Einführung in die Mathematische Logik und Theoretische Informatik

Ralf Schindler

geT$_{\mathrm{E}}$Xt von Martina Pfeifer

14. Juli 2017

Dieses Skript basiert auf Vorlesungen, die ich im WiSe 99-00 (Einf. i. d. theoretische Informatik), WiSe 01-02 (Logik 1), SoSe 02 (Grundbegriffe der Logik), WiSe 02-03 (Einf. i. d. theoretische Informatik), SoSe 03 (Grundbegriffe der Logik) an der Universität Wien und im WiSe 05-06 (Einf. i. d. math. Logik und theoretische Informatik) an der WWU Münster gehalten habe. Dabei habe ich vor allem die beiden folgenden sehr schönen Textbücher verarbeitet:

Herbert B. Enderton, *A mathematical introduction to logic*, San Diego, 1972 + 2001.

Michael Sipser, *Introduction to the theory of computation*, Boston, 1997.

Ralf Schindler

Inhaltsverzeichnis

Kapitel 0

Worum geht es in der Logik?

Betrachten Sie den folgenden Satz, nennen wir ihn S:

Dieser Satz ist nicht wahr.

Der Satz S stellt eine Behauptung auf. Ist S wahr oder falsch? Nun, wenn S wahr ist, dann ist S offensichtlich falsch, da S ja sagt, dass S *nicht* wahr ist; und wenn S falsch ist, dann ist S wahr, da S ja sagt, dass S *nicht wahr* ist. Damit ist dann wohl S weder wahr noch falsch, also vielleicht einfach unsinnig ... Aber ein unsinniger Satz ist ja wohl doch immerhin *nicht wahr* und damit stimmt dann doch, was S sagt, und S ist am Ende doch wahr? Ihre Gedanken fangen an, sich in einem Kreis zu bewegen. Vielleicht ist der Satz S *so* unsinnig, dass er eben wirklich ganz und gar sinnlos ist, so wie wohl die Begriffe "Hump" und "Dump"[1]. Aber ganz so einfach ist es nicht. Sie haben ja eben eine Weile lang durchaus logisch über den Satz S nachgedacht. Also scheint nichts zu passen: S ist weder wahr, noch falsch, noch unsinnig, noch ganz und gar sinnlos — also insbesondere ist S nicht wahr! Daher ...

Sie geben auf. Der Satz S ist eine verfeinerte Variante der "Lügner–Paradoxie", die in der Bibel zwar besprochen, aber nicht gelöst wurde[2]. Hier ist also ein Satz, S, der in geringfügigen Varianten schon seit mehr als 2000 Jahren durchdiskutiert wird; aber: Eine "Lösung der Lügner–Paradoxie" ist nicht in Aussicht und wird auch in diesem Skript nicht präsentiert werden. Während Sie über S nachgedacht haben, haben Sie jedenfalls *logisch* nachgedacht: In Ihren Gedanken hat sich eins aus dem anderen zwangsweise

[1] Siehe http://www.funfocus.net/bilder/cartoons/deix5.htm

[2] Titus, 1, 12 – 13: "Einer ihrer eigenen Propheten hat gesagt: 'Kreter sind von jeher verlogen, wilde Tiere, faule Bäuche'. Dies Zeugnis ist wahr".

ergeben. Die Logik geht diesem Zwang nach. Sie fragt: "Was bedeutet es, dass etwas *logisch* aus etwas anderem *folgt*?"

Sie haben nun zwei Möglichkeiten:

(1) Die Beschäftigung mit S hat Ihnen gereicht, Sie wollen Ihre Zeit sinnvoller verbringen und nichts weiter über Logik wissen.

(2) Wenn man Ihnen schon keine befriedigende Auskunft hinsichtlich des Satzes S geben kann, dann wollen Sie wenigstens einen Einblick bekommen, was die Logik in den letzten 2000 Jahren Positives erreicht hat (und insbesondere, was die mathematische Logik in den letzten 100 Jahren geleistet hat!). Im letzteren Falle sollten Sie dranbleiben.[3]

Der bedeutendste Logiker des 20. Jahrhunderts heißt KURT GÖDEL (*1906, †1978). Er wurde in Brünn geboren und kam dann nach Wien. Er bewies den so genannten Vollständigkeitssatz in seiner Dissertation und den sogenannten Unvollständigkeitssatz in seiner Habilitation. Diese Resultate haben die Logik revolutioniert. Er beschäftigte sich erfolgreich mit dem 1. HILBERTschen Problem. Es hat aber nichts genützt, er bekam keine feste Stelle in Wien. Er emigrierte 1938 in die USA. Er fuhr zunächst mit dem Zug nach Wladiwostok, dann von dort mit dem Schiff nach San Francisco und von dort weiter nach Princeton, NJ, wo er den Rest seines Lebens als Kollege von ALBERT EINSTEIN (*1897, †1955) am 'Institute for Advanced Studies' wirkte. In Princeton bewies Kurt Gödel, dass Einsteins Feldgleichungen eine Lösung besitzen, die Zeitreisen in die Vergangenheit möglich machen.

Wir wollen in dieser Vorlesung den Gödelschen Vollständigkeitssatz beweisen. Man kann leicht missverstehen, was dieser Satz sagt. Man kann es aber schön an einem Beispiel illustrieren.

Eine (additive) *Gruppe* ist ein Objekt der Gestalt

$$(G; 0, +),$$

wobei G eine nichtleere Menge, $0 \in G$ das neutrale Element und $+$ eine zweistellige Verknüpfung ist (die Addition von Elementen aus G), und wobei $(G; 0, +)$ die Gruppenaxiome erfüllt. Gruppentheoretiker möchten herausfinden, welche Aussagen (in der Sprache der Gruppentheorie) in allen Gruppen gelten. Z.B. ist das Zentrum einer beliebigen Gruppe immer eine Untergruppe dieser Gruppe (und diese Aussage lässt sich in der Sprache der

[3] Ach ja, es gibt eine dritte Möglichkeit: Sie wollen zwar eigentlich nichts weiter über Logik wissen, benötigen aber einen Schein.

Gruppentheorie formulieren). Wie zeigt man aber, dass eine solche Aussage in allen Gruppen gilt?

Nun, auf diese Frage gibt es spätestens seit EUKLID (ca. *325, †265 v.u.Z.) nur eine mögliche Antwort: Man *beweist* die fragliche Aussage aus den Gruppenaxiomen. Was aber ist ein Beweis? Wir haben oben davon gesprochen, dass sich (beim Nachdenken über den Satz S) in Ihren Gedanken eins aus dem anderen zwangsweise ergeben hat; Sie haben logische Folgerungen vollzogen. In der Logik (als Teil der Mathematik) wird nun zunächst dieser Folgerungsbegriff (d.h. der Begriff des Beweisens) präzisiert. Es wird sodann nicht schwer sein zu sehen, dass alles, was aus den Gruppenaxiomen beweisbar ist, auch in allen Gruppen gelten muss. Aber gilt auch die Umkehrung? D.h., *wenn eine Aussage in allen Gruppen gilt, haben wir dann die Methoden zur Hand, diese Aussage aus den Gruppenaxiomen streng zu beweisen?*

Der Gödelsche Vollständigkeitssatz beantwortet diese Frage positiv (und das nicht nur für die Gruppentheorie). Ich habe oben gesagt: Man kann leicht missverstehen, was der Vollständigkeitssatz sagt. Eigentlich sollte man sagen: Es liegt nicht unmittelbar auf der Hand, dass er überhaupt etwas sagt. Man muss die Fragestellung wirklich erfassen um zu sehen, welch tiefes Resultat er ist. Wir wollen uns dieser Fragestellung langsam (wenn auch zugegebenermaßen abstrakt!) annähern.

Wir werden im Zuge des Beweises des Gödelschen Unvollständigkeitssatzes ein Resultat kennen lernen, mit dessen Hilfe wir Nichtstandard–Modelle der Zahlentheorie konstruieren werden. Was eine Gruppe ist, das wird von den Gruppenaxiomen ausgedrückt. Kann es eine (natürliche) Menge von Axiomen geben, die uns erklärt, was die natürlichen Zahlen sind? Manche von Ihnen kennen vielleicht die Axiome der PEANO–Arithmetik. Die Menge der natürlichen Zahlen ist ein Modell dieses Axiomensystems, so wie jede Gruppe ein "Modell" der Gruppenaxiome ist. Es gibt viele Gruppen, aber es gibt nur eine Menge der natürlichen Zahlen. Wir werden sehen, dass die Peano–Arithmetik viele (zueinander nicht isomorphe) Modelle besitzt. Das mag auf den ersten Blick etwas paradox erscheinen, die natürlichen Zahlen sind doch etwas so "Konkretes", dass es eigentlich möglich sein müsste, sie vollständig zu beschreiben. Nun ja. Der Unterschied zur "Lügner–Paradoxie" ist der, dass hier alles "Paradoxe" verschwindet, sobald man die Sache verstanden hat.

Apropos: Sei Franz K. ein Kreter. Angenommen Franz K. sagt:

"Jeder Aussagesatz, den ein Kreter jemals sagt, ist nicht wahr".

Diese äußerung kann nicht wahr sein, da Franz K. selbst ein Kreter ist. Also muss sie falsch sein; d.h. Franz K. sagt in diesem Moment die Unwahrheit und es gibt äußerungen von Kretern, die wahr sind. Dies allerdings ist konsistent! Die "Lügner–Paradoxie" der Bibel ist also auch keine Paradoxie!

Bleibt der oben diskutierte Satz S als Problem. Er liefert immerhin die Idee zum Beweis des Gödelschen *Un*vollständigkeitssatzes. Gödel hat es geschafft, in der Sprache der Zahlentheorie einen Satz zu formulieren, der sagt: "Dieser Satz ist in der Zahlentheorie nicht beweisbar". Er kann nicht beweisbar sein, wenn die Zahlentheorie konsistent ist. Also ist er wahr. Es gibt also wahre nicht beweisbare Sätze in der Zahlentheorie.

Entsprechendes gilt für die meisten mathematischen Theorien, sogar für die Mengenlehre, die die stärksten Theorien der Mathematik liefert: Für jede ihrer Theorien gibt es wahre, jedoch nicht beweisbare Sätze.

Ein Verwandter des Gödelschen Unvollständigkeitssatzes ist der Unentscheidbarkeitssatz von Alonzo Church (*1903, †1995). Er besagt, dass es kein Entscheidungsverfahren gibt, mit dessen Hilfe wir errechnen können, ob eine vorgelegte Aussage der Zahlentheorie (oder der Mengenlehre) in der Zahlentheorie (bzw. einer Theorie der Mengenlehre) beweisbar ist.

Die mathematische Forschung sähe anders aus, wäre der Churchsche Satz falsch! Er besagt, dass es prinzipiell keinen Computer geben kann, der uns Mathematikern die Arbeit abnimmt.

Bevor Konrad Zuse (*1910, †1995) den ersten programmierbaren Rechner der Welt gebaut hatte, wurden die Möglichkeiten und Grenzen der Berechenbarkeit systematisch untersucht. Das mathematische Modell eines Computers geht auf Alan Turing (*1912, †1954) zurück, der im zweiten Weltkrieg an der Entschlüsselung der Enigma beteiligt war. 1952 hat ihn die rückständige Nachkriegszeit wegen Homosexualität ins Gefängnis gebracht und später in den Selbstmord getrieben.

Neben der Betrachtung der Gödelschen Sätze werden wir in dieser Vorlesung die Grundlagen der Theorie der Berechenbarkeit kennen lernen. Dabei werden wir auch sehen, dass prinzipiell (durch Berechnung) löbare Probleme nicht unbedingt auch praktisch lösbar sind, wenn nämlich etwa die Rechenzeit zu großist. Dies führt uns beispielsweise zur Frage, ob $P = NP$. Wir werden diese Frage formulieren, aber nicht beantworten. Falls Sie eine Antwort finden, dann sind Sie um 1 Mio. $ reicher[4]. Dies ist eine letzte Möglichkeit, inwiefern sich ein Besuch dieser Vorlesung lohnen könnte.

[4]Siehe http://www.claymath.org/millenium/P_vs_NP/

Kapitel 1

Aussagenlogik

Die ganze Logik ist in vier Teile geteilt: Beweistheorie, Rekursionstheorie (Theorie der Berechenbarkeit), Modelltheorie und die Mengenlehre. Die Theoretische Informatik ist nah an der Rekursionstheorie angesiedelt.

Die Begriffe des Beweisens und Rechnens sind grundlegend in der Mathematik. In der Logik werden sie systematisch untersucht. Beweise sind Beweise auf der Grundlage von Theorien. Theorien besitzen Modelle. Wir betrachten hier die grundlegenden Zusammenhänge zwischen Beweisen, Berechnungen und Modellen.

Die einfachsten Beweise sind Beweise in der Aussagenlogik. Beispiel:

Die Werse ist ein Berg oder ein Fluss.
Die Werse ist kein Berg.

Die Werse ist ein Fluss.

Mit A_0 = "Die Werse ist ein Berg" und A_1 = "Die Werse ist ein Fluss" sieht dieser "Beweis" (dieser logische Schluss) so aus:

A_0 oder A_1
nicht A_0

A_1

Die logischen *Junktoren* sind Zeichen für: *nicht, und, oder, wenn ... , dann ... , genau dann, wenn.*

"$\neg$" steht für "nicht",
"$\wedge$" steht für "und",
"$\vee$" steht für "oder",

"$\rightarrow$" steht für "wenn ... , dann ... ",
"$\leftrightarrow$" steht für "genau dann, wenn".

Unser logischer Schluss sieht dann so aus:

$$\begin{array}{l} A_0 \vee A_1 \\ \neg A_0 \\ \hline A_1 \end{array}$$

Dieser Schluss ist logisch, da es auf die Bedeutung der darin vorkommenden Aussagenvariablen nicht ankommt.

Die *Symbole* der Aussagenlogik sind die folgenden:

Klammern: (und)
Junktoren: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
Aussagenvariablen: $A_0, A_1, A_2 \ldots$

Ein *Ausdruck* ist eine endliche Folge von Symbolen. Z.B. ist $(\wedge A_3 \leftrightarrow)$ ein Ausdruck. Nicht alle Ausdrücke sind "Formeln", d.h. haben Bedeutung.

Welche Ausdrücke sind Formeln? Wir wollen:

(a) Jede Aussagenvariable ist eine Formel.

(b) Wenn φ und ψ Formeln sind, dann sind auch $\neg\varphi, (\varphi \wedge \psi), (\varphi \vee \psi), (\varphi \rightarrow \psi)$ und $(\varphi \leftrightarrow \psi)$ Formeln.

(c) Kein Ausdruck ist eine Formel, der es nicht auf Grund von (a) oder (b) sein muss.

Wie lässt sich dies mathematisch präziser fassen? Eine *Formel* der Aussagenlogik ist ein Ausdruck, der in jeder Menge M von Ausdrücken enthalten ist, so dass

(a)' jede Aussagenvariable in M enthalten ist, und

(b)' wenn φ und ψ in M enthalten sind, dann auch $\neg\varphi, (\varphi \wedge \psi), (\varphi \vee \psi), (\varphi \rightarrow \psi)$ und $(\varphi \leftrightarrow \psi)$.

Wenn wir (nur für diesen Zweck!) eine Menge M von Ausdrücken "gut" nennen genau dann, wenn M (a)' und (b)' erfüllt, dann gilt also: Ein Ausdruck ist eine Formel genau dann, wenn er im Durchschnitt aller "guten" Mengen von Ausdrücken liegt.

Die Aussagenvariablen heißen auch *atomare Formeln*, die übrigen Formeln *zusammengesetzte Formeln*. Eine zusammengesetzte Formel ist entweder Negation (d.h. von der Gestalt $\neg\varphi$), oder Konjunktion (von der Gestalt $(\varphi \wedge \psi)$), oder Disjunktion (von der Gestalt $(\varphi \vee \psi)$), oder Konditional (von der Gestalt $(\varphi \to \psi)$) oder Bikonditional (von der Gestalt $(\varphi \leftrightarrow \psi)$).

Die Definition des Formelbegriffs ist Beispiel für eine *rekursive Definition*. Es gilt das *Induktionsprinzip* für Formeln: Sei E eine beliebige Eigenschaft. Angenommen, E gilt für alle atomaren Formeln. Sei weiterhin Folgendes angenommen: wenn E auf die Ausdrücke φ und ψ zutrifft, dann gilt E auch für $\neg\varphi, (\varphi \wedge \psi), (\varphi \vee \psi), (\varphi \to \psi)$ und $(\varphi \leftrightarrow \psi)$. Dann gilt E für alle Formeln.

Hier ist ein Beispiel. Wir zeigen mit Hilfe des Induktionsprinzips, dass jede Formel die gleiche Anzahl von linken und rechten Klammern hat. Diese Aussage gilt nämlich offensichtlich für alle atomaren Formeln, und sie vererbt sich von φ und ψ weiter auf $\neg\varphi, (\varphi \wedge \psi), (\varphi \vee \psi), (\varphi \to \psi)$ und $(\varphi \leftrightarrow \psi)$. Also gilt die Aussage für alle Formeln.

Warum gilt das Induktionsprinzip? Sei M die Menge aller Ausdrücke, auf die E zutrifft. Dann gilt für M nach Voraussetzung (a)' und (b)'. Also ist (nach Definition dessen, was es heißt, eine Formel zu sein) die Menge aller Formeln eine Teilmenge von M; d.h., auf jede Formel trifft E zu!

Rekursive Definitionen sind in der Mathematik sehr häufig. Für sie alle gibt es ein entsprechendes Induktionsprinzip. Das Induktionsprinzip für $\mathbb{N}$ ist ein weiteres Beispiel. Wir werden im Folgenden viele Beispiele zur Rekursion und Induktion sehen.

Wir wollen nun definieren, was es heißt, dass eine Formel logisch aus einer anderen Formel folgt. Hierzu definieren wir zunächst den Begriff der Belegung; eine Belegung β ordnet jeder Formel φ einen "Wahrheitswert" $\beta(\varphi) \in \{0, 1\}$ zu. Dabei lesen wir $\beta(\varphi) = 0$ auch als "φ ist falsch" und $\beta(\varphi) = 1$ als "φ ist wahr".

Sei F_0 die Menge aller atomaren Formeln (d.h. $F_0 = \{A_0, A_1, A_2, \dots\}$), und sei $F \supset F_0$ die Menge aller Formeln. Sei $\overline{\beta} : F_0 \to \{0, 1\}$ eine beliebige Funktion, die allen atomaren Formeln "Wahrheitswerte" zuordnet. Wir definieren dann die *zu $\overline{\beta}$ gehörige Belegung* $\beta : F \to \{0, 1\}$ rekursiv wie folgt:

(1) $\beta(\varphi) = \overline{\beta}(\varphi)$ für alle $\varphi \in F_0$

und für alle $\varphi, \psi \in F$:

(2) $\beta(\neg\varphi) = 1 - \beta(\varphi)$,

(3) $\beta((\varphi \wedge \psi)) = \beta(\varphi) \cdot \beta(\psi) = \min\{\beta(\varphi), \beta(\psi)\}$

(4) $\beta((\varphi \vee \psi)) = \max\{\beta(\varphi), \beta(\psi)\}$

(5) $\beta((\varphi \to \psi)) = \max\{1 - \beta(\varphi), \beta(\psi)\}$

(6) $\beta((\varphi \leftrightarrow \psi)) = \beta((\varphi \to \psi)) \cdot \beta((\psi \to \varphi))$

Es gilt also Folgendes: $\neg\varphi$ ist wahr gdw.[1] φ falsch ist; $(\varphi \wedge \psi)$ ist wahr gdw. φ und ψ beide wahr sind; $(\varphi \vee \psi)$ ist wahr gdw. φ oder ψ wahr ist (im nicht ausschließenden Sinne); $(\varphi \to \psi)$ ist wahr gdw. $\neg\varphi$ oder ψ wahr ist; $(\varphi \leftrightarrow \psi)$ ist wahr gdw. φ und ψ dieselben Wahrheitswerte besitzen.

Zu jedem $\overline{\beta} : F_0 \to \{0,1\}$ ist dadurch eindeutig eine zugehörige Belegung $\beta : F \to \{0,1\}$ definiert.

Wir nennen $\beta(\varphi)$ auch den *zu $\overline{\beta}$ gehörigen Wahrheitswert von φ*. Für eine gegebene Formel errechnet sich der Wahrheitswert am leichtesten mit Hilfe einer "Wahrheitstafel". Sei etwa die Formel $(A_0 \to (\neg A_1 \to A_0))$ gegeben. Sei $\overline{\beta}(A_0) = 0$, $\overline{\beta}(A_1) = 1$. Dann ergibt sich für den zugehörigen Wahrheitswert $\beta((A_0 \to (\neg A_1 \to A_0)))$:

A_0	A_1	$(A_0$	$\to$	$(\neg$	A_1	$\to$	$A_0))$
0	1		1	0		1	

D.h. $\beta((A_0 \to (\neg A_1 \to A_0))) = 1$. Offensichtlich hängt der Wahrheitwert $\beta(\varphi)$ einer Formel φ nur von den (endlich vielen) Werten $\overline{\beta}(A_n)$ ab, für die die atomare Formel A_n in φ vorkommt. In unserem Falle errechnen sich die zu verschiedenen $\overline{\beta}$ gehörigen Wahrheitswerte von $(A_0 \to (\neg A_1 \to A_0))$ wie folgt:

A_0	A_1	$(A_0$	$\to$	$(\neg$	A_1	$\to$	$A_0))$
0	0		1	1		0	
0	1		1	0		1	
1	0		1	1		1	
1	1		1	0		1	

Sei $\overline{\beta} : F_0 \to \{0,1\}$. Wir sagen, dass $\overline{\beta}$ die Formel φ *erfüllt* gdw. $\beta(\varphi) = 1$, wobei β die zu $\overline{\beta}$ gehörige Belegung ist. Sei Σ eine Menge von Formeln. (D.h. $\Sigma \subset F$.) Wir sagen, dass $\overline{\beta}$ die Formelmenge Σ *erfüllt* gdw. $\beta(\varphi) = 1$ für alle $\varphi \in \Sigma$, wobei β die zu $\overline{\beta}$ gehörige Belegung ist.

Definition 1.1 *Sei $\Sigma \cup \{\varphi\}$ eine Menge von Formeln.* Σ impliziert tautologisch φ, *in Zeichen $\Sigma \models \varphi$, gdw. für alle $\overline{\beta} : F_0 \to \{0,1\}$ Folgendes gilt: wenn $\overline{\beta}$ Σ erfüllt, dann erfüllt $\overline{\beta}$ auch φ. Anstelle von $\emptyset \models \varphi$ schreiben wir auch $\models \varphi$; in diesem Falle nennen wir φ eine* Tautologie.

[1] "gdw." steht für "genau dann wenn"

Die oben betrachtete Formel $(A_0 \to (\neg A_1 \to A_0))$ ist also eine Tautologie. Hier sind einige Beispiele für $\Sigma \models \varphi$:

$$\{\neg A_1\} \models A_1 \to A_0,$$
$$\{A_3 \to A_5, A_5 \to A_3\} \models A_3 \leftrightarrow A_5,$$
$$\{\neg A_0 \vee A_1\} \models A_0 \to A_1.$$

Die Menge Σ in der obigen Definition muss übrigens nicht endlich sein. Wir schreiben auch $\psi \models \varphi$ anstelle von $\{\psi\} \models \varphi$. ψ und φ heißen *tautologisch äquivalent* gdw. $\psi \models \varphi$ und $\varphi \models \psi$ gelten.

Sei $J \subset \{\neg, \wedge, \vee, \to, \leftrightarrow\}$ eine Menge von Junktoren. J heißt *vollständig* gdw. es zu jeder Formel φ eine tautologisch äquivalente Formel ψ gibt, in der nur Junktoren aus J vorkommen. Z.B. ist $\{\neg\}$ nicht vollständig. Es gilt aber:

Satz 1.2 *Die folgenden Mengen J von Junktoren sind vollständig.*

(1) $J = \{\neg, \wedge, \vee\}$,

(2) $J = \{\neg, \wedge\}$,

(3) $J = \{\neg, \vee\}$,

(4) $J = \{\neg, \to\}$.

Beweis von (1): Wir konstruieren rekursiv eine Funktion $\Phi : F \to F$, die jeder Formel φ eine tautologisch äquivalente Formel $\Phi(\varphi)$ zuordnet, in der höchstens die Junktoren $\neg, \wedge, \vee$ vorkommen.

Wir setzen $\Phi(\varphi) = \varphi$ für jedes atomare φ. Seien $\Phi(\varphi)$ und $\Phi(\varphi')$ bereits definiert. Dann setzen wir

$$\Phi(\neg\varphi) = \neg\Phi(\varphi),$$
$$\Phi((\varphi \wedge \varphi')) = (\Phi(\varphi) \wedge \Phi(\varphi')),$$
$$\Phi((\varphi \vee \varphi')) = (\Phi(\varphi) \vee \Phi(\varphi')),$$
$$\Phi((\varphi \to \varphi')) = \neg\Phi(\varphi) \vee \Phi(\varphi'),$$
$$\Phi((\varphi \leftrightarrow \varphi')) = \Phi((\varphi \to \varphi')) \wedge \Phi((\varphi' \to \varphi)).$$

Damit ist $\Phi : F \to F$ (durch "Rekursion längs der Formelkomplexität") wohldefiniert.

Man überlegt sich leicht induktiv, dass φ und $\Phi(\varphi)$ immer tautologisch äquivalent sind. Für atomares φ ist das trivial. Sei nun etwa φ ein Konditional, etwa φ gleich $(\psi \to \psi')$. Dann sind nach Induktionsvoraussetzung ψ

und $\Phi(\psi)$ sowie ψ' und $\Phi(\psi')$ jeweils tautologisch äquivalent. Damit sind aber auch $\neg\psi$ und $\neg\Phi(\psi)$ sowie schließlich $(\neg\psi \vee \psi')$ und

$$(\neg\Phi(\psi) \vee \Phi(\varphi') = \Phi((\psi \to \psi')))$$

tautologisch äquivalent. Es sind aber $(\psi \to \psi')$ (d.h. φ) und $(\neg\psi \vee \psi')$ tautologisch äquivalent. □

Im Beweis dieses Satzes sahen wir ein Beispiel für eine Definition durch "Rekursion längs der Formelkomplexität" und ein Beispiel für den Beweis einer Aussage durch "Induktion nach der Formelkomplexität".

Wir beweisen nun den Kompaktheitssatz der Aussagenlogik. Sei Σ eine Menge von Formeln. Σ heißt *erfüllbar* gdw. es ein $\overline{\beta} : F_0 \to \{0,1\}$ gibt, so dass $\overline{\beta}$ jedes φ aus Σ erfüllt. Σ heißt *endlich* erfüllbar gdw. jede endliche Teilmenge $\overline{\Sigma}$ von Σ erfüllbar ist. Offensichtlich ist jedes erfüllbare Σ auch endlich erfüllbar.

