ETH Zürich
Dept. Informatik

Prof. Markus Gross
WS 06/07

Informationstheorie

Lösung 9

9.1 Lempel-Ziv

a) Der Input wird wie folgt in Teilstücke zerlegt:
`0|01|00|1|11|010|011|0101`
Der entstehende Code ist dann:
`000,0|001,1|001,0|000,1|100,1|010,0|010,1|110,1`

b) Wenn das Präfix mit 3 bit codiert worden ist, wird der Code wie folgt unterteilt:
`000,1|001,1|001,0|000,0|011,0|101,1|100,0|100,1`
Daraus ergibt sich der Klartext:
`1|11|10|0|100|1001|00|01`

c) Ein modifiziertes Verfahren benutzt immer k = ceil(log(x)) Bits zum Codieren des Präfix, wobei x die aktuelle Anzahl von Einträgen im Wörterbuch ist. Der Decoder kennt an jeder Position im String die aktuelle Länge des Wörterbuchs, kann also auch k berechnen. Somit ist kein Präfixfreier Code zur Codierung der Präfixindexlänge notwendig, die Indizes können einfach in binärer Form gespeichert werden.

9.2 Vergleich von Codierungsverfahren

- Huffmann: Codiert Symbole aus einer Quelle über einem beliebigen Alphabet. Die Menge der Codewörter ist präfixfrei, und damit eindeutig decodierbar. Huffmann-Codes sind optimal. Um einen Huffmann-Code zu erzeugen, muss die Quellenstatistik, d.h. die Wahrscheinlichkeiten der einzelnen Elemente des Alphabets bekannt sein. Huffmann-Codes werden in der Praxis haeufig zur Kompression eingesetzt, wenn die Quellenstatistik bekannt ist oder geschätzt werden kann.

- Shannon-Fano: Codiert Symbole aus einer Quelle über einem beliebigen Alphabet. Die Menge der Codewörter ist präfixfrei, und damit eindeutig decodierbar. Shannon-Fano-Codierung ist nicht notwendigerweise optimal. Um einen Shannon-Fano-Code zu erzeugen, muss die Quellenstatistik bekannt sein. Shannon-Fano-Codes könnten ähnlich wie Huffmann Codes zur Kompression eingesetzt werden, in der Praxis wird Shannon-Fano allerdings praktisch nie benutzt, da Huffmann dieselben Voraussetzungen hat und bessere Codes erzeugt.

- Codierung der ganzen Zahlen: Codiert ganze Zahlen. Die Menge der Codewoerter ist präfixfrei, und damit eindeutig decodierbar. Man kann jeden endlichen String als ganze Zahl auffassen und so codieren, ohne weitere Annahmen zu machen. Die Codierung der ganzen Zahlen wird benötigt, um Zahlen präfixfrei abzuspeichern.

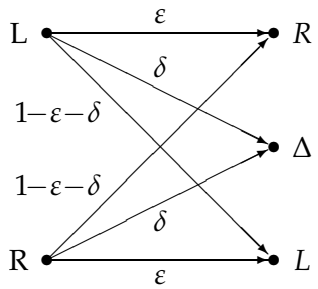
- Arithmetische Codierung: Codiert einen String über einem beliebigen Alphabet. Der sich ergebende Code repräsentiert den String, aber den Elemente des Alphabets werden keine Codewörter zugewiesen. Die Frage nach eindeutiger Decodierbarkeit der Codewortmenge stellt sich daher nicht. Ein mittels Arithmetischer Codierung erzeugter Codestring ist allerdings nur dann decodierbar, wenn die Länge des Eingabestrings bekannt ist (oder durch spezielle konventionen gekennzeichnet). Arithmetische Codierung ist in dem Sinne optimal, dass sie asymptotisch auf die Entropie codiert. Da es keine Codewörter gibt, kann das Kriterium der mittleren Codewortlänge nicht angewandt werden. Die Quellenstatistik des zu codierenden Strings muss bekannt sein. Arithmetische Codierung wird zur Kompression von Strings mit bekannter Länge bei bekannter Quellenstatistik eingesetzt.

- Intervalllängencodierung: Codiert Symbole aus einer Quelle mit beliebigem Alphabet. Auch bei der Intervalllängencodierung wird den Elementen des Quellalphabets keine Codewörter zugeordnet. Die erzeugten Codestrings sind allerdings ohne weitere Information decodierbar. Die Intervallängencodierung komprimiert Quellen mit endlichem Alphabet nicht auf die Entropie, ist also in diesem Sinne nicht optimal. Durch Einführen von Blockcodes kann dieses Problem behoben werden. Die Quellenstatistik muss weder zur Codierung noch zur decodierung bekannt sein. Die Intervalllängencodierung wurde und wird zur Codierung von Information auf physikalischen Datenträgern verwendet, z.B. zur Darstellung von "sparse files" im ext2/3-Dateisystem.

- Lempel-Ziv: Codiert Strings über einem beliebigem Alphabet. Den einzelnen Elementen des Alphabets werden keine Codewörter zugeordnet. Der Code ist jedoch ohne weiteres Wissen eindeutig decodierbar. Lempel-Ziv codiert asymptotisch auf die Entropie, ist also in diesem Sinne optimal. Die Quellenstatistik wird nicht benötigt. Das Lempel-Ziv Verfahren ist ein klassisches Kompressionsverfahren, das ohne Vorwissen eingesetzt werden kann. Das Programm "compress" benutzt dieses Verfahren, Varianten davon sind in allen gängigen Kompressionsprogrammen zu finden.

9.3 Blockcode auf binär-symmetrischem Auslöschungskanal

a) Der Kanal zwischen Alice und Bob kann folgendermassen dargestellt werden (wir bezeichnen ihn als binär-symmetrischen Auslöschungskanal):



b) Der Kanal ist symmetrisch und deshalb ist $H(Y|X = x)$ für alle x gleich. Es muss also nur $H(Y)$ maximiert werden. Folgende Überlegungen zeigen, dass $H(Y)$ maximal ist, wenn

die Werte am Kanalinput gleichverteilt sind (X gleichverteilt). Wir führen eine Zufallsvariable S ein:

$$S = \begin{cases} 0 & \text{falls } Y = \Delta \\ 1 & \text{falls } Y \neq \Delta. \end{cases}$$

Dann ist

$$H(Y) = H(YS) = H(Y|S) + H(S) = P_S(0)H(Y|S=0) + P_S(1)H(Y|S=1) + H(S)$$

Die erste Gleichung gilt, weil S durch Y vollständig definiert ist. Da $H(Y|S=0) = 0$ erhalten wir

$$H(Y) = P_S(1)H(Y|S=1) + H(S) = (1-\delta)H(Y|S=1) + H(S)$$

$(1-\delta)$ und $H(S)$ hängen nicht von der Verteilung von X ab. $H(Y)$ ist also maximal, falls $H(Y|S=1)$ maximal. Dies wird erreicht, wenn $Y = L$ und $Y = R$ gleichwahrscheinlich sind. Dies wiederum ist dann der Fall, wenn $P_X(L) = \frac{1}{2}$ und $P_X(R) = \frac{1}{2}$.

Deshalb ist:

$$C = H([\frac{1}{2}(1-\delta), \delta, \frac{1}{2}(1-\delta)]) - H([\varepsilon, \delta, 1-\varepsilon-\delta]).$$

c) Die Kapazität des Kanals ist $C = 0.15098$ und die Rate $\frac{2}{15}$; somit ist die Rate kleiner als die Kapazität. Angenommen, Bob muss auf N Fragen jeweils eine von vier möglichen Antworten bestimmen. Gemäss dem Kanalcodierungstheorem 2.Teil kann Alice einen Code mit Codewortlänge $15N$ für die N Antworten finden, so dass die Decodierfehler-Wahrscheinlichkeit für Bob beliebig klein gemacht werden kann, falls N nur genügend gross gewählt wird.

d) Die einfache ML-Decodierregel ist:

1. ignoriere die Δ's
2. dekodiere zu demjenigen Wort mit minimaler Hammingdistanz