

---

---

# Logik

Prof. Dr. Madlener

TU Kaiserslautern

SS 2009

---

---

# Logik

Studiengang „Informatik“, „Technoinformatik“ und „ WiWi/Inf“  
SS'09

Prof. Dr. Madlener TU - Kaiserslautern

Vorlesung:

Mi 11.45-13.15 52/207

- ▶ Informationen  
<http://www-madlener.informatik.uni-kl.de/teaching/ss2009/logik/logik.html>
- ▶ Grundlage der Vorlesung: Skript  
Einführung in die Logik und Korrektheit von Programmen.

- 
- 
- ▶ Bewertungsverfahren:  
Zulassungsvoraussetzungen zu Abschlussklausur:  
Übungen: mind. 50 %  
Aufsichtsarbeit: mind. 50 %
  - ▶ Abschlussklausur: Montag, 2009/08/10
  - ▶ Übungen: Gruppen  
Einschreiben, Sprechzeiten siehe Homepage

---

---

## Grundlagen der Aussagenlogik

Syntax

Semantik

Deduktiver Aufbau der Aussagenlogik

Natürliche Kalküle

## Algorithmischer Aufbau der Aussagenlogik

Semantische Tableaux

Normalformen

Davis-Putman-Algorithmen

Resolutions-Verfahren

---

---

## Grundlagen der Prädikatenlogik

Beziehungen zwischen Eigenschaften von Elementen

Semantik der P-Logik 2-Stufe – Interpretationen, Belegungen, Bewert

Transformationen von Termen und Formeln

## Entscheidbarkeit in der Prädikatenlogik

Unentscheidbarkeit der Allgemeingültigkeit

Hauptsätze der Prädikatenlogik erster Stufe

Theorien erster Stufe

## Algorithmen der Prädikatenlogik

Aufzählungsverfahren für PL-1

Resolventenmethode – (Allg. Resolutionsverfahren)

Logisches Programmieren und Prolog

---

---

# Einleitung

## Methoden zur Lösung von Problemen mit Hilfe von Rechnern Formalisierung ( $\equiv$ Festlegung)

- ▶ **Logik::** „Lehre vom folgenrichtigen Schließen“ bzw. „Lehre von formalen Beziehungen zwischen Denkinhalten“

**Zentrale Fragen:** Wahrheit und Beweisbarkeit von Aussagen  $\rightsquigarrow$   
**Mathematische Logik.**

- ▶ **Logik in der Informatik:**
  - ▶ **Aussagenlogik:** Boolesche Algebra. Logische Schaltkreise (Kontrollsystemen), Schaltungen, Optimierung.
  - ▶ **Prädikatenlogik:** Spezifikation und Verifikation von Softwaresystemen.
  - ▶ **Modal- und Temporallogik:** Spezifikation und Verifikation reaktiver Systeme.

---

---

# Logik in der Informatik

1. Semantik von Programmiersprachen (Hoarscher Kalkül).
2. Spezifikation von funktionalen Eigenschaften.
3. Verifikationsprozess bei der SW-Entwicklung.  
Beweise von Programmeigenschaften.
4. Spezielle Programmiersprachen (z.B. PROLOG)

► **Automatisierung des logischen Schließens**

1. Automatisches Beweisen (Verfahren,...)
2. Grundlagen von Informationssystemen (Verarbeitung von Wissen, Reasoning,...)

---

---

## Voraussetzungen

1. **Mathematische Grundlagen.** Mengen, Relationen, Funktionen. Übliche Formalisierungen: „Mathematische Beweise“, Mathematische Sprache, d.h. Gebrauch und Bedeutung der üblichen Operatoren der naiven Logik. Also Bedeutung von **nicht, und, oder, impliziert, äquivalent, es gibt, für alle.**
2. **Grundlagen zur Beschreibung formaler Sprachen.** Grammatiken oder allgemeiner **Kalküle** (Objektmenge und Regeln zur Erzeugung neuer Objekte aus bereits konstruierter Objekte), Erzeugung von Mengen, Relationen und Funktionen, Hüllenoperatoren (Abschluss von Mengen bzgl. Relationen).
3. **Vorstellung von Berechenbarkeit**, d.h. entscheidbare und rek.aufzählbare Mengen, Existenz nicht entscheidbarer Mengen und nicht berechenbarer Funktionen.

---

---

# Berechnungsmodelle/Programmiersprachen

Algorithmische Unlösbarkeit?

prinzipielle Lösbarkeit



effiziente Lösbarkeit



algorithmischer Entwurf



$P$ : Programm in einer HPS



Problem  
Spezifikation

(Formalisiert)

---

---

# Syntaktische und semantische Verifikation von $P$ .

- ▶ **Syntaxanalyse**  
Sprachen Chomski-Hierarchie  
Kontext freie Sprachen  
Grammatiken/Erzeugungsprozess
  
- ▶ **Programmverifikation**  
Tut  $P$  auch was erwartet wird.  
Gilt  $P \rightsquigarrow$  Problem Spezifikation

---

---

## Typische Ausdrücke

- ▶  $(x + 1)(y - 2)/5$  Terme als Bezeichner von Objekten.
- ▶  $3 + 2 = 5$  Gleichungen als spezielle Formeln.
- ▶ „29 ist (k)eine Primzahl “ Aussagen.
- ▶ „ $3 + 2 = 5$  und 29 ist keine Primzahl “ Aussage.
- ▶ „wenn 29 keine Primzahl ist, dann ist  $0 = 1$  “ Aussage.
- ▶ „jede gerade Zahl, die größer als 2 ist, ist die Summe zweier Primzahlen “ Aussage.
- ▶  $2 \leq x$  und  $(\forall y \in \mathbb{N})$   
 $((2 \leq y$  und  $y + 1 \leq x) \rightarrow$  nicht  $(\exists z \in \mathbb{N})y * z = x)$   
Aussage.

---

---

## Typische Ausdrücke (Fort.)

- ▶  $(\forall X \subseteq \mathbb{N})(0 \in X \wedge (\forall x \in \mathbb{N})(x \in X \rightarrow x + 1 \in X) \rightarrow X = \mathbb{N})$   
**Induktionsprinzip.**
- ▶  $(\forall X \subseteq \mathbb{N})(X \neq \emptyset \rightarrow X \text{ hat ein kleinstes Element})$   
Jede nichtleere Menge natürlicher Zahlen enthält ein minimales Element.

**Zweiwertige Logik** Jede Aussage ist entweder **wahr** oder **falsch**.

- ▶ Es gibt auch andere Möglichkeiten (Mehrwertige Logik).
- ▶ Prädikatenlogik erster Stufe (PL1): Nur Eigenschaften von Elementen und Quantifizierung von Elementvariablen erlaubt.



# Kapitel I

## Grundlagen der Aussagenlogik













# Belegungen und Bewertungen (Fort.)

- ▶ Sprechweise:  $A$  ist „falsch“ unter  $\varphi$ , falls  $\varphi(A) = 0$   
 $A$  ist „wahr“ unter  $\varphi$  oder  $\varphi$  „erfüllt“  $A$ , falls  $\varphi(A) = 1$ .
- ▶ Darstellung von Bewertungen durch **Wahrheitstafeln**:

$A$	$\neg A$
1	0
0	1

$A$	$B$	$A \vee B$	$A \wedge B$	$A \rightarrow B$	$A \leftrightarrow B$
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	0
1	1	1	1	1	1







# Bewertungen

$p$	$q$	$r$	$q \rightarrow r$	$p \wedge q$	$p \rightarrow (q \rightarrow r)$	$(p \wedge q) \rightarrow r$	$A$
0	0	0	1	0	1	1	1
0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1
0	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1
1	0	1	1	0	1	1	1
1	1	0	0	1	0	0	1
1	1	1	1	1	1	1	1

$A$  ist „wahr“ unabhängig von den Werten von  $p, q, r$ , d.h. für jede Bewertung  $\varphi$ . Weitere solche Formeln sind etwa:

$(A \rightarrow (B \rightarrow A)), (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$  oder  $((\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A))$ .







## Bemerkung (Fort.)

- 2.(a) Sei  $\Sigma = \{p\}$  und  $A = p \vee q$ . Dann gilt  $\Sigma \models A$ , denn falls  $\varphi(p) = 1$ , dann auch  $\varphi(p \vee q) = 1$ . Jede Bewertung, die  $\Sigma$  erfüllt, erfüllt also auch  $A$ .
- (b) Ist  $\Sigma = \emptyset$ , dann gilt  $\Sigma \models A$  genau dann, wenn  $A$  Tautologie ist, d.h.  $\text{Folg}(\emptyset) = \text{Taut}$ .
- (c) Ist  $\Sigma$  nicht erfüllbar, dann gilt  $\Sigma \models A$  für alle  $A \in F$ , d.h.  $\text{Folg}(\Sigma) = F$ . Insbesondere  $\Sigma \models A, \neg A$  für ein  $A$ .
- (d) Sei  $\Sigma \subseteq \Sigma'$ . Ist  $\Sigma'$  erfüllbar, dann ist auch  $\Sigma$  erfüllbar.
- (e) Es gilt  $\Sigma \subseteq \text{Folg}(\Sigma)$  und  $\text{Folg}(\text{Folg}(\Sigma)) = \text{Folg}(\Sigma)$ .
- (f) Falls  $\Sigma \subseteq \Sigma'$ , dann gilt  $\text{Folg}(\Sigma) \subseteq \text{Folg}(\Sigma')$ .
3.  $\Sigma \models A$  gilt genau dann, wenn  $\Sigma \cup \{\neg A\}$  nicht erfüllbar. Ist  $\Sigma$  endlich, dann ist es entscheidbar, ob  $\Sigma$  erfüllbar ist, und die Menge  $\text{Folg}(\Sigma)$  ist entscheidbar.

# Deduktionstheorem und Modus-Ponens Regel

## Lemma 1.9

a) **Deduktionstheorem:**

$$\Sigma, A \models B \quad \text{gdw} \quad \Sigma \models (A \rightarrow B).$$

( $\Sigma, A$  ist Kurzschreibweise für  $\Sigma \cup \{A\}$ )

b) **Modus-Ponens-Regel:**

Es gilt  $\{A, A \rightarrow B\} \models B$ .

*Insbesondere ist  $B$  eine Tautologie, falls  $A$  und  $(A \rightarrow B)$  Tautologien sind.*





## Anwendungen Kompaktheitssatz

### Beispiel 1.11

Sei  $\Sigma \subseteq F$ . Gibt es zu jeder Bewertung  $\varphi$  ein  $A \in \Sigma$  mit  $\varphi(A) = 1$ , so gibt es  $A_1, \dots, A_n \in \Sigma$  ( $n > 0$ ) mit  $\models A_1 \vee \dots \vee A_n$ .

- Betrachte die Menge  $\Sigma' = \{\neg A \mid A \in \Sigma\}$ , nach Voraussetzung ist sie unerfüllbar. Also gibt es eine endliche nichtleere Teilmenge  $\{\neg A_1, \dots, \neg A_n\}$  von  $\Sigma'$  die unerfüllbar ist. Also gilt für jede Bewertung  $\varphi$  gibt es ein  $i$  mit  $\varphi(\neg A_i) = 0$  oder  $\varphi(A_i) = 1$  und somit  $\varphi(A_1 \vee \dots \vee A_n) = 1$ .

- Der zweite Teil des Satzes ist die Grundlage für Beweisverfahren für  $\Sigma \models A$ . Dies ist der Fall wenn  $\Sigma \cup \{\neg A\}$  unerfüllbar ist.

Widerspruchbeweise versuchen systematisch eine endliche Menge  $\Sigma_0 \subset \Sigma$  zu finden, so dass  $\Sigma_0 \cup \{\neg A\}$  unerfüllbar ist.

# Logische Äquivalenz

## Definition 1.12 (Logische Äquivalenz)

Seien  $A, B \in F$  heißen **logisch äquivalent** mit der Schreibweise  $A \models\equiv B$ , falls für jede Bewertung  $\varphi$  gilt:  $\varphi(A) = \varphi(B)$ .

Insbesondere ist dann  $A \models B$  und  $B \models A$ .

► Einige Beispiele für logisch äquivalente Formeln:

1.  $A \models\equiv \neg(\neg A)$ ,  $A \models\equiv A \vee A$ ,  $A \models\equiv A \wedge A$
2.  $A \wedge B \models\equiv B \wedge A$  und  $A \vee B \models\equiv B \vee A$ ,
3.  $A \wedge (B \wedge C) \models\equiv (A \wedge B) \wedge C$  und  $A \vee (B \vee C) \models\equiv (A \vee B) \vee C$ ,
4.  $A \wedge (B \vee C) \models\equiv (A \wedge B) \vee (A \wedge C)$  und  
 $A \vee (B \wedge C) \models\equiv (A \vee B) \wedge (A \vee C)$       **(Distributiv)**



## Logische Äquivalenz (Fort.)

► Folgende Aussagen sind äquivalent:

- $\models (A \leftrightarrow B)$
- $A \models B$  und  $B \models A$
- $A \models\!\!\!\not\equiv B$
- $\text{Folg}(A) = \text{Folg}(B)$

### Folgerung 1.13

Zu jedem  $A \in F$  gibt es  $B, C, D \in F$  mit

1.  $A \models\!\!\!\not\equiv B$ ,  $B$  enthält nur  $\rightarrow$  und  $\neg$  als log. Verknüpfungen
2.  $A \models\!\!\!\not\equiv C$ ,  $C$  enthält nur  $\wedge$  und  $\neg$  als log. Verknüpfungen
3.  $A \models\!\!\!\not\equiv D$ ,  $D$  enthält nur  $\vee$  und  $\neg$  als log. Verknüpfungen

## Logische Äquivalenz (Fort.)

### Definition 1.14 (Vollständige Operatoren Mengen)

Eine Menge  $OP \subseteq \{\neg, \vee, \wedge, \rightarrow, \leftrightarrow, \dots\}$  heißt **vollständig**, falls es zu jedem  $A \in F$  eine logisch äquivalente  $A$ -Form  $B \in F(OP)$  gibt.

- ▶ Vollständige Operatoren Mengen für die Aussagenlogik sind z.B.:  
 $\{\neg, \rightarrow\}, \{\neg, \vee\}, \{\neg, \wedge\}, \{\neg, \vee, \wedge\}, \{\text{false}, \rightarrow\}$
- ▶ Dabei ist `false` eine Konstante, mit  $\varphi(\text{false}) = 0$  für jede Bewertung  $\varphi$ . Offenbar gilt  $\neg A \models (A \rightarrow \text{false})$ .
- ▶ **Normalformen**:: DNF (Disjunktive Normalform), KNF (Konjunktive Normalform), KDNF, KKNF (Kanonische Formen).

## Boolsche Funktionen

Jede Aussageform  $A(p_1, \dots, p_n)$  stellt in natürlicher Form eine Boolesche Funktion  $f_A : \mathbb{B}^n \rightarrow \mathbb{B}$  dar. Nämlich durch die Festlegung  $f_A(b_1, \dots, b_n) = \varphi_{\vec{b}}(A)$  mit der Bewertung  $\varphi_{\vec{b}}(p_i) = b_i$

- ▶ Man kann leicht durch Induktion nach  $n$  zeigen, dass jede Boolesche Funktion  $f : \mathbb{B}^n \rightarrow \mathbb{B}$  ( $n > 0$ ) sich in obiger Form durch eine Aussageform in  $p_1, \dots, p_n$  und einer vollständigen Operatorenmenge darstellen lässt.
- ▶ Die Boolesche Algebra über  $\mathbb{B}$  hat als übliche Operatormenge **true, false, not, or, and** mit der standard Interpretation.
- ▶ Für andere Operatormengen die etwa **nand, nor** enthalten, siehe Digitale Logik  
(Gatter: Ein- Ausgabesignale, Verzögerung. **nand, nor** Gattern werden bevorzugt, da nur zwei Transistoren nötig).

## Beispiel Patientenüberwachungssystem

**Beispiel** *Patientenüberwachungssystem* erhält gewisse Daten über den Zustand eines Patienten. Z.B. Temperatur, Blutdruck, Pulsrate. Die Schwellenwerte für die Daten seien etwa wie folgt festgelegt:

Zustände	
Ein/ Ausgaben	Bedeutung
A	Temperatur außerhalb 36-39°C.
B	Blutdruck außerhalb 80-160 mm.
C	Pulsrate außerhalb 60-120 Schläge pro min.
O	Alarmaktivierung ist notwendig.

Die Anforderungen, d.h. bei welchen Kombinationen der Werte der Zustände eine Alarmaktivierung notwendig ist, werden durch den Medizin-Experten festgelegt. Sie seien in folgender Tabelle fixiert.

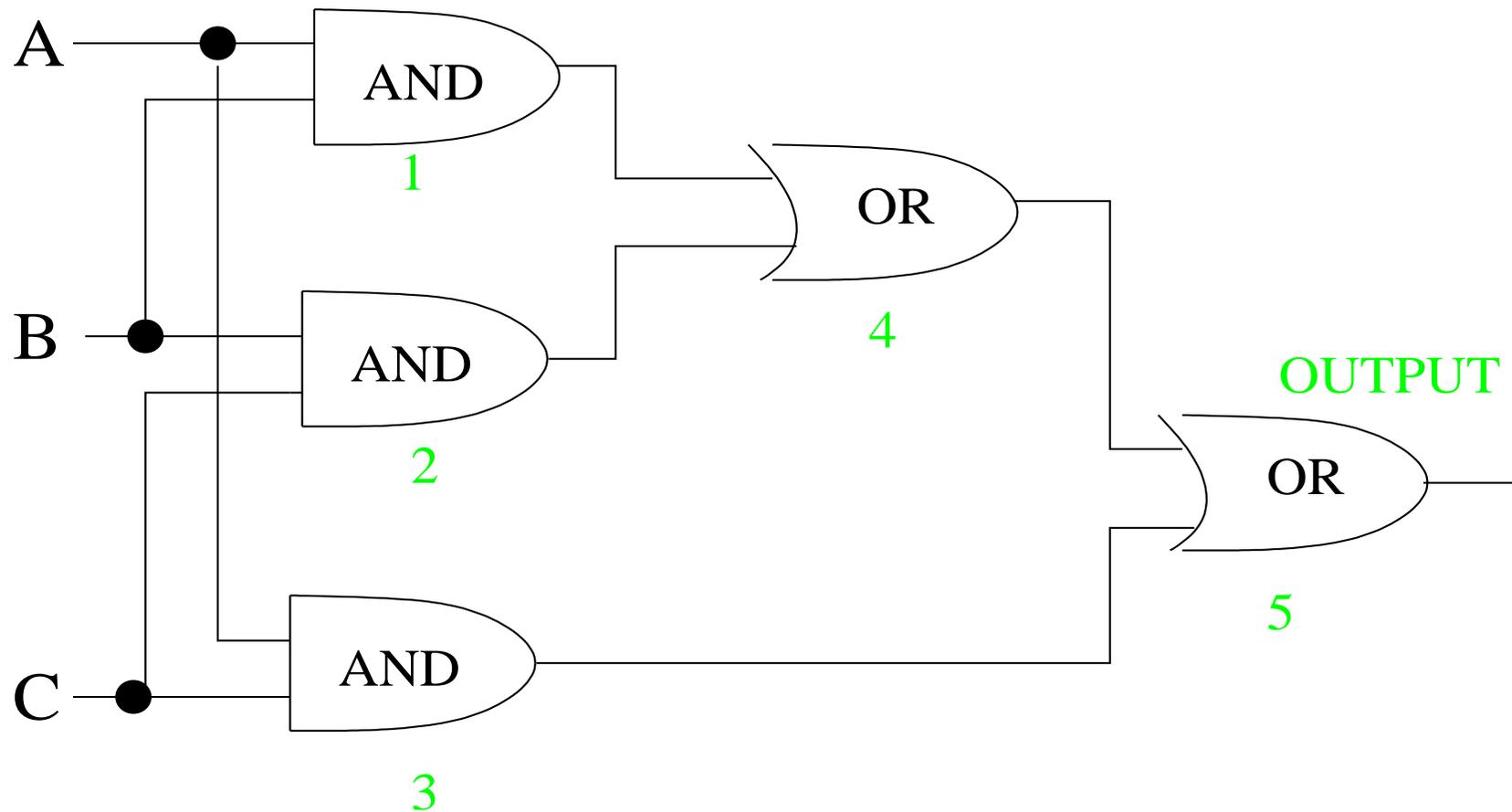
**I/O - Tabelle**

A	B	C	O
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

**Logischer Entwurf:** Betrachte die Zeilen in denen O den Wert 1 hat und stelle eine KDNF auf (Disjunktion von Konjunktionen von Literalen, wobei ein Literal eine atomare Form oder die Negation einer solchen ist).  
 $(\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C)$

Als eine Realisierung könnte man das folgende Schaltnetz nehmen:

INPUTS



# Deduktiver Aufbau der Aussagenlogik

Dieser Abschnitt beschäftigt sich mit einem axiomatischen Aufbau der Aussagenlogik mittels eines **“Deduktiven Systems”** oder eines **„Kalküls“**.

Eine syntaktisch korrekte Formel in einem Deduktiven System wird **“Theorem”** genannt, wenn sie durch rein mechanische Anwendungen der Regeln des Systems aus den Axiomen des Systems **“abgeleitet”** werden kann.

Man kann mehrere deduktive Systeme angeben, in denen aussagenlogische Formeln genau dann Theoreme sind, wenn sie auch Tautologien sind.

# Deduktive Systeme-Kalküle

## Definition 1.15 (Deduktives System)

Ein **Deduktives System**  $\mathcal{F} = \mathcal{F}(Ax, R)$  besteht aus

- ▶ einem Alphabet  $\Delta$  (hier  $\Delta = V \cup K \cup \{\rightarrow, \neg\}$ ),
- ▶  $F \subseteq \Delta^*$ , einer Menge von (wohldefinierten) Formeln (hier die Aussageformen),
- ▶  $Ax \subseteq F$ , einer Menge von Axiomen und
- ▶  $R$ , einer Menge von Regeln der Form  $\frac{A_1, \dots, A_n}{A}$  ( $n \in \mathbb{N}^+$ ).  
( $A_1, \dots, A_n, A \in F$ )

Die Mengen  $F$ ,  $Ax$  und  $R$  sind im allgemeinen rekursiv entscheidbar.

## Deduktive Systeme-Kalküle

- ▶ Die Menge  $T = T(\mathcal{F})$  der **Theoreme** ist definiert durch:
  1.  $Ax \subseteq T$  (d.h. alle Axiome sind Theoreme)
  2. Sind  $A_1, \dots, A_n \in T$  und ist die Regel  $\frac{A_1, \dots, A_n}{A}$  in  $R$ , dann ist  $A \in T$ .
  3.  $T$  ist die kleinste Menge von Formeln, die (1) und (2) erfüllt.
- ▶ Man schreibt für  $A \in T(\mathcal{F})$  auch  $\vdash_{\mathcal{F}} A$  oder einfach  $\vdash A$  und sagt “ $A$  ist in  $\mathcal{F}$  herleitbar”.
- ▶ **Deduktiver Folgerungsbegriff:** Sei  $\Sigma \subseteq F$ ,  $A \in F$ , dann bedeutet  $\Sigma \vdash_{\mathcal{F}(Ax, R)} A$  nichts anderes als  $\vdash_{\mathcal{F}(Ax \cup \Sigma, R)} A$ . Sprechweise: “ $A$  ist in  $\mathcal{F}$  aus  $\Sigma$  herleitbar”.
- ▶ Sind  $\mathcal{F}_1$  und  $\mathcal{F}_2$  deduktive Systeme über der Formelmenge  $F$  und gilt  $T(\mathcal{F}_1) = T(\mathcal{F}_2)$  so nennt man sie **äquivalent**.



## Bemerkung (Fort.)

2. Die Menge  $T$  der Theoreme ist *rekursiv aufzählbar* (denn  $Ax$  und  $R$  sind rekursiv). Die Menge der Beweise

$$\text{Bew} := \{B_1 \star B_2 \star \dots \star B_n \mid B_1, \dots, B_n \text{ ist Beweis}\}$$

ist rekursiv. (Siehe Argumentation von  $L(G)$  ist rekursiv aufzählbar für Grammatiken  $G$ ).

- ▶ Ist  $\Sigma$  rekursiv entscheidbar, so gelten obige Aussagen entsprechend. Insbesondere ist  $\text{Fol}_{\mathcal{F}}(\Sigma) = \{A \mid \Sigma \vdash_{\mathcal{F}(Ax,R)} A\}$  rekursiv aufzählbar.
- ▶ **Beachte:** Beweise sind im allgemeinen nicht eindeutig. Es wird im allgemeinen nicht verlangt, dass  $T$  von  $R$  *frei* erzeugt wird.

## Bemerkung (Fort.)

3. Gibt es ein deduktives System  $\mathcal{F}_0$ , so dass  $\vdash_{\mathcal{F}_0} A$  genau dann gilt, wenn  $\models A$  gilt?

- Hierzu werden  $Ax$  und  $R$  häufig endlich beschrieben durch **Schemata**.

Beispielsweise beschreibt das Axiom  $(A \rightarrow (B \rightarrow A))$  die Menge

$\{A_0 \mid \text{es gibt } A, B \in F \text{ mit } A_0 \equiv (A \rightarrow (B \rightarrow A))\}$

und die Regel

$\frac{A, A \rightarrow B}{B}$  beschreibt die Menge von Regeln

$\left\{ \frac{A_0, A_1}{B_0} \mid \text{Es gibt } A, B \in F \text{ mit} \right.$   
 $\left. A_0 \equiv A, B_0 \equiv B \text{ und } A_1 \equiv A \rightarrow B \right\}$ .

## Das deduktive System $\mathcal{F}_0$

### Definition 1.17 (Das deduktive System $\mathcal{F}_0$ )

Das deduktive System  $\mathcal{F}_0$  für die Aussagenlogik besteht aus der Formelmengemenge  $F_0$  der Formeln in  $V, \neg, \rightarrow, ($  und  $)$ . Die Axiomenmenge  $Ax$  wird durch folgende Axiomenschemata beschrieben:

$$Ax1: A \rightarrow (B \rightarrow A)$$

$$Ax2: (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$$

$$Ax3: ((\neg A) \rightarrow (\neg B)) \rightarrow (B \rightarrow A)$$

Dabei beschreiben  $Ax1$ ,  $Ax2$  und  $Ax3$  disjunkte Formelmengen.

## Das deduktive System $\mathcal{F}_0$

Die Regelmengens  $R$  wird beschrieben durch das Regelschema

$$\text{MP: } \frac{A, (A \rightarrow B)}{B} \quad (\text{modus ponens}).$$

- ▶ Beachte: Ax und  $R$  sind rekursiv entscheidbar.
- ▶ Es genügt zunächst nur Axiome für Formeln in  $\rightarrow$  und  $\neg$  zu betrachten, da alle anderen Formeln zu einer Formel in  $\rightarrow$  und  $\neg$  logisch äquivalent sind.
- ▶ Die Menge der Theoreme von  $\mathcal{F}_0$  wird nicht frei erzeugt. Die Modus-Ponens-Regel ist hochgradig nicht eindeutig.  
 $\frac{A, A \rightarrow B}{B}$  und  $\frac{A', A' \rightarrow B}{B}$  sind beides Regeln mit gleicher Folgerung. Das erschwert sehr das Finden von Beweisen.

## Beispiel

### Beispiel 1.18

Für jedes  $A \in \mathcal{F}_0$  gilt  $\vdash (A \rightarrow A)$ , d.h.  $(A \rightarrow A) \in T(\mathcal{F}_0)$

Beweis:

$$\begin{array}{ll}
 B_0 \equiv (A \rightarrow ((A \rightarrow A) \rightarrow A)) \rightarrow & \\
 \quad ((A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A)) & \text{Ax2} \\
 B_1 \equiv A \rightarrow ((A \rightarrow A) \rightarrow A) & \text{Ax1} \\
 B_2 \equiv (A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A) & \text{MP}(B_0, B_1) \\
 B_3 \equiv A \rightarrow (A \rightarrow A) & \text{Ax1} \\
 B_4 \equiv A \rightarrow A & \text{MP}(B_2, B_3)
 \end{array}$$

■

- ▶ Wie findet man Beweise im System  $\mathcal{F}_0$ ?  
 Einziger Hinweis: Die Zielformel  $B$ , sofern sie kein Axiom ist, muss in der Form  $(A_1 \rightarrow (\dots(A_n \rightarrow B)\dots))$  vorkommen. Wähle geeignete  $A$ 's.
- ▶ **Beachte:** Alle Axiome sind Tautologien der Aussagenlogik. Da diese abgeschlossen gegenüber Modus Ponens sind, sind alle Theoreme von  $\mathcal{F}_0$  Tautologien. D.h.  $T(\mathcal{F}_0) \subseteq Taut(F_0)$ .
- ▶ Will man in ganz  $F$  Beweise führen, so muss man weitere Axiome einführen.  
 Z.B.  
 $Ax1\wedge : (A \wedge B) \rightarrow (\neg(A \rightarrow (\neg B)))$   
 $Ax2\wedge : (\neg(A \rightarrow (\neg B))) \rightarrow (A \wedge B)$

# Deduktiver Folgerungsbegriff

## Definition 1.19 (Axiomatischer Folgerungsbegriff)

Sei  $\Sigma \subseteq F_0, A \in F_0$ .

1.  $A$  ist aus  $\Sigma$  in  $\mathcal{F}_0$  **herleitbar**, wenn  $A$  sich aus  $Ax \cup \Sigma$  mit den Regeln aus  $R$  herleiten lässt, d.h.  $A$  ist Theorem im deduktiven System  $\mathcal{F}$  mit Axiomenmenge  $Ax \cup \Sigma$  und gleicher Regelmenge wie  $\mathcal{F}_0$ . Schreibweise  $\Sigma \vdash_{\mathcal{F}_0} A$ , einfacher  $\Sigma \vdash A$ .

$B_0, \dots, B_n$  ist ein **Beweis** für  $\Sigma \vdash A$ , falls  $A \equiv B_n$  und für alle  $0 \leq i \leq n$  gilt:  $B_i \in Ax \cup \Sigma$  oder es gibt  $j, k < i$  mit  $B_k \equiv (B_j \rightarrow B_i)$ .

2.  $\Sigma$  heißt **konsistent**, falls für keine Formel  $A \in F_0$  gilt  $\Sigma \vdash A$  und  $\Sigma \vdash \neg A$ .

Gibt es eine solche Formel, so heißt  $\Sigma$  **inkonsistent**.

## Folgerung 1.20 (Beweishilfsmittel)

1. Gilt  $\Sigma \vdash A$ , so folgt unmittelbar aus der Definition 1.19, dass es eine endliche Teilmenge  $\Sigma_0 \subseteq \Sigma$  gibt mit  $\Sigma_0 \vdash A$ .  
*Dies entspricht dem Kompaktheitssatz für " $\models$ ".*
2. Ist  $\Sigma$  inkonsistent, dann gibt es eine endliche Teilmenge  $\Sigma_0 \subseteq \Sigma$ , die inkonsistent ist  
(denn ist  $\Sigma \subseteq \Gamma$  und  $\Sigma \vdash A$ , dann gilt auch  $\Gamma \vdash A$ ).
3. Ist  $\Sigma \subseteq \Gamma$  so  $\text{Fol}_{\mathcal{F}_0}(\Sigma) \subseteq \text{Fol}_{\mathcal{F}_0}(\Gamma)$ .
4. Aus  $\Sigma \vdash A$  und  $\Gamma \vdash B$  für alle  $B \in \Sigma$  folgt  $\Gamma \vdash A$ .  
Ist also  $\Sigma \subseteq \text{Fol}_{\mathcal{F}_0}(\Gamma)$  so  $\text{Fol}_{\mathcal{F}_0}(\Sigma) \subseteq \text{Fol}_{\mathcal{F}_0}(\Gamma)$ .  
*Beweise lassen sich also zusammensetzen.*
5. Gilt  $\Sigma \vdash A$ , so ist  $\{\Sigma, \neg A\}$  inkonsistent.  
*Gilt auch die Umkehrung?*
6. Es gilt stets  $T(\mathcal{F}_0) \subseteq \text{Fol}_{\mathcal{F}_0}(\Sigma)$  für jede Menge  $\Sigma$ .

## Satz 1.21 (Deduktionstheorem)

Sei  $\Sigma \subseteq F_0$  und seien  $A, B \in F_0$ .

Dann gilt:  $\Sigma, A \vdash B$  gdw  $\Sigma \vdash (A \rightarrow B)$

**Beweis:**

„  $\Leftarrow$  “ Klar wegen MP-Regel.

„  $\Rightarrow$  “ Sei  $B_0, \dots, B_m$  ein Beweis für  $\Sigma, A \vdash B$ , d.h.  $B \equiv B_m$ .

**Beh.:** Für  $i = 0, \dots, m$  gilt  $\Sigma \vdash (A \rightarrow B_i)$

Induktion nach  $i$  und Fallunterscheidung, je nachdem ob  $B_i$  gleich  $A$  ist, in  $Ax \cup \Sigma$  liegt oder mit MP-Regel aus  $B_j, B_k$  mit  $j, k < i$  entsteht. ■

## Anwendungen des Deduktionstheorems

### Beispiel 1.22 (Beweistransformationen. Wiederverwendung von Beweisen. )

- ▶ Vereinbarungen zur Darstellung von Beweisen:  
 $B_1, \dots, B_n$  heißt **abgekürzter Beweis** für  $\Sigma \vdash B_n$ , falls für jedes  $j$  mit  $1 \leq j \leq n$  gilt:  $\Sigma \vdash B_j$  oder es gibt  $j_1, \dots, j_r < j$  mit  $B_{j_1}, \dots, B_{j_r} \vdash B_j$ .
  - ▶ Gibt es einen abgekürzten Beweis für  $\Sigma \vdash A$ , dann gibt es auch einen Beweis für  $\Sigma \vdash A$ .
1.  $\vdash (A \rightarrow A)$  folgt aus dem Deduktionstheorem, da  $A \vdash A$  gilt.
  2. Um  $A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$  zu zeigen, zeige  
 $A, A \rightarrow B, B \rightarrow C \vdash C$ .

## Anwendungen des Deduktionstheorems (Fort.)

3.  $\vdash (\neg\neg A \rightarrow A)$  dazu genügt es zu zeigen  
 $\neg\neg A \vdash A$

Beweis:

$B_1 \equiv \neg\neg A$	
$B_2 \equiv \neg\neg A \rightarrow (\neg\neg\neg\neg A \rightarrow \neg\neg A)$	Ax1
$B_3 \equiv \neg\neg\neg\neg A \rightarrow \neg\neg A$	MP
$B_4 \equiv (\neg\neg\neg\neg A \rightarrow \neg\neg A) \rightarrow (\neg A \rightarrow \neg\neg\neg A)$	Ax3 ■
$B_5 \equiv \neg A \rightarrow \neg\neg\neg A$	MP
$B_6 \equiv (\neg A \rightarrow \neg\neg\neg A) \rightarrow (\neg\neg A \rightarrow A)$	Ax3
$B_7 \equiv \neg\neg A \rightarrow A$	MP
$B_8 \equiv A$	MP

# Anwendungen des Deduktionstheorems (Fort.)

- 4.  $\vdash (A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C))$   
(zeige:  $A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$ )
- 5.  $\vdash (B \rightarrow ((B \rightarrow A) \rightarrow A))$
- 6.  $\vdash (\neg B \rightarrow (B \rightarrow A))$  (zu zeigen:  $\neg B, B \vdash A$ )

Beweis:

$B_1 \equiv \neg B$	Vor
$B_2 \equiv \neg B \rightarrow (\neg A \rightarrow \neg B)$	Ax1
$B_3 \equiv \neg A \rightarrow \neg B$	MP
$B_4 \equiv (\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$	Ax3
$B_5 \equiv B \rightarrow A$	MP
$B_6 \equiv B$	Vor
$B_7 \equiv A$	MP ■

7.  $\vdash B \rightarrow \neg\neg B$
8.  $\vdash ((A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A))$  und  
 $\vdash ((\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B))$
9.  $\vdash (B \rightarrow (\neg C \rightarrow \neg(B \rightarrow C)))$
10.  $\vdash ((B \rightarrow A) \rightarrow ((\neg B \rightarrow A) \rightarrow A))$
11.  $\vdash (A \rightarrow B) \rightarrow ((A \rightarrow \neg B) \rightarrow \neg A)$

**Frage:** Lassen sich alle Tautologien als Theoreme im System  $\mathcal{F}_0$  herleiten ?

# Vollständigkeit und Entscheidbarkeit von $\mathcal{F}_0$

## Satz 1.23 (Korrektheit und Vollständigkeit von $\mathcal{F}_0$ )

Sei  $A \in \mathcal{F}_0$  eine Formel der Aussagenlogik.

a) *Korrektheit:* Aus  $\vdash_{\mathcal{F}_0} A$  folgt  $\models A$ , d.h. nur Tautologien können als Theoreme in  $\mathcal{F}_0$  hergeleitet werden.

b) *Vollständigkeit:* Aus  $\models A$  folgt  $\vdash_{\mathcal{F}_0} A$ , d.h. alle Tautologien lassen sich in  $\mathcal{F}_0$  herleiten.

# Vollständigkeit und Entscheidbarkeit von $\mathcal{F}_0$ (Fort.)

Als Hilfsmittel dient:

### Lemma 1.24

Sei  $A \equiv A(p_1, \dots, p_n) \in F_0, n > 0$ , wobei  $p_1, \dots, p_n$  die in  $A$  vorkommenden Aussagevariablen sind. Sei  $\varphi$  eine Bewertung. Ist

$$P_i := \begin{cases} p_i & , \text{ falls } \varphi(p_i) = 1 \\ \neg p_i & , \text{ falls } \varphi(p_i) = 0 \end{cases} \quad A' := \begin{cases} A & , \text{ falls } \varphi(A) = 1 \\ \neg A & , \text{ falls } \varphi(A) = 0 \end{cases}$$

$(1 \leq i \leq n)$ , dann gilt  $P_1, \dots, P_n \vdash A'$ .

Angenommen das Lemma gilt und sei  $\models A$ , d.h.  $\varphi(A) = 1$  für alle Bewertungen  $\varphi$ . Sei  $\varphi$  eine Bewertung mit  $\varphi(p_n) = 1$ . Es gilt  $P_1, \dots, P_n \vdash A$  und wegen  $P_n \equiv p_n$  gilt  $P_1, \dots, P_{n-1}, p_n \vdash A$ . Betrachtet man eine Bewertung  $\varphi'$  mit  $\varphi'(p_n) = 0$  und sonst gleich  $\varphi$ , erhält man  $P_1, \dots, P_{n-1}, \neg p_n \vdash A$ .



## Vollständigkeit und Entscheidbarkeit von $\mathcal{F}_0$ (Fort.)

- ▶ Durch Anwenden des Deduktionstheorems entstehen daraus  $P_1, \dots, P_{n-1} \vdash p_n \rightarrow A$  und  $P_1, \dots, P_{n-1} \vdash \neg p_n \rightarrow A$ .  
Gleichzeitig gilt nach dem 10. Beispiel von 1.22 auch  $P_1, \dots, P_{n-1} \vdash ((p_n \rightarrow A) \rightarrow ((\neg p_n \rightarrow A) \rightarrow A))$ .
- ▶ Durch zweimaliges Anwenden des Modus-Ponens entsteht  $P_1, \dots, P_{n-1} \vdash A$ .
- ▶ Dies gilt für jede Wahl der  $P_i, i = 1, \dots, n - 1$  und somit lässt sich das Argument iterieren. D.h. in einer Herleitung von  $A$  muss kein  $p_i$  verwendet werden, also  $\vdash A$ .
- ▶ Das Lemma wird durch Induktion über den Aufbau von  $A$  nachgewiesen. D.h. für  $A \equiv p_1, \neg C, B \rightarrow C$  unter Verwendung von Deduktionen aus Beispiel 1.22.

# Folgerung

## Folgerung 1.25

Sei  $\Sigma \subseteq F_0, A \in F_0$ .

1.  $\Sigma \vdash_{\mathcal{F}_0} A$  gilt genau dann, wenn  $\Sigma \models A$  gilt.
2.  $\Sigma$  ist genau dann konsistent, wenn  $\Sigma$  erfüllbar ist.
3.  $\Sigma$  ist genau dann inkonsistent, wenn  $\Sigma$  unerfüllbar ist.
4. Ist  $\Sigma$  endlich und  $A \in F_0$ , dann ist  $\Sigma \vdash_{\mathcal{F}_0} A$  entscheidbar.

## Beweis der Folgerung

Beweis:

1.

$$\Sigma \vdash_{\mathcal{F}_0} A$$

$$\begin{array}{l} \text{1.20} \\ \iff \end{array} \text{Es gibt } A_1, \dots, A_n \in \Sigma \text{ mit } A_1, \dots, A_n \vdash_{\mathcal{F}_0} A$$

$$\begin{array}{l} \text{D.T.} \\ \iff \end{array} \text{Es gibt } A_1, \dots, A_n \in \Sigma \text{ mit} \\ \vdash_{\mathcal{F}_0} (A_1 \rightarrow (A_2 \rightarrow \dots (A_n \rightarrow A) \dots))$$

$$\begin{array}{l} \text{1.23} \\ \iff \end{array} \text{Es gibt } A_1, \dots, A_n \in \Sigma \text{ mit} \\ \models (A_1 \rightarrow (A_2 \rightarrow \dots (A_n \rightarrow A) \dots))$$

$$\begin{array}{l} \text{D.T.} \\ \iff \end{array} \text{Es gibt } A_1, \dots, A_n \in \Sigma \text{ mit } A_1, \dots, A_n \models A$$

$$\begin{array}{l} \text{K.S.} \\ \iff \end{array} \Sigma \models A$$

## Beweis der Folgerung

2.  $\Sigma$  ist konsistent.  $\iff$   
Es gibt kein  $A$  mit  $\Sigma \vdash A$  und  $\Sigma \vdash \neg A$ .  $\iff$   
Es gibt kein  $A$  mit  $\Sigma \models A$  und  $\Sigma \models \neg A$ .  $\iff$   
 $\Sigma$  ist erfüllbar (Bemerkung 1.8 c)).

## Natürliche Kalküle

Es gibt andere deduktive Systeme, für die Satz 1.23 gilt. Das deduktive System  $\mathcal{F}_0$  wurde von **S.C. Kleene** eingeführt. Das folgende deduktive System geht auf G. Gentzen zurück.

### Definition 1.26 (Gentzen-Sequenzenkalkül)

Eine Sequenz ist eine Zeichenreihe der Form  $\Gamma \vdash \Delta$  mit zwei endlichen Mengen von Formeln  $\Gamma$  und  $\Delta$ .

Seien  $\Gamma, \Delta \subseteq F$  endliche Mengen von Formeln und  $A, B \in F$ .

Der Kalkül für Objekte der Form  $\Gamma \vdash_G \Delta$  wird definiert durch folgende Axiome und Regeln:

# Gentzen-Sequenzenkalkül: Axiome und Regeln

Ax1:  $\Gamma, A \vdash_G A, \Delta$

Ax2:  $\Gamma, A, \neg A \vdash_G \Delta$

Ax3:  $\Gamma \vdash_G A, \neg A, \Delta$

$R_{\wedge, \vee}$ : 
$$\frac{\Gamma, A, B \vdash_G \Delta}{\Gamma, A \wedge B \vdash_G \Delta}$$

$$\frac{\Gamma \vdash_G A, B, \Delta}{\Gamma \vdash_G (A \vee B), \Delta}$$

$R_{\rightarrow}$ : 
$$\frac{\Gamma, A \vdash_G \Delta, B}{\Gamma \vdash_G (A \rightarrow B), \Delta}$$

$$\frac{\Gamma \vdash_G A, \Delta; \quad \Gamma, B \vdash_G \Delta}{\Gamma, (A \rightarrow B) \vdash_G \Delta}$$

$R_{\neg}$ : 
$$\frac{\Gamma, A \vdash_G \Delta}{\Gamma \vdash_G \neg A, \Delta}$$

$$\frac{\Gamma \vdash_G A, \Delta}{\Gamma, \neg A \vdash_G \Delta}$$

$R_{\wedge'}$ : 
$$\frac{\Gamma \vdash_G A, \Delta; \quad \Gamma \vdash_G B, \Delta}{\Gamma \vdash_G (A \wedge B), \Delta}$$

$R_{\vee'}$ : 
$$\frac{\Gamma, A \vdash_G \Delta; \quad \Gamma, B \vdash_G \Delta}{\Gamma, (A \vee B) \vdash_G \Delta}$$

## Gentzen-Sequenzkalkül

$\Gamma \vdash_G \Delta$  ist ableitbar bedeutet: Es gibt ein  $r \in \mathbb{N}$  und eine Folge von Sequenzen  $\Gamma_1 \vdash_G \Delta_1, \dots, \Gamma_r \vdash_G \Delta_r$  mit

1.  $\Gamma_r \equiv \Gamma$  und  $\Delta_r \equiv \Delta$
2. Jedes  $\Gamma_j \vdash_G \Delta_j$  mit  $1 \leq j \leq r$  ist Axiom oder geht aus vorangehenden Folgliedern aufgrund einer Regel hervor.

### Bemerkung 1.27 (Semantische Interpretation)

*Die Aussage  $\Gamma \vdash_G \Delta$  kann wie folgt anschaulich interpretiert werden: Für jede Bewertung  $\varphi$  gibt es eine Formel  $A \in \Gamma$  mit  $\varphi(A) = 0$  oder es gibt eine Formel  $B \in \Delta$  mit  $\varphi(B) = 1$ . Sind  $\Gamma = \{A_1, \dots, A_n\}$  und  $\Delta = \{B_1, \dots, B_m\}$ , also endlich, entspricht dies also der Formel  $(A_1 \wedge \dots \wedge A_n) \rightarrow (B_1 \vee \dots \vee B_m)$ .*

- ▶ Der Semantische Folgerungsbegriff  $\Sigma \models A$  für eine Menge von Formeln  $\{\Sigma, A\}$  kann wie folgt auf Mengenpaare  $\Gamma, \Delta$  erweitert werden:

$$\Gamma \models \Delta \quad \text{gdw} \quad \Gamma \models A$$

wobei  $A$  die Disjunktion der Formeln in  $\Delta$  ist.

- ▶ Interpretiert man in einer Sequenz  $\Gamma \vdash_G \Delta$  die Menge  $\Gamma$  als Voraussetzungen, und die Menge  $\Delta$  als Konklusion, so lässt sich die Korrektheit des Kalküls leicht nachweisen.
- ▶ Es gilt also:  
Aus  $\Gamma \vdash_G \Delta$  folgt  $\Gamma \models \Delta$ . (**Übung**)
- ▶ Es gilt auch die Umkehrung dieser Aussage, d.h. der Sequenzenkalkül von Gentzen ist korrekt und vollständig.  
(Bew. siehe z.B. Kleine Büning/Lettmann: Aussagenlogik, Deduktion und Algorithmen)

# Beispiel

## Beispiel 1.28

Es gilt  $p \vee q, (\neg p) \vee r \vdash_G q \vee r$

Beweis:

$B_1 \equiv q, r \vdash q, r$	Ax1
$B_2 \equiv q, \neg p \vdash q, r$	Ax1
$B_3 \equiv q, (\neg p) \vee r \vdash q, r$	$R_{\vee'}(1,2)$
$B_4 \equiv p, r \vdash q, r$	Ax1
$B_5 \equiv \neg p, p \vdash q, r$	Ax2
$B_6 \equiv p, (\neg p) \vee r \vdash q, r$	$R_{\vee'}(4,5)$
$B_7 \equiv p \vee q, (\neg p) \vee r \vdash q, r$	$R_{\vee'}(3,6)$
$B_8 \equiv p \vee q, (\neg p) \vee r \vdash q \vee r$	$R_{\vee}(7)$

■

## Bemerkung 1.29 (Weitere Kalküle für die Aussagenlogik)

*Man findet in der Literatur eine Vielzahl von „natürlichen“ Kalkülen (deduktiven Systemen), die ebenfalls korrekt und vollständig sind. Für diese werden auch Beweisstrategien für so genannte „Goals“ und „Subgoals“ vorgestellt.*

*Als Beispiel **Hilberts Kalkül**, das z.B. für jeden Operator eine Regel für die Einführung und eine für die Entfernung des Operators enthält.*



Natürliche Kalküle

- Transitivität  $\Leftrightarrow I : \frac{p \leftrightarrow q, q \leftrightarrow r}{p \leftrightarrow r}$
- Deduktions - Theorem  $\rightarrow I : \frac{p, \dots, r, \underline{s} \vdash t}{p, \dots, r \vdash s \rightarrow t}$
- Reductio ad absurdum  $\neg I : \frac{p, \dots, r, \underline{s} \vdash t, p, \dots, r, \underline{s} \vdash \neg t}{p, \dots, r \vdash \neg s}$
- Hypothetischer Syllogismus  $\frac{p \rightarrow q, q \rightarrow r}{p \rightarrow r}$
- Konstruktives Dilemma  $\frac{p \rightarrow q, r \rightarrow s, p \vee r}{q \vee s}$

Hinzukommen die üblichen Assoziativitäts-, Kommutativitäts-,  
Distributivitäts-, Negations-, Implikations- und de Morgan Regeln.

## Beispiele in Hilberts Kalkül

### Beispiel 1.30

Zeige  $(p \wedge q) \vee r \vdash \neg p \rightarrow r$

Beweis:

#### ► Transformationsbeweis

- |    |                                |                     |
|----|--------------------------------|---------------------|
| 1. | $(p \wedge q) \vee r$          | Prämisse            |
| 2. | $r \vee (p \wedge q)$          | Kommutativität      |
| 3. | $(r \vee p) \wedge (r \vee q)$ | Distributivität     |
| 4. | $(r \vee p)$                   | $\wedge\_E$         |
| 5. | $(p \vee r)$                   | Kommutativität      |
| 6. | $(\neg\neg p \vee r)$          | Negations-Gesetz    |
| 7. | $\neg p \rightarrow r$         | Implikations-Gesetz |



# Beispiele in Hilberts Kalkül

Zeige  $(p \wedge q) \vee r \vdash \neg p \rightarrow r$

Beweis:

► **Bedingter Beweis**

- |    |                                      |  |
|----|--------------------------------------|--|
| 1. | $(p \wedge q) \vee r$                | Prämisse                                   |
| 2. | $\neg(\neg p \vee \neg q) \vee r$    | Doppelnegation, de Morgan                  |
| 3. | $(\neg p \vee \neg q) \rightarrow r$ | Implikationsgesetz                         |
| 4. | $\neg p$                             | Annahme                                    |
| 5. | $\neg p \vee \neg q$                 | $\vee\text{-}I$                            |
| 6. | $r$                                  | Modus Ponens $\rightarrow E$ aus 3. und 5. |
| 7. | $\neg p \rightarrow r$               | Aus 4, 5, 6 mit Ded. Theo. $\rightarrow I$ |

■

## Beispiele in Hilberts Kalkül

Zeige  $(p \wedge q) \vee r \vdash \neg p \rightarrow r$

Beweis:

### ► Indirekter Beweis

- |    |                                  |  |
|----|----------------------------------|--|
| 1. | $(p \wedge q) \vee r$            | Prämisse                               |
| 2. | $(p \vee r) \wedge (q \vee r)$   | Distributivgesetz                      |
| 3. | $(p \vee r)$                     | $\wedge\_E$                            |
| 4. | $\neg(\neg p \rightarrow r)$     | Annahme                                |
| 5. | $\neg(p \vee r)$                 | Implikations- und Negationsgesetz      |
| 6. | $\neg\neg(\neg p \rightarrow r)$ | Red. Abs. $\rightarrow I$ aus 3, 4, 5. |
| 7. | $\neg p \rightarrow r$           | Doppelnegation                         |



# Algorithmischer Aufbau der Aussagenlogik

In diesem Abschnitt betrachten wir Verfahren die bei gegebener endlichen Menge  $\Sigma$  und A-Form  $A$  entscheiden ob  $\Sigma \models A$  gilt. Die bisher betrachteten Verfahren prüfen **alle Belegungen** der in den Formeln vorkommenden Variablen oder zählen effektiv die Theoreme eines geeigneten deduktiven Systems auf. **Dies ist sicherlich recht aufwendig**. Obwohl die Komplexität dieses Problems groß ist (Entscheidbarkeit von  $SAT$  ist bekanntlich NP-vollständig), ist die Suche nach Verfahren, die „oft“ schneller als die „brute force Methode“ sind, berechtigt.

Wir betrachten drei solcher Verfahren die alle Erfüllbarkeitsverfahren sind, d.h. sie basieren auf:

$$\Sigma \models A \quad \text{gdw} \quad \{\Sigma, \neg A\} \text{ unerfüllbar:}$$

Semantische Tableaux    Davis-Putman    Resolution





# Feststellungen

Zwei Arten von Formeln, solche, die zu Verzweigungen führen ( $\beta$ -Formeln), und solche, die nicht zu Verzweigungen führen ( $\alpha$ -Formeln).

- ▶  **$\alpha$ -Formeln** mit Komponenten  $\alpha_1$  und  $\alpha_2$ , die zu Knoten mit den Markierungen  $\alpha_1$  und  $\alpha_2$  führen:

$\alpha$	$\neg\neg A$	$A_1 \wedge A_2$	$\neg(A_1 \vee A_2)$	$\neg(A_1 \rightarrow A_2)$
$\alpha_1$	$A$	$A_1$	$\neg A_1$	$A_1$
$\alpha_2$	$(A)$	$A_2$	$\neg A_2$	$\neg A_2$

- ▶  **$\beta$ -Formeln** mit Komponenten  $\beta_1$  und  $\beta_2$ , die zu Verzweigungen führen mit Knotenmarkierungen  $\beta_1$  und  $\beta_2$ :

$\beta$		$\neg(A_1 \wedge A_2)$	$A_1 \vee A_2$	$A_1 \rightarrow A_2$
$\beta_1$	$\beta_2$	$\neg A_1$	$A_1$   $A_2$	$\neg A_1$   $A_2$





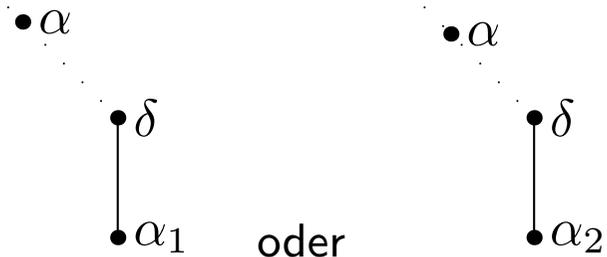




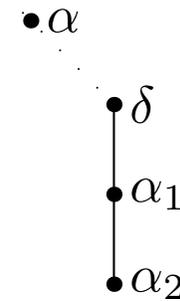
## Formalisierung (Fort.)

1. (b)  $(\alpha)$  einen Knoten hinzufügt, der mit  $\alpha_1$  oder  $\alpha_2$  markiert ist, falls eine  $\alpha$ -Formel  $\alpha$  auf dem Ast zu  $\delta$  vorkommt und  $\alpha_1$  und  $\alpha_2$  die Komponenten von  $\alpha$  sind.

Graphisch:



In der Praxis werden jedoch an  $\delta$  nacheinander die Knoten für beide Komponenten hinzugefügt:



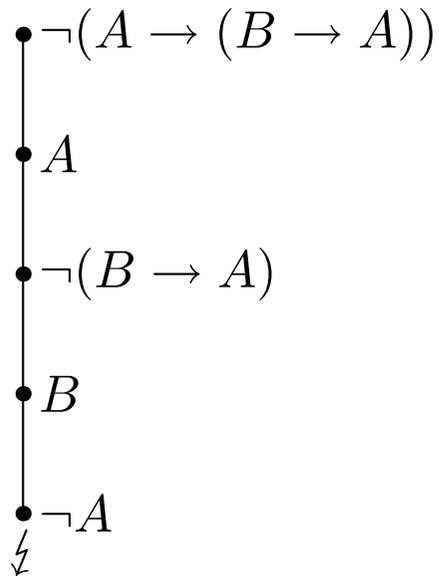




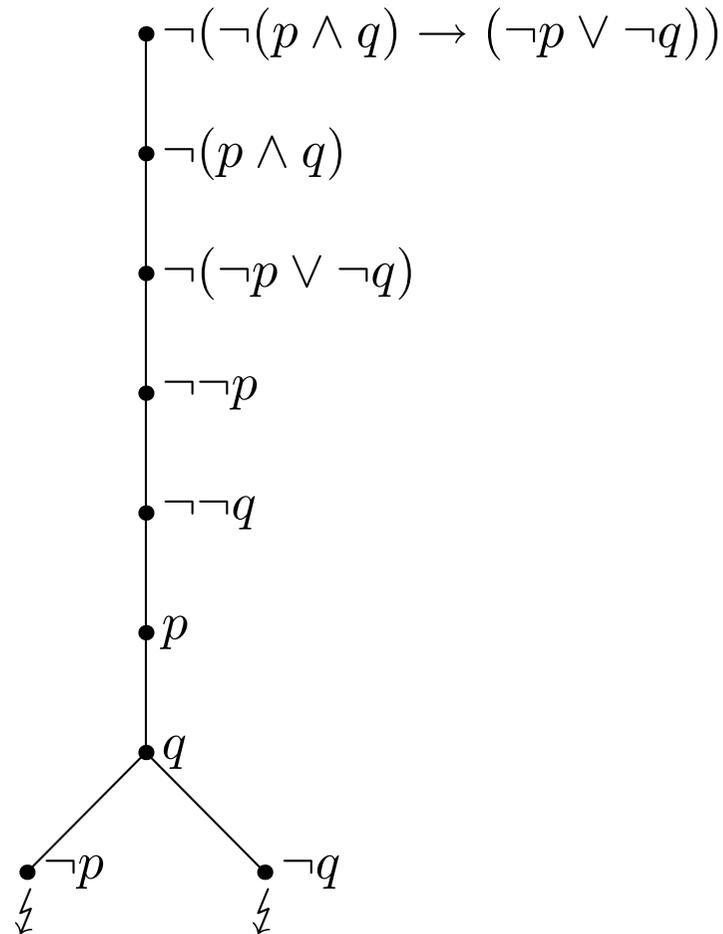


## Beispiel 2.4

$\vdash_{\tau} A \rightarrow (B \rightarrow A)$ :



$\vdash_{\tau} \neg(p \wedge q) \rightarrow (\neg p \vee \neg q)$ :

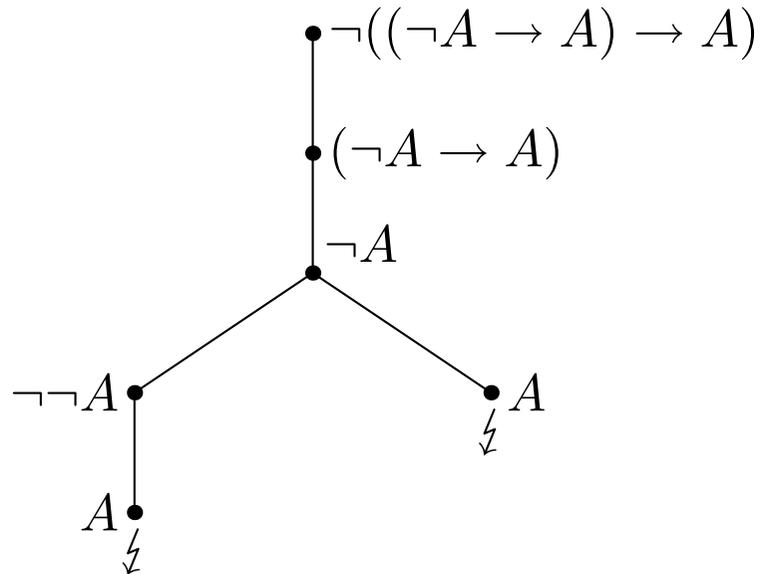




# Beispiele (Fort.)

## Beispiel 2.6

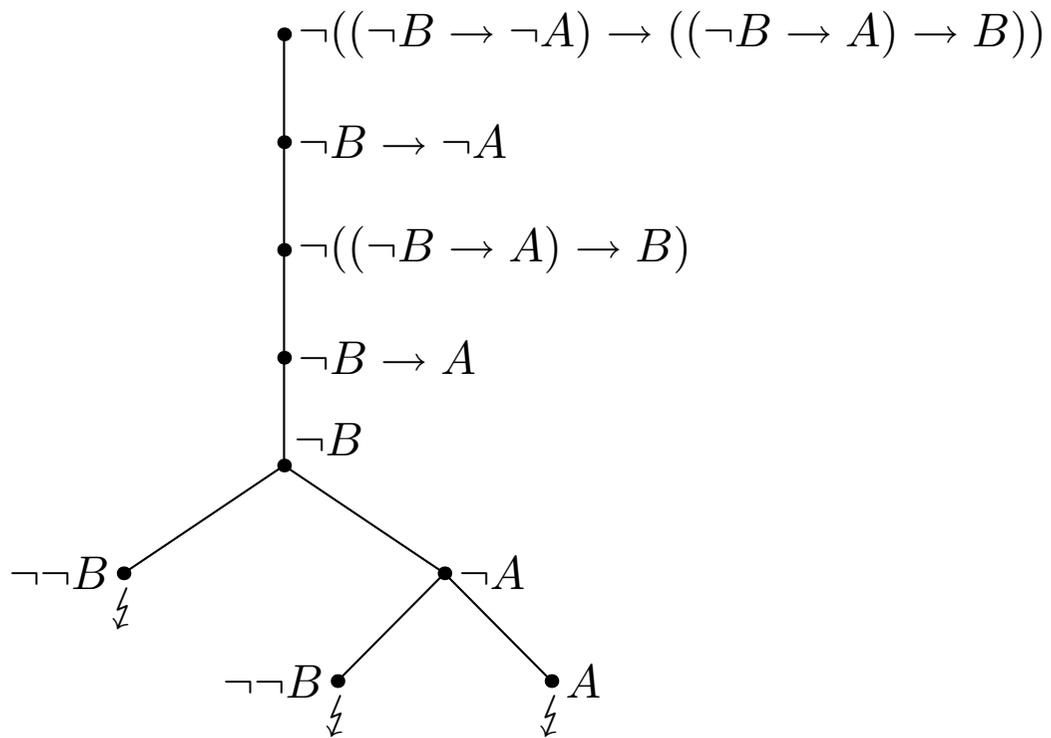
$$\vdash_{\tau} (\neg A \rightarrow A) \rightarrow A$$



# Beispiele (Fort.)

## Beispiel 2.7

$$\vdash_{\tau} (\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$$



# Vollständige Tableaux

## Definition 2.8

Sei  $\tau$  ein Tableau,  $\Theta$  ein Ast von  $\tau$ .

- ▶  $\Theta$  heißt **vollständig**, falls für die Menge der Formeln in  $\Theta$  gilt:  
Mit jeder  $\alpha$ -Formel  $\alpha \in \Theta$  ist stets  $\{\alpha_1, \alpha_2\} \subseteq \Theta$  und mit jeder  $\beta$ -Formel  $\beta \in \Theta$  ist stets  $\beta_1 \in \Theta$  oder  $\beta_2 \in \Theta$ .
- ▶  $\tau$  heißt **vollständig**, falls jeder Ast in  $\tau$  abgeschlossen oder vollständig ist.
- ▶ Sei  $\Sigma \subseteq F$ ,  $\Sigma$  endlich.  $\tau \in \mathcal{T}_\Sigma$  heißt **vollständig für  $\Sigma$** , falls  $\tau$  vollständig ist und jeder offene Ast  $\Sigma$  enthält.
- ▶ Sei  $\Sigma \subseteq F$ ,  $\Sigma$  unendlich, so verallgemeinerte Tableaux erlaubt (d.h. jeder offene Ast ist unendlich und enthält  $\Sigma$ ).



## Vollständige Tableaux (Fort.)

### Lemma 2.10

*Jede Menge  $\Sigma$  von Formeln, die 1, 2 und 3 aus der Bemerkung 2.9.2 genügt, ist erfüllbar. Insbesondere sind vollständige offene Äste von Tableaux erfüllbar.*

*Gibt es offene vollständige Tableaux für  $\Gamma$ , so ist  $\Gamma$  erfüllbar.*

**Beweis:**

Definiere:

$$\varphi(p) = \begin{cases} 0 & \neg p \in \Sigma \\ 1 & \text{sonst} \end{cases}$$

Offensichtlich ist  $\varphi$  wohldefiniert.

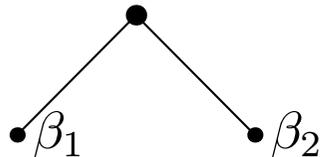
Beh.: Falls  $A \in \Sigma$ , dann  $\varphi(A) = 1$ . (Induktion) ■





oder:

- 4. Ist  $Y$  die erste unmarkierte  $\beta$ -Formel in  $\tau_n$ , durch die ein offener Ast geht, so markiere  $Y$  und erweitere jeden offenen Ast, der durch  $Y$  geht, um



Markiere  $\beta_1$  und/oder  $\beta_2$ , falls diese Literale sind. Dadurch werden möglicherweise Äste abgeschlossen.

oder:

- 5. Gibt es eine Formel  $A_j \in \Gamma$ , die noch nicht in jedem offenen Ast vorkommt, so erweitere alle diese Äste um:



Falls möglich, Knoten markieren und Äste abschließen.

- ▶ **Verfahren:** Beginne mit  $\tau_1$ . Wiederhole 3. solange wie möglich. Dann 4.. Sind weder 3. noch 4. möglich so 5. Geht nichts mehr, so stopp.
- Aus  $\tau_n$  ( $n \geq 1$ ) erhält man kein weiteres Tableau, falls  $\tau_n$  abgeschlossen ist oder alle Formeln von  $\tau_n$  markiert sind und  $\Gamma$  endlich und ausgeschöpft ist.
- ▶ Setze  $\tau_\infty := \bigcup_{n \in \mathbb{N}} \tau_n$ . Dann ist  $\tau_\infty$  ein binärer Baum.

Behauptung:  $\tau_\infty$  ist vollständig!

## Beweis:

1.  $\tau_\infty = \tau_k$  für ein  $k \in \mathbb{N}$ .
  - ▶ Ist  $\tau_k$  abgeschlossen, gilt die Behauptung.
  - ▶ Ist  $\tau_k$  nicht abgeschlossen, so ist  $\tau_k$  **vollständig**: Alle Formeln sind markiert und  $\Gamma$  muss endlich sein. Alle Formeln von  $\Gamma$  sind in den offenen Ästen von  $\tau_k$ . Somit ist  $\Gamma$  nach Lemma 2.10 erfüllbar.
2.
  - ▶ Es gibt kein  $k \in \mathbb{N}$  mit  $\tau_\infty = \tau_k$ . Dann ist  $\tau_\infty$  ein unendlicher Baum.
  - ▶ Es gibt eine Folge von Knoten  $\{Y_n\}$ ,  $n \in \mathbb{N}$ , die unendlich viele Nachfolger haben: Setze  $Y_1 = A_1$ , die Wurzel mit unendlich vielen Nachfolgerknoten. Ist  $Y_n$  bereits gefunden, dann hat  $Y_n$  entweder einen oder zwei direkte Nachfolger, von denen einer unendlich viele Nachfolger hat. Wähle als  $Y_{n+1}$  diesen Knoten. Dann ist der Ast  $\{Y_n | n \in \mathbb{N}\}$  in  $\tau_\infty$ , offen, vollständig und enthält  $\Gamma$ , d.h.  $\Gamma$  ist erfüllbar.

## Bemerkung und Folgerung

### Bemerkung 2.12

1. *Ist  $\Gamma$  eine rekursiv aufzählbare Menge, so ist das Hinzufügen einer Formel  $A_n \in \Gamma$  zu einem Tableau effektiv, d.h. falls  $\Gamma$  rekursiv aufzählbar aber nicht erfüllbar ist, so stoppt die systematische Tableau-Konstruktion. Insbesondere stoppt die systematische Tableau-Konstruktion immer, wenn  $\Gamma$  endlich ist. Sie liefert dann entweder:*
  - ▶  $\Gamma$  ist nicht erfüllbar, d.h. es gibt eine  $n \in \mathbb{N}$ , so dass  $\tau_n$  abgeschlossen ist, oder:
  - ▶  $\Gamma$  ist erfüllbar und die (offenen) Äste von  $\tau_n$  liefern alle Belegungen, die  $\Gamma$  erfüllen.

*Die systematische Tableau-Konstruktion liefert also für endliche Mengen in den offenen vollständigen Äste alle Belegungen der wesentlichen Variablen, die  $\Gamma$  erfüllen.*



## Folgerungen (Fort.)

3.

$$\begin{aligned}
 \Gamma \models A &\iff \Gamma \cup \{\neg A\} \text{ unerfüllbar} \\
 &\iff \mathcal{T}\{\Gamma, \neg A\} \text{ enthält abgeschlossenes Tableau} \\
 &\iff \Gamma \vdash_{\tau} A
 \end{aligned}$$

Für  $\Gamma$  endlich beginne also mit Anfangstableau für  
 $\{\neg A, A_1, \dots, A_n\}$

## Folgerungen (Fort.)

4. ●(a)  $\models ((p \vee q) \wedge (\neg p \vee r)) \rightarrow (q \vee r)$  oder  
 $(p \vee q) \wedge (\neg p \vee r) \models (q \vee r)$

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

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

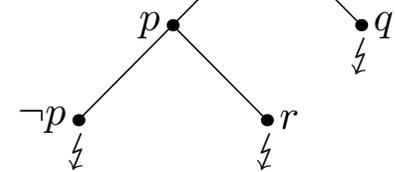
$$\neg(q \vee r)$$

$$(p \vee q)$$

$$\neg p \vee r$$

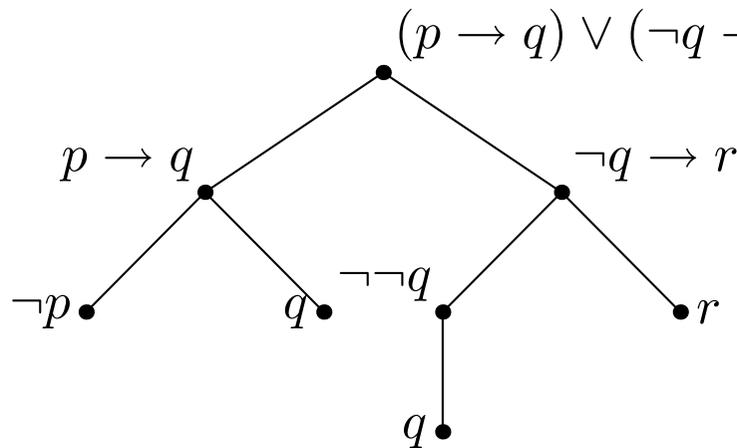
$$\neg q$$

$$\neg r$$



# Folgerungen (Fort.)

- (b) Bestimme alle Belegungen, die  $A \equiv (p \rightarrow q) \vee (\neg q \rightarrow r)$  erfüllen!



Demnach ist  $\{\varphi \mid \varphi \text{ ist Bewertung mit } \varphi(p) = 0 \text{ oder } \varphi(q) = 1 \text{ oder } \varphi(r) = 1\}$  die Menge aller Belegungen, die  $A$  erfüllen. An den Blättern des Baumes lässt sich auch eine äquivalente Disjunktive Normalform (DNF) zur Formel  $A$  ablesen, nämlich  $\neg p \vee q \vee r$ .

# Normalformen

- ▶ **Normalformen** haben oft den Vorteil, dass man aus Formeln in dieser Form gewisse Informationen leicht ablesbar sind. So lassen sich z.B. aus einer KDNF (kanonische disjunktive Normalform) alle erfüllende Belegungen aus den Elementarkonjunktionen direkt ablesen. Aus minimalen DNF lassen sich leicht die Schaltnetze (mit UND, ODER, NEG Gattern) herleiten. Die systematische Tableau Konstruktion erlaubt es diese Normalformen aus einem vollständigen Tableau abzulesen.

# Normalformen

**Motivation:** Oft will man eine beliebige A-Form in eine Form transformieren die „einfachere“ Gestalt hat und spezielle Algorithmen zur Lösung einer bestimmten Fragestellung für Formeln in dieser Gestalt verfügbar sind. Die Transformation sollte nicht zu teuer sein, sonst würde sich der Aufwand dafür nicht lohnen.

- ▶ Transformiert werden kann in einer
  - ▶ logisch äquivalenten Formel, d.h.  $A \models T(A)$  oder
  - ▶ erfüllungs äquivalenten Formel, d.h.  
 $A$  erfüllbar gdw.  $T(A)$  erfüllbar
- ▶ Wir behandeln drei dieser Normalformen:
  - ▶ **Negationsnormalform (NNF)** Form in  $\neg, \vee, \wedge$
  - ▶ **Konjunktive Normalform (KNF)** Form in  $\neg, \vee, \wedge$
  - ▶ **Disjunktive Normalform (DNF)** Form in  $\neg, \vee, \wedge$



## Klauseln

### Definition 2.15 (Klausel)

Eine Formel  $A \equiv (L_1 \vee \dots \vee L_n)$  mit Literalen  $L_i$  für  $i = 1, \dots, n$  wird **Klausel** genannt.

- Sind alle Literale einer Klausel **negativ**, so ist es eine **negative** Klausel. Sind **alle Literale positiv**, so ist es eine **positive** Klausel. Klauseln die **maximal ein positives Literal** enthalten, heißen **Horn Klauseln**.
- $A$  wird  **$k$ -Klausel** genannt, falls  $A$  maximal  $k$  Literale enthält. 1-Klauseln werden auch **Unit-Klauseln** genannt.
- Eine Formel  $A$  ist in **KNF** gdw.  $A$  eine Konjunktion von Klauseln ist. D.h.  
 $A \equiv (A_1 \wedge \dots \wedge A_m)$  mit Klauseln  $A_i$  für  $i = 1, \dots, m$ .
- Sind die  $A_i$   $k$ -Klauseln, so ist  $A$  in  **$k$ -KNF**.

## Normalformen (Fort.)

### Beispiel 2.16

$A \equiv (p \vee q) \wedge (p \vee \neg q) \wedge (\neg p \vee q) \wedge (\neg p \vee \neg q)$  ist in 2-KNF (Beachte ist unerfüllbar). Betrachtet man Klauseln als Mengen von Literalen, so lassen sich Formeln in KNF als Mengen von Literalismengen darstellen, etwa

$$A \equiv \{\{p, q\}, \{p, \neg q\}, \{\neg p, q\}, \{\neg p, \neg q\}\}.$$

### Lemma 2.17

Zu jeder Formel  $A \in F(\{\neg, \wedge, \vee\})$  gibt es eine logisch äquivalente Formel  $B$  in KNF mit  $|B| \in O(2^{|A|})$ .

- **Beachte:** Es gibt eine Folge von Formeln  $A_n$  mit  $|A_n| = 2n$ , für die jede logisch äquivalente Formel  $B_n$  in KNF mindestens die Länge  $2^n$  besitzt.

## Definition 2.18 (DNF)

Eine A-Form  $A$  ist in **DNF** gdw.  $A$  eine Disjunktion von Konjunktionen von Literalen ist, d.h.  $A \equiv (A_1 \vee \dots \vee A_i)$  mit  $A_i \equiv (L_{i1} \wedge \dots \wedge L_{im_i})$ .

## Definition 2.19 (Duale Formel)

Die **duale Formel** von  $A$ ,  $d(A)$  (auch  $A^*$ ) ist definiert durch:

- ▶  $d(p) \equiv p$  für  $p \in V$
- ▶  $d(\neg A) \equiv \neg d(A)$
- ▶  $d(B \vee C) \equiv (d(B) \wedge d(C))$
- ▶  $d(B \wedge C) \equiv (d(B) \vee d(C))$

## Lemma 2.20

Für jede  $A$ -Form  $A$  gilt:

1. Sei  $A$  in  $KNF$ , dann ist  $NNF(\neg A)$  in  $DNF$ .
2. Ist  $A$  in  $KNF$ , so ist  $d(A)$  in  $DNF$ .
3.  $A$  ist Tautologie gdw.  $d(A)$  widerspruchsvoll.
4.  $A$  ist erfüllbar gdw.  $d(A)$  ist keine Tautologie.

Setzt man  $\varphi'(p) = 1 - \varphi(p)$ , so gilt  $\varphi'(d(A)) = 1 - \varphi(A)$



## Definition 2.21

Sei  $A$  A-Form,  $p \in \mathbb{V}$  definiere  $A[p/1]$  (bzw.  $A[p/0]$ ) als Ergebnis des folgenden Ersetzungsprozesses:

1. Ersetze in  $A$  jedes Vorkommen von  $p$  durch 1.
2.
  - Tritt nun eine Teilform  $\neg 1$  auf, ersetze sie durch 0,
  - $\neg 0$  ersetze durch 1.
  - Teilformeln  $B \wedge 1$ , sowie  $B \vee 0$  werden durch  $B$  ersetzt,
  - Teilformeln  $B \vee 1$  durch 1 und
  - Teilformeln  $B \wedge 0$  durch 0 ersetzt.
3. Schritt 2 wird so lange durchgeführt, bis keine weitere Ersetzung möglich ist.

Analog für  $A[p/0]$ .

Allgemeiner verwende  $A[l/1]$  bzw.  $A[l/0]$  für Literale  $l$ .

- ▶ **Beachte:** Für  $A$  in KNF und Literal  $l$  gilt:  
 $A[l/1]$  entsteht aus  $A$  durch Streichen aller Klauseln, die das Literal  $l$  enthalten und durch Streichen aller Vorkommen des Literals  $\neg l$  in allen anderen Klauseln.
- ▶  $A[p/1]$  (bzw.  $A[p/0]$ ) sind wohldefiniert. (Warum ?)
- ▶ Als Ergebnis des Ersetzungsprozesses  $A[p/i]$   $i = 1, 0$  erhält man:
  - ▶ eine A-Form (in NNF bzw. KNF wenn  $A$  diese Form hatte)
  - ▶ 1 die „leere Formel“
  - ▶ 0 die „leere Klausel“ ( $\perp, \square$ )

Die leere Formel wird als wahr interpretiert. Die leere Klausel als falsch (nicht erfüllbar), d. h.  $A[p/i]$  als A-Form behandelbar

Für jedes Atom  $p \in \mathbb{V}$  und  $A \in F$  gilt:

1.  $A[p/i]$   $i \in \{1, 0\}$  ist entweder die leere Formel oder die leere Klausel oder eine  $A$ -Form in NNF in der  $p$  nicht vorkommt. Ist  $\varphi$  eine Bewertung mit  $\varphi(p) = i$ , so gilt  $\varphi(A) = \varphi(A[p/i])$ .
  2.  $A \wedge p \models \equiv A[p/1] \wedge p \qquad A \wedge \neg p \models \equiv A[p/0] \wedge \neg p$
  3.  $A$  ist erfüllbar gdw  $A[p/1] = 1$  oder  $A[p/0] = 1$  oder eine der Formeln  $A[p/1], A[p/0]$  erfüllbar ist.
- ↪ Durch Testen der Teilbewertungen  $A[p/1]$  und  $A[p/0]$  kann rekursiv die Erfüllbarkeit von  $A$  entschieden werden.

# Regelbasierter Aufbau von DP-Algorithmen

## Definition 2.22 (Regeln für Formeln in **NNF**)

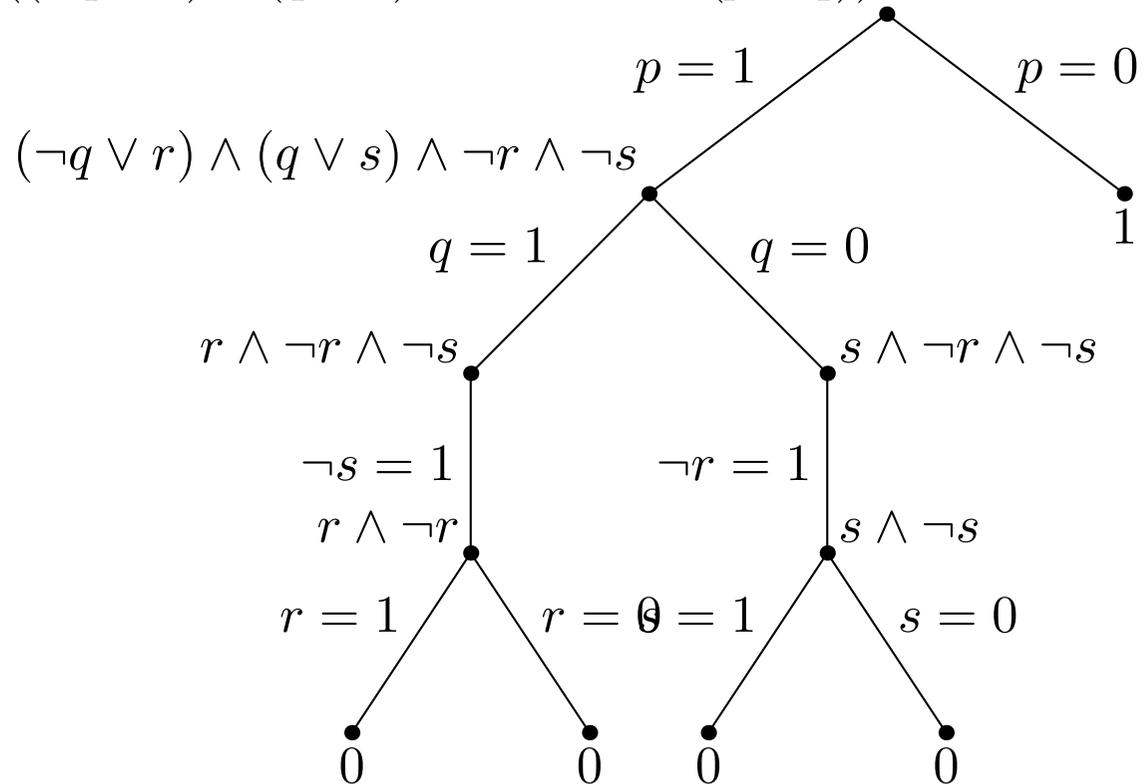
1. **Pure-Literal Regel** Kommt ein Atom  $p \in \mathbb{V}$  in einer A-Form  $A$  nur positiv oder nur negativ vor, so können wir  $p$  mit 1 bzw. 0 belegen und die Formel dementsprechend kürzen.  
 $\hookrightarrow$  (Es gilt  $A[p/0] \models A[p/1]$  bzw.  $A[p/1] \models A[p/0]$ ), genauer  $A$  erfüllungsäquivalent  $A[p/1]$  bzw.  $A[p/0]$ .
2. **Splitting-Regel** Kommt jedes Atom sowohl positiv als auch negativ vor, so wähle ein solches Atom  $p$  in  $A$  aus und bilde aus  $A$  die zwei A-Formen  $A[p/1]$  und  $A[p/0]$ .  
 $\hookrightarrow$  Die Ausgangsformel  $A$  ist genau dann erfüllbar, wenn bei einer der Kürzungen der Wert 1 oder eine erfüllbare Formel als Ergebnis auftritt.

# Regelbasierter Aufbau von DP-Algorithmen

- ▶ Regeln reduzieren das Erfüllbarkeitsproblem für eine Formel mit  $n$  Atomen auf EP für Formeln mit maximal  $(n - 1)$  Atomen.
- ▶ Algorithmen, die mit Hilfe dieser beiden Regeln mit verschiedenen Heuristiken (zur Auswahl des splitting Atoms) und weiteren Verfeinerungen arbeiten, werden als **Davis-Putman-Algorithmen** bezeichnet.

## Beispiel 2.23 (Darstellung der Abarbeitung als Baum)

$$A \equiv \neg p \vee ((\neg q \vee r) \wedge (q \vee s) \wedge \neg r \wedge \neg s \wedge (p \vee q))$$

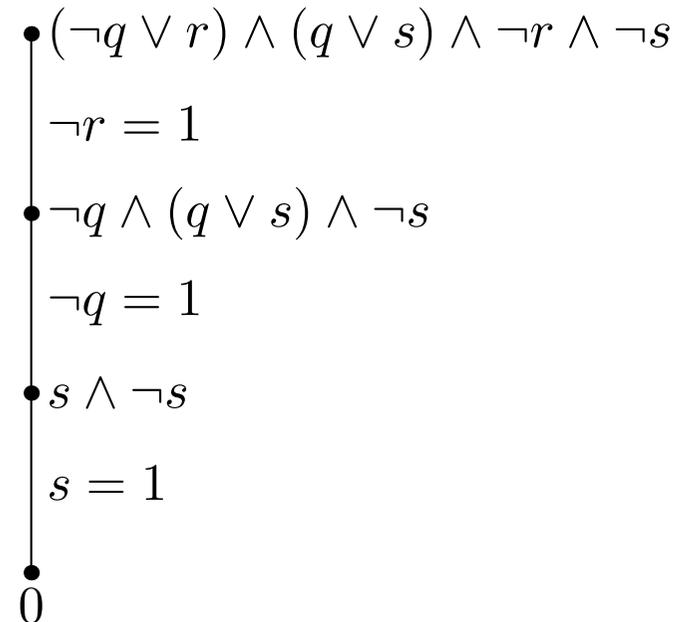


# Weitere Verfeinerungen

## Definition 2.24 (Regeln für Formeln in KNF)

### 3. Unit-Regel

Sei  $A$  in KNF und  $A$  enthält eine Unit Klausel  $A_i \equiv l$ . Bilde  $A[l/1]$  ( $A$  ist erfüllbar gdw  $A[l/1]$  erfüllbar), da das Literal einer Unit-Klausel durch eine erfüllende Bewertung auf wahr gesetzt werden muss.



- ▶ Seien  $A_1$  und  $A_2$  Klauseln.  $A_1$  **subsumiert**  $A_2$  ( $A_1 \subseteq A_2$ ) gdw jedes Literal aus  $A_1$  auch in  $A_2$  auftritt.
- ▶ Aus der Erfüllbarkeit einer Klausel folgt sofort die Erfüllbarkeit aller Klauseln, die sie subsummiert.

#### 4. **Subsumption-Rule**

Sei  $A$  in KNF. Streiche aus  $A$  alle Klauseln, die von anderen subsumiert werden::  $SR(A)$ .

Streiche insbesondere tautologische Klauseln (solche die  $p$  und  $\neg p$  für ein  $p$  enthalten).

- ▶ Da in KNF alle Klauseln konjunktiv verknüpft sind, braucht man nur diejenigen zu berücksichtigen, die von keiner anderen subsumiert werden.

**procedure** Davis/Putman//Eingabe:  $A$  in KNF//

//Ausgabe: Boolescher Wert für Erfüllbarkeit (1,0)//

**begin****if**  $A \in \{0, 1\}$  **then** return( $A$ ); $p :=$  pure( $A, s$ );//*liefert Atom und Belegung, falls nur positiv oder nur negativ vorkommt sonst null*//**if**  $p \neq$  null **then** return(DPA( $A[p/s]$ )); $p :=$  unit( $A, s$ ); //*Unit Klausel mit Belegung sonst null*//**if**  $p \neq$  null **then** return(DPA( $A[p/s]$ )); $A :=$  Subsumption\_Reduce( $A$ ); //*entfernt subs. Klauseln*// $p :=$  split( $A$ ); //*liefert Atom in  $A$* //**if** DPA( $A[p/1]$ ) = 1 **then** return(1);return(DPA( $A[p/0]$ ));**end**

## Auswahlkriterien für die Splitting Regel

- ▶ Wähle das erste in der Formel vorkommende Atom,
- ▶ wähle das Atom, welches am häufigsten vorkommt,
- ▶ ... das in den kürzesten Klauseln am häufigsten vorkommt,
- ▶ wähle Atom mit  $\sum_{p \text{ in } A_i} |A_i|$  minimal,
- ▶ berechne die Anzahl der positiven und negativen Vorkommen in den kürzesten Klauseln und wähle das Atom mit der größten Differenz.

# Resolutions-Verfahren

- ▶ Das **Resolutionskalkül** als Deduktionssystem operiert auf Klauselmengen, d. h. Formeln in KNF mit nur einer Schlussregel:  
Aus Klauseln  $(A \vee I)$  und  $(B \vee \neg I)$  kann eine neue Klausel  $(A \vee B)$  erzeugt werden.
- ▶ **Ziel:** Leere Klausel zu erzeugen.
- ▶ Klauseln als Mengen  $(p \vee \neg q \vee p) \leftrightarrow \{p, \neg q\}$   
 $I \equiv p$  so  $\neg I \equiv \neg p$        $I \equiv \neg p$  so  $\neg I \equiv p$

# Resolutions-Verfahren

## Definition 2.25 (**Resolutionsregel** (Resolventenregel))

Seien  $A, B$  Klauseln,  $I$  ein Literal.  $I$  kommt in  $A$  und  $\neg I$  kommt in  $B$  vor. Dann können  $A$  und  $B$  über  $I$  (bzw.  $\neg I$ ) resolviert werden.

- ▶ Die **Resolvente** der Klauseln  $A$  und  $B$  ist die Klausel  $(A \setminus \{I\}) \cup (B \setminus \{\neg I\})$ .

$A$  und  $B$  sind die **Elternklauseln** der Resolvente

Schema  $\frac{A, B}{Res_I(A, B) \equiv (A \setminus \{I\}) \cup (B \setminus \{\neg I\})}$  (**Resolutionsregel**)

# Eigenschaften der Resolvente

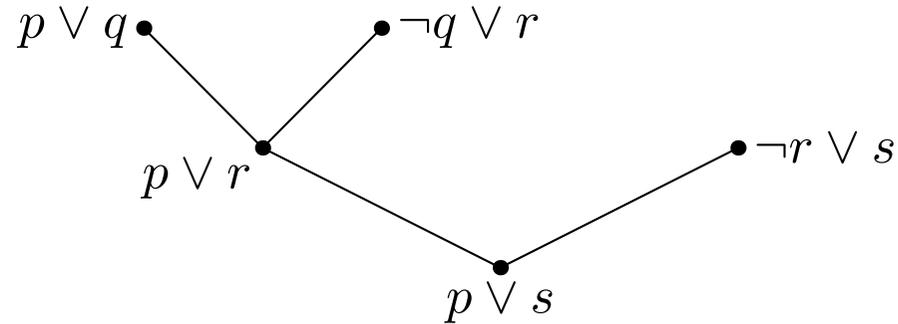
**Beachte:**

- ▶ Enthält die Resolvente ein Literal  $l'$ , so muss dieses bereits in  $A$  oder  $B$  enthalten sein.
- ▶ Schreibe auch  $A \vee l, B \vee \neg l \vdash_{Res} A \vee B$ .
- ▶  $A \wedge B$  erfüllbar   gdw    $A \wedge B \wedge Res_l(A, B)$  erfüllbar.  
  gdw    $Res_l(A, B)$  erfüllbar.
- ▶  $A \wedge B \models Res(A, B)$ .
- ▶ Resolvente kann **leere Klausel**  $\square$  sein.

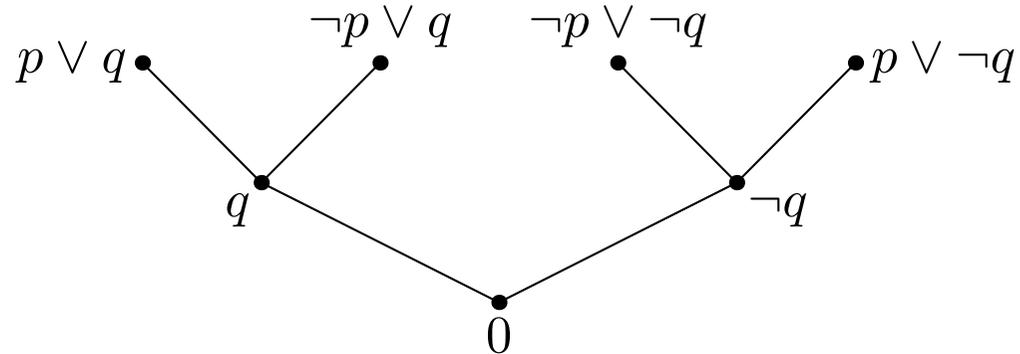


# Beispiele

b)  $A \equiv \{(p \vee q), (\neg q \vee r), (\neg r \vee s)\}$



c)  $A \equiv \{(p \vee q), (\neg p \vee q), (p \vee \neg q), (\neg p \vee \neg q)\}$





## Ableitungen im Resolutionskalkül

### Definition 2.27 (Herleitungen (Ableitungen))

Sei  $A \equiv \{C_1, \dots, C_n\}$  eine Formel in KNF und  $C$  ein Klausel. Eine Folge  $D_1, \dots, D_k$  von Klauseln ist eine **Herleitung** der Klausel  $C$  aus  $A$ . Wenn  $C \equiv D_k$  und für alle  $j$  mit  $1 \leq j \leq k$  Klauseln  $E, F \in A \cup \{D_1, \dots, D_{j-1}\}$  existieren mit  $E, F \stackrel{Res}{\vdash} D_j$ .

- ▶  $C$  ist (mit der Resolventenregel oder im Resolutionskalkül) **herleitbar** aus  $A$

Schreibweise:  $A \stackrel{+}{\vdash}_{Res} C$  ( $A$  Ausgangs-Klauseln)

- ▶  $k$  ist die **Länge** der Herleitung.



# Korrektheit und Widerlegungsvollständigkeit

## Satz 2.28

1. Der Resolutionskalkül ist korrekt.

*A in KNF, C Klausel dann  $A \stackrel{+}{\underset{Res}{\vdash}} C$ , so  $A \models C$*

2. Der Resolutionskalkül ist nicht vollständig.

*Es gibt A in KNF, C Klausel mit  $A \models C$  aber nicht  $A \stackrel{+}{\underset{Res}{\vdash}} C$*

3. Der Resolutionskalkül ist widerlegungsvollständig.

*A in KNF, A widerspruchsvoll (unerfüllbar), so  $A \stackrel{+}{\underset{Res}{\vdash}} \perp$*

## Korrektheit und Widerlegungsvollständigkeit (Forts.)

Beweis:

1.  $\checkmark$ , 2.  $A \equiv p$ ,  $C \equiv p \vee q \rightsquigarrow$  Behauptung.

3. Induktion nach Länge der Formel:

Kürzeste Formel:  $\{\{p\}, \{\neg p\}\}$ , dann  $p, \neg p \underset{Res}{\vdash} \perp$ .

Verwende dabei:  $A$  widerspruchsvoll, so auch  $A[p/1]$  und  $A[p/0]$  widerspruchsvoll.

- Sei  $A$  mit Länge  $n + 1$ ,  $A$  widerspruchsvoll. Es gibt ein Atom  $p$  in  $A$  das sowohl positiv als auch negativ vorkommt. Betrachte  $A[p/1]$  und  $A[\neg p/1]$ , beide nicht erfüllbar. Angenommen nicht Wert 0.

- Nach Ind.Vor.:  $A[p/1] \underset{Res}{\overset{+}{\vdash}} \perp$ ,  $A[p/0] \underset{Res}{\overset{+}{\vdash}} \perp$ .

## Korrektheit und Widerlegungsvollständigkeit (Forts.)

- ▶  $A$  in KNF.  $A[p/1]$  entsteht durch Streichen der Klauseln, die  $p$  enthalten und durch Streichen von  $\neg p$  aus Klauseln, die  $\neg p$  enthalten.
- ↳ Fügt man in  $A[p/1]$  die eliminierten Literale  $\neg p$  und zu  $A[\neg p/1]$  die Atome  $p$  wieder hinzu, so sind diese Formeln  $A[p/1](\neg p)$  und  $A[p/0](p)$  Teilformen von  $A$ .

## Korrektheit und Widerlegungsvollständigkeit (Forts.)

- ▶ Dann aber entweder  $A[p/1](\neg p) \stackrel{+}{\underset{Res}{\vdash}} \perp$  bzw.  $A[\neg p/1](p) \stackrel{+}{\underset{Res}{\vdash}} \perp$

auch aus  $A$  herleitbar oder

$$A[p/1](\neg p) \stackrel{+}{\underset{Res}{\vdash}} \neg p \text{ und } A[\neg p/1](p) \stackrel{+}{\underset{Res}{\vdash}} p.$$

Dann aber  $\neg p, p \stackrel{+}{\underset{Res}{\vdash}} \perp$  und somit  $A \stackrel{+}{\underset{Res}{\vdash}} \perp$ .

- ▶ Ergibt  $A[p/1]$  oder  $A[\neg p/1]$  den Wert 0, so enthält  $A$  eine Klausel  $p$  (falls  $A[\neg p/1] = 0$ ) oder eine Klausel  $\neg p$  (falls  $A[p/1] = 0$ ). Dann analoger Schluss. ■

## Lemma 2.29

Sei  $A$  in KNF,  $C$  eine Klausel, dann gilt  $A \models C$  gdw es gibt eine Teilklausel  $C' \subseteq C$ :

$$A \overset{+}{\vdash}_{\text{Res}} C'$$

► Ist  $A$  erfüllbar und  $C$  Primimplikant von  $A$ , dann gilt  $A \overset{+}{\vdash}_{\text{Res}} C$ .

# Resolventenmethode: Strategien/Heuristiken

## ▶ Verfeinerungen der Methode

**Starke Herleitungen:**  $A$  in KNF, widerspruchsvoll.

Dann gibt es eine Herleitung  $C_1, \dots, C_n \equiv \perp$  mit

1. in der Herleitung tritt keine Klausel mehrfach auf,
2. in der Herleitung tritt keine Tautologie auf,
3. in der Herleitung tritt keine schon subsumierte Klausel auf:  
Es gibt kein  $C_i, C_j, j < i, C_j \subseteq C_i$ .

### ► **Strategien, Heuristiken**

- Stufenstrategie (Resolutionsabschluss)  
( Alle erfüllende Bewertungen)
- Stützmengenrestriktion  
(Set of Support, Unit-Klauseln bevorzugen)
- P-N Resolution
- Lineare Resolution (SL-Resolution)  
(PROLOG-Inferenzmaschine)

Resolutions-Verfahren

Beispiel:  $A \equiv \{\{\neg p, \neg q, \neg r\}, \{p, \neg s\}, \{q, \neg r\}, \{r, \neg t\}, \{t\}\}$

Stufen:

0	1	2	3
1. $\{\neg p, \neg q, \neg r\}$	6. $\{\neg q, \neg r, \neg s\}$ (1,2)	11. $\{\neg r, \neg s\}$ (6,3)	21. $\{\neg s, \neg r\}$ (11,4)
2. $\{p, \neg s\}$	7. $\{\neg p, \neg r\}$ (1,3)	12. $\{\neg q, \neg s, \neg t\}$ (6,4)	22. $\{\neg s\}$ (11,10)
3. $\{q, \neg r\}$	8. $\{\neg p, \neg q, \neg t\}$ (1,4)	13. $\{\neg p, \neg t\}$ (7,4)	⋮
4. $\{r, \neg t\}$	9. $\{q, \neg t\}$ (3,4)	14. $\{\neg p, \neg r, \neg t\}$ (8,3)	
5. $\{t\}$	10. $\{r\}$ (4,5)	15. $\{\neg p, \neg q\}$ (8,5)	
		16. $\{q\}$ (10,3)	
		17. $\{\neg r, \neg s, \neg t\}$ (6,9)	
		18. $\{\neg q, \neg s\}$ (6,10)	
		19. $\{\neg p\}$ (7,10)	
		20. $\{\neg p, \neg t\}$ (8,9)	

$\varphi(q) = 1, \varphi(p) = 0, \varphi(s) = 0, \varphi(r) = 1, \varphi(t) = 1$







Hilberts-10-Problem: Gibt es ein Verfahren, um zu entscheiden, ob eine diophantische Gleichung eine Lösung hat?

- ▶ Hat ein Polynom in mehreren Veränderlichen mit Koeffizienten in  $\mathbb{Z}$  eine Lösung in  $\mathbb{Z}$  ?

$$\exists x, y, z. 3x^2 - 2xy^3 + 5z^3 + 6 = 0$$

- Entscheide effektiv, ob eine Aussage in  $\mathbb{Z}$  gilt.

- ▶ Beschreibung von Eigenschaften:

$$U(x) \equiv \exists z. x = 2 \cdot z + 1, \quad G(x) \equiv \exists z. x = 2 \cdot z$$

$$\forall x(U(x) \vee G(x))$$





# Interpretationen

$$1. \quad D = \mathbb{N} \quad a = 0, b = 1 \quad f(x) = x \dot{-} 1 \\ g(x, y) = x \cdot y \quad p(x) \equiv x > 0$$

► Gibt es eine Funktion  $F : \mathbb{N} \rightarrow \mathbb{N}$  :

$$F(0) = 1 \qquad F(x) = x \cdot F(x \dot{-} 1) \quad (x > 0)$$

$$2. \quad D = \mathbb{N} \quad a = 0, b = 1 \quad f(x) = x \\ g(x, y) = y + 1 \quad p(x) \equiv x > 0$$

► Gibt es eine Funktion  $F : \mathbb{N} \rightarrow \mathbb{N}$  :

$$F(0) = 1 \qquad F(x) = F(x) + 1 \quad (x > 0)$$

# Allgemeine Sprache der Prädikatenlogik zweiter Stufe

## Definition 3.2 (Syntax)

### (a) Alphabet:

1 **Wahrheitswerte:**  $W, F$  (Log-Konstanten)

2 **Logische Symbole:**

2.1 **Junktoren:**  $\neg, \rightarrow, \wedge, \vee, \leftrightarrow, \dots$ , **If - Then - Else -**

2.2 **Operatoren:** **=, if - then - else -**

2.3 **Quantoren:**  $\forall$  (Allquantor),  $\exists$  (Existenzquantor)

3 **Variablensymbole:**

3.1  $n$ -stellige **Funktionsvariablen:**  $F_j^n (j \geq 1, n \geq 0)$   
 $n = 0$  **Individuenvariablen:** Bezeichnung  $x_j$

3.2  $n$ -stellige **Prädikatenvariablen:**  $P_j^n (j \geq 1, n \geq 0)$   
 $n = 0$  **aussagenlogische Variablen**









## Bemerkung 3.3

- a) **Term, Form** sind rekursiv entscheidbar, zusammengesetzte Terme und Formeln lassen sich eindeutig zerlegen. Freie und gebundene Vorkommen lassen sich effektiv bestimmen.
  
- b) **Vereinbarungen: Stelligkeit aus Kontext**  
*a, b, c* Individuenkonstanten, *f, g, h, ...* Funktionskonstanten  
*p, q, r ...* Prädikatskonstanten, *x, y, z ...* Individuenvariablen  
*F, G, H, ...* Funktionsvariablen *P, Q, R, ...*  
Prädikatsvariablen, *A, B, C, ...* Formeln, *t, s* Terme.  
Die Mengen *Var(t)*, *Var(A)* seien die Variablen, die in *t* bzw. in *A* vorkommen.





# Eingeschränkte Teilsprachen

## Konstanten, Variablen einschränken

### 1. Allgemeine Sprache der Prädikatenlogik erster stufe: PL1

- ▶ Nur Individuenvariablen  $x_j$   
(keine Funktion- und Prädikatsvariablen)

$$x_1 \neq x_2 \wedge \forall x_2 (\exists x_3 p(x, f(x_2, x_3)) \rightarrow (p(x_2, x_1) \vee p(x_2, a)))$$

### 2. ▶ Sprache der Gleichheitslogik

Nur Individuenvariablen  $x_j$  ( $j \geq 1$ ), Funktionskonstanten.  
Keine Prädikatskonstanten und Variablen, keine  
Funktionsvariablen.

Oft noch eingeschränkt, nur Individuenkonstanten

- ▶ reine Gleichheitslogik

**Term** :  $x_i, a_i, \text{if } A \text{ then } t_1 \text{ else } t_2$

**AForm** :  $W, F, t_1 = t_2$

$$\forall x_1 \forall x_2 \forall x_3 (((x_1 = x_2) \wedge (x_2 = a)) \rightarrow (x_1 = a))$$





# Semantik der P-Logik 2-Stufe

## Interpretationen, Belegungen, Bewertungen

- !  $D \neq \emptyset$ , Funktionen  $f : D^n \rightarrow D$  (totale Funktion),
- ! Prädikate  $P \subseteq D^n$  (als Relationen)
- ! oder  $P : D^n \rightarrow \{0, 1\}$
- ! 0-stellige Funktionen (Element aus  $D$ ),
- ! Prädikate (Element aus  $\{0, 1\}$ ).

## Definition 3.4 (Interpretationen, Belegungen)

Sei  $\mathcal{L}$  Sprache der P-Logik 2-Stufe

(festgelegt durch die Menge der Konstantensymbole i. R. endlich).

a) Eine **Interpretation**  $I$  für  $\mathcal{L}$  ist ein Tripel  $I = (D, I_c, I_v)$  mit

- ▶  $D \neq \emptyset$  Individuenbereich (Definitionsbereich).
- ▶  $I_c$  ist eine Interpretation (Belegung) der Konstanten  
 $f^n \in \mathcal{L}$ , so  $I_c(f^n) : D^n \rightarrow D$   
 $p^m \in \mathcal{L}$ , so  $I_c(p^m) \subseteq D^m$  (oder  $I_c : D^m \rightarrow \{0, 1\}$ )
- ▶  $I_v$  ist eine Belegung der Variablen:  
 $F^n$  Funktionsvariablen  $I_v(F^n) : D^n \rightarrow D$  ( $n \geq 0$ )  
 $P^m$  Prädikatsvariablen  $I_v(P^m) \subseteq D^m$  ( $D^m \rightarrow \{0, 1\}$ )

$(D, I_c)$  heißt auch **Relationalstruktur**.

Kommen keine Prädikatskonstanten vor, so **Algebra**.



## Interpretationen, Belegungen (Forts.)

- b) ii) ▶  $I(\neg A)$ ,  $I(A \wedge B)$ ,  $I(A \vee B)$ ,  $I(A \rightarrow B)$ ,  $I(A \leftrightarrow B)$ ,  
 $I(\text{If } A \text{ Then } B \text{ Else } C)$

Wie in A-Logik.

- ▶  $I((\forall x)A) = \begin{cases} 1 & \text{falls für alle } d \in D \text{ gilt } I^{x,d}(A) = 1 \\ 0 & \text{sonst} \end{cases}$
- ▶  $I((\exists x)A) = \begin{cases} 1 & \text{falls es } d \in D \text{ gibt mit } I^{x,d}(A) = 1 \\ 0 & \text{sonst} \end{cases}$

$$I^{x,d} = (D, I_c, I'_v), \quad I'_v(y) = \begin{cases} d & y \equiv x \\ I_v(y) & \text{sonst} \end{cases}$$

- ▶ Entsprechend für Quantifizierungen mit den anderen Funktions- und Prädikatsvariablen.

## Interpretationen, Belegungen (Forts.)

Jede Interpretation  $I = (D, I_c, I_v)$  induziert durch 3 (i) und ii)) eine Bewertung aller Terme und Formeln, die die bewerteten Konstanten und Variablen als freie Variablen enthalten.

Umgekehrt wird jede Bewertung  $I$ , die i) und ii) genügt eindeutig durch eine solche Interpretation induziert.

**Beachte:** 3-Parameter:  $D, I_c, I_v$ .

# Interpretationen, Belegungen (Forts.)

- c) Gilt  $I(A) = 1$ , so ist  $A$  wahr in der Interpretation  $I$  oder  $I$  **erfüllt**  $A$ .

Schreibweise  $\models_I A$  oder  $I \models A$

(Beachte  $A$  ist hier eine beliebige Formel, kann also freie Variablen enthalten).

## Folgerungen

### Bemerkung 3.5

- ▶ *Um die Bewertung einer Formel  $A$  zu bestimmen, genügt es die Bewertung der in ihr vorkommenden Konstanten und frei vorkommenden Variablen zu kennen!!*

$$I = (\underline{D}, \underline{I}_c, I_v)$$

- ▶ *Insbesondere: Ist  $A$  abgeschlossen, so genügt es Interpretationen der Form  $(D, I_c)$ , d. h. Definitionsbereich und Belegung der Konstanten zu betrachten.*
- ▶ *Seien  $I_1, I_2$  Interpretationen mit  $D_1 = D_2$  und  $A$  eine Formel. Stimmen  $I_1$  und  $I_2$  auf allen Konstanten und freien Variablen, die in  $A$  vorkommen, überein, so gilt  $I_1(A) = I_2(A)$ .*

## Beispiel

### Beispiel 3.6

i)  $\exists x \forall y (p(y) \rightarrow x = y)$

Stimmt in allen Interpretationen für die  $I(p)$  höchstens ein Element enthält. „Es gibt höchstens ein  $x$ , so dass  $p(x)$  wahr ist“.

ii) „Es gibt genau ein  $x$ , so dass  $p(x)$  wahr ist“.

$$\exists x [p(x) \wedge \forall y [p(y) \rightarrow x = y]]$$

iii)  $\forall z \exists u \exists v ((z = u \vee z = v) \wedge u \neq v)$   
 $\wedge \forall x \forall y \forall P [x \neq y \vee P(x, x) \vee \neg P(y, y)]$

- ▶ Wahr in jeder Interpretation mit  $|D| \geq 2$
- ▶ Falsch in jeder Interpretation mit  $|D| = 1$



# Definition

## Definition 3.7

Sei  $\mathcal{L}$  Sprache der P-Logik 2-Stufe

- a)  $A \in \mathbf{Form}$  heißt **allgemeingültig**, falls  $I(A) = 1$  für jede Interpretation  $I$  für  $\mathcal{L}$ .  
**Schreibweise:**  $\models A$
- b)  $A \in \mathbf{Form}$  heißt **erfüllbar**, falls es eine Interpretation  $I$  für  $\mathcal{L}$  gibt mit  $I(A) = 1$ .  $I$  heißt auch **Modell** für  $A$  (nur für abg.  $A$ ).  
Gibt es keine solche Interpretation, so heißt  $A$  **unerfüllbar**.
- c)  $\Sigma \subseteq \mathbf{Form}$  heißt **erfüllbar**, falls es eine Interpretation  $I$  für  $\mathcal{L}$  gibt, die alle Formeln  $A \in \Sigma$  erfüllt.

## Einfache Folgerungen

### Bemerkung 3.8

*A allgemeingültig gdw  $\neg A$  unerfüllbar.*

*Es genügt Interpretationen zu betrachten, die die Konstanten und freien Variablen der Formel A belegen.*

*- unendlich viele, da  $D \neq \emptyset$  beliebig.*

### Bemerkung 3.9

*Es gibt allgemeingültige Formeln: **Tautologie-Theorem:***

*Sei  $A(p_1, \dots, p_n)$  eine Formel der Aussagenlogik in A-Variablen  $p_1, \dots, p_n$ .  $A'$  entstehe aus A durch simultane Ersetzung von  $p_i$  durch  $B_i \in \mathbf{Form}$ . Dann  $A' \in \mathbf{Form}$ .*

*Ist A Tautologie, so ist A' allgemeingültig.*

*(z. B.  $A_1 \vee \neg A_1, A_1 \rightarrow (A_2 \rightarrow A_1) \dots$ )*

# Einfache Folgerungen

## Bemerkung 3.10

i)  $\forall x \forall y \forall P (x \neq y \vee (P(x, x) \vee \neg P(y, y)))$  ist allgemeingültig.  
 Es genügt Interpretationen mit  $I = (D) \quad D \neq \emptyset$  zu betrachten.

$|D| = 1$  ok,  $|D| > 1$

$I_V(x, y, P), \quad x \rightarrow d_1, \quad y \rightarrow d_2$

$$d_1 \neq d_2 \text{ ok, } d_1 = d_2 \rightsquigarrow I_V(P)(d_1, d_2) = \begin{cases} 1 \\ 0 \end{cases}$$

ii)  $A \equiv \exists P \forall x \exists y (P(x, x) \wedge \neg P(x, y))$

Weder allgemeingültig noch unerfüllbar.

$|D| = 1 \rightsquigarrow I(A) = 0, \quad |D| \geq 2 \rightsquigarrow I(A) = 1$

iii) Allgemeingültig sind:

$t = t, \quad \forall x A \leftrightarrow \neg \exists x (\neg A), \quad \exists x A \leftrightarrow \neg \forall x (\neg A)$



# Ordnungsrelationen

## Bemerkung 3.11 (Eigenschaften von Ordnungsrelationen:)

*p 2-stellige P-Konstante*

$A_1 \equiv \forall x \forall y \forall z ((p(x, y) \wedge p(y, z)) \rightarrow p(x, z))$  *Tra.*

$A_2 \equiv \forall x \forall y (p(x, y) \vee p(y, x) \vee x = y)$  *Trichot.*

$A_3 \equiv \forall x \neg p(x, x)$  *Antireflex.*

$A_4 \equiv \forall x \forall y (p(x, y) \rightarrow \exists z (p(x, z) \wedge p(z, y)))$  *dicht*

$A_5 \equiv \forall x \exists y p(x, y)$  *ohne letztes Elem.*

$A_6 \equiv \forall x \exists y p(y, x)$  *ohne erstes Elem.*

*Keine der Formeln ist allgemeingültig! (Überzeugen Sie sich)*

*Sie sind erfüllbar:  $I_1 = (\{0, 1, 2\}, <)$*

*$I_2 = (\mathbb{N}, <)$*

*$I_3 = ([0, 1], <)$*

*$I_4 = (\mathbb{Q}, <)$*

	$<$	0	1	2
0	F	W	W	
1	F	F	W	
2	F	F	F	





## Arithmetik

### Beispiel 3.12

Die Sprache der Arithmetik  $\mathbb{N}$

- ▶ Konstanten:  $0, S, +, \cdot$ ,  $I = (\mathbb{N}, 0, ', +, \cdot)(n' = n + 1)$
- ▶ Stelligkeiten 0, 1, 2, 2
  1.  $\forall x \forall y (S(x) = S(y) \rightarrow x = y)$
  2.  $\forall x \quad S(x) \neq 0$
  3.  $\forall x \quad x + 0 = x$
  4.  $\forall x \forall y (x + S(y)) = S(x + y)$
  5.  $\forall x \quad x \cdot 0 = 0$
  6.  $\forall x \forall y \quad x \cdot S(y) = (x \cdot y) + x$
  7.  $\forall P [(P(0) \wedge \forall x (P(x) \rightarrow P(S(x)))) \rightarrow \forall x P(x)]$
- ▶ Sind gültig in  $I$ .  
 Man beachte 7. ist Induktionsprinzip für Teilmengen von  $\mathbb{N}$ .  
 Es ist eine Formel der P-Logik 2-Stufe.

# Arithmetik Beispiel (Forts.)

Frage nach der **Axiomatisierbarkeit** der Arithmetik:

- ▶ Ist die Allgemeingültigkeit für Formeln einer Sprache  $\mathcal{L}$  entscheidbar?
- ▶ Rekursiv aufzählbar?
- ▶ Welche effektiven Methoden gibt es?

# Allgemeingültigkeit: Entscheidbare Fälle

## Lemma 3.13

*Die Allgemeingültigkeit für Formeln der quantifizierten A-Logik ist entscheidbar.*

### Beweis:

Methode der **Quantorenelimination**: Finde zu Formel der Q-A-Logik eine logisch äquivalente der A-Logik.

(**Problemreduktion!**)

$$(\forall P_i^0)B \leftrightarrow B_{P_i^0}[W] \wedge B_{P_i^0}[F] \qquad (P_i^0 \leftarrow W, P_i^0 \leftarrow F)$$

$$(\exists P_i^0)B \leftrightarrow B_{P_i^0}[W] \vee B_{P_i^0}[F]$$

$$I( \quad ) = I( \quad )$$

- ▶ Nach Transformation bleibt eine Formel der A-Logik: Entscheide, ob diese eine Tautologie ist.

# Allgemeingültigkeit: Entscheidbare Fälle (Forts.)

## Beispiel 3.14

$\forall P \exists Q ((P \leftrightarrow \neg Q) \vee (p \rightarrow Q))$       ist Allgemeingültig

$\rightsquigarrow$

$\exists Q ((W \leftrightarrow \neg Q) \vee (p \rightarrow Q)) \wedge \exists Q ((F \leftrightarrow \neg Q) \vee (p \rightarrow Q))$

$\rightsquigarrow$

$[((W \leftrightarrow \neg W) \vee (p \rightarrow W)) \vee ((W \leftrightarrow \neg F) \vee (p \rightarrow F))] \wedge [(F \leftrightarrow \neg W) \vee (p \rightarrow W) \vee ((F \leftrightarrow \neg F) \vee (p \rightarrow F))]$

$\rightsquigarrow W \wedge W \quad \rightsquigarrow W$

# Allgemeingültigkeit: Entscheidbare Fälle (Forts.)

## Ausblick

Andere:

- ▶ Gleichheitslogik
- ▶ Monadische P-Logik 1-Stufe mit =
- ▶ Monadische P-Logik 2-Stufe mit =
- ▶ Pressburgerarithmetik: Gültige PL1-Formeln in  $(\mathbb{N}, 0, ', +, <)$
- ▶ Syntaktisch eingeschränkte Formelklassen  
 $\forall( \quad ), \forall\exists, \exists\forall, \dots?$

# Transformationen von Termen und Formeln

## Einschränkung auf PL1

### Definition 3.15 (Substitution)

$A \in \mathbf{Form}$ ,  $t, \hat{t} \in \mathbf{Term}$ ,  $x$  Individuen-Variable.

- ▶ **Substitution** von  $x$  durch  $t$  in  $A$  bzw. (in  $\hat{t}$ ):
  - $A_x[t]$  ( $\hat{t}_x[t]$ ) ist die Formel (der Term), die aus  $A$  (bzw.  $\hat{t}$ ) entsteht, wenn man jedes **freie** Vorkommen von  $x$  in  $A$  (bzw.  $\hat{t}$ ) durch  $t$  ersetzt.
- ▶ Analog **simultane Substitutionen**
  - $A_{x_1, \dots, x_n}[t_1, \dots, t_n]$  bzw.  $t_{x_1, \dots, x_n}[t'_1, \dots, t'_n]$

## Transformationen von Termen und Formeln (Forts.)

- ▶ Allgemeiner ist eine **Substitution** durch  $\sigma : \{x_i : i \in \mathbb{N}\} \rightarrow \mathbf{Term}$  gegeben.  
 $\sigma(A)$ ,  $\sigma(t)$  können entsprechend durch Induktion über den Aufbau der Formeln bzw. Terme definiert werden.
- ▶ Die Substitution heißt **erlaubt**, falls kein Vorkommen einer Variablen in  $t$  bzw.  $t_i$  nach der Substitution in  $A$  gebunden vorkommt.
- Dies ist der Fall z. B. wenn die Variablen von  $t$  nicht in  $A$  vorkommen.  
(Kann durch Umbenennung der gebundenen Variablen erreicht werden).

## Beispiel 3.16 (Substitutionen)

$$A \equiv \exists y \ x = 2 \cdot y \qquad t \equiv y + 1$$

▶  $A_x[t] \equiv \exists y \ y + 1 = 2 \cdot y$

keine erlaubte Substitution

▶  $A_y[t] \equiv \exists y \ x \equiv 2 \cdot y$

da keine freie Vorkommen

▶  $t_y[t] \equiv (y + 1) + 1$

▶ Wie Ändert sich die Bedeutung einer Formel bei Substitutionen?

●  $I = (\mathbb{N}, +, \cdot) \quad x \leftarrow 3 \quad y \leftarrow 2 \quad t \leftarrow 3 \quad I(x) = I(t)$

▶ Ersetzt man  $x$  durch  $t$ , sollte sich die Bedeutung einer Formel nicht verändern.

●  $I(A) = I(\exists y \ x = 2 \cdot y) = 0$

●  $I(A_x[t]) = I(\exists y \ y + 1 = 2y) = 1$  (keine erlaubte Substitution)

Betrachte die Formel:

●  $B \equiv \forall x(P(x, y) \rightarrow Q(x)) \quad t \equiv f(y, z)$

↪  $B_y[t] \equiv \forall x(P(x, f(y, z)) \rightarrow Q(x))$  erlaubte Substitution.

### Lemma 3.17 (Substitutionslemma)

Sei  $A$  Term oder Formel,  $x$  Individuenvariable,  $t \in \mathbf{Term}$  und  $A_x[t]$  eine *erlaubte Substitution*. Dann gilt für jede Interpretation

$$I = (D, I_c, I_v)$$

- $I(A_x[t]) = I^{x, I(t)}(A)$

► Insbesondere: Sind  $I, I'$  Interpretationen, die sich höchstens für  $x$  unterscheiden und gilt  $I'(x) = I(t)$ , so gilt  $I'(A) = I(A_x[t])$ .

- **Beweis:**

Induktion über Aufbau von Formeln bzw. Terme. ■





## Semantischer Folgerungsbegriff

### Definition 3.19

Sei  $\mathcal{L}$  eine (Teil-)Sprache der PL2

$\Gamma \subseteq \mathbf{Form}, A, B \in \mathbf{Form}$

- a)  $A$  ist **logische Folgerung** aus  $\Gamma : \Gamma \models A$ . Wenn jede Interpretation, die  $\Gamma$  erfüllt auch  $A$  erfüllt ( $I(\Gamma) = 1 \rightsquigarrow I(A) = 1$ ).
  - ▶ Sei  $\text{Fol}(\Gamma) = \{A \in \mathbf{Form} : \Gamma \models A\}$ .
- b)  $A$  und  $B$  sind **logisch äquivalent**  $A \models\equiv B$ , falls  $A \models B$  und  $B \models A$ .  
(Lässt sich auf Mengen verallgemeinern!)

Bemerkung 3.20

- 1.  $\Gamma \models A$  gdw  $\{\Gamma, \neg A\}$  nicht erfüllbar.
- 2.  $\emptyset \models A$  gdw  $\models A$  (d.h.  $A$  ist allgemeingültig).
- 3.  $\Gamma$  nicht erfüllbar gdw  $\Gamma \models A$  für alle  $A \in \mathbf{Form}$ .
- 4.  $\Gamma \subset \Sigma, \Gamma \models A$ , so  $\Sigma \models A$ . (Monotonieeigenschaft)
- 5.  $\Gamma \models \equiv \Sigma$ , d. h.  $\Gamma \models B$  für  $B \in \Sigma$  und  $\Sigma \models C$  für  $C \in \Gamma$ .  
 Insbesondere  $\Gamma$  erfüllbar gdw  $\Sigma$  erfüllbar und  
 $\Gamma \models A$  gdw  $\Sigma \models A$ . Also  $\text{Folg}(\Gamma) = \text{Folg}(\Sigma)$ .
- 6.  $A \models \equiv B$  gdw  $\models A \leftrightarrow B$  gdw  $I(A) = I(B)$  für jede  
 Interpretation  $I$ .  
 $A \models \equiv B$  dann  $\Gamma \models A$  gdw  $\Gamma \models B$ .

## Beispiel 3.21

i)  $\forall x Q(x) \models Q(y)$

(Spezialfall von  $\forall x A \rightarrow A_x[t]$  allgemeingültig)

ii)  $A(y) \not\models \forall y A(y)$  ( $y$  kommt frei in  $A$  vor)

$$A(y) \equiv p(y), I = (\{0, 1\}, I(p)(x) \equiv (x = 0), y \leftarrow 0)$$

$$I(A(y)) = 1, \text{ jedoch } I(\forall y A(y)) = 0 \quad I(p)(1) = 0$$

iii)  $\models \exists x(p(x) \rightarrow \forall x p(x))$

Sei  $\mathcal{I} = (D, I(p))$  eine Interpretation, d. h.  $I(p) \subseteq D$

$I(\exists x(p(x) \rightarrow \forall x p(x))) = 1$  gdw es gibt  $d \in D$ , so dass

$$I' = (D, I(p), x \leftarrow d), I'(p(x) \rightarrow \forall x p(x)) = 1$$

gdw  $d \notin I(p)$  oder  $I'(\forall x p(x)) = 1$  für ein  $d \in D$

gdw es gibt ein  $d \in D$  mit  $d \notin I(p)$  oder  $I(p) = D$

iv) Beispiele für äquivalente Formeln

- $\neg\neg A \models A$
- $W \vee A \models A \vee W \models W$   
 $F \wedge A \models A \wedge F \models F$   
 $A \vee A \models A \quad A \wedge A \models A$
- $B \circ QvA \models Qv(B \circ A)$  Q Quantor, v nicht frei in B
- Z.B.  $B \vee QvA \models Qv(B \vee A)$
- $\forall vA \models \neg\exists v(\neg A), \exists vA \models \neg\forall v(\neg A)$
- $\forall xA(x) \rightarrow B \models \exists y(A(y) \rightarrow B)$   
falls y weder in A(x) noch in B frei vorkommt
- $\forall x(A \rightarrow B) \models \forall xA \rightarrow \forall xB$
- $\forall vA \models \forall yA_v[y] \quad \exists vA \models \exists yA_v[y]$   
Falls Sub. erlaubt und y nicht frei in A
- **Beachte:**  $\forall vB \models B \quad \exists vB \models B$   
Falls v nicht frei in B vorkommt.

## Satz 3.22 (Wichtige Sätze)

Sei  $\Gamma \subseteq \mathbf{Form}$ ,  $A, B \in \mathbf{Form}$ .

1. **Deduktionstheorem** •  $\Gamma, A \models B$  gdw  $\Gamma \models A \rightarrow B$
2. **Modus-Ponens-Regel** •  $\Gamma \models A, \Gamma \models A \rightarrow B$  so  $\Gamma \models B$
3. **Kontrapositionsregel** •  $\Gamma, A \models \neg B$  gdw  $\Gamma, B \models \neg A$
4. **Generalisierungstheorem**

*Kommt  $v \in \mathit{Var}$  in keiner Formel von  $\Gamma$  frei vor, so*

- $\Gamma \models A$  gdw  $\Gamma \models \forall v A$

*Insbesondere:  $A \models \forall v A$  bzw.  $\models A \rightarrow \forall v A$ ,*

*falls  $v$  nicht frei in  $A$  vorkommt.*

5. **Ersetzungstheorem**

Sei  $A' \in \mathbf{Form}$ ,  $A$  Teilformel von  $A'$ . Entsteht  $B'$  aus  $A'$  indem man einige Vorkommen der Teilformel  $A$  durch  $B$  ersetzt und gilt  $A \models B$ , so auch  $A' \models B'$ .

### Beispiel 3.23 (Anwendung der Sätze)

- a)  $\models \exists x \forall y A \rightarrow \forall y \exists x A$   
 gdw  $\underbrace{\exists x \forall y A}_{\text{Deduktionstheorem}} \models \forall y \exists x A$   
 gdw  $\exists x \forall y A \models \exists x A$       **Generalisierungstheorem**  
 gdw  $\neg \forall x \neg \forall y A \models \neg \forall x \neg A$       **Ersetzungstheorem**  
 gdw  $\forall x \neg A \models \forall x \neg \forall y A$       **Kontrapositionsregel**  
 gdw  $\forall x \neg A \models \neg \forall y A$       **Generalisierungstheorem**  
 gdw  $\{\forall x \neg A, \forall y A\}$       **nicht erfüllbar**
- b) **Variante des Ersetzungstheorems**  
 $A'$  entstehe aus  $A$  durch Substitution (erlaubte) einiger Vorkommen von  $x$  in  $A$  durch  $y$ . Dann gilt  
 $\models \forall x \forall y (x = y \rightarrow (A \leftrightarrow A'))$   
 (z. B.  $A \equiv f(x, y) = g(x)$        $A' \equiv f(y, y) = g(x)$ )

## Normalformen

### Definition 3.24 (Normalformen (Präfix Normalformen))

Eine Formel ist in **PKNF** (Pränex Konjunktiver NF), falls sie die Gestalt

$$\underbrace{(\Delta v_1) \cdots (\Delta v_n)}_{\text{Präfix}} \left\{ \underbrace{[A_{11} \vee \cdots \vee A_{1l_1}] \wedge \cdots \wedge [A_{m1} \vee \cdots \vee A_{ml_m}]}_{\text{Matrix}} \right\}$$

$\Delta \in \{\exists, \forall\}$ ,  $v_j$  Variablen, die in mindestens einem  $A_{kl}$  vorkommen und paarweise verschieden sind.

$A_{kl}$  Literale, d. h. atomare oder negierte atomare Formeln.

### Beispiel 3.25

$$\forall x \exists Q \forall y \{ [\neg p \vee x \neq a \vee x = b] \wedge [Q(y) \vee y = b] \}$$

# Normalformen

## Satz 3.26 (Verfahren PKNF)

Jede Formel  $A \in \mathbf{Form}$  lässt sich effektiv in eine logisch äquivalente Formel in PKNF (PDNF) transformieren.

(Beachte die Länge der Formel kann exponentiell in der Länge der ursprünglichen Formel wachsen. Dies kann man vermeiden, wenn man nur erfüllungsäquivalente Formeln benötigt!).

# Verfahren PKNF, PPDF

## Schritte:

1. Eliminiere überflüssige Quantoren.
2. Umbenennung gebundener Variablen.
3. Eliminiere logische Verknüpfungen und Operatoren.  
 $\rightarrow, \leftrightarrow, \mathbf{if} \dots, \mathbf{if} \dots$
4. NNF: Negation vor Atome.
5. Quantoren nach außen.
6. Matrix in KNF (DNF).

## Verfahren PKNF, PDNF - Beispiele

### Beispiel 3.27

$$\forall x [(\forall y p(x) \vee \forall z q(z, y)) \rightarrow \neg \forall y r(x, y)]$$

1.  $\forall x [(p(x) \vee \forall z q(z, y)) \rightarrow \neg \forall y r(x, y)]$
2.  $\forall x [(p(x) \vee \forall z q(z, y)) \rightarrow \neg \forall y' r(x, y')]$
3.  $\forall x [\neg(p(x) \vee \forall z q(z, y)) \vee \neg \forall y' r(x, y')]$
4.  $\forall x [(\neg p(x) \wedge \exists z \neg q(z, y)) \vee \exists y' \neg r(x, y')]$
5.  $\forall x \exists z \exists y' [(\neg p(x) \wedge \neg q(z, y)) \vee \neg r(x, y')]$
6.  $\forall x \exists z \exists y' [(\neg p(x) \vee \neg r(x, y')) \wedge (\neg q(z, y) \vee \neg r(x, y'))]$   
(Ist PKNF)
7. PDNF  $\forall x \exists z \exists y' [(\neg p(x) \wedge \neg q(z, y)) \vee \neg r(x, y')]$









# Beweis der Behauptung

:

„↪“: Angenommen  $A_S$  allgemeingültig, / Interpretation mit  $D = \{0, 1\}^*$ ,  $a \leftarrow \varepsilon$ ,  $f_0 : x \rightarrow x0$ ,  $f_1 : x \rightarrow x1$   
 $p(x, y)$  gdw  $x \equiv \alpha_{j_1} \cdots \alpha_{j_m}$ ,  $y \equiv \beta_{j_1} \cdots \beta_{j_m}$  für  $j_1, \dots, j_m$   
( $j_i \in [1 \dots n]$   $m > 0$ ).

„↪“: Angenommen PCP  $S$  habe Lösung  $j_1 \dots j_m$ , d. h.

$\alpha_{j_1} \alpha_{j_2} \cdots \alpha_{j_m} \equiv \beta_{j_1} \beta_{j_2} \cdots \beta_{j_m}$   
 $\rightsquigarrow p(f_{\alpha_{j_1} \alpha_{j_2} \cdots \alpha_{j_m}}(a), f_{\beta_{j_1} \beta_{j_2} \cdots \beta_{j_m}}(a))$  wahr,  
also ist  $A_S$  allgemeingültig.

! Es gibt weitere Unentscheidbarkeitsresultate die wir später noch behandeln werden. Es sei jedoch erwähnt, dass die Grenzen zwischen den entscheidbaren und unentscheidbaren Fälle der Allgemeingültigkeit sehr genau bekannt sind. (Siehe etwa Börger).



## Hauptsätze der Prädikatenlogik erster Stufe (Forts.)

b) **Kompaktheitssatz für PL1**: Sei  $\Sigma \subseteq \mathbf{Form}(\mathcal{L})$ .  
 $\Sigma$  erfüllbar gdw jede endliche Teilmenge von  $\Sigma$  ist erfüllbar.

c) Insbesondere:  $\Sigma \subseteq \mathbf{Form}(\mathcal{L}), A \in \mathbf{Form}(\mathcal{L})$ :

$\Sigma \models A$  gdw es gibt  $\Sigma_0 \subseteq \Sigma, \Sigma_0$  endlich :  $\Sigma_0 \models A$

## Hauptsätze der Prädikatenlogik erster Stufe (Forts.)

d) **Satz von Löwenheim-Skolem:**

$\Sigma \subseteq \mathbf{Form}(\mathcal{L})$  erfüllbar gdw es gibt eine Interpretation  $I = (D, I_c, I_v)$ , wobei  $D$  abzählbar oder endlich ist, die  $\Sigma$  erfüllt.

(  $D$  kann als eine Termmenge über die konstanten Symbole gewählt werden).

**Beachte jedoch: nicht gültig für PL2**



# Deduktive Systeme für PL1

## Definition 4.5 (Deduktive Systeme für PL1)

Sei  $\mathcal{L}$  Sprache 1-Stufe mit Formeln in  $\neg, \rightarrow, \forall, =$ .  $\mathcal{F} = (\mathbf{Ax}, \mathbf{R})$  bestimmt durch Axiomenmenge  $Ax$  (Axiomenschema) und Menge  $R$  von Regeln (Regelschema).

- ▶  $Ax$  enthält **alle Generalisierungen** von folgenden durch Schemata beschriebenen Formelmengen:

$$\mathbf{Ax1:} \quad A \rightarrow (B \rightarrow A)$$

$$\mathbf{Ax2:} \quad (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$$

$$\mathbf{Ax3:} \quad (\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B)$$

$$\mathbf{Ax4:} \quad \forall x A \rightarrow A_x[t], \quad \text{falls } A_x[t] \text{ erlaubt}$$

## Deduktive Systeme für PL1 (Forts.)

**Ax5:**  $\forall x (A \rightarrow B) \rightarrow (\forall x A \rightarrow \forall x B)$

**(Ax5':**  $\forall x (A \rightarrow B) \rightarrow (A \rightarrow \forall x B)$   $x$  nicht frei in  $A$ )

**Ax6:**  $A \rightarrow \forall x A$ , falls  $x$  nicht frei in  $A$  vorkommt

**Ax7:**  $x = x$

**Ax8:**  $x = y \rightarrow (A \rightarrow A')$ , wobei  $A'$  aus  $A$  durch Ersetzen einiger freier Vorkommen von  $x$  durch  $y$  (erlaubt)

- ▶  $R$  enthält alle Regeln, die vom **Regelschema Modus Ponens**

**MP**  $\frac{A, A \rightarrow B}{B}$  Modus Ponens beschrieben werden.

## Deduktive Systeme für PL1 (Forts.)

- **Alternatives** Deduktionssystem  $\mathcal{F}' = (\mathbf{Ax}, \mathbf{R}')$
- ▶  $R'$  enthält MP-Regel und **Generalisierungsregel**.
- GR**  $\frac{A}{\forall xA}$  Generalisierung (ohne Einschränkungen)
- ! **Beachte:**  $Ax$  enthält nur allgemeingültige Formeln. MP und GR-Regel führen nicht aus der Menge der allgemeingültigen Formeln hinaus.

## Ziel

- $\vdash_{\mathcal{F}} A$  gdw  $\vdash_{\mathcal{F}'} A$  gdw  $\models A$   
 $\rightsquigarrow$   $\rightsquigarrow$  Korrektheit
- $\Sigma \vdash_{\mathcal{F}} A$  gdw  $\Sigma \models A$   
 $\rightsquigarrow$  Korrektheit
- $\Sigma \vdash_{\mathcal{F}} A$  , so  $\Sigma \vdash_{\mathcal{F}'} A$  Umkehrung i. Allg. nicht
- $\Sigma \vdash_{\mathcal{F}'} A \not\leftrightarrow \Sigma \vdash_{\mathcal{F}} A$  (**gilt nur für  $\Sigma$  Abg. Formeln**).

z. B.  $p(x) \vdash_{\mathcal{F}'} \forall x p(x)$  aber  $p(x) \not\models \forall x p(x)$

- ▶ **Bemerkung:** Alle Tautologien (taut. Theorem) sind herleitbar in  $\mathcal{F}$ , d. h. Theoreme von  $\mathcal{F}$ .

# Beispiele

## Beispiel 4.6

Verwende  $\exists y A$  als Abkürzung für  $\neg\forall y \neg A$

$$1. \vdash_{\mathcal{F}} \forall x (p(x) \rightarrow \exists y p(y))$$

$$B_1 \equiv \forall x [(\forall y \neg p(y) \rightarrow \neg p(x)) \rightarrow (p(x) \rightarrow \neg\forall y \neg p(y))] \quad (\text{Ax3, Gen})$$

$$B_2 \equiv \forall x ((\forall y \neg p(y) \rightarrow \neg p(x)) \rightarrow (p(x) \rightarrow \neg\forall y \neg p(y))) \rightarrow [\forall x (\forall y \neg p(y) \rightarrow \neg p(x)) \rightarrow \forall x (p(x) \rightarrow \neg\forall y \neg p(y))] \quad (\text{Ax5})$$

$$B_3 \equiv \forall x (\forall y \neg p(y) \rightarrow \neg p(x)) \rightarrow \forall x (p(x) \rightarrow \neg\forall y \neg p(y)) \quad (\text{MP})$$

$$B_4 \equiv \forall x (\forall y \neg p(y) \rightarrow \neg p(x)) \quad (\text{Ax4, Gen})$$

$$B_5 \equiv \forall x (p(x) \rightarrow \exists y p(y)) \quad (\text{MP})$$

## Beispiele (Forts.)

$$2. \underset{\mathcal{F}}{\vdash} \forall x A \rightarrow \exists x A$$

Beweis:

$$\begin{array}{ll} \vdash \forall x \neg A \rightarrow \neg A & (\text{Ax4}) \\ \vdash (\forall x \neg A \rightarrow \neg A) \rightarrow (A \rightarrow \neg \forall x \neg A) & (\text{Ax3}) \\ \vdash A \rightarrow \neg \forall x \neg A & (\text{MP}) \\ \vdash \forall x A \rightarrow A & (\text{Ax4}) \\ \vdash (\forall x A \rightarrow A) \rightarrow & \\ \quad ((A \rightarrow \neg \forall x \neg A) \rightarrow (\forall x A \rightarrow \neg \forall x \neg A)) & (\text{Taut}) \\ \vdash (A \rightarrow \neg \forall x \neg A) \rightarrow (\forall x A \rightarrow \neg \forall x \neg A) & (\text{MP}) \\ \vdash \forall x A \rightarrow \exists x A & (\text{MP}) \end{array}$$

## Beispiele (Forts.)

$$3. \vdash t = t \quad \vdash (x = y) \rightarrow (y = x)$$

$$\vdash ((x = y) \rightarrow ((y = z) \rightarrow (x = z)))$$

Folgen aus  $Ax, \forall x(x = x), (x = y) \rightarrow A(x) \leftrightarrow A(y)$  für  $A$  (erlaubt).

$$\vdash ((t_1 = t'_1) \wedge \dots \wedge (t_n = t'_n)) \rightarrow (A(t_1, \dots, t_n) \leftrightarrow$$

$A(t'_1, \dots, t'_n)$ ), wobei  $A(x_1, \dots, x_n)$  eine Formel mit

mindestens  $n$ -freien Variablen und Substitutionen erlaubt

$(x_i \leftarrow t_i$  bzw.  $x_i \leftarrow t'_i)$ .

Spezialfall:

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

# Beweisidee

- $\forall x[(x = y) \rightarrow (A(x) \leftrightarrow A(y))]$
- $\forall x[(x = y) \rightarrow (A(x) \leftrightarrow A(y))] \rightarrow ((t = y) \rightarrow (A(t) \leftrightarrow A(y)))$
- $(t = y) \rightarrow (A(t) \leftrightarrow A(y))$
- $\forall y(t = y) \rightarrow (A(t) \leftrightarrow A(y))$
- $(t = t') \rightarrow (A(t) \leftrightarrow A(t'))$

# Deduktionstheorem - Generalisierungstheorem

## Satz 4.7 (Hauptsätze für $\mathcal{F}$ und $\mathcal{F}'$ )

Seien  $\Gamma \subseteq \mathbf{Form}$ ,  $A, B \in \mathbf{Form}$ .

### a) Deduktionstheorem

- 1)  $\Gamma \vdash_{\mathcal{F}} A \rightarrow B$  gdw  $\Gamma, A \vdash_{\mathcal{F}} B$
- 2)  $\Gamma \vdash_{\mathcal{F}'} A \rightarrow B$  gdw  $\Gamma, A \vdash_{\mathcal{F}'} B$ , falls die Generalisierung nicht auf eine in  $A$  frei vorkommende Variable angewandt wurde.
- 3)  $\Gamma, A \vdash_{\mathcal{F}'} B$  gdw  $\Gamma \vdash_{\mathcal{F}'} \tilde{A} \rightarrow B$ ,  
wobei  $\tilde{A}$  ein universeller Abschluss von  $A$  ist.

### b) Generalisierungstheorem:

- 1) Falls  $\Gamma \vdash_{\mathcal{F}} A$  und  $x$  nicht frei in  $\Gamma$  vorkommt, so  $\Gamma \vdash_{\mathcal{F}} \forall x A$
- 2)  $\Gamma \vdash_{\mathcal{F}'} A$  gdw  $\Gamma \vdash_{\mathcal{F}'} \forall x A$

### c) Kontrapositionstheorem: Für $\mathcal{F}$ und $\mathcal{F}'$

- ▶  $\Gamma, A \vdash \neg B$  gdw  $\Gamma, B \vdash \neg A$

# Deduktionstheorem - Generalisierungstheorem (Forts.)

- ↪ Es gelten somit für die hier vorgestellten prädikatenlogischen Systeme die für das deduktive System der Aussagenlogik entsprechenden Sätze. Vorsicht muss man beim System  $\mathcal{F}'$  mit dem Deduktionstheorem haben, da wir dafür eine allgemeinere Generalisierungsregel zugelassen haben die semantisch nicht immer korrekt ist.
- ▶ Hinzu kommt das Generalisierungstheorem in den zwei Varianten.

# Beweis Deduktionstheorem

1. „ $\Leftarrow$ “ Aus  $\Gamma \vdash A \rightarrow B$  folgt auch  $\Gamma, A \vdash A \rightarrow B$ . Da auch  $\Gamma, A \vdash A$  gilt, folgt  $\Gamma, A \vdash B$ , wegen MP ( $\mathcal{F}$  und  $\mathcal{F}'$ ).

„ $\Rightarrow$ “ Ang.  $\Gamma, A \underset{\mathcal{F}}{\vdash} B$ . Behauptung:  $\Gamma \underset{\mathcal{F}}{\vdash} A \rightarrow B$

- Induktion über Beweislänge: Axiom oder Hypothese

$$\Gamma \vdash B \quad (\text{Ax})$$

$$\Gamma \vdash B \rightarrow (A \rightarrow B) \quad (\text{Ax})$$

$$\Gamma \vdash A \rightarrow B \quad (\text{MP})$$

- ▶ Schritt ist MP-Schritt

Schritt $j$ :	$\Gamma, A \vdash C$	IV: $\Gamma \vdash A \rightarrow C$
---------------	----------------------	-------------------------------------

$\vdots \quad \vdots$

Schritt $k$ :	$\Gamma, A \vdash C \rightarrow B$	IV : $\Gamma \vdash A \rightarrow (C \rightarrow B)$
---------------	------------------------------------	--

$\vdots \quad \vdots$

Schritt $n + 1$ :	$\Gamma, A \vdash B$	Ax2+MP $\Gamma \vdash A \rightarrow B$
-------------------	----------------------	---

## Beweis Deduktionstheorem in $\mathcal{F}'$

2. In  $\mathcal{F}'$  „ $\forall$ “ Generalisierungsregel

$$\Gamma, A \vdash C$$

$$\vdots$$

$$\Gamma, A \vdash \forall x C \quad x \text{ nicht frei in } A$$

Dann IV:

$$\Gamma \vdash A \rightarrow C$$

$$\Gamma \vdash \forall x(A \rightarrow C) \quad (\text{Gen})$$

$$\Gamma \vdash \forall x(A \rightarrow C) \rightarrow (A \rightarrow \forall x C) \quad (\text{Ax5})$$

$x$  nicht frei in  $A$

$$\Gamma \vdash A \rightarrow \forall x C$$

### Definition 4.8

Sei  $\Gamma \subseteq \mathbf{Form}$ ,  $\Gamma$  heißt **konsistent**, falls es kein  $A \in \mathbf{Form}$  gibt, mit

$$\Gamma \vdash_{\mathcal{F}} A \text{ und } \Gamma \vdash_{\mathcal{F}} \neg A.$$

## Konsistenz im deduktiven System

### Bemerkung 4.9

- ▶  $\Gamma$  ist konsistent gdw jede endliche Teilmenge von  $\Gamma$  ist konsistent.
- ▶ Ist  $\Gamma$  inkonsistent, dann gilt  $\Gamma \vdash_{\mathcal{F}} A$  für jede Formel  $A$ .
- ▶  $\Gamma \cup \{\neg A\}$  inkonsistent gdw  $\Gamma \vdash A$ .
- ▶  $\Gamma \cup \{A\}$  inkonsistent gdw  $\Gamma \vdash \neg A$ .
- ▶ Ist  $\Gamma$  inkonsistent, so ist  $\Gamma$  nicht erfüllbar: Sei nämlich  $A$  mit  $\Gamma \vdash A$  und  $\Gamma \vdash \neg A$ .  $I$  Interpretation, die  $\Gamma$  erfüllt (wegen  $\Gamma \models A$  und  $\Gamma \models \neg A$ ) folgt aber  $I$  erfüllt  $\{A, \neg A\}$  ↯
- ▶ Die Menge der allgemeingültigen Formeln ist konsistent.
- ▶ Die Menge der Theoreme von  $\mathcal{F}$  ( $\mathcal{F}'$ ) ist konsistent.

# Vollständigkeit der Axiomatisierung

## Satz 4.10 (Gödel)

*Vollständigkeit der Axiomatisierung*

Seien  $A \in \mathbf{Form}$ ,  $\Sigma \subseteq \mathbf{Form}$ , dann gilt:

- a)  $\models A$  gdw  $\vdash_{\mathcal{F}} A$  gdw  $\vdash_{\mathcal{F}'} A$ .
- b)  $\Sigma$  konsistent gdw  $\Sigma$  erfüllbar.
- c)  $\Sigma \vdash_{\mathcal{F}} A$  gdw  $\Sigma \models A$ .

Beweis:

Siehe Yashuhara oder Enderton. ■

# Theorien erster Stufe

## Definition 4.11

Sei  $\mathcal{L}$  eine Sprache erster Stufe (fixiert durch die Funktions- und -Prädikatskonstanten).  $\Gamma \subseteq \mathbf{Form}(\mathcal{L})$  heißt **logische Theorie erster Stufe**, falls  $\Gamma$  abgeschlossen ist gegenüber logischer Folgerung, d. h.  $A \in \mathbf{Form} \quad \Gamma \models A$ , so  $A \in \Gamma$ .

- ▶ **Beachte:** Alternative Definitionen in Literatur:
  - $\Gamma$  Theorie, falls  $\Gamma$  abgeschlossen gegen MP und Generalisierung.
  - $\Gamma$  Menge **abgeschlossener Formeln**, abgeschlossen gegen logische Folgerung.
- ▶  $T$  als generische Bezeichnung für Theorien.

## Theorien erster Stufe (Forts.)

### Bemerkung 4.12

Sei  $\mathcal{L}$  Sprache 1-Stufe.

- a)  $T_{\mathcal{L}} = \{A \mid A \in \mathbf{Form}(\mathcal{L}), \text{ allgemeingültig}\}$  ist Theorie. Sie ist in jeder Theorie über  $\mathcal{L}$  enthalten.
- b)  $T_{\Sigma} = \{A \mid A \in \mathbf{Form}(\mathcal{L}), \Sigma \models A\}$  für  $\Sigma \subseteq \mathbf{Form}(\mathcal{L})$  ist eine Theorie, **die von  $\Sigma$  erzeugte Theorie** oder durch die Axiome  $\Sigma$  definierte Theorie.
- c) Ist  $T$  eine Theorie, so  $T \vdash_{\mathcal{F}} A$  gdw  $A \in T$ .  
 $T$  inkonsistent gdw es gibt  $A$  mit  $A, \neg A \in T$ , d. h.  $T = \mathbf{Form}$ .

## Theorien erster Stufe (Forts.)

d) Sei  $\mathcal{R}$  Relationalsystem (Struktur) für  $\mathcal{L}$   $I = (D, I_c)$ . Dann ist  $T_{\mathcal{R}} = \{A \mid A \in \mathbf{Form}(\mathcal{L}), \mathcal{R} \models A\}$  eine Theorie:

**Die Theorie von  $\mathcal{R}$**  Schreibe auch:  $Th(\mathcal{R})$ .

- ▶ Verwende hier  $\mathcal{R} \models A$ , falls für jede Interpretation der Variablen gilt  $\mathcal{R}, I_v \models A$ . Insbesondere  $\mathcal{R} \models A$  gdw  $\mathcal{R} \models \hat{A}$ , wobei  $\hat{A}$  ein universeller Abschluss von  $A$  ist.

( $T_{\mathcal{R}} \models A$  zeige  $\mathcal{R} \models A$  klar, da  $\mathcal{R} \models T_{\mathcal{R}}$ )

↪ Insbesondere:

- $T_{\mathcal{R}}$  ist konsistent für jede Struktur  $\mathcal{R}$ .

e)  $T \subseteq \mathbf{Form}(\mathcal{L})$  ist Theorie gdw  $\{A \mid A \in \mathbf{Form}(\mathcal{L}), \models A\} \subseteq T$  und  $T$  ist abgeschlossen gegenüber MP.

## Theorien erster Stufe (Forts.)

### Definition 4.13

Sei  $T$  eine Theorie erster Stufe über  $\mathcal{L}$

- a)  $T$  heißt **vollständig**, falls für jede abgeschlossene Formel  $A$  gilt:  $A \in T$  oder  $\neg A \in T$ .
- b)  $T$  heißt **(endlich) rekursiv axiomatisierbar**, falls es eine (endliche) rekursive Teilmenge  $\Sigma \subseteq \mathbf{Form}$  gibt mit  $T_\Sigma = \{A \in \mathbf{Form}(\mathcal{L}) \mid \Sigma \models A\} = T$ .
- c)  $T$  heißt **entscheidbar**, falls  $T$  eine rekursiv entscheidbare Teilmenge von  $\mathbf{Form}$  ist.

**Fragen:** Finde (endliche) Axiomatisierungen wichtiger Theorien. Insbesondere wann gilt  $T_{\mathcal{R}} = T_\Sigma$  für  $\Sigma$  rekursiv.

# Folgerungen

## Bemerkung 4.14

- a)  $T_{\mathcal{R}}$  ist vollständig für jede Struktur  $\mathcal{R}$ .  
 $T_{\mathcal{R}}$  ist somit konsistent und vollständig.
- b)  $T$  erfüllbar ( $T$  hat eine Modell) gdw  $T$  konsistent.
- c)  $T$  ist rekursiv axiomatisierbar, so  $T$  rekursiv aufzählbar.
- d) Ist  $T$  vollständig, konsistent und rekursiv axiomatisierbar.  
Dann ist die Menge der Aussagen von  $T$  rekursiv entscheidbar.
  - $A$  abgeschlossen, so  $A$  oder  $\neg A$  in  $T$ .  
Da  $T$  rekursiv aufzählbar, findet man  $A$  oder  $\neg A$  in dieser Aufzählung effektiv.

## Folgerungen (Forts.)

e) *Ist  $T$  vollständig und konsistent, dann gilt  $T = T_{\mathcal{R}}$  für eine Struktur  $\mathcal{R}$ .*

*•  $\mathcal{R} \models T$  existiert, da  $T$  erfüllbar, d. h.  $T \subseteq T_{\mathcal{R}}$ .*

*Angenommen  $T \subsetneq T_{\mathcal{R}}$ . Dann gibt es abgeschlossene Formel  $A \in T_{\mathcal{R}}$  mit  $A \notin T$ . Da  $T$  vollständig ist, muss  $\neg A \in T$  gelten, d. h.  $A, \neg A \in \Gamma_{\mathcal{R}}$  ↯*

**Frage:** Wann ist  $T_{\Sigma}$  vollständig für rekursive  $\Sigma$ ?

↷ Entscheidbarkeit!

# Beispiele

## Beispiel 4.15

$Th(\mathbb{N})$  Theorie der natürlichen Zahlen.

$\mathcal{R} = \langle \mathbb{N}; 0, S, +, *, = \rangle$  natürliche Interpretation der Sprache der Arithmetik  
Konst.  $0, S, +, *$  Funktionskonstante  $n \in \mathbb{N}$ ,

$\tilde{n} \equiv S(S \cdots (S(0) \cdots))$  Schreibe auch:  $(S^n 0)$ .

### Die Sätze von Gödel:

- a)  $Th(\mathbb{N})$  ist nicht rekursiv entscheidbar.
  - b)  $Th(\mathbb{N})$  ist nicht rekursiv axiomatisierbar.
- Oder
- c) jede rekursive Axiomenmenge ist nicht vollständig für  $Th(\mathbb{N})$ .

## Beispiele (Forts.)

↪ Insbesondere: **Peano Axiome**

$$P_1 \quad \forall x \forall y (S(x) = S(y) \rightarrow x = y)$$

$$P_2 \quad \forall x S(x) \neq 0$$

$$P_3 \quad \forall x x + 0 = x$$

$$P_4 \quad \forall x \forall y x + S(y) = S(x + y)$$

$$P_5 \quad \forall x x * 0 = 0$$

$$P_6 \quad \forall x \forall y x * S(y) = x * y + x$$

$$P_7 \quad A_x[0] \rightarrow (\forall x (A \rightarrow A_x[S(x)]) \rightarrow \forall x A),$$

( $A$  Formel mit  $x$  als einzige frei vorkommende Variable in  $A$ )

Sind **keine** Axiomatisierung von  $Th(\mathbb{N})$ .

## Beispiele (Forts.)

### Beispiel 4.16 (Elementare Arithmetik (Arith. 1-Stufe))

Basis der Sprache:  $(\{0, 1, +, *\}, \{<\})$  Interpretation:  $I_A = (\mathbb{N}, I_c)$   
natürliche Interpretation  $I_c$

$\hookrightarrow Th(I_A)$  vollständig nicht rekursiv axiomatisierbar.

## Beispiele (Forts.)

### Beispiel 4.17

**Pressburger Arithmetik:** Sprache  $(\{0, 1, +\}, \{<\})$

Interpretation  $I_{PA} = (\mathbb{N}, I_c)$ ,  $I_c$  wie gehabt.

- ▶  $Th(I_{PA})$  ist vollständig, entscheidbar und endlich axiomatisierbar.

$$Ax < \begin{cases} \forall x \neg(x < 0) \\ \forall x \forall y \ x < Sy \leftrightarrow x < y \vee x = y \\ \forall x \forall y \ x < y \vee x = y \vee y < x \end{cases}$$

## Beispiele (Forts.)

### Beispiel 4.18

$\mathbb{R}$  reelle Zahlen, Sprache  $(\{0, 1, +, *, \dots\}, \{<\})$

Sprache der Körpertheorie.

$Th(\mathbb{R})$  ist rekursiv entscheidbar (Quantorenelimination).

$Th(\mathbb{R})$  ist rekursiv axiomatisierbar.

# Beispiele (Forts.)

Axiomatisierung:

- ▶ Axiome für Körper
- ▶ Nullstellen für Polynome mit ungeradem Grad +  
Ordnungsaxiome:

$$\forall x \neg(x < x)$$

$$\forall x \forall y \forall z ((x < y \wedge y < z) \rightarrow x < z)$$

$$\forall x \forall y x < y \vee y = x \vee y < x$$

$$\forall x \forall y \forall z x < y \rightarrow x + z < y + z$$

$$\forall x \forall y 0 < x \wedge 0 < y \rightarrow 0 < x * y$$

$$\forall x 0 < x \rightarrow \exists y (y * y = x)$$

# Beispiele (Forts.)

## Reell abgeschlossene Körper

$\mathbb{R}$  ist „Beispiel“ dafür, Tarski

↪ Wichtige Folgerungen: Entscheidbarkeit der Ebenen  
euklidische Geometrie!

## Beispiel 4.19

Theorie der Ordnung mit Gleichheit ( $<$ ,  $=$ ) ist weiteres Beispiel  
einer entscheidbaren Theorie.









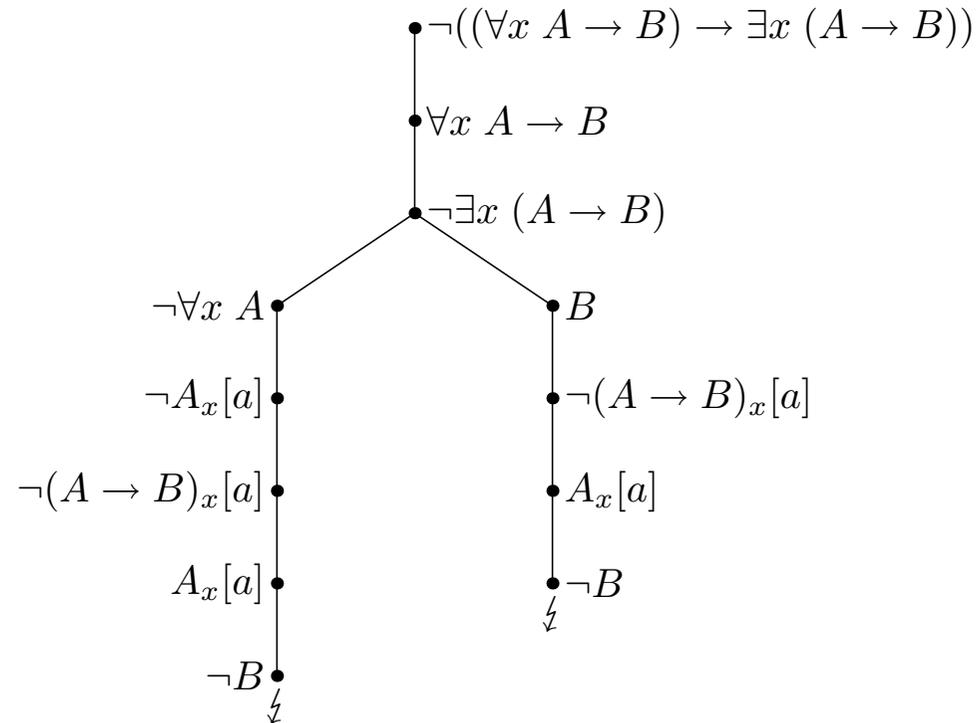






# Beispiele

Beispiel 5.4 ( $\models (\forall x A \rightarrow B) \rightarrow \exists x(A \rightarrow B)$ ,  $x$  nicht frei in  $B$ )







↪ Interpretation:

$p(x, y)$	$a$	$b$	$c$	$d$	$e$	$\dots$
$a$	0	1				
$b$			1			
$c$				1		
$d$					1	
$e$						1
						$\dots$

- ▶ Es gibt Interpretation mit nur 2 Elementen  
 $\neg p(a, a), p(a, b), p(b, a), p(b, b)$ , die Modell sind.

↪ **Vorschrift „neu“ kann aufgeweicht werden.**

- ! Erst alle Konstanten verwenden, falls diese Wahl zu  $\downarrow$  führt müssen neue Konstanten einführt werden, sonst Modell gefunden.







# Beispiele (Forts.)

Interpretation:

$$\begin{aligned}
 p(c_1, c_1) &= F & p(c_1, c_2) &= W & p(c_2, c_2) &= F \\
 p(c_2, c_1) &= F & p(c_2, c_3) &= W & p(c_3, c_3) &= F \dots
 \end{aligned}$$

	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$
$c_1$	0	1			
$c_2$	0	0	1		
$c_3$			0	1	
$c_4$				0	1
$\vdots$					





# Entscheidbare Fälle (Präfixe) Allgemeingültigkeit

## $\mathcal{L}$ Sprache ohne Funktionskonstanten

### abgeschlossene Formeln

- |   |  |
|---|--|
| <ol style="list-style-type: none"> <li>1. <math>\forall x_1 \dots \forall x_m \exists y_1 \dots \exists y_m A</math></li> <li>2. <math>\forall x_1 \dots \forall x_m \exists y \forall z_1 \dots \forall z_n A</math></li> <li>3. <math>\forall x_1 \dots \forall x_m \exists y_1 \exists y_2 \forall z_1 \dots \forall z_n A</math></li> <li>4. <math>\exists y_1 \dots \exists y_n \forall x_1 \dots \forall x_m \exists z A</math></li> <li>5. <math>\forall x_1 \dots \forall x_m \exists y_1 \exists y_2 \exists y_3 \forall z_1 \dots \forall z_n A</math></li> </ol> | <p><math>A</math> quantorenfrei</p><br><p><math>n, m \geq 1</math> n. entsch.</p> <p><math>m, n \geq 0</math> n. entsch.</p> |
|---|--|

## Beispiel 5.10 (Typ 2. Formel)

- $\neg \forall x_1 \forall x_2 \exists y \forall z A$
- $\neg \forall x_2 \exists y \forall z A_{x_1} [a]$
- $\neg \exists y \forall z A_{x_1, x_2} [a, b]$
- $\neg \forall z A_{x_1, x_2, y} [a, b, a]$
- $\neg \forall z A_{x_1, x_2, y} [a, b, b]$
- $A_{x_1, x_2, y, z} [a, b, a, a_1]$
- $A_{x_1, x_2, y, z} [a, b, b, a_2]$

## Resolventenmethode (PL1 - Resolutionsverfahren)

- ▶ **Formeln in Klauselnormalform auf Erfüllbarkeit testen.**

### Definition 5.11

Eine Formel  $A \in \mathbf{Form}(\mathcal{L})$  ist in **Klauselform** (KLF), wenn sie die Gestalt

$$A \equiv \forall x_1 \forall x_2 \cdots \forall x_n [C_1 \wedge C_2 \wedge \cdots \wedge C_k]$$

hat. Dabei sind die  $C_j$  Klauseln (d. h. Disjunktionen von Literalen ohne Wiederholungen) und die Variablen  $x_1, \dots, x_n$  sind alle Variablen, die in den  $C_j$  vorkommen.

- ▶  $\forall x_1, \dots, \forall x_n$  **Präfix**      $[C_1 \wedge \cdots \wedge C_k]$  **Mantisse (Matrix).**

↔ PKNF und Präfix enthält kein  $\exists$  Quantor.





## Skolemisierung: Verfahren

### Beweis:

$A$  gegeben. Transformiere  $A$  wie folgt:

- Schritt 1:** Bilde den existentiellen Abschluss von  $A$ .
- Schritt 2:** Eliminiere alle überflüssigen Quantoren.
- Schritt 3:** Umbenennung mehrfach quantifizierter Variablen.
- Schritt 4:** Ersetze Operatoren, die von  $\wedge, \vee, \neg$  verschieden sind. z. B.  $\rightarrow, \mathbf{if} \dots, \leftrightarrow \dots$
- Schritt 5:** Schiebe  $\neg$  nach innen, bis sie vor den Atomen stehen. Insbesondere  $\neg\neg A$  durch  $A$  ersetzen.  
 $\neg\forall xA \rightsquigarrow \exists x\neg A, \neg(A \vee B) \rightsquigarrow (\neg A \wedge \neg B)$   
 $\neg(A \wedge B) \rightsquigarrow (\neg A \vee \neg B)$

## Skolemisierung: Verfahren (Fort.)

**Schritt 6:** Schränke Quantoren auf ihren Wirkungsbereich ein:

$$Qx(A * B) \rightsquigarrow A * QxB \quad x \text{ nicht frei in } A$$

$$\rightsquigarrow QxA * B \quad x \text{ nicht frei in } B$$

**Schritt 7:** Eliminiere existentielle Quantoren durch Einführung von **Skolem Funktionen**. Wähle dabei die erste Teilformel (von links) der Form  $\exists yB(y)$  und ersetze sie durch  $B_y[f(x_1, \dots, x_n)]$ ,  $n \geq 0$ , wobei

a)  $x_1 \dots x_n$  alle unterschiedlichen freien Variablen in  $\exists yB(y)$  sind, die links von  $\exists yB(y)$  universell quantifiziert sind.

b)  $f$  eine „frische“  $n$ -stellige Funktionskonstante.

**Schritt 8:** Schiebe  $\forall$  Quantoren nach links.

**Schritt 9:** Bringe Matrix in KNF und simplifiziere.

# Skolemisierung

- ▶ **Korrektheit**  $A_j \equiv \dots \exists y B(y) \rightsquigarrow A_{j+1} \equiv \dots B_y[f(x_1, \dots, x_n)]$
- ▶ **Behauptung:**  $A_j$  ist erfüllbar gdw  $A_{j+1}$  ist erfüllbar.  
 Sei  $I$  Modell für  $A_j$ : Für jede mögliche Belegung der Variablen  $x_1, \dots, x_n$  muss es ein oder mehrere Werte für  $y$  geben, mit denen  $A_j$  wahr wird.  
 $I'$  wie  $I$  zusätzlich eine  $n$ -stellige Funktionskonstante  $f : f(x_1, \dots, x_n)$  gibt für Wertetupel für  $x_1 \dots x_n$  ein (den)  $y$  Wert als Funktionsergebnis.

## Beispiele

### Beispiel 5.13

$$\forall x \{p(x) \rightarrow \exists z \{\neg \forall y [q(x, y) \rightarrow p(f(x_1))] \wedge \forall y [q(x, y) \rightarrow p(x)]\}\}$$

1. Existentieller Abschluss und Elimination von überf. Quantoren:

$$\exists x_1 \forall x \{p(x) \rightarrow \{\neg \forall y [q(x, y) \rightarrow p(f(x_1))] \wedge \forall y [q(x, y) \rightarrow p(x)]\}\}$$

2. Umbenennung von  $y$ :

$$\exists x_1 \forall x \{p(x) \rightarrow \{\neg \forall y [q(x, y) \rightarrow p(f(x_1))] \wedge \forall z [q(x, z) \rightarrow p(x)]\}\}$$

3. Elimination von  $\rightarrow$ :

$$\exists x_1 \forall x \{\neg p(x) \vee \{\neg \forall y [\neg q(x, y) \vee p(f(x_1))] \wedge \forall z [\neg q(x, z) \vee p(x)]\}\}$$

## Beispiele (Forts.)

4.  $\neg$  nach innen:

$$\exists x_1 \forall x \{ \neg p(x) \vee \{ \exists y [q(x, y) \wedge \neg p(f(x_1))] \wedge \forall z [\neg q(x, z) \vee p(x)] \} \}$$

5. Quantoren auf Geltungsbereich einschränken:

$$\exists x_1 \forall x \{ \neg p(x) \vee \{ [\exists y q(x, y) \wedge \neg p(f(x_1))] \wedge [\forall z \neg q(x, z) \vee p(x)] \} \}$$

6. Eliminieren der Ex.-Quantoren  $\exists x_1$  und  $\exists y$  (0 - stellige bzw. 1-stellige Funktionseinführung):

$$\forall x \{ \neg p(x) \vee \{ [q(x, g(x)) \wedge \neg p(f(a))] \wedge [\forall z \neg q(x, z) \vee p(x)] \} \}$$

## Beispiele (Forts.)

7. Quantoren nach links:

$$\forall x \forall z \{ \neg p(x) \vee \{ [q(x, g(x)) \wedge \neg p(f(a))] \wedge [\neg q(x, z) \vee p(x)] \} \}$$

8. KNF + Simplifikation:

$$\forall x \forall z \{ [\neg p(x) \vee q(x, g(x))] \wedge [\neg p(x) \vee \neg p(f(a))] \wedge [\neg p(x) \vee \neg q(x, z) \vee p(x)] \}$$

$$\rightsquigarrow A' \equiv \forall x \{ [\neg p(x) \vee q(x, g(x))] \wedge \neg p(f(a)) \}$$

►  $A' \equiv \forall x [ [\neg p(x) \vee q(x, g(x))] \wedge [\neg p(f(a))] ]$  Erfüllbar?







## Herbrand-Universum (Forts.)

### Beispiel 5.15

- $A \equiv \forall x \{ [\neg p(x) \vee q(x, g(x))] \wedge \neg p(f(a)) \}$

$$\hookrightarrow H_A = \{ a, f(a), g(a), g(g(a)), g(f(a)), \dots \}$$

Die Menge der Grundterme in  $a, f$  1-stellig,  $g$  1-stellig.

- $B \equiv \forall x \exists y [p(f(x), y, g(x, y))]$

$$\hookrightarrow H_B \equiv \{ a, f(a), g(a, a), f(g(a, a)), g(a, f(a)), \dots \}$$

## Das Herbrand-Universum (Forts.)

### Definition 5.16 (Herbrand-Interpretation)

Eine **Herbrand-Interpretation** für eine (abgeschlossene) Formel  $A$  ist eine Interpretation  $I = (H_A, \dots)$ , die

- a) jeder Individuenkonstante  $a$  in  $A$  den Term  $a \in H_A$  zuordnet.
- b) jeder  $n$ -stelligen Funktionskonstante  $f$ , die in  $A$  vorkommt, eine Funktion  $I(f) : H_A^n \rightarrow H_A$  zuordnet, die die Terme  $t_1, \dots, t_n \in H_A$  auf den Term  $f(t_1, \dots, t_n) \in H_A$  abbildet.

# Der Satz von Herbrand

## Satz 5.17

- *Eine Formel  $A$  in Klauselform ist erfüllbar gdw  $A$  ist in einer Herbrand-Interpretation erfüllbar.*
- *$A$  ist unerfüllbar gdw  $A$  unerfüllbar in allen Herbrand-Interpretationen.*

► Festlegung der Definitionsbereiche der Interpretationen.

Sei  $A \equiv \forall x_1 \cdots \forall x_n [C_1 \wedge \cdots \wedge C_k]$ .

Eine **Grundinstanz** einer Klausel  $C_j$  ( $1 \leq j \leq k$ ) von  $A$  ist eine Klausel, die man erhält, wenn man alle Individuenvariablen in  $C_j$  durch Terme aus  $H_A$  ersetzt, d. h. eine Grundsubstitution  $x_i \leftarrow t_i$  auf  $C_j$  anwendet. Solche Klauseln, die keine Individuenvariablen enthalten heißen **Grundklauseln**.





## Grundresolventenmethode

- ▶  $A \equiv [\forall x \exists y p(x, y) \wedge \forall y \exists z q(y, z)] \rightarrow \forall x \exists y \exists z (p(x, y) \wedge q(y, z))$

gilt  $\models A$ ? Transformiere  $\neg A$  in KLF

- ▶  $B \equiv \forall x \forall y \forall y_1 \forall z [p(x, f(x)) \wedge q(y, g(y)) \wedge (\neg p(a, y_1) \vee \neg q(y_1, z))]$

- ▶  $H_B = \{a, f(a), g(a), f(f(a)), f(g(a)), \dots\}$

- ▶ **Bilde Grundinstanzen:**

$$[p(a, f(a)) \wedge q(a, g(a)) \wedge [\neg p(a, a) \vee \neg q(a, a)]]$$

$$\wedge$$

$$\vdots$$

$$\wedge [p(a, f(a)) \wedge q(f(a), g(f(a))) \wedge [\neg p(a, f(a)) \vee \neg q(f(a), g(f(a)))]]$$



# Substitutionen

1.	$p(x, f(x))$	$x \leftarrow a$	$p(a, f(a))$
2.	$q(y, g(y))$	Substitutionen	
3.	$\neg p(a, y_1) \vee \neg q(y_1, z)$	$y_1 \leftarrow f(a)$	$\neg p(a, f(a)) \vee \neg q(f(a), z)$
4.	$\neg q(f(a), z)$	1R3	$x \leftarrow a \quad y_1 \leftarrow f(a)$
5.	$\square$	2R4	$y \leftarrow f(a) \quad z \leftarrow g(f(a))$

## Definition 5.19 (Erinnerung)

Eine Substitution  $\theta$  ist eine endliche Menge der Form

$\{\langle v_1, t_1 \rangle, \dots, \langle v_n, t_n \rangle\}$ ,  $v_i$  ist Individuenvariable  $v_i \neq v_j$ ,  $t_i$  Terme ( $\mathbf{Term}(\mathbb{V}, \mathbb{F})$ ),  $\mathbb{F}$  Funktionskonstanten  $t_i \neq v_i$ ,  $i = 1, \dots, n$ .

$\{v_1, \dots, v_n\}$  ist der Definitionsbereich der Substitution  $\langle v_i, t_i \rangle$

► „Bindung“ für  $v_i$ .

►  $\theta$  induziert Abbildungen  $\theta : \mathbf{Term}(\mathbb{V}, \mathbb{F}) \rightarrow \mathbf{Term}(\mathbb{V}, \mathbb{F})$   
bzw.  $\theta : \mathbf{AForm} \rightarrow \mathbf{AForm}$

Wie üblich:

- $\theta(v_i) \equiv t_i \quad i = 1, \dots, n$
- $\theta(v) \equiv v \quad \text{sonst}$
- $\theta(f(t_1, \dots, t_n)) \equiv f(\theta(t_1), \dots, \theta(t_n))$
- $\theta(p(t_1, \dots, t_n)) \equiv p(\theta(t_1), \dots, \theta(t_n))$

## Substitutionen (Forts.)

- ▶ Komposition von Substitutionen: wie üblich.  
 $\theta\sigma$  erst  $\theta$ , dann  $\sigma$ .  
Identität als Substitution erlaubt.
- ▶ Betrachte  $t_1 \equiv f(g(x), y)$ ,  $t_2 \equiv f(z, g(a))$   
**Frage:** Gibt es Substitution  $\theta$  mit  $t_1\theta = t_2\theta$  d.h.

$$\theta(f(g(x), y)) \equiv \theta(f(z, g(a)))$$

- ▶  $\theta$  heißt dann **Unifikator** von  $t_1$  und  $t_2$ .  
 $\{x \leftarrow a \quad y \leftarrow g(a) \quad z \leftarrow g(a)\}$  sind  
 $\{y \leftarrow g(a) \quad z \leftarrow g(x)\}$  Unifikatoren

$\rightsquigarrow$  **Unifikationsalgorithmen**

**Allgemeinster Unifikator:**  $\sigma$  (MGU) (**Most General Unifier**).

**Eigenschaft:** Ist  $\theta$  Unifikator, so gibt es  $\tau$  mit  $\theta = \sigma\tau$ .

# Unifikation

## Definition 5.20 (Unifikator)

Sei  $S = A_1 \vee \dots \vee A_n$   $\{A_1, \dots, A_n\}$  eine Disjunktion (Menge) atomarer Formeln  $A_j$  ( $1 \leq j \leq n$ ). Eine Substitution  $\theta$  heißt **Unifikator** für  $S$ , falls  $A_1\theta \equiv A_2\theta \equiv \dots \equiv A_n\theta$  (insbesondere müssen die Prädikatskonstanten der  $A_j$  alle identisch sein).

- ▶ Gibt es für  $S$  einen solchen Unifikator, dann heißt  $S$  **unifizierbar**.
- ▶ Ein Unifikator  $\theta$  heißt **allgemeinster Unifikator** oder **MGU** für  $S$ , wenn es für jeden Unifikator  $\sigma$  von  $S$  eine Substitution  $\tau$  gibt, so dass  $\sigma = \theta\tau$ .





## Resolutionsverfahren

### Definition 5.22 (Allgemeine Resolventenregel)

Seien  $C_1$  und  $C_2$  Klauseln ohne gemeinsame Variablen.

Seien  $A_1 \vee \dots \vee A_k$  und  $\neg B_1 \vee \neg B_2 \vee \dots \vee \neg B_l$  Teildisjunktionen von  $C_1$  bzw.  $C_2$ , so dass  $A_1, \dots, A_k, B_1, \dots, B_l$  unifizierbar sind, mit allgemeinsten Unifikator  $\theta$  und sei

$$A_i\theta \equiv B_j\theta \equiv p(r_1, \dots, r_n) \text{ [Faktor]}$$

Die **Resolvente** von  $C_1$  und  $C_2$  ist dann die Klausel

$C_1\theta \setminus p(r_1, \dots, r_n) \cup C_2\theta \setminus \neg p(r_1, \dots, r_n)$  als Menge von Literalen.

## Resolutionsverfahren (Forts.)

### Satz 5.23 (Robinson)

*Eine Formel  $A$  in Klauselform ist genau dann unerfüllbar, wenn die leere Klausel  $\square$  aus  $A$  mit der Resolventenregel hergeleitet werden kann.*

- ▶ [Annahme: Klauseln in  $A$  sind variablendisjunkt] Immer erreichbar da

$$\forall \bar{x} [C_1(\bar{x}) \wedge \dots \wedge C_k(\bar{x})] \models \exists \bar{x}_1 \forall \bar{x}_2 \dots \forall \bar{x}_k [C_1(\bar{x}_1) \wedge \dots \wedge C_k(\bar{x}_k)]$$

- ▶ Beachte allgemeine Resolventenregel  $k, l \geq 1$
- ▶ Viele Varianten: **Unit-Resolution**, **lineare Resolventenregel**...
- ! Ziel ist es, so schnell wie möglich die leere Klausel herzuleiten, d. h. soviel Literale wie möglich zu „faktorisieren“.

## Beispiel

### Beispiel 5.24

$$A \equiv \neg \exists y \forall z [p(z, y) \leftrightarrow \neg \exists x [p(z, x) \wedge p(x, z)]]$$

- Frage gilt  $\models A$ ?

$$\neg A \equiv \neg \neg \exists y \forall z [p(z, y) \leftrightarrow \neg \exists x [p(z, x) \wedge p(x, z)]]$$

↪ KLF( $\neg A$ )

$$\forall z \forall x [ \quad [\neg p(z, a) \vee \neg p(z, x) \vee \neg p(x, z)] \wedge \\ [p(z, f(z)) \vee p(z, a)] \wedge \\ [p(f(z), z) \vee p(z, a)]]$$

## Beispiel (Forts.)

- ▶ Klauseln (nach Umbenennen der Variablen, variablendisjunkt)
  1.  $\neg p(z_1, a), \neg p(z_1, x_1), \neg p(x_1, z_1)$
  2.  $p(z_2, f(z_2)), p(z_2, a)$
  3.  $p(f(z_3), z_3), p(z_3, a)$
- Resolvente von 2. und 1. mit Teildisjunktionen  
 $\neg p(z_1, a), \neg p(z_1, x_1), \neg p(x_1, z_1)$  in 1. und  $p(z_2, a)$  in 2.  
Der Unifikator  $\langle x_1, a \rangle, \langle z_2, a \rangle, \langle z_1, a \rangle$  MGU.
  - 4.  $p(a, f(a))$

## Beispiel (Forts.)

- Resolvente 3. und 1. mit Teildisjunktionen

$$\neg p(z_1, a), \neg p(z_1, x_1), \neg p(x_1, z_1), p(z_3, a)$$

▶ DS:

$$\left. \begin{array}{ll} \{z_1, z_1, x_1, z_3\} & x_1 \leftarrow z_1 \quad z_3 \leftarrow z_1 \\ \{a, z_1\} & z_1 \leftarrow a \end{array} \right\} \begin{array}{ll} x_1 \leftarrow a & z_1 \leftarrow a \\ z_3 \leftarrow a & \end{array}$$

- 5.  $p(f(a), a)$

- Resolvente 4., 1. mit Teildisjunktionen

$$p(a, f(a)), \neg p(x_1, z_1) \quad x_1 \leftarrow a, z_1 \leftarrow f(a)$$

- 6.  $\neg p(f(a), a)$

## Beispiel (Forts.)

- Resolvente 5., 6.

- 7. □

↪ Also gilt

$$\models A \equiv \neg \exists y \forall z [p(z, y) \leftrightarrow \neg \exists x [p(z, x) \wedge p(x, z)]]$$

## Logisches Programmieren und Prolog

- ▶ Sprache der Logik zur Darstellung von Wissen (deklarativ) über Strukturen. **Wissensherleitung** durch logische Folgerung (deduktiv, Resolution usw.)

- ▶ **Prozedurale Sicht von Resolution**

$$Z = ((Y \cdot 2) + X) - Y$$

Funktion von  $X, Y$

$$2 \equiv \text{succ}(\text{succ}(0))$$

- 0  $\text{goal}(X, Y, Z) : - \text{mult}(Y, 2, A), \text{add}(A, X, B), \text{subt}(B, Y, Z).$
- 1  $\text{add}(R, \text{succ}(S), \text{succ}(T)) : - \text{add}(R, S, T).$
- 2  $\text{add}(T, 0, T).$
- 3  $\text{subt}(\text{succ}(R), \text{succ}(S), T) : - \text{subt}(R, S, T).$
- 4  $\text{subt}(T, 0, T).$
- 5  $\text{mult}(R, \text{succ}(S), T) : - \text{mult}(R, S, U), \text{add}(R, U, T).$
- 6  $\text{mult}(R, 0, 0).$

# Logisches Programmieren und Prolog

- ▶ `goal`, `add`, `subt`, `mult` sind 3-stellige  $P$ -Konstanten zur Darstellung der Funktionen, die den ersten beiden Argumenten als Wert 3-Argumente zuordnen.
- ▶ Wie „berechnet“ sich  $((Y \cdot 2) + X) - Y$ , wenn  $X \leftarrow \text{succ}(\text{succ}(0))$  und  $Y \leftarrow \text{succ}(0)$   
 ? – `goal(succ(succ(0)), succ(0), Z)`

?



. With  $Z = \text{succ}(\text{succ}(\text{succ}(0)))$

# Horn-Logik

**Erinnerung:** Klauseln  $\{L_1, L_2, \dots, L_n\}$  Menge von Literalen  
 $= \{A_1, \dots, A_n\} \cup \{\neg B_1, \dots, \neg B_m\}$ ,  $A_j, B_j$  Atome.

- ▶ **Hornlogik:** Klauseln mit höchstens einem positiven Literal,  
d. h.  $n \leq 1$ . Einteilung:

$\{A, \neg B_1, \dots, \neg B_m\}$ $m > 0$	<b>Regel Klausel</b>	◀
$\{A\}$	<b>Fakt Klausel</b>	◀
$\{\neg B_1, \dots, \neg B_m\}$ $m > 0$	<b>Goal Klausel</b>	◀
$\emptyset$	<b>leeres Goal</b>	◀

# Horn-Logik

- ▶ Formeln in KLF: Endliche Mengen von Literalen.

Hornklausel	Formel von PL-1
<ul style="list-style-type: none"> <li>• <math>\{A, \neg B_1, \dots, \neg B_m\}</math></li> </ul>	$\forall (A \vee \neg B_1 \vee \dots \vee \neg B_m)$ bzw. $(\forall ((B_1 \wedge \dots \wedge B_m) \rightarrow A))$
<ul style="list-style-type: none"> <li>• <math>\{A\}</math></li> </ul>	$\forall (A)$
<ul style="list-style-type: none"> <li>• <math>\{\neg B_1, \dots, \neg B_m\}</math></li> </ul>	$\forall (\neg B_1 \vee \dots \vee \neg B_m)$ bzw. $\neg \exists (B_1 \wedge \dots \wedge B_m)$
□	false

↪ **Notationen:**

Formel	logisches Programm	Prolog
<ul style="list-style-type: none"> <li>• <math>\forall ((B_1 \wedge \dots \wedge B_m) \rightarrow A)</math></li> </ul>	$A \leftarrow B_1, \dots, B_m$	$A : - B_1, \dots, B_m.$
<ul style="list-style-type: none"> <li>• <math>\forall (A)</math></li> </ul>	$A \leftarrow$	$A.$
<ul style="list-style-type: none"> <li>• <math>\neg \exists (B_1 \wedge \dots \wedge B_m)</math></li> </ul>	$\leftarrow B_1, \dots, B_m$	$? - B_1, \dots, B_m.$

## Horn-Logik (Forts.)

- ▶ Eine Menge von Regeln und Fakten ist ein **logisches Programm**  $P$   
d. h. die Programmformeln sind entweder Regeln oder Fakten.
- Eine Struktur  $\mathcal{R}$  ist **Modell** von  $P$ , falls  $\mathcal{R}$  Modell der entsprechenden Formeln in  $P$  ist.
- $P \models \varphi$  hat die übliche Bedeutung.
- ! **Beachte:** Sei  $P$  logisches Programm und  $? - B_1, \dots, B_m$  ein nichtleeres Ziel. Dann sind äquivalent
  1.  $P \cup \{? - B_1, \dots, B_m\}$  hat kein Modell (unerfüllbar)
  2.  $P \models \exists(B_1 \wedge \dots \wedge B_m)$
- ↪ Herbrand Interpretationen reichen aus: d. h. Termmengen und Funktionen sind fest durch die Formeln (Programm) definiert.

## Horn-Logik (Forts.)

- ▶ Offen ist nur noch die Interpretation der  $P$ -Konstanten.
 
$$\mathcal{R} \leftrightarrow I(\mathcal{R}) = \{r(t_1, \dots, t_n) \mid r \text{ } P\text{-Konstante, } n\text{-stellig}$$

$$t_1, \dots, t_n \in H_P \text{ Grundterme}$$

$$(t_1, \dots, t_n) \in r\mathcal{R}\}$$

### Semantik logischer Programme (deklarative Semantik).

Sei  $P$  ein logisches Programm. Unter den Herbrand Interpretationen die Modelle von  $P$  sind, gibt es ein minimales Herbrand Modell  $M_P$ :

- $M_P = \{r(t_1, \dots, t_n) \mid t_1, \dots, t_n \text{ Grundterme und}$   
 $P \models r(t_1, \dots, t_n)\}$
- ▶  $M_P$  lässt sich rekursiv definieren.

## Horn-Logik (Forts.)

### Satz 5.25

Sei  $\exists X_1 \cdots \exists X_k (B_1 \wedge \cdots \wedge B_m)$  eine abgeschlossene existentielle Formel und  $P$  logisches Programm. Dann sind äquivalent:

1.  $P \models \exists X_1 \cdots \exists X_k (B_1 \wedge \cdots \wedge B_m)$
2.  $P \models (B_1 \wedge \cdots \wedge B_m)[X_1/t_1, \dots, X_k/t_k]$  für Grundterme  $t_1, \dots, t_k$
3.  $M_p$  ist Modell von  $\exists X_1 \cdots \exists X_k (B_1 \wedge \cdots \wedge B_m)$
4.  $M_p \models (B_1 \wedge \cdots \wedge B_m)[X_1/t_1, \dots, X_k/t_k]$  für Grundterme  $t_1, \dots, t_k$

**!** Grundlage für  $M_p$  ist die Semantik (Bedeutung) von  $P$ .  
 (Beachte der Satz gilt nicht für universelle Formeln!)

## Horn-Logik (Forts.)

### Beispiel 5.26

$P$  über Signatur  $0, \text{succ}$  (Fkt. Symbole),  $\text{add}$  (3 St. Pr.-Konstante)

$P$  :  $\text{add}(X, 0, X).$   
 $\text{add}(X, \text{succ}(Y), \text{succ}(Z)) : \neg \text{add}(X, Y, Z).$

↪ Offenbar ist

$$M_P = \{\text{add}(\text{succ}^n(0), \text{succ}^m(0), \text{succ}^{n+m}(0)) \mid n, m \in \mathbb{N}\}$$

## Wie werden existentielle Anfragen beantwortet?

- ▶ (Frage 1) ?  $\text{-add}(\text{succ}^3(0), \text{succ}^8(0), Z)$   
JA mit  $Z = \text{succ}^{11}(0)$
  
  - ▶ (Frage 2) ?  $\text{-add}(X, \text{succ}^8(0), Z)$   
 $(P \stackrel{?}{\models} \exists X \exists Z \text{ add}(X, \text{succ}^8(0), Z))$   
 JA mit  $X = 0, Z = \text{succ}^8(0)$   
 mit  $X = \text{succ}(0), Z = \text{succ}^9(0) \dots$
- $\left. \begin{array}{l} \\ \\ \end{array} \right\} Z = \text{succ}^8(X)$   
 ist allgemeinste Lösung.

## Wie werden existentielle Anfragen beantwortet? (Forts.)

- ▶ (Frage 3) ?  $\text{--add}(\text{succ}^3(0), Y, Z)$   
 $(P \stackrel{?}{\models} \exists Y \exists Z \text{ add}(\text{succ}^3(0), Y, Z))$

JA mit  $Y = 0, Z = \text{succ}^3(0)$   
 mit  $Y = \text{succ}(0), Z = \text{succ}^4(0)$   
 mit  $Y = \text{succ}^2(0), Z = \text{succ}^5(0) \dots$

- $Z = \text{succ}^3(Y)$  ist jedoch keine allgemeinste Lösung:
- ▶ Da  $\forall Y \text{ add}(\text{succ}^3(0), Y, \text{succ}^3(Y))$  nicht logische Folgerung von  $P$  ist (Übung!)  
 (Sie ist jedoch in  $M_p$  gültig!! **Induktives Theorem**)

## Wie werden existentielle Anfragen beantwortet? (Forts.)

- (Frage 4) ?  $\neg \text{add}(X, Y, \text{succ}^3(0))$   
 $(P \stackrel{?}{\models} \exists X \exists Y \text{add}(X, Y, \text{succ}^3(0)))$

**Lösungssubstitutionen:**

$X = 0$	$Y = \text{succ}^3(0)$
$\text{succ}(0)$	$\text{succ}^2(0)$
$\text{succ}^2(0)$	$\text{succ}(0)$
$\text{succ}^3(0)$	$0$

## Lösungssubstitutionen

Sei  $G = ? - B_1, \dots, B_m$  ein goal,  $P$  logisches Programm,  $\sigma$  Substitution,  $\sigma|_G$  Einschränkung von  $\sigma$  auf die Variablen, die in  $G$  vorkommen.

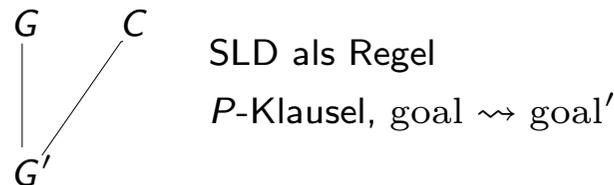
$\sigma = \{X_1 \leftarrow t_1, \dots, X_n \leftarrow t_n\}$  ist eine **korrekte Lösungssubstitution** für  $P \cup \{G\}$  gdw  $X_1, \dots, X_n$  kommen in  $G$  vor und  $P \models \forall ((B_1 \wedge \dots \wedge B_m)\sigma)$ .

(Beachte: nicht äquivalent zu  $M_p \models \forall ((B_1 \wedge \dots \wedge B_m)\sigma)$  nur, falls variablenfrei).

- ? Wie operationalisiert man die Bestimmung von korrekten Lösungssubstitutionen?  $\rightsquigarrow$  **Operationale Semantik Varianten der Resolution.**

# SLD-Resolution – (selective linear Resolution for definitive clauses)

- ▶ Sei  $G$  goal  $? - A_1, \dots, A_m$  und  $C \equiv A : - B_1, \dots, B_q$  eine Programmformel ( $q = 0$  erlaubt). Seien weiterhin  $G$  und  $C$  variablendisjunkt und sei  $\mu$  ein MGU von  $A_k$  und  $A$ . (Die Klauseln können resolviert werden).  
Das goal
- ▶  $G' \equiv ? - A_1\mu, \dots, A_{k-1}\mu, B_1\mu, \dots, B_q\mu, A_{k+1}\mu, \dots, A_k\mu$  ist eine **SLD-Resolvente** von  $G$  und  $C$  über  $\mu$ .



# SLD-Resolution: SLD-Ableitungen

- ▶ **SLD-Ableitungen**, wie üblich definieren:  
SLD-Ableitungen sind Ableitung der Form

$G_0, G_1, \dots, G_n, \dots$  Programmformeln  $C_0, C_1, \dots, C_n, \dots$   
 Substitutionen  $\mu_0, \mu_1, \dots, \mu_n, \dots$ , so dass  
 $G_{n+1}$  SLD-Resolvente von  $G_n$  und  $C_n$  über  $\mu_n$

(Hierbei kommen die Variablen in  $C_{n+1}$  nicht in  $G_0, G_1, \dots, G_n, C_0, \dots, C_n, \mu_0, \dots, \mu_n$  vor).

## Lösungssubstitution

### Definition 5.27 (Lösungssubstitution)

Sei  $P$  logisches Programm,  $G$  goal.

Eine **SLD-Widerlegung** von  $P \cup \{G\}$  ist eine endliche SLD-Ableitung, bis zum Ziel  $G_n$  aus  $G$ , wobei  $G_n$  leer ist.

$C_0, \dots, C_{n-1}$  sind Varianten (Umbenennungen) von Programmformeln aus  $P$ .

$\mu = (\mu_0 \mu_1 \cdots \mu_{n-1})|_G$  ist die berechnete **Lösungssubstitution**.

## Bemerkung und Beispiel

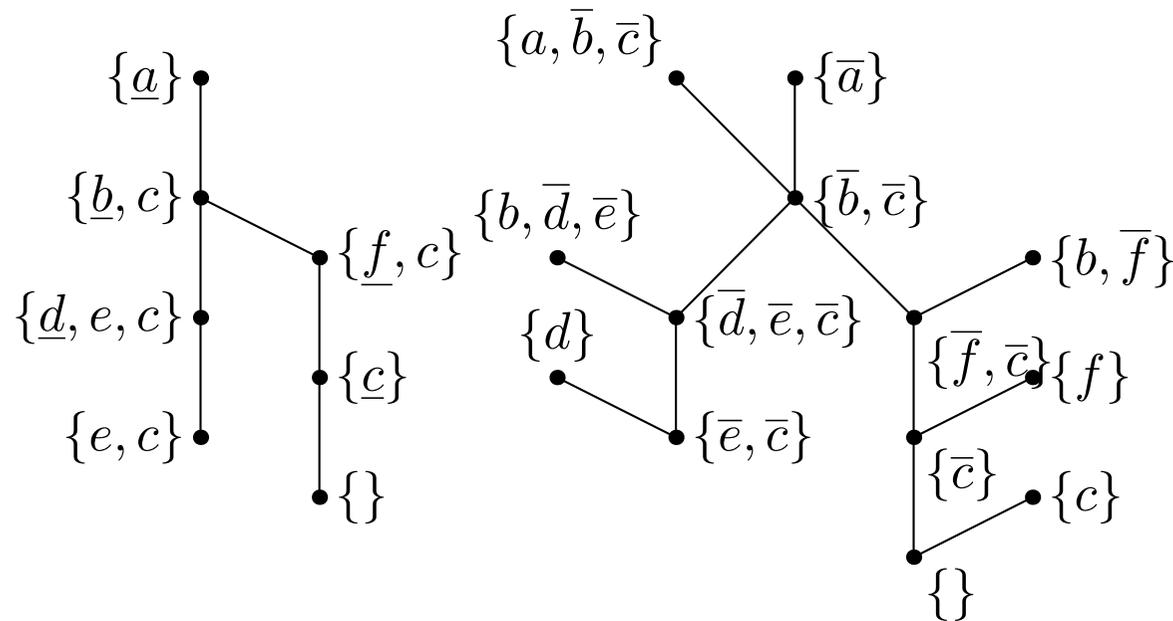
### Bemerkung 5.28

- *Korrektheit und Vollständigkeit lassen sich beweisen!*  
*Siehe etwa Leitsch: Resolutionskalküle*

### Beispiel 5.29

	Klauseln
$a : - b, c.$	$\{a, \bar{b}, \bar{c}\}$
$a : - d.$	$\{a, \bar{d}\}$
$b : - d, e.$	$\{b, \bar{d}, \bar{e}\}$
$b : - f.$	$\{b, \bar{f}\}$
$c.$	$\{c\}$
$c : - d, f.$	$\{c, \bar{d}, \bar{f}\}$
$d.$	$\{d\}$
$f.$	$\{f\}$

# Beispiel: Goal ? - a





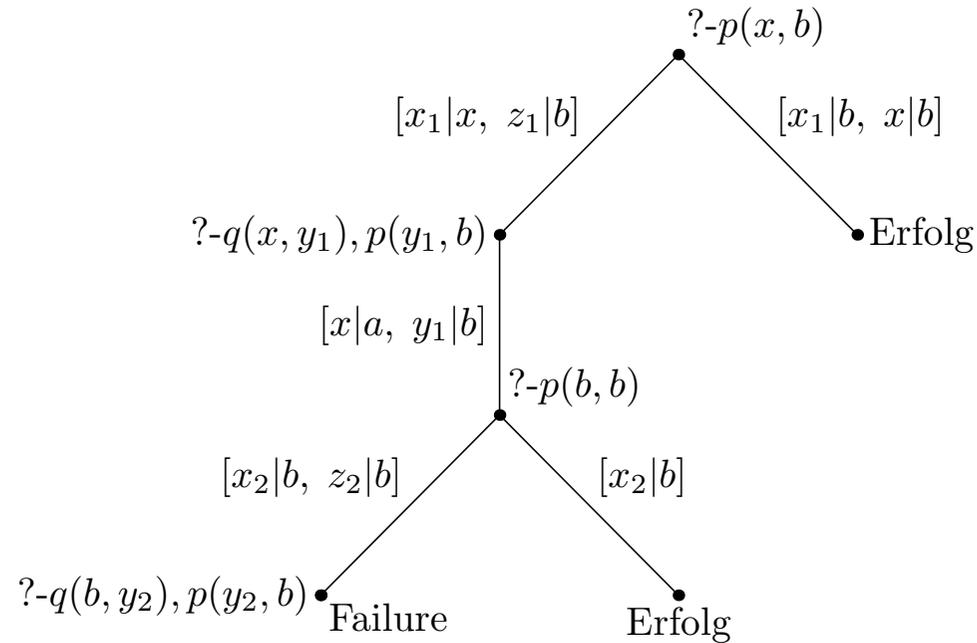
# Beispiele

►  $P$  sei gegeben durch:

- 1  $p(X, Z) : - q(X, Y), p(Y, Z).$
- 2  $p(X, X).$
- 3  $q(a, b).$

$G \equiv ? - p(X, b)$

SLD( $P, G$ ) Baum aller N-SLD Resolventen



## Beispiele (Forts.)

### ► Umordnung der Literale in Programmformeln

$$1 \quad p(X, Z) : - p(Y, Z), q(X, Y).$$

$$2 \quad p(X, X).$$

$$3 \quad q(a, b).$$

$$G \equiv? - p(X, b)$$

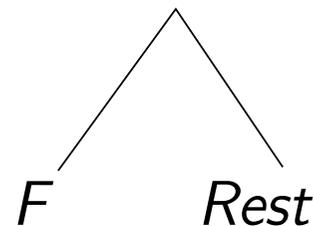
→ Depth First  $\rightsquigarrow$  kein Ergebnis.

$\rightsquigarrow$  Umordnung des SLD-Baums

## Beispiel 5.30 (Listen über $\Sigma$ )

Gegeben Signatur  $\Sigma$  definiere Listen über  $\Sigma$ :

- ▶ 2-stellige Funktion Infix-Notation  $[\cdot|\cdot] : L(\Sigma)^2 \rightarrow L(\Sigma)$
- ▶ Konstante  $[ ]$  bezeichne die leere Liste.
- ▶ **Rekursive Definition:**
  - ▶  $[ ]$  ist Liste über  $\Sigma$
  - ▶  $[F|Rest]$  ist Liste über  $\Sigma$ , falls  $F$   $\Sigma$ -Term oder Liste über  $\Sigma$
  - ▶  $Rest$  Liste über  $\Sigma$



- $[F_1, F_2, \dots, F_n|Rest] := [F_1|[F_2|\dots|[F_n|Rest]\dots]]$
- $[F_1, \dots, F_n] := [F_1, \dots, F_n|[ ]]$



