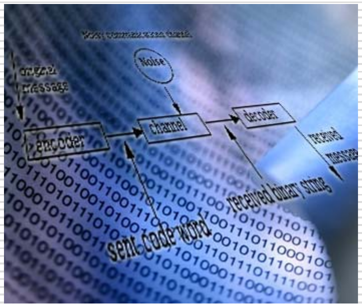
Kapitel 8: Optimalcodierung II



Ziele des Kapitels

ETH

- Shannon-Fano Coding
- Arithmetic Coding

Shannon-Fano Coding

ETH

- Ein weiteres wichtiges, klassisches Verfahren ist das Shannon-Fano Verfahren:
  Gegeben sind die Codewort-Wahrscheinlichkeiten $[p_1 \dots p_L]$:
  1. Ordne die Wahrscheinlichkeiten nach fallenden Werten
  2. Teile das Feld in zwei Gruppen, so dass jede Gruppe möglichst gleich grosse Teilsummen besitzt (Entropiemaximierung)
  3. Weise der ersten Gruppe eine 0 zu, der zweiten Gruppe eine 1
  4. Wiederhole 2. und 3. bis jede Gruppe nur noch ein Element enthält

Codewortlängen ergeben sich zwangsläufig, wobei ensprechend nach dem Optimierungsprinzip $l_j=1/p_j$ auf- bzw. abgerundet wird

Shannon-Fano Coding

ETH

- Gegeben sei eine diskrete Quelle mit Wahrscheinlichkeitsverteilung

$$P_X(x|=[0.18; 0.10; 0.40; 0.08; 0.05; 0.05;0.14])$$

Sym	p_j	1	2	3	4	Codewörter	Länge l_j	$p_j \cdot l_j$
a	0.4		0			00	2	0.80
b	0.18	0	1			01	2	0.36
c	0.14	1		0		100	3	0.42
d	0.1		0	1		101	3	0.30
e	0.08		1	0		110	3	0.24
f	0.05			1	0	1110	4	0.20
g	0.05				1	1111	4	0.20

Shannon-Fano Coding

- $E[l_C(x)] = \sum p_j \cdot l_j = 2.52$
- $$H(X) = -0.4 \cdot \log 0.4 - 0.18 \cdot \log 0.18 - 0.14 \cdot \log 0.14 - 0.1 \cdot \log 0.1$$
  $$- 0.08 \cdot \log 0.08 - 2 \cdot 0.05 \cdot \log 0.05$$
  $$= 2.43 \,\text{Bit/QZ}$$
- Coderedundanz ist damit
  $$R_C = E[l_C(x)] - H(X) = 2.52 \,\text{Bits/QZ} - 2.43 \,\text{Bit/QZ}$$
  $$= 0.09 \,\text{Bits/QZ}$$

Bemerkungen

- Shannon-Fano Coding ist suboptimal
- Es kann gezeigt werden, dass die mittlere Codelänge wie folgt begrenzt ist:
  $$H(X) \le E[l_C(X)] \le H(X) + 1$$
- Wir erhalten also die gleichen Grenzen, wie bei Huffman-Coding
- Dieses Verfahren ist nicht immer eindeutig
- Es kann durchaus mehrere Codes geben, die gleiche Coderedundanz aufweisen
- Praktisch weniger relevant

Shannon-Fano vs. Huffman

- Gegeben: Quellalphabet mit 5 Symbolen {A,B,C,D,E}

Symbol	$P(x)$	S-F Code	Huffman
A	0.35	00	1
B	0.17	01	011
C	0.17	10	010
D	0.16	110	001
E	0.15	111	000
		$E[l]=2.31$	$E[l]=2.3$

Erweiterte Quellen

- Wir erinnern uns an das Konzept der erweiterten Quellen
- Es gilt für die mittlere Codelänge
  $$mH(X) \le mE[l_C(X)] < mH(X) + 1$$
- Sowie für die Coderedundanz
  $$R_C < \frac{1}{m}$$

Shannon-Fano Coding (m=1) ETH

- Gegeben eine Binärqellemit unabhängigen Zeichen und den Auftrittswahrscheinlichkeiten $p_1 = 0.2$ und $p_2 = 0.8$
- $E[l_C(x)] = 1$
- $H(X) = -0.2 \cdot \log 0.2 - 0.8 \cdot \log 0.8$
  $= 0.722\,\text{Bit/QZ}$
- $R_C = E[l_C(x)] - H(X) = 0.278\,\text{Bits/QZ}$

Shannon-Fano Coding (m=2) ETH

- Wir erhalten eine erweiterte Quelle
  $X^2 = \{(x_1x_1),(x_1x_2),(x_2x_1),(x_2x_2)\}$
- Mit den zugehörigen Auftrittswahrscheinlichkeiten
  $p_1^2 = 0.2 \cdot 0.2 = 0.04 \qquad p_2^2 = 0.2 \cdot 0.8 = 0.16$
  $p_3^2 = 0.8 \cdot 0.2 = 0.16 \qquad p_4^2 = 0.8 \cdot 0.8 = 0.64$
- Anwendung des Shannon-Fano Algorithmus'

p_j^2	Optimalcode	$p_j \cdot l_j$
0.64	0	0.64
0.16	10	0.32
0.16	110	0.48
0.04	111	0.12

Shannon-Fano Coding (m=2) ETH

- $m \cdot E[l_C(x)] = \sum p_j^2 \cdot l_j = 1.56$
  $\Rightarrow E[l_C(x)] = \frac{1.56}{2} = 0.780$
- $H(X) = -0.2 \cdot \log 0.2 - 0.8 \cdot \log 0.8$
  $= 0.722\,\text{Bit/QZ}$
- $R_C = E[l_C(x)] - H(X) = 0.058\,\text{Bits/QZ}$

Shannon-Fano Coding (m=3) ETH

	p_j^3	p_j^3 geordnet	Optimalcode	$p_j^3 \cdot l_j$
$p_1p_1p_1$	0.512	0.512	0	0.512
$p_1p_1p_2$	0.128	0.128	100	0.384
$p_1p_2p_1$	0.128	0.128	101	0.384
$p_1p_2p_2$	0.032	0.128	110	0.384
$p_2p_1p_1$	0.128	0.032	11100	0.160
$p_2p_1p_2$	0.032	0.032	11101	0.160
$p_2p_2p_1$	0.032	0.032	11110	0.160
$p_2p_2p_2$	0.008	0.008	11111	0.040

Shannon-Fano Coding (m=3)

- $m \cdot E[l_C(x)] = \sum p_j^3 \cdot l_j = 2.184$

$$\Rightarrow E[l_C(x)] = \frac{2.184}{3} = 0.728$$

- $H(X) = -0.2 \cdot \log 0.2 - 0.8 \cdot \log 0.8$

$$= 0.722\,\text{Bit/QZ}$$

- $R_C = E[l_C(x)] - H(X) = 0.006\,\text{Bits/QZ}$

Arithmetic Coding

- Die generelle Zuordnung von $l_j = 1/p_j$ führt zu einer optimalen Codierung an der Entropiegrenze
- Huffman Codes sind optimal und in $O(n)$ erzeugbar
- Typischerweise muss man die Codelängen aufrunden
- Die Coderedundanz liegt daher innerhalb von einem Bit der Quellenentropie
- Blockcode helfen weiter, das Rundungsproblem bleibt jedoch
- Idee: Codiere einen String als reelle Zahl $0 <= R < 1$

Arithmetic Coding

- Arithmetische Codierung:
  Gegeben sind die Codewort-Wahrscheinlichkeiten $[p_1 \ldots p_L]$ aller Zeichen der Quelle sowie ein String $s = x_1 || \ldots || x_S$ der Länge S.
  1. Teile das Intervall [0,1) in Subintervalle gemäss der Symbol-Wahrscheinlichkeiten p_i
  2. Nimm das nächste Symbol x_i aus dem String und ermittle das Subintervall, welches seiner Wahrscheinlichkeit $P(x_i)$ entspricht
  3. Teile dieses Intervall gemäss der Symbolwahrscheinlichkeiten, so wie in Schritt 1
  4. Führe die Schritte 2 und 3 aus, bis der String zu Ende ist
  5. Weise den Binärcode einer Zahl aus diesem Intervall dem String zu

Arithmetic Coding

- Insbesondere können die neuen Grenzen mit folgenden beiden Gleichungen berechnet werden
  - `newleft = prevleft + left[x[i]]*prevsize`
  - `newsize = prevsize*size[x[i]]`
- Hierbei sind `left[x[i]]` die linke Intervallgrenze des aktuellen Symbols x_i sowie `size[x[i]]` seine Wahrscheinlichkeit
- Die Grösse des resultierenden Intervalls ergibt sich aus dem Produkt der Symbolwahrscheinlichkeiten
- des Strings $P(s) = \prod_{i=1}^{S} P_X(s(i))$

Arithmetic vs. Huffman

- Code 1: Gegeben χ = {a,e,i,o,u} sowie *p*(a)=0.12, *p*(e)=0.42, *p*(i)=0.09, *p*(o)=0.3 und *p*(u)=0.07
- Huffman:
  Beispielstring: S=„iou“
  => 1011111010 = 10 Bit
- Arithmetic-Coding:
- Beispielstring: S= „iou“
  => .011000001 = 9 Bit

Arithmetic Coding

- Die Grösse des resultierenden Intervalls ergibt sich aus dem Produkt der Symbolwahrscheinlichkeiten
- des Strings $P(s) = \prod_{i=1}^{S} P_X(s(i))$
- Zur Codierung dieses Intervalls benötigen wir

$$-\log_2 P(s) = -\sum_{i=1}^{S} \log_2 P_X(s(i))\ Bit$$

- Ist *S* sehr gross, so tauchen die Einzelsymbole gemäss ihrer Wahrscheinlichkeiten auf und die mittlere Codelänge wird

$$\frac{-\log_2 P(s)}{S} = -\frac{1}{S}\sum_{i=1}^{S} \log_2 P_X(s(i)) = -\sum_{i=1}^{L} P_X(x_i)\log_2 P_X(x_i) = H(X)$$