Grundlagen der Logik in der Informatik

Vorlesungsskript Wintersemester 2020/21
Friedrich-Alexander-Universität Erlangen-Nürnberg
Department Informatik, Lehrstuhl 8 (Theoretische Informatik)

Aufbauend auf einer Mitschrift der Vorlesung von Lutz Schröder im Sommersemester 2012 von

Johannes Schilling
Dominik Paulus
Ulrich Rabenstein
Tobias Polzer

überarbeitet von Lutz Schröder, mit weiteren Beiträgen von

Andreas Schieb

Inhaltsverzeichnis

0.1 Literatur

U. Schöning Logik für Informatiker, Spektrum akademischer Verlag, 2000

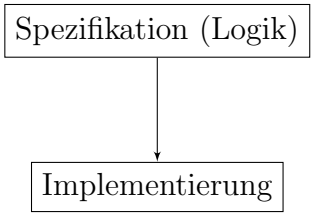
J. Barwise, J. Etchemendy Language, Proof & Logic, CSLI, 2000
(deutsche Version verfügbar)

M. Huth, M. Ryan Logic in Computer Science, CUP, 2000

S. Reeves, M. Clarke Logic for Computer Science, Addison-Wesley, 1990

0.2 Logik in der Informatik

- Logik als Problem für Informatiker
  - SAT
  - Automatisches/Halbautomatisches Theorembeweisen
    (SPASS, Vampire, Isabelle, Coq, Z3)
- Logik als Programmierparadigma
  - Prolog
  - Mercury
- Logik als Abfrageformalismus
  - SQL
  - Datalog
- Logik als (Wissens-) Repräsentationsformalismus
  - Ontologien
  - Semantic Web
  - OWL
- Logik als Entwicklungsmethode



  - Z
  - B
  - VDM
  - CASL
  - Design-by-Contract: Spec#, JML, Dafny, Boogie, FramaC
  - Isabelle/Coq/Z3

1 Aussagenlogik (informelle Einführung)

Redet über atomare Aussagen A, B, C, ... ohne Rücksicht auf deren innere Struktur.

$$(\text{z. B. EsRegnet} \rightarrow \text{HabeSchirm} \vee \text{WerdeNass})$$

und deren Wahrheitswerte, hier klassisch: $\{\underbrace{\bot}_{\text{falsch}}, \underbrace{\top}_{\text{wahr}}\}$

Grundbestandteile einer Logik

- *Syntax*: „Was kann ich hinschreiben?“
- *Semantik*: „Was bedeutet das?“
- *Beweismethoden*: „Wie ziehe ich daraus Schlüsse?“ (manuell /automatisch)

2 Induktion

Induktion ist das Prinzip der Reduktion einer Aussage über ein Objekt auf gleichartige Aussagen über „einfachere“ Objekte in einem jeweils geeigneten Sinn. Wenn Objekte nur endlich oft „einfacher werden“ können, erreicht man so nach endlich vielen Reduktionen ein Objekt, das nicht mehr einfacher werden kann; wenn man außerdem für solche Objekte die entsprechende Aussage beweisen kann („Induktionsanfang“), hat man die Aussage für alle Objekte bewiesen. Das Prinzip ist das gleiche wie bei der Programmierung rekursiver Funktionen – dort ruft man (jedenfalls dann, wenn man an Terminierung interessiert ist) eine Funktion rekursiv mit Argumenten auf, die „einfacher“ als das ursprüngliche Argument sind, und hat Basisfälle, in denen keine rekursiven Aufrufe stattfinden.

Wir fassen kurz die wesentlichen im weiteren benötigten Induktionsprinzipien zusammen. Wir verweisen auch auf den auf der Veranstaltungshomepage verfügbaren Text von Thomas Voß.

2.1 Induktion über natürliche Zahlen

Das vermutlich bekannteste Induktionsprinzip ist die Induktion über natürliche Zahlen (die, mit gewisser Variation, immer in irgendeiner Form entweder als Axiom postuliert oder konstruktiv sichergestellt wird). In der einfachsten Form lautet das Prinzip wie folgt. Wenn eine Aussage $P(n)$ über natürliche Zahlen n (die *Induktionsbehauptung*) die beiden Eigenschaften

1. *Induktionsanfang*: P gilt für die 0 (als Formel: $P(0)$) und
2. *Induktionsschritt*: für jede natürliche Zahl n folgt $P(n+1)$ aus $P(n)$, als Formel:

   $$\forall n \in \mathbb{N}\,(\,P(n) \implies P(n+1)\,)$$

   (man bezeichnet hier $P(n)$ als die *Induktionsvoraussetzung*)

erfüllt, dann gilt P für jede natürliche Zahl, als Formel:

$$\forall n \in \mathbb{N}\,(\,P(n)\,).$$

(Wir verwenden hier später formal eingeführte Notation vorwegnehmend, insbesondere Allquantifizierung $\forall x\,(\phi)$, zu lesen „für alle x gilt ϕ“. Der Geltungsbereich des Quantors ist immer der folgende geklammerte Ausdruck.)

Als Beispiel diene hier folgende einfache Identität:

$$\sum_{i=1}^{n}(2i-1) = n^2.$$

Zum besseren Abgleich mit dem allgemeinen Induktionsprinzip bezeichnen wir diese Aussage mit $P(n)$. Man beweist $\forall n(P(n))$ durch Induktion über n:

- Induktionsanfang: Es gilt $\sum_{i=1}^{0}(2i-1) = 0 = 0^2$ (also $P(0)$).
- Induktionsschritt: Sei $n \in \mathbb{N}$, so dass $\sum_{i=1}^{n}(2i-1) = n^2$ (also $P(n)$); zu zeigen ist dann $P(n+1)$, also $\sum_{i=1}^{n+1}(2i-1) = (n+1)^2$. Man rechnet wie folgt:

$$\begin{aligned}\sum_{i=1}^{n+1}(2i-1) &= \sum_{i=1}^{n}(2i-1) + 2(n+1) - 1\\ &\overset{IV}{=} n^2 + 2(n+1) - 1\\ &= n^2 + 2n + 1 = (n+1)^2.\end{aligned}$$

  Hierbei haben wir mit IV den Umformungsschritt markiert, in dem die Induktionsvoraussetzung $P(n)$ angewendet wird.

Course-Of-Values Induction Nicht immer führt Induktion nach obigem Schema zum Ziel. Wenn wir z.B. den Fundamentalsatz der Arithmetik

Jede positive natürliche Zahl ist ein endliches Produkt von Primzahlen

beweisen wollen, wird uns die Annahme, dass n ein Produkt von Primzahlen ist, erkennbar nicht weiterhelfen beim Beweis der Behauptung, dass $n+1$ ein Produkt von Primzahlen ist (im Gegenteil teilen ja die Primzahlen, aus denen n zusammengesetzt ist, $n+1$ gerade *nicht*). Stattdessen verwenden wir folgendes stärkere Induktionsprinzip:

Satz 1 (Course-of-Values Induction)**.** *Sei $P(n)$ eine Aussage über natürliche Zahlen.*[1] *Wenn für jedes n aus $\forall k < n\,(P(k))$ bereits $P(n)$ folgt, als Formel:*

$$\forall n\,(\,\forall\, k < n\,(P(k)) \implies P(n)\,),$$

so gilt P für jede natürliche Zahl (als Formel: $\forall n\,(P(n))$).

[1] Wir sind hier, wie schon vorher, ungenau bezüglich der Ausdrucksmittel, die zur Formulierung von P zur Verfügung stehen, insofern bleibt der Satz hier zum Teil informell. Man könnte mit weiter unten eingeführtem Wissen z.B. verlangen, dass P eine Formel in Logik erster Stufe ist, die nur 0 und Nachfolger erwähnt.

Beweis. In der Tat lässt sich diese Prinzip mittels normaler Induktion über n beweisen. Dies ist gleichzeitig eine Illustration des Prinzip der *Verstärkung des Induktionsziels*: Wenn sich eine Aussage $P(n)$ nicht durch Induktion beweisen lässt, kommt man oft weiter, wenn man stattdessen eine *stärkere* Aussage $Q(n)$ (d.h. eine Aussage $Q(n)$, so dass für jedes n die Aussage $P(n)$ aus $Q(n)$ folgt) per Induktion beweist. Man hat dann zwar im Induktionsschritt mehr zu zeigen, hat aber dazu eine stärkere Induktionsannahme zur Verfügung. Auch im vorliegenden Fall kann man sich überzeugen, dass die Induktionsannahme $P(n)$ unter den Annahmen des Satzes nicht ausreicht, um $P(n+1)$ zu folgern (dazu braucht man $P(k)$ für alle $k < n+1$, die Induktionsannahme liefert dies aber nur für den Fall $k = n$). Stattdessen beweisen wir unter den Annahmen des Satzes die stärkere Aussage

$$\forall\, k \leqslant n\, (P(n))$$

durch Induktion über n:

- Induktionsanfang: nach Annahme können wir $P(0)$ folgern, wenn $P(k)$ für alle natürlichen Zahlen $k < 0$ gilt. Da es keine solchen Zahlen gibt, ist dies der Fall, also gilt $P(0)$. Damit gilt natürlich auch $\forall\, k \leqslant 0\, (P(k))$.
- Induktionsschritt: Es gelte $\forall\, k \leqslant n\, (P(n))$; zu zeigen ist $\forall\, k \leqslant n+1\, (P(n))$. Für die meisten k folgt dies sofort aus der Induktionsannahme; zu zeigen bleibt $P(n+1)$. Nach der Annahme des Satzes reicht es dazu aus, zu zeigen, dass $P(k)$ für alle $k < n+1$ gilt; das ist aber gerade unsere Induktionsannahme $\forall\, k \leqslant n\, (P(n))$. □

Mit diesem Prinzip beweisen wir nun den eingangs erwähnten Fundamentalsatz der Arithmetik: Sei $n > 0$. Wir nehmen an, jede Zahl $k < n$ sei ein Produkt von endlich vielen Primzahlen (Sonderfälle hierbei per Konvention: 1 ist ein Produkt von 0 Primzahlen, und jede Primzahl ist Produkt aus einer einzigen Primzahl). Wir müssen zeigen, dass dann n selbst ein Produkt endlich vieler Primzahlen ist. Wir unterscheiden dazu zwei Fälle: Wenn n selbst prim ist oder $n = 1$, dann ist n per eben vereinbarter Konvention ein endliches Produkt von Primzahlen. Andernfalls ist n zusammengesetzt, also $n = km$ mit $k, m < n$. Nach Induktionsvoraussetzung sind dann k und m endliche Produkte von Primzahlen, also $k = p_1 \dots p_r$, $m = q_1 \dots q_s$ mit $p_1, \dots, p_r, q_1, \dots, q_s$ Primzahlen; damit ist auch $n = km = p_1 \dots p_r q_1 \dots q_s$ ein Produkt endlich vieler Primzahlen.

2.1.1 Backus-Naur-Form

Die Backus-Naur-Form (BNF) ist eine verbreitete Art, die Syntax von formalen Sprachen (genauer: sogenannten kontextfreien Sprachen, s. BFS) darzustellen. Man arbeitet mit zwei Alphabeten T und N von *terminalen* und *nichtterminalen* Symbolen; wir beschränken uns hier der Einfachheit halber auf den Fall mit nur einem nichtterminalen Symbol. Eine BNF hat dann die Form

$$n \ ::= \ B_1 \mid \dots \mid B_m$$

mit $n \in N$ und $B_1, \dots, B_m \in (N \cup T)^*$; die B_i heißen *Alternativen*. Eine Alternative

$$B_i = w_0 n w_1 n \dots n w_k \text{ mit } w_0, \dots, w_k \in T^*$$

lesen wir als eine Regel zur Einführung von *Instanzen* von n:

wenn $v_1, \ldots, v_k$ Instanzen von n sind, dann auch $w_0 v_1 w_1 v_2 \ldots v_k w_k$.

Die so entstehende Regelmenge lesen wir *induktiv*, d.h. ein Wort über T ist dann eine Instanz von n, wenn sich dies durch endlich viele Regelanwendungen herleiten lässt. Insbesondere sind alle Instanzen von n endliche Wörter.

Unser erstes Beispiel ist die Grammatik

$$\varphi, \psi ::= \bot \mid A \mid \varphi \wedge \psi \mid \neg\, \varphi \qquad (A \in \mathscr{A}),$$

wobei wir zwecks leichterer Notation zwei verschiedene Namen ϕ, ψ für dasselbe Nichtterminal verwenden. Dabei ist $\mathscr{A}$ eine gegebene Menge von *Atomen*, d.h. nicht weiter unterteilbaren Aussagen. Wir haben hier also $T = \mathscr{A} \cup \{\bot, \wedge, \neg\}$ (bzw. $T = \mathscr{A} \cup \{\bot, \wedge, \neg,), (\}$, wobei wir aber Klammern in der Grammatik implizit lassen und Terme nur bei Bedarf klammern). Instanzen von ϕ nennen wir *(aussagenlogische) Formeln.* Dies entspricht in der Lesart als Regeln dem Regelsystem

1. $\bot$ und $A \in \mathscr{A}$ sind Formeln.
2. Wenn ϕ eine Formel ist, dann auch $\neg\phi$.
3. Wenn ϕ und ψ Formeln sind, dann auch $\phi \wedge \psi$.

Z.B. ist $A \wedge \neg\bot$ eine Formel; durch Rückwärtsanwenden der Regeln sieht man dies wie folgt:

- $A \wedge \neg\bot$ ist eine Formel, denn (3):
- $A \in \mathscr{A}$ ist eine Formel (1) und $\neg\bot$ ist eine Formel, denn (2):
- $\bot$ ist eine Formel (1).

2.2 Strukturelle Induktion

Man kann nun Induktion nicht nur über den natürlichen Zahlen verwenden, sondern auch über im wesentlichen allen endlichen azyklischen Datenstrukturen (und sogar noch allgemeineren Objekten, was hier aber zu weit führt) – insbesondere z.B. über mittels einer Grammatik definierten Objekten, wie etwa aussagenlogischen Formeln.

Wir stellen uns ein solches Objekt dabei eher als eine baumförmige Struktur als einen flachen String vor (d.h. wir stellen uns z.B. Formeln fertig geparst vor). In ihrer einfachsten Form besagt strukturelle Induktion dann, dass jede Eigenschaft, die für alle Blätter gilt und sich von direkten Kindern auf Elternknoten vererbt, für alle Bäume gilt.

Formal stellt sich dies wie folgt dar: Aus einer BNF

$$n ::= B_1 \mid \cdots \mid B_m$$

erhalten wir ein Induktionsprinzip zum Beweis einer Eigenschaft P für alle Instanzen von n, in dem m verschiede Induktionsschritte durchzuführen sind, einer für jedes B_i. Der Induktionsschritt für $B_i = w_0 n w_1 n \ldots n w_k$ verlangt, dass man unter der Annahme, dass Instanzen $v_1, \ldots, v_k$ von n bereits P erfüllen (Induktionsvoraussetzung), zeigt, dass auch die neu erzeugte Instanz

$$w_0 v_1 w_1 \ldots v_k w_k$$

die Eigenschaft P erfüllt. Wenn n nicht in B_i vorkommt, B_i also nur aus terminalen Symbolen besteht, ist der Induktionschritt für B_i natürlich eher eine Art Induktionsanfang (von denen es dann mehrere geben kann), da man keine Induktionsvoraussetzung hat. Die Rechtfertigung dieses Induktionsprinzips, d.h. der Beweis der Tatsache, dass man nach Durchführung aller Induktionsschritte tatsächlich folgern kann, dass $P(w)$ für alle Instanzen w von n gilt, ist per Course-of-Values-Induktion über die Länge von w, bei Fallunterscheidung über die Regel, mit der man w erzeugt hat.

Ein erstes Beispiel dieses Prinzips ist die eingangs diskutierte Induktion über natürliche Zahlen. Wir können nämlich die natürlichen Zahlen als die Instanzen der Grammatik

$$n ::= z \mid s(n)$$

ansehen – diese sind z, $s(z)$, $s(s(z))$, $s(s(s(z)))$, ... Dann haben wir gemäß den obigen Vorschriften beim Beweis einer Eigenschaft P für alle Instanzen von n zwei Induktionsschritte, einen für jede Alternative der Grammatik:

- z: Hier kommt n nicht vor, zu zeigen ist also einfach $P(z)$. Dies entspricht dem üblichen Induktionsanfang (z steht für 0).
- $s(n)$: Hier ist zu zeigen, dass, wenn n die Eigenschaft P hat, dann auch $s(n)$, wobei jetzt n als Platzhalter für eine beliebige Instanz steht. Da s für die Nachfolgerfunktion steht, entspricht dies genau dem üblichen Induktionsschritt.

Für aussagenlogische Formeln erhalten wir ein strukturelles Induktionsprinzip mit vier Induktionsschritten (von denen zwei in Wirklichkeit Induktionsanfänge sind): Um zu zeigen, dass eine Eigenschaft $P(\phi)$ für alle aussagenlogischen Formeln ϕ gilt, zeigt man

- $P(\bot)$;
- $P(A)$ für alle $A \in \mathscr{A}$;
- wenn $P(\phi)$, dann auch $P(\neg\phi)$; und
- wenn $P(\phi)$ und $P(\psi)$, dann auch $P(\phi \wedge \psi)$.

Wir verwenden dieses Prinzip ganz entsprechend auch zur *rekursiven Definition* von Funktionen, wie etwa in der folgenden Definition.

Definition 2 (Atome einer Formel)**.** Die Menge $\mathrm{At}(\varphi)$ der in φ vorkommenden Atome ist rekursiv definiert durch

$$\begin{aligned} \mathrm{At}(A) &= \{A\} \\ \mathrm{At}(\top) &= \varnothing \\ \mathrm{At}(\neg\varphi) &= \mathrm{At}(\varphi) \\ \mathrm{At}(\varphi \wedge \psi) &= \mathrm{At}(\varphi) \cup \mathrm{At}(\psi) \end{aligned}$$

Das Schema der rekursiven Aufrufe ist dasselbe wie das der Induktionsvoraussetzungen im Induktionsprinzip, d.h. rekursive Aufrufe erfolgen immer auf die Bestandteile des aktuellen Arguments – bei der Klausel für $\mathrm{At}(\varphi \wedge \psi)$ z.B. auf ϕ und ψ. Als Beispiel zeigen wir

Für jede Formel ϕ ist die Menge $\mathrm{At}(\phi)$ *endlich*

durch Induktion über ϕ:

- $\mathrm{At}(\bot) = \varnothing$ ist endlich.
- Für $A \in \mathscr{A}$ ist $\mathrm{At}(A) = \{A\}$ endlich.
- Sei $\mathrm{At}(\phi)$ endlich; dann ist auch $\mathrm{At}(\neg\phi) = \mathrm{At}(\phi)$ endlich.
- Seien $\mathrm{At}(\phi)$ und $\mathrm{At}(\psi)$ endlich; dann ist auch $\mathrm{At}(\phi \wedge \psi) = \mathrm{At}(\phi) \cup \mathrm{At}(\psi)$ endlich, da die Vereinigung zweier endlicher Mengen wieder eine endliche Menge ist.

3 Aussagenlogik

Wir kommen nunmehr zur formalen Behandlung der Aussagenlogik hinsichtlich Syntax, Semantik und Beweistheorie.

3.1 Syntax

Wir definieren (wie im Abschnitt über Induktion) die Menge der aussagenlogischen Formeln φ, ψ durch die Grammatik

$$\varphi, \psi ::= \bot \mid A \mid \varphi \wedge \psi \mid \neg\, \varphi \qquad (A \in \mathscr{A}).$$

Wir vereinbaren:

- $\neg$ bindet am stärksten
- $\top = \neg\bot$
- $(\varphi \vee \psi) = \neg(\neg\varphi \wedge \neg\psi)$
- $\varphi \to \psi = \neg\varphi \vee \psi$
- $\varphi \leftrightarrow \psi = (\varphi \to \psi) \wedge (\psi \to \varphi)$
- $\wedge$ bindet stärker als $\vee$, und $\vee$ bindet stärker als $\to$ und $\leftrightarrow$.

(Hierbei bezeichnet $=$ syntaktische Gleichheit; z.B. $A \wedge B \neq B \wedge A$!)

3.2 Semantik

3.2.1 Informelle Semantik

Die Aussprache und intuitive Bedeutung der logischen Operatoren ist

$$\begin{aligned} \varphi \wedge \psi &\quad \text{„}\phi \text{ und } \psi\text{“} \\ \neg\varphi &\quad \text{„nicht } \phi\text{“} \\ \varphi \vee \psi &\quad \text{„}\phi \text{ oder } \psi\text{“} \\ \varphi \to \psi &\quad \text{„wenn } \phi\text{, dann } \psi\text{“} \\ \varphi \leftrightarrow \psi &\quad \text{„genau dann } \varphi\text{, wenn } \psi\text{“} \\ \top &\quad \text{„Wahr“} \\ \bot &\quad \text{„Falsch“} \end{aligned}$$

Hierbei ist „oder“ als *inklusives Oder* zu lesen, d.h. es dürfen auch beide Aussagen wahr sein; „wenn“-„dann“ ist eine *materielle Implikation*, d.h. wenn ϕ nicht gilt, ist $\phi \to \psi$ stets wahr – insbesondere auch dann, wenn ψ falsch ist! Ferner ist $\phi \to \psi$ stets wahr, wenn ψ gilt, ohne Rücksicht darauf, ob diese Tatsache etwas mit ϕ zu tun hat, oder ϕ auch nur wahr ist.

Für die logischen Operatoren werden folgende sprachliche Bezeichnungen verwendet:

$\neg$	Negation
$\wedge$	Konjunktion
$\vee$	Disjunktion
$\to$	Implikation
$\leftrightarrow$	Biimplikation, Äquivalenz
$\top$	Wahrheit, konstant wahre Aussage, Verum
$\bot$	Falschheit, konstant falsche Aussage, Absurdität, Falsum

3.2.2 Wahrheitsbelegungen und Erfülltheit

Im Allgemeinen sind Formeln nicht schlechthin wahr oder falsch, sondern erhalten erst durch die Festlegung konkreter Wahrheitswerte für die propositionalen Atome einen Wahrheitswert. Formal stellt sich dies wie folgt dar:

Definition 3 (Wahrheitsbelegung, Erfülltheit)**.** Wir schreiben $2 = \{\bot, \top\}$ für die Menge der *Wahrheitswerte*. Eine *Wahrheitsbelegung (WB)* ist eine Abbildung

$$\kappa : \mathscr{A} \to 2,$$

legt also Wahrheitswerte für alle Atome fest.

Wir definieren die *Erfüllheitsrelation* $\kappa \models \varphi$ (lies: κ erfüllt φ) rekursiv (genauer gesagt per struktureller Rekursion wie in Abschnitt 2 eingeführt) durch

$$\begin{aligned} \kappa &\not\models \bot \\ \kappa \models A &\iff \kappa(A) = \top \\ \kappa \models \varphi \wedge \psi &\iff \kappa \models \varphi \text{ und } \kappa \models \psi \\ \kappa \models \neg\varphi &\iff \kappa \not\models \varphi. \end{aligned}$$

In Worten: κ erfüllt $\bot$ nicht; κ erfüllt ein Atom A, wenn es A den Wert „wahr“ zuweist; κ erfüllt die Konjunktion zweier Formeln, wenn κ beide Formeln erfüllt; κ erfüllt die Negation einer Formel ϕ, wenn κ die Formel ϕ nicht erfüllt.

Erfülltheit verhält sich für die abgeleiteten Operatoren wie folgt:

$$\begin{aligned}
\kappa &\vDash \top \text{ stets} \\
\kappa \vDash \varphi \vee \psi &\iff (\kappa \vDash \varphi \text{ oder } \kappa \vDash \psi) \\
\kappa \vDash \varphi \to \psi &\iff (\text{falls } \kappa \vDash \varphi, \text{ so auch } \kappa \vDash \psi) \\
\kappa \vDash \varphi \leftrightarrow \psi &\iff (\kappa \vDash \varphi \text{ genau dann, wenn } \kappa \vDash \psi)
\end{aligned}$$

Beispiel 4. Die Wahrheitsbelegung κ ordne den Atomen A und B die Werte $\kappa(A) = \top$ und $\kappa(B) = \bot$ zu; Kurzschreibweise: $\kappa = [A \mapsto \top, B \mapsto \bot]$ (das spezifiziert κ nicht eindeutig, was im Folgenden aber nicht stört). Dann gilt

$$\kappa \vDash ((A \vee \neg B) \to B) \iff (\text{falls } \kappa \vDash (A \vee \neg B), \text{ so auch } \kappa \vDash B)$$

Es gilt $\kappa \vDash A \vee \neg B \iff (\kappa \vDash A \text{ oder } \kappa \vDash \neg B)$. Nun gilt $\kappa(A) = \top$, also $\kappa \vDash A$, somit $\kappa \vDash A \vee \neg B$. Es gilt aber $\kappa \nvDash B$ (da $\kappa(B) = \bot$), also $\kappa \nvDash (A \vee \neg B) \to B$.

Andere Wahrheitsbelegungen können die Formel aber erfüllen, z. B. gilt $\kappa_2 \vDash (A \vee \neg B) \to B$ für $\kappa_2 = [A \mapsto \top, B \mapsto \top]$, oder in der Tat für jedes κ_2 mit $\kappa_2(B) = \top$ (nie dagegen, wenn $\kappa_2(B) = \bot$).

Wir dehnen den Begriff der *Erfülltheit* auf Mengen Φ von Formeln aus: Eine Wahrheitsbelegung κ *erfüllt* Φ genau dann, wenn κ alle Formeln φ erfüllt, die in Φ enthalten sind, d.h.

$$\kappa \vDash \Phi \ :\iff \ \forall\, \varphi \in \Phi\, (\kappa \vDash \varphi).$$

(Insbesondere gilt also $\kappa \vDash \varnothing$ stets.)

3.2.3 Erfüllbarkeit, Gültigkeit, logische Konsequenz

Aus der eben angegebenen Semantik ergeben sich verschiedene von einzelnen Wahrheitsbelegungen unabhängige, nur auf Formeln anzuwendende semantische Begriffe:

Definition 5. Seien ϕ, ψ Formeln und Φ eine Menge von Formeln.

- Φ ist *erfüllbar*, wenn eine Wahrheitsbelegung κ mit $\kappa \vDash \Phi$ existiert; ϕ ist *erfüllbar*, wenn $\{\phi\}$ erfüllbar ist.
- ϕ ist *gültig*, wenn $\kappa \vDash \phi$ für *alle* Wahrheitsbelegungen κ; wir schreiben dann $\vDash \phi$.
- ϕ und ψ sind *logisch äquivalent*, in Symbolen $\phi \equiv \psi$, wenn $\phi \leftrightarrow \psi$ gültig ist, d.h. wenn für jede Wahrheitsbelegung κ genau dann $\kappa \vDash \phi$ gilt, wenn $\kappa \vDash \psi$ gilt.
- ψ ist eine *logische Folgerung* oder *logische Konsequenz* aus Φ, in Symbolen $\Phi \vDash \psi$, wenn für jede Wahrheitsbelegung κ mit $\kappa \vDash \Phi$ auch $\kappa \vDash \psi$ gilt.

Logische Folgerung bezeichnet also das Verhältnis, das zwischen einer gegebenen Menge von Formeln, den *Annahmen*, und einer einzelnen Formel, der *Folgerung*, besteht, wenn immer dann, wenn die Annahmen erfüllt sind, auch die Folgerung erfüllt ist.

Achtung: Für $\Phi \models \psi$ wird ausdrücklich *nicht* verlangt, dass umgekehrt auch aus $\kappa \models \psi$ stets $\kappa \models \Phi$ folgt.

Beispiel 6 (Erfüllbarkeit und logische Konsequenz)**.**

Erfüllbar: $A \to \neg A$ (eine erfüllende Wahrheitsbelegung ist $\kappa(A) = \bot$)

Unerfüllbar: $A \wedge \neg A$

Gültig: $A \vee \neg A$, $(A \wedge B) \to A$

Logische Konsequenz: $\{A \to B, A\} \models B$ (diesen Schluss nennt man „modus ponens“)

Es ergibt sich unmittelbar aus den Definitionen, dass eine Formelmenge Φ genau dann unerfüllbar ist, wenn $\Phi \models \bot$, und dass eine Formel ϕ genau dann gültig ist, wenn $\varnothing \models \phi$; im letzteren Fall schreiben wir vereinfacht auch $\models \phi$. Wir haben folgende weitere Beziehungen zwischen den einzelnen Begriffen:

Lemma 7. *Seien ψ eine Formel und Φ eine Menge von Formel. Dann gilt*

1. *ψ ist genau dann gültig, wenn $\neg\psi$ unerfüllbar ist.*
2. *ψ ist genau dann erfüllbar, wenn $\neg\psi$ nicht gültig ist.*
3. *ψ ist genau dann logische Folgerung aus Φ, wenn die Vereinigung von Φ und der Negation von ψ unerfüllbar ist:*
$$\Phi \models \psi \iff (\Phi \cup \{\neg\psi\}) \models \bot.$$

Beweis. Teil 1 ergibt sich einfach durch Spezialisierung von 3 auf $\Phi = \varnothing$. Ferner erhält man 2 per Anwendung von 1 auf $\neg\psi$, mittels der logischen Äquivalenz $\neg\neg\psi \equiv \psi$. Für 3 zeigen wir zwei Implikationen:

„$\Longleftarrow$“: Sei $\kappa \models \Phi$; wir müssen $\kappa \models \psi$ zeigen. Beweis durch Widerspruch: wenn $\kappa \not\models \psi$, dann per Definition $\kappa \models \neg\psi$, also $\kappa \models \Phi \cup \{\neg\psi\}$. Letzteres ist aber nach Annahme unerfüllbar, Widerspruch.

„$\Longrightarrow$“: Wir nehmen zwecks Widerspruchs an, es gäbe κ mit $\kappa \models \Phi \cup \{\neg\psi\}$. Dann gilt $\kappa \models \Phi$, aber $\kappa \not\models \psi$, im Widerspruch zu $\Phi \models \psi$. □

3.3 Wahrheitstafeln

Eine Wahrheitstafel ist eine tabellarische Auflistung der Wahrheitswerte einer Formel in Abhängigkeit von den Wahrheitswerten der (endlich vielen!) in ihr vorkommenden Atome. Wahrheitstafeln liefern Entscheidungsalgorithmen für Erfüllbarkeit, Gültigkeit, logische Konsequenz und logische Äquivalenz in der Aussagenlogik; diese sind aber offenbar in der Praxis nicht skalierbar, da stets die gesamte (exponentiell große) Wahrheitstafel erzeugt werden muss. Konkret:

- φ ist gültig genau dann, wenn alle Werte für φ in der Wahrheitstafel $\top$ sind .
- φ ist erfüllbar genau dann, wenn $\top$ als Wert für φ in der Wahrheitstafel für φ vorkommt.
- $\varphi \equiv \psi$ genau dann, wenn ϕ und ψ in der gemeinsamen (!) Wahrheitstafel in jeder Zeile den gleichen Wert haben.
- $\Phi \models \phi$ (für Φ endlich) genau dann, wenn in der gemeinsamen Wahrheitstafel für Φ und ψ die Formel ψ in jeder Zeile, in der alle $\varphi \in \Phi$ den Wert $\top$ haben, ebenfalls den Wert $\top$ hat.

Beispiel 8. Wahrheitstafel von $A \to B = \neg A \vee B$:

A	B	$\neg A$	$\neg A \vee B$
$\bot$	$\bot$	$\top$	$\top$
$\bot$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\bot$
$\top$	$\top$	$\bot$	$\top$

$\neg A \vee A$ ist gültig:

A	$\neg A$	$\neg A \vee A$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\top$

$A \to \neg A$ ist erfüllbar:

A	$\neg A$	$A \to \neg A$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\bot$

$\neg(A \to B) \models A$:

A	B	$A \to B$	$\neg(A \to B)$
$\bot$	$\bot$	$\top$	$\bot$
$\bot$	$\top$	$\top$	$\bot$
$\top$	$\bot$	$\bot$	$\top$
$\top$	$\top$	$\top$	$\bot$

$\{A \to B, A\} \models B$:

A	B	$A \to B$
$\bot$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\bot$
$\top$	$\top$	$\top$

(Im letzten Beispiel muss man Erfülltheit von B nur in der letzten Zeile prüfen, da nur dort die beiden Annahmen A und $A \to B$ erfüllt sind.)

$B \vee \neg B \equiv A \vee \neg A$: bei beiden Formeln steht nur $\top$ in der Wahrheitstafel.

Im letzten Beispiel sieht man, dass zwei Formeln trotz unterschiedlicher verwendeter Atome äquivalent sein können. Wir erinnern noch einmal an die formale Definition der in einer Formel verwendeten Atome per Rekursion:

Definition 9 (Atome einer Formel)**.** Die Menge $\mathrm{At}(\varphi)$ der in φ vorkommenden Atome ist wie im Abschnitt über Induktion rekursiv definiert durch

$$\begin{aligned}\mathrm{At}(A) &= \{A\}\\ \mathrm{At}(\top) &= \varnothing\\ \mathrm{At}(\neg\varphi) &= \mathrm{At}(\varphi)\\ \mathrm{At}(\varphi \wedge \psi) &= \mathrm{At}(\varphi) \cup \mathrm{At}(\psi)\end{aligned}$$

Beispiel 10. Wir berechnen $\mathrm{At}((A \wedge B) \wedge \neg A)$:

$$\begin{aligned}\mathrm{At}((A \wedge B) \wedge \neg A) &= \mathrm{At}(A \wedge B) \cup \mathrm{At}(\neg A)\\ &= \mathrm{At}(A) \cup \mathrm{At}(B) \cup \mathrm{At}(A)\\ &= \{A\} \cup \{B\} \cup \{A\} = \{A, B\}\end{aligned}$$

Das folgende Lemma ist die formale Legitimation dafür, dass Wahrheitstafeln sich auf die in φ vorkommenden Atome beschränken dürfen.

Lemma 11. *Die Erfülltheit $\kappa \vDash \varphi$ hängt nur von den Belegungen der Atome von φ, also von den Werten $\kappa(A)$ für $A \in \mathrm{At}(\varphi)$ ab; d.h. wenn κ und κ' auf $\mathrm{At}(\varphi)$ übereinstimmen, so erfüllt κ genau dann φ, wenn κ' dies tut. Formal: Wenn $\kappa, \kappa' : \mathscr{A} \to 2$ mit $\kappa(A) = \kappa'(A)$ für alle $A \in \mathrm{At}(\varphi)$, dann gilt*

$$\kappa \vDash \varphi \iff \kappa' \vDash \varphi.$$

Beweis. Strukturelle Induktion über ϕ. Wir gehen alle Alternativen der Grammatik durch:

$\bot$: Es gilt $\kappa \nvDash \bot$ und $\kappa' \nvDash \bot$.

A: Es $A \in \mathrm{At}(A)$, also $\kappa \vDash A \iff \kappa(A) = \top \iff \kappa'(A) = \top \iff \kappa' \vDash A$.

$\neg\phi$: Es gilt $\mathrm{At}(\neg\phi) = \mathrm{At}(\phi)$, d.h. κ und κ' stimmen auf $\mathrm{At}(\phi)$ überein. Daher $\kappa \vDash \neg\phi \iff \kappa \nvDash \phi \overset{\text{IV}}{\iff} \kappa' \nvDash \phi \iff \kappa' \vDash \neg\phi$.

$\phi \wedge \psi$: Es gilt $\mathrm{At}(\phi \wedge \psi) \supseteq \mathrm{At}(\phi), \mathrm{At}(\psi)$, d.h. κ und κ' stimmen auf $\mathrm{At}(\phi)$ und auf $\mathrm{At}(\psi)$ überein. Daher $\kappa \vDash \phi \wedge \psi \iff (\kappa \vDash \phi \text{ und } \kappa \vDash \psi) \overset{\text{IV}}{\iff} (\kappa' \vDash \phi \text{ und } \kappa' \vDash \psi) \iff \kappa' \vDash \phi \wedge \psi$.

□

Die Semantik von φ ist also bestimmt durch endliche Tabellierung von $\kappa \vDash \varphi$ für alle $\kappa : A_0 \to 2$ mit $A_0 \subseteq \mathscr{A}$ endlich, $\mathrm{At}(\varphi) \subseteq A_0$. Dies ist die formale Rechtfertigung für das Wahrheitstafelverfahren; wir geben für den Fall der logischen Äquivalenz eine explizite Präzisierung an:

Lemma 12. *Für Formeln ϕ, ψ gilt $\varphi \equiv \psi$ genau dann wenn φ, ψ identische Wahrheitstafeln über $\mathrm{At}(\varphi) \cup \mathrm{At}(\psi)$ haben.*

3.4 Logische Äquivalenzen

Im folgenden geben wir eine Übersicht wichtiger logischer Äquivalenzen.

$\neg\neg\varphi \equiv \varphi$ (Doppelnegationselimination)

$\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi)$ (De Morgansche Gesetze)
$\neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$

$\varphi \wedge (\psi \vee \chi) \equiv (\varphi \wedge \psi) \vee (\varphi \wedge \chi)$ (Distributivgesetze)
$\varphi \vee (\psi \wedge \chi) \equiv (\varphi \vee \psi) \wedge (\varphi \vee \chi)$

$(\varphi \wedge \psi) \wedge \chi \equiv \varphi \wedge (\psi \wedge \chi)$ (Assoziativgesetze)
$\varphi \vee (\psi \vee \chi) \equiv (\varphi \vee \psi) \vee \chi$

$\chi \wedge \top \equiv \chi$ (Neutrale Elemente)
$\chi \vee \bot \equiv \chi$

$\varphi \wedge \psi \equiv \psi \wedge \varphi$ (Kommutativität)
$\varphi \vee \psi \equiv \psi \vee \varphi$

$\varphi \wedge \varphi \equiv \varphi$ (Idempotenz)
$\varphi \vee \varphi \equiv \varphi$

4 Formale Deduktion in Aussagenlogik

Systeme formalen Schließens dienen der rein syntaxbasierten *Herleitung* von Formeln, die einfach als zu manipulierende Zeichenketten angesehen werden. Ein Deduktionssystem besteht typischerweise aus *Axiomen*, also Formeln, die ohne weitere Voraussetzungen als hergeleitet hingeschrieben werden können, sowie aus *Regeln*, die festlegen, wie man aus bereits hergeleiteten Formeln bzw. schon durchgeführten Herleitungen neue Formeln herleitet. Im einfachsten Fall hat eine Regel die Form

$$\frac{\text{Prämissen}}{\text{Konklusion}}.$$

Die *Prämissen* repräsentieren Formeln, die zur Anwendung der Regel bereits hergeleitet sein müssen; die Regel gestattet dann die Herleitung der *Konklusion*. In komplizierteren Fällen können die Prämissen statt Formeln auch ganze Herleitungen sein; dazu sehen wir später Beispiele. Zusätzlich kann eine Regel *Seitenbedingungen* haben, also außerhalb der logischen Syntax ausgedrückte Bedingungen für die Anwendbarkeit der Regel, idealerweise einfache Zusatzforderungen an die syntaktische Struktur wie etwa das Vorkommen oder Nichtvorkommen gewisser Symbole.

Man unterscheidet verschiedene Typen von formalen Deduktionssystemen, u.a. nach der Gewichtung zwischen Axiomen und Regeln:

Hilbert Viele Axiome, wenig Regeln; meist nur *modus ponens*

$$(mp)\ \frac{\phi \quad \phi \to \psi}{\psi}$$

Gentzen Regeln für *Sequenten* $\phi_1, \ldots, \phi_n \vdash \psi_1, \ldots, \psi_k$, zu lesen als „die Konjunktion der ϕ_i impliziert die Disjunktion der ψ_j“. Axiome (in der Grammatik mit nur $\bot, \wedge, \neg$) sind nur Sequenten der Form $\phi, \cdots \vdash \phi, \ldots$ oder $\bot, \cdots \vdash \ldots$.

Natürliches Schließen Variante des Sequentenkalküls, in der nur eine Formel rechts von $\vdash$ steht[2] und die linke Seite implizit gelassen wird; stattdessen hat man lokale Mengen von Annahmen, die in hierarchisch strukturierten Beweisen rekursiv aufgebaut werden.

Wir lernen im folgenden ein System natürlichen Schließens kennen, den Fitch-Kalkül.

Oft sind die Regeln eines Kalküls so organisiert, dass man für jedes logische Konnektiv Regeln sowohl zur Einführung (I wie *Introduction*) als auch zur Entfernung (E wie *Elimination*) bereitstellt. Diese Struktur weist auch das folgende einführende Beispiel auf.

Beispiel 13 (Regeln für eine auf Konjunktion beschränkte Logik)**.**

$$(\wedge I)\ \frac{\phi \quad \psi}{\phi \wedge \psi} \qquad (\wedge E1)\ \frac{\phi \wedge \psi}{\phi} \qquad (\wedge E2)\ \frac{\phi \wedge \psi}{\psi}$$

Die Regel $(\wedge I)$ („Und-Einführung“) erlaubt es uns, sofern wir bereits ϕ und ψ hergeleitet haben, auch $\phi \wedge \psi$ herzuleiten; die Regeln $(\wedge E1)$ und $(\wedge E2)$ („Und-Elimination“) erlauben es, aus einer bereits hergeleiteten Konjunktion deren Konjunkte herzuleiten.

Notation: Für eine Formel ψ und eine Menge Φ von Formeln schreiben wir $\Phi \vdash \phi$, wenn ϕ mittels der gegebenen Regeln (also insbesondere rein syntaktisch) aus Annahmen in Φ herleitbar ist. Für Beweise verwenden wir oft eine baumartige Schreibweise, wie z.B. in folgender Herleitung von $\psi \wedge \phi$ aus der Annahme $\phi \wedge \psi$, die also $\{\phi \wedge \psi\} \vdash \psi \wedge \phi$ bezeugt:

$$\dfrac{(\wedge E2)\dfrac{\phi \wedge \psi}{\psi} \qquad (\wedge E1)\dfrac{\phi \wedge \psi}{\phi}}{\psi \wedge \phi}(\wedge I)$$

Der Übersichtlichkeit halber verwenden wir im folgenden alternativ auch eine linearisierte Darstellung des Beweisablaufs:

1	$\phi \wedge \psi$		
2	ψ	$(\wedge E2)$	(1)
3	ϕ	$(\wedge E1)$	(1)
4	$\psi \wedge \phi$	$(\wedge I)$	$(2, 3)$

Wir notieren in jedem Schritt die hergeleitete Formel (erste Spalte) sowie die verwendete Regel und die Prämissen (zweite Spalte). Die erste Zeile enthält eine Annahme; Annahmen werden von Schlüssen durch einen waagerechten Strich getrennt.

In Beweisen in ausdrucksstärkeren Logiken als der im obigen Beispiel betrachteten Spielzeuglogik kommt es auch vor, dass ein *Unterbeweis* als Prämisse verwendet wird; es ergibt sich daraus eine hierarchische Strukturierung von Beweisen. Unterbeweise haben *lokale* Annahmen, die wie im Beispiel oben durch einen waagerechten Strich abgetrennt werden. Die lokale Annahme

[2]Für sehr Interessierte: Wenn man weiter nachliest, mag zunächst irritieren, dass die Beschränkung auf genau eine Formel auf der rechten Seite gerade das ist, was Gentzens intuitionistischen Sequentenkalkül LJ vom klassischen Kalkül LK unterscheidet. In der Tat machen wir unser System gewissermaßen mit Gewalt klassisch, indem wir ausdrücklich die Doppelnegationsregel hinzunehmen, während diese in LK aus deutlich natürlicher wirkenden Regeln herleitbar ist.

verschwindet mit dem Ende des Unterbeweises (daher der Name), d.h. sie steht zum einen außerhalb des Unterbeweises nicht als Prämisse für Regelanwendungen zur Verfügung, zum anderen ist aber der äußere Beweis auch von ihr unabhängig, d.h. liefert Herleitungen mit entsprechend weniger Annahmen (also stärkere Aussagen). Solche Unterbeweise werden benötigt, wenn wir unser Regelwerk auf Negation und $\bot$ erweitern:

$$(\neg I)\ \dfrac{\begin{array}{|l} \phi \\ \hline \vdots \\ \bot \end{array}}{\neg\phi} \qquad (\bot\ \mathrm{I})\ \dfrac{\phi \quad \neg\phi}{\bot} \qquad (\bot\ \mathrm{E})\ \dfrac{\bot}{\Phi} \qquad (\neg\ \mathrm{E})\ \dfrac{\neg\neg\phi}{\phi}$$

D.h.

- um eine negierte Formel $\neg\phi$ zu beweisen, nimmt man in einem Unterbeweis ϕ an und leitet daraus einen Widerspruch her;
- um einen Widerspruch herzuleiten, zeigt man für irgendeine Formel ϕ sowohl ϕ als auch $\neg\phi$;
- aus einem Widerspruch folgt beliebiges (*ex falso quodlibet*);
- aus einer zweifach negierten Formel $\neg\neg\phi$ folgt ϕ.

Allein in der letzten Regel verbirgt sich übrigens das Prinzip vom ausgeschlossenen Dritten (*tertium non datur*, *excluded middle*).

Beispiel 14. $\vdash \neg(\phi \wedge \neg\phi)$:

1	⊢	
2	⊢ $\phi \wedge \neg\phi$	
3	ϕ	$(\wedge E1), 2$
4	$\neg\phi$	$(\wedge E2), 2$
5	$\bot$	$(\bot I), 2, 3$
6	$\neg(\phi \wedge \neg\phi)$	$(\neg I), 1-4$

Damit ist unser System zunächst einmal fast komplett; aus technischen Gründen gönnen wir uns noch eine *Reiterationsregel*, die uns erlaubt, Annahmen oder Schlüsse aus dem aktuellen Unterbeweis oder aus Unterbeweisen oberhalb des aktuellen noch einmal hinzuschreiben (das Kopieren von Formeln aus anderen Unterbeweisen als den genannten ist natürlich nicht erlaubt, insbesondere z.B. von *unterhalb* des aktuellen Unterbeweises). Wir erwähnen Verwendung dieser Regel meist nicht ausdrücklich.

Erweiterung auf Disjunktion und Implikation Wir haben Disjunktion und Implikation durch Konjunktion und Negation codiert und können daher folgende Regeln für diese Konnektive aus den Regeln für Negation und Konjunktion herleiten:

$$(\vee I1)\frac{\phi}{\phi \vee \psi} \qquad (\vee I2)\frac{\psi}{\phi \vee \psi} \qquad (\vee E)\frac{\phi \vee \psi \qquad \begin{array}{|l} \phi \\ \hline \vdots \\ \chi \end{array} \qquad \begin{array}{|l} \psi \\ \hline \vdots \\ \chi \end{array}}{\chi}$$

$$(\to E)\frac{\phi \to \psi \qquad \phi}{\psi} \qquad (\to I)\frac{\begin{array}{|l} \phi \\ \hline \vdots \\ \psi \end{array}}{\phi \to \psi}$$

Die Eliminationsregel $(\vee E)$ verkörpert dabei das Prinzip der *Fallunterscheidung*: Um χ aus einer Disjunktion $\phi \vee \psi$ herzuleiten, zeigt man, dass man χ sowohl aus der Annahme ϕ als auch aus der Annahme ψ herleiten kann. Die Einführungsregel $(\to I)$ entspricht gerade der üblichen Art, eine Implikation $\phi \to \psi$ herzuleiten: man nimmt ϕ an und leitet daraus ψ her. Die Eliminationsregel $(\to E)$ ist, wie oben schon einmal angedeutet, auch als *modus ponens* bekannt.

Wir führen die Herleitung der Regeln an zwei Beispielen vor:

Herleitung von $(\to I)$ bei Codierung $\phi \to \psi = \neg(\neg\neg\phi \wedge \neg\psi)$

1	$\neg\neg\phi \wedge \neg\psi$	
2	$\neg\neg\phi$	$(\wedge E_1)$ 1
3	ϕ	$(\neg E)$ 2
4	$\vdots$	Annahme
5	ψ	
6	$\neg\psi$	$(\wedge E_2)$ 1
7	$\bot$	$(\bot I)$ 5,6
8	$\neg(\neg\neg\phi \wedge \neg\psi)$	$(\neg I)$ 1–7

Herleitung von $(\to E)$ bei Codierung $\phi \to \psi = \neg\phi \vee \psi$

1	ϕ	
2	$\neg\phi \vee \psi$	
3	$\quad\neg\phi$	
4	$\quad\bot$	$(\bot I)$ 1, 3
5	$\quad\psi$	$(\bot E)$
6	$\quad\psi$	
7	$\quad\psi$	(Reiteration)
8	ψ	$(\vee E)$ 3–5, 6–7, 2

Achtung: In Übungs- und Klausuraufgaben verlangen wir üblicherweise, dass $\vee$ und $\to$ gerade *nicht* dekodiert werden, sondern mittels der oben eingeführten Regeln behandelt werden. So ein Vorgehen ist meist auch einfacher als formale Beweise über die dann doch schnell recht komplizierten dekodierten Formeln.

Satz 15. *(Korrektheit)*

$$\Phi \vdash \psi \Rightarrow \Phi \models \psi$$

Beweis. Wir verallgemeinern zunächst die Behauptung auf Aussagen ψ, die in Unterbeweisen gefolgert werden: Wir behaupten, dass für solche Aussagen $\Phi' \models \psi$ gilt, wobei Φ' aus allen im betreffenden Unterbeweis aktiven Annahmen besteht (inklusive der globalen Annahmen Φ). Wir beweisen die verallgemeinerte Behauptung per Induktion über die Länge n der Herleitung von ψ. Hierbei verwenden wir *course-of-values induction*, d.h. wir beweisen, dass die Behauptung für $n = k$ gilt, wenn sie für alle $n < k$ gilt.

Wir unterscheiden dann nach der zuletzt angewandten Regel, z.B. a) $\wedge I$ (hier im Skript ausgelassen), b) $(\neg I)$: Der Unterbeweis in der Prämisse bedeutet, dass $\Phi \cup \{\phi\} \vdash \bot$, wobei Φ die Menge der bei Anwendung der Regel aktiven Annahmen ist und im Unterbeweis eben die Annahme ϕ dazukommt. Da der Unterbeweis echt kürzer ist als der Gesamtbeweis, können wir auf ihn die Induktionsvoraussetzung anwenden, d.h. es folgt $\Phi \cup \{\phi\} \models \bot$; mit anderen Worten, $\Phi \cup \{\phi\}$ ist unerfüllbar. Damit folgt per Lemma 7 wie verlangt $\Phi \models \neg\phi$. Die anderen Regeln für $\wedge$, $\neg$ und $\bot$ sind noch wesentlich einfacher; die Regeln für $\vee$ und $\to$ sind dann ebenfalls korrekt, da wir sie ja aus denen für $\wedge$, $\neg$ und $\bot$ herleiten. □

4.1 Vollständigkeit

In Umkehrung zur Korrektheit gilt auch

Satz 16. *Vollständigkeit*

$$\Phi \models \psi \Rightarrow \Phi \vdash \psi$$

Definition 17. Eine Formelmenge Φ heißt *konsistent*, wenn $\Phi \not\vdash \bot$, d.h. wenn sich aus Φ kein Widerspruch herleiten lässt. Ferner ist Φ *maximal konsistent*, wenn Φ maximal bezüglich $\subseteq$ unter den konsistenten Mengen ist, d.h.

1. Φ ist konsistent, und
2. wenn Ψ konsistent ist und $\Phi \subseteq \Psi$, dann folgt $\Phi = \Psi$.

Beweis. Strategie des Vollständigkeitsbeweises ist

(A) Zeige

$$\Phi \text{ konsistent} \Rightarrow \Phi \text{ erfüllbar.}$$

Damit folgt dann Vollständigkeit wie im Satz behauptet: Sei $\Phi \models \psi$; dann ist $\Phi \cup \{\neg\psi\}$ unerfüllbar, also nach obigem inkonsistent, also $\Phi \cup \{\neg\psi\} \vdash \bot$. Mittels Regel $(\neg I)$ folgt $\Phi \vdash \neg\neg\psi$, und mittels $(\neg E)$ $\Phi \vdash \psi$.

Slogan: Vollständigkeitsbeweise bestehen in *Modellkonstruktionen*.

(B) Zeige (A) zunächst für den Spezialfall, dass Φ *maximal* konsistent ist.

(C) Beweise das

Lindenbaumlemma: Wenn Φ konsistent ist, dann existiert ein maximal konsistentes $\overline{\Phi}$ mit $\Phi \subseteq \overline{\Phi}$.

Damit folgt dann (A) aus (B): Wenn Φ konsistent ist, dann existiert $\overline{\Phi}$ wie im Lindenbaumlemma. Nach (B) ist $\overline{\Phi}$ erfüllbar, und damit trivialerweise auch Φ.

□

Wir beginnen mit (C) und zeigen dazu:

Lemma 18 (Konsistenzlemma)**.** *Sei Φ konsistent. Dann ist mindestens eine der Mengen $\Phi \cup \{\psi\}$ und $\Phi \cup \{\neg\psi\}$ konsistent.*

Beweis. Per Kontraposition: Seien $\Phi \cup \{\psi\} \vdash \bot$ und $\Phi \cup \{\neg\psi\} \vdash \bot$. Mittels $(\neg I)$ folgt $\Phi \vdash \neg\psi$ und $\Phi \vdash \neg\neg\psi$, also per $(\bot I)$ $\Phi \vdash \bot$, d.h. Φ ist inkonsistent. □

Beweis (Lindenbaumlemma). **a) Zorn**: Vereinigungen aufsteigender Ketten von konsistenten Mengen sind konsistent, also hat die mittels $\subseteq$ geordnete Menge der konsistenten Mengen oberhalb einer gegebenen nach dem Zornschen Lemma maximale Elemente.

oder b): Sei $\varphi_1, \varphi_2, \varphi_3, \ldots$ Aufzählung aller Formeln. Wir konstruieren eine aufsteigende Kette $\Phi = \Phi_0 \subseteq \Phi_1 \subseteq \Phi_2 \subseteq \ldots$ konsistenter Formelmengen per

$$\Phi_{i+1} = \begin{cases} \Phi_i \cup \{\varphi_i\} & \text{wenn konsistent} \\ \Phi_i \cup \{\neg\varphi_i\} & \text{sonst (konsistent per Konsistenzlemma).} \end{cases}$$

Setze $\overline{\Phi} = \bigcup_{i=0}^{\infty} \Phi_i \supseteq \Phi$. Zu zeigen ist dann folgendes:

1. $\overline{\Phi}$ ist konsistent: Nimm an $\overline{\Phi} \vdash \bot$. Der Beweis ist endlich, verwendet also nur endlich viele Annahmen aus $\overline{\Phi}$. Jede dieser Annahmen kommt in einem Φ_i vor; durch Wahl des größten unter diesen endlich vielen Indizes i erhalten wir ein Φ_i, das *alle* verwendeten Annahmen enthält. Dann $\Phi_i \vdash \bot$, im Widerspruch zur Konsistenz von Φ_i.

2. $\overline{\Phi}$ maximal: Sei Ψ konsistent und $\overline{\Phi} \subseteq \Psi$. Zu zeigen ist dann $\Psi \subseteq \overline{\Phi}$. Sei also $\psi \in \Psi$. Es existiert n mit $\psi = \phi_n$. Da $\Phi_n \cup \{\phi_n\} \subseteq \Psi$, ist $\Phi_n \cup \{\phi_n\}$ konsistent, also $\phi_n \in \Phi_{n+1} \subseteq \Phi$.

□

Es bleibt Schritt (B) durchzuführen, d.h. wir müssen zeigen, dass jede maximal konsistente Menge Φ erfüllbar ist. Wir halten folgende Eigenschaften maximal konsistenter Mengen fest:

Lemma 19. *(Hintikka-Eigenschaften) Sei Φ maximal konsistent. Dann gilt*

1. $\bot \notin \Phi$
2. $\neg\psi \in \Phi \iff \psi \notin \Phi$
3. $\phi \wedge \psi \in \Phi \iff \phi \in \Phi$ *und* $\psi \in \Phi$

Beweis. ad (1): Klar.

ad (2): „$\Rightarrow$“: sonst $\Phi \vdash \bot$ per $(\bot I)$, im Widerspruch zur Konsistenz von Φ. „$\Leftarrow$“: Sei $\psi \notin \Phi$. Dann gilt $\Phi \cup \{\psi\} \supsetneq \Phi$. Da Φ maximal konsistent ist, ist $\Phi \cup \{\psi\}$ inkonsistent; nach Konsistenzlemma folgt, dass $\Phi \cup \{\neg\psi\}$ konsistent ist, und per maximaler Konsistenz von Φ folgt $\Phi \cup \{\neg\psi\} \subseteq \Phi$, also $\neg\psi \in \Phi$.

ad (3): Allgemeiner folgt sogar stets aus $\Phi \vdash \psi$, dass $\psi \in \Phi$: Dann ist nämlich $\Phi \cup \{\psi\}$ konsistent (denn aus $\Phi \cup \{\psi\} \vdash \bot$ und $\Phi \vdash \psi$ würde $\Phi \vdash \bot$ folgen), also per Maximalität enthalten in Φ. □

Um nun die verlangte erfüllende Wahrheitsbelegung κ für eine maximal konsistente Menge Φ zu erhalten, setzen wir nun

$$\kappa(A) = \top \iff A \in \Phi.$$

Lemma 20. *(Wahrheitslemma) $\kappa \models \psi \iff \psi \in \Phi$.*

Beweis. Induktion über ψ per Definition und Hintikka-Eigenschaften, z.B.: $\kappa \models \neg\psi \iff \kappa \not\models \psi \overset{\text{IV}}{\iff} \psi \notin \Phi \overset{\text{Hintikka}}{\iff} \neg\psi \in \Phi$. □

Damit gilt wie verlangt $\kappa \models \Phi$. Der Vollständigkeitsbeweis ist damit beendet.

Eine unmittelbare Folgerung aus der Vollständigkeit ist die folgende Eigenschaft:

Korollar 21. *(Kompaktheit) Sei Φ eine Formelmenge, so dass alle endlichen Teilmengen $\Phi_0 \subseteq \Phi$ erfüllbar sind (so eine Formelmenge heißt* endlich erfüllbar*). Dann ist Φ erfüllbar.*

Beweis. Nach Vollständigkeit ist Erfüllbarkeit gleichbedeutend mit Konsistenz. Konsistenz hat offenbar die behauptete Eigenschaft: nach Negation beider Seiten ist zu zeigen, dass $\Phi \vdash \bot$ genau dann, wenn es eine endliche Teilmenge $\Phi_0 \subseteq \Phi$ gibt mit $\Phi_0 \vdash \bot$. Das ist aber klar, da ein Beweis von $\bot$ aus Φ nur endlich viele der Annahmen in Φ verwendet. □

4.2 Anwendungen des Kompaktheitssatzes

Kompaktheit ist eine durchaus überraschende Eigenschaft, die zahlreiche ebenfalls überraschende Anwendungen in der Theorie diskreter Strukturen hat. Wir diskutieren im folgenden ein Beispiel aus der Graphfärbungstheorie.

Definition 22. Ein (ungerichteter) Graph mit Knotenmenge V und Kantenmenge E heißt *k-färbbar*, wenn es eine Abbildung $c : V \to \{1, \ldots, k\}$ (*Colouring/Färbung*) gibt, so dass $c(v) \neq c(w)$ für jede Kante $\{v, w\} \in E$ gilt.

Der berühmte Vierfarbensatz besagt beispielsweise, dass jeder planare Graph 4-färbbar ist.

Satz 23. *Ein (möglicherweise unendlicher) Graph ist genau dann k-färbbar, wenn alle seine endlichen Untergraphen k-färbbar sind.*

Beweis. „Nur dann, wenn“ ist trivial; wir zeigen „wenn“.

Sei V die Knotenmenge und E die Kantenmenge des Graphen. Wir führen Atome $A_{v,i}$ für $v \in V$, $i \in \{1, \ldots, k\}$ ein, mit der Lesart „Knoten v hat Farbe i“ (also $c(v) = i$). Wir definieren Formelmengen Φ_1, Φ_2, Φ_3 wie folgt:

$$\Phi_1 = \{\textstyle\bigvee_{i=1}^{k} A_{v,i} \mid v \in V\}$$

– d.h. jeder Knoten hat mindestens eine Farbe.

$$\Phi_2 = \{\neg(A_{v,i} \wedge A_{v,j}) \mid v \in V, i, j \in \{1, \ldots, k\}, i \neq j\}$$

– d.h. kein Knoten hat mehr als eine Farbe.

$$\Phi_3 = \{\neg(A_{v,i} \wedge A_{w,i}) \mid \{v, w\} \in E, i \in \{1, \ldots, k\}\}$$

– d.h. keine zwei adjazenten Knoten haben die gleiche Farbe.

Wir setzen nun $\Phi = \Phi_1 \cup \Phi_2 \cup \Phi_3$. Dann ist Φ endlich erfüllbar: Sei $\Psi \subseteq \Phi$ endliche Teilmenge; dann ist die Menge $V_0 = \{v \mid \exists i.\, A_{v,i} \in \mathrm{At}(\Psi)\}$ endlich. Sei G_0 der von V_0 aufgespannte Untergraph. Nach Annahme ist G_0 k-färbbar; sei c eine entsprechende Färbung. Wir definieren eine Wahrheitsbelegung κ_0 durch

$$\kappa_0(A_{v,i}) = \top \text{ gdw. } c(v) = i$$

für $v \in V_0$ (und beliebig auf anderen Atomen). Dann gilt $\kappa_0 \models \Phi_0$.

Nach dem Kompaktheitssatz ist also Φ erfüllbar; sei $\kappa \models \Phi$. Dann existiert für jedes v genau ein i, so dass $\kappa(A_{v,i}) = \top$; wir erhalten eine k-Färbung c des Graphen, indem wir $c(v) = i$ setzen. □

Sehr ähnlich erhält man z.B. einfache Beweise von Königs Lemma (jeder endlich verzweigende unendliche Graph hat einen unendlichen Pfad) oder der Aussage, dass man, gegeben ein Satz K von quadratischen Kacheln mit gefärbten Kanten, genau dann die $\mathbb{N} \times \mathbb{N}$-Ebene mit Kacheln aus K farblich passend anschließend überdecken kann, wenn dies für jede $n \times n$-Fläche geht.

5 Normalformen und Resolution

Wir entwickeln nunmehr ein Entscheidungsverfahren für die Erfüllbarkeit aussagenlogischer Formeln, das zwar immer noch in schlechten Fällen in exponentieller Zeit läuft (und wegen der NP-Vollständigkeit des SAT-Problems, s. BFS, wird sich das wahrscheinlich, d.h. wenn P≠NP, grundsätzlich nicht vermeiden lassen), aber in der Praxis wesentlich schneller ist als das bisher verwendete Wahrheitstafelverfahren: das sogenannte Resolutionsverfahren. Dieses Verfahren erwartet die Eingabeformel in einem besonderen Format, der sogenannten *konjunktiven Normalform*, die wir in zwei Schritten einführen.

5.1 Negationsnormalform (NNF)

Die ab jetzt betrachteten Normalformen sind dadurch definiert, dass sie eine bestimmte Reihenfolge der logischen Operatoren bei der Traversierung des Syntaxbaums einer Formel von der Wurzel zu den Blättern festlegen. Wir behandeln zunächst nur die Negation, von der wir verlangen, dass sie hierbei zuletzt kommt; da sich aufeinanderfolgende Negationen aufheben, läuft dies darauf hinaus, dass es Negationen nur unmittelbar vor Blättern, also Atomen, geben darf. Mit unserer bisherigen Auswahl an Basisoperatoren (nur $\neg$ und $\wedge$) lässt sich allerdings eine solche Normalform im allgemeinen nicht herstellen; wir betrachten daher ab jetzt auch $\top$ und die Disjunktion $\vee$ als Basisoperationen.

Definition 24 (Negationsnormalform)**.** Eine aus Atomen sowie $\neg, \wedge, \vee, \bot, \top$ gebildete Formel ϕ ist in *Negationsnormalform (NNF)* (oder *eine NNF*), wenn die Negation $\neg$ in ϕ nur direkt vor Atomen vorkommt, d.h. wenn ϕ von der Grammatik

$$\phi, \psi ::= \bot \mid \top \mid A \mid \neg A \mid \phi \wedge \psi \mid \phi \vee \psi \qquad (A \in \mathscr{A})$$

erzeugt wird. Eine NNF χ ist *NNF von* ϕ, wenn $\chi \equiv \phi$.

Beispiel 25 (Terme in NNF)**.** Die Formel $\neg A \vee (B \vee \neg C)$ ist in NNF, $\neg(A \vee \neg B)$ dagegen nicht.

Satz 26. *Jede Formel hat eine NNF.*

Beweis. Auf eine gegebene Formel wende man so lange wie möglich die Äquivalenzumformungen

$$\begin{aligned} \neg\neg\phi &\equiv \phi \\ \neg(\phi \wedge \psi) &\equiv \neg\phi \vee \neg\psi \\ \neg(\phi \vee \psi) &\equiv \neg\phi \wedge \neg\psi \\ \neg\top &\equiv \bot \\ \neg\bot &\equiv \top \end{aligned}$$

an. □

Beispiel 27. Die Formel $\neg(\neg(A \vee \neg B) \wedge C)$ bringen wir nach der im vorigen Beweis angegebenen Methode folgendermaßen in NNF:

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

5.2 Konjunktive Normalformen

Wir wollen nun zusätzlich darauf bestehen, dass bei Traversierung des Syntaxbaums einer Formel von der Wurzel zu einem Blatt stets die Konjunktionen vor den Disjunktionen kommen (und weiterhin zuletzt ggf. die Negation).

Formal definieren wir *Literale*, *Klauseln* und *konjunktive Normalformen (CNFs)* durch die folgende Grammatik:

Literale

$$L ::= A \mid \neg A \qquad A \in \mathscr{A}$$

Klauseln

$$C ::= \bot \mid neC$$
$$neC ::= L \mid L \vee neC$$

CNFs

$$\phi ::= \top \mid \psi$$
$$\psi ::= C \mid C \wedge \psi$$

Eine CNF ist demnach eine Konjunktion von Disjunktionen von Literalen, hat also die allgemeine Form

$$\bigwedge_{i=1}^{n} (\bigvee_{j=1}^{k_i} L_{ij})$$

Meistens, insbesondere zum Zwecke der Repräsentation im Rechner, verwenden wir eine alternative Darstellung von CNFs als Mengen, d.h. wir abstrahieren in Konjunktionen und Disjunktionen von der Reihenfolge und Wiederholungen (unter Verwendung von Assoziativität, Kommutativität und Idempotenz von $\wedge$ und $\vee$):

Klauseln sind endliche Mengen von Literalen, d. h. die Disjunktion $L_1 \vee \cdots \vee L_k$ wird repräsentiert als die Menge $\{L_1, \ldots, L_k\}$. Die leere Klausel repräsentiert $\bot$ und wird als $\square$ notiert.

CNFs sind endliche Mengen von Klauseln, wobei die leere Menge $\top$ repräsentiert: $\top \mathrel{\hat{=}} \varnothing$; $C_1 \wedge \cdots \wedge C_n \mathrel{\hat{=}} \{C_1, \ldots, C_n\}$

Definition 28 (CNF einer Formel)**.** Sei ϕ eine Formel. Eine CNF ϕ' heißt *CNF von* ϕ, wenn $\phi \equiv \phi'$.

Lemma 29. *Jede Formel hat eine CNF.*

Beweis. Sei ϕ eine Formel. Wir können nach Obigem annehmen, dass ϕ bereits in NNF ist. Wir sehen $\wedge$ und $\vee$ als assoziativ und kommutativ an. Wir definieren dann eine CNF $\mathsf{CNF}(\phi)$ von ϕ rekursiv:

$$\begin{aligned} \mathsf{CNF}(\phi \wedge \psi) &= \mathsf{CNF}(\phi) \wedge \mathsf{CNF}(\psi) \\ \mathsf{CNF}((\phi \wedge \psi) \vee \chi) &= \mathsf{CNF}(\phi \vee \chi) \wedge \mathsf{CNF}(\psi \vee \chi) \\ \mathsf{CNF}(\phi) &= \phi \qquad (\phi \text{ Klausel}). \end{aligned}$$

Man beachte, dass immer einer der Fälle anwendbar ist: Der erste Fall ist anwendbar, wenn das oberste Konnektiv eine Konjunktion ist. Wenn das oberste Konnektiv eine Disjunktion ist, die Formel aber auch noch Konjunktionen enthält, ist nach geeigneter Umformung gemäß Assoziativität und Kommutativität der zweite Fall anwendbar. (Was als $\mathsf{CNF}(\phi)$ genau herauskommt, hängt von der jeweils aus eventuell mehreren Möglichkeiten gewählten Umformung ab, d.h. der Algorithmus ist bis zu einem gewissen Grad erst einmal nichtdeterministisch, was uns aber egal ist, so lange wir irgendeine CNF von ϕ erhalten.) Andernfalls, d.h. wenn die Formel keine Konjunktionen enthält, ist der dritte Fall anwendbar, da wir annehmen, dass die Formel bereits in NNF ist.

Die Rekursion terminiert, weil die Argumente in den rekursiven Aufrufen stets kleiner sind als das Argument im gerade definierten Aufruf. Man zeigt durch Indukion über die Größe von ϕ, dass $\phi \equiv \mathsf{CNF}(\phi)$ und dass $\mathsf{CNF}(\phi)$ in der Tat eine CNF ist. Der einzig interessante Teil der Induktion ist der Induktionsschritt für $(\phi \wedge \psi) \vee \chi$, der auf dem Distributivgesetz

$$(\phi \wedge \psi) \vee \chi \equiv (\phi \vee \chi) \wedge (\psi \vee \chi)$$

beruht. □

Das Problem an der CNF ist, dass, wenn $\mathsf{CNF}(\phi)$ n Klauseln hat und $\mathsf{CNF}(\psi)$ k Klauseln hat, dann $\mathsf{CNF}(\phi \vee \psi)$ $n \cdot k$ Klauseln hat, was bei längeren Disjunktionen einen exponentiellen Zuwachs in der Größe bewirkt. Z.B. hat $\mathsf{CNF}((A_1 \wedge B_1) \vee \cdots \vee (A_n \wedge B_n))$ 2^n Klauseln. Durch Einführung zusätzlicher Atome lässt sich eine polynomielle Größe der CNF sicherstellen; dann ist aber eben wegen der zusätzlichen Atome die so gewonnene CNF einer Formel ϕ nur noch *erfüllbarkeitsäquivalent* zu ϕ, d.h. sie ist *erfüllbar* genau dann, wenn ϕ erfüllbar ist, aber i.a. nicht mehr logisch äquivalent zu ϕ.

Beispiel 30.

$$\begin{aligned}
&\mathsf{CNF}((\neg B \wedge C) \vee (\neg A \wedge B)) \\
&= \mathsf{CNF}(\neg B \vee (\neg A \wedge B)) \wedge \mathsf{CNF}(C \vee (\neg A \wedge B)) \\
&= \mathsf{CNF}(\neg B \vee \neg A) \wedge \mathsf{CNF}(\neg B \vee B) \wedge \mathsf{CNF}(C \vee \neg A) \wedge \mathsf{CNF}(C \vee B)) \\
&= (\neg B \vee \neg A) \wedge (\neg B \vee B) \wedge (C \vee \neg A) \wedge (C \vee B)).
\end{aligned}$$

5.3 Resolution

Das Resolutionsverfahren ist ein Algorithmus zur Entscheidung der Erfüllbarkeit einer CNF, hier dargestellt als eine Klauselmenge ϕ. Er beruht auf der *Resolutionsregel*

$$(\text{Res})\ \frac{C_1 \cup \{A\} \qquad C_2 \cup \{\neg A\}}{C_1 \cup C_2}. \tag{1}$$

Diese Regel wird im Resolutionsverfahren auf die Menge der Klauseln in einer CNF angewendet, in dem Sinne, dass, wenn in der Menge alle Prämissen, d.h. die Klauseln über dem Strich, enthalten sind, die Konklusion, also die Klausel $C_1 \cup C_2$, hinzugefügt werden darf; letztere heißt *Resolvente* von $C_1 \cup \{A\}$ und $C_2 \cup \{\neg A\}$.

Die Resolutionsregel ist im folgenden Sinn korrekt:

Lemma 31. *Für Klauseln C_1, C_2 und $A \in \mathscr{A}$ gilt $C_1 \cup \{A\}, C_2 \cup \{\neg A\} \models C_1 \cup C_2$.*

Beweis. Sei $\kappa \models C_1 \cup \{A\}, \kappa \models C_2 \cup \{\neg A\}$. Wir führen eine Fallunterscheidung nach $\kappa(A)$ durch:

$\kappa(A) = \top$: Dann $\kappa \models C_2$, also $\kappa \models C_1 \cup C_2$

$\kappa(A) = \bot$: Dann $\kappa \models C_1$, also $\kappa \models C_1 \cup C_2$ □

Auf der Resolutionsregel basiert der folgende Algorithmus zur Entscheidung der Erfüllbarkeit einer CNF:

Algorithmus 32 (Resolutionsverfahren)**.** Eingabe: CNF ϕ. Ausgabe: „ja“, wenn ϕ erfüllbar, „nein“ sonst.

Verwende ϕ als globale Variable:

1. If $\square \in \phi$ return „nein“.
2. Suche $C_1 \cup \{A\}, C_2 \cup \{\neg A\} \in \phi$, $C_1 \cup C_2 \notin \phi$. Falls keine solchen C_1, C_2 existieren, return „ja“.
3. $\phi := \phi \cup \{C_1 \cup C_2\}$.
4. Gehe zu Schritt 1.

Beispiel 33. 1. $\{D, B, \neg C\}, \{D, C\}, \{\neg D, B\}, \{\neg C, B, \neg A\}, \{C, B, \neg A\}, \{\neg B, \neg A\}, \{\neg B, A\}$

2. $\{A, \neg B\}, \{A, B\}$

Wir beweisen die totale Korrektheit des Algorithmus:

Terminierung: Der Algorithmus arbeitet ausschließlich auf der Menge $\mathrm{At}(\phi)$ der in ϕ vorkommenden Atome, d.h. die Anwendung der Regel führt nie neue Atome ein. In Mengendarstellung gibt es nur endlich viele unterschiedliche Klauseln über $\mathrm{At}(\phi)$, nämlich $4^{|\mathrm{At}(\phi)|}$ (jedes Atom kann sowohl als positives als auch als negatives Literal jeweils vorkommen oder nicht vorkommen). Insbesondere kann also nur endlich oft eine Klausel hinzugefügt werden, die nicht bereits in der aktuellen CNF enthalten ist.

Korrektheit (Antwort „nein“ ist richtig): Per Korrektheit der Resolutionsregel (Lemma 31) folgen während der Ausführung des Algorithmus alle jeweils in ϕ enthaltenen Klauseln aus der Eingabe (diese Eigenschaft ist also eine *Invariante* des Algorithmus). Wenn $\square$, also nach unseren Konventionen $\bot$, eine dieser Klauseln ist, ist somit die Eingabe unerfüllbar.

Vollständigkeit (Antwort „ja“ ist richtig): Wir sagen, dass ϕ *resolutionsabgeschlossen (ra.)* ist, wenn mit $C_1 \cup \{A\}, C_2 \cup \{\neg A\} \in \phi$ stets $C_1 \cup C_2 \in \phi$ gilt. Der Algorithmus bricht also genau dann bei Schritt 2 ab, wenn ϕ ra. ist. Sei also ϕ ra., $\square \notin \phi$. Zu zeigen ist dann, dass ϕ erfüllbar ist. Wir verwenden Induktion über $|\mathrm{At}(\phi)|$:

Induktionsanfang: Ohne Atome kann ϕ als Klauselmenge nur leer sein, da die einzige Klausel ohne Atome, nämlich $\square$, nach Voraussetzung nicht in ϕ enthalten ist. Die leere Klausel repräsentiert nach unseren Konventionen die Formel $\top$ und ist damit natürlich erfüllbar.

Induktionsschritt: Wähle $A \in \mathrm{At}(\phi)$. Sei

$$\phi/A = \{\, D \backslash \{\neg A\} \mid A \notin D \in \phi \,\}$$
$$\phi/\neg A = \{\, D \backslash \{A\} \mid \neg A \notin D \in \phi \,\}.$$

ϕ/A ist logisch äquivalent zu ϕ, wenn wir A annehmen, entsprechend für $\phi/\neg A$ und $\neg A$; formal: $A \models \phi \leftrightarrow \phi/A$ und $\neg A \models \phi \leftrightarrow \phi/\neg A$. Es gilt $\mathrm{At}(\phi/A) \not\ni A \notin \mathrm{At}(\phi/\neg A))$.

Dann ist entweder $\square \notin \phi/A$ oder $\square \notin \phi/\neg A$: sonst $\{\neg A\} \in \phi \ni \{A\}$; da ϕ ra., würde folgen $\square \in \phi$, Widerspruch.

Ohne Einschränkung[3] sei $\square \notin \phi/A$.

Nun ist auch ϕ/A ra.: Sei $E_1 \cup \{B\}, E_2 \cup \{\neg B\} \in \phi/A$. Da dann $B \neq A$, haben die E_i nach Definition von ϕ/A die Form

$$E_1 = C_1 \backslash \{\neg A\}, \quad E_2 = C_2 \backslash \{\neg A\}$$

für geeignete C_1, C_2 mit $C_1 \cup \{B\}, C_2 \cup \{\neg B\} \in \phi$, $A \notin C_1 \cup \{B\}$ und $A \notin C_2 \cup \{\neg B\}$. Da ϕ ra. ist, folgt $C_1 \cup C_2 \in \phi$; da $A \notin C_1 \cup C_2$, folgt $E_1 \cup E_2 = (C_1 \cup C_2) \backslash \{\neg A\} \in \phi/A$.

Da $|At(\phi/A)| < |\,\mathrm{At}(\phi)|$, folgt nach Induktionsvoraussetzung, dass ϕ/A erfüllbar ist, d.h. es existiert κ mit $\kappa \models \phi/A$. Wir behaupten nun, dass $\kappa[A \mapsto \top] \models \phi$, womit dann ϕ wie verlangt erfüllbar ist. Sei also $D \in \phi$; zu zeigen ist $\kappa[A \mapsto \top] \models D$.

Fall 1: $A \in D$ ✓

Fall 2:

$$\begin{aligned} A \notin D &\Rightarrow D \backslash \{\neg A\} \in \phi/A \\ &\Rightarrow \kappa \models D \backslash \{\neg A\} \\ &\Rightarrow \kappa[A \mapsto \top] \models D \backslash \{\neg A\} && (A \notin \mathrm{At}(D \backslash \{\neg A\}), \text{ Lemma 11}) \\ &\Rightarrow \kappa[A \mapsto \top] \models D. \end{aligned}$$

□

6 Prädikatenlogik erster Stufe

Ein *Prädikat* ist eine Eigenschaft, die Individuen haben (oder nicht haben) können, d.h. eine Aussage mit Parametern, für die Individuen einzusetzen sind; Beispiel: $\mathsf{positiv}(n)$ oder $\mathsf{verheiratetMit}(\mathsf{x}, \mathsf{y})$. Diese Parametrisierung unterscheidet Prädikate von den Atomen der Aussagenlogik, die nur schlechthin wahr oder falsch sein können (wie z.B. $\mathsf{esRegnet}$). Logiken, die über Prädikate sprechen, heißen *Prädikatenlogiken*. Sie beinhalten typischerweise auch die Möglichkeit, Aussagen zu *quantifizieren*, etwa dahingehend, ob sie für *alle* möglichen Belegungen von Variablen oder für *mindestens eine* Belegung gelten. Wir beschäftigen uns hier nur mit dem Fall, in dem die betreffenden Variablen für Individuen stehen, also mit der Prädikatenlogik *erster Stufe*. (Es gibt auch Logiken, die Variablen für komplexere Gebilde, z.B. für Mengen von Individuen, zulassen; man spricht dann von Prädikatenlogik *höherer Stufe*, im Beispielfall *zweiter Stufe*.) Gelegentlich wird einfach der Begriff „Prädikatenlogik“ verwendet, womit meist die Prädikatenlogik erster Stufe gemeint ist. Letztere wird auf Englisch mit dem deutlich kürzeren Begriff *first order logic* bezeichnet, abgekürzt *FOL*; der Knappheit halber werden wir oft diese Abkürzung verwenden.

[3] „ohne Einschränkung“ ist gleichbedeutend mit (aber kürzer als) „ohne Beschränkung der Allgemeinheit“. Konkret heißt es hier, dass der Beweis für den Fall $\square \notin \phi/\neg A$ genauso geführt werden kann und deshalb weggelassen wird.

Definition 34 (Syntax der Prädikatenlogik erster Stufe)**.** Die Syntax der Prädikatenlogik hängt ab von einem Vorrat an Symbolen für Konstanten sowie für Prädikate und Funktionen gegebener Stelligkeit. Dieser Symbolvorrat wird oft als die *Sprache* bezeichnet; wir bevorzugen hier zur Vermeidung von Verwechslungen den Ausdruck *Signatur*. Formal besteht eine Signatur $\Sigma = (P_\Sigma, F_\Sigma)$ aus

- einer Menge P_Σ von *Prädikatensymbolen* und
- einer Menge F_Σ von *Funktionssymbolen*.

Wenn keine Gefahr von Missverständnissen besteht, bezeichnen wir Prädikats- und Funktionssymbole auch kurz als *Prädikate* bzw. *Funktionen*. Jedes Funktions- oder Prädikatensymbol s hat eine endliche *Stelligkeit* $\mathrm{ar}(s) \geqslant 0$. Für $s \in P_\Sigma \cup F_\Sigma$ und $\mathrm{ar}(s) = n$ schreiben wir oft $s/n \in \Sigma$, wobei wir typischerweise dadurch die Unterscheidung zwischen Funktionen- und Prädikatensymbolen sicherstellen, dass wir erstere mit Kleinbuchstaben und letztere mit Großbuchstaben bezeichnen. Ein nullstelliges Funktionssymbol $c/0 \in \Sigma$ heißt auch *Konstante*; nullstellige *Prädikaten*symbole entsprechen gerade den Atomen der Aussagenlogik.

Diese Daten bestimmen nun zunächst den Begriff des *Terms*: Wir unterstellen einen Vorrat V an Variablen $x, y \ldots$; Terme $E, D, \ldots$ sind dann gegeben durch die BNF

$$E ::= x \mid f(E_1, \ldots, E_n) \qquad (x \in V, f/n \in \Sigma).$$

Die Syntax von Formeln ist dann gegeben durch die folgende BNF:

$$\phi ::= \quad E = D \mid P(E_1, \ldots, E_n) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \forall x(\phi) \qquad (x \in V, P/n \in \Sigma).$$

Die durch die ersten zwei Klauseln konstruierten Formeln heißen *atomare Formeln*. Der Existenzquantor $\exists$ wird dann definiert durch $\exists x(\phi) := \neg\forall x(\neg\phi)$, außerdem $\bot \equiv \neg\forall x(x = x)$. Weitere aussagenlogische Junktoren wie Implikation, Disjunktion etc. werden wie gewohnt definiert.

Die Sprechweise für $\forall x(\phi)$ ist „für alle x gilt ϕ“, und für $\exists x(\phi)$ „es existiert ein x, so dass ϕ“ oder „es existiert ein x mit ϕ“.

Wie in der Grammatik (ausnahmsweise) angedeutet, setzen wir zur Vermeidung von Uneindeutigkeiten quantifizierte Formeln stets in Klammern; allerdings lassen wir bei Sequenzen von Quantoren gleicher Reichweite alle Klammerpaare bis auf das innerste weg, d.h. wir schreiben z.B. $\forall y \exists z(\phi)$ statt $\forall y(\exists z(\phi))$.

Beispiel 35 (Wissenswertes über Erlangen)**.** Der Satz „Jeder Erlanger mag einen anderen Erlanger“ wird in Logik erster Stufe ausgedrückt als

$$\forall x(E(x) \to \exists y(E(y) \wedge L(x, y)))$$

(mit E für „Erlanger“ und L für „likes“). Der etwas unwahrscheinlicher klingende Satz „Es gibt einen Erlanger, den alle Erlanger mögen“ dagegen wird formalisiert als

$$\exists y(E(y) \wedge \forall x(E(x) \to L(x, y))).$$

Diese Beispiele illustrieren zum einen die typische und meist allein sinnvolle Kombination von $\forall$ mit $\to$ und von $\exists$ mit $\wedge$ gemäß den klassischen *aristotelischen Formen*:

$$\begin{aligned} \text{Alle Ps sind Qs:} \quad & \forall x(P(x) \to Q(x)) \\ \text{Einige Ps sind Qs:} \quad & \exists x(P(x) \wedge Q(x)). \end{aligned}$$

Zum anderen sieht man hier, obwohl wir natürlich noch keinerlei formale Bedeutung von Formeln diskutiert haben, schon anhand der Lesart, dass man Quantoren $\forall x$ und $\exists y$ im allgemeinen keineswegs bedeutungserhaltend vertauschen kann.

Wir sehen die Variable x in $\forall x(\phi)$ als durch den Quantor *gebunden* an, d.h. ihre Lebensdauer ist begrenzt auf die Unterformel ϕ. Wenn $\forall x(\phi)$ Teil einer größeren Formel ist, die außerhalb von $\forall x(\phi)$ die Variable x bereits erwähnt, so wird die äußere Verwendung von x durch den Quantor *verschattet*. Diese Phänomene sind letztlich ähnlich der Verwendung lokaler Variablen in üblichen Programmiersprachen.

Formal definieren wir die *freien Variablen* in einem Term oder einer Formel wie folgt.

Definition 36 (Freie Variable)**.** Die Mengen $FV(E)$ und $FV(\phi)$ der freien Variablen eines Terms E bzw. einer Formel ϕ sind rekursiv definiert durch

$$\begin{aligned}
FV(x) &= \{x\} \\
FV(f(E_1, \ldots, E_n)) &= \bigcup_{i=1}^{n} FV(E_i) \\
FV(E = D) &= FV(E) \cup FV(D) \\
FV(P(E_1, \ldots, E_n)) &= \bigcup_{i=1}^{n} FV(E_i) \\
FV(\neg\phi) &= FV(\phi) \\
FV(\phi \wedge \psi) &= FV(\phi) \cup FV(\psi) \\
FV(\forall x(\phi)) &= FV(\phi)\backslash\{x\}
\end{aligned}$$

Z.B. gilt $FV(P(x, y) \wedge \forall y \exists z(Q(y, w, z))) = \{x, y, w\}$.

Definition 37 (Satz)**.** Eine Formel ϕ heißt *Satz*, wenn $FV(\phi) = \varnothing$, d.h. wenn alle in ϕ vorkommenden Variablen durch Quantoren gebunden sind.

Z.B. ist die Formel $\forall x(E(x) \rightarrow \exists y(E(y) \wedge L(x, y)))$ aus unserem obigen Beispiel ein Satz.

Wir sehen die Namen gebundener Variablen als unwichtig an und betrachten daher zwei Formeln, die sich nur durch die Namen gebundener Variablen unterscheiden (also *α-äquivalent* sind) als im wesentlichen gleich; z.B. sind $\forall x(x = z)$ und $\forall y(y = z)$ (nicht aber $\forall z(z = z)$!) im wesentlichen dieselbe Formel.

Definition 38 (Substitution, Umbenennung)**.** Eine *Substitution* ist eine Abbildung σ, die jeder Variablen x einen Term $\sigma(x)$ zuordnet, so dass die Menge

$$\mathrm{Dom}(\sigma) = \{x \mid \sigma(x) \neq x\}$$

endlich ist, d.h. σ verändert nur eine endliche Anzahl an Variablen. $\mathrm{Dom}(\sigma)$ („Domain“, „Bereich“ von σ) ist die Menge aller Variablen, mit denen σ „etwas tut“, d.h. denen σ einen anderen Term zuordnet. Mit $E\sigma$ bezeichnen wir die Anwendung von σ auf den Term E. Dabei wird jede Variable wie in σ angegeben ersetzt: $x\sigma = \sigma(x)$, $f(E_1, \ldots, E_n)\sigma = f(E_1\sigma, \ldots, E_n\sigma)$. Die Anwendung $\phi\sigma$ einer Substitution σ auf eine Formel ϕ ist wegen eventuell gebundener

Variablen komplizierter:

$$\begin{aligned}(E = D)\sigma &= (E\sigma = D\sigma)\\ P(E_1, \ldots, E_n)\sigma &= P(E_1\sigma, \ldots, E_n\sigma)\\ (\neg\phi)\sigma &= \neg(\phi\sigma)\\ (\phi \wedge \psi)\sigma &= \phi\sigma \wedge \psi\sigma\\ (\forall x(\phi))\sigma &= \forall y(\phi\sigma'),\end{aligned}$$

wobei $\sigma'(x) = y$, $\sigma'(z) = \sigma(z)$ für jedes $z \neq x$, und wobei ferner y so gewählt ist, dass $y \notin FV(\sigma(z))$ für alle $z \in FV(\forall x(\phi))$ („y ist frisch“). (Falls x diese Bedingung bereits erfüllt, kann für y auch x gewählt werden.)

Eine Substitution σ heißt eine *Umbenennung*, wenn $\sigma(x)$ für alle x eine Variable ist.

Die Substitution $[E_1/x_1, \ldots, E_n/x_n]$ ist die Substitution σ mit

$$\sigma(x) = \begin{cases} E_i & \text{wenn } x = x_i \\ x & \text{sonst} \end{cases}$$

Die Identität wird denotiert durch die leere Substitution $[\,]$, also $E\,[\,] \equiv E$. Für Substitutionen σ, τ bezeichnet $\sigma\tau$ die Substitution mit $(\sigma\tau)(x) = (\sigma(x))\tau$, d.h. die Substitution, die entsteht, wenn man zuerst σ und dann τ ausführt. (Achtung: Das ist gerade andersherum als bei der üblichen Notation für Funktionskomposition, in der $f \circ g$ die Funktion bezeichnet, die entsteht, indem man zuerst g und dann f anwendet. Dies ist der Tatsache geschuldet, dass die Anwendung einer Substitution σ auf eine Formel ϕ in Postfixnotation, also als $\phi\sigma$, geschrieben wird.)

Beispiel 39. Mit $\sigma = [add(z, y)/x, suc(x)/y]$ haben wir $\mathrm{Dom}(\sigma) = \{x, y\}$ und z.B.

$$add(add(x, z), add(x, y))\sigma = add(add(add(z, y), z), add(add(z, y), suc(x))).$$

Dieses Beispiel illustriert insbesondere, dass die Ersetzung von x und y durch σ definitionsgemäß *gleichzeitig* stattfindet; wenn man zuerst die Substitution $[add(z, y)/x]$ und anschließend die Substitution $[suc(x)/y]$ anwendet, kommt etwas anderes heraus, da dann auch im für x eingesetzten Term $add(z, y)$ die Variable y durch $suc(x)$ ersetzt wird.

Als weiteres Beispiel betrachten wir die Formel $\phi = \forall x(x \leqslant y)$, wobei $\leqslant$ ein in Infixnotation geschriebenes binäres Prädikat ist; wenn wir uns x, y etc. (*nur für dieses Beispiel!!*) als Zahlen aus einem gegebenen Bereich vorstellen und $\leqslant$ als die übliche kleiner-gleich-Relation interpretieren, ist dies die Aussage, dass y die größte Zahl des gegebenen Zahlbereichs ist. Ferner sei σ die Substitution $[z/x, a(x, y)/y]$, wobei a eine zweistellige Funktion ist. Die Seitenbedingung in der Definition der Anwendung $\phi\sigma$ von σ auf ϕ verbietet in diesem Fall gerade, dass wir den Namen der allquantifizierten Variablen x unverändert lassen, da x nicht frisch im Sinne der Seitenbedingung ist: x kommt frei im Term $a(x, y)$ vor, den σ für die freie Variable y in $\forall x(x \leqslant y)$ substituiert. Wenn wir uns über diese Beschränkung hinwegsetzen, ergibt sich denn auch eine Formel, die offenbar nicht im beabsichtigten Sinne eine Instanz von ϕ ist (formaler wird es gerade noch nicht, da wir noch keine Semantik definiert haben): Mit $\sigma'(x) = x$ und $\sigma'(y) = \sigma(y) = a(x, y)$ erhielten wir hier als Resultat der Substitution

$$\forall x((x \leqslant y)\sigma') = \forall x(x \leqslant a(x, y)),$$

also eine gänzlich unverwandte Aussage, die z.B., wenn wir wir a als Addition interpretieren, besagt, dass y nichtnegativ ist.

Wählen wir dagegen wie vorgeschrieben eine frische Variable, beispielsweise z (das ist wirklich frisch im verlangten Sinn: z kommt zwar im Term $\sigma(x)$ vor, aber x ist nicht frei in ϕ), so erhalten wir mit $\sigma'(x) = z$, $\sigma'(y) = \sigma(y) = a(x, y)$ korrekterweise

$$\phi\sigma = (\forall x(x \leqslant y))\sigma = \forall z((x \leqslant y)\sigma') = \forall z(z \leqslant a(x, y)),$$

also die Aussage, dass $a(x, y)$ die größte Zahl im gegebenen Zahlbereich ist; diese Aussage ist in der erwarteten Form eine Instanz der ursprünglichen Aussage $\forall x(x \leqslant y)$.

6.1 Natürliches Schließen in Prädikatenlogik

Wir erweitern nun das System des natürlichen Schließens um Regeln für die neuen Ausdrucksmittel $\exists$, $\forall$ und $=$. Die Regeln für Gleichheit sind

$$(=I)\ \frac{}{E = E} \qquad (=E)\ \frac{\phi[E/x] \qquad E = D}{\phi[D/x]}.$$

Es überrascht eventuell zunächst, dass das schon genügt; man beweist hiermit aber eben andere erwartete Eigenschaften von Gleichheit, etwa Symmetrie

1	$E = D$	
2	$E = E$	(=I)
3	$D = E$	(=E) 1,2

und Transitivität

1	$E = D$	
2	$D = F$	
3	$E = F$	(=E) 1,2

Die Regeln für $\forall$ formalisieren die erwarteten Schlussweisen: Die $\forall$-Eliminationsregel lautet

$$\forall E\ \frac{\forall x(\phi)}{\phi[E/x]}$$

für beliebig gewählte Terme E, also umgangssprachlich „Wenn ϕ für alle x gilt, dann gilt ϕ auch für ein gegebenes E“.

Die Einführungsregel liest sich umgangssprachlich „Wenn man ϕ für ein beliebiges c zeigen kann, dann gilt ϕ für alle x“, formal

$$(\forall I)\ \frac{\begin{array}{|l} \boxed{c} \\ \hline \vdots \\ \phi[c/x] \end{array}}{\forall x(\phi)}$$

wobei c eine *frische* Konstante ist, d.h. außer in dem betreffenden Unterbeweis nirgendwo vorkommt (dies wird durch die Box um das c angedeutet). Durch diese Einschränkung an c wird

der informelle Ausdruck „beliebig“ eingefangen: c kann dann insbesondere nicht in irgendwelchen gerade aktiven Annahmen vorkommen, d.h. der Unterbeweis in der Prämisse zeigt $\phi[c/x]$, ohne über c irgendwelche unzulässigen Voraussetzungen zu machen.

Die Einführungsregel für $\exists$ ist ebenso intuitiv:

$$(\exists \mathrm{I})\ \frac{\phi[E/x]}{\exists x(\phi)}$$

In Worten: Wenn ϕ für einen Term E gilt, dann gibt es ein x, für das ϕ gilt – nämlich eben gerade E.

Etwas gewöhnungsbedürftiger ist die Eliminationsregel für $\exists$:

$$(\exists \mathrm{E})\ \frac{\exists x(\phi) \qquad \begin{array}{|l} \boxed{c}\ \phi[c/x] \\ \hline \vdots \\ \psi \end{array}}{\psi}$$

wiederum mit der Seitenbedingung, dass c frisch ist; dies bedeutet hier insbesondere, dass c nicht in ψ vorkommen darf. In Worten: Wenn es ein x gibt, für das ϕ gilt, und wenn ich für beliebiges c eine Aussage ψ daraus folgern kann, dass ϕ für c gilt, dann gilt ψ. In natürlichsprachlichen Beweisen entspricht dies folgendem Vorgehen: Man hat Existenz eines Elements mit einer gegebenen Eigenschaft ϕ, nimmt sich also ein solches Element c her und argumentiert damit weiter.

Herleitung der Regen für $\exists$ Als erste Anwendung der Regeln für $\forall$ können wir die Regeln für $\exists$ bei Kodierung von $\exists$ per $\exists x(\phi) \equiv \neg\forall x(\neg\phi)$ herleiten:

Beweis von ($\exists I$)

1	$\phi[c/x]$	
2	$\quad \forall x(\neg\phi)$	
3	$\quad \neg\phi[c/x]$	($\forall$E, 2)
4	$\quad \bot$	($\bot$I 1, 3)
5	$\neg\forall x(\neg\phi)$	($\neg$I 1-4)

Beweis von ($\exists E$)

1	$\neg\forall x(\neg\phi)$	
2	$\quad\neg\psi$	
3	$\quad\quad\boxed{c}$	
4	$\quad\quad\phi[c/x]$	
5	$\quad\quad\vdots$	
6	$\quad\quad\psi$	(Unterbeweis aus Prämisse)
7	$\quad\quad\bot$	($\bot$I) 2,6
8	$\quad\neg\phi[c/x]$	($\neg$I) 4–7
9	$\quad\forall x(\neg\phi)$	($\forall$I) 3–8
10	$\quad\bot$	($\bot$I) 1,9
11	$\neg\neg\psi$	($\neg$I) 2–10
12	ψ	($\neg$E) 11

Man beachte im obigen Beweis insbesondere, dass c bei seiner Einführung in Zeile 3 in der Tat frisch ist, da wir bei der Formulierung von ($\exists E$) darauf bestanden haben, dass c auch frisch für ψ ist.

Wie schon im Falle der Aussagenlogik schreiben wir $\Phi \vdash \psi$, wenn ein Beweis von ψ aus Annahmen in Φ existiert.

Weitere Beispiele für Herleitungen, jetzt im vollen Kalkül (also mit den Regeln für $\exists$):

Folgerung der Existenz aus der Universalität $\forall x(P(x)) \vdash \exists x(P(x))$

1	$\forall x(P(x))$	
2	$P(c)$	($\forall$E, 1)
3	$\exists x(P(x))$	($\exists$I, 2)

Transitivität der Subsumption

1	$\forall x(P(x) \to Q(x))$	
2	$\forall z(Q(z) \to R(z))$	
3	$\boxed{\text{d}}$	
4	$P(d)$	
5	$P(d) \to Q(d)$	($\forall$E) 1
6	$Q(d)$	($\to$E) 3, 4
7	$Q(d) \to R(d)$	($\forall$E) 2
8	$R(d)$	($\to$E) 5, 6
9	$P(d) \to R(d)$	($\to$I) 4–8
10	$\forall x(P(x) \to R(x))$	($\forall$I) 3–9

(Nicht-)-Vertauschung von $\forall$ und $\exists$ Wir können aus $\exists x \forall y(P(x, y))$ die Formel $\forall y \exists x(P(x, y))$ folgern. Die Umkehrung gilt nicht, wie man sich anhand der bisher nur informell beschriebenen Bedeutung der Formeln klarmacht; wir werden dies später auch formal zeigen.

1	$\exists x \forall y(P(x, y))$	
2	$\boxed{\text{c}}$ $\forall y(P(c, y))$	
3	$\boxed{\text{d}}$	
4	$P(c, d)$	($\forall$E) 2
5	$\exists x(P(x, d))$	($\exists$I) 4
6	$\forall y \exists x(P(x, y))$	($\forall$I) 3–5
7	$\forall y \exists x(P(x, y))$	($\exists$E) 2–6

Noch mehr Beispiele:

$$\exists x(\phi \vee \psi) \equiv \exists x(\phi) \vee \exists x(\psi)$$
$$\phi \wedge \exists x(\psi) \equiv \exists x(\phi \wedge \psi) \qquad (x \notin FV(\phi))$$
$$\phi \vee \exists x(\psi) \equiv \exists x(\phi \vee \psi) \qquad (x \notin FV(\phi))$$

6.2 Semantik

Wir legen nunmehr formal die *Bedeutung* prädikatenlogischer Formeln fest. Ein verbreitetes Prinzip der Logik ist, dass Bedeutung hierbei die Gestalt eines Begriffs von *Erfülltheit* von Formeln relativ zu einem *Modell* annimmt, d.h. die Semantik ist eine binäre Relation $\models$ zwischen Modellen und Formeln. Im Falle der Aussagenlogik war ein Modell schlicht und einfach eine

Wahrheitsbelegung κ. Abhängig von einer gegebenen Signatur Σ definieren wir nun den Begriff des Σ-*Modells* (weitere geläufige Begriffe: Σ-Struktur, Σ-Algebra, FO-Modell, FO-Struktur, Interpretation). Ein solches Modell muss offenbar hinreichend Struktur für die Interpretation von Funktionen- und Prädikatensymbolen zur Verfügung stellen.

Definition 40 (Σ-Modell)**.** Ein Σ-*Modell* $\mathfrak{M}$ besteht aus

- einer nichtleeren (!) Menge M (dem *Universum*, *(Grund)bereich* oder *Träger*);
- einer *Interpretation* in $\mathfrak{M}$ für jedes n-stellige Funktionssymbol $f/n \in \Sigma$, gegeben durch eine Funktion $\mathfrak{M}[\![f]\!] : M^n \to M$
- einer *Interpretation* in $\mathfrak{M}$ für jedes n-stelliges Prädikatensymbol $P/n \in \Sigma$, gegeben durch eine Teilmenge $\mathfrak{M}[\![P]\!] \subseteq M^n$.

Eine *Umgebung* η (in $\mathfrak{M}$) ist eine Abbildung $\eta : V \to M$, ordnet also jeder Variablen $v \in V$ einen Wert $\eta(v)$ im Universum M zu.

Da Konstanten einfach nullstellige Funktionssymbole sind, werden sie durch Abbildungen $M^0 \to M$ interpretiert. Da es genau ein leeres Tupel () gibt, d.h. $M^0 = \{()\}$, läuft dies einfach auf die Auswahl eines Elements von M hinaus; wir werden daher im folgenden so tun, als würden Konstanten einfach durch Elemente von M interpretiert, wenn dies die Schreibweise vereinfacht.

Definition 41 (Interpretation von Termen, Erfülltheit)**.** Gegeben ein Modell $\mathfrak{M}$ und eine Umgebung η definieren wir die *Interpretation*

$$\mathfrak{M}[\![E]\!]\eta \in M$$

eines Terms E rekursiv durch

$$\begin{aligned}\mathfrak{M}[\![x]\!]\eta &= \eta(x)\\ \mathfrak{M}[\![f(E_1,\ldots,E_n)]\!]\eta &= \mathfrak{M}[\![f]\!]\,(\mathfrak{M}[\![E_1]\!]\eta,\ldots,\mathfrak{M}[\![E_n]\!]\eta)\end{aligned}$$

Die *Erfülltheit* einer Formel ϕ durch ein Modell $\mathfrak{M}$ und eine Umgebung η, geschrieben $\mathfrak{M},\eta \models \phi$ (sprich „$\mathfrak{M},\eta$ erfüllt ϕ“), ist dann rekursiv definiert durch

$$\begin{aligned}\mathfrak{M},\eta \models (E = D) &\iff \mathfrak{M}[\![E]\!]\eta = \mathfrak{M}[\![D]\!]\eta\\ \mathfrak{M},\eta \models P(E_1,\ldots,E_n) &\iff (\mathfrak{M}[\![E_1]\!]\eta,\ldots,\mathfrak{M}[\![E_n]\!]\eta) \in \mathfrak{M}[\![P]\!]\\ \mathfrak{M},\eta \models \forall x(\phi) &\iff \text{für alle } m \in M \text{ gilt } \mathfrak{M},\eta[x \mapsto m] \models \phi\end{aligned}$$

und durch die erwarteten Klauseln für die booleschen Fälle. Dabei bezeichnet $\eta[x \mapsto m]$ die Umgebung mit

$$\eta[x \mapsto m](y) = \begin{cases} m & (y = x)\\ \eta(y) & (\text{sonst}).\end{cases}$$

Beispiel 42. Wir betrachten $\Sigma = \{z/0, s/1, O/1\}$, also eine Signatur, die aus einer Konstante z (*zero*), einem einstelligen Funktionssymbol s (*successor*) und einem einstelligen Prädikat O (*odd*) besteht, und suchen nach Σ-Modellen, die den Satz

$$\phi = (\neg O(z) \wedge \forall x(O(s(x)) \leftrightarrow \neg O(x)))$$

erfüllen. (Hierfür spielt die Umgebung offenbar zunächst keine Rolle; wir beweisen dies weiter unten auch formal.)

- $M = \mathbb{N}$, $\mathfrak{M}[\![z]\!] = 0$, $\mathfrak{M}[\![s]\!](n) = n + 1$, $\mathfrak{M}[\![O]\!] = \{2n + 1 \mid n \in M\}$.
- $M = \{0, \ldots, 2n - 1\}$, $\mathfrak{M}[\![z]\!] = 0$, $\mathfrak{M}[\![s]\!](x) = x + 1$ für $x < 2n - 1$, $\mathfrak{M}[\![s]\!](2n - 1) = 0$, $\mathfrak{M}[\![O]\!] = \{2i - 1 \mid 1 \leqslant i \leqslant n\}$.
- $M = \mathbb{N} \times \mathbb{N}$, $\mathfrak{M}[\![z]\!] = (0, 0)$, $\mathfrak{M}[\![s]\!](n, k) = (n + 1, k)$, $\mathfrak{M}[\![O]\!] = \{(n, k) \mid n + k \text{ ungerade}\}$.

Nun wollen wir zeigen, dass die freien Variablen das tun, was wir haben wollten, als wir sie definiert haben, dass nämlich eine Formel höchstens von den in ihr vorkommenden freien Variablen abhängt. („Höchstens“, weil es z.B. auch Tautologien gibt, die von ihren freien Variablen nicht abhängen, wie etwa $\forall x(x = y \vee \neg(x = y))$.)

Lemma 43. *Sei $\eta_1(x) = \eta_2(x)$ für alle $x \in FV(\phi)$; dann gilt $\mathfrak{M}, \eta_1 \models \phi \iff \mathfrak{M}, \eta_2 \models \phi$.*

Beweis. Da in Formeln Terme vorkommen, müssen wir zunächst eine entsprechende Aussage für Terme beweisen, nämlich, dass für alle Terme E und alle Umgebungen η_1, η_2, wenn $\eta_1(x) = \eta_2(x)$ für alle $x \in FV(E)$ gilt, dann

$$\mathfrak{M}[\![E]\!]\eta_1 = \mathfrak{M}[\![E]\!]\eta_2. \tag{2}$$

Der Beweis per Induktion über E wird dem Leser überlassen.

Wir beweisen nun die Aussage des Lemmas durch Induktion über ϕ: Atomare Formeln werden per (2) abgehandelt, z.B. $\mathfrak{M}, \eta_1 \models E = D \iff \mathfrak{M}[\![E]\!]\eta_1 = \mathfrak{M}[\![D]\!]\eta_1 \overset{(2)}{\iff} \mathfrak{M}[\![E]\!]\eta_2 = \mathfrak{M}[\![D]\!]\eta_2 \iff \mathfrak{M}, \eta_2 \models E = D$. Die Booleschen Fälle sind trivial. Es bleibt der Allquantor:

$$\begin{aligned} \mathfrak{M}, \eta_1 \models \forall x(\psi) &\iff \text{für alle } m \in M \text{ gilt } \mathfrak{M}, \eta_1[x \mapsto m] \models \psi \\ &\overset{\text{IV}}{\iff} \text{für alle } m \in M \text{ gilt } \mathfrak{M}, \eta_2[x \mapsto m] \models \psi \\ &\iff \mathfrak{M}, \eta_2 \models \forall x(\psi) \end{aligned}$$

Zur Anwendung der Induktionsvoraussetzung brauchen wir hierbei, dass

$$\eta_1[x \mapsto m](y) = \eta_2[x \mapsto m](y) \text{ für alle } y \in FV(\psi) \subseteq FV(\forall x(\psi)) \cup \{x\}.$$

Dies zeigt man durch Fallunterscheidung über y: Wenn $y \in FV(\forall x(\psi))$, dann gilt $\eta_1[x \mapsto m](y) = \eta_1(y)$, da dann $y \neq x$, entsprechend für η_2, und $\eta_1(y) = \eta_2(y)$ gilt nach Voraussetzung. Falls $y = x$, so gilt $\eta_1[x \mapsto m](y) = m = \eta_2[x \mapsto m](y)$. □

Damit ist für einen *Satz* ϕ (der ja keine freien Variablen hat) $\mathfrak{M}, \eta \models \phi$ von η unabhängig; wir schreiben in diesem Fall kurz $\mathfrak{M} \models \phi$ für $\mathfrak{M}, \eta \models \phi$.

Analog wie schon für aussagenlogische Formeln schreiben wir für eine Menge Φ von Formeln $\mathfrak{M}, \eta \models \Phi$, wenn $\mathfrak{M}, \eta \models \phi$ für alle $\phi \in \Phi$. Logische Konsequenz, Erfüllbarkeit, Gültigkeit und logische Äquivalenz werden analog wie bisher definiert, bis auf Ersetzung der Wahrheitsbelegung κ durch $\mathfrak{M}, \eta$. Z.B. gilt per Definition $\Phi \models \psi$ (ψ ist logische Folgerung aus Φ) dann, wenn aus $\mathfrak{M}, \eta \models \Phi$ stets $\mathfrak{M}, \eta \models \psi$ folgt. Wie bisher gilt $\Phi \models \psi$ genau dann, wenn $\Phi \cup \{\neg\psi\}$ unerfüllbar ist.

Beispiel 44 (Logische Konsequenz)**.** $\exists x(\forall y(L(y, x))) \models \forall y(\exists x(L(y, x)))$

Beweis. Aus der Annahme folgt, dass $m \in M$ existiert mit

$$\mathfrak{M}, [x \mapsto m] \models \forall y(L(y, x)). \tag{3}$$

Sei $n \in M$; zu zeigen ist $\mathfrak{M}, [y \mapsto n] \models \exists x(L(y, x))$. Wegen (3) gilt $\mathfrak{M}, [y \mapsto n, x \mapsto m] \models L(y, x)$, woraus die Behauptung folgt. □

Andererseits gilt $\forall y(\exists x(L(y,x))) \not\models \exists x(\forall y(L(y,x)))$: Ein Gegenmodell ist z.B. $M = \{0,1\}$, $\mathfrak{M}[\![L]\!] = \{(1,0),(0,1)\}$.

In Beispielen wie dem obigen ist eine Sichtweise hilfreich, in der man die Erfülltheit einer Formel über das Gewinnen eines Spiels zwischen zwei Spielern, *Eloise* ($\exists$) und *Abaelard* ($\forall$), definiert. Eloise versucht zu zeigen, dass eine Formel erfüllt ist, und Abaelard, dass sie nicht erfüllt ist. Spielpositionen sind Paare (η, ϕ), wobei η eine Umgebung in $\mathfrak{M}$ und ϕ eine Formel in NNF (also mit Negation nur vor atomaren Formeln; Existenz einer NNF für jede Formel wird weiter unten behandelt) ist; das Spiel ist beendet, wenn eine atomare Formel oder deren Negation erreicht ist (für die man dann unmittelbar im Modell nachsehen kann, wer gewonnen hat). Regeln des Spiels sind z.B.

- $\phi \wedge \psi$: $\forall$ ist dran und wählt ϕ oder ψ als neue Formel
- $\phi \vee \psi$: $\exists$ ist dran und wählt ϕ oder ψ als neue Formel
- $\forall x(\phi)$: $\forall$ ist dran und wählt ein Element des Grundbereichs als neuen Wert für x (in η)
- $\exists x(\phi)$: $\exists$ ist dran und wählt ein Element des Grundbereichs als neuen Wert für x

Atomare Formeln und deren Negationen sind Gewinnpositionen für einen der Spieler, je nachdem, ob die Formel eben gerade gilt oder nicht; z.B. gewinnt Eloise die Position $(\eta, L(x,y))$, wenn $\mathfrak{M}, \eta \models L(x,y)$ (d.h. wenn $(\eta(x), \eta(y)) \in \mathfrak{M}[\![L]\!]$). Der entscheidende Unterschied zwischen $\forall y \exists x(L(y,x))$ und $\exists x \forall y((L(y,x)))$ ist in dieser Sichtweise, dass im Erfülltheitsspiel für $\forall y(\exists x(L(y,x)))$ Abaelard den ersten Zug macht, in dem für $\exists x(\forall y(L(y,x)))$ dagegen Eloise.

→ Welche Spielregeln müsste man für nichtatomare Negation einführen?

Beispiel 45 (Logische Äquivalenz)**.**

$$\neg\forall x(\phi) \equiv \exists x(\neg\phi) \qquad \neg\exists x(\phi) \equiv \forall x(\neg\phi)$$

Mittels der Äquivalenzen im Beispiel lässt sich die Transformation in Negationsnormalform (NNF) auf prädikatenlogische Formeln erweitern. Dazu müssen natürlich sowohl $\forall$ als auch $\exists$ als vollberechtigte Bestandteile der Grammatik von Formeln angesehen werden (ebenso wie schon im aussagenlogischen Fall sowohl $\wedge$ als auch $\vee$); eine NNF ist dann eine Formel in diesem Sinne, die Negation nur direkt vor atomaren Formeln (also Gleichungen und Prädikatenanwendungen) enthält. Wir werden später noch weitergehende Normalformresultate behandeln.

Leicht zu zeigen, aber zentral ist

Lemma 46 (Substitutionslemma)**.** *Es gilt*

$$\mathfrak{M}, \eta \models \phi\sigma \iff \mathfrak{M}, \eta_\sigma \models \phi,$$

wobei $\eta_\sigma(x) = \mathfrak{M}[\![\sigma(x)]\!]\eta$ *für* $x \in V$*.*

(Man beachte, dass $\eta_\sigma(x) = \eta(x)$ für $x \notin \mathrm{Dom}(\sigma)$: dann haben wir nämlich $\mathfrak{M}[\![\sigma(x)]\!]\eta = \mathfrak{M}[\![x]\!]\eta = \eta(x)$.)

Beweis. Induktion über ϕ. Die benötigte Variante der Aussage für Terme ist

$$\mathfrak{M}[\![E\sigma]\!]\eta = \mathfrak{M}[\![E]\!]\eta_\sigma.$$

□

Wir halten abschließend fest, dass unser formales Deduktionssystem zur gerade definierten Semantik passt:

Satz 47 (Korrektheit)**.** *Wenn* $\Phi \vdash \psi$*, dann auch* $\Phi \models \psi$*.*

Beweis. Wie schon im Fall der Aussagenlogik führen wir den Beweis über die Länge der Herleitung von ψ aus Φ, mit Fallunterscheidung über die zuletzt angewandte Regel und Verallgemeinerung der Induktionsbehauptung auf in Unterbeweisen aus lokalen Annahmen hergeleitete Aussagen. Die Regeln für Gleichheit sind einfach; die Regeln für $\exists$ sind aus denen für $\forall$ hergeleitet, so dass wir sie hier ignorieren können. Wir behandeln die Eliminationsregel für $\forall$, um die Verwendung des Substitutionslemmas zu illustrieren: Wenn im letzten Beweisschritt bei lokalen Annahmen Φ' die Formel $\psi[E/x]$ aus $\forall x(\psi)$ hergeleitet wird, dann gilt per Induktionsannahme $\Phi' \models \forall x(\psi)$. Sei nun $\mathfrak{M}, \eta \models \Phi'$; dann gilt nach Voraussetzung $\mathfrak{M}, \eta \models \forall x(\psi)$. Zu zeigen ist $\mathfrak{M}, \eta \models \psi[E/x]$. Dies ist per Substitutionslemma äquivalent zu $\mathfrak{M}, \eta[x \mapsto \mathfrak{M}[\![E]\!]\eta] \models \psi$, was aber unmittelbar nach Definition der Semantik aus $\mathfrak{M}, \eta \models \forall x(\psi)$ folgt. □

7 Unifikation

Unifikation bezeichnet das Problem, ob und wie zwei gegebene Terme durch Substitution strukturell gleich gemacht, sprich *unifiziert* werden können. Mit anderen Worten ist Unifikation Gleichungslösen mit rein syntaktischen Mitteln: Wir können z.B. eine Lösung für die Gleichung $f(g(x)) = f(y)$ angeben, ohne zu wissen, was f und g für Abbildungen sind, nämlich $y = g(x)$. Unifikation spielt eine zentrale Rolle in der logischen Programmierung und in automatischen Deduktionsalgorithmen, z.B. in der prädikatenlogischen Resolution (siehe Kapitel 9).

7.1 Problemstellung

Definition 48 (Gleichung, Unifikatoren, Allgemeinheitsvergleich)**.** Eine *Gleichung* $E \doteq D$ ist ein Paar (E, D) von Termen. Ein *Gleichungssystem* ist eine Menge S von Gleichungen; die freien Variablen von S sind die der in S vorkommenden Gleichungen, also $FV(S) = \bigcup_{(E \doteq D) \in S} FV(E) \cup FV(D)$. Eine Substitution σ ist ein *Unifikator* von $E \doteq D$, wenn $E\sigma = D\sigma$, wobei wir (wie auch bisher schon) mit dem undekorierten Gleichheitszeichen '=' *syntaktische* Gleichheit bezeichnen, d.h. mit der voranstehenden Gleichung meinen wir, dass $E\sigma$ und $D\sigma$ wörtlich identische Terme sind. Ein *Unifikator* eines Gleichungssystems S ist eine Substitution, die Unifikator aller Gleichungen in S ist. Das System S ist *unifizierbar*, wenn es einen Unifikator hat. Wir bezeichnen mit

$$\mathsf{Unif}(S) = \{\sigma \mid \sigma \text{ ist Unifikator von } S\}$$

die Menge aller Unifikatoren von S. Eine Substitution σ_1 ist *allgemeiner als* σ_2, und σ_2 heißt dann umgekehrt eine *Spezialisierung von* σ_1, wenn eine Substitution τ existiert mit $\sigma_1\tau = \sigma_2$, wenn also σ_2 durch Einsetzen aus σ_1 hervorgeht.

Ein Unifikator σ von S ist *allgemeinster Unifikator* (mgu, für *most general unifier*) *von* S ($\sigma = mgu(S)$), wenn σ allgemeiner als jeder Unifikator von S ist.

Beispiel 49. Zur Gleichung

$$add(suc(x), y) \doteq add(y, suc(z))$$

haben wir z.B. den Unifikator $\sigma = [suc(x)/y, x/z]$. Man kann sich überzeugen, dass dies sogar ein mgu ist. In jedem Fall ist z.B. $\sigma' = [suc(suc(z))/y, suc(z)/z, suc(z)/x]$ ein weiterer Unifikator, und σ ist allgemeiner als σ': wir haben

$$\sigma' = \sigma[suc(z)/x].$$

Unifikatoren sind abgeschlossen unter Spezialisierung, d.h. eine Spezialisierung eines Unifikators ist selbst wieder ein Unifikator:

$$\sigma \in \mathsf{Unif}(S) \Rightarrow \sigma\tau \in \mathsf{Unif}(S).$$

Es folgt, dass wir *alle* Unifikatoren eines Gleichungssystems kennen, sobald wir einen mgu gefunden haben; dann sind nämlich die Unifikatoren gerade die Spezialisierungen des mgu:

Lemma 50. *Wenn $\sigma = mgu(S)$, dann gilt $\mathsf{Unif}(S) = \{\sigma\tau \mid \tau \text{ Substitution}\}$.*

Der mgu ist im allgemeinen nicht eindeutig bestimmt; z.B. hat die Gleichung $f(x) = f(g(y))$ sowohl $[g(y)/x]$ als auch $[g(z)/x, z/y]$ als mgu. Es gilt aber:

Lemma 51 (Eindeutigkeit des mgu)**.** *Der mgu ist eindeutig bis auf injektive Umbenennung von Variablen.*

Beweis. Seien σ, σ' mgu eines Gleichungssystems S. Per Definition existieren dann Substitutionen τ, τ' mit

$$\sigma' = \sigma\tau \quad \sigma = \sigma'\tau',$$

also

$$\sigma = \sigma\tau\tau'$$

– d.h. τ' macht τ rückgängig (genauer gesagt gilt dies auf Variablen, die in Termen der Form $\sigma(x)$ mit $x \in FV(S)$ vorkommen, und nur die interessieren uns). Da man durch Substitution nie aus einem komplexen Term wieder eine Variable machen kann, folgt damit, dass τ eine Umbenennung ist, und da man die Identifikation zweier Variablen nicht wieder rückgängig machen kann, folgt, dass τ injektiv ist. □

Wir beweisen nun die *Existenz* eines mgu durch Angabe eines Algorithmus:

7.2 Unifikationsalgorithmus von Martelli/Montanari

Folgende Definition fängt einen Begriff von *Lösung* eines Gleichungssystems formal ein, den wir im Grunde seit der Mittelstufe kennen:

Definition 52 (Gelöstes Gleichungssystem)**.** Ein Gleichungssystem S heißt *gelöst*, wenn jede Gleichung in S von der Form $x \doteq E$ ist, wobei x nur dieses eine Mal in S vorkommt (formal: $x \notin FV(E) \cup FV(S \backslash \{x \doteq E\})$). Zwei Gleichungssysteme S_1, S_2 sind *äquivalent*, wenn $\mathsf{Unif}(S_1) = \mathsf{Unif}(S_2)$. Ein zu S äquivalentes gelöstes Gleichungssystem heißt *gelöste Form* von S.

Wenn wir ein Gleichungssystem in diesem Sinne gelöst haben, kennen wir seinen mgu:

Lemma 53. *Wenn $S' = \{x_1 \doteq E_1, \ldots, x_n \doteq E_n\}$ eine gelöste Form von S ist, dann gilt $[E_1/x_1, \ldots, E_n/x_n] = mgu(S)$.*

Beweis. Da S' gelöst ist, ist $\sigma = [E_1/x_1, \ldots, E_n/x_n]$ Unifikator von S' und damit auch von S. Wenn σ' Unifikator von S ist, dann auch von S'; es gilt also $\sigma'(x_i) = E_i\sigma' = \sigma(x_i)\sigma'$ für alle i. Wenn $y \notin \{x_1, \ldots, x_n\}$, dann gilt $\sigma(y)\sigma' = y\sigma' = \sigma'(y)$. Insgesamt gilt also $\sigma' = \sigma\sigma'$, d.h. σ ist allgemeiner als σ'. □

Der Algorithmus (ursprünglich Herbrand, später Martelli/Montanari) besteht nun in erschöpfender Anwendung der folgenden Umformungsregeln:

(delete):
$S \cup \{x \doteq x\} \quad \to S$

(decomp):
$S \cup \{f(E_1, \ldots, E_n) \doteq f(D_1, \ldots, D_n)\} \quad \to S \cup \{E_1 \doteq D_1, \ldots, E_n \doteq D_n\}$

(conflict) :
$S \cup \{f(E_1, \ldots, E_n) \doteq g(D_1, \ldots, D_k)\} \quad \to \bot \qquad (\text{für } f \neq g)$

(orient) :
$S \cup \{E \doteq x\} \quad \to S \cup \{x \doteq E\} \qquad (E \text{ keine Variable})$

(occurs)/(elim) :

$$S \cup \{x \doteq E\} \quad \to \begin{cases} \bot & (x \in FV(E), x \neq E) \\ S[E/x] \cup \{x \doteq E\} & (x \notin FV(E), x \in FV(S) \end{cases}$$

Wenn $\bot$ erreicht wird, gibt der Algorithmus „nicht unifizierbar“ aus; ansonsten berechnet er, wie wir noch zeigen, eine gelöste Form und damit einen mgu von S.

Beispiele:

- $f(x, g(y)) \doteq f(g(z), z)$
- $f(x, g(x), h(y)) \doteq f(k(y), g(z), z)$
- $f(x, g(x)) \doteq f(z, z)$

Wir zeigen zunächst Terminierung. Dazu erinnern wir an den Begriff des *Terminationsmaßes*: Wir ordnen jedem Zustand (hier: jedem Gleichungssystem S so wie aktuell umgeformt) ein Maß $\nu(S)$ zu, so dass $\nu(S)$ in jedem Schritt abnimmt. Wenn wir als Wertebereich des Maßes eine geordnete Menge wählen, in der es keine unendlichen absteigenden Ketten

$$x_0 > x_1 > x_2 > \ldots$$

gibt, beweist das, dass immer nur endlich viele Zustände durchlaufen werden, bevor der Prozess stoppt. Ein Beispiel einer solchen Menge ist die Menge $\mathbb{N}^n$ aller n-Tupel von natürlichen Zahlen in *lexikographischer Ordnung* $\preceq$. Diese definieren wir durch Rekursion über n wie folgt: Wir setzen

- $() \preceq ()$
- für $n > 0$: $(a_1, \ldots, a_n) \preceq (b_1, \ldots, b_n)$ genau dann, wenn $a_1 \leqslant b_1$ und, falls $a_1 = b_1$, außerdem $(a_2, \ldots, a_n) \preceq (b_2, \ldots, b_n)$.

Es gilt nun in der Tat

Satz 54. *Die lexikographische Ordnung ist eine* Wohlordnung*, d.h. es gibt keine unendliche absteigende Kette*

$$\vec{a}^0 \succ \vec{a}^1 \succ \vec{a}^2 \succ \dots \tag{4}$$

Beweis. Induktion über n, mit trivialem Induktionsanfang. Wir nehmen zwecks Herleitung eines Widerspruchs an, wir hätten eine unendliche absteigende Kette (4), mit $\vec{a}^i = (a_1^i, \dots, a_n^i)$. Dann gilt

$$a_1^0 \geqslant a_1^1 \geqslant a_1^2 \geqslant \dots .$$

Diese Kette wird, da es in $\mathbb{N}$ keine unendlichen absteigenden Ketten gibt, irgendwann *stationär*, d.h. es existiert k mit $a_1^i = a_1^k$ für alle $i \geqslant k$. Dann gilt aber per Definition

$$(a_2^k, \dots, a_n^k) \succ (a_2^{k+1}, \dots, a_n^{k+1}) \succ (a_2^{k+2}, \dots, a_n^{k+2}) \succ \dots ,$$

im Widerspruch zur Induktionsvoraussetzung, die besagt, dass es in $\mathbb{N}^{n-1}$ keine unendlichen absteigenden Ketten gibt. □

Im vorliegenden Fall verwenden wir als Terminationsmaß das Tripel $(n_0 - n_1, n_2, n_3)$ in lexikographischer Ordnung, wobei

- n_0 = Anzahl der *ursprünglich* im System vorkommenden Variablen
- n_1 = Anzahl der Variablen x, die *erledigt* sind, d.h. als linke Seite einer Gleichung vorkommen und sonst nirgends.
- n_2 = Anzahl Symbole in aktuellen System
- n_3 = Anzahl Gleichungen der Form $E \doteq x$ im aktuellen System mit E keine Variable.

Dieses Maß nimmt offenbar in jedem Umformungsschritt ab, so dass der Algorithmus terminiert. Die erste Komponente $n_0 - n_1$ ist nichtnegativ, da der Algorithmus nie neue Variablen hinzufügt, also alle erledigten Variablen von vornherein im System vorgekommen sind.

Wenn der Algorithmus ohne Erreichen von $\bot$ terminiert, also keine Regel mehr anwendbar ist, dann ist das so erreichte System S gelöst: wenn $E \doteq D$ eine Gleichung in S ist, dann kann E nicht zusammengesetzt sein (sonst greift eine der Regeln *(decomp)*, *(conflict)* oder *(orient)*). Also ist E eine Variable x. Wegen der Regeln *(delete)*, *(occurs)* und *(elim)* kommt dann x weder in D noch in den restlichen Gleichungen vor.

Es bleibt zu zeigen, dass die Regeln das gegebene Gleichungssystem stets in ein äquivalentes transformieren, wobei $\bot$ das unlösbare System repräsentiert. Das ist klar in allen Fällen bis vielleicht auf *(occurs)*. Eine Gleichung der Form $x \doteq E$ mit $x \in FV(E)$ und $x \neq E$ ist aber offenbar nicht unifizierbar: Da E mindestens ein Vorkommen von x und mindestens ein weiteres Symbol enthält, ist der Term $E\sigma$ stets größer als der Term $x\sigma$.

8 Normalformen in Logik erster Stufe

Wir erinnern daran, dass wir für Aussagenlogik in Abschnitt 5 verschiedene Normalformen eingeführt haben, insbesondere die Negationsnormalform (NNF) und die konjunktive Normalform

(CNF). Die NNF haben wir in Abschnitt 6.2 bereits auf die Logik erster Stufe ausgedehnt; zur Erinnerung: Eine Formel in Logik erster Stufe (mit $\vee$ und $\exists$ als eigenständigen Symbolen) ist in NNF, wenn in ihr Negationen nur direkt vor atomaren Formeln (also Gleichungen oder Prädikatenanwendungen) vorkommen, und wir formen eine gegebene Formel in NNF um, indem wir die für die Aussagenlogik in Abschnitt 5 angegebenen Umformungsregeln (de Morgansche Regeln, Entfernung von Doppelnegationen) sowie die Äquivalenzen

$$\neg\forall x(\phi) \equiv \exists x(\neg\phi) \qquad \neg\exists x(\phi) \equiv \forall x(\neg\phi)$$

erschöpfend anwenden. Wir führen nun weitere Normalisierungen ein, die sich spezifisch mit der Form des Vorkommens von Quantoren befassen.

8.1 Pränexe Normalform

Unsere erste Normalisierung betrifft die Position der Quantoren, die wir an den Anfang der Formel ziehen:

Definition 55. Eine *pränexe Normalform* ist eine Formel der Form

$$Q_1 x_1 . \ldots Q_n x_n . \phi$$

mit $Q_1, \ldots, Q_n \in \{\forall, \exists\}$ und ϕ quantorenfrei.

Satz 56. *Zu jeder Formel in Prädikatenlogik erster Stufe lässt sich eine äquivalente pränexe Normalform berechnen.*

Beweis. Wir können annehmen, dass die Formel bereits in NNF ist. Die Berechnung erfolgt dann durch erschöpfende Anwendung der Umformungsregeln

$$\begin{aligned} \phi \wedge \exists x(\psi) &\equiv \exists x(\phi \wedge \psi) & \qquad \phi \wedge \forall x(\psi) &\equiv \forall x(\phi \wedge \psi) \\ \phi \vee \exists x(\psi) &\equiv \exists x(\phi \vee \psi) & \qquad \phi \vee \forall x(\psi) &\equiv \forall x(\phi \vee \psi), \end{aligned}$$

wobei für die Korrektheit dieser Umformungen $x \notin FV(\phi)$ vorausgesetzt wird, was durch geeignete Umbenennung der gebundenen Variablen x stets erreicht werden kann. Es sind jeweils die symmetrischen Fälle mit gemeint. Jede Umformung beseitigt einen *Fehlstand*, d.h. ein Vorkommen eines Booleschen Konnektivs oberhalb eines Quantors, so dass das Verfahren mit einer Formel, auf die keine Umformungsregel mehr anwendbar ist, terminiert; eine solche ist eine pränexe Normalform. □

Beispiel 57. Die Formel

$$\forall x(\forall y(L(y,x)) \to \exists y(M(x,y)))$$

wird wie folgt umgeformt:

$$\begin{aligned} &\forall x(\forall y(L(y,x)) \to \exists y(M(x,y))) \\ &\equiv \forall x(\neg\forall y(L(y,x)) \vee \exists y(M(x,y))) \\ &\equiv \forall x(\exists y(\neg L(y,x)) \vee \exists y(M(x,y))) \\ &\equiv \forall x \exists y((\exists y(\neg L(y,x)) \vee\ M(x,y))) \\ &\equiv \forall x \exists y \exists y'(\neg L(y',x) \vee\ M(x,y)) \end{aligned}$$

.

8.2 Skolemform

Als nächstes eliminieren wir den Existenzquantor:

Definition 58. Eine *Skolemform* ist eine pränexe Normalform, die nur Allquantoren enthält.

Es gibt ersichtlich nicht zu jeder Formel eine äquivalente Skolemform; z.B. ist jede Skolemform über der leeren Signatur Σ äquivalent zu entweder $\top$ oder $\bot$ oder $\forall x, y(x = y)$ (warum?), und damit nicht äquivalent zu $\exists x, y(x \neq y)$. Es gilt jedoch

Definition 59. Formeln ϕ, ψ heißen *erfüllbarkeitsäquivalent*, wenn ϕ genau dann erfüllbar ist, wenn ψ erfüllbar ist.

Satz 60. *Zu jeder Formel in Prädikatenlogik erster Stufe lässt sich eine erfüllbarkeitsäquivalente Skolemform berechnen.*

Diese *Skolemisierung* beruht auf der Ersetzung von Existenzquantoren durch frische Konstanten, basierend auf der Beobachtung, dass $\exists x(\phi)$ genau dann erfüllbar ist, wenn $\phi[c/x]$ für eine frische Konstante c (eine *Skolemkonstante*) erfüllbar ist. Im allgemeinen hängen die für x einzusetzenden Werte noch von vorhergehenden allquantifizierten Variablen ab, so dass man es mit Skolemfunktionen statt nur Skolemkonstanten zu tun hat. Formal:

Beweis (Satz 60). Man berechnet zunächst eine pränexe Normalform

$$Q_1 x_1 \ldots Q_n x_n(\chi),$$

und beseitigt dann die Existenzquantoren durch wiederholte Anwendung der Umformung

$$\forall x_1 \ldots \forall x_k \exists y(\phi) \rightsquigarrow \forall x_1 \ldots \forall x_k(\phi[f(x_1, \ldots, x_n)/y])$$

(Achtung: Nur auf ganze Formeln, nicht auf Teilformeln innerhalb größerer Formeln! Umformung von Teilformeln in erfüllbarkeitsäquivalente Formeln ist insbesondere im allgemeinen nicht selbst eine erfüllbarkeitsäquivalente Umformung.) mit jeweils einem frischen (insbesondere bisher nicht zur Signatur gehörendem) Funktionssymbol f, das wir als *Skolemfunktkion* bezeichnen. (Bei der Transformation von Formelmengen darf jedes f nur zur Transformation einer Formel verwendet werden.)

Wir bezeichnen die linke Seite dieser Umformung mit ψ und die rechte mit $\bar{\psi}$; wir müssen zeigen, dass ψ und $\bar{\psi}$ erfüllbarkeitsäquivalent sind:

Wenn einerseits $\mathfrak{M}, \eta \models \bar{\psi},$, dann auch $\mathfrak{M}, \eta \models \psi$, da für jedes $\eta' = \eta[x_1 \mapsto m_1, \ldots, x_k \mapsto m_k]$ $\mathfrak{M}[\![f(x_1, \ldots, x_k)]\!]\eta'$ per Substitutionslemma einen Zeugen für den Existenzquantor für x_j liefert.

Wenn umgekehrt $\mathfrak{M}, \eta \models \phi$, dann kann man $\mathfrak{M}$ zu einem Modell $\overline{\mathfrak{M}}$ der um die Skolemfunktionen vergrößerten Signatur mit $\overline{\mathfrak{M}}, \eta \models \bar{\phi}$ *erweitern*, indem man für jedes Tupel $(m_1, \ldots, m_k) \in M^k$ den Wert $\mathfrak{M}[\![f]\!](m_1, \ldots, m_k) = m$ so *wählt*, dass

$$\overline{M}, \eta[x_1 \mapsto m_1, \ldots, x_k \mapsto m_k, y \mapsto m] \models \phi$$

gilt (was allerdings am sogenannten *Auswahlaxiom* der Mengenlehre hängt); dann gilt per Substititionslemma wie verlangt auch

$$\overline{M}, \eta[x_1 \mapsto m_1, \ldots, x_k \mapsto m_k] \models \phi[f(x_1, \ldots, x_n)/y].$$

Es folgt $\overline{M}, \eta \models \bar{\psi}$. □

(In der Praxis kann man die Skolemisierung in einem Schritt durchführen, indem man in $\phi = Q_1x_1 \ldots Q_nx_n(\chi)$ alle Existenzquantoren $\exists x_j$ streicht und jede existenzquantifizierte Variable x_j durch

$$f_j(x_{j_1}, \ldots, x_{j_k})$$

substituiert, wobei f_j ein jeweils neu eingeführtes k-stelliges Funktionssymbol ist und $j_1, \ldots, j_k$ diejenigen Indizes $\leqslant j$ sind, für die $Q_j = \forall$; d.h. $\{j_1, \ldots, j_k\} = \{i \leqslant j \mid Q_i = \forall\}$)

Beispiel 61. Wir führen die Normalisierung aus Beispiel 57 fort:

$$\forall x \exists y \exists y' (\neg L(y', x) \vee M(x, y))$$

ist erfüllbarkeitsäquivalent zu seiner Skolemisierung

$$\forall x (\neg L(f_3(x), x) \vee M(x, f_2(x))).$$

Bemerkung 62. Das im Beweis von Satz 56 verwendete Verfahren zur Berechnung einer pränexen Normalform beinhaltet Wahlfreiheiten hinsichtlich der jeweils als nächstes anzuwendenden Umformung, und verschiedene Umformungsabfolgen können verschiedene pränexe Normalformen liefern. Z.B. kann man $\forall x(P(x)) \vee \exists y(Q(y))$ zu sowohl $\forall x \exists y(P(x) \vee Q(y))$ also auch $\exists y \forall x(P(x) \vee Q(y))$ umformen. Zweiteres ist für Zwecke der Skolemisierung günstiger, da dann die Skolemfunktion für den Existenzquantor ein Argument weniger (nämlich keins) hat. Aus ähnlichen Gründen kann es vorteilhaft sein, vor der Bildung von pränexen Normalformen die Reihenfolge in Konjunktionen oder Disjunktionen geeignet zu vertauschen.

8.3 Klauselform

Man kann nach unseren Resultaten über Normalformen in Aussagenlogik jede pränexe Normalform $Q_1x_1 \ldots Q_nx_n(\phi)$ so weiter normalisieren, dass ϕ in CNF ist. Wenn die Formel zusätzlich in Skolemform ist, also $Q_i = \forall$ für alle i, dann kann man die führenden Quantoren in der Notation weglassen, so dass man also alle Variablen als implizit allquantifiziert ansieht. Für die so erhaltene CNF verwendet man dann üblicherweise die bereits eingeführte Schreibweise als Menge von Klauseln. Eine solche Klauselmenge heißt *Klauselform* bzw. *ist in Klauselform*.

Beispiel 63. Die Formel aus Beispiel 61 hat bereits eine CNF als quantorenfreien Anteil; sie lautet in Klauselform (unter Weglassung der äußeren Mengenklammern)

$$\{\neg L(f_3(x), x),\ M(x, f_2(x))\}$$

(besteht also nur aus einer Klausel).

9 Resolution in Prädikatenlogik erster Stufe

Das Resolutionsverfahren für Aussagenlogik lässt sich auf Prädikatenlogik erster Stufe erweitern, ist dort allerdings nur noch ein Halbentscheidungsverfahren für Unerfüllbarkeit (d.h. für erfüllbare Eingabeformeln terminiert es möglicherweise nicht). Das Verfahren arbeitet mit Formeln in Klauselform.

Die sogenannte *generalisierte Reduktionsregel*, auf der das Verfahren beruht, kombiniert die aussagenlogische Resolutionsregel

$$\frac{C_1, A \qquad C_2, \neg A}{C_1, C_2}$$

(wobei wir nunmehr Mengenklammern einsparen und Mengenvereinigungen einfach mit Kommata andeuten) mit der Spezialisierungsregel

$$\frac{C}{C\sigma},$$

die aus einer Klausel C eine Substitutionsinstanz für eine Substitution σ folgert, sowie mit der *Faktorisierungsregel*

$$\frac{C, A, A}{C, A}.$$

Die kombinierte Regel, auch *Resolution mit impliziter Faktorisierung* genannt, verwendet Unifikation. Sie lautet

$$(RIF)\frac{C_1, A_1, \ldots, A_n \qquad C_2, \neg B}{C_1\sigma, C_2\sigma} \qquad (\sigma = mgu(A_1, \ldots, A_n, B)).$$

Hierbei schreiben wir kurz $mgu(A_1, \ldots, A_n, B)$ für den mgu des Systems $\{A_i \doteq B \mid i = 1, \ldots, n\}$. Vor Anwendung der Regel werden die Variablen in den beiden ursprünglichen Klauseln so umbenannt, dass die Klauseln disjunkte Variablenmengen haben (aus Lesbarkeitsgründen nehmen wir diese Umbenennung nicht in die Notation der Regel mit auf). Das ist offenbar zulässig, da die Klauseln ja allquantifizierte Formeln darstellen, und vergrößert die Menge der Unifikatoren.

Der Algorithmus zum Beweis der Gültigkeit einer Formel ϕ lautet damit

1. Bilde $\neg\phi$
2. Transformiere $\neg\phi$ in Klauselform
3. Wende (RIF) an, bis $\square$ erreicht ist.

(Achtung: anders als bei der aussagenlogischen Resolution kann man hier i.a. nicht erwarten, dass, wenn ϕ nicht gültig ist, irgendwann eine Formelmenge erreicht wird, auf die (RIF) nicht mehr anwendbar ist; d.h. der Algorithmus terminiert in diesem Fall eventuell nicht.)

Beispiel 64. Die Negation der Formel

$$P(a) \wedge \forall x(P(x) \to P(f(x))) \to \exists x(P(f(f(x))))$$

wird in Klauselform zu

$$(1)\ \{P(a)\} \qquad (2)\ \{\neg P(x), P(f(x))\} \qquad (3)\ \{\neg P(f(f(x)))\}.$$

Resolution von (1) mit (2) liefert

$$(4)\ \{P(f(a))\},$$

Resolution von (4) mit (2) liefert

$$(5)\ \{P(f(f(a)))\},$$

und Resolution mit (3) liefert schließlich $\square$, womit die ursprüngliche Formel bewiesen ist.

9.1 Herbrand-Modelle

Wir zeigen nun die *Vollständigkeit* des Resolutionsalgorithmus, d.h. dass der Algorithmus auf jeder *unerfüllbaren* Klauselform mit der Antwort 'unerfüllbar' terminiert. (Auf *erfüllbaren* Klauselformen terminiert der Algorithmus eventuell nicht.) Der Schlüsselbegriff ist hierbei der des sogenannten *Herbrandmodells*; Herbrandmodelle zeichnen sich dadurch aus, dass in ihnen jedes Element durch einen Term ohne Variablen bezeichnet wird. Insbesondere sind Herbrandmodelle damit abzählbar; ihre Konstruktion impliziert daher den berühmten *Satz von Löwenheim-Skolem*, der besagt, dass jede erfüllbare Formelmenge in Prädikatenlogik ein höchstens abzählbares Modell hat (sofern die Signatur höchstens abzählbar ist).

Zur technischen Vereinfachung schränken wir ab jetzt auf Prädikatenlogik ohne „=“ ein; d.h. als atomare Formeln gibt es nur noch Prädikatenanwendungen $P(E_1, \dots, E_n)$. Für den Rest des Abschnitts sei eine Signatur Σ gegeben; o.E. nehmen wir an, dass Σ mindestens eine Konstante enthält (dass das o.E. ist, liegt daran, dass per Definition alle Modelle nichtleeren Träger haben).

Definition 65 (Herbrand-Universum)**.** Das *Herbrand-Universum* U_Σ ist die Menge aller *Grundterme* (Ground Terms), d.h. Terme ohne Variablen:

$$U_\Sigma = \{E \mid E\ \Sigma\text{-Term}, FV(E) = \varnothing\}.$$

Für $E \in U_\Sigma$ und ein Σ-Modell $\mathfrak{M}$ schreiben wir $\mathfrak{M}[\![E]\!]$ statt $\mathfrak{M}[\![E]\!]\eta$ für die Auswertung von E in $\mathfrak{M}$, da Umgebungen für die Auswertung geschlossener Terme irrelevant sind.

Beispiel 66. Bei $\Sigma = \{s/1, z/0, O/2\}$ haben wir $U_\Sigma = \{z, s(z), s(s(z)), \dots\}$.

Definition 67 (Herbrand-Modell)**.** Ein *Herbrand-(Σ-)Modell* ist ein Σ-Modell $\mathfrak{M}$ mit folgenden Eigenschaften:

- $M = U_\Sigma$
- Für $f/n \in \Sigma$ und $E_1, \dots, E_n \in U_\Sigma$ gilt

$$\mathfrak{M}[\![f]\!](E_1, \dots, E_n) = f(E_1, \dots, E_n)$$

Beispiel 68. Mit Σ wie oben gilt also in Herbrand-Σ-Modellen $\mathfrak{M}$ z.B. $\mathfrak{M}[\![s]\!](s(z)) = s(s(z))$.

Lemma 69 (Auswertungslemma für Terme)**.** *Sei $\mathfrak{M}$ ein Herbrand-Σ-Modell. Dann gilt*

$$\mathfrak{M}[\![E]\!]\eta = E\eta$$

für jeden Term E. Insbesondere gilt $\mathfrak{M}[\![E]\!] = E$ für Grundterme E.

Zum Verständnis der Aussage des Lemmas beachte man, dass eine Umgebung η in $\mathfrak{M}$ gleichzeitig eine Substitution ist, eben eine Abbildung von Variablen auf Terme (bei Unterschlagung der hier nicht so wichtigen Endlichkeitsbedingung an $\mathrm{Dom}(\eta)$).

Beweis. Induktion über E. □

Definition 70 (Grundinstanz)**.** Sei φ eine Formel. Eine *Grundinstanz* (Ground Instance) von φ ist eine Formel der Form $\varphi\sigma$, wobei σ eine *Grundsubstitution* (Ground Substitution) ist, d.h. für alle $x \in FV(\varphi)$ ist $\sigma(x)$ ein Grundterm; äquivalenterweise ist $\varphi\sigma$ eine geschlossene Formel, also $FV(\varphi\sigma) = \varnothing$. Für eine Menge Φ von Formeln bezeichnet $E(\Phi)$ die Menge der Grundinstanzen von Formeln aus Φ, d.h.

$$E(\Phi) = \{\phi\sigma \mid \phi \in \Phi, FV(\phi\sigma) = \varnothing\}.$$

Lemma 71 (Auswertungslemma für Formeln)**.** *Sei φ eine Formel und $\mathfrak{M}$ ein Herbrand-Modell. Dann gilt, unter erneuter Verwechslung von Substitutionen und Umgebungen,*

$$\mathfrak{M}, \eta \models \varphi \iff \mathfrak{M} \models \varphi\eta.$$

Man beachte dabei, dass oben $\varphi\eta$ eine Grundinstanz von φ ist.

Beweis. Nach dem Substitutionslemma gilt zunächst

$$\mathfrak{M}, \eta' \models \varphi \iff \mathfrak{M} \models \varphi\eta,$$

wobei $\eta'(X) = \mathfrak{M}[\![\eta(X)]\!]$. Nach dem Auswertungslemma für Terme gilt aber $\eta' = \eta$. □

Definition 72 (Universeller Abschluss)**.** Der *universelle Abschluss* $\forall\varphi$ einer Formel φ mit $FV(\varphi) = \{x_1, \ldots, x_n\}$ ist die Formel $\forall x_1 \ldots \forall x_n(\varphi)$. Wir dehnen diesen Begriff auf Formelmengen Φ aus per $\forall\Phi = \{\forall\varphi \mid \varphi \in \Phi\}$.

In Worten: Der universelle Abschluss von φ allquantifiziert alle freien Variablen in φ.

Lemma 73 (Grundinstanzlemma)**.** *Sei $\mathfrak{M}$ ein Herbrand-Σ-Modell und φ eine Formel. Dann gilt*

$$\mathfrak{M} \models \forall\varphi \iff \mathfrak{M} \models \varphi\sigma \text{ für jede Grundinstanz } \varphi\sigma \text{ von } \varphi.$$

Beweis. Gemäß der Semantik des Allquantors gilt $\mathfrak{M} \models \forall\phi$ genau dann, wenn $\mathfrak{M}, \eta \models \phi$ für alle Umgebungen η gilt. Letzteres ist nach dem Auswertungslemma äquivalent zur rechten Seite der behaupteten Äquivalenz. □

Entscheidend ist nunmehr

Satz 74 (Herbrand-Vollständigkeit)**.** *Sei Φ eine Menge von quantorenfreien Formeln über Σ. Dann sind äquivalent:*

1. *$\forall\Phi$ ist erfüllbar.*
2. *Es gibt ein ein Herbrandmodell $\mathfrak{M}$ mit $\mathfrak{M} \models \forall\Phi$.*
3. *$E(\Phi)$ ist aussagenlogisch erfüllbar.*

Dabei meinen wir in 3 mit *aussagenlogisch erfüllbar*, dass die Menge aussagenlogischer Formeln, die man erhält, wenn man alle atomaren Unterformeln als Atome mit merkwürdigen Namen ansieht, erfüllbar ist.

Beweis. 1. $\implies$ 3. und 2. $\implies$ 1. sind trivial. Wir zeigen 3. $\implies$ 2. Sei also $\kappa \models E(\Phi)$. Wir konstruieren dann ein Herbrand-Modell $\mathfrak{M}$ mit $\mathfrak{M} \models \forall\Phi$ per

$$\mathfrak{M}[\![P]\!] = \{(E_1, \ldots, E_n) \mid \kappa(P(E_1, \ldots, E_n)) = \top\}$$

für $P/n \in \Sigma$. Um zu zeigen, dass tatsächlich $\mathfrak{M} \models \forall\Phi$ gilt, reicht es nach dem Grundinstanzlemma zu zeigen, dass $\mathfrak{M} \models E(\Phi)$. Man zeigt durch Induktion über quantorenfreies ϕ, dass für jede Grundinstanz $\phi\sigma$

$$\mathfrak{M} \models \phi\sigma \iff \kappa \models \phi\sigma \tag{5}$$

gilt. Damit folgt dann die Behauptung, da ja $\kappa \models E(\Phi)$. Die Booleschen Fälle in der Induktion für (5) sind (wie meistens) leicht, und für atomare Formeln gilt die Äquivalenz nach der Konstruktion von $\mathfrak{M}$ und dem Auswertungslemma für (geschlossene) Terme:

$$\begin{aligned}
\mathfrak{M} \models P(E_1, \ldots, E_n)\sigma &\iff (\mathfrak{M}[\![E_1\sigma]\!], \ldots, \mathfrak{M}[\![E_n\sigma]\!]) \in \mathfrak{M}[\![P]\!] && \text{(Semantik)} \\
&\iff (E_1\sigma, \ldots, E_n\sigma) \in \mathfrak{M}[\![P]\!] && \text{(Auswertungslemma)} \\
&\iff \kappa(P(E_1\sigma, \ldots, E_n\sigma)) = \top && \text{(Definition von } \mathfrak{M}[\![P]\!]\text{)} \\
&\iff \kappa(P(E_1, \ldots, E_n)\sigma) = \top && \text{(Definition der Substitution)} \\
&\iff \kappa \models P(E_1, \ldots, E_n)\sigma && \text{(Semantik)}
\end{aligned}$$

(da ϕ quantorenfrei ist, gibt es keine weiteren Fälle). □

Der Satz gilt nicht für beliebige Formelmengen, z.B.

Beispiel 75. Man betrachte

$$\begin{aligned}
\Sigma &= \{a/0, P/1\} \\
\Phi &= \{\neg P(a), \exists x(P(x))\} \\
U_\Sigma &= \{a\}.
\end{aligned}$$

Die Formelmenge Φ ist offenbar in keinem einelementigen Modell erfüllbar, insbesondere also in keinem Herbrand-Modell, da Herbrandmodelle Grundbereich $U_\Sigma = \{a\}$ haben. Andererseits ist Φ aber offenbar erfüllbar, z.B. wähle $M = \{0, 1\}$, $\mathfrak{M}[\![a]\!] = 0$, $\mathfrak{M}[\![P]\!] = \{1\}$.

Da Herbrandmodelle über abzählbarem Σ abzählbar sind, folgt hieraus wie angekündigt

Satz 76 (Löwenheim/Skolem)**.** *Jede erfüllbare Menge von Sätzen über einer höchstens abzählbaren Signatur hat ein höchstens abzählbares Modell.*

Beweis. Durch Skolemisierung der gegebenen Formelmenge Ψ erhalten wir eine Formelmenge $\forall\Phi$ mit Φ quantorenfrei, über einer vergrößerten, aber weiterhin abzählbaren Signatur. Per Herbrand-Vollständigkeit hat $\forall\Phi$ ein abzählbares Modell (nämlich ein Herbrandmodell); dieses erfüllt dann auch Ψ. □

Als weitere Anwendung der Herbrand-Vollständigkeit erben wir den Kompaktheitssatz von der Aussagenlogik:

Satz 77. *Logik erster Stufe ist* kompakt*, d.h. wenn Φ eine Menge von Sätzen ist, so dass jede endliche Teilmenge von Φ erfüllbar ist, dann ist Φ erfüllbar.*

Beweis. Wir können annehmen, dass Φ aus Skolemformen besteht, d.h. $\Phi = \forall\Phi_0$ für eine Menge Φ_0 von quantorenfreien Formeln. Per Herbrand-Vollständigkeit ist dann Φ erfüllbar genau dann, wenn $E(\Phi_0)$ aussagenlogisch erfüllbar ist. Jede endliche Teilmenge Ψ von $E(\Phi_0)$ ist enthalten in einer Menge der Form $E(\bar{\Psi})$ für eine endliche Teilmenge $\bar{\Psi}$ von Φ_0. Nach Voraussetzung ist $\bar{\Psi}$ erfüllbar, damit auch $E(\bar{\Psi})$ und somit erst recht auch $\Psi \subseteq E(\bar{\Psi})$, d.h. $E(\Phi_0)$ ist endlich erfüllbar, also nach Kompaktheit der Aussagenlogik erfüllbar. □

Ferner erhalten wir Halbentscheidbarkeit des Gültigkeitsproblems der Prädikatenlogik erster Stufe:

Satz 78. *Es gibt einen Algorithmus, der alle gültigen Formeln in Logik erster Stufe aufzählt.*

Beweis. Dual zeigen wir, dass es einen Algorithmus gibt, der die *unerfüllbaren* Formeln auflistet. Wir können uns auf Skolemformen φ einschränken; es reicht, einen Algorithmus anzugeben, der mit der Antwort „unerfüllbar“ terminiert, wenn φ unerfüllbar ist, und ansonsten entweder „erfüllbar“ antwortet oder nicht terminiert. Wir erzeugen einfach nacheinander alle Grundinstanzen von φ; sobald wir eine aussagenlogisch unerfüllbare Menge erreichen, antworten wir „unerfüllbar“; falls die Menge der Grundinstanzen unendlich und aussagenlogisch erfüllbar ist, terminiert der Algorithmus also nicht. Im Sonderfall, dass die Menge der Grundinstanzen endlich und aussagenlogisch erfüllbar ist, antworten wir „erfüllbar“. □

9.2 Vollständigkeit der Resolution

Wir zeigen nunmehr wie angekündigt die Vollständigkeit der prädikatenlogischen Resolution. Wir reduzieren diese mittels Herbrandvollständigkeit auf die schon gezeigte Vollständigkeit der aussagenlogischen Resolution. Hierzu benötigen wir eine technische Aussage über das Verhältnis der beiden Resolutionsverfahren, für die wir wiederum die genaue Gestaltung der prädikatenlogischen Resolutionsregel eingehender diskutieren müssen.

Wir erinnern daran, dass die Anwendung von (RIF) auf den mgu als Substitution beschränkt ist. Wir bezeichnen mit (lRIF) die *liberalisierte RIF-Regel*, in der wir diese Einschränkung aufgeben und stattdessen beliebige Unifikatoren von $A_1, \ldots, A_n, B$ zulassen. Dies ändert offenbar nicht die Stärke des Systems:

Lemma 79. *Wenn sich aus einer Klauselmenge mittels (lRIF) die leere Klausel herleiten lässt, dann auch mittels (RIF).*

Beweis. Man zeigt durch Induktion über die Länge der Herleitung, dass jede mittels (lRIF) herleitbare Klausel eine Substitutionsinstanz einer mittels (RIF) herleitbaren Klausel ist. Damit folgt die Behauptung, da eine Klausel, die die leere Klausel als Substitutionsinstanz hat, selbst leer sein muss. □

Wir bezeichnen ferner mit (RI2F) (*resolution with implicit two-sided factoring*) die Regel

$$\frac{C, A_1, \ldots, A_n \qquad \neg B_1, \ldots, \neg B_k, D}{C\sigma, D\sigma} \quad \sigma = mgu(A_1, \ldots, A_n, B_1, \ldots, B_k)$$

Lemma 80. *(RI2F) ist aus (lRIF) herleitbar.*

Beweis. Ohne Einschränkung bestehe $FV(\sigma)$ aus frischen Variablen, so dass $\sigma\sigma = \sigma$. Wir wenden (lRIF) mit der Substitution σ auf $A_1, \ldots, A_n, \neg B_1$ an und erhalten

$$C\sigma, \neg B_2\sigma, \ldots, \neg B_k\sigma, D\sigma.$$

Wir wenden dann (lRIF) mit der Substitution σ an auf $A_1, \ldots, A_n, B_2\sigma$ und erhalten (da $\sigma\sigma = \sigma$)

$$C\sigma, \neg B_3\sigma, \ldots, \neg B_k\sigma, D\sigma.$$

Wir fahren so fort und erhalten schließlich $C\sigma, D\sigma$. □

Bemerkung 81. Dagegen ist (RIF) nicht herleitbar aus der einfachen Resolutionsregel ohne Faktorisierung, Gegenbeispiel: Klauseln $\{P(x), P(y)\}, \{\neg P(z), \neg P(w)\}$.

Lemma 82 (Lifting-Lemma)**.** *Seien $C, A_1, \ldots, A_n$ und $\neg B_1, \ldots, \neg B_k, D$ prädikatenlogische Klauseln, und sei σ eine Grundsubstitution mit $A_i\sigma = B_j\sigma$ für alle i, j. Dann ist die zugehörige aussagenlogische Resolvente $C\sigma, D\sigma$ Grundinstanz einer prädikatenlogischen Resolvente von $C, A_1, \ldots, A_n$ und $\neg B_1, \ldots, \neg B_k, D$ nach (RI2F).*

Beweis. Wähle $\sigma' = mgu(A_1, \ldots, A_n, B_1, \ldots, B_k)$. Dann gilt $\sigma = \sigma'\tau$ für eine Grundsubstitution τ, und $C\sigma, D\sigma$ ist unter τ Grundinstanz der prädikatenlogischen Resolvente $C\sigma', D\sigma'$. □

Bemerkung 83. Der Beweis des Lifting-Lemmas ist also eher trivial; man beachte aber, dass das entsprechende Lemma für die Regel (RIF) mit nur einseitiger Faktorisierung *nicht* gilt: Gegeben prädikatenlogische Klauseln und σ wie im Lemma ist die aussagenlogische Resolvente $C\sigma, D\sigma$ i.a. nicht Grundinstanz einer Resolvente von $C, A_1, \ldots, A_n$ und $\neg B_1, \ldots, \neg B_k, D$ nach (RIF), da in einer solchen Resolvente bei $k > 1$ stets noch Literale der Form $\neg B_i\sigma$ verbleiben.

Damit gilt

Satz 84 (Vollständigkeit der prädikatenlogischen Resolution)**.** *Wenn Φ eine unerfüllbare Menge von prädikatenlogischen Klauseln ist, dann existiert eine Herleitung der leeren Klausel $\square$ aus Φ mittels prädikatenlogischer Resolution.*

Beweis. Nach Herbrandvollständigkeit ist $E(\Phi)$ aussagenlogisch unerfüllbar; nach der Vollständigkeit der aussagenlogischen Resolution existiert also eine Herleitung von $\square$ aus Grundinstanzen mittels aussagenlogischer Resolution. Indem wir das Lifting-Lemma auf alle Schritte dieses Resolutionsbeweises anwenden, erhalten wir eine Herleitung einer Klausel C aus Φ mittels (RI2F), und damit nach Lemma 80 auch per (lRIF), so dass $\square$ eine Grundinstanz von C ist. Dann ist aber $C = \square$, also per Lemma 79 auch per (RIF) herleitbar. □

10 Quantorenelimination

Erfüllbarkeit von Formeln in Prädikatenlogik ist im allgemeinen unentscheidbar; dies zeigt man z.B. durch Reduktion des Postschen Korrespondenzproblems (siehe Skript „Ontologien im Semantic Web“). Erfüllbarkeit unter bestimmten Theorien kann dagegen in günstigen Fällen entscheidbar sein. Oft basieren Entscheidungsverfahren auf der Methode der *Quantorenelimination*, d.h. der äquivalenten Ersetzung von Formeln durch quantorenfreie Formeln.

Definition 85 (Theorie)**.** Eine *Theorie* $\mathcal{T} = (\Sigma, \Phi)$ besteht aus einer Signatur Σ und einer Menge Φ von Σ-Sätzen, ihren *Axiomen*. Ein *Modell* von $\mathcal{T}$ ist ein Σ-Modell $\mathfrak{M}$ mit $\mathfrak{M} \models \Phi$. Eine Σ-Formel ψ heißt *erfüllbar unter* $\mathcal{T}$, wenn ψ in einem Modell von $\mathcal{T}$ erfüllbar ist, d.h. wenn $\Phi \cup \{\psi\}$ erfüllbar ist. Wir schreiben $\mathcal{T} \models \psi$, wenn $\neg\psi$ unerfüllbar unter $\mathcal{T}$ ist, d.h. wenn $\Phi \models \psi$.

Beispiel 86. Die Theorie $\mathcal{T}_<$ der *dichten linearen Ordnungen ohne Endpunkte* hat die Signatur

$$\Sigma_< = \{</2\},$$

d.h. ein binäres Prädikat $<$, geschrieben in Infixnotation, sowie die Axiome

$$\begin{aligned}
&\forall x(\neg(x < x)) && \text{(Irreflexivität)}\\
&\forall x, y, z(x < y \wedge y < z \to x < z) && \text{(Transitivität)}\\
&\forall x, y(x < y \vee x = y \vee y < x) && \text{(Trichotomie)}\\
&\forall x, y(x < y \to \exists z(x < z \wedge z < y)) && \text{(Dichte)}\\
&\forall x \exists y(y < x) &&\\
&\forall x \exists y(x < y) && \text{(Endpunktfreiheit).}
\end{aligned}$$

Ein Modell dieser Theorie sind z.B. die rationalen Zahlen mir der üblichen Interpretation von $<$; ein weiteres Modell sind die reellen Zahlen.

Wir werden mittels Quantorenelimination zeigen, dass Erfüllbarkeit unter $\mathcal{T}_<$ entscheidbar ist; man sagt kurz, dass $\mathcal{T}_<$ *entscheidbar* ist. Der allgemeine Ansatz ist hierbei wie folgt.

Definition 87. Eine Theorie $\mathcal{T} = \{\Sigma, \Phi\}$ *hat Quantorenelimination*, wenn für jede Σ-Formel ϕ eine quantorenfreie Σ-Formel ϕ' berechenbar ist, so dass $\mathcal{T} \models \phi \leftrightarrow \phi'$.

Fakt 88. *Wenn $\mathcal{T} = (\Sigma, \Phi)$ Quantorenelimination hat und Erfüllbarkeit von quantorenfreien Formeln unter $\mathcal{T}$ entscheidbar ist, dann ist $\mathcal{T}$ entscheidbar.*

Quantorenelimination lässt sich auf einen Spezialfall reduzieren:

Lemma 89. *Wenn für jede Σ-Formel ϕ der Form $\exists x(\alpha_1 \wedge \cdots \wedge \alpha_n)$ mit Literalen α_i eine quantorenfreie Formel ϕ' mit $\mathcal{T} \models \phi \leftrightarrow \phi'$ berechenbar ist, dann hat $\mathcal{T}$ Quantorenelimination.*

Beweis. Wir zeigen per Induktion über ψ, dass für jede Σ-Formel ψ eine äquivalente quantorenfreie Formel berechenbar ist. Wir nehmen dabei an, dass ψ nur $\exists$ enthält (d.h. $\forall$ durch $\exists$ definiert ist). Für Atome ist nichts zu zeigen, und die Booleschen Fälle sind trivial. Sei also ψ von der Form $\exists x(\chi)$. Nach Induktionsvoraussetzung existiert ein quantorenfreies χ' mit $\mathcal{T} \models \chi \leftrightarrow \chi'$. Wir bringen χ' in DNF $\chi' \equiv \chi'_1 \vee \cdots \vee \chi'_n$, wobei die χ'_i Konjunktionen von Literalen sind. Per Vertauschung von $\exists$ mit $\vee$ (s.o.) gilt dann $\mathcal{T} \models \exists x(\chi) \leftrightarrow \exists x(\chi'_1) \vee \cdots \vee \exists x(\chi'_n)$, und die Quantoren auf der rechten Seite sind nach Voraussetzung eliminierbar. □

Damit zeigen wir

Satz 90. *Die Theorie der dichten linearen Ordnungen ohne Endpunkte hat Quantorenelimination.*

Beweis. Sei $\phi = \exists x(\alpha_1 \wedge \cdots \wedge \alpha_n)$ mit Literalen α_i. Wir gehen schrittweise vor:

1. Wir eliminieren zunächst Negation: Per Irreflexivität, Transitivität und Trichotomie gilt $\mathcal{T} \models (\neg y = z) \leftrightarrow (y < z \vee z < y)$ und $\mathcal{T} \models (\neg(y < z) \leftrightarrow (y = z \vee z < y)$.

2. Damit erreichen wir eine Formel, die keine Negationen mehr, dafür aber wieder Disjunktionen enthält. Wie im Beweis von Lemma 89 formen wir ϕ dann in eine Disjunktion von existenzquantifizierten Formeln des richtigen Formats um, und machen mit einer solchen existenzquantifizierten Formel weiter.

3. Damit erreichen wir eine Formel $\psi = \exists x(\beta_1 \wedge \cdots \wedge \beta_n)$ mit *Atomen* β_i. Wir eliminieren nun $=$:

- Wenn $\beta_i = (y = y)$, dann streichen wir β_i.
- Wenn $\beta_i = (y = x)$ oder $\beta_i = (x = y)$, dann streichen wir β_i und ersetzen in die verbleibenden β_j jeweils durch $\beta_j[x \mapsto y]$. Abschließend streichen wir den Existenzquantor (x kommt ja nun unter ihm nicht mehr vor) und sind fertig.

Falls wir noch nicht fertig sind, ziehen wir von den verbleibenden Atomen nunmehr diejenigen, die x nicht erwähnen, vor den Existenzquantor (s.o.), unter dem dann kein $=$ mehr vorkommt.

4. Falls unter den verbleibenden β_j $x < x$ vorkommnt, ersetzen wir ψ durch $\bot$ (per Irreflexivität).

5. Der verbleibende existenzquantifizierte Anteil hat nunmehr (nach einfacher Umformung) die Form
$$\exists x(\bigwedge_{i=1}^{n} u_i < x \wedge \bigwedge_{j=1}^{m} x < v_j), \tag{6}$$
wobei die u_i und die v_j von x verschiedene Variablen sind. Diese Formel ist in $\mathcal{T}_<$ äquivalent zu
$$\bigwedge_{i=1}^{n}\bigwedge_{j=1}^{m} u_i < v_j. \tag{7}$$
Dabei ist die Implikation (6) $\Longrightarrow$ (7) klar per Transitivität. Für die Umkehrung bemerken wir, dass es per Trichotomie unter den u_i ein größtes geben muss; sei dies u_{i_0} (formal beweist man per Induktion eine Disjunktion mit n Disjunkten, eins für jedes in Frage kommende u_i). Entsprechend gibt es unter den v_j ein kleinstes, v_{j_0}. Dann gilt nach Voraussetzung $u_{i_0} < v_{j_0}$, so dass per Dichte x existiert mit $u_{i_0} < x$ und $x < v_{j_0}$; dieses x ist dann per Transitivität ein Zeuge für (6). Diese Argumentation setzt natürlich voraus, dass $n, m > 0$. Für $n > 0, m = 0$ und für $n = 0, m > 0$ führt man ein ähnliches Argument mittels Endpunktfreiheit durch; der Fall $n = m = 0$ ist trivial.

□

Um nun tatsächlich Entscheidbarkeit von $\mathcal{T}_<$ zu bekommen, fehlt uns noch die Entscheidbarkeit der Erfüllbarkeit quantorenfreier Formeln unter $\mathcal{T}$. Wir beobachten zunächst, dass bei der Quantorenelimination für $\mathcal{T}_<$ auch $\neg$ eliminiert wird. Ferner können wir mit Disjunktionen stets umgehen, indem wir zunächst in DNF transformieren und dann die Disjunkte der DNF der Reihe nach auf Erfüllbarkeit prüfen (natürlich hören wir damit auf, sobald wir einen erfüllbaren Disjunkt gefunden haben). Das verbleibende Problem ist, die Erfüllbarkeit einer Konjunktion atomarer Formeln der Form $x = y$ oder $x < y$ zu entscheiden. Da wir jetzt nur noch erfüllbarkeitsäquivalent umformen müssen, können wir atomare Formeln $x = y$ jeweils entfernen und in der verbleibenden Formel x durch y substituieren (das ist nicht mehr äquivalent, da ja die Aussage $x = y$ verloren geht, aber offenbar erfüllbarkeitsäquivalent). Zur Vereinfachung der Notation schreiben wir die verbleibende Konjunktion einfach als Menge C von Formeln der Form $x < y$. Folgender Algorithmus entscheidet Erfüllbarkeit von C unter $\mathcal{T}_<$:

1. Wenn in C Formeln $x < y$, $y < z$ enthalten sind mit $x < z$ nicht in C, füge $x < z$ zu C hinzu und mache bei Schritt 1 weiter.

2. Falls C eine Formel der Form $x < x$ enthält, antworte „unerfüllbar“, sonst „erfüllbar“.

Schritt 1 terminiert, da nur endlich (genauer: quadratisch) viele Formeln $x < z$ hinzugefügt werden können. Per Transitivität und Irreflexivität ist klar, dass der Algorithmus recht hat, wenn er „unerfüllbar“ antwortet. Um zu sehen, dass er auch im anderen Fall recht hat, zeigen wir

Lemma 91. *Wenn mit $x < y$ und $y < z$ stets auch $x < z$ in C ist und C keine Formel der Form $x < x$ enthält, dann ist C erfüllbar unter $\mathcal{T}_<$.*

Beweis. Nach Voraussetzung ist $<$ eine strikte partielle Ordnung auf den Variablen. Man kann dann die Variablen topologisch sortieren, d.h. in einer Folge $x_1, \ldots, x_n$ anordnen, so dass $i < j$, wenn $x_i < x_j$ in C ist. Dann $(\mathbb{Q}, <), \eta \models C$ mit $\eta(x_i) = i$. [Siehe AuD; zur Erinnerung: beim topologischen Sortieren entnimmt man einfach wiederholt ein minimales Element aus der partiellen Ordnung und fügt es als nächstes Element in eine Liste ein.] □

11 Vollständigkeit der Prädikatenlogik erster Stufe

Zur Vereinfachung arbeiten wir in Prädikatenlogik ohne Gleichheit (=) und ohne Funktionssymbole positiver Stelligkeit (aber mit Konstanten). Wir beweisen die Umkehrung des Korrektheitssatzes, d.h.

Satz 92 (Vollständigkeit der Prädikatenlogik erster Stufe)**.** *Sei Φ eine Menge von Sätzen und ψ ein Satz über einer Signatur Σ in Prädikatenlogik erster Stufe. Wenn ψ logische Folgerung aus Φ ist ($\Phi \models \psi$), dann ist ψ aus Φ herleitbar ($\Phi \vdash \psi$).*

Bemerkung 93. Eine Version des Satzes für beliebige Formeln (d.h. auch solche mit freien Variablen) ist leicht reduzierbar auf die obige Version, indem man einfach Variablen durch Konstanten ersetzt.

Wie schon im Falle der Aussagenlogik ist der Beweis der Vollständigkeit wesentlich schwieriger als der der Korrektheit. Der Satz wurde zuerst von Gödel bewiesen. Der Beweis, wie wir ihn hier präsentieren, geht auf Leon Henkin zurück; er basiert auf einer Reduktion der Prädikatenlogik erster Stufe auf die Aussagenlogik, für die wir die Vollständigkeit ja schon gezeigt haben. Im Groben besteht der Beweis aus den folgenden Schritten:

(A) Wir erweitern zunächst die Signatur Σ zu einer Signatur Σ_H, die für jede Formel $\phi(x)$ (mit höchstens einer freien Variablen x) über Σ_H eine *Zeugenkonstante* $c_{\phi(x)}$ enthält, gedacht als Name für ein $\phi(x)$ erfüllendes Element, wenn ein solches existiert. Ebenso erweitern wir die gegebene Formelmenge Φ um eine unendliche Menge $\mathscr{H}$ zusätzlicher Axiome, die *Henkin-Theorie*. Diese enthält insbesondere für jede Formel $\phi(x)$ ein Axiom $\exists x(\phi(x)) \to \phi(c_{\phi(x)})$.

(B) (*Henkin-Elimination*) Wir beweisen dann für jede Formel ρ über Σ (die also keine Zeugenkonstanten erwähnt), dass aus $\Phi \cup \mathscr{H} \vdash \rho$ bereits $\Phi \vdash \rho$ folgt, in Worten: wenn sich ρ aus Φ unter Zuhilfenahme der Henkin-Theorie herleiten lässt, dann kann man ρ schon aus Φ allein herleiten .

(C) (*Henkin-Konstruktion*) Wir fassen nun Formeln in Logik erster Stufe als bloße aussagenlogische Formeln auf, indem wir alle Teilformeln der Form $\forall x(\phi)$, $\exists x(\phi)$ oder $P(\ldots)$ als Atome ansehen. Aus einer Wahrheitsbelegung κ mit $\kappa \models \Phi \cup \mathscr{H}$ (*aussagenlogische* Erfülltheit) konstruieren wir dann ein Σ-Modell $\mathfrak{M}_\kappa$ mit $\mathfrak{M}_\kappa \models \Phi$ (*prädikatenlogische* Erfülltheit).

(Zu (C): Z.B. ist die Formel $(\forall x.\phi) \wedge \exists x.\neg\phi$ aussagenlogisch strukturgleich zur Formel $A \wedge B$, insbesondere zwar als prädikatenlogische Formel unerfüllbar, aber aussagenlogisch erfüllbar. Dagegen ist die Formel $(\forall x.\phi) \wedge \neg\forall x.\phi$ schon aussagenlogisch unerfüllbar, da sie strukturgleich zur aussagenlogischen Formel $A \wedge \neg A$ ist.) Damit läuft der Beweis des Vollständigkeitssatzes dann wie folgt: Wie schon im Falle der Aussagenlogik reicht es zu zeigen, dass jede konsistente Menge Φ von Sätzen in Logik erster Stufe erfüllbar ist. Per Henkin-Elimination ist mit Φ auch $\Phi \cup \mathscr{H}$ konsistent — als Menge von Formeln in Logik erster Stufe, und damit erst recht als Menge von aussagenlogischen Formeln, da ja nach der Uminterpretation allenfalls weniger Information und weniger deduktive Mittel als vorher zur Verfügung stehen, um einen Widerspruch herzuleiten. Per Vollständigkeit der Aussagenlogik ist damit $\Phi \cup \mathscr{H}$ erfüllbar als Menge aussagenlogischer Formeln; nach der Henkinkonstruktion folgt dann, dass Φ auch als Menge von Formeln in Logik erster Stufe erfüllbar ist.

Im einzelnen werden die Schritte der Beweisstrategie wie folgt umgesetzt.

(A) (Henkin-Theorie) Wir brauchen für jede Formel $\phi(x)$ eine Zeugenkonstante $c_{\phi(x)}$, *aber*: Zeugenkonstanten erzeugen neue Formeln, z.B. $\exists y(P\left(y, c_{\phi(x)}\right))$. Die Lösung besteht darin, die Hinzufügung von Zeugenkonstanten zu iterieren. Wir definieren also eine aufsteigende Folge $\Sigma = \Sigma_0 \subseteq \Sigma_1 \subseteq \Sigma_2 \subseteq \ldots$ von Signaturen, wobei jeweils Σ_{i+1} die vorhergehende Signatur Σ_i um neue Zeugenkonstanten $c_{\phi(x)}$ für alle Σ_i-Formeln $\phi(x)$ (mit $FV(\phi) = \{x\}$) erweitert, und setzen $\Sigma_H = \bigcup_{i=0}^{\infty} \Sigma_i$. Wir können dann für jedes $c_{\phi(x)}$ seinen *Geburtstag*

$$\min\{i \mid c_{\phi(x)} \in \Sigma_i\}$$

definieren. Damit sieht man auch, dass Σ_H in der Tat für jede Formel $\phi(x)$ über Σ_H eine Zeugenkonstante $c_{\phi(x)}$ enthält: wenn i der Geburtstag der jüngsten Zeugenkonstante in $\phi(x)$ ist, dann enthält $\Sigma_{i+1} \subseteq \Sigma_H$ eine Zeugenkonstante $c_{\phi(x)}$.

Die Henkin-Theorie $\mathscr{H}$ ist dann definiert als die (unendliche) Menge aller Instanzen der folgenden beiden Axiomenschemata:

H1: $\exists x(\phi(x)) \rightarrow \phi(c_{\phi(x)})$

H2: $\phi(c) \rightarrow \exists x(\phi(x))$

(B) (Henkin-Elimination) Formeln das Typs H2 sind beweisbar in Logik erster Stufe und können daher offensichtlich in Beweisen in Logik erster Stufe als Annahmen weggelassen werden. Es bleibt zu zeigen, dass alle Instanzen von H1 eliminierbar sind. Wir wählen in einem gegebenen Beweis von ρ unter den verwendeten Instanzen von H1 die mit dem jüngsten $c_{\phi(x)}$. Wir bezeichnen die Menge aller anderen im Beweis verwendeten Instanzen von H1 mit $\mathscr{H}_0$, so dass

$$\Phi \cup \mathscr{H}_0 \cup \{\exists x(\phi(x)) \rightarrow \phi(c_{\phi(x)})\} \vdash \rho.$$

Da $\exists x(\phi(x)) \rightarrow \phi(c_{\phi(x)})$ sowohl aus $\neg\exists x(\phi(x))$ als auch aus $\phi(c_{\phi(x)})$ herleitbar ist, folgt

$$\Phi \cup \mathscr{H}_0 \cup \{\neg\exists x(\phi(x))\} \vdash \rho \text{ und } \Phi \cup \mathscr{H}_0 \cup \{\phi(c_{\phi(x)})\} \vdash \rho.$$

Da $c_{\phi(x)}$ in Φ, $\mathscr{H}_0$ und ρ nicht vorkommt (in $\mathscr{H}_0$ deswegen nicht, weil $c_{\phi(x)}$ die jüngste vorkommende Zeugenkonstante ist), folgt aus dem rechten Teil per ($\exists$E)

$$\Phi \cup \mathscr{H}_0 \cup \{\exists x(\phi(x))\} \vdash \rho,$$

also per Fallunterscheidung über $\exists x(\phi(x)) \vee \neg\exists x(\phi(x))$ letztlich $\Phi \cup \mathscr{H}_0 \vdash \rho$. Damit ist eine Instanz von H1 eliminiert; Iterieren des Verfahrens eliminiert alle Instanzen von H1.

(C) (Henkin-Konstruktion): Wir setzen

$$\begin{aligned} M_\kappa &= \text{Menge der Konstanten in } \Sigma_{\mathscr{H}} \\ \mathfrak{M}_\kappa[\![c]\!] &= c \\ \mathfrak{M}_\kappa[\![P]\!] &= \{(c_1, \ldots, c_n) \mid \kappa(P(c_1, \ldots, c_n)) = \top\} \end{aligned}$$

Die Behauptung $\mathfrak{M}_\kappa \models \Phi$ folgt dann unmittelbar aus

Lemma 94 (Wahrheitslemma)**.** *Für jeden Satz ρ gilt $\mathfrak{M}_\kappa \models \rho$ genau dann, wenn $\kappa \models \rho$.*

Beweis. Induktion über die Größe von ρ. Wir nehmen an, dass in ρ nur Existenzquantoren vorkommen; das ist für Zwecke eines Vollständigkeitsbeweises deswegen zulässig, weil die Äquivalenz von $\forall x(\phi)$ und $\neg\exists x(\neg\phi)$ in unserem System herleitbar ist und alle logischen Konstrukte herleitbare Äquivalenz bewahren. Der Induktionsanfang ist $\rho = P(c_1, \ldots, c_n)$ (da ρ ein Satz ist, kommen keine Variablen unter den Argumenten von P vor); für solche Formeln gilt die Behauptung nach Konstruktion von $\mathfrak{M}_\kappa$. Die Booleschen Fälle ($\neg$, $\wedge$) sind klar.

Es bleibt der Fall $\rho \equiv \exists x(\phi(x))$; wir handeln die Implikationen getrennt ab:

„$\Rightarrow$“*:* Sei $\mathfrak{M}_\kappa \models \exists x(\phi(x))$. Da jedes Element von M_κ die Interpretation einer Konstanten ist, existiert dann per Substitutionslemma $c \in \Sigma_{\mathscr{H}}$ mit $\mathfrak{M}_\kappa \models \phi(c)$. Da $\phi(c)$ ein Satz und kleiner als $\exists x(\phi(x))$ ist, folgt nach Induktionsvoraussetzung $\kappa \models \phi(c)$. Mit H2 erhalten wir schließlich $\kappa \models \exists x(\phi(x))$.

„$\Leftarrow$“*:* Sei $\kappa \models \exists x(\phi(x))$. Nach H1 folgt dann $\kappa \models \phi(c_{\phi(x)})$; da $\phi(c_{\phi(x)})$ ein Satz und kleiner als $\exists x(\phi(x))$ ist, folgt nach Induktionsvoraussetzung $\mathfrak{M}_\kappa \models \phi(c_{\phi(x)})$ und damit $\mathfrak{M}_\kappa \models \exists x(\phi(x))$. □

Es folgt nunmehr erneut

Korollar 95 (Kompaktheit)**.** *Jede* endlich erfüllbare *Formelmenge Φ (d.h. jede endliche Teilmenge von Φ ist erfüllbar) ist erfüllbar*

Beweis. Da Beweise endliche Objekte sind, gilt die analoge Eigenschaft mit „konsistent“ statt „erfüllbar“; nach Vollständigkeit und Korrektheit sind aber Konsistenz und Erfüllbarkeit äquivalent. □

Beispiel 96. Sei Φ eine Axiomatisierung der natürlichen Zahlen in Logik erster Stufe (z.B. die Peano-Axiome, mit 0 und Sukzessorfunktion s sowie einem Axiomenschema für Induktion). Dann ist $\Phi \cup \{c > s^n(0) | n \in \mathbb{N}\}$ endlich erfüllbar, somit nach Kompaktheit erfüllbar — d.h. wir können in Logik erster Stufe durch keine noch so raffinierte Axiomatisierung die Existenz von Nicht-Standard-Zahlen ausschließen.

Bemerkung 97. Vollständigkeit ist durchaus keine selbstverständliche Eigenschaft einer Logik. Man unterscheidet im allgemeinen zwischen *starker Vollständigkeit*, wie wir sie oben für Prädikatenlogik erster Stufe bewiesen haben, und *schwacher Vollständigkeit*. Eine Logik, mit logischer Folgerungsrelation $\models$, ist per Definition stark vollständig, wenn aus $\Phi \models \psi$ stets $\Phi \vdash \psi$ folgt (*Korrektheit* ist die umgekehrte Implikation), und schwach vollständig, wenn jede gültige Formel herleitbar ist, d.h. wenn aus $\models \psi$ stets $\vdash \psi$ folgt. Wenn eine Logik korrekt und stark vollständig ist und außerdem das Beweissystem *finitär* ist, d.h. wenn jede Regel nur endlich viele Prämissen hat (das schließt Regeln aus, die z.B. aus $\phi(n)$ für alle konkreten natürlichen Zahlen n die Formeln $\forall x \in \mathbb{N}(\phi(x)$ schließen), dann ist die Logik, mit der gleichen Argumentation wie oben, *kompakt*, d.h. jede endlich erfüllbare Formelmenge ist erfüllbar.

In *monadischer Prädikatenlogik zweiter Stufe* (MSO) hat man zusätzlich zu Quantoren über Individuenvariablen wie in Prädikatenlogik erster Stufe noch Variablen für Teilmengen des Grundbereichs, also für unäre Prädikate, und Quantoren darüber. Wir nennen die neuen Variablen *Mengenvariablen* und schreiben sie $P, Q, \ldots$; wir verwenden sie ansonsten wie Prädikatensymbole. Sei dann Φ die Formelmenge über der Signatur $\Sigma = \{0, suc_{\text{¸}}, <\}$ bestehend aus den Formeln

$$\forall P((P(0) \wedge \forall x(P(x) \to P(s(x)))) \to \forall x(P(x)))$$
$$\forall x(\neg s(x) = 0)$$
$$\forall x, y(s(x) = s(y) \to x = y)$$
$$\forall x((0 < x \vee 0 = x) \wedge \neg x < 0)$$
$$\forall x, y(s(x) < s(y) \leftrightarrow x < y).$$

Dann definiert Φ eindeutig (bis auf irrelevante Umbenennungen der Elemente des Grundbereichs) die natürlichen Zahlen mit der üblichen Lesart von $<$: das zweite und dritte Axiom stellen sicher, dass man lauter verschiedene Elemente $s^n(0)$ für $n \geqslant 0$ hat, das erste Axiom sorgt dafür, dass dies tatsächlich alle Elemente sind, und die letzten beiden Axiom definieren $<$ per Rekursion. Damit ist dann offenbar

$$\Phi \cup \{x < s^n(0) \mid n \in \mathbb{N}\}$$

endlich erfüllbar, aber nicht erfüllbar. Somit hat MSO keine stark vollständige finitäre Axiomatisierung. (Nach dem deutlich schwerer zu beweisenden Gödelschen Unvollständigkeitssatz ist MSO auch nicht schwach vollständig.)