Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

GTI – ÜBUNG 2

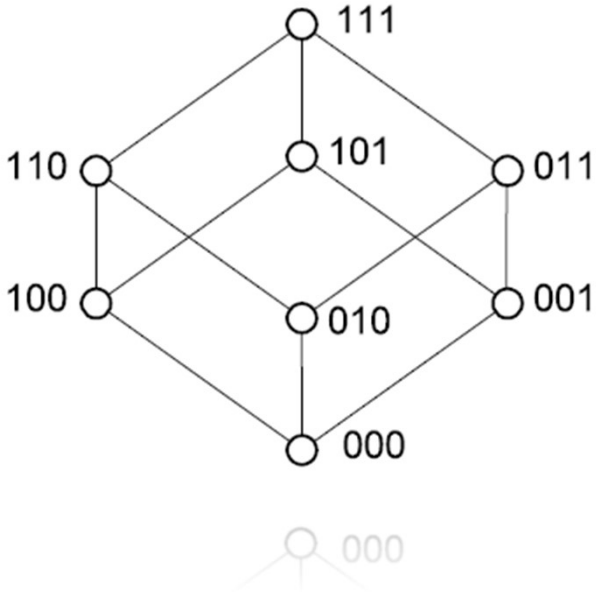
FEHLERERKENNUNG, FEHLERKORREKTUR UND HUFFMAN

Aufgabe 1 – Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Beschreibung

- Das Bild zeigt ein Diagramm, dass die Nachbarschaftsbeziehungen für einen Code mit drei Binärstellen darstellt.



Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Begriffsklärung:

Hamming-Distanz:	Anzahl der unterschiedlichen Binärstellen zwischen zwei gleich langen Codewörtern ($c1, c2 \in \{0, 1\}^n$)
Minimale H-D:	minimale Anzahl an Stellen, die sich bei dem Übergang zwischen zwei beliebigen Codewörtern eines Codes ändern
N-fach Fehler:	durch externe Störeinflüsse (z.B. Magnetfeld) ändern sich bei einem vorher korrekten Binärwort n unterschiedliche Stellen

Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Hamming-Distanz müssen die gültigen Codeworte aufweisen, damit Einzelfehler erkannt werden können?
- Wie viele Zeichen können so mit drei Binärstellen maximal codiert werden?

Hinweis: die minimale Hamming-Distanz ist entscheidend

Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Hamming-Distanz müssen die gültigen Codeworte aufweisen, damit Einzelfehler erkannt werden können?

Sei die minimale Hamming-Distanz gegeben als $HD_{min} = d$

Dann können (d-1)-Fehler erkannt werden.

Lösung: $1 = d - 1 \qquad d = 2$

man benötigt also eine minimale Hamming-Distanz von 2

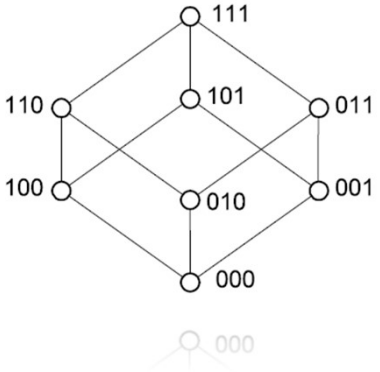
Erklärung: $HD_{min} = d$ bedeutet, dass sich zwischen jedem Codewort genau d Binärstellen ändern. Ist also ein d-1 Fehler aufgetreten, so ändern sich von den richtigen Codewörtern zu dem falschen Codewort nicht genau d Stellen und das fehlerhafte Codewort kann erkannt werden.

Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wie viele Zeichen können so mit drei Binärstellen maximal codiert werden?

Sei die minimale Hamming-Distanz 2.



Lösung: visuell:

man betrachte den Würfel, starte bei einer Ecke und schaut, wie viele Codewörter erreichbar sind (4)

rechnerisch:

das Startwort besitzt drei Stellen. Die Möglichkeiten zwei Binärstellen zu verändern liegen bei $\binom{3}{2}$ = 3. Macht mit dem Ausgangswort 4.

Im Spezialfall: $\frac{2^{Binärstellen}}{2^{H_{min}-1}} = 2^{Stellen-H_{min}+1}$

Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Hinweis

Es geht um die minimale Hamming-Distanz.

Beispiel: 4 Binärstellen, minimale Hamming-Distanz sei 2

Codewörter: Gegeben sei Codewort 0000. Wie lauten die Restlichen?

0000, 1100, 1010, 1001, 0110, 0101, 0011 und 1111 (da minimal!)

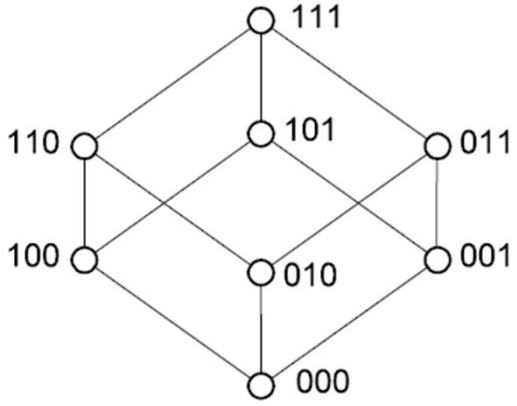
Das hat auch die Konsequenz, dass der Binomialkoeffizient zur Bestimmung nur eine Untergrenze liefert!

Aufgabe 1 – Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Hamming-Distanz müssen die gültigen Codewörter aufweisen, damit die Korrektur von Einzelfehlern möglich ist?
- Wie viele Zeichen können jetzt maximal codiert werden?

Hinweis: auch hier spielt HD_{min} eine Rolle



Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Hamming-Distanz müssen die gültigen Codewörter aufweisen, damit die Korrektur von Einzelfehlern möglich ist?

Sei die minimale Hamming-Distanz gegeben als $HD_{min} = d$

Dann können (d-1)/2-Fehler korrigiert werden.

Lösung: $1 = (d - 1)/2 \qquad d = 3$

man benötigt also eine minimale Hamming-Distanz von 3

Erklärung: $HD_{min} = d$ bedeutet, dass sich zwischen jedem Codewort minimal d Binärstellen ändern. Ist also ein (d-1)/2 Fehler aufgetreten, so kann der Fehler sicher erkannt werden (d-1) und auch dem ursprünglichen Codewort zugeordnet werden (erlaubt die Schrittweite /2).

Aufgabe 1 - Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wie viele Zeichen können jetzt maximal codiert werden?

Sei die minimale Hamming-Distanz 3.

Lösung: visuell:

man betrachte den Würfel, starte bei einer Ecke und schaut, wie viele Codewörter erreichbar sind (2)

rechnerisch:

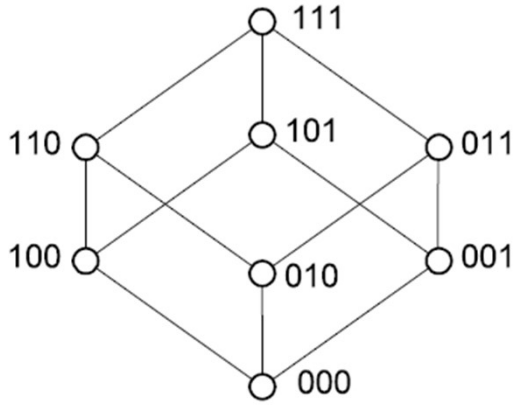
das Startwort besitzt drei Stellen. Die Möglichkeiten drei Binärstellen zu verändern liegen bei $\binom{3}{3}$ = 1. Macht mit dem Ausgangswort 2.

Aufgabe 1 – Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Die beiden Zeichen A und B sollen so codiert werden, dass Einzelfehler korrigierbar sind. Wie viele Lösungen sind für die Codierung der beiden Zeichen mit drei Binärstellen möglich? Geben Sie eine Lösung an.

Hinweis: hier ist der Würfel als Visualisierung sehr hilfreich



Aufgabe 1 – Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Die beiden Zeichen A und B sollen so codiert werden, dass Einzelfehler korrigierbar sind. Wie viele Lösungen sind für die Codierung der beiden Zeichen mit drei Binärstellen möglich? Geben Sie eine Lösung an.

Lösung:

visuell:

der Würfel besitzt acht Ecken, also acht Startmöglichkeiten und somit auch acht Lösungen, da eine minimale Hamming-Distanz von drei vorliegt (diagonal gegenüber liegende Ecken als Lösungspaar)

rechnerisch:

$HD_{min} = 3 \rightarrow$ Startwort wird nur ein weiteres Codewort zugeordnet

Anzahl Startwörter: $2^{STELLEN} = 2^3 = 8 \rightarrow 8$ Lösungen

Aufgabe 1 – Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Bei der Datenübertragung mit einer Codierung nach c, wurde genau eine Binärstelle falsch übertragen. Die folgenden Daten wurden empfangen:

  0 1 1 1 1 0 0 0 1 1 1 0

  Korrigieren Sie den Fehler.

