Logik für Informatiker

Prof. Dr. W. Vogler

Wintersemester 2007/2008

Diese Vorlesungsnotizen beruhen auf einer LATEX-Mitschrift von Nils Drews, Matthias Görig, Florian Lautenbacher, Anselm Aggensteiner, Robert Klaus, Elger Thiele und Stefan Grottenthaler, bei denen ich mich sehr bedanke.

Inhaltsverzeichnis

Einführung

Logik: formale Behandlung des exakten Denkens und Folgerns;
beantwortet die Frage „Was ist ein mathematischer Beweis?“

in der Informatik:

- Korrektheitsbeweise (Verifikation) von Programmen, Schaltkreisen etc.; diese benötigen Rechnerunterstützung; dazu: *formale* Angabe von Aussagen, speziell von
- Spezifikationen (Logik ist *ausdrucksstark* bzw. soll es sein)
- automatisches Beweisen
- logische Programmierung (PROLOG) (Beweis als Programmablauf)
- semantic web: Webseite um Inhaltsbeschreibung/-katalogisierung ergänzen – „Logik pur“

Literatur zum Thema Logik

M. Ben-Ari: Mathematical Logic for Computer Science. Prentice Hall

H.-D. Ebbinghaus, J. Flum, W. Thomas: Einführung in die mathematische Logik. Spektrum Akademischer Verlag, 4. Auflage

M. Kreuzer, S. Kühling: Logik für Informatiker. Pearson Studium 2006

U. Schöning: Logik für Informatiker. BI-Wissenschaftsverlag, Reihe Informatik Bd. 56

Logische Formeln (wie auch Programme) sind Zeichenketten bzw. *syntaktische Objekte*, z.B. $P(x)$. Durch *Interpretation* der Symbole (z.B. $P \widehat{=}$ „ist prim“) geben wir der Formel eine *Bedeutung (Semantik).*

Auch bei Programmen müssen wir nicht nur wissen, was syntaktisch korrekt ist, also vom Compiler akzeptiert wird, sondern auch die Bedeutung kennen; unter Bedeutung kann man die Ausführung bzw. das Ergebnis des Programms verstehen. Auch ein syntaktisch korrektes Programm kann das falsche Ergebnis liefern, also semantisch falsch sein.

Ein essentieller Begriff ist *Folgerung. Aus* A *folgt* B heißt: Ist A unter einer Interpretation wahr, so auch B (semantischer Begriff). Bsp.: aus $P(x)$ folgt $\neg\neg P(x)$

Wichtiges Ziel: B aus A durch Zeichenmanipulation (syntaktisch) *herleiten* (vgl. Chomsky-Grammatiken).

Ein *Kalkül* beschreibt die Herleitbarkeit durch logische *Axiome* und *Schlußregeln.* Ein Kalkül ist *korrekt*, wenn alles Herleitbare gefolgert werden kann (also wahr ist), und *vollständig*, wenn alles Folgerbare (Wahre) hergeleitet werden kann. Ein solcher Kalkül erlaubt die genaue Erfassung der Wahrheit (semantisch) durch Zeichenmanipulationen (syntaktisch)! Dies gilt nur im Rahmen des Formulierbaren, es ist also wichtig, daß die Logik *ausdrucksstark* ist.

Syntaktisches Herleiten bedeutet, dass die Axiome und Schlußregeln sich nur an der syntaktischen Struktur der Aussagen, nicht aber an ihrer Bedeutung (Semantik) orientieren sollen.

Beispiel: Gegeben seien die Annahmen
– „Arbeit macht reich.“
– „Armut macht schön.“
Welche der nachfolgenden Aussagen folgen logisch aus den folgenden Annahmen?
– „Arbeit macht häßlich.“
– „Arme arbeiten nicht.“
– „Schöne können nicht reich sein.“
– „Arbeitende können nicht schön sein.“
– „Durch Arbeit verliert man eine Chance, schön zu werden.“

Eine ganz analoge Struktur haben die Annahmen
– „Aufräumen schafft Ordnung.“
– „Unordnung ist kreativ.“
und die Aussagen
– „Aufräumen tötet die Kreativität.“
– „Unordentliche räumen nicht auf.“
– „Kreative können keine Ordnung erreichen.“
– „Aufräumer können nicht kreativ sein.“
– „Durch Aufräumen verliert man eine Chance, kreativ zu werden.“

Ein brauchbarer logischer Kalkül sollte also in der Lage sein, beide Aufgabengruppen einheitlich zu behandeln.

Später:
- Programmverifikation
- temporale Logik

1 Syntax und Semantik der Prädikatenlogik 1. Stufe

Dies ist eine Logik mit Elementen, Funktionen, Prädikaten und Quantifikation über Elemente ($\forall x\ P(x)$).

Prädikat: Relation wie $<$ (2-stellig) oder auch „ist prim“ (1-stellig).

In der 2. Stufe wird auch die Quantifikation über Prädikate behandelt. (Erwartet hätte man vielleicht Quantifikation über Mengen; dies ist ein Spezialfall: 1-stellige Prädikate entsprechen Mengen, s.u.)

Beispiel: Um in $\mathbb{N}_0$ $\forall x\ P(x)$ zu zeigen, verwenden wir Induktion, denn es gilt das Induktionsprinzip: $\forall P\ \ P(0) \wedge (\forall x\ \ P(x) \rightarrow P(x+1)) \rightarrow \forall x P(x)$

1.1 Syntax

Gegeben seien Zeichen $x_0, x_1, \ldots$ (Variablen); auch: $x, y, z \in \{x_0, x_1, \ldots\} =: \mathcal{X}$.

Eine *Signatur* ist ein Paar $(\mathcal{F}, \mathcal{P})$, wobei $\mathcal{F}$ und $\mathcal{P}$ disjunkte, endliche oder abzählbar unendliche (entscheidbare) Zeichenmengen sind, mit $x_i, \forall, \ldots \notin \mathcal{F} \cup \mathcal{P}$; implizit sei:

$\mathcal{F} = \mathcal{F}^0 \dot\cup \mathcal{F}^1 \dot\cup \mathcal{F}^2 \dot\cup \ldots$

$\mathcal{P} = \mathcal{P}^0 \dot\cup \mathcal{P}^1 \dot\cup \mathcal{P}^2 \dot\cup \ldots$

$f, g, h \in \mathcal{F}^n$: *n-stellige Funktionssymbole*

$c \in \mathcal{F}^0$: *Konstante*

$P, Q, R \in \mathcal{P}^n$: *n-stellige Prädikatensymbole* (auch: $p, q, r \in \mathcal{P}^0$)

Die Menge $Term_{\mathcal{F},\mathcal{P}}$ (kurz *Term*) der $\mathcal{F}$,$\mathcal{P}$-*Terme* (kurz: *Terme*) ist die kleinste Menge mit:

(T1) $x_0, x_1, \cdots \in Term$

(T2) Wenn $f \in \mathcal{F}^n$ und $t_1, \ldots, t_n \in Term$, dann $f(t_1, \ldots, t_n) \in Term$; speziell: $c \in Term$

„kleinste Menge“ heißt: *Term* enthält nichts, was nicht aufgrund dieser Regeln in *Term* sein muß.

Ein Term ist also z.B. $g(f(x_0), c)$; wie üblich stehen die Funktionssymbole vor den Argumenten – *Präfix-Schreibweise*. In arithmetischen Termen haben wir zweistellige Funktionen $+$, $-$ etc., schreiben aber üblicherweise nicht $+(x_0, -(c, d))$ sondern $x_0 + (c - d)$. (*Infix-Schreibweise*; diese werden wir gelegentlich auch verwenden.) Graphische Darstellung von Termen: Syntaxbaum.

Anderes Beispiel für Terme (aus der Theoretischen Informatik): rationale (bzw. reguläre) Ausdrücke über einem Alphabet Σ; dabei ist $\mathcal{F} = \mathcal{F}^0 \dot\cup \mathcal{F}^1 \dot\cup \mathcal{F}^2$, $\mathcal{F}^0 = \Sigma \cup \{\lambda, \emptyset\}$, $\mathcal{F}^1 = \{^*\}$ und $\mathcal{F}^2 = \{+, \cdot\}$; z.B. $(a + b \cdot c)^*$.

Aus der Sicht der funktionalen Programmierung sind die x_i, f und c Konstruktoren.

Wegen oben „kleinste Menge“ sind nur nach den beiden Regeln aufgebaute Objekte Terme . Der Aufbau nach den Regeln wird durch eine *Herleitung* beschrieben, d.h. jeder Term hat eine Herleitung; z.B. sei $f \in \mathcal{F}^1, g \in \mathcal{F}^2$:

(1)	x_0 (d.h. $x_0 \in \mathit{Term}$)	T1 (Begründung)
(2)	$f(x_0)$	T2 (1)
(3)	c	T2
(4)	$g(f(x_0), c)$	T2 (2),(3)

Herleitung: bottom-up (erst Teile herleiten)

Bem.: T1 + T2 entsprechen „unendlicher kontextfreier Grammatik“

$$\text{Term} ::= x \mid c \mid f(\underbrace{\text{Term}, \ldots, \text{Term}}_{n})$$

mit $x \in \{x_0, x_1, \ldots\}$, $c \in \mathcal{F}^0$, $f \in \mathcal{F}^n$, $n \geq 1$.

Ableitung in Grammatik: top-down (im Ableitungsbaum) z.B.: Term $\Rightarrow$ g(Term, Term) $\Rightarrow$ $g(f(\text{Term}), \text{Term}) \Rightarrow g(f(x_0), \text{Term}) \Rightarrow g(f(x_0), c)$ □

Regeln wie T1, T2 (analog A0 – A5 unten) kann man in folgender Form schreiben:

$$\frac{t_1, \ldots, t_n}{t} \qquad \frac{\mathit{Prämissen}}{\mathit{Konklusion}}$$

Idee: Wenn $t_1, \ldots, t_n$ Terme (oder Formeln (s.u.), gültige Aussagen,...) sind, so auch t.

Speziell für n=0:

$$\frac{}{t} \qquad \mathit{Axiom}$$

Damit sind T1 und T2:

(T1') $\dfrac{}{x}$ $\quad x \in \{x_0, \ldots\}$ (Nebenbedingung)

(T2') $\dfrac{t_1, \ldots, t_n}{f(t_1, \ldots, t_n)}$ $\quad f \in \mathcal{F}^n$ $\quad$ speziell: $\dfrac{}{c}$ $\quad c \in \mathcal{F}^0$

Eine *Herleitung* von t_0 für solche Regeln ist eine Folge $w_1, \ldots, w_m$ von Zeichenketten mit $w_m \equiv t_0$ (syntaktisch gleich, d.h. gleiche Zeichenketten), so daß für jedes $i = 1, \ldots, m$ eine Regel $\frac{t_1, \ldots, t_n}{t}$ mit $w_i \equiv t$ existiert, für die $t_1, \ldots, t_n$ unter den $w_1, \ldots, w_{i-1}$ auftreten. Wir numerieren die w_i explizit und geben jeweils die angewandte Regel an zusammen mit den Zeilennummern der t_i.

Beispiel: siehe oben; $w_1 \equiv x_0$, $w_2 \equiv f(x_0)$, $w_3 \equiv c$, $w_4 \equiv g(f(x_0), c)$.

Z.B. für w_4 besagt Regel (T2) $\frac{f(x_0),\ c}{g(f(x_0),c)}$, und $f(x_0)$ und c sind w_2 und w_3. □

Man kann nun Definitionen/Beweise für herleitbare Objekte durch *Induktion* (über die Herleitungslänge m) durchführen. Ausführlich:

Objekt t hat mindestens eine Herleitung; sei also eine solche Herleitung der Länge m gegeben. Nach Induktion liegt Definition/Beweis für alle Objekte mit kürzerer Herleitung schon vor (Noethersche Ind.!).

Die zuletzt angewandte Regel zur Herleitung von t sei $\frac{t_1, \ldots, t_n}{t}$ – hier muß man jeweils eine Fallunterscheidung nach den gerade behandelten Regeln machen. Die t_i erscheinen in der gegebenen Herleitung vor t; sie haben also kürzere Herleitungen, so daß nach Ind. Definition/Beweis für sie schon vorliegt. Gib basierend darauf Definition/Beweis für t.

Bei einem solchen im Prinzip immer wieder gleichen Vorgehen hat man für jede Regel einen Fall. Man gibt nur die Liste dieser Fälle und schreibt jeweils kurz:

Regel $\frac{t_1,\ldots,t_n}{t}$: Definition/Beweis für t basierend auf Definition/Beweis für die t_i

Bei Termen (Formeln s.u.) sind die t_i Teilterme von t; man spricht von *Induktion über den Termaufbau* bzw. *struktureller Induktion.* Vgl. rationale Ausdrücke.

Beispiele:

1) Die Funktionszahl $\#_F t$ von t ist definiert durch:

   (T1) $\#_F x = 0$

   (T2) $\#_F f(t_1,\ldots,t_n) =$
   (Spezialfall: $\#_F c = 1$)

   Die Fallunterscheidung nach der Form von t wird deutlicher, wenn man „$t \equiv x \in \mathcal{X}$“ statt (T1) und „$t \equiv f(t_1,\ldots,t_n)$ mit $f \in \mathcal{F}^n$, $t_1,\ldots,t_n \in \mathit{Term}$“ statt (T2) schreibt.

   Achtung: Definition ist eindeutig, weil Zerlegung in Teilterme eindeutig ist.

2) Jeder Term hat eine gerade Anzahl von Klammern.
   Beweis durch strukturelle Induktion:

   (T1) x hat 0 Klammern.

   (T2) t_i habe k_i Klammern, nach Induktionsbeh. sei k_i gerade. Dann hat $f(t_1,\ldots,t_n)$ $2+k_1+k_2+\cdots+k_n$ Klammern, was als Summe gerader Zahlen gerade ist. (Speziell: c hat 0 Klammern) □

Bem.:

- Warum Noethersche Ind. (Schluß von „alle $n < m$“ auf m) statt Schluß von m auf $m+1$?

  Die kürzeren Herleitungen sind in der Regel nicht um 1 kürzer.

- Warum keine Verankerung? bzw.: Sind nicht die Axiome die Verankerung?

  Verankerung nicht nötig. Im Fall $m = 1$ ist die Regel ein Axiom; in diesem Fall machen wir gar keine Annahmen, d.h. wir behandeln die Verankerung implizit.

  Axiome können in einer Herleitung häufiger angewandt werden, also auch beim Schritt auf $m > 1$ auftreten; wir müssen sie also auf jeden Fall im Induktionsschritt berücksichtigen und sparen uns daher die Verankerung.

□

Atomare $\mathcal{F}$,$\mathcal{P}$-Formeln: kleinste Menge $At_{\mathcal{F},\mathcal{P}}$ (At) mit:

(A1) Sind $t_1, t_2 \in \mathit{Term}$, so ist $t_1 = t_2 \in At$.

$$\frac{}{t_1 = t_2} \quad t_1, t_2 \in \mathit{Term}$$

(A2) Sind $P \in \mathcal{P}^n$, $t_1,\ldots,t_n \in \mathit{Term}$, so $P(t_1,\ldots,t_n) \in At$; speziell: $p \in At$

Bem.: "=“ ist ein spezielles 2-stelliges Prädikat. Man kann auch *Logik ohne Gleichheit* betrachten, dann fällt (A1) weg. □

$\mathcal{F}$,$\mathcal{P}$-Formeln: kleinste Menge $For_{\mathcal{F},\mathcal{P}}$ (For) mit:

(A0) $At \subseteq For$

(A3) Ist $A \in For$, so auch $(\neg A) \in For$

(A4) Sind $A, B \in For$, so auch $(A \wedge B), (A \vee B), (A \to B), (A \leftrightarrow B) \in For$

(A5) Ist $A \in For$, so $(\forall x A), (\exists x A) \in For$

Wir führen die Formeln *true* und *false* ein als Abkürzungen für $p_0 \vee \neg p_0$ und $\neg true$ für ein festes $p_0 \in \mathcal{P}^0$. Zur verkürzenden Schreibweise lassen wir Klammern weg:

- Außenklammern
- Klammern bei aufeinanderfolgenden Negationen oder Quantoren: $\neg\neg A$ bzw. $\forall x \exists y\, Q(x, y)$
- Linksklammerung bei $A \vee B \vee C, A \wedge B \wedge C$
- Bindungsstärke: $\neg \gg \wedge \gg \vee \gg \to \gg \leftrightarrow \gg \forall, \exists$

  Der Bindungsbereich von $\forall, \exists$ endet so spät wie möglich.

  Also steht $\forall x \ \neg B \to (\forall y \ \ Q(x, y) \to P)$ für ...

Beispiel: Eine Formel, die besagt, daß $R \in \mathcal{P}^2$ eine Äquivalenzrelation ist:
$(\forall x R(x, x)) \ \wedge \ (\forall x \forall y \ \ R(x, y) \ \to \ R(y, x)) \ \wedge \ (\forall x \forall y \forall z \ \ R(x, y) \wedge R(y, z) \ \to \ R(x, z))$

Bem.:

- Da die Menge Σ^* der Wörter über einer abzählbaren Menge Σ abzählbar ist, ist auch $For_{\mathcal{F},\mathcal{P}}$ abzählbar.
- Die Zerlegung von Formeln und Termen in Teilformeln und Teilterme, aus denen sie nach (T1)-(T2) und (A0)-(A5) aufgebaut sind, ist eindeutig (und berechenbar).

1.2 Semantik

$f \in \mathcal{F}$ bzw. $P \in \mathcal{P}$ sollen als Funktion bzw. Relation über einer Menge verstanden werden, "=" immer als Gleichheit.

Eine *Interpretation* I einer Signatur $(\mathcal{F}, \mathcal{P})$ stützt sich auf eine nichtleere Menge D (oder D_I) (*Grundbereich*, domain) und ordnet jedem $f \in \mathcal{F}^n$ eine Funktion $f^I : D^n \to D$ (speziell: $c^I \in D$) und jedem $P \in \mathcal{P}^n$ ein Prädikat $P^I : D^n \to \{T, F\}$ (Spezialfall: $p^I \in \{T, F\}$) zu.

Bem.:

- f, P, p sind Zeichen, f^I, P^I, p^I sind Funktionen/Prädikate.
- Hier haben Werte nur eine Sorte D. Man kann auch mehrere Sorten und entsprechend getypte Variablen, Funktionen und Prädikate verwenden.
- Das Prädikat P^I entspricht der Relation $\{(d_1, \ldots, d_n) \mid P^I(d_1, \ldots, d_n)\}$. (Man beachte die Schreibweise.)

  (1-stellige Prädikate entsprechen Mengen: $P^I : D \to \{T, F\}$ entspricht der Menge $\{d \in D \mid P^I(d)\}$; vgl. charakteristische Funktion einer Menge, Darstellung einer Menge als Bitvektor bzw. „Eigenschaft aufzählbarer Sprachen" .)

□

Die Auswertung von Termen und Formeln wird nun induktiv definiert und folgt dabei den induktiven Definitionen (T1), ..., (A5) s.o.

Eine *Belegung* β zu I ist eine Funktion $\beta : \{x_0, x_1, \ldots\} \to D_I$. Ist $d \in D_I$, so ist β_x^d die Belegung mit:

$$\beta_x^d(y) = \begin{cases} d & \text{für } y \equiv x \\ \beta(y) & \text{für } y \not\equiv x \end{cases}$$

Die Funktionen β und β_x^d unterscheiden sich also (allenfalls) nur an der Stelle x.

Die *Auswertung eines Terms* t unter I und β ist $t_{I,\beta}$ mit

(vgl. T1) $x_{I,\beta} :=$

(vgl. T2) $f(t_1, \ldots, t_n)_{I,\beta} :=$

(Spezialfall: $c_{I,\beta} := \ldots$

Beispiel: $D = \mathbb{R}$, $g^I : (x, y) \to (x - y)$, $f^I : x \to x^2$, $\beta(x) = 3$.
Dann ist $g(f(x), x)_{I,\beta} = g^I(f(x)_{I,\beta}, x_{I,\beta}) = g^I(f^I(3), 3) = g^I(9, 3) = 6$.

Bem.: Der Mathematiker schreibt einfach $x := 5$ statt $x_{I,\beta} := 5$. Wir untersuchen Terme/Formeln (Zeichenketten) wie x, unterscheiden also x und seinen Wert deutlich. □

Wir definieren nun $I, \beta \models B$, d.h. B *gilt bei* (I, β), durch:

(vgl. A1) $I, \beta \models (t_1 = t_2) :\Leftrightarrow (t_1)_{I,\beta} = (t_2)_{I,\beta}$

Bemerkung: $\Leftrightarrow$ ist *unsere* Sprache, mit der wir über Formeln sprechen (*Metasprache*); $\leftrightarrow$ gehört zur Sprache der Formeln (*Objektsprache*). Analog: Im Deutschen über Englisch reden.

Beispiel: $D = \mathbb{Z}$, $f^I = +$, $\beta(x) = 1$ und $\beta(y) = 0$; weiter sei $f^{I'}$ die Multiplikation.

Dann ist $f(x,y)_{I,\beta} = 1$ und $f(x,y)_{I',\beta} = 0$; demnach gilt $I, \beta \models f(x,y) = x$, nicht aber $I', \beta \models f(x,y) = x$.

(vgl. A2) $I, \beta \models P(t_1, \ldots, t_n) :\Leftrightarrow$

(vgl. A3) $I, \beta \models \neg A :\Leftrightarrow$ nicht $I, \beta \models A \Leftrightarrow$: $I, \beta \not\models A$

(vgl. A4) $I, \beta \models A \wedge B \quad :\Leftrightarrow \quad I, \beta \models A$ und $I, \beta \models B$

$I, \beta \models A \vee B \quad :\Leftrightarrow \quad I, \beta \models A$ oder $I, \beta \models B$

$I, \beta \models A \rightarrow B \quad :\Leftrightarrow \quad I, \beta \not\models A$ oder $I, \beta \models B$

$I, \beta \models A \leftrightarrow B \quad :\Leftrightarrow \quad (I, \beta \not\models A$ und $I, \beta \not\models B)$ oder $(I, \beta \models A$ und $I, \beta \models B)$

($\forall x A$ heißt: $A(x)$ gilt für alle x-Werte; $I, \beta \models A$ heißt: $A(x)$ gilt für $\beta(x)$; also:)

(vgl. A5) $I, \beta \models \forall x A \quad :\Leftrightarrow \quad$ für alle $d \in D$ gilt $I, \beta^d_x \models A$

$I, \beta \models \exists x A \quad :\Leftrightarrow$

Definition von $\neg A, A \wedge B$, etc. tabellarisch; dabei steht A, $A \wedge B$ etc. für $I, \beta \models A$, $I, \beta \models A \wedge B$ etc.

A	B	$\neg A$	$A \wedge B$	$A \vee B$	A xor B	$A \rightarrow B$	$A \leftrightarrow B$
T	T	F	T	T	F	T	T
T	F	F	F	T	T	F	F
F	T	T	F	T	T	T	F
F	F	T	F	F	F	T	T

$\vee$ ist das inklusive, xor das exklusive Oder.

Bem.: $I, \beta \models A$ (bzw. $I, \beta \models A \wedge B$ etc.) ist *keine* $\mathcal{F}, \mathcal{P}$-Formel; daher macht $\square$

Beispiele: 1. p: Augsburg liegt in Thailand.

q: Augsburg liegt in Asien.

r: Augsburg liegt in Afrika.

$p \rightarrow r$ *gilt* mit der Realität als Interpretation — $p \rightarrow q$ natürlich auch. Wir benutzen $\rightarrow$ *extensional*, d.h. nur die Wahrheitswerte sind interessant.

Intentional ist $p \rightarrow r$ („Wenn Augsburg in Thailand *läge*, dann läge es in Afrika.“) wohl falsch.

$p \mathrel{\widehat{=}}$ Satz, der wahr oder falsch ist. Also nicht:

- Hau ab !
- Nachts ist es kälter als draußen.
- Der gegenwärtige König von Frankreich hat eine Glatze.
- Diser Saz enthält drei Fehler. (falsch, da nur 2 Schreibfehler; wahr, da ein Sinnfehler dazukommt)

Der letzte Satz zeigt, daß Sätze, die sich auf sich selbst beziehen, problematisch sind; weitere Beispiele:

- Dieser Satz ist falsch.
- Ein Kreter sagt: „Alle Kreter lügen."

2.

a) Max sagt: „Paul lügt."

b) Paul sagt: „Max lügt."

Sei p „Paul lügt"und m „Max lügt". Dann besagen a) und b) $(\neg m \leftrightarrow p)$ und $(\neg p \leftrightarrow m)$.

3. $D := \mathbb{N}_0, \qquad P^I :=<, \qquad I, \beta \models \exists y : P(y, x)$

– ist wahr für $\beta(x) \in \{1, \dots\}$; wähle $d := \beta(x) - 1$ oder $d := 0$.

– ist falsch für $\beta(x) = 0$.

Also ist $I, \beta \models \forall x \exists y : P(y, x)$ falsch. □

Proposition 1.1 *(Hier wie auch sonst bedeutet das folgende: „Für alle Interpretationen I, Belegungen β und Formeln A, B gilt. . . ")*

i) $I, \beta \models A \Leftrightarrow I, \beta \not\models \neg A \Leftrightarrow I, \beta \models \neg\neg A$

ii) $I, \beta \models A \wedge B \Leftrightarrow I, \beta \models \neg(A \rightarrow \neg B)$

iii) $I, \beta \models A \vee B \Leftrightarrow I, \beta \models \neg A \rightarrow B$

iv) $I, \beta \models A \leftrightarrow B \Leftrightarrow I, \beta \models (A \rightarrow B) \wedge (B \rightarrow A)$

v) $I, \beta \models \exists x A \Leftrightarrow I, \beta \models \neg\forall x \neg A$

Beweis:

i) $I, \beta \not\models \neg A$ bzw. $I, \beta \models \neg\neg A$
$\Leftrightarrow$ nicht $I, \beta \models \neg A$ (Definition $\not\models$ und $\neg$)
$\Leftrightarrow$ nicht nicht $I, \beta \models A$ (Definition $\neg$)
$\Leftrightarrow I, \beta \models A$

iii) $I, \beta \models A \vee B$
$\Leftrightarrow I, \beta \models A$ oder $I, \beta \models B$ (Definition $\vee$)
$\Leftrightarrow I, \beta \not\models \neg A$ oder $I, \beta \models B$ (Teil i))
$\Leftrightarrow I, \beta \models \neg A \rightarrow B$ (Definition $\rightarrow$)

v) $I, \beta \models \exists x A$
$\Leftrightarrow$ es gibt $d \in D$ mit $I, \beta_x^d \models A$ (Definition $\exists x$)
$\Leftrightarrow$ nicht für alle $d \in D$ gilt nicht $I, \beta_x^d \models A$
$\Leftrightarrow$ nicht für alle $d \in D : I, \beta_x^d \models \neg A$ (Definition $\neg$)
$\Leftrightarrow I, \beta \models \neg \forall x \neg A$ (Definition $\forall x, \neg$) □

□

Nur üblicher math. Beweis – wir streben größere Präzision an; dazu müssen wir unser formales Vorgehen mit „gesundem (math.) Menschenverstand“ absichern.

1.3 Modelle und Folgerbarkeit

Definition 1.2 (*Modell*)
Seien $A \in For$, $M \subseteq For$ und I eine Interpretation. Wir sagen A bzw. M *gilt* in I, I *erfüllt* A bzw. M und I ist ein *Modell* von A bzw. M, wenn $I \models A$ bzw. $I \models M$ gilt gemäß folgender Definition:

- $I \models A :\Leftrightarrow$ für alle Belegungen β gilt $I, \beta \models A$
- $I \models M :\Leftrightarrow$ für alle $A \in M$ gilt $I \models A$
  (Analog: $I, \beta \models M :\Leftrightarrow$ für alle $A \in M$ gilt $I, \beta \models A$

□

Bem.: $I \models \emptyset$ gilt immer □

Beispiele: 3) (wie vor Proposition 1.1)

- I ist kein Modell von $\forall x \exists y \;\; P(y, x)$ bzw. $\exists y \;\; P(y, x)$
- $I \models P(x, y) \vee P(y, x) \vee x = y$ (Die Formel ist gewissermaßen all-quantifiziert.)
- ACHTUNG: $\underbrace{I \models P(x, y),\ I \models P(y, x) \text{ bzw. } I \models x = y}_{\text{...sind alles falsche Aussagen!}}$
  Die Definition von $I, \beta \models A \vee B$ läßt sich also nicht auf $I \models A \vee B$ übertragen.

4) $D := \mathbb{N} \setminus \{0\}, \quad f^I := +, \quad P^I :=$ „ist prim“
Die sogenannte ... besagt ungefähr, daß I ein Modell ist von: $(\exists x_1 \;\; f(x_1, x_1) = x_0) \rightarrow \exists x_2 \exists x_3 \;\; P(x_2) \wedge P(x_3) \wedge f(x_2, x_3) = x_0$

Definition 1.3 (*Folgerbarkeit und Äquivalenz*)
Seien $A, B \in For$ und $M \subseteq For$; A *folgt aus* M (in Zeichen: $M \models A$), wenn für alle Interpretationen I und Belegungen β gilt: $I, \beta \models M \Rightarrow I, \beta \models A$.
Konvention: Wir schreiben auch $A_1, \ldots, A_n \models A$ für $\{A_1, \ldots, A_n\} \models A$.
A und B sind *logisch äquivalent*, $A \models\!\!\models B$, wenn $A \models B$ und $B \models A$. □

Beispiel: Für eine 2-stellige Funktion $\circ$ in Infix-Schreibweise und eine Konstante e seien

$$M_{gr} = \{\forall x \forall y \forall z \;\; (x \circ y) \circ z = x \circ (y \circ z), \;\; \forall x \;\; x \circ e = x, \;\; \forall x \exists y \;\; x \circ y = e\}$$

$$A :\equiv \forall x \exists y \;\; y \circ x = e$$

$M_{gr} \models A$ bedeutet: A ist ein *Satz der ...*, d.h. in jeder Interpretation / jedem Modell von M_{gr} ... gilt A, d.h. jedes Element .... Belegungen sind hier egal, siehe unten.

Bem.: Mathematik handelt also von Aussagen $M \models A$. □

Nach Proposition 1.1 sind $A \wedge B$ und $\neg(A \rightarrow \neg B)$, $A \vee B$ und $\neg A \rightarrow B$, $A \leftrightarrow B$ und $(A \rightarrow B) \wedge (B \rightarrow A)$, sowie $\exists x A$ und $\neg \forall x \neg A$ logisch äquivalent. Wir können jede Formel (auch als Teilformel einer größeren Formel – ohne Beweis) durch eine logisch äquivalente ersetzen; also gilt z.B.

$$\begin{aligned} A \vee B \wedge \neg C &\quad =|\models\quad \neg A \to B \wedge \neg C \\ &\quad =|\models\quad \neg A \to \neg(B \to \neg\neg C) \quad \text{(Teilformel ersetzt)} \\ &\quad =|\models\quad \neg A \to \neg(B \to C) \end{aligned}$$

Damit können wir uns also bei Bedarf auf $\neg, \to$ und $\forall x$ beschränken und $\wedge, \vee, \leftrightarrow, \exists x$ als Abkürzungen ($A \wedge B$ für $\neg(A \to \neg B)$ etc.) auffassen, die wir bei Bedarf zur besseren Lesbarkeit benützen.

Damit haben wir weniger Formeln und weniger Fallunterscheidungen, z.B. beim Beweis von Satz 2.3 oder im Hilbert-Kalkül in Abschnitt 2.2.

Bem.: Zum Vergleich der verschiedenen Äquivalenzsymbole:

- $\leftrightarrow$ steht in Formeln
- $=|\models$ steht zwischen Formeln; Meta-Sprache
- $\Leftrightarrow$ steht zwischen (meta-sprachlichen) Aussagen, die wahr oder falsch sind; $A \vee B \Leftrightarrow \neg A \to B$ ist syntaktisch falsch, denn $A \vee B$ ist nicht wahr oder falsch – $I, \beta \models A \vee B$ schon.

□

Satz 1.4 *i) Für alle $n \geq 1$ gilt: $I \models \{A_1, \ldots, A_n\} \Leftrightarrow I \models A_1 \wedge \cdots \wedge A_n$*

ii) $M \models A \to B \Leftrightarrow M \cup \{A\} \models B$

Beweis: i) per Induktion über n

ii) "$\Rightarrow$" Gelte $I, \beta \models M \cup \{A\}$ (– z.z. ist $I, \beta \models B$). Dann gilt $I, \beta \models M$ und damit nach Vorraussetzung $I, \beta \models A \to B$, ferner $I, \beta \models A$. Also gilt einerseits $I, \beta \not\models A$ oder $I, \beta \models B$, andererseits $I, \beta \models A$; demnach $I, \beta \models B$.

"$\Leftarrow$" Gelte $I, \beta \models M$ (– z.z. ist $I, \beta \models A \to B$). Entweder $I, \beta \models A$ ist falsch (fertig!) oder $I, \beta \models M \cup \{A\}$ und nach Vorraussetzung $I, \beta \models B$; fertig nach der Definition von $\to$. □

Definition 1.5 $A \in \mathit{For}$ ist *(allgemein)gültig* (eine *Tautologie*), in Zeichen: $\models A$, wenn für alle Interpretationen I gilt $I \models A$. (D.h. für alle I, β: $I, \beta \models A$.) Analog definieren wir $\models M$ für $M \subseteq \mathit{For}$.

A bzw. $M \subseteq \mathit{For}$ sind *erfüllbar*, wenn es ein I und ein β gibt mit $I, \beta \models A$ bzw. $I, \beta \models M$, andernfalls *unerfüllbar*. □

$\models A$ bzw. $M \models A$ sind die Wahrheiten, die uns interessieren. Vgl. oben: $M_{gr} \models A$ heißt, daß A aus den Formeln in M_{gr} folgt – unabhängig von einem konkreten I bzw. I, β, d.h. der konkreten Gruppe; analog für Körper, Vektorräume etc. M könnte auch Gesetze für Datenstrukturen enthalten, z.B. $\forall x \forall s \ \mathit{first}(\mathit{prefix}(x, s)) = x$.

Proposition 1.6 *Sei $A \in For$.*

i) *A gültig $\Rightarrow$ A erfüllbar, d.h.*
A unerfüllbar $\Rightarrow$ A nicht gültig.

ii) *A gültig ($\models A$) $\Leftrightarrow \emptyset \models A$*

Beweis: ii)

$\emptyset \models A \quad \Leftrightarrow$
$\Leftrightarrow$ □
$\Leftrightarrow$ A gültig.

Satz 1.7 *Sei $A \in For$ und $M \subseteq For$. Dann gilt $M \models A$ genau dann, wenn $M \cup \{\neg A\}$ unerfüllbar ist; speziell ist A gültig genau dann, wenn $\neg A$ unerfüllbar ist.*

Beweis: für alle $I, \beta : I, \beta \models M \Rightarrow I, \beta \models A$
genau dann wenn
für alle $I, \beta : I, \beta \models M \Rightarrow I, \beta \not\models \neg A$ (Proposition 1.1 i))
genau dann wenn
$M \cup \{\neg A\}$ ist unerfüllbar. □

Algorithmische Idee: Versuche, erfüllende I, β für $\neg A$ zu konstruieren; gelingt dies, ist A nicht gültig (und wir haben I, β als Beleg dafür), sonst ist es gültig (wenn die Versuche geeignet sind).

2 Aussagenlogik

2.1 Allgemeines, Wahrheitstafeln

In diesem Kapitel sei $\mathcal{P} = \mathcal{P}^0 = \{p_0, p_1, p_2, ...\}$ [1] und "=" ist verboten. Es treten in Formeln also keine Terme auf, demnach auch keine Variablen. $\mathcal{P}$ *ist* demnach die Menge der atomaren Formeln, kurz *Atome*. Wie bei nullstelligen Funktionszeichen schreiben wir bei nullstelligen Prädikatzeichen keine Klammern beim „Aufruf".

Belegungen sind irrelevant, d.h. $I \models A$ genau dann wenn $I, \beta \models A$.
O.E. gibt es keine Variablen, Belegungen, $\forall x$ – und $\mathcal{F} = \emptyset$; es gibt also nur: $\mathcal{P}^0, \rightarrow, \neg$. Interpretation haben demnach die Form $I : \mathcal{P} \rightarrow \{T, F\}$. (Diese werden auch oft Belegungen genannt, wenn man nur Aussagenlogik behandelt – wir werden das also nicht tun.)

Wir können alle Interpretationen (bzw. ihren relevanten Teil) mittels *Wahrheitstafel* durchprobieren; genau dann, wenn alle Einträge einer Formel T sind, ist die Formel gültig. Gültigkeit ist also *entscheidbar*.

Beispiel 2.1

$$\begin{aligned} A &\equiv p \rightarrow (q \rightarrow p) \\ B &\equiv (\neg p \rightarrow \neg q) \rightarrow (q \rightarrow p) \quad \text{(Kontraposition)} \\ C &\equiv (\neg p \rightarrow \neg q) \rightarrow (p \rightarrow q) \end{aligned}$$

p	q	$q \rightarrow p$	A	$\neg p \rightarrow \neg q$	B	$p \rightarrow q$	C
T	T	T	T	T	T	T	T
T	F	T	T	T	T	F	F
F	T	F	T	F	T	T	T
F	F	T	T	T	T	T	T

A und B sind gültig, C nicht. □

Eine Formel ist erfüllbar genau dann, wenn mindestens ein Eintrag in der Wahrheitstafel T ist. Also ist Erfüllbarkeit entscheidbar.

Beispiele: 1. C in Beispiel 2.1 ist erfüllbar.

2. Bei vielen Atomen (n) ist dies Verfahren sehr aufwendig (2^n Zeilen); evtl. "gezielte Suche", z.B.:

Beim Antrag auf das Vordiplom stehen ein weißes, ein rotes und ein blaues Formular zur Wahl. Die Prüfungsordnung enthält folgende Vorschriften:

a) $w \rightarrow \neg b$ (In Worten: Wer ein weißes Formular einreicht, darf kein blaues einreichen.)

b) $w \vee r$

c) $r \rightarrow b \wedge w$

Angenommen $r = T$, dann

Also $r = F$, damit c) ✓

wegen

wegen

Ergibt die einzige erfüllende Interpretation.

In der Aussagenlogik heißt "$\dashv\models$" einfach „haben dieselben Modelle"; auch das können wir mit Wahrheitstafeln durchprobieren.

[1] Menge der nullstelligen Prädikate, s. Kapitel 1

Beispiel 2.2 $p \to (q \to r) \not\models\models (p \wedge q) \to r$

p	q	r	$q \to r$	links	$p \wedge q$	rechts
.	.	.				
T	T	F	F	F	T	F
T	F	T	T	T	F	T
.	.	.				
.	.	.				
F	T	F	F	T	F	T
.	.	.				

Da die Spalten zu links und rechts gleich sind, folgt die Äquivalenz. □

$A \equiv p \to (q \to p)$ aus Beispiel 2.1 ist nur einzelne Formel; "natürlich" gilt auch $\neg p \to (q \wedge r \to \neg p)$. Allgemein:

Satz 2.3 *Sei $B \in For$ gültig mit Atomen $p_0, p_1, ..., p_n$ und seien $A_0, ..., A_n \in For$. B' entstehe aus B, indem man jedes p_i durch A_i ersetzt. Dann ist B' gültig.*

Bem.: Dies gilt analog auch, wenn $A_0, ..., A_n$ Formeln der allgemeinen *Prädikatenlogik* sind; B' heißt dann *aussagenlogisch gültig.* □

Beweis: Werten wir B' unter einer Interpretation I' aus, bestimmen wir, ob $I' \models A_i$ oder nicht.

$I' \models B'$ hängt nicht von A_i, sondern nur von diesen Ergebnissen ab, die mit $\to$ und $\neg$ kombiniert werden. D.h. $I' \models B'$ genau dann, wenn $I \models B$, wobei $p_i^I = T$ genau dann wenn $I' \models A_i$.

Da $I \models B$ immer gilt, ist B' gültig.

Formaler:

Wir betrachten nur die Ersetzung eines Atoms $p \in \mathcal{P}^0$ durch $A \in For$.

Beispiel: Ersetze p durch $p_0 \to p_1$ in $(p \to p_0) \to \neg p$.

Strukturelle Definition von *Ersetzung*: Zu $B \in For$ sei B' definiert wie folgt:

i) $B \equiv q \in \mathcal{P}$:

a) $q \equiv p$: $B' :\equiv$

b) $q \not\equiv p$: $B' :\equiv$

ii) $B \equiv \neg B_1$: $B' :\equiv \neg(B_1')$

iii) $B \equiv B_1 \to B_2$: $B' :\equiv$

Sei I' beliebige Interpretation; wir definieren I, indem wir für alle $q \in \mathcal{P}$ setzen:

$$q^I := \begin{cases} q^{I'} & \text{wenn } q \not\equiv p \\ T & \text{wenn } q \equiv p \text{ und } I' \models A \\ F & \text{sonst} \end{cases}$$

Beh.: Für alle $B \in For$ gilt: $I \models B \Leftrightarrow I' \models B'$

Beweis: durch strukturelle Induktion über B

i) a) $B \equiv p$: $I \models B \Leftrightarrow p^I = T \overset{\text{(Def. } I\text{)}}{\Longleftrightarrow} \ldots \overset{\text{(Def. Ersetzung)}}{\Longleftrightarrow} I' \models B'$

b) $B \equiv q \not\equiv p$: $I \models B \Leftrightarrow q^I = T \Leftrightarrow \ldots \Leftrightarrow I' \models B'$

ii) $B \equiv \neg B_1$: $I \models B \Leftrightarrow I \not\models B_1 \ldots \Leftrightarrow I' \models \neg(B_1') \Leftrightarrow I' \models B'$

iii) analog für $\to$

Jetzt sehen wir: Für alle I' gilt $I' \models B'$ genau dann, wenn für alle I' gilt $I \models B$. Ist also B gültig, so auch B'. □

Beispiel 2.1 (Fortsetzung): Nach Satz 2.3 gilt für alle $A, B \in For$: $A \to (B \to A)$

Analog zu 2.3 folgt aus $B_1 \models\!\models B_2$ auch $B_1' \models\!\models B_2'$. Nach Beispiel 2.2 ist also für alle $A, B, C \in For$: $A \to (B \to C) \models\!\models A \wedge B \to C$; die zweite Formel ist wohl besser zu verstehen.

Exkurs:

Literale : $p, \neg p$ mit $p \in \mathcal{P}^0$

Klausel : $l_1 \vee \cdots \vee l_n$ mit Literalen l_i.

Eine Formel ist in bzw. eine *konjunktive(r) Normalform* (konj. NF), wenn sie eine Konjunktion von Klauseln ist; z.B. $(p_1 \vee \neg p_2 \vee p_3) \wedge (\neg p_1 \vee p_3) \wedge (p_2 \vee \neg p_3)$. (Erfüllbarkeit ist schwer zu entscheiden, s.u.)

Analog: *disjunktive Normalform*: Disjunktion von Konjunktionen von Literalen

Erfüllbarkeit leicht:

(Verwendet zur Beschreibung von Schaltkreisen; es gibt Verfahren zu ihrer Minimierung).

(*) Zu jeder Formel gibt es eine logisch äquivalente disjunktive Normalform.

Wir lesen ihre Konjunktionen aus den Zeilen der Wahrheitstafel mit T ab:

Beispiel: $p \vee q \to \neg q \mathrel{=\!|\!=}$??

p	q	$p \vee q$	$\neg q$	$\bullet$
T	T	T	F	F
T	F	T	T	T
F	T	T	F	F
F	F	F	T	T

(*) gilt auch für konjunktive Normalformen.

Man erhält eine konjunktive Normalform von A aus der disjunktiven Normalform D von $\neg A$, indem man die De Morganschen Gesetze auf $\neg D$ anwendet.

Problem **SAT** (*satisfiability*): Geg. sei A in konj. NF; ist A erfüllbar ?

Dies ist (vermutlich) nur mit hohem Aufwand entscheidbar! (*NP-vollständig*) – Mit Wahrheitstafel entscheidbar; die hat aber bei n Atomen 2^n Zeilen!

$\Longrightarrow$ logische Formeln können

- sehr kompakt sein
- den Schwierigkeitsgrad von Problemen charakterisieren. (z.B. *NP-vollständig*) $\rightsquigarrow$ Informatik III; Komplexitätstheorie.

2.2 Der Hilbert-Kalkül

Ziel: *Herleitung* aller gültigen Formeln durch Zeichenmanipulation aus *Axiomen* mittels *Schlußregeln*. Trotz Wahrheitstafeln ist dies aus folgenden Gründen interessant:

- Herleitung ist evtl. schneller (vgl. Exkurs und gezielte Suche)
- Wiederverwendbarkeit von alten Ergebnissen bzw. Zwischenergebnissen (Sätze, Lemmata)
- Wahrheitstafeln existieren nicht für Prädikatenlogik.

$M \vdash A$: A kann aus M *hergeleitet* werden. ($\vdash$ = turnstyle, Drehkreuz)

Ziel:

- $M \vdash A \Rightarrow M \models A$ Kalkül *korrekt*, alles Herleitbare ist wahr.
- $M \models A \Rightarrow M \vdash A$ Kalkül *vollständig*, alles Wahre kann hergeleitet werden.

Definition 2.4 Der *Hilbert-Kalkül* besteht aus den Axiomen (Regeln ohne Prämissen)

$Ax1 := \{A \to (B \to A) \mid A, B \in For\}$

$Ax2 := \{(A \to (B \to C)) \to ((A \to B) \to (A \to C)) \mid A, B, C \in For\}$

$Ax3 := \{(\neg A \to \neg B) \to (B \to A) \mid A, B \in For\}$

(endlich viele Axiomenschemata, s. Bem. u.)

und der Schlußregel *Modus Ponens MP*

$$\frac{A,\ A \to B}{B}$$

□

Bedeutung dieser Notation: Wenn wir A und $A \to B$ (*Prämissen* von MP) bewiesen haben, beweist dies B (*Konklusion* von MP).

Bem.: $Ax1$ und $Ax2$ sind besser einsichtig in der Form $A \wedge B \to A$ und $((A \wedge B \to C) \wedge (A \to B) \wedge A) \to C$; s. Satz 2.3. □

Bem.: $Ax1$ ist eine Menge von Axiomen. Man kann z.B. $A \to (B \to A)$ als *ein* Axiom mit "Meta-Variablen" A und B (also kein x) auffassen, die für Formeln stehen, und hat dann endl. viele Axiome. Man erhält jede Formel aus $Ax1$ (die man ja in den Herleitungen verwenden will), indem man jedes A durch dieselbe Formel ersetzt und analog für B etc., d.h. als sogenannte *Instantiierung* des Axioms. (vgl. Satz 2.3) □

Was eine Herleitung zu Def. 2.4 ist, ist eigentlich schon klar; neu: Herleitung aus M.

Definition 2.5 $A \in For$ ist *aus* $M \subseteq For$ *H-herleitbar*, $M \vdash_H A$, (oder *herleitbar aus*, $M \vdash A$, in Abschnitt 2.2 und 2.3), wenn es eine Folge $A_1, \ldots A_n$ in *For* gibt (ein "Beweis") mit:

(i) $A_n \equiv A$

(ii) Für $i = 1, \ldots n$ gilt:

- $A_i \in Ax1 \cup Ax2 \cup Ax3 \cup M$ oder
- (Modus Ponens) Es gibt $j, k < i$ mit $A_j \equiv A_k \to A_i$
  (Beweis enthält A_k und $A_k \to A_i$; nach MP ist auch A_i bewiesen).

Die Folge heißt *Herleitung von A aus M* (vgl. Abschnitt 1.1)
$\vdash A$ steht für $\emptyset \vdash A$: A ist *herleitbar*. □

Beobachtung:

i) Für $1 \leq j < n$ ist auch $A_1, \ldots A_j$ eine Herleitung (von A_j).

ii) Sind $A_1, \ldots A_n$ und $B_1, \ldots B_m$ Herleitungen, so auch $A_1, \ldots A_n, B_1, \ldots B_m$ (von B_m).

Beispiel: Für alle $A, B \in For$ gilt $\neg A \vdash A \to B$. Zum Beweis geben wir eine Herleitung aus $\neg A$ an – wieder mit einer Begründung für jede Zeile:

(1) $\neg A \to (\neg B \to \neg A)$	Ax1
(2) $\neg A$	trivial ($\in M$)
(3) $\neg B \to \neg A$	MP (2),(1)
(4) $(\neg B \to \neg A) \to (A \to B)$	Ax3
(5) $A \to B$	MP (3),(4)

Konstruktion (hier und meistens) rückwärts: $A \to B$ ist kein Axiom, also müssen wir es aus einem C und $C \to (A \to B)$ mit MP herleiten. $C \to (A \to B)$ ist in Ax1 für $C \equiv B$, aber B können wir sicher nicht aus $\neg A$ herleiten. $C \to (A \to B)$ ist in Ax3 für $C \equiv \neg B \to \neg A$; also suchen wir jetzt analog ein C für $\neg B \to \neg A$. $\neg A \to (\neg B \to \neg A)$ ist in Ax1 und diesmal liegt $\neg A$ einfach in M.

Beispiel 2.6 Für alle $A \in For$: $\vdash A \to A$.
Beweis:

(1) $(A \to (\underbrace{(A \to A)}_{B} \to \underbrace{A}_{C})) \to ((A \to (\underbrace{A \to A)}_{B}) \to (A \to \underbrace{A}_{C}))$	Ax2
(2) $A \to (\underbrace{(A \to A)}_{B} \to A)$	Ax1
(3) $(A \to (A \to A)) \to (A \to A)$	MP (2),(1)
(4) $A \to (A \to A)$	Ax1
(5) $A \to A$	MP (4),(3)

□

- $A \to A$ ist klar (semantisch, d.h. im Sinne von $\models$), seine Herleitbarkeit aber nicht. Man könnte mehr Axiome nehmen, aber Ziel ist Minimalität: Möglichst wenige Fälle („Eleganz“) und starkes Vollständigkeitsresultat.

Für praktische Herleitungen scheint der Kalkül also weniger geeignet; tatsächlich ist die Situation aber nicht so unbequem wie man denken könnte, denn jetzt können wir einfach $A \to A$ in jeder Herleitung direkt (als Lemma) verwenden. Streng genommen müßten wir jeweils obige Herleitung einbauen, aber das können wir ja immer, so daß wir einfach darauf verzichten wollen.

- Wie kommt man auf den Beweis?
  $A \to A$ ist kein Axiom, z.z. sind also C und $C \to (A \to A)$.
  $C \equiv A$ unmöglich herleitbar, wie auch $C \equiv \neg A$.
  $C \equiv A \to A$ wollen wir ja gerade zeigen.
  $C \equiv A \to (A \to A)$: Axiom, relativ einfach, hängt mit $A \to A$ zusammen.
  $(A \to (A \to A)) \to (A \to A)$ sieht aus wie rechte Seite von Ax2 $\rightsquigarrow$ (1).
  Beweisen ist nicht einfach, aber endlich eine klare Def. – für Aussagenlogik.

Um das Beweisen zu vereinfachen, benötigen wir Regeln zum "Zusammenstricken" von Beweisen, d.h. zum Verwenden alter Ergebnisse wie $A \to A$. Proposition 2.7 i) ist fast dasselbe wie MP ; wir werden den Unterschied in Zukunft nicht immer machen. Proposition 2.7 ii) ist eine *abgeleitete Regel*.

Proposition 2.7 *i) Wenn $M \vdash A$ und $M \vdash A \to B$, dann $M \vdash B$.*

ii) Wenn $M \vdash \neg A \to \neg B$, dann $M \vdash B \to A$.

$$\frac{\neg A \to \neg B}{B \to A}$$

.

Beweis:

1. Herleitung für A und $A \to B$ hintereinanderschreiben – das ist eine Herleitung! Dann $MP \rightsquigarrow B$.
2. Folgt aus $M \vdash (\neg A \to \neg B) \to (B \to A)$ gemäß Ax3 und i).

□

Haben wir also in einer Herleitung $\neg A \to \neg B$ erreicht ($\neg A \to \neg B$ ist also aus dem jeweiligen M herleitbar), können wir in einer folgenden Zeile unmittelbar $B \to A$ schreiben, da wir nun wissen, daß wir eine Herleitung von $B \to A$ aus M einfügen können. Wir können also im Beispiel vor 2.6 Zeile (4) weglassen und (5) mit 2.7 ii) (3) begründen.

Der folgende Satz ist ein wichtiges Hilfsmittel zur Vereinfachung von Beweisen.

Satz 2.8 *(**Deduktionstheorem**) (vgl. Satz 1.4)*
$M \vdash A \to B \Leftrightarrow M \cup \{A\} \vdash B$

Bem.: „⇐" wird implizit in Situationen wie der folgenden angewandt: Um einen Satz der Gruppentheorie zu beweisen, leiten wir ihn aus der Menge M_{gr} von Voraussetzungen her (s. Bsp. nach 1.3). Hat der Satz die Form $A \to B$, beginnen wir den Beweis of „Gelte also A ..."; dann leiten wir B her, wobei wir A als weitere Voraussetzung hinzunehmen, die wir bei

Bedarf also jederzeit in die Herleitung einflechten können – wie ein Axiom. Wir zeigen also $M_{gr} \cup \{A\} \vdash B$; das bedarf einer Rechtfertigung, die nicht offensichtlich ist. □

Beweis:

$\Rightarrow$	n. Vor.	$M \cup \{A\} \vdash A \to B$	(selbe Herleitung)
	ferner	$M \cup \{A\} \vdash A$	(triviale Herleitung)
	also	M $\cup\{A\} \vdash B$	(2.7)

$\Leftarrow$ (wichtig) Es sei $A_1, ..., A_n$ eine Herleitung von B aus $M \cup \{A\}$, d.h. $A_n \equiv B$. Wir zeigen allgemeiner durch Induktion über i, daß für $i = 1, ..., n$ gilt $M \vdash A \to A_i$. Sei dies also wahr für alle $j < i$. (Noethersche Induktion)

1. $A_i \in M \cup Ax1 \cup Ax2 \cup Ax3$

$M \vdash A_i$	trivial
$M \vdash A_i \to (A \to A_i)$.	Ax1
$M \vdash A \to A_i$	(2.7)

2. $A_i \equiv A$

$M \vdash A \to A_i$	(2.6)

3. Es gibt $k, j < i$ mit $A_j \equiv A_k \to A_i$. Nach Induktion gilt also $M \vdash A \to A_k$ und $M \vdash A \to (A_k \to A_i)$. Damit

$M \vdash (A \to (A_k \to A_i)) \to ((A \to A_k) \to (A \to A_i))$	Ax2
$M \vdash A \to A_i$	(2 x 2.7)

Bem.: Noethersche Induktion, denn Schluß von i auf $i + 1$ funktioniert nicht in 3.; in einer Verankerung würde 1. und 2. stehen, die dann im Ind.schritt wiederholt werden müssten. □

Der Beweis gibt eine Konstruktion, wie aus einer Herleitung von A_n eine von $A \to A_n$ wird; dabei wird jeder Schritt der ersten Herleitung durch einen (abgeleiteten) oder durch drei Schritte ersetzt. Beachte: In den Herleitungen steht nicht „$M \vdash$“.)

Anwendungen:

Satz 2.9 *Für alle $A, B, C \in For$:*

i)	$\vdash (A \to B) \to ((B \to C) \to (A \to C))$	*(Transitivität von $\to$)*
ii)	$\vdash \neg A \to (A \to B)$	*($\neg A \wedge A \to B$)(ex falso quod libet)*
iii)	$\vdash \neg\neg A \to A$	
iv)	$\vdash A \to \neg\neg A$	
v)	$\vdash (\neg A \to A) \to A$	*(ein Widerspruchsbeweis)*

Beweis: Die folgenden Beweise sind keine Herleitungen aus einer festen Menge M; trotzdem sind sie wie Herleitungen aufgebaut, wobei wir zusätzlich in jeder Zeile angeben, aus welcher Menge die jeweilige Formel herleitbar ist.

i) $M = \{A \to B, B \to C, A\}$

	(1) $M \vdash A$	trivial
	(2) $M \vdash A \to B$	trivial
	(3) $M \vdash B$	MP (1),(2)
	(4) $M \vdash B \to C$	trivial
	(5) $M \vdash C$	MP (3),(4)
	(6)	Deduktion (5)
	(7)	
	(8) $\vdash (A \to B) \to ((B \to C) \to (A \to C))$	

ii)	(1) $\vdash \neg A \to (\neg B \to \neg A)$	Ax1
	(2) $\neg A \vdash \neg B \to \neg A$	Deduktion (1)
	(3) $\neg A \vdash A \to B$	2.7 ii) (2) (vgl. obiges Bsp. – jetzt kürzer)
	(4) $\vdash \neg A \to (A \to B)$	Deduktion (3)

iv)	(1) $\vdash$	
	(2) $\vdash A \to \neg\neg A$	2.7 ii) (1)

□

abgeleitete Regeln:

Proposition 2.10 *i)* $\dfrac{A \to B, \quad B \to C}{A \to C}$ *ii)* $\dfrac{\neg\neg A}{A}$

Beweis: i) Wenn $M \vdash A \to B$ und $M \vdash B \to C$ (mit derselben Menge M!), dann: $M \vdash (A \to B) \to ((B \to C) \to (A \to C))$ wg. Satz 2.9 i), also $M \vdash A \to C$ mit 2 x MP (bzw. 2.7 i))

ii) Übung

□

2.3 Korrektheit und Vollständigkeit

Satz 2.11 *(Korrektheitssatz)*
Wenn $M \vdash A$ für $A \in For$ und $M \subseteq For$, dann gilt $M \models A$. Speziell: Wenn $\vdash A$, dann ist A gültig.

Beweis: Sei $A_1, ..., A_n$ eine Herleitung von A aus M, und sei I eine Interpretation mit $I \models M$. Wir zeigen durch Induktion über i: $I \models A_i$ für $i = 1, \ldots, n$.

Dann gilt $I \models A$ wegen

Sei also $I \models A_j$ für alle $j < i$.

i) $A_i \in M$: $I \models A_i$ nach Voraussetzung.

ii) $A_i \in Ax1 \cup Ax2 \cup Ax3$: $I \models A_i$, denn A_i ist gültig nach Satz 2.3 und Beispiel 2.1 bzw. Übung.

iii) $j, k < i$ mit $A_j \equiv A_k \to A_i$. Nach Induktion ist $I \models A_k$ und $I \models A_k \to A_i$, nach Def. von $\models$ also $I \models A_i$.

□

Schritte zur Vollständigkeit:

- Konsistenz (= keine widersprüchlichen Herleitungen)
- Modell-Lemma
- Vollständigkeitssatz

Definition 2.12 $M \subseteq For$ heißt *konsistent*, wenn für kein $A \in For$ zugleich $M \vdash A$ und $M \vdash \neg A$ gilt. □

Lemma 2.13 *i) Ist M inkonsistent, so $M \vdash B$ für alle $B \in For$.*

ii) $M \nvdash A \Rightarrow M \cup \{\neg A\}$ ist konsistent.

Beweis:

i) Sei $M \vdash A$ und $M \vdash \neg A$; wg. Satz 2.9 ii) ist $M \vdash \neg A \to (A \to B)$. Also $M \vdash B$ mit 2x MP.

ii) Sei $M \cup \{\neg A\}$ inkonsistent. Dann gilt $M \cup \{\neg A\} \vdash A$ nach i), d.h. $M \vdash \neg A \to A$ nach Satz 2.8. Also $M \vdash A$ nach Satz 2.9 v) und MP.

□

Mit Lemma 2.13 ii) kann man jede konsistente Menge M erweitern zu einer konsistenten Menge M', so daß für alle $A \in For$ entweder $M' \vdash A$ oder $M' \vdash \neg A$. (Entweder es gilt bereits $M \vdash A$, oder man nimmt $\neg A$ hinzu.)

Dann erhält man ein Modell I von M durch $p^I := T$, falls $M' \vdash p$.

Lemma 2.14 *(Modell-Lemma)*
Jede konsistente Menge ist erfüllbar, d.h. (in der Aussagenlogik): Sie besitzt ein Modell.

Satz 2.15 *(Vollständigkeitssatz)*
Für alle $A \in For$ und $M \subseteq For$ gilt $M \models A \Rightarrow M \vdash A$.

Beweis: Angenommen $M \nvdash A$. Dann ist $M \cup \{\neg A\}$ konsistent (Lemma 2.13 ii)) und hat ein Modell I (Lemma 2.14) mit $I \models M$ und $I \models \neg A$, d.h. $I \not\models A$. Also $M \not\models A$. □

Zusammenfassung 2.16 *i) $M \models A \Leftrightarrow M \vdash A$ (Sätze 2.11 und 2.15)*

ii) Tautologie $\widehat{=}$ herleitbar (Spezialfall von i))

iii) M erfüllbar $\Leftrightarrow$ M konsistent

Beweis: zu iii):

$\Rightarrow$ Wenn $I \models M$ gilt und $M \vdash A$ und $M \vdash \neg A$, dann gilt wg. Satz 2.11 auch $M \models A$ und $M \models \neg A$ und damit $I \models A$ und $I \models \neg A$ – Widerspruch!

$\Leftarrow$ Lemma 2.14

□

Satz 2.17 *(Endlichkeits- bzw. Kompaktheitssatz)*

i) $M \models A \Leftrightarrow$ es gibt eine endliche Teilmenge $M' \subseteq M$ mit $M' \models A$.

ii) M erfüllbar $\Leftrightarrow$ jede endliche Teilmenge von M ist erfüllbar.

Beweis: i) $\Leftarrow$ ist klar. Da eine Herleitung von A aus M nur endlich viele Formeln ($\widehat{=}M'$) von M benutzt, folgt $\Rightarrow$ aus 2.16 i).

ii) Wegen 2.16 iii) genügt es, zu zeigen:
M ist inkonsistent $\Leftrightarrow$ eine endliche Teilmenge $M' \subseteq M$ ist inkonsistent.

$\Rightarrow$ da Herleitungen von $M \vdash A$ und $M \vdash \neg A$ nur endlich viele Formeln ($\widehat{=}M'$) brauchen.

$\Leftarrow$ klar.

□

Es kann also nicht sein,

- daß man unendlich viele Formeln braucht, die in einem Modell gelten müssen, damit A gilt;
- daß jedes endliche $M' \subseteq M$ ein Modell hat, das man für größere M' aber immer wieder ändern muß, so daß letzlich M kein Modell hat.

3 Hilbert-Kalkül für Prädikatenlogik

3.1 Vorbereitung

Jetzt ist wieder β wichtig; z.B. gilt nach Definition: $M \models A \Leftrightarrow$ für alle $I, \beta : (I, \beta \models M \Rightarrow I, \beta \models A)$. In Kapitel 2 galt $M \models A \Leftrightarrow$ für alle I: $(I \models M \Rightarrow I \models A)$; jetzt haben wir:

Proposition 3.1 *$M \models A \Rightarrow$ für alle I: $(I \models M \Rightarrow I \models A)$, aber nicht umgekehrt.*

Beweis: nur "aber":
Sei $M = \{P(x)\}$ und $A \equiv P(y)$. $M \not\models A$; z.B. gilt für $P \mathrel{\widehat{=}}$ "ist prim", $\beta(x) = 2$ und $\beta(y) = 4$ sowohl $I, \beta \models M$ als auch $I, \beta \not\models A$.

Andererseits gilt für alle I: $I \models M \Rightarrow I \models A$. Für jedes I mit $I \models P(x)$ gilt nämlich: Sei β eine beliebige Belegung, so gilt u.a. $I, \beta_x^{\beta(y)} \models P(x)$, also $P^I(\beta(y))$ und $I, \beta \models P(y)$. Demnach gilt $I, \beta \models A$ für alle β, d.h. $I \models A$. □

Wichtig ist auch wieder „$\forall x$". Zur Erinnerung:
$I, \beta \models A$ heißt: $A(x)$ gilt für $\beta(x)$. $\forall x A$ heißt: $A(x)$ gilt für alle x-Werte, d.h. für alle $d \in D$ gilt $I, \beta_x^d \models A$. Zum Eingewöhnen:

Proposition 3.2 *i)* $\forall x A \models A$

ii) i.a. nicht $A \models \forall x A$

Beweis:

i) Wenn $I, \beta \models \forall x A$, so speziell $I, \beta_x^{\beta(x)} \models A$ und $\beta_x^{\beta(x)} = \beta$, d.h. $I, \beta \models A$.

ii) $A \equiv P(x)$, $P \mathrel{\widehat{=}}$ "ist prim", $\beta(x) = 2$. Dann gilt in $\mathbb{N}_0$ $I, \beta \models P(x)$, nicht aber $I, \beta \models \forall x P(x)$.

□

Man sagt, dass x in der Formel $P(x)$ (für sich betrachtet) *frei* auftritt, weil über seinen Wert weiter nichts bekannt ist; darüber müssen wir eine Belegungen $\beta(x)$ befragen. Dagegen ist das zweite Auftreten von x in $\forall x P(x)$ durch den Allquantor *gebunden*; es müssen alle möglichen Werte von x durchprobiert werden.

Vergleiche dazu die Blockstruktur in Programmiersprachen:

```
x=5; {int x; ... x=7;... if (x==5)... } if (x==5)...
```

`x == 5` hat innen den Wert `false`, außen `true`.

Gebunden $\mathrel{\widehat{=}}$ (neu) vereinbart; das Auftreten von `x` ist im inneren Block gebunden, Belegung außerhalb ist irrelevant.

Definition 3.3 *FV* bzw. *BV*, die Menge der *freien* bzw. *gebundenen Variablen*, ist wie folgt definiert:

i) Für $t \in Term$ ist $FV(t)$ die Menge der in t auftretenden Variablen, $BV(t) = \emptyset$.

ii) – $FV(t_1 = t_2) := FV(t_1) \cup FV(t_2)$
$BV(t_1 = t_2) := \emptyset$

- $FV(P(t_1,...,t_n) := \bigcup_{i=1}^{n} FV(t_i)$ (P bewirkt keine Bindung.) $BV = \emptyset$
- $FV(\neg A) := FV(A)$ ($\neg$ bewirkt ebenfalls keine Bindung)
  $BV(\neg A) := BV(A)$ (wegen evtl. Quantoren)
- $FV(A \to B) :=$
  $BV(A \to B) :=$
- $FV(\forall x A) := FV(A) \backslash \{x\}$ (vgl. Bew. zu 3.2 ii): $\beta(x)$ ist egal für $\forall x P(x)$)
  $BV(\forall x A) := BV(A) \cup \{x\}$

$A \in For$ heißt *geschlossen*, wenn $FV(A) = \emptyset$. □

Beispiele:

- $A \equiv (\forall x P(x)) \to P(x)$ (verschiedene *Auftreten* von x)
- $A \equiv \forall x P(y)$
  $BV(A) =$
  $FV(A) = \begin{cases} \text{falls } x \not\equiv y \\ \text{falls } x \equiv y \end{cases}$

Bem.: x und y stehen für Elemente von $\mathcal{X}$, könnten also z.B. beide für x_3 stehen; oft unterstellen wir $x \not\equiv y$. □

Definition 3.4 $\forall x A$ heißt *Generalisierung* von A. Ist $FV(A) = \{y_1, \ldots, y_n\}$, so heißt jede der $n!$ Formeln $\forall y_1 \forall y_2 \ldots \forall y_n \;\; A$ ein *universeller Abschluß* von A. □

Beispiel: Kommutativgesetz: Wir schreiben oft $x+y = y+x$ und meinen $\forall x \forall y \;\; x+y = y+x$.
(Dabei unterstellen wir $x \not\equiv y$; dazu sollten wir eigentlich besser $\forall x_0 \forall x_1 \;\; x_0 + x_1 = x_1 + x_0$ schreiben, denn wir meinen natürlich nicht $\forall x_0 \forall x_0 \;\; x_0 + x_0 = x_0 + x_0$.)

Satz 3.5 *(Koinzidenzlemma)*
Seien $A \in For$ und β_1, β_2 Belegungen mit $\beta_1|_{FV(A)} = \beta_2|_{FV(A)}$. ($\beta_1$ und β_2 stimmen auf den freien Variablen von A überein.) Dann $I, \beta_1 \models A \Leftrightarrow I, \beta_2 \models A$.

Beweis: strukturelle Induktion □

Korollar 3.6 *Seien A, M geschlossen und β_1, β_2 Belegungen.*

i) $I, \beta_1 \models M \Leftrightarrow I, \beta_2 \models M$

ii) $I, \beta_1 \models M \Leftrightarrow I \models M$ *(mit i))*

iii) M *erfüllbar* $\Leftrightarrow M$ *hat ein Modell (mit ii))*

iv) $M \models A \Leftrightarrow$ *für alle I: ($I \models M \Rightarrow I \models A$)*

Beweis: zu iv):

" $\Rightarrow$ "

" $\Leftarrow$ "

□

Bem.: zu iv):
Beachte Unterschied zu 3.1.
Vgl. M_{gr} im Beispiel nach 1.3; dort sind Belegungen also in der Tat irrelevant. □

Proposition 3.7 *i)* $I \models A \Leftrightarrow I \models \forall x A$

ii) $\models A \Leftrightarrow \models \forall x A$

Bem.: Beachte den Kontrast zu 3.2; man muß mit offenen Formeln also vorsichtig sein. □

Beweis: Idee: A bzw. $\forall x A$ gilt jeweils für alle β, also für alle Werte $\beta(x)$; daher ist $\forall x$ egal.

i) für alle $\beta : I, \beta \models A$

$\Leftrightarrow_*$ für alle β und alle $d \in D : I, \beta^d_A \models A$

$\Leftrightarrow_{\text{Def.}}$ für alle $\beta : I, \beta \models \forall x A$

"$\Rightarrow_*$" β^d_x ist eine spezielle Belegung, die also auch A erfüllt.

"$\Leftarrow_*$" wähle $d = \beta(x)$; dann $\beta^d_x = \beta$.

ii) i) gilt für alle I.

□

Aus $\forall x P(x)$ wollen wir $P(c)$ oder $P(f(x, y))$ herleiten (z.B. wenn jede natürliche Zahl einen Nachfolger hat, so auch $3 + 2$); dazu ersetzen wir in $\underline{P(x)}$ x durch beliebige Terme, z.B. auch durch y. Allgemein: in $\forall x A$ redet A über x, wir wollen x in A substituieren. Substitution evtl. problematisch, wenn Formel A selbst Quantoren enthält:

- Enthält A $\forall x Q(x)$, wollen wir x dort nicht substituieren; $\forall x Q(x)$ redet (im Gegensatz zu $Q(x)$) gar nicht über x; genausogut könnten wir $\forall y Q(y)$ schreiben – sog. *α-Konversion.*

  Also: nur freie x ersetzen. Analog zum Umbenennen von Variablen in Programmen: Enthält das Programm einen Block {int x; ...}, so können wir alle x in diesem Block durch y ersetzen, ohne das Verhalten des Programms zu ändern.

- $\forall y P(x, y)$ sagt etwas über x (z.B. "x ist kleinstes Element"). Ersetzt man darin x durch y, so entsteht die Formel $\forall y P(y, y)$; sie sagt nichts über y, sondern daß P reflexiv ist; beim Ersetzen des freien x durch y wurde dies "gefangen", als es in den *Bindungsbereich* von $\forall y$ geriet (vgl. Programmiersprachen).

  Das müssen wir vermeiden, in dem wir zuvor $\forall y P(x, y)$ zu $\forall z P(x, z)$ umschreiben.

Definition 3.8 Wir definieren die *Substitution* $[t/x]$ von x durch $t \in Term$ induktiv (vgl. Beweis zu 2.3):

T1) $x[t/x] : \equiv$

$y[t/x] : \equiv$ für $y \not\equiv x$

T2) $f(t_1, \ldots, t_n)\ [t/x] :\equiv f(t_1[t/x], \ldots, t_n[t/x])$

speziell: $c[t/x] :\equiv c \quad c \in \mathcal{F}^0$

A1, A2) $P(t_1, \ldots, t_n)\ [t/x] :\equiv$

„$=$“ analog

A3) $(\neg A)\ [t/x] :\equiv \neg A[t/x]$ (steht für $\neg(A[t/x])$, d.h. $[t/x]$ bindet stärker als $\neg$)

A4) $(A \to B)[t/x] :\equiv$

A5) $(\forall y A)[t/x] :\equiv \begin{cases} \forall y\ A & \text{für } x \notin FV(\forall y A) \\ \forall y\ A[t/x] & \text{sonst, und } y \notin FV(t) \\ \forall z\ A[z/y][t/x] & \text{sonst, wobei } z \text{ eine } \textit{frische} \text{ Variable ist,} \\ & \text{d.h. } z \notin FV(t) \cup FV(A) \end{cases}$

□

Beispiel:

$$\Big(\big(\forall z\ P(x,z) \to \forall x\ Q(x,z)\big) \to \forall y\ R(y,x)\Big)\ [f(y,x)/x]$$
$$\equiv \Big(\forall z\ \big(P(x,z) \to \forall x\ Q(x,z)\big)\Big)[f(y,x)/x] \to \big(\forall y\ R(y,x)\big)[f(y,x)/x]$$
$$\equiv \Big(\forall z\ \big(P(x,z) \to \forall x\ Q(x,z)\big)[f(y,x)/x]\Big) \to \forall z\ R(z,x)[f(y,x)/x]$$
$$\equiv \Big(\forall z\ P(x,z)[f(y,x)/x] \to \big(\forall x\ Q(x,z)\big)[f(y,x)/x]\Big) \to \forall z\ R(z,x)[f(y,x)/x]$$
$$\equiv \big(\forall z\ P(f(y,x),z) \to \forall x\ Q(x,z)\big) \to \forall z\ R(z,f(y,x))$$

3.2 Der Kalkül

Definition 3.9 Der *Hilbert-Kalkül* hat als *Axiome* für alle $A, B, C \in For$ und $t \in Term$ alle *Generalisierungen* von:

Ax1: $A \to (B \to A)$ (wie in Kapitel 2)

Ax2: $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ (wie in Kapitel 2)

Ax3: $(\neg B \to \neg A) \to (A \to B)$ (wie in Kapitel 2)

Ax4: $(\forall x\ A) \to (A[t/x])$ SP (Spezialisierung)
(z.B. Wenn jede natürliche Zahl einen Nachfolger hat, dann auch $3+2$; s.o.)

Ax5: $A \to \forall x\ A$, falls $x \notin FV(A)$ GE (Generalisierung)

Ax6: $(\forall x\ A \to B) \to \big((\forall x\ A) \to (\forall x\ B)\big)$ DA (Distributivität Allquantor)

Ax7: $x = x$ RE (Reflexivität)

Ax8: $x = y \to (A \to A')$, wobei A quantorenfrei ist und A' aus A entsteht, indem einige (oder kein) x durch y ersetzt werden. GL (Gleichheit)

und als *Schlußregel* nur Modus Ponens.

Die Definition einer Herleitung und von $M \vdash A$ wird analog aus Definition 2.5 übernommen. Für eine aussagenlogische Formel B heißt $\vdash_H B$, daß man B gemäß Definition 2.5 (mit dem "Aussagen-Hilbert-Kalkül") herleiten kann. □

Satz 3.10 *Sei B eine aussagenlogische Formel mit Atomen $p_0, \ldots, p_n$ und seien $A_0, \ldots, A_n \in For$. B' entsteht aus B wie in Satz 2.3. Gilt $\vdash_H B$ (also $\models B$), so auch $\vdash B'$ und $\models B'$. (syntaktisches Gegenstück zu Satz 2.3)*

Beweis: $\vdash_H B \Leftrightarrow \models B$ wegen 2.16; $\models B'$ wegen Satz 2.3.

Man kann in einer Herleitung $B_1, \ldots, B_m$ von B gemäß $\vdash_H B$ jedes B_i durch B_i' ersetzen; denn:

- Ist B_i Axiom in Definition 2.5, so B_i' Axiom in Definition 3.9.
- Wird B_j aus B_i und $B_i \to B_j$ hergeleitet, so auch B_j' aus B_i' und $(B_i \to B_j)' \equiv B_i' \to B_j'$.

□

Damit sind $A \to A$ und Satz 2.9 i) – v) auch im Prädikaten-Kalkül herleitbar. Die Propositionen 2.7 und 2.10 lassen sich hier auch genau wie in Kapitel 2 beweisen.

Beispiel: $\vdash A[t/x] \to \exists x A$ (Gilt A für t, so gibt es einen x-Wert, der A erfüllt.)

Bemerkung: $\exists x\ A \equiv \neg\ \forall x\ \neg\ A$; wir haben folgende Herleitung:

(1) $(\forall x \neg A) \to (\neg A)[t/x]$ Ax4 SP (beachte: $(\neg A)[t/x] \equiv \neg A[t/x]$)

(2) $\big((\underbrace{\forall x \neg A}_{C}) \to (\neg \underbrace{A[t/x]}_{D})\big) \to (\underbrace{A[t/x]}_{D} \to \neg \underbrace{\forall x \neg A}_{C})$ (ist ein B' mit $\models B$, Satz 3.10)

(3) $A[t/x] \rightarrow \neg\forall x \neg A$ MP (1),(2)

Wir haben hier benutzt, daß (2) aussagenlogisch gültig ist; wir haben uns also auf eine semantische Argumentation gestützt und damit die Herleitung deutlich verkürzt. Das ist gerechtfertigt, da wir vor allem Aussagen herleiten wollen, die man nicht so einfach überblicken kann wie (2).

Beispiel: $\vdash (\forall x \forall y \ \ P(x,y)) \rightarrow \forall x \ \ P(x,x)$

Im ersten Schritt der Herleitung verwenden wir eine Generalisierung von Ax4!

(1) $\forall x \ \ (\forall y \ \ P(x,y)) \rightarrow P(x,x)$	Ax4 SP
(2) $\big(\forall x \ \ (\forall y \ \ P(x,y)) \rightarrow P(x,x)\big) \rightarrow \big((\forall x \forall y \ \ P(x,y)) \rightarrow \forall x \ \ P(x,x)\big)$	Ax6 DA
(3) $(\forall x \forall y \ \ P(x,y)) \rightarrow \forall x \ \ P(x,x)$	MP (1), (2)

Satz 3.11 *(Deduktionstheorem)*
$M \vdash A \rightarrow B \ \Leftrightarrow M \cup \{A\} \vdash B$

Beweis: praktisch derselbe wie zu Satz 2.8

□

Satz 3.12 *(Generalisierungstheorem)*
Aus $M \vdash A$ und $x \notin FV(B)$ für alle $B \in M$ folgt $M \vdash \forall x A$.

Beweis: Sei A_1,...,A_n Herleitung von A aus M. Wir zeigen, daß man $\forall x A_1, \ldots, \forall x A_n$ zu einer Herleitung "auffüllen" kann. (Konstruktion einer Herleitung!)

i) Ist A_i Axiom, so auch $\forall x A_i$ ("Generalisierung" in Definition 3.9)

ii) Ist $A_i \in M$, fügen wie ein:

(1) A_i	Vor.
(2) $A_i \rightarrow \forall x A_i$	Ax5 GE
(3) $\forall x \ A_i$	MP (1),(2)

iii) Wird A_i aus A_j und $A_j \rightarrow A_i$ hergeleitet, leiten wir aus $\forall x \ (A_j \rightarrow A_i)$ und Ax6 mit MP $(\forall x A_j) \rightarrow \forall x A_i$ her, daraus mit $\forall x \ A_j$ und MP $\forall x \ A_i$.

□

Korollar 3.13 $\vdash A \ \Rightarrow \vdash \forall x A$ *(vgl. Prop. 3.7 und 3.2)*

Proposition 3.14 *Sei $y \notin FV(\forall x A)$. Dann gilt $\vdash (\forall x A) \rightarrow \forall y (A[y/x])$ (Umbenennen von Hilfsvariablen (dummy variables) – α-Konversion)*

Beweis: $\vdash (\forall x A) \to (A[y/x])$ wegen Ax4, also $\forall x A \vdash A[y/x]$ nach Satz 3.11. Nach Satz 3.12 gilt $\forall x A \vdash \forall y A[y/x]$ und fertig mit Satz 3.11. □

Beispiel: $\vdash (\forall x \;\; P(x) \to Q(f(x), z)) \to \forall y \;\; P(y) \to Q(f(y), z)$

Wenn man t mit $x \in FV(t)$ für z in $\forall x A$ substituieren will, nimmt man oft an, daß die Einsetzung eines frischen y "automatisch" geschieht, d.h. o.E. geschehen ist; also: Wenn wir "$A[t/z]$" schreiben und $x \in FV(t)$, so gilt immer $x \notin BV(A)$. Der schwierige 3. Fall von Definition 3.8 A5 tritt unter dieser Annahme (*Barendregt-Konvention*) nicht mehr auf; daß wir bei diesem 3. Fall ein beliebiges frisches z zugelassen und uns daher nicht auf genau einen Term festgelegt haben, ist jetzt durch 3.14 gerechtfertigt.

3.3 Korrektheit und Vollständigkeit

Satz 3.15 *(Korrektheit)*
$M \vdash A \Rightarrow M \models A$ *(speziell:* $\vdash A \Rightarrow \models A$*)*

Beweis: wie zu Satz 2.11. Wir müssen "nur noch" die Gültigkeit von Ax1 bis Ax8 beweisen. Wegen Proposition 3.7 ii) können wir das Wort "Generalisierung" in Definition 3.9 ignorieren;

damit sind Ax1 – Ax3 nach Satz 2.3 erledigt. Ax7 und Ax8 sind nicht schwer.

Ax4 SP Ansatz: Induktion über die Länge der Herleitung von A; langwierig, besonders für den Fall $A \equiv \forall y B$

Ax5 GE Es genügt zu zeigen: $A \models \forall x A$ (wegen Satz 1.4 ii)). Sei also $I, \beta \models A$ und $d \in D$ beliebig. Da $x \notin FV(A)$, sind β_x^d und β auf $FV(A)$ gleich. Demnach $I, \beta_x^d \models A$ nach Satz 3.5 und daher $I, \beta \models \forall x A$.

Ax6 DA einfacher (mit 2x 1.4 ii))

□

Zum Beweis der Vollständigkeit verwendet man wie in Kapitel 2 die Konsistenz (vgl. Definition 2.12) und zeigt analog zu Lemma 2.13:

Lemma 3.16 *i)* $M \nvdash A \Rightarrow M \cup \{\neg A\}$ *ist konsistent.*

ii) $M \nvdash \forall x A \Rightarrow M \cup \{\neg \forall x A, \neg A[^c/_x]\}$ *ist konsistent für jede Konstante c, die nicht in M und A vorkommt.*

Die Vereinigung in ii) enthält nicht nur $\neg \forall x A$, sondern auch einen "Zeugen" dafür.

Damit kann man zum Beweis des Modell-Lemmas wieder jede konsistente Menge in gewissem Sinne vervollständigen; aus der resultierenden Formelmenge liest man für Logik *ohne Gleichheit* erfüllende I_0 und β_0 mit $D_0 \subseteq Term(!)$ ab. Sind $P(c)$ und $P(f(c))$ in der Menge, nehmen wir c und $f(c)$ selbst als Elemente von D_0 und lassen P für diese Elemente wahr sein; s.a.u.

Für Logik mit Gleichheit ist D eine Menge von Äquivalenzklassen von D_0. (Terme, die gleich sein müssen – z.B. $c = c \circ e$ ist in M – sind äquivalent; als Terme, also als Elemente von D_0, sind c und $c \circ e$ verschieden – $[c]$ und $[c \circ e]$ sind aber dieselbe Äquivalenzklasse, also dasselbe Element von D.) Da *Term* abzählbar ist, ergibt sich:

Lemma 3.17 *(Modell-Lemma)*
konsistent $\Leftrightarrow$ *erfüllbar* *(vgl. Lemma 2.14 und 2.16)*

Satz 3.18 *(Löwenheim-Skolem)*
Jede erfüllbare Menge M geschlossener Formeln hat ein höchstens abzählbares Modell bzw. im Falle von Logik ohne Gleichheit ein abzählbar unendliches Modell.

Anwendung: Daher kann kein M $\mathbb{R}$ charakterisieren; neben $\mathbb{R}$ hätte M auch ein abzählbares, also anderes Modell.

Der Beweis folgt aus obigen Überlegungen; beachte: Aus „geschlossen “ folgt $I, \beta \models M$ (M erfüllbar) $\Leftrightarrow I \models M$ (M hat ein Modell). Der Satz besagt natürlich nur, daß es jeweils zumindest ein solches Modell gibt – evtl. gibt es natürlich auch andere.

Jetzt folgt Vollständigkeit wie in Kapitel 2 ($I \rightsquigarrow I, \beta$):

Satz 3.19 *(Vollständigkeit)*
$M \models A \Rightarrow M \vdash A$

Damit gilt auch Satz 2.17:

Satz 3.20 *(Endlichkeits- bzw. Kompaktheitssatz der Prädikatenlogik)*

i) $M \models A \Leftrightarrow$ *es gibt eine endliche Teilmenge* $M' \subseteq M$ *mit* $M' \models A$.

ii) M *erfüllbar* $\Leftrightarrow$ *jede endliche Teilmenge von* M *ist erfüllbar.*

Anwendungen

Anwendung 3.21 *Die Menge der gültigen Formeln ist aufzählbar bzw. semi-entscheidbar. (Verfahren (z.B. eine Turingmaschine) sagt "ja" (akzeptiert), wenn Formel A gültig; sonst: "nein" oder keine Terminierung)*

Beweis: Mit Sorgfalt kann man alle Herleitungen nach und nach generieren, ggf. bis A hergeleitet wurde; 3.15 & 3.19. □

Bem.:

- Analog für $M \models A$ statt $\models A$, wenn M aufzählbar.
- $\models A$ ist aber unentscheidbar !!!

□

Anwendung 3.22 *Jeder Satz der Gruppentheorie ($M_{gr} \models A$, vgl. Beispiel nach Definition 1.3) hat einen gruppentheoretischen Beweis, d.h. eine Herleitung aus M_{gr} ($M_{gr} \vdash A$).*

Ein Satz könnte ja auch in verschiedenen Gruppen aus verschiedenen Gründen gelten; der Beweis gibt *einen* einheitlichen Grund. Analog für andere in Prädikatenlogik formulierte Theorien.

Logik ist wichtig, um Sachverhalte exakt auszudrücken, d.h. formal zu definieren; dies hat jedoch auch seine Grenzen! Dies zeigen wir jetzt.

Eine Klasse $\mathcal{C}$ von Interpretationen bzw. Modellen (d.h. eine Eigenschaft) ist *definierbar*, wenn es $M \subseteq For$ gibt mit $I \models M \Leftrightarrow I \in \mathcal{C}$, *elementar definierbar*, wenn $|M| = 1$. Z.B. ist für eine geeignete Signatur $(\mathcal{F}, \mathcal{P})$ die Eigenschaft „I ist eine Gruppe“ definierbar – durch M_{gr}.

Anwendung 3.23 *Es gibt kein $A \in For$ mit:*

i) $I \models A \Leftrightarrow D$ *ist endlich*

(Endlichkeit ist nicht elementar definierbar, wohl aber $|D| = 3$, siehe Übung)

ii) (ohne Gleichheit) $I \models A \Leftrightarrow |D| = n$, *wobei* $n \in \mathbf{N}$ *fest*

(in Logik ohne Gleichheit ist n-Elementigkeit nicht elementar definierbar)

Beweis: Angenommen A existiert.

i) Setze $C_i :\equiv \exists x_1 \ldots \exists x_i \quad ((x_1 \neq x_2 \wedge \cdots \wedge x_1 \neq x_i) \wedge (x_2 \neq x_3 \wedge \cdots \wedge x_2 \neq x_i) \wedge \cdots \wedge (x_{i-1} \neq x_i))$; also: $C_i \Leftrightarrow$ es gibt $\geq i$ Elemente.

$M := \{A\} \cup \{C_i \mid i \geq 2\}$. Jede endliche Menge $M' \subseteq M$ ist erfüllbar – wähle D' mit $max\{i \mid C_i \in M'\}$ Elementen. Also auch M durch ein D nach Satz 3.20; $|D|$ ist endlich nach Annahme und $C_{|D|+1}$ ist verletzt. Widerspruch!!

ii) Wegen Proposition 3.7 i) ist A o.E. geschlossen. Wende Satz 3.18 auf {A} an, das natürlich erfüllbar ist. Widerspruch!!

□

i) und ii) gelten auch für "definierbar", siehe Übung.

Modelle, die auf Termen basieren, interessieren bei Datenbanken (logische Programmierung, Prolog); speziell: Ein *Herbrand-Modell* I von $M \subseteq For$ erfüllt:

i) D ist die Menge der Terme, die aus Konstanten in M (falls nicht existent, Konstante c hinzunehmen) und Funktionssymbolen in M gebildet werden

ii) Für $f \in \mathcal{F}^n$ und $t_1, \ldots, t_n \in D$ gilt $f^I(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$ speziell: $c^I = c$.

Jeder Term ist also seine eigene Interpretation. Prädikate sind nicht festgelegt.

Beispiel: $M = \{\forall x\ P(f(x), c),\ \forall x \forall y\ Q(g(x, c),\ y)\}$. Dann $D \ni c,\ f(c),\ g(c, c),\ f(f(c)),\ g(f(c), c), \ldots$.

g^I bildet $(f(c), g(c, c))$ auf $g(f(c), g(c, c))$ ab.

Satz: Eine erfüllbare Menge universeller, geschlossener Formeln ohne Gleichheit hat ein Herbrand-Modell.

Wir können uns bei der Suche nach erfüllenden Modellen also auf Herbrand-Modelle beschränken.

Dabei ist eine Formel *universell*, wenn sie aus quantorfreien Formeln mittels Konjunktion, Disjunktion und All-Quantifizierung aufgebaut ist.

4 Korrektheit von Programmen – der Zusicherungskalkül

Ziele:
– formale Spezifikation von Programmen (Vor- und Nachbedingungen);
– Nachweis der Korrektheit, d.h. dass Programme ihre Spezifikation erfüllen.
Varianten des Kalküls gibt es allg. für imperative Programme; hier: C-Programmstücke mit Zuweisungen, `if` und `while` .

Man verwendt als Aussagen *Hoare-Tripel* der Form $\{A\}\ S\ \{B\}$, wobei

A, B: prädikatenlogische Formeln – sogenannte *Zusicherungen* –, die u.a. die Konstanten, Funktionen und Prädikate der Grundtypen (int, char, ...) und Programmvariablen verwenden (1.Verwendung von Logik).

A: *Vorbedingung*

B: *Nachbedingung*

S: Anweisung, d.h. ein Programmstück

A und B sind Kommentare, die man ins Programm schreiben kann; in C muß man dann schreiben /*A*/ S /*B*/ .

Hoare-Tripel sind selbst Aussagen, mit denen man folgern und herleiten kann.
(2.Verwendung von Logik)

4.1 Semantik

Hoare-Tripel kann man zwei verschiedene Bedeutungen geben.

schwache Semantik: Gilt A vor Ausführung von S, so gilt B danach, *falls* S ohne Fehlerabbruch terminiert. (Beachte: S ändert die Variablenbelegung β, d.h. sowohl $x = 1$ als auch $x > 1$ können *jeweils zu gegebener Zeit* wahr sein.) Man sagt dann, daß S *partiell korrekt* bzgl. A und B ist.

Beispiele:

i) $\{x \geq 1\}\ x = x - 1;\ \{x \geq 0\}$

ii) $\{true\}\ x = 2;\ \{x > 0\}$

iii) $\{x > 0\}\ x = x/0;\ \{x = 25\}$ endlicher Fehler

iv) $\{true\}$ `while` () $x = x;\ \{x = 42\}$ unendlicher Fehler

strenge Semantik: Gilt A vor S, so terminiert S ohne Fehlerabbruch und dann gilt B. Man sagt dann, daß S *total korrekt* bzgl. A und B ist.

Beispiele: i), ii); nicht: iii), iv)

4.2 Zusicherungskalkül

(Floyd, Hoare (bekannt für Quicksort, TCSP), Dijkstra (Kürzeste-Wege-Algorithmus)):
Regeln, die es gestatten, gültige Aussagen der Form $\{A\}\ \ S\ \ \{B\}$ herzuleiten. Da wir zwei verwandte Semantiken haben, gibt es auch zwei Versionen von Gültigkeit; also haben wir genaugenommen auch zwei (sehr ähnliche) Kalküle.

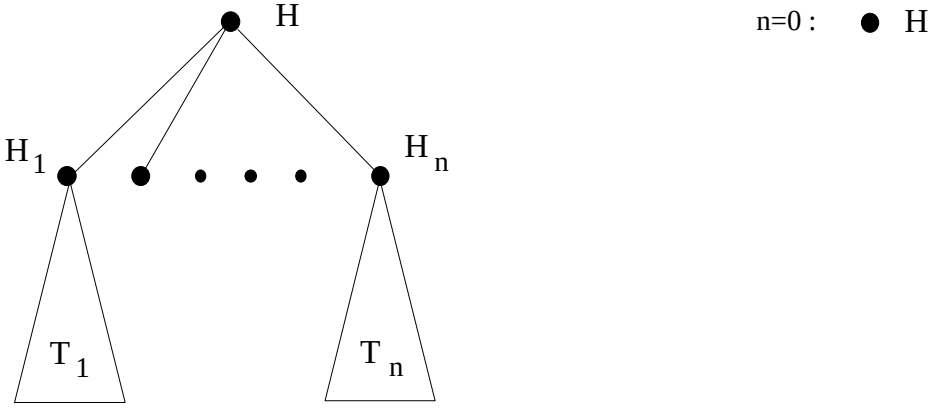
Ein Kalkül soll *korrekt* sein – und möglichst *vollständig*, d.h. die Herleitung aller gültigen Hoare-Tripel erlauben.

Regeln haben wieder die Form $\frac{H_1,\ldots,H_n}{H_{n+1}}$, wobei $n \geq 0$ und die H_i Hoare-Tripel sind. Ein *Axiom* liegt wieder vor bei $n = 0$, also $\frac{}{H_1}$.

Aus Regeln kann man wieder Herleitungen gewinnen; das sind Folgen $K_1,\ldots, K_m$, wobei für alle i eine Regel $\frac{H_1,\ldots,H_n}{H_{n+1}}$ existiert mit $K_i \equiv H_{n+1}$ und $H_1,\ldots, H_n$ erscheinen in $K_1, \ldots, K_{i-1}$.

Alternativ kann man (hier und in der Prädikatenlogik) eine Herleitung auch als Baum auffassen: Eine *Herleitung* (von H) ist (auch) ein Baum, dessen Knoten mit Hoare-Tripeln (und die Wurzel mit H) beschriftet sind, gemäß folgender induktiver Definition:

Sind $T_1, \ldots, T_n$ Herleitungen von $H_1, \ldots, H_n$ und $\frac{H_1,\ldots,H_n}{H}$ eine Regel, so ist T eine Herleitung von H:



Beispiele und Notation, s.u.

Der Kalkül

In einer Zuweisung "$x = E$;" ist E ein Ausdruck. Sei D_E eine Formel, die sicherstellt, daß die Auswertung von E ohne Fehler terminiert. (Definiertheitsbedingung)

Beispiel: Zu $E \equiv x/y$ ist $D_E \equiv y \neq 0$.

4.2.1 Zuweisungsaxiom

$$\frac{}{\{B[^E/_x]\}\quad x = E;\quad \{B\}} \quad (= p)$$

$$\frac{}{\{D_E \wedge B[^E/_x]\}\quad x = E;\quad \{B\}} \quad (= t)$$

(=p) ist das Axiom für partielle, (=t) Axiom für totale Korrektheit. Diese Axiome sind jeweils korrekt, denn:

Gilt B, falls wir x mit dem Wert des Ausdrucks E belegen, so gilt nach der Zuweisung B mit dem aktuellen Wert von x. D_E garantiert zusätzlich Terminierung; ist E stets definiert, so ist $D_E \equiv \mathit{true}$ und kann wegfallen.

Beobachtung: Zu verschiedenen Zeitpunkten gelten verschiedene Formeln – anders als in Kapitel 1-3!! Man sollte Substitution beherrschen!!

Beispiel: 1. a) $\{2 > 0\}\ x = 2;\ \{x > 0\}$ ($D_E \equiv \mathit{true}$)

4.2.2 Konsequenzregel

$$\frac{A \Rightarrow B, \quad \{B\}\ S\ \{C\}, \quad C \Rightarrow D}{\{A\}\ S\ \{D\}} \quad (K)$$

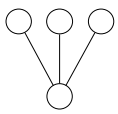
- Beide Varianten unseres Kalküls haben dieselbe Regel – wie auch in den nächsten beiden Fällen.
- Regel korrekt
- $A \Rightarrow B$ heißt: gilt A für unsere Grundtypen, so auch B. Werden die Grundtypen durch Axiome M beschrieben, so heißt $A \Rightarrow B$ also $M \cup \{A\} \models B$. Dies können wir evtl. durch Herleitung $M \cup \{A\} \vdash B$ nach Kapitel 3 zeigen. (3. Verwendung von Logik)
- Ist $A \Rightarrow B$ ein Hoare-Tripel?
- Ist $A \equiv B$ (und analog für $C \equiv D$), so gilt offenbar $A \Rightarrow B$; wir können die Regel anwenden, ohne diese Prämisse aufzuführen. (abgeleitete Regel)

Beispiel: 1. b) $\{\mathit{true}\}\ \ x = 2;\ \ \{x > 0\}$, denn $\mathit{true} \Rightarrow 2 > 0$ und 1.a) (und natürlich $x > 0 \Rightarrow x > 0$)

2. $\{x \geq 1\}\ \ x = x - 1;\ \ \{x \geq 0\}$,

Herleitung (Wurzel unten) zu diesem Beispiel – dabei kann $x \geq 0 \Rightarrow x \geq 0$ entfallen:

$$\frac{x \geq 1 \Rightarrow x - 1 \geq 0, \quad \dfrac{}{\{x - 1 \geq 0\}\ \ x = x - 1;\ \ \{x \geq 0\}}\ (=), \quad x \geq 0 \Rightarrow x \geq 0}{\{x \geq 1\}\ \ x = x - 1;\ \ \{x \geq 0\}}\ (\mathrm{K})$$



Beherrscht man (=), argumentiert man nur: $x \geq 1$ (oder auch $x > 0$ für int x !) impliziert $x - 1 \geq 0$.

Beachte: x tritt in $E \equiv x - 1$ auf!

3. $\{x \geq 1\}\ \ x = x + y;\ \ \{x \geq y\}$, denn $x \geq 1 \Rightarrow x \geq 0 \Rightarrow x + y \geq y$

4. $\{x \geq 0\}\ \ x = -x;\ \ \{x \leq 0\}$, denn $x \geq 0 \Rightarrow -x \leq 0$

Bem.: $B[^E/_x]$ ist die allgemeinste bzw. schwächste Vorbedingung, die die Nachbedingung B garantiert. (weakest precondition, wp-Kalkül, Dijkstra) □

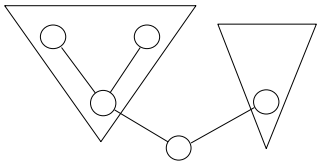
4.2.3 sequentielle Komposition

$$\frac{\{A\}\ \ S\ \ \{B\},\quad \{B\}\ \ T\ \ \{C\}}{\{A\}\ \ S;T\ \ \{C\}}\ (sK)$$

korrekt

Beispiel: Herleitung für int x

$$\dfrac{\dfrac{x>0 \Rightarrow x-1\geq 0,\quad \dfrac{}{\{x-1\geq 0\}\ \ x=x-1;\ \ \{x\geq 0\}}\ (=)}{\{x>0\}\ \ x=x-1;\ \ \{x\geq 0\}}\ (K),\quad \dfrac{\textit{siehe Beispiel 4}}{\{x\geq 0\}\ \ x=-x;\ \ \{x\leq 0\}}}{\{x>0\}\ \ x=x-1;x=-x;\ \ \{x\leq 0\}}\ (sK)$$



2 Bäume werden mit (sK) zusammengefügt.

4.2.4 bedingte Anweisung

$$\frac{\{A\wedge B\}\ \ S\ \ \{C\},\quad \{A\wedge\neg B\}\ \ T\ \ \{C\}}{\{A\}\ \ \texttt{if}\ (B)\ S\ \texttt{else}\ T\ \ \{C\}}\quad (if)$$

korrekt

Beispiel: In diesem Beispiel wird (K) „wirklich“ gebraucht (bei der ersten Anwendung); wir geben zunächst zwei Teilbäume $T1$ und $T2$ an, denen die Wurzel folgt.

$$\frac{true\wedge x>3\Rightarrow -x\leq 0,\quad \dfrac{}{\{-x\leq 0\}\ \ x=-x;\ \ \{x\leq 0\}}\ (=)}{\{true\wedge x>3\}\ \ x=-x;\ \ \{x\leq 0\}}\ (K)$$

$$\frac{true\wedge x\leq 3\Rightarrow x-3\leq 0,\quad \dfrac{}{\{x-3\leq 0\}\ \ x=x-3;\ \ \{x\leq 0\}}\ (=)}{\{true\wedge x\leq 3\}\ \ x=x-3;\ \ \{x\leq 0\}}\ (K)$$

$$\frac{T1\quad T2}{\{true\}\ \ \texttt{if}\ (x>3)\ x=-x;\ \texttt{else}\ x=x-3;\ \ \{x\leq 0\}}\ (if)$$

4.2.5 while-Schleife

Zwei Begriffe, die auch unabhängig vom Kalkül wichtig sind:

- *Schleifeninvariante*: Formel A, die bei jedem Schleifendurchlauf wahr bleibt. Man könnte denken, daß dies $\{A\}\ \ S\ \ \{A\}$ entspricht und $\{A\}\ \ \texttt{while}\ (B)\ S\ \ \{A\}$ impliziert; die nächste Regel ist genauer – und dadurch eher zu verwenden.

- *Terminierungsgröße* t: ein int-wertiger Term in den Programmvariablen, der jeweils kleiner wird, aber nach unten beschränkt ist.

$$\left.\frac{\{A \wedge B\}\ \ S\ \ \{A\}}{\{A\}\ \ \texttt{while}\ (B)\ S\ \ \{A \wedge \neg B\}}\right\}\ \text{(Wp)}$$

$$\left.\begin{array}{ll} (1) & \forall z \in \mathbb{Z}:\ \{A \wedge B \wedge t = z\}\ \ S\ \ \{A \wedge t < z\} \\ (2) & A \wedge B \Rightarrow t \geq 0 \\ \hline & \{A\}\ \ \texttt{while}\ (B)\ S\ \ \{A \wedge \neg B\} \end{array}\right\}\ \text{(Wt)}$$

In (1) werden Schleifeninvariante und Terminierungsgröße gemeinsam behandelt; die Verwendung von z ist ein „Trick“, um den alten und den neuen Wert von t zu vergleichen.

Korrektheit Es gelte A. Wir zeigen zunächst für (Wt), daß `while` $(B)\ S$ terminiert.

Ist B falsch, so Termination; sei also B wahr, so daß wegen (2) $t = z \geq 0$. Wir zeigen durch Induktion über $z \in \mathbb{N}_0$:
Falls $A \wedge B \wedge t = z \in \mathbb{N}_0$, so terminiert `while` $(B)\ S$.

Werde also Termination erreicht, falls $A \wedge B \wedge t = z' < z$, $z' \in \mathbb{N}_0$. Führen wir in der gegebenen Situation S einmal aus, so gelten A und $t < z$ wegen (1). Ist nun B wahr und damit wegen (2) $t = z' \in \mathbb{N}_0$, wenden wir Induktion an; sonst sind wir wie oben fertig.

Wir können nun annehmen, daß A gilt und `while` $(B)\ S$ terminiert, und zeigen für (Wp) und (Wt), daß danach $A \wedge \neg B$ gilt. Beweis durch Induktion über die Anzahl der Schleifendurchläufe; ist die Anzahl 0, so ist B von Anfang an falsch, und wir sind fertig.

Sei die Aussage für $n \geq 0$ Durchläufe wahr, und in der gegebenen Situation werde S $n+1$-mal ausgeführt. Es ist also zunächst B wahr; nach Prämisse gilt dann A nach dem ersten Durchlauf. Jetzt terminiert die Schleife nach n weiteren Durchläufen, nach Induktion fertig. □

4.2.6 Abgeleitete Schlußregel für Schleifen

In Anwendungen stehen Schleifen natürlich nicht isoliert, sondern meist nach einer initalisierenden Anweisung *init*.

Lemma 4.1 *Folgende Schlußregeln sind korrekt:*

$$\left.\begin{array}{ll} \textit{(1)} & \{C\}\ \ \textit{init}\ \ \{A\} \qquad \text{(Invariante muß bei Erreichen der Schleife gelten)} \\ \textit{(2)} & \{A \wedge B\}\ \ S\ \ \{A\} \\ \textit{(3)} & A \wedge \neg B \Rightarrow D \quad \text{(Invariante ist stark genug zum Beweis der Spezifikation } D\text{)} \\ \hline & \{C\}\ \ \textit{init}\ \ \texttt{while}\ (B)\ S\ \ \{D\} \end{array}\right\}\ \text{(Sp)}$$

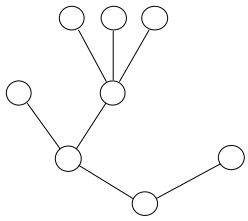
$$\left.\begin{array}{ll} \textit{(1)} & \{C\}\ \ \textit{init}\ \ \{A\} \qquad \text{(Invariante muß bei Erreichen der Schleife gelten)} \\ \textit{(2)} & \forall z \in \mathbb{Z}:\ \{A \wedge B \wedge t = z\}\ \ S\ \ \{A \wedge t < z\} \\ \textit{(3)} & A \wedge B \Rightarrow t \geq 0 \\ \textit{(4)} & A \wedge \neg B \Rightarrow D \quad \text{(Invariante ist stark genug zum Beweis der Spezifikation } D\text{)} \\ \hline & \{C\}\ \ \textit{init}\ \ \texttt{while}\ (B)\ S\ \ \{D\} \end{array}\right\}\ \text{(St)}$$

Bem.: In Anwendungen hat man C und D und muß A finden – Problem!! □

Beweis: für (St):

$$\dfrac{(1)\quad \dfrac{\dfrac{}{(2)\quad (3)}}{\{A\}\ \texttt{while}\ (B)\ S\ \{A \wedge \neg B\}}\ (Wt)}{\{C\}\ \mathit{init}\ \texttt{while}\ (B)\ S\ \{A \wedge \neg B\}}\ (sK) \qquad (4)}{\{C\}\ \mathit{init}\ \texttt{while}\ (B)\ S\ \{D\}}\ (K)$$

Baumstruktur: □



Beispiel: Seien $m_0, n_0 > 0$ und g kurz für $\mathrm{ggt}(m_0, n_0)$.

int m, n
$A \equiv \mathrm{ggt}(m, n) = g$ (impliziert $m, n > 0$) $t \equiv m + n$
$GGT \equiv \mathit{init}\ \mathit{loop}$
$\mathit{init} \equiv m = m_0;\ n = n_0;$
$\mathit{loop} \equiv \texttt{while}\ (m \neq n)\ \mathit{body}$
$\mathit{body} \equiv \texttt{if}\ (m > n)\ m = m - n;\ \texttt{else}\ n = n - m;$

Behauptung: In der starken Semantik gilt $\{\mathit{true}\}\ \ GGT\ \ \{m = g\}$

Beweis: (abgekürzte) Herleitung als Liste. (L) $\widehat{=}$ Logik

(1)	$true \Rightarrow \text{ggt}(m_0, n_0) = g$	(L, vgl. Ax7)
(2)	$\{\text{ggt}(m_0, n_0) = g\}\ \ m = m_0;\ \ \{\text{ggt}(m, n_0) = g\}$	(=t)
(3)	$\{\text{ggt}(m, n_0) = g\}\ \ n = n_0;\ \ \{A\}$	(=t)
(4)	$\{\text{ggt}(m_0, n_0) = g\}\ \ init\ \ \{A\}$	(sK) (2),(3)
(5)	$\{true\}\ \ init\ \ \{A\}$	(K) (1),(4)
(6)	$A \wedge m \neq n \wedge m > n \wedge m + n = z \Rightarrow$ ggt$(m-n, n)$ ist definiert und $=$ ggt$(m, n) = g$ und $m - n + n < z$	(Arithmetisch: $(m-n, n)$ und (m, n) haben dieselben gemeinsamen Teiler und $A \Rightarrow n > 0$)
(7)	$\{\text{ggt}(m-n, n) = g \wedge m - n + n < z\}\ \ m = m - n;$ $\{A \wedge m + n < z\}$	(=t)
(8)	$\{A \wedge m \neq n \wedge m > n \wedge m + n = z\}\ \ m = m - n;$ $\{A \wedge m + n < z\}$	(K)(6),(7)
(9)	$\{A \wedge m \neq n \wedge \neg(m > n) \wedge m + n = z\}\ \ n = n - m;$ $\{A \wedge m + n < z\}$	analog zu (8)
(10)	$\{A \wedge m \neq n \wedge m + n = z\}\ \ body\ \ \{A \wedge m + n < z\}$	(if) (8),(9)
(11)	$A \wedge m \neq n \Rightarrow m, n > 0 \Rightarrow m + n \geq 0$	(Arithmetik)
(12)	$A \wedge m = n \Rightarrow m = \text{ggt}(m, n) = g$	(Arithmetik)
(13)	$\{true\}\ \ GGT\ \ \{m = g\}$	(St) (5),(10),(11),(12)

Graphisch (L: Logik A: Arithmetik) :

$$\dfrac{\dfrac{\dfrac{(L)}{(1)} \quad \dfrac{\overline{(2)}\ (=t) \qquad \overline{(3)}\ (=t)}{(4)}\ (sK)}{(5)}\ (K) \qquad \dfrac{\dfrac{(A)}{(6)} \quad \overline{(7)}\ (=t)}{(8)}\ (K) \qquad \dfrac{analog}{(9)}}{(10)}\ (\texttt{if}\ ) \qquad \dfrac{(A)}{(11)} \quad \dfrac{(A)}{(12)}}{(13)}\ (St)$$

4.3 Verwendungsmöglichkeiten des Zusicherungskalküls

- Verifikation eines gegebenen Programms S mit gegebener Nachbedingung (Spezifikation) B:

  - zeige $\{A\}\ \ S\ \ \{B\}$ für gegebene Vorbedingung A

    oder

  - bestimme A und zeige $\{A\}\ \ S\ \ \{B\}$

  Schlußweise eher "bottom-up", d.h. von einfachen zu zusammengesetzten Programmstücken

- Die oder einige Zusicherungen des Beweises (als Kommentar) ins Programm schreiben $\Rightarrow$ Dokumentation

- Konstruktion von Programmen mit gleichzeitigem Korrektheits- und Terminierungsbeweis: Gegeben B und eventuell A, suche S mit $\{A\}\ \ S\ \ \{B\}$.

Schlußweise "top-down": Aus Anforderungen an einen größeren, noch nicht ausprogrammierten Programmteil werden Anforderungen an kleinere Teile und eine Struktur, gemäß der jene zum größeren Teil zusammengesetzt werden sollen. Vgl. Lemma oben: Sind C und D gegeben und man hat die Idee, daß das Programm die Form $init+$ Schleife haben soll, so ergeben sich Anforderungen an diese Teile.

Bem.:

1. Für ernsthafte Anwendung ist Rechnerunterstützung nötig

   - prüfen der Regelanwendung
   - halbautomatisches Beweisen: einige Beweisschritte automatisch, aber der Mensch gibt z.B. Schleifeninvarianten und Terminierungsgröße an.
   - oder (einfacher): nur einige Zusicherungen als (vermutete) Beweisskizze ins Programm schreiben (Dokumentation!); diese werden zur Laufzeit für den konkreten Fall geprüft; ggf. Stop mit Fehlermeldung. Prüfung nur im debugging-Modus , da diese

2. Beide Varianten sind nur anwendbar, wenn Programm terminieren soll; sonst (z.B. Betriebssystem): temporale Logik für Zusicherungen und Spezifikation.

□

5 Temporale Logik

In Systemabläufen ändern Formeln ihren Wahrheitswert, vgl. Kapitel 4. (Ist $x = 1$, so nicht mehr nach der Zuweisung $x = x + 1$.) Der Zusicherungskalkül ist nur geeignet für terminierende Systeme (z.B. Programme, die *ein* Ergebnis liefern). Viele Systeme terminieren nicht, sondern sollen immer wieder auf Anforderungen reagieren (sogenannte *reaktive* Systeme): Betriebssysteme, Schaltkreise, Kommunikationsprotokolle ... (oft parallel / verteilt)

Beispiel 5.1 Spezifikationen, die über Abläufe reden:

i) Es greifen *nie* zwei Prozesse gleichzeitig auf denselben Datensatz zu. (wechselseitiger Ausschluß, mutual exclusion, MUTEX).

ii) Es erscheint *immer wieder* eine Eingabeaufforderung (prompt).

iii) *Immer, wenn* eine Anforderung (request, r) kommt, wird diese *nach einer Weile* erfüllt (granted, g).

Formulierung in Prädikatenlogik: $\forall t_1 \;\; r(t_1) \rightarrow \exists t_2 > t_1 \;\; g(t_2)$

□

Idee: keine explizite Modellierung der Zeit (Axiome dafür sind kompliziert, Formeln unübersichtlich); besonders günstig, wenn wir sonst nur Aussagenlogik brauchen.

$\Box$: (von jetzt an) immer, always; G

$\Diamond$: irgendwann (ab jetzt), eventually; F

Damit 5.1 iii) formal als $\Box(r \rightarrow \Diamond g)$; hier soll g jeweils irgendwann „ab jetzt“, also nach r gelten.

Eigenschaften i) - iii) aus 5.1 sollen für alle Abläufe gelten, die typischerweise unendliche Folgen von Zuständen sind. (Systeme, die wir betrachten, machen unendlich viele Schritte.) Zeit ist hier also diskret:

$\circ$: im nächsten Zustand, next(time), X

Im folgenden sei wie in Kapitel 2 eine Menge $\mathcal{P} = \mathcal{P}^0$ von *Atomen* (atomare Formeln) gegeben. Systemzustände sind evtl. sehr detailliert; wir können von diesen Details abstrahieren – uns interessiert nur, welche Atome wahr sind. (L in 5.2 leistet gerade diese Abstraktion.) Anhand dieser Information können wir z.B. prüfen, ob jedem Zustand, in dem r wahr ist, ein Zustand folgt, in dem g wahr ist.

Definition 5.2 Ein *Ablauf* $\pi = s_0, s_1, \ldots$ ist eine unendliche Folge von Zuständen s_i aus einer Menge S von *Zuständen* mit einer *Bewertung* $L : \; S \rightarrow \not\!\!\mathcal{P}(\mathcal{P})$.
$\pi^j = s_j, s_{j+1}, \ldots$ ist das j-te *Suffix* von π. □

Die Idee ist, daß p im Zustand s gilt, falls $p \in L(s)$; $L(s)$ ist also im Sinne von Kapitel 2 eine Interpretation (Modell) – *hier* sind Abläufe die Modelle, siehe Definition 5.4. Wir schreiben in Abläufen auch $L(s_i)$ statt s_i; dann ist $\pi = \{p\}, \; \{p, r\}, \; \{p\}, \; \{p, r\} \ldots$ ein Ablauf, den wir formaler schreiben als $\pi = (\{p\}, \; \{p, r\})^\omega$; ähnlich wie $*$ in regulären Ausdrücken für beliebige endliche Wiederholung steht, steht ω für unendliche Wiederholung.

Definition 5.3 (Syntax von PLTL)
Formeln von PLTL (propositional linear time logic): $TFor_{\mathcal{P}}(TFor)$ ist die kleinste Menge mit:

(T1) $p \in \mathcal{P} \Rightarrow p \in TFor$

(T2) $A, B \in TFor \Rightarrow \neg A,\ A \wedge B,\ A \vee B,\ A \rightarrow B,\ A \leftrightarrow B\ \in TFor$
(Klammerung wie üblich; temporale Operatoren (T3, T4) binden am stärksten)

(T3) $A \in TFor \Rightarrow \Box A,\ \Diamond A,\ \circ A\ \in TFor$

(T4) $A, B \in TFor \Rightarrow A\,\mathtt{U}\,B \in TFor$ (until)

□

Definition 5.4 (Semantik von PLTL)
Sei $\pi = s_0, s_1, \ldots$ ein Ablauf. $A \in TFor$ *gilt* für π (π *erfüllt* A, $\pi \models A$) ist wie folgt definiert:

(T1) $\pi \models p$, wenn $p \in L(s_0)$ (p gilt in einem Zustand)

(T2) • $\pi \models \neg A$, wenn nicht $\pi \models A$.

• $\pi \models A \vee B$, wenn $\pi \models A$ oder $\pi \models B$ etc.

(T3) • $\pi \models \circ A$, wenn $\pi^1 \models A$

Mit $\pi = (\{p\},\ \{p, r\},\ \emptyset,\ \{p, r'\})^\omega$ gilt also $\pi \models \circ r$, nicht aber $\pi \models r$ oder $\pi \models \circ\circ r$.

Warum brauchen wir π^1? Können wir nicht einfach sagen, daß A im Zustand s_1 gelten muß?

• $\pi \models \Box A$, wenn

Bei obigem π gilt $\pi \models \Box(r \rightarrow p)$, aber nicht $\pi \models \Box p$

• $\pi \models \Diamond A$, wenn

Oben gilt $\pi \models \Diamond(r' \wedge \circ p)$.

(T4) $\pi \models A\,\mathtt{U}\,B$, wenn es ein $j \geq 0$ gibt mit: $\pi^j \models B$ und für alle $0 \leq i < j:\ \pi^i \models A$.

Oben gilt $\pi \models p\,\mathtt{U}\,r$, aber nicht $\pi \models p\,\mathtt{U}\,r'$

A ist *gültig / erfüllbar*, wenn alle π / ein π A erfüllen.
Wie bisher bedeutet *logisch äquivalent*, $=\!|\!\models$: dieselben π erfüllen links wie rechts; entsprechend bedeutet $M \models A$: "A folgt aus M." □

Beispiel: Sei $\pi = (\{p\},\ \{p,\ r\},\ \{r,\ q\},\ \{r,\ q'\})^\omega$.

$\pi \models p \quad \ldots \qquad \pi \models \Box p \quad \ldots \qquad \pi \models \circ p \quad \ldots \qquad \pi \models r \quad \ldots$
$\pi \models p\,\mathtt{U}\,q \quad \ldots \qquad \pi \models \Box(p\,\mathtt{U}\,q) \quad \ldots \qquad \pi \models p\,\mathtt{U}\,q' \quad \ldots$
$\pi \models \Box(p \rightarrow \Diamond q') \quad \ldots \qquad \pi \models \Box(r \rightarrow \Diamond p) \quad \ldots \qquad \pi \models \Box(p \rightarrow \circ r) \quad \ldots$

Es ist wieder $A \vee B =\!|\!\models \neg A \rightarrow B$ etc., d.h. man kann $\vee$, $\wedge$, $\leftrightarrow$ als Abkürzungen aufffassen.

Proposition 5.5 *i)* $\Box A =\!|\!\models \neg\Diamond\neg A$

ii) $\Diamond A =\!|\!\models \mathit{true}\,\mathtt{U}\,A$

Beweis:

i) $\pi^i \models A$ gilt für alle i gdw. es für kein i falsch ist. $(\forall x A \models\!\!\!\models \neg\exists x \neg A)$

ii) Übung

□

Man kann also bei Bedarf auch $\Box$ und $\Diamond$ als Abkürzungen auffassen. Man kann wieder Axiome, Regeln, Herleitungen und vollständige Kalküle untersuchen. Meist steht aber das sogenannte *model-checking* im Vordergrund (s.u.); wir beschränken uns deshalb auf semantische Überlegungen.

Satz 5.6 *i)* $\models \Box(A \to B) \to (\Box A \to \Box B)$,

ii) $\models \Box \circ A \leftrightarrow \circ \Box A$

iii) $\models (A \wedge \Box(A \to \circ A)) \to$

iv) $\models \circ \Diamond A \to \Diamond A$

Beweis:

i) Bemerkung: $C \to (D \to E) \models\!\!\!\models C \wedge D \to E$

Erfüllt jedes π^i $A \to B$ und A, so auch B; d.h. $\pi \models \Box B$. (vgl. Ax6: $(\forall x A \to B) \to ((\forall x A) \to (\forall x B))$.

ii) Für alle $i \geq 0$ gilt: $\pi^i \models \circ A$ gdw. für alle $i \geq 0$: $(\pi^i)^1 = \pi^{i+1} = (\pi^1)^i \models A$ gdw. $\pi^1 \models \Box A$

iii) $\pi \models A \wedge \Box(A \to \circ A)$ heißt $\pi^0 \models A$ und für alle $i \in \mathbb{N}_0$ $(\pi^i \models A \Rightarrow \pi^{i+1} \models A)$. Nach vollständiger Induktion gilt also $\pi^i \models A$ für alle $i \in \mathbb{N}_0$.

iv) Übung

□

Beispiel: MUTEX: Zwei (oder mehr) Prozesse fordern eine Ressource an (z.B. Zugriff auf Datensatz oder Drucker); wenn sie das tun, gilt (ggf. wiederholt) r_1 bzw. r_2 (vgl. 5.1 iii)). Wenn sie die Ressource bekommen / haben (man sagt: "sie sind in ihrem *kritischen Bereich*"), „erlischt" r_1 bzw. r_2 und es gilt g_1 bzw. g_2, bis sie freigegeben wird.
Anforderungen für Lösung / Scheduler:

α) Nur einer zur Zeit hat die Ressource (vgl. 5.1 i))

$\Box\neg(g_1 \wedge g_2)$

β) Jede Anforderung wird erfüllt (vgl. 5.1 iii))

$\Box(r_1 \to \Diamond g_1)$, $\Box(r_2 \to \Diamond g_2)$

α ist eine typische *Sicherheitseigenschaft* ("Etwas Schlechtes geschieht nie."), wie alle Formeln $\Box$ A mit $A \in For$ (nicht $TFor$). β ist eine *Lebendigkeitseigenschaft* ("Etwas Gutes geschieht schließlich."); weitere Beispiele sind 5.1 ii) ($\Box\Diamond p$) bzw. „Programm hält (h) schließlich endgültig" $\Diamond\Box h$.

Typisch: Ist α bzw. β zu einem Zeitpunkt nicht wahr, so wird α auch in keiner Fortsetzung wahr; β kann immer noch erfüllt werden.

Satz: *Jede Eigenschaft ist Konjunktion einer Sicherheits- und einer Lebendigkeitseigenschaft.*

Einfache Lösung: Ressource immer wieder abwechselnd vergeben; funktioniert aber nur wenn jeder immer wieder anfordert, also $\Box\Diamond r_1 \wedge \Box\Diamond r_2$; das können wir *nicht* annehmen.

Anwendung temporaler Logik oft als **Model-Checking:**

Spezifikation ist eine PLTL-Formel A. Man beschreibt alle System-Abläufe der (evtl. geplanten) Implementierung durch eine Struktur K und prüft, ob *alle* Abläufe von K A erfüllen (d.h. *Modelle* von A sind).

Definition 5.7 Eine *Kripke-Struktur* $K = (S, \rightarrow, L, s_0)$ besteht aus einer Menge S von *Zuständen* mit *Startzustand* s_0, einer Bewertung $L : S \rightarrow \mathfrak{P}(\mathcal{P})$ und einer *Transitionsrelation* $\rightarrow \subseteq S \times S$, so daß für alle $s \in S$ ein s' mit $s \rightarrow s'$ (d.h. $(s, s') \in \rightarrow$) existiert. (vgl. endliche Automaten)

K modelliert z.B. die Abläufe eines interaktiven Programms, wobei ein Zustand durch die Variablenwerte und den Programmzähler (das Restprogramm) gegeben ist.

Ein Ablauf π von K ist eine unendliche Folge von Zuständen beginnend mit s_0 (in der ein s ohne $s \rightarrow s'$ nicht auftreten könnte), also $\pi = s_0, s_1, s_2, \ldots$ mit $s_i \rightarrow s_{i+1}$ für alle $i \geq 0$.

K *erfüllt* ein $A \in TFor$, falls für alle Abläufe π von K gilt $\pi \models A$. □

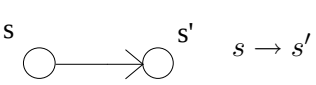
Es gibt viele Ideen (und tools) wie man auch für sehr große (z.T. sogar unendliche) K automatisch prüfen kann, ob K ein A erfüllt.

Als Beispiel für Definition 5.7 diskutieren wir die Modellierung einer MUTEX-Lösung: Eine solche Lösung ist eine Kripke-Struktur, die die Formeln unter α und β (s.o.) erfüllt.

Bem.: Tatsächlich müssen noch weitere Eigenschaften gelten; ist z.B. $\rightarrow = \{(s_0, s_0)\}$ und $L(s_0)$ leer, gelten die gewünschten Formeln – die Prozesse können gar nicht erst anfordern. □

Graphische Konventionen:

Zustand s mit $L(s) = \{p_1, p_2\}$



$s \rightarrow s'$

Startzustand

Idee für K_1 (vgl. Abb. 1): Jeder Prozeß kann anfordern, erhält die Ressource – soweit frei – und gibt sie wieder zurück; falls beide Prozesse anfordern (Zustand s_3) würfelt der Scheduler.

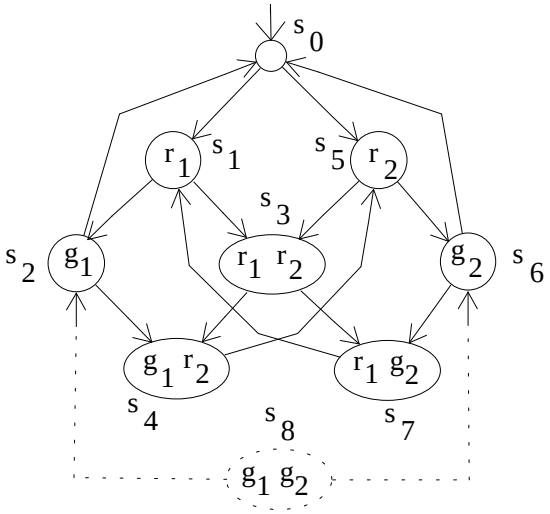


Abbildung 1:

In jedem Ablauf von K_1 wird angefordert ($\Diamond(r_1 \vee r_2)$); das ist in Wirklichkeit nicht so, also ergänzen wir bei s_0 eine Schleife (vgl. Abb. 2). Die Schleife repräsentiert Aktivitäten der Prozesse, die nichts mit der Ressource zu tun haben. (Bzw.: Schleife als technischer Trick, um im Prinzip doch endliche Abläufe einzubeziehen.)

Inspektion von K_1 zeigt, daß K_1 $\Box\neg(g_1 \wedge g_2)$ erfüllt.

Bem.: Wird K_1 aus einem Programm erzeugt, liegt es zunächst nicht explizit vor. In einer ersten Phase wird K_1 also systematisch erzeugt – BFS/DFS auf einem noch nicht existenten Graphen. Diese Erreichbarkeitsanalyse zeigt, daß s_8 von s_0 nicht erreichbar ist, also in Abläufen gar nicht auftreten kann. □

Aber: $\pi = s_0, (s_5, s_3, s_4)^\omega \not\models \Box(r_2 \rightarrow \Diamond g_2)$, denn $\pi^1 = (s_5, s_3, s_4)^\omega \models r_2$, jedoch $\pi^1 \not\models \Diamond g_2$.

Verbesserung (K_2 in Abb. 2): Wer zuerst kommt, mahlt zuerst – aber dann kommt der andere dran.

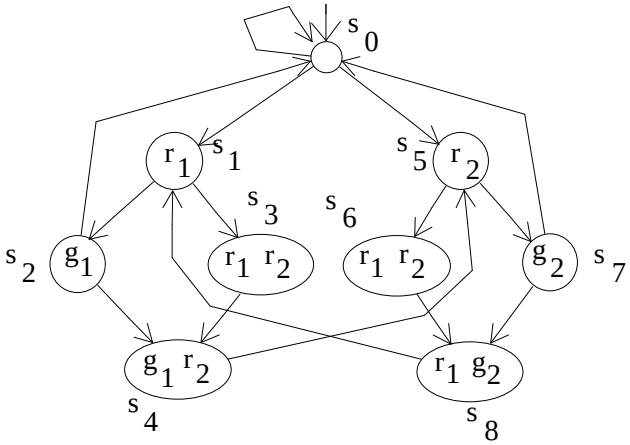


Abbildung 2:

Beachte, daß $L(s_3) = L(s_6)$; L abstrahiert also von (für diese Lösung wesentlichen!) Details, in denen s_3 und s_6 sich unterscheiden. Offenbar erfüllt K_2 die Formel $\Box\neg(g_1 \wedge g_2)$. Wir stellen fest:

- $K_2[s_2]$ (K_2 mit s_2 als Startzustand) und $K_2[s_4]$ erfüllen $\Diamond g_1$.
- Erfüllt $K_2[s']$ $\Diamond g_1$ für alle s' mit $s \to s'$, so auch $K_2[s]$; vgl. Satz 5.6 iv).
- Also erfüllen $K_2[s_3]$, damit $K_2[s_1]$, damit $K_2[s_8]$ und schließlich $K_2[s_6]$ $\Diamond g_1$.
- Erfüllt ein Ablaufsuffix π' r_1, so beginnt π' mit s_1, s_3, s_6 oder s_8. Daher auch $\pi' \models \Diamond g_1$.
- Damit gilt für jeden Ablauf π die Formel $\Box(r_1 \to \Diamond g_1)$.

Also: K_2 erfüllt $\Box(r_1 \to \Diamond g_1)$ und mit Symmetrie auch $\Box(r_2 \to \Diamond g_2)$; K_2 ist also eine Lösung für das MUTEX-Problem.

Typisches Vorgehen: Untersuchung, welche Zustände Teilformeln von A erfüllen.

K_2 is stark vereinfacht: Genauso, wie der erste Prozeß im Zustand s_0 beliebig „für sich“ arbeiten kann, kann er das in s_5. Wir müssen also bei s_5 eine Schleife einfügen – und analog in s_7, s_1 und s_2. Dann ergibt sich aber $s_0, (s_5)^\omega \not\models \Box(r_2 \to \Diamond g_2)$. Ist unsere Lösung falsch?

Argument: Dieser Ablauf ist ein Artefakt; 2 *unabhängige* Prozesse wollen stets einen Schritt machen – in der Realität wird schließlich jeder einen machen, so daß s_5 verlassen wird.

Wir wollen also nur Abläufe betrachten, die die *schwache Fairness* (*progress assumption*) erfüllen: Ist ein Prozeß ständig aktiviert (z.B. *nicht* der erste Prozeß in s_6, wohl aber der Scheduler in s_5), so macht er einen Schritt. Um dies formal zu erreichen, gibt es 2 Möglichkeiten:

1. Struktur mit Fairness-Bedingung $\mathcal{F} \subseteq S$; (Analogon zu Endzuständen in endlichen Automaten, hier für *unendliche* Abläufe; vgl. Büchi-Automaten)

   Eine unendliche Folge wie bisher ist nur dann ein Ablauf, wenn ein $s \in \mathcal{F}$ unendlich oft durchlaufen wird. Wenn ein fairer Ablauf in den Schlingen bei s_1, s_2, s_5 und s_7 nicht hängenbleiben soll, so wählen wir z.B. $\mathcal{F} = \{s_0, s_4, s_8\}$. Jetzt ist s_0, s_5^ω kein Ablauf mehr, wohl aber $s_0, (s_1, s_2, s_4, s_5, s_6, s_8)^\omega$ und s_0^ω.

2. Fairness als Teil der Spezifikation

   Spezifikation besagt: wenn Ablauf fair, dann $\Box(r_1 \to \Diamond g_1)$ etc.
   Dazu müssen wir in der Kripke-Struktur beschreiben, wer aktiviert ist und wer einen Schritt macht bzw. gemacht hat.

   Wir fügen Atome en_1, en_2 (1 bzw. 2 aktiviert/enabled) und $last_1$, $last_2$ (letzter Schritt kam von 1 bzw. 2) hinzu und verlangen $\Box(\Box en_i \to \Diamond last_i)$, $i = 1, 2, \ldots$, d.h. die Lebendigkeitseigenschaften heißen jetzt:

   $$\Box((\Box en_1 \to \Diamond last_1) \wedge (\Box en_2 \to \Diamond last_2)) \to \Box(r_i \to \Diamond g_i), \quad i \in \{1, 2\}$$

   $\Rightarrow$ Aufblähung der Kripke-Struktur, um die Idee von K_2 mit den neuen zusätzlichen Atomen zu modellieren, vgl. Abb. 3.

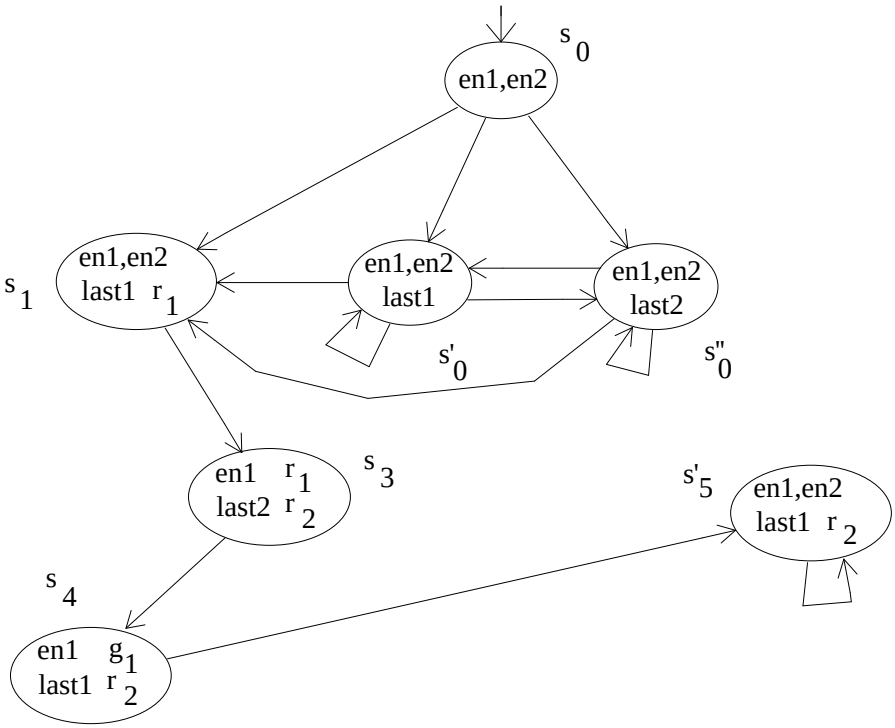

s 0
en1,en2
en1,en2
last1 r 1
s 1
en1,en2
last1
en1,en2
last2
s'0
s"0
en1 r 1
last2 r 2
s 3
s'5
en1,en2
last1 r 2
s 4
en1 g 1
last1 r 2

Abbildung 3: