Lernzettel

Pascal Diller

November 9, 2024

Contents

Logik

- "$\wedge$": **Und**
- "$\vee$": **Oder**
- "$\neg$": **Nicht** (Verneinung)
- $A \implies B$: A **impliziert**B
- $A \impliedby B$: A wird durch B **impliziert**
- $A \iff B$: A ist äquivalent zu B
  Es gilt: $A \implies B$ und $A \impliedby B$
- $\forall$: **Für alle**
- $\exists$: **Es existiert (mindestens) ein**

Mengen

Eine **Menge** ist eine Zusammenfassung von (mathematischen) Objekten.
Die Objekte in einer Menge werden als **Elemente** bezeichnet.

- $x \in M$: x in/Element M
- $x \notin M$: x nicht in/Element M

Defintion einer Menge:

- Aufzählung:

  $M_1 = \{0, 1, 2, 3, 5, 8, -1\}; \quad M_2 = \{1, 2, 3, 4, 5, \dots\}$

  Es kommt **nicht** auf die **Reihenfolge** und **nicht** auf **Verdopplungen** an: $\{1, 3, 2, 3\} = \{3, 2, 1\} = \{1, 2, 3\}$

- Beschreibung:

$$M_3 = \{x \in \mathbb{R} : x \geq -1 \wedge x \leq 1\} = [-1, 1]$$

Menge B ist eine **Teilmenge** von Menge A, wenn für jedes $x \in A$ auch $x \in B$ gilt.

- $A \subset B$ ("A ist eine Teilmenge von B")
- $A \supset B$ ("B ist eine Teilmenge von A")

Mengenoperationen:

- **Vereinigung** der Mengen A und B
  $A \cup B = \{x : x \in A \vee x \in B\}$ ("A vereinigt B")
- **Durchschnitt** der Mengen A und B
  $A \cap B = \{x : x \in A \wedge x \in B\}$ ("A geschnitten B")
- **Differenzmenge** der Mengen A und B
  $A \setminus B = \{x : x \in A \wedge x \notin B\}$ ("A ohne B")

Kartesisches Produkt:
sei $n \in \mathbb{N}$ und seien $X_1, \ldots, X_n$ Mengen, dann ist

$$X_1 \times \cdots \times X_n = \{(x_1, \ldots, x_n) : x_i \in X_i, \text{ für } i = 1, \ldots, n\}$$

die Menge der n-**Tupel** mit i-ter Koordinate x_i in X_i für $i = 1, \ldots, n$.

Potenzmenge:
Die Menge aller Teilmengen einer Menge X heißt Potenzmenge von X und wird mit $\mathcal{P}(\mathcal{X})$ bezeichnet:

$$\mathcal{P}(X) = \{Y : Y \subset X\}$$

Es gilt immer: $\emptyset \in \mathcal{P}(X)$ und $X \in \mathcal{P}(X)$.
Beispiel: Sei $X = \{1, 2, 3\}$. Dann ist

$$\mathcal{P}(X) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2.3\}, \{1, 2, 3\}\}$$

Sei P eine Menge bestehend aus Mengen. Dann steht

$$\bigcup_{Y \in P} Y = \{y : \text{ es gibt } Y \in P \text{ so dass } y \in Y\}$$

für die (möglicherweise unendliche) Vereinigung aller Mengen in P.

Partitionen:
Sei X eine Menge. Eine Partition von X ist eine Teilmenge $P \in \mathcal{P}(X) \setminus \{\emptyset\}$ sodass

- für alle $Y, Z \in P$ mit $Y \neq Z, Y \cap Z = \emptyset$ (Y und Z sind disjunkt).
- $\bigcup_{Y \in P} Y = X$.

Definierte Mengen:

- Leere Menge: $\emptyset = \{\}$
- Natürliche Zahlen: $\mathbb{N} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \dots\}$ ($0 \notin \mathbb{N}$)
- Ganze Zahlen: $\mathbb{Z} = \{0, 1, -1, 2, -2, 3, -3, 4, -4, \dots\}$
- Rationale Zahlen: $\mathbb{Q} = \left\{\frac{p}{q} : p, q \in \mathbb{Z}, q \neq 0\right\}$
- Relle Zahlen: $\mathbb{R}$, Menge aller **rellen Zahlen**, die man **nicht abzählen** kann

Es gilt: $\mathbb{N} \in \mathbb{Z} \in \mathbb{Q} \in \mathbb{R}$

boolesche Algebra

Als eine **boolesche Algebra** bezeichnet man eine Menge $V = \{a, b, c, \dots\}$, auf der zwei zweistellige Operationen $\oplus$ und $\otimes$ derart definiert sind, dass durch ihre Anwendung auf Elemente aus V wieder Elemente aus V enstehen (Abgeschlossenheit).

Abgeschlossenheit: für alle $a, b \in V$ gilt:

$$a \otimes b \in V$$

$$a \oplus b \in V$$

Zudem müssen die vier **Huntingtonischen Axiome** gelten:

- H1: **Kommutativgesetz**

$$a \otimes b = b \otimes a$$

$$a \oplus b = b \oplus a$$

- H2: **Distributivgesetz**

$$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$$

$$a \oplus (b \otimes c) = (a \oplus b) \otimes (a \oplus c)$$

- H3: **Neutrale Elemente**
  Es existieren zwei Elemente $e, n \in V$, so dass gilt:

$$a \otimes e = a \qquad (e \text{ wird } \textbf{Einselelement} \text{ genannt})$$
$$a \oplus n = a \qquad (n \text{ wird } \textbf{Nullelement} \text{ genannt})$$

- H4: **Inverse Elemente**
  Für jedes $a \in V$ existiert ein Element $a^{-1} \in V$, so dass gilt:

$$a \otimes a^{-1} = n$$

$$a \oplus a^{-1} = e$$

Schaltalgebra

Die Schaltalgebra $(\{0, 1\}, \wedge, \vee)$ ist eine spezielle boolesche Algebra. 0 und 1 können als die logischen Werte wahr und falsch interpretieren.
Es gelten die vier Huntingtonischen Axiome:

(H1)	Kommutativgesetz	$a \vee b = b \vee a$
		$a \wedge b = b \wedge a$
(H2)	Distributivgesetz	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
		$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
(H3)	Neutrale Elemente	$a \wedge 1 = a$
		$a \vee 0 = a$
(H4)	Invere Elemente	$a \wedge \neg a = 0$
		$a \vee \neg a = 1$

Es lassen sich folgende Sätze ableiten:

(R1)	Assoziativgesetz	$(a \wedge b) \wedge c = a \wedge (b \wedge c)$
		$(a \vee b) \vee c = a \vee (b \vee c)$
(R2)	Idempotenzgesetz	$a \wedge a = a$
		$a \vee a = a$
(R3)	Absorptionsgesetz	$a \wedge (a \vee b) = a$
		$a \vee (a \wedge b) = a$
(R4)	DeMorgan-Gesetz	$\neg(a \wedge b) = \neg a \vee \neg b$
		$\neg(a \vee b) = \neg a \wedge \neg b$

boolesche Funktionen

Eine Funktion $f : \{0,1\}^n \rightarrow \{0,1\}$ wird als boolesche Funktion bezeichnet.

boolescher Ausdruck

Sei $V = \{x_1, x_2, \ldots, x_n\}$ eine Menge boolescher Variablen. Dann ist die Menge der booleschen Ausdrücke wie folgt definiert:

- $0, 1, x_i$ sind boolesche Ausdrücke.
- Ist Φ ein boolescher Ausdruck, dann ist auch $\neg\Phi$ ein boolescher Ausdruck.
- Wenn Φ und Ψ boolesche Ausdrücke sind, dann sind auch $\Phi \wedge \Psi$ und $\Phi \vee \Psi$ boolesche Ausdrücke.
- Ist Φ ein boolescher Ausdruck, dann ist auch (Φ) ein boolescher Ausdruck.

Äquivalenz boolescher Ausdrücke

Zwei boolesche Ausdrücke Φ und Ψ sind äquivalent, falls sie dieselbe Funktion repräsentieren.
Sie sind genau dann äquivalent, wenn für alle Variablenbelgungen $x_1, \cdots, x_n$ die folgende Beziehung gilt:

$$\Phi(x_1, \cdots, x_n) = \Psi(x_1, \cdots, x_n)$$

Tautologie

Ein boolescher Ausdruck, der immer wahr ist, wird als **Tautologie** bezeichnet.
Das heißt zwei boolesche Ausdrücke A und B sind äquivalen, wenn $A \leftrightarrow B$ eine Tautologie ist.

Vollständiges Operatorensystem

M sei eine beliebige Menge von Operatoren. M ist ein **vollständiges Operatorensystem**, wenn sich jede boolesche Funktion auch durch einen Ausdruck bescreiben lässt, in dem neben den Variablen $x_1, \cdots, x_n$ ausschließlich Operatoren aus M vorkommen.

Normalformen

Kanonische Normalformen

Eine kanonische Normalform ist eine **eindeutige Darstellung** mit UND, ODER und NICHT.

- kanonische **disjunktive Normalform (DNF)**

  Die Disjunktion (Verbinden mit ODER) von Mintermen der Funktion

  Die **nicht kanonische** Form ist eine Disjunktion beliebiger konjunktiv verknüpfter boolescher Ausdrücke.

  Konstruktion einer DNF: Für jede **Einszeile** der Wahrheitstabelle wird ein **Minterm** konstruiert, welcher für genau diese Variablenbelgungen 1 wird. Alle so erstellten Minterme werden **disjunktiv verknüpft**.

- kanonische **konjuntive Normalform (KNF)**

  Die Konjunktion (Verbinden mit UND) von Maxtermen der Funktion

  Die **nicht kanonische** Form ist eine Konjunktion beliebiger disjuntiv verknüpfter boolescher Ausdrücke.
  Konstruktion einer KNF: Für jede **Nullzeile** der Wahrheitstabelle wird ein **Maxterm** konstruiert, welcher für genau diese Variablenbelgungen 0 wird. Alle so erstellten Maxterme werden **konjunktiv verknüpft**.

Es wird **ausschließlich** die **kanonische Form** behandelt und deswegen das Wort kanonisch häufig wegelassen.

Minterm, Maxterm

Sei $f(x_1, \ldots, x_n)$ eine beliebige n-stellige boolesche Funktion.
Ein **Minterm** ist jeder Ausdruck der Form

$$\hat{x}_1 \wedge \cdots \wedge \hat{x}_n \text{ mit } \hat{x}_i \in \{\overline{x}_i, x_i\}$$

Ein **Maxterm** ist jeder Ausdruck der Form

$$\hat{x}_1 \vee \cdots \vee \hat{x}_n \text{ mit } \hat{x}_i \in \{\overline{x}_i, x_i\}$$

Ein **Literal** ist der Teilausdruck $\hat{x}_i$, der entweder aus einer negierten oder einer unnegierten Variablen besteht.

Relationen

Eine **(binäre) Relation** zwischen zwei Mengen X und Y ist eine Teilmenge

$$\mathrm{R} \subset X \times Y$$

Im Falle $X = Y$ sprechen wir von einer Relation auf X.
$x \in X$ steht in Relation zu $y \in Y$ genau dann wenn $(x, y) \in \mathrm{R}$.
Auch geschrieben: $x \,\mathrm{R}\, y$ oder $x \sim_{\mathrm{R}} y$ für $(x, y) \in \mathrm{R}$ und $x \,\not{\mathrm{R}}\, y$ oder $x \not\sim_{\mathrm{R}} y$ für $(x, y) \notin \mathrm{R}$.

Seien X, Y und Z Mengen und $\mathrm{R} \subset X \times Y$, $\mathrm{S} \subset Y \times X$ Relationen.

- Die zu R **inverse Relation** ist

$$\mathrm{R}^{-1} = \{(y, x) \in Y \times X : (x, y \in \mathrm{R})\}$$

- Die Verkettung von R und S ist

  $$\mathrm{S} \circ \mathrm{R} = \{(x,z) \in X \times Z : \text{es gibt } y \in Y \text{ mit } (x,y) \in \mathrm{R} \text{ und } (y,z) \in \mathrm{S}\}$$

Eine binäre Relation R auf der Menger X heißt:

- **relfexiv**, wenn $x \,\mathrm{R}\, x$ für alle $x \in X$.
- **symmetrisch**, wenn für alle $x, y \in X$ aus $x \,\mathrm{R}\, y$ stets $y \,\mathrm{R}\, x$ folgt.
- **antisymmetrische**, wenn für alle $x, y \in X$ aus $x \,\mathrm{R}\, y$ und $y \,\mathrm{R}\, x$ stets $x = y$ folgt.
- **asymmetrisch**, wenn für alle $x, y \in X$ aus $x \,\mathrm{R}\, y$ stets $y \not\mathrm{R}\, x$ folgt.
- **transitiv**, wenn für alle $x, y, z \in X$ aus $x \,\mathrm{R}\, y$ und $y \,\mathrm{R}\, z$ stets $x \,\mathrm{R}\, z$ folgt.

Äquivalenzrelationen

Sei X eine nicht leere Menge. Eine Relation R auf X die relfexiv, symmetrisch und transitiv ist, heißt **Äquivalenzrelationen**. Für $x \in X$ nennt man die Menge

$$[x] \sim_{\mathrm{R}} = \{y \in X : x \,\mathrm{R}\, y\}$$

die **Äquivalenzklasse** von x. Man nennt x und jedes andere Element aus $[x] \sim_{\mathrm{R}}$ einen **Vertreter** oder **Repräsentanten** dieser Äquivalenzklasse.

Sei X eine Menge und $\sim$ eine Äquivalenzrelation auf X. Ein **Vertretersystem** ist eine Teilmenge von X, die für jede Äquivalenzklasse genau ein Element enthält.

Ordnungsrelationen

Sei X eine Menge. Eine **Ordnung** auf X ist eine reflexive, antisymmetrische und transitive Relation. Eine **strikte Ordnung** auf X ist eine asymmetrisch und transitive Relation. Wir nennen eine (strikte) Ordnung $\preceq$ **total**, wenn je zwei Elemente vergleichbar sind:

$$\text{für alle } x, y \in X \text{ gilt } x \preceq y \text{ oder } y \preceq x$$

Ansonsten nennen wir sie **partiell**.

Hüllen

Sei R eine Relation auf der Menge X. Wir definieren:

- Für $n \in \mathbb{N}_0$

  $$R^n = \begin{cases} \mathrm{I}_X & n = 0 \\ \mathrm{R} \circ \mathrm{R}^{n-1} & n \geq 1 \end{cases}$$

  Es gilt, dass $\mathrm{R}^1 = \mathrm{R}$

- Die **transitive Hülle** von R ist

  $$\mathrm{R}_{\text{trans}} = \bigcup_{n \in \mathbb{N}} \mathrm{R}^n$$

- Die **reflexive Hülle** von R ist

  $$\mathrm{R}_{\text{refl}} = \mathrm{R} \cup \mathrm{I}_X$$

- Die **symmetrische Hülle** von R ist

  $$\mathrm{R}_{\text{sym}} = \mathrm{R} \cup \mathrm{R}^{-1}$$

Vollständige Induktion

Das **Prinzip der vollständigen Induktion** ist ein Beweisverfahren, mit dem man Aussagen $A(n)$ beweisen kann, die von $n \in \mathbb{N}_0$ abhängen.

Idee der vollständigen Induktion

Zu zeigen sei die Aussage $A(n)$ für alle $n \in \mathbb{N}_0$ mit $n \geq n_0$ für ein $n_0 \in \mathbb{N}$.
Angenommen, man kann zeigen, dass $A(n_0)$ gilt, und weiter kann man beweisen, dass $A(n+1)$ gilt, wenn man voraussetzt, dass $A(n)$ gilt, d.h. die Implikation $A(n) \implies A(n+1)$ ist für alle $n \in \mathbb{N}, n \geq n_0$ gültig.
Dann gilt $A(n)$ für alle $n \in \mathbb{N}$ mit $n \geq n_0$.

Beweis durch vollständige Induktion

1. **Induktionsanfang (I.A.)**: Es gibt ein $n_0 \in \mathbb{N}_0$, sodass $A(n_0)$ wahr ist.

2. **Induktionsvoraussetzung (I.V.)**: Annahme: $A(n)$ ist wahr (für ein $n \geq n_0$).

3. **Induktionsschluss (I.S.)**: Zeige: $A(n) \implies A(n+1)$.

Abbildungen

Eine **Abbildung** $f : X \to Y$ besteht aus:

- einer Menge X, der **Definitionsbereich** von f;
- einer Menge Y, der **Wertebereich** von f;
- einer **Vorschrift**, die jedem $x \in X$ eindeutig ein $y \in Y$ zuordnet.

Notation: $f : X \to Y, x \mapsto f(x)$

Seien X, Y Mengen, $f : X \to Y$ eine Abbildung und $x \in X, y \in Y$ sodass $f(x) = y$.
Dann ist y das **Bild** von x und x ein **Urbild** von y.
Für eine Teilmenge $X_0 \subset X$ ist

$$f(X_0) := \{y \in Y : \text{ es gibt } x \in X_0, \text{sodass } f(x) = y\} \subset Y$$

das **Bild** von X_0 und für eine Teilmenge $Y_0 \subset Y$ ist

$$f^{-1}(Y_0) := \{x \in X : f(x) \in Y_0\} \subset X$$

das **Urbild** von Y_0.

Seien X und Y Mengen und $f : X \to Y$ eine Abbildung.

f ist **injektiv** falls aus $x_1, x_2 \in X$ mit $f(x_1) = f(x_2)$ stets $x_1 = x_2$ folgt.

"zu jedem y höchstens 1 x-Wert"

f ist **surjektiv** falls es für jedes $y \in Y$, ein $x \in X$ existiert so dass $f(x) = y$.

"zu jedem y mindestens 1 x-Wert"

f ist **bijektiv** falls f injektiv und surjektiv ist.

Seien X, Y, Z Mengen und $f : X \to Y$ und $g : Y \to Z$ Abbildungen.
Die **Komposition** oder **Verknüpfung** von f und g ist die
Abbildung $g \circ f : X \to Z$, definiert durch $(g \circ f)(x) = g(f(x))$.

Zahlensysteme

Binärsystem

Eine Binärzahl b mit $n+1$ Stellen hat die Form $b_n \dots b_1 b_2$ mit $b_i \in \{0, 1\}$.

Sie entspricht der Dezimalzahl d mit $d = b_n \cdot 2^n + \dots + b_1 \cdot 2^1 + b_0 \cdot 2^0$

Beispiel: $1101_2 = 1 \cdot 2^3 +^{\cdot} 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 13_{10}$

Carry-Flag

Wenn bei einer Addition oder Subtraktion ein **Übertrag in der höchsten Stelle** auftritt, wird die Carry-Flag gesetzt. Dieser kann von nachfolgenden Befehlen aufgerufen werden.

Zweierkomplement

Um **negative Zahlen** darzustellen wird der entsprechende Wert des **höchsten Bits negiert**.

Beispiel bei 4 Bit: $1011_{2c} = 1 \cdot (-2^3) + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = -5$

Um von einer positiven ganzen Zahl zur negativen Zahl (oder umgekehrt) gleichen Betrags zu gelangen werden **alle Bits invertiert** und **1 zum Ergebnis addiert**.

Hexadezimalsystem

Eine Hexadezimalzahl h mit $n+1$ Stellen hat die Form $h_n \dots h_1 h_0$ mit $h_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(\widehat{=}10), B(\widehat{=}11), C(\widehat{=}12), D(\widehat{=}13), E(\widehat{=}14), F(\widehat{=}15)\}$.

Sie entspricht der Dezimalzahl d mit $d = h_n \cdot 16^n + h_1 \cdot 16^1 + h_0 \cdot 16^0$.

Beispiel: $\texttt{5F}_{16} = 5 \cdot 16^1 + 15 \cdot 16^0 = 95_{10}$

4 Binärziffern lassen sich zu einer Hexadezimalzahl zusammenfassen:
$\underbrace{1101}_{13_{10}=D_{16}}\ \underbrace{0011_2}_{3_{16}} = \texttt{D3}_{16}$

Oktalsystem

Eine Oktalzahl o mit $n + 1$ Stellen hat die Form $o_n \dots o_1 o_0$ mit $o_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$

Sie entspricht der Dezimalzahl d mit $d = o_n \cdot 8^n + \cdots + o_1 \cdot 8^1 + o_0 \cdot 8^0$.

Beispiel: $36_8 = 3 \cdot 8^1 + 6 \cdot 8^0 = 30_{10}$

3 Binärziffern lassen sich zu einer Oktalzahl zusammenfassen:
$\underbrace{11}_{3_8}\underbrace{010}_{2_8}\underbrace{011_2}_{3_8} = 323_8$

Festkommazahlen

Eine Festkommazahl besteht aus einer **festen Anzahl von Ziffern vor und nach dem Komma**.

2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}
1	1	0	1	.	0	1	0	1

Gleitkommazahlen: IEEE 754

3 Formate:

- Single Precision: 32 Bit
- Double Precision: 64 Bit
- Extended Precision: 80 Bit

Basiert auf der wissenschaftlichen Notation.

Aufbau

Single Precision:	1 Bit Vorzeichen	8 Bit Exponent	23 Bit normalisierte Mantisse
Double Precision:	1 Bit Vorzeichen	11 Bit Exponent	52 Bit normalisierte Mantisse

Vorzeichen: $0 = +$; $1 = -$

Exponent: wird gespeichert, indem man den festen Biaswert (127:SP, 1023:DP) addiert.

Die Mantisse beginnt mit einem "Hidden Bit" (immer 1).

Dezimal zu IEEE 754

Beispiel: -62.058

1. Vorzeichen Bit bestimmen

   Vorzeichen Bit = 1

2. Zu pur Binär umwandeln

   $62.058_{10} = 111110.10010100_2$

3. Normalisieren für Mantisse und Exponent (ohne Bias)

   $111110.10010100_2 = 1.1111010010100_2 \cdot 2^5$

4. Exponent mit Bias bestimmen

   $5 + 127 = 132_{10} = 10000100_2$

5. Führende 1 der Mantisse abschneiden

   $1.1111010010100_2 \rightarrow 1111010010100_2$

6. Zusammenfügen

   $-62.058_{10} = \underbrace{1}_{\substack{\text{Vorzeichen}\\\text{Bit}}} \underbrace{10000100}_{\text{Exponent}} \underbrace{1111010010100}_{\text{Mantisse}}$

IEEE 754 zu Dezimal

Beispiel: 01000010011010100000000000000000

1. Vorzeichen bestimmen

   Vorzeichen: +

2. Exponent bestimmen (Bias muss abgezogen werden)

   $10000100_2 - 127_{10} = 132_{10} - 127_{10} = 5_{10}$

3. Mantisse bestimmen

$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
1	1	0	1	0	1

   $\frac{1}{2} + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} = \frac{53}{64} = 0.828125$

4. 1 zur Mantisse addieren (Hidden Bit) und Vorzeichen einrechnen

   1.828125

5. Ergebnis berechnen

   $1.828125 \cdot 2^5 = 58.5_{10}$

Fehlererkennung

Redundanzen

Eine Einheite von n Datenbits und k Redundanzbits nennt man **Codewort**.
Die **Länge** eines Codeworts ist insgesammt $n + k$.
Die Menge aller gültigen Codewörter nennt man **Code**.

Hamming-Distanz

Die Hamming Distanz zweier Codewörter ist gegeben als die Anzahl der Bitpositionen, in denen sie sich unterscheiden.

Beispiel: 11110000 und 11001100 $\implies$ Hamming-Distanz beträgt 4

Die Hamming Distanz eines Codes ist die kleinste Hamming-Distanz zweier Codewörter

Beispiel: $\{1100, 0011, 1111\}$ $\implies$ Hamming-Distanz beträgt 2

c-Bit Fehler können erkannt werden, wenn die Hamming-Distanz c+1 beträgt.
c-Bit Fehler können korrigiert werden, wenn die Hamming-Distanz $2c + 1$ beträgt.

Parität

Durch Hinzufügen eines **Paritätsbits** wird ein Code mit Hamming-Distanz 2 erzeugt.

Das Paritätsbit wird gesetzt sodass die Gesamtzahl der 1en...

... gerade ist

$$\underbrace{00100101}_{\textit{Datenbits}}\ \underbrace{1}_{\text{Paritätsbit}}$$

... ungerade ist

$$\underbrace{00100101}_{\textit{Datenbits}}\ \underbrace{0}_{\text{Paritätsbit}}$$

Zweidimensionale Parität

Die zweidimensionale Parität konstruiert einen Code mit Hamming-Distanz 4.
Dabei werden n Wörter zu je n Bits in einer $n \times n$-Matrix untereinandergeschrieben und über jede Zeile und jede Spalte je ein Paritätsbit berechnet.

Bei einem 1-Bit-Fehler stimmen die Paritätsbits genau einer Zeile und Spalte nicht.
Dann ist die Position des Fehlers klar und er kann korrigiert werden.

$$\text{fehlerfrei:}\quad \begin{array}{cccc|c} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 1 \end{array} \qquad \text{1-Bit-Fehler:}\quad \begin{array}{cccc|c} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & \color{red}{1} & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 1 \end{array}$$

Das Bit ganz unten rechts wird zur Paritätsberechnung der Paritätszeile und -spalte genutzt.

Hamming-Code

Ein Hamming-Code mit n **Redundanzbits** hat maximal $2^n - 1$ Bits und maximal $2^n - 1 - n$ Datenbits (mit $n \in \mathbb{N}$)
Die Bits des Codewortes werden, beginnend bei 1, durchnummeriert.
Das i-te **Prüfbit**(auch Redundanzbit) steht im Codewort an Position 2^i ($\implies 1, 2, 4, 8, \ldots$)

Beispiel (gerade Parität):

Position	Bits des Codewortes	
1_{10}	0	Prüfbit
2_{10}	1	Prüfbit
3_{10}	0	
4_{10}	0	Prüfbit
5_{10}	1	
6_{10}	0	
7_{10}	1	

Gespeichertes Datenwort: 0101

Berechnung der Prüfbits

Jedes Prüfbit ist ein Paritätsbit über eine eindeutige Menge von Bits.
Das i-te Prüfbit an Position 2^i wird über alle Stellen aus dem Codewort berechnet, für die in der Binärdarstellung für 2^i das Bit auf der jeweiligen Position auf 1 gesetzt ist.
Beispiel: 0. Prüfbit an Stelle $2^2 = 4_{10} = 100_2 \implies$ jedes Bit aus dem Codewort, in dessen Binärdarstellung der Position das Bit auf Position 2^2 gesetzt ist, wird zur Berechnung des Prüfbits verwendet.

Summenzeichen und Produktzeichen

Summenzeichen

Seien $m, n \in \mathbb{Z}$ mit $m \leq n$. Die Summen der Zahlen $a_m, a_{m+1}, \ldots, a_n$ wird folgendermaßen bezeichnet:

$$\sum_{i=m}^{n} a_i = a_m + a_{m+1} + \ldots + a_n$$

Dabei gilt: $i \widehat{=}$ **Summationsindex**; $m/n \widehat{=}$ **untere/obere Summationsgrenze**.
Rechenregeln:

$$\sum_{i=m}^{n} c \cdot a_i = c \cdot \sum_{i=m}^{n} a_i$$

$$\sum_{i=m}^{n} (a_i + b_i) = \sum_{i=m}^{n} a_i + \sum_{i=m}^{n} b_i$$

Leere Summe:

$$\sum_{i=m}^{n} := 0, \text{ für } m > n$$

Produktzeichen

Seien $m, n \in \mathbb{Z}$ mit $m \leq n$. Das Produkt der Zahlen $a_m, a_{m+1}, \ldots, a_n$ wird folgendermaßen bezeichnet:

$$\prod_{i=m}^{n} a_i = a_m \cdot a_{m+1} \cdot \ldots \cdot a_n$$

Dabei gilt: $i \widehat{=}$ **Laufindex**; $m/n \widehat{=}$ **untere/obere Grenze**.
Leeres Produkt:

$$\prod_{i=m}^{n} := 1, \text{ für } m > n$$

Rechenregeln

Bruchregeln

$$\frac{a}{b} = \frac{a \cdot c}{b \cdot c} \qquad \frac{a}{b} + \frac{c}{b} = \frac{a+c}{b}$$

$$\frac{a}{b} \cdot \frac{c}{d} = \frac{ac}{bd} \qquad \frac{a}{b} : \frac{c}{d} = \frac{a}{b} \cdot \frac{d}{c} = \frac{ad}{bc}$$

Potenzgesetze

$$a^n \cdot a^m = a^{n+m} \qquad a^n \cdot b^n = (a \cdot b)^n$$

$$(a^n)^m = (a^m)^n = a^{n \cdot m} \qquad a^{-n} = \frac{1}{a^n}, a \neq 0$$

$$a^0 = 1, a \in \mathbb{R}$$

Wurzelgesetze

$$\sqrt[n]{a^n} = a \qquad (\sqrt[n]{a})^n = a$$

$$\sqrt[n]{a \cdot b} = \sqrt[n]{a} \cdot \sqrt[n]{b} \qquad \sqrt[n]{\frac{a}{b}} = \frac{\sqrt[n]{a}}{\sqrt[n]{b}}, \; b \neq 0$$

$$a^{\frac{1}{n}} = \sqrt[n]{a} \qquad a^{-\frac{1}{n}} = \frac{1}{\sqrt[n]{a}}, a > 0$$

Logarithmengesetze

$$\log 1 = 0 \qquad \log e = 1$$

$$a^x = b \Leftrightarrow x = \log_a(b) \qquad \log(a^x) = x \log a$$

$$\log(x \cdot y) = \log x + \log y \qquad \log\left(\frac{x}{y}\right) = \log x - \log y$$

Trigonometrie

Bogenmaß

Der Bogenmaß ist die Länge des Kreisbogens des Einheitskreises und gibt den Betrag des Winkels an. Der Umfang des Einheitskreises beträgt 2π.

Bogenmaß	0	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{\pi}{2}$	π	$\frac{3\pi}{2}$	2π
Gradmaß	$0°$	$30°$	$45°$	$60°$	$90°$	$180°$	$270°$	$360°$

Umwandlung von Winkel α von Gradmaß zu Bogenmaß: Bogenmaß $= \alpha \frac{\pi}{180°}$
Umwandlung von Winkel α von Bogenmaß zu Gradmaß: Gradmaß $= \alpha \frac{180°}{\pi}$