Hinweis: welche Basis-Codewörter wurden hier wohl verwendet?

Aufgabe 1 – Hamming-Distanz

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Bei der Datenübertragung mit einer Codierung nach c, wurde genau eine Binärstelle falsch übertragen. Die folgenden Daten wurden empfangen:

  0 1 1 1 1 0 0 0 1 1 1 0

  Korrigieren Sie den Fehler.

Lösung: Gliederung:

011 110 001 110

Ursprünglichen Codewörter:

Es gibt nur zwei Codewörter (c). Und auch nur einen Fehler. Somit ist das Codewort 110 auf jeden Fall korrekt. Das andere Codewort lautet somit 001 (HD_{min} = 3). 011 ist also falsch. Eine falsche Binärstelle führt zu 001 (hier (d-1)/2 Aussage vergewissern)

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Beschreibung

- Am Ende einer längeren Übertragungs-strecke wird die folgende Nachricht im ASCII-Code empfangen:
- Es ist bekannt, dass der Sender die sieben Bits des ASCII-Codes um ein sogenanntes Paritätsbit (ganz links) ergänzt hat.

Empfangener Code	ASCII-Zeichen
0 100 0111	'G'
1 101 0100	'T'
1 100 1001	'I'
1 010 0100	'$'
0 110 1001	'i'
1 111 0011	's'
0 111 0100	't'
1 010 0000	' '
1 111 0110	'v'
0 110 1111	'o'
0 110 1100	'l'
0 110 1100	'l'
0 ?10 ?100	'?'
0 ?10 ?100	'?'

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Begriffsklärung:

ASCII-Code: American Standard Code for Information Interchange
codiert mit 7 Bit 128 Zeichen (für Amerika ausreichend)

Paritätsbit: zu übertragende Code-wörter werden mit einem zusätzlichen Bit gesichert

Gerade Parität: Anzahl der Einsen des neuen Codeworts gerade

Ungerade Par.: Anzahl der Einsen des neuen Codeworts ungerade

LSB	MSB							
Binär	000	001	010	011	100	101	110	111
	Steuerzeichen				Großbuchstaben		Kleinbuchstaben	
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	„	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	´	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welches Zeichen wurde offensichtlich falsch übertragen?

Empfangener Code	ASCII-Zeichen
0 100 0111	'G'
1 101 0100	'T'
1 100 1001	'I'
1 010 0100	'$'
0 110 1001	'i'
1 111 0011	's'
0 111 0100	't'
1 010 0000	' '
1 111 0110	'v'
0 110 1111	'o'
0 110 1100	'l'
0 110 1100	'l'

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welches Zeichen wurde offensichtlich falsch übertragen?

Semantisch:

Das $

Logisch:

Das $ weist eine ungerade Parität auf.

Die restlichen Zeichen eine gerade Parität.

Empfangener Code	ASCII-Zeichen
0 100 0111	'G'
1 101 0100	'T'
1 100 1001	'I'
1 010 0100	'$'
0 110 1001	'i'
1 111 0011	's'
0 111 0100	't'
1 010 0000	' '
1 111 0110	'v'
0 110 1111	'o'
0 110 1100	'l'
0 110 1100	'l'

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Das letzte Wort lautete vor der Übertragung „toll“ und nicht „voll“. Warum ist der von der Übertragungsstrecke verursachte Fehler nicht erkennbar?

Empfangener Code	ASCII-Zeichen
0 100 0111	'G'
1 101 0100	'T'
1 100 1001	'I'
1 010 0100	'$'
0 110 1001	'i'
1 111 0011	's'
0 111 0100	't'
1 010 0000	' '
1 111 0110	'v'
0 110 1111	'o'
0 110 1100	'l'
0 110 1100	'l'

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Das letzte Wort lautete vor der Übertragung „toll“ und nicht „voll“. Warum ist der von der Übertragungsstrecke verursachte Fehler nicht erkennbar?

Hinweis: welche Arten von Fehlern sind erkennbar

Empfangener Code	ASCII-Zeichen
0 100 0111	'G'
1 101 0100	'T'
1 100 1001	'I'
1 010 0100	'$'
0 110 1001	'i'
1 111 0011	's'
0 111 0100	't'
1 010 0000	' '
1 111 0110	'v'
0 110 1111	'o'
0 110 1100	'l'
0 110 1100	'l'

Aufgabe 2 - Fehlererkennung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Das letzte Wort lautete vor der Übertragung „toll“ und nicht „voll“. Warum ist der von der Übertragungsstrecke verursachte Fehler nicht erkennbar?

Welche Arten von Fehlern sind erkennbar?

Fehler mit ungerader Anzahl an Fehlern (1, 3, …)

Was kann also passiert sein?

Zweifachfehler, Vierfachfehler, Sechsfachfehler, …

Hier: Zweifachfehler, da Codierung von ‚t‘ schon gegeben

Empfangener Code	ASCII-Zeichen
0 100 0111	'G'
1 101 0100	'T'
1 100 1001	'I'
1 010 0100	'$'
0 110 1001	'i'
1 111 0011	's'
0 111 0100	't'
1 010 0000	' '
1 111 0110	'v'
0 110 1111	'o'
0 110 1100	'l'
0 110 1100	'l'

Aufgabe 3 - Blocksicherung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Beschreibung

- Bei der Übertragung wichtiger Daten im ASCII-Code wird eine Blocksicherung durchgeführt. Die Prüfbits werden so erzeugt, dass eine gerade Parität entsteht. Die folgende Tabelle zeigt die empfangenen Daten. Offensichtlich wurden nicht alle Daten richtig übermittelt.

	Binärcode	Prüfbit	Hex	ASCII
Block 1	101 0010	1	52	'R'
	100 1001	1	49	'I'
	101 0011	1	53	'S'
	010 1000	0	48	'H'
	111 0000	1	Prüfsumme	

	Binärcode	Prüfbit	Hex	ASCII
Block 2	101 0000	1	50	'P'
	100 1001	1	49	'I'
	100 0111	0	47	'G'
	010 0001	0	21	'!'
	111 1011	0	Prüfsumme	

Aufgabe 3 - Blocksicherung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Begriffsklärung:

Blocksicherung: Erweiterung der Paritätssicherung
Parität wird spalten- und zeilenweise geprüft
Dazu werden empfangene Daten in Blöcke aufgeteilt

Beispiel: Block besteht aus drei Binärwörtern (drei Stellen), ungerade Parität

Eingabestrom: 0 1 0 1 1 1 0 1 0 0 1 1 0 1 0 1 1 1

Blöcke:

	0 1 0 | 0	0 1 1 | 1
	1 1 1 | 0	0 1 0 | 0
	0 1 0 | 0	1 1 1 | 0
	0 0 0 | 1	0 0 1 | 0

Aufgabe 3 - Blocksicherung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Fehler (Anzahl, Einfach-/Mehrfachfehler) sind korrigierbar?

Hinweis: wie entdeckt man einen Fehler?

	Binärcode	Prüfbit	Hex	ASCII
Block 1	101 0010	1	52	'R'
	100 1001	1	49	'I'
	101 0011	1	53	'S'
	010 1000	0	48	'H'
	111 0000	1	Prüfsumme	

	Binärcode	Prüfbit	Hex	ASCII
Block 2	101 0000	1	50	'P'
	100 1001	1	49	'I'
	100 0111	0	47	'G'
	010 0001	0	21	'!'
	111 1011	0	Prüfsumme	

Aufgabe 3 - Blocksicherung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Fehler (Anzahl, Einfach-/Mehrfachfehler) sind korrigierbar?

Wie entdeckt man einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen und Spalten (falsche Parität).

Beispiel: Block besteht aus drei Binärwörtern (drei Stellen), ungerade Parität

Blöcke:

0	1	0	|	0
0	1	1	|	0
0	1	0	|	0
0	0	0	|	1

0	1	1	|	0
0	1	0	|	0
0	1	1	|	0
0	0	0	|	1

Konklusion:

Nur Einfachfehler pro Block korrigierbar

Denn (s. zweites Beispiel) bei 2-Fehler:

Zwei mögliche Fehlerkonstellationen

(Links oben, Rechts Unten) oder
(Rechts oben, Links Unten) der SP

Aufgabe 3 - Blocksicherung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Die aufgetretenen Fehler seien korrigierbar. Korrigieren Sie die entsprechenden Binärstellen in der Tabelle. Bestimmen Sie für die korrigierten Codewörter das zugehörige ASCII-Zeichen

	Binärcode	Prüfbit	Hex	ASCII
Block 1	101 0010	1	52	'R'
	100 1001	1	49	'I'
	101 0011	1	53	'S'
	010 1000	0	48	'H'
	111 0000	1	Prüfsumme	

	Binärcode	Prüfbit	Hex	ASCII
Block 2	101 0000	1	50	'P'
	100 1001	1	49	'I'
	100 0111	0	47	'G'
	010 0001	0	21	'!'
	111 1011	0	Prüfsumme	

Aufgabe 3 - Blocksicherung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Die aufgetretenen Fehler seien korrigierbar. Korrigieren Sie die entsprechenden Binärstellen in der Tabelle. Bestimmen Sie für die korrigierten Codewörter das zugehörige ASCII-Zeichen

	Binärcode	Prüfbit	Hex	ASCII
Block 1	1011 0010	1	52	'R'
	1001 1001	1	49	'I'
	1011 0011	1	53	'S'
	0101 1000	0	48	'H'
	1111 0000	1	Prüfsumme	

	Binärcode	Prüfbit	Hex	ASCII
Block 2	1010 0000	1	50	'P'
	1001 1001	1	49	'I'
	1001 0111	0	47	'G'
	0100 0001	0	21	'!'
	1110 1011	0	Prüfsumme	

Codewort 1: 1 0 1 0 0 1 1 → 100 0011 (= 43 HEX → ‚C')

Codewort 2: 1 0 1 0 0 0 0 → 101 0100 (= 54 HEX → ‚T')

Halbzeit: kurz durchschnaufen

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Folgendes Spiel wird mit einem zweiten Partner gespielt:
Jeder der beiden Mitspieler nennt abwechselnd eine Zahl zwischen 1-10. Alle genannten Zahlen werden aufeinander aufaddiert. Wer in seiner Runde durch nennen seiner Zahl die Summe 100 erreichen kann, hat das Spiel gewonnen.

Du bist Erster an der Reihe. Was ist deine Strategie, um das Spiel sicher zu gewinnen (mit welcher Zahl solltest du anfangen und wie lautet deine Taktik)?

Halbzeit: Lösung

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Eine Lösung: mit der Zahl 1 anfangen und die gegnerischen Zahlen immer auf 11 ergänzen.

$$1 + 9 \cdot 11 = 100$$

Begründung: du musst es schaffen, dass dein Gegner mit seiner letzten Zahl gerade nicht die 100 erreicht und egal welche Zahl er nennt, du auf 100 ergänzen kannst.
Das funktioniert nur mit der Zahl 89 $(1 + 8 \cdot 11)$. 11 ist dabei die Zahl, auf die du immer ergänzen kannst unabhängig der gewählten Zahl des Mitspielers.

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Beschreibung

- Gegeben sei ein nicht fehlertolerantes Kommunikationssystem, welches in der Lage ist, einstellige Codeworte in hexadezimaler Darstellung zu übertragen. Es soll nun dahingehend erweitert werden, dass es mittels eines Hamming-Codes Zweifachfehler erkennen oder Einfachfehler korrigieren kann.

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Begriffsklärung:

Hexadezimalsystem:

- System zur Basis 16 (0-9, A, B, C, D, E, F)
- leichte Umwandlung einer vierstelligen Binärzahl (da 2^4)
- erspart viel Schreibaufwand (01011010 → 5A)

Hamming-Code:

- es lässt sich pro Zeile nach bisherigen Wissen nur 1 Paritätsbit konstruieren
- Ziel: mehrere Prüfbits (y_1, y_2, …) für ein Codewort (x_1, x_2, …)
- Lösung: Überdeckungstabellen (y_1 betrachtet x_1, x_2, x_4; y_2 betrachtet …)
- Konstruktion der Zuordnung Prüfbit → Informationsbits durch eine Tabelle

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Hamming-Distanz (HD) wird benötigt, um die geforderte Fehlertoleranz zu erreichen?

Hinweis: Fehlertoleranz = Zweifachfehler erkennen/ Einfachfehler korrigieren

vgl. Aufgabe 1

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Welche Hamming-Distanz (HD) wird benötigt, um die geforderte Fehlertoleranz zu erreichen?

Lösung: Zweifachfehler erkennen

$d - 1 = 2 \rightarrow d = 3$

Einfachfehler korrigieren

$(d - 1)/2 = 1 \quad \Leftrightarrow \quad d - 1 = 2 \rightarrow d = 3$

Beides Mal also eine Hamming-Distanz von 3.

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wie viele Bits werden nun jeweils benötigt, um die Informationen und die Paritätsbits nach Hamming zu codieren?
- Wie lang wird das gesamte zu übertragende Codewort?

Hinweis: es müssen das Codewort und die Paritätsbits übertragen werden

wie viele Bits umfasst eine Hexadezimalzahl?

wie viele Paritätbits lassen sich aus dieser Codewortlänge kreieren?

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wie viele Bits werden nun jeweils benötigt, um die Informationen und die Paritätsbits nach Hamming zu codieren?
- Wie lang wird das gesamte zu übertragende Codewort?

Wie viele Bits umfasst eine Hexadezimalzahl?

Immer vier ($2^4 = 16$). $0_{16} = 0000_2$, $A_{16} = 1010_2$, …

Wie viele Paritätsbits lassen sich aus dieser Codewortlänge kreieren?

Minimales k, für das gilt: $2^k - k - 1 \geq m$

k: Anzahl an Paritätsbits m: Anzahl an Informationsstellen

Bei Hexadezimalzahlen also: $2^k - k - 1 \geq 4$ $2^k - k \geq 5$ k = 3

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wie viele Bits werden nun jeweils benötigt, um die Informationen und die Paritätsbits nach Hamming zu codieren?
- Wie lang wird das gesamte zu übertragende Codewort?

Gesamte Länge:

4 (Informationsbits) + 3 (Prüfbits) = 7 Bit pro Codewort

Beispiel: C = 1100 $\rightarrow_{PB}$ $x_1x_2x_3x_4y_1y_2y_3$ (Infobit als x, Pbit als y)

Nach der genannten Formel ergibt sich folgende Verteilung:

Informationsbits	**2-4**	**5-11**	**12-26**	**27-57**	**…**
Paritätsbits	3	4	5	6	…

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie nun den Hamming-Code und ordnen Sie den Codeworten die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codeworte soll wie folgt aussehen: x_m; … ;x_1;y_k; … ;y_1.

Hinweis: die Tabelle lässt sich durch die absteigende Darstellung der natürlichen Zahlen in Binärform erzielen (B_N, B_{N-1}, B_{N-2}, …, B_1) beginnend bei:

$N = 2^{PARITAETSBITS} - 1$

Prüfbits (y) sind Spalten mit nur einer 1

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie nun den Hamming-Code und ordnen Sie den Codeworten die entsprechenden Hexadezimalwerte zu.

1: Natürlichen Zahlen binär in absteigender Reihenfolge

7:	**6:**	**5:**	**4:**	**3:**	**2:**	**1:**
1	1	1	1	0	0	0
1	1	0	0	1	1	0
1	0	1	0	1	0	1

2: Spalten mit genau einer 1 werden einem Paritätsbit zugeordnet

x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	1	1	0	0	0
1	1	0	0	1	1	0
1	0	1	0	1	0	1

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie nun den Hamming-Code und ordnen Sie den Codeworten die entsprechenden Hexadezimalwerte zu.

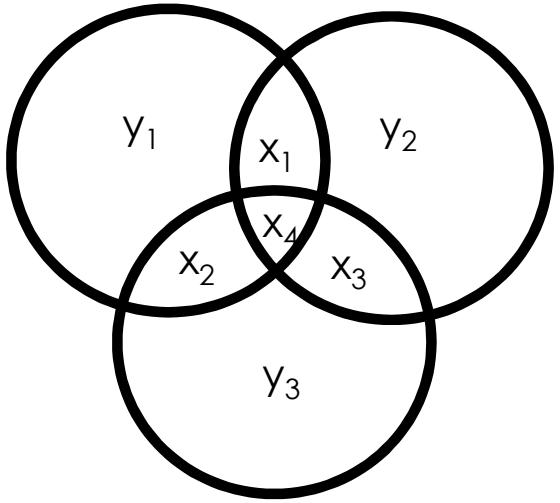
3: Jedes y ist Ergebnis eines XORs aller x=1-Komponenten seiner 1 Reihe

x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	1	1	0	0	0
1	1	0	0	1	1	0
1	0	1	0	1	0	1

$Y_1 = x_1$ XOR x_2 XOR x_4

$Y_2 = x_1$ XOR x_3 XOR x_4

$Y_3 = x_2$ XOR x_3 XOR x_4



Visualisierung der Zuordnung

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Einschub: XOR/Kontra – oder Antivalenz

  - mathematischer Operator zur Verknüpfung zweier Binärstellen
  - Exklusives Oder, d.h. es darf nur genau ein Operator erfüllt sein
    als Merkhilfe: ODER exklusive UND
  - im Deutschen: entweder a oder b, aber nicht beide

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

$C = A \oplus B$

Assoziativ: A XOR B XOR C = (A XOR B) XOR C = A XOR (B XOR C)

Für die schnelle Anwendung im Fall multipler XORs (als gerade Parität):

Die Anzahl der Einsen muss immer ungerade sein, damit eine 1 gesetzt wird

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie nun den Hamming-Code und ordnen Sie den Codeworten die entsprechenden Hexadezimalwerte zu.

4: Aufsteigende Darstellung aller Hex-Werte im Binärsystem

X4	X3	X2	X1	Y3	Y2	Y1	HEX
0	0	0	0				0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

X4	X3	X2	X1	Y3	Y2	Y1	HEX
1	0	0	0				8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie nun den Hamming-Code und ordnen Sie den Codeworten die entsprechenden Hexadezimalwerte zu.

5: Zuordnung der jeweiligen Prüfbits durch XOR-Anwendung

X4	X3	X2	X1	Y3	Y2	Y1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1	0	0	0	7

X4	X3	X2	X1	Y3	Y2	Y1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1	1	1	1	F

Aufgabe 4 - Fehlerkorrektur

- Bei einer Übertragung mit diesem Kommunikationssystem wurde folgende Binärfolge empfangen:
- 0101000010010010011001110111111010 0
- Überprüfen Sie anhand Ihrer Code-Tabelle, ob der Empfang der Codeworte fehlerfrei erfolgt ist und führen Sie falls notwendig eine Korrektur durch.

Hinweis: wie lang war ein Codewort?

überprüfe, ob die Bits richtig gesetzt sind.

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Bei einer Übertragung mit diesem Kommunikationssystem wurde folgende Binärfolge empfangen:
- 0101000010010010011001110111111 0100
- Überprüfen Sie anhand Ihrer Code-Tabelle, ob der Empfang der Codeworte fehlerfrei erfolgt ist und führen Sie falls notwendig eine Korrektur durch.

Zergliederung: ein Codewort besteht aus 7 Binärstellen

0101000 0100100 1001100 1110111 1110100

Auf vier Informationsbits folgen drei Prüfbits:

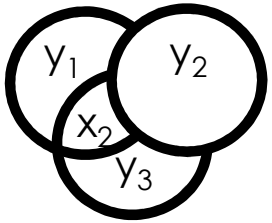
0101000 0100100 1001100 1110111 1110100

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Zwei falsche Prüfbits:

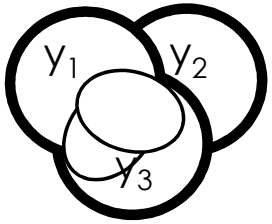
0101000 0101000 0111



Folge: Fehlerhaftes Informationsbit als Schnittmenge der beiden falschen Prüfbits ohne Schnitt des richtigen Bits (hier: $y1 \cap y3 \setminus y2$)

Ein falsches Prüfbit:

0100100 0100100 0100



Folge: Fehlerhaftes Paritätsbit, da die Menge des falschen Prüfbits ohne Schnittmenge der richtigen Bits leer ist (hier: $y2 \setminus y1 \setminus y3$)

0 falsche Prüfbits → richtig 3 falsche Prüfbits → x4 falsch

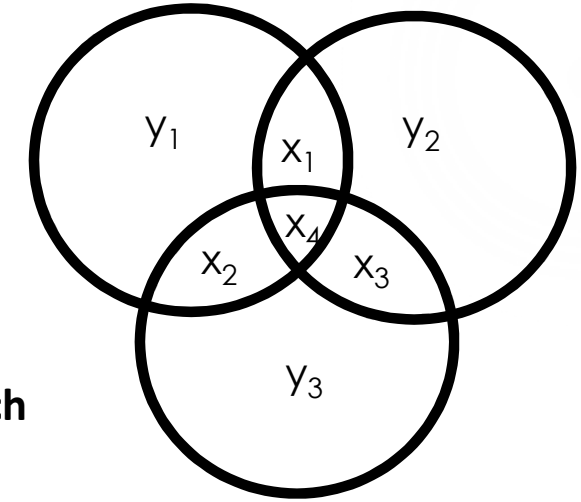
Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Eine weitere Art der Korrektur:

Dazu malt man sich die Zuweisung der Prüfbits tabellarisch auf:

Prüfbit	**x1**	**x2**	**x3**	**x4**
y1	1	1		1
y2	1		1	1
y3		1	1	1



Jede Spalte der Tabelle zeigt auf, welche Prüfbits falsch sein müssen, damit jenes Informationsbit falsch ist.

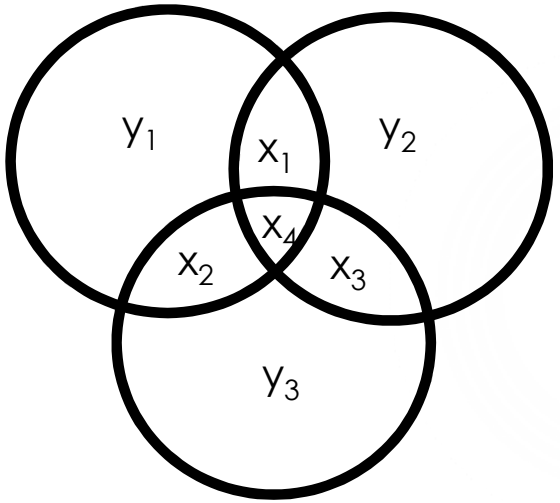
Sind y2 und y3 falsch, so ist x3 falsch. Dies funktioniert wunderbar auch für mehr als drei Prüfstellen (hier versagt die Kreistechnik).

Aufgabe 4 - Fehlerkorrektur

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Empfangen	Falsche PB	Korrektes Wort	
0101000	0101000	0111	Bit 2 falsch
0100100	0100100	0100	PB 2 falsch
1001100	1001100	1001	Richtig!
1110111	1110111	1111	Bit 1 falsch
1110100	1110100	1110	Richtig!

Visualisierung der Zuordnung



Prüfbit	**x1**	**x2**	**x3**	**x4**
y1	1	1		1
y2	1		1	1
y3		1	1	1

Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Begriffsklärung:

Kanalcodierung:	Hinzufügen von redundanten Informationen, um Sicherheit einer zu übertragenden Information zu erhöhen (vgl. Aufgabe 1-4); Schutz gegen Übertragungsfehler
Quellencodierung:	Reduzierung der Redundanz einer Codierung. Senken des benötigten Speicherplatzes und der Übertragungszeit
Huffman-Code:	Form einer Quellencodierung durch effiziente Zuweisung von Codewörtern variabler Länge unter Ausnutzung ihres Informationsgehaltes (bzw. generell der Entropie). Präfixfrei.
	Alternative Codierungen: z.B. Shannon-Fano-Code
Präfixfreiheit:	Kein Codewort ist Präfix eines anderen Codewortes

Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

Begriffsklärung:

Optimaler Code: Minimierung der durchschnittlichen Codewortlänge m

$\bar{m} = \sum_{i=1}^{n} p(x_i) \cdot m(x_i)$ m: Länge eines Zeichens

Entropie: Idealwert eines optimalen Codes (durchschnittlicher Informationsgehalt)

$H = \sum_{i=1}^{n} p(x_i) \cdot I(x_i)$ $I(x_i)$ bzw. $H_e(x_i)$

Der Informationsgehalt gibt also an, mit wie vielen Bits ein Codewort optimal codiert werden sollte. (deshalb Einheit Bit!)

Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie den Huffman-Codierungsbaum für die folgende Zeichenkette:

  ABRAKADABRASIMSALABIM

Hinweis:

1: Sortieren nach Häufigkeit

2: Geringste Häufigkeit als Blätter

3: Jeweils Knoten geringster Häufigkeit zum Baum hinzufügen

4: Beim fertigen Baum an jeden linken Ast eine 0, an jeden rechten Ast eine 1 schreiben

5: Codewörter als Pfad des Baumes von der Wurzel aus auslesen

Aufgabe 5 – Huffman-Code

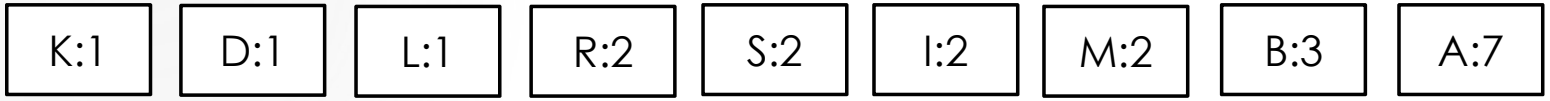
- Erstellen Sie den Huffman-Codierungsbaum für die folgende Zeichenkette:

  ABRAKADABRASIMSALABIM

1: Sortieren nach Häufigkeit

Buchstabe	**A**	**B**	**R**	**K**	**D**	**S**	**I**	**M**	**L**
Anzahl	7	3	2	1	1	2	2	2	1

2: Aufstellen der Blätter (nach Nummern sortiert, nicht alphabetisch)



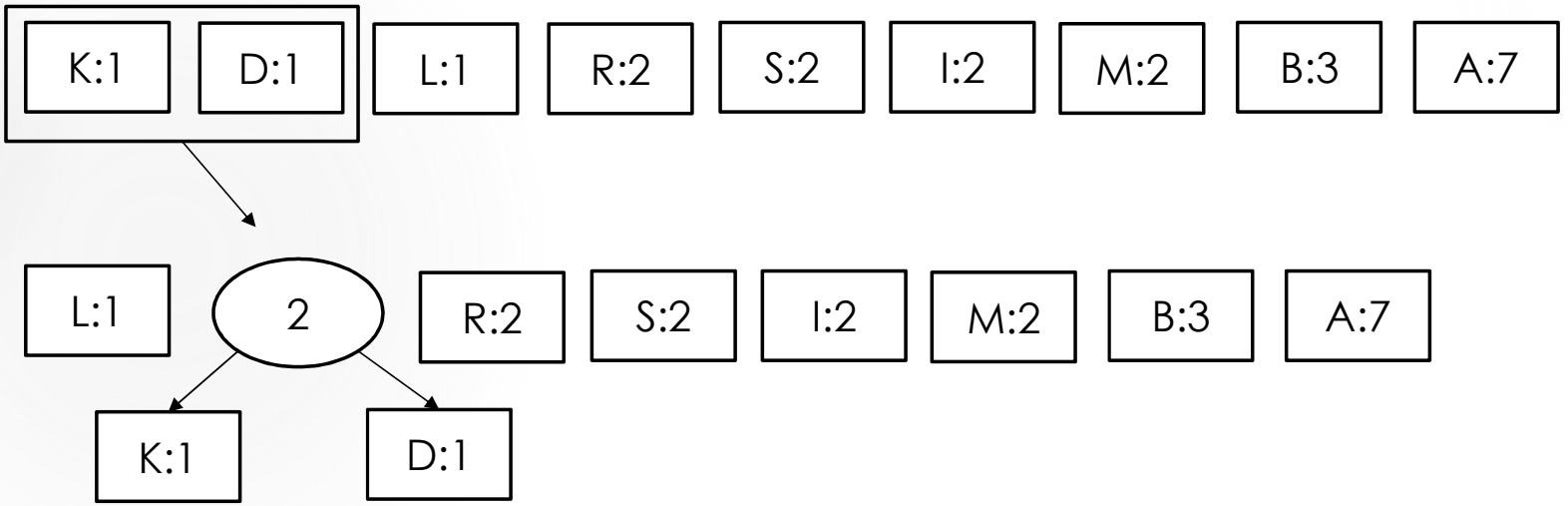
Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Erstellen Sie den Huffman-Codierungsbaum für die folgende Zeichenkette:

  ABRAKADABRASIMSALABIM

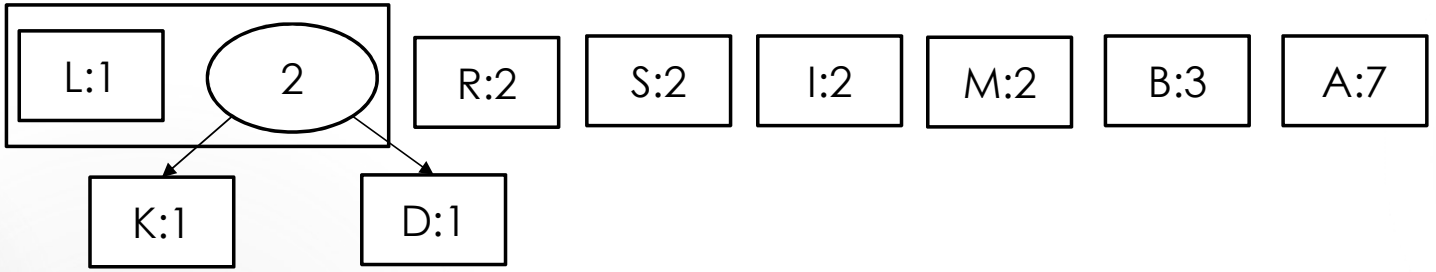
3: Blätter mit geringster Häufigkeit zusammenfassen und neu einordnen

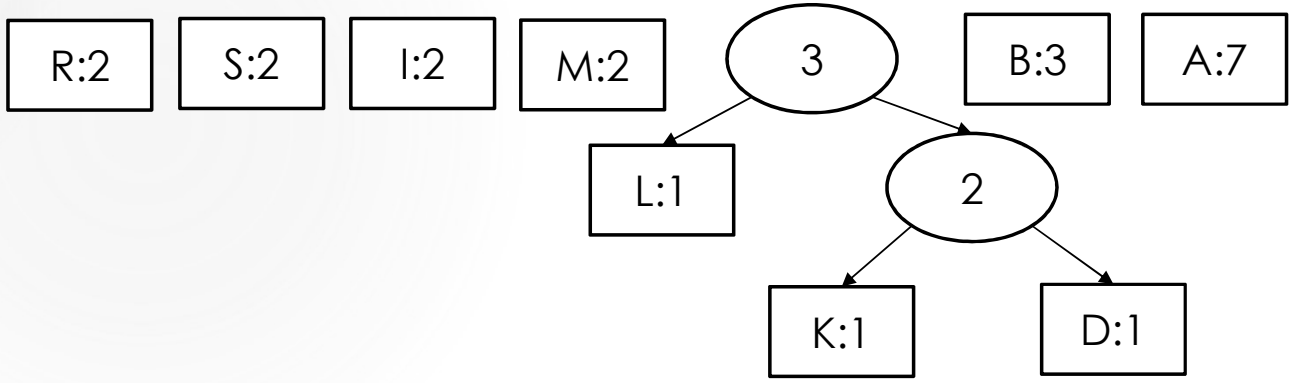


Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

3: Knoten mit geringster Häufigkeit zusammenfassen und neu einordnen

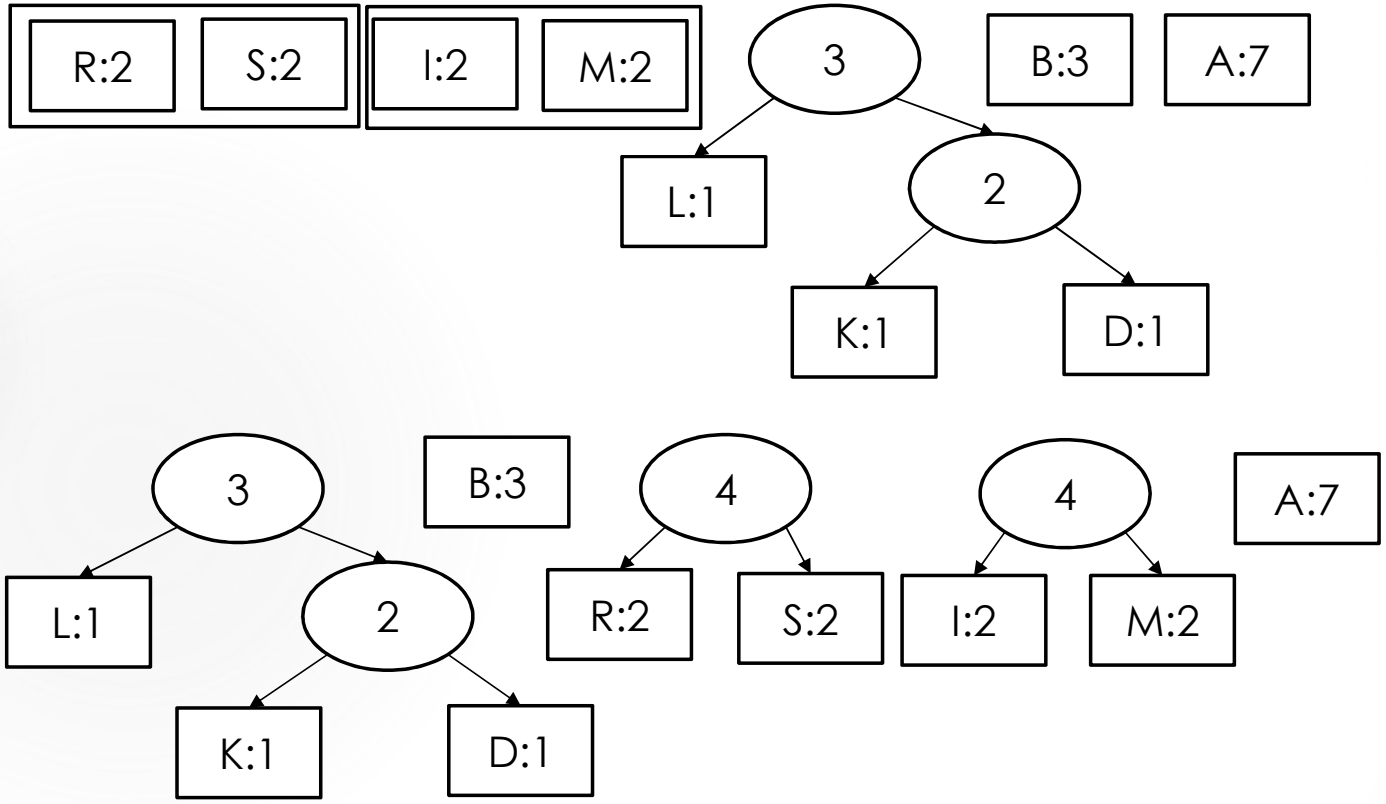




Aufgabe 5 – Huffman-Code

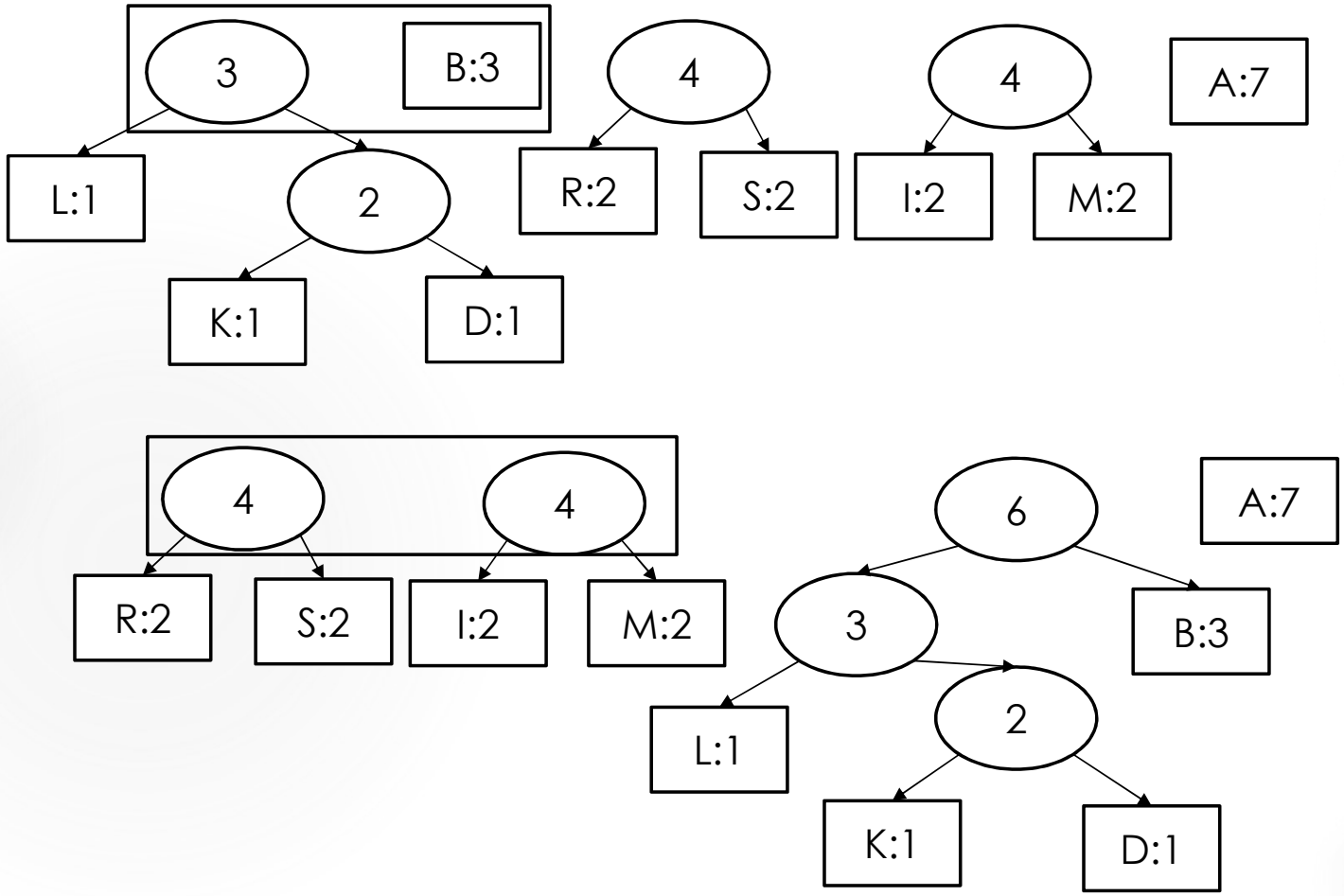
Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

3: Knoten mit geringster Häufigkeit zusammenfassen und neu einordnen



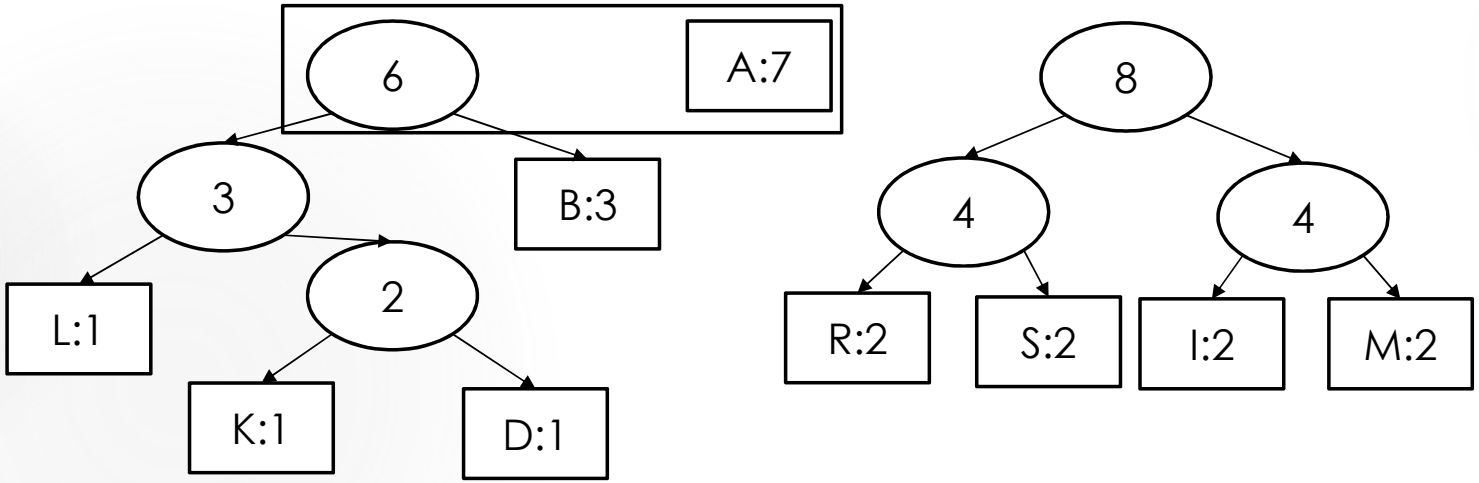
Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck



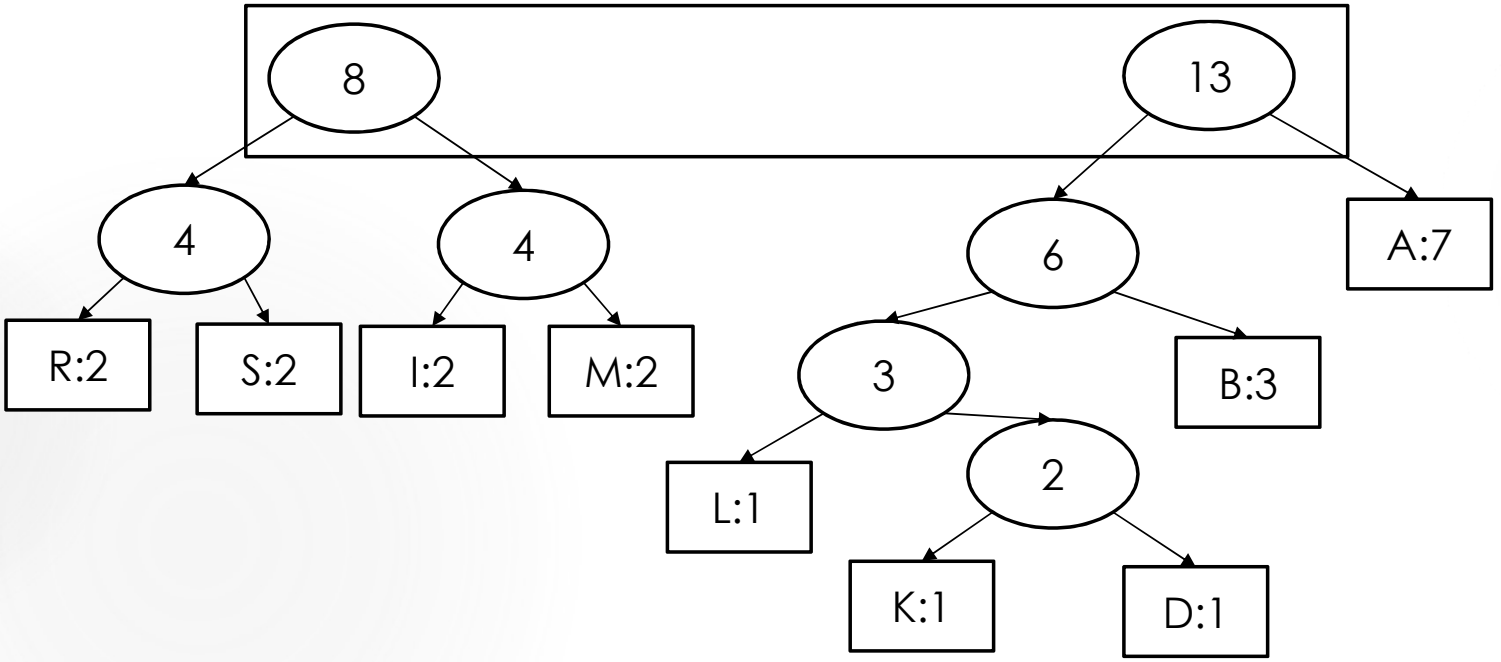
Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck



Aufgabe 5 – Huffman-Code

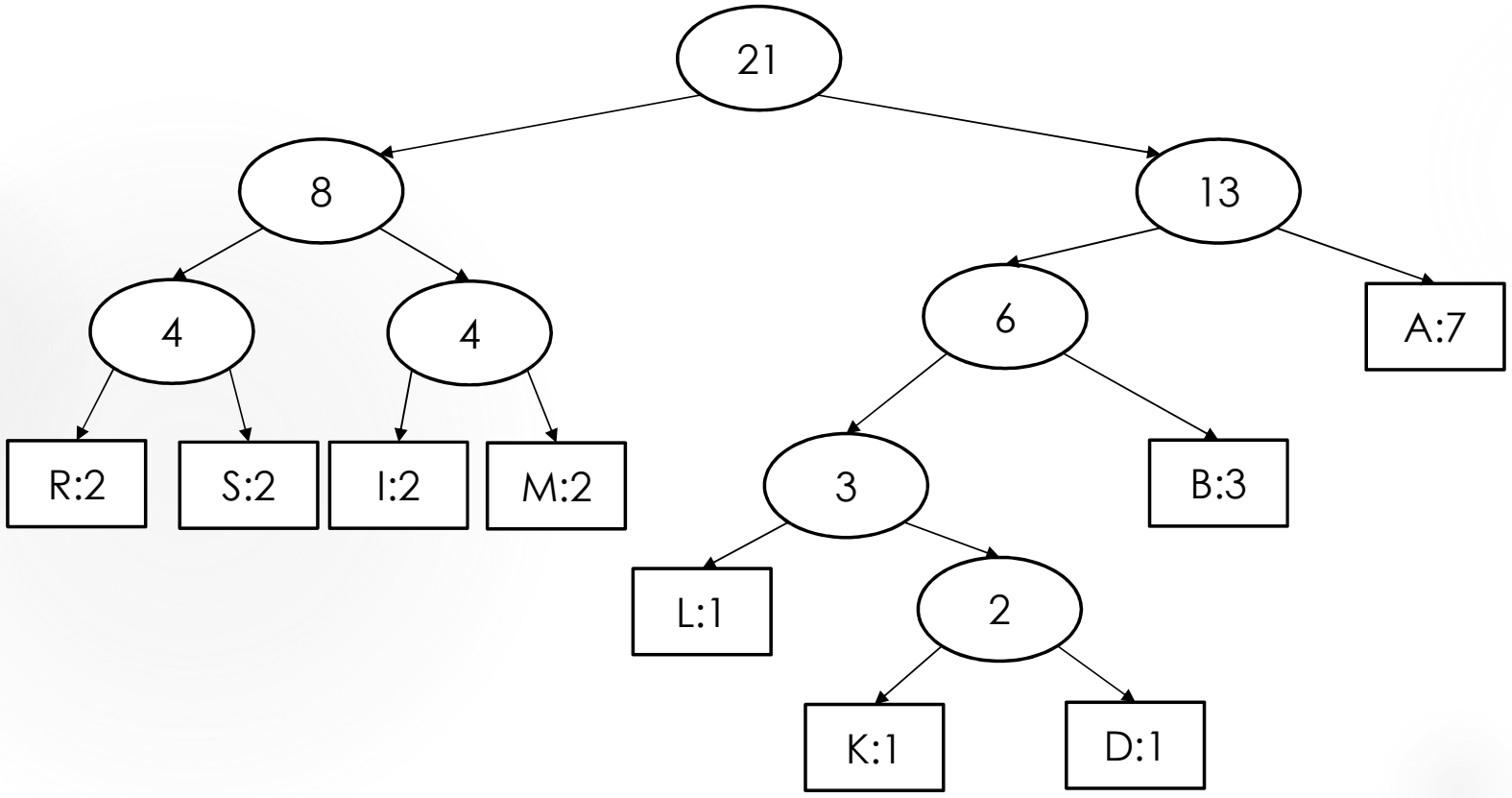
Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck



Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

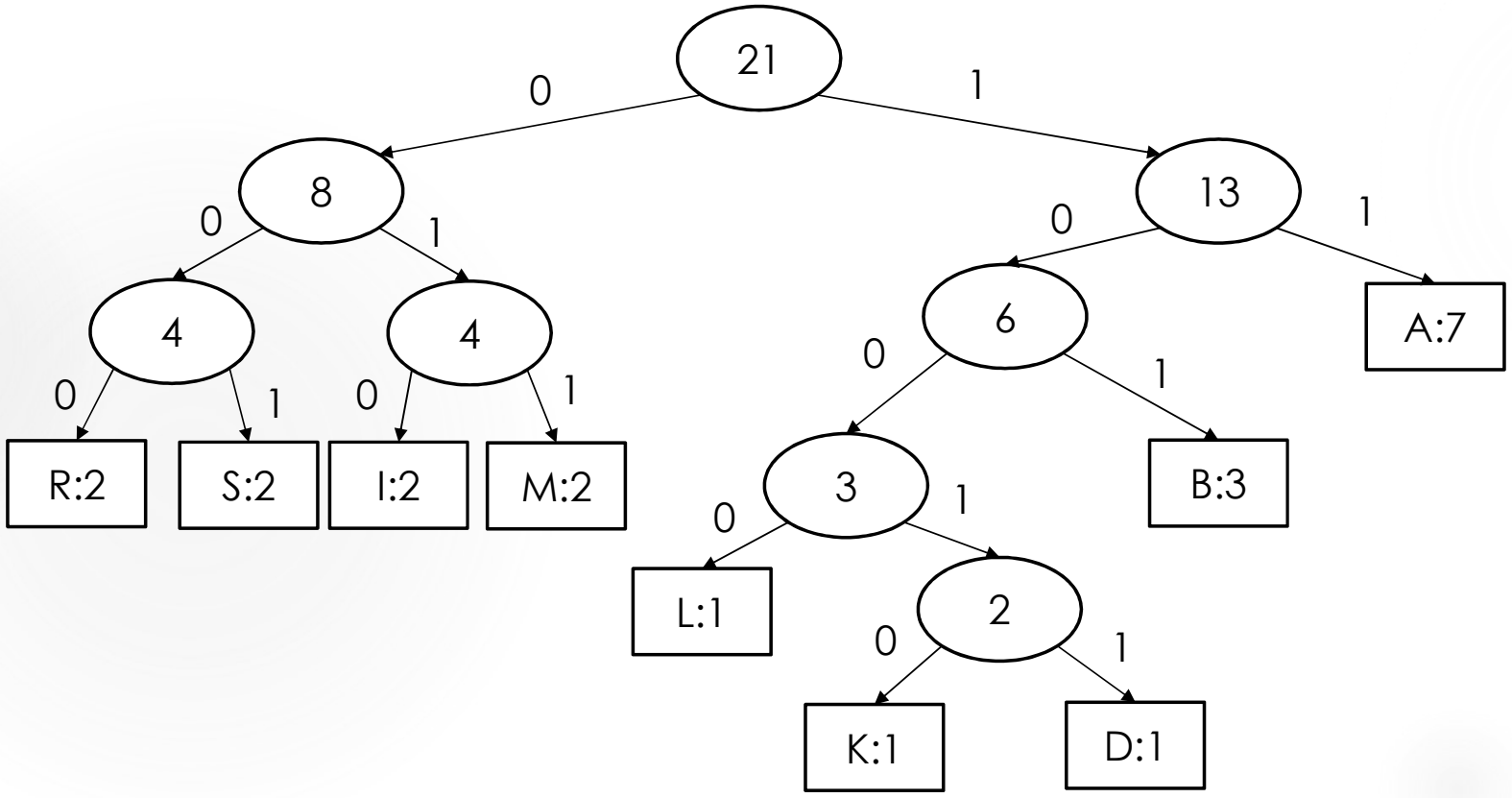
3: Fertig!



Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

4: Beim fertigen Baum an jeden linken Ast eine 0, an jeden rechten Ast eine 1 schreiben

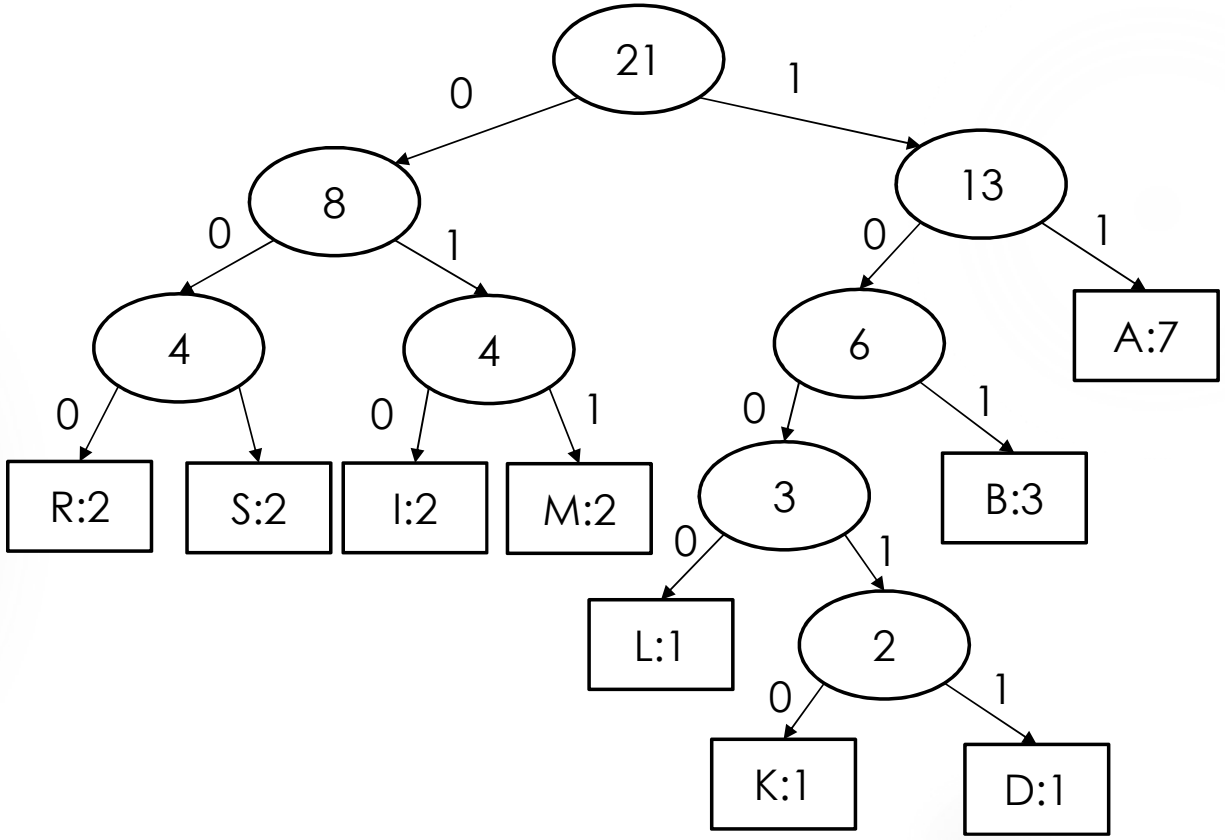


Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

5: Codewörter als Pfad des Baumes von oben nach unten auslesen

Zeichen	Code
A	11
B	101
R	000
K	10010
D	10011
S	001
I	010
M	011
L	1000



Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart? (Das Codebuch ist zu vernachlässigen)

Hinweis:
1: Summieren der Längen aller Codewörter (· Anzahl)
2: wie viele Bits benötigt man bei einer festen Codewortlänge?

Aufgabe 5 – Huffman-Code

- Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart? (Das Codebuch ist zu vernachlässigen)

1: Summieren aller bestimmten Längen

M = Länge(A) · Anzahl(A) + … + Länge(L) · Anzahl(I) = 61 bit

2: wie viele Bits bei fester Codewortlänge?

9 Zeichen = 2^{BITS} → mindestens 4 Bit pro Wort, da $2^3 = 8 < 9$

Bei 21 Zeichen macht das: 21 · 4Bit = 84 Bit

Man spart also 23 Bit. Das ist nahezu ein Viertel!

Zeichen	Code
A	11
B	101
R	000
K	10010
D	10011
S	001
I	010
M	011
L	1000

Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wieviel Bits sind minimal nötig (optimale Codierung)?
- Wieviel Prozent schlechter ist der Huffman-Code?

Hinweis:

1: Die minimale Anzahl Bits ist durch den Informationsgehalt bestimmt. Genau das ist nämlich dessen Bedeutung!

2: Das Huffman-Verfahren erlaubt nur Codierung mit Längen natürlicher Zahlen, deshalb gibt es meistens einen Verlust

3: Es gibt tatsächlich Codierungsverfahren, die auch Zwischenwerte annehmen können (→ Arithmetische Codierung)

Dazu eine Beschreibung: http://de.wikipedia.org/wiki/Arithmetisches_Kodieren

Aufgabe 5 – Huffman-Code

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

- Wieviel Bits sind minimal nötig (optimale Codierung)?

$H_e/I(x) = -ld(p) = -ld$ (Anzahl/Gesamtzeichenzahl)

Zeichen	Anzahl	Relativer Anteil	Bitlänge (I(x))
A	7	0.333	1.59
B	3	0.143	2.81
R, S, I, M	2	0.095	3.39
K, D, L	1	0.048	4.39

Min(#Bits) = Länge(A) · Anzahl(A) + … = 1.59 · 7 + 2.81 · 3 + 3.39 · 8 + 4.39 · 3 = 59.83

- Wieviel Prozent schlechter ist der Huffman-Code?

Der Huffman-Code (61 bit) ist also fast so gut, wie ein optimaler Code (59.83 bit).

P = 61/59.83 = 1.02 Der Unterschied beträgt nur etwa 2 %.

Vielen Dank für eure Aufmerksamkeit!

Friedrich-Alexander Universität Erlangen-Nürnberg
Jan Spieck

De wens is de vader van de gedachte