Satz 1.3 Kompaktheitssatz. *Sei Σ eine Menge von Formeln. Wenn Σ endlich erfüllbar ist, dann ist Σ auch erfüllbar.*

Beweis: Sei $(\varphi_n : n \in \mathbb{N})$ eine Aufzählung *aller* Formeln. Eine solche Aufzählung erhält man z.B., indem man im m^{ten} Schritt alle Formeln der Länge $\leq m$ aufzählt, in denen höchstens die Aussagenvariablen $A_0, \ldots, A_{m-1}$ vorkommen.

Sei nun Σ endlich erfüllbar. Wir konstruieren nun rekursiv eine Folge $(\Sigma_n : n \in \mathbb{N})$ von Formelmengen Σ_n wie folgt. Setze $\Sigma_0 = \Sigma$. Sei nun Σ_n konstruiert. Setze dann $\Sigma_{n+1} = \Sigma_n \cup \{\varphi_n\}$, falls $\Sigma_n \cup \{\varphi_n\}$ endlich erfüllbar ist; ansonsten setze $\Sigma_{n+1} = \Sigma_n \cup \{\neg\varphi_n\}$. Schließlich sei Σ_∞ die Vereinigung aller Σ_n. Wir zeigen zunächst durch Induktion, dass alle Σ_n endlich erfüllbar sind. Dies gilt nach Annahme für $n = 0$. Sei nun Σ_n endlich erfüllbar. Falls $\Sigma_n \cup \{\varphi_n\}$ endlich erfüllbar ist, dann ist der Induktionsschritt trivial. Sei also o.B.d.A. $\Sigma_n \cup \{\varphi_n\}$ nicht endlich erfüllbar und $\Sigma_{n+1} = \Sigma_n \cup \{\neg\varphi_n\}$. Sei $\overline{\Sigma} \subset \Sigma_n \cup \{\neg\varphi_n\}$ endlich. Sei $\overline{\Sigma}' \subset \Sigma_n \cup \{\varphi_n\}$ endlich und nicht erfüllbar.

Nach Induktionsvoraussetzung existiert ein $\overline{\beta} : F_0 \to \{0,1\}$, das $(\overline{\Sigma} \cup \overline{\Sigma}')\backslash\{\varphi_n, \neg\varphi_n\}$ erfüllt. Sei β die zu $\overline{\beta}$ gehörige Belegung. Da Σ_n endlich erfüllbar und $\overline{\Sigma}'$ nicht erfüllbar ist, muss φ_n in $\overline{\Sigma}'$ enthalten sein und $\beta(\varphi_n) = 0$. Dann gilt aber $\beta(\neg\varphi_n) = 1$ und $\overline{\beta}$ erfüllt $(\overline{\Sigma} \cup \overline{\Sigma}' \cup \{\neg\varphi_n\})\backslash\{\varphi_n\}$, also auch $\overline{\Sigma}$.

Damit ist nun sofort auch Σ_∞ endlich erfüllbar. Nach Konstruktion haben wir, dass für jede Formel φ entweder φ oder $\neg\varphi$ in Σ_∞ liegt.

Wir definieren nun ein $\overline{\beta} : F_0 \to \{0,1\}$ wie folgt. Wir setzen $\overline{\beta}(A_n) = 1$ gdw. A_n in Σ_∞ enthalten ist.

Sei $\beta : F \to \{0,1\}$ die zu $\overline{\beta}$ gehörige Belegung. Wir zeigen jetzt induktiv, dass $\beta(\varphi) = 1$ gdw. φ in Σ_∞ enthalten ist. Dies gilt zunächst für atomare Formeln φ. Sei die Aussage nun für φ gezeigt. Wir zeigen dann, dass sie auch für $\neg\varphi$ gilt. Es gilt aber $\beta(\neg\varphi) = 1$ gdw. $\beta(\varphi) = 0$ gdw. φ nicht in Σ_∞ enthalten ist gdw. $\neg\varphi$ in Σ_∞ enthalten ist. Sei die Aussage für φ und ψ gezeigt. Sie gilt dann auch für $(\varphi \wedge \psi) : \beta((\varphi \wedge \psi)) = 1$ gdw. $\beta(\varphi) = 1 = \beta(\psi)$ gdw. φ und ψ beide in Σ_∞ sind gdw. $(\varphi \wedge \psi)$ in Σ_∞ ist. Letztere äquivalenz ergibt sich leicht aus der endlichen Erfüllbarkeit von Σ_∞. ähnlich argumentiert man für die übrigen zusammengesetzten Formeln. □

Aus Satz 1.3 ergibt sich unmittelbar:

Korollar 1.4 *Sei $\Sigma \cup \{\varphi\}$ eine Menge von Formeln. Wenn $\Sigma \models \varphi$, dann gibt es ein endliches $\overline{\Sigma} \subset \Sigma$ mit $\overline{\Sigma} \models \varphi$.*

Diese Aussage ist sogar äquivalent zu 1.3.

Kapitel 2

Turing–Maschinen

Mit Hilfe der Wahrheitstafelmethode können Sie entscheiden, ob eine vorgelegte Formel der Aussagenlogik eine Tautologie ist oder nicht. Eine solche Entscheidung könnte Ihnen ein Computer abnehmen.

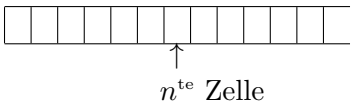
Eine (aussagenlogische) Formel φ heißt *erfüllbar* gdw. $\neg\varphi$ keine Tautologie ist, d.h. wenn es ein $\overline{\beta} : F_0 \to \{0,1\}$ gibt, das φ erfüllt. Wir definieren

$$SAT = \{\varphi : \varphi \text{ ist eine erfüllbare Formel}\}.$$

Wir wollen nun sehen, dass (in einem mathematisch präzisen Sinne) SAT (d.h. Elementschaft in SAT) entscheidbar ist. Später werden wir sehen, dass SAT "NP–vollständig" ist.

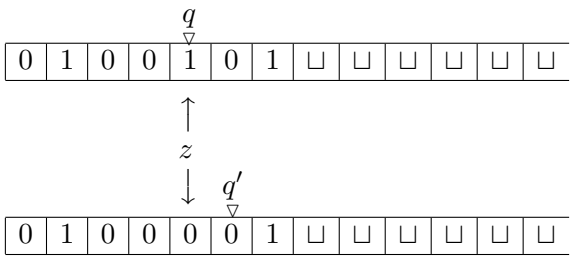
Im praktischen Leben verhalten sich Computer teilweise scheinbar chaotisch. In der Mathematik ist das anders: Computer folgen hier strengen Regeln. (Wir werden später auch "nicht–deterministische" Computer kennen lernen.)

Das mathematische Modell eines Computers ist die Turing–Maschine. Eine Turing–Maschine führt gemäßeines Programms Berechnungen auf einem Blatt Papier durch. Das "Blatt Papier" wird auch als *Schreibband* bezeichnet und sieht so aus:



Genau für jede natürliche Zahl n existiert eine n^{te} Zelle auf dem Schreibband; diese Zelle kann leer sein oder mit einem Symbol eines festen *Alphabets* Γ beschrieben sein. Die Turing–Maschine besitzt einen Kopf, der zu einem gegebenen Rechenzeitpunkt auf einer Zelle des Schreibbandes steht. Im nächsten

Rechenschritt wird die Turing–Maschine abhängig von der Inschrift der Zelle z, auf der der Kopf steht, und abhängig vom gegenwärtigen *Zustand* der Maschine die Zelle z neu beschrieben, mit dem Kopf einen Schritt nach rechts oder links gehen und in einen neuen Zustand geraten:



Formal besteht für uns eine *Turing–Maschine* $\top$ aus den folgenden Objekten:

(1) Einer endlichen Menge Q von *Zuständen*, wobei es drei ausgezeichnete Zustände gibt: den *Anfangszustand* $q_0 \in Q$, den *positiven Endzustand* q_+ und den *negativen Endzustand* q_-.

(2) Ein endliches *Alphabet* Γ, d.h. eine Menge von Symbolen, die das Leerzeichen $\sqcup$ enthalten soll; eine nichtleere Teilmenge Σ von Γ, die nicht $\sqcup$ enthält, ist als *Eingabealphabet* ausgezeichnet.

(3) Eine *übergangsfunktion*

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\},$$

die Paaren (q, a) mit $q \in Q$ und $a \in \Gamma$ jeweils ein Tripel (q', b, x) mit $q' \in Q, b \in \Gamma$ und $x \in \{L, R\}$ zuordnet.

Ein *Rechenvorgang* der Turing–Maschine $\top$ ist eine (endliche oder unendliche) Folge von *Konfigurationen* K_n. Die n^{te} Konfiguration ist gegeben durch:

(a) Einen Zustand $q \in Q$, in dem sich $\top$ nach n Rechenschritten befindet; dabei ist $q = q_0$ für $n = 0$, d.h. nach 0 Rechenschritten ist $\top$ im Anfangszustand.

(b) Eine Position des Kopfes, der auf einer Zelle des Rechenbandes steht; für $n = 0$ steht der Kopf am linken Ende des Bandes.

(c) Ein endliches *Wort*, das auf dem Rechenband niedergeschrieben ist; im Falle $n = 0$ ist dieses das *Eingabewort* (oder die *Eingabe*): für ein $l \in \mathbb{N}$ sind nach 0 Rechenschritten die ersten l Zellen mit jeweils einem Symbol aus Σ beschrieben.

Insbesondere liest der Kopf nach n Rechenschritten ein Symbol a (u.U. $\sqcup$), mit dem die Zelle, auf der er steht, beschrieben ist.

Der n^{te} Rechenschritt ist nun durch δ gegeben. Für $(q', b, x) = \delta(q, a)$ ist q' der Zustand, in dem $\top$ nach $n+1$ Rechenschritten ist; das Wort, das nach $n+1$ Rechenschritten auf dem Band steht, ergibt sich aus dem Wort, das nach n Rechenschritten auf dem Band steht, indem das Symbol a in der Zelle, auf der der Kopf nach n Schritten stand, durch das Symbol b ersetzt wird; der Kopf schließlich geht um einen Schritt nach links oder rechts, je nachdem, ob $x = L$ oder $x = R$ (falls der Kopf am linken Bandende steht, so bleibt er bei $x = L$ dort stehen).

Jeder Rechenvorgang von $\top$ ist offensichtlich durch die Eingabe determiniert. Ein Rechenvorgang bricht nach n Rechenschritten ab, falls $\top$ dann im Zustand q_+ oder im Zustand q_- ist. Im ersteren Fall sagen wir dann, dass $\top$ die Eingabe *akzeptiert*, im letzteren Fall sagen wir, dass $\top$ die Eingabe *verwirft*. (Es gibt dann keinen n^{ten} Rechenschritt.)

Falls $\top$ eine gegebene Eingabe weder verwirft, noch akzeptiert, dann ist der zugehörige Rechenvorgang ergebnislos und unendlich lange.

Sei w eine Eingabe. Wir schreiben

$$\top(w) \downarrow +, \text{ falls } \top w \text{ akzeptiert, und } \top(w) \downarrow -, \text{ falls } \top w \text{ verwirft.}$$

Falls der Rechenvorgang von $\top$ bei Eingabe von w nicht abbricht, so schreiben wir $\top(w) \uparrow$.

Wir wollen uns nun einige Beispiele ansehen. Es ist nicht auf Anhieb zu glauben, dass alles, was überhaupt "berechenbar" ist, mit Hilfe einer Turing–Maschine berechenbar ist. Dies wird aber die Aussage der These von Church sein.

Wir wollen zunächst eine Turing–Maschine $\top$ bauen, die entscheidet, ob die Eingabe eine gerade Anzahl von Nullen enthält.

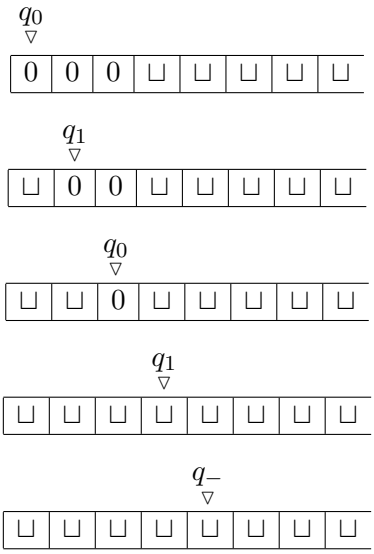
Sei $Q = \{q_0, q_1, q_+, q_-\}$. Sei etwa $\Sigma = \{0\}$ und $\Gamma = \{0, \sqcup\}$. Das Eingabealphabet hat also die Null als einziges Symbol. Die übergangsfunktion δ werde folgendermaßen beschrieben:

q_0	0	q_1	$\sqcup$	R
q_1	0	q_0	$\sqcup$	R
q_0	$\sqcup$	q_+	$\sqcup$	R
q_1	$\sqcup$	q_-	$\sqcup$	R

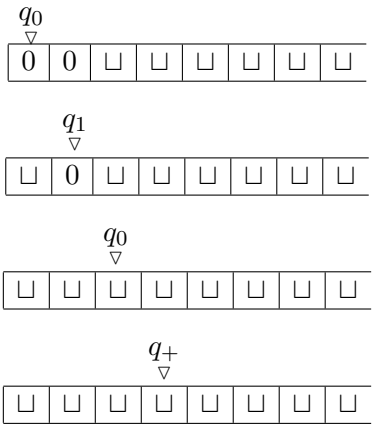
$\top$ oszilliert zwischen den Zuständen q_0 und q_1 hin und her, wobei der Kopf in jedem Schritt nach rechts geht, solange, bis der Kopf das Eingabeende,

d.h. $\sqcup$ erreicht. In diesem Fall wird akzeptiert/verworfen gdw. $\top$ eine gerade/ungerade Anzahl von Malen zwischen q_0 und q_1 oszilliert ist.

Ein typischer Rechenvorgang sieht wie folgt aus:



Die Eingabe 000 wird also verworfen. Die Eingabe 00 wird aber akzeptiert:



Etwas schwieriger ist es, eine Turing–Maschine zu konstruieren, die entscheidet, ob die Eingabe 2^n Nullen enthält, wobei $n \in \mathbb{N}$. Die Idee ist die Folgende. Der Kopf liest wiederholt die Eingabe von links nach rechts, wobei jede zweite 0 durchgestrichen wird. Wenn dann eine ungerade Anzahl von

Nullen auf dem Band bleibt, dann wird die Eingabe verworfen (es sei denn, es stand am Anfang genau eine 0 auf dem Band).

Es sei $Q = \{q_0, q_1, q_2, q_3, q_4, q_+, q_-\}, \Sigma = \{0\}$ und $\Gamma = \{0, x, \sqcup\}$. Die übergangsfunktion δ werde wie folgt beschrieben.

q_0	$\sqcup$	q_-	$\sqcup$	R
q_0	0	q_1	$\sqcup$	R
q_1	$\sqcup$	q_+	$\sqcup$	R
q_1	x	q_1	x	R
q_1	0	q_2	x	R
q_2	x	q_2	x	R
q_2	$\sqcup$	q_4	$\sqcup$	L
q_4	0	q_4	0	L
q_4	x	q_4	x	L
q_4	$\sqcup$	q_1	$\sqcup$	R
q_2	0	q_3	0	R
q_3	x	q_3	x	R
q_3	0	q_2	x	R
q_3	$\sqcup$	q_-	$\sqcup$	R

Da die Maschine im Allgemeinen die Eingabe mehrfach lesen muss, ist es erforderlich, das linke Bandende beim Start zu *markieren*, so dass später beim Zurücksetzen des Kopfes erkannt werden kann, wann das linke Bandende erreicht ist. Dies geschieht, indem beim Start ein $\sqcup$ in die linkeste Zeile geschrieben wird.

Beim Nach–rechts–Gehen oszilliert die Maschine zwischen q_2 und q_3, wobei bereits durchgestrichene Nullen (d.h. x) ignoriert werden und jede zweite Null neu durchgestrichen wird. Es dauert sicherlich einige Zeit, bis man sich in die Vorgehensweise dieser Turing–Maschine eingedacht hat!

Ein Wort im Alphabet Γ ist eine endliche Folge von Symbolen aus Γ. In jedem Rechenschritt ist (genau) ein Wort auf dem Schreibband niedergeschrieben. Das Wort endet mit dem ersten $\sqcup$, so dass nur noch weitere $\sqcup$ folgen.

Wir bezeichnen die Menge aller Worte im Alphabet Γ mit Γ^*. Eine Menge $L \subset \Gamma^*$ heißt auch *Sprache*. Die beiden grundlegenden Begriffe der Berechenbarkeitstheorie sind die folgenden:

Sei $L \subset \Sigma^*$ eine Sprache, wobei Σ eine nichtleere endliche Menge von Symbolen ist. L heißt *rekursiv aufzählbar* (oder *Turing–erkennbar*) gdw. es eine Turing–Maschine $\top$ mit Eingabealphabet Σ gibt, so dass für alle $w \in \Sigma^*$:

$$w \in L \Leftrightarrow \top(w) \downarrow +.$$

L heißt *rekursiv* (oder *Turing–entscheidbar*, oder einfach *entscheidbar*) gdw. es eine Turing–Maschine $\top$ mit Eingabealphabet Σ gibt, so dass für alle $w \in \Sigma^*$:

$$w \in L \Leftrightarrow \top(w) \downarrow +, \text{ und}$$
$$w \notin L \Leftrightarrow \top(w) \downarrow -.$$

Offensichtlich ist jede entscheidbare Sprache rekursiv aufzählbar. Wir werden allerdings rekursiv aufzählbare Sprachen kennen lernen, die nicht entscheidbar sind (etwa das Halteproblem).

Der Beweis der folgenden Aussage ist nicht schwierig.

Lemma 2.1 *Eine Sprache $L \subset \Sigma^*$ ist Turing-entscheidbar gdw. sowohl L als auch $\Sigma^* \setminus L$ rekursiv aufzählbar ist.*

Beweisskizze: Sei sowohl L als auch $\Sigma^* \backslash L$ rekursiv aufzählbar. Sei $\top_0$ eine Turing–Maschine, die bezeugt, dass L rekursiv aufzählbar ist, und sei $\top_1$ eine Turing–Maschine, die bezeugt, dass $\Sigma^* \backslash L$ rekursiv aufzählbar ist. Wenn wir $\top_0$ und $\top_1$ gleichzeitig laufen lassen, dann erhalten wir bzgl. eines beliebigen $w \in \Sigma^*$ irgendwann eine Entscheidung, ob $w \in L$ oder $w \in \Sigma^* \backslash L$. Man kann nun recht einfach eine Turing–Maschine $\top$ bauen, die bei Eingabe von $w \in \Sigma^*$ in der n^{ten} Runde zunächst die ersten n Rechenschritte von $\top_0$ und sodann die ersten n Rechenschritte von $\top_1$ simuliert. □

Wir wollen nun zeigen, dass SAT entscheidbar ist. Offensichtlich können wir jede (aussagenlogische) Formel φ als Wort im Alphabet $\{(,), \neg, \wedge, \vee, \rightarrow, \leftrightarrow, *\}$ auffassen; dabei identifizieren wir die Aussagenvariable A_n mit der Folge von $n+1$ vielen $*$. Aus $(A_3 \rightarrow (A_0 \vee A_1))$ wird also das Wort $(*** \rightarrow (* \vee **))$. Teil des SAT–Problems ist es zu entscheiden, ob ein gegebener (aussagenlogischer) Ausdruck eine Formel ist. Hier ist eine Turing–Maschine $\top$ die entscheidet, ob im gegebenen Ausdruck eine Teilformel der Gestalt

$$(A_n \wedge A_m), \text{ d.h. } (* \ldots * \wedge * \ldots *)$$

vorkommt.

ein beliebiges q	(	q_1	(	R
q_1	$*$	q_2	$*$	R
q_2	$*$	q_2	$*$	R
q_2	$\wedge$	q_3	$\wedge$	R
q_3	$*$	q_4	$*$	R
q_4	$*$	q_4	$*$	R
q_4	)	q_+	)	R
ein beliebiges q	$\sqcup$	q_-	$\sqcup$	R
ein sonstiges Paar q	x	q_0	x	R

$\top$ hat also die 7 Zustände $q_0, \ldots, q_4, q_+, q_-$. Der Kopf läuft von links nach rechts, ohne die Zelleneintragungen zu ändern. Man erkennt unschwer, dass $\top$ die Eingabe akzeptiert (verwirft) gdw. eine Teilformel der Form $(* \ldots * \wedge * \ldots *)$ (nicht) vorkommt.

Wir können $\top$ leicht so variieren, dass $\top$ anstelle zu akzeptieren zunächst das gefundene Vorkommnis von $(* \ldots * \wedge * \ldots *)$ durch die "atomare Formel" $** \ldots *** \ldots **$ ersetzt:

ein beliebiges q	(	q_1	(	R
q_1	$*$	q_2	$*$	R
q_2	$*$	q_2	$*$	R
q_2	$\wedge$	q_3	$\wedge$	R
q_3	$*$	q_4	$*$	R
q_4	$*$	q_4	$*$	R
q_4	)	q_5	$*$	L
q_5	$*$ oder $\wedge$	q_5	$*$	L
q_5	(	q_+	$*$	L
ein beliebiges q	$\sqcup$	q_-	$\sqcup$	R
ein sonstiges Paar q	x	q_0	x	R

Anstatt zu akzeptieren könnte diese neue Maschine auch mit dem Kopf zum linken Bandende zurücklaufen und von neuem starten und sehen, ob eine weitere Teilformel der Gestalt $(* \ldots * \wedge * \ldots *)$ im neu entstandenen Ausdruck auftritt und im positiven Falle diese Teilformel abermals durch $** \ldots *** \ldots **$ ersetzen.

Wir können schließlich sogar eine Turing–Maschine $\top_F$ bauen, die in jedem Lesedurchgang des Kopfes von links nach rechts nachsieht, ob eine Teilformel der Gestalt $\neg * \ldots *, (* \ldots * \wedge * \ldots *), (* \ldots * \vee * \ldots *), (* \ldots * \rightarrow * \ldots *)$ oder $(* \ldots * \leftrightarrow * \ldots *)$ im (jeweils neu entstandenen) Ausdruck auftritt und im positiven Falle diese Teilformel durch $** \ldots *$ bzw. $** \ldots *** \ldots **$

(d.h. durch eine "atomare Formel") ersetzt. Damit $\top_F$ keinen Unsinn produziert, sollte $\top_F$ allerdings in einem allerersten Lesedurchgang nachsehen, ob die Eingabe keine Teilfolgen der Form $*($ oder $)*$ enthält.[1] Falls nun dieser Prozess solange weiterläuft bis am Ende ein Ausdruck der Gestalt $* \ldots *$ (d.h. eine "atomare Formel") auf dem Band steht, dann akzeptiert $\top_F$ (und genau dann war die Eingabe eine (aussagenlogische) Formel); ansonsten verwirft $\top_F$ die Eingabe.

Ein kleines Problem hierbei ist wiederum: wie findet der Kopf von $\top_F$ jeweils das linke Bandende wieder? Eine einfache Lösung ist abermals, die linkeste Zelle sofort bei Rechenstart zu *markieren*, d.h. etwa

$$(,), \neg, \wedge, \vee, \rightarrow, \leftrightarrow, *$$

durch

$$(',)', \neg', \wedge', \vee', \rightarrow', \leftrightarrow', *'$$

zu ersetzen, wobei die Symbole $(', \ldots, *'$ nur für diesen Markierungszweck vorbehalten bleiben. Sobald der Kopf dann später ein Symbol der Form $(', \ldots, *'$ liest, erkennt $\top_F$, dass der Kopf am Bandanfang steht.

Wir haben also eine Turing–Maschine T_F mit folgender Eigenschaft gebaut: Für jeden (aussagenlogischen) Ausdruck w gilt $\top_F(w) \downarrow +$ gdw. w eine Formel ist, und $\top_F(w) \downarrow -$ gdw. w keine Formel ist. D.h. $\top_F$ bezeugt, dass die Menge aller Formeln entscheidbar ist.

Wenn wir nun entscheiden wollen, ob ein gegebenes φ eine erfüllbare Formel ist, so werden wir zunächst $\top_F$ entscheiden lassen, ob φ eine Formel ist; allerdings sollten wir in Wahrheit eine Variante von $\top_F$ rechnen lassen, die φ "intakt" lässt, d.h. nicht durch $* \ldots *$ ersetzt, da sich ansonsten φ nicht mehr rekonstruieren lässt. Es ist leicht, eine solche Variante von $\top_F$ zu bauen. Wenn dann der Formeltest positiv verlaufen ist, so wollen wir als nächstes entscheiden, ob φ erfüllbar ist. Dazu benutzen wir im Wesentlichen die Methode der Wahrheitstafeln: für jedes $\overline{\beta} : F_0 \rightarrow \{0, 1\}$ wollen wir ausrechnen, ob die zugehörige Belegung $\beta : F \rightarrow \{0, 1\}$ das gegebene φ mit 0 (falsch) oder 1 (wahr) belegt. Sobald ein $\overline{\beta}$ mit $\beta(\varphi) = 1$ gefunden ist, akzeptieren wir φ. Falls niemals ein solches $\overline{\beta}$ gefunden wird, verwerfen wir φ. Da es bei $\overline{\beta}$ nur auf die Werte von Aussagenvariablen ankommt, die in φ tatsächlich vorkommen, kann tatsächlich nach endlich vielen Rechenschritten eine Entscheidung gefunden werden. Seien etwa $A_{i_0}, \ldots, A_{i_{n-1}}$ die in φ vorkommenden Aussagenvariablen. Es gibt dann 2^n Möglichkeiten, wie ein $\overline{\beta} : F_0 \rightarrow \{0, 1\}$ diese Aussagenvariablen mit Wahrheitswerten belegt.

[1] Andernfalls würde $\top_F$ z.B. den Ausdruck $*(** \wedge ***)$ akzeptieren.

Die Lösung des SAT–Problems erfordert eine gute Buchhaltung. Wir bauen jetzt eine Turing–Maschine, die nach und nach die Zahlen von 0 bis 2^n in Dualdarstellung auf das Band schreibt und dann hält. Sei

$$\Sigma = \{\#, 0\}, \Gamma = \{\#, 0, 1, \sqcup\}.$$

$\#$ dient zur Markierung des linken Bandendes. Wir starten mit der Eingabe $\#0 \dots 0$, d.h. $\#$, gefolgt von n Nullen. Die Maschine sei gegeben durch

	q_0	$\#$	q_0	$\#$	R
	q_0	0	q_0	0	R
	q_0	1	q_0	1	R
	q_0	$\sqcup$	q_1	$\sqcup$	L
	q_1	0	q_2	1	L
	q_1	1	q_3	0	L
	q_2	0	q_2	0	L
	q_2	1	q_2	1	L
	q_3	0	q_2	1	L
	q_3	1	q_3	0	L
q_1 oder	q_2	$\#$	q_0	$\#$	R
	q_3	$\#$	q_+	$\#$	L

Der Kopf geht zunächst nach rechts, wonach die Maschine schriftlich 1 addiert. Dabei entspricht der Zustand q_2 "Null gemerkt" und der Zustand q_3 "Eins gemerkt". Die Maschine hält, sobald der Kopf am linken Bandende bei "Eins gemerkt" steht. Die Bandinschrift lautet dann $\#1 \dots 1$, d.h. $\#$, gefolgt von n Einsen.

Wir beschreiben schließlich eine Turing–Maschine $\top_{SAT}$, die das SAT–Problem entscheidet.

Es werde φ eingegeben. Zunächst lässt $\top_{SAT}$ die Maschine $\top_F$ laufen und entscheidet, ob φ eine Formel ist. Im negativen Falle verwirft $\top_{SAT}$ die Eingabe. Sei nun diese erste Entscheidung positiv.

Zunächst markiert jetzt $\top_{SAT}$ das linke Bandende (dies ist wohl ohnehin durch $\top_F$ bereits geschehen). Sodann kopiert $\top_F$ die Formel φ nach rechts, vom Urbild etwa durch eine weitere Markierung $\#$ getrennt. Dann schreibt $\top_{SAT}$ weiter rechts davon, etwa nochmals durch $\#$ (oder besser $\#\#$) getrennt, eine Folge $0 \dots 0$ von n Nullen, wobei etwa einfach n die Länge von φ sei. (Wenn die Aussagenvariablen A_i, d.h. $i+1$ viele $*$, in φ vorkommen, dann ist $i < n$.) Wir benutzen diesen hintersten Teil zur Buchführung bzgl. der abgehandelten Belegungen der Aussagenvariablen $A_0, \dots, A_{n-1}$ mit Wahrheitswerten 0, bzw. 1.

Auf dem Band sieht es nun so aus:

$$\varphi\#\varphi\#\#0\ldots0.$$

Zu einem späteren analogen Zeitpunkt der Berechnung sieht es auf dem Band so aus:

$$\varphi\#\varphi\#\#d,$$

wobei d die Dualdarstellung einer Zahl zwischen 0 und $2^n - 1$ ist.

Wir ersetzen nun gemäßd jedes Vorkommnis von $A_i, i < n$, (d.h. jedes weder nach links, noch nach rechts verlängerbare Vorkommnis einer $*$–Folge der Länge $i + 1$) in der Kopie von φ zwischen # und ## durch 0, bzw. 1. Sodann berechnen wir den sich ergebenden Wahrheitswert. Nach und nach ersetzen wir

$\neg 0$	durch	1
$\neg 1$		0
$(0 \wedge 0)$		0
$(0 \wedge 1)$		0
$(1 \wedge 0)$		0
$(1 \wedge 1)$		1

usw.

Falls sich am Ende der Wert 1 ergibt, so akzeptieren wir die ursprüngliche Eingabe φ; wir haben dann eine Belegung gefunden, die φ erfüllt.

Falls sich 0 ergibt, so ersetzen wir diese 0 wieder durch φ und erhöhen den "Zähler" d um 1. Falls der Zählerstand gleich 2^n ist und sich auch in diesem letzten Falle als Wahrheitswert von φ die 0 ergibt, so verwerfen wird die ursprüngliche Eingabe; es gibt dann keine Belegung, die φ erfüllt.

Wir haben also gesehen, wie sich eine Turing–Maschine $\top_{SAT}$ bauen lässt, so dass $\top_{SAT}$ bezeugt, dass SAT entscheidbar ist.

Das Problem SAT lässt sich also mit relativ primitiven Mitteln lösen. Die *These von Church* sagt: wenn sich ein Problem im intuitiven Sinne durch Rechnen (d.h. durch einen Algorithmus irgendwelcher Art) lösen lässt, dann lässt es sich mit Hilfe einer Turing–Maschine lösen.

Kapitel 3

Logik erster Stufe

Nicht alle logischen Schlüsse sind aussagenlogische Schlüsse. Betrachten wir das Beispiel des Sokrates–Schülers Aristoteles:

> Alle Flüsse fließen ins Meer.
> Die Aa ist ein Fluss .
> ---
> Die Aa fließt ins Meer.

In der Logik erster Stufe lässt sich dieser Schluss nachvollziehen. Analog zu den aussagenlogischen Tautologien wird es "logische Wahrheiten" geben, d.h. (logisch) gültige Formeln (der Logik erster Stufe). Die Menge der gültigen Formeln ist zwar rekursiv aufzählbar, aber nicht entscheidbar. Dies wird sich aus den Gödelschen Sätzen ergeben.

Die Logik erster Stufe ist die Logik, die wir in der Mathematik benutzen. Allerdings ist es nicht ganz richtig, von der Logik erster Stufe im Singular zu sprechen, da wir mit verschiedenen Mengen von Prädikatsymbolen, Konstanten und Funktoren arbeiten können.

Seien I, K, J (womöglich leere, womöglich unendlich große) paarweise disjunkte Indexmengen, und sei $n : I \cup J \to \mathbb{N}$; wir schreiben n_i für $n(i)$, wobei $i \in I \cup J$. Die zu I, K, J, n gehörige Sprache der Logik erster Stufe besitzt die folgenden *Symbole*:

> Klammern: (und)
> Junktoren: $\neg$ und $\to$
> Allquantor: $\forall$
> Variablen: $v_0, v_1, v_2, \ldots$
> Gleichheitszeichen: $=$
> Prädikatsymbole: für jedes $i \in I$ ein n_i–stelliges Prädikatsymbol P_i

Konstanten: für jedes $k \in K$ eine Konstante c_k
Funktoren: für jedes $j \in J$ einen n_j–stelligen Funktor f_j.

Ein Beispiel für eine solche Sprache ist die Sprache der Gruppentheorie: diese enthält einen zweistelligen Funktor für die Addition (bzw. Multiplikation) und eine Konstante für das neutrale Element. Die Sprache der Körpertheorie enthält zwei zweistellige Funktoren für die Addition und die Multiplikation und zwei Konstanten für die 0 und die 1. Die Sprache der *elementaren Zahlentheorie* enthält ein zweistelliges Prädikatsymbol $<$, eine Konstante 0 für die Null, einen einstelligen Funktor S für die Nachfolgeroperation $n \mapsto n+1$ und drei zweistellige Funktoren $+, \cdot, E$ für die Addition, die Multiplikation und die Exponentiation.[1] Die Sprache der Mengenlehre schließlich hat lediglich das zweistellige Prädikatsymbol $\in$.

Sei $\mathcal{L}$ eine Sprache der Logik erster Stufe. Ein *$\mathcal{L}$–Ausdruck* ist eine endliche Folge von Symbolen von $\mathcal{L}$. Wir wollen nun die Begriffe "$\mathcal{L}$–Term" und "$\mathcal{L}$–Formel" definieren. Dies geschieht wieder rekursiv. Sei $\mathcal{L}$ durch I, K, J, n gegeben.

Ein *$\mathcal{L}$–Term* ist ein $\mathcal{L}$–Ausdruck, der in jeder Menge M von $\mathcal{L}$–Ausdrücken liegt mit:

(a) jede Konstante und jede Variable liegt in M, und

(b) für $j \in J$ und $\tau_0, \dots \tau_{n_j-1} \in M$ ist auch $f_j\tau_0 \dots \tau_{n_j-1} \in M$.

Beispielsweise sind die folgenden Ausdrücke Terme der elementaren Zahlentheorie:

$$+S0v_3,\ SSS0,\ \cdot SS0SSS0.$$

Wir werden natürlich $S(0) + v_3$ (oder besser $1 + v_3$) für $+S0v_3$ schreiben, usw.

Eine *atomare $\mathcal{L}$–Formel* ist jeder Ausdruck der Form $P\tau_0 \dots \tau_{n-1}$, wobei P ein n–stelliges Prädikatsymbol ist und $\tau_0, \dots, \tau_{n-1}$ $\mathcal{L}$–Terme sind, oder ein Ausdruck der Form $\tau_0 = \tau_1$, wobei τ_0, τ_1 $\mathcal{L}$–Terme sind.

Schließlich ist eine *$\mathcal{L}$–Formel* ein Ausdruck, der in jeder Menge M von Ausdrücken liegt mit:

(1) jede atomare $\mathcal{L}$–Formel ist in M,

(2) wenn φ und ψ in M sind, dann auch $\neg\varphi$ und $(\varphi \to \psi)$,

(3) wenn φ in M ist und $n \in \mathbb{N}$, dann ist auch $\forall v_n \varphi$ in M.

[1] Diese Sprache werden wir später mit $\mathcal{L}_A$ bezeichnen.

Beispiele für Formeln der Sprache der Mengenlehre sind etwa $\in v_3 v_0$ oder $\forall v_3 \in v_3 v_0$. Wir werden natürlich $v_3 \in v_0$ anstelle von $\in v_3 v_0$ und entsprechend $\forall v_3\ v_3 \in v_0$ anstelle von $\forall v_3 \in v_3 v_0$ schreiben. $v_3 \in v_0$ ist eine atomare Formel der Sprache der Mengenlehre.

Wir vereinbaren einige Schreibkonventionen, um Formeln besser lesen zu können. So schreiben wir:

$$\begin{array}{lll} (\varphi \vee \psi) & \text{für} & (\neg\varphi \to \psi), \\ (\varphi \wedge \psi) & \text{für} & \neg(\varphi \to \neg\psi), \\ (\varphi \leftrightarrow \psi) & \text{für} & ((\varphi \to \psi) \wedge (\psi \to \varphi)), \\ \exists v_n \varphi & \text{für} & \neg\forall v_n \neg\varphi, \\ \tau = \tau' & \text{für} & = \tau\tau', \text{ und} \\ \tau \neq \tau' & \text{für} & \neg\tau = \tau', \end{array}$$

wobei φ, ψ Formeln und τ, τ' Terme sind. Diese abkürzenden Schreibweisen sind sinnvoll: für die ersten drei Zeilen überlegt man sich leicht, dass, wären φ und ψ aussagenlogische Formeln, die jeweiligen Formeln (also $(\varphi \vee \psi)$ und $(\neg\varphi \to \psi)$, usw.) tautologisch äquivalent wären.

Mit diesen Konventionen ist nun also auch $\forall v_0 \exists v_1\ v_0 \in v_1$ eine Formel der Sprache der Mengenlehre, bei der es sich in nicht abgekürzter Schreibweise um die Formel

$$\forall v_0 \neg \forall v_1 \neg \in v_0 v_1$$

handelt. Als weitere Schreibkonvention lassen wir von nun an die äußeren Klammern weg, schreiben also z.B. $\varphi \to \psi$ anstelle von $(\varphi \to \psi)$.

Wir wollen nun wieder definieren, wann eine Formelmenge Σ eine Formel φ impliziert. Wieder arbeiten wir mit Belegungen, diesmal aber von Variablen mit Elementen eines Modells; dies entspricht der Einsetzung solcher Elemente für freie Variablen in Formeln.

Wir fixieren nun I, K, J, n, wodurch eine Sprache $\mathcal{L}$ gegeben ist. Anstelle von $\mathcal{L}$–Termen und (atomaren) $\mathcal{L}$–Formeln sprechen wir nun auch einfach von Termen und Formeln.

Wir definieren zunächst den Begriff der "freien Variablen".

Sei $n \in \mathbb{N}$. Wir definieren "v_n kommt frei in der Formel φ vor". Wenn φ atomar ist, dann kommt v_n frei in φ vor gdw. v_n überhaupt in φ vorkommt. v_n kommt frei in $\neg\varphi$ vor gdw. v_n frei in φ vorkommt. v_n kommt frei in $\varphi \to \psi$ vor gdw. v_n frei in φ oder frei in ψ vorkommt. Schließlich kommt v_n frei in $\forall v_m \varphi$ vor gdw. $m \neq n$ und v_n frei in φ vorkommt.

Ein *$\mathcal{L}$–Satz* ist eine *$\mathcal{L}$–Formel*, in der keine freien Variablen vorkommen.

Beispielsweise ist $\forall v_1 \neg v_1 \in v_0$ eine Formel der Sprache der Mengenlehre (aber kein Satz) und $\exists v_0 \forall v_1 \neg v_1 \in v_0$ ein Satz der Sprache der Mengenlehre.

Wir wollen jetzt den Sätzen der Sprache $\mathcal{L}$ Bedeutung verleihen. Oft hat eine Sprache einen "intendierten Objektbereich" im Auge. Mit der Sprache der elementaren Zahlentheorie wollen wir über die Struktur der natürlichen Zahlen sprechen, mit der Sprache der Mengenlehre über die Struktur des Universums aller Mengen.

Ebenso oft gibt es aber auch keinen intendierten Objektbereich: die Sprache der Gruppentheorie hat nicht eine feste Gruppe im Auge, sondern "beliebige Gruppen".

Allgemein bekommen Sätze Bedeutung, indem wir ein Modell von $\mathcal{L}$ betrachten. Ein *Modell von* $\mathcal{L}$ ist eine Struktur der Form

$$\mathcal{M} = (M; (R_i : i \in I), (a_k : k \in K), (F_j : j \in J)),$$

wobei M eine nichtleere Menge ist (das *Universum*, oder die *Trägermenge* von $\mathcal{M}$), jedes R_i eine n_i–stellige Relation auf M ist (d.h. $R_i \subset M^{n_i}$), jedes a_k ein Element von M ist und jedes F_j eine n_j–stellige Funktion auf M ist (d.h. $F_j : M^{n_j} \to M$).

Ein solches Modell interpretiert $\mathcal{L}$ in offensichtlicher Weise: das Prädikatsymbol P_i wird durch die Relation R_i interpretiert (in Zeichen: $P_i^{\mathcal{M}} = R_i$), die Konstante c_k wird durch a_k interpretiert (in Zeichen: $c_k^{\mathcal{M}} = a_k$), und der Funktor f_j wird durch die Funktion F_j interpretiert (in Zeichen: $f_j^{\mathcal{M}} = F_j$).

Wir wollen nun sagen, was es heißt, dass ein gegebener Satz im Modell $\mathcal{M}$ gilt. Hierzu gehen wir einen kleinen (aber sehr effektiven) Umweg und definieren allgemeiner, was es heißt, dass eine gegebene Formel unter einer gegebenen Belegung in $\mathcal{M}$ gilt.

Eine *$\mathcal{M}$–Belegung* ist eine Funktion $\overline{\beta} : \{v_0, v_1, \dots\} \to M$, die allen Variablen Elemente der Trägermenge von $\mathcal{M}$ zuordnet. Durch eine $\mathcal{M}$–Belegung ist eine Interpretation beliebiger Terme gegeben. Eine solche durch $\overline{\beta}$ induzierte *Terminterpretation* ist eine Funktion $\beta : T \to M$ (wobei T die Menge aller Terme ist), für die gilt:

(1) $\beta(v_n) = \overline{\beta}(v_n)$ für $n \in \mathbb{N}$,

(2) $\beta(c_k) = a_k$ für $k \in K$, und

(3) $\beta(f_j \tau_0 \dots \tau_{n_j - 1}) = F_j(\beta(\tau_0), \dots, \beta(\tau_{n_j - 1}))$ für $j \in J$ und Terme $\tau_0, \dots, \tau_{n_j - 1}$.

Wir definieren nun "φ gilt in $\mathcal{M}$ unter der Belegung $\overline{\beta}$". Hierzu ist die folgende Schreibweise hilfreich.

Sei $n \in \mathbb{N}$, und sei $a \in M$. Dann bezeichnet $\overline{\beta}(v_n|a)$ diejenige Belegung, die mit $\overline{\beta}$ überall übereinstimmt, außer an der Stelle v_n, wo der Wert a angenommen wird, d.h.

$$\overline{\beta}(v_n|a)(v_m) = \begin{cases} \overline{\beta}(v_m), \text{ falls } m \neq n \\ a \qquad\quad , \text{ falls } m = n. \end{cases}$$

Wir schreiben "φ gilt in $\mathcal{M}$ unter der Belegung $\overline{\beta}$" oder "$\mathcal{M}$ ist Modell von φ unter $\overline{\beta}$" als

$$\mathcal{M} \models \varphi[\overline{\beta}]$$

und definieren diese Relation wie folgt. Hierbei sei β die durch $\overline{\beta}$ induzierte Terminterpretation. Die folgende Definition ist rekursiv "nach der Formelkomplexität" und simultan für alle Belegungen.

(1) $\mathcal{M} \models \tau_0 = \tau_1[\overline{\beta}]$ gdw. $\beta(\tau_0) = \beta(\tau_1)$ für Terme τ_0, τ_1

(2) $\mathcal{M} \models P_i\tau_0 \dots \tau_{n_j-1}[\overline{\beta}]$ gdw. $(\beta(\tau_0), \dots, \beta(\tau_{n_j-1})) \in R_i$

(3) $\mathcal{M} \models \neg\varphi[\overline{\beta}]$ gdw. $\mathcal{M} \models \varphi[\overline{\beta}]$ nicht gilt

(4) $\mathcal{M} \models \varphi \to \psi[\overline{\beta}]$ gdw. $\mathcal{M} \models \varphi[\overline{\beta}]$ nicht gilt oder $\mathcal{M} \models \psi[\overline{\beta}]$ gilt

(5) $\mathcal{M} \models \forall v_n\varphi[\overline{\beta}]$ gdw. für alle $a \in M$, $\mathcal{M} \models \varphi[\overline{\beta}(v_n|a)]$.

Betrachten wir den Satz $\forall v_0\ v_0 = v_0$. Es gilt $\mathcal{M} \models \forall v_0\ v_0 = v_0[\overline{\beta}]$ gdw. für alle $a \in M$, $\mathcal{M} \models v_0 = v_0[\overline{\beta}(v_0|a)]$ gdw. für alle $a \in M$, $\beta(v_0) = \beta(v_0)$ (d.h. $\overline{\beta}(v_0) = \overline{\beta}(v_0)$). $\forall v_0\ v_0 = v_0$ gilt also in jedem Modell unter jeder Belegung.

Definition 3.1 *Sei $\Sigma \cup \{\varphi\}$ eine Menge von Formeln. Wir sagen, dass Σ (logisch) φ impliziert, in Zeichen: $\Sigma \models \varphi$ gdw. für jedes Modell $\mathcal{M}$ und für jede Belegung $\overline{\beta}$ gilt: wenn $\mathcal{M} \models \psi[\overline{\beta}]$ für alle $\psi \in \Sigma$, dann auch $\mathcal{M} \models \varphi[\overline{\beta}]$. Anstelle von $\emptyset \models \varphi$ schreiben wir auch $\models \varphi$ und sagen in diesem Falle, dass φ (logisch) gültig ist.*

Für jede Menge Σ von Formeln schreiben wir $\mathcal{M} \models \Sigma[\overline{\beta}]$ anstelle von: für alle $\varphi \in \Sigma$, $\mathcal{M} \models \varphi[\overline{\beta}]$. Es gilt also $\Sigma \models \varphi$ gdw. für jedes Modell $\mathcal{M}$ und für jede Belegung $\overline{\beta}$ gilt: wenn $\mathcal{M} \models \Sigma[\overline{\beta}]$, dann $\mathcal{M} \models \varphi[\overline{\beta}]$.

Der oben betrachtete Satz $\forall v_0\ v_0 = v_0$ ist also gültig. Dasselbe gilt übrigens auch für $\exists v_0\ v_0 = v_0$, das abkürzend die Formel $\neg\forall v_0\ \neg v_0 = v_0$ darstellt. Dies ist leicht nachzurechnen. Hingegen ist $\exists v_0 \exists v_1\ v_0 \neq v_1$ nicht gültig, aber es gilt z.B.

$$\{\exists v_0 \exists v_1 \exists v_2 (v_0 \neq v_1 \wedge v_1 \neq v_2 \wedge v_0 \neq v_2)\} \models \exists v_0 \exists v_1 v_0 \neq v_1.$$

Wir schreiben $\varphi \models \psi$ anstelle von $\{\varphi\} \models \psi$. φ und ψ heißen *(logisch) äquivalent* gdw. $\varphi \models \psi$ und $\psi \models \varphi$ gelten.

Satz 3.2 *Sei φ eine $\mathcal{L}$–Formel, und sei $\mathcal{M}$ ein Modell von $\mathcal{L}$. Seien $\overline{\beta}_0$ und $\overline{\beta}_1$ $\mathcal{M}$–Belegungen, so dass $\overline{\beta}_0(v_n) = \overline{\beta}_1(v_n)$ für alle v_n, die frei in φ vorkommen. Dann gilt*

$$\mathcal{M} \models \varphi[\overline{\beta}_0] \text{ gdw. } \mathcal{M} \models \varphi[\overline{\beta}_1].$$

Beweis: Wir zeigen diesen Satz durch Induktion "nach der Formelkomplexität". Die Aussage gilt zunächst für atomares φ, da dann v_n frei in φ vorkommt gdw. v_n überhaupt in φ vorkommt. Sodann ergibt sich aus der Tatsache, dass die Aussage für ψ_0 und ψ_1 gilt, sofort, dass die Aussage auch für φ gleich $\neg\psi_0$ und für φ gleich $\psi_0 \to \psi_1$ gilt.

Sei nun φ gleich $\forall v_m \psi$. v_n kommt frei in φ vor gdw. $n \neq m$ und v_n frei in ψ vorkommt. Für ein beliebiges a gilt also

$$\overline{\beta}_0(v_m|a)(v_n) = \overline{\beta}_1(v_m|a)(v_n)$$

für alle v_n, die frei in ψ vorkommen.

Damit gilt aber mit Hilfe der Induktionsvoraussetzung: $\mathcal{M} \models \varphi[\overline{\beta}_0]$ gdw. für alle $a \in M$, $\mathcal{M} \models \psi[\overline{\beta}_0(v_m|a)]$ gdw. für alle $a \in M$, $\mathcal{M} \models \psi[\overline{\beta}_1(v_m|a)]$ gdw. $\mathcal{M} \models \varphi[\overline{\beta}_1]$. □

Korollar 3.3 *Sei φ ein $\mathcal{L}$–Satz, und sei $\mathcal{M}$ ein Modell von $\mathcal{L}$. Seien $\overline{\beta}_0$ und $\overline{\beta}_1$ beliebige $\mathcal{M}$-Belegungen. Dann gilt*

$$\mathcal{M} \models \varphi[\overline{\beta}_0] \text{ gdw. } \mathcal{M} \models \varphi[\overline{\beta}_1].$$

Für Sätze φ schreiben wir im Folgenden $\mathcal{M} \models \varphi$ anstelle von $\mathcal{M} \models \varphi[\overline{\beta}]$ und sagen, dass φ *in $\mathcal{M}$ gilt* (oder, dass $\mathcal{M}$ *Modell von* φ ist).

Eine $\mathcal{L}$–Forml φ heißt *erfüllbar* gdw. $\neg\varphi$ nicht logisch gültig ist, d.h. wenn es ein Modell $\mathcal{M}$ und eine $\mathcal{M}$–Belegung $\overline{\beta}$ gibt, so dass $\mathcal{M} \models \varphi[\overline{\beta}]$. Die Erfüllbarkeit aussagenlogischer Formeln ist — wie wir gesehen haben — Turing–entscheidbar. Wir werden später sehen, dass hingegen die Erfüllbarkeit von $\mathcal{L}$–Formeln *nicht* Turing–entscheidbar ist (dies ist der Satz von Church, aus dem wir dann auch den ersten Gödelschen Unvollständigkeitssatz ableiten werden).

Der Gödelsche Vollständigkeitssatz auf der anderen Seite sagt, dass die Menge der logisch gültigen $\mathcal{L}$–Formeln Turing–erkennbar ist.

Sei $\mathcal{M}$ ein Modell von $\mathcal{L}$. Sei φ eine Formel, in der genau die Variablen $v_{i_0}, \ldots, v_{i_{n-1}}$ frei vorkommen. Sei $a_{i_0}, \ldots, a_{i_{n-1}} \in M$, der Trägermenge von $\mathcal{M}$. Wir schreiben dann kurz

$$\mathcal{M} \models \varphi(a_{i_0}, \ldots, a_{i_{n-1}})$$

anstelle von: $\mathcal{M} \models \varphi[\overline{\beta}]$, wobei $\overline{\beta}(v_{i_j}) = a_{i_j}$ für alle $j < n$.

Eine n–stellige Relation $R \subset M^n$ heißt *über* $\mathcal{M}$ *definierbar* gdw. es eine Formel φ gibt, in der genau die Variablen $v_0, \ldots, v_{n-1}$ frei vorkommen, so dass für alle $a_0, \ldots, a_{n-1} \in M$:

$$(a_0, \ldots, a_{n-1}) \in R \text{ gdw. } \mathcal{M} \models \varphi(a_0, \ldots, a_{n-1}).$$

Für $\mathcal{M} = (M; (R_i : i \in I), (c_k : k \in K), (F_j : j \in J))$ sind trivialerweise alle Relationen R_i über $\mathcal{M}$ definierbar; es gibt aber im Allgemeinen noch weitere definierbare Relationen.

Betrachten wir die Struktur

$$\mathcal{N} = (\mathbb{N}; <, 0, S, +, \cdot, E),$$

wobei $<$ die Kleiner–Relation, 0 die Null, S die Nachfolgeroperation $n \mapsto n + 1$, $+$ die Addition, $\cdot$ die Multiplikation und E die Exponentiation ist. Offenbar ist $\mathcal{N}$ ein Modell der Sprache der elementaren Zahlentheorie[2] (das "Standardmodell"; wir werden sehen, dass es noch weitere Modelle gibt). Es ist nicht wirklich nötig, $<$ zu $\mathcal{N}$ hinzuzunehmen: die Relation $<$ ist über dem Modell

$$(\mathbb{N}; 0, +)$$

mit Hilfe der Formel $\exists v_2(v_2 \neq 0 \wedge v_0 + v_2 = v_1)$ definierbar:

$$n < m \text{ gdw. } (\mathbb{N}; 0, +) \models \exists v_2(v_2 \neq 0 \wedge n + v_2 = m).$$

Wir werden später sehen, dass die Exponentiation über

$$(\mathbb{N}; <, 0, S, +, \cdot)$$

definierbar ist, d.h. dass es eine Formel φ gibt, so dass

$$n^m = q \text{ gdw. } (\mathbb{N}; <, 0, S, +, \cdot) \models \varphi(n, m, q).$$

[2]Damit bezeichnet $<$ sowohl das Symbol für die Kleiner–Relation als auch die Relation selbst, usw.

Dies wird einige Arbeit erfordern.

Auf der anderen Seite ist z.B. die Addition nicht über

$$(\mathbb{N}; \cdot)$$

und die Multiplikation nicht über

$$(\mathbb{N}; <, 0, S, +)$$

definierbar. Ersteres lässt sich folgendermaßen zeigen.

Seien $\mathcal{M}$ und $\mathcal{P}$ zwei Modelle von $\mathcal{L}$ mit Trägermenge M bzw. P. Eine Abbildung $\pi\colon M \to P$ heißt ein *Homomorphismus von $\mathcal{M}$ nach $\mathcal{P}$* gdw.

(1) $(a_0, ..., a_{n-1}) \in P^{\mathcal{M}}$ gdw. $(\pi(a_0), ..., \pi(a_{n-1})) \in P^{\mathcal{P}}$,

(2) $\pi(f^{\mathcal{M}}(a_0, ..., a_{n-1})) = f^{\mathcal{P}}(\pi(a_0), ..., \pi(a_{n-1}))$, und

(3) $\pi(c^{\mathcal{M}}) = c^{\mathcal{P}}$

für alle Prädikatsymbole P, alle Funktoren f, alle a_0, ..., $a_{n-1} \in M$ und für alle Konstanten c ist (hierbei ist n, abhängig von P bzw. f, jeweils die Stelligkeit von P bzw. f). Eine Abbildung $\pi\colon M \to P$ heißt ein *Isomorphismus von $\mathcal{M}$ nach $\mathcal{P}$* gdw. π ein Homomorphismus von $\mathcal{M}$ nach $\mathcal{P}$ ist. Die folgende Aussage ist leicht zu zeigen.

Satz 3.4 *Sei $\pi\colon M \to P$ ein Homomorphismus von $\mathcal{M}$ nach $\mathcal{P}$. Sei $\bar{\beta}$ eine M-Belegung, und sei β die durch $\bar{\beta}$ induzierte Terminterpretation. Sei die P-Belegung $\bar{\beta}'$ definiert durch $\bar{\beta}'(v_i) = \pi(\bar{\beta}(v_i))$ für $i \in \mathbb{N}$, und sei β' die durch $\bar{\beta}'$ induzierte Terminterpretation. Dann haben wir:*

(a) Für alle Terme τ ist $\pi(\beta(\tau)) = \beta'(\tau)$.

(b) Wenn π ein Isomorphismus von $\mathcal{M}$ nach $\mathcal{P}$ ist, dann gilt für alle Formeln φ, dass $\mathcal{M} \models \phi[\bar{\beta}]$ gdw. $\mathcal{P} \models \phi[\bar{\beta}']$.

Die äquivalenz in (b) bedeutet, dass

$$\mathcal{M} \models \phi(a_{i_0}, ..., a_{i_{n-1}}) \text{ gdw. } \mathcal{P} \models \phi(\pi(a_{i_0}), ..., \pi(a_{i_{n-1}})),$$

wenn in der Formel φ die Variablen $v_{i_0}, ..., v_{i_{n-1}}$ frei vorkommen und $\bar{\beta}(v_{i_0}) = a_{i_0}$, ..., $\bar{\beta}(v_{i_{n-1}}) = a_{i_{n-1}}$.

Angenommen nun, die Addition wäre über der Struktur $(\mathbb{N}; \cdot)$ definierbar. Dann hätten wir für jeden Isomorphismus[3] $\pi\colon \mathbb{N} \to \mathbb{N}$ von $(\mathbb{N}; \cdot)$ nach $(\mathbb{N}; \cdot)$,

[3] d.h. Automorphismus

dass $\pi(n+m) = \pi(n) + \pi(m)$ für alle n, $m \in \mathbb{N}$. Ein Isomorphismus π ist aber wie folgt gegeben. Sei $p_0 = 2$, $p_1 = 3$, ... die natürliche Aufzählung aller Primzahlen. Setze $\pi(0) = 0$, $\pi(1) = 1$, $\pi(2) = 3$, $\pi(3) = 2$ und $\pi(p_i) = p_i$ für alle $i \geq 2$. Für natürliche Zahlen $n \geq 2$ mit Primfaktorzerlegung

$$n = \prod p_{i_0}^{k_0} \cdot \ldots \cdot p_{i_{m-1}}^{k_{m-1}}$$

ist

$$\pi(n) = \prod \pi(p_{i_0})^{k_0} \cdot \ldots \cdot \pi(p_{i_{m-1}})^{k_{m-1}}.$$

Es gilt aber dann $\pi(1+1) = \pi(2) = 3 \neq 2 = 1 + 1 = \pi(1) + \pi(1)$.

Definition 3.5 *Die Klasse der primitiv rekursiven (p.r.) Funktionen ist die kleinste Klasse K von Funktionen $f : A \to \mathbb{N}$, wobei $A \subset \mathbb{N}^k$ für ein $k > 0$, so dass gilt:*

(1) K enthält die Null– *und die* Nachfolgerfunktion 0 *und S, wobei $0 : \mathbb{N} \to \mathbb{N}, 0(n) = 0$ für alle n, und $S : \mathbb{N} \to \mathbb{N}, S(n) = n + 1$ für alle n.*

(2) Für $1 \leq i \leq n$ enthält K die Projektionsfunktion *$U_i^n : \mathbb{N}^n \to \mathbb{N}$, wobei $U_i^n(m_1, \ldots, m_n) = m_i$ für allem$_1, \ldots, m_n \in \mathbb{N}$.*

(3) Wenn $f : A \to \mathbb{N}$ in K ist, wobei $A \subset \mathbb{N}^k$, und wenn $g_1, \ldots, g_k$ in K sind, dann ist auch die Komposition von f und $g_1, \ldots, g_k$, *nämlich die Funktion h mit $h(\vec{m}) = f(g_1(\vec{m}), \ldots, g_k(\vec{m}))$ für alle $\vec{m}$,[4] in K enthalten. Hierbei ist h genau für diejenigen $\vec{m}$ definiert, so dass $g_i(\vec{m})$ für alle i und sodann $f(g_1(\vec{m}), \ldots, g_k(\vec{m}))$ definiert ist.*

(4) K ist abgeschlossen bzgl. primitiver Rekursion, *d.h. wenn f und g in K sind, dann ist auch h in K, wobei für alle $\vec{m}$ gilt:*

$$h(\vec{m}, 0) = f(\vec{m})$$
$$h(\vec{m}, n+1) = g(\vec{m}, n, h(\vec{m}, n)).$$

Hierbei ist $h(\vec{m}, 0)$ definiert gdw. $f(\vec{m})$ definiert ist, und $h(\vec{m}, n+1)$ ist definiert gdw. $h(\vec{m}, n)$ und sodann auch $g(\vec{m}, n, h(\vec{m}, n))$ definiert ist.

Lemma 3.6 *Die folgenden Funktionen sind p.r.*

[4] Wir schreiben hier und im Folgenden $\vec{m}$ für $m_1, \ldots, m_n$ (für ein n) und setzen hier voraus, dass der Urbildbereich von allen g_i Teilmenge von $\mathbb{N}^n$ ist.

(1) $n + m$ [5]

(2) $n \cdot m$

(3) $n \dot{-} 1$, *wobei* $n \dot{-} 1 = n - 1$ *für* $n > 0$ *und* $n \dot{-} 1 = 0$ *für* $n = 0$

(4) $n \dot{-} m$, *wobei* $n \dot{-} m = n - m$ *für* $n \geq m$ *und* $n \dot{-} m = 0$ *für* $n < m$

(5) $\max(n, m)$

(6) $\min(n, m)$

(7) $eq(n, m)$, *wobei* $eq(n, m) = 1$ *für* $n = m$ *und* $eq(n, m) = 0$ *für* $n \neq m$

(8) $lt(n, m)$, *wobei* $lt(n, m) = 1$ *für* $n < m$ *und* $lt(n, m) = 0$ *für* $n \geq m$.

Definition 3.7 *Die Klasse der* partiell rekursiven Funktionen *ist die kleinste Klasse* K *von Funktionen* $f : A \to \mathbb{N}$, *wobei* $A \subset \mathbb{N}^k$ *für ein* $k > 0$, *so dass gilt:*

(1) – (4) wie in Definition 3.5, sowie

(5) K *ist abgeschlossen unter* Minimierung, *d.h. wenn* f *in* K *ist, dann ist auch* g *in* K, *wobei* $g(\vec{m}) =$ *das kleinste* n, *so dass* $f(\vec{m}, 0), \ldots, f(\vec{m}, n)$ *alle definiert sind und* $f(\vec{m}, n) = 0$.

$g(\vec{m})$ *ist nicht definiert, falls es kein solches* n *gibt.*

Lemma 3.8 *Sei* K *entweder die Klasse der p.r. Funktionen oder die Klasse der partiell rekursiven Funktionen.* K *besitzt die folgenden Abschlusseigenschaften:*

(1) Wenn $g, f_0, \ldots, f_k$ *alle in* K *sind, dann ist auch* h *in* K, *wobei*

$$h(\vec{m}) = \begin{cases} f_0(\vec{m}), \text{ falls } g(\vec{m}) = 0 \\ \vdots \\ f_{k-1}(\vec{m}), \text{ falls } g(\vec{m}) = k - 1 \\ f_k(\vec{m}), \text{ falls } g(\vec{m}) \geq K \end{cases}$$

$h(\vec{m})$ *ist genau dann definiert, wenn sowohl* $g(\vec{m})$ *als auch das entsprechende* $f_i(\vec{m})$ *definiert ist.*

[5] Darunter verstehen wir die Funktion $n, m \mapsto n + m$. Analoges gilt für das Folgende.

(2) Wenn f in K ist, dann ist auch g in K, wobei

$$g(\vec{m},n) = \begin{cases} \text{das kleinste } q \leq n, \text{ so dass} \\ f(\vec{m},0),\ldots,f(\vec{m},q) \text{ alle definiert} \\ \text{sind und } f(\vec{m},q) = 0, \text{ falls} \\ \text{ein solches } q \text{ existiert} \\ \\ n, \text{ falls alle } f(\vec{m},0),\ldots,f(\vec{m},n) \\ \text{definiert aber ungleich } 0 \text{ sind.} \end{cases}$$

$g(\vec{m},n)$ ist nicht definiert, falls wir nicht in einer dieser beiden Fälle sind.

Wenn f und g wie in (5) von Definition 3.7 sind, dann schreibt man oft $\mu n.\ f(\vec{m},n) = 0$ für $g(\vec{m})$. Wenn f und g wie in (2) von Lemma 3.8 sind, dann schreibt man oft $\mu q \leq n.\ g(\vec{m},q) = 0$ für $g(\vec{m},n)$.

Jede partiell rekursive Funktion ist Turing–berechenbar. Es gilt aber auch die Umkehrung:

Satz 3.9 *Sei $f : A \to \mathbb{N}$, wobei $A \subset \mathbb{N}^k$ für ein $k > 0$. Dann ist f partiell rekursiv gdw. f Turing–berechenbar ist.*

Definition 3.10 *Sei f partiell rekursiv, wobei der Urbildbereich von f gleich $\mathbb{N}^k$ für ein $k > 0$ ist. Dann heißt f* rekursiv.

Der Begriff der rekursiven Menge (Sprache) wurde in Kapitel 2 definiert. Es gilt nun:

Lemma 3.11 *Eine Menge $A \subset \mathbb{N}^k$ ist rekursiv gdw. die charakteristische Funktion χ_A von A rekursiv ist, wobei*

$$\chi_A(\vec{m}) = \begin{cases} 1 \text{ gdw. } \vec{m} \in A \\ 0 \text{ gdw. } \vec{m} \notin A. \end{cases}$$

Definition 3.12 *Eine Menge $A \subset \mathbb{N}^k$ heißt* primitiv rekursiv (p.r.) *gdw. χ_A p.r. ist.*

Kapitel 4

Was ist ein Beweis?

Mathematiker beweisen Sätze. Was aber ist ein Beweis? Seit EUKLID gibt es darauf im Wesentlichen *eine* Antwort: ein Beweis ist die logische Ableitung einer Aussage aus einer Menge von vorausgesetzten Aussagen, den "Axiomen". Dies wollen wir nun präzisieren. Wir werden sehen, dass Turing–Maschinen Beweise führen können.

Logische Ableitungen werden mit Hilfe von logischen Regeln vollzogen. Es stellt sich heraus, dass die einzige derartige Regel, die wir wirklich benötigen, der *modus ponens* ist. Der modus ponens ist ein logischer Schluss der folgenden Form:

$$\frac{\varphi \to \psi \qquad \varphi}{\psi}$$

Dieser Schluss ist korrekt in dem Sinne, dass für beliebige Modelle $\mathcal{M}$ und $\mathcal{M}$–Belegungen $\overline{\beta}$ gilt: aus $\mathcal{M} \models \varphi \to \psi[\overline{\beta}]$ und $\mathcal{M} \models \varphi[\overline{\beta}]$ folgt $\mathcal{M} \models \psi[\overline{\beta}]$.

Zusätzlich zu den Axiomen einer bestimmten Theorie benötigen wir immer auch "logische Axiome". Beispielsweise ist die Aussage $\forall v_0\ v_0 = v_0$ ein logisches Axiom: wir wollen sie nicht im Rahmen der Gruppentheorie (oder einer anderen Theorie) jedes Mal eigens fordern wollen.

Bis auf Weiteres fixieren wir eine Sprache $\mathcal{L}$ der Logik erster Stufe. Wir wollen definieren, was es heißt, dass eine $\mathcal{L}$–Formel φ aus einer Menge Γ von $\mathcal{L}$–Formeln beweisbar ist.

Wir definieren zunächst (unabhängig von Γ!) eine feste Menge Λ, die Menge der logischen Axiome.

Eine Formel ψ heißt *Verallgemeinerung* von φ gdw. ψ von der Gestalt

$$\forall v_{i_0} \dots \forall v_{i_{n-1}} \varphi$$

ist. (Dabei erlauben wir $n = 0$; φ ist auch Verallgemeinerung von sich selbst.) Eine $\mathcal{L}$–Formel φ ist ein *logisches Axiom* gdw. φ Verallgemeinerung einer Formel ist, die zu einer der folgenden Klassen gehört:

(1) Tautologien,

(2) Ersetzungsaxiome: $\forall v_i \varphi \to \varphi^{v_i}_\tau$, falls τ für v_i in φ einsetzbar ist,

(3) $\forall v_i(\varphi \to \psi) \to (\forall v_i \varphi \to \forall v_i \psi)$,

(4) $\varphi \to \forall v_i \varphi$, falls v_i in φ nicht frei vorkommt,

(5) $v_i = v_i$, und

(6) $v_i = v_j \to (\varphi \to \varphi')$, wobei φ atomar ist und φ' aus φ dadurch entsteht, dass an null oder mehr Stellen v_i durch v_j ersetzt wird.

Wir werden nun diese Klassen erklären. Wir werden sodann sehen, dass jedes logische Axiom logisch gültig ist. (1) Eine Formel φ ist eine *Tautologie* der Logik erster Stufe gdw. es möglich ist, φ auf folgende Art und Weise zu generieren. Es existiert eine aussagenlogische Tautologie ψ, in der als Junktoren lediglich $\neg$ und $\to$ vorkommen und es gibt ein Verfahren der Ersetzung von Aussagenvariablen durch Formeln der Logik erster Stufe, wobei gleiche Aussagenvariablen immer durch gleiche derartige Formeln ersetzt werden, so dass aus ψ die Formel φ hervorgeht.

Beispielsweise ist $A_0 \to (\neg A_1 \to A_0)$ eine aussagenlogische Tautologie. Angenommen, $\mathcal{L}$ enthält das einstellige Prädikatssymbol P. Dann ist

$$\forall v_2 P v_2 \to (\neg P v_0 \to \forall v_2 P v_2)$$

eine Tautologie der Logik erster Stufe. Damit ist übrigens auch

$$\forall v_0(\forall v_2 P v_2 \to (\neg P v_0 \to \forall v_2 P v_2))$$

ein logisches Axiom. (2) Wir denfinieren zunächst $\varphi^{v_i}_\tau$; dies ist das Resultat der Ersetzung von v_i durch τ an allen Stellen, an denen v_i frei vorkommt. Sei φ eine Formel und τ ein Term. Für atomares φ ist $\varphi^{v_i}_\tau$ das Resultat der Ersetzung aller Vorkommnisse von v_i in φ durch τ. Sodann gelte[1]

$$(\neg\varphi)^{v_i}_\tau \equiv \neg\varphi^{v_i}_\tau,\ (\varphi \to \psi)^{v_i}_\tau \equiv \varphi^{v_i}_\tau \to \psi^{v_i}_\tau$$

[1]Wir benutzen $\equiv$ zur Mitteilung der Gleichheit zweier Ausdrücke.

und

$$(\forall v_j \varphi)^{v_i}_{\tau} \equiv \begin{cases} \forall v_j \varphi^{v_i}_{\tau}, \text{ falls } j \neq i \\ \forall v_j \varphi, \text{ falls } j = i \end{cases}$$

Ist $\forall v_i \varphi \to \varphi^{v_i}_{\tau}$ immer ein vernünftiges Axiom? Betrachten wir $i = 0$ und $\varphi \equiv \exists v_1\ v_1 \neq v_0$. (D.h. $\varphi \equiv \neg \forall v_1 \neg \neg v_1 = v_0$!) Dann wird aus $\forall v_i \varphi \to \varphi^{v_i}_{\tau}$ die Formel

$$\forall v_0 \exists v_1\ v_1 \neq v_0 \to \exists v_1\ v_1 \neq v_1,$$

die nicht logisch gültig ist: dies sieht man schnell, indem man ein Modell $\mathcal{M}$ betrachtet, dessen Trägermenge M mindestens zwei Elemente enthält.

Wir lösen dieses Problem, indem wir dieses Axiom nur verlangen, falls τ für v_i in φ eingesetzt werden kann. Wir definieren diese Phrase wie folgt: Für atomares φ kann jedes τ für jedes v_i in φ eingesetzt werden. τ kann für v_i in $\neg\varphi$ eingesetzt werden gdw. τ für v_i in φ eingesetzt werden kann. τ kann für v_i in $\varphi \to \psi$ eingesetzt werden gdw. τ für v_i sowohl in φ als auch in ψ eingesetzt werden kann. τ kann für v_i in $\forall v_j \varphi$ eingesetzt werden gdw. v_i in $\forall v_j \varphi$ nicht frei vorkommt oder v_j in τ nicht vorkommt und τ für v_i in φ eingesetzt werden kann.

Damit kann beispielsweise v_1 für v_0 in $\exists v_1\ v_1 \neq v_0$ nicht eingesetzt werden.

Die Klassen (3), (4), (5) und (6) bedürfen keiner weiteren Erklärung.

Definition 4.1 *Sei $\Gamma \cup \{\varphi\}$ eine Menge von Formeln. Ein* Beweis von φ aus Γ *ist eine (endliche!) Folge $(\varphi_0, \varphi_1, \ldots, \varphi_{N-1})$ von Formeln, so dass Folgendes gilt:*

(a) $\varphi_{N-1} \equiv \varphi$,

(b) für jedes $i < N$ ist $\varphi_i \in \Gamma \cup \Lambda$ oder es existieren $j, k < i$, so dass φ_i mit Hilfe des modus ponens aus φ_j und φ_k hervorgeht, d.h. $\varphi_j \equiv \varphi_k \to \varphi_i$.

Die Formel φ ist *aus* Γ *beweisbar*, in Zeichen $\Gamma \vdash \varphi$, gdw. es einen Beweis von φ aus Γ gibt. Anstelle von $\emptyset \vdash \varphi$ schreiben wir auch $\vdash \varphi$.

Wir wollen nun den folgenden Korrektheitssatz zeigen: Sei φ aus Γ beweisbar, und sei $\mathcal{M} \models \Gamma[\overline{\beta}]$; dann gilt auch $\mathcal{M} \models \varphi[\overline{\beta}]$.

Wir beschäftigen uns zunächst mit den logischen Axiomen. Die meisten Umstände bereitet die Klasse (2). Für Terme ρ, τ ist $\rho^{v_i}_{\tau}$ das Resultat der Ersetzung aller Vorkommnisse von v_i in ρ durch τ.

Lemma 4.2 *Sei $\mathcal{M}$ ein Modell, und sei $\overline{\beta}$ eine $\mathcal{M}$–Belegung. Seien τ, ρ Terme. Sei β die durch $\overline{\beta}$ induzierte Terminterpretation, und sei $\widetilde{\beta}$ die durch $\overline{\beta}(v_i|\beta(\tau))$ induzierte Terminterpretation. Dann gilt*

$$\beta(\rho_\tau^{v_i}) = \widetilde{\beta}(\rho).$$

Beweis durch Induktion nach der "Komplexität" von ρ: Wenn ρ eine Konstante oder eine Variable v_j mit $j \neq i$ ist, dann ist $\rho_\tau^{v_i} \equiv \rho$ und $\widetilde{\beta}(\rho) = \beta(\rho)$. Wenn $\rho \equiv v_i$, dann ist $\rho_\tau^{v_i} \equiv \tau$, also $\widetilde{\beta}(\rho) = \widetilde{\beta}(v_i) = \beta(\tau) = \beta(\rho_\tau^{v_i})$.

Sei schließlich $\rho \equiv f\rho_0 \dots \rho_{n-1}$ für einen n–stelligen Funktor f. Dann gilt

$\widetilde{\beta}(\rho) = \widetilde{\beta}(f\rho_0 \dots \rho_{n-1}) =$
$f^{\mathcal{M}}(\widetilde{\beta}(\rho_0), \dots, \widetilde{\beta}(\rho_{n-1})) =$
$f^{\mathcal{M}}(\beta((\rho_0)_\tau^{v_i}, \dots, \beta(\rho_{n-1})_\tau^{v_i}))$ nach Induktionsvoraussetzung, $=$
$\beta(f(\rho_0)_\tau^{v_i} \dots (\rho_{n-1})_\tau^{v_i}) =$
$\beta((f\rho_0 \dots \rho_{n-1})_\tau^{v_i}) = \beta(\rho_\tau^{v_i})$.

□

Lemma 4.3 *Sei $\mathcal{M}$ ein Modell, und sei $\overline{\beta}$ eine $\mathcal{M}$–Belegung. Sei φ eine Formel und τ ein Term, der für v_i in φ eingesetzt werden kann. Sei β die durch $\overline{\beta}$ induzierte Terminterpretation. Dann gilt*

$$\mathcal{M} \models \varphi_\tau^{v_i}[\overline{\beta}] \text{ gdw. } \mathcal{M} \models \varphi[\overline{\beta}(v_i|\beta(\tau))].$$

Beweis durch Induktion nach der "Komplexität" von φ: Sei zunächst φ atomar, etwa $\varphi \equiv P\rho$, wobei P einstelliges Prädikatssymbol und ρ ein Term ist (alle übrigen Fälle sind analog). Dann gilt $\mathcal{M} \models \varphi_\tau^{v_i}[\overline{\beta}]$ gdw. $\mathcal{M} \models P\rho_\tau^{v_i}[\overline{\beta}]$ gdw. $\beta(\rho_\tau^{v_i}) \in P^{\mathcal{M}}$ gdw. $\widetilde{\beta}(\rho) \in P^{\mathcal{M}}$, nach obigem Lemma, wobei $\widetilde{\beta}$ die durch $\overline{\beta}(v_i|\beta(\tau))$ induzierte Terminterpretation ist, gdw. $\mathcal{M} \models P\rho[\overline{\beta}(v_i|\beta(\tau))]$ gdw. $\mathcal{M} \models \varphi[\overline{\beta}(v_i|\beta(\tau))]$. Für $\varphi \equiv \neg\psi$ oder $\varphi \equiv \psi \to \psi'$ ergibt sich die Aussage leicht aus der Induktionsvoraussetzung.

Sei nun $\varphi \equiv \forall v_j \psi$. Falls v_i nicht frei in φ vorkommt, dann stimmen $\overline{\beta}$ und $\overline{\beta}(v_i|\beta(\tau))$ auf den freien Variablen von φ überein und es gilt $\varphi_\tau^{v_i} \equiv \varphi$, so dass die Aussage sehr einfach folgt.

Angenommen nun, v_i kommt frei in φ vor. τ kann für v_i in φ eingesetzt werden, so dass also jetzt v_j in τ nicht vorkommt und τ für v_i in ψ eingesetzt werden kann. Es gilt somit $\mathcal{M} \models \varphi_\tau^{v_i}[\overline{\beta}]$ gdw. $\mathcal{M} \models \forall v_j \psi_\tau^{v_i}[\overline{\beta}]$ gdw. für alle

$a \in M$, $\mathcal{M} \models \psi^{v_i}_{\tau}[\overline{\beta}(v_j|a)]$ gdw. für alle $a \in M$, $\mathcal{M} \models \psi[\overline{\beta}(v_j|a)(v_i|\beta(\tau))]$ wegen Induktionsvoraussetzung, gdw. $\mathcal{M} \models \forall v_j \psi[\overline{\beta}(v_i|\beta(\tau))]$ gdw. $\mathcal{M} \models \varphi[\overline{\beta}(v_i|\beta(\tau))]$. □

Satz 4.4 (Gültigkeitssatz). *Jedes $\varphi \in \Lambda$, d.h. jedes logische Axiom, ist logisch gültig.*

Beweis: Es ist leicht zu sehen, dass mit φ auch jede Verallgemeinerung von φ logisch gültig ist. Es genügt daher zu zeigen, dass jedes Axiom aus einer der Klassen (1) bis (6) logisch gültig ist.

Sei φ zunächst eine Tautologie. Dann existiert eine aussagenlogische Tautologie ψ, in der lediglich die Junktoren $\neg$ und $\rightarrow$ vorkommen, und ein Ersetzungsverfahren $A_0 \mapsto \varphi_0$, $A_1 \mapsto \varphi_1, \dots$, das φ produziert. Sei $\mathcal{M}$ ein Modell und $\overline{\beta}$ eine $\mathcal{M}$–Belegung. Wir definieren ein $\overline{\alpha} : \{A_0, A_1, \dots\} \rightarrow \{0, 1\}$ durch $\overline{\alpha}(A_n) = 1$ gdw. $\mathcal{M} \models \varphi_n[\overline{\beta}]$. Wenn α die zu $\overline{\alpha}$ gehörige Belegung (im Sinne der Aussagenlogik) ist, dann gilt $\alpha(\psi) = 1$, da ψ Tautologie ist. Daraus folgt aber offensichtlich $\mathcal{M} \models \varphi[\overline{\beta}]$. Da $\mathcal{M}$ und $\overline{\beta}$ beliebig waren, ist φ logisch gültig.

Sei nun eines der Ersetzungsaxiome gegeben. Sei $\mathcal{M}$ ein Modell und $\overline{\beta}$ eine $\mathcal{M}$–Belegung. Sei $\mathcal{M} \models \forall v_i \varphi[\overline{\beta}]$, und sei τ für v_i in φ einsetzbar. Dann gilt, wenn β die durch $\overline{\beta}$ induzierte Terminterpretation ist, $\mathcal{M} \models \varphi[\overline{\beta}(v_i|\beta(\tau))]$, also auch $\mathcal{M} \models \varphi^{v_i}_{\tau}[\overline{\beta}]$ nach Lemma 4.3. Dies zeigt $\mathcal{M} \models \forall v_i \varphi \rightarrow \varphi^{v_i}_{\tau}[\overline{\beta}]$, falls τ für v_i in φ einsetzbar ist.

Wir zeigen jetzt $\mathcal{M} \models \forall v_i(\varphi \rightarrow \psi) \rightarrow (\forall v_i \varphi \rightarrow \forall v_i \psi)[\overline{\beta}]$ für alle $\mathcal{M}$ und alle $\mathcal{M}$–Belegungen $\overline{\beta}$. Es genügt zu zeigen, dass aus $\mathcal{M} \models \forall v_i(\varphi \rightarrow \psi)[\overline{\beta}]$ und $\mathcal{M} \models \forall v_i \varphi[\overline{\beta}]$ folgt, dass $\mathcal{M} \models \forall v_i \psi[\overline{\beta}]$. Sei aber $a \in M$ beliebig. Dann liefert $\mathcal{M} \models \forall v_i \varphi[\overline{\beta}]$, dass $\mathcal{M} \models \varphi[\overline{\beta}(v_i|a)]$, und $\mathcal{M} \models \varphi