

# 1 Grundlagen

## 1.1 Logische Grundgesetze

<b>Doppelte Negation</b>	$x = \neg(\neg x)$
<b>Kommutativgesetz</b>	$x \wedge y = y \wedge x$ $x \vee y = y \vee x$
<b>Assoziativgesetz</b>	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$ $x \vee (y \vee z) = (x \vee y) \vee z$
<b>Distributivgesetz</b>	$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
<b>Idempotenz</b>	$x \wedge x = x$ $x \vee x = x$
<b>Tautologie</b>	$x \vee \neg x = 1$
<b>Kontradiktion</b>	$x \wedge \neg x = 0$
<b>Absorption</b>	$x \wedge (x \vee y) = x$ $x \vee (x \wedge y) = x$
<b>Neutralität</b>	$(x \wedge \neg x) \vee y = y$ $(x \vee \neg x) \wedge y = y$
<b>Negation</b>	$x \wedge \neg x = 0$ $x \vee \neg x = 1$
<b>De Morgan</b>	$\neg(x \wedge y) = \neg x \vee \neg y$ $\neg(x \vee y) = \neg x \wedge \neg y$
<b>Operationen mit 0 und 1</b>	$0 \wedge x = 0$ $1 \wedge x = x$ $\neg 0 = 1$

## 1.2 Schlussregeln

<b>Modus Ponens</b>	$(a \Rightarrow b) \wedge a \Rightarrow b$
<b>Modus Tollens</b>	$(a \Rightarrow b) \wedge \neg b \Rightarrow \neg a$
<b>Hypothetischer Syllogismus</b>	$(a \Rightarrow b) \wedge (b \Rightarrow c) \Rightarrow (a \Rightarrow c)$
<b>Disjunktiver Syllogismus</b>	$(a \vee b) \wedge \neg a \Rightarrow b$

## 1.3 Quantoren

$\forall x p = \neg(\exists x \neg p)$	$\exists x p = \neg(\forall x \neg p)$
$\neg \forall x p = (\exists x \neg p)$	$\neg \exists x p = (\forall x \neg p)$

## 1.4 Implikation

$(a \Rightarrow b) = \neg a \vee b$
$(a \Rightarrow b) = (\neg b \Rightarrow \neg a)$
$(a \Rightarrow b) = \neg(a \wedge \neg b)$

## 1.5 Operatorenpräzedenz

Von stark bindend zu schwach bindend:

Negation  $\neg$ , {Konjunktion  $\cap$ , Disjunktion  $\cup$ }, {Implikation  $\rightarrow$ , Äquivalenz  $\leftrightarrow$ }

# 2 Mengen

**Teilmenge:** Eine Menge A heisst Teilmenge einer Menge B, wenn jedes Element von A auch in B enthalten ist.  $A \subseteq B :\Leftrightarrow \forall x(x \in A \rightarrow x \in B)$

**Schnittmenge:** Die Menge aller Elemente, die sowohl in A als auch in B enthalten sind.  $A \cap B := \{x | (x \in A) \wedge (x \in B)\}$

**Vereinigung:** Menge aller Elemente, die in A oder in B enthalten sind.  $A \cup B := \{x | (x \in A) \vee (x \in B)\}$

**Komplement:** Menge aller Elemente, die nicht in A enthalten sind.  $\bar{A} = \{x | x \notin A\}$

**Differenz:** Menge aller Elemente, die in A, jedoch nicht in B enthalten sind.  $A \setminus B = \{x | (x \in A) \wedge (x \notin B)\}$

**Symmetrische Differenz:** Menge aller Elemente, die entweder

A oder in B, nicht jedoch in beiden enthalten sind.  $A \Delta B := (A \setminus B) \cup (B \setminus A) = (A \cup B) \setminus (A \cap B)$

**Potenzmenge:** Die Potenzmenge  $P(A)$  von A ist die Menge aller Teilmengen von A. Sie enthält  $2^n$  Elemente (jedes Element kann enthalten sein oder nicht).

**Beispiel:** Potenzmenge von  $A = \{1, 2, 3\}$   
 $\mathbb{P}(A) = \{\{\emptyset\}, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$

## 2.1 Paare

**Menge:**  $\{a, b\} = \{b, a\}$

**Geordnetes Paar:**  $(a, b) \neq (b, a)$

## 2.2 Vollständige Induktion

Annahme, Verankerung ( $n=0$ ), Induktionsschritt ( $n \rightarrow n+1$ )

# 3 Aussagenlogik

## 3.1 Syntax der Aussagenlogik

Eine **atomare Formel** hat die Form  $A_i$  mit  $i = 1, 2, 3, \dots$  **Formeln** werden folgendermassen definiert:

1. Alle atomaren Formeln sind Formeln.
2. Für alle Formeln F und G sind  $(F \wedge G)$  und  $(F \vee G)$  Formeln.
3. Für jede Formel F ist  $\neg F$  eine Formel.

**Beispiel: Formeln**

$(A \wedge B \vee C)$  ist falsch, da Präzedenz nicht definiert ist.

$((A \wedge B) \vee C)$  ist korrekt

## 3.2 Semantik der Aussagenlogik

<b>Konjunktion</b>		<b>Negation</b>		<b>Disjunktion</b>	
$\wedge$	$\begin{matrix} 1 & 0 \\ 1 & 0 \\ 0 & 0 \end{matrix}$	$\neg$	$\begin{matrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{matrix}$	$\vee$	$\begin{matrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{matrix}$
<b>Implikation</b>		<b>Äquivalenz</b>			
$\rightarrow$	$\begin{matrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{matrix}$	$\leftrightarrow$	$\begin{matrix} 1 & 0 \\ 1 & 1 \\ 0 & 0 \end{matrix}$		

## 3.3 Begriffe

**Tautologie:** Eine Formel ist **allgemeingültig**, wenn sie immer wahr ist, unabhängig von der Belegung der Aussagevariablen.

**Erfüllbar:** Eine Formel ist erfüllbar, wenn es möglich ist, sie wahr zu machen.

**unerfüllbar:** Es existiert *keine* erfüllende Belegung.

## 3.4 Negationsnormalform (NNF)

1. **Äquivalenzen entfernen**

$$p \leftrightarrow q = (p \rightarrow q) \wedge (q \rightarrow p)$$

2. **Implikationen entfernen**

$$p \rightarrow q = \neg p \wedge q$$

3. **Negation in Formel hineinziehen** → De Morgan

$$\neg(p \vee q) = \neg p \wedge \neg q$$

$$\neg(p \wedge q) = \neg p \vee \neg q$$

$$\neg\neg p = p$$

Beispiel: NNF

?

3.5 **Konjunktive Normalform (KNF)**

1. **Formel in Negationsnormal umformen**
2. **Distributivgesetze**  $p \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$   
 $(q \wedge r) \vee p = (q \vee p) \wedge (r \vee p)$
3. **Vereinfachungen**

Die KNF lässt sich aus den 0en der Wertetabelle herauslesen.  
 Beispiel:

A	B	F
1	1	0
1	0	1
0	1	1
0	0	0

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

Beispiel: KNF

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

3.6 **Disjunktive Normalform (DNF)**

1. **Formel in die negationsnormalform umformen**
2. **Distributivgesetze**  $(p \vee q) \wedge r = (p \wedge r) \vee (q \wedge r)$   
 $r \wedge (p \vee q) = (r \wedge p) \vee (r \wedge q)$
3. **Vereinfachungen**

A	B	F
1	1	0
1	0	1
0	1	1
0	0	0

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

Beispiel: DNF

$$(p \wedge q \wedge r) \vee (p \wedge q \wedge q \wedge \neg r) \dots$$

3.7 **Modell**

Sei F eine Formel und A eine Belegung. A ist ein **Modell** für F, falls  $\mathbb{A}(F) = 1$ . Man schreibt:

$$\mathbb{A} \models F$$

G ist eine **semantische Folgerung** von  $F_1, F_2, \dots, F_k$  falls gilt: Wenn A Modell von  $\{F_1, F_2, \dots, F_k\}$  ist (d.h.  $F_1 = \dots = F_k = 1$ ), dann ist A auch Modell von G. Man schreibt:

$$F_1, F_2, \dots, F_k \models G$$

1. G ist eine Folgerung von  $F_1, \dots, F_k$
2.  $((\bigwedge_{i=1}^k F_i) \rightarrow G)$  ist eine Tautologie
3.  $((\bigwedge_{i=1}^k F_i) \wedge \neg G)$  ist unerfüllbar

Beispiel: Modell

$$F = (p \wedge q) \rightarrow (\neg r)$$

$\mathbb{A}(p) = 1, \mathbb{A}(q) = 1, \mathbb{A}(r) = 0$  ist ein Modell für F

4 **Resolution**

**Literal L:** Aussagevariable p oder deren Negation  $\neg p$ . p heisst **positives Literal**,  $\neg p$  heisst **negatives Literal**.

**Klausel C:** Endliche Menge von Literalen  $\{L_1, L_2, \dots, L_n\}$ . Eine Klausel C entspricht einer **Disjunktion:**  $L_1 \vee L_2 \vee \dots \vee L_n$

**Beispiel: Klausel**  $\{p, \neg q, r, \neg p\} = p \vee \neg q \vee r \vee \neg p$

**Klauselmeng S:** (Evtl. unendliche) Menge von Klauseln. Eine Klauselmeng S entspricht einer **Konjunktiven Normalform:**  $\{C_1, C_2, \dots, C_n\} = C_1 \wedge C_2 \wedge \dots \wedge C_n$

Beispiel:  $\{\{p, q\}, \{\neg r, s\}, \{\neg p, \neg s\}\} = (p \vee q) \wedge (\neg r \vee s) \wedge (\neg p \vee \neg s)$

4.1 **Vorgehen**

$$R = (K_1 - \{L\}) \cup (K_2 - \{\neg L\})$$

1. Wähle ein Literal L, so dass L in C positiv und in D negativ vorkommt
2. Streiche L in C und  $\neg L$  in D
3. Vereinige C und D

Beispiel:  $\{p, q, r\}$  und  $\{\neg p, s\}$  Resolvente:  $\{q, r, s\}$

**Resolvente:**

$$\text{Res}(S) = S \cup \{R | R \text{ ist Resolvente zweier Klauseln in } S\}$$

4.2 **Resolutionsbeweis**

Das Ziel der Resolution ist es, aus einer Klauselmeng S die leere Klausel  $\square$  herzuleiten. Gelingt dies, ist bewiesen, dass S **unerfüllbar** ist.

**Was lässt sich mit einem Resolutionsbeweis zeigen?**

1. Eine Klauselmeng S ist unerfüllbar
2. Eine Klauselmeng ist eine Tautologie:  
 $S$  ist eine Tautologie  $\leftrightarrow \neg S$  ist unerfüllbar
3. Eine Klausel C ist eine semantische Folgerung aus einer Klauselmeng S, d.h.  $S \models C$  g.d.w.  $S \cup \{\neg C\}$  unerfüllbar ist

4.3 **Hornformeln**

Eine Formel ist eine **Hornformel**, falls F in KNF ist und jedes Disjunktionsglied in F höchstens ein positives Literal enthält.

Es gibt 3 Typen von Klauseln:

$$\{A\}, \{\neg A_1, \dots, \neg A_n, B\}, \{\neg A_1, \dots, \neg A_n\}$$

Beispiel:  $F = (A \vee \neg B) \wedge (\neg C \vee \neg A \vee D) \wedge (\neg A \vee \neg B) \wedge (D) \wedge (\neg E)$

**Resolution ist mit Hornformeln viel effizienter!**

4.4 **Markierungsalgorithmus**

Klauseln einer Hornformel können auch als Implikation geschrieben werden.

$$F = (A \vee \neg B) \wedge (\neg C \vee \neg A \vee D) \wedge (\neg A \vee \neg B) \wedge (D) \wedge (\neg E)$$

$$\downarrow$$

$$F = (B \rightarrow A) \wedge ((C \wedge A) \rightarrow D) \wedge ((A \wedge B) \implies 0) \wedge (1 \rightarrow D) \wedge (E \rightarrow 0)$$

**Algorithmus**

Eingabe: Hornformel F (mit Implikationen)

1. Markiere alle Vorkommen einer atomaren Formel A in F, falls es in F eine Teilformel der Form  $(1 \rightarrow A)$  gibt.
2. **while** es gibt in F eine Teilformel G der Form  $((A_1 \wedge \dots \wedge A_n) \rightarrow B)$  oder  $((A_1 \wedge \dots \wedge A_n) \rightarrow 0), n \geq 1$ , wobei  $A_1, \dots, A_n$  schon markiert sind (und B noch nicht markiert ist).  
**do**  
 if G hat die erste Form  
 markiere alle Vorkommen von B in F  
**else**  
 Formel ist unerfüllbar  
**end**
3. Formel ist erfüllbar. Erfüllbare Belegung durch Markierung ersichtlich.

**Beispiel: Markierungsalgorithmus: Geburtstagsparty**

**Fakten:** E

**Regeln:**  $C \rightarrow D, D \wedge C \rightarrow A, B \wedge E \rightarrow C, E \rightarrow B$

**Query:** nimmt A teil?  $\{\neg A\}$

**Klauseln:**  $\{E\}, \{\neg C, \neg D, A\}, \{\neg B, \neg E, C\}, \{\neg E, B\}, \{\neg A\}$

- E
- C → D
- D ∧ C → A
- B ∧ E → C
- E → B
- ¬A

**4.5 SLD Resolution**

- Lineare Resolution für Klauseln mit Selektionsregeln
- Wird zur Logikprogrammierung mit Prolog verwendet
- Man beginnt mit dem negierten Ziel und versucht die leere Menge herzuleiten. Somit ist das Ziel in positiver Form aus den gegebenen Klauseln herleitbar.

**5 Natürliches Schliessen**

- Aus Annahmen will man Folgerungen schliessen
- Folgerungen sind Implikationen ( $\models, =, \rightarrow$ )
- Natürliches Schliessen ist ein Beweis von Tautologien.
- Für den Beweis steht ein Set von Herleitungsschritten zur Verfügung
- $F \rightarrow G$  g.d.w. es einen Ableitungsbaum von F nach G gibt

Semantik	Beweistheorie
Wahrheitstabellen	Beweise
$A_1, \dots, A_n \models B$	$A_1, \dots, A_n \vdash B$
$\models A$ Tautologie	$\vdash A$ Theorem
Junktorenvelfalt	Resolution
$\wedge, \vee, \rightarrow$	Natürliches Schliessen

Eine **Schlussregel** sieht folgendermassen aus:

$$\frac{A_1 \dots A_n}{B} \text{ (Name)}$$

$A_1 \dots A_n$  nennt man **Prämissen / Annahmen**. B nennt man **Konklusion / Folgerung**. Sind keine Prämissen vorhanden, so redet man von einem **Axiom**.

**Grundregeln**

$\frac{A \ B}{A \wedge B} (\wedge i)$	$\frac{A \wedge B}{A} (\wedge e)$	$\frac{A \wedge B}{B} (\wedge e)$	$\frac{A \ A \rightarrow B}{B} (MP)$
$\frac{A}{A \vee B} (\vee i)$	$\frac{B}{A \vee B} (\vee i)$	$\frac{A \vee B \ A \rightarrow C \ B \rightarrow C}{C} (\vee e)$	
$\frac{A \leftrightarrow B}{A \rightarrow B} (\leftrightarrow e)$	$\frac{A \leftrightarrow B}{B \rightarrow A} (\leftrightarrow e)$	$\frac{A \rightarrow B \ B \rightarrow A}{A \leftrightarrow B} (\leftrightarrow i)$	$\top (\top i)$
$\frac{A \rightarrow \perp}{\neg A} (\neg i)$	$\frac{A \ \neg A}{\perp} (\neg e)$	$\frac{\perp}{A} (EFQ)$	$\frac{}{A \vee \neg A} (TND)$

**Indirekter Beweis (IB)**

$[A]$	$[\neg A]$
$\vdots$	$\vdots$
$\frac{}{\perp}$	$\frac{}{\perp}$
$\frac{}{\neg A}$	$\frac{}{A}$

**Weitere Regeln**

$A \vee B$	$[A]$	$[B]$	$[A]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\frac{}{C}$	$\frac{}{C}$	$\frac{}{C}$	$\frac{}{B}$
$\frac{}{C}$	$\frac{}{C}$	$\frac{}{A \rightarrow B}$	

**Indirekter Beweis**

**Beispiel:**  $\neg \neg A \models A$

$\neg \neg A$	$[\neg A]$
$\frac{}{\perp}$	
$\frac{}{A}$	

**6 Prädikatenlogik**

**6.1 Prädikate und Quantoren**

Prädikatenlogik ist eine Erweiterung der Aussagenlogik mit Quantoren ( $\forall, \exists$ ), Funktionen und Prädikaten.

**Universum  $U_{\mathbb{A}}$ :** beliebige, aber nicht leere Menge. Beispiel:  $U_{\mathbb{A}} = \mathbb{N}$

**Interpretation  $I_{\mathbb{A}}$ :** Ordnet jedem Prädikat P, Funktionssymbol f und Variablen x der Formel eine Bedeutung zu. Beispiel:  $I_{\mathbb{A}}(P) = P^{\mathbb{A}} = \{(m, n) \mid m, n \in U_{\mathbb{A}} \text{ und } m < n\}$

**Struktur:** Universum und Interpretation. Beispiel:  $\mathbb{A} = (U_{\mathbb{A}}, I_{\mathbb{A}})$

**Variablen:** bezeichnen beliebige Objekte des Universums.

Beispiel: x, y, z

**Konstanten:** bezeichnen feste Objekte des Universum. Beispiel: Die Zahl 0

**Funktionen:** operieren auf dem Universum, bzw. liefert ein Objekt aus diesem. Beispiel: add(x,y)

**Gleichungen:**  $s \approx t$  bedeutet, dass die Terme s und t denselben Wert haben im Universum.

**Prädikate:**  $R(t_1, t_2, \dots, t_n)$  heisst, dass das Prädikat R auf  $t_1, \dots, t_n$  zutrifft (liefert wahr / falsch). Beispiel:  $I_{\mathbb{A}}(Q) = \{n \in U_{\mathbb{A}} \mid n \text{ ist Primzahl}\}$

**Allquantor:**  $\forall x A$  ist wahr, falls A wahr ist für jedes x aus dem Universum.  $\forall x A$  ist falsch, wenn es ein Objekt gibt, so dass A falsch ist.

**Existenzquantor:**  $\exists x A$  ist wahr, falls mindestens ein x existiert, für das A wahr ist.  $\exists x A$  ist falsch, falls A falsch ist für alle x aus dem Universum.

**6.2 Syntax der Prädikatenlogik**

**Term:**

1. Jede Variable ist ein Term
2. Ist  $f$  eine Funktion und  $t_1, \dots, t_k$  Terme, dann ist  $f(t_1, \dots, t_k)$  ein Term

**Formel:**

1.  $P$  ist ein Prädikat,  $t_1, \dots, t_k$  Terme, dann ist  $P(t_1, \dots, t_k)$  eine Formel
2. Sind  $t_1, t_2$  Terme, dann ist  $(t_1 = t_2)$  eine Formel
3.  $F, G$  Formeln, dann sind  $(\neg F), (F \wedge G), (F \vee G)$  Formeln
4.  $F$  Formel,  $x$  Variable, dann sind  $\forall x(F)$  und  $\exists x(F)$  Formeln

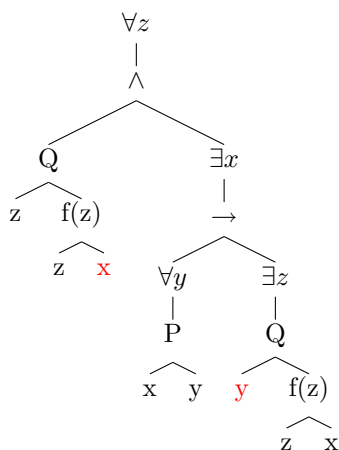
Begriff	Namen	Beispiel
Variablen	$x, y, z$	
Relationssymbole	$P, Q, R$	
Funktionssymbole	$f, g$	
Konstanten	$a, b, c$	
Terme	$r, s, t$	$\text{push}(\text{add}(x2,1),x3)$
Formeln	$A, B, C$	$\forall x(Q(x) \wedge R(x) \wedge \neg \exists y(P(y)))$
Menge von Formeln	$\Gamma, \Phi, \Psi$	

**Präzedenz**

Quantoren binden stärker als aussagenlogische Operatoren:  
 $\exists x Q(x) \wedge R(x) = (\exists x Q(x)) \wedge R(x)$   
 Eine Folge von gleichen Quantoren kann zusammengefasst werden:  
 $\forall x, y, z(R(x, y, z))$  steht für  $\forall x \forall y \forall z(R(x, y, z))$

**Syntaxbaum**

$\forall z(Q(z, f(z, x)) \wedge \exists x(\forall y(P(x, y)) \rightarrow \exists z(Q(y, f(z, x))))$   
 Freie Variablen sind rot markiert.



**6.3 Semantik der Prädikatenlogik**

Zwei Formel  $F$  und  $G$  sind semantisch äquivalent, falls **jede** zu  $F$  passende Struktur auch zu  $G$  passt und umgekehrt. D.h.  $F \models G$  und  $G \models F$

**Beispiel: Semantische Äquivalenz:**  $\neg \forall x F \equiv \exists x \neg F$

**Erfüllbarkeitsäquivalenz**

Zwei Formeln  $F$  und  $G$  sind genau dann **erfüllbarkeitsäquivalent**, wenn gilt:

$$F \text{ ist erfüllbar} \leftrightarrow G \text{ ist erfüllbar}$$

oder umgekehrt:

$$F \text{ ist unerfüllbar} \leftrightarrow G \text{ ist unerfüllbar}$$

Die beiden Formeln brauchen nicht äquivalent zu sein und brauchen auch nicht durch die selben Interpretationen erfüllt zu werden. Es ist somit eine recht schwache Eigenschaft.

**6.4 Freie und gebundene Variablen**

**Freie Variable:** Ein Vorkommen einer Variable  $x$  in einer Formel  $A$  heisst **gebunden**, falls es im Bereich eines Quantors  $\exists x$  oder  $\forall x$  liegt, sonst heisst das Vorkommen **frei**.

**Satz:** Formel die keine freie Variable enthält.

**Umbenennung gebundener Variablen:**

Formeln, die gleich sind bis auf die Namen der gebundenen Variablen, werden identifiziert.

**Beispiel: Umbenennung gebundener Variablen**

$$\forall x(P(x) \wedge Q(z)) = \forall t(P(t) \wedge Q(z))$$

$$\forall x(P(x) \vee Q(y)) \neq \forall t(P(t) \vee Q(z))$$

Mit  $A \frac{t}{x}$  bezeichnet man die Formel, die man aus  $A$  erhält, indem man alle freien Vorkommen von  $x$  durch den Term  $t$  ersetzt.

- Es können nur freie Variablen ersetzt werden.
- Gebundene Variablen müssen umbenannt werden, falls es zu einer Kollision kommt.

**Beispiel: Freie Variablen ersetzen**

$$A = \forall y \exists z(R(x, z) \wedge Q(y, z))$$

Gesucht:  $A \frac{f(y)}{x}$

1. Gebundene Variablen umbenennen:  $A = \forall t \exists z(R(x, z) \wedge Q(t, z))$
2. Freie Variablen ersetzen:  $A = \forall t \exists z(R(f(y), z) \wedge Q(t, z))$

**6.5 Begriffe**

**Modell:** Eine Struktur  $\mathbb{A}$  ist Modell für  $F$ , falls  $\mathbb{A}(F) = 1$

$$\mathbb{A} \models F$$

$F$  ist **gültig**, falls jede zu  $F$  passende Struktur ein Modell für  $F$  ist.

$$\models F$$

$F$  ist **erfüllbar**, falls es mindestens ein Modell für  $F$  gibt.

Zwei prädikatenlogische Formeln  $F$  und  $G$  sind **äquivalent**, falls für alle sowohl zu  $F$  als auch zu  $G$  passende Strukturen  $\mathbb{A}$  gilt:

$$\mathbb{A}(F) = \mathbb{A}(G)$$

**Wie kann ich zeigen, dass eine Formel  $F$  gültig ist? Wie kann ich zeigen, dass zwei Formeln  $F$  und  $G$  semantisch äquivalent sind?**

1. **Prädikate durch Aussagen ersetzen**

$$F = (Q(a) \vee \neg R(f(b), c)) \wedge P(a, b)$$

$$Q(a) \leftrightarrow A_1$$

$$R(f(b), c) \leftrightarrow A_2$$

$$P(a, b) \leftrightarrow A_3$$

2. **Gleichheitsregel**

$$t_1 = t'_1 \wedge t_2 = t'_2 \wedge \dots \wedge t_n = t'_n \rightarrow f(t_1, \dots, t_n) = f(t'_1, \dots, t'_n)$$

$$t_1 = t'_1 \wedge t_2 = t'_2 \wedge \dots \wedge t_n = t'_n \rightarrow P(t_1, \dots, t_n) = P(t'_1, \dots, t'_n)$$

3. **Ersetzungsregel**

Falls eine Formel F Teilformel von H ist und  $F \leftrightarrow G$ , dann kann F in H durch G ersetzt werden:

$$H = (\underbrace{Q(a) \vee \neg R(f(b), c)}_F) \wedge P(a, b)$$

$$H' = (\underbrace{R(f(b), c) \rightarrow Q(a)}_G) \wedge P(a, b)$$

4. **Quantorenregeln**

*Negation von Quantoren:*

$$\neg \forall x P(x) \leftrightarrow \exists x \neg P(x)$$

$$\neg \exists x P(x) \leftrightarrow \forall x \neg P(x)$$

$$\neg \forall x (P(x) \rightarrow Q(x)) \leftrightarrow \exists x (P(x) \wedge \neg Q(x))$$

$$\neg \exists x (P(x) \wedge Q(x)) \leftrightarrow \forall x (P(x) \rightarrow \neg Q(x))$$

*Vertauschen von gleichen Quantoren:*

$$\forall x \forall y R(x, y) \leftrightarrow \forall y \forall x R(x, y)$$

$$\exists x \exists y R(x, y) \leftrightarrow \exists y \exists x R(x, y)$$

*Hineinziehen von Quantoren:*

$$\forall x (P(x) \wedge Q(x)) \leftrightarrow \forall x P(x) \wedge \forall x Q(x)$$

$$\exists x (P(x) \vee Q(x)) \leftrightarrow \exists x P(x) \vee \exists x Q(x)$$

*Nullquantifikation*

Regeln gelten nur, falls x in A keine freie Variable ist.

$$\forall x A \leftrightarrow A$$

$$\exists x A \leftrightarrow A$$

$$\forall x (A \vee P(x)) \leftrightarrow A \vee \forall x P(x)$$

$$\exists x (A \wedge P(x)) \leftrightarrow A \wedge \exists x P(x)$$

*Einseitige Regeln:*

$$\forall x P(x) \rightarrow \exists x P(x)$$

$$\exists x \forall y R(x, y) \rightarrow \forall y \exists x R(x, y)$$

$$\forall x P(x) \vee \forall x Q(x) \rightarrow \forall x (P(x) \vee Q(x))$$

$$\exists x (P(x) \wedge Q(x)) \rightarrow \exists x P(x) \wedge \exists x Q(x)$$

6.6 **Normalformen**

**Normalisierung**

1. **Bereinigte Form**

Keine Variablen, die frei und gebunden vorkommen.

2. **Pränexform**

Alle Quantoren vor der Formeln, gegebenenfalls umbenennen

3. Freie Variablen mit Existenzquantoren  $\exists$  binden, die an den Anfang der Formel gestellt werden

4. **Skolemform**

Nur noch Allquantoren  $\forall$ , keine Existenzquantoren  $\exists$ . Ersetzen von jeder durch  $\exists$  gebundene Variable durch eine Funktion  $f(y_1, \dots, y_n)$ , abhängig von den vorangehenden durch  $\forall$  gebundenen Variablen  $y_1, \dots, y_n$ , bzw. durch eine Konstante c.

5. **Matrix der Formel**

(=Formel ohne Quantoren) in KNF umwandeln

6. **Klauselmenge**

z.B.  $\{\{P(x, y)\}, \{P(y, z)\}, \{P(x, z)\}\}$

**Beispiel: Normalisierung**

**Gegeben:**

$$\neg(\exists y(P(x, y) \rightarrow \exists x(Q(y, x))) \vee \forall s(R(s, y)))$$

**Bereinigen:** Gebundene Variablen umbenennen, so dass keine Variable mehr gleichzeitig frei und gebunden vorkommt. [y / u, x / v]:

$$\neg(\exists u(P(x, u) \rightarrow \exists v(Q(y, v))) \vee \forall s(R(s, y)))$$

**Pränexform:** Die Quantoren können an den Anfang der Formel genommen werden ( $\rightarrow$  muss übersetzt werden).  $\neg$  werden nach innen geschoben durch Austauschen von  $\exists$  mit  $\forall$ .

$$\exists u \forall v \forall s ((P(x, u) \wedge \neg Q(y, v)) \vee R(s, y))$$

**Skolemform:** Wir machen u zur Konstante (bzw. null-stelligen Funktion).

$$\forall v \forall x ((P(x, c) \wedge \neg Q(y, v)) \vee R(s, y))$$

**Bereinigte Form:** Es gibt keine Variable, die in der Formel sowohl frei, als auch gebunden vorkommt. Hinter allen vorkommenden Quantoren stehen verschiedene Variablen.

$$F = \forall x \exists y (P(x, f(y)) \wedge \forall y (Q(x, y) \vee R(x)))$$

↓ umbenennen

$$F = \forall t \exists r (P(t, f(r)) \wedge \forall y (Q(x, y) \vee R(x)))$$

Eine Formel heisst in **bereinigter Pränexform**, falls sie die folgende Bauart hat:  $Q_1 y_1 Q_2 y_2 \dots Q_n y_n F$ .  $Q_i \in \{\forall, \exists\}$  und  $y_i$  sind Variablen. In F kommt kein Quantor vor.

$$F = \forall t \exists r (P(t, f(r)) \wedge \forall y (Q(x, y) \vee R(x)))$$

↓ Nullquantifikation

$$F = \forall t \forall y \exists r (P(t, f(r)) \wedge (Q(x, y) \vee R(x)))$$

Eine Formel heisst **Skolemform**, falls sie in bereinigter Pränexform ist und keine Existenzquantoren  $\exists$  enthält.

$$F = \forall t \forall y \exists r (P(t, f(r)) \wedge (Q(x, y) \vee R(x)))$$

↓

$$F = \forall t \forall y (P(t, f(g(t))) \wedge (Q(x, y) \vee R(x)))$$

6.7 **Unifikation**

Methode zur Vereinheitlichung prädikatenlogischer Ausdrücke.

**Beispiel: Unifikation**

$$\{h(g(X), Y, c), h(Y, g(b), c), h(Z, Z, U)\}$$

$$\text{sub} = [X \setminus b, Y \setminus g(b), Z \setminus g(b), U \setminus c]$$

sub heisst **kleinster gemeinsamer Unifikator**

**Mehrere Auftreten einer Variable:**

- innerhalb einer Klausel: gleiche Substitution
- in verschiedenen Klauseln: eine davon umbenennen  $\rightarrow$  verschiedene Substitutionen

**Beispiel: Unifikation**

$$F = \{\{P(x, a)\}, \{\neg P(b, y)\}\} \equiv \forall x \forall y (P(x, a) \wedge \neg P(b, y))$$

 sub:  $[x \setminus b] [y \setminus a]$ 
**6.8 Resolution der Prädikatenlogik****1. Variablen umbenennen**

Die Variablen sind so umzubennenen, dass keine in mehreren Formeln vorkommt

**2. Resolvieren**

Durchzuführen wie eine normale Resolution, nur kann bei jedem Schritt eine Substitution zur Unifikation vorgenommen werden.

**7 Prolog**

**Term:** Prolog's single data type is the term. Terms are either atoms, numbers, variables or compound terms.

**Atom:** Atoms can be regarded as a special case of compound terms (of arity zero). Examples are: `x`, `blue`.

**Numbers:** In addition to floats and integers which are prescribed by the Prolog ISO standard, many Prolog implementations also provide rational numbers and unbounded integers.

**Variable:** A variable is a string of upper-case letters, lower-case letters, digits and underscore characters that starts either with an upper-case letter or with underscore. A single underscore (`_`) means any term.

**Compound terms:** A compound term has a functor and a number of arguments, which are again terms. The number of arguments is called the term's arity. Examples for terms are `f(a,b)` and `p(x,y,z)`. Some operators can be used in infix notation. For example, the terms `+(a,b)` and `=(X,Y)` can also be written as `a+b` and `X=Y`, respectively. Users can also declare arbitrary sequences of symbols as new infix or postfix operators to allow for domain-specific notations.

**Rule:** `Head :- Body.` is read "Head is true if Body is true".

**Facts:** Clauses with empty bodies are called facts: `cat(tom).`, which is equivalent to the rule `cat(tom) :- true.`

**Query:** `"Is tom a cat?"` can be written as `?- cat(tom).` and, with the fact above, returns `"Yes"`.

`"What things are cats?"`

```
?- cat(X).\
```

**7.1 Beispiel**

Ein Arithmetik-Beispiel aus der Vorlesung.

```
add(a,X,X).
add(f(X),Y,f(Z)) :- add(X,Y,Z).
mult(a,X,a).
mult(f(X),Y,U) :- mult(X,Y,Z), add(Z,Y,U).
exp(X,a,f(a)).
exp(X,f(Y),V) :- exp(X,Y,Z), mult(Z,X,V).
```

**Bedeutung**

`add(a,X,X)` bedeutet, dass  $0+x=x$ . Die Regel bedeutet: um  $(x+1)+y$  zu erhalten, kann ich  $x+y$  ausrechnen und noch eins dazuzählen. `a` bedeutet also 0.

`mult(a,X,a)` bedeutet, dass  $0*x=0$ . Die Regel bedeutet: um  $((x+1)*y)$  zu erhalten, kann ich  $x*y$  ausrechnen und noch  $y$  addieren.

Beispiel:  $3 * 5 = 2 * 5 + 5 = 5 * 5 + 5 + 5$

`exp(X,a,f(a))` bedeutet, dass  $x$  hoch 0 immer 1 ergibt (den Nachfolger von 0). Die Regel bedeutet: um  $x^{(y+1)}$  zu erhalten, muss ich  $x^y$  ausrechnen und mit  $x$  multiplizieren.

Beispiel:  $5^3 = 5 * 5^2 = 5 * 5 * 5^1 = 5 * 5 * 5 * 1$