

## Lösungsblatt Nr. 12

### Aufgabe 41

- a) Zunächst bestimmen wir die Auftrittswahrscheinlichkeiten für die Zeichenketten  $0^n 1$  mit  $0 \leq n \leq n_{\max}$ .<sup>1</sup>

Metazeichen	Auftrittsw.	Zahlenwert
1	$1 - \beta$	0,1500
01	$\beta(1 - \beta)$	0,1275
001	$\beta^2(1 - \beta)$	0,1084
0001	$\beta^3(1 - \beta)$	0,0921
00001	$\beta^4(1 - \beta)$	0,0783
000001	$\beta^5(1 - \beta)$	0,0666
0000001	$\beta^6(1 - \beta)$	0,0566
00000001	$\beta^7(1 - \beta)$	0,0481
000000001	$\beta^8(1 - \beta)$	0,0409
0000000001	$\beta^9(1 - \beta)$	0,0347
0000000000	$\beta^{10}$	0,1969

Die Huffman-Kodierung werden wir nun auf die übliche Weise ermitteln. Wir halten uns dabei an die Konvention, daß unwahrscheinlichere Zweige mit "0", wahrscheinlichere mit "1" kodiert werden sollen.

Da die  $\beta^n(1 - \beta)$  eine geometrische Folge darstellen, sind sie (bis auf das "Restglied"  $\beta^{n_{\max}+1}$ ) bereits nach Größe sortiert:

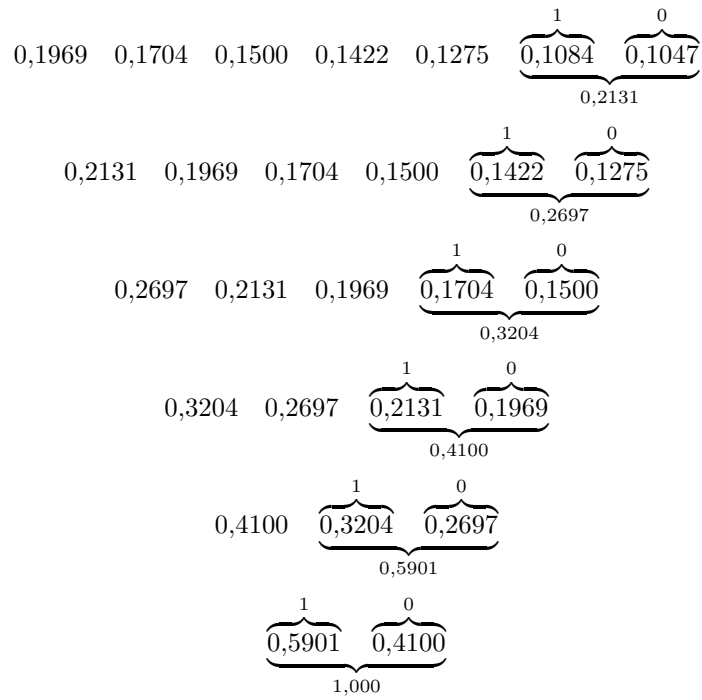
$$0,1969 \quad 0,1500 \quad 0,1275 \quad 0,1084 \quad 0,0921 \quad 0,0783 \quad 0,0666 \quad 0,0566 \quad 0,0481 \quad \underbrace{\overbrace{0,0409}^1 \quad \overbrace{0,0347}^0}_{0,0756}$$

$$0,1969 \quad 0,1500 \quad 0,1275 \quad 0,1084 \quad 0,0921 \quad 0,0783 \quad 0,0756 \quad 0,0666 \quad \underbrace{\overbrace{0,0566}^1 \quad \overbrace{0,0481}^0}_{0,1047}$$

$$0,1969 \quad 0,1500 \quad 0,1275 \quad 0,1084 \quad 0,1047 \quad 0,0921 \quad 0,0783 \quad \underbrace{\overbrace{0,0756}^1 \quad \overbrace{0,0666}^0}_{0,1422}$$

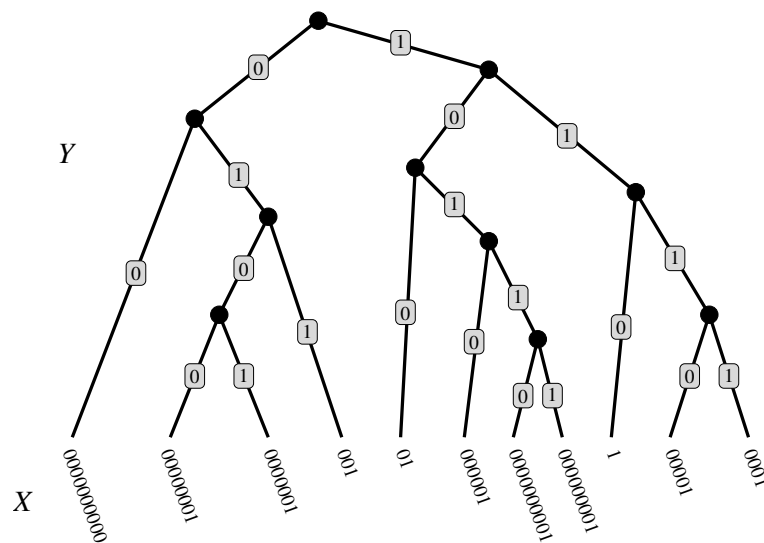
$$0,1969 \quad 0,1500 \quad 0,1422 \quad 0,1275 \quad 0,1084 \quad 0,1047 \quad \underbrace{\overbrace{0,0921}^1 \quad \overbrace{0,0783}^0}_{0,1704}$$

<sup>1</sup>wir rechnen im folgenden mit vier Nachkommastellen



Aus dieser Aufstellung lässt sich die (bzw. eine) Huffman-Kodierung ermitteln:

Eingabe $X$	Auftrittsw.	Kodierung $Y$
1	0,1500	110
01	0,1275	100
001	0,1084	011
0001	0,0921	1111
00001	0,0783	1110
000001	0,0666	1010
0000001	0,0566	0101
00000001	0,0481	0100
000000001	0,0409	10111
0000000001	0,0347	10110
0000000000	0,1969	00



- b) Die relative Redundanz  $R$  der Kodierung  $Y$  ermitteln wir bekanntlich, indem wir die durchschnittliche Kodelänge  $\tilde{H}$  mit dem Shannonschen Informationsgehalt  $H$  wie folgt verrechnen:

$$R(Y) = 1 - \frac{H(Y)}{\tilde{H}(Y)}$$

Dazu berechnen wir zunächst die Nominalinformation  $\tilde{H}$  numerisch:

$$\begin{aligned} \tilde{H}(Y) &= \sum_i p(Y=y_i) \cdot l(y_i) \\ &\approx (15\% + 12,75\% + 10,84\%) \cdot 3 \text{ bit} \\ &\quad + (9,21\% + 7,83\% + 6,66\% + 5,66\% + 4,81\%) \cdot 4 \text{ bit} \\ &\quad + (4,09\% + 3,47\%) \cdot 5 \text{ bit} + 19,69\% \cdot 2 \text{ bit} \\ &= 3,2963 \text{ bit} \end{aligned}$$

Auch die Realinformation  $H(Y)$  berechnen wir "zu Fuß":

$$\begin{aligned} H(Y) &= - \sum_i p(Y=y_i) \cdot \log p(Y=y_i) \\ &\approx H(15\%, 12,75\%, 10,84\%, 9,21\%, 7,83\%, 6,66\%, \\ &\quad 5,66\%, 4,81\%, 4,09\%, 3,47\%, 19,69\%) \\ &\approx 3,2654 \text{ bit} \end{aligned}$$

Die Redundanz beträgt daher

$$R(Y) = 1 - \frac{H(Y)}{\tilde{H}(Y)} \approx 1 - \frac{3,2654 \text{ bit}}{3,2963 \text{ bit}} \approx 0,009374,$$

d. h. nur knapp 1%.

Das relativ kleine Ausgabealphabet – es umfaßt 11 Kodewörter – komprimiert die Eingabe-Bitfolge  $X$  also schon beträchtlich, verglichen mit ihrer ursprünglichen Redundanz, die

$$\begin{aligned} R(X) &= 1 - \frac{H(X)}{\tilde{H}(X)} = 1 + \frac{\beta \cdot \log \beta + (1 - \beta) \cdot \log(1 - \beta)}{1 \text{ bit}} \\ &= 1 + \frac{85\% \cdot \log 85\% + 15\% \cdot \log 15\%}{\text{bit}} \approx 1 - \frac{0,6098 \text{ bit}}{\text{bit}} \approx 0,3902 \end{aligned}$$

beträgt, also ärgerliche 39%.

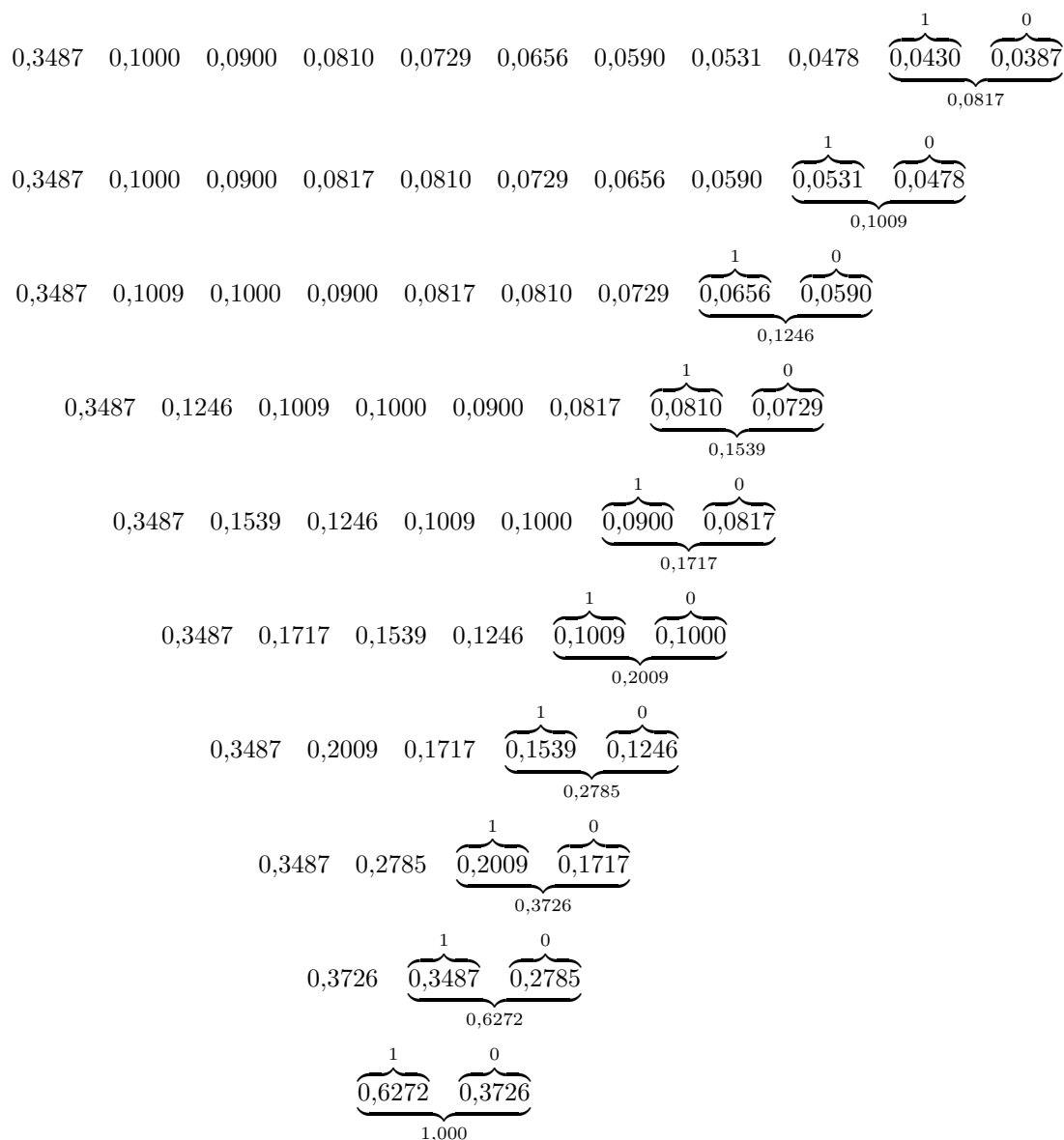
- c) Wenn sich  $\beta$  signifikant ändert, ist zu erwarten, daß sich auch die Huffman-Kodierung – d.h. die Wortlängen für die einzelnen Metazeichen – ändert.

Nun sei also  $\beta = 90\%$ . Wir gehen vor wie in (a) und (b):

Die Auftrittswahrscheinlichkeiten für die Eingabe-Zeichenketten  $0^n1$  sind (abermals mit  $n_{\max} = 9$  und einer Rechengenauigkeit von vier Nachkommastellen)

Metazeichen	Auftrittsw.	Zahlenwert
1	$1 - \beta$	0,1000
01	$\beta(1 - \beta)$	0,0900
001	$\beta^2(1 - \beta)$	0,0810
0001	$\beta^3(1 - \beta)$	0,0729
00001	$\beta^4(1 - \beta)$	0,0656
000001	$\beta^5(1 - \beta)$	0,0590
0000001	$\beta^6(1 - \beta)$	0,0531
00000001	$\beta^7(1 - \beta)$	0,0478
000000001	$\beta^8(1 - \beta)$	0,0430
0000000001	$\beta^9(1 - \beta)$	0,0387
0000000000	$\beta^{10}$	0,3487

Die Huffman-Kodierung ermitteln wir nun wie gewohnt:





Die Realinformation  $H(Y)$  beläuft sich nach Claude Shannon auf:

$$\begin{aligned}
 H(Y) &= - \sum_i p(Y = y_i) \cdot \log p(Y = y_i) \\
 &\approx H(10\%, 9\%, 8,10\%, 7,29\%, 6,56\%, 5,90\%, \\
 &\quad 5,31\%, 4,78\%, 4,30\%, 3,87\%, 34,87\%) \\
 &\approx 3,0540 \text{ bit}
 \end{aligned}$$

Die Redundanz beträgt daher

$$R(Y) = 1 - \frac{H(Y)}{\tilde{H}(Y)} \approx 1 - \frac{3,0540 \text{ bit}}{3,1118 \text{ bit}} \approx 0,0186,$$

d. h. knapp 2%. (Dieser Wert ist vor allem dadurch höher als bei Teil (a), daß der häufiger gewordene Fall, daß mehr als  $n_{\max}$  Nullen bei der Eingabe hintereinander auftauchen, für unser größeres  $\beta = 90\%$  mit der erhöhten Wahrscheinlichkeit von 35% vorkommt; diese 35% sind aber leider nicht nahe einer (ganzzahligen) Zweierpotenz wie  $1/2$  oder  $1/4$  gelegen. Vgl. Aufgabe 35!)

Dieses relativ kleine Ausgabealphabet komprimiert die Eingabe-Bitfolge  $X$  also – wie in Teil (a) – schon beträchtlich, verglichen mit ihrer ursprünglichen Redundanz, die

$$\begin{aligned}
 R(X) &= 1 - \frac{H(X)}{\tilde{H}(X)} = 1 + \frac{90\% \cdot \log 90\% + 10\% \cdot \log 10\%}{\text{bit}} \\
 &\approx 1 - \frac{0,4690 \text{ bit}}{\text{bit}} \approx 0,5310
 \end{aligned}$$

beträgt, also stolze 53%!

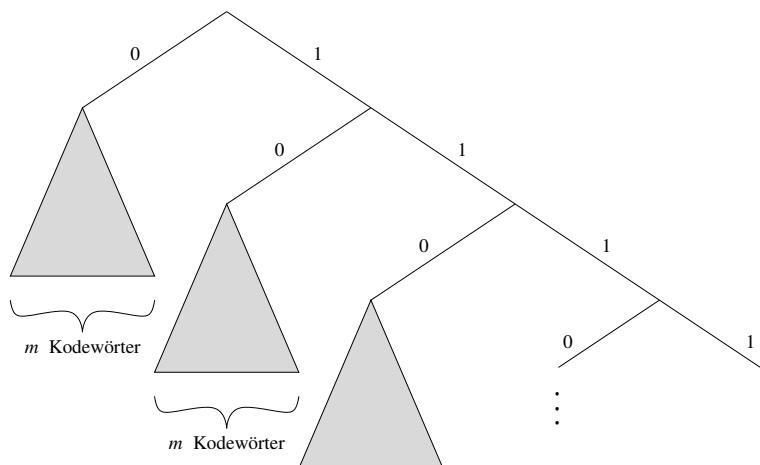
- d) Läßt man (für festes  $\beta$ )  $n_{\max} \rightarrow \infty$  gehen, so ergibt sich ein interessantes Bild: Eine Huffman-Kodierung läßt sich dann zwar nicht mehr im strengen Sinne anwenden, da ja immer die beiden Alternativen mit den niedrigsten Wahrscheinlichkeiten gegeneinander kodiert werden sollen, wir es aber mit unendlich vielen – und beliebig unwahrscheinlichen – Ereignissen  $0^n 1$  zu tun haben (für  $n \leq n_{\max} \rightarrow \infty$ ).

Man behilft sich hier mit einem Trick: sei

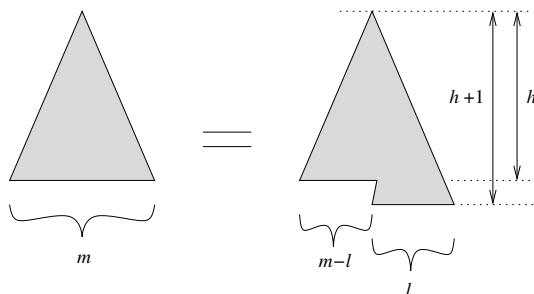
$$m := \left\lceil \log_{\beta} \frac{1}{2} \right\rceil$$

Es ist also das Ereignis  $0^{n+m} 1$  nur noch halb so wahrscheinlich wie  $0^n 1$  (für alle  $n \in \mathbb{N}$ ). Dies bedeutet aber im Kontext von Huffmans Methode, daß das Kodewort für  $0^{n+m} 1$  um ein Bit länger sein muß als das für  $0^n 1$ .

