

# Inhaltsverzeichnis

<b>Inhaltsverzeichnis</b>	<b>1</b>
<b>1 Logik</b>	<b>3</b>
1.1 Aussagenlogik . . . . .	3
1.2 Prädikatenlogik . . . . .	5
1.3 Beweistechniken . . . . .	6
<b>2 Naive Mengenlehre</b>	<b>9</b>
2.1 Quantitative Relationen . . .	11
2.2 Abbildungen . . . . .	12
2.3 Relationen . . . . .	14
<b>3 Analysis</b>	<b>17</b>
3.1 Reelle Zahlen . . . . .	17
3.2 Intervalle . . . . .	21
3.3 Folgen . . . . .	22
3.4 Grenzwerte . . . . .	24
3.5 Grenzwertsätze . . . . .	26
3.6 Reihen . . . . .	28
<b>4 Algorithmen auf Datenstrukturen</b>	<b>33</b>
4.1 Effizienz . . . . .	33
4.2 Suchverfahren . . . . .	36
4.3 Verkettete Listen . . . . .	39
4.4 Sortiervverfahren . . . . .	41
<b>5 Bäume</b>	<b>51</b>
5.1 Binärbäume . . . . .	53
<b>Index</b>	<b>55</b>



# 1

## Logik


### 1.1 Aussagenlogik


**Aussage** Satz/Formel entweder wahr oder falsch; „-form“ bei zu wenig Infos.

**Theoreme** sind wahre Aussagen.

#### Junktoren

**Negation**  $\neg A$  „Nicht“ (!, ~, )


**Konjunkt.**  $A \wedge B$  „und“ (&&, )

**Disjunkt.**  $A \vee B$  „oder“ (||, )

**Implikat.**  $A \Rightarrow B$  „Wenn, dann“ / „B“ ( $\rightarrow$ , **if**)

$A \Rightarrow B$  „A hinreichend“

$B \Rightarrow A$  „A notwendig“

**Äquiv.**  $A \Leftrightarrow B$  „Genau dann, wenn“ ( $\leftrightarrow$ ,  $\equiv$ ,  $==$ , )

**Wahrheitswertetabelle** mit  $2^n$  Zeilen für  $n$  Atome. Konstruktionssystematik: Frequenz pro Atom verdoppeln.

# 1. LOGIK

$A$	$B$	$\neg A$	$A \wedge B$	$A \vee B$	$A \Rightarrow B$	$A \Leftrightarrow B$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

☹ ☹ ☹

Äquivalente Formeln $\Leftrightarrow$		Bezeichnung
$A \wedge B$	$B \wedge A$	Kommutativ
$A \vee B$	$B \vee A$	
$A \wedge (B \wedge C)$	$(A \wedge B) \wedge C$	Assoziativ
$A \vee (B \vee C)$	$(A \vee B) \vee C$	
$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$	Distributiv
$A \vee (B \wedge C)$	$(A \vee B) \wedge (A \vee C)$	
$A \wedge A$	$A$	Idempotenz
$A \vee A$	$A$	
$\neg \neg A$	$A$	Involution
$\neg(A \wedge B)$	$\neg A \vee \neg B$	
$\neg(A \vee B)$	$\neg A \wedge \neg B$	DE-MORGAN
$A \wedge (A \vee B)$	$A$	
$A \vee (A \wedge B)$	$A$	Absorption
$A \Rightarrow B$	$\neg A \vee B$	
$\neg(A \Rightarrow B)$	$A \wedge \neg B$	Elimination
$A \Leftrightarrow B$	$(A \Rightarrow B) \wedge (B \Rightarrow A)$	

## Axiomatik

**Axiome** als wahr angenommene Aussagen; an Nützlichkeit gemessen.

Anspruch, aber nach GÖDELS Unvollständigkeitssatz nicht möglich:

- Unabhängig
- Vollständig
- Widerspruchsfrei

## 1.2 Prädikatenlogik

**Quantoren** Innerhalb eines Universums:

**Existenzq.**  $\exists$  „Mind. eines“

**Individuum**  $\exists!$  „Genau eines“

**Allq.**  $\forall$  „Für alle“

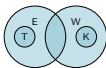
**Quantitative Aussagen**

**Erfüllbar**  $\exists x F(x)$

**Widerlegbar**  $\exists x \neg F(x)$

**Tautologie**  $\top = \forall x F(x)$  (alle Schlussregeln)

**Kontradiktion**  $\perp = \forall x \neg F(x)$



Klassische Tautologien	Bezeichnung
$A \vee \neg A$	Ausgeschlossenes Drittes
$A \wedge (A \Rightarrow B) \Rightarrow B$	Modus ponens
$(A \wedge B) \Rightarrow A$	Abschwächung
$A \Rightarrow (A \vee B)$	

### Negation (DE-MORGAN)

$$\neg \exists x F(x) \Leftrightarrow \forall x \neg F(x)$$

$$\neg \forall x F(x) \Leftrightarrow \exists x \neg F(x)$$

### Häufige Fehler

- $U = \emptyset^{\mathbb{C}}$  nicht notwendig
- $\exists x(P(x) \Rightarrow Q(x)) \not\Rightarrow \exists x P(x)$
- $\neg \exists x \exists y P(x, y) \Leftrightarrow \forall x \neg \exists y P(x, y)$

## 1.3 Beweistechniken



**Achtung:** Aus falschen Aussagen können wahre *und* falsche Aussagen folgen.

**Direkt**  $A \Rightarrow B$  Angenommen  $A$ , zeige  $B$ . Oder:  
Angenommen  $\neg B$ , zeige  $\neg A$   
(*Kontraposition*).

$$(A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)$$

**Fallunters.** Aufteilen, lösen, zusammenführen. O.B.d.A = „Ohne Beschränkung der Allgemeinheit“

**Widerspruch**  $(\neg A \Rightarrow \perp) \Rightarrow A$  Angenommen  $A \wedge \neg B$ , zeige Kontradiktion. (Reductio ad absurdum)

**Ring** (Transitivität der Implikation)

$$\begin{aligned} A &\Leftrightarrow B \Leftrightarrow C \Leftrightarrow \dots \\ &\equiv A \Rightarrow B \Rightarrow C \Rightarrow \dots \Rightarrow A \end{aligned}$$

**Induktion**  $F(n) \quad \forall n \geq n_0 \in \mathbb{N}$

1. **Anfang:** Zeige  $F(n_0)$ .
2. **Schritt:** Angenommen  $F(n)$  (Hypothese), zeige  $F(n+1)$  (Behauptung).

**Starke Induktion:** Angenommen  
 $F(k) \quad \forall n_0 \leq k \leq n \in \mathbb{N}$ .

### Häufige Fehler

- Nicht voraussetzen, was zu beweisen ist
- Äquival. von Implikat. unterscheiden (Zweifelsfall immer Implikat.)





## 2

# Naive Mengenlehre

**Mengen** Zusammenfassung versch. Objekte „Elemente“.


**Element**  $x \in M$  „enthält“


**Leere M.**  $\emptyset = \{\}$

**Universum**  $U$

**Einschränkung**  $\{x \mid F(x)\}$

## Relationen

**Teilmenge**  $N \subseteq M$   
 $\Leftrightarrow \forall n \in N : n \in M$  

**Gleichheit**  $M = N$   
 $\Leftrightarrow M \subseteq N \wedge N \subseteq M$  

## Mächtigkeit

$$|M| \begin{cases} = n & \text{endlich} \\ \geq \infty & \text{unendlich} \end{cases}$$
$$= |N| \Leftrightarrow \exists f_{\text{bijekt.}} : M \rightarrow N$$

**Abzählbar**  $\exists f_{\text{surj.}} : \mathbb{N} \rightarrow M$

- Endliche Mengen,  $\emptyset$ ,  $\mathbb{N}$ ,  $\mathbb{Z}$ ,  $\mathbb{Q}$
- $M_{\text{abz.}} \wedge N_{\text{abz.}} \Rightarrow (M \cup N)_{\text{abz.}}$   
 $(= \{m_1, n_1, m_2, n_2, \dots\})$
- $M_{\text{abz.}} \wedge N \subseteq M \Rightarrow N_{\text{abz.}}$

$\div$	1	2	3	4	...
1	$\frac{1}{1}$	$\frac{2}{1}$	$\frac{3}{1}$	$\frac{4}{1}$	...
2	$\frac{1}{2}$	$\frac{2}{2}$	$\frac{3}{2}$	$\frac{4}{2}$	...
3	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{3}{3}$	$\frac{4}{3}$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

*Note: Red diagonal lines and arrows in the original image indicate the sequence of elements visited in a zig-zag pattern across the grid of fractions.*

$$\begin{aligned}
 f(1) &= 0, \mathbf{r_{11}} r_{12} r_{13} r_{14} \dots \\
 f(2) &= 0, r_{21} \mathbf{r_{22}} r_{23} r_{24} \dots \\
 f(3) &= 0, r_{31} r_{32} \mathbf{r_{33}} r_{34} \dots \\
 f(4) &= 0, r_{41} r_{42} r_{43} \mathbf{r_{44}} \dots \\
 &\vdots
 \end{aligned}$$


(CANTORS Diagonalargumente)


## Operationen


**Vereinig.**  $M \cup N$

$$\Leftrightarrow \{x \mid x \in M \vee x \in N\}$$



**Schnitt**  $M \cap N \Leftrightarrow \{x \mid x \in M \wedge x \in N\}$   
(=  $\emptyset$  „disjunkt“) 

**Diff.**  $M \setminus N \Leftrightarrow \{x \mid x \in M \wedge x \notin N\}$  

**Komplement**  $M^c = \{x \mid x \notin M\}$  

Alle logischen Äquivalenzen gelten auch für die Mengenoperationen.

### Häufige Fehler

- $\forall M : \emptyset \subseteq M$  , nicht  $\forall M : \emptyset \in M$

## 2.1 Quantitative Relationen

Sei Indexmenge  $I$  und Mengen  $M_i \quad \forall i \in I$ .

$$\bigcup_{i \in I} M_i := \{x \mid \exists i \in I : x \in M_i\}$$

$$\bigcap_{i \in I} M_i := \{x \mid \forall i \in I : x \in M_i\}$$

### Neutrale Elemente

- $\bigcup_{i \in \emptyset} M_i = \emptyset$  („hinzufügen“)
- $\bigcap_{i \in \emptyset} M_i = U$  („wegnehmen“)

### Potenzmenge

$$\mathcal{P}(M) := \{N \mid N \subseteq M\}$$

$$|\mathcal{P}(M)| = 2^{|M|} \quad (\in / \notin \text{ binär})$$

### Auswahlaxiom (AC)

Für Menge  $\mathcal{X}$  nicht-leerer Mengen:

$$\begin{aligned}\exists c : \mathcal{X} &\rightarrow \bigcup \mathcal{X} \\ \forall X \in \mathcal{X} : c(X) &\in X\end{aligned}$$

Nutzung kennzeichnen!

## 2.2 Abbildungen

**Abbildung**  $f$  von  $X$  (Definitions- ) nach  $Y$  (Werteb. ) ordnet jedem  $x \in X$  eindeutig ein  $y \in Y$  zu.

**Totalität**  $\forall x \in X \exists y \in Y : f(x) = y$

**Eindeutigkeit**  $\forall x \in X \forall a, b \in Y : f(x) = a \wedge f(x) = b \Rightarrow a = b$

$$f : X \rightarrow Y$$

**Bilder**  $f(X') = \{f(x) \mid x \in X'\} \quad X' \subseteq X$

**Urbilder**  $f^{-1}(Y') = \{x \in X \mid f(x) \in Y'\} \quad Y' \subseteq Y$

**Graph**  $\text{gr}(f) := \{(x, f(x)) \mid x \in X\}$

**Identität**

$$\begin{aligned}\text{id}_A : A &\rightarrow A \\ \text{id}_A(a) &:= a \quad \forall a \in A\end{aligned}$$

**Umkehrfunktion**  $f^{-1} : Y \rightarrow X$  wenn  $f$  bijektiv und  $(f \circ f^{-1})(y) = y$  bzw.  $f; f^{-1} = \text{id}_Y \wedge f^{-1}; f = \text{id}_X$

Für die Relation  $f^{-1}$  gilt:

- $x \in f^{-1}(\{f(x)\})$
- $f(f^{-1}(\{y\})) = \{y\}$  falls  $f$  surjektiv

### Eigenschaften

**Injektiv**  $\forall x_1, x_2 \in X :$   
 $x_1 \neq x_2 \Leftrightarrow f(x_1) \neq f(x_2)$

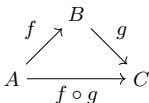
**Surjektiv**  $\forall y \in Y \exists x \in X : y = f(x)$

**Bijektiv/Invertierbar** wenn injektiv und surjektiv

**Verkettung**  $f \circ g : A \rightarrow C$

$$(f \circ g)(a) = f(g(a))$$

(der Reihenfolge nach)



## 2.3 Relationen

### Kartesisches Produkt

$$X_1 \times \cdots \times X_n := \{(x_1, \cdots, x_n) \\ | x_1 \in X_1, \cdots, x_n \in X_n\}$$

**Relation**  $\sim$  von/auf  $M$  nach  $N$  ist Teilmenge  $R \subseteq M \times N$ . ( $R' \subseteq N \times P$ )

$$m \sim n \Leftrightarrow (m, n) \in R$$

$$\equiv \text{Reflexiv } \forall x \in M : (\mathbf{x}, \mathbf{x}) \in R \\ \Leftrightarrow \text{id}_M \subseteq R$$

$$\text{Irreflexiv } \forall x \in M : (x, x) \notin R \\ \Leftrightarrow \text{id}_M \cap R = \emptyset$$

$$\equiv \text{Sym. } \forall (x, \mathbf{y}) \in R : (\mathbf{y}, x) \in R \\ \Leftrightarrow R \subseteq R^{-1}$$

$$\text{Antis. } \forall x, y : ((x, y) \in R \wedge (y, x) \in R) \Rightarrow \\ \mathbf{x} = \mathbf{y} \\ \Leftrightarrow R \cap R' \subseteq \text{id}_M$$

$$\equiv \text{Transitiv } \forall \mathbf{x}, y, \mathbf{z} : ((x, y) \in R \wedge (y, z) \in R) \Rightarrow (\mathbf{x}, \mathbf{z}) \in R \\ \Leftrightarrow R; R \subseteq R$$

$$\text{Vollst. } \forall \mathbf{x}, \mathbf{y} \in M : (x, y) \in R \vee (y, x) \in R \\ \Leftrightarrow R \cup R^{-1} = M \times M$$

## Spezielle Relationen

**Inverse Relation**  $R^{-1}$  mit  $R \in M \times N := \{(n, m) \in N \times M \mid (m, n) \in R\}$

**Komposition**  $R; R'$  mit  $R' \in N \times P := \{(m, p) \in M \times P \mid \exists n \in N : (m, n) \in R \wedge (n, p) \in R'\}$

**Leere Relation**  $\emptyset$

**Identität**  $\text{id}_M := \{(m, m) \mid m \in M\}$  (=)

**Allrelation**  $M \times M$

**Äquivalenzrelation**  $\equiv$  reflexiv, symmetrisch und transitiv. (Gleichheit\*\*\*)

**Äquivalenzklasse**  $[m]_{\equiv}$  auf  $M$ , Vertreter  $m \in M$ .

$$[m]_{\equiv} := \{x \in M \mid m \equiv x\}$$

$$\Leftrightarrow [m]_{\equiv} = [x]_{\equiv}$$

**Zerlegung**  $\mathcal{N} \subseteq \mathcal{P}(M)$  von  $M$ .

- $\emptyset \notin \mathcal{N}$
- $M = \bigcup \mathcal{N}$
- $N \cap N' = \emptyset$   
( $N, N' \in \mathcal{N} : N \neq N'$ )
- (Korrespondiert zur ÄR.)

**Quotient**  $(M / \equiv)$  Sei  $\equiv$  ÄR. auf  $M$ . (ist Zerlegung)

$$(M / \equiv) := \{[m]_{\equiv} \mid m \in M\}$$





# 3

## Analysis

### 3.1 Reelle Zahlen $\mathbb{R}$

#### Angeordnete Körper

(Gilt auch für  $\mathbb{Z}$  und  $\mathbb{Q}$ )

**Körperaxiome**  $(\mathbb{R}, +, *)$   $a, b, c \in \mathbb{R}$

**Addition**  $(\mathbb{R}, +)$

**Assoziativität**

$$a + (b + c) = (a + b) + c$$

**Kommutativität**

$$a + b = b + a$$

**Neutrales Element Null**

$$a + 0 = a \quad 0 \in \mathbb{R}$$

**Inverses „Negativ“**

$$a + (-a) = 0 \quad (-a) \in \mathbb{R}$$

**Multiplikation**  $(\mathbb{R}, *)$

**Assoziativität**  $a * (b * c) = (a * b) * c$

**Kommutativität**  $a * b = b * a$

**Neutrales Element Eins**

$$a * 1 = a \quad 1 \in \mathbb{R} \setminus \{0\}$$

**Inverses „Kehrwert“**

$$a * (a^{-1}) = 1$$

$$a \neq 0, (a^{-1}) \in \mathbb{R}$$

#### Distributivität

$$\mathbf{a} * (b + c) = \mathbf{a} * b + \mathbf{a} * c$$

#### Totale Ordnung

#### Transitivität

$$a < b \wedge b < c \Rightarrow a < c$$

#### Trichotomie Entweder

$$a < b \text{ oder } a = b \text{ oder } b < a \\ \Rightarrow \textit{Irreflexivität} (a < b \Rightarrow a \neq b)$$

#### Addition

$$a < b \Rightarrow a + c < b + c$$

#### Multiplikation

$$a < b \Rightarrow a * c < b * c \quad 0 < c$$

Bei Additiver oder Multiplikativer Inversion dreht sich die Ungleichung.

### Archimedes Axiom

$$\forall x \in \mathbb{R} \exists n \in \mathbb{N} : n > x \\ n > \frac{1}{x}$$

### Teilbarkeit

$$a|b \Leftrightarrow \exists n \in \mathbb{Z} : b = a * n$$

( $\Rightarrow \sqrt{2} \notin \mathbb{Q}$ , da mit  $\frac{a}{b} = \sqrt{2}$  nicht teilerfremd)

## Häufige Fehler

- Nicht durch Null teilen/kürzen
- Nicht  $-x < 0$  annehmen
- Multiplikation mit negativen Zahlen kehrt Ungleichungen

## Operationen

### Brüche

- $\frac{a}{b} * \frac{c}{d} = \frac{ac}{bd}$
- $\frac{a}{b} \stackrel{*d}{=} \frac{ad}{bd}$
- $\frac{a}{c} + \frac{b}{c} = \frac{a+b}{c}$
- $\frac{a}{b} + \frac{c}{d} = \frac{ad+cb}{bd}$

### Wurzeln $b^n = a \Leftrightarrow b = \sqrt[n]{a}$

- $\sqrt[n]{a * b} = \sqrt[n]{a} * \sqrt[n]{b}$
- $\sqrt[n]{\sqrt[m]{a}} = \sqrt[n * m]{a}$
- $\sqrt[n]{a} < \sqrt[n]{b} \quad 0 \leq a < b$
- ${}^{n+1}\sqrt{a} < \sqrt[n]{a} \quad 1 < a$
- $\sqrt[n]{a} < {}^{n+1}\sqrt{b} \quad 0 < a < 1$

$$\sqrt[n]{a^n} = |a| \quad a \in \mathbb{R}$$

**Potenzen**  $a^{\frac{x}{y}} = \sqrt[y]{a^x}$

- $a^{\times} * b^{\times} = (a * b)^{\times}$
- $a^x * a^y = a^{x+y}$
- $(a^x)^y = a^{x*y}$

## Dezimaldarstellung

**Gauss-Klammer**  $[y] := \max\{k \in \mathbb{Z} \mid k \leq y\} = \lfloor y \rfloor$

$$[y] = k \Leftrightarrow k \leq y < k + 1$$

**Existenz**  $\forall x \geq 0 \exists! (a_n)_{n \in \mathbb{N}}$  mit

- $a_n \in \{0, \dots, 9\} \quad \forall n \in \mathbb{N}$
- $\sum_{i=0}^n \frac{a_i}{10^i} \leq x < \sum_{i=0}^n \frac{a_i}{10^i} + \frac{1}{10^n} \quad \forall n \in \mathbb{N}_0$

Die Umkehrung gilt mit Lemma:

$$x = \sum_{n=0}^{\infty} \frac{a_n}{10^n}$$

**Lemma**  $x \geq 0, (a_n)_{n \in \mathbb{N}}$  Dezi. von  $x$

$$\neg (\exists N \in \mathbb{N} \forall n \geq N : a_n = 9)$$

$$x \in \mathbb{Q} \Leftrightarrow (a_n)_{n \in \mathbb{N}} \text{ periodisch}$$

## 3.2 Intervalle

Sei  $A \subseteq \mathbb{R}$ ,  $A \neq \emptyset$ ,  $a_0 \in A$ .

**Geschlossen**  $[a; b] := \{x \in \mathbb{R} \mid a \leq x \leq b\}$   
(„Ecken sind mit enthalten“)

**Offen**  $(a; b) := \{x \in \mathbb{R} \mid a < x < b\}$   
(Bei  $\infty$  immer offen, da  $\infty \notin \mathbb{R}$ )

### Kleinstes/Größtes Element

**Minimum**  $\min(A) := a_0$   
 $\Leftrightarrow \forall a \in A : a_0 \leq a$

**Maximum**  $\max(A) := a_0$   
 $\Leftrightarrow \forall a \in A : a \leq a_0$

$(\#^{\min}/_{\max}(a; b))$

**Beschränktheit**  $A$  heißt

**Oben beschränkt**  $\exists s \in \mathbb{R} \forall a \in A : a \leq s$

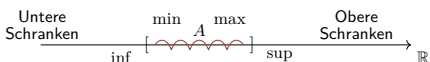
**Unten beschränkt**  $\exists s \in \mathbb{R} \forall a \in A : s \leq a$

### Vollständigkeit

**Infimum (klein)**  $\inf(A)$   
 $:= \max\{s \in \mathbb{R} \mid \forall a \in A : s \leq a\}$

**Supremum (groß)**  $\sup(A)$   
 $:= \min\{s \in \mathbb{R} \mid \forall a \in A : a \leq s\}$

**Vollständigkeitsaxiom**  $\exists \sup(A)$ .



## 3.3 Folgen

**Folge**  $(a_n)_{n \in \mathbb{N}}$  in  $A$  ist eine Abb.  $f : \mathbb{N} \rightarrow A$  mit  $a_n = f(n)$ .

**Arithmetische Folge**  $a_{n+1} = a_n + d$   
 $a_n = a + (n - 1) * d \quad d, a \in \mathbb{R}$

**Geometrische Folge**  $a_{n+1} = a_n * q$   
 $a_n = q^n \quad q \in \mathbb{R}$

**Rekursion**  $a_n$  ist auf  $a_{n-1}$  definiert.

$$a_{n+1} = F(n, a_n) \quad \forall n \in \mathbb{N}$$
$$F : A \times \mathbb{N} \rightarrow A$$

**Primfaktorzerlegung**  $n \in \mathbb{N}, n \geq 2$

$$\exists p_1, \dots, p_n \in \mathbb{P} : n = p_1 * \dots * p_n$$

## Summen und Produkte

**Summe**  $\sum_{i=1}^n i = 1 + 2 + \dots + n$

**Produkt**  $\prod_{i=1}^n i = 1 * 2 * 3 * \dots * n$

**Fakultät**  $n! = \prod_{i=1}^n i \quad (0! = 1)$

**Gaussche Summe**  $n \in \mathbb{N}$

$$\sum_{i=0}^n i = \frac{n * (n + 1)}{2}$$

**Geom. Summe**  $q \in \mathbb{R} \setminus \{0\}, n \in \mathbb{N}_0$

$$\sum_{i=0}^n q^i = \frac{1 - q^{n+1}}{1 - q}$$

**Bernoulli Unglei.**  $n \in \mathbb{N}_0, x \geq -1$

$$(1 + x)^n \geq 1 + nx$$

**Binom. Koeff.**  $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

- Rechnen:  $\frac{n > k}{0 < (n-k)}$
- $\binom{n}{0} = \binom{n}{n} = 1$
- $\binom{n+1}{k+1} = \binom{n}{k} + \binom{n}{k+1}$

**Binomischer Satz**  $n \in \mathbb{N}$

$$(a + b)^n = \sum_{k=0}^n \binom{n}{k} * a^{n-k} b^k$$

## 3.4 Grenzwerte

**Betrag**  $|x| := \begin{cases} x & 0 \leq x \\ -x & x < 0 \end{cases}$

**Lemma**  $|x * y| = |x| * |y|$

**Dreiecksungleichung**  $|x + y| \leq |x| + |y|$

**Umgekehrte Dreiecksungleichung**

$$||x| - |y|| \leq |x - y|$$

## Konvergenz

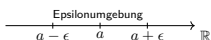
Sei  $(a_n)_{n \in \mathbb{N}} \subseteq \mathbb{R}, a \in \mathbb{R}$ .

$$a_n \xrightarrow{n \rightarrow \infty} a \Leftrightarrow$$

$$\forall \epsilon > 0 \exists n_0 \in \mathbb{N} \forall n \in \mathbb{N} n \geq n_0 :$$

$$|a_n - a| \leq \epsilon$$

$$(a - \epsilon \leq a_n \leq a + \epsilon)$$



•  $a_n \xrightarrow{n \rightarrow \infty} a \Leftrightarrow \lim_{n \rightarrow \infty} a_n = a$

Beschränkt + monoton  $\Rightarrow$  konvergent:

$$\lim_{n \rightarrow \infty} a_n = \begin{cases} \inf\{a_n \mid n \in \mathbb{N}\} & (a_n)_{\text{fall.}} \\ \sup\{a_n \mid n \in \mathbb{N}\} & (a_n)_{\text{steig.}} \end{cases}$$

**Nullfolgen**  $\lim_{n \rightarrow \infty} a_n = 0$

•  $\lim_{n \rightarrow \infty} \frac{1}{n^k} = 0 \quad k \in \mathbb{N}$



- $\lim_{n \rightarrow \infty} nq^n = 0$

### Folgen gegen 1

- $\lim_{n \rightarrow \infty} \sqrt[n]{a} = 1 \quad a > 0$
- $\lim_{n \rightarrow \infty} \sqrt[n]{n} = 1$

### Bestimmt Divergent

$$a_n \xrightarrow{n \rightarrow \infty} \infty \Leftrightarrow \forall R > 0 \exists n \geq n_0 \in \mathbb{N} : a_n \geq R$$

$$a_n \xrightarrow{n \rightarrow \infty} -\infty \Leftrightarrow \forall R < 0 \exists n \geq n_0 \in \mathbb{N} : a_n \leq R$$

$$\lim_{n \rightarrow \infty} q^n \begin{cases} = 0 & (-1; 1) \\ = 1 & = 1 \\ \geq \infty & > 1 \\ \text{div.} & \leq -1 \end{cases}$$

### Monotonie

#### Monoton fallend

$$a_n \underset{(\text{streng})}{\geq} a_{n+1} \quad \forall n \in \mathbb{N}$$

#### Monoton steigend

$$a_n \underset{(\text{streng})}{\leq} a_{n+1} \quad \forall n \in \mathbb{N}$$

### Beschränktheit

$$\exists k > 0 \forall n \in \mathbb{N} : |a_n| \leq k$$

- Konvergent  $\Rightarrow$  beschränkt
- Unbeschränkt  $\Rightarrow$  divergent

## 3.5 Grenzwertsätze

$$\lim_{n \rightarrow \infty} a_n = a, \lim_{n \rightarrow \infty} b_n = b$$

- $a_n \xrightarrow{n \rightarrow \infty} a \wedge a_n \xrightarrow{n \rightarrow \infty} b \Rightarrow a = b$  (Max. einen Grenzw.)
- $a = 0 \wedge (b_n)_{\text{beschr.}} \Leftrightarrow \lim_{n \rightarrow \infty} a_n b_n = 0$
- $a_n \leq b_n \Leftrightarrow a \leq b$  (nicht  $<$ )
- $\lim_{n \rightarrow \infty} \begin{cases} a_n \pm b_n = a \pm b \\ a_n * b_n = a * b \\ a_n * c = a * c \\ \sqrt[k]{a_n} = \sqrt[k]{a} \\ |a_n| = |a| \end{cases}$

### Einschachtelungssatz

$$\lim_{n \rightarrow \infty} a_n = \lim_{n \rightarrow \infty} b_n = a$$

$$\forall n \geq N \in \mathbb{N} : a_n \leq c_n \leq b_n$$

$$(\exists) \lim_{n \rightarrow \infty} c_n = a$$

### Spezielle Folgen

**Teilfolge** *streng mnt.* Folge  $(b_k)_{k \in \mathbb{N}}$  mit  $(n_k)_{k \in \mathbb{N}}$  sodass  $b_k = a_{n_k} \quad \forall k \in \mathbb{N}$ .

$$\lim_{n \rightarrow \infty} a_n = a \Rightarrow \lim_{n \rightarrow \infty} a_{n_k} = a$$

(da  $n_k$  mnt. steigend)

$$\forall (a_n)_{n \in \mathbb{N}} \exists (a_{n_k})_{k \in \mathbb{N}} \text{ mnt.}$$

(nicht streng!)

**Häufungspunkt**  $h$  mit einer Teilfolge

$$\lim_{n \rightarrow \infty} a_{n_k} = h$$

$$\bullet \lim_{n \rightarrow \infty} a_n = a \Leftrightarrow \exists! : h = a$$

**Bolzano-Weierstraß**

$$(a_n)_{n \in \mathbb{N}} \text{ beschr.} \Rightarrow \exists h \text{ Häuf.}$$

(Beschränkte Teilfolgen besitzen mind. einen Häufungspunkt)

**Cauchy-Folge**

$$\forall \epsilon > 0 \exists n_0 \in \mathbb{N} \forall n, m \geq n_0 :$$

$$|a_n - a_m| \leq \epsilon$$

(Konv. ohne bekannten Grenzwert)

**Vollständigkeit von  $\mathbb{R}$**

$$(a_n)_{n \in \mathbb{N}} \text{ CAUCHY} \Leftrightarrow \exists \lim_{n \rightarrow \infty} a_n$$

$$(\exists \lim_{n \rightarrow \infty} a_n \Rightarrow (a_n)_{n \in \mathbb{N}} \text{ CAUCHY})$$

$$\Rightarrow (a_n)_{n \in \mathbb{N}} \text{ beschr.}$$

$$\Rightarrow \exists h \quad (\text{BW})$$

$$\Rightarrow \lim_{n \rightarrow \infty} a_n = h)$$

## 3.6 Reihen

**Reihe**  $(s_n)_{n \in \mathbb{N}} = \sum_{k=1}^{\infty} a_k$  mit den Gliedern  $(a_k)_{k \in \mathbb{N}}$ .

**n-te Partialsumme**  $s_n = \sum_{k=1}^n a_k$

**Grenzwert** ebenfalls  $\sum_{k=1}^{\infty} a_k$ , falls  $s_n$  konvergiert

### Spezielle Reihen

**Geom.**  $\sum_{k=0}^{\infty} q^k = \frac{1}{1-q}$   $q \in (-1; 1)$

**Harmon.**  $\sum_{k=1}^{\infty} \frac{1}{k}$  divergent

**Allg. Harmon.**  $\sum_{k=1}^{\infty} \frac{1}{k^\alpha}$  konvergiert  $\forall \alpha > 1$

### Lemma

- $\sum_{k=1}^{\infty} a_k, \sum_{k=1}^{\infty} b_k$  konvergent
  - $\sum_{k=1}^{\infty} \mathbf{a}_k + \sum_{k=1}^{\infty} \mathbf{b}_k = \sum_{k=1}^{\infty} (\mathbf{a}_k + \mathbf{b}_k)$
  - $\mathbf{c} * \sum_{k=1}^{\infty} \mathbf{a}_k = \sum_{k=1}^{\infty} \mathbf{c} * \mathbf{a}_k$
- $\exists N \in \mathbb{N} : (\sum_{k=N}^{\infty} a_k)_{\text{konv.}} \Rightarrow (\sum_{k=1}^{\infty} a_k)_{\text{konv.}}$   
(Es reicht spätere Glieder zu betrachten)
- $(\sum_{k=1}^{\infty} a_k)_{\text{konv.}}$ 
  - $\Rightarrow \forall N \in \mathbb{N} : (\sum_{k=N}^{\infty} a_k)_{\text{konv.}}$
  - $\Rightarrow \lim_{N \rightarrow \infty} \sum_{k=N}^{\infty} a_k = 0$

## Konvergenzkriterien

### Cauchy

$$\Leftrightarrow \left( \sum_{k=1}^n a_k \right)_{n \in \mathbb{N}} \text{ CAUCHY}$$

$$\left( \sum_{k=1}^{\infty} a_k \right)_{\text{konv.}}$$

$$\Leftrightarrow \forall \epsilon > 0 \exists n_0 \in \mathbb{N} \forall n > m > n_0 :$$

$$\left| \sum_{k=m+1}^n a_k \right| \leq \epsilon$$

### Notwendig

$$\left( \sum_{n=1}^{\infty} a_n \right)_{\text{konv.}} \Rightarrow \lim_{n \rightarrow \infty} a_n = 0$$

$$\lim_{n \rightarrow \infty} a_n \neq 0 \Rightarrow \left( \sum_{n=1}^{\infty} a_n \right)_{\text{div.}}$$

**Beschränkt**  $a_n \geq 0 \ (\Rightarrow \text{mnt.}) \ \forall n \in \mathbb{N}$

$$\left( \sum_{n=1}^{\infty} a_n \right)_{\text{beschr.}} \Leftrightarrow \left( \sum_{n=1}^{\infty} a_n \right)_{\text{konv.}}$$

**Majorante**  $0 \leq a_n \leq b_k \ \forall n \in \mathbb{N}$

$$\left( \sum_{n=1}^{\infty} b_n \right)_{\text{konv.}} \Leftrightarrow \left( \sum_{n=1}^{\infty} a_n \right)_{\text{konv.}}$$

**Quotient**  $a_n \geq 0 \quad \forall n \in \mathbb{N}$

$$\lim_{n \rightarrow \infty} \frac{a_{n+1}}{a_n} \begin{cases} < 1 \rightarrow (\sum_{n=1}^{\infty} a_n)_{\text{konv.}} \\ > 1 \rightarrow (\sum_{n=1}^{\infty} a_n)_{\text{div.}} \end{cases}$$

**Wurzel**  $a_n \geq 0 \quad \forall n \in \mathbb{N}$

$$\lim_{n \rightarrow \infty} \sqrt[n]{a_n} \begin{cases} < 1 \rightarrow (\sum_{n=1}^{\infty} a_n)_{\text{konv.}} \\ > 1 \rightarrow (\sum_{n=1}^{\infty} a_n)_{\text{div.}} \end{cases}$$

**Absolut**

$$(\sum_{n=1}^{\infty} |a_n|)_{\text{konv.}} \Rightarrow (\sum_{n=1}^{\infty} a_n)_{\text{konv.}}$$

$$|\sum_{n=1}^{\infty} a_n| \leq \sum_{n=1}^{\infty} |a_n|$$

(Dreiecksungleichung)

**Leibniz**  $(a_n)_{n \in \mathbb{N}}$  mnt. Nullfolge

$$(\sum_{n=1}^{\infty} (-1)^n * a_n)_{\text{konv.}}$$

**Grenzwert**  $a_n, b_n \geq 0 \quad \forall n \in \mathbb{N}$

$$\lim_{n \rightarrow \infty} \frac{a_n}{b_n} > 0 \Rightarrow$$

$$(\sum_{n=1}^{\infty} a_n)_{\text{konv.}} \Leftrightarrow (\sum_{n=1}^{\infty} b_n)_{\text{konv.}}$$

## Exponentialfunktion

$$\exp(x) := \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

- $\exp(0) = 1$
- $\exp(1) = e \approx 2,71828 \notin \mathbb{Q}$   
 $e = \lim_{n \rightarrow \infty} (1 + \frac{1}{n})^n$

$$\exp(x) * \exp(y) = \exp(x + y)$$

## Cauchy-Produkt

$$(\sum_{n=0}^{\infty} a_n)(\sum_{n=0}^{\infty} b_n) = \sum_{n=0}^{\infty} \sum_{k=0}^n a_k b_{n-k}$$

## Korollar

- $\exp(x) > 0$
- $\frac{1}{\exp(x)} = \exp(-x)$
- $x < y \Rightarrow \exp(x) < \exp(y)$
- $\exp(r * x) = (\exp(x))^r$
- $\exp(r) = e^r$





# 4

## Algorithmen auf Datenstrukturen

**Algorithmus** Handlungsvorschrift aus endlich vielen Einzelschritten zur Problemlösung.

- Korrektheit (Test-based dev.)
- Terminierung (TOURING)
- Effizienz (Komplexität)

**Formen (High to low)** Menschl. Sprache, Pseudocode, Mathematische Ausdrücke, Quellcode, Binärcode

### **Divide & Conquer**

**Divide** Zerlegen in kleinere Teilprobleme

**Conquer** Lösen der Teilprobleme mit gleicher Methode (rekursiv)

**Merge** Zusammenführen der Teillösungen

### **4.1 Effizienz**

Raum/Zeit-Tradeoff: Zwischenspeichern vs. Neuberechnen

## 4. ALGORITHMEN AUF DATENSTRUKTUREN

---

Programmlaufzeit/-allokationen	Komplexität
Einfluss äußerer Faktoren	Unabh.
Konkrete Größe	Asymptotische Schätzung

**Inputgröße  $n$**  Jeweils

- Best-case  $C_B$
- Average-case
- Worst-case  $C_W$

### Asymptotische Zeit-/Speicherkomplexität

**Groß-O-Notation** Kosten  $C_f(n)$  mit  $g : \mathbb{N} \rightarrow \mathbb{R} \exists c > 0 \exists n_0 > 0 \forall n \geq n_0$

**Untere Schranke  $\Omega(f)$**   
 $C_f(n) \geq c * g(n)$

**Obere Schranke  $O(f)$**   
 $C_f(n) \leq c * g(n)$

**Exakte Schranke  $\Theta(f)$**   
 $C_f(n) \in \Omega(f) \cap O(f)$   
Polynom  $k$ ten Grades  $\in \Theta(n^k)$

(Beweis:  $g$  und  $c$  finden)

Groß-O	Wachstum	Klasse	
$O(1)$	Konstant		lösbar
$O(\log n)$	Logarithmisch		
$O(n)$	Linear		
$O(n \log n)$	Nlogn		
$O(n^2)$	Quadratisch	Polynomiell $O(n^k)$	hart
$O(n^3)$	Kubisch		
$O(2^n)$	Exponentiell	Exponentiell $O(\alpha^n)$	
$O(n!)$	Fakultät		
$O(n^n)$			

## Rechenregeln

**Elementare Operationen, Kontrollstr.**  $\in O(1)$

**Schleifen**  $\in i$  Wiederholungen  $* O(f)$  teuerste Operation

**Abfolge**  $O(g)$  nach  $O(f) \in O(\max(f; g))$

**Rekursion**  $\in k$  Aufrufe  $* O(f)$  teuerste Operation

**Mastertheorem**  $a \geq 1, b > 1, \Theta \geq 0$

$$T(n) = a * T\left(\frac{n}{b}\right) + \Theta(n^k)$$

$$\Rightarrow \begin{cases} \Theta(n^k) & a < b^k \\ \Theta(n^k \log n) & a = b^k \\ \Theta(n^{\log_b a}) & a > b^k \end{cases}$$

### Floor/Ceiling Runden

**Floor**  $\lfloor x \rfloor$  nach unten

**Ceiling**  $\lceil x \rceil$  nach oben

## 4.2 Suchverfahren

**Lineare Liste** endlich, geordnete (nicht sortierte) Folge  $n$  Elemente  $L := [a_0, \dots, a_n]$  gleichen Typs.

**Array** Sequenzielle Abfolge im Speicher, statisch, Index  $O(1)$ , schnelle Suchverfahren  $L[0] \mid$

**Sequenziell**  $C_A(n) = \frac{1}{n} * \sum^n i = \frac{n+1}{2} \in O(n)$

**Algorithm:** Sequential Search

**Input:** Liste  $L$ , Predikat  $x$

**Output:** Index  $i$  von  $x$

```
for  $i \leftarrow 0$  to  $L.len - 1$  do
    if  $x = L[i]$  then
        | return  $i$ 
    end
end
return  $-1$ 
```

**Auswahlproblem** Finde  $i$ -kleinstes Element in unsortierter Liste  $\in \Theta(n)$

**Algorithm:** *i*-Smallest Element

**Input:** Unsortierte Liste  $L$ , Level  $i$

**Output:** Kleinstes Element  $x$

```

 $p \leftarrow L[L.len - 1]$ 
for  $k = 0$  to  $L.len - 1$  do
    if  $L[k] < p$  then
        |  $\text{Push}(L_{<}, L[k])$ 
    if  $L[k] > p$  then
        |  $\text{Push}(L_{>}, L[k])$ 
    end
end
if  $L_{<}.len = i - 1$  then
    | return  $p$ 
if  $L_{<}.len > i - 1$  then
    | return i-Smallest Element  $L_{<}$ 
if  $L_{<}.len < i - 1$  then
    | return i-Smallest Element  $(L_{>}, i - 1 - L_{<}.len)$ 
end

```

## Sortierte Listen

**Binär**  $C_W(n) = \lfloor \log_2 n \rfloor + 1$ ,  $C_A(n) \stackrel{n \rightarrow \infty}{\approx} \log_2 n \in O(\log n)$

**Algorithm:** Binary Search

**Input:** Sortierte Liste  $L$ , Predikat  $x$

**Output:** Index  $i$  von  $x$

```

if  $L.len = 0$  then
    | return  $-1$ 
else
    |  $m \leftarrow \lfloor \frac{L.len}{2} \rfloor$ 
if  $x = L[m]$  then
    | return  $m$ 
if  $x < L[m]$  then
    | return Binary Search  $[L[0], \dots, L[m - 1]]$ 
if  $x > L[m]$  then
    | return  $m + 1 + \text{Binary Search}$ 
    |  $[L[m + 1], \dots, L[L.len - 1]]$ 
end

```

**Sprung** Kosten Vergleich  $a$ , Sprung  $b$  mit optimaler Sprungweite:

$$m = \lfloor \sqrt{\left(\frac{a}{b}\right) * n} \rfloor$$

$$C_A(n) = \frac{1}{2}(\lceil \frac{n}{m} \rceil * a + mb) \in O(\sqrt{n})$$

## 4. ALGORITHMEN AUF DATENSTRUKTUREN

---

**Algorithm:** Jump Search

**Input:** Sortierte Liste  $L$ , Predikat  $x$

**Output:** Index  $i$  von  $x$

$m \leftarrow \lfloor \sqrt{n} \rfloor$

**while**  $i < L.len$  **do**

$i \leftarrow i + m$

**if**  $x < L[i]$  **then**

**return** Search  $[L[i - m], \dots, L[i - 1]]$

**end**

**end**

**return**  $-1$

- $k$ -Ebenen Sprungsuche  $\in O(\sqrt[k]{n})$
- Partitionierung in Blöcke  $m$  möglich

### Exponentiell $\in O(\log x)$

**Algorithm:** Exponential Search

**Input:** Sortierte Liste  $L$ , Predikat  $x$

**Output:** Index  $i$  von  $x$

**while**  $x > L[i]$  **do**

$i \leftarrow 2 * i$

**end**

**return** Search  $[L \lfloor i/2 \rfloor, \dots, L[i - 1]]$

- Unbekanntes  $n$  möglich

### Interpolation $C_A(n) = 1 + \log_2 \log_2 n$ , $C_W(n) = O(n)$

**Algorithm:** Searchposition

**Input:** Listengrenzen  $[u, v]$

**Output:** Suchposition  $p$

**return**  $\lfloor u + \frac{x - L[u]}{L[v] - L[u]} (v - u) \rfloor$

**Algorithm:** Interpolation Search

**Input:** Sortierte Liste  $[L[u], \dots, L[v]]$ , Predikat  $x$

**Output:** Index  $i$  von  $x$

**if**  $x < L[u] \vee x > L[v]$  **then**

**return**  $-1$

$p \leftarrow \text{Searchposition}(u, v)$

**if**  $x = L[p]$  **then**

**return**  $p$

**if**  $x > L[p]$  **then**

**return** Interpolation Search( $p + 1, v, x$ )

**else**

**return** Interpolation Search( $u, p - 1, x$ )

**end**

**Häufigkeitsordnungen** mit Zugriffswahrscheinlichkeit  $p_i$ :  $C_A(n) = \sum_{i=0}^n i p_i$

**Frequency-count** Zugriffszähler pro Element

**Transpose** Tausch mit Vorgänger

**Move-to-front**

### 4.3 Verkettete Listen

**Container** Jedes Element  $p$  ist in der Form  $p \rightarrow \boxed{(\text{key}) \mid \text{value} \mid \text{next}}$ . Index ist seq. Suche  $\in O(n)$

**Löschen**  $\in O(1)$

**Algorithm:** Delete

**Input:** Zeiger  $p$  auf *Vorgänger* des löschendes Elements

```
if  $p \neq \emptyset \wedge p \rightarrow \text{next} \neq \emptyset$  then  
  |  $p \rightarrow \text{next} \leftarrow (p \rightarrow \text{next}) \rightarrow \text{next}$   
end
```

- desh. sehr dynamisch

**Suchen**  $C_A(n) = \frac{n+1}{2} \in O(n)$

**Algorithm:** Search Linked List

**Input:** Verkettete Liste  $L$ , Predikat  $x$

**Output:** Zeiger  $p$  auf  $x$

```
 $p \leftarrow L.\text{head}$  while  $p \rightarrow \text{value} \neq x$  do  
  |  $p \leftarrow p \rightarrow \text{next}$   
end  
return  $p$ 
```

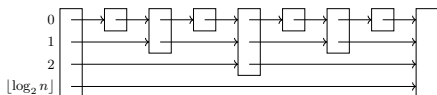
## 4. ALGORITHMEN AUF DATENSTRUKTUREN

### Doppelt Verkettet Zeiger auf Vorgänger

(key)		value		prev		next
-------	--	-------	--	------	--	------

- Bestimmung des Vorgängers (bei Einfügen, Löschen)  $\in O(1)$  statt  $O(n)$
- Höherer Speicheraufwand

### Skip



- Zeiger auf Ebene  $i$  zeigt zu nächstem  $2^i$  Element
- Suchen  $\in O(\log n)$

**(Perfekt)** Einfügen, Löschen  $\in O(n)$  (Vollst. Reorga.)

**Randomisiert** Höhe zufällig (keine vollst. Reorga.)

$P(h) = \frac{1}{2^{h+1}}$ : Einfügen, Löschen  $\in O(\log n)$

## Spezielle Listen

**ADT** „Abstrakte Datentypen“

**Stack**  $S = |\text{TOP}, \dots$  Operationen nur auf letztem Element  $\in O(1)$

**Queue**  $Q = ||\text{HEAD}, \dots, \text{TAIL}$  Vorne Löschen, hinten einfügen  $\in O(1)$



**Priority Queue**  $P = \begin{bmatrix} p_0 & p_1 & \cdots & p_n \\ a_0 & a_1 & \cdots & a_n \end{bmatrix}$  Jedes Element  $a$  hat Priorität  $p$ ; Entfernen von Element mit höchster (MIN) Priorität

## 4.4 Sortiervverfahren

### Sortierproblem

**Gegeben** (endliche) Folge von Schlüsseln (von Daten)  $(K_i)_{i \in I}$

**Gesucht** Bijektive Abbildung  $\pi : I \rightarrow I$  (Permutation), sodass  $K_{\pi(i)} \leq K_{\pi(i+1)}$   $\forall i \in I$

mit Optimierung nach geringen

- Schlüsselvergleichen  $C$
- Satzbewegungen  $M$

### Eigenschaften

**Ordnung** *Allgemein* vs. *speziell*: Ordnung wird nur über Schlüsselvergleiche hergestellt

**Relation** *Stabil* vs. *instabil*: Vorherig relative Reihenfolge bleibt erhalten

**Speicher** *In situ* vs. *ex situ*: Zusätzlicher Speicher notwendig

**Lokal** *Intern* vs. *extern*: Alles im RAM oder Mischung vorsortierter externer Teilfolgen

## 4. ALGORITHMEN AUF DATENSTRUKTUREN

---

**Ordnung**  $\forall x, y \in X$

**Reflexiv**  $x \leq x$

**Antisym.**  $x \leq y \wedge y \leq x \Rightarrow x = y$

**Transitiv**  $x \leq y \wedge y \leq z \Rightarrow x \leq z$

**Total (Vollständig)**  $x \leq y \vee y \leq x$

(ohne Total: „*Halbordnung*“)

### Grad der Sortierung

**Anzahl der Inversionen** Anzahl kleinerer Nachfolger für jedes Element:

$$\begin{aligned} \text{inv}(L) &:= |\{(i, j) \mid \\ &0 \leq i < j \leq n - 1, \\ &L[i] \geq L[j]\}| \end{aligned}$$

**Anzahl der Runs** Ein *Run* ist eine sortierte Teilliste, die nicht nach links oder rechts verlängert werden kann. Die Anzahl der Runs ist:

$$\begin{aligned} \text{runs}(L) &:= |\{i \mid \\ &0 \leq i < n - 1, \\ &L[i + 1] < L[i]\}| + 1 \end{aligned}$$

**Längster Run** Anzahl der Elemente der längsten sortierten Teilliste:

$$\begin{aligned} \text{las}(L) &:= \max\{r.\text{len} \mid \\ &r \text{ ist Run in } L\} \\ \text{rem}(L) &:= L.\text{len} - \text{las}(L) \end{aligned}$$

## Einfache Sortiervverfahren $O(n^2)$

**Selection** Entferne kleinstes Element in unsortierter Liste und füge es sortierter Liste an.

**Algorithm:** Selectionsort

**Input:** Liste  $L$

**Output:** Sortierte Liste  $L$

```

for  $i \leftarrow 0$  to  $L.len - 2$  do
    min  $\leftarrow i$ 
    for  $j \leftarrow i + 1$  to  $L.len - 1$  do
        if  $L[i] < L[min]$  then
            min  $\leftarrow j$ 
        end
    if min  $\neq i$  then
        Swap  $L[min], L[i]$ 
    end
end
if  $L.len = 0$  then
    return -1

```

**Insertion** Verschiebe erstes Element aus unsortierter Liste von hinten durch sortierte Liste, bis das vorgehende Element kleiner ist.

**Algorithm:** Insertionsort

**Input:** Liste  $L$

**Output:** Sortierte Liste  $L$

```

for  $i \leftarrow 1$  to  $L.len - 1$  do
    if  $L[i] < L[i - 1]$  then
        temp  $\leftarrow L[i]$ 
         $j \leftarrow i$ 
        while temp  $< L[j - 1] \wedge j > 0$  do
             $L[j] \leftarrow L[j - 1]$ 
             $j \leftarrow j - 1$ 
        end
         $L[j] \leftarrow temp$ 
    end
end

```

**Bubble** Vertausche benachbarte Elemente, durchlaufe bis nichts vertauscht werden muss.

**Achtung:** Die hinteren Elemente können im Durchlauf ignoriert werden!

## 4. ALGORITHMEN AUF DATENSTRUKTUREN

---

**Algorithm:** Bubblesort

**Input:** Liste  $L$

**Output:** Sortierte Liste  $L$

$i \leftarrow L.\text{len}$

swapped  $\leftarrow 1$

**while** swapped **do**

    swapped  $\leftarrow 0$

**for**  $j \leftarrow 0$  **to**  $i - 2$  **do**

**if**  $L[j] > L[j + 1]$  **then**

            Swap  $L[j], L[j + 1]$

            swapped  $\leftarrow 1$

**end**

$i \leftarrow i - 1$

**end**

### Verbesserte Sortiervverfahren $O(n \log n)$

**Shell** Insertionsort, nur werden Elemente nicht mit Nachbarn getauscht, sondern in  $t$  Sprüngen  $h_i$ , die kleiner werden (Kamm). Im letzten Schritt dann Insertionsort ( $h_t = 1$ ); somit Sortierung von grob bis fein, also Reduzierung der Tauschvorgänge.

**Algorithm:** Shellsort

**Input:** Liste  $L$ , Absteigende Liste von Sprunggrößen  $H$

**Output:** Sortierte Liste  $L$

**foreach**  $h$  **in**  $H$  **do**

**for**  $i \leftarrow h$  **to**  $L.\text{len} - 1$  **do**

        temp  $\leftarrow L[i]$

**for**  $j \leftarrow i$ ; temp  $< L[j - h] \wedge j \geq h$ ;

$j \leftarrow j - h$  **do**

$L[j] \leftarrow L[j - h]$

**end**

$L[j] \leftarrow \text{temp}$

**end**

**end**

**Quick** Rekursiv: Pivot-Element in der Mitte, Teillisten  $L_{<}$ ,  $L_{>}$ , sodass  $\forall l_{<} \in L_{<} \forall l_{>} \in L_{>} : l_{<} < x < l_{>}$ . Zerlegung: Durchlauf von Links bis  $L[i] \geq x$  und von Rechts bis  $L[j] \leq x$ , dann tauschen.

**Algorithm:** Quicksort

**Input:** Liste  $L$ , Indices  $l, r$

**Output:**  $L$ , sortiert zwischen  $l$  und  $r$

```

if  $l \geq r$  then
  | return
 $i \leftarrow l$ 
 $j \leftarrow r$ 
 $piv \leftarrow L[\lfloor \frac{l+r}{2} \rfloor]$ 
do
  | while  $L[i] < piv$  do
  | |  $i++$ 
  | end
  | while  $L[j] > piv$  do
  | |  $j--$ 
  | end
  | if  $i \leq j$  then
  | | Swap  $L[i], L[j]$ 
  | |  $i++$ 
  | |  $j--$ 
while  $i \leq j$ ;
Quicksort( $L, l, j$ )
Quicksort( $L, i, r$ )

```

**Turnier** Liste also Binärbaum, bestimme  $\min(L)$  durch Austragen des Turniers, entferne Sieger und wiederhole von Siegerpfad aus.

**Heap** Stelle Max-Heap (größtes Element in der Wurzel) her, gib Wurzel aus und ersetze mit Element ganz rechts in unterster Ebene.

**Algorithm:** Max-Heapify

**Input:** Liste  $L$ , Index  $i$  der MHE widerspricht und  $\forall j > i$  erfüllen MHE

**Output:** Liste  $L$  mit MHE  $\forall j \geq i$

```

 $l \leftarrow 2i + 1$ 
 $r \leftarrow 2i + 2$ 
if  $l < L.len \wedge L[l] > L[i]$  then
  |  $largest \leftarrow l$ 
else
  |  $largest \leftarrow i$ 
end
if  $r < L.len \wedge L[r] > L[largest]$  then
  |  $largest \leftarrow r$ 
if  $largest \neq i$  then
  | Swap  $L[i], L[largest]$ 
  | Max-Heapify  $L, largest$ 
end

```

## 4. ALGORITHMEN AUF DATENSTRUKTUREN

---

**Algorithm:** Build-Max-Heap

**Input:** Liste  $L$

**Output:** Liste  $L$  mit MHE

```
for  $i \leftarrow \lfloor \frac{L.len}{2} \rfloor - 1$  to 0 do
    | Max-Heapify  $L, i$ 
end
```

**Algorithm:** Heapsort

**Input:** Liste  $L$

**Output:** Sortierte Liste  $L$

Build-Max-Heap  $L$

```
for  $i \leftarrow L.len - 1$  to 1 do
    | Swap  $L[0], L[i]$ 
    |  $L.len \leftarrow$ 
    | Max-Heapify  $L, 0$ 
end
```

**Merge** Zerlege Liste in  $k$  Teile, sortiere diese (mit Mergesort) und verschmelze die sortierten Teillisten (merge).

**Algorithm:** 2-Merge

**Input:** Liste  $L$  mit  $L[l \dots m - 1]$  und  $L[m \dots r]$  sortiert, Indices  $l, m, r$

**Output:** Liste  $L$  mit  $L[l \dots r]$  sortiert

```
 $j \leftarrow l$ 
 $k \leftarrow m$ 
for  $i \leftarrow 0$  to  $r - l$  do
    | if  $k > r \vee (j < m \wedge L[j] \leq L[k])$  then
    |     |  $B[i] \leftarrow L[j]$ 
    |     |  $j \leftarrow j + 1$ 
    | else
    |     |  $B[i] \leftarrow L[k]$ 
    |     |  $k \leftarrow k + 1$ 
    | end
end
 $B$ 
for  $i \leftarrow 0$  to  $r - l$  do
    |  $L[l + i] \leftarrow B[i]$ 
end
```

**Algorithm:** Rekursives 2-Mergesort

**Input:** Liste  $L$ , Indices  $l, r$

**Output:** Liste  $L$  mit  $L[l \dots r]$  sortiert

```
if  $l \geq r$  then
    | return
else
    |  $m \leftarrow \lfloor \frac{l+r+1}{2} \rfloor$ 
    | Mergesort  $L, l, m - 1$ 
    | Mergesort  $L, m, r$ 
    | Merge  $L, l, m, r$ 
end
```

### Iteratives 2-Mergesort

**Algorithm:** Iteratives 2-M

**Input:** Liste  $L$

**Output:** Sortierte Liste  $L$

**for**  $k \leftarrow 2; k < n; k \leftarrow 2k$

**for**  $i \leftarrow 0; i + k < n; i \leftarrow i + k$

        Merge  $L, i, i + k$

**end**

**end**

Merge  $L, 0, n - 1, \frac{k}{2}$

**Natürliches Mergesort** Verschmelzen von benachbarten Runs (Ausnutzen der Vorselektion)

### Untere Schranke allgemeiner Sortierverfahren

Jedes allgemeine Sortiervverfahren benötigt im Worst- und Average-case Schlüsselvergleiche von mindestens:

$$\Omega(n \log n)$$

(Siehe Pfadlänge auf Entscheidungsbaum)

### Spezielle Sortiervverfahren $O(n)$

**Distribution** Abspeichern der Frequenz jedes Elementes  $k$  auf  $F[k]$ ; Ausgeben jedes Index  $F[k]$  mal.

**Lexikographische Ordnung**  $\leq$  Sei  $A = \{a_1, \dots, a_n\}$  ein Alphabet, dass sich mit gegebener Ordnung  $a_1 < \dots < a_n$  wie folgt auf dem Lexi-

## 4. ALGORITHMEN AUF DATENSTRUKTUREN

---

kon  $A^* = \bigcup_{n \in \mathbb{N}_0} A^n$  fortsetzt:

$$\begin{aligned} v &= (v_1, \dots, v_p) \leq w = (w_1, \dots, w_q) \\ \Leftrightarrow \forall 1 \leq i \leq p : v_i &= w_i \quad p \leq q \\ \vee \forall 1 \leq j \leq i : v_j &= w_j \quad v_i < w_i \end{aligned}$$

**Fachverteilen** Sortieren von  $n$   $k$ -Tupeln in  $k$  Schritten: Sortieren nach letztem Element, vorletztem usw.

### Große Datensätze sortieren

**Indirekt** Liste von Zeigern  $Z[i] = i$  auf die eigentlichen Listenelemente. Schlüsselvergleiche mit  $L[Z[i]]$ , Satzbewegungen nur als Zeigertausch in  $Z$ . Anschließend linear kopieren.

**Extern** Zerlegen in  $m$  Blöcke, sortieren im Hauptspeicher (Run) der mind.  $m+1$  Blöcke groß ist, verschmelzen der Runs ( $m$ -Wege-Merge).

**Ausgeglichenes 2-Wege-Mergesort** Daten auf Band  $n$ , sortieren von Block  $r_1 < n$  auf zweites Band und  $r_2$  auf drittes Band, löschen des ersten Bandes und Merge  $2r$  abwechselnd auf erstes (neues  $2r_1$ ) und viertes Band (neues  $2r_2$ ) und wiederholen.

**Replacement Selectionsort** Lese  $r < n$  Elemente auf Priority-Queue  $Q$ . Falls  $x =$



## 4.4. Sortierverfahren

$\min(Q) \geq$  letztem Element auf zweiten Band, schreibe  $x$  aus, sonst schreibe  $Q$  auf Band. Wiederhole auf dritten Band und dann merge.



Algo.	Stabil	Mem.	Schlüsselvergleiche			Satzbewegungen			
			$C_B$	$C_A$	$C_W$	$M_B$	$M_A$	$M_W$	
Selection	✗	1	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	$3(n-1)$	$3(n-1)$	$3(n-1)$	$O(n^3)$
Insertion	✓	1	$n-1$	$\stackrel{n \rightarrow \infty}{\sim} \frac{n(n-1)}{2} + n - \ln n$	$\frac{n(n-1)}{2}$	$2(n-1)$	$\frac{n^2+3n-4}{4} + n - 1$	$\frac{n^2+3n-4}{2}$	
Bubble	✓	1	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	0	$\frac{3n(n-1)}{4}$	$\frac{3n(n-1)}{2}$	
			Best-case		Average-case		Worst-case		
Shell	✗	1	-		-		-		$O(n \log n)$
Quick	✗	$\log n$	$n \log n$		$n \log n$		$n^2$		
Turnier	✗	$2n-1$	$n \log n$		$n \log n$		$n \log n$		
Heap	✗	1	$n \log n$		$n \log n$		$n \log n$		
Merge	✓	$n$	$n \log n$		$n \log n$		$n \log n$		
Untere Schranke $\Omega(n \log n)$ für allgemeine Sortierverfahren									
Distribution	✓	$n$	$n$		$n$		$n \log n, n^2$	$O(n)$	



# 5

## Bäume

- Verallg. von Listen: Element/Knoten kann mehrere Nachfolger haben
- Darstellung von Hierarchien

**Ungerichteter Graph**  $(V, E)$  mit einer Menge Knoten  $V$  und Kanten  $E \subseteq V \times V$

**Baum** Ungerichteter Graph mit

**Einfach** keine Schleife () oder Doppelkanten ()

**Zusammenhängend** Für jede zwei Knoten gibt es genau eine Folge von Kanten die sie verbindet

**Azyklisch** kein Zyklus (Cycle)

**Wurzelbaum** Baum mit genau einem Knoten der Wurzel heißt

**Orientierter Wurzelbaum** Alle Knoten sind Wurzel ihrer disjunkten Unterbäume und haben verschiedene Werte gleichen Typs. (Im Nachfolgenden einfach nur „Baum“)

### Darstellungsarten

**Array**

**Verkettete Liste**

**Graph**

**Einrückung**

**Menge**

**Klammer**

### Größen

**Ordnung** Max. Anzahl von Kindern jedes Knoten eines Baums

**Tiefe** Anzahl Kanten zwischen einem Knoten und Wurzel

**Stufe** Alle Knoten gleicher Tiefe

**Höhe** Max. Tiefe +1

### Eigenschaften

**Geordnet** Kinder erfüllen Ordnung von links nach rechts

**Vollständig** Alle Blätter auf gleicher Stufe, jede Stufe hat max. Anzahl von Kindern

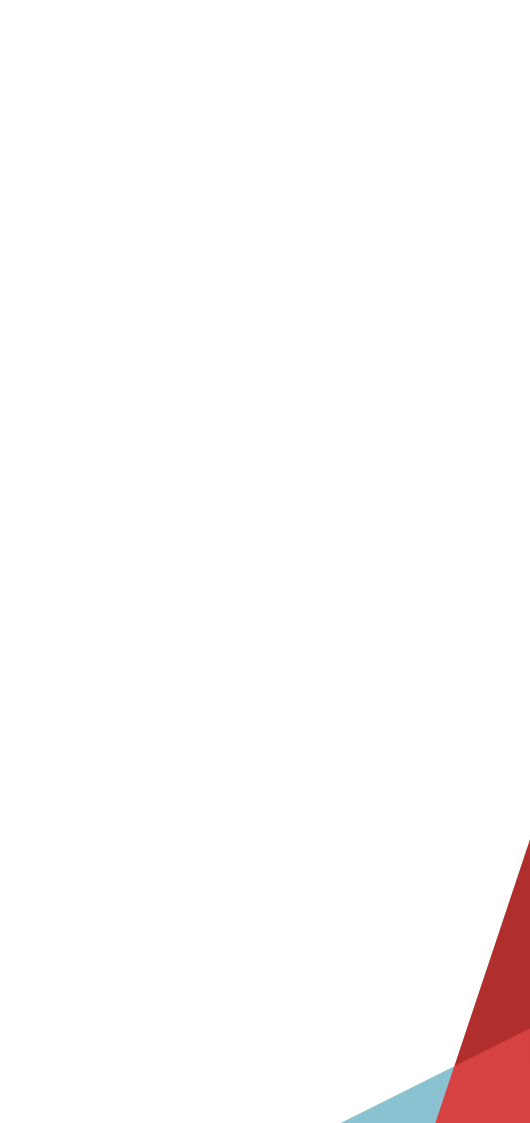
### 5.1 Binärbäume

Geordneter, orientierter Wurzelbaum der Ordnung 2 (oft auch nur 0 oder 2 Kinder)

Somit mit für  $i$  Stufen max.  $2^i$  Knoten

**Strikt** Jeder Knoten hat 0 oder 2 Kinder  
(Kein Knoten hat genau 1 Kind).

**Vollständig** Jeder Knoten auSSer der letzten Stufe hat genau 2 Kinder.



# Index

DE-MORGAN, 4  
ARCHIMEDES Axiom, 18  
BERNOULLI Ungleichung, 23  
BOLZANO-WEIERSTRASS, 27  
CAUCHY-Folge, 27  
CAUCHY-Kriterium, 29  
CAUCHY-Produkt, 31  
GAUSSCHE Summenformel, 23  
GAUSS-Klammer, 20

Abbildung, 12  
Abbildungsidentität, 12  
Abgeschlossen, 21  
Abschwächung, 5  
Absolute Konvergenz, 30  
Absorption, 4  
Abstrakter Datentyp, 40  
Abzählbarkeit, 10  
Addition, 17  
Algorithmus, 33  
Allgemeine Harmonische Reihe,  
28  
Allgemeine Suche, 41  
Allquantor, 5  
Allrelation, 15  
Angeordnete Körper, 17  
Antisymmetrie, 14  
Anzahl der Inversionen, 42  
Arithmetische Folge, 22  
Array, 36  
Assoziativität, 17  
Ausgeglichenes 2-Wege-Mergesort,  
48  
Ausgeschlossenes Drittes, 5  
Aussage, 3  
Auswahlaxiom, 12  
Auswahlproblem, 36  
Average-case Komplexität, 34  
Axiom, 4  
Azyklischer Graph, 51

Baum, 51  
Baumhöhe, 52  
Baumordnung, 52  
Baumstufe, 52  
Beschränkte Folge, 25  
Beschränkte Menge, 21  
Best-case Komplexität, 34  
Bestimmt Divergent, 25  
Betrag, 24  
Bijektiv, 13  
Bild, 12

Binomial Koeffizient, 23  
Binomischer Satz, 23  
Binärbaum, 53  
Binäre Suche, 37  
Bruch, 19  
Bubblesort, 43  
Bucketsort, 47

Ceiling, 36

Definitionsbereich, 12  
Dezimaldarstellung, 20  
Diagonalargumente, 10  
Differenz, 11  
Direkter Beweis, 6  
Disjunktion, 3  
Distributionsort, 47  
Distributivität, 18  
Divide & Conquer, 33  
Doppelt Verkettete Listen, 40  
Dreiecksungleichung, 24

Effizienz, 33  
Einfacher Graph, 51  
Eins, 17  
Einschachtelungssatz, 26  
Einschränkung, 9  
Element, 9  
Elimination, 4  
endlich, 9  
Epsilonumgebung, 24  
Erfüllbar, 5  
Eulersche Zahl, 31  
Ex situ Suche, 41  
Exakte Schranke Komplexität,  
34  
Existenzquantor, 5  
Exponentialfunktion, 31  
Exponentielle Spruchsuche, 38  
Externe Suche, 41  
Externes Sortieren, 48

Fachverteilen, 48  
Fakultät, 22  
Fallunterscheidung, 6  
Floor, 36  
Folge, 22  
Frequency-count Regel, 39  
Funktion, 12  
Funktionsgraph, 12

Geometrische Folge, 22  
Geometrische Reihe, 28  
Geometrische Summe, 23  
Geordneter Baum, 52  
Gleichmächtigkeit, 9  
Grad der Sortierung, 42  
Grenzwertkriterium, 30  
Grenzwertsätze, 26  
Groß-O-Notation, 34

Halbordnung, 42  
Heapsort, 45  
High-level Sprache, 33  
Hinreichende Bedingung, 3  
Häufigkeitsgeordnete Listen, 39  
Häufungspunkt, 27

Idempotenz, 4  
Implikation, 3  
In situ Suche, 41  
Indirektes Sortieren, 48  
Individuum, 5  
Induktion, 7  
Induktionsanfang, 7  
Induktionsbehauptung, 7  
Induktionshypothese, 7  
Induktionsschritt, 7  
Infimum, 21  
Injektiv, 13  
Insertionsort, 43  
Instabile Suche, 41  
Interne Suche, 41  
Interpolationssuche, 38  
Intervall, 21  
Inverse Relation, 15  
Involution, 4  
Irrationalität, 18  
Irreflexivität, 14  
Irreflexivität der Ordnung, 18

Kartesches Produkt, 14  
Kehrwert, 17  
Knotentiefe, 52  
Kommutativität, 17  
Komplement, 11  
Komplexitätsklassen, 35  
Komposition, 15  
Konjunktion, 3  
Kontradiktion, 5  
Kontraposition, 6  
Konvergenz, 24  
Kreuzprodukt, 14  
Körperaxiome, 17

Leere Menge, 9

Leere Relation, 15  
Leibniz-Kriterium, 30  
Lexikographische Ordnung, 47  
Limes, 24  
Lineare Liste, 36  
Logische Assoziativität, 4  
Logische Distributivität, 4  
Logische Kommutativität, 4  
Low-level Sprache, 33  
Längster Run, 42

Majorante, 29  
Majorantenkriterium, 29  
Mastertheorem, 35  
Maximum, 21  
Menge, 9  
Mengengleichheit, 9  
Mengenquotient, 15  
Mergesort, 46  
Minimum, 21  
Minorante, 29  
Modus ponens, 5  
Monoton steigend, 25  
Monotonie, 25  
Monoton fallend, 25  
Move-to-front Regel, 39  
Multiplikation, 17  
Mächtigkeit, 9

Natürliches Mergesort, 47  
Negation, 3, 6  
Negatives, 17  
Neutrale Mengenelemente, 11  
Notwendige Bedingung, 3  
Null, 17  
Nullfolge, 24

O.B.d.A, 6  
Obere Schranke Komplexität, 34  
Obere Schranken, 21  
Offen, 21  
Operation, 10  
Ordnung, 42  
Orientierter Wurzelbaum, 51

Partialsumme, 28  
Perfekte Skip-Liste, 40  
Potenz, 20  
Potenzmenge, 11  
Primfaktorzerlegung, 22  
Priority Queue, 41  
Produkt, 22  
Prädikatenlogik, 5

Quantor, 5



- Queue, 40
- Quicksort, 44
- Quotientenkriterium, 30
  
- Randomisierte Skip-Liste, 40
- Raum/Zeit-Tradeoff, 33
- Reductio ad absurdum, 6
- Reflexivität, 14
- Reihe, 28
- Reihendreiecksungleichung, 30
- Rekursion, 22
- Relation, 14
- Relationsidentität, 15
- Replacement Selectionsort, 48
- Run, 42
  
- Sandwichtheorem, 26
- Schnitt, 11
- Selectionsort, 43
- Sequenzielle Suche, 36
- Shellsort, 44
- Skip-Liste, 40
- Sortierproblem, 41
- Spezielle Suche, 41
- Sprungsuche, 37
- Stabile Suche, 41
- Stack, 40
- Starke Induktion, 7
- Strikter Binärbaum, 53
- Summ, 22
- Supremum, 21
- Surjektiv, 13
- Symmetrie, 14
  
- Tautologie, 5
- Teilbarkeit, 18
- Teilfolge, 26
- Teilmenge, 9
- Theorem, 3
- Transitivität, 14
- Transitivität der Ordnung, 18
- Transpose Regel, 39
- Trichotomie, 18
- Turniersortierung, 45
  
- Umgekehrte Dreiecksungleichung, 24
- Umkehrfunktion, 13
- unendlich, 9
- Ungerichteter Graph, 51
- Universum, 9
- Untere Schranke allgemeiner Sortierverfahren, 47
- Untere Schranke Komplexität, 34
- Untere Schranken, 21
- Unvollständigkeitssatz, 4
- Urbild, 12
  
- Vereinigung, 10
- Verkettete Liste, 39
- Verkettung, 13
- Vollständiger Baum, 52
- Vollständiger Binärbaum, 53
- Vollständigkeit, 14, 21
- Vollständigkeit der Reellen Zahlen, 27
- Vollständigkeitsaxiom, 22
  
- Wahrheitswertetabelle, 3
- Wertebereich, 12
- Widerlegbar, 5
- Widerspruch, 6
- Worst-case Komplexität, 34
- Wurzel, 19
- Wurzelbaum, 51
- Wurzelkriterium, 30
  
- Zerlegung, 15
- Zusammenhängender Graph, 51
  
- Äquivalenz, 3
- Äquivalenzklasse, 15
- Äquivalenzrelation, 15