

# Modul 114

Thema 7/11

Speicherplatz als rares Gut – Kompression

# Agenda

2

Thema	Inhalte
1	Zahlensysteme BIN - DEZ - HEX
2	Arithmetische und logische Grundoperationen im Binärsystem
3	Die Logik und den Prozessor verstehen
4	Grosse Zahlen in kleinen Variablen ablegen
5	Fehler in der Datenübertragung finden und korrigieren
6	Speicherplatz als rares Gut - Dateien und ihr Platzbedarf
7	Speicherplatz als rares Gut - Kompression
8	Speicherplatz als rares Gut - Reduktion
9	Vektorgrafiken - Eine Alternative zu den Pixeln
10	Verschlüsselung - Geschichte und Grundsätzliches
11	Verschlüsselung - Moderne Verfahren



# Tagesziele

3

Ich kann...

- den Unterschied zwischen verlustfreier und verlustbehafteter Kompression erklären.
- die Huffman-Codierung auf einfache Texte anwenden.
- den Kompressionsfaktor und die Kompressionsrate berechnen.



# Kompression allgemein

# Kompression: verlustfrei...

5



...oder verlustbehaftet

6



# Grundsätzliches

7

- Wikipedia:** Die Datenkompression oder Datenkomprimierung ist ein Vorgang, bei dem die Menge digitaler Daten reduziert wird.
- Ziele:**
- Speicherplatz wird eingespart
  - Übertragungszeit wird kürzer
- Grundsatz:** Es wird versucht, überflüssige Informationen zu vermeiden oder zu entfernen.
- Zwei Arten:**
- Verlustfreie Kompression**  
Es werden **Redundanzen** ausgenutzt, es geht keine Information verloren.
- Verlustbehaftete Kompression**  
Es wird **«irrelevante» Information** weggelassen  
(Bild-, Ton- und Videoübertragung)



# Verlustfreie Kompression



# Lösungsansätze für Binärwerte

9

## Laufängencodierung

Gibt an, wie viele Nullen oder Einsen hintereinander kommen und versucht so, Platz zu sparen. Diese Methode bewährt sich nur in Spezialfällen!

## Wörterbuchverfahren

Häufig auftretende Muster werden identifiziert und durch einen kurzen Code ersetzt. Bei gleichartigen Mustern sehr effizient.

## Huffman-Code

Es wird ein für die zu komprimierende Datei optimierter Code erstellt. Dieser Code codiert häufig vorkommende Zeichen möglichst kurz und kommt ohne Trennzeichen aus. Er hat sich stark durchgesetzt.



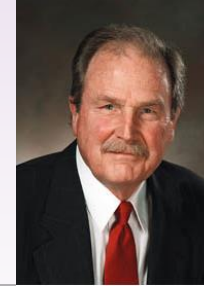
# Welche Bytes kommen wie oft vor?

10

01100111	11111100	01111101	01111111	10110011	10010101	11101000	10011110	11101111	10100000	10010111	00100001	00010111	01000011	00011001
11100100	10011100	01100010	01011111	11010011	10001100	10000101	01110101	10010000	01010111	11110000	10111110	10110010	10110101	10001101
11001011	01010001	11100010	01010111	11000101	10001011	10000101	01000111	00011110	01001101	00000010	11100100	00011010	10010010	10000010
00100011	00011111	00001000	01011011	01001111	10100101	01111001	00111001	11001011	01101010	01000110	11011111	10010011	10000000	01100110
10111001	01010010	01100011	00000100	00001011	10011100	00101000	00010110	00111010	10000100	10011001	11010100	10001101	01100111	00000101
00100011	11100010	10000111	00011000	10011101	01111011	01011010	10001100	10101010	11101100	11100001	10100100	01001000	00110111	00100001
01000101	01010001	00100101	11110001	00110111	00100100	11010110	10101100	01011111	00111001	01100111	11111100	01111101	01111101	11011001
11001010	11101000	10011110	11101111	10100000	10010111	00100001	00010111	01000011	00011100	11100010	10011100	01100010	01011111	11010011
10001100	10001101	01110101	10010000	01011011	01110000	10111110	10110010	10110101	10011001	11001011	01001001	11100010	01011011	11000101
10001011	01000011	01000111	00011110	01001101	00000010	11100100	00011010	10000010	10000010	00100011	00011111	00001000	01011011	01001111
11010010	01111001	00111001	11001011	01101010	01000110	11011111	10010011	10000000	01100110	10111001	01010010	01100011	00000100	00001011
10011100	00101000	00010110	00111010	10000100	10011001	11010100	10001101	01100111	00000101	00100011	11010010	10000111	00011000	10011101
01111011	01011010	10001100	10101010	11101100	11100001	10100100	01001000	00110111	00100001	01000101	01010001	00100101	11110001	00110111
00100100	11010110	10101100	01011111	00111001	01111001	11111100	01111101	01111011	11011001	11010010	11101000	10011110	11101111	10100000
10010111	00100001	00010111	01000011	00011100	11100010	10011100	01100010	01011111	11010011	10001100	10001101	01110101	10010000	01011011
01110000	10111110	10110010	10110101	10011001	11001011	01001001	11100010	01011011	11000101	10001101	01000011	01000111	00011110	01001101
00000010	11100100	00010101	10010010	10000010	00100011	00011111	00001000	01011011	01001111	11001011	01111001	00111001	11001011	01010101
01000110	11101111	10010011	10000000	01100110	10111001	01010010	01100011	00000100	00001011	10011100	00101000	00010110	00111010	10000100
10011001	10101000	10001101	01100111	00000101	00100011	11010010	10000111	00011000	10011101	01111011	01011010	10001100	10101010	11101100
11100001	10100100	01001000	00110111	00100001	01000101	01010001	00100101	11110001	00110111	00100100	11010110	10101100	01011111	00111001
01100111	11111100	01111101	01111101	11011001	11001010	11101000	10011110	111011						



# Huffman-Code



11

## Huffman – Codierung

Mit dem Huffman-Algorithmus wird ein Code generiert, welcher für den vorliegenden Text optimal gestaltet ist:

- Häufige Zeichen → kürzester Code
- Seltene Zeichen → längerer Code
- Aus allen Zeichen-Codes wird ein eindeutiger Bitstrom (keine Trennzeichen), d.h. präfix-frei
- Liefert einen von meist mehreren optimalen Codes



# Huffman-Code

12

## Anleitung zur Huffman – Codierung

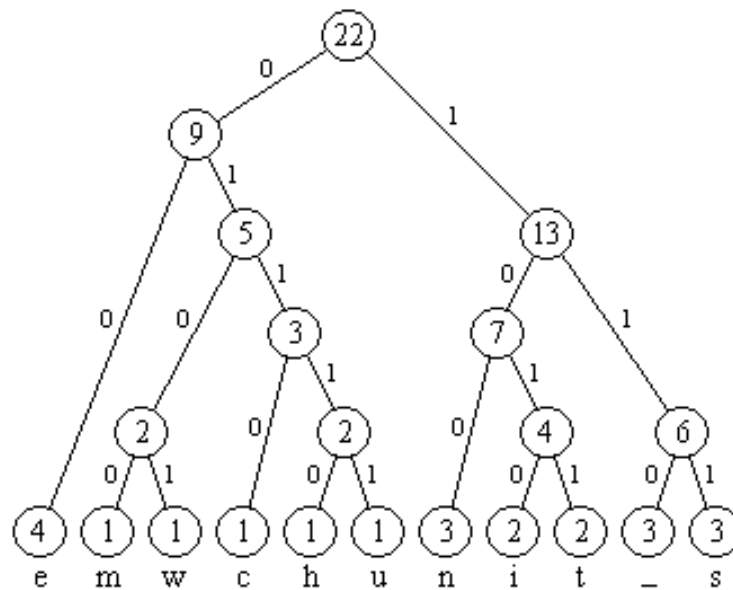
Zu komprimierender Text: **IM WESTEN NICHTS NEUES**

1. Schreibe jedes vorkommende Zeichen unten auf ein Blatt.
2. Schreibe über jedes Zeichen seine Häufigkeit.
3. Verbinde immer die beiden tiefsten (freien) Häufigkeitswerte (oder Knoten nach oben zu einem Summenknoten.
4. Wiederhole Schritt 3 bis Du den Stammknoten gebildet hast (Totalsumme)
5. Male unter jeden Knoten links eine null und rechts eine eins.
6. Nun kannst Du den Binärcode jedes Zeichens vom Stammknoten her ablesen.



# Huffman-Code

13



E	00
M	0100
W	0101
C	0110
H	01110
U	01111
N	100
I	1010
T	1011
[ ]	110
S	111

Codierter Bitstrom (73 Bit):

101001001100101001111011001001101001010011001110101111110100000111100111



# Kompressionsfaktor und Kompressionsrate

# Kompressionsfaktor und -rate

15

Kompressionsfaktor:

Wie gross ist der Kompressionsfaktor, wenn wir das Resultat unserer Huffman-Codierung mit dem Originaltext in Unicode vergleichen?

Lösung:

$$KF = \frac{73 \text{ Bit}}{22 \cdot 16 \text{ Bit}} = 0.207 = 20.7\%$$

Merke: Die Kompressionsrate ist der Kehrwert des Kompressionsfaktors!

$$KF = \frac{\text{komprimierte Grösse}}{\text{ursprüngliche Grösse}} \quad KR = \frac{\text{ursprüngliche Grösse}}{\text{komprimierte Grösse}}$$



# Einsatzgebiete Huffman-Code

16

- JPEG
- ZIP (Implode)
- MPEG

Und viele weitere...





# Übungsaufgaben



17

- › Das Gelernte können Sie mit Hilfe von AB 114-07 üben

**Ziel:** Repetition und Vertiefung des Stoffes  
**SF:** Einzelarbeit/Partnerarbeit  
**Zeit:** 45 Minuten



# Abschluss



18

- › **Offene Punkte / Fragen**
- › **Feedback**
- › **Hausaufgaben**
  - Arbeitsblatt AB114-07 fertig lösen