Die Struktur des (unendlichen) Kodierungsbaumes wird daher so aussehen wie in der folgenden Abbildung; die Länge der Kodewörter steige von links nach rechts monoton.



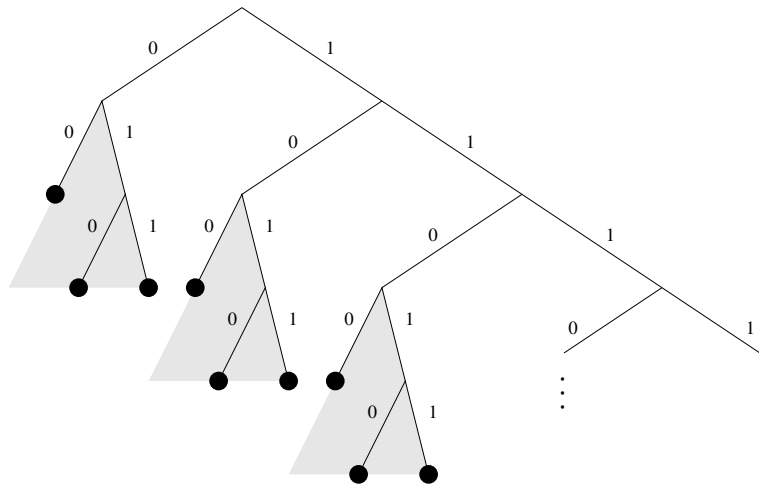
Die Blätter (Kodewörter-Suffixe) der grau schraffiert dargestellten identischen Unterbäume werden allerdings i. d. R. nicht alle auf derselben Höhe (von derselben Länge) sein, wenn nicht  $m$  zufällig eine Zweierpotenz ist. Sei  $h := \lfloor \log_2 m \rfloor$ . Dann sind auf der Höhe  $h + 1$  die “überzähligen” Blätter anzutreffen, es sind derer genau  $l := 2 \cdot (m - 2^h)$ . Alle anderen  $m - l$  Blätter des Unterbaumes<sup>2</sup> sind noch auf der Höhe  $h$ .



Damit läßt sich feststellen: es gibt für jede Kodelänge  $m$  Kodewörter; nur der ersten (kürzesten) Kodewörter sind es bloß  $m - l$  Stück.

Für jedes  $m \in \mathbb{N}$  ergibt sich so ein anderer, Huffman-ähnlicher Kode. Er hat unendlich viele Kodewörter, aber die Wahrscheinlichkeit für längere Kodewörter fällt ja auch exponentiell. Zu  $m = 3$  ( $\Leftrightarrow \beta \approx 0,794$ ) läßt sich z. B. der folgende Kode aufbauen:<sup>3</sup>

<sup>2</sup>es ist ja  $2^h = (m - l) + l/2$ ; außerdem ist stets  $l \in \{0, \dots, 2^{h+1} - 2\}$  und  $m - l \in \{1, \dots, 2^h\}$   
<sup>3</sup>wie man leicht sieht, ist hier  $h = \lfloor \log_2 3 \rfloor = 1$  und  $l = 2 \cdot (3 - 2^1) = 2$  und  $m - l = 3 - 2 = 1$

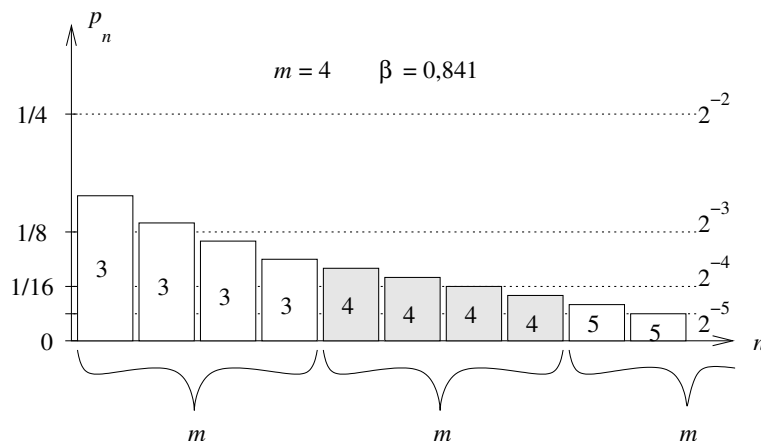
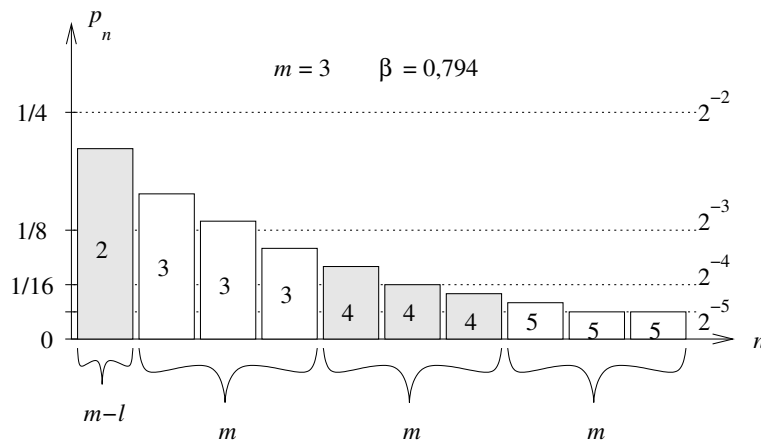


$X$	$Y$	Präfix von $Y$	$Y$ -Kodelänge
1	00	0	2
01	010	0	3
001	011	0	3
0001	100	10	3
00001	1010	10	4
000001	1011	10	4
0000001	1100	110	4
00000001	11010	110	5
000000001	11011	110	5
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Die beiden folgenden Graphen zeigen, wie Kodewort-Längen (in den Balken der Diagramme angegeben), Auftretswahrscheinlichkeiten  $p_n = P(X = 0^n1)$  und Perioden  $m$  für die Beispiele  $m \in \{3, 4\}$  zusammenhängen. Je größer der Anfangswert  $P(X = 1) = 1 - \beta$  ist, desto schneller fallen die Werte für  $p_n$  ab.<sup>4</sup>

Deutlich ist zu sehen, daß die Kodewort-Längen  $k$  in Zusammenhang stehen mit Auftretswahrscheinlichkeiten  $p_n \approx 2^{-k}$ :

<sup>4</sup>Denn die (unendliche) geometrische Reihe muß ja  $\sum_n p_n = 1$  werden!



Die so entstehenden Codes sind nach ihrem Erfinder Salomon Golomb benannt, der sie 1966 eingeführt hat.<sup>5</sup>

### Aufgabe 42

**Alternative 1:** (Für eine genauere Herleitung der allgemeineren Sachverhalte vgl. Aufgabe 43)

- a) Sofort sieht man, daß im Produkt von  $f_1$  und  $f_2$  die höchste vorkommende Potenz  $x^3$  sein muß, d. h. das Ergebnispolynom wird mit vier Koeffizienten darstellbar sein.

Es genügen uns daher  $n = 4$  Stützstellen für die Schnelle Fouriertransformation.

Das FFT-Schema selbst wird  $\log_2 n = 2$  Stufen haben.

Die  $n = 4$ -te primitive Einheitswurzel von Eins ist  $\omega = i$ ; anschaulich wird dies auch anhand einer kleinen Skizze der Gaußschen Zahlenebene ( $i$  entspricht dem Winkel  $90^\circ$ , also einem Viertel des Einheitskreises).

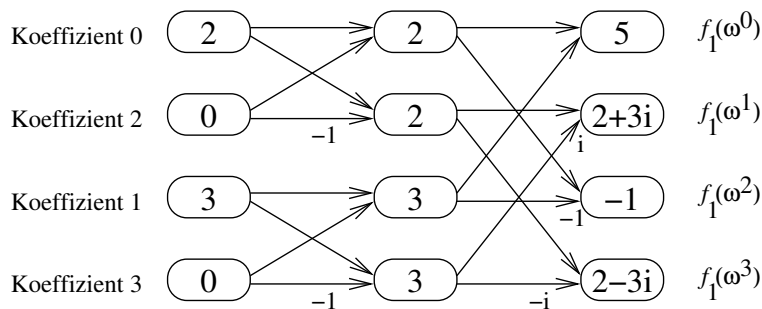
Dann sind die Potenzen von  $\omega$  die vier Werte  $1, i, -1$  und  $-i$ .

- b) Die Koeffizienten des ersten Polynoms  $f_1(x) = 3x + 2$  sind offenbar  $2, 3, 0$  und  $0$ , aufsteigend geordnet. Diese müssen dem zweistufigen FFT-Schema *in der*

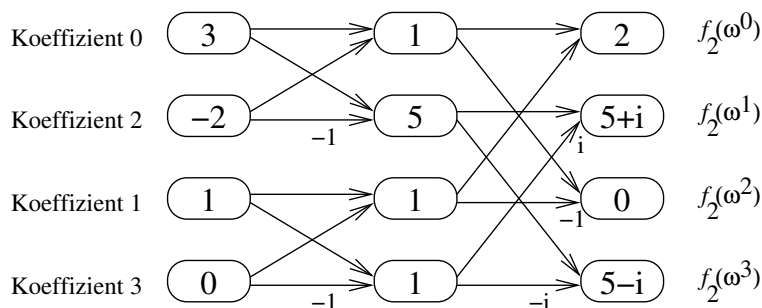
<sup>5</sup>Die Spezialfälle mit  $l = 0$  werden auch als Rice-Codes bezeichnet (1979).

korrekten Permutation (0,2,1,3) als Eingabe zur Verfügung stehen, also als 2, 0, 3, 0.

