Dezimal, Zweierkomplement & Exzess:

|  |  |  |
| --- | --- | --- |
| **Dezimal** | **Zweierkomplement** | **Exzess\*** |
| 127 | 0111 1111 | 1111 1111 |
| 0 | 0000 0000 | 1000 0000 |
| -1 | 1111 1111 | 0111 1111 |
| -128 | 1000 0000 | 0000 0000 |

\*Exzess = Hälfte des ganzen Bereichs = 2n-1

Gleitkommazahlen:

6 Billionen = 6'000'000'000'000 = 6 \* 1012 = 60 \* 1011 Bsp: 32 Bit

\/

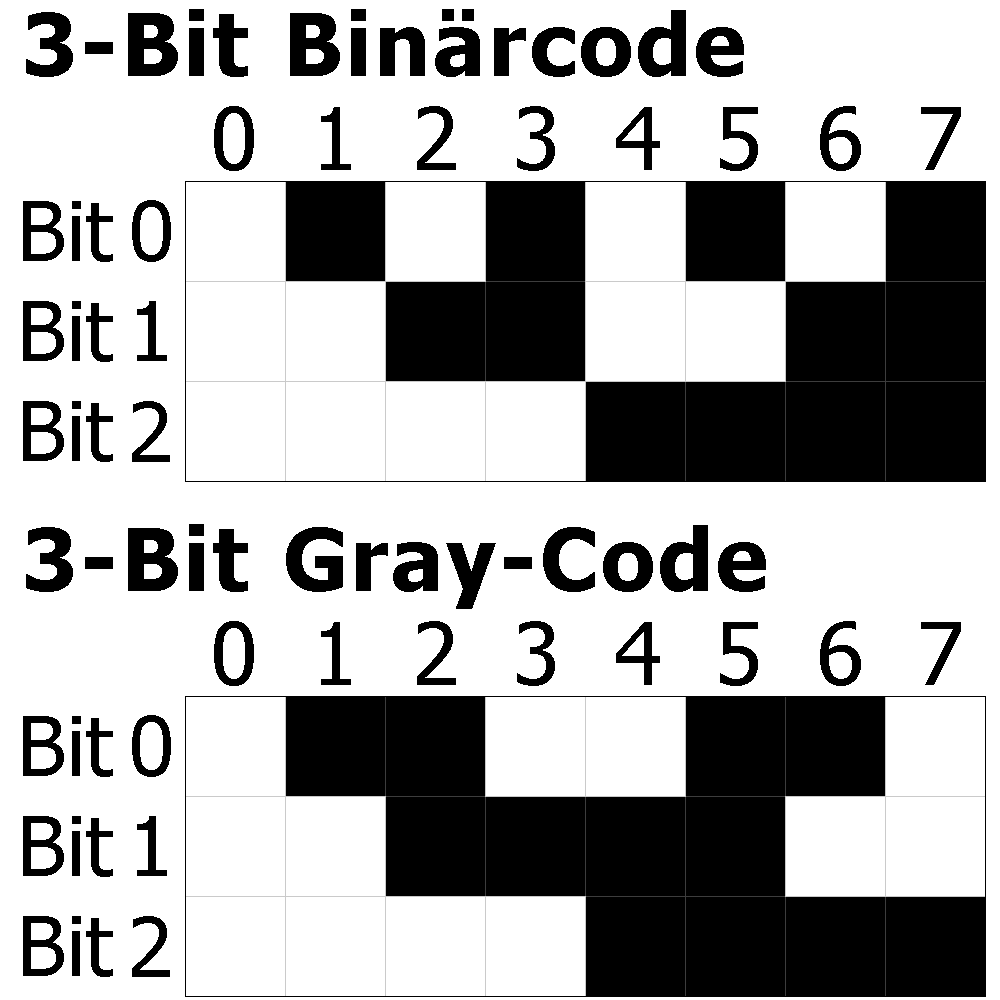
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

/\|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

Vorzeichen | |

Exponent «12» Mantisse «6»

Gray-Code:

nur ein Bit wechselt pro Übergang

EAN-Code:

Barcode, Strichcode

Artikelnummerierung im Detailhandel

ASCII-Code:

7 Bit = 128 Zeichen, Standard ASCII (American Standard Code for Information Interchange)

8 Bit = 256 Zeichen, Extended ASCII

Unicode:

Für ALLE möglichen Zeichen , UTF-8, UTF-16, UTF-32 (UTF = UCS Transformation Format, UCS = Universal Coded Character Set)

Hammingdistanz (HD):

maus // haus 0110 // 0101

|\_\_\_\_\_\_| |\_\_\_\_\_\_\_|

| |

HD = 1 HD = 2

= Anzahl unterschiedlicher Zeichen/Bits zweier Zeichenketten

Paritätsbit:

b: 0110 0010 1 --+ - Even Parity =

c: 0110 0011 0 --| gerade Zahl Einsen

HD = 2 - Odd Parity =

ungerade Zahl Einsen

Fehler: 0111 0010 1

Redundanzen berechnen:

log2 6 => 2? = 6

= Taschenrechner 2nd, logyx

Formel: log2 6 = log10 6 / log10 2 = 0.778 / 0.301 = 2.58 ~ 2.6 Bit

| |

Endzahl |

Basis

Redundanz: 3 Bit – 2.6 Bit = 0.4 Bit

Hamming-Code:

Fehlerkorrektur für 1 Bitfehler -> Übung 5

CRC-Prüfsumme:

CRC = Cyclic Redundancy Check

111010011100100000 / 100110 = ...(uninteressant)

100110 |

111000 +-----> Generator Polynom (x5+x2+x1)

100110

111101 -> versendet wird

100110 111010011100100100

110111 |

100110 |

100011 |

100110 |

101001 |

100110 |

111100 |

100110 |

110100 |

100110 |

100100 |

100110 |

100------------------------------+

Datenkomprimierung

| |

verlustfrei <---+ +---> verlustbehaftet

+------------------+ +-------------+

| |

z.B.: zip, rar, 7-zip, png z.B. mp3, mpeg4

* Huffman
* Lempel-Ziv LZ77
* Lempel-Ziv-Welch LZ78
* RLE: Runlength Encoding

Lauflängenkodierung

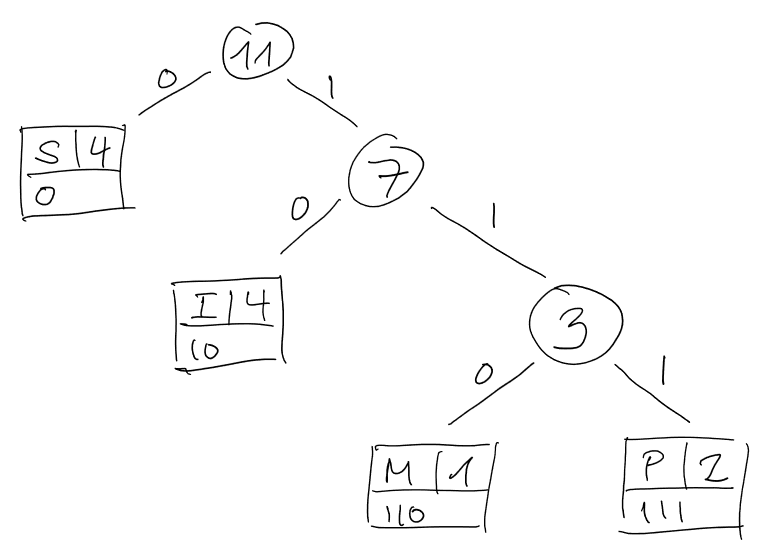
* BWT: Burrow Wheeler Transformation

Huffman-Baum:

MISSISSIPPI

|  |  |  |  |
| --- | --- | --- | --- |
| M | I | S | P |
| 1 | 4 | 4 | 2 |

Häufigkeitsanalyse



11010001000101111110

21 Bit

ASCII -> 88 Bit (11\*8)

Einsparung:

88 - 21 = 67 Bit Einsparung

LZ77 (Lempel-Ziv 1977):

+-> LZMA (LZ Markov): 7zip, XZ (.tar.xz in Unix)

| |

LZ77 ---+ +---> sehr hohe Komprimierungsrate

|

+-> LZSS (LZ Storer Szymanski): WinRAR

|

+-> DEFLATE: Phil Katz, Kombination LZSS + Huffman: Zip, PNG, GZIP

LZ78 -----> LZW (LZ Welch): Basis für GIF-Bilder

Huffman: MP3, bzip2

BWT (1994): bzip2, GZIP, PKZIP

Beispiel LZ77: BANANE

Puffer Vorschaufenster Code

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2  B | 3  A | 4  N | 5  B  A | 6  B  A  N | 7  B  A  N  E |  | B  A  N  A | A  N  A  N | N  A  N  E | A  N  E |  |  | (0, 0, B)  (0, 0, A)  (0, 0, N)  (6, 2, E)  (-, -, -) |

(0, 0, B)(0, 0, A)( 0, 0, N)(6, 2, E)(-, -, -)

Rekonstruktion:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2  B  - | 3  A  - | 4  N  - | 5  B  A  - | 6  B  A  N  - | 7  B  A  N  E  - |  | (0, 0, B)  (0, 0, A)  (0, 0, N)  (6, 2, E)  (-, -, -) |

Beispiel LZ78: BAUMSAUMRAUM = 1 2 3 4 5 8 4 6 12

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Wörterbuch | letztes Wort | akt. Wort | WB-Eintrag | Code |
| B=1 |  | B |  |  |
| A=2 | B | A | BA=7 | 1 |
| U=3 | A | U | AU=8 | 2 |
| M=4 | U | M | UM=9 | 3 |
| S=5 | M | S | MS=10 | 4 |
| R=6 | S | A | SA=11 | 5 |
| BA=7 | A | U |  |  |
| AU=8 | AU | M | AUM=12 | 8 |
| UM=9 | M | R | MR=13 | 4 |
| MS=10 | R | A | RA=14 | 6 |
| SA=11 | A | U |  |  |
| AUM=12 | AU | M |  |  |
| MR=13 | AUM | EoF |  | 12 |
| RA=14 |  |  |  |  |

Burrow-Wheeler-Transformation (BWT):

* Keine Komprimierung, sondern Vorbereitung für RLE

|

Run Length Encoding

Lauflängenkodierung

Beispiel: ANANAS$



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A | N | A | N | A | S | $ |
| N | A | N | A | S | $ | A |
| A | N | A | S | $ | A | N |
| N | A | S | $ | A | N | A |
| A | S | $ | A | N | A | N |
| S | $ | A | N | A | N | A |
| $ | A | N | A | N | A | S |

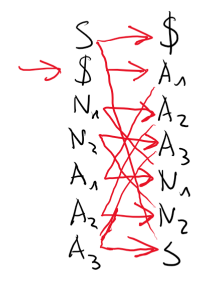
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| $ | A | N | A | N | A | S |
| A | N | A | N | A | A | $ |
| A | N | A | S | $ | A | N |
| A | S | $ | A | N | A | N |
| N | A | N | A | S | $ | A |
| N | A | S | $ | A | N | A |
| S | $ | A | N | A | N | A |

S$NNAAA = S$N2A$

| |

BWT RLE

S $ A1N1A2N2A3S$

$ A1

N1 A2

N2 A3

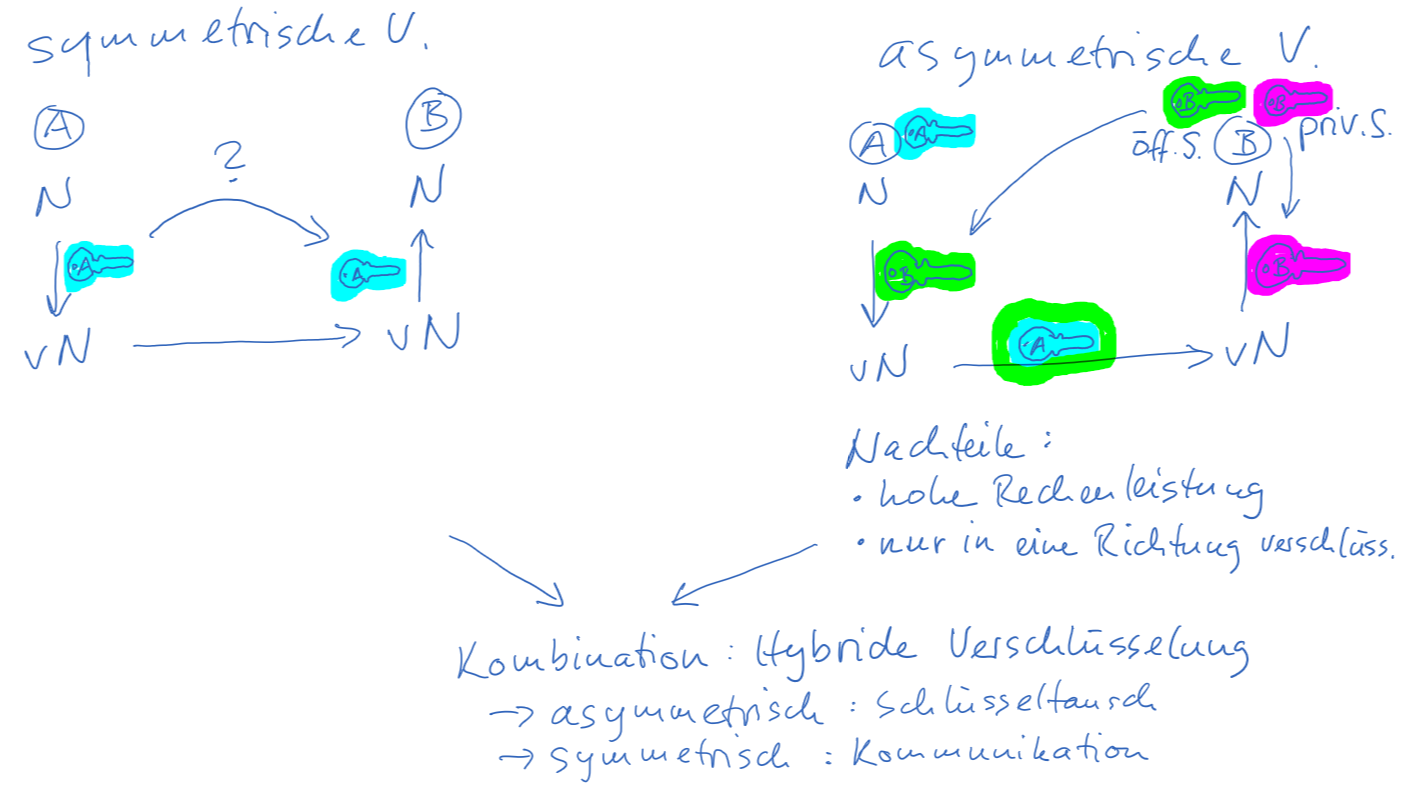
A1 N1

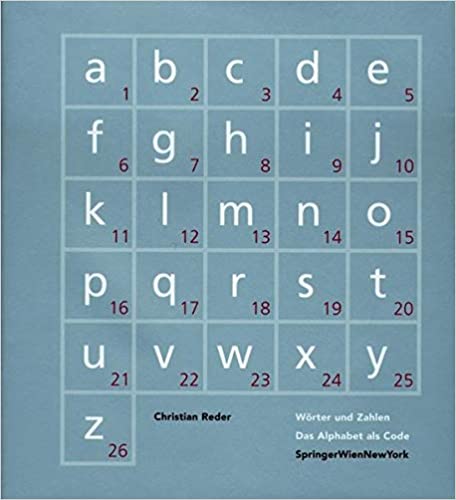
A2 N2

A3 S

Verschlüsselung

* Symmetrische Verschlüsselung
* Asymmetrische Verschlüsselung



Caesarchiffre

* Buchstaben verschieben

Bsp.: +2, hallo --> jbnnq

Vigenèrechiffre

* Variable Verschiebung durch «Passwort»

Bsp: h a l l o , Passwort «j a»

8 1 12 12 15 10 1

10 1 10 1 10

18 2 22 13 25

r b v m y

Vernam

* Gleich vie Vigenèrechiffte, nur das der Schlüssel gleich lang ist, wie die Nachricht

Bsp.: h a l l o , Passwort «v o g e l »

8 1 12 12 15 22 15 7 5 12

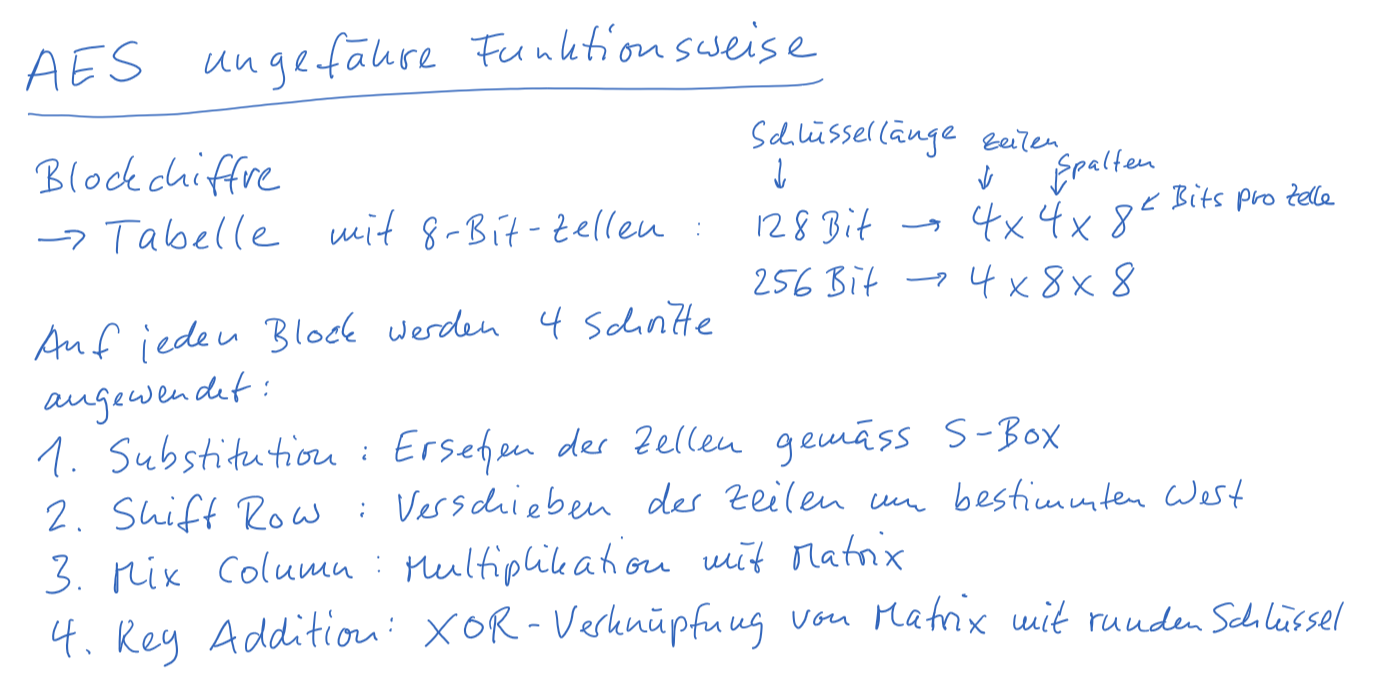
22 15 7 5 12

30 16 19 17 27

d p s q a

Verschlüsselungsalgorithmen

|  |  |  |
| --- | --- | --- |
|  | früher | heute |
| symetrische V. | DES  Data Encryption Standard | AES  Advanced Encryption Standard |
| asymetrische V. | DH  Diffie-Hellman | RSA  Rivest-Shamir-Adleman |
| hybride V. | SSL  Secure Sockets Layer | TLS  Transport Layer Security |
| [Hash](https://de.wikipedia.org/wiki/Hashfunktion) |  | SHA  Secure Hash Algorithm |
| Email |  | PGP (OpenPGP)  Pretty Good Privacy  S/MINE  Secure Multipurpose Internet Mail Extensions |
| Public Key Infrastructure | | PKI |



PKI Grundlagen

