Universität Mannheim
Lehrstuhl für Praktische Informatik IV
Prof. Dr. W. Effelsberg
Christoph Kuhmünch, Gerald Kühne

Praktische Informatik II
SS 2000

Übungsblatt 5 - Musterlösung

Aufgabe 1: Huffman-Codierung [6 Punkte]

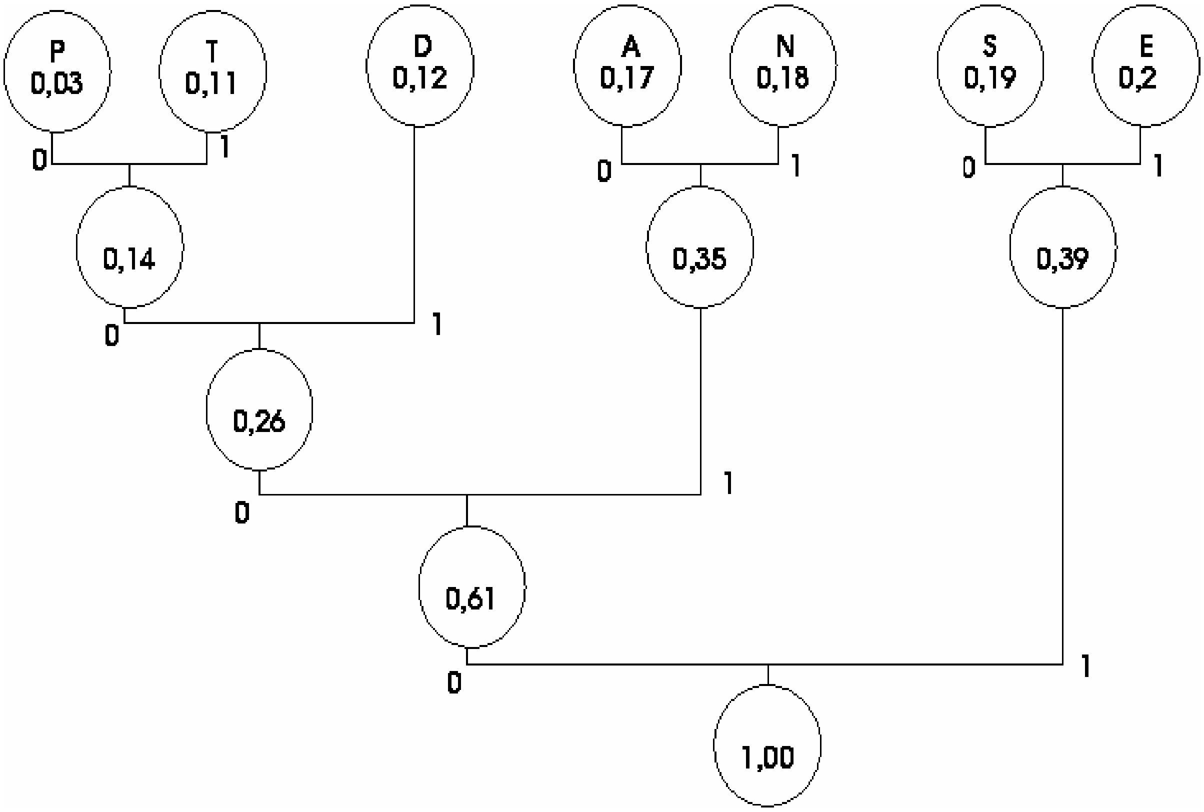
In dieser Aufgabe sollen Sie die Zeichenkette PANSADETTA mit Hilfe des Huffman-Algorithmus kodieren. Sortieren Sie dazu zunächst die Buchstaben der Zeichenkette aufsteigend nach der Häufigkeit ihres Vorkommens. Erstellen Sie anschließend einen Kodier-Baum. In Aufgabe a) soll der Baum nach der vorgegebenen Wahrscheinlichkeitstabelle erstellt werden, während Sie in Aufgabe b) die Wahrscheinlichkeitstabelle anhand der Häufigkeit der Buchstaben in der Zeichenkette PANSADETTA bestimmen sollen.

a) [3Punkte]
Die folgende Tabelle gibt die Wahrscheinlichkeiten für die Zeichen an. Benutzen Sie die Tabelle zur Erstellung des Baumes!

Buchstabe	E	A	T	N	P	S	D
Wahrscheinlichkeit	0.2	0.17	0.11	0.18	0.03	0.19	0.12

Geben Sie den entstehenden Huffman-Code für das Wort PANSADETTA sowie die Einsparung in Prozent an (verglichen mit einer 8-Bit-Kodierung der Zeichen).

Kodierbaum:



Buchstabe	Kodierung	alternativ
E	11	00
A	010	101
T	0001	1110
N	011	100
P	0000	1111
S	10	01
D	001	110

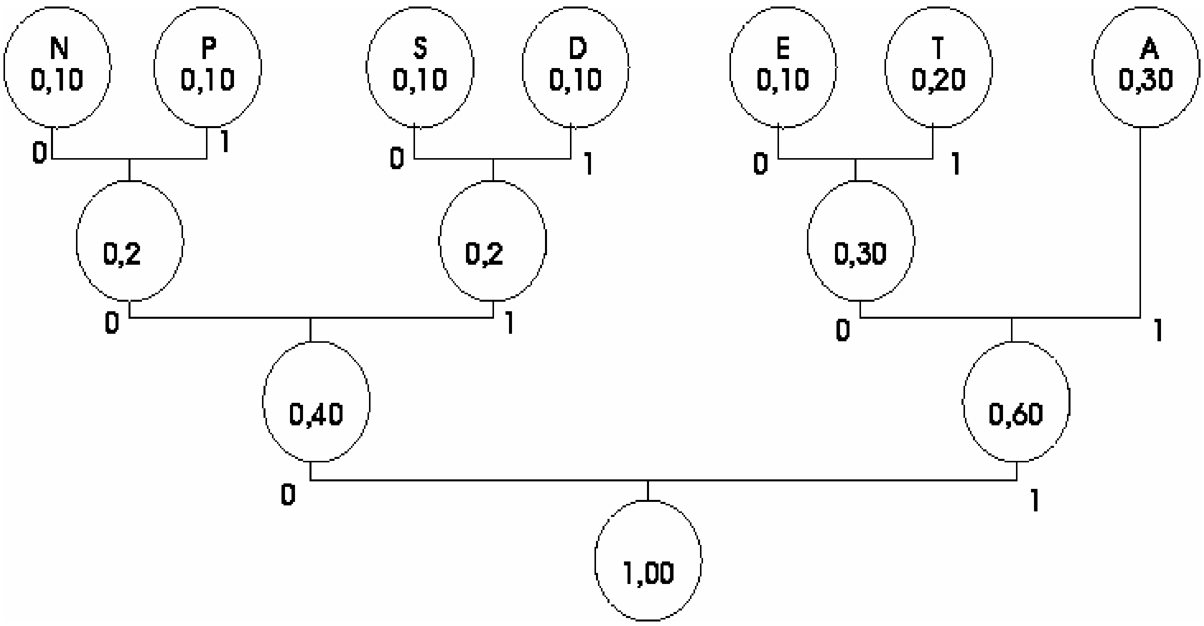
Kodierung: 0000 010 011 10 010 001 11 0001 0001 010
(1111 101 100 01 101 110 00 1110 1110 101)
Einsparung [%]: (80-31)/80 = 61,25%

b) [3 Punkte]
Erstellen Sie nun eine eigene Tabelle ausgehend von den Auftrittswahrscheinlichkeiten in der gegebenen Zeichenkette. Geben Sie wiederum den Code für das Wort PANSADETTA sowie die Einsparung in Prozent an.

Die Buchstaben besitzen folgende Häufigkeiten:

Buchstabe	Häufigkeit	Wahrscheinlichkeit
E	1	0,1
A	3	0,3
T	2	0,2
N	1	0,1
P	1	0,1
S	1	0,1
D	1	0,1

Daraus ergibt sich folgender Kodierbaum (eine von mehreren Alternativen):



Buchstabe	Kodierung	alternativ
E	100	011
A	11	00
T	101	010
N	000	111
P	001	110
S	010	101
D	011	100

Kodierung: 001 11 000 010 11 011 100 101 101 11 (110 00 111 101 00 100 011 010 010 00)
Einsparung [%]: (80-27)/80= 66,25%

Aufgabe 2: Hamming-Distanz [3 Punkte]

Gegeben sei folgender Code:

0000 0000	A
0000 0111	B
0011 1000	C
1100 0001	D
0001 1110	E

(a) [1 Punkt]
Wie groß ist die Hammingdistanz dieses Codes?

Die Hamming-Distanz `D` eines Codes `C` wird folgendermaßen bestimmt:
`D(C):= min {d(c1, c2) | c1, c2 aus C, c1 != c2}`

Im Beispiel-Code unterscheiden sich alle Zeichen um mindestens drei Bitpositionen, die Hamming-Distanz beträgt also 3.

(b) [1 Punkt]
Bei wievielen Bitfehlern können also fehlerhafte Codewörter noch eindeutig decodiert werden?

Zum Beheben von `e`–Bitfehlern wird mindestens eine Hamming-Distanz von `2e+1` benötigt. Bei einer Hamming-Distanz von `3` können daher maximal `1-Bit` Fehler korrigiert werden.

c) [1 Punkt]
Folgende Codewörter werden über ein Netzwerk an einen Empfänger übertragen:
(11000001; 00011000; 00001110; 00000111; 00011100)
Welche Zeichen dekodiert der Empfänger?

1100 0001	D	Kein Übertragungsfehler
0001 1000	C	Fehler in Bit 3
0000 1110	E	Fehler in Bit 4
0000 0111	B	Kein Übertragungsfehler
0001 1100	E	Fehler in Bit 7

Aufgabe 3: Hamming-Code mit 1-Bit Fehlerkorrektur [8 Punkte]

In dieser Aufgabe soll ein Schema zur Konstruktion eines 1-Bit-Fehler-korrigierenden Codes erarbeitet werden. Angenommen, es soll ein 5-Bit Code (D_0,...,D_4) gegen 1-Bit Fehler gesichert werden. Dann werden dazu 4 Redundanz-Bits (H_0,..., H_3) (vgl. Vorlesung, Seite 2-49) benötigt. Die Redundanz-Bits werden Hamming-Bits genannt und folgendermaßen angeordnet:

Bit	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Bedeutung	H_0	H_1	D_0	H_2	D_1	D_2	D_3	H_3	D_4

Das Hamming-Bit an Position 2^i *ist das even-parity Bit* P_i *all derer Daten-Bits, in deren Bit-Position* 2^i *vorkommt (die Bit-Position binär dargestellt, siehe auch nachfolgende Tabelle).*

Bit	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Bedeutung	H_0	H_1	D_0	H_2	D_1	D_2	D_3	H_3	D_4
P0	*1*	*0*	*1*	*0*	*1*	*0*	*1*	*0*	*1*
P1	*0*	*1*	*1*	*0*	*0*	*1*	*1*	*0*	*0*
P2	*0*	*0*	*0*	*1*	*1*	*1*	*1*	*0*	*0*
P3	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*1*

Beim Dekodieren werden die Hamming-Bits des empfangenen Code-Wortes neu berechnet und per XOR mit den empfangenen Hamming-Bits verknüpft. Das Eregbnis ist bei fehlerfreier Übertragung gleich 0. Tritt ein 1-Bit Fehler auf, dann gibt die XOR-Verknüpfung die Position des Fehlers an.

Folgende Daten sind mit dem oben vorgestellten Hammingcode kodiert (even parity):

(a) [4 Punkte] *0 1001 1011*

(b) [4 Punkte] *0 0100 0111*

Wie heißen die ursprünglichen Nutzdaten?

a) 010011011

Bitposition(3) des ersten Datenbits (0)	Bitposition(5) des zweiten Datenbits (1)	Bitposition(6) des dritten Datenbits (1)	Bitposition(7) des vierten Datenbits (0)	Bitposition(9) des fünften Datenbits (1)	Even-parity der 1-bits bei Datenbit = 1
1	1	0	1	1	0 (H_0)
1	0	1	1	0	1 (H_1)
0	1	1	1	0	0 (H_2)
0	0	0	0	1	1 (H_3)

Die berechneten Hamming-Bits stimmen mit den übertragenenen überein (0101 XOR 0101 = 0000). Die übertragenen Daten sind also korrekt: 01101.

b) 001000111

Bitposition(3) des ersten Datenbits (1)	Biposition(5) des zweiten Datenbits (0)	Bitposition(6) des dritten Datenbits (0)	Bitposition(7) des vierten Datenbits (1)	Bitposition(9) des fünften Datenbits (1)	Even-parity der 1-bits bei Datenbit = 1
1	1	0	1	1	1 (H_0)
1	0	1	1	0	0 (H_1)
0	1	1	1	0	1 (H_2)
0	0	0	0	1	1 (H_3)

Die berechneten Hamming-Bits stimmen mit den übertragenenen nicht überein. 0001 XOR 1011 = 1010 => die fehlerhafte Bit-Position ist fünf (das least significant bit steht hier vorne). Bit-Position fünf steht das zweite Daten-Bit, dieses muß demnach 1 und nicht 0 sein. Die korrekten Daten lauten also 11011.

Aufgabe 4: Hamming-Code mit n-Bit Fehlerkorrektur [3 Punkte]

Es sei folgender, fehlerkorrigierender Hamming-Code gegeben:
(0000000000; 0000011111; 1111100000; 1111111111)

a) [2 Punkte]
Wie groß ist das Verhältnis der von der in diesem Code codierten Nachricht benötigten Bits zur Anzahl der Bits der Nachricht selber?

Ein Wort des Hamming-Codes codiert 2 Bits Information.
10 Bits zur Übertragung von 2 Bits: Verhältnis 5:1

b) [1 Punkt]
Wie groß ist das Verhältnis, wenn man jedes Wort der Nachricht mit einem Parity-Bit versieht, statt diesen Hamming-Code zu verwenden?

3 Bits zur Übertragung von 2 Bits: Verhältnis 3:2