Im folgenden Diagramm sind die Gewichte – der Übersicht halber – nur angegeben, wenn sie  $\neq 1$  sind:



c) Analog zu (b) erhalten wir für das zweite Polynom  $f_2(x) = -2x^2 + x + 3$ , das also die Koeffizienten 3, 1,  $-2$  und 0 hat:



d) Wir multiplizieren<sup>6</sup> die beiden Polynome in Stützstellendarstellung:

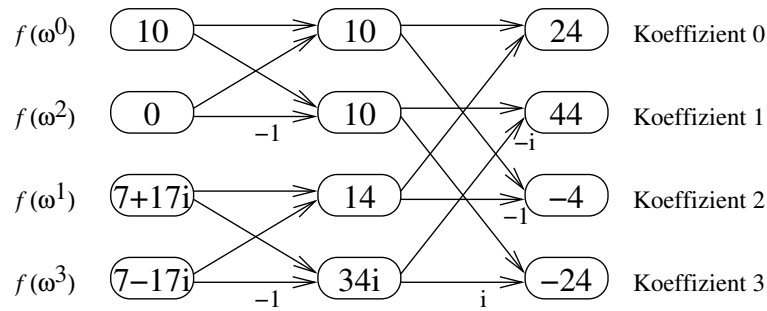
Polynom	Wert bei $\omega^0$	Wert bei $\omega^1$	Wert bei $\omega^2$	Wert bei $\omega^3$
$f_1$	5	$2 + 3i$	$-1$	$2 - 3i$
$f_2$	2	$5 + i$	0	$5 - i$
$f := f_1 \cdot f_2$	10	$7 + 17i$	0	$7 - 17i$

Um nun zur Koeffizientendarstellung zurückkehren zu können, nehmen wir diese Ergebniswerte an den vier Stützstellen  $\omega^i$  als Eingabe für die inverse FFT. Die  $\text{FFT}^{-1}$  (Rücktransformation) unterscheidet sich von der vorwärts-Variante der Schnellen Fouriertransformation nur dadurch,

- daß statt  $\omega$  nun  $1/\omega = \omega^*$  als Einheitswurzel verwendet wird (was sich in den veränderten Gewichten im Schema niederschlägt), und
- daß das Endergebnis schließlich mit dem Faktor  $1/n$  zu normieren ist.

Die Eingaben müssen abermals in der korrekten, immergleichen Permutation (0,2,1,3) anliegen:

<sup>6</sup>Ausführlich ist  $(2 + 3i)(5 + i) = 10 + 15i + 2i + 3i^2 = 10 + 15i + 2i - 3 = 7 + 17i$  und  $(2 - 3i)(5 - i) = 10 - 15i - 2i + 3i^2 = 10 - 15i - 2i - 3 = 7 - 17i$ .



Das Endergebnis ist also nach Normierung: **6, 11, -1, -6** bzw. das Polynom

$$[f_1 \cdot f_2](x) = -6x^3 - x^2 + 11x + 6$$

e) Auch ohne FFT erhält man als Produkt

$$\begin{aligned} f_1(x) \cdot f_2(x) &= (3x + 2)(-2x^2 + x + 3) \\ &= (3 \cdot (-2))x^3 + (3 \cdot 1 + 2 \cdot (-2))x^2 + (3 \cdot 3 + 2 \cdot 1)x + (2 \cdot 3) \\ &= -6x^3 - x^2 + 11x + 6 = [f_1 \cdot f_2](x) \end{aligned}$$

Die Probe ergibt somit, daß in Teil (b)–(d) richtig gerechnet wurde.

### Alternative 2:

a) Mithilfe aller  $n = 10$  zyklischen Verschiebungen der Zeichenkette  $w$  erhalten wir die Matrix

$$M = \begin{bmatrix} S & U & P & E & R & D & U & P & E & R \\ U & P & E & R & D & U & P & E & R & S \\ P & E & R & D & U & P & E & R & S & U \\ E & R & D & U & P & E & R & S & U & P \\ R & D & U & P & E & R & S & U & P & E \\ D & U & P & E & R & S & U & P & E & R \\ U & P & E & R & S & U & P & E & R & D \\ P & E & R & S & U & P & E & R & D & U \\ E & R & S & U & P & E & R & D & U & P \\ R & S & U & P & E & R & D & U & P & E \end{bmatrix} \begin{matrix} 8 \\ 9 \\ 4 \\ 2 \\ 6 \\ 1 \\ 10 \\ 5 \\ 3 \\ 7 \end{matrix}$$

Zeilenweise sortiert erhalten wir:

$$M' = \begin{bmatrix} D & U & P & E & R & S & U & P & E & R \\ E & R & D & U & P & E & R & S & U & P \\ E & R & S & U & P & E & R & D & U & P \\ P & E & R & D & U & P & E & R & S & U \\ P & E & R & S & U & P & E & R & D & U \\ R & D & U & P & E & R & S & U & P & E \\ R & S & U & P & E & R & D & U & P & E \\ S & U & P & E & R & D & U & P & E & R \\ U & P & E & R & D & U & P & E & R & S \\ U & P & E & R & S & U & P & E & R & D \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{matrix}$$

Die Ausgabe  $(v, k)$  besteht nun in der letzten Spalte  $L$  dieser Matrix  $M'$  und dem Zeilenindex  $k$  der Originalzeichenkette  $w$  in  $M'$ :

$$v := L = R P P U U E E R S D \quad k = 8$$

- b) Die **Laufängnenkodierung** – nach dem Schema (Zeichen,Länge) – von  $v$  sieht wie folgt aus:

$$(R, 1), (P, 2), (U, 2), (E, 2), (R, 1), (S, 1), (D, 1)$$

bzw. unter Benutzung des Kodes aus der Aufgabenstellung:

$$(3, 1), (2, 2), (5, 2), (1, 2), (3, 1), (4, 1), (0, 1)$$

Die **move-to-front-Kodierung** von  $v$  gestaltet sich folgendermaßen:

Eingabe	Ausgabe	Kode bisher:	0	1	2	3	4	5
R	3		D	E	P	R	S	U
P	3		R	D	E	P		
P	0		P	R	D	E		
U	5							
U	0		U	P	R	D	E	S
E	4							
E	0		E	U	P	R	D	
R	3							
S	5		R	E	U	P		
D	5		S	R	E	U	P	D

Ausgabe ist also die Folge 3, 3, 0, 5, 0, 4, 0, 3, 5, 5.

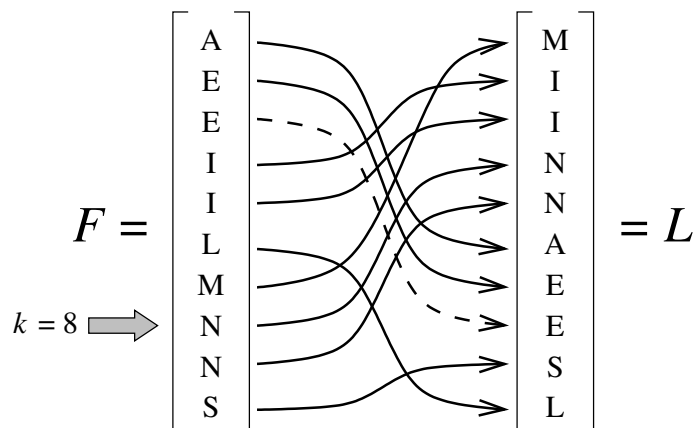
- c) Mit der Zeichenkette  $v$  ist die letzte Spalte  $L$  der (unbekannten) Matrix  $M'$  gegeben:

$$L = \left[ \begin{array}{cccccccccc} M & I & I & N & N & A & E & E & S & L \end{array} \right]^T$$

Die erste Spalte  $F$  von  $M'$  erhalten wir, indem die  $n = 10$  Zeichen von  $v$  sortiert werden:<sup>7</sup>

$$F = \left[ \begin{array}{cccccccccc} A & E & E & I & I & L & M & N & N & S \end{array} \right]^T$$

Mit dem gegebenen Startindex  $k = 8$  beginnen wir nun die Dekodierung in der (nur durch  $F$  und  $L$  gegebenen) Matrix  $M'$ :



Dabei haben wir in der Spalte  $F$  nacheinander folgende Zeichen besucht:

$$N, I, E, M, A, L, S, N, I, E$$

Also lautet die dekodierte Zeichenkette:

$$w = N I E M A L S N I E$$

<sup>7</sup>denn dadurch, daß  $M'$  zeilenweise sortiert ist, wissen wir ja insbesondere, daß die ersten Zeichen jeder Zeile in sortierter Reihenfolge vorliegen

