4 Codierungstheorie

Wir beschäftigen uns mit dem Problem, Nachrichten über einen störungsanfälligen Kanal (z.B. Internet, Satelliten, Schall, Speichermedium) zu übertragen. Wichtigste Aufgabe in diesem Zusammenhang ist es, Methoden zu entwickeln, um die ursprünglich gesendete Nachricht aus der gestörten Nachricht zu rekonstruieren. Um dies zu erreichen, muss die Nachricht zusammen mit zusätzlicher, eigentlich redundanter Information gesendet werden.

4.0.1 Definition. Sei X eine Menge (von zu übertragenden Informationen). Dann legt jede injektive Abbildung $g : X \to Y$ (die *Generatorabbildung*) einen *Code* fest. Die Menge $g(X) = \{\, g(x) \mid x \in X \,\}$ heißt die Menge der *Codewörter* oder einfach der *Code*.

Eine Abbildung $h : Y \to X$ mit $h \circ g = g$ heißt *Dekodierung*.

Ist $Y = \mathbb{B}^n$, dann handelt es sich um einen *binären Code* der Länge n.

Die Codierung einer Information ergibt somit ein Codewort $g(x)$. Wegen der Injektivität von g gibt es zu jedem gegebenen Codewort y genau ein $x \in X$ mit $g(x) = y$.

4.0.2 Beispiel (Einfacher Wiederholungscode). Im einfachsten Fall wird die Nachricht einfach wiederholt. D.h. man verwendet die Generatorabbildung

$$w : X \to X \times X, \quad x \mapsto (x, x).$$

Kommt beidemal dieselbe Nachricht, so ist sie wahrscheinlich richtig übertragen worden; anderenfalls weiß man, daß ein Fehler aufgetreten ist, und kann eine passende Maßnahme ergreifen, z.B. nocheinmal senden.

Das Wort `PILL` wird damit als `PILLPILL` codiert. Falls dann, wegen der Störung, das Wort `PILLKILL` ankommt, weiss man immerhin, daß ein Fehler passiert sein muß. Die ursprüngliche Nachricht kann aber nicht rekonstruiert werden – sie könnte `PILL`, `KILL`, oder auch etwa `PILZ` gewesen sein. Kommt dagegen `PILZPILZ` an, so bleibt der Fehler natürlich unerkannt – aber das ist doch ein eher unglücklicher Zufall.

4.0.3 Beispiel (Mehrfacher Wiederholungscode). Um Fehler auch automatisch zu korrigieren, kann man die Nachricht dreimal senden, als z.B. `PILLPILLPILL`. Kommt dann `PILLKILLPILL` an, so kann man halbwegs sicher sein, daß ursprünglich `PILL` gesendet werden sollte.

Man kann auch noch öfter wiederholen, um auch mehrfache Fehler zu korrigieren und so die Wahrscheinlichkeit einer fälschlichen Übertragung beliebig klein gestalten.

Diese eben besprochenen *Wiederholungscodes* haben den Nachteil, daß das zu übertragene Codewort ziemlich lang wird, was teuer sein kann, und obendrein die Anzahl der Fehler

erhöht (wenn das zu übertragende Wort doppelt so lange ist, ist auch mit der doppelten Fehleranzahl zu rechnen), was den eigentlichen Zweck für die Wiederholung zum Teil wieder kompromittiert. Ziel der Codierungstheorie ist es, bessere Codes zu entwickeln.

Da der Kanal, über den die Nachrichten übertragen werden, störungsanfällig ist, kommt beim Empfänger statt des Codewortes y ein Wort $S(y)$ an, welche kein Codewort sein muß. Dabei hängt die Störung S nicht nur von y, sondern außerdem vom Zufall ab (es handelt sich also um eine Zuvallsvariable mit Werten in Y). Es ist nun Aufgabe der Codierungstheorie, das ursprüngliche Codewort y aus $S(y)$ möglichst zuverlässig zu rekonstruieren, d.h. mittels einer Dekodierung h zu dekodieren. Dabei sollte mit großer Wahrscheinlichkeit gelten

$$h(S(y)) = y.$$

Ist $S(y)$ kein Codewort, so weiß man jedenfalls, daß ein Fehler passiert ist. Der Fehler wurde somit *erkannt*. Wenn $S(y)$ dagegen ein Codewort ist, dann nimmt man an, daß kein Fehler passiert ist. Es könnte aber auch zufällig ein anderes Codewort entstanden sein, was natürlich bei einem guten Code eher selten vorkommen sollte.

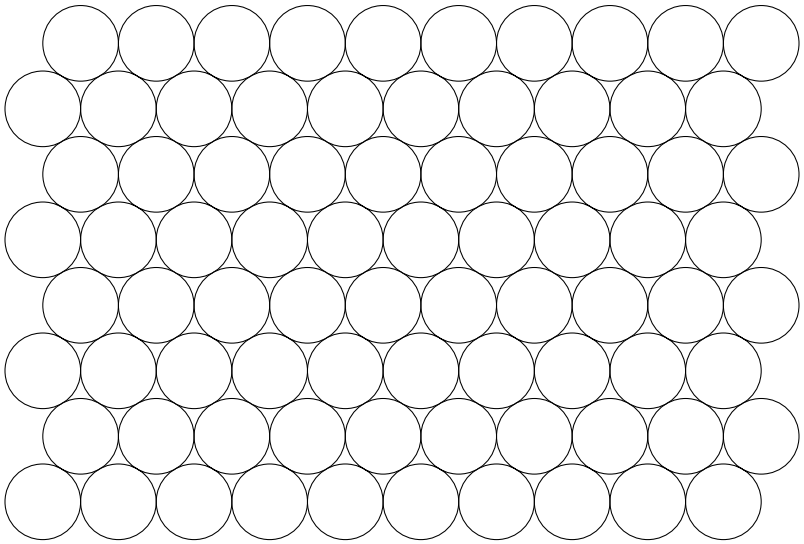
4.0.4 Definition. Sei $C \subseteq Y$ ein Code, $y \epsilon C$, und S eine Störung.

1. Ein Fehler wird *erkannt*, wenn $S(y) \notin C$.
2. Ein Fehler wird durch eine Decodierung h *korrigiert*, wenn $h(S(y)) = y$.

Ein mit der Codierung nahe verwandtes Problem ist es, „möglichst viele Orangen in einen Schiffstank reinzupferchen“:

4.0.5 Problem (Kugelpackung). Wie kann man eine große Menge gleichgroßer Kugeln möglichst dicht anordnen?

Für 2-dimensionale „Kugeln“ ist dieses Problem relativ einfach zu lösen:



Der Nachweis, daß diese Kreise tatsächlich nicht dichter angeordnet werden können, ist nicht allzu schwer. Ganz anders die Situation im 3-dimensionalen Fall: Eine sicherlich gute Lösung ist, die unterste Schicht Kugeln so wie oben anordnen und darauf die nächste Schicht so legen, daß die Kugeln in die Vertiefungen gelegt werden. Darauf kommt dann in derselben Weise die nächste Schicht, usw. Es ist aber nicht bekannt, ob es nicht doch eine bessere Lösung gibt.

Aus einer Kugelpackung gewinnt man leicht einen Code, indem man die Mittelpunkte der Kugeln als Codewörter verwendet. Man dekodiert dann jeden Punkt zum nächstgelegenen Mittelpunkt. Dies funktioniert dann vernünftig, wenn große Störungen deutlich unwahrscheinlicher sind als kleine Störungen, und es funktioniert besonders gut, wenn die Kugeln möglichst dicht gepackt sind.

Dieses Problem läßt sich auch auf höherdimensionale Räume ausdehnen. Und anstatt der normalen Euklid'schen Metrik (Abstandbegriff) kann auch eine andere verwendet werden, um den Begriff *Kugel* zu definieren.

4.0.6 Definition. Sei X eine beliebige Menge. Eine Abbildung $d\colon X \times X \to [0, \infty]$ heißt *Metrik* falls für alle $x, y, z \in X$ gilt:

$$\begin{aligned} &d(x,y) = 0 \iff x = y && \text{(Definitheit)} \\ &d(x,y) = d(y,x) && \text{(Symmetrie)} \\ &d(x,z) \leq d(x,y) + d(y,z) && \text{(Dreiecksungleichung)} \end{aligned}$$

Die Menge X, zusammen mit der Metrik, heißt dann ein *metrischer Raum.*

4.0.7 Definition. Sei (X, d) ein metrischer Raum, $x \in X$ und $\varepsilon \geq 0$. Dann heißt

$$K(x, \varepsilon)\{\, x \in X \mid d(x,y) \leq \varepsilon \,\}\}$$

die *Kugel* mit Mittelpunkt x und Radius ε.

Für $\mathbb{B}^n$ bietet sich die *Hamming-Distanz* als Metrik an:

$$\begin{aligned} d(\mathbf{x}, \mathbf{y}) &= \text{die Anzahl der Stellen, an denen sich } \mathbf{x} \text{ und } \mathbf{y} \text{ unterscheiden} \\ &= |\{\, i \in \{\, 1, \ldots, n \,\} \mid x_i \neq y_i \,\}| \qquad \text{wobei } \mathbf{x} = x_1 \ldots x_n,\ \mathbf{y} = y_1 \ldots y_n. \end{aligned}$$

Ab sofort nehmen wir generell an, daß alle zu übertragenden Nachrichten Folgen aus 0 und 1 sind. Die Nachrichten sind also Elemente von $\mathbb{B}^n$, für ein passendes n. Dabei ist zu beachten, daß, um 2^n verschiedene Wörter zu übertragen, prinzipiell eine Bitfolge der Länge n notwendig ist (d.h. die Nachricht hat n bit Information).

4.0.8 Definition. Ein Code heißt *t-fehlererkennend* falls alle fehlerhaften Codewörter mit Fehlern an höchstens t Stellen als fehlerhaft erkannt werden, d.h. für alle Codewörter y gilt

$$z \text{ ist kein Codewort, falls } d(y,z) \leq t$$

Ein Code heißt *s-fehlerkorrigierend* falls es eine Dekodierung h gibt, sodaß alle Codewörter mit höchstens s Fehlern automatisch korrigiert werden können d.h. für alle Codewörter y gilt

$$h(z) = y, \text{ falls } d(y, z) \leq s$$

Ein Code hat die *Informationsrate* $\frac{k}{n}$, falls er zur Übertragung von k bit Information ein Codewort der Länge n verwendet.

4.0.9 Beispiel. Der einfache Wiederholungscode ergibt einen 1-fehlererkennenden Code. Der zweifachen Wiederholungscode ergibt einen 1-fehlerkorrigierenden Code. Die Informationsrate dieser Codes ist $\frac{1}{2}$ bzw. $\frac{1}{3}$.

4.0.10 Beispiel (Paritätskontrolle). Eine Folge $x_1 \dots x_k$, mit $x_i \in \{0, 1\}$ wird codiert, indem eine Prüfziffer x_{k+1} angehängt wird, sodaß die Anzal der Einser gerade wird, also

$$x_{k+1} = x_1 + \cdots + x_k \pmod 2$$

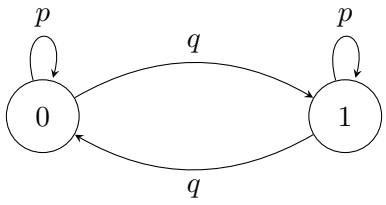
Dieser Code ist 1-fehlerkorrigierend, so wie der einfache Wiederholugscode, aber mit der deutlich besseren Informationsrate $\frac{k}{k+1}$.

4.0.11 Beispiel. Wir führen die Idee der Paritätskontrolle fort, indem wir gleich 3 Prüfsymbole einführen. Für $k = 3$ sieht das dann so aus:

$$\begin{aligned} x_4 &= x_1 + x_2 \\ x_5 &= x_1 \qquad + x_3 \\ x_6 &= \qquad x_2 + x_3 \end{aligned}$$

Dieser Code kann doppelte Fehler erkennen und einfache Fehler korrigieren, so wie der doppelte Wiederholungscode, hat aber die deutlich bessere Informationsrate $\frac{1}{2}$.

4.0.12 Definition. Ein *binärer symmetrischer Kanal* ist ein Übertragungskanal für Bitfolgen, in dem jedes einzelne Bit, unabhängig von den anderen, mit einer Wahrscheinlichkeit p korrekt und mit einer Wahrscheinlichkeit $q := 1 - p$ falsch übertragen wird.



Die die Anzahl der Fehler in einer Bitfolge der Länge n ist dann binomialverteilt. Insbesondere beträgt die mittlere Fehleranzahl nq, die Wahrscheinlichkeit für eine fehlerfreie Übertragung ist p^n, für einen Fehler $np^{n-1}q$, für t Fehler $\binom{n}{t}p^{n-t}q^t$. Für $p = 99\,\%$

und $n = 50$ bedeutet dies konkret:

t	Wahrscheinlichkeit für t Fehler
0	$60,50\,\%$
1	$30,56\,\%$
2	$7,56\,\%$
3	$1,22\,\%$
4	$0,15\,\%$
5	$0,01\,\%$

Tatsächlich geht die Wahrscheinlichkeit für sehr viele Fehler sehr schnell gegen 0 geht, falls $p > \frac{1}{2}$ ist.

4.0.13 Definition. Ein *binärer Code* C der *Länge* n ist eine Teilmenge von $\mathbb{B}^n$. Wird ein Codewort $\mathbf{x} \,\epsilon\, C$ übermittelt und $\mathbf{y} \,\epsilon\, X$ erhalten, so heißt $\mathbf{e} := \mathbf{y} - \mathbf{x}$ der *Fehler*.

Aufgabe des Dekodierverfahrens ist es nun, einem gegebenen $\mathbf{y}$ ein passendes Codewort zuzuordnen. Gemäß unserer Diskussion nach Definition 4.0.12 ist es sinnvoll, dasjenige $\mathbf{x}$ zu wählen, welches zu $\mathbf{y}$ den kleinsten Abstand hat (*maximum likelihood decoding*).

Ist für $\mathbb{B}^n$ eine Kugelpackung gegeben, so kann man die Mittelpunkte der Kugeln als gültige Codewörter auffassen. Beim maximum likelihood decoding werden dann alle Wörter innerhalb einer Kugel dem entsprechenden Mittelpunkt zugeordnet. Haben die Kugeln der Kugelpackung einen Radius t, dann ist dieser Code t-fehlerkorrigierend und $(2t-1)$-fehlererkennend.

Allgemeiner nennt man den minimalen Abstand zweier verschiedener Codewörter die *Hamming-Distanz* des Codes, also

$$d_{\min} := \min_{\substack{\mathbf{x},\mathbf{y}\epsilon C \\ \mathbf{x}\neq\mathbf{y}}} d(\mathbf{x},\mathbf{y}).$$

4.0.14 Satz. *Ein Code mit Hamming-Distanz d kann bis zu $d-1$ Fehler erkennen und $\lfloor \frac{d-1}{2} \rfloor$ Fehler korrigieren.*

4.0.15 Definition. Ein k-dimensionaler linearer Unterraum des $\mathbb{B}^n$ mit Minimaldistanz d heißt ein heißt ein *linearer Code* oder ein (n,k,d)-*Code*.

Wir haben bisher fast nur lineare Codes kennengelernt.

4.0.16 Definition. Sei $\mathbf{H}$ eine $(n-k) \times n$ Matrix von vollem Zeilenrang. Dann ist der k-dimensionale Unterraum $C := \{\mathbf{Hx} = \mathbf{o}\}$ ein linearer Code und C heißt eine *Paritätskontrollmatrix* (*parity-check matrix*) von C.

Eine $n \times k$ Matrix $\mathbf{G}$ mit vollem Spaltenrang, sodaß $\mathbf{H} \cdot \mathbf{G} = \mathbf{O}$, heißt eine zugehörige *Generatormatrix*.

Die Spalten von $\mathbf{G}$ bilden dabei offensichtlich eine Basis von C, die Zeilen von $\mathbf{H}$ dagegen eine Basis des orthogonalen Komplements von C.

Durch die Generatormatrix wird eine Codierung festgelegt:

$$\mathbf{x} \mapsto \mathbf{G} \cdot \mathbf{x},$$

und mit mit dem Test

$$\mathbf{H} \cdot \mathbf{y} = \mathbf{o}$$

kann man leicht überprüfen, ob $\mathbf{y}$ ein gültiges Codewort ist.

Von den Dekodierungsverfahren sei hier nur dasjenige für eine sehr spezielle Klasse linearer Codes erwähnt.

4.0.17 Definition. Ein (binärer) linearer Code C_m der Länge $n = 2^m - 1$, $m \geq 2$, dessen Paritätskontrollmatrix die Dimension $m \times n$ hat und deren Spalten aus allen nicht-verschwindenden (binären) Vektoren besteht, heißt ein binärer *Hamming Code*.

4.0.18 Theorem. *Jeder Hamming Code C_m ist 1-fehlererkennend und hat Informationsrate $\frac{n-m}{n}$. Die Dekodierung gestaltet sich denkbar einfach: Wird das Wort $\mathbf{y}$ erhalten, so bestimmt man das sogenannte Syndrom $\mathbf{H}\cdot\mathbf{y}$. Falls dieses nicht der Nullvektor ist, ist ein Fehler passiert, und zwar am wahrscheinlichsten an jener Stelle, deren Binärdarstellung das Syndrom ist.*

4.0.19 Beispiel. Der Hamming Code C_3 hat die Paritätskontrollmatrix

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

(Die Spalten entsprechen gerade den Binärdarstellungen der Zahlen von 1 bis 7.) Angenommen es wird das Wort $\mathbf{y} = 0011101$ erhalten. Wir berechnen das Syndrom $\mathbf{H}\cdot y = 101$, was der Zahl 5 entspricht. An der 5. Stelle trat also ein Fehler auf. Das ursprünglich kodierte Wort war daher 0011001. Die eigentliche Nachricht, die damit kodiert wurde, ist daher 1001 (oder die Zahl 9). Letztere entspricht den Stellen x_3, x_5, x_6, x_7. Die anderen Stellen sind Prüfziffern: $x_1 = x_3 + x_5 + x_7$, $x_2 = x_3 + x_6 + x_7$ und $x_4 = x_5 + x_6 + x_7$. Die Generatormatrix dieses Codes ist somit

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Es gibt natürlich noch viel mehr brauchbare Codes als die besprochenen (lineare und nicht-lineare), und auch zahlreiche Methoden, um aus bekannten Codes neue zu gewinnen. Die Brauchbarkeit eines Codes ergibt sich einerseits aus dessen Qualität, d.h.

einer möglichst hohen Informationsrate für den gewünschten Grad der Fehlererkennung und -korrektur, und andererseits aus der Verfügbarkeit von effizienten Codierungs- und Decodierungsverfahren. Allgemein läßt sich sagen: Je mehr Bit zu einem Codewort zusammengefaßt werden, desto bessere Codes können gefunden werden; allerdings muß man dann typischerweise weniger effiziente (De-)Codierungsverfahren in Kauf nehmen.