

Datenkomprimierung (u. -expansion)

Kriterien, nach denen Daten komprimiert werden:

- Zeit
- Speicherplatz

Im Augenblick besitzen Daten einen hohen Grad von **Redundanz** (=Weitschweifigkeit)

Lauf längencodierung

Einfachster Typ der Redundanz sind lange Folgen sich wiederholender Zeichen, sog. **Läufe** (runs).

Codierung variabler Lauf längen

Codierung von Buchstaben mit 5 Bits unter Verwendung des Index i des Buchstabens innerhalb des Alphabets:

A	B	R	A	C	...
00001	00010	10010	00001	00011	...

Feste Länge der Zahlen ist Voraussetzung!

Kodierung mit variabler Länge:

Platzeinsparung dadurch, dass häufig verwendete Zeichen mit möglichst wenig Bits verschlüsselt werden.

Bsp.: A=>0 , B=>1 , R=>01 , C=>10 , D=>11

ABRACADABRA => 0|1|01|0|11|0|11|0|1|01|0

Bei diesem Code ist es notwendig, dass ein Begrenzer, ein zusätzliches Zeichen, den Code zwischen zwei Zeichen abtrennt.

Andere Überlegung:

Code ersetzen, bei dem kein Zeichencode mit dem Anfang eines anderen übereinstimmt => spart den Begrenzer

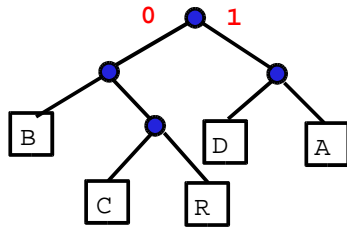
Bsp.: A=>11, B=>00, C=>010, D=10, R=>011

=> codierte Zeichenfolge:

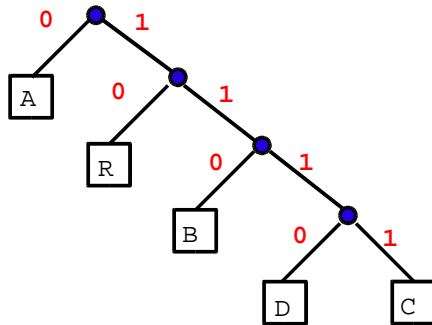
1100011110101110110001111

(ohne Begrenzer möglich)

=> einfachste und zugleich wirksame Methode zur Kodierung ist der **TRIE**.



oder:



=> 01101001111011100110100

Erzeugung des Huffman-Codes

1952 von D. Huffman (franz. Math.) entwickelt.

Schritte zur Erzeugung:

- 1) Ermittlung der Häufigkeit des Auftretens der verwendeten Zeichen
- 2) Aufbau des Kodierungs-Tries nach der Häufigkeit der Zeichen.

(während der Erzeugung des Tries betrachtet man ihn als Binärbaum mit Häufigkeiten, die in den Knoten gespeichert sind)

Es werden jeweils die beiden Knoten (Teil-Tries) mit den geringsten Häufigkeiten ausgewählt, aus denen ein neuer Knoten mit der Summe der beiden Häufigkeiten gebildet wird.

Nachfolger dieses Knotens sind die beiden Ausgangsknoten (Teil-Tries).

- 3) Der Hoffman-Code wird aus den Tries abgeleitet, indem man den Trie von der Wurzel bis zu einem Blatt (codiertes Zeichen) durchläuft und dabei nach "links" den Wert **0** und nach "rechts" den Wert **1** codiert.

Bsp.: Datei: EIN_SONNIGES_WOCHENENDE

Hoffman-Code: 01 111 10 0010 0011 0000 10 10 111 1101 01 0011 0010
11000 0000 11001 00010 01 10 01 10 00011 01


1.

Zeichen	Häufigkeit
E	5
I	2
N	5
—	2
S	2
O	2
G	1
W	1
C	1
H	1
D	1



2.

Z	H
E	5
N	5
I	2
—	2
S	2
O	2
G	1
W	1
C	1
H	1
D	1



3.

Z	H
E	5
N	5
I	2
—	2
S	2
O	2
G	1
W	1
C	1
	2

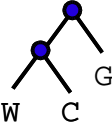

4.

Z	H
E	5
N	5
I	2
—	2
S	2
O	2
	2
	2
G	1

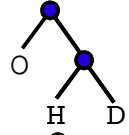
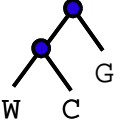
5.

Z	H
E	5
N	5
I	2
—	2
S	2
O	2
	2
	3

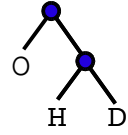

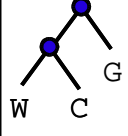
6.

Z	H
E	5
N	5
	3
I	2
—	2
S	2
O	2
	2

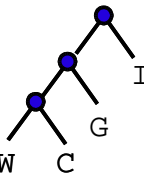
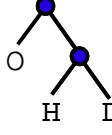

7.

Z	H
E	5
N	5
	4
	3
I	2
-	2
S	2

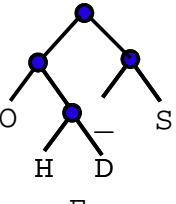
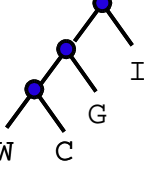
8.

Z	H
E	5
N	5
	4
	4
	3
I	2

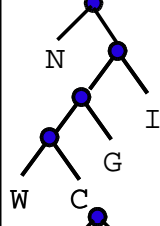
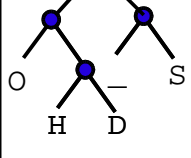
9.

Z	H
E	5
N	5
	5
	4
	4

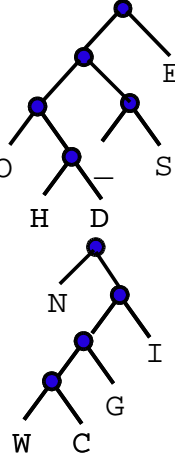
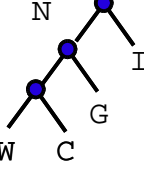
10.

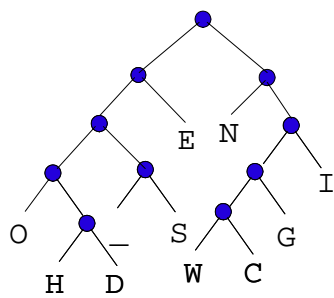
Z	H
	8
	5

11.

Z	H
	10
	8
E	5

12.

Z	H
	13
	8



E: 01
 N: 10
 I: 111
 _: 0010
 S: 0011
 O: 0000

beim Codieren sind die Zeichen
 mit der größten Häufigkeit am
 weitesten oben

=> kürzester Code

G: 1101
 W: 11000
 C: 11001
 H: 00010
 D: 00011