### Aufgabe 43

**Allgemeines zum Verständnis.** Wie im Skript beschrieben, erhalten wir gemäß der “grundlegenden Formel”

$$f(\omega) = f_g(\omega^2) + \omega \cdot f_u(\omega^2)$$

die folgenden Rechenvorschriften für die Zahlenwerte  $r_i := f(\omega^i)$ :

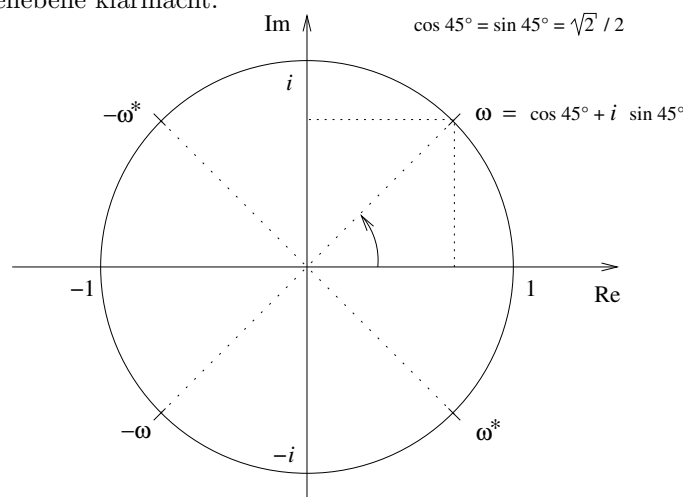
$$r_0 = g_0 + \omega^0 u_0 \qquad r_4 = g_0 + \omega^4 u_0$$

$$r_1 = g_2 + \omega^1 u_2 \qquad r_5 = g_2 + \omega^5 u_2$$

$$r_2 = g_4 + \omega^2 u_4 \qquad r_6 = g_4 + \omega^6 u_4$$

$$r_3 = g_6 + \omega^3 u_6 \qquad r_7 = g_6 + \omega^7 u_6$$

Dieses Vorgehen entspricht der **dritten (also letzten) Stufe** des Schemas für  $n = 8 = 2^3$ . Die Potenzen von  $\omega = \sqrt{2}/2 + i \cdot \sqrt{2}/2$  sind  $(\omega^i)_{i=0,\dots,7} = (1, \omega, i, -\omega^*, -1, -\omega, -i, \omega^*)$ , was man sich leicht anhand einer Skizze der Gaußschen Zahlenebene klarmacht:



In dieser dritten Stufe sind noch alle 4 Basisschemata des Butterfly-Algorithmus verschieden: wir haben

$$\begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a+b \\ a-b \end{bmatrix} \qquad \begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a+\omega b \\ a-\omega b \end{bmatrix} \qquad \begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a+ib \\ a-ib \end{bmatrix} \qquad \begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a-\omega^* b \\ a+\omega^* b \end{bmatrix}$$

Die Werte  $g_i$  und  $u_i$  lassen sich in einer **vorletzten (also zweiten) Stufe** analog erhalten als:

$$g_0 = gg_0 + (\omega^2)^0 ug_0 \qquad u_0 = gu_0 + (\omega^2)^4 uu_0$$

$$g_2 = gg_4 + (\omega^2)^1 ug_4 \qquad u_2 = gu_4 + (\omega^2)^5 uu_4$$

$$g_4 = gg_0 + (\omega^2)^2 ug_0 \qquad u_4 = gu_0 + (\omega^2)^6 uu_0$$

$$g_6 = gg_4 + (\omega^2)^3 ug_4 \qquad u_6 = gu_4 + (\omega^2)^7 uu_4$$

Die  $(\omega^2)^i$  vereinfachen sich hierbei<sup>8</sup> zu eigentlich nur vier verschiedenen Werten  $(\omega^{2i})_{i=0,\dots,7} = (\omega^0, \omega^2, \omega^4, \omega^6, \omega^8, \omega^{10}, \omega^{12}, \omega^{14}) = (1, i, -1, -i, 1, i, -1, -i)$ . Daher

<sup>8</sup>gemäß der zyklischen Gruppeneigenschaft beim Potenzieren von Einheitswurzeln

sind jeweils zwei der 4 Basisschemata auf dieser zweiten FFT-Stufe identisch; wir haben die Varianten

$$\begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a+b \\ a-b \end{bmatrix} \quad \text{und} \quad \begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a+ib \\ a-ib \end{bmatrix}$$

In der **ersten Stufe** des FFT-Algorithmus werden die Zwischenergebnisse  $gg_i, ug_i, gu_i, uu_i$  ermittelt:

$$\begin{aligned} gg_0 &= ggg + (\omega^4)^0 ugg & ug_0 &= gug + (\omega^4)^4 uug \\ gg_4 &= ggg + (\omega^4)^1 ugg & ug_4 &= gug + (\omega^4)^5 uug \\ gu_0 &= ggu + (\omega^4)^2 ugu & uu_0 &= guu + (\omega^4)^6 uuu \\ gu_4 &= ggu + (\omega^4)^3 ugu & uu_4 &= guu + (\omega^4)^7 uuu \end{aligned}$$

Die  $(\omega^4)^i$  sind hierbei nur noch Potenzen von  $-1$ ; es ist nämlich  $(\omega^4)^{i=0,\dots,7} = (\omega^0, \omega^4, \omega^8, \omega^{12}, \omega^{16}, \omega^{20}, \omega^{24}, \omega^{28}) = (1, -1, 1, -1, 1, -1, 1, -1)$ . Dies ist die Ursache dafür, daß alle 4 Basisschemata des Butterfly-Algorithmus auf dieser (ersten) Stufe gleich sind, also nur eine Variante vorkommt:

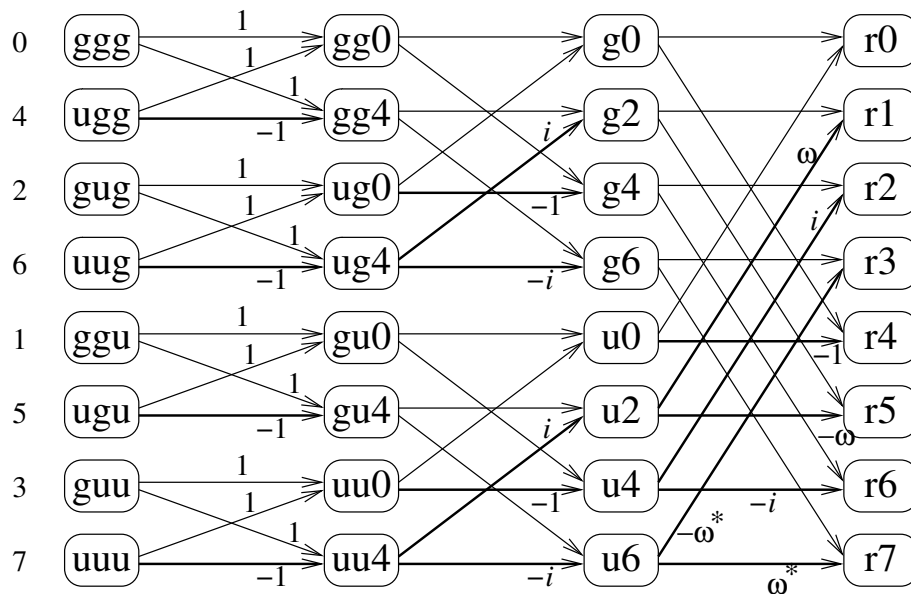
$$\begin{bmatrix} a \\ b \end{bmatrix} \rightarrow \begin{bmatrix} a+b \\ a-b \end{bmatrix}$$

(Quasi die komplementären Operationen “Mittelung” und “Differenzierung”)

Die Werte  $ggg, ugg, gug, \dots, uuu$  sind dabei die acht Koeffizienten des gegebenen Polynoms, die ja die Eingabewerte der FFT sind.

Aus selbsttätiger Hirnzermarterung<sup>9</sup> geht hervor, daß die Koeffizienten an der Eingabe in der Reihenfolge bzw. Permutation  $(a_0, a_4, a_2, a_6, a_1, a_5, a_3, a_7) =: (ggg, ugg, gug, uug, ggu, ugu, guu, uuu)$  vorliegen müssen.

Schließlich erhält man aus diesen Überlegungen (bzw. aus der Skizze im Skript) das Gesamtschema:<sup>10</sup>



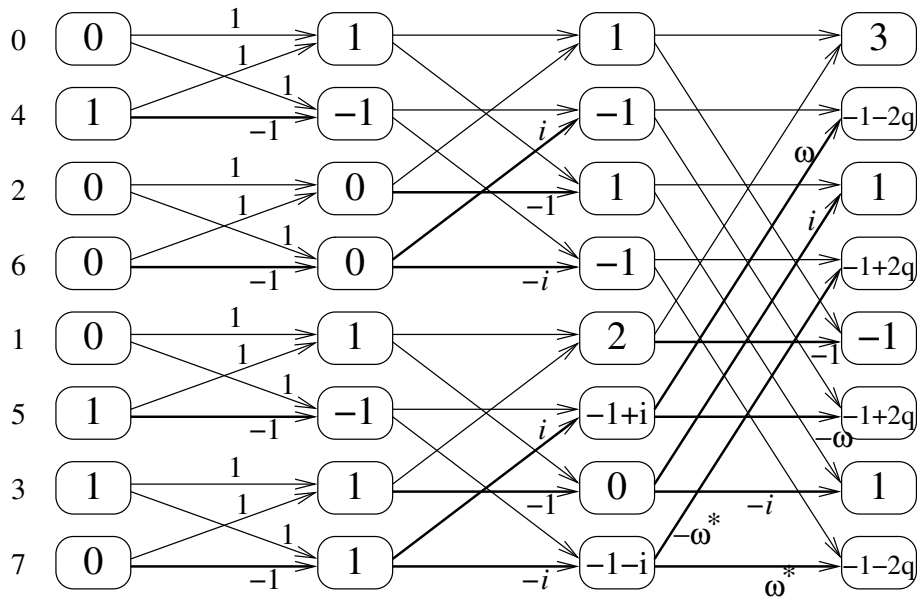
<sup>9</sup>oder aber einem Blick ins Skriptum: Kapitel 5, Seite 21

<sup>10</sup>Das kaskadierte Butterfly-Schema der FFT ist übrigens ein Paradebeispiel für ein nützliches Datenfluß-Diagramm. Man kann daraus direkt ablesen, daß die übereinander stehenden Werte parallel verarbeitet werden dürfen, weil zwischen ihnen keine Abhängigkeiten bestehen. Die horizontale Achse kann dann als Zeitachse der Verarbeitung interpretiert werden; der optimierte Aufwand  $\Theta(n \log n)$  der FFT (ohne FFT:  $\Theta(n^2)$ ) ist so nochmal zu verringern, indem Parallelverarbeitung der  $n$  Werte stattfindet: es bleiben bloß  $\Theta(\log n)$  Zeitschritte!

Hier sind (aus Übersichtsgründen) alle Gewichte, die nicht explizit angegeben sind, = 1. Explizit angegebene Gewichte  $\neq 1$  sind den fett gezeichneten Kanten zugeordnet.

- a) Mit der korrekten Reihenfolge der gegebenen  $a_i$  am Eingang der dreistufigen Struktur ergibt sich als Inhalt der FFT-Einträge:

( $q$  ist hierbei Abkürzung für den irrationalen Wert  $\cos 45^\circ = \sin 45^\circ = \sqrt{2}/2 = \sqrt{1/2} \approx 0,707$ )



Wir haben also die Werte

$$r_0 = f(\omega^0) = 3 \quad r_4 = f(\omega^4) = -1$$

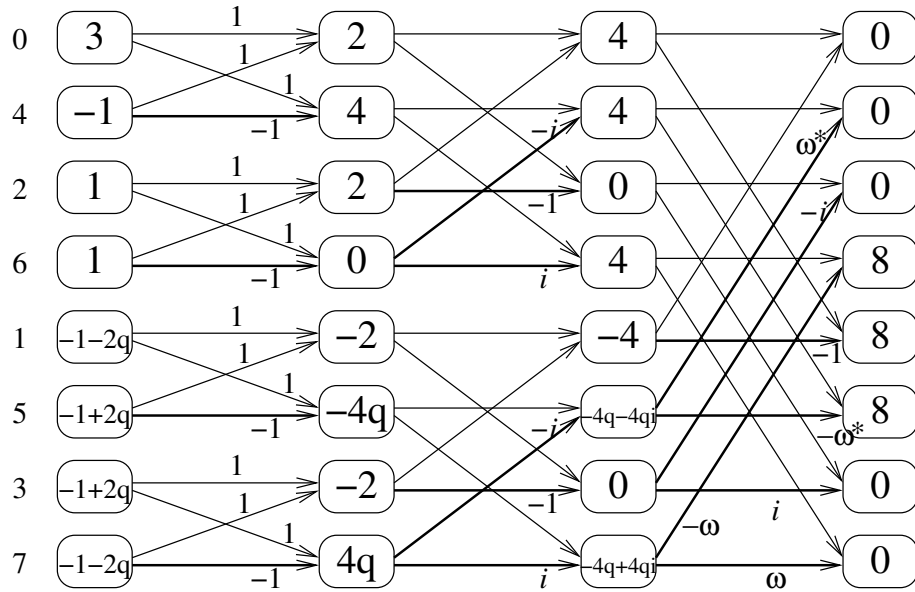
$$r_1 = f(\omega^1) = -1 - \sqrt{2} \quad r_5 = f(\omega^5) = -1 + \sqrt{2}$$

$$r_2 = f(\omega^2) = 1 \quad r_6 = f(\omega^6) = 1$$

$$r_3 = f(\omega^3) = -1 + \sqrt{2} \quad r_7 = f(\omega^7) = -1 - \sqrt{2}$$

erhalten. ( $q = \sqrt{2}/2$ )

- b) Analog zu Teil (a) wenden wir dasselbe Schema für  $\bar{\omega} = \omega^{-1}$  an. An den Eingängen müssen die  $r_i = f(\omega^i)$  abermals in der gewohnten Reihenfolge permutiert werden:



Die Gewichte sind in der Skizze schon für  $\bar{\omega}$  korrigiert:

$$\begin{aligned} \bar{\omega} &= \omega^* & -\bar{\omega} &= -\omega^* & i &\leftrightarrow -i \\ \bar{\omega}^* &= \omega & -\bar{\omega}^* &= -\omega & -i &\leftrightarrow i \end{aligned}$$

Man sieht, daß der Ausgabevektor  $(0, 0, 0, 8, 8, 8, 0, 0)$  das 8-fache der Eingabe  $(a_0, a_1, \dots, a_7)$  ist. Durch den Normierungsfaktor  $1/n = 1/2^3$  (was dem Normierungsfaktor  $1/2$  pro Stufe entspricht) wird die Identität wiedergewonnen.

Was zu zeigen war.

- c) Wenn man die Gewichte inspiziert, die in Teil (b) zu  $a_0$  bzw. dem  $n$ -fachen von  $a_0$  führen, so sind (inklusive der Normierung) alle  $n$  Eingabewerte  $r_i = f(\omega^i)$  mit demselben Gewicht  $1/n$  in diesen Koeffizienten  $a_0$  eingegangen.

Zum selben Ergebnis kommt man auch, wenn man durch rekursive Anwendung der “grundlegenden Formel” analysiert, wie die Schnelle Fouriertransformation  $a_0$  eigentlich berechnet:<sup>11</sup>

$$\begin{aligned} a_0 &= r_0 = g_0 + \omega^0 u_0 \\ &= [gg_0 + (\omega^2)^0 u g_0] + \omega^0 [g u_0 + (\omega^2)^4 u u_0] \\ &= \left( [ggg + (\omega^4)^0 u g g] + (\omega^2)^0 [g u g + (\omega^4)^4 u u g] \right) + \omega^0 \left( [g g u + (\omega^4)^2 u g u] + (\omega^2)^4 [g u u + (\omega^4)^6 u u u] \right) \\ &= \left( [ggg + u g g] + [g u g + u u g] \right) + \left( [g g u + u g u] + [g u u + u u u] \right) = \sum_{i=0}^7 f(\omega^i) \end{aligned}$$

Die Operation entspricht also einer (arithmetischen) Mittelung. Was ja auch zum ursprünglichen Polynom  $f(x) = \sum a_i x^i$  paßt, dessen konstanter Anteil eben  $a_0$  ist. Dieser Koeffizient  $a_0$  ist in allen Funktionswerten  $r_i$  gleich stark “enthalten”, weil er in der Linearkombination<sup>12</sup> mit  $x^0 \equiv 1$  gewichtet wird.

Andere Koeffizienten  $a_i$  symbolisieren (und quantifizieren!) das Vorhandensein anderer “Frequenzen” im Funktionsverlauf ...

<sup>11</sup>hier statt  $\text{FFT}^{-1}$  die vorwärts-FFT, damit wir  $a_0$  mit  $r_0$  identifizieren, die Bezeichner  $g_i, u_i, gg_i, \dots$  verwenden und  $\omega$  statt  $\omega^*$  schreiben können; dadurch fehlt hier auch der Normierungsfaktor  $1/n$

<sup>12</sup>eine solche ist ja das Polynom!

#### Aufgabe 44

- a) Um die Orthogonalität der  $T_u : [-\frac{1}{2}, 7\frac{1}{2}] \rightarrow [-1, 1]$  zu zeigen,<sup>13</sup> ist das Skalarprodukt dieser vektoriellen Basis der Kosinustransformation zu bestimmen. Das Skalarprodukt zweier Funktionen  $T_u$  und  $T_v$  ist aber das Integral ihres Produkts:

$$T_u \cdot T_v = \int_I T_u(x) \cdot T_v(x) dx$$

Wenn dieses für  $u \neq v$  Null ist, bilden die Komponenten  $T_u$  ein Orthogonalsystem:

$$\begin{aligned} T_u \cdot T_v &= \int_{-\frac{1}{2}}^{7\frac{1}{2}} T_u(x) \cdot T_v(x) dx \\ &= \int_{-\frac{1}{2}}^{7\frac{1}{2}} \cos\left(u \frac{(2x+1)\pi}{16}\right) \cdot \cos\left(v \frac{(2x+1)\pi}{16}\right) dx \\ &= \frac{1}{2} \int_{-1}^{15} \cos\left(u \frac{(x+1)\pi}{16}\right) \cdot \cos\left(v \frac{(x+1)\pi}{16}\right) dx \\ &= \frac{1}{2} \int_0^{16} \cos\left(u \frac{x\pi}{16}\right) \cdot \cos\left(v \frac{x\pi}{16}\right) dx \\ &= \frac{1}{2} \cdot \frac{16}{\pi} \int_0^\pi \cos(u \cdot x) \cdot \cos(v \cdot x) dx \end{aligned}$$

Von diesen Termen  $(\cos(u \cdot x))_{u=0,1,2,\dots}$  wissen wir aber bereits aus der Fouriertransformation, daß sie auf dem Intervall  $[0, 2\pi] \ni x$  paarweise orthogonal sind!

Da der Kosinus bei  $\cos 0 = 1$  beginnt und an unserer (skalierten) Intervallgrenze den Wert  $\cos \pi = -1$  annimmt, braucht für  $u = 1$  (und infolge dessen für  $u = 3, 5, 7, \dots$ ) nicht über die ganze Periode  $[0, 2\pi]$  integriert zu werden – für  $u = 2, 4, 6, \dots$  ohnehin nicht; die Produkte  $T_u T_v$  sind in  $[0, \pi]$  und  $[\pi, 2\pi] \cong [-\pi, 0]$  symmetrisch, weil der Kosinus symmetrisch ist:  $\cos(x) \equiv \cos(-x)$ .

Das heißt, wenn  $\int_0^{2\pi} \dots dx = \int_0^\pi \dots dx + \int_\pi^{2\pi} \dots dx = 2 \int_0^\pi \dots dx$  ist, muß  $\int_0^{2\pi} \dots dx = 0$  bedeuten, daß schon  $\int_0^\pi \dots dx = 0$  ist. Die  $T_u$  sind also paarweise orthogonal.

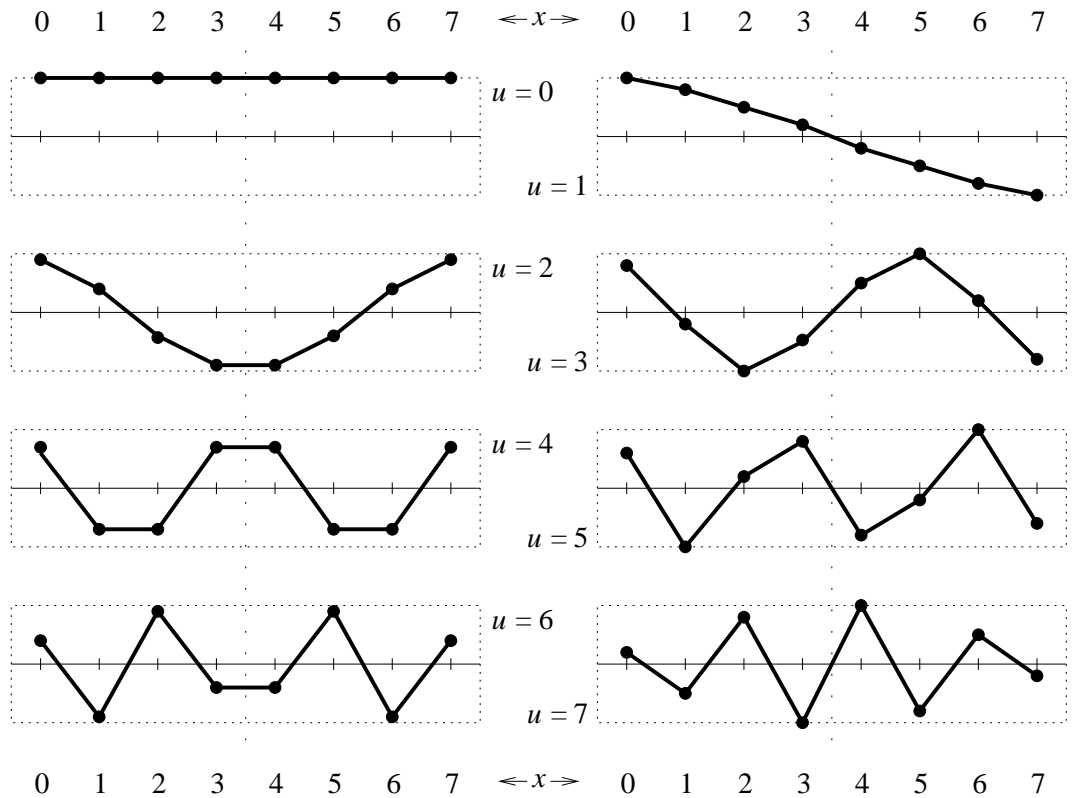
- b) (Wenn wir im folgenden von einer *phasenverschobenen Funktion*  $f(x + \alpha)$  sprechen ( $\alpha, x \in \mathbb{N}$ ), dann ist selbstverständlich

$$f(x + \alpha) := f((x + \alpha) \bmod 8)$$

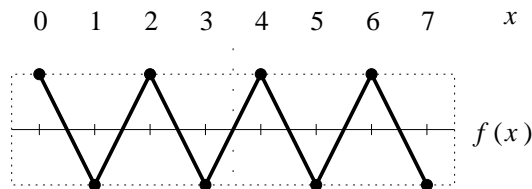
gemeint, denn wir bewegen uns nach wie vor im Definitionsbereich des kleinen  $8 \times 8$ -Kästchens, das JPEG bearbeitet.)

Wir betrachten im vorliegenden linearen Fall die acht Basisfunktionen  $T_u$ :

<sup>13</sup>die im übrigen auch aus der Matrix  $M$  auf Seite 42 des Vorlesungsskriptums (Kapitel 5) ersichtlich ist, dort ist ja  $M \cdot M^T = I$ , wenn auch numerisch, berechnet worden



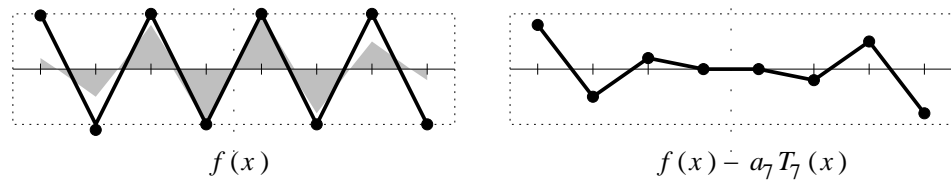
• Die Funktion  $f : x \mapsto (-1)^x$



Wie anhand der Diagramme zu sehen ist,<sup>14</sup> wird diese Funktion eine Linearkombination (gewichtete Summe) der ungeraden Komponenten  $u = 2i + 1$  sein:

$$f = \sum_{i=0}^3 a_{2i+1} T_{2i+1}$$

Den betragsmäßig größten Koeffizienten  $a_7 \approx 1$  wird dabei die Höchsfrequenz-Komponente  $T_7$  abbekommen; der Rest  $f(x) - a_7 T_7(x)$  wird durch eine Linearkombination der  $T_5$ ,  $T_3$  und  $T_1$  approximiert:



(grau unterlegt ist  $T_7$ )

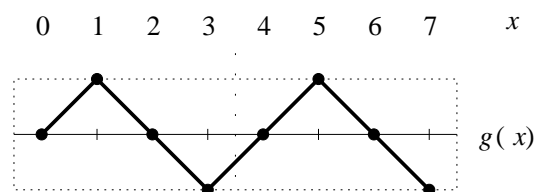
<sup>14</sup>Wem das Hinsehen nicht reicht, der/die berechne für  $i = 0, \dots, 3$  die "diskreten Integrale"  $\sum_{x=0}^7 T_{2i}(x)f(x)$  ( $= 0$ ), aus denen ja die  $a_{2i}$  hervorgehen.

Die einzige nichttriviale **Phasenverschiebung**, die im diskreten Fall für  $f$  möglich ist, ist die um eine Stelle, dann haben wir  $\tilde{f}(x) = f(x+1) = (-1)^{x+1}$ .

Dies ist aber genau  $\tilde{f}(x) = -f(x)$ . Das bedeutet, daß sich auch die Koeffizienten  $a_i$  von  $f$  negieren müssen:

$$\tilde{f} = \sum_{i=0}^3 -a_{2i+1} T_{2i+1}$$

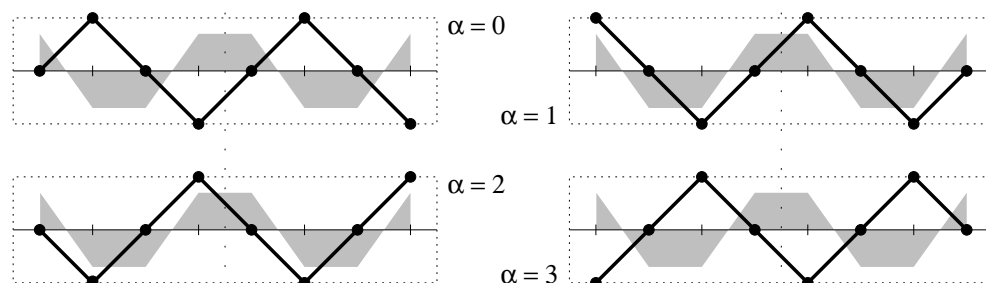
- Die Funktion  $g : 2x \mapsto 0, 2x+1 \mapsto (-1)^x$



Sie hat die halbe Frequenz von  $f$  und mit  $\lambda = 4$  dieselbe Periode wie die gerade Funktion  $T_4$  (siehe Skizze). Der betragsmäßig größte Koeffizient von

$$g = \sum_{i=0}^7 a_i T_i$$

wird daher  $a_4$  sein, und zwar *unabhängig* von einer eventuellen Phasenverschiebung  $\alpha \in \{0, \dots, \lambda-1\}$ :



Für  $\alpha = 2$  ist die phasenverschobene Funktion  $g(x+\alpha)$  genau  $g(x+2) = -g(x)$  und die Koeffizienten  $a_i$  müssen sich komplett negieren. Wie leicht an der Gestalt von  $T_4$  (grau unterlegt) zu sehen ist, wird der Hauptkoeffizient  $a_4$  dadurch positiv, wie auch bei  $\alpha = 1$ , wo er exakt denselben Wert annimmt.

Es ist für  $\alpha \in \{1, 2\}$  symmetrischerweise  $g(x+2) = g(7-(x+1))$ , was bedeutet, daß die Koeffizienten der ungeraden Komponenten  $a_{2i+1}$  sich negieren, während die der geraden Komponenten  $a_{2i}$  gleich bleiben.

Selbiges gilt für  $\alpha \in \{0, 3\}$ ; es ist  $g(x+3) = g(7-x)$ .

Dies alles bedeutet schließlich, daß sich für die Funktion  $g$  die vier möglichen **Phasenverschiebungen** in einer Kombination

- der evtl. Negation der geraden Koeffizienten  $a_{2i}$  mit
- der evtl. Negation der ungeraden Koeffizienten  $a_{2i+1}$

niederschlagen, während die Beträge aller Koeffizienten sich nicht ändern:

Vorzeichen von	$a_{2i}$	$a_{2i+1}$
$\alpha = 0$	+	+
$\alpha = 1$	-	+
$\alpha = 2$	-	-
$\alpha = 3$	+	-

Die Kosinustransformation schafft es also ohne jegliche ergänzenden Sinus-Terme (!), Phasenverschiebungen in die Koeffizienten zu "kodieren".

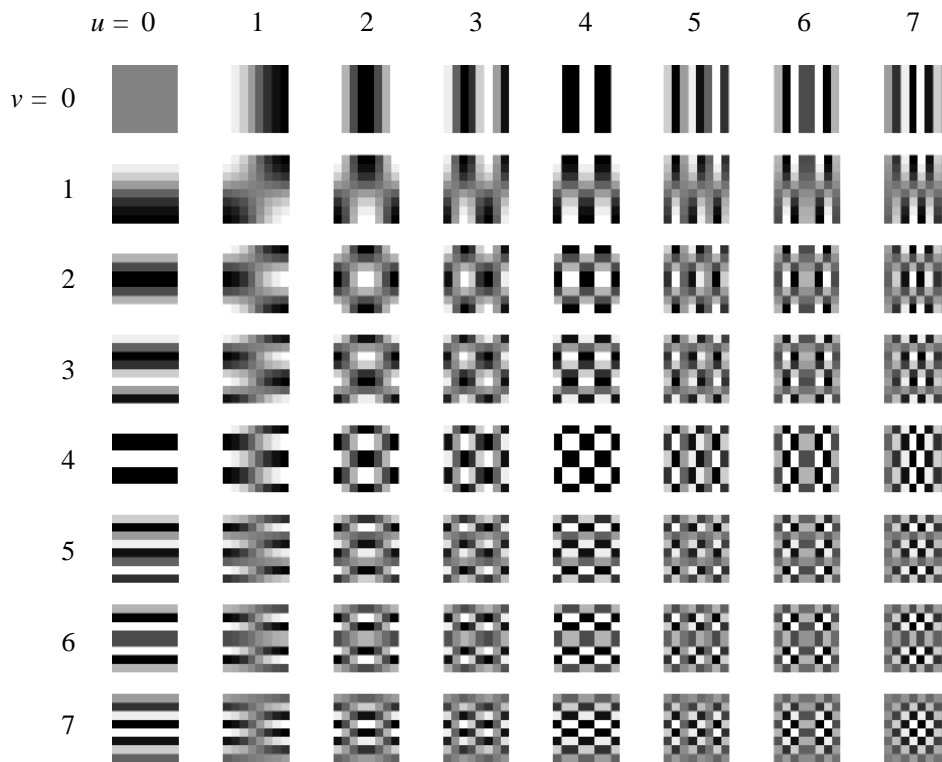
Bei einer einzigen ganzzahligen Eigenfrequenz (wie in unserm Beispiel der Funktionen  $f$  und  $g$ ) reichen offenbar sogar die Vorzeichen der Koeffizienten aus, um die Phaseninformation aufzunehmen.

Durch das Erhöhen um Eins (bzw. um  $u\pi$ ) im Zähler  $u(2x + 1)\pi$  des Kosinargumentes wird ja für ungerade  $u$  genau diejenige Phasenverschiebung erzeugt, die aus dem geraden Kosinus einen ungeraden Pseudo-Sinus macht, wenn man den Mittelpunkt  $x = 3\frac{1}{2}$  des diskreten Intervalls  $x \in \{0, 1, \dots, 7\}$  als Ursprung nimmt. Während die Fouriertransformation für jede Frequenz  $u$  je eine gerade und ungerade Basisfunktion bereithält, hat also die Kosinustransformation *abwechselnd* gerade und ungerade Terme (siehe die Skizzen bei (b)). Da die Basis der Diskreten Kosinustransformation denselben Umfang hat wie das zu transformierende Datenmaterial, wird die Information über die acht Skalare  $f(x)$  sowohl restlos als auch nicht redundant in den acht reellwertigen Koeffizienten  $a_i$  gespeichert.

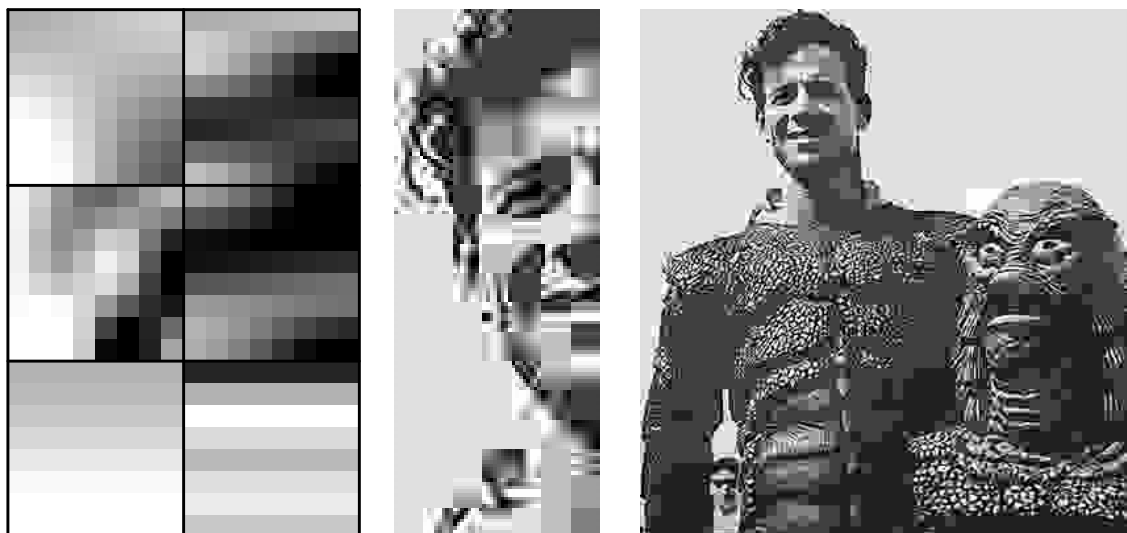
Im zweidimensionalen Fall der Bildkompression mittels JPEG besteht daher die Diskrete Kosinustransformation in der bijektiven Abbildung

$$DCT : \mathbb{R}^{8 \times 8} \rightarrow \mathbb{R}^{8 \times 8}$$

Eine Diskrete Fouriertransformation, die dasselbe Frequenzspektrum betrachtet, muß hingegen redundant sein, wenn sich dort acht Werte  $f(x)$  zu *jeweils* acht Koeffizienten  $a_i$  und  $b_i$ , also auf sechzehn verdoppeln. (Im 2D-Fall der Bildkompression dürfte sich die Anzahl der Koeffizienten sogar von  $8^2 = 64$  auf  $(2 \cdot 8)^2 = 256$  vervierfachen ...)



Die Basis der (zweidimensionalen) Diskreten Kosinustransformation



Die  $8 \times 8$ -Kästchen des komprimierten Bildes sind Linearkombinationen der oben abgebildeten Basisfunktionen, hier gut sichtbar durch eine besonders drastische Komprimierung (einen hohen Q-Faktor und wenige der  $8 \cdot 8 = 64$  Koeffizienten  $\neq 0$ ).