

**Modelle der Informatik**  
Wintersemester 2019/2020

## **Kapitel 2: Logik**

Lehrstuhl Prof. Dr. Volker Gruhn  
Software Engineering, insbesondere mobile Anwendungen

Arbeitsgruppe Prof. Dr. Fabian Beck  
Juniorprofessur für Informatik – Visualisierung

# Inhaltsverzeichnis

## 2. Logik

- 2.1 Einleitung
- 2.2 Aussagenlogik
- 2.3 Resolution in der Aussagenlogik
- 2.4 Prädikatenlogik

## 2.1 Einleitung

Die formale Logik entstand als Modell für das menschliche (logische) Denken.

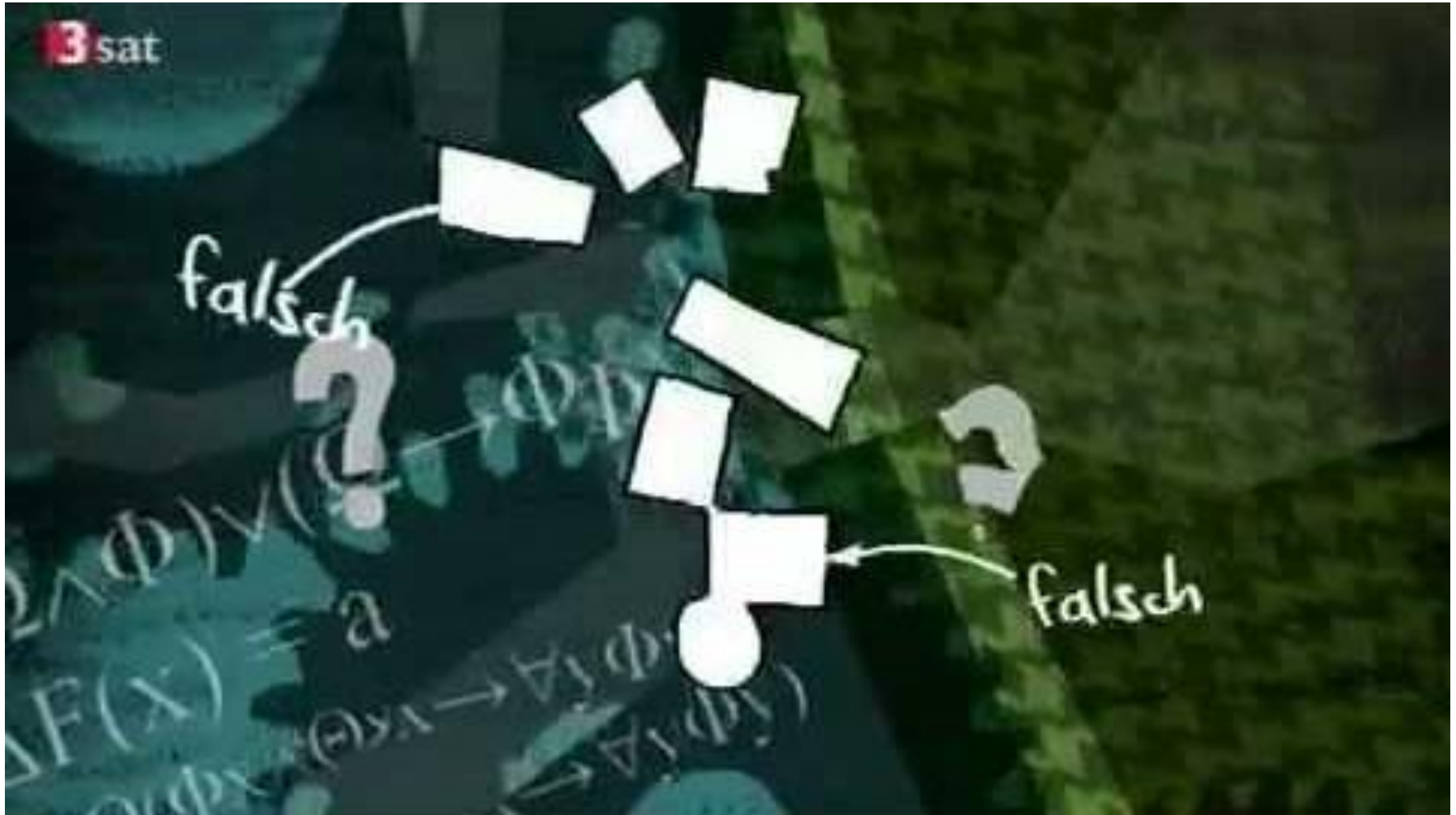
### Logik beschäftigt sich ...

- mit den Prinzipien zum Ziehen von Schlussfolgerungen (Konklusionen),
- mit der Gültigkeit von Begründungen und Widerspruchsfreiheit von Aussagen,
- mit der formalen Darstellung für die Form von Begründungen,
- mit dem Wahrheitsbegriff in abstrakter Form, aber **nicht mit dem Inhalt** von Aussagen.

Dieses Kapitel erweitert das Thema „**Elementare Aussagen und Prädikatenlogik**“ der Vorlesung „**Lineare Algebra für Informatiker und Wirtschaftsinformatiker**“.

# Einleitung

## Was ist Logik?



Was ist Logik? Philosophisches Kopfkino (3sat) – 2:42: <https://www.youtube.com/watch?v=aXKR0Ld9dTU>

# Einleitung

## Anwendungen der Logik

Die Anwendungen der Logik in der Informatik betreffen

- Formulierung von logischen Ausdrücken und Bedingungen in Programmen (z.B. `if`, `while`) und in Anfragesprachen für Datenbanken (z.B. SQL)
- Entwurf von Schaltungen
- Logik-basierte Programmiersprachen (z.B. Prolog)
- Maschinengestütztes Beweisen und Programmverifikation
- Expertensysteme und wissensbasierte Systeme
- Und vieles mehr...

## 2.2 Aussagenlogik

Die Aussagenlogik untersucht, wie sich der Wahrheitswert von verknüpften Aussagen aus den Wahrheitswerten der einfachen Aussagen ergibt.

Klassische Logik (zweiwertige Logik):

**Prinzip der Zweiwertigkeit:** Jede **Aussage** ist entweder **wahr** oder **falsch**.

- Schreibweisen:

**Wahr (Verum):** w, TRUE, 1,  $\top$

**Falsch (Falsum):** f, FALSE, 0,  $\perp$

- **Beispiel:** „7 ist eine Primzahl.“ ist eine wahre Aussage.

- Anmerkung: es gibt auch mehrwertige Logiken

Siehe z.B. [https://de.wikipedia.org/wiki/Mehrwertige\\_Logik](https://de.wikipedia.org/wiki/Mehrwertige_Logik)

**Definition:** Aussagen A und B heißen **logisch äquivalent** (Schreibweise  $A \equiv B$ ), genau dann wenn sie denselben Wahrheitswert besitzen.

# Aussagenlogik

## Beispiele

Wir betrachten folgende vier Sätze.

- (1) „Berlin ist die Hauptstadt von Deutschland.“
- (2) „Die Sonne ist eine Lichtquelle.“
- (3) „Die Zahl  $x$  ist eine Primzahl.“
- (4) „6 ist kleiner 13.“

- Die Sätze 1, 2 und 4 können unmittelbar auf ihre Wahrheitswerte untersucht werden. Eine Aussage ist wahr, wenn der Sachverhalt vorliegt.
- Die Sätze 1, 2 und 4 sind wahre Aussagen. Somit gilt:  $(1) \equiv (2) \equiv (4)$ . Obwohl inhaltlich völlig verschieden, sind sie aussagenlogisch gleich, weil alle den gleichen Wahrheitswert *wahr* besitzen.
- Der Wahrheitswert von Satz 3 kann erst bestimmt werden, wenn  $x$  einen Wert zugewiesen bekommt. Im Fall  $x = 6$ , wäre Satz 3 eine falsche Aussage.

# Aussagenlogik

## Aussagenvariablen und Aussagenschemata

Meist ist man nicht an elementaren Einzelaussagen, sondern an **Aussagenschemata** interessiert, die über **Aussagenvariablen** gebildet werden.

- Aussagenvariablen sind Variablen im Sinne von Booleschen Variablen in Programmiersprachen (Pascal, Java, C , C++).
- Folgend repräsentieren die Buchstaben p, q, r, ... oder auch A, B,... **Aussagenvariablen**.
- Erst durch eine Belegung der Variablen mit konkreten Werten kann ein Aussagenschema ausgewertet werden.



# Aussagenlogik

## Verknüpfungen von Aussagen und Aussagenschemata

Mit Hilfe von **logischen Operatoren** (Verknüpfungen, Junktoren) lassen sich ein, zwei oder mehrere logische Aussagen (bzw. Aussagenschemata) zu einer neuen logischen Aussage (bzw. Aussagenschema) verknüpfen.

- Diese nennt man dann **aussagenlogische Formel** oder **aussagenlogischer Ausdruck**.
- Grundlegende Operatoren:

Operator		Schreibweisen			Sprich
Negation	$\neg$	$\neg A$	NOT A	$\bar{A}$	„nicht A“
Disjunktion	$\vee$	$A \vee B$	A OR B	$A + B$	„A oder B“
Konjunktion	$\wedge$	$A \wedge B$	A AND B	$AB$	„A und B“

# Aussagenlogik

## Wahrheitstafeln

**Wahrheitstafeln** (Wahrheitstabellen) bieten eine vollständige Darstellung der Wahrheitswerte der einzelnen Komponenten eines logischen Ausdrucks.

- Die Auswertung erfolgt durch das Nebeneinanderschreiben der Ergebnisspalten von Teilausdrücken mit sukzessiver Auswertung.
- Eine Wahrheitstafel mit  $n$  Variablen besitzt  $2^n$  Zeilen mit verschiedenen Belegungen der Variablen.

Wahrheitstafel  $\neg$

A	$\neg A$
f	w
w	f

Wahrheitstafel  $\vee$

A	B	$A \vee B$
f	f	f
f	w	w
w	f	w
w	w	w

Wahrheitstafel  $\wedge$

A	B	$A \wedge B$
f	f	f
f	w	f
w	f	f
w	w	w

# Aussagenlogik

## Präzedenzen und Assoziativität der Operatoren

**Präzedenz** (Rangfolge, Bindungsstärke) von logischen Operatoren:

- Um Klammern zu sparen, wird folgende Bindungsstärke der logischen Operatoren vereinbart:

$\neg$  vor  $\wedge$  vor  $\vee$

d.h.  $\neg$  bindet am stärksten,  $\vee$  bindet am schwächsten

### Beispiele:

- $\neg \neg A \vee B$  ist gleichwertig zu  $(\neg (\neg A)) \vee B$
- $\neg A \wedge B \vee C$  ist gleichwertig zu  $((\neg A) \wedge B) \vee C$

**Assoziativität** der logischen Operatoren  $\wedge$  und  $\vee$ :


- $E \wedge (F \wedge G) \equiv (E \wedge F) \wedge G$
- $E \vee (F \vee G) \equiv (E \vee F) \vee G$

# Aussagenlogik

## Implikation

### Implikation $A \rightarrow B$

(alternative Schreibweise:  $A \text{ IMPL } B$ ,  $A \Rightarrow B$ )

- Die Implikation ist dann falsch, wenn aus einer wahren Voraussetzung die Folgerung nicht erfüllt ist.
- Insbesondere: Die Aussage „falsch impliziert wahr“ ist wahr. 
- $A \rightarrow B$  ist gleichbedeutend zu  $\neg A \vee B$

Wahrheitstafel  $\rightarrow$

A	B	$A \rightarrow B$
f	f	w
f	w	w
w	f	f
w	w	w

**Beispiel:** Betrachten wir folgende Aussagen

- A: „Der Himmel ist blau.“ (bzw. die Negation „Der Himmel ist nicht blau.“)
  - B: „Die Sonne scheint.“ (bzw. die Negation „Die Sonne scheint nicht.“)
- ... und überlegen, was dies für die Implikation  $A \rightarrow B$  bedeutet.

# Aussagenlogik

## Äquivalenz

### Äquivalenz $A \leftrightarrow B$

(alternative Schreibweisen:  $A \equiv B$ ,  $A \Leftrightarrow B$ )

- A und B heißen äquivalent, wenn die Wahrheitswerte von A und B übereinstimmen.
- $A \leftrightarrow B$  ist gleichbedeutend zu  $(\neg A \wedge \neg B) \vee (A \wedge B)$

Wahrheitstafel  $\leftrightarrow$

A	B	$A \leftrightarrow B$
f	f	w
f	w	f
w	f	f
w	w	w

# Aussagenlogik

## NAND und NOR

Es existieren zwei logische Operatoren, die **jeweils allein** ausreichen, um alle anderen logischen Operatoren darzustellen:

**NAND:  $A \mid B$**  (Sheffer-Strich  $\mid$ )

$A \mid B$  ist gleichbedeutend zu  $\neg (A \wedge B)$

**NOR:  $A \downarrow B$**  (Pierce-Pfeil  $\downarrow$ )

$A \downarrow B$  ist gleichbedeutend zu  $\neg (A \vee B)$

Wahrheitstafel  $\mid$  und  $\downarrow$

A	B	$A \mid B$	$A \downarrow B$
f	f	w	w
f	w	w	f
w	f	w	f
w	w	f	f

Darstellung von NOT, OR, AND durch Sheffer-Strich:

Negation:  $\neg A \equiv A \mid A$

Konjunktion:  $A \wedge B \equiv (A \mid B) \mid (A \mid B)$

Disjunktion:  $A \vee B \equiv (A \mid A) \mid (B \mid B)$

Darstellung von NOT, OR, AND durch Pierce-Pfeil:

Negation:  $\neg A \equiv A \downarrow A$

Konjunktion:  $A \wedge B \equiv (A \downarrow A) \downarrow (B \downarrow B)$

Disjunktion:  $A \vee B \equiv (A \downarrow B) \downarrow (A \downarrow B)$

# Aussagenlogik

Negation:  $\neg A \equiv A \mid A$

**NAND** Konjunktion:  $A \wedge B \equiv (A \mid B) \mid (A \mid B)$

Disjunktion:  $A \vee B \equiv (A \mid A) \mid (B \mid B)$

A	A   A	$\neg A$
f	w	w
w	f	f

A	B	A   B	(A   B)   (A   B)	$A \wedge B$
f	f	w	f	f
f	w	w	f	f
w	f	w	f	f
w	w	f	w	w

A	B	A   A	B   B	(A   A)   (B   B)	$A \vee B$
f	f	w	w	f	f
f	w	w	f	w	w
w	f	f	w	w	w
w	w	f	f	w	w

# Aussagenlogik

## NOR

Negation:  $\neg A \equiv A \downarrow A$

Konjunktion:  $A \wedge B \equiv (A \downarrow A) \downarrow (B \downarrow B)$

Disjunktion:  $A \vee B \equiv (A \downarrow B) \downarrow (A \downarrow B)$

A	$A \downarrow A$	$\neg A$
f	w	w
w	f	f

A	B	$A \downarrow A$	$B \downarrow B$	$(A \downarrow A) \downarrow (B \downarrow B)$	$A \wedge B$
f	f	w	w	f	f
f	w	w	f	f	f
w	f	f	w	f	f
w	w	f	f	w	w

A	B	$A \downarrow B$	$(A \downarrow B) \downarrow (A \downarrow B)$	$A \vee B$
f	f	w	f	f
f	w	f	w	w
w	f	f	w	w
w	w	f	w	w



# Aussagenlogik

## Präzedenz

**Präzedenz aller bisher genannten Operatoren:**

$\neg$  vor  $|$  vor  $\downarrow$  vor  $\wedge$  vor  $\vee$  vor  $\rightarrow$  vor  $\leftrightarrow$

d.h.  $\neg$  bindet am stärksten,  $\leftrightarrow$  bindet am schwächsten

**Beispiel:**

Ohne Klammerung:  $A \wedge B \leftrightarrow C \vee D \rightarrow \neg E \wedge F$

Mit äquivalenter Klammerung:  $(A \wedge B) \leftrightarrow ((C \vee D) \rightarrow ((\neg E) \wedge F))$

# Aussagenlogik

## Quiz 1 (per Moodle)

- Wahrheitstafel für  $(A \wedge B) \rightarrow (B \vee C)$ :

A	B	C	$A \wedge B$	$B \vee C$	$(A \wedge B) \rightarrow (B \vee C)$
f	f	f			
f	f	w			
f	w	f			
f	w	w			
w	f	f			
w	f	w			
w	w	f			
w	w	w			



Vorgehen:

- Werte die Teilausdrücke aus (Spalte 4 und 5).
- Werte den Gesamtausdruck aus (Spalte 6).

Zur Erinnerung:

A	B	$A \rightarrow B$
f	f	w
f	w	w
w	f	f
w	w	w

# Aussagenlogik

## Quiz 1 (per Moodle)



- Wahrheitstafel für  $(A \wedge B) \rightarrow (B \vee C)$ :

A	B	C	$A \wedge B$	$B \vee C$	$(A \wedge B) \rightarrow (B \vee C)$
f	f	f	f	f	w
f	f	w	f	w	w
f	w	f	f	w	w
f	w	w	f	w	w
w	f	f	f	f	w
w	f	w	f	w	w
w	w	f	w	w	w
w	w	w	w	w	w

Vorgehen:

- Werte die Teilausdrücke aus (Spalte 4 und 5).
- Werte den Gesamtausdruck aus (Spalte 6).

**Beobachtung:** Endausdruck (letzte Spalte) des aussagenlogischen Ausdrucks ist immer w (wahr). Wir nennen solch einen Ausdruck eine **Tautologie**.

# Aussagenlogik

## Tautologien und Kontradiktionen

- **Definition:** Als **Tautologien** (allgemeingültige Aussage) bezeichnet man logische Ausdrücke, die unabhängig von der Belegung der Variablen immer den Wert *wahr* ergeben.
- **Definition:** Als **Kontradiktion** (widersprüchliche Aussage) bezeichnet man logische Ausdrücke, die unabhängig von der Belegung der Variablen immer den Wert *falsch* ergeben.
- **Beispiele:**
  - **w** (wahr) ist eine Tautologie und **f** (falsch) ist eine Kontradiktion.
  - Eine Aussage der Form  $(A \vee \neg A)$  ist eine Tautologie und eine Aussagen der Form  $(A \wedge \neg A)$  ist eine Kontradiktion.

A	$\neg A$	$A \vee \neg A$	$(A \wedge \neg A)$
f	w	w	f
w	f	w	f

# Aussagenlogik

## Verknüpfen von Tautologien und Kontradiktionen

Es gilt:  $p \wedge 1 \equiv p$  und  $p \wedge 0 \equiv 0$  (Kontradiktion).

<b>p</b>	<b>1</b>	<b><math>p \wedge 1</math></b>
f	w	f
w	w	w

<b>p</b>	<b>0</b>	<b><math>p \wedge 0</math></b>
f	f	f
w	f	f

Es gilt:  $p \vee 1 \equiv 1$  (Tautologie) und  $p \vee 0 \equiv p$ .

<b>p</b>	<b>1</b>	<b><math>p \vee 1</math></b>
f	w	w
w	w	w

<b>p</b>	<b>0</b>	<b><math>p \vee 0</math></b>
f	f	f
w	f	w

Anmerkung zur Notation: wir verwenden hier **1** für **w** (bzw. **0** für **f**) in den Ausdrücken, damit **w** bzw. **f** nicht mit Variablen verwechselt werden.

# Aussagenlogik

## Einsetzungsregel

**Einsetzungsregel** (auch **Substitutionsregel** genannt):

Tautologien bleiben erhalten, wenn man eine oder mehrere darin enthaltene Aussagenvariablen konsistent ersetzt (d.h. gleiche Aussagenvariablen werden jeweils durch einen identischen Ausdruck ersetzt).

### Beispiel

- $p \wedge q \equiv q \wedge p$  ist eine Tautologie
- Wird in dieser Formel für  $p$  der Ausdruck  $(r \vee s)$  eingesetzt und für  $q$  der Ausdruck  $\neg r$  gewählt, dann ist auch  $(r \vee s) \wedge (\neg r) \equiv (\neg r) \wedge (r \vee s)$  eine Tautologie.

p	q	$p \wedge q$	$q \wedge p$	$p \wedge q \equiv q \wedge p$
f	f	f	f	w
f	w	f	f	w
w	f	f	f	w
w	w	w	w	w

# Aussagenlogik

## Vereinfachungen

Die verschiedenen Äquivalenzen zwischen logischen Ausdrücken geben uns die Möglichkeit, Ausdrücke zu vereinfachen.

### Beispiel:

$$\begin{aligned}(p \vee 0) \wedge (p \vee \neg p) &\equiv p \wedge (p \vee \neg p) && | \text{ weil } (p \vee 0) \equiv p \\ &\equiv p \wedge 1 && | \text{ weil } (p \vee \neg p) \text{ eine Tautologie ist} \\ &\equiv p && | \text{ weil } (p \wedge 1) \equiv p\end{aligned}$$

# Aussagenlogik

## Rechenregeln der Aussagenlogik 1/3

Des Weiteren gelten die folgenden Gesetze für beliebige Ausdrücke  $p$ ,  $q$  und  $r$  (jede einzelne Regel lässt sich leicht mit Hilfe einer Wahrheitstafel überprüfen):

- **Kommutativgesetz**

$$p \wedge q \equiv q \wedge p$$

$$p \vee q \equiv q \vee p$$

- **Assoziativgesetz**

$$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$$

$$(p \vee q) \vee r \equiv p \vee (q \vee r)$$

- **Distributivgesetz**

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$$

$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

- **Reflexivität der Äquivalenz**

$$p \equiv p$$

- **Kommutativität der Äquivalenz**

$$(p \equiv q) \equiv (q \equiv p)$$

- **Transitivität der Äquivalenz**

$$(p \equiv q) \wedge (q \equiv r) \rightarrow (p \equiv r)$$

- **Äquivalenz von Negationen**

$$(p \equiv q) \equiv (\neg p \equiv \neg q)$$



# Aussagenlogik

## Rechenregeln der Aussagenlogik 2/3

Die **de Morgan'schen Regeln** formulieren einen Zusammenhang zwischen NOT und AND bzw. OR:

- $\neg(p \wedge q) \equiv \neg p \vee \neg q$
- $\neg(p \vee q) \equiv \neg p \wedge \neg q$

Diese Regeln können für beliebige Konjunktionen bzw. Disjunktionen verallgemeinert werden (wegen Assoziativgesetze):

- $\neg(p_1 \wedge p_2 \wedge \dots \wedge p_n) \equiv \neg p_1 \vee \neg p_2 \vee \dots \vee \neg p_n$
- $\neg(p_1 \vee p_2 \vee \dots \vee p_n) \equiv \neg p_1 \wedge \neg p_2 \wedge \dots \wedge \neg p_n$

# Aussagenlogik

## Rechenregeln der Aussagenlogik 3/3

- Identität von AND  $p \wedge 1 \equiv p$
- Neutralität von AND  $p \wedge 0 \equiv 0$
- Identität von OR  $1 \vee p \equiv 1$
- Neutralität von OR  $0 \vee p \equiv p$
- doppelte Negation  $\neg\neg p \equiv p$
- Idempotenz von AND  $p \wedge p \equiv p$
- Idempotenz von OR  $p \vee p \equiv p$
- Kontradiktion  $p \wedge \neg p \equiv 0$
- Tautologie  $p \vee \neg p \equiv 1$

# Aussagenlogik

## Logische (Boolesche) Funktionen

Logische Ausdrücke mit Variablen können auch als logische (oder Boolesche) Funktionen aufgefasst werden ( $f: \{0,1\}^n \rightarrow \{0,1\}$ ).

### Beispiele:

$$f_1(p) \quad := \quad \neg p$$

$$f_2(p, q) \quad := \quad p \wedge (p \vee q)$$

$$f_3(p, q, r) := \quad p \wedge (p \vee q) \wedge \neg r$$

# Aussagenlogik

## Anwendung in Programmen: Ein Beispiel in Java

```
1 public class EligibilityToVote {  
2  
3     public static void main(String args[]) {  
4         System.out.println("Who is eligible to vote? Some examples ... ");  
5         System.out.println("Underage child with citizenship, living in the country: "  
6             + isEligibleToVote(true, true, true));  
7         System.out.println("Adult with citizenship, living in the country: "  
8             + isEligibleToVote(false, true, true));  
9         System.out.println("Adult foreigner, living in the country: "  
10            + isEligibleToVote(false, false, true));  
11     }  
12  
13     public static boolean isEligibleToVote(  
14         boolean underage,  
15         boolean citizen,  
16         boolean resident) {  
17         return !underage && (citizen || resident);  
18     }  
19 }
```

```
Who is eligible to vote? Some examples ...  
Underage child with citizenship, living in the country: false  
Adult with citizenship, living in the country: true  
Adult foreigner, living in the country: true
```

# Aussagenlogik

## Boolesche Ausdrücke in Java

```
return !underage && (citizen || resident);
```

NOT

AND

OR

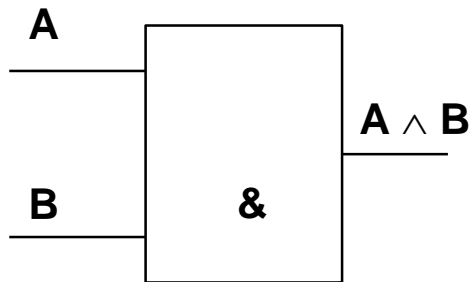
- AND: Für `&&` wird zunächst die linke Seite ausgewertet.
  - Die Berechnung stoppt, wenn die linke Seite `false` ergibt (der Wert der rechten Seite ist dann für das Ergebnis irrelevant). Das Ergebnis ist `false`.
  - Alternativ: `&` wertet immer beide Seiten aus.
- OR: `||` (stoppt bei linker Seite `true`) und `|` funktionieren analog.
- In der Praxis werden meist `&&` und `||` verwendet.
  - Schnellere Ausführung durch teilweises Überspringen von Berechnungen.
  - Die Reihenfolge kann optimiert werden: Ausdrücke, die häufig `false` (AND) bzw. `true` (OR) ergeben, sollten links stehen.
  - Die ist besonders relevant, wenn die Auswertung der Seiten umfangreichere Berechnungen erfordern (z.B. den Aufruf weiterer Methoden).

# Aussagenlogik

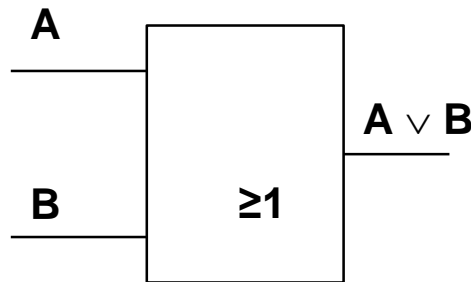
## Anwendung Entwurf digitaler Schaltungen

Logische Funktionen können auch mittels elektronischer Schaltungen (in Hardware) umgesetzt werden.

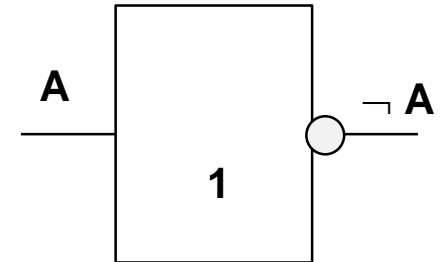
- Die logischen Grundfunktionen  $\wedge$  (AND),  $\vee$  (OR) und  $\neg$  (NOT) können durch sog. **Logikgatter** realisiert werden, die miteinander kombiniert werden können.



AND-Gatter



OR-Gatter



NOT-Gatter

- Die Gatter werden als Bauteile auf Basis einfacher Transistoren realisiert.
- Elektrische Leitungen verbinden die Ein- bzw. Ausgänge der Gatter.
- Spannungswerte werden binär als Belegung der logischen Variablen interpretiert (Low = 0; High = 1).

# Aussagenlogik

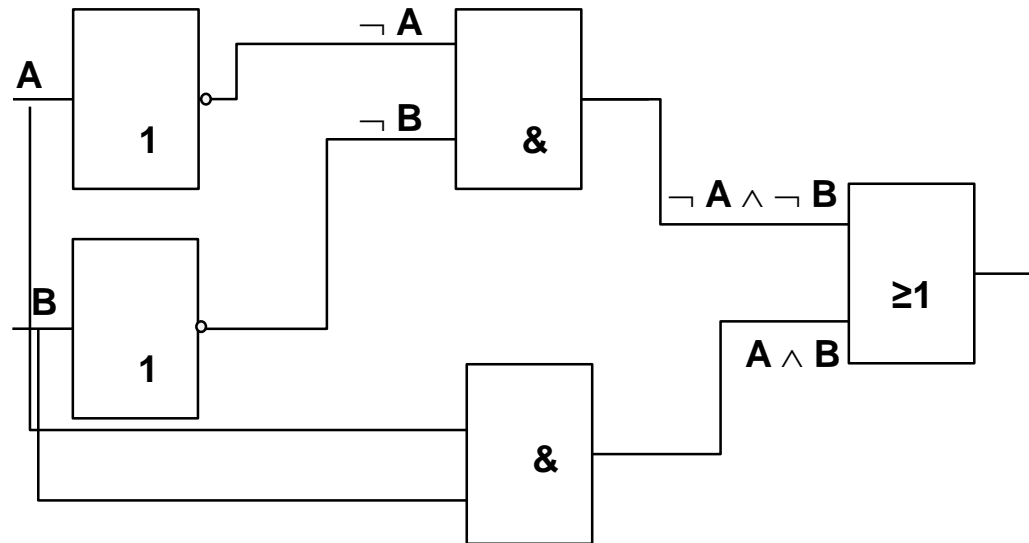
## Beispielschaltung

**Beispiel:** Implementiert werden soll die Äquivalenz  $A \leftrightarrow B$  als Schaltung.

- Umformung:

$$A \leftrightarrow B \equiv (\neg A \wedge \neg B) \vee (A \wedge B)$$

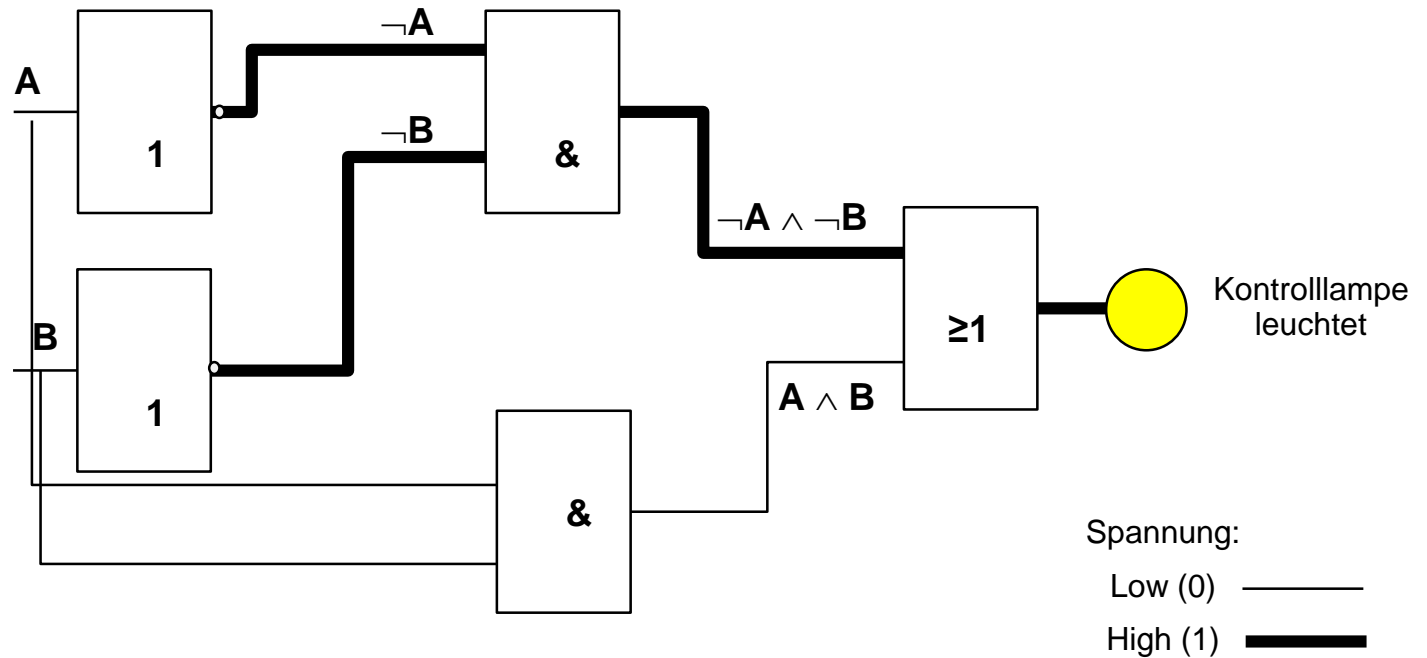
- Damit kann die Schaltung aus NOT-, AND- und OR-Gatter umgesetzt werden:



# Aussagenlogik

## Beispielschaltung: Test 1

A	B	$\neg A$	$\neg B$	$\neg A \wedge \neg B$	$A \wedge B$	$(\neg A \wedge \neg B) \vee (A \wedge B)$
0	0	1	1	1	0	1

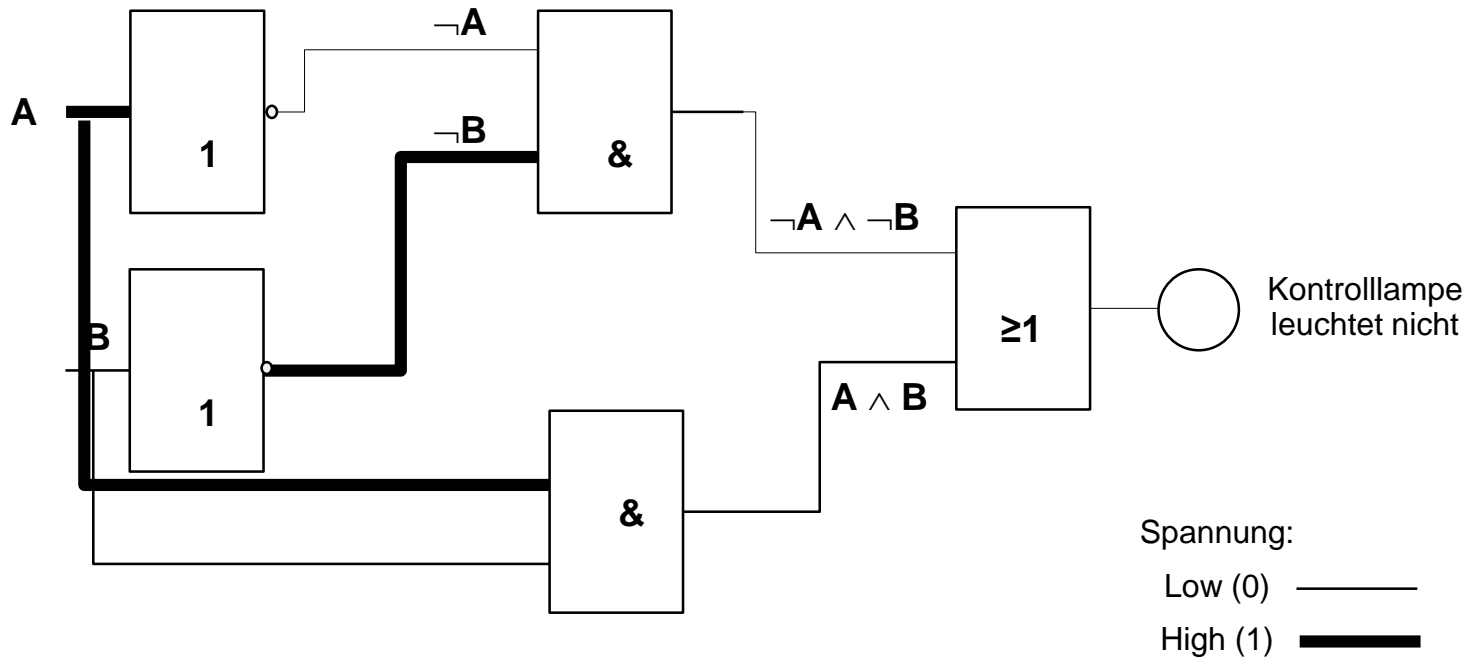




# Aussagenlogik

## Beispielschaltung: Test 2

A	B	$\neg A$	$\neg B$	$\neg A \wedge \neg B$	$A \wedge B$	$(\neg A \wedge \neg B) \vee (A \wedge B)$
1	0	0	1	0	0	0





Lionel Page  
@page\_eco



Such a cool way to explain logic gates.

[Tweet übersetzen](#)



[https://twitter.com/page\\_eco/status/1188749430020698112](https://twitter.com/page_eco/status/1188749430020698112)

# Aussagenlogik

## Konsistenz von Aussagen

**Definition:** Eine Menge von Aussagen heißt **konsistent** (widerspruchsfrei), wenn es mindestens eine Belegung der Variablen gibt, so dass alle Aussagen wahr sind (bzw. kein Widerspruch abgeleitet werden kann – siehe später).

**Beispiel:** Es seien

- S: Die Durchfallquote bei Prüfungen ist niedrig.
- R: Die Studierenden lernen viel.
- H: Die Professor\*innen sind unglücklich.
- Wir definieren die folgenden Menge von vier Aussagen:  

(1)  $R \rightarrow S$ (2)  $S \rightarrow \neg H$ (3)  $R$ (4)  $H$
- **Frage:** Sind diese Aussagen konsistent (widerspruchsfrei)?
- **Vorgehensweise:** Um zu erkennen, ob die Aussagen konsistent sind, muss ermittelt werden, ob die Konjunktion  $(R \rightarrow S) \wedge (S \rightarrow \neg H) \wedge R \wedge H$  den Wert wahr annehmen kann.

# Aussagenlogik

## Lösung

$$\underbrace{(R \rightarrow S)}_a \wedge \underbrace{(S \rightarrow \neg H)}_b \wedge \underbrace{R}_c \wedge \underbrace{H}_d$$

Eine Überprüfung erfolgt mit Hilfe einer Wahrheitstafel.

R	S	H	a	b	c	d	$a \wedge b \wedge c \wedge d$
f	f	f	w	w	f	f	f
f	f	w	w	w	f	w	f
f	w	f	w	w	f	f	f
f	w	w	w	f	f	w	f
w	f	f	f	w	w	f	f
w	f	w	f	w	w	w	f
w	w	f	w	w	w	f	f
w	w	w	w	f	w	w	f

→ Die Aussagen sind nicht widerspruchsfrei!

## 2.3 Resolution in der Aussagenlogik

Unter **Resolution** (auch **Resolvierung** oder **Resolutionskalkül** genannt) versteht man eine systematische und automatisierbare Methode zur Behandlung logischer Probleme.

Wir wollen nachweisen, dass aus gegebenen Aussagen Q und R (**Annahmen** oder **Prämissen** genannt) eine **Schlussfolgerung** S (Konklusion, Konsequenz) gezogen werden kann.

- Wir schreiben hierfür  $\{Q, R\} \vdash S$ .
- Wir sagen S ist **ableitbar** (oder resolvierbar) aus Q und R

Die Grundlage des Resolutionskalküls bilden **Normalformen**, wobei man die folgenden beiden Normalformen unterscheidet:

- **Konjunktive Normalform (CNF)**
- **Disjunktive Normalform (DNF)**

# Resolution in der Aussagenlogik

## Literale

Als **atomare Ausdrücke** oder **Atome** bezeichnen wir nicht weiter zerlegbare Grundaussagen.

Ein **Literal** ist ein Atom oder die Negation eines Atoms.

Sei  $p$  ein atomarer Ausdruck.

- Dann wird  $p$  das **positive** und  $\neg p$  das **negative Literal** genannt.
- Beide Literale  $p$  und  $\neg p$  heißen **komplementär**.

# Resolution in der Aussagenlogik

## Normalformen

**Definition:** Ein aussagenlogischer Ausdruck  $A$  ist in **konjunktiver Normalform (CNF)**, falls er eine Konjunktion von Disjunktionen von Literalen ist, d.h.:

$$A = A_1 \wedge A_2 \wedge \dots \wedge A_n ,$$

wobei jedes  $A_i$  die Form  $\lambda_1 \vee \lambda_2 \dots \vee \lambda_j \dots \vee \lambda_{m_i}$  hat  
und jedes  $\lambda_j$  ein Literal ist.

**Definition:** Ein aussagenlogischer Ausdruck  $A$  ist in **disjunktiver Normalform (DNF)**, falls er eine Disjunktion von Konjunktionen von Literalen ist, d.h.:

$$A = A_1 \vee A_2 \vee \dots \vee A_n ,$$

wobei jedes  $A_i$  die Form  $\lambda_1 \wedge \lambda_2 \dots \wedge \lambda_j \dots \wedge \lambda_{m_i}$  hat  
und jedes  $\lambda_j$  ein Literal ist.

### Beispiele für CNF:

$$(p_1 \vee p_2) \wedge (p_3 \vee p_4)$$

$$(p_1 \vee p_2) \wedge (p_2 \vee p_3 \vee p_4) \wedge (p_3 \vee \neg p_4)$$

### Beispiele für DNF:

$$(p_1 \wedge p_2) \vee (p_3 \wedge p_4)$$

$$(p_1 \wedge p_2) \vee (p_2 \wedge p_3 \wedge p_4) \vee (p_3 \wedge \neg p_4)$$

# Resolution in der Aussagenlogik

## Konstruktion von DNF und CNF per Wahrheitstafel

Jede aussagenlogische Formel kann in DNF oder CNF überführt werden.

Konstruktion der **CNF (bzw. DNF)** per Wahrheitstafel:

- Pro Zeile mit dem **Ergebniswert f (bzw. w)** wird ein **ge-oderter/disjunktiver (bzw. ge-undeter/konjunktiver) Teilausdruck** erzeugt, der ausschließlich für diese Variablenbelegung den **Ergebniswert f (bzw. w)** liefert.
- Dann erfolgt eine **konjunktive (bzw. disjunktive) Verknüpfung** der Teilausdrücke.

### Beispiel: XOR

A	B	A XOR B	CNF
f	f	f	$A \vee B$
f	w	w	-
w	f	w	-
w	w	f	$\neg A \vee \neg B$

Ergebnis:  $(A \vee B) \wedge (\neg A \vee \neg B)$

A	B	A XOR B	DNF
f	f	f	-
f	w	w	$\neg A \wedge B$
w	f	w	$A \wedge \neg B$
w	w	f	-

$(\neg A \wedge B) \vee (A \wedge \neg B)$



# Resolution in der Aussagenlogik

## CNF/DNF-Umformung durch Nutzung der Äquivalenzbeziehungen

Alternative: Zur Umwandlung von aussagenlogischen Ausdrücken in CNF oder DNF können Äquivalenzbeziehungen (siehe Rechenregeln) verwendet werden.

**Geeignete Reihenfolge** der Anwendung der Äquivalenzbeziehungen (Ziel: CNF):

1. Eliminierung von  $\leftrightarrow$ :

$$A \leftrightarrow B \equiv (A \rightarrow B) \wedge (B \rightarrow A)$$

2. Eliminierung von  $\rightarrow$ :

$$A \rightarrow B \equiv \neg A \vee B$$

3. Negationen vor die Atome ziehen:

$$\text{de Morgan, z.B. } \neg(A \wedge B) \equiv \neg A \vee \neg B$$

4. Eliminierung von doppelten Negationen:

$$\neg\neg A \equiv A$$

5. Anwendung der Regeln für Distributivität, Kommutativität, Identität, Absorptionen, Tautologie und Kontradiktion.

# Resolution in der Aussagenlogik

## Beispiele

Die folgenden Ausdrücke werden in CNF umgewandelt:

$$\begin{aligned}\neg (p_1 \rightarrow p_3) &\equiv \neg(\neg p_1 \vee p_3) \\ &\equiv p_1 \wedge \neg p_3\end{aligned}$$

$$\begin{aligned}p_1 \leftrightarrow (p_2 \wedge p_3) &\equiv (p_1 \rightarrow (p_2 \wedge p_3)) \wedge ((p_2 \wedge p_3) \rightarrow p_1) \\ &\equiv (\neg p_1 \vee (p_2 \wedge p_3)) \wedge (\neg(p_2 \wedge p_3) \vee p_1) \\ &\equiv (\neg p_1 \vee p_2) \wedge (\neg p_1 \vee p_3) \wedge (\neg p_2 \vee \neg p_3 \vee p_1)\end{aligned}$$

# Resolution in der Aussagenlogik

## Wechsel zwischen den Normalformen (CNF $\leftrightarrow$ DNF)

Durch Anwendung des Distributivgesetzes und der Absorptionsregeln können DNF-Ausdrücke in CNF-Ausdrücke umgewandelt werden und umgekehrt.

### Beispiele:

#### ▪ DNF $\rightarrow$ CNF

$$\begin{aligned}(\neg A \wedge B) \vee (A \wedge \neg B) &\equiv (\neg A \vee A) \wedge (\neg A \vee \neg B) \wedge (B \vee A) \wedge (B \vee \neg B) \\&\equiv (1) \wedge (\neg A \vee \neg B) \wedge (B \vee A) \wedge (1) \\&\equiv (\neg A \vee \neg B) \wedge (B \vee A)\end{aligned}$$

#### ▪ CNF $\rightarrow$ DNF

$$\begin{aligned}(A \vee B) \wedge (\neg A \vee \neg B) &\equiv (A \wedge \neg A) \vee (\neg A \wedge B) \vee (\neg B \wedge A) \vee (\neg B \wedge B) \\&\equiv (0) \vee (\neg A \wedge B) \vee (\neg B \wedge A) \vee (0) \\&\equiv (\neg A \wedge B) \vee (\neg B \wedge A)\end{aligned}$$

# Resolution in der Aussagenlogik

## Klauseln und Klauselmengen

**Definition:** Eine **Klausel** ist eine endliche Disjunktion von Literalen, d.h. entspricht der Form

$$p_1 \vee p_2 \vee \dots \vee p_k,$$

wobei jedes  $p_i$  ein (positives oder negatives) Literale ist.

**Definition:** Eine **Klauselmenge** ist eine Menge  $\{K_1, \dots, K_n\}$  von Klauseln.

- **Bemerkung:** Damit können Ausdrücke in CNF auch als Klauselmenge aufgefasst werden.
- **Schreibweise:** Klauselmengen werden in der Form  $\{\{L_1, L_2, L_3, \dots\}, \{M_1, M_2, \dots\}, \{N_1, \dots\}, \dots\}$  angegeben.
- **Beispiel:**

CNF-Ausdruck	$(\neg A \vee B) \wedge (\neg B \vee C) \wedge A \wedge \neg C$
entspricht der Klauselmenge	$\{\{\neg A, B\}, \{\neg B, C\}, \{A\}, \{\neg C\}\}$

# Resolution in der Aussagenlogik

## Resolventen

**Definition:** Seien  $C_1, C_2$  Klauseln, die die komplementären Literale  $\lambda$  und  $\neg\lambda$  enthalten, d.h.  $\lambda \in C_1$  und  $\neg\lambda \in C_2$ . Dann bezeichnet

$$R = C_1 \setminus \{\lambda\} \cup C_2 \setminus \{\neg\lambda\}$$

eine **Resolvente** der Klauseln  $C_1$  und  $C_2$ .

## Beispiele für Resolution:

- $\{B\}$  ist eine Resolvente von  $\{\{\neg A, B\}, \{A\}\}$
- $\{A, B, D\}$  ist eine Resolvente von  $\{\{A, B, C\}, \{\neg C, D\}\}$
- Die leere Klausel  $\emptyset$  ist eine Resolvente von  $\{\{A\}, \{\neg A\}\}$  (d.h. es ergibt sich ein Widerspruch; wir verwenden das Symbol  $\perp$  zur Darstellung)
- $\{A, C, \neg C, D\}$  und  $\{A, \neg B, B, D\}$  sind Resolventen von  $\{\{A, \neg B, C, D\}, \{A, B, \neg C, D\}\}$



**Bemerkung:** Am letzten Beispiel sieht man, dass die Resolventenbildung nicht eindeutig sein muss. Es wird immer nur ein Paar komplementärer Literale entfernt.

# Resolution in der Aussagenlogik

## Resolutionsprinzip und Resolutionslemma

**Resolutionsprinzip:** Eine Resolvente  $R$  zweier Klauseln  $C_1, C_2$  ist eine logische Folgerung (Konklusion) aus  $C_1 \wedge C_2$ , d.h.:  $C_1 \wedge C_2 \vdash R$ .

**Beispiel:** Aus  $(A \rightarrow B)$  und  $(B \rightarrow C)$  folgt  $(A \rightarrow C)$ , weil  $(\neg A \vee C)$  Resolvente von  $(\neg A \vee B)$  und  $(\neg B \vee C)$  ist (Zur Erinnerung:  $A \rightarrow B$  ist gleichbedeutend zu  $\neg A \vee B$ ), d.h.  $(A \rightarrow B) \wedge (B \rightarrow C) \vdash (A \rightarrow C)$ .

### Resolutionslemma:

- Sei  $F$  eine Formel in CNF.
- Ferner sei  $R$  eine Resolvente zweier Klauseln  $C_1$  und  $C_2$  aus der Klauselmenge von  $F$ .

Dann gilt:  $F \equiv (F \wedge R)$



**Bemerkung:** Aber es gilt in der Regel nicht:  $F \equiv R$ , d.h. die Ausgangsklauseln und ihre Resolvente sind nicht äquivalent.

# Resolution in der Aussagenlogik

## Beispiel für das Resolutionslemma

Atome			C1	C2	F	R	$(F \wedge R)$	$F \equiv (F \wedge R)$	$F \equiv R$
A	B	C	$(\neg A \vee B)$	$(\neg B \vee C)$	$(\neg A \vee B) \wedge (\neg B \vee C)$	$(\neg A \vee C)$			
f	f	f	w	w	w	w	w	w	w
f	f	w	w	w	w	w	w	w	w
f	w	f	w	f	f	w	f	w	f
f	w	w	w	w	w	w	w	w	w
w	f	f	f	w	f	f	f	w	w
w	f	w	f	w	f	w	f	w	f
w	w	f	w	f	f	f	f	w	w
w	w	w	w	w	w	w	w	w	w

- $C_1, C_2 = \text{Klauseln}, F = \text{CNF-Formel}, R = \text{Resolvente}$

# Resolution in der Aussagenlogik

## Beispiel

Wir beweisen  $(p_i \rightarrow p_j) \wedge (\neg(p_j \rightarrow p_k) \rightarrow \neg p_i) \vdash (p_i \rightarrow p_k)$ .

Durch Überführung in CNF ergibt sich:

$$\begin{aligned} (p_i \rightarrow p_j) \wedge (\neg(p_j \rightarrow p_k) \rightarrow \neg p_i) &\vdash (p_i \rightarrow p_k) && | \text{ Implikation } \rightarrow \text{ ersetzen} \\ \equiv (\neg p_i \vee p_j) \wedge ((p_j \rightarrow p_k) \vee \neg p_i) &\vdash (\neg p_i \vee p_k) && | \text{ Implikation } \rightarrow \text{ ersetzen} \\ \equiv (\neg p_i \vee p_j) \wedge (\neg p_j \vee p_k \vee \neg p_i) &\vdash (\neg p_i \vee p_k) && | \text{ als Mengen notieren} \\ \equiv \{\{\neg p_i, p_j\}, \{\neg p_j, p_k, \neg p_i\}\} &\vdash \{\{\neg p_i, p_k\}\} \end{aligned}$$

Gemäß dem Resolutionsprinzip und da  $\{\neg p_i, p_k\}$  eine Resolvente von  $\{\{\neg p_i, p_j\}, \{\neg p_j, p_k, \neg p_i\}\}$  (mit  $\lambda = p_j$ ) ist, haben wir die Aussage bewiesen.



# Resolution in der Aussagenlogik

## Resolutionsableitung

**Definition:** Eine Resolutionsableitung einer Klausel  $C$  aus einer Klauselmenge  $S$  ist eine endliche Folge von Klauseln  $C_1, C_2, \dots, C_n$  (mit  $C = C_n$  als Ergebnis der Ableitung), so dass für jedes  $C_i$  aus dieser Folge gilt, entweder

1.  $C_i$  ist ein Element aus  $S$  (d.h.  $C_i \in S$ ) oder
2.  $C_i$  ist eine Resolvente zweier vorhergehender Klauseln der Folge  $C_j$  und  $C_k$  mit  $j, k < i$ .

## Quiz 2 (per Moodle)

Sei  $S = \{\{A, \neg B\}, \{A, B\}, \{\neg C\}\}$ . Welche der folgenden Folgen von Klauseln ist keine gültige Resolutionsableitung aus  $S$ ? Begründen Sie die Antwort kurz.



- a)  $\{A, \neg B\}, \{A, B\}$
- b)  $\{A, \neg B\}, \{A, B\}, \{\neg C\}, \{A\}$
- c)  $\{A, \neg B\}, \{A\}, \{A, B\}, \{\neg C\}$
- d)  $\{A, \neg B\}, \{A, B\}, \{A\}, \{\neg C\}, \{A\}$

# Resolution in der Aussagenlogik

## Resolutionsableitung

**Definition:** Eine Resolutionsableitung einer Klausel  $C$  aus einer Klauselmenge  $S$  ist eine endliche Folge von Klauseln  $C_1, C_2, \dots, C_n$  (mit  $C = C_n$  als Ergebnis der Ableitung), so dass für jedes  $C_i$  aus dieser Folge gilt, entweder

1.  $C_i$  ist ein Element aus  $S$  (d.h.  $C_i \in S$ ) oder
2.  $C_i$  ist eine Resolvente zweier vorhergehender Klauseln der Folge  $C_j$  und  $C_k$  mit  $j, k < i$ .

## Quiz 2 (per Moodle)

Sei  $S = \{\{A, \neg B\}, \{A, B\}, \{\neg C\}\}$ . Welche der folgenden Folgen von Klauseln ist keine gültige Resolutionsableitung aus  $S$ ? Begründen Sie die Antwort kurz.



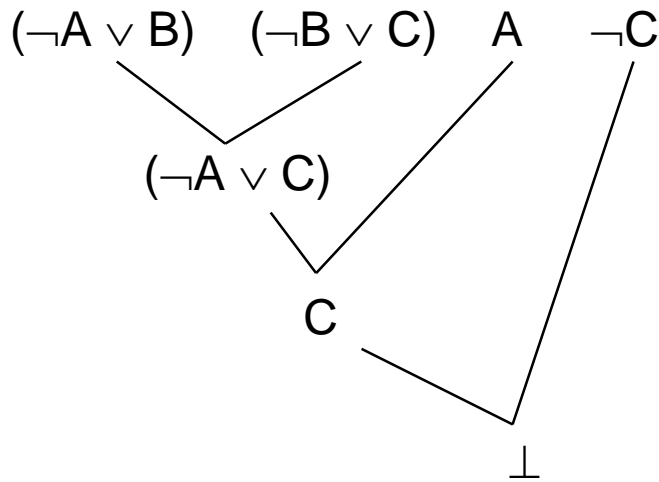
- |  |   |
|--|---|
| a) $\{A, \neg B\}, \{A, B\}$                           | <b>Gültig:</b> keine Ableitung erfolgt (alles Fall 1), $\{\neg C\}$ nicht verwendet     |
| b) $\{A, \neg B\}, \{A, B\}, \{\neg C\}, \{A\}$        | <b>Gültig:</b> $\{A\}$ ist Resolvente von vorherigen Klauseln $\{A, \neg B\}, \{A, B\}$ |
| c) $\{A, \neg B\}, \{A\}, \{A, B\}, \{\neg C\}$        | <b>Nicht gültig:</b> $\{A\}$ nicht Resolvente aus vorherigen Klauseln                   |
| d) $\{A, \neg B\}, \{A, B\}, \{A\}, \{\neg C\}, \{A\}$ | <b>Gültig:</b> vergleiche (b), nur unnötige Dopplung                                    |

# Resolution in der Aussagenlogik

## Grafische Darstellung einer Resolutionsableitung

Eine Resolutionsableitung kann wie folgt grafisch dargestellt werden (z.B. von links nach rechts und oben nach unten als Folge gelesen).

**Beispiel:** Gegeben sei der CNF Ausdruck  $(\neg A \vee B) \wedge (\neg B \vee C) \wedge A \wedge \neg C$ .

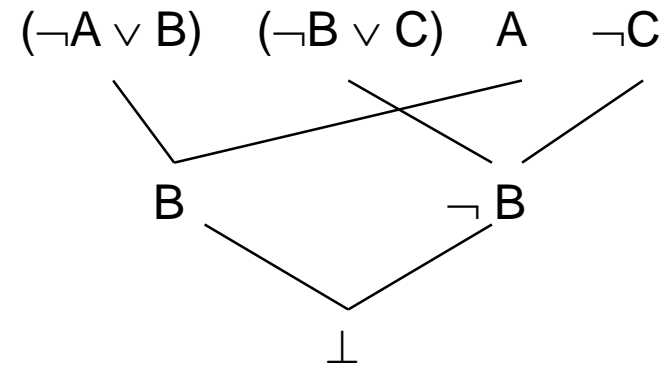
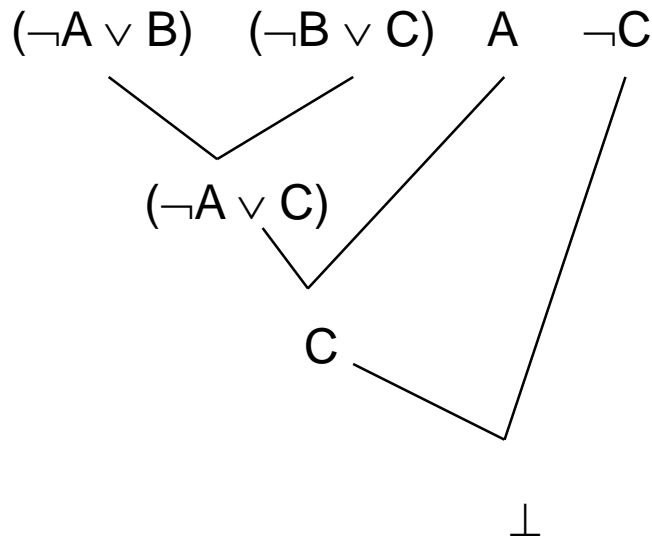


1. Die beiden Klauseln  $(\neg A \vee B)$  und  $(\neg B \vee C)$  haben die Resolvente  $(\neg A \vee C)$ .
2. Die Klauseln  $(\neg A \vee C)$  und  $A$  haben die Resolvente  $C$ .
3. Die Klauseln  $C$  und  $\neg C$  haben die leere Klausel als Resolvente (Widerspruch  $\perp$ ).

# Resolution in der Aussagenlogik

## Nichtdeterministische Resolutionsableitung

Grundsätzlich kann die Resolution in unterschiedlicher Reihenfolge durchgeführt werden. Somit entstehen verschiedene alternative Resolutionsableitungen.



**Beobachtung:** Die Resolventen können sich unterscheiden. Wenn also im Verlauf der Resolution nicht die geforderte Konklusion resolviert wird, müssen auch alternativen Resolutionsableitungen geprüft werden.

# Resolution in der Aussagenlogik

## Widerspruchsfreiheit von Klauselmengen

Durch Resolution können folgende Aussagen können zur Widerspruchsfreiheit von Klauselmengen gemacht werden:

- Falls eine leere Klausel resolviert werden kann, ist die gesamte Klauselmenge **nicht widerspruchsfrei**. Der Resolutionsvorgang kann somit abgebrochen werden.
- Falls in jeder Klausel das gleiche Literal vorhanden ist (entweder immer negiert oder immer nicht negiert), ist diese Klauselmenge **widerspruchsfrei**.

### Beispiel:

- Die Klauselmenge  $\{\{\neg A, B\}, \{\neg B, C\}, \{A\}, \{\neg C\}\}$  ist nicht widerspruchsfrei (siehe Resolution im vorherigen Beispiel).
- Die Klauselmenge  $\{\{A, B, C\}, \{A\}, \{A, \neg C\}, \{A, \neg B\}\}$  ist widerspruchsfrei, da in jeder Klausel das Literal A vorhanden ist.

# Resolution in der Aussagenlogik

## Lineare Resolution

Eine Einschränkung der Resolutionsableitung, um systematisch zu überprüfen, ob ein Widerspruch abgeleitet werden kann, ist die **lineare Resolution**:

Sei  $S$  eine Klauselmenge und  $C_1, C_2, \dots, C_n$  eine Folge von Klauseln aus einer Resolutionsableitung aus  $S$  derart, dass für jede Klausel  $C_i$  gilt:

1.  $C_i$  ist ein Element aus  $S$  (d.h.  $C_i \in S$ ) oder
2.  $C_i$  ist eine Resolvente zweier vorhergehender Klauseln der Folge  $C_j$  und  $C_{i-1}$  mit  $j < i$ .

**Beobachtung:** Die Einschränkung gegenüber einer allgemeinen Resolutionsableitung bezieht sich auf Fall (2) (ursprünglich: „ $C_i$  ist eine Resolvente zweier vorhergehender Klauseln der Folge  $C_j$  und  $C_k$  mit  $j, k < i$ .“): eine der Klauseln muss nun genau vor der Resolventen stehen ( $C_{i-1}$ ).

Die lineare Resolution ist **widerlegungsvollständig** (d.h. sind die Klauseln **nicht widerspruchsfrei**, dann existiert eine lineare Resolutionsableitung, die zum Widerspruch führt). Ohne Beweis.

# Resolution in der Aussagenlogik

## Widerlegung durch Resolution

**Resolutionswiderlegung (Resolutionsrefutation):** Ableitung eines Widerspruchs per Resolution um eine Schlussfolgerung zu beweisen.

### Ablauf:

1. Negation der Konklusion einer Schlussfolgerung durch konjunktive Verknüpfung mit den Prämissen (z.B.  $\{Q, R\} \vdash S$  wird zu  $Q \wedge R \wedge \neg S$ ).
2. Überführung des Gesamtausdrucks in die CNF und Erzeugung der Klauselmenge.
3. Durchführung der Resolution: Erzeugung alternativer Resolutionsableitung bis ein Widerspruch abgeleitet oder die Resolution vollständig erzeugt wurde (z.B. nach linearer Resolution).

Konnte ein Widerspruch erzeugt werden, so ist die ursprüngliche Aussage wahr (denn die Negation der Konklusion ist widerspruchsbehaftet).  
Sonst ist die ursprüngliche Aussage falsch.



# Resolution in der Aussagenlogik

## Beispiel

Es ist zu prüfen, ob aus den beiden Prämissen

$$Q := P \rightarrow (\neg H \rightarrow C) \text{ und } R := P \rightarrow \neg H$$

die Schlussfolgerung

$$S := P \rightarrow C$$

gezogen werden kann, d.h. ob gilt  $\{Q, R\} \vdash S$ .

Als Interpretation für die atomaren Aussagen  $P$ ,  $C$  und  $H$  könnten z. B. die Aussagen herangezogen werden:

$P$ : „Der Geiger gibt ein Konzert.“

$C$ : „Viele werden kommen.“

$H$ : „Die Preise sind zu hoch.“



# Resolution in der Aussagenlogik

## Beispiel

### Lösung des Beispiels:

1. Negation der Konklusion:  $\neg S \equiv \neg (P \rightarrow C)$

2. Transformation in Klauselform:

- Q:  $P \rightarrow (\neg H \rightarrow C) \quad \equiv \neg P \vee (\neg \neg H \vee C)$   
 $\quad \quad \quad \equiv \neg P \vee H \vee C$
- R:  $P \rightarrow \neg H \quad \equiv \neg P \vee \neg H$
- $\neg S$ :  $\neg(P \rightarrow C) \quad \equiv \neg(\neg P \vee C)$   
 $\quad \quad \quad \equiv \neg \neg P \wedge \neg C$   
 $\quad \quad \quad \equiv P \wedge \neg C$

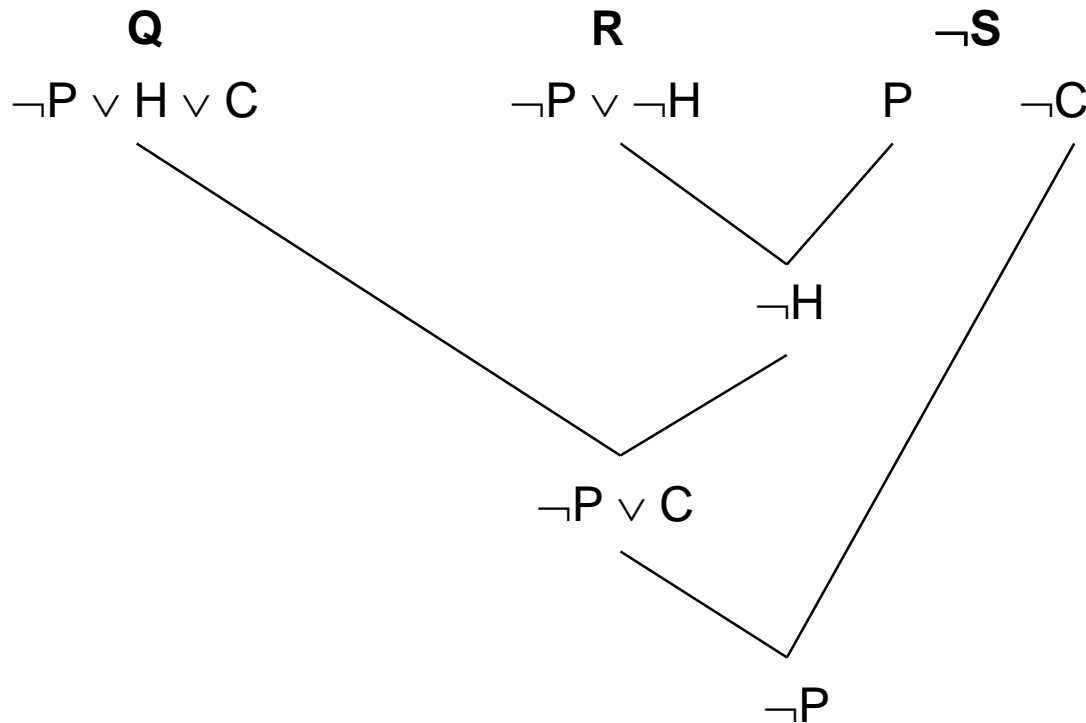
Wir erhalten somit folgende Klauseln:

$$\{\{\neg P, H, C\}, \{\neg P, \neg H\}, \{P\}, \{\neg C\}\}$$

# Resolution in der Aussagenlogik

## Beispiel

3. Bilden der Resolvente (lineare Resolution, Variante 1):



Ableitung als Folge

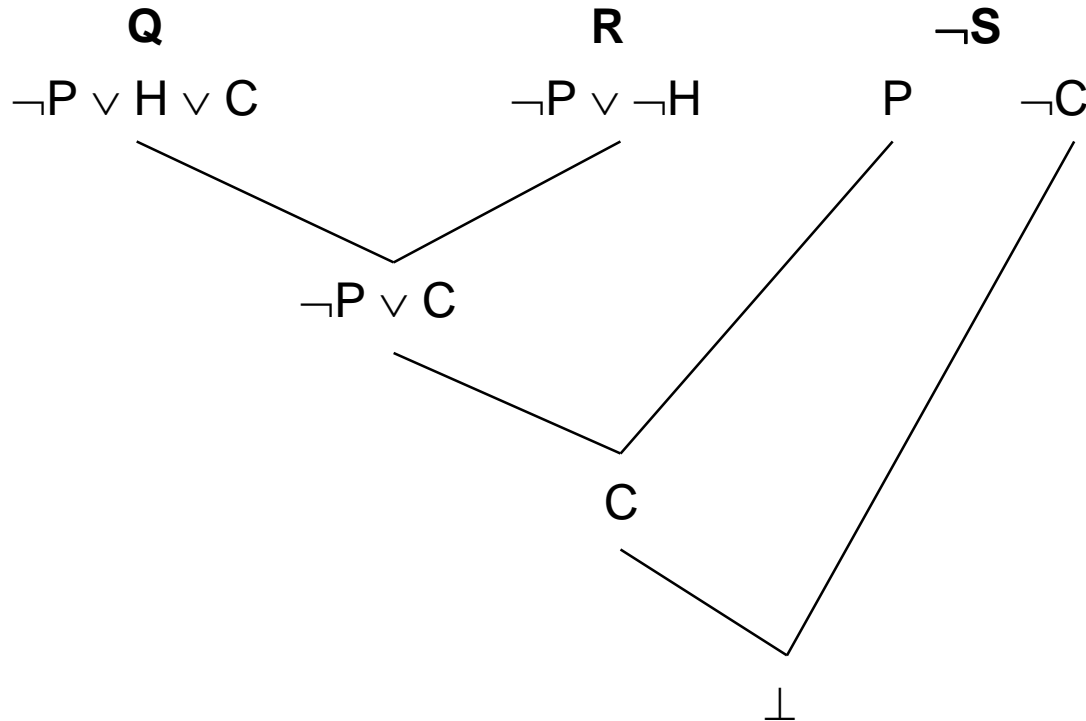
$(\{\neg P \vee H \vee C\},$   
 $\{\neg P \vee \neg H\},$   
 $\{P\},$   
 $\{\neg H\},$   
 $\{\neg P \vee C\},$   
 $\{\neg C\},$   
 $\{\neg P\})$

➔ Resolution führt nicht zum Widerspruch, nächste Variante...

# Resolution in der Aussagenlogik

## Beispiel

3. Bilden der Resolvente (lineare Resolution, Variante 2):



Ableitung als Folge

$(\{\neg P \vee H \vee C\},$

$\{\neg P \vee \neg H\},$

$\{\neg P \vee C\},$

$\{P\},$

$\{C\},$

$\{\neg C\},$

$\emptyset)$

➔ Resolution führt zum Widerspruch, d.h. die ursprüngliche Aussage ist wahr. Es gilt also  $\{Q, R\} \vdash S$ .

## 2.4 Prädikatenlogik

Die Prädikatenlogik stellt eine Erweiterung der Aussagenlogik dar.

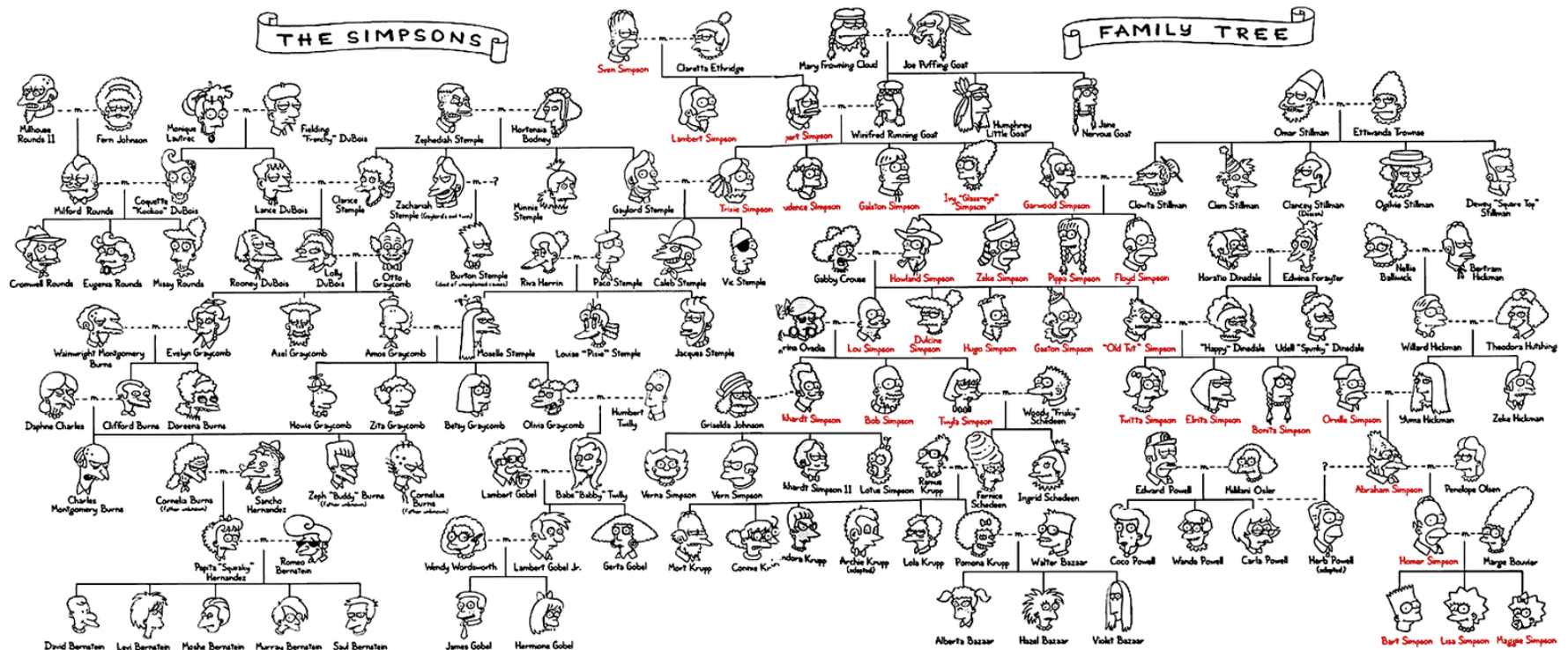
Prädikatenlogik dient

- zur Formulierung von Axiomen (Grundannahmen, die nicht begründet werden) und Schlussfolgerungen daraus,
- als Grundlage logischer Programmiersprachen,
- zum automatisierten Beweisen von Theoremen und Programmen (siehe *Theorem Proving* und *Model Checking*),
- als Grundlage für Expertensysteme und künstlichen Intelligenz.

Wir betrachten zunächst einige der Grundbegriffe, insbesondere **Prädikate** und **Quantoren** anhand von einfachen Beispielen.

# Prädikatenlogik

## Verwandschaft



Quelle: [https://simpsons.fandom.com/wiki/File:The Simpson's Family Tree of Homer.png](https://simpsons.fandom.com/wiki/File:The_Simpson's_Family_Tree_of_Homer.png)

# Prädikatenlogik

## Grenzen der Aussagenlogik

Mittels Aussagenlogik kann festgehalten werden, ob zwei Personen verwandt sind.

**Beispiel:** „Bart und Lisa sind Geschwister.“

Wir können mehrere Aussagen bisher jedoch nicht in Beziehung setzen.

**Beispiel:**

- $A :=$  „Homer und Marge sind ein Ehepaar.“
- $B :=$  „Lisa ist eine Tochter von Marge.“

Aber wie hängen diese beiden Aussagen zusammen? Die Schlussfolgerung, dass dann auch Lisa und Homer verwandt sind, lässt sich mit Aussagenlogik nicht ableiten.

# Prädikatenlogik

## Prädikatenlogik erweitert die Aussagenlogik

Mittels der Prädikatenlogik können Eigenschaften und Beziehungen angegeben werden.

### Beispiele:

- $W(m)$  „Marge ist weiblich.“
- $E(h,m)$  „Homer und Marge sind ein Ehepaar.“
- $T(l,m)$  „Lisa ist eine Tochter von Marge.“

Nun können mit Prädikatenlogik Aussagen getroffen werden wie z.B.:

- „Wenn x eine Tochter von y ist sowie y und z ein Ehepaar sind, dann gilt x auch mit z als verwandt.“
- Formal:  $(T(x,y) \wedge (E(y,z) \vee E(z,y))) \rightarrow V(x,z)$

# Prädikatenlogik

## Prädikate

**Prädikate** beschreiben Eigenschaften von Objekten und Beziehungen zwischen Objekten. Wir stellen Prädikate in der Regel durch großgeschriebene Buchstaben dar, gefolgt von den Objekten in Klammern, d.h. in der Form  $P(x_1, \dots, x_n)$  mit  $n \geq 0$ .

### Beispiele:

- |                        |          |  |           |
|------------------------|----------|--|-----------|
| ■ 0-stelliges Prädikat | R        | „Es regnet.“                           | (Aussage) |
| ■ 1-stelliges Prädikat | N(x)     | „x wird nass.“                         |           |
| ■ 2-stelliges Prädikat | L(x,y)   | „x liebt y.“                           |           |
| ■ 3-stelliges Prädikat | K(x,y,z) | „x ist Kind von Mutter y und Vater z.“ |           |



**Anmerkung:** Der Begriff Prädikat wird hier anders verwendet als in der deutschen Grammatik (dort entspricht ein Prädikat einer Satzaussage, die das Hauptverb enthält).



# Prädikatenlogik

## Formale Definitionen

**Definition:** Ein **n-stelliges Prädikat** über einer Menge  $M$  (**Diskursuniversum**) ist eine Abbildung:

$$P: \underbrace{M \times M \times \dots \times M}_{n\text{-mal}} \rightarrow \{0, 1\}.$$

**Definition:** Ein **n-stelliges Prädikat  $P$**  heißt **erfüllbar**, wenn es mindestens eine Belegung  $x_1, x_2, \dots, x_n \in M$  gibt, so dass  $P(x_1, x_2, \dots, x_n) = 1$ .

**Definition:** Ein **n-stelliges Prädikat  $P$**  heißt **gültig**, wenn für alle  $x_1, x_2, \dots, x_n \in M$  gilt, dass  $P(x_1, x_2, \dots, x_n) = 1$ .

# Prädikatenlogik

## Verknüpfung von Prädikaten

Wir können Prädikate als Teil logischer Aussagen verwenden und miteinander zu einer **prädikatenlogischen Formel** verknüpfen.

### Beispiele:

- „x ist ein Mensch, daher ist x sterblich.“  $M(x) \rightarrow S(x)$
- „x hat einen Schirm, daher wird x nicht nass.“  $S(x) \rightarrow \neg N(x)$
- „Es regnet nicht, daher wird x nicht nass.“  $\neg R \rightarrow \neg N(x)$

### Erfüllbarkeit und Gültigkeit:

- Eine **Formel** heißt **erfüllbar**, wenn sie mindestens eine Interpretation besitzt, welche die Formel zu einem erfüllbaren Prädikat macht.
- Eine **Formel** heißt **gültig**, wenn jede Interpretation die Formel zu einem erfüllbaren Prädikat macht.

# Prädikatenlogik

## Quantoren

Neben den Junktoren der Aussagenlogik  $\neg$ ,  $\vee$ ,  $\wedge$ , ... verwendet die Prädikatenlogik **Quantoren**:

$\forall$  heißt **Allquantor** und bedeutet: „für alle“ (oder „jeden“).

$\exists$  heißt **Existenzquantor** und steht für: „es existiert“ (oder „es gibt“).

### Beispiele:

- $(\exists x) N(x)$  „Es gibt jemanden, der nass wird.“
- $\neg(\exists x) N(x)$  „Es gibt niemanden, der nass wird.“  
( $\neg(\exists x)$  wird auch geschrieben als  $\nexists x$ .)
- $(\forall x) S(x)$  „Alle haben einen Schirm.“
- $\neg(\forall x) S(x)$  „Nicht alle haben einen Schirm.“
- $R \rightarrow (\forall x) (S(x) \vee N(x))$  „Wenn es regnet, gilt für jede\*n: sie\*er hat einen Schirm oder sie\*er wird nass.“

**Anmerkung:** Die Klammerung um  $\exists$  bzw.  $\forall$  dient der Lesbarkeit, kann aber auch weggelassen werden (z.B.  $\exists x N(x)$ ).

# Prädikatenlogik

## Freie und gebundene Variablen

### Definition:

- Eine Variable heißt (durch einen Quantor) **gebunden**, wenn sie innerhalb des Wirkungsbereichs dieses Quantors vorkommt.
- Nicht gebundene Variablen heißen **frei**. Ob eine Aussage wahr ist, hängt in der Regel auch von der Belegung der freien Variablen ab.

### Beispiel: $(\exists y) P(x,y)$

- Die Variable  $y$  ist gebunden durch den Existenzquantor.
- $x$  ist eine freie Variable.

**Definition:** Eine Formel ohne freie Variablen wird **geschlossen** genannt, ansonsten heißt sie **offen**.


# Prädikatenlogik

## Kombination von Quantoren

Es ist auch möglich, mehrere Quantoren in einer Aussage zu verwenden.

### Beispiel:

„Jede\*r liebt irgendjemandem.“  $(\forall x) (\exists y) L(x,y)$   
(sprich: „für jedes x existiert ein y, so dass ...“)

 **Anmerkung:** Man beachte, dass beim Vertauschen der Quantoren eine andere Aussage entsteht.

„Jemand wird von allen geliebt.“  $(\exists y) (\forall x) L(x,y)$

Vertauschen von gleichen Quantoren ist möglich:

„Jede\*r liebt jede\*n.“  $(\forall x) (\forall y) L(x,y) \equiv (\forall y) (\forall x) L(x,y)$

„Irgendjemand liebt irgendjemand.“  $(\exists x) (\exists y) L(x,y) \equiv (\exists y) (\exists x) L(x,y)$

Keine Spalte/Zeile ist leer:

	a	b	c	d	e
a					
b					
c					
d					
e					

1.  $\forall x \exists y L y x$ :  
Jeder wird von jemandem geliebt.

	a	b	c	d	e
a					
b					
c					
d					
e					

2.  $\forall x \exists y L x y$ :  
Jeder liebt jemanden.

Eine Zeile/Spalte ist voll:

	a	b	c	d	e
a					
b					
c					
d					
e					

3.  $\exists x \forall y L x y$ :  
Jemand liebt alle.

	a	b	c	d	e
a					
b					
c					
d					
e					

4.  $\exists x \forall y L y x$ :  
Jemand wird von allen geliebt.

Die Diagonale ist nichtleer/voll:

	a	b	c	d	e
a					
b					
c					
d					
e					

5.  $\exists x L x x$ :  
Jemand liebt sich selbst.

	a	b	c	d	e
a					
b					
c					
d					
e					

6.  $\forall x L x x$ :  
Alle lieben sich selbst.

Die Matrix ist nichtleer/voll:

	a	b	c	d	e
a					
b					
c					
d					
e					

7.  $\exists x \exists y L x y$ :  
Einer liebt einen.

8.  $\exists x \exists y L y x$ :  
Einer wird von einem geliebt.

	a	b	c	d	e
a					
b					
c					
d					
e					

9.  $\forall x \forall y L x y$ :  
Jeder liebt jeden.

10.  $\forall x \forall y L y x$ :  
Jeder wird von jedem geliebt.

Quelle: Wikipedia, Prädikatenlogik – <https://de.wikipedia.org/wiki/Pr%C3%A4dikatenlogik>

# Prädikatenlogik

## Quiz 3 (per Moodle) – Fressen und gefressen werden

Betrachten wir Tierarten als Diskursuniversum. Seien folgende Prädikate gegeben:

- $F(x,y)$ : „x fressen y.“ (z.B. „Katzen fressen Mäuse.“)
- $L(x)$ : „x sind Landtiere.“ (z.B. „Katzen sind Landtiere.“)
- $W(x)$ : „x sind Wassertiere.“ (z.B. „Fische sind Wassertiere.“)



Formulieren Sie die folgenden Aussagen in Worten unter Verwendung der Begriffe „Tierart“, „Landtiere“, „Wassertiere“.

1.  $(\exists x) (L(x) \wedge W(x))$
2.  $(\exists x) (\exists y) (F(x,y) \wedge L(x) \wedge W(y))$
3.  $\neg(\exists x) (\forall y) (L(x) \wedge (W(y) \rightarrow F(x,y)))$
4.  $(\forall x) (L(x) \rightarrow \neg(\exists y) (W(y) \wedge F(y,x)))$

# Prädikatenlogik

## Quiz 3 (per Moodle) – Fressen und gefressen werden

Betrachten wir Tierarten als Diskursuniversum. Seien folgende Prädikate gegeben:

- $F(x,y)$ : „x fressen y.“ (z.B. „Katzen fressen Mäuse.“)
- $L(x)$ : „x sind Landtiere.“ (z.B. „Katzen sind Landtiere.“)
- $W(x)$ : „x sind Wassertiere.“ (z.B. „Fische sind Wassertiere.“)



Formulieren Sie die folgenden Aussagen in Worten unter Verwendung der Begriffe „Tierart“, „Landtiere“, „Wassertiere“.

1.  $(\exists x) (L(x) \wedge W(x))$   
„Es gibt eine Tierart, die sowohl Landtiere als auch Wassertiere sind.“
2.  $(\exists x) (\exists y) (F(x,y) \wedge L(x) \wedge W(y))$   
„Es gibt Landtiere, die Wassertiere fressen.“
3.  $\neg(\exists x) (\forall y) (L(x) \wedge (W(y) \rightarrow F(x,y)))$   
„Es gibt keine Landtiere, die alle Arten von Wassertieren fressen.“
4.  $(\forall x) (L(x) \rightarrow \neg(\exists y) (W(y) \wedge F(y,x)))$   
„Landtiere werden nicht von Wassertieren gefressen.“



# Prädikatenlogik

## Ergänzung zu Quiz 3.3

„Es gibt keine Landtiere, die alle Arten von Wassertieren fressen.“

$$\begin{aligned} & \neg(\exists x) (\forall y) (L(x) \wedge (W(y) \rightarrow F(x,y))) \\ & \equiv (\forall x) \neg(\forall y) (L(x) \wedge (W(y) \rightarrow F(x,y))) \\ & \equiv (\forall x) (\exists y) \neg(L(x) \wedge (W(y) \rightarrow F(x,y))) \\ & \equiv (\forall x) (\exists y) (\neg L(x) \vee \neg(W(y) \rightarrow F(x,y))) \\ & \equiv (\forall x) (\exists y) (\neg L(x) \vee \neg(\neg W(y) \vee F(x,y))) \\ & \equiv (\forall x) (\exists y) (\neg L(x) \vee (W(y) \wedge \neg F(x,y))) \\ & \equiv (\forall x) (\exists y) (L(x) \rightarrow (W(y) \wedge \neg F(x,y))) \end{aligned}$$

„Für alle Landtiere gibt es eine Art von Wassertieren, die diese nicht fressen.“

# Prädikatenlogik

## Beziehung zwischen ( $\forall$ und $\wedge$ ) sowie ( $\exists$ und $\vee$ )

Sei  $\{x_1, \dots, x_n\}$  eine endliche Menge des Diskursuniversums.

- Eine durch den **Allquantor** dargestellte Aussage ist äquivalent mit einer **n-fachen Konjunktion**:  $(\forall x) P(x) \equiv \bigwedge_{i=1}^n P(x_i)$ .
- Eine durch den **Existenzquantor** dargestellte Aussage ist äquivalent mit einer **n-fachen Disjunktion**:  $(\exists x) P(x) \equiv \bigvee_{i=1}^n P(x_i)$ .

**Beispiele:** Sei  $G(x)$  das Prädikat „ $x$  ist eine gerade Zahl“.

- Sei  $\{2, 4, 6, 8\}$  eine endliche Menge. Dann gilt
$$(\forall x) G(x) \equiv G(2) \wedge G(4) \wedge G(6) \wedge G(8) \equiv 1.$$
- Sei  $\{2, 3, 5, 11\}$  eine endliche Menge. Dann gilt
$$(\exists x) G(x) \equiv G(2) \vee G(3) \vee G(5) \vee G(11) \equiv 1.$$

**Anmerkung:** Bei unendlichen Diskursuniversen bezieht sich die Konjunktion/Disjunktion auf unendlich viele Elemente.

# Prädikatenlogik

## Beziehung zwischen $\exists$ und $\forall$

Quantoren können jeweils durch den anderen ausgedrückt werden:

$$\begin{aligned} (\exists x) P(x) &\equiv \neg(\forall x)(\neg P(x)) \\ \text{„Es gibt ein } x, \text{ sodass } P(x) \text{ gilt.“} &\equiv \text{„Nicht für alle } x \text{ gilt } P(x) \text{ nicht.“} \end{aligned}$$

$$\begin{aligned} \neg(\exists x) P(x) &\equiv (\forall x)(\neg P(x)) \\ \text{„Es gibt kein } x, \text{ sodass } P(x) \text{ gilt.“} &\equiv \text{„Für alle } x \text{ gilt } P(x) \text{ nicht.“} \end{aligned}$$

$$\begin{aligned} (\exists x) (\neg P(x)) &\equiv \neg (\forall x) P(x) \\ \text{„Es gibt ein } x, \text{ sodass } P(x) \text{ nicht gilt.“} &\equiv \text{„Nicht für alle } x \text{ gilt } P(x) \text{.“} \end{aligned}$$

$$\begin{aligned} \neg(\exists x) (\neg P(x)) &\equiv (\forall x) P(x) \\ \text{„Es gibt kein } x, \text{ sodass } P(x) \text{ nicht gilt.“} &\equiv \text{„Für alle } x \text{ gilt } P(x) \text{.“} \end{aligned}$$

# Prädikatenlogik

## Logische Programmierung

Logische Programmiersprachen verwenden Prädikatenlogik als Basis.

- Programme setzen sich aus Fakten und Regeln zusammen, die mittels Prädikaten formuliert werden.
- Im Gegensatz zu imperativen Sprachen (wie z.B. Java) sind logische Programmiersprachen **deklarativ** (d.h. das Problem wird beschrieben, nicht der Lösungsweg).
- Aus Gründen der Komplexität der Berechnungen werden nur Teile der Prädikatenlogik verwendet und basiert auf sog. Horn-Klauseln.
- Logische Programmiersprachen werden vor allem im Kontext künstlicher Intelligenz verwendet (z.B. IBM Watson).
- **Prolog** ist der bekannteste Vertreter dieser Klasse von Programmiersprachen.

# Prädikatenlogik

## Horn-Klauseln

**Definition:** Eine **Horn-Klausel** ist eine Klausel in der alle Literale negiert sind, mit Ausnahme höchstens eines Literals, d.h.

$$\neg p_1 \vee \neg p_2 \vee \dots \vee \neg p_{k-1} \vee p_k \text{ oder } \neg p_1 \vee \neg p_2 \vee \dots \vee \neg p_k.$$

**Bemerkung:** Eine Horn-Klausel mit einem positiven Literal lässt sich wie folgt als Implikation auffassen (auch geschrieben als umgekehrte Implikation):

$$\begin{aligned} \neg p_1 \vee \neg p_2 \vee \dots \vee \neg p_{k-1} \vee p_k &\equiv \neg(p_1 \wedge p_2 \wedge \dots \wedge p_{k-1}) \vee p_k \\ &\equiv (p_1 \wedge p_2 \wedge \dots \wedge p_{k-1}) \rightarrow p_k \\ &\equiv p_k \leftarrow (p_1 \wedge p_2 \wedge \dots \wedge p_{k-1}) \end{aligned}$$

## Anwendung in der logischen Programmierung:

- Umgekehrte Implikationen wie diese werden in logischen Programmiersprachen als Regeln notiert.
- Die Implementierung logische Programmiersprachen basiert auf Horn-Klauseln und Resolutionen (siehe oben).
- Die Resolvente zweier Horn-Klauseln ist wieder eine Horn-Klausel.

# Prädikatenlogik

## Prolog

```
% append(Xs, Ys, Zs) :-  
%     Zs is the result of concatenating lists Xs and Ys.  
  
append([ ], Ys, Ys).  
append([X | Xs], Ys, [X | Zs]) :-  
    append(Xs, Ys, Zs).
```

A Taste of Prolog – 5:39: <https://www.youtube.com/watch?v=pJvd4Fk95XI>

# Prädikatenlogik

## Beispiel in Prolog

**/\*Fakten:\*/**

```
landtiere (mäuse) .  
landtiere (katzen) .  
landtiere (krokodile) .  
wassertiere (fische) .  
wassertiere (krokodile) .  
fressen (katzen,mäuse) .  
fressen (krokodile,fische) .  
fressen (krokodile,katzen) .  
fressen (katzen,fische) .
```

**/\*Abfragen:\*/**

```
?- landtiere(X), wassertiere(X) .  
?- fressen(X,Y) .  
?- retter(X,Y) .  
?- retter(X,Y), \+echterretter(X,Y) .
```

**/\*Regeln:\*/**

```
retter(X, Y) :-  
    fressen(X, Z) ,  
    fressen(Z, Y) .  
echterretter(X, Y) :-  
    fressen(X, Z) ,  
    fressen(Z, Y) ,  
    \+ fressen(X, Y) .
```

### Prolog-Syntax:

- :- entspricht  $\leftarrow$   
(umgekehrte Implikation)
- , entspricht  $\wedge$
- \+ entspricht  $\neg$

### Online Prolog-Umgebung:

<https://swish.swi-prolog.org/>

# Zusammenfassung

- **Prinzip der Zweiwertigkeit:** Aussagen sind entweder wahr oder falsch.
- **Junktoren** ( $\neg$ ,  $\vee$ ,  $\wedge$ , ...) erlauben das Zusammensetzen von Aussagen.
- Aussagenlogische Funktionen können in **Programmen** als Methoden implementiert werden oder in Hardware als **Schaltung** mittels logischer Gatter.
- Beliebige aussagenlogische Formeln können in **konjunktive und disjunktive Normalformen** gebracht werden.
- Basierend auf der konjunktiven Normalform werden diese als Klauselmengen geschrieben aus denen sich Schlussfolgerungen nach dem **Resolutionsprinzip** ableiten lassen.
- **Resolutionswiderlegung:** Zum Beweis einer Schlussfolgerung wird diese negiert als Klauselmenge geschrieben und zu einem Widerspruch geführt.
- **Prädikate** beschreiben Eigenschaften und Beziehungen zwischen Objekten zu beschreiben
- **Quantoren** ( $\exists$  und  $\forall$ ) spezifizieren die Erfüllbarkeit und Gültigkeit von prädikatenlogischen Aussagen.
- In der **logischen Programmierung** werden Prädikatenlogik und Resolutionen angewendet.



# Literaturempfehlungen

1. V. Aho, J. D. Ullman: „Informatik - Datenstrukturen und Konzepte der Abstraktion“; International Thomson Publishing, 1996, ISBN 3-8266-0242-0.
2. H. Eirund, B. Müller, G. Schreiber: „Formale Beschreibungsverfahren der Informatik“; Teubner, 2000 (Kap. 2).
3. J. Kelly: „Logik im Klartext“<sup>3</sup>; Pearson-Studium, 2003 (Kap. 1, 3, 5 und 6).