# Digitale Codierungen | DigCod

Zusammenfassung

## Mathematische Grundlagen

### Zahlensysteme

Jede Stelle der Zahl hat den Wert der entsprechenden Potenz, zum Beispiel 16 (Hexadezimal). Die rechte Ziffer entspricht 16º=1, die zweite von rechts 16¹=16 usw. Nimm jede Ziffer bzw. ihren Zahlenwert (A=10, B=11, ... ) mal mit der entsprechenden Potenz und summiere.

* + 1. Bit

Ein Bit ist eine Grösse, die zwei Zustände annehmen kann. Welchem Zustand und welchem entspricht, hängt dabei von Hersteller, Standards, Konventionen usw. ab.

* Drei oder mehr Zustände sind technisch aufwändiger zu realisieren und zu verarbeiten
* Direkter Zusammenhant mit klassischer (boolescher) Logik (wahr/falsch)
* Einfachste Entscheidungen im Programmfluss sind binär (ja/nein)

Das Binärsystem bildet die Basis aller gängigen Computersysteme.

* + 1. Binäres Zahlensystem

Im binären Zahlensystem ist die Basis und die Ziffern sind

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | 1/2 0.5 | 1/4 0.25 | 1/8 0.125 | 1/16 0.0625 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

* Table

  Description automatically generatedBit = 1: Gesetztes Bit (set Bit)
* Bit = 0: Gelöschtes Bit (cleared bit)
* LSB: Least Significant Bit, niedrigstwertiges Bit, Bit
* MSB: Most significant Bit, höchstwertigstes Bit, Bit in einer -stelligen Zahl
* Nibble: Binärzahl mit 4 Bit. Grössere Binärzahlen werden als Nibble gruppiert, z.B.
* Oktett/Byte: Eine Binärzahl mit 8 Bit oder zwei Nibbles, z.B.
* Notation: Die Bits einer Binärzahl werden häufig durch Indizes einzeln berechnet. Zusammenhängende Bereiche von Bits in einer Binärzahl werden «/» bezeichnet.

Minimal benötigte Bits zur Adressierung:

Beispiel: Wieviele Bits benötigt man, um 20MB Speicher adressieren zu können?

Bit

* + 1. Hexadezimales Zahlensystem

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **16^0** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| **16^1** | 16 | 32 | 48 | 64 | 80 | 96 | 112 | 128 | 144 | 160 | 176 | 192 | 208 | 224 | 240 |
| **16^2** | 256 | 512 | 768 | 1024 | 1280 | 1536 | 1792 | 2048 | 2304 | 2560 | 2816 | 3072 | 3328 | 3584 | 3840 |
| **16^3** | 4096 | 8192 | 12 288 | 16 384 | 20 480 | 24 576 | 28 672 | 32 768 | 36 864 | 40 960 | 45 056 | 49 152 | 53 248 | 57 344 | 61 440 |

Sonst einfach Zahl / 16, Rest ergibt den hintersten HEX Wert, dann geteilte Zahl / 16 etc.

|  |  |  |
| --- | --- | --- |
| **Decimal** | **Binary** | **Hexadecimal** |
| 0 | 0000 | 0 |
| 1 | 0001 | 1 |
| 2 | 0010 | 2 |
| 3 | 0011 | 3 |
| 4 | 0100 | 4 |
| 5 | 0101 | 5 |
| 6 | 0110 | 6 |
| 7 | 0111 | 7 |
| 8 | 1000 | 8 |
| 9 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |

### Aussagenlogik

Eine Aussage ist ein feststellender Satz, dem eindeutig einer der beiden Wahrheitswerte wahr oder falsch zugeordnet werden kann.»

Auch wenn der Wahrheitswert unbekannt ist, können Behauptungen Aussagen sein, wenn sie im Prinzip beweis- oder widerlegbar sind. Auch Behauptungen über die Zukunft können Aussagen sein. Für Aussagen werden oft Symbole (z.B. A, B, C, …) verwendet.  
Wenn die Behauptung in eine «Wenn <Behauptung>, dann …» Form gebracht werden kann, handelt es sich um eine Aussage.

* + 1. **Junktoren**

|  |  |
| --- | --- |
| **¬** Negation nicht **∧** Konjunktion und **∨** Disjunktion oder | **⇒** Implikation wenn … dann …**¬** **⇔** Äquivalenz … genau dann, wenn … |

* + 1. **Negation / Verneinung / ¬**

Die Negation einer Aussage ist genau dann wahr, wenn die Aussage falsch ist.

* Heute regnet es ⇒ Heute regnet es nicht
* x > 1000 ⇒ x ≤ 1000
* A: x < 7 ⇒ ¬A: ¬(x<7) oder ¬A: x≥7

|  |  |  |
| --- | --- | --- |
| A | ¬A | ¬(¬A) |
| w | f | w |
| f | w | f |

* + 1. Konjunktion / und / ∧

Die Konjunktion zweier Aussagen ist genau dann wahr, wenn beide Aussagen wahr sind.

* A: Heute regnet es. B: Die Lufttemperatur ist unter null. ⇒ A ∧ B: Heute regnet es und die Lufttemperatur ist unter null.
* A: x < 3, B: y > 5 ⇒ A ∧ B: x < 3 und y > 5

|  |  |  |
| --- | --- | --- |
| A | B | A ∧ B |
| w | w | w |
| w | f | f |
| f | w | f |
| f | f | f |

Achtung: Mit der Konjunktion wird kein kausaler oder zeitlicher Zusammenhang berücksichtigt.   
**In der Aussagenlogik gilt A ∧ B ⇔ B ∧ A**

* + 1. **Disjunktion / (einschliessliches) ODER / ∨**

Die Disjunktion zweier Aussagen ist genau dann wahr, wenn mindestens eine der beiden Aussagen wahr ist.

* A: Heute regnet es. B: Die Lufttemperatur ist unter null. 🡪 A ∨ B: Heute regnet oder und die Lufttemperatur ist unter null.
* A: x < 3, B: y > 5 🡪 A ∨ B: x < 3 oder y > 5 oder beides

|  |  |  |
| --- | --- | --- |
| A | B | A ∨ B |
| w | w | w |
| w | f | w |
| f | w | w |
| f | f | f |

Eine Kontradiktion ist eine logische Aussage, die nie wahr ist. Alle Zeilen der Wahrheitstafel ergeben falsch. Eine Tautologie ist eine logische Aussage, die immer wahr ist. Alle Zeilen der Wahrheitstafel ergeben wahr.

* + 1. **Implikation / wenn x, dann y /** ⇒

Wenn die Aussage A wahr ist, dann gilt auch die Aussage B. Aus A folgt B.   
A ist hinreichend für B. B ist notwendig für A. A ist Voraussetzung / Annahme, B ist die Folgerung.

|  |  |  |
| --- | --- | --- |
| A | B | A ⇒ B |
| w | w | w |
| w | f | f |
| f | w | w |
| f | f | w |

Abtrennungsregel: Wenn A wahr ist, und A ⇒ B wahr ist, dann ist auch B wahr. (A ∧ (A ⇒ B)) ⇒ B

* + 1. **Äquivalenz / y genau dann, wenn x /** ⇔

Aussage A gilt genau dann, wenn B gilt. Es gilt A ⇒ B und B ⇒ A

|  |  |  |
| --- | --- | --- |
| A | B | A ⇔ B |
| w | w | w |
| w | f | f |
| f | w | f |
| f | f | w |

Hinweis: ein senkrechter Strich *x | y* bedeutet *x ist ein Teiler von y*

* + 1. Bindungsstärke

Die Bindungsstärke legt fest, welche Junktoren in aussagenlogischen Formeln zuerst ausgeführt werden. Das Nicht hat die grösste Bindungsstärke, gefolgt von den beiden Junktoren Und und Oder. Die kleinste Bindungsstärke besitzen die Implikationen und die Äquivalenz

### Aussagenlogische Formeln

A und B seien Aussagen. Die verknüpften Aussagen ¬A, A ∧ B, A ∨ B, A ⇒ B und A ⇔ B sind aussagenlogische Formeln. Die verknüpften Aussagen sind auch aussagenlogische Formeln, wenn A oder B bereits selbst aussagenlogische Formeln sind. Eine aussagenlogische Formel ist selbst wieder eine Aussage. Die logische Konstante W bedeutet «wahr» und F bedeutet «falsch».

Satz 1: A, B, und C seien aussagenlogische Formeln. Dann gilt

1. Kommutativität: A ∧ B ⇔ B ∧ A, A ∨ B ⇔ B ∨ A  
Wenn A und B zutreffen, gilt auch B und A. Wenn A oder B zutreffen, gilt auch B oder A (umgekehrt)

2. Assoziativität: A ∧ (B ∧ C) ⇔ (A ∧ B) ∧ C, A ∨ (B ∨ C) ⇔ (A ∨ B) ∨ C

Wenn A und (B und C) zutreffen, können die Klammern beliebig verschoben oder weggelassen werden (gleich bei oder)

3. Distributivität: A ∧ (B ∨ C) ⇔ (A ∧ B) ∨ (A ∧ C), A ∨ (B ∧ C) ⇔ (A ∨ B) ∧ (A ∨ C)  
A und (B oder C) ist das gleiche wie (A und B) oder (A und C), A oder (B und C) ist das gleiche wie (A oder B) und (A oder C)

4. Absorption: A ∨ (A ∧ B) ⇔ A, A ∧ (A ∨ B) ⇔ A  
A oder (A und B) ist das Gleiche wie A. A und (A oder B) ist das Gleiche wie A

5. Idempotenz: A V A = A, A ∧ A = A  
A und A = A, A oder A = A

6. Doppelte Negation: ¬(¬A) ⇔ ¬¬A ⇔ A

Nicht nicht A ist gleich A

7. Konstanten: W = wahr, F = falsch. A ∨ W ⇔ W, A ∧ W ⇔ A, A ∨ ¬A ⇔ W  
A oder wahr = wahr, A und wahr = A, A oder nicht A = wahr

Implikation ersetzen: A ⇒ B kann man durch ¬A ∨ B ersetzen

Äquivalenz ersetzen: A ⇔ B kann man durch (A ∧ B) ∨ (¬A ∧ ¬B) ersetzen

Satz 2: A, B und C seien aussagenlogische Formeln. Dann bedeutet A ⇒ B ⇒ C, dass beide Implikationen, sowohl A ⇒ B als auch B ⇒ C gelten. Formal kann man also schreiben: (A ⇒ B ⇒ C) ⇔ (A ⇒ B) ∧ (B ⇒ C)

Satz 3 (de Morgan): Für beliebige Aussagen A und B gilt:

a) ¬(A ∧ B) ⇔ ¬A ∨ ¬B Nicht (A und B) ist das gleiche wie nicht A oder nicht B

b) ¬(A ∨ B) ⇔ ¬ A ∧ ¬B Nicht (A oder B) ist das gleiche wie nicht A und nicht B

### Normalformen

Dienen der Übersichtlichkeit – standardisierte Form

* + 1. Negationsnormalform

Eine aussagenlogische Formel steht in Negationsnormalform, wenn die Negation (¬) nur direkt vor Aussagen oder vor Konstanten steht.  
Beispiel: ¬(A ∨ B) ist keine Negationsnormalform, ¬A ∧ ¬B hingegen schon

* + 1. Verallgemeinerte Konjunktion

Eine verallgemeinerte Konjunktion ist…

* Eine einzelne Aussage oder seine Negation (also A oder ¬A) oder
* Einer der logischen Konstante T=wahr oder F=falsch oder
* Eine Konjunktion A ∧ B, falls A und B selbst verallgemeinerte Konjunktionen sind

Eine Verbindung von Aussagen, Negationen und logischen Konstanten mit «und».   
Liegt immer in **Negationsnormalform** vor.

* + 1. Disjunktive Normalform

Eine Aussage liegt in disjunktiver Normalform vor, wenn sie eine **verallgemeinerte Konjunktion** ist, oder wenn sie eine Disjunktion von verallgemeinerten Konjunktionen ist.

Eine Verbindung von **verallgemeinerten Konjunktionen** mit «oder».

* + 1. Verallgemeinerte Disjunktion

Eine verallgemeinerte Disjunktion ist…

* Eine einzelne Aussage oder seine Negation (also A oder ¬A) oder
* Einer der logischen Konstante T=wahr oder F=falsch oder
* Eine Disjunktion A ∨ B, falls A und B selbst verallgemeinerte Disjunktionen sind
  + 1. Konjunktive Normalform

Eine Aussage liegt in konjunktiver Normalform vor, wenn sie eine **verallgemeinerte Disjunktion** ist, oder wenn sie eine Konjunktion von verallgemeinerten Disjunktionen ist.

Beispiel: Aussage A: (X ∧ ¬Y) ∨ (¬X ∧ (Y ∨ ¬ Z))

|  |  |  |  |
| --- | --- | --- | --- |
| X | Y | Z | A |
| w | w | w | **f** |
| w | w | f | **f** |
| w | f | w | **w** |
| w | f | f | **w** |
| f | w | w | **w** |
| f | w | f | **w** |
| f | f | w | **f** |
| f | f | f | **w** |

Um A in **disjunktiver Normalform** zu schreiben, werden die Zeilen der Tabelle ausgewertet, in denen A richtig ist. Um A in **konjunktiver Normalform** zu schreiben, werden die Zeilen der Tabelle ausgewertet, in denen A falsch ist.

Das heisst, die **konjunktive Normalform** wäre wie folgt:  
A ⇔ ¬ ((X ∧ Y ∧ Z) ∨ (X ∧ Y ∧ ¬Z) ∨ (¬X ∧ ¬Y ∧ Z))

Daraus lässt sich mit der Regel von de Morgan eine **Negationsnormalform** erstellen:

A ⇔ (¬X ∨ ¬Y ∨ ¬Z) ∧ (¬X ∨ ¬Y ∨ Z) ∧ (X ∨ Y ∨ ¬Z))

### Table Description automatically generatedBinäre Funktionen

**Table

Description automatically generated**

* Implikation:
* **A picture containing text, clock

  Description automatically generated**NAND (not and / nicht alle wahr):   
  Nur dann 0, wenn x=1 und y=1. Die Basisoperationen können ausschliesslich aus | dargestellt werden (Sheffer stroke). NAND-Bausteine sind technisch leicht als Transistoren zu realisieren.
* NOR ():Nur dann 1, wenn x = 0 und y = 0
* XOR: Exklsuive Diskunktion/Exklusives oder
* Ein Bild, das Screenshot, Zahl, Schrift, Reihe enthält.

  Automatisch generierte BeschreibungBasis der Addition: XOR bildet die Addition zweier Bits ab,   
  AND den Übertrag
* Literal: Variable oder Negation einer Variablen, bspw. (positives Literal) oder (negatives Literal)
* Konkunktionsterm: Konjunktion von Literalen, bspw.
* Ein Bild, das Schrift, Text, weiß, Typografie enthält.

  Automatisch generierte BeschreibungDiskunktionsterm: Disjunktion von Literalen, bspw.
* Minterm: Konjunktionsterm, der alle Parameter der Funktion enthält, z.B.
* Maxterm: Disjunktionsterm, der alle Parameter der Funktion enthält, z.B.
* KDNF (kanonische disjunktive Normalform): Disjunktion von Mintermen.

### Computerarithmetik

* + 1. Ein Bild, das Schrift, Screenshot, Diagramm, Zahl enthält.

       Automatisch generierte BeschreibungAddition

Überträge können bei signed verloren gehen   
(carry bit) oder erfordern eine   
 stellige Binärzahl

* + 1. Ein Bild, das Schrift, Text, Zahl, Screenshot enthält.

       Automatisch generierte BeschreibungSubtraktion

Ist der Minuend kleiner als der Subtrahend,   
erhöht man den Minuenden um   
auf eine stellige Binärzahl:

Diagram

Description automatically generatedZweierkomplement

**Von negativ zu positiv**

darstellen (nur signed)

* Minus 1 rechnen,
* Zahl invertieren

**Von positiv zu negativ**

darstellen (nur signed)

* Zahl invertieren
* Plus 1 rechnen

Ein Bild, das Handschrift, Text, Schrift, Tinte enthält.

