"Source coding is what Alice uses to save money on her telephone bills. It is usually used for data compression, in other words, to make messages shorter."

John Gordon

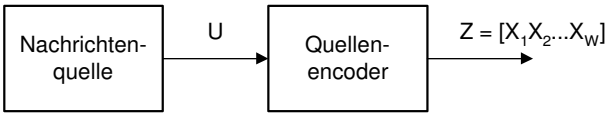
3 Quellencodierung

3.1 Einleitung

Im vorangegangenen Kapitel wurde der Begriff der Ungewissheit eines Zufallsexperiments eingeführt und gezeigt, dass dieser einige gefühlsmässig einleuchtende Eigenschaften besitzt. Daraus können wir jedoch noch nicht schliessen, dass die angegebene Definition die „richtige"[3] ist. Wir müssen uns erst davon überzeugen, dass praktische Probleme der Informationsübertragung und -speicherung mit Hilfe der Ungewissheit effizient untersucht werden können. Genau dies bezweckt dieses Kapitel.

Bei der Quellencodierung geht es darum, die zu übertragende Information möglichst effizient in eine Folge von Ziffern (nicht notwendigerweise binär) zu codieren. Als Mass für die Effizienz wird gewöhnlich die im Mittel zu erwartende Länge der Ziffernfolge verwendet.

Um das Problem genauer untersuchen zu können, betrachten wir die in Figur 3 wiedergegebene Versuchsanordnung.



Figur 3: Codierungsschema

Die Nachrichtenquelle liefert ein zufällig gewähltes Symbol U aus einem N-wertigen Alphabet $\{u_1, u_2, \ldots u_N\}$. Der Quellenencoder wandelt das Symbol U in eine Folge von Ziffern $Z = [X_1X_2\ldots X_W]$ um, wobei jede Ziffer X_i aus einem D-wertigen Alphabet $\{0, 1, \ldots, D-1\}$ stammt.

Häufig ist U ein Buchstabe aus dem lateinischen Alphabet {‚a', ‚b', … ‚z'}. Dies ist jedoch keineswegs zwingend. Man könnte beispielsweise als Alphabet auch die Menge aller deutschen Wörter verwenden. Dann würde der Quellenencoder eben ganze Wörter und nicht einzelne Buchstaben

[3] Definitionen (definitio lat., Bestimmung) sind durch Menschen festgelegte, möglichst klare Beschreibungen eines Begriffs. Es macht deshalb keinen Sinn, sie als richtig oder falsch zu bezeichnen. Hingegen können Definitionen durchaus mehr oder weniger zweckdienlich sein.

codieren. Das Codewort Z am Ausgang des Encoders ist meistens, aber nicht immer, eine binäre Folge. Wir werden uns im Folgenden dennoch auf D = 2 beschränken.

Die Länge W des Codewortes Z ist gemeinhin nicht konstant, sondern vom Symbol U abhängig. Ein effizienter Quellencodierer wandelt häufig auftretende Symbole in kurze Ziffernfolgen um und verwendet für seltenere Symbole längere Ziffernfolgen. Ein Mass für die Effizienz des Quellencodierers ist die durchschnittliche Codewortlänge, welche wir mit E[W] bezeichnen. Falls das Symbol u_i in ein Codewort der Länge w_i umgewandelt wird, kann die durchschnittliche Codewortlänge wie folgt berechnet werden

$$E[W] = \sum_{i=1}^{N} w_i \cdot P(u_i),$$

wobei $P(u_i)$ die Auftretenswahrscheinlichkeit des Symbols u_i bezeichnet.

Wir stellen zwei grundlegende Anforderungen an den Quellenencoder:

1. Zwei unterschiedliche Symbole u_i und u_j müssen auch in unterschiedliche Codewörter z_i und z_j umgewandelt werden. Ferner darf der Code keine leeren Codewörter (Codewörter der Länge null) enthalten. Falls diese Bedingungen nicht erfüllt sind, spricht man von einem degenerierten Code.
2. Kein Codewort soll das Präfix (Vorsilbe) eines längeren Codewortes sein. Damit stellen wir sicher, dass ein Codewort decodiert werden kann, sobald das letzte Zeichen empfangen wurde. Es ist dann auch möglich, den Quellenencoder mehrmals nacheinander für das Codieren von Symbolen zu verwenden, ohne dass es beim Empfang zu Mehrdeutigkeiten kommt.

Jeder präfixfreie Code ist eindeutig decodierbar, aber nicht alle eindeutig decodierbaren Codes sind präfixfrei! Es wurde jedoch bewiesen, dass für jeden eindeutig decodierbaren Code ein präfixfreier Code mit gleichen Codewortlängen existiert. Man kann sich deshalb bei der Suche nach optimalen Codes auf präfixfreie Codes beschränken.

Beispiel

Codes können in Tabellenform wiedergegeben werden, indem für jedes Nachrichtensymbol u_i das zugehörige Codewort z_i dargestellt wird.

Der Code

U	Z
u_1	0
u_2	10
u_3	11

erfüllt die obigen Bedingungen. Würde man die Reihenfolge der Ziffern der Codewörter umkehren, so wäre der Code immer noch eindeutig decodierbar, aber nicht mehr präfixfrei.

Beim Code

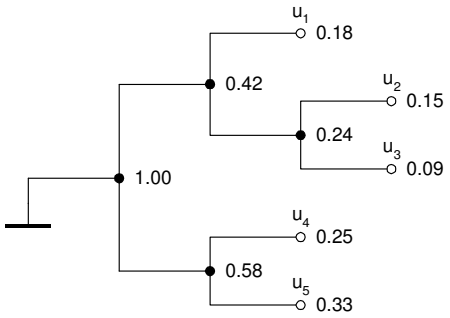
U	Z
u_1	1
u_2	00
u_3	11

ist das Codewort z_1 ein Präfix des Codewortes z_3. Bei Empfang der Folge [1 1 1] ist nicht klar, ob $[u_1\ u_1\ u_1]$, $[u_1\ u_3]$ oder $[u_3\ u_1]$ codiert wurde. ■

3.2 Codebäume mit Wahrscheinlichkeiten

Zur graphischen Darstellung von Codes eignen sich Verzweigungsbäume. Die einzelnen Zeichen eines Codewortes entsprechen dabei den Ästen des Baums. Bei einem Code ohne Präfix ist jedes Codewort ein Endknoten des Baums. Es macht deshalb Sinn, den Endknoten die Wahrscheinlichkeit des entsprechenden Codewortes $P(u_i)$ zuzuordnen. Selbstverständlich muss die Summe dieser Wahrscheinlichkeiten 1 betragen. Den Nichtendknoten werden ebenfalls Wahrscheinlichkeiten zugeordnet und zwar ergibt sich die Wahrscheinlichkeit eines Knotens aus der Summe der Wahrscheinlichkeiten der direkt darüber liegenden Knoten[4]. Die Wahrscheinlichkeit des Wurzelknotens ist deshalb immer gleich 1.

U	$P(u_i)$	Z
u_1	0.18	11
u_2	0.15	101
u_3	0.09	100
u_4	0.25	01
u_5	0.33	00



Figur 4: Darstellung eines Codes als Verzweigungsbaum

Zwischen den Wahrscheinlichkeiten der Nichtendknoten und der mittleren Länge eines Astes besteht ein einfacher Zusammenhang, der wie folgt lautet:

[4] Wir bezeichnen einen Knoten als darüber liegend, falls er näher beim Endknoten, resp. weiter entfernt vom Wurzelknoten liegt.

Pfadlängensatz

In einem Codebaum mit Wahrscheinlichkeiten ist die mittlere Länge der Äste gleich der Summe der Wahrscheinlichkeiten der Nichtendknoten (inklusive der Wurzel).

Beweis

Betrachten wir einen Endknoten, so erkennen wir, dass dessen Wahrscheinlichkeit $P(u_i)$ in allen darunter liegenden Knoten genau einmal als Summand auftritt. Die Anzahl darunter liegender Knoten ist gleich der Länge w_i des zum Endknoten gehörigen Codeworts. Dieser Endknoten liefert aus diesem Grunde den Beitrag $w_i \cdot P(u_i)$ an die Gesamtsumme. Addiert man die Beiträge aller Endknoten, so erhält man die Formel zur Bestimmung der mittleren Länge eines Codes. ■

Beispiel

Addieren wir die Wahrscheinlichkeiten der Nichtendknoten des Codes aus Figur 4, so erhalten wir:

$$1.0 + 0.42 + 0.58 + 0.24 = 2.24.$$

Berechnen wir die mittlere Codewortlänge, so ergibt sich:

$$E[W] = 0.18 \cdot 2 + 0.15 \cdot 3 + 0.09 \cdot 3 + 0.25 \cdot 2 + 033 \cdot 2 = 2.24.$$ ■

3.3 Kraft'sche Ungleichung

Die Antwort auf die Frage, ob für eine gegebene Liste von Codewortlängen ein binärer präfixfreier Code existiert, liefert die Kraft'sche Ungleichung.

Kraft'sche Ungleichung

Ein binärer präfixfreier Code mit den Codewortlängen $w_1, w_2, \ldots w_N$ existiert genau dann, falls die Bedingung

$$\sum_{i=1}^{N} 2^{-w_i} \leq 1$$

erfüllt ist.

Beweis

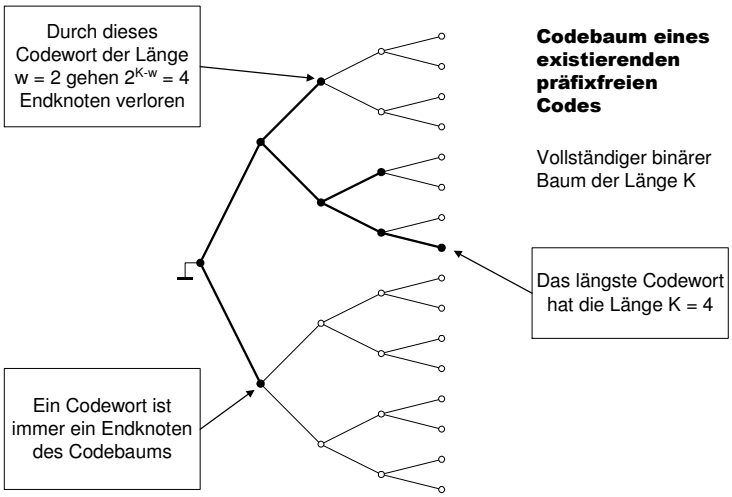
Nehmen wir zuerst an, ein solcher Code existiere. Wir wollen zeigen, dass in diesem Fall die Ungleichung erfüllt ist. Zu diesem Zweck betrachten den vollständigen binären Baum der Länge $K = \max(w_i)$. Das erste Codewort z_1 entspricht einem Knoten in diesem Baum. Falls $w_1 < K$ ist, stutzen wir den Baum an diesem Knoten und machen ihn so zu einem Endknoten. Wir verlieren dadurch 2^{K-w_1} Endknoten des vollständigen Baums. Indem wir das mit allen Codewörtern durchführen, verlieren wir insgesamt

$$2^{K-w_1} + 2^{K-w_2} + \cdots + 2^{K-w_K}$$

Endknoten. Gesamthaft besitzt der vollständige Baum jedoch genau 2^K Endknoten. Da nach unserer Voraussetzung der Code existiert, muss demnach die Bedingung

$$2^{K-w_1} + 2^{K-w_2} + \cdots + 2^{K-w_K} \leq 2^K$$

gelten. Durch Division mit 2^K erhält man die Kraft'sche Ungleichung.

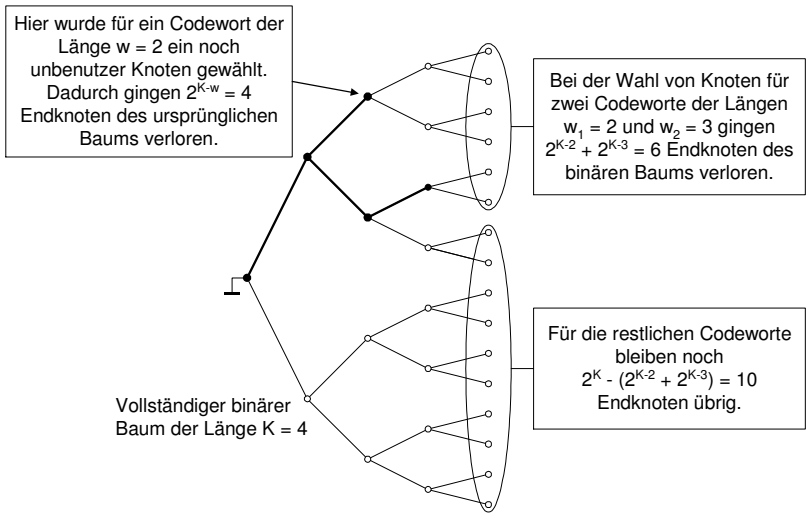


Figur 5: Ein existierender präfixfreier Code erfüllt die Kraft'sche Ungleichung

Nun nehmen wir umgekehrt an, eine Menge positiver ganzer Zahlen, welche die Kraft'sche Ungleichung erfüllen, sei gegeben. Ohne Beschränkung der Allgemeinheit dürfen wir annehmen, dass die Codewortlängen sortiert wurden, d.h. es gilt $w_1 \leq w_2 \leq \ldots \leq w_N = K$.

Ausgehend von einem vollständigen binären Baum der Länge $K = \max(w_i)$ versuchen wir einen binären präfixfreien Code mit Hilfe des folgenden Algorithmus zu konstruieren.

1. $i := 1$
2. Wähle für z_i einen noch unbenutzten Knoten der Tiefe w_i und, falls $w_i < K$, stutze den Baum an diesem Knoten.
3. Falls $i \neq N$ setze $i := i + 1$ und gehe zu Schritt 2.



Figur 6: Vorgehen zur Konstruktion eines Codes mit vorgegebenen Codewortlängen.

Falls wir jedes Mal im Schritt 2 einen Knoten wählen können, haben wir einen binären Code ohne Präfix mit den vorgegebenen Codewortlängen konstruiert. Wir müssen uns nun davon überzeugen, dass dieser Algorithmus zum Ziel führt, falls die Kraft'sche Ungleichung erfüllt ist. Dazu nehmen wir an, die Codeworte $z_1, z_2, \ldots z_{j-1}$ seien zugewiesen worden und stellen uns die Frage, ob es in jedem Fall möglich ist, für das j-te Codewort einen noch unbenutzten Knoten der Tiefe w_j zu finden. Durch die Wahl eines Codeworts der Länge w_i fallen im Schritt 2 jeweils 2^{K-w_i} Endknoten des vollständigen binären Baums weg. Nach der Festlegung von j - 1 Codeworten existieren also noch

$$2^K - \left(2^{K-w_1} + 2^{K-w_2} + \cdots + 2^{K-w_{j-1}}\right) = 2^K \cdot \left(1 - \sum_{i=1}^{j-1} 2^{-w_i}\right)$$

Endknoten der Tiefe K. Da aber $j \le N$ gilt, folgt mit Hilfe der Ungleichung von Kraft, dass diese Zahl grösser null ist

$$\sum_{i=1}^{j-1} 2^{-w_i} < 1 \qquad \Rightarrow \qquad \underbrace{2^K \cdot \left(1 - \underbrace{\sum_{i=1}^{j-1} 2^{-w_i}}_{<1}\right)}_{>0} \ge 1 .$$

Wenn aber mindestens ein Endknoten der Tiefe K existiert, kann sicherlich auch ein Knoten der Tiefe w_j gefunden und somit ein Codewort z_j zugewiesen werden. ■

3.4 Huffman Code

Wir zeigen nun, wie ein optimaler präfixfreier Code konstruiert werden kann. Unter optimal verstehen wir in diesem Zusammenhang einen Code mit möglichst kurzer mittlerer Codewortlänge.

Wir betrachten dazu eine Nachrichtenquelle mit N unterscheidbaren Symbolen $U = \{u_1, u_2, ... u_N\}$ und den dazugehörigen Wahrscheinlichkeiten $P_U(u_i) = p_i$. Ohne Beschränkung der Allgemeinheit dürfen wir annehmen, dass die Liste $[p_1, p_2, ... p_N]$ der Wahrscheinlichkeiten sortiert ist, d.h. es gelte $p_1 \geq p_2 \geq ... \geq p_N$.

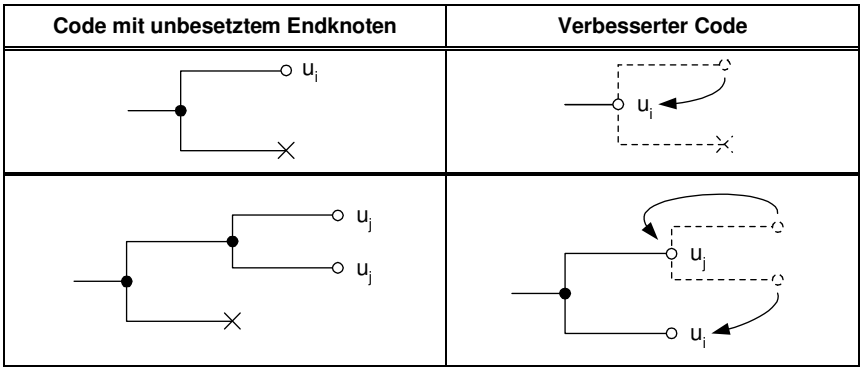
Zunächst beweisen wir die nachstehenden Hilfssätze.

Hilfssätze

1. Ein optimaler binärer präfixfreier Code besitzt keine unbesetzten Endknoten.
2. Zu jeder beliebigen Liste von Wahrscheinlichkeiten existiert ein optimaler binärer präfixfreier Code, bei dem die Codewörter für die beiden unwahrscheinlichsten Symbole sich nur im letzten Bit unterscheiden.

Beweis der Bedingung 1

Wir gehen von der Annahme aus, dass ein optimaler Code mit unbesetztem Endknoten existiert. Der Nachbarknoten des unbesetzten Endknotens kann entweder ein Codewort oder ein Teilbaum sein. Wie Figur 7 zeigt, gelingt es in beiden Fällen einen besseren Code zu finden, bei dem alle Endknoten besetzt sind.



Figur 7: Ein optimaler binärer präfixfreier Code besitzt keine unbesetzten Endknoten.

■

Beweis der Bedingung 2

Nicht alle optimalen Codes müssen diese Bedingung erfüllen, aber es gibt immer einen optimalen Code, der sie erfüllt. Gemäss dem soeben bewiesenen Hilfssatz, besitzt das längste Codewort einen gleich langen Nachbarn, der sich lediglich im letzten Bit unterscheidet. Ohne die Optimalität zu gefährden, können diesen beiden Codeworten die Symbole u_{N-1}, u_N mit den kleinsten Wahrscheinlichkeit zugeordnet werden. ■

Hilfssatz

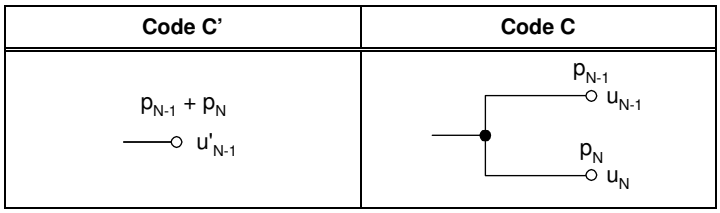
Es sei C' der optimale binäre präfixfreie Code für die Symbole $U' = \{u'_1, u'_2, ..., u'_{N-1}\}$ mit den Wahrscheinlichkeiten $[p_1, p_2, ..., p_{N-1} + p_N]$.

Den optimalen Code C für die Symbole $U = \{u_1, u_2, ..., u_N\}$ mit den Wahrscheinlichkeiten $[p_1, p_2, ..., p_{N-1}, p_N]$ erhält man, indem der Knoten für das Symbol u'_{N-1} erweitert wird und die beiden neu entstandenen Endknoten den Symbolen u_{N-1} und u_N zugeordnet werden.

Anstatt eines Endknotens der Tiefe w_{N-1} und der Wahrscheinlichkeit $p_{N-1} + p_N$ enthält der neue Code C zwei Endknoten der Tiefe $w_{N-1} + 1$ und den Wahrscheinlichkeiten p_{N-1} und p_N. Die durchschnittliche Codewortlänge des derart konstruierten Codes C ist deshalb um

$$p_{N-1} \cdot (w_{N-1} + 1) + p_N \cdot (w_{N-1} + 1) - (p_{N-1} + p_N) \cdot w_{N-1} = p_{N-1} + p_N$$

länger als die durchschnittliche Codewortlänge des Codes C'.

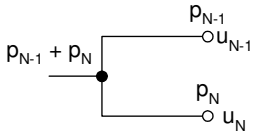


Figur 8: Zusammenhang zwischen dem Code C' und dem Code C

Beweis des Hilfssatzes

Es ist sofort klar, dass der Code C binär und präfixfrei ist. Wir müssen noch beweisen, dass C optimal für $[p_1, p_2, ... p_N]$ ist, falls C' optimal für $[p_1, p_2, ..., p_{N-1} + p_N]$ ist. Dazu nehmen wir an, dass C* ein besserer Code für die Liste $[p_1, p_2, ... p_N]$ als C wäre. Ohne die durchschnittliche Codewortlänge zu ändern, könnten wir die Codeworte für die beiden unwahrscheinlichsten Symbole u_{N-1} und u_N von C* so vertauschen, dass sie sich lediglich im letzten Bit unterscheiden würden. Diese beiden Endknoten könnten anschliessend zusammengefasst werden und als Codewort für das Symbol u'_{N-1} dienen. Es entstünde so ein Code C'* für die Liste $[p_1, p_2, ..., p_{N-1} + p_N]$, dessen durchschnittliche Codewortlänge um $p_{N-1} + p_N$ kürzer wäre als diejenige des Codes C*. Da C* gemäss Annahme besser als C ist, der Unterschied aber in beiden Fällen $p_{N-1} + p_N$ beträgt, wäre die durchschnittliche Codewortlänge von C'* kürzer als diejenige von C', woraus folgt, dass C' kein optimaler Code wäre. ■

Der soeben bewiesene Hilfssatz zeigt uns, wie der erste Nichtendknoten des optimalen Codes konstruiert werden muss. Dazu müssen offensichtlich die beiden Endknoten mit kleinster Wahrscheinlichkeit zusammengeführt werden.



Figur 9: Ein neuer Knoten entsteht durch Verbindung der am wenigsten wahrscheinlichen Endknoten.

Wir tun nun so, als ob der derart konstruierte Nichtendknoten ein neuer Endknoten sei und weisen ihm die Wahrscheinlichkeit $p_N + p_{N-1}$ zu. Mit diesem neuen und den verbleibenden N - 2 Endknoten fahren wir anschliessend fort und konstruieren den nächsten Nichtendknoten durch Verbinden der beiden Endknoten mit kleinster Wahrscheinlichkeit.

Das Vorgehen ist nachfolgend nochmals zusammengefasst.

Huffmans binärer Algorithmus [3]

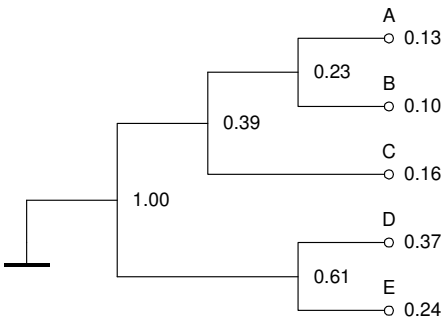
1. Bezeichne die N Endknoten mit $u_1, u_2, \ldots u_N$ und weise ihnen die Wahrscheinlichkeiten $P(u_i)$ zu. Betrachte diese Knoten als aktiv.
2. Verbinde die zwei am wenigsten wahrscheinlichen Knoten. Deaktiviere diese beiden Knoten, aktiviere den neu entstandenen Knoten und weise ihm die Summe der beiden Wahrscheinlichkeiten zu.
3. Falls noch ein aktiver Knoten existiert, gehe zu 2.

Beispiel

Wir betrachten eine Nachrichtenquelle, welche die Symbole A, B, C, D und E mit den folgenden Wahrscheinlichkeiten generiert.

u =	A	B	C	D	E
P(u) =	0.13	0.10	0.16	0.37	0.24

Durch Anwendung des Huffman Algorithmus erhält man einen optimalen binären präfixfreien Code



Die im Mittel zu erwartende Codewortlänge errechnet sich zu E[W] = 2.23. ■

Es sei hier noch erwähnt, dass der Algorithmus bei nichtbinärer Codierung ($D \neq 2$) angepasst werden muss.