R enthält alle Regeln, die durch das folgende Regelschema gegeben sind:

$$\frac{A, A \to B}{B} \quad \text{(Modus Ponens)}$$

Man schreibt $\vdash_{\mathcal{F}} A$ (oder kurz $\vdash A$), falls A in $\mathcal{F}$ herleitbar ist bzw. $\Sigma \vdash_{\mathcal{F}} A$, (oder kurz $\Sigma \vdash A$, falls A in $\mathcal{F}$ aus Σ herleitbar ist.

Beachte: Alle Axiome sind allgemeingültige Formeln, d.h. die Axiome sind korrekt. Sind die Formeln A und $A \to B$ allgemeingültig, so ist auch B allgemeingültig. Ist die Formel A allgemeingültig, dann ist auch $\forall x\, A$ (eine Generalisierung) allgemeingültig. Deshalb folgt aus $\vdash A$ stets $\models A$ und aus $\Sigma \vdash A$ folgt $\Sigma \models A$. Das heißt $\mathcal{F}$ ist korrekt!

Man kann andere Axiomatisierungen angeben. Betrachte das deduktive System $\mathcal{F}' = (\mathrm{Ax}', R')$, wobei Ax' alle Generalisierungen der Axiome Ax1,..,Ax8 umfasst. Die Regelmenge R' enthalte zusätzlich zum Modus-Ponens noch die *Generalisierungsregel*:

$$\frac{A}{\forall x\, A}$$

Dann gilt für $A \in$ Form und $\Sigma \subseteq$ Form :

$\vdash_{\mathcal{F}} A$ genau dann, wenn $\vdash_{\mathcal{F}'} A$ gilt, also genau dann, wenn $\models A$ gilt und aus $\Sigma \vdash_{\mathcal{F}} A$ folgt $\Sigma \vdash_{\mathcal{F}'} A$ Gilt zusätzlich, dass Σ nur abgeschlossene Formeln enthält, dann gilt auch $\Sigma \vdash_{\mathcal{F}'} A$ genau dann, wenn $\Sigma \models A$ gilt.

Für eine Menge $\Gamma \subseteq$ Form folgt wegen der Generalisierungsregel jedoch nicht immer aus $\Gamma \vdash_{\mathcal{F}'} A$ auch $\Gamma \models A$.

Denn beispielsweise gilt für alle einstelligen Prädikatskonstanten p zwar $p(x) \vdash_{\mathcal{F}'} \forall x\, p(x)$, aber $p(x) \not\models \forall x\, p(x)$.

2.24 Bemerkung. Alle Formeln, die sich aus Ax1, Ax2, Ax3 und Modus-Ponens herleiten lassen, sind allgemeingültig (genauer: Tautologien). Ist andererseits A eine Tautologie der Aussagenlogik und entsteht A' aus A durch Substitution der aussagenlogischen Konstanten durch Formeln erster Stufe, so gilt $\vdash_{\mathcal{F}} A'$.

2.25 Beispiele.

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

   Beweis:

   $B_1 \equiv \forall x\, [(\forall y\, \neg p(y) \to \neg p(x)) \to (p(x) \to \neg \forall y\, \neg p(y))]$ (Ax3,Gen)

   $B_2 \equiv \forall x\, ((\forall y\, \neg p(y) \to \neg p(x)) \to (p(x) \to \neg \forall y\, \neg p(y))) \to$
   $[\forall x\, (\forall y\, \neg p(y) \to \neg p(x)) \to \forall x\, (p(x) \to \neg \forall y\, \neg p(y))]$ (Ax5)

   $B_3 \equiv \forall x\, (\forall y\, \neg p(y) \to \neg p(x)) \to \forall x\, (p(x) \to \neg \forall y\, \neg p(y))$ (MP)

   $B_4 \equiv \forall x\, (\forall y\, \neg p(y) \to \neg p(x))$ (Ax4,Gen)

   $B_5 \equiv \forall x\, (p(x) \to \exists y\, p(y))$ (MP)

2. $\vdash_{\mathcal{F}} \forall x\ A \to \exists x\ A$.

Beweis:

$\vdash \forall x\ \neg A \to \neg A$	(Ax4)
$\vdash (\forall x\ \neg A \to \neg A) \to (A \to \neg\forall x\ \neg A)$	(Taut)
$\vdash A \to \neg\forall x\ \neg A$	(MP)
$\vdash \forall x\ A \to A$	(Ax4)
$\vdash (\forall x\ A \to A) \to$ $\quad((A \to \neg\forall x\ \neg A) \to (\forall x\ A \to \neg\forall x\ \neg A))$	(Taut)
$\vdash (A \to \neg\forall x\ \neg A) \to (\forall x\ A \to \neg\forall x\ \neg A)$	(MP)
$\vdash \forall x\ A \to \exists x\ A$	(MP)

3. $\vdash_{\mathcal{F}'} \forall x\ A \to \forall z\ A_x[z]$, falls z nicht in A vor kommt.

Beweis:

$\vdash \forall x\ A \to A_x[z]$	
$\vdash \forall z\ (A \to A_x[z]) \to (A \to \forall z\ A_x[z])$	(Ax5, Bem)
$\vdash \forall z\ (\forall x\ A \to A_x[z])$	(Gen)
$\vdash \forall x\ A \to \forall z\ A_x[z]$	(MP)

Es soll nun der Frage nachgegangen werden, ob sich die Sätze aus 2.15 auch auf die deduktiven Systeme $\mathcal{F}$ und $\mathcal{F}'$ anwenden lassen.

2.26 Satz. Seien $\Gamma \subseteq \text{Form}, A, B \in \text{Form}$.

1. *Deduktionstheorem*:
   (a) Genau dann gilt $\Gamma \vdash_{\mathcal{F}} A \to B$, wenn $\Gamma, A \vdash_{\mathcal{F}} B$ gilt.
   (b) Genau dann gilt $\Gamma \vdash_{\mathcal{F}'} A \to B$, wenn $\Gamma, A \vdash_{\mathcal{F}'} B$ gilt und die Generalisierung nicht auf eine in A frei vorkommende Variable angewandt wurde.
   (c) Genau dann gilt $\Gamma, A \vdash_{\mathcal{F}'} B$, wenn $\Gamma \vdash_{\mathcal{F}'} \tilde{A} \to B$ gilt, wobei $\tilde{A}$ ein universeller Abschluss von A ist.
2. *Generalisierungstheorem:*
   (a) Falls $\Gamma \vdash_{\mathcal{F}} A$ gilt und x eine Variable ist, die in Γ nicht frei vorkommt, dann gilt $\Gamma \vdash_{\mathcal{F}} \forall x\ A$.
   (b) Genau dann gilt $\Gamma \vdash_{\mathcal{F}'} A$, wenn $\Gamma \vdash_{\mathcal{F}'} \forall x\ A$ gilt.
3. *Kontrapositionstheorem:* Genau dann gilt $\Gamma, A \vdash_{\mathcal{F}} \neg B$, wenn $\Gamma, B \vdash_{\mathcal{F}} \neg A$ gilt.

Beweis. 1.(a).
"$\Longrightarrow$": Falls $\Gamma \vdash_{\mathcal{F}} A \to B$, dann gibt es dafür einen Beweis $C_1 C_2 \ldots C_n$ der Form

$$\begin{array}{ll} C_1 & \Gamma \\ \vdots & \vdots \\ C_n & A \to B \end{array}$$

Dann gibt es aber auch einen Beweis $D_1 D_2 \dots D_{n+1}$ für $\Gamma, A \vdash_{\mathcal{F}} A \to B$, nämlich

$$\begin{array}{ll} D_1 & A \\ D_2 & \Gamma \\ \vdots & \vdots \\ D_{n+1} & A \to B \end{array}$$

mit $D_{j+1} \equiv C_j$ für $1 \leq j \leq n$.
Somit:

$$\begin{array}{lll} B_1 & \Gamma \vdash A \to B & \text{Voraussetzung} \\ B_2 & \Gamma, A \vdash A \to B & \text{s.o.} \\ B_3 & \Gamma, A \vdash A & A \text{ ist aus } A \text{ herleitbar} \\ B_4 & \Gamma, A \vdash B & \text{MP}(B_2, B_3) \end{array}$$

"$\Longleftarrow$": Induktion über die Zahl n der Beweisschritte für $\Gamma, A \vdash B$:
$n = 1$: B ist selbst Axiom oder Hypothese. Falls B ein Axiom ist oder $B \in \Gamma$, dann

$$\begin{array}{lll} B_1 & \Gamma \vdash B & \text{Axiom oder Hypothese} \\ B_2 & \Gamma \vdash B \to (A \to B) & \text{Ax1} \\ B_3 & \Gamma \vdash A \to B & \text{MP}(B_1, B_2) \end{array}$$

Ist $B \equiv A$, dann (da $\vdash_{\mathcal{F}} A \to A$ (zeigen!))

$$B_1 \qquad \Gamma \vdash A \to A$$

$n \to n+1$: Für den Fall, dass B Axiom oder Hypothese ist, wurde der Beweis bereits angegeben. Ansonsten resultiert B im Beweis für $\Gamma, A \vdash B$ aus vorangegangenen Schritten durch Anwendung der Modus-Ponens-Regel. Es stellt sich also folgende Situation dar:

$$\begin{array}{rl} \vdots & \vdots \\ \text{Schritt } j: & \Gamma, A \vdash C \\ \vdots & \vdots \\ \text{Schritt } k: & \Gamma, A \vdash C \to B \\ \vdots & \vdots \\ \text{Schritt } n+1: & \Gamma, A \vdash B \end{array}$$

mit $j, k \leq n$. Da für den j-ten und k-ten Schritt die Induktionsvoraussetzung bereits gilt, erhält man den folgenden Beweis für $\Gamma \vdash A \to B$:

$$\begin{array}{lll} B_1 & \Gamma \vdash A \to C & \text{nach Induktionsvoraussetzung} \\ B_2 & \Gamma \vdash A \to (C \to B) & \text{nach Induktionsvoraussetzung} \\ B_3 & \Gamma \vdash (A \to (C \to B)) \to ((A \to C) \to (A \to B)) & \text{Ax2} \\ B_4 & \Gamma \vdash A \to B & \text{2mal MP} \end{array}$$

1.(b).

Der Beweis lässt sich analog zu dem Beweis aus 1.(a). führen. Es muss im Beweis "$\Longleftarrow$" beim Induktionsschritt jedoch zusätzlich der Fall betrachtet werden, dass der $n+1$-te (letzte) Schritt im Beweis von $\Gamma, A \vdash B$ durch die Anwendung der Generalisierungsregel erfolgte, d.h.

$$\begin{array}{rl} \vdots & \vdots \\ \text{Schritt } j: & \Gamma, A \vdash C \\ \vdots & \vdots \\ \text{Schritt } n+1: & \Gamma, A \vdash \forall x\ C \end{array}$$

mit $B \equiv \forall x\ C$, $j \leq n$ und x kommt in A nicht frei vor. Man erhält dann als Beweis für $\Gamma \vdash A \to B$:

B_1	$\Gamma \vdash A \to C$	nach Induktionsvoraussetzung
B_2	$\Gamma \vdash \forall x\ (A \to C)$	Gen(B_1)
Da x in A nicht frei vorkommt:		
B_3	$\Gamma \vdash \forall x\ (A \to C) \to (A \to \forall x\ C)$	Ax5
B_4	$\Gamma \vdash A \to \forall x\ C$	MP(B_2, B_3)

1.(c).

Der Beweis für die Variante $\Gamma, A \vdash_{\mathcal{F}'} B$ genau dann, wenn $\Gamma \vdash_{\mathcal{F}'} \tilde{A} \to B$, wobei $\tilde{A}$ universeller Abschluss von A ist :

"$\Longleftarrow$":

B_1	$\Gamma \vdash \tilde{A} \to B$	Voraussetzung
B_2	$\Gamma, A \vdash \tilde{A} \to B$	vgl. 1.(a)$\Rightarrow$
B_3	$\Gamma, A \vdash A$	A ist aus A herleitbar
B_4	$\Gamma, A \vdash \forall x_1\ A$	Anwendung der
B_5	$\Gamma, A \vdash \forall x_2 \forall x_1\ A$	Generalisierungsregel bis alle freien
	$\vdots$	Vorkommen von Variablen x_j in A
B_{3+k}	$\Gamma, A \vdash \tilde{A}$	$(1 \leq j \leq k)$ gebunden sind
B_{3+k+1}	$\Gamma, A \vdash B$	MP(B_2, B_{3+k})

"$\Longrightarrow$": Induktion über die Zahl n der Beweisschritte für $\Gamma, A \vdash B$:

$n = 1$: B ist selbst Axiom oder Hypothese. Falls B ein Axiom ist oder $B \in \Gamma$, dann

B_1	$\Gamma \vdash B$	Axiom oder Hypothese
B_2	$\Gamma \vdash B \to (\tilde{A} \to B)$	Ax1
B_3	$\Gamma \vdash \tilde{A} \to B$	MP(B_1, B_2)

Ist $B \equiv A$, dann (da $\vdash \tilde{A} \to A$ (zeigen!)):

$$B_1 \quad \Gamma \vdash \tilde{A} \to A$$

$n \to n+1$: Für den Fall, dass B Axiom oder Hypothese ist, wurde die Aussage bereits gezeigt. Ansonsten resultiert B aus vorangegangenen Schritten im Beweis für $\Gamma, A \vdash B$ durch Anwendung der Modus-Ponens-Regel oder der Generalisierungsregel. Es stellt sich also eine der beiden folgenden Situationen ein:

$$(\star) \qquad \begin{array}{rl} \vdots & \vdots \\ \text{Schritt } j: & \Gamma, A \vdash C \\ \vdots & \vdots \\ \text{Schritt } k: & \Gamma, A \vdash C \to B \\ \vdots & \vdots \\ \text{Schritt } n+1: & \Gamma, A \vdash B \end{array}$$

mit $j, k \leq n$ oder

$$(\star\star) \qquad \begin{array}{rl} \vdots & \vdots \\ \text{Schritt } j: & \Gamma, A \vdash C \\ \vdots & \vdots \\ \text{Schritt } n+1: & \Gamma, A \vdash \forall x\ C \end{array}$$

mit $B \equiv \forall x\ C,\ j \leq n$.
Als Beweis für $\Gamma \vdash \tilde{A} \to B$ erhält man im Fall $(\star)$:

B_1	$\Gamma \vdash \tilde{A} \to C$	nach Induktionsvoraussetzung
B_2	$\Gamma \vdash \tilde{A} \to (C \to B)$	nach Induktionsvoraussetzung
B_3	$\Gamma \vdash (\tilde{A} \to (C \to B)) \to ((\tilde{A} \to C) \to (\tilde{A} \to B))$	Ax2
B_4	$\Gamma \vdash \tilde{A} \to B$	2mal MP

Im Fall $(\star\star)$ ergibt sich folgender Beweis:

B_1 $\quad \Gamma \vdash \tilde{A} \to C$ $\qquad$ nach Induktionsvoraussetzung
B_2 $\quad \Gamma \vdash \forall x\ (\tilde{A} \to C)$ $\qquad$ Gen(B_1)
Da in $\tilde{A}$ keine Variable frei vorkommt (also auch x nicht) :
B_3 $\quad \Gamma \vdash \forall x\ (\tilde{A} \to C) \to (\tilde{A} \to \forall x\ C)$ $\qquad$ Ax5
B_4 $\quad \Gamma \vdash \tilde{A} \to \forall x\ C$ $\qquad$ MP(B_2, B_3)

2.(a).
Sei $B_0, \ldots, B_m$ ein Beweis für $\Gamma \vdash_{\mathcal{F}} A$. Zeige: $\Gamma \vdash \forall x\ B_j$ $(j = 0, \ldots, m)$.
Induktion nach j:

- $B_j \in \text{Ax}$ $\surd$
- $B_j \in \Gamma$

B_j	Voraussetzung
$B_j \to \forall x\ B_j$	Ax6, da x nicht frei in Γ
$\forall x\ B_j$	MP

- B_j entsteht durch MP aus B_h und B_k, d.h. $B_k \equiv B_h \to B_j$, dann lässt sich folgender Beweis formulieren:

C_1	$\Gamma \vdash \forall x\ B_h$	Induktionsvoraussetzung
C_2	$\Gamma \vdash \forall x\ (B_h \to B_j)$	Induktionsvoraussetzung
C_3	$\Gamma \vdash \forall x(B_h \to B_j) \to (\forall x\ B_h \to \forall x\ B_j)$	Ax5
C_4	$\Gamma \vdash \forall x\ B_h \to \forall x\ B_j$	MP(C_2, C_3)
C_5	$\Gamma \vdash \forall x\ B_j$	MP(C_1, C_4)

2b.

"$\Longrightarrow$":

B_1	$\Gamma \vdash A$	
B_2	$\Gamma \vdash \forall x\ A$	Gen(B_1)

"$\Longleftarrow$":

B_1	$\Gamma \vdash \forall x\ A$	
B_2	$\Gamma \vdash \forall x\ A \to A_x[x]$	Ax4, Substitution erlaubt
B_3	$\Gamma \vdash A$	MP(B_1, B_2)

3.

Mit der Voraussetzung $\vdash_{\mathcal{F}} (D \to \neg E) \to (E \to \neg D)$ (Variante von Ax3 (zeigen!)):

"$\Longrightarrow$": Es gelte $\Gamma, A \vdash_{\mathcal{F}} \neg B$. Dann folgt aus dem Deduktionstheorem $\Gamma \vdash_{\mathcal{F}} A \to \neg B$. Ein Beweis für $\Gamma \vdash_{\mathcal{F}} B \to \neg A$ ist dann:

$\Gamma \vdash A \to \neg B$	
$\Gamma \vdash (A \to \neg B) \to (B \to \neg A)$	
$\Gamma \vdash B \to \neg A$	(MP)

Die Behauptung $\Gamma, B \vdash_{\mathcal{F}} \neg A$ folgt nun mit dem Deduktionstheorem.

"$\Longleftarrow$": symmetrisch zu "$\Longrightarrow$" ■

2.27 Definition. Sei $\Gamma \subseteq$ Form. Γ heißt *konsistent*, falls es kein $A \in$ Form gibt, mit $\Gamma \vdash_{\mathcal{F}} A$ und $\Gamma \vdash_{\mathcal{F}} \neg A$.

Beachte: Γ ist genau dann konsistent, wenn jede endliche Teilmenge von Γ konsistent ist. Ist Γ *inkonsistent* (d.h. nicht konsistent), dann gilt $\Gamma \vdash_{\mathcal{F}} A$ für jede Formel A.

Genau dann ist $\Gamma \cup \{\neg A\}$ inkonsistent, wenn $\Gamma \vdash_{\mathcal{F}} A$ gilt; genau dann ist $\Gamma \cup \{A\}$ inkonsistent, wenn $\Gamma \vdash_{\mathcal{F}} \neg A$ gilt.

Ist Γ inkonsistent, so ist Γ nicht erfüllbar: Sei nämlich A eine Formel mit $\Gamma \vdash_{\mathcal{F}} A$ und $\Gamma \vdash_{\mathcal{F}} \neg A$ und sei I eine Interpretation, die Γ erfüllt, dann folgt wegen $\Gamma \models A$ und $\Gamma \models \neg A$, dass I sowohl A als auch $\neg A$ erfüllt. Widerspruch.

Die Menge der allgemeingültigen Formeln ist konsistent.

2.28 Satz (Gödel). *Vollständigkeit der Axiomatisierung:* Seien $A \in$ Form, $\Sigma \subseteq$ Form, dann gilt:

1. äquivalent sind
   - $\models A$,
   - $\vdash_{\mathcal{F}} A$ und
   - $\vdash_{\mathcal{F}'} A$.
2. Σ ist genau dann konsistent, wenn Σ erfüllbar ist.
3. Genau dann gilt $\Sigma \vdash_{\mathcal{F}} A$, wenn $\Sigma \models A$ gilt.

Beweis. s. Yasuhara [1971] bzw. Enderton [1972]

2.29 Definition. Sei L eine Sprache erster Stufe. $\Gamma \subseteq \text{Form}_L$ heißt *logische Theorie erster Stufe*, falls Γ abgeschlossen ist gegenüber logischer Folgerung, d.h. für alle $A \in \text{Form}$ gilt $\Gamma \models A$ genau dann, wenn $A \in \Gamma$.

2.30 Bemerkungen. Sei L Sprache erster Stufe.

1. $T = \{A \mid A \text{ allgemeingültig}\}$ ist eine Theorie. Sie ist in jeder Theorie über L enthalten.
2. $T_\Sigma = \{A \mid \Sigma \models A\}$ ist eine Theorie ($\Sigma \subseteq \text{Form}$).

   Angenommen $T_\Sigma \models A$ und I sei eine Interpretation, die Σ erfüllt, dann ist $I(T_\Sigma) = 1$ und $I(A) = 1$, d.h. $\Sigma \models A$ und $A \in T_\Sigma$. T_Σ ist die von Σ erzeugte Theorie erster Stufe.
3. Ist T eine Theorie, dann gilt $T \vdash A$ genau dann, wenn $A \in T$.

   Insbesondere: Ist T inkonsistent, dann $T = \text{Form}$.
4. Sei R eine Struktur in L (d.h. es genügt Interpretationen $I = (D, I_c)$ zu betrachten), dann ist

   $$T_R := \{A \in \text{Form} \mid R \models \tilde{A},\ \text{wobei } \tilde{A} \text{ universeller Abschluss von } A \text{ ist}\}$$

   eine Theorie.
5. $T \subseteq \text{Form}$ ist genau dann eine Theorie, wenn $\{A \mid\ \models A\} \subseteq T$ und T abgeschlossen gegenüber Modus Ponens ist.

2.31 Definition. Sei T eine Theorie.

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

2.32 Bemerkungen.

1. T_R ist vollständig für jede Struktur R. Außerdem ist T_R konsistent.
2. T ist genau dann erfüllbar, wenn T konsistent ist.
3. Ist T rekursiv axiomatisierbar, dann ist T auch rekursiv aufzählbar.
4. Ist T vollständig, konsistent und rekursiv axiomatisierbar, dann ist T entscheidbar.

   Falls für jede abgeschlossene Formel A genau dann $A \in T$ gilt, wenn $\neg A \notin T$ gilt, so ist "$A \in T$" entscheidbar für abgeschlossene Formeln, da T rekursiv aufzählbar ist.
5. Ist T vollständig und konsistent, dann gilt $T = T_R$ für eine Struktur R.

2.33 Beispiele.

1. Th($\mathbb{N}$) ist das Relationalsystem $< \mathbb{N}; 0, S, +, \star; = >$, das die "natürliche" Interpretation der Sprache der Arithmetik über den Konstanten $0, S, +$ und $\star$ bezeichnet. (Ist $n \in \mathbb{N}$, dann hat das entsprechende Element $\tilde{n} \in L$ die Form $\tilde{n} = S(S(\ldots S(0)\ldots))$. Es gilt (Gödel):

   (a) Th($\mathbb{N}$) ist nicht rekursiv!

   (b) Th($\mathbb{N}$) ist nicht rekursiv axiomatisierbar!

   Eine Konsequenz ist, dass die Peano-Axiome $P_1 \ldots P_6$ keine Axiomatisierung von Th($\mathbb{N}$) bilden.

   P_1 $\forall x \forall y\ S(x) = S(y) \rightarrow x = y$
   P_2 $\forall x\ S(x) \neq 0$
   P_3 $\forall x\ x + 0 = x$
   P_4 $\forall x \forall y\ x + S(y) = S(x + y)$
   P_5 $\forall x\ x \star 0 = 0$
   P_6 $\forall x \forall y\ x \star S(y) = x \star y + x$
   P_7 $A_x[0] \rightarrow (\forall x\ (A \rightarrow A_x[S(x)]) \rightarrow \forall x\ A)$,
   falls x die einzige in Formel A frei vorkommende Variable ist.

2. Die Peano-Arithmetik (auch Arithmetik erster Stufe genannt), ist die Menge aller Formeln (erster Stufe) über der Basis $B = (\{0, 1, +, \star\}, \{<\})$, die in der Interpretation $I = (\mathbb{N}, I_c)$ allgemeingültig sind. Dabei soll I_c den Funktionssymbolen und dem Prädikatssymbol die übliche Bedeutung geben. Die Preßburger-Arithmetik entspricht der Peano-Arithmetik, es entfällt jedoch das Multiplikationssymbol.

   Es gilt: Die Peano-Arithmetik und die Preßburger-Arithmetik sind vollständige Theorien. Die Preßburger-Arithmetik ist entscheidbar und axiomatisierbar. Die Peano-Arithmetik ist nicht entscheidbar, nicht rekursiv aufzählbar und nicht axiomatisierbar.

3. Für die Theorie $\mathrm{Th}(\mathbb{R})$ ($\mathbb{R}$ bezeichnet die reellen Zahlen) über einer Sprache in den Konstanten $0, 1, +, \star$ und $<$ mit ihrer "natürlichen" Interpretation gilt:

   $\mathrm{Th}(\mathbb{R})$ ist rekursiv entscheidbar!

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

   Eine Axiomatisierung für $\mathrm{Th}(\mathbb{R})$ besteht aus den Axiomen für Körper (+ Nullstellen für Polynome) und zusätzlich:

$$\begin{array}{l} \neg(x < x) \\ x < y \land y < z \rightarrow x < z \\ x < y \lor y = x \lor y < x \\ x < y \rightarrow x + z < y + z \\ 0 < x \lor 0 < y \rightarrow 0 < x \star y \\ 0 < x \rightarrow \exists y \ (y \star y = x) \end{array}$$

4. Sei L eine Sprache erster Stufe mit Gleichheit und dem zweistelligen Prädikatssymbol $<$. L habe keine Konstanten und Funktionssymbole. Sei DLO die zu L gehörende Theorie mit folgenden Axiomen:

$$\begin{array}{l} \forall x \forall y \forall z \ (x < y \land y < z \rightarrow x < z) \\ \forall x \forall y \ (x < y \lor x = y \lor y < x) \\ \forall x \ \neg(x < x) \\ \forall x \forall y \exists z \ (x < y \rightarrow (x < z \land z < y)) \\ \forall x \exists y \exists z \ (y < x \land x < z) \end{array}$$

   Interpretationen, die DLO erfüllen, sind dichte, linear geordnete Systeme ohne Endpunkte. Beispiele dafür sind die reellen Zahlen und die rationalen Zahlen, sowie offene Intervalle davon. DLO ist eine vollständige und entscheidbare Theorie [Yasuhara, 1971].

2.4 Semantische Tableaux

Im Rest dieses Kapitels werden zwei Aufzählungsverfahren für die Prädikatenlogik erster Stufe vorgestellt:

Frage: Gilt $\Sigma \models A$ für eine Formelmenge Σ und eine Formel A? Sei $\Sigma \subseteq \mathrm{Form}$. Dann ist $T_\Sigma = \{A \mid \Sigma \models A\}$ rekursiv aufzählbar. Gesucht wird also ein "einfaches" (effektives) Aufzählungsverfahren für die Formeln aus T_Σ. Es gibt dafür verschiedene Möglichkeiten: Zum einen legt die Möglichkeit der Axiomatisierung die Idee nahe, alle möglichen Beweise ihrer Länge nach aufzuzählen. Diese Möglichkeit ist aber normalerweise umständlich. Eine andere Möglichkeit beruht auf der Äquivalenz von $\Sigma \models A$ und "$\{\Sigma, \neg A\}$ nicht erfüllbar". Man versucht aus der Voraussetzung, dass $\{\Sigma, \neg A\}$ ein Modell hat, so schnell und so systematisch wie möglich einen Widerspruch herzuleiten, um $\Sigma \models A$ zu beweisen. Die Tableaux-Methode für die Prädikatenlogik erster Stufe beruht auf dieser Strategie.

2.34 Definition. *Tableaux* für die Prädikatenlogik erster Stufe.

Sei L eine prädikatenlogische Sprache erster Stufe in $\wedge, \vee, \neg, \rightarrow, \exists$ und $\forall$. Wie für die Aussagenlogik (vgl. 1.29) werden die Formeln der Sprache in Klassen unterteilt:

1. Atomare und negierte atomare Formeln
2. α-Formeln (analog der Aussagenlogik: $A \wedge B, \neg(A \vee B), \neg(A \rightarrow B)$ und $\neg\neg A$)
3. β-Formeln (analog zur Aussagenlogik: $\neg(A \wedge B), (A \vee B)$ und $(A \rightarrow B)$)

Mit der Einteilung in atomare, α- und β-Formeln sind jedoch nicht alle möglichen Formeln einer prädikatenlogischen Sprache erster Stufe behandelt. Für Formeln mit Quantoren werden zusätzlich die folgenden Klassen definiert:

4. γ-Formeln: Formeln der Struktur $\forall x\ A$ und $\neg\exists x\ A$ ($A \in$ Form)
5. δ-Formeln: Formeln der Struktur $\exists x\ A$ und $\neg\forall x\ A$ ($A \in$ Form)

Als Regeln werden neben den Regeln für α- und β-Formeln aus Definition 1.29 für γ- und δ-Formeln zusätzliche Regeln eingeführt:

<u>γ-Regeln:</u>

γ	$\forall x\ A$	$\neg\exists x\ A$
$\gamma[t]$	$A_x[t]$	$\neg A_x[t]$

dabei ist t ein beliebiger Term, der so gewählt ist, dass die Substitution erlaubt ist.

<u>δ-Regeln:</u>

δ	$\exists x\ A$	$\neg\forall x\ A$
$\delta[y]$	$A_x[y]$	$\neg A_x[y]$

dabei ist y eine Variable, mit der Einschränkung, dass y "neu" ist, d.h. y kommt noch nicht im Ast zu $A_x[y]$ (bzw. $\neg A_x[y]$) vor, und die Substitution ist erlaubt. Statt y kann auch eine neue Konstante "a" verwendet werden.

γ-Regeln und δ-Regeln werden bei der systematischen Tableau-Konstruktion wie früher α- bzw. β-Regeln angewendet. Es ist bei jeder Anwendung jedoch darauf zu achten, dass $A_x[t]$ eine erlaubte Substitution ist bzw. dass y "neu" ist.

δ-Formeln brauchen nur einmal berücksichtigt zu werden, und können dann abgeharkt werden. Dagegen können γ-Formeln nicht abgeharkt werden.

2.35 Lemma. Sei Γ eine Menge von Formeln mit Universum $\mathcal{U}$ (von Termen), für die folgendes gilt:

1. Für keine Formel A gilt $A \in \Gamma$ und $\neg A \in \Gamma$.

2. Für jede α-Formel in Γ gilt $\alpha_1, \alpha_2 \in \Gamma$.
3. Für jede β-Formel in Γ gilt $\beta_1 \in \Gamma$ oder $\beta_2 \in \Gamma$.
4. Für jede γ-Formel in Γ gilt $\gamma[t] \in \Gamma$ für alle $t \in \mathcal{U}$.
5. Für jede δ-Formel in Γ gibt es ein $t \in \mathcal{U}$ mit $\delta[t] \in \Gamma$.

Dann ist Γ erfüllbar!

Beweis. Der Beweis kann analog des Beweises zu Lemma 1.33 geführt werden. Dazu wird eine Interpretation $I = (\mathcal{U}, \ldots)$ betrachtet, mit

$$\begin{aligned} I(P(t_1, \ldots, t_n)) &= 0, \text{ falls } \neg P(t_1, \ldots t_n) \in \Gamma \text{ und} \\ I(P(t_1, \ldots, t_n)) &= 1 \text{ sonst,} \end{aligned}$$

für eine Formel P, die aus den Termen $t_1, .., t_n \in \mathcal{U}$ und Prädikatskonstanten zusammengesetzt ist.

Behauptung: $I(A) = 1$, für alle $A \in \Gamma$.

Induktion über den Aufbau von A :

Ist A atomar, eine α-Formel oder eine β-Formel, dann ist $I(A) = 1$, wenn $A \in \Gamma$ analog zum Beweis von Lemma 1.33.

Ist $(\Delta x\ A) \in \Gamma$ eine γ-Formel, $\Delta \in \{\forall, \neg\exists\}$, dann gilt für jedes $k \in \mathcal{U}$, dass $A_x[k] \in \Gamma$ und somit nach gilt Induktionsvoraussetzung $I(A_x[k]) = 1$, also ist auch $I(\Delta x\ A) = 1$.

Ist $(\Delta x\ A) \in \Gamma$ eine δ-Formel, $\Delta \in \{\exists, \neg\forall\}$, dann gibt es ein $k \in \mathcal{U}$, so dass $A_x[k] \in \Gamma$ und somit gilt nach Induktionsvoraussetzung $I(A_x[k]) = 1$. Dann ist aber auch nach Definition $I(\Delta x\ A) = 1$. ■

2.36 Bemerkung. In Sprachen, die die Gleichheit "=" enthalten, wird zusätzlich eine <u>ϵ-Regel</u> eingeführt:

$$\frac{t_1 = t_2}{A[t_1 \leftarrow t_2]},$$

d.h. in einer Formel A kann ein Term t_1 durch einen "gleichen" Term t_2 ersetzt werden, wenn im Ast zu A der Knoten mit der Markierung $t_1 = t_2$vorkommt.

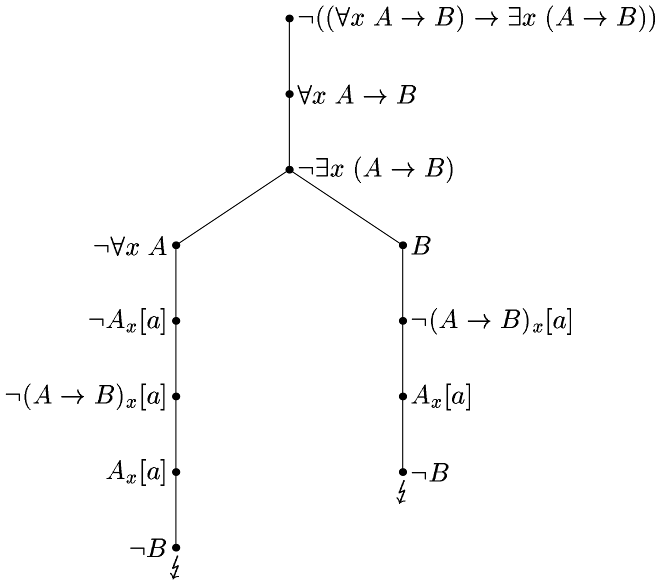
2.37 Satz. Sei L eine Sprache erster Stufe und sei $A \in \mathrm{Form}_L, \Sigma \subseteq \mathrm{Form}_L$.

1. $\models A$ gilt genau dann, wenn es gibt abgeschlossenes Tableau für $\neg A$ gibt.
2. $\Sigma \models A$ gilt genau dann, wenn es ein abgeschlossenes Tableau für $\Sigma \cup \{\neg A\}$ gibt.

Beweis. s. Smullyan [1968]

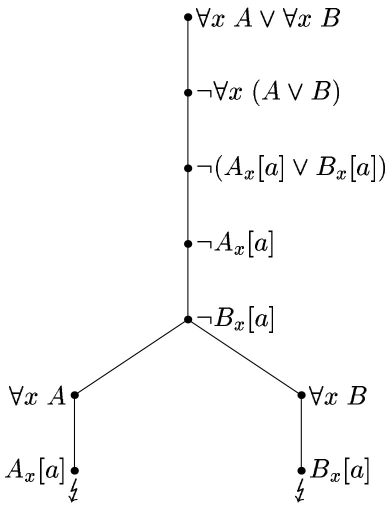
2.38 Beispiele.

1. $\models (\forall x\ A \to B) \to \exists x\ (A \to B)$, x soll in B nicht frei vorkommen



   Beachte dabei, dass $\neg B_x[a] \equiv \neg B$ ist, da x in B nicht frei vorkommt.

2. Sei $\Gamma = \{\forall x\ A \vee \forall x\ B\}$, $F = \forall x\ (A \vee B)$.
   Gilt $\Gamma \models F$?



   γ-Formeln können nicht abgeharkt werden. Stattdessen können immer wieder neue Terme t eingeführt werden, für die $\gamma[t]$ erfüllt werden muss.

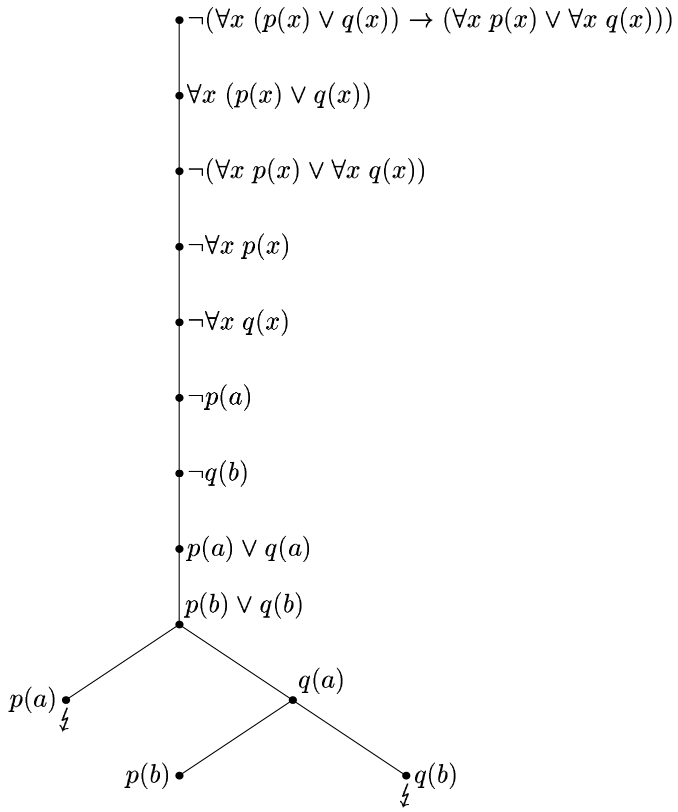
3. $\exists x\ \neg p(x,x) \not\models \neg\forall x \exists y\ p(x,y)$ Gesucht ist eine Interpretation, die Modell für $\Sigma = \{\exists x\ \neg p(x,x), \forall x \exists y\ p(x,y)\}$ ist.

$$\exists x\ \neg p(x,x)$$
$$\forall x \exists y\ p(x,y)$$
$$\neg p(a,a)$$
$$\exists y\ p(a,y)$$
$$p(a,b)$$
$$\exists y\ p(b,y)$$
$$p(b,c)$$
$$\vdots$$

In diesem Fall terminiert das Tableau nicht. Aus dem Tableau lässt sich jedoch eine Interpretation I über dem unendlichen Definitionsbereich $\{a, b, c, \ldots\}$ ablesen mit $I(p)(a,a) = 0, I(p)(a,b) = 1, I(p)(b,c) = 1, I(p)(c,d) = 1, \ldots,$ die Σ erfüllt.

4. Sei $A \equiv \forall x\ (p(x) \vee q(x)) \rightarrow (\forall x\ p(x)\ \vee\ \forall x\ q(x))$.

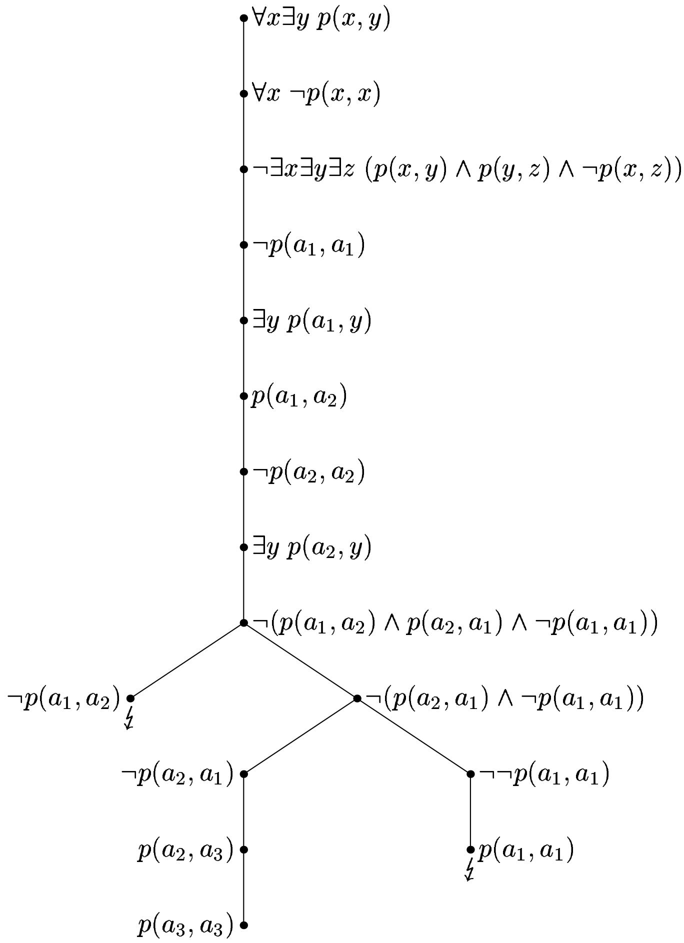
Frage: Ist A allgemeingültig ($\models A$)?



Offene vollständige Äste liefern Interpretationen, die $\neg A$ wahr machen. Dieses Tableau liefert die Interpretation $I = (\{a, b\}, \ldots)$ mit $I(p)(a) = 0, I(p)(a) = 1, I(q)(a) = 1$ und $I(q)(b) = 0$. Also ist $I(A) = 0$ und es gilt nicht $\models A$.

5. Seien $A_1 \equiv \forall x \exists y\ p(x, y)$, $A_2 \equiv \forall x\ \neg p(x, x)$ und $B \equiv \exists x \exists y \exists z\ (p(x, y) \wedge p(y, z) \wedge \neg p(x, z)$.

Frage: Gilt $\{A_1, A_2\} \models B$?



Aus dem Tableau kann die Interpretation I über $\{a_1, a_2, a_3\}$ abgelesen werden, mit $I(p)(a_1,a_1) = 0, I(p)(a_1,a_2) = 1, I(p)(a_2,a_2) = 0, I(p)(a_2,a_1) = 0, I(p)(a_2,a_3) = 1, I(p)(a_3,a_3) = 0$. Somit ist I ein Gegenbeispiel zu $\{A_1, A_2\} \models B$, d.h. $\{A_1, A_2\} \not\models B$,

Die Allgemeingültigkeit für beliebige Formeln über Sprachen der Prädikatenlogik erster Stufe ist unentscheidbar. Wie sieht das aber für spezielle Formelklassen aus? In folgenden Formeln sei A eine quantorenfreie Formel ohne Funktionskonstanten. $(n, m \in \mathbb{N})$

1. $\forall x_1 \ldots \forall x_m \exists y_1 \ldots \exists y_n \; A$,
2. $\forall x_1 \ldots \forall x_m \exists y \forall z_1 \ldots \forall z_n \; A$,
3. $\forall x_1 \ldots \forall x_m \exists y_1 \exists y_2 \forall z_1 \ldots \forall z_n \; A$,

4. $\exists y_1 \ldots \exists y_n \forall x_1 \ldots \forall x_m\ A,\ (m, n \geq 1)$ und

5. $\forall x_1 \ldots \forall x_m \exists y_1 \exists y_2 \exists y_3 \forall z_1 \ldots \forall z_n\ A,\ (m, n \geq 0)$.

Die Allgemeingültigkeit für Formeln der Art 1.,2. und 3. ist entscheidbar, für Formeln der Art 4. und 5. ist die Allgemeingültigkeit nicht entscheidbar [Manna].

2.5 Die Resolventen-Methode

Die Resolventen-Methode (auch Resolutionsmethode genannt) ist ein partielles Entscheidungsverfahren für die Unerfüllbarkeit einer gegebenen Formel einer Sprache der Prädikatenlogik erster Stufe. Das bedeutet, dass das Verfahren bei Eingabe einer nicht erfüllbaren Formel B die Unerfüllbarkeit von B entdeckt und hält, bei Eingabe einer erfüllbaren Formel jedoch "unendlich" weiter laufen kann. Um zu testen, ob eine gegebene Formel A allgemeingültig ist, wird man also die Resolventen-Methode auf die Formel $\neg A$ anwenden. Ist $\neg A$ unerfüllbar (d.h. A ist allgemeingültig), dann wird das Resolventenverfahren das feststellen und halten. Die Resolventenmethode ist Grundlage für automatische Beweisverfahren und wird zur Interpretation von "Logik programmen" eingesetzt.

Die Resolventen-Methode wird angewandt auf Formeln in einer speziellen *Klauselform*. In diesem Abschnitt wird zunächst ein effektives Verfahren eingeführt, mit dem man eine beliebige Formel A der Prädikatenlogik erster Stufe in eine Formel A' in Klauselform transformieren kann, ohne dass dabei die Eigenschaft von A erfüllbar zu sein oder nicht verloren geht. Danach wird auf den Satz von Herbrand eingegangen, der einen grundlegenden Einfluss auf die Entscheidungsprozeduren hat, die auf der Resolventen-Methode beruhen. Schließlich werden zwei solche Entscheidungsprozeduren vorgestellt, nämlich die Methode von Davis und Putnam und die Grundresolventen-Methode.

2.39 Definition. 1. Ein *Literal* ist eine atomare Formel oder eine negierte atomare Formel.

2. Eine *Klausel* ist die Disjunktion einer oder mehrerer Literale.

3. Eine Formel der Prädikatenlogik erster Stufe ist in *Klauselform* (KLF), wenn sie die Form hat

$$\forall x_1 \forall x_2 \ldots \forall x_n\ [C_1 \land C_2 \land \ldots \land C_k]$$

Dabei bezeichnen die C_j Klauseln, und die $x_1, x_2, \ldots, x_n$ sind alle Variablen, die in den C_j vorkommen $(1 \leq j \leq k)$. $\forall x_1 \forall x_2 \ldots \forall x_n$ heißt *Präfix* und $[C_1 \land C_2 \land \ldots \land C_k]$ die *Mantisse* der Formel.

Beachte: Eine Formel A ist genau dann in KLF, wenn A abgeschlossen und in PKNF ist und der Präfix kein $\exists$-Quantor enthält.

2.40 Lemma (Skolem). Jede Formel D der Prädikatenlogik erster Stufe kann in eine Formel D' in Klauselform transformiert werden, so dass D genau dann erfüllbar ist, wenn D' erfüllbar ist.

Beweis. Die Transformation geschieht durch die Konstruktion einer endlichen Folge von Formeln $D_1, \ldots, D_n$, so dass $D_1 \equiv D$ und $D_n \equiv D'$ und für jedes j, mit $1 \leq j < n$, gilt: D_j ist genau dann erfüllbar, wenn D_{j+1} erfüllbar ist. Das Transformationsverfahren besteht aus 9 Schritten. Bis auf Schritt 7 ist die Invarianz im Bezug auf die Erfüllbarkeit für alle Schritte offensichtlich. Eine Rechtfertigung für Schritt 7 wird im Anschluss gegeben.

Verfahren: Gegeben eine Formel D.

Schritt 1: Bilde den existentiellen Abschluss von D.

Schritt 2: Eliminiere in D alle überflüssigen Quantoren.

Schritt 3: Benenne gebundene Variablen, die mehrfach in D quantifiziert werden um.

Schritt 4: Eliminiere durch logische Umformungen in D alle Operatoren, die von $\wedge, \vee, \neg$ verschieden sind.

Beispiel: Ersetze $(A \to B)$ durch $(\neg A \vee B)$ und (IF A THEN B ELSE C) durch $(\neg A \vee B) \wedge (A \vee C)$.

Schritt 5: Schiebe alle $\neg$ so weit nach innen, bis sie vor einem (positiven) Literal stehen.

Beispiel: Ersetze $(\neg \forall x\ A)$ durch $(\exists x\ \neg A)$, $\neg(A \vee B)$ durch $(\neg A \wedge \neg B)$ und $\neg\neg A$ durch A.

Schritt 6: Ziehe die Quantoren nach rechts.

Beispiel: Ersetze $\Delta x\ (A \star B)$ durch $(A \star \Delta x\ B)$, falls x nicht frei in A vorkommt, und durch $(\Delta x\ A \star B)$, falls x in B nicht frei vorkommt ($\Delta \in \{\exists, \forall\}, \star \in \{\vee, \wedge\}$).

Schritt 7 : Eliminiere existentielle Quantoren durch Einführung von *Skolem-Funktionen* (eventuell auch nullstelligen): Wähle die erste Teilformel (von links) der Form $\exists y\ B(y)$ und ersetze sie durch $B_y[f(x_1, \ldots, x_n)]$, $(n \geq 0)$, wobei

1. $x_1, \ldots, x_n$ alle unterschiedlichen freien Variablen in $\exists y\ B(y)$ sind, die links von $\exists y\ B(y)$ universell quantifiziert sind, und
2. f eine n-stellige Funktionskonstante ist, die noch nicht vorkommt.

Schritt 8 : Schiebe die $\forall$-Quantoren nach links.

Schritt 9 : Bringe die Matrix (durch Anwendung der Distributivgesetze) in Konjunktive Normalform.

Für die durch einmalige Anwendung der Ersetzung aus Schritt 7 aus einer Formel $D_j = \exists y\ B(y)$ entstehende Formel $D_{j+1} = B_y[f(x_1, \ldots, x_n)]$ gilt: D ist genau dann erfüllbar, wenn D_{j+1} erfüllbar ist. Denn: Ist D_j erfüllbar, dann muss es dafür ein Modell I geben, so dass es für jede mögliche Belegung für die Variablen $x_1, \ldots, x_n$ ein oder mehrere Werte für y gibt, mit denen D_j wahr wird. Betrachte nun eine Interpretation I', die zu I identisch ist, die aber zusätzlich eine n-stellige Funktionskonstante f so interpretiert, dass $f(x_1, \ldots, x_n)$ für jede Belegung von $x_1, \ldots, x_n$ gerade das Funktionsergebnis liefert, das y annehmen kann um D_j wahr zu machen. Dann ist I' ein Modell für D_{j+1} (d.h. D_{j+1} ist erfüllbar). Mit einem ähnlichen Argument (ausgehend von einem Modell für D_{j+1}) kann die Umkehrrichtung gezeigt werden. ∎

2.41 Beispiel. Transformieren einer Formel in eine Formel in KLF. Betrachte die Formel

$$A \equiv \forall x \, \{p(x) \to \exists z \, \{\neg \forall y \, [q(x,y) \to p(f(x_1))] \land \forall y \, [q(x,y) \to p(x)]\}\}.$$

1. Existentieller Abschluss und Elimination von Quantoren:

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

2. Umbenennung von y:

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

3. Elimination von $\to$:

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

4. $\neg$ nach innen:

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

5. Schiebe Quantoren nach rechts:

$$\exists x_1 \forall x \, \{\neg p(x) \lor \{[\exists y \; q(x,y) \land \neg p(f(x_1))] \land [\forall z \; \neg q(x,z) \lor p(x)]\}\}$$

6. Eliminieren der Ex.-Quantoren $\exists x_1$ und $\exists y$:

$$\forall x \, \{\neg p(x) \lor \{[q(x,g(x)) \land \neg p(f(a))] \land [\forall z \; \neg q(x,z) \lor p(x)]\}\}$$

7. Quantoren nach links:

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

8. Distributivgesetze anwenden:

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

Schließlich erhält man nach Vereinfachungen die Formel in Klauselform:

$$\forall x \, \{[\neg p(x) \lor q(x,g(x))] \land \neg p(f(a))\}.$$

Nach Definition ist eine Formel A unerfüllbar genau dann, wenn $I(A) = 0$ für jede Interpretation I, also insbesondere für jeden Definitionsbereich für I. Beim Beweis für die Unerfüllbarkeit von A wäre es jedoch von großem Nutzen, wenn man irgendeinen ("kleineren") festen Definitionsbereich D so bestimmen könnte, dass A in D genau dann unerfüllbar ist, wenn A überhaupt unerfüllbar ist. Tatsächlich gibt es einen solchen Definitionsbereich.

2.42 Definition. Sei $A \in$ Form. Das *Herbrand-Universum* (H_A) für eine Formel A ist die Menge aller Terme, die aus den Individuenkonstanten und den Funktionskonstanten, die in A vorkommen, gebildet werden können. Falls aus den Individuen- und den Funktionskonstanten, die in A vorkommen, keine Terme gebildet werden können, ist $H_A := \{a\}$ mit einer Individuenkonstanten a.

2.43 Beispiel. Sei $A \equiv \forall x \; \{[\neg p(x) \vee q(x, g(x))] \wedge \neg p(f(a))\}$, dann ist $H_A = \{a, f(a), g(a), g(g(a)), g(f(a)), f(f(a)), f(g(a)), g(g(g(a))), g(g(f(a))), \ldots\}$

2.44 Definition. Eine *Herbrandinterpretation* für eine (geschlossene) Formel A ist eine Interpretation $I = (H_A, \ldots)$, die

1. jeder Individuenkonstante a in A den Term a aus H_A zuordnet,
2. jeder n-stelligen ($n \geq 1$) Funktionskonstanten f, die in A vorkommt, eine Funktion $I(f) : {H_A}^n \rightarrow H_A$ zuordnet, die die Terme $t_1, t_2, \ldots, t_n \in H_A$ auf den Term (als Zeichenkette) $f(t_1, \ldots, t_n) \in H_A$ abbildet.

Herbrandinterpretationen unterscheiden sich also nur auf den Prädikatskonstanten.

2.45 Satz. Eine Formel A in KLF ist genau dann unerfüllbar, wenn A in allen Herbrandinterpretationen unerfüllbar ist [Manna, 1974].

Sei A eine Formel in KLF der Form $A \equiv \forall x_1 \ldots \forall x_n \; [C_1 \wedge \ldots \wedge C_k]$. Die *Grundinstanz* einer Klausel C_j $(1 \leq j \leq k)$ von A ist die Klausel, die man erhält, wenn man alle Individuenvariablen in C_j durch Terme aus H_A ersetzt. Solche Klauseln, die keine Individuenvariablen enthalten heißen *Grundklauseln.*

2.46 Satz (Herbrand). Eine Formel A in Klauselform ist genau dann unerfüllbar, wenn es eine endliche Konjunktion von Grundinstanzen ihrer Klauseln gibt, die unerfüllbar ist.

Dieser Satz legt ein Verfahren nahe die Unerfüllbarkeit einer Formel in Klauselform zu testen. Das Verfahren ist unter dem Namen "Herbrand-Prozedur" bekannt.

2.47 Herbrand-Prozedur. Sei A eine Formel in KLF mit n Klauseln. Erzeuge der Reihe nach alle Grundklauseln G_j aus den Klauseln C_j $(1 \leq j \leq n)$ von A und prüfe, ob ihre Konjunktion unerfüllbar ist.

Bei Durchführung der Herbrand-Prozedur tauchen zwei Probleme auf. Zum einen stellt sich die Frage, wie man die Grundklauseln systematisch erzeugen kann, und zum anderen, wie man testen kann, ob eine Konjunktion von Grundklauseln unerfüllbar ist.

Es gibt zwei bekannte Verfahren, die Unerfüllbarkeit einer Konjunktion von Klauseln zu testen: die Methode von Davis und Putnam und die Grundresolventen-Methode.

Hier soll die *Grundresolventen-Methode* vorgestellt werden. Sie beruht auf der (Grund-) Resolventenregel.

2.48 Definition. Die *(Grund-) Resolventenregel* kann wie folgt beschrieben werden:

Für je zwei Grundklauseln G und H, von denen eine ein Literal l und die andere dessen Komplement $\neg l$ enthält, konstruiere eine neue Klausel K als Disjunktion von G' und H'. Dabei bezeichne G' bzw. H' die Formeln, die man aus G bzw. H erhält, wenn man alle Vorkommen von l bzw. $\neg l$ in G und H streicht. Doppelte Vorkommen von Literalen in K werden gestrichen. K heißt *Resolvent(e)* von G und H. Im Spezialfall, dass die betrachteten Klauseln jeweils nur aus dem Literal l bzw. $\neg l$ bestehen, ist die entstehende Klausel K die *leere Klausel*, notiert "$\Box$".

Das folgende Lemma beschreibt das Hauptresultat des Verfahrens. Es beruht auf der Beobachtung, dass $S = G_1 \wedge G_2 \wedge \ldots \wedge G_n$ genau dann unerfüllbar ist, wenn $S \wedge G_{n+1}$ unerfüllbar ist, wobei G_{n+1} eine Resolvente aus zwei Klauseln von S ist.

2.49 Lemma. Eine endliche Konjunktion von Grundklauseln $S = G_1 \wedge G_2 \wedge \ldots \wedge G_n$ ist genau dann unerfüllbar, wenn durch wiederholte Anwendung der Grundresolventenregel die leere Klausel $\Box$ aus S hergeleitet werden kann.

Verfahren:

Starte für eine Klauselform $S = G_1 \wedge G_2 \wedge \ldots \wedge G_n$ mit

$$\begin{array}{c} G_1 \\ G_2 \\ \vdots \\ G_n \end{array}$$

und bilde alle möglichen Resolventen. Wird $\Box$ hergeleitet, dann ist S unerfüllbar, sonst ist S erfüllbar. Das Verfahren stoppt, da mit endlich vielen Literalen nur endlich viele Klauseln gebildet werden können.

2.50 Beispiele.

1.

$$\begin{aligned} S \quad \equiv \quad & (p_1(a) \vee p_2(b)) \wedge \\ & (p_1(a) \vee \neg p_3(a)) \wedge \\ & (\neg p_1(a) \vee p_3(a)) \wedge \\ & (\neg p_1(a) \vee \neg p_2(b)) \wedge \\ & (p_3(a) \vee \neg p_2(b)) \wedge \\ & (\neg p_3(a) \vee p_2(b)) \end{aligned}$$

1.	$p_1(a) \vee p_2(b)$	
2.	$p_1(a) \vee \neg p_3(a)$	
3.	$\neg p_1(a) \vee p_3(a)$	
4.	$\neg p_1(a) \vee \neg p_2(b)$	
5.	$p_3(a) \vee \neg p_2(b)$	
6.	$\neg p_3(a) \vee p_2(b)$	
7.	$p_2(b) \vee p_3(a)$	$R(1,3)$
8.	$p_2(b)$	$R(6,7)$
9.	$\neg p_3(a) \vee \neg p_2(b)$	$R(2,4)$
10.	$\neg p_2(b)$	$R(5,9)$
11.	$\square$	$R(8,10)$

2. $S \equiv (p_1(a) \vee p_2(b)) \wedge \neg p_2(b) \wedge (\neg p_1(a) \vee p_2(b) \vee p_3(a))$

1.	$p_1(a) \vee p_2(b)$	
2.	$\neg p_2(b)$	
3.	$\neg p_1(a) \vee p_2(b) \vee p_3(a)$	
4.	$p_1(a)$	$R(1,2)$
5.	$p_2(b) \vee p_3(a)$	$R(1,3)$
6.	$\neg p_1(a) \vee p_3(a)$	$R(2,3)$
7.	$p_3(a)$	$R(2,5)$

Es können keine weiteren Klauseln erzeugt werden. S ist also erfüllbar in Interpretationen I, mit $I(p_1)(a) = 1, I(p_3)(a) = 0$ und $I(p_2)(b) = 0$.

Zwar liegt mit der Grundresolventenregel ein Verfahren vor, das Unerfüllbarkeitsproblem für eine endliche Konjunktion von Grundklauseln zu entscheiden. Das zweite wichtige Problem bei der Implementierung des Herbrandverfahrens ist damit jedoch noch nicht gelöst: Die systematische Erzeugung der Grundinstanzen zu einer Klausel. Für die meisten prädikatenlogischen Formeln müssen sehr viele Grundinstanzen erzeugt werden, bevor eine nichterfüllbare Konjunktion darunter ist. Um das Erzeugen von Grundinstanzen zu umgehen, führte Robinson die Resolutions-Regel ein. Sie ist eine Generalisierung der Grundresolventenregel: Man kann sie direkt auf eine Konjunktion von Klauseln (also nicht unbedingt auf Grundklauseln) anwenden, um deren Unerfüllbarkeit zu zeigen.

2.51 Beispiel. Frage: Ist

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

allgemeingültig? Zuerst wird $\neg A$ in eine Formel B in Klauselform transformiert:

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

Das Herbranduniversum für B lautet $H_B = \{a, f(a), g(a), f(f(a)), f(g(a)), \ldots\}$
Um mit der Grundresolventenmethode zu zeigen, dass B unerfüllbar ist, generiert man nacheinander die Grundinstanzen für die Klauseln aus B und testet sukzessive, ob deren Konjunktion unerfüllbar ist: $[p(a, f(a)) \wedge q(a, g(a)) \wedge [\neg p(a,a) \vee \neg q(a,a)] \ldots \wedge [p(a, f(a)) \wedge q(f(a), g(f(a))) \wedge [\neg p(a, f(a)) \vee \neg q(f(a), g(f(a)))]]$.

Erst aus dieser Konjunktion kann die leere Klausel hergeleitet werden:

1.	$p(a, f(a))$	
2.	$q(f(a), g(f(a)))$	
3.	$\neg p(a, f(a)) \vee \neg q(f(a), g(f(a)))$	
4.	$\neg q(f(a), g(f(a)))$	$R(1,3)$
5.	$\square$	$R(2,4)$

Die Idee der allgemeinen Resolutionsmethode ist, die Klauseln (einer Formel wie B) direkt zur Resolution heranzuziehen, ohne den Umweg über ihre Grundinstanzen zu machen. Am Beispiel der Formel B soll hier zunächst illustriert werden, wie das Verfahren arbeitet:

Zunächst die Klauseln von B :

1. $p(x, f(x))$
2. $q(y, g(y))$
3. $\neg p(a, y_1) \vee \neg q(y_1, z)$

Betrachte die Klauseln 1. und 3. Zwar ist das Literal aus 1. zu keinem Literal der dritten Klausel invers, ersetzt man jedoch die Variable x durch die Konstante a und die Variable y_1 durch den (konstanten) Term $f(a)$, dann erhält man die Klauseln

$1'$. $p(a, f(a))$

$3'$. $\neg p(a, f(a)) \vee \neg q(f(a), z)$

Eine solche Ersetzung heißt *Substitution* und wird notiert als

$$\{\langle x, a\rangle, \langle y_1, f(a)\rangle\}.$$

Die Substitution leistet im obigen Fall etwas besonderes, durch sie werden $p(x, f(x))$ und $p(a, y_1)$ zur selben Formel $p(a, f(a))$. Die Substitution $\{\langle x, a\rangle, \langle y_1, f(a)\rangle\}$ wird deshalb auch *Unifikator* genannt. Man könnte natürlich noch zusätzlich die Variable z durch a oder $f(a)$ usw. substituieren. Die dann entstehende Substitution wäre ebenfalls ein Unifikator. Sie würde aber Informationen enthalten, die zum Unifizieren ("Gleichmachen") von $p(x, f(x))$ und $p(a, y_1)$ nicht notwendig sind. In diesem Sinne ist der Unifikator $\{\langle x, a\rangle, \langle y_1, f(a)\rangle\}$ allgemeiner. Ja, es gibt in diesem Falle keinen Unifikator, der allgemeiner ist. $\{\langle x, a\rangle, \langle y_1, f(a)\rangle\}$ wird deshalb auch *Allgemeinster Unifikator* oder *Most General Unifier* (MGU) genannt.

Resolviert man nun die Klauseln $1'$. und $3'$. miteinander, dann erhält man

4. $\neg q(f(a), z)$

als Resolvente. Durch Anwenden der Substitution $\{\langle y, f(a)\rangle, \langle z, g(f(a))\rangle\}$ auf die Klauseln 2. und 4. werden diese unifiziert und man erhält als Resolvente aus beiden:

5. $\square$

2.52 Definition. Eine *Substitution* σ ist eine endliche Menge der Form

$$\{\langle v_1, t_1\rangle, \ldots, \langle v_n, t_n\rangle\},$$

wobei die v_j $(1 \leq j \leq n)$ voneinander verschiedene Variablen sind und die t_j von v_j verschiedene Terme sind. Jedes Element $\langle v_j, t_j\rangle$ heißt *Bindung* für v_j.

2.53 Definition. Sei $\sigma = \{\langle v_1, t_1\rangle, \ldots, \langle v_n, t_n\rangle\}$ eine Substitution und E ein Literal. Dann ist $E\sigma$, die *Instanz* von E, das Literal, das aus E durch gleichzeitiges Ersetzen jedes Vorkommens der Variablen v_j in E durch die Terme t_j $(j = 1, \ldots, n)$ entsteht.

2.54 Definition. Seien $\sigma = \{\langle u_1, s_1\rangle, \ldots, \langle u_n, s_n\rangle\}$ und $\tau = \{\langle v_1, t_1\rangle, \ldots, \langle v_m, t_m\rangle\}$ Substitutionen. Die *Komposition* $\sigma\tau$ von σ und τ ist die Substitution, die man aus der Menge $\{\langle u_1, s_1\tau\rangle, \ldots, \langle u_n, s_n\tau\rangle, \langle v_1, t_1\rangle, \ldots, \langle v_m, t_m\rangle\}$ erhält, wenn man jede Bindung $\langle u_j, s_j\tau\rangle$ mit $u_j = s_j\tau$ $(1 \leq j \leq n)$ und jede Bindung $\langle v_k, t_k\rangle$ mit $v_k \in \{u_1, \ldots, u_n\}$ $(1 \leq k \leq m)$ daraus entfernt.

2.55 Definition. Die durch die leere Menge gegebene Substitution e heißt *Identitätssubstitution*, d.h. $xe = x$ für jede Variable x.

2.56 Definition. Sei $S = A_1 \vee \ldots \vee A_n$ eine Disjunktion atomarer Formeln A_j $(1 \leq j \leq n)$. Eine Substitution σ heißt *Unifikator* für S, falls $A_1\sigma = A_2\sigma = \ldots = A_n\sigma$ gilt. Gibt es für S einen solchen Unifikator, dann heißt S *unifizierbar*. Ein Unifikator heißt *allgemeinster Unifikator* oder *most general unifier* (MGU) für S, wenn es für jeden Unifikator τ von S eine Substitution υ gibt, so dass $\tau = \sigma\upsilon$.

2.57 Definition. Sei S eine endliche Disjunktion von Literalen. Das *"disagreement set"* von S wird wie folgt definiert: Suche die am weitesten links stehende Symbolposition in den Literalen von S, an denen sich die Literale unterscheiden und bestimme für jedes Literal in S den Teilterm, der an der gefundenen Position beginnt. Die Menge all dieser Teilausdrücke ist das "disagreement set".

2.58 Beispiel. Das "disagreement set" für die Disjunktion

$$p(x, g(f(y,z), x), y) \vee p(x, g(a, b), b) \vee p(x, g(g(h(x), a)y, h(x))$$

ist die Menge $\{f(y,z), a, g(h(x), a)\}$.

2.59 Satz (Unifikationstheorem). Gegeben folgender Unifikationsalgorithmus:

Unifikations - Algorithmus

Eingabe: $S \equiv A_1 \vee \ldots \vee A_n$, eine Disjunktion von atomaren Formeln $A_1, \ldots, A_n$.

Ausgabe: MGU σ_k für S, falls dieser existiert, oder eine Meldung, dass S nicht unifizierbar ist sonst.

Verfahren:

1. Setze $k = 0$ und $\sigma_k = e$.

2. Falls σ_k Unifikator für S ist, dann stoppen, σ_k ist gesuchter MGU.

   Ansonsten finde das "disagreement set" D_k von $S\sigma_k \equiv A_1\sigma_k \vee \ldots \vee A_n\sigma_k$.

3. Gibt es v und t in D_k derart, dass v eine Variable ist, die nicht in t vorkommt, dann setze $\sigma_{k+1} := \sigma_k \cup \{\langle v, t\rangle\}$, erhöhe k um 1 und fahre mit Schritt 2 fort.

   Ansonsten melde, dass S nicht unifizierbar ist und stoppe.

Falls S unifizierbar ist, hält der Algorithmus und liefert einen MGU für S als Ergebnis. Ist S nicht unifizierbar, hält der Algorithmus und meldet, dass S nicht unifizierbar ist.

Die nachfolgend vorgestellte Resolutions-Regel basiert auf der Resolventenregel.

2.60 Definition (Resolutions-Regel). Seien C_1 und C_2 Klauseln ohne gemeinsame Variablen (Diese Bedingung ist keine Einschränkung, denn sie kann durch die Transformation einer Klausel C_j in eine äquivalente Klausel C'_j ($j \in \{1, 2\}$), in der die Variablen, durch die die Bedingung verletzt wurde umbenannt wurden, stets erfüllt werden.) und ist $p(t_1^{(1)}, \ldots, t_k^{(1)}) \vee \ldots \vee p(t_1^{(n)}, \ldots, t_k^{(n)})$ eine Teildisjunktion von C_1 und $\neg p(s_1^{(1)}, \ldots, s_k^{(1)}) \vee \ldots \vee \neg p(s_1^{(m)}, \ldots, s_k^{(m)})$ eine Teildisjunktion von C_2, so dass es einen MGU β gibt für die Disjunktion $p(t_1^{(1)}, \ldots, t_k^{(1)}) \vee \ldots \vee p(t_1^{(n)}, \ldots, t_k^{(n)}) \vee \neg p(s_1^{(1)}, \ldots, s_k^{(1)}) \vee \ldots \vee \neg p(s_1^{(m)}, \ldots, s_k^{(m)})$ mit dem Faktor $p(r_1, \ldots, r_k)$ der Disjunktion. Dann konstruiere als *Resolvente* von C_1 und C_2 die Klausel C_3 aus der Disjunktion von $C_1\beta$ und $C_2\beta$ nach Streichen von $p(r_1, \ldots, r_k)$ aus $C_1\beta$ und $\neg p(r_1, \ldots, r_k)$ aus $C_2\beta$.

2.61 Satz (Robinson). Eine Formel S in Klauselform ist genau dann unerfüllbar, wenn die leere Klausel aus S mit der Resolutions-Regel hergeleitet werden kann.

2.62 Beispiel. Um zu beweisen, dass die Formel

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

allgemeingültig ist, wird mit der Resolutionsmethode gezeigt, dass

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

unerfüllbar ist. Die $\neg A$ entsprechende Formel in Klauselform lautet

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

und enthält die Klauseln (nach Umbenennung der Variablen)

1. $\neg p(z_1, a) \vee \neg p(z_1, x_1) \vee \neg p(x_1, z_1)$
2. $p(z_2, f(z_2)) \vee p(z_2, a)$
3. $p(f(z_3), z_3) \vee p(z_3, a)$

Um die Klauseln 1. und 2. zu resolvieren, werden die Teildisjunktionen $\neg p(z_1, a) \vee \neg p(z_1, x_1) \vee \neg p(x_1, z_1)$ aus Klausel 1. und $p(z_2, a)$ aus Klausel 2. betrachtet. Der MGU von $\neg p(z_1, a) \vee \neg p(z_1, x_1) \vee \neg p(x_1, z_1) \vee p(z_2, a)$ lautet

$$\{\langle x_1, a\rangle, \langle z_2, a\rangle, \langle z_1, a\rangle\}.$$

Deshalb erhält man als Resolvente der 1. und 2. Klausel

$$4. \quad p(a, f(a))$$

Ähnlich erhält man aus den Klauseln 1. und 3. mit den Teildisjunktionen $\neg p(z_1, a) \vee \neg p(z_1, x_1) \vee \neg p(x_1, z_1)$ und $p(z_3, a)$ den MGU $\{\langle x_1, a\rangle, \langle z_1, a\rangle, \langle z_3, a\rangle\}$ und die Resolvente

$$5. \quad p(f(a), a)$$

Als MGU von $\neg p(x_1, z_1) \vee p(a, f(a))$ erhält man $\{\langle x_1, a\rangle, \langle z_1, f(a)\rangle\}$ und deshalb als Resolvente von Klausel 1. und Klausel 4:

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

Schließlich erhält man die leere Klausel als Resolvente der Klauseln 5. und 6.

$$7. \quad \square$$

Also ist die Formel $\neg A$ unerfüllbar und A allgemeingültig.

Literatur

Dies war nur ein kleiner Einblick in Methoden der formalen Logik. Mittlerweile sind die Entwicklungen auf diesem Gebiet fortgeschritten und eine nähere Betrachtung ist nicht Ziel dieser Einführung. Der interessierte Leser kann Hinweise auf weiterführende Literatur in den Referenzen finden.

Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. *Einfhrung in die mathematische Logik.* Spektrum, Heidelberg, 1998.

H. B. Enderton. *A Mathematical Introduction to Logic.* Academic Press, 1972.

Jean H. Gallier. *Logic for Computer Science.* Harper & Row, New York, 1986.

B. Heinemann and K. Weihrauch. *Logik für Informatiker.* Teubner, Stuttgart, 1991.

H.-J. Kreowski. *Logische Grundlagen der Informatik.* Oldenbourg, München, 1991.

Z. Manna. *Mathematical Theory of Computation.* McGraw-Hill, 1974.

Nimal Nissanke. *Introductory Logic and Sets for Computer Scientists.* Addison-Weslay, Harlow, 1999.

Uwe Schöning. *Logik für Informatiker.* Number 56 in Reihe Informatik. Wissenschaftsverlag, Mannheim, 1989.

R. M. Smullyan. *First-Order Logic.* Springer, 1968.

V. Sperschneider and G. Antoniou. *Logic: a foundation for computer science.* Addison-Wesley, Wokingham, 1991.

A. Yasuhara. *Recursive Function Theory Logic.* Academic Press, 1971.