Automatisch generierte BeschreibungRechnen von

* darstellen,
* Im 4bit-System fällt das vorderste Zeichen weg,
  + 1. Multiplikation

110

\* 101

\_\_\_\_\_\_\_

110

+ 0000

+ 11000

\_\_\_\_\_\_\_

11110

11011

\* 101

\_\_\_\_\_\_\_\_\_

11011

+ 000000

+ 1101100

\_\_111\_\_\_\_

10000111

1011

\* 1111

\_\_\_\_\_\_\_\_\_

1011

+ 10110

+ 101100

+ 1011000

\_\_\_\_\_\_\_\_\_

10100101

* + 1. Fehler von Kommazahlen

Beschränkte Genauigkeit: Nicht alle Zahlen sind genau darstellbar. Es sind nur ganzzahlige Vielfache von darstellbar. Alle anderen Zahlen bräuchten eine unendliche periodische Darstellung und werden auf die entsprechende Anzahl Bits abgekürzt. Unflexibel und fehleranfällig, aber performant.

* Absoluter Fehler:
* Relativer Fehler:

Bei mehreren Additionen/Subtraktionen addieren sich die absoluten Fehler.

* + 1. Fixkommazahlen

Umrechnung Dezimal zu Binär: Der Vorteil dieser Methode ist, dass man sofort feststellen kann, wenn sich die Binärstellen wie im folgenden Beispiel periodisch wiederholen:

(die erste Binärstelle nach dem Komma ist 1)  
 (für die zweite Binärstelle werden die Nachkommastellen von 1.1 dupliziert)  
   
   
   
   
 (ab hier wiederholt sich bis die Anzahl Stellen erreicht ist)

Fest-Komma:

**Ein Bild, das Text, Schrift, Reihe, Diagramm enthält.

Automatisch generierte Beschreibung**Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte BeschreibungExzess-q

Nullpunkt verschieben um z.B. negative Zahlen ohne Minus Zeichen darzustellen.

Zum Beispiel   
Kleinste Zahl mit 3 Stellen: -> Das wird die . Daraus folgt:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Binär |  |  |  |  |  |  |  |  |
| Betrag |  |  |  |  |  |  |  |  |
| Exz-4 |  |  |  |  |  |  |  |  |

* + 1. Gleitkommazahlen (32bit)

Bei 32bit float sind 8bit für den Exponenten reserviert.

Wenn die erste Nummer des Exponenten 0 ist, dann ist der Exponent positiv. Das gleiche mit der Mantissa.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

### Gruppentheorie

Ein Ring ist ein Körper, wenn jedes Element des Rings ausser der Null ein multiplikatives Inverses hat, die Multiplikation kommutativ ist (ab = ba), das Distributivgesetzt gilt (a(1-b) = a – ab) und wenn er eine 1 hat (multiplikatives neutrales Element)

### Interpretation eines Datenworts

* Im endlichen Ganzzahlkörper gibt es imme eine grösste und eine kleinste Zahl
* Begrenzt wird die Darstellung dieser Zahl durch den zur Verfügung stehenden Platz des Speichers und der definierten Wortgrösse
* In der Welt der Zahlen im Rechner oder der Codierung existiert der Begriff «unendlich» somit nicht in unserer Vorstellung
* In der Informatik werden alle Informationen in sogenannten Codewörtern abgelegt
* Die Anzahl der dastellbaren Codewörtern wird durch die Codewortlänge bestimmt. Ein Byte besteht aus 8 bit, ein Word besteht aus 16 oder 32 bit, ein TCP Paket besteht maximal aus 1024 bit.
  + 1. Tupel, Zahl, Vektor, Polynom

Diese Darstellungsformen sind äquivalent und beschreiben alle das gleiche Codewort.

* Tupel:
* Zahl: . Es gelten die üblichen Operationen der Ganzzahlrechnung.
* Vektor: . Es gelten die üblichen Operationen der Vektorrechnung.
* Polynom: . Es gelten die üblichen Operationen der Polynomrechnung.

Alle Berechnungen erfolgen in Z2

* + 1. Multiplikation zweier Polynome mod 2

### Zyklische Gruppe

Ein Bild, das Diagramm enthält.

Automatisch generierte BeschreibungIn der Gruppentheorie ist eine zyklische Gruppe eine Gruppe, die von einem einzelnen Element erzeugt wird. Sie besteht nur aus Potenzen des Erzeugers.

Hat dieses Polynom in eine Lösung?

Nach dem Hauptsatz der Algebra hat jedes Polynom so viele Nullstellen, wie durch die höchste Potenz angezeigt wird, hier 3 Nullstellen.

Eine zyklische Gruppe wird von einem einzelnen Element erzeugt und besteht nur aus Potenzen des Erzeugers. Definiert ist:

Dann können wir zunächst festhalten:

, aber:: der Zyklus beginnt von vorne

Damit entsteht aus der Erweiterungskörper

Die Elemente können auch codiert/interpretiert werden durch:

Derartige Zyklen entstehen z.B. beim zyklischen Code, oder der Erzeugung von Zufallszahlen.

### Wahrscheinlichkeit

Definition: Zufallsvorgan, Zufallsexperiment

Unter einem Zufallsvorgang verstehen wir einen Vorgang, bei dem:

- im Voraus feststeht, welche mögliche Ausgänge dieser theoretisch haben kann (z.B. ein Bit wird gedreht,   
 0 oder 1, oder ein lesbares Zeichen wird in ein anderes lesbares Zeichen überführt).

- der sich einstellende, tatsächliche Ausgang im Voraus jedoch unbekannt ist (Tritt ein Bitfehler bei der   
 Datenübertragung auf oder nicht? Unsicherheit bei einem Folgezeichen).

Zufallsexperimente, die geplant sind und kontrolliert ablaufen, heissen Zufallsexperimente.

Definition Ergebnismenge

Die Menge aller möglichen Ausgänge (Ergebnisse) eines Zufallsvorgangs heiss Ergebnismenge und wird mit bezeichnet. Ein einzelnes Element heisst Ergebnis. Wir notieren die Anzahl aller Elemente von , das heisst die Anzahl aller Ergebnisse mit .

* + 1. Eigenschaften von Wahrscheinlichkeiten
* Wahrscheinlichkeit des sicheren Ereignisses:
* Wahrscheinlichkeit des Komplementärereignisses:
* Wahrscheinlichkeit des unmöglichen Ereignisses:
* Wertebereich der Wahrscheinlichkeit:
* Additionssatz für Wahrscheinlichkeiten
* Additionssatz für 3 Ereignisse (Wahrscheinlichkeit, dass A, B oder C eintritt):

Wenn ein Experiment eine Anzahl verschiedener und gleich möglicher Ausgänge hervorbringen kann, dann ist die Wahrscheinlichkeit eines gewünschten Ausgangs gleich dem Verhältnis der Anzahl der gewünschten zur Anzahl der möglichen Ausgänge:

### Kombinatorik

Wahrscheinlichkeit erfordert Berechnung von Anzahlen. Dafür wird die Kombinatorik verwendet. Einige grundsätzliche Fragen der Kombinatorik: Wie viele Möglichkeiten gibt es, bestimmte Objekte anzuordnen? Wie viele Möglichkeiten gibt es, bestimmte Objekte aus einer Menge auszuwählen? Hier betrachen wir nur soweit nötig die geordnete und die ungeordnete Probe.

* + 1. Geordnete Proben
* Die Anzahl der **k-Tupel** aus einer **n-Menge** mit Wiederholung ist **nk**Beispiel: Bei einem Ziffernschloss muss man eine **5-stellige Zahl** einstellen, die aus den Ziffern **0,1,…,9** gebildet wird. Es gilt: Menge (n) = 10, k = 5, ergibt Kombinationen.
* Ein Bild, das Diagramm enthält.

  Automatisch generierte BeschreibungDie Anzahl der **k-Tupel** aus einer **n-Menge** ohne Wiederholung ist  
     
  Beispiel: Beim Pferde-Toto «3 aus 18» muss man von 18 Pferden 3 gemäss der Reihenfolge ihres Zieleinlaufs tippen.   
  Anzahl =
* Die Zahl der Permutationen einer **n-Menge** ohne Wiederholung für **n** = **k** ist **n**!   
  Beispiel: Zur Festlegung einer Sitzordnung bei einer Feier mit 10 Personen gibt es Möglichkeiten.
  + 1. Ungeordnete Proben
* Die Anzahl der **k-Tupel** aus einer **n-Menge** ist:  
  Beispiel: Eine Schulklasse mit 25 Schülern mächte ein Schachturnier austragen, bei dem jeder Schüler einmal gegen jeden anderen Schüler spielt. Wie viele Spiele werden ausgetragen?

**Ein Bild, das Diagramm enthält.

Automatisch generierte Beschreibung**

## Informationstheorie Grundlagen

Eine Nachricht wird als «Information» bezeichnet, wenn sie «relevant» und «nicht-redundant» ist.

* Relevant: Der Empfänger kann die Nachricht verstehen
* Ein Bild, das Diagramm enthält.

  Automatisch generierte BeschreibungNicht-redundant: Der Empfänger kann die Nachricht nicht voraussagen.

Modell der Informationsverarbeitung

Ein Bild, das Diagramm enthält.

Automatisch generierte Beschreibung

|  |  |  |
| --- | --- | --- |
| **Nachricht** | redundant | nicht-redundant |
| irrelevant | Zeichenvorrat bei Quelle und Senke verschieden | |
| relevant | Vorhersagbar | Information |

### Definitionen

Ein Bild, das Diagramm enthält.

Automatisch generierte BeschreibungDefinition Entscheidungsgehalt: Der Entscheidungsgehalt ist das Mass für   
den Aufwand, der zur Bildung einer Nachricht bzw. für die Entscheidung   
einer Nachricht notwendig ist.

Anders ausgedrückt: «Anzahl Knoten», die auf dem Weg zu diesem   
Zeichen durchlaufen werden müssen, also die Anzahl   
Entscheidungen, die man zwischen «links» und «rechts» treffen   
muss, bis man beim Zeichen angelangt ist.

Definition Entscheidungsfluss: Der Entscheidungsfluss ist definiert als

Wobei t die Zeit ist, die zur Übertragung eines Quellzeichens benötigt wird.

Definition Informationsgehalt: Der Informationsgehalt eines Zeichens sagt aus, wie viele Elementarentscheidungen zur Bestimmung dieses Zeichens zu treffen sind.

Definition Entropie: Die Entropie bezeichnet den mittleren Informationsgehalt der Quelle. Sie zeigt also auf, wie viele Elementarentscheidungen die Quelle/Senke im Mittel pro Zeichen treffen muss.

### Codewortlänge L(xk) vgl. Informationsgehalt

Die tatsächliche Bitgrösse (als Integer) der obigen, oftmals gebrochenen Bits bezeichnet man als «Codewortlänge».



* + 1. Mittlere Codewortlänge vgl. Entropie

Wir können nun auch die mittlere Codewortlänge von allen Zeichen berechnen (kann eine reelle Zahl sein).

Ein Bild, das Schrift, Text, weiß, Handschrift enthält.

Automatisch generierte Beschreibung

Günstig ist, wenn der Wert 𝐿 möglichst klein ist.

### Präfixeigenschaft

Ein Bild, das Diagramm, Reihe, Kreis enthält.

Automatisch generierte BeschreibungEine Gruppe von Codeworten besitzt die Präfixeigenschaft, wenn alle Codes ohne Trennzeichen identifizierbar sind. Die Zeichen befinden sich in den «Blättern» des Baumes. ASCII hat die Präfixeigenschaft, der Morsecode nicht.

### Shannon’sches Codierungstheorem

Das Codierungstheorem besagt, dass:

* Für jede beliebige Binärcodierung mit Präfixeigenschaft ist die mittlere Codewortlänge nicht kleiner als die Entropie:
* Für jede beliebige Quelle kann eine Binärcodierung gefunden werden, so dass gilt:

Der Begriff der Redundanz der Quelle wird erweitert um den Begriff der Redundanz des Codes:

Redundanz der Quelle:   
Redundanz des Codes:

### Quellen ohne Gedächtnis

Die Auftrittswahrscheinlichkeit eines Zeichens ist nicht abhängig von dem zuvor gesendeten Zeichen d.h. die Verbundwahrscheinlichkeit für die beiden Zeichen x und y ist

Ein Bild, das Text, Schrift, Screenshot, Zahl enthält.

Automatisch generierte BeschreibungEntscheidungsgehalt: hz(1.2) im TR

h0(1.1) im TR

Informationsgehalt: iz (1.2) im TR

Entropie:

Redundanz der Quelle:

Ein Bild, das Text, Schrift, Screenshot, Zahl enthält.

Automatisch generierte BeschreibungMittlere Codewortlänge:

Redundanz des Codes:

Bessere Codierung: Ziel ist es, für häufige Zeichen möglichst kurze Codeworte zu verwenden. Die könnte erreicht werden, indem folgende Änderung vorgenommen wird: .

Was sagt das Codierungstheorem von Shannon in diesem Fall?

Wie gross ist die minimale Redundanz eines Codes für diese Quelle (Shannon):

Code mit weniger Redundanz? Wie sieht dieser aus und wie gross ist die Redundanz?:   
Huffman Code anwenden, L = mittlere Codewortlänge, angeben

🡪 Für die Quelle sind wir mit dieser Codierung nahe an der minimalen mittleren Codewortlänge angelangt.

### Quellen mit Gedächtnis

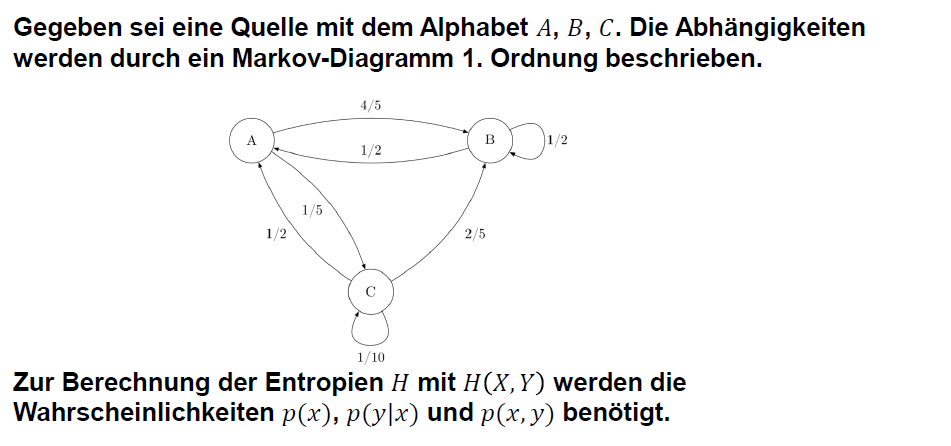
Allgemein kann nicht von einer gedächtnislosen Quelle ausgegangen werden:

Es galt:

Um zu bestimmen, setzten wir statt , in die obige Gleichung ein, es folgt:

Es kann gezeigt werden, dass gilt:

* + 1. Interpretation:
* Die mittlere Entropie einer Quelle ohne Gedächtnis ist stehts grösser oder gleich der Entropie einer Quelle mit Gedächtnis.
* In der Quellencodierung sind daher nicht Einzelzeichen zu codieren, sondern stets Zeichenketten.
* Beispiel der Redundanz der deutschen Sprache:  
  Entropie der Einzelzeichen   
  Entropie bei Ausnutzung aller Abhängigkeiten

Beispiel: Diskrete Quelle mit Gedächtnis  
Ein Bild, das Text, Diagramm, Reihe, Plan enthält.

Automatisch generierte Beschreibung

3 Unbekannte, . Bei drei Unbekannten werden drei Gleichungen gebraucht:

Die Gesamtwahrscheinlichkeit muss 1 sein, also gibt es noch eine weitere Gleichung:

## Quellencodierung & Komprimierung

* Datenkomprimierung: verlustfrei oder verlustbehaftet
* Verschlüsselung: symmetrisch oder asymmetrisch

### Datenkomprimierung

Komprimierung hat das Ziel, den Aufwand der Datenspeicherung und Datenübertragung zu reduzieren. Es können auch verschiedene Komprimierungsverfahren kombiniert werden.

**Anforderungen:**

* Hohe Komprimierungsrate für alle Typen von Daten (idealerweise ohne Kenntnis über die Eigenart der Daten)
* Hohe Encode- und Decode-Geschwindigkeit
* Geringe Ansprüche an die Hardware

Verfahren zur Datenkomprimierung

* Statische Verfahren, z.B. Huffmann-Codierung für die deutsche Sprache   
  Eigenart der Daten werden berücksichtigt
* Adaptive Verfahren, z.B. Huffman-Codierung mit gemessener Häufigkeitsverteilung  
  Eigenart der Daten werden berücksichtigt
* Dynamische Verfahren, z.B. ITU Standard V42.bis basiert auf LZ77 (Lempel, ziv)  
  Eigenart der Daten werden nicht berücksichtigt

### Huffman-Codierung

Huffman ist ein rekursives (Start bei Blättern) Verfahren für die Bildung eines kommafreien (Code hat Präfixeigenschaft - Kommafrei) Codes mit minimaler mittleren Codewortlänge.

Ein Bild, das Diagramm, Reihe, Kreis enthält.

Automatisch generierte BeschreibungVorgehen

* Häufigkeit/Auftrittswahrscheinlichkeiten der Zeichen notieren (gelb)
* Häufigkeit in aufsteigener Reihenfolge anordnen
* Baum zeichnen: Knoten mit den geringsten Häufigkeiten verbinden - neu entstehender Knoten mit neuer Wahrscheinlichkeit beschriften (orange)
* Kanten beschriften (link 0, rechts 1)
* Binärcode von Wurzel zu Blättern ablesen (Resultate der Codierung)  
  A = 0, B = 10, C = 110, D = 1110, E = 1111

Minimale Redundanz: 0, erreichbar bei günstigen Auftrittswahrscheinlichkeiten.

### Lauflängenkomprimierung

Erkennung von Wiederholungen: Bei der Lauflängencodierung werden Sequenzen von identischen Symbolen durch deren Anzahl und (falls notwendig) das Symbol ersetzt.   
RLE (Run Length Encoding) oder RLC (Run Length Coding) genannt.

Beispiel:   
Quelltext

Codiert

Lauflängenkomprimierung für Bit-Folgen

Bei der Kodierung von Bitfolgen existieren nur zwei Möglichkeiten: Eine Folge von Nullen oder eine Folge von Einsen. Auf jede Sequenz von Nullen folgt garantiert mindestens eine Eins – und umgekehrt. Ausnahme ist, wenn das Ende der Nachricht erreicht ist.

* Der Kodierer einigt sich mit dem Dekodierer, mit welchem Bit begonnen wird, 0 oder 1
* Anschliessend werden abwechselnd die Längen der 0 und 1 Folgen übertragen
* Der Dekodierer muss zu jedem empfangenen Wert entsprechend viele 0 oder 1 Bits ausgeben

Beispiel:

Originalnachricht   
Start mit einer 1  
Übertragener Code: Um jede Stelle von 0 .. 7 zu codieren reichen 3 Bit je Stelle aus, das Codewort wird zu

### Datenkomprimierung nach Lempel-Ziv (LZ)

Erkennen von wiederkehrenden Mustern. Komprimiert dadurch, dass zuvor eingelesener Text als Tabelle genutzt wird. Phrasen aus dem Eingabetext werden durch Zeiger in der Tabelle ersetzt. Dadurch wird die Komprimierung erreicht. Der Grad der Komprimierung hängt von der Länge der Phrasen, Fenstergrösse und Entropie des Ursprungstextes ab. Bei gleichen nah beieinander liegenden Textsequenzen kommt es sehr schnell zur Kompression.

Grundüberlegungen

* Der zu komprimierende Code hat wiederkehrende Muster oder Phrasen
* Anstatt den Code vollständig zu übertragen, werden «nur» die codierten Phrasen übertragen
* Dazu müssen die Phrasen zur Laufzeit erfasst und in einem Phrasenspeicher gespeichert und codiert werden. Die Grösse des Wörterbuchs und des «look ahead buffers» muss bestimmt werden.

Umsetzung

* Während des Durchlaufens der Daten wird ein ständig wachsender Baum erzeugt.
* Der Baum dient als Wörterbuch und zeigt Regularitäten auf.
* Die Knoten dienen als Referenzen.
* Werden gleiche Subdaten wiederholt geparst, so kann auf den entsprechenden Knoten des Wörterbuches referenziert wreden.

Die Datenstruktur besteht aus

* Einem Textfenster, dem search buffer: hier stehen die schon codierten Symbole, wird dynamisch aufgebaut
* Einem nach vorn gerichteten Puffer look-ahead buffer, dieser zeigt auf die als nächstes zu codierenden Symbole
* Sollte eine Sequenz von Symbolen in dem look-ahead buffer und dem search buffer übereinstimmen, so wird ein Code bestehend aus Position und Länge im search buffer gebildet und abgespeichert.
* Ansonsten wir der Code so gespeichert, wie er vorlag
* Anschliessend schieben sich beide Puffer eine Position nach vorn, deshalb wird diese Methode auch die Methode der gleitenden Fenster genannt.

Beispiel

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

* Ein Bild, das Text enthält.

  Automatisch generierte BeschreibungSuche in der Tabelle eine mäglichst lange Zeichenfolge, die mit den nächsten n zu codierenden Zeichen übereinstimmt
* Bilde ein Token und speichere es
* Verschiebe das Fenster um (n+1) Zeichen
* Wiederhole, bis alle Zeichen codiert sind

Lempel-Ziv für Bitfolgen

* Beginne mit einem Binärbaum, der einen Root-Knoten mit dem Index 0 besitzt
* Suche im Binärbaum die längste Zeichenfolge (bzw. «Knotenfolge»), die mit den nächsten n Zeichen übereinstimmt.
* Ein Bild, das Tisch enthält.

  Automatisch generierte BeschreibungCodiere den Eintrag in der Form (I,N): I = Index vom aktuellen Knoten, N = nächstes Zeichen
* Erstelle beim aktuellen Knoten einen neuen Kindknoten mit dem Index Imax+1 und dem Zeichen an der Position n+1 als Kantenbeschriftung.
* Verschiebe das «Fenster» hinter die n+1 Zeichen auf das nächste Zeichen.
* Wiederhole, bis alle Zeichen codiert sind.

## Quellencodierung & Verschlüsselungsverfahren

### Symmetrische Verfahren

Beide Teilnehmer verwenden *denselben Schlüssel*

Anzahl der erforderlichen Schüssel wächst stark, da für jedes Paar, welches Daten austauscht, ein eigener Schlüssel erstellt werden muss.

Anzahl Schlüssel bei 100 Mitgliedern:

* Caesar Chiffre / Substitutionsverfahren
* Transpositionsverfahren
* Vigenère-Chiffre Quadratische Anordnung von untereinander stehenden verschobenen Alphabeten
* DES Data Encryption Standard

DES / Data Encryption Standard

Wurde 1977 als offizieller Standard für die US-Regierung bestätigt und wird seither international vielfach eingesetzt. Heute wird er aufgrund der verwendeten Schlüssellänge von nur 56 Bits für viele Anwendungen als nicht ausreichend sicher erachtet.

### Asymmetrische Verfahren

Private und öffentliche Schlüssel

Ein Schlüssel zur Codierung und einen zweiten zur De-Codierung.

* + 1. RSA-Verfahren

Die RSA-Verschlüsselung ist ein Public-Key-Verfahren. Funktionsprinzip: Sie vergeben einen öffentlichen Schlüssel, mit dem jeder Botschaften an Sie so verschlüsseln kann, dass nur Sie sie entschlüsseln können.

Public & Private Key erhalten

1. Zwei Primzahlen (**p**,**q**) multiplizieren zum Produkt **n**:

2. Die Eulersche -Funktion von n berechnen.ergibt dasselbe Resultat und da beide Zahlen Primzahlen sind, kann einfach gerechnet werden.

3. Eine beliebige Zahl a zwischen 1 und auswählen, die mit teilerfrend ist:   
z.B. **a** = **3**, Test: ggt(20,3)=1

4. Das multiplikative Inverse von in berechnenKleinstmögliche Zahl => durch **20** teilbar, also

5. Nun haben wir den Public Key des Empfängers (b & n) und den Private Key(a) zur Verschlüsselung:

Ablauf für Prüfung:

* Beide Primzahlen und angeben
* angeben mit ausrechnen
* Es gilt: – ausrechnen
* Privaten Schlüssel angeben
* Nachricht entschlüsseln

Text verschlüsseln

Ein Bild, das Text, Wasser, schließen, hell enthält.

Automatisch generierte Beschreibung1. Buchstabentabelle generieren, zum Beispiel

2. Text in Zahlen aus Tabelle umwandeln: T = 20, E = 5, S = 19, T = 20

3. Zahlen mit **b** potenzieren und Modulo **n** rechnen:

4. Nun kann die Nachricht «26, 14, 13, 26» gesendet werden.

Text entschlüsseln

1. Dieselbe Buchstabentabelle wie beim Verschlüsseln verwenden.

2. Die empfangene Nachricht mit potenzieren und Modulo **n** rechnen:

3. Zahlen in Buchstaben umwandeln: 20 = **T**, 5 = **E**, 19 = **S**, 20 = **T**

### Euklidischer Algorighmus (Modulo)

Gibt ggT (grösster gemeinsamer Teiler) & kgV (kleinstes gemeinsames Vielfaches) aus.

Wenn der ggT von zwei Zahlen 1 ist, sind die beiden Zahlen teilerfremd (sie haben keinen gemeinsamen Teiler). Bei geraden Zahlen sind alle kleineren geraden Zahlen **nicht** teilerfremd. Bei Primzahlen sind alle Zahlen kleiner als die Primzahl teilerfremd. Die Zahl 1 ist bei jeder Zahl teilerfremd.

Teilerfremde Zahlen der Zahlen 1-10

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 1 | 1,2 | 1,3 | 1,2,3,4 | 1,5 | 1,2,3,4,5,6 | 1,3,5,7 | 1,2,4,5,7,8 | 1,3,7,9 |

Primfaktorzerlegung

12 = **2** \* ~~2~~ \* ~~3~~, 18 = ~~2~~ \* ~~3~~ \* **3** => 2 \* 3 = 6 => ggT. (12 \* 18) / 6 = 36 => kgV

Bei grossen Zahlen sehr rechenaufwendig. Deshalb bestimmen wir den ggT mit dem Euklidischen Algorithmus, und das kgV aus (a\*b) / ggT(a,b)

Euklidischer Algorithmus (ggT finden)

Seien

Initialisierung: Setze und (wie oft passt y in x)und (d.h. bestimme q und r so, dass ist.

Wiederhole, bis r = 0 ist.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ablauf | x | y |  |  |
| Initialisierung | 122 | 72 | 1 | 122 – 72 = 50 |
| 1. Wiederholung | 72 | 50 | 1 | 72 – 50 = 22 |
| 2. Wiederholung | 50 | 22 | 2 | 50 – 44 = 6 |
| 3. Wiederholung | 22 | 6 | 3 | 22 – 18 = 4 |
| 4. Wiederholung | 6 | 4 | 1 | 6 – 4 = 2 |
| 5. Wiederholung | 4 | 2 = ggT | 2 | 0 |

Erweiterter Euklidischer Algorithmus (ggT finden und als Linearkombination darstellen)

Seien

Initialisierung: Setze , (wie oft passt y in x), (d.h. bestimme q und r so, dass ist), und ***(u, s, v, t) = (1,0,0,1)***

Wiederhole, bis r = 0 ist.

**x = y** aus der vorangegangenen Zeile

**y = r** aus der vorangegangenen Zeile

**q = x div y**, **r = x mod y** = x-y\*q

**u = s** aus der vorangegangenen Zeile

**s = u – q \* s** mit u, q & s aus der vorangegangenen Zeile

**v = t** aus der voangegangenen Zeile

**t = v – q \* t** mit v, q & t aus der vorangegangenen Zeile

Ergebnis: In der letzten Zeile gilt y = ggT, s \* a + t \* b. Wenn ggT = 1 ist, dann folgt: t\*b = 1 mod a

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ablauf | x | y |  |  | ***u*** | ***s*** | ***v*** | ***t*** |
| Initialisierung | 99 | 79 | 1 | 99 – 79 = 20 | 1 | 0 | 0 | 1 |
| 1. Wiederholung | 79 | 20 | 3 | 79 – 60 = 19 | 0 | 1 – **1** \* 0 = **1** | 1 | 0 – **1** \* 1 = **-1** |
| 2. Wiederholung | 20 | 19 | 1 | 20 – 19 = 1 | 1 | 0 – **3** \* 1 = **-3** | -1 | 1 – **3** \* -1 = **4** |
| 3. Wiederholung | 19 | 1 | 19 | 19 – 19 = **0** | -3 | 1 – 1\*-3 = **4** | 4 | -1 – 1\*4 = **-5** |

Bedeutet:

Satz von Euler

Sei und mit ggT(**z**,**n**) = 1. Dann ist

Eulersche -Funktion

Sei und . Dann berechnet sich die Eulersche -Funktion als

Anzahl Zahlen mit und ggT(n,m) = 1 => Anzahl der teilerfremden Zahlen zwischen 0 und n

Anzahl Elemente mit multiplikativem Inversen

**Formen zur Berechung:**

1. Sei eine Primzahl: Dann ist

2. Sei eine Primzahl und : Dann ist

3. Seien und ggT(m,n) = 1: Dann ist

Beispiel 1:

Beispiel 2:

Beispiel 3:

Multiplikatives Inverses

Seien und . Dann gilt:   
Es gibt eine Zahl mit

Beispiel: Multiplikatives Inverse von **7** in

X kann auf 2 Arten berechnet werden. Entweder durch Durchprobieren oder durch den erweiterten euklidischen Algorithmus.

**Durchprobieren:**

nicht durch 11 teilbar

nicht durch 11 teilbar

…

durch 11 teilbar, **x** **= 8**

**Erweiterter euklidischer Algorithmus:** Siehe unten, danach das letzte Resultat von t (in diesem Fall -3) durch Modulo der Zahl in der Originalrechnung nehmen.= multiplikatives Inverses von 7 in

Liste von Primzahlen  
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997, 1009, 1013, 1019, 1021, 1031, 1033, 1039, 1049, 1051, 1061, 1063, 1069, 1087, 1091, 1093, 1097, 1103, 1109, 1117, 1123, 1129, 1151, 1153, 1163, 1171, 1181, 1187, 1193, 1201, 1213, 1217, 1223, 1229, 1231, 1237, 1249, 1259, 1277, 1279, 1283, 1289, 1291, 1297, 1301, 1303, 1307, 1319, 1321, 1327, 1361, 1367, 1373, 1381, 1399, 1409, 1423, 1427, 1429, 1433, 1439, 1447, 1451, 1453, 1459, 1471, 1481, 1483, 1487, 1489, 1493, 1499, 1511, 1523, 1531, 1543, 1549, 1553, 1559, 1567, 1571, 1579, 1583, 1597, 1601, 1607, 1609, 1613, 1619, 1621, 1627, 1637, 1657, 1663, 1667, 1669, 1693, 1697, 1699, 1709, 1721, 1723, 1733, 1741, 1747, 1753, 1759, 1777, 1783, 1787, 1789, 1801, 1811, 1823, 1831, 1847, 1861, 1867, 1871, 1873, 1877, 1879, 1889, 1901, 1907, 1913, 1931, 1933, 1949, 1951, 1973, 1979, 1987, 1993, 1997, 1999

## Kanalmodell

Ein Bild, das Text, Diagramm, Reihe, Screenshot enthält.

Automatisch generierte BeschreibungAbstrakte Abbildung eines Kanals. Beschreibt u.a. die Schwierigkeiten bei der Datenübertragung im Bezug auf den Kanal, z.B. die Fehlerwahrscheinlichkeit.

Aufgrund von «Rauschen» können bei der Übertragung Fehler auftreten. «Rauschen» kann z.B. eine schlechte Verbindung sein. Dieses Phänomen kann in einer Kanalmatrix abgebildet werden.

### Ein Bild, das Text, Schrift, Reihe, Diagramm enthält. Automatisch generierte BeschreibungKanalmatrix

Die Kanalmatrix beschreibt die Wahrscheinlichkeit, dass ein Zeichen auf ein korrektes oder inkorrektes Zeichen abgebildet wird.

bedeutet: Die Wahrscheinlichkeit, dass ein ankommt, wenn ein gesendet wurde.

Rechenbeispiel

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Eigenschaften

* Ist die Wahrscheinlichkeit für eine inkorrekte Zuweisung 0, so ist der Kanal nicht gestört.
* Sind alle Zuweisungen gleich wahrscheinlich, so ist der Kanal «vollständig» gestört.
* Ist , so ist der Kanal symmetrisch

Ausgangswahrscheinlichkeit

Wir können nun die Wahrscheinlichkeit für das Auftreten eines Zeichens anhand der Kanalmatrix berechnen. (Die Summe aus den inkorrekten und korrekten Zuweisungen.)

### Maximum Likelihood (Restwahrscheinlichkeit)

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungIst ein Kanal gestört, so müssen wir anhand des erhaltenen Zeichens entscheiden, welches Zeichen tatsächlich gesendet wurde. Man nehme die höchste Wahrscheinlichkeit in jeder Spalte einer Kanalmatrix und bilde diese auf ab.

Aus jeder Spalte () die höchste Wahrscheinlichkeit verwenden -> y-Wert   
Jede Zeile () ist ein x-Wert, also Zeile 1 bezieht sich auf das , Zeile zwei auf das usw.

### Transinformation

Wir können feststellen, dass bei der Datenübertragung über einen gestörten Kanal «Informationen» verloren gehen. Das bedeutet, der mittlere Informationsgehalt (die Entropie) nimmt ab.

: Eingangsentropie, : Ausgangsentropie

* Die Transformation beschreibt den maximalen, fehlerfreien Informationsfluss über einen Kanal.
* Verändert sich die Entropie der Quelle, verändert sich auch die Transinformation.
* Nimmt die Fehlerwahrscheinlichkeit zu, so verringert sich die Transinformation.
* Transinformation wird durch die Quelle bestimmt.
* Sind alle Positionen der Kanalmatrix gleich besetzt, so wird die Transinformation , d.h. , unabhängig von der Entropie am Kanaleingang. (-> Transinformation maximal)
* : ungestörter Kanal, : vollständig gestörter Kanal.

**Wie gross ist die Transinformation?**

*im TR*

Entropie am Kanalausgang:   
Irrelevanz:

Transinformation

Maximale Transinformation:

Ein Bild, das Text, Screenshot, Schrift, Diagramm enthält.

Automatisch generierte Beschreibung

Verbundentropie

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte BeschreibungDas paarweise Auftreten aller möglichen Kombinationen am Kanalein- und ausgang.

### Äquivokation (Verlust )

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte BeschreibungBeschreibt die Ungewissheit über ein gesendetes Zeichen bei bekannten Empfangszeichen. Ist der Kanal fehlerfrei, ist die Äquivokation gleich 0. Wird auch Rückschlussentropie genannt.

### Irrelevanz (Rauschen)

Beschreibt die Ungewissheit der empfangenen Zeichen bei vorgegebenen Sendezeichen.

### Beispiel

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

## Kanalcodierung & Blockcodes

Kanalcodierung beinhaltet Blockcodes und Faltungscodes. Die Kanalcodierung hat zum Ziel, bewusst Redundanz in eine Nachricht zu bringen, um Fehlern bei der Übertragung entgegenzuwirken. Der Coderaum wird dafür in gültige und ungültige Codeworte aufgeteilt.

### Der n-Dimensionale Coderaum

Weisse Punkte: Gültig, graue Punkte: ungültig

Ein Bild, das Text, Diagramm, Reihe, Screenshot enthält.

Automatisch generierte Beschreibung

Definition

* Anzahl der sicher erkennbaren Fehler:
* Anzahl der sicher korrigierbaren Fehler, wenn h gerade:
* Ein Bild, das Text, Reihe, Diagramm, Kreis enthält.

  Automatisch generierte BeschreibungAnzahl der sicher korrigierbaren Fehler, wenn h ungerade:

Blockcode «Voci»:

* Anzahl Worte = 2 bei Binärcode
* Anzahl Codestellen ( falls )
* Anzahl Nachrichtenstellen
* Anzahl Kontrollstellen
* Gültige Codewörter: (Anzahl Worte) / Binär
* Mögliche Codewörter: (Anzahl Worte) / Binär
* **Hammingdistanz** : beschreibt den minimalen Abstand zwischen zwei gültigen Codeworten im gesamten Coderaum.   
   Ein Bild, das Text enthält.

  Automatisch generierte Beschreibung

Treten mehr Fehler auf als korrigierbar sind, so wird entweder falsch korrigiert oder der Fehler wird nicht erkannt.

Der Coderaum ist Dichtgepackt, wenn sich alle Codewörter (gültige und ungültige) in einer Korrigierkugel befinden.

Sei:

* die Dimension des Code (Anzahl aller Codewörter )
* die Dimension der Nachrichten (Anzahl aller gültigen Codewörter )
* die Dimension der Kontrollstellen mit

So folgt die Codeabschätzung:

Ein Bild, das Text, Schrift, Diagramm, Zeichnung enthält.

Automatisch generierte Beschreibung

Wenn die linke Seite der Gleichung die rechte Seite, ist der Coderaum dichtgepackt.

### Hamming Blockcode

Beim Hamming-Code werden Gleichungen basierend auf den einzelnen Stellen des Codewortes definiert. Ein Codewort ist gültig, wenn es all diese Gleichungen erfüllt.

Das Generatorpolynom entspricht der ersten Prüfgleichung. Der Code hat Stellen.  
Anzahl Kontrollstellen entspricht Anzahl Prüfgleichungen. Wenn es nur eine Prüfgleichung gibt, gibt es auch nur eine Kontrollstelle.  
Anzahl Codeworte:

Ein Bild, das Text, Diagramm, Plan, Screenshot enthält.

Automatisch generierte Beschreibung

Generatormatrix

Ein Bild, das Text, Schrift, Diagramm, Screenshot enthält.

Automatisch generierte Beschreibung

Formell können wir nun definieren

Fehlersyndrom

Bei einem fehlerhaften Codewort liefert uns die obige Formel keinen Nullvektor, sondern genau die Spalte der Generatormatrix, in der ein Fehler aufgetreten ist. (Funktioniert nicht, wenn mehr als ein Fehler aufgetreten ist)  
Beispiel: Generatorpolynom , Fehler bei

Prüfmatrix nach Prüfgleichungen angeben

Bsp: Folgende Gleichungen sind gegeben

**Matrix:**

Kontrollstellen: Anzahl der Spalten der Einheitsmatrix, hier   
Gültige Codeworte:   
Hamming-Distanz: Spalten müssen ergeben

* Spalte mit am meisten 1er nehmen + Spalte mit am zweit-meisten 1er.
* Welche Spalten müssen noch ergänz werden, damit alles ergibt (minimale Anzahl)?
* Anzahl verwendete Spalten = Hamming-Distanz.  
  In diesem Fall , das bedeutet, erkennbare Fehler Korrigierbare Fehler

Ein Bild, das Text, Screenshot, Diagramm, Kreis enthält.

Automatisch generierte BeschreibungFehlersyndrom, falls gestört sind:

### Zyklische Codes

Idee: Generatormatrix kann durch ein Generatorpolynom beschrieben werden.  
Ziel: Vereinachte Berechnung der Kontrollstellen durch rückgekoppelte Schieberegister.

Das Generatorpolynom lässt sich in der Polynom- und Binärschreibweise notieren.

* Polynom:
* Binär / Vektor:

Der höchste Grad des Polynoms bestimmt die Anzahl der Kontrollstellen, hier 3

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungErmitteln der Kontrollstellen (Nicht Anzahl, sondern welche)

Die Berechnung der Kontrollstellen einer gegebenen Nachricht funktioniert über die Polynomdivision (=Mehrfachaddition).

* Beginne mit der Nachricht
* Schreibe nun unter die erste 1 das Generatorpolynom aus der Aufgabe hin
* Berechne die Summe mit
* Wiederhole mit dem aktuellen Resultat, bis alle Kontrollstellen berechnet wurden
* Sollte das Generatorpolynom länger sein als das Codewort (Raushängtwenn es noch einmal darunter gesetzt wird) ist das Codewort ungültig

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungNachricht im Bild: , Generator Kontrollstellen:

Über die Polynomdivision kann auch bestimmt werden, ob ein Codewort gültig ist oder nicht. Wenn die Kontrollstellen ans Codewort angehängt geteilt durch den Generator 0 ergibt, ist das Codewort gültig. Bei einem fehlerhaften Codewort liefert uns die Polynomdivision genau die Spalte, in der ein Fehler aufgetreten ist.

Mit einem gültigen Codewort kann also die Generatormatrix   
hergeleitet werden, indem jedes Bit einmal invertiert wird.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Diese Darstellung der Polynomdivision wird als «Mehrfachaddition» bezeichnet, funktioniert aber identisch. Ist die Anzahl der Kontrollstellen bekannt, so kann man sich die Berechnung sparen (Einheitsmatrix)

### Ein Bild, das Text enthält. Automatisch generierte BeschreibungSpezielle Codes

Zyklischer Hamming-Code / Blockcode

Anzahl Kontrollstellen   
Anzahl Nachrichtenstellen   
Anzahl Codestellen (Nachrichtenstellen + Kontrollstellen)   
Anzahl Codewörter   
Anzahl gültige Codeworte

Welches sind die gültigen Codeworte? Mehrfachaddition mit Vektor des Generatorpolynoms

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungZyklischer Abramson-Code (CRC-Code)

Ermittlung der Kontrollstellen durch rückgekoppeltes Schieberegister

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

Automatisch generierte Beschreibung

* Gleich viele Quadrate zeichnen wie höchster Grad vom Polynom (Jedes Polynom bildet eine Linie)
* Jeder Grad des Polynoms (jedes Element, welches eine 1 hat im Vektor) bildet ein . Höchster und tiefster Grad sind ausserhalb der Quadrate ganz links und ganz rechts.

## Faltungscodes

Faltungscodes erlauben die fortlaufende Codierung eines kontinuierlichen Datenstroms (z.B. Stream), wobei keine Blockbildung oder Synchronisation benötigt wird. Gute Faltungscodes werden durch Rechnersimulation gefunden.

### Encoderschaltung

Bei Faltungscodes werden mehrere Generatorpolynome in eine Encoderschaltung abgebildet, wobei gilt:

* Jedes Generatorpolynom bildet eine «Linie»
* Der höchste Grad bestimmt die «Kastenzahl»
* Jeder Grad eines Polynoms bildet ein / XOR
* Encodergedächtnis = Anzahl Kästchen

Ein Bild, das Text, Diagramm, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

Encoder Schaltung / (Output, Input, Gedächtnis)  
Tailbits = Encodergedächtnis

Code-Rate für 185 Bits:

Ein Bild, das Schwarz, Dunkelheit enthält.

Automatisch generierte BeschreibungBeispiel skizzieren einer Encoder-Schaltung (1,2,2)

Generatorpolynome:

* oder
* oder

Encodergedächtnis   
Impulsantwort Encoder:   
Impulsantwort Decoder: encode

### Zustandsdarstellung des (1,2,2) Encoders

Ein Bild, das Schwarz, Dunkelheit enthält.

Automatisch generierte BeschreibungEin Bild, das Reihe, Diagramm enthält.

Automatisch generierte Beschreibung

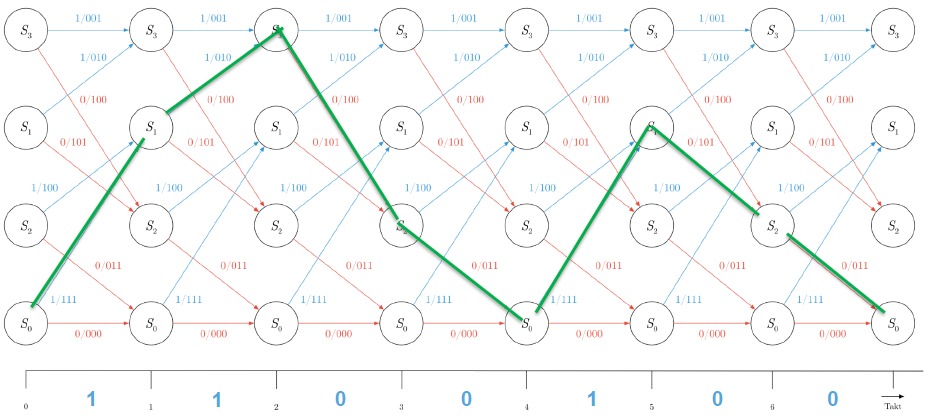
### Zustandsdarstellung des (2,1,3) Encoders

Ein Bild, das Text, Diagramm, Reihe, Plan enthält.

Automatisch generierte Beschreibung

### Netzdiagramm

Das Netzdiagramm bezeichnet ein aufgespanntes Zustandsdiagramm bei einer Folge von Eingabezeichen.



Ein Bild, das Text, Diagramm, Reihe, Plan enthält.

Automatisch generierte BeschreibungEin Bild, das Diagramm, Kreis, Reihe, Schrift enthält.

Automatisch generierte BeschreibungWir können damit Zeichenketten decodieren. Dekodieren der Zeichenfolge (anhand vom vorherigen Beispiel).

### Definitionen

Anzahl Zustände

Zustände = EntscheidungsmöglichkeitenGedächtnisstellen

Ausgehend vom oberen Schieberegister (2 Entscheidungsmöglichkeiten, 3 Gedächtnisstellen: Zustände (kann im Zustandsdiagramm kontrolliert werden)

Anzahl Tailbits

Die Anzahl Tailbits entspricht der Anzahl Gedächtnisstellen, damit sicher wieder alle Gedächtnisstellen mit 0 belegt sind.

Gewicht

Das Gewicht des Codes ist die Anzahl von Bitstellen eines Codeworts, die von «0» verschieden sind.

Fundamentalweg

Ist der (Teil-)Weg eines Codes, der im Zustand beginnt und wieder im Zustand endet. Die Analyse der Fundamentalwege liefert die Struktur des Faltungscodes.

Metrik:

* bezeichnet den Zustandsübergang, der durch «1» ausgelöst wird.
* bezeichnet die Anzahl der durch den Übergang zur Codefolge hinzukommenden «1» Bitstellen (Gewichtszunahme)
* Eine Zählvariable, die die Anzahl der Übergänge zählt
* Jede Kante eines Fundamentalweges lässt sich durch das Triplet beschreiben (Kantengewicht)

Impulsantwort

Man sendet eine 1 und dann eine Anzahl von Nullen, die dem Grad des Polynoms entspricht. Z.B.

* + 1. Was ist ein «guter» Faltungscode?

Ein Faltungscode ist gut, wenn der Unterschied der Ausgabe bei einem Zustandsübergang immer maximal ist.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

* + 1. Generatorpolynome für optimale Faltungscodes

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung