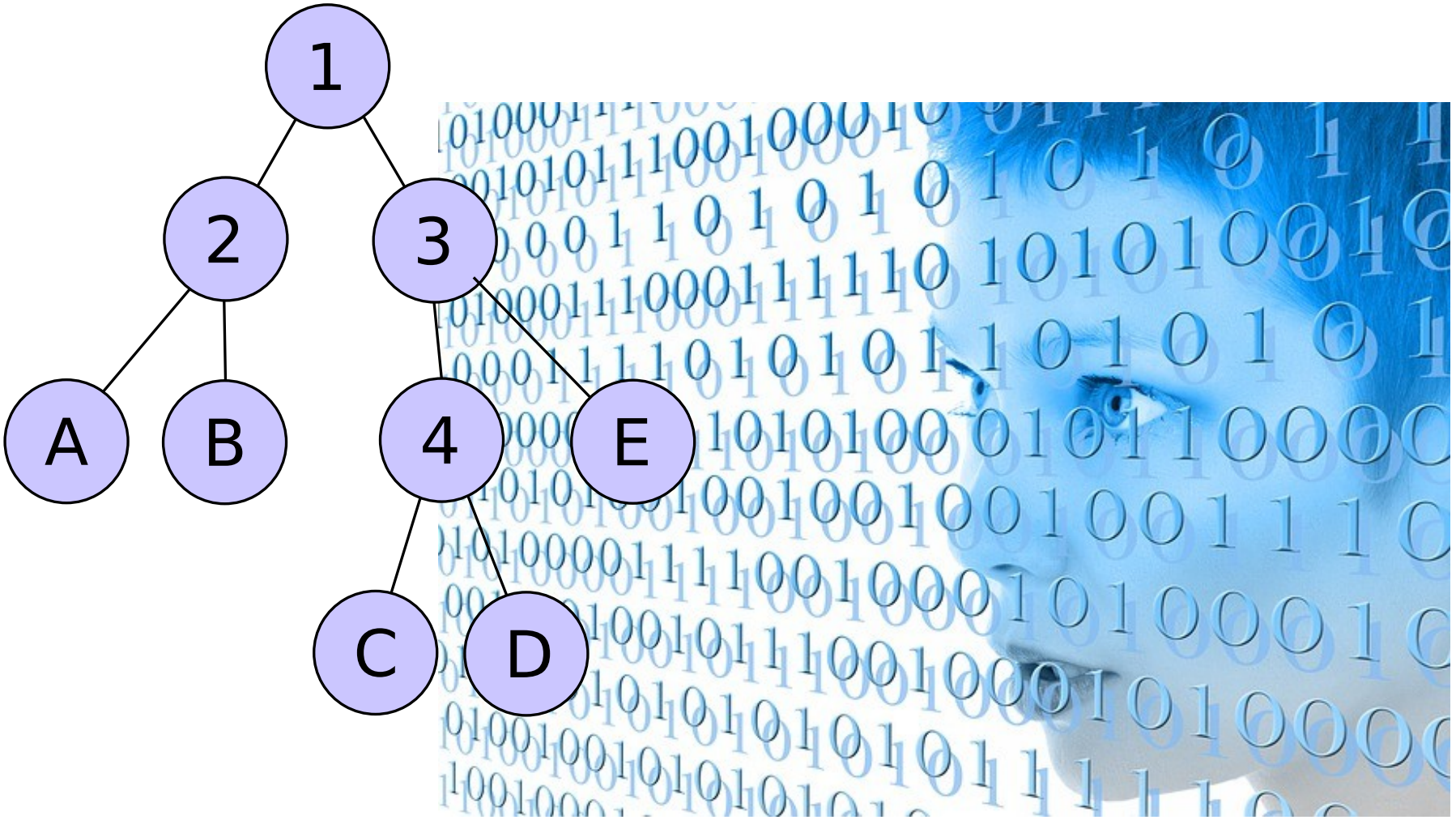
Komprimierung



Kaneiderdaniel in der Wikipedia auf Deutsch

Image by Gerd Altmann from Pixabay

Kleinste Informationseinheit

1 Bit

1 Bit ist die **kleinste Informationseinheit**
(man sagt auch **atomare** Informationseinheit)

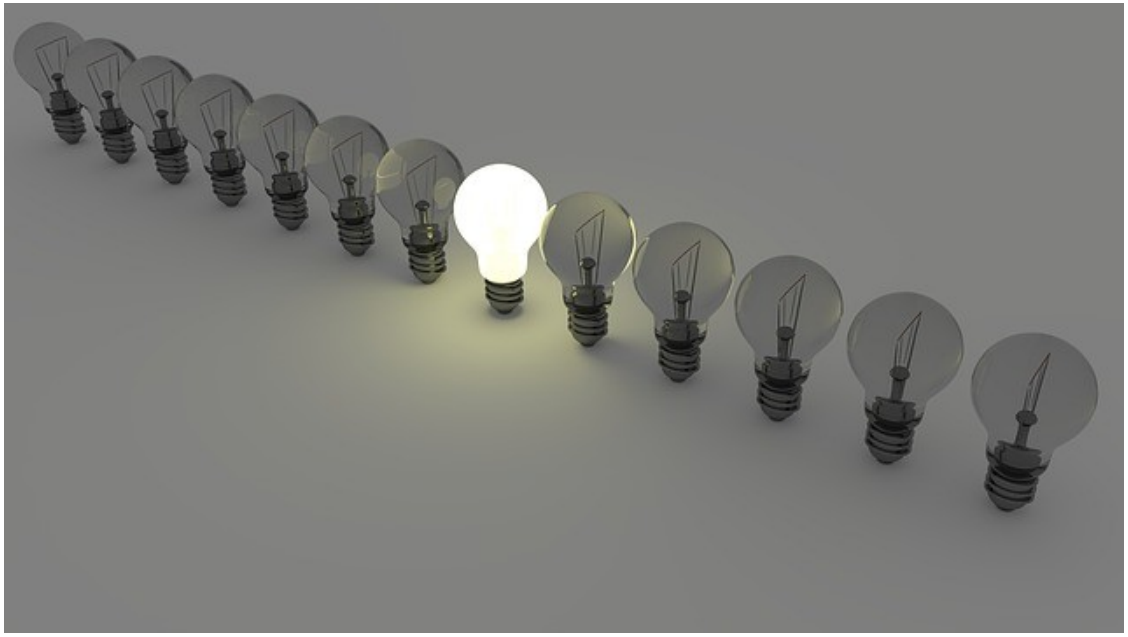


Bild von Colin Behrens auf Pixabay

Zechnall (September 2021)
Schiebel (Oktober 2023)

Codierung und Information

LIEBE

Im Deutschen gibt es 26 Buchstaben, mit Umlauten 30. Unterscheidet man Klein- und Großbuchstaben so sind es 60. Nimmt man den 8-Bit-ASCII zur Codierung kann man 256 Buchstaben und Zeichen codieren.

→ **Speicherplatz für „LIEBE“: 5 * 8 Bit = 40 Bit**

Das chinesische Zeichen für LIEBE

Im Chinesischen gibt es ca. 10.000 Zeichen

→ bei fester Bitlänge benötigt jedes Zeichen 14 Bit

→ **Speicherplatz für „LIEBE“: 14 Bit**

Dezimal	Binär		Dezimal	Binär		Dezimal	Binär	
32	00100000	SP	64	01000000	@	96	01100000	`
33	00100001	!	65	01000001	A	97	01100001	a
34	00100010	"	66	01000010	B	98	01100010	b
35	00100011	#	67	01000011	C	99	01100011	c
36	00100100	$	68	01000100	D	100	01100100	d
37	00100101	%	69	01000101	E	101	01100101	e
38	00100110	&	70	01000110	F	102	01100110	f
39	00100111	'	71	01000111	G	103	01100111	g
40	00101000	(	72	01001000	H	104	01101000	h
41	00101001	)	73	01001001	I	105	01101001	i
42	00101010	*	74	01001010	J	106	01101010	j
43	00101011	+	75	01001011	K	107	01101011	k
44	00101100	,	76	01001100	L	108	01101100	l
45	00101101	-	77	01001101	M	109	01101101	m
46	00101110	.	78	01001110	N	110	01101110	n
47	00101111	/	79	01001111	O	111	01101111	o
48	00110000	0	80	01010000	P	112	01110000	p
49	00110001	1	81	01010001	Q	113	01110001	q
50	00110010	2	82	01010010	R	114	01110010	r
51	00110011	3	83	01010011	S	115	01110011	s
52	00110100	4	84	01010100	T	116	01110100	t
53	00110101	5	85	01010101	U	117	01110101	u
54	00110110	6	86	01010110	V	118	01110110	v
55	00110111	7	87	01010111	W	119	01110111	w
56	00111000	8	88	01011000	X	120	01111000	x
57	00111001	9	89	01011001	Y	121	01111001	y
58	00111010	:	90	01011010	Z	122	01111010	z
59	00111011	;	91	01011011	[	123	01111011	{
60	00111100	<	92	01011100	\	124	01111100	|
61	00111101	=	93	01011101	]	125	01111101	}
62	00111110	>	94	01011110	^	126	01111110	~
63	00111111	?	95	01011111	_	127	01111111	DEL

Zechnall (September 2021)
Schiebel (Oktober 2023)

Chinesisches Zeichen von Clker-Free-Vector-Images auf Pixabay

Codierung und Information

LIEBE → 40 Bit

Trägt das deutsche **Wort** „Liebe“ **mehr Information** als das chinesische Schrift**zeichen**?

L → 8 Bit

Trägt ein **chinesisches Zeichen mehr Information** als ein **Deutsches**?

→ 14 Bit

Tragen **deutsche Zeichen unterschiedlich viel Information**?

Chinesisches Zeichen von Clker-Free-Vector-Images auf Pixabay

Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – Kann man das lesen?

an atte ieen are ei eine errn
eient, a ra er u i ,err, eine eit
it eru, nun ote i erne ieer ei u
einer utter, et ir einen on.‘ er
err antortete ,u at ir treu un
eri eient, ie er ient ar, o o er on
ein,‘ un a i ein tü o, a o roß a
anen o ar. an o ein Tüein au er
Tae, iete en uen inein, ette in au
ie uter un ate i au en e na au.

Zechnall (September 2021)
Schiebel (Oktober 2023)

https://www.abenteuer-informatik.de/tafeln.html

Komprimierung – Kann man das lesen?

Hns htt sbn Jhr b snm Hrrn
gdnt, d sprch r z hm 'Hrr, mn
Zt st hrm, nn wllt ch grn wdr
hm z mnr Mttr, gbt mr mnn
Lhn.' Dr Hrr ntwrtt 'd hst mr tr
nd hrlch gdnt, w dr Dnst wr, s
sll dr Lhn sn,' nd gb hm n Stck
Gld, ds s grß ls Hnsns Kpf wr.
Hns zg n Tchln s dr Tsch, wcklt
dn Klmpn hnn, stzt hn f d

Zechnall (September 2021)
Schiebel (Oktober 2023)

https://www.abenteuer-informatik.de/tafeln.html

Komprimierung durch Entropiecodierung

Variable Bitlänge

Seltene Buchstaben tragen mehr Information → Codierung mit **mehr Bit**

Bsp.: B → 001

Häufige Buchstaben tragen weniger Information → Codierung mit **weniger Bit**

Bsp.: E → 1

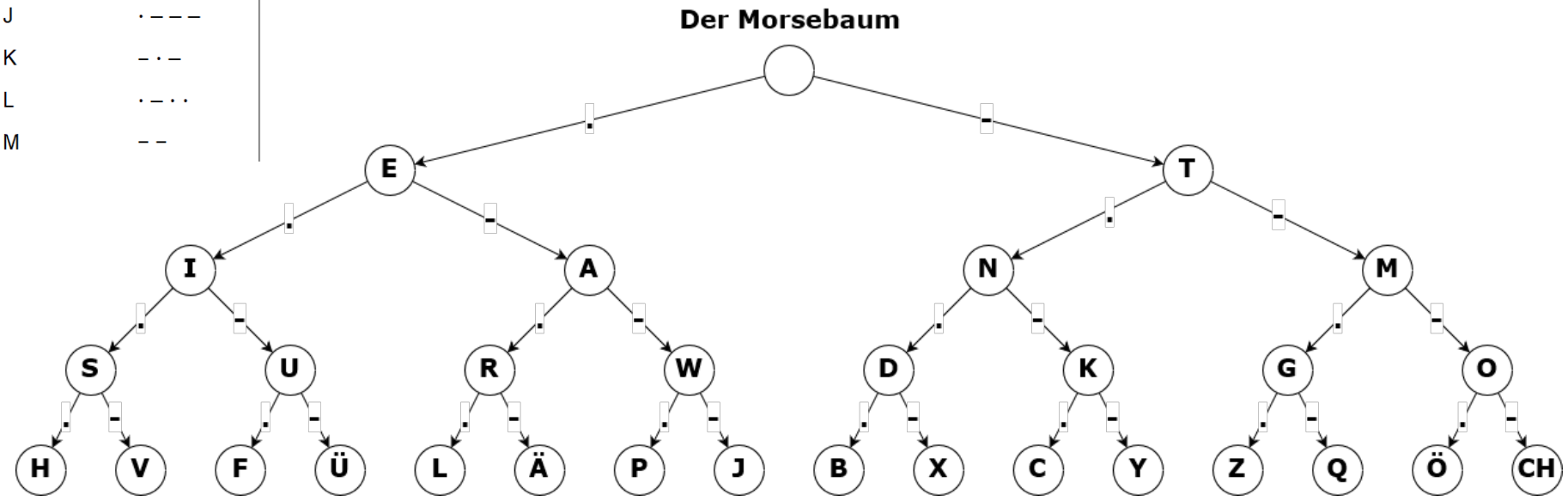
Fazit 1: Texte lassen sich **komprimieren**, indem **häufig vorkommende Buchstaben** mit **weniger Bit codiert** werden!

Zechnall (September 2021)
Schiebel (Oktober 2023)

Variable Bitlänge im Morsebaum

Buchstabe	Code
A	· –
B	– · · ·
C	– · – ·
D	– · ·
E	·
F	· · – ·
G	– – ·
H	· · · ·
I	· ·
J	· – – –
K	– · –
L	· – · ·
M	– –

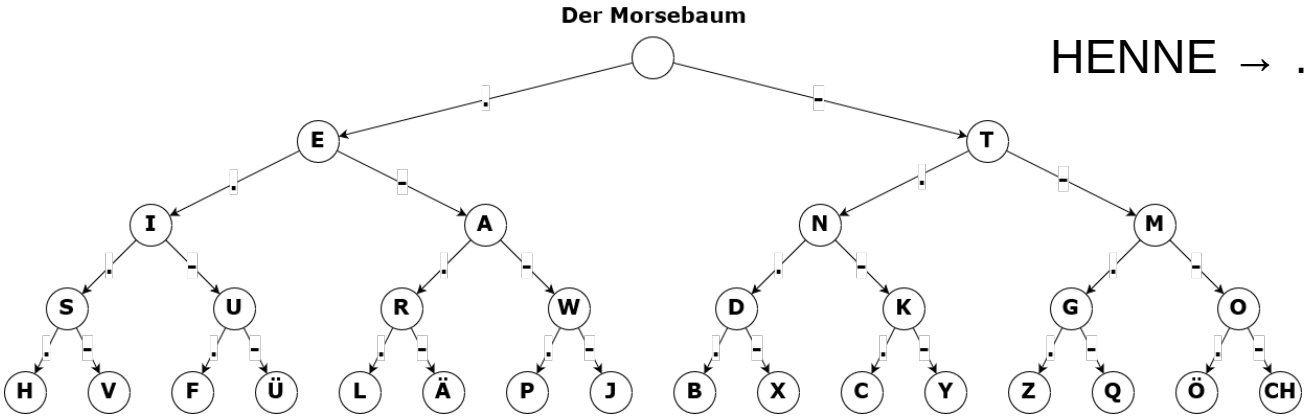
Morsecode und Morsebaum



Zechnall (September 2021)
Schiebel (Oktober 2023)

Präfixfreiheit

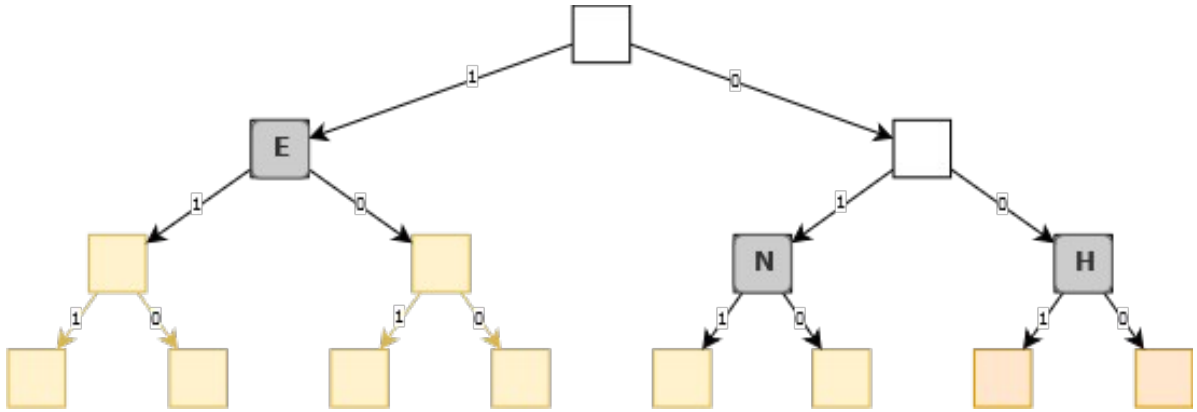
Der Morsecode ist nicht präfixfrei



HENNE → .... . -. -. . → .....-.-..

Unterschied?

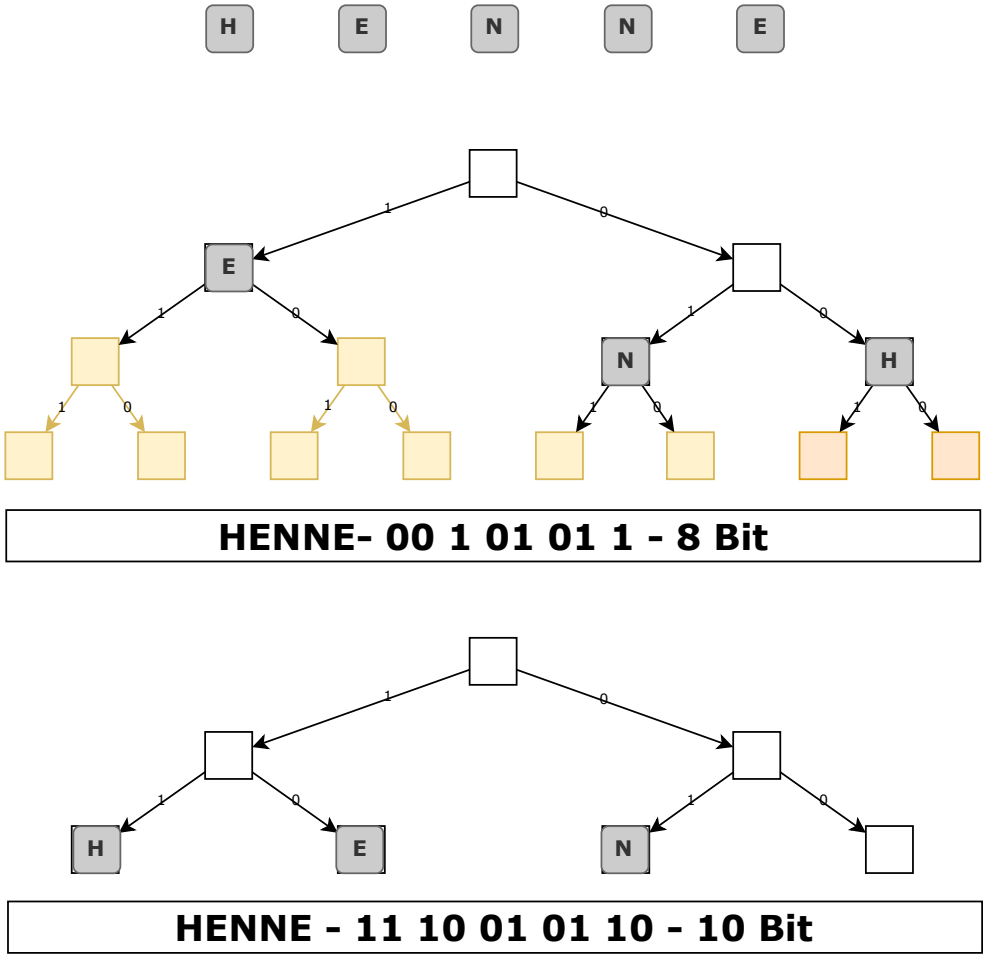
Präfixfreiheit bei variabler Bitlänge



HENNE → 00101011

Zechnall (September 2021)
Schiebel (Oktober 2023)

Präfixfreiheit



Unterschied?

Fazit 2: Präfixfreie Codes benötigen **keine Trennzeichen** (Pausen) und **sparen Speicherplatz** → Komprimierung

Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Codierung (Entropiecodierung)

Zum besseren Verständnis wird hier ein Text Huffman-codiert. Selbstverständlich funktioniert das Verfahren auch mit Binärdaten.

TERRASSENTONNE

1. Schritt:
Symbole und ihre Häufigkeit feststellen

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14

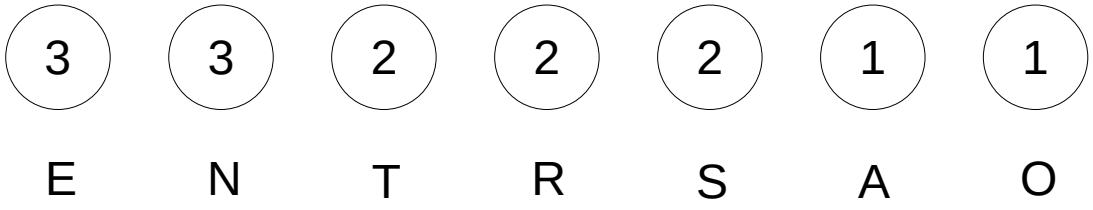
Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



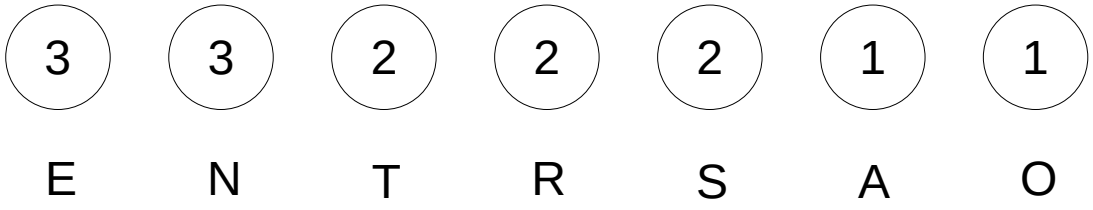
Huffman-Codierung: Schritt-für-Schritt

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14

2. Schritt:
Huffman-Baum nach Algorithmus generieren

a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

b) Zwei Knoten mit den **geringsten Häufigkeiten** werden zu Kindknoten eines **neuen Mutterknotens**, der die Summe der Häufigkeiten als Schlüssel trägt.



Huffman-Codierung: Schritt-für-Schritt

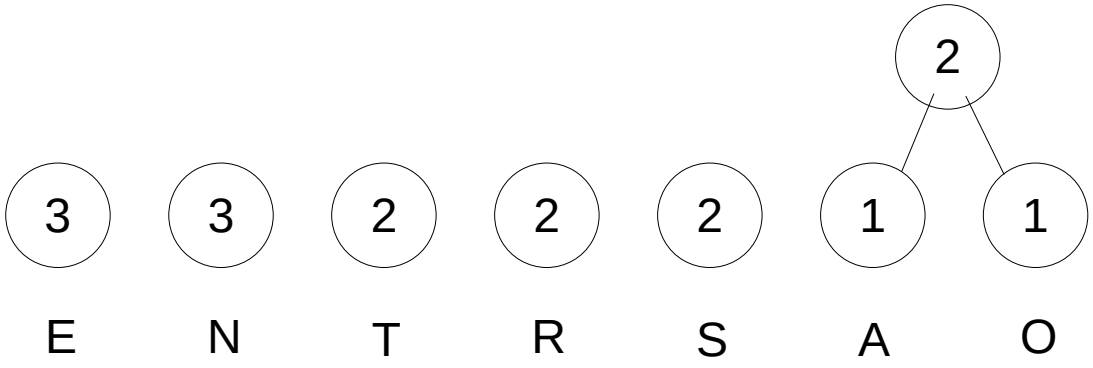
Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14

2. Schritt:
Huffman-Baum nach Algorithmus generieren

a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

b) Zwei Knoten mit den **geringsten Häufigkeiten** werden zu Kindknoten eines **neuen Mutterknotens**, der die Summe der Häufigkeiten als Schlüssel trägt.

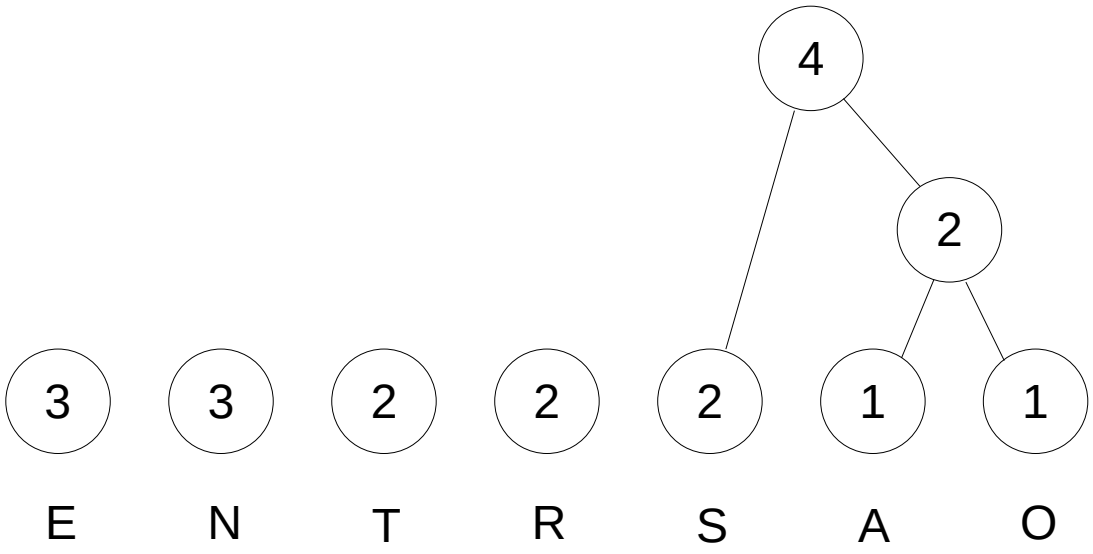
c) Vorgang b) wird so lange wiederholt, bis der letzte Mutterknoten (=Wurzel!) erzeugt ist.



Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

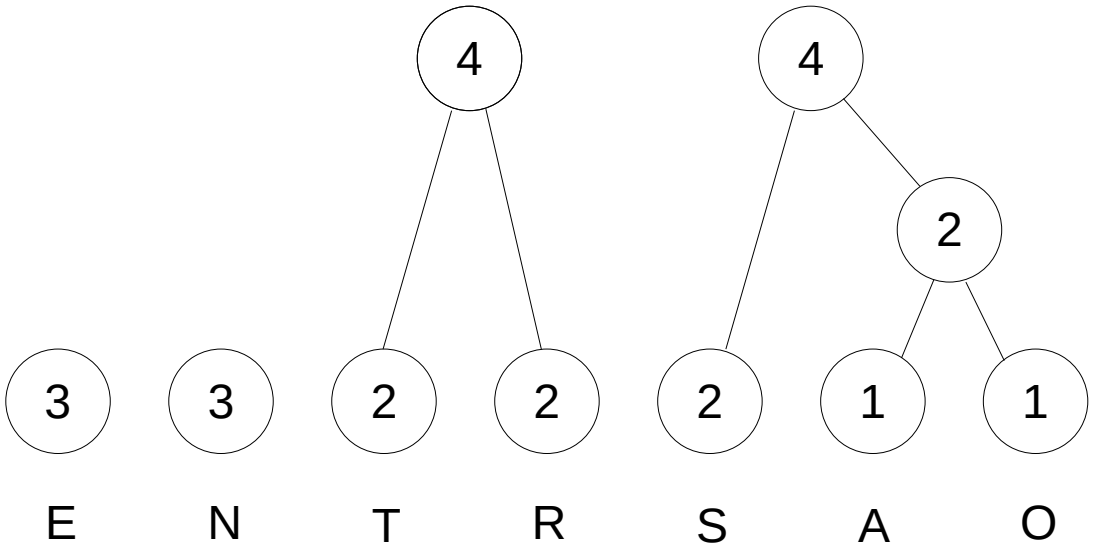
b) Zwei Knoten mit den geringsten Häufigkeiten werden zu Kindknoten eines neuen Mutterknotens, der die Summe der Häufigkeiten als Schlüssel trägt.

c) Vorgang b) wird so lange wiederholt, bis der letzte Mutterknoten (=Wurzel!) erzeugt ist.

Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

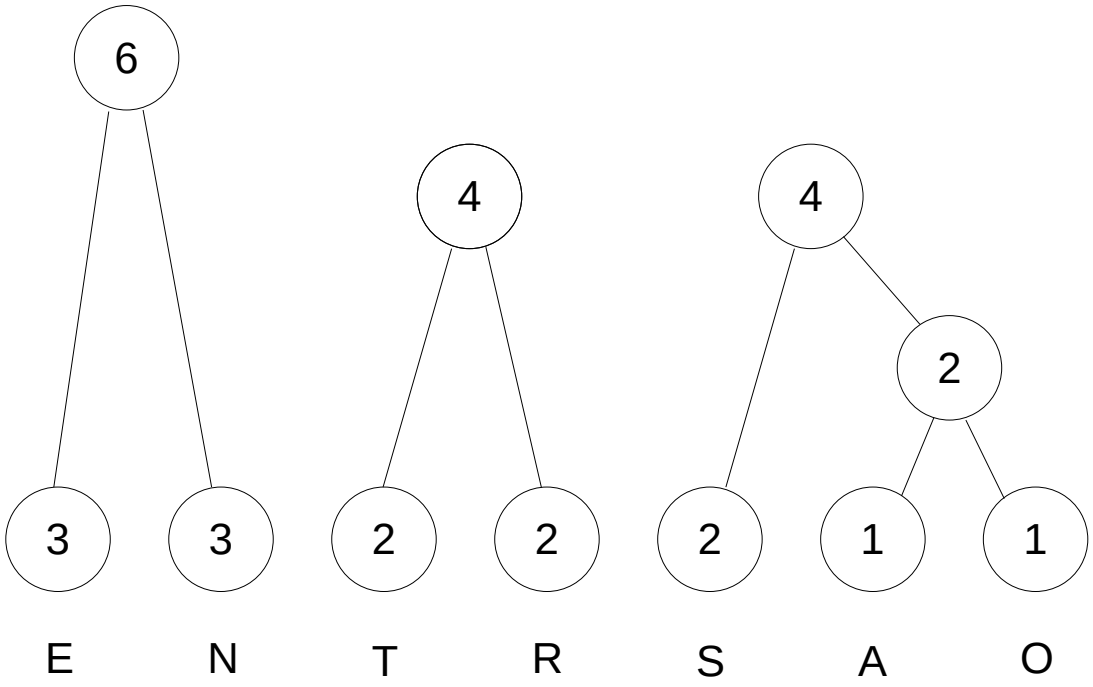
b) Zwei Knoten mit den geringsten Häufigkeiten werden zu Kindknoten eines neuen Mutterknotens, der die Summe der Häufigkeiten als Schlüssel trägt.

c) Vorgang b) wird so lange wiederholt, bis der letzte Mutterknoten (=Wurzel!) erzeugt ist.

Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

b) Zwei Knoten mit den geringsten Häufigkeiten werden zu Kindknoten eines neuen Mutterknotens, der die Summe der Häufigkeiten als Schlüssel trägt.

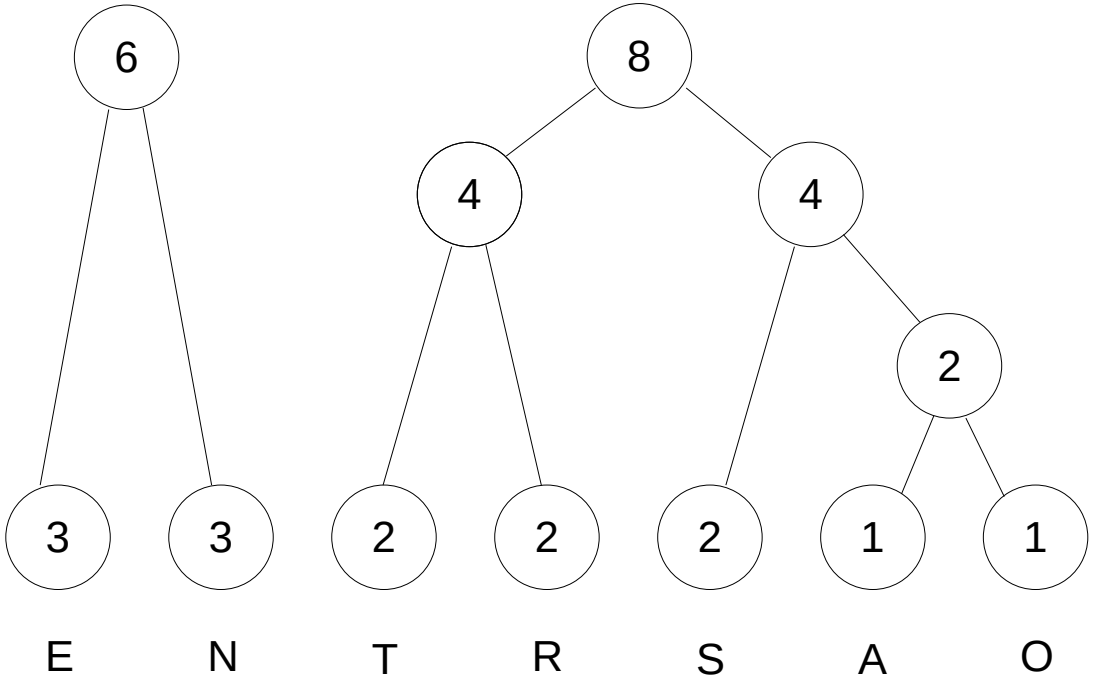
c) Vorgang b) wird so lange wiederholt, bis der letzte Mutterknoten (=Wurzel!) erzeugt ist.

Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

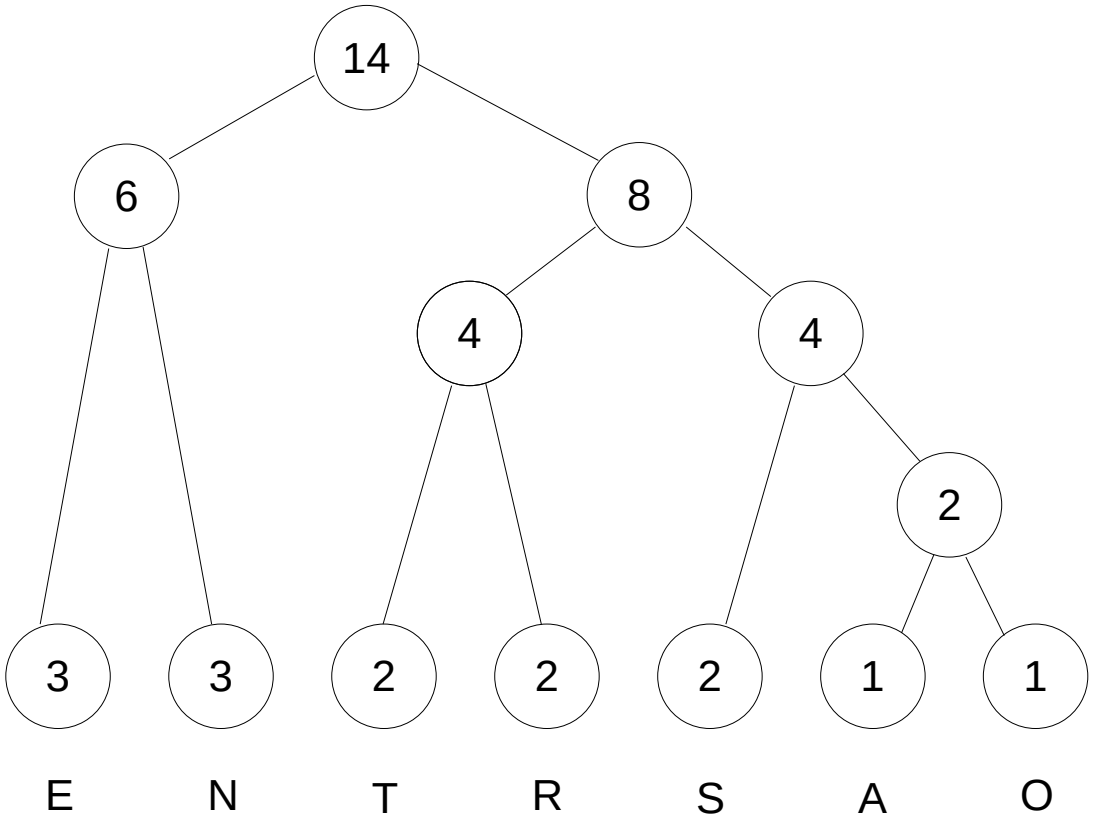
b) Zwei Knoten mit den geringsten Häufigkeiten werden zu Kindknoten eines neuen Mutterknotens, der die Summe der Häufigkeiten als Schlüssel trägt.

c) Vorgang b) wird so lange wiederholt, bis der letzte Mutterknoten (=Wurzel!) erzeugt ist.

Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



a) Blätter des Binärbaums tragen die Häufigkeit als Schlüssel

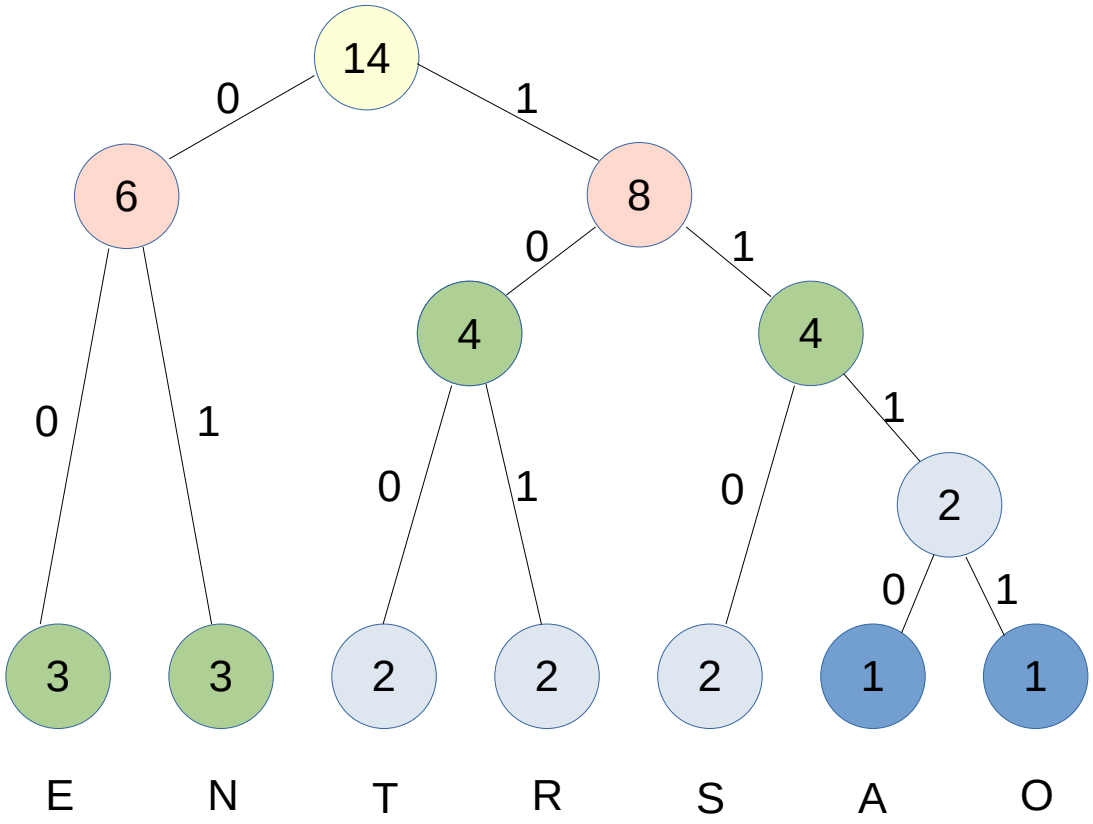
b) Zwei Knoten mit den geringsten Häufigkeiten werden zu Kindknoten eines neuen Mutterknotens, der die Summe der Häufigkeiten als Schlüssel trägt.

c) Vorgang b) wird so lange wiederholt, bis der letzte Mutterknoten (=Wurzel!) erzeugt ist.

Huffman-Codierung: Schritt-für-Schritt

2. Schritt:
Huffman-Baum nach Algorithmus generieren

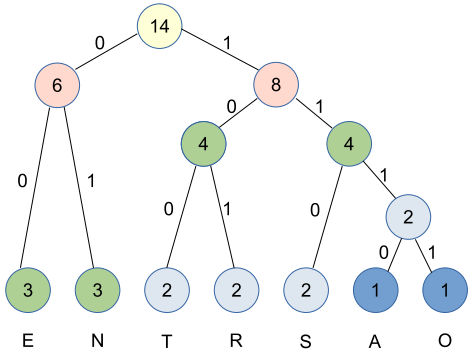
Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14



Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Codierung: Schritt-für-Schritt

3. Schritt:
Codierung notieren



Zeichen	E	N	T	R	S	A	O
Abs. H.	3	3	2	2	2	1	1
Rel. H.	3/14	3/14	2/14	2/14	2/14	1/14	1/14
Code	00	01	100	101	110	1110	1111
Bits	2	2	3	3	3	4	4

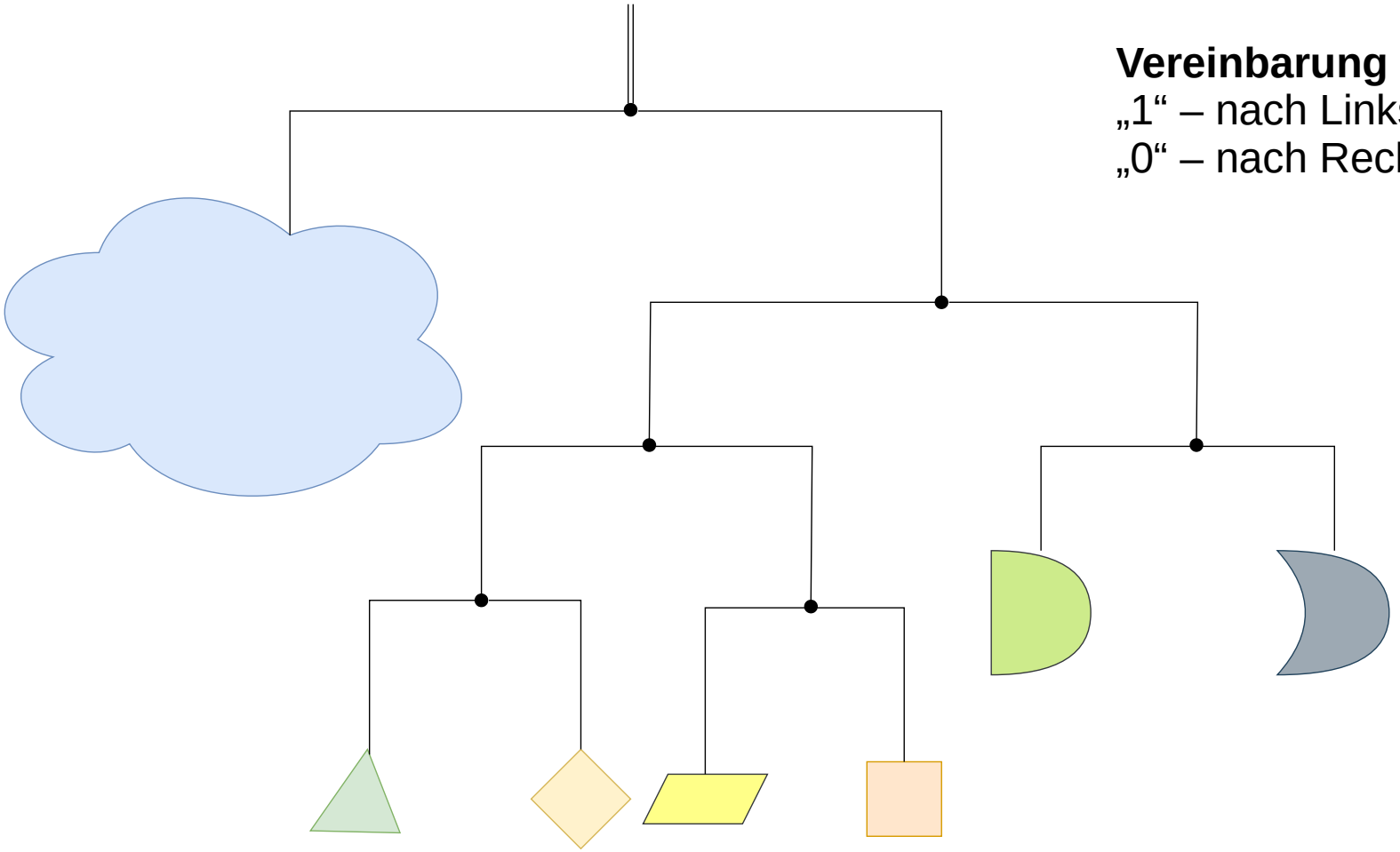
Speicherbedarf:

8-Bit-Codierung 14*8 bit = 112 Bit

Huffman-Codierung (3*2 + 3*2 + 2*3 + 2*3 + 2*3 + 1*4 + 1*4) bit = 38bit Bit

Durchschnitt je Zeichen 38 bit/14 ≈ 2,7 Bit

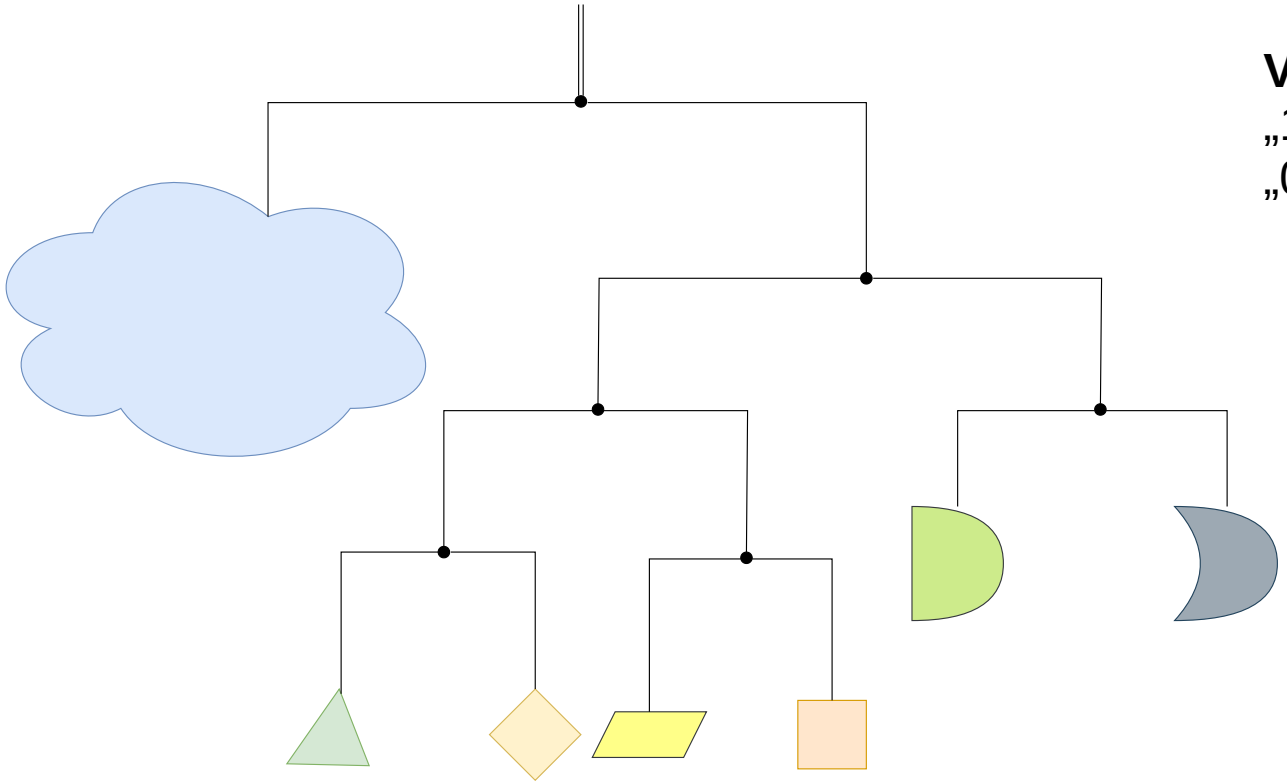
Das Huffmanverfahren und die Mobile-Idee



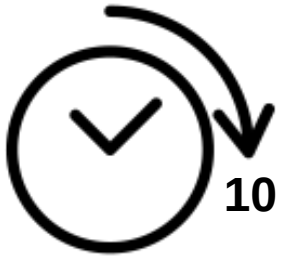
Vereinbarung
„1“ – nach Links
„0“ – nach Rechts

Zechnall (September 2021)
Schiebel (Oktober 2023)

Das Huffmanverfahren und die Mobile-Idee



Vereinbarung
„1“ – nach Links
„0“ – nach Rechts



10 Minuten

Arbeitsphase: Komprimiere den Satz „EBER_ESSEN_EIER“
(mit weniger als 45 Bit) mit der „Mobile-Idee“

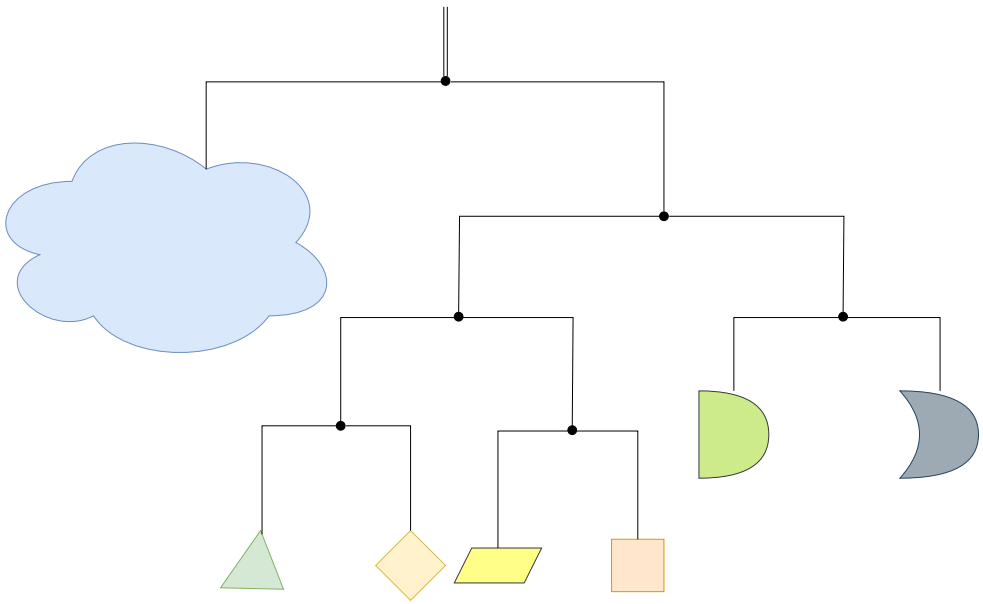
Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Komprimierung Hilfe

Hilfe zur Mobile-Idee

EBER_ESSEN_EIER

Zeichen	E	S	R	_	B	I	N
n							
Code							



1. Bestimme zuerst die Häufigkeit der Zeichen.

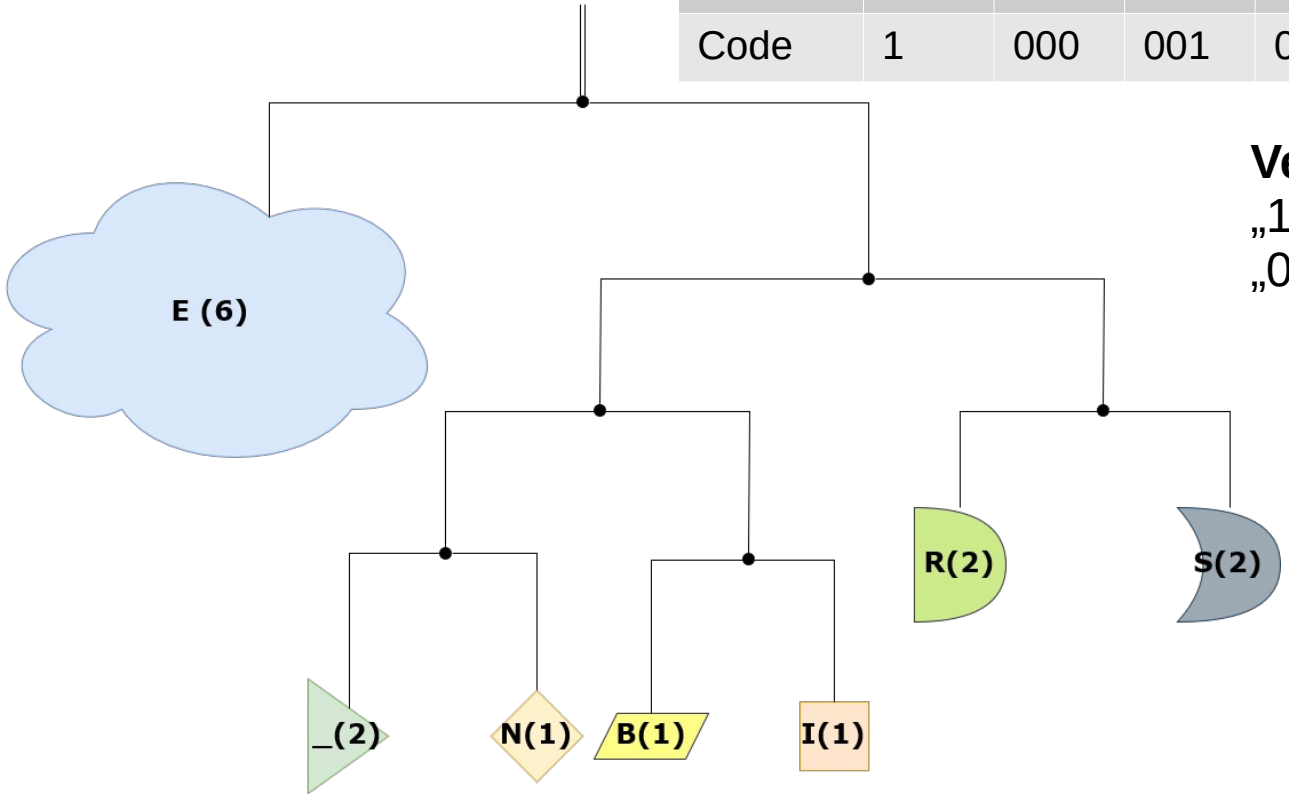
2. Hänge die Zeichen dann an ein Mobile, so dass dieses „ausgeglichen“ ist.

3. Erstelle dann unter der **Vereinbarung**
„1“ – nach Links
„0“ – nach Rechts
die Codierung der Zeichen

Huffman-Komprimierung Hilfe

Lösung

Zeichen	E	S	R	_	B	I	N
n	6	2	2	2	1	1	1
Code	1	000	001	0111	0101	0100	0110



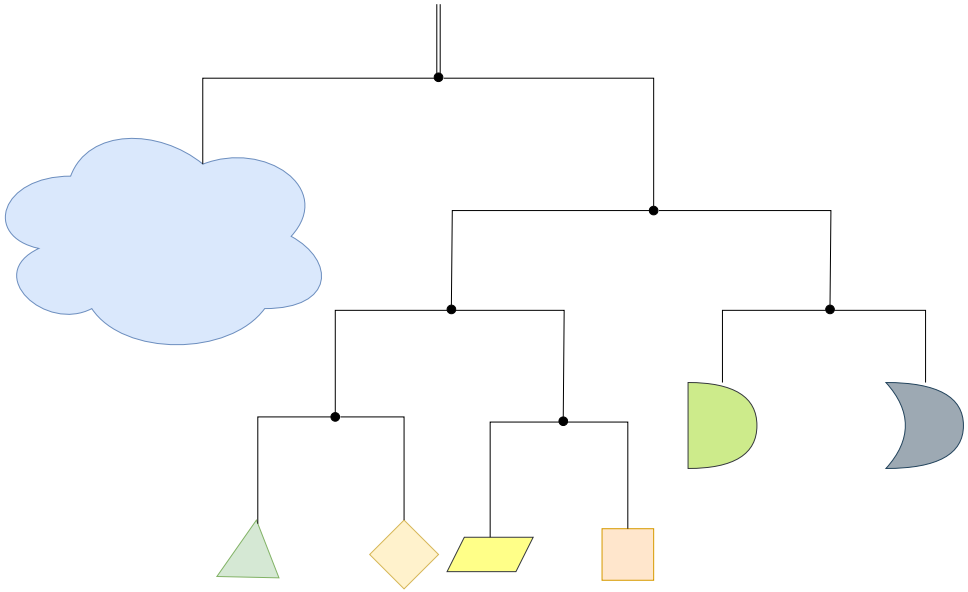
Vereinbarung
„1“ – nach Links
„0“ – nach Rechts

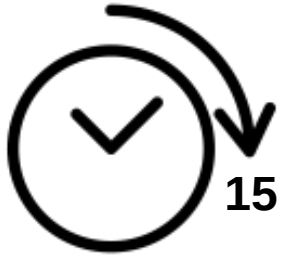
```
6x1+2x3+1x4+1x4+1x4+2x3+2x4 = 38Bit
```


Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Komprimierung Übung

Übung zum Huffman-Verfahren





15 Minuten

Auftrag: Komprimiere die Sätze

a) „**Liebe_geht_durch_den_Magen**“ (mit weniger als 104 Bit)

b) „**EBER_ESSEN_EHER_SELTEN_EIER**“ (mit weniger als 108 Bit)

Zechnall (September 2021)
Schiebel (Oktober 2023)

Huffman-Code: Fazit

Der Huffman-Code ist präfixfrei, weil kein Symbol in einem inneren Knoten des Decodierungsbaums steht.

Im Gegensatz dazu: Morse