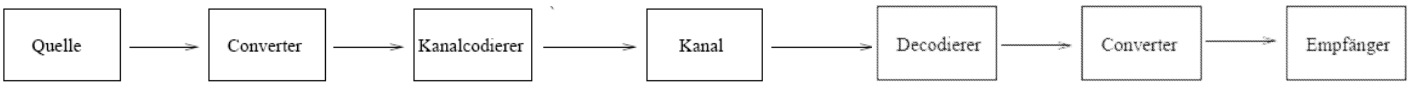
I. Einführung in die Codierungstheorie

1. Allgemeines

Codierung: Sicherung von Daten und Nachrichten gegen zufällige Fehler bei der Übertragung oder Speicherung.



Ziel der Codierung: Möglichst viele bei der Übertragung aufgetretene (zufällige) Fehler sollen bei der Decodierung erkannt werden und evtl. sogar korrigiert werden.

Verwandte Gebiete:

1. Quellencodierung: Nachricht wird so codiert, dass häufig auftretende Strings in kurzer Form codiert werden und seltenere in längerer Form (Datenkompression).
2. Kryptographie: Sicherung von Nachrichten/Daten gegen Abhören oder Änderungen durch unbefugte Dritte (Datenverschlüsselung).

Zwei Prinzipien der Kanalcodierung:

- *FEC-Verfahren (Forward Error Correction):*
  Aufgetretene Fehler werden (aufgrund der zugefügten Redundanz) erkannt und korrigiert. (Vorteil: keine Verzögerung bei Übertragung; aber gegebenenfalls große Redundanz notwendig!)
- *ARQ-Verfahren (Automatic Repeat Request)*
  Aufgetretene Fehler sollen erkannt werden, werden aber nicht korrigiert. Stattdessen wird eine Wiederholung der Übertragung beim Sender angefordert. (Vorteil: geringe Redundanz; aber ggf. erhebliche Verzögerung bei wiederholter Übertragung.)

2. Definitionen

2.1. Alphabet

Unter einem Alphabet A versteht man den Zeichenvorrat der Quelle.

2.2. Wort

Unter einem Wort der Länge $n \in N$ über einem Alphabet A versteht man ein n-Tupel $(c_1, c_2, \ldots, c_n)$ von Elementen aus A.

2.3. Code

Seinen A und B nichtleere Mengen und $n \in N$, dann läst sich eine injektive Abbildung $C^*: A \rightarrow \cup B^i$ zu einer Abbildung C von der Menge A* der Wörter über A wie folgt fortsetzen: $C(a_1a_2...a_n) = C^*(a_1)\, C^*(a_2)...C^*(a_n)$ $(C(\varnothing) = \varnothing)$
C heißt Codierung, das Bild von C Code und seine Elemente Codewörter.

2.4. Präfixcode (irreduzibler Code)

Ein Code C heißt Präfixcode, wenn kein Codewort aus Bild(C) Präfix eines anderen Codeworts von C ist.

→ eindeutig decodierbar

Bsp.: Paritätsbit

2.5. Fehlertypen

Fehlertyp		**rel. Häufigkeit / %**
Einzelfehler (1 Ziffer falsch)	a -> b	79
(Nachbar-) Transposition	ab -> ba	10,2
Sprungtransposition	acb -> bca	0,8
Zwillingsfehler	aa -> bb	0,6

2.6. Prüfziffercodierung

Seien $a_1,a_2...a_{n-1}$ ($n \geq 2$) ein Wort über dem Alphabet A= {0,...,9} dann ist eine *Prüfziffercodierung modulo 10 mit einem redundanten Prüfbit* $a_n \in A$ definiert durch n Permutationen $\delta_1, ..., \delta_n$ von A zusammen mit der Kontrollgleichung

$\sum \delta_i(a_i) \equiv c \pmod{10}$.

Beispiele: ISBN, EAN

3. Blockcodes

3.1. Definition Blockcode

Es sei A eine endliche Menge (Alphabet) und $n \in \mathbb{N}$. Ein *Blockcode* $\mathcal{C}$ der *(Block-)Länge* n über A ist eine Teilmenge von $A^n = \underbrace{A \times \ldots \times A}_{\leftarrow n \rightarrow}$.

Die Elemente von $\mathcal{C}$ heißen *Codewörter*.
Ist $|A| = 2$ (i.Allg. $A = \{0,1\}$), so heißt $\mathcal{C}$ *binärer (Block-)Code*.

3.2. Hamming-Abstand

Sei A ein endliches Alphabet, $n \in \mathbb{N}$.
Für $a = (a_1, \ldots, a_n),\ b = (b_1, \ldots, b_n) \in A^n$ sei

$$d(a,b) = \#\{i :\ 1 \leq i \leq n,\ a_i \neq b_i\}$$

d heißt *Hamming-Abstand* von a und b.

3.2.1. d ist eine Metrik, da:

(1) $0 \leq d(a,b) \leq n \quad$ *für alle* $a, b \in A^n$

(2) $d(a,b) = 0 \Leftrightarrow a = b \quad$ *für alle* $a, b \in A^n$

(3) $d(a,b) = d(b,a) \quad$ *für alle* $a, b \in A^n$

(4) $d(a,b) \leq d(a,c) + d(c,b) \quad$ *für alle* $a, b, c \in A^n$ *(Dreiecks-Ungleichung)*

Wird ein Wort x gesendet und Wort y empfangen, so bestimmt der Hammingabstand d(x,y) = k die Anzahl der aufgetretenen Fehler.

3.2.2. Hamming-Decodierung (Maximum-Likelyhood)

Wird ein Wort $y \in A^n$ empfangen, so wird y als ein x' decodiert mit

$$d(x', y) = \min_{x \in \mathcal{C}} d(x, y).$$

3.3. Definition Kugel

Ist $z \in A^n$, $r \in N_0$, dann heißt $K_r(z) := \{ x \in A^n \mid d(x,z) \leq r \}$ Kugel vom Radius r um z.

3.4. Minimalabstand

Der *Minimalabstand* von $\mathcal{C}$ ist:

$$d(\mathcal{C}) = \min_{\substack{x, x' \in \mathcal{C} \\ x \neq x'}} d(x, x')$$

3.4.1. r-Fehler-erkennend

Ein Blockcode C heißt r-Fehler-erkennend, falls $d(C) \geq r+1$.

3.4.2. r-Fehler-korrigierend

Ein Blockcode C heißt r-Fehler-korrigierend, falls **$d(C) \geq 2r+1$**.

3.5. Perfekter Code

3.4.1. Kugelpackung

Eine Kugelpackung bezeichnet eine disjunkte Überdeckung von A^n (bzw. $A^* \subset A^n$) durch Kugeln des Radius r. Die Kugelmittelpunkte als Codewörter gewählt ergeben einen r-fehlerkorrigierenden Code.

3.4.3. Perfekter Code

Sei $\mathcal{C}$ ein Code der Länge n über A. $\mathcal{C}$ heißt *perfekt*, falls ein $e \in \mathbb{N}$ existiert mit:

1. $K_e(x) \cap K_e(x') = \emptyset \quad \forall\, x, x' \in \mathcal{C},\ x \neq x'$
2. $A^n = \bigcup_{x \in \mathcal{C}} K_e(x)$.

3.4.2. Kugelpackungsschranke

Sei $\mathcal{C}$ ein Code der Länge n über A, $|A| = q$. Sei $e \in \mathbb{N}_0$ maximal mit $d(\mathcal{C}) \geq 2e+1$.

(a) (Hamming-Schranke, Kugelpackungsschranke) Es gilt:

$$|\mathcal{C}| \leq \frac{q^n}{\sum_{j=0}^{e} \binom{n}{j}(q-1)^j}$$

(b) $\mathcal{C}$ ist genau dann ein perfekter Code, wenn in (a) die Gleichheit gilt:

$$|\mathcal{C}| = \frac{q^n}{\sum_{j=0}^{e} \binom{n}{j}(q-1)^j}$$

4. Lineare Codes

4.1. Definitionen

4.1.1. Gewicht

K endlicher Körper.

(a) $x \in K^n$, so heißt $wt(x) = d(x,0) = |\{i \mid x_i \neq 0\}|$ das *Gewicht* von x.

(b) Ist $\{0\} \neq \mathcal{C} \subseteq K^n$, so heißt $wt(\mathcal{C}) = \min_{0 \neq x \in \mathcal{C}} wt(x)$ das *Minimalgewicht* von $\mathcal{C}$.

4.1.2. Linearer Code

Sei K ein endlicher Körper und $n \in \mathbb{N}$. Ein *linearer Code* $\mathcal{C}$ ist ein Unterraum des K-Vektorraums K^n.
Ist $\dim(\mathcal{C}) = k\,(\leq n)$ und $d(\mathcal{C}) = d$, so heißt $\mathcal{C}$ ein $[n,k]$-*Code* oder $[n,k,d]$-*Code über* K.

4.2. Darstellung

Ein linearer Code C kann angegeben werden durch die Angabe einer Basis von C, oder durch die Angabe eines linearen Gleichungssystems mit der Lösungsmenge C.

4.2.1. Erzeuger-, Basis- oder Generatormatrix

Sei $\mathcal{C}$ ein $[n,k]$-Code über K und
$g_1 = (g_{11}, \ldots, g_{1n}), \ldots, g_k = (g_{k1}, \ldots, g_{kn})$ eine Basis von $\mathcal{C}$. Dann heißt die $k \times n$-Matrix

$$G = \begin{pmatrix} g_1 \\ \vdots \\ g_k \end{pmatrix} = \begin{pmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & & \vdots \\ g_{k1} & \cdots & g_{kn} \end{pmatrix}$$

eine *Erzeugermatrix* für $\mathcal{C}$.

4.2.2. Kontroll-, Basis- oder Generatormatrix

Sei $\mathcal{C}$ ein $[n,k]$-Code über K

Ist H eine Matrix derart, dass C = { x ∈ K^n | H x^T = **0** } gilt, so heißt H eine Kontrollmatrix von C.

4.3. Singleton-Schranke, MDS-Code und Hamming-Code

4.3.1. Singleton-Schranke (ohne Beweis)

Ist C ein linearere [n,k]-Code über K und d(C) = d, dann gilt: $d \leq n - k - 1$.

4.3.2. MDS-Code (Maximum Distance Separable)

Ein linearere [n,k]-Code C über K mit d(C) = n – k – 1 heißt MDS-Code.

4.3.3. Hamming-Code

Sei q eine Primzahlpotenz, K Körper mit $|K| = q$.
Sei $l \in \mathbb{N}$, $n = \frac{q^l-1}{q-1}$. Dann existiert ein perfekter $[n, n-l]$-Code $\mathcal{C}$ über K mit $d(\mathcal{C}) = 3$, der *Hamming-Code*.

5. Gewichtsverteilung

5.1. Definition Weight Enumerator

Sei $\mathcal{C}$ linearer $[n,k]$-Code über K, $|K| = q$.
Für $0 \leq w \leq n$ sei A_w die Anzahl der Codewörter von Gewicht w.
Dann heißt das Polynom

$$\sum_{w=0}^{n} A_w z^w \in \mathbb{R}[z]$$

Gewichtsverteilungspolynom, oder Gewichtszähler (weight enumerator) von C.

5.2. Definition Dualer Code

Sei $\mathcal{C} \subseteq K^n$ ($\mathcal{C}$ braucht kein Unterraum zu sein). Dann heißt

$$\mathcal{C}^\perp = \{y \in K^n : < y, x > = 0 \text{ für alle } x \in \mathcal{C}\}$$

der *duale Code* zu $\mathcal{C}$. $\mathcal{C}^\perp$ ist immer ein Unterraum, also ein linearer Code, selbst wenn $\mathcal{C}$ nicht linear ist. $\mathcal{C}^\perp = \langle \mathcal{C} \rangle_K^\perp$.

5.2. Mac-Williams-Identität

Sei $\mathcal{C}$ ein $[n,k]$-Code über K, $|K| = q$, mit Gewichtszähler $A(z) = \sum_{w=0}^{n} A_w z^w$
Dann hat $\mathcal{C}^\perp$ den Gewichtszähler

$$B(z) = \frac{1}{q^k}(1 + (q-1)z)^n A\left(\frac{1-z}{1+(q-1)z}\right).$$

II. Codierung mittels Uncoverings

1.1 Wiederholung: Scharf k-transitive Gruppen

Sei G eine Permutationsgruppe die auf $\Omega = \{1, ..., n\}$ agiert. Dann heißt G scharf k-transitiv, wenn für zwei geordnete k-Tupel $\{x_1, ..., x_k\}, \{y_1, ..., y_k\} \in \{1, ..., n\}^k$ mit $x_i = x_j \Leftrightarrow i = j$, $y_i = y_j \Leftrightarrow i = j$ genau ein $g \in G$ gibt mit $(x_1, ..., x_k)g = (y_1, ..., y_k)$.

1.2 Wiederholung: Uncoverings

Eine Menge U von k-Teilmengen von $\Omega = \{1, ..., n\}$ heißt (n,k,r)-Uncovering, falls für jede r-Teilmenge R ein $K \in U$ existiert, mit K$\bigcap R = \emptyset$.

1.3 Fehlerkorrektur durch Uncoverings

Die maximale Korrektur durch den Unvocering-Algorithmus einer scharf k-transitiven Gruppe G vom Grad n liegt bei $r = \lfloor \frac{d-1}{2} \rfloor$, wobei d=n-k+1.

Nebenbemerkung: Die S_n ist scharf n-1-transitiv.
Damit gilt, d = n-(n-1)+ 1= 2. Somit ist $r = \lfloor \frac{2-1}{2} \rfloor = 0$.
Also bildet die S_n einen 0-Fehlerkorrigierenden Code.

1.4 Beispiele von Uncoverings

G=$PGL(2,7) \rightarrow n = 8, k = 3, r = \lfloor \frac{8-3+1-1}{2} \rfloor = \lfloor \frac{5}{2} \rfloor = 2.$

Finde also ein (8,3,2)-Uncovering. Zum Beispiel:
$$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 2 & 3 & 7 \\ 1 & 7 & 8 \end{matrix}.$$

1.5 Der Algorithmus Uncoverings

Starte, wähle dazu ein erstes k-Tupel.
Prüfe, ob dieses Element im Codebuch steht.
Hat es eine Fehlerdistanz kleiner r, so ist es das gesuchte Wort$\rightarrow$ *Ende*.
Hat es eine größere Fehlerdistanz, wähle das nächste k-Tupel.

1.6 Beispiel zu G=PGL(2,7)

g=(1 2 8 3 4 5 6 7) wird gesendet und
w=(1 2 8 3 4 1 5 7) wird empfangen. Betrachte das (8,3,2)-Uncovering.
Eintrag bei 1, 2, 3= 128.
Prüfe im Codebuch:
c=(1 2 8 3 4 5 6 7). r = 2. Es ist das gesuchte Wort.

1.7 Beispiel zur M_{12}

Die M_{12} <(1 2)(3 4)(5 6)(7 8)(9 10)(11 12),(1 3 2)(4 7 5)(8 9 11)>
ist scharf 5-transitiv. Der Minimalabstand beträgt: 12-5+1=8.
Damit ist M_{12} 3-Fehlerkorrigierend. Ein (12,5,3) Uncovering liegt in:

1	2	3	4	5
1	2	6	11	12
1	3	7	8	9
1	4	6	7	10
1	5	8	9	11
2	4	8	9	12 .
2	5	7	10	11
3	4	7	11	12
3	5	6	10	12
3	6	8	9	11
6	7	8	9	10

Sei nun g=1 2 3 4 5 6 7 8 9 10 11 12 gesendet und w=2 2 3 5 5 7 8 9 10 11 1 empfangen.

1.8 Permutation Decoding [25]

Gäbe es für einen t-Fehlerkorrigierenden Code C eine Untermenge $S \subset Aut(C)$, so dass ein Element$\sigma \in S$, wenn es auf das empfangene Wort angewendet wird, die Symbole, welche fehlerhaft sind, auf Nichtinformationspositionen (Redudanzpositionen) setzt. Nach Paritätscheck und Anwendung von σ^-1 erhält man das ursprünglich gesendete Wort.

1.9 Theorem

Sei C ein (n,k) t-Fehlerkorrigierender Linearcode mit Paritätskontrollmatrix H und der Identität I_{n-k} in Redundanzposition. Sei r=c+e mit c und $wt(e) \leq t$. Dann sind die Informationssymbole in r korrekt, $\Leftrightarrow wt(syn(r)) \leq t$, mit $syn(r) = Hr^T$.

1.10 Algorithmus Permutation Decoding

Sei $S = \{\sigma_1, ..., \sigma_p\}$ Permutationdecoding-Menge (PD-Menge) für C. Dann:

I. Für ein empfangenes Wort r, berechne $wt(H(r\sigma_i)^T$, für i=1,2,...
solange bis das Gewicht kleiner gleich t ist.
II. Durch Extrahieren der Informationssymbole von rσ_i
und Bestätigen der Paritätskontrollgleichungen finde die Redundanzsymbole und bilde das Codewort c.
III. r wird decodierd zu $c\sigma_i^-1$.

5. Literatur

[1] R.-H. Schulz, Codierungstheorie. Vieweg, 2003

[2] W. C. Huffman, Codes and groups, in *Handbook of Coding Theory*, (eds V. S. Pless and W. C. Huffman), Elsevier, Amsterdam, 1998.

[3] Robert Francis Bailey, Permutation Groups, Error-Correcting Codes and Uncoverings, Queen Mary, University of London, 2005

[4] Prof. Peter Hauck, Codierungstheorie, Skript zur Vorlesung im WS 2005/06, Tübingen 2005 (http://www-dm.informatik.uni-tuebingen.de/skripte/Codierungstheorie/CodierungstheorieWS0506.pdf)