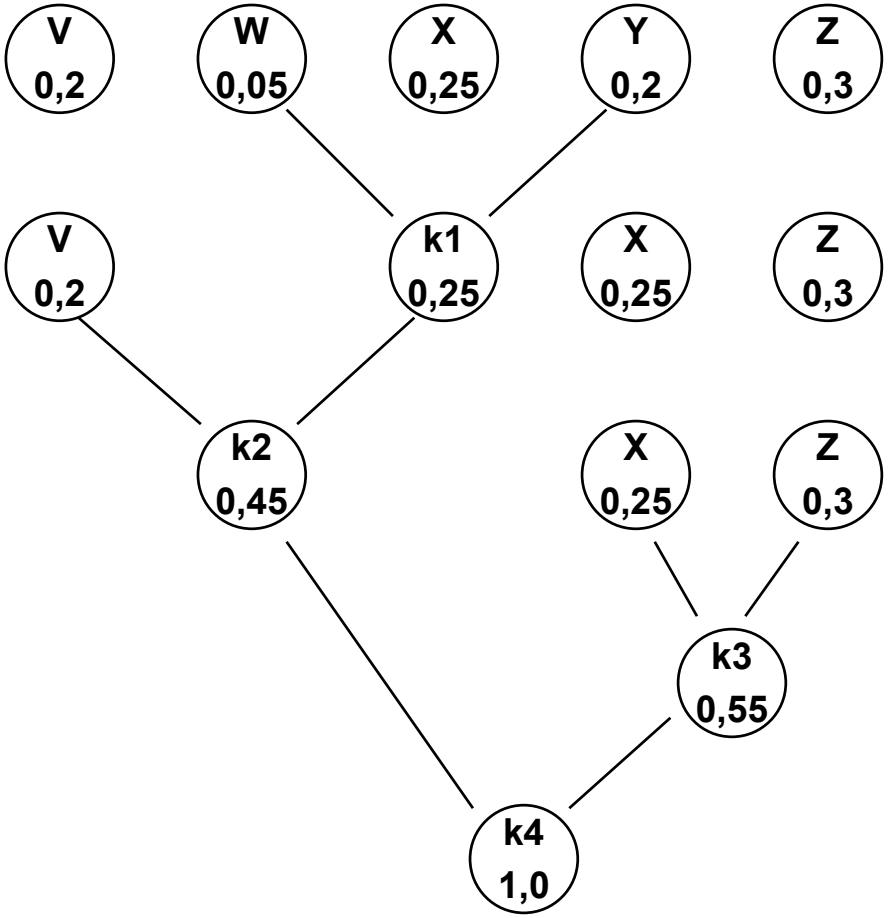
1. **Fasse zwei Elemente e_i und e_k mit den kleinsten Wahrscheinlichkeiten zu einem Knoten k_m zusammen mit Wahrscheinlichkeit $p(k_m) = p(e_i) + p(e_k)$.**

2. **Ersetze die Elemente e_i und e_k in E durch k_m und wiederhole Punkt 1. bis nur noch ein Element übrigbleibt.**

3. **Setze den Codebaum zusammen.**

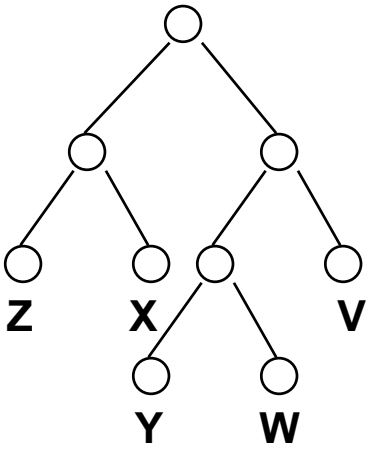
Satz: Für eine gegebene Shannon'sche Informationsquelle hat keine Codierung der Einzelzeichen eine kleinere mittlere Wortlänge als die Huffman-Codierung.

Beispiel zum Huffman-Code



E = { V, W, X, Y, Z}
0,2 0,05 0,25 0,2 0,3

Codebaum



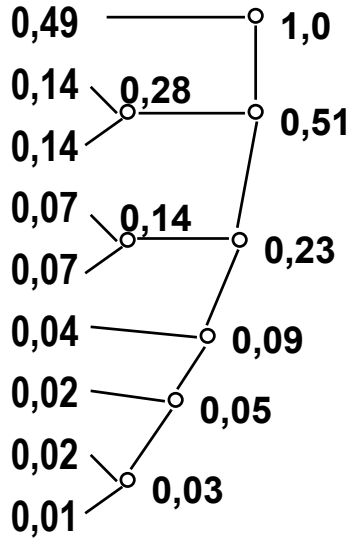
Wort-Codierung

- Um die Differenz H - L möglichst klein zu machen, kann es nötig sein, nicht einzelne Zeichen, sondern Wörter der Länge > 2 zu codieren.

Beispiel: Codierung von Zeichenpaaren

ZV = { A, B, C}, p(A) = 0,7, p(B) = 0,2, p(C) = 0,1

Paar	p	Codierung	Länge
AA	0,49	0	1
AB	0,14	100	3
BA	0,14	101	3
AC	0,07	1100	4
CA	0,07	1101	4
BB	0,04	1110	4
BC	0,02	11110	5
CB	0,02	111110	6
CC	0,01	111111	6



Mittlere Wortlänge:

$L_2 = 2{,}33$ (pro Paar)

$L_1 = 1{,}165$ (pro Zeichen)

relative Redundanz:

0,7 %

Fehlererkennung und -behebung

- Eine Redundanz des Codes kann wünschenswert sein und dazu genutzt werden, Fehler bei der Codierung und Speicherung bzw. Übertragung von Daten zu erkennen und/oder zu beheben.

- Nicht jede Code-Redundanz kann zur Fehlererkennung genutzt werden.

Beispiel: Die relative Redundanz von 17% des BCD-Codes (s.o.) kann nicht zur Fehlererkennung genutzt werden.

- Häufig wird eine bereits codierte Nachricht nachträglich mit zusätzlichen Bits versehen, um einen fehlererkennenden oder fehlerkorrigierenden Code zu erhalten.
  - Die Methode ist unabhängig von der ursprünglichen Codierung.
  - Meist wird die Nachricht in Abschnitte fester Länge aufgeteilt und zusätzliche Bits (sog. Prüfbits) pro Abschnitt eingefügt.

Stellendistanz und Hammingdistanz

Definition: Die **Stellendistanz** d(v,w) zweier Codewörter v, w gleicher Länge ist die Anzahl der Positionen, an denen sich die beiden Wörter unterscheiden.

Beispiele:

0 0 0 0 und 0 0 0 1 haben Stellendistanz 1

101101 und 010101 haben Stellendistanz 3

Definition: Sei $c: A \dashrightarrow B^n$ eine Codierung mit fester Wortlänge. Die **Hammingdistanz** h von c ist definiert als die minimale Stellendistanz aller Paare verschiedener Codewörter von c, also

$$h = \min \{ d(v,w);\ v, w \in c(A),\ v \neq w \}$$

Bemerkung: Für zwei beliebige Codewörter eines Codes ist stets $d \geq h$.

Hammingdistanz und Fehlererkennung

- **Die Hammingdistanz gibt an, wieviele Fehler (falsche Zeichen) ausreichen können, um ein neues Codewort zu erhalten.**

Satz: **Für jeden Code mit fester Wortlänge gilt:**

a) Die Zahl der erkennbaren Fehler ist h - 1.

b) Die Zahl der korrigierbaren Fehler ist [(h-1)/2]
(größte ganze Zahl ≤ (h-1)/2).

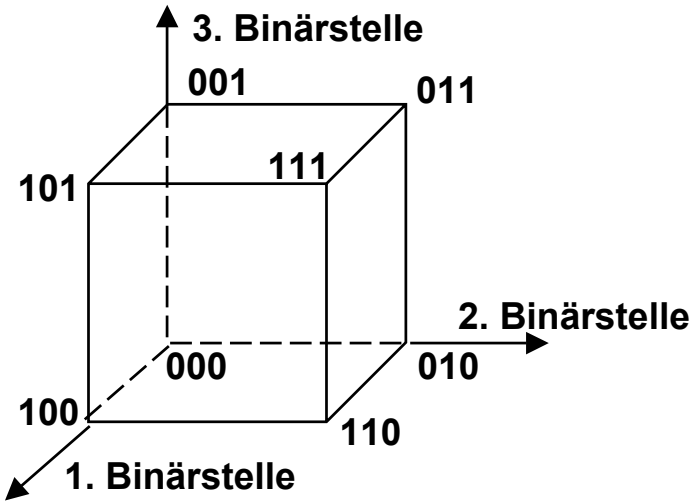
- **Die Hammingdistanz eines Binärcodes mit ungerader Hammingdistanz h = 2k-1 kann durch ein Paritätsbit um 1 erhöht werden. Bei gerader Parität erhält das Paritätsbit**
  - **den Wert 0, wenn die Zahl der Einsen im Codewort gerade ist,**
  - **den Wert 1, wenn die Zahl der Einsen im Codewort ungerade ist.**

  Alle neuen Codewörter haben dann eine gerade Zahl von Einsen. Insbesondere kann auf diese Weise ein Binärcode mit h=1 zu einem 1-bit-fehlererkennenden Code erweitert werden.

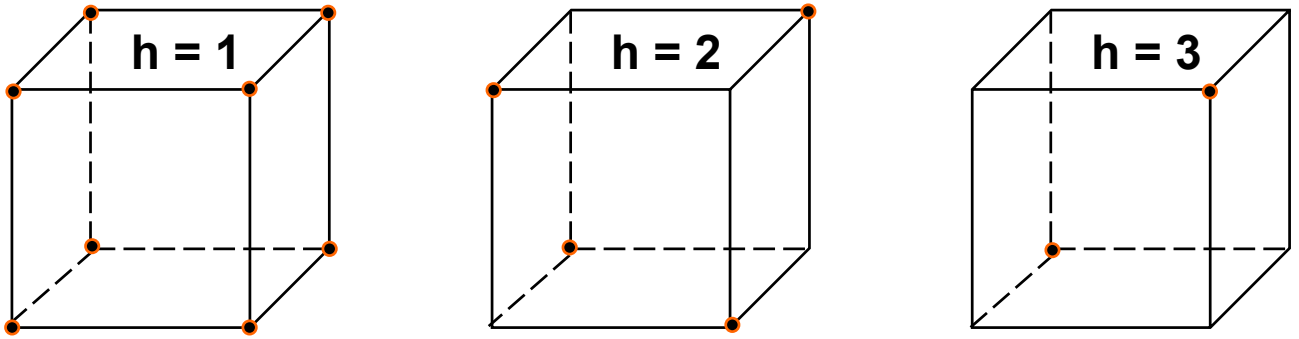
Räumliche Darstellung von Stellendistanz und Hammingdistanz

- Räumliche Darstellung der Stellendistanz d:

Kante:	d = 1
Flächendiagonale:	d = 2
Raumdiagonale:	d = 3



- Räumliche Darstellung der Hammingdistanzen h = 1, h = 2, h = 3:



- genutzte Codewörter

Cyclic Redundancy Check (CRC) Codes

- CRC-Codes sind zyklische Codes, d.h. jede zyklische Vertauschung eines Codewortes ist wieder ein Codewort.

- Man kann zeigen, daß zyklische Codes genau die Codes sind, deren Konstruktion im folgenden beschrieben wird.

Grundidee:

- Ein binäres Wort $a_{n-1} \ldots a_0$ wird dem Polynom $a_{n-1} x^{n-1} + \ldots + a_0 x^0$ zugeordnet.
- Gegeben ist ein erzeugendes Polynom G(x) vom Grad r (mit Koeffizienten 0 und 1).
- Ein Codewort $a_{m-1} \ldots a_0$ wird mit r Prüfbits $p_r \ldots p_1$ so ergänzt, daß das zum binären Wort $a_{m-1} \ldots a_0\ p_r \ldots p_1$ gehörende Polynom durch G(x) teilbar ist (dabei wird die Division modulo 2 ausgeführt).
- Der Empfänger dividiert das zum erhaltenen Wort gehörende Polynom durch G(x). Tritt dabei ein Rest $\neq 0$ auf, liegt ein Fehler vor.

Bestimmen der Prüfbits bei CRC-Codes

1. Hänge an das zu übertragende Wort r (= Grad von G(x)) Nullen an.

2. Dividiere das zum entstandenen Wort gehörende Polynom durch G(x) (Rechnung modulo 2). Benötigt wird nur der Rest der Division.

3. Subtrahiere den entstandenen Rest von dem Polynom. Das zu dem jetzt entstandenen Polynom gehörende Binärwort wird übertragen.

   Der Rest ist ein Polynom vom Grad $\leq$(r-1), die Subtraktion ändert also nur die letzten r Bits des Wortes, nicht die ersten m Bit, die das ursprüngliche Wort enthalten.

Beispiel: Das erzeugende Polynom sei $G(x) = x^3 + x^2 + 1$, das zu übertragende Wort sei 0110.

1. Betrachte 0110000, was dem Polynom $x^5 + x^4$ entspricht.
2. $(x^5 + x^4) / (x^3 + x^2 + 1) = x^2$ Rest x^2
3. $x^5 + x^4 - x^2 = x^5 + x^4 + x^2$ entspricht dem Binärwort 0110100.

Was leisten CRC-Codes?

- Das (korrekt) übertragene Wort entspreche dem Polynom T(x). Falls der Empfänger das Polynom T(x) + E(x) erhält, rechnet er:

  Rest von (T(x) + E(x)) / G(x) = 0 + Rest von E(x) / G(x)

  Er erhält somit einen von 0 verschiedenen Rest (und erkennt somit einen Übertragungsfehler) genau dann, wenn der Fehler E(x) nicht durch G(x) teilbar ist.

- CRC-Codes sind somit fehlererkennend (aber nicht fehlerkorrigierend).

  Wie alle fehlererkennenden Codes können sie dann sinnvoll eingesetzt werden, wenn bei Erkennen eines Fehler eine Wiederholung der Übertragung durchgeführt werden kann.

Beispiel: CRC-Codes werden z.B. bei der Speicherung von Daten auf Magnetbändern eingesetzt.

Beispiele für weitere Prüfzeichenverfahren

- Bei ISBN-Nummern (International Standard Book Number) ist die letzte von 10 Ziffern eine Prüfziffer:

  $$a_{10} = (a_1 + 2a_2 + 3a_3 + \ldots + 9a_9) \mod 11$$

  (Ist der Wert 10, so wird das Zeichen X verwendet.)

- Bei Barcodes (EAN = European Article Number) ist die letzte von 13 Ziffern eine Prüfziffer:

  $$a_{13} = (a_1 + 3\,a_2 + a_3 + 3a_4 + \ldots + a_{11} + 3a_{12}) \mod 10$$

- Die obigen beiden Beispiele können Einzelfehler erkennen.

- Mit Methoden der Algebra (Gruppentheorie) kann man allgemeinere Prüfzeichenverfahren finden, die nicht nur Einzelfehler, sondern z.B. auch die Vertauschung benachbarter Zeichen erkennen können.

  Beispiel: Die Nummern der deutschen Banknoten verwendeten ein Prüfzeichenverfahren, das auf der sog. Diedergruppe D_5 basiert.

Fehlerkorrigierende Codes

- **Um einzelne Bit-Fehler korrigieren zu können, muß die Hamming-Distanz des Codes mindestens 3 sein.**

- **Angenommen, wir wollen Binärwörter der Länge m mit r Prüfbits zu einem 1-bit-fehlerkorrigierenden Code ergänzen (neue Wortlänge also n=m+r).**

  Jedes Binärwort der Länge n hat n Nachbarn mit Stellendistanz 1. Die Codewörter der 2^m Binärwörter der Länge m "belegen" somit $(n+1) \cdot 2^m$ der 2^n existierenden Binärwörter, also muß gelten: $(n+1) \cdot 2^m \leq 2^n$, bzw. mit n = m+r:

  $$(m + r + 1) \leq 2^r$$

  Dies ergibt eine untere Schranke für die Anzahl der nötigen Prüfbits.

- **Diese untere Schranke für die Anzahl von Prüfbits wird vom Hamming-Code tatsächlich erreicht.**

Hamming-Code

- Der **Hamming-Code** wird wie folgt gebildet:
  - Die Bits der Codewörter seien von links nach rechts mit 1, 2, ..., n durchnumeriert.
  - Die Bits 1, 2, 4, 8, 16, ... (Zweierpotenzen) dienen als Prüfbits.
  - Die restlichen Bits 3, 5, 6, 7, 9, ... enthalten die "Nutz"-Bits.
  - Das Prüfbit Nummer 2^n ist das Paritätsbit für die Menge derjenigen Datenbits, die eine Nummer haben, in deren Binärdarstellung der Summand 2^n vorkommt.
- Der Empfänger eines Codewortes
  - berechnet die Prüfbits k (k = 1, 2, 4, 8, ...) für das erhaltene Wort,
  - summiert die Nummern derjenigen Prüfbits, deren übertragener Wert nicht mit dem berechneten Wert übereinstimmt.
  - Falls es fehlerhafte Prüfbits gibt, gibt die Summe ihrer Nummern die Nummer des falsch übertragenen Bits an.

Beispiel: 4 Datenbits, 3 Prüfbits. Anordnung: $p_1\ p_2\ a_3\ p_4\ a_5\ a_6\ a_7$.

$p_1 = a_3 \oplus a_5 \oplus a_7$

$p_2 = a_3 \oplus a_6 \oplus a_7$

$p_4 = a_5 \oplus a_6 \oplus a_7$

$\oplus$ bezeichnet die Operation zur Bildung des Paritätsbits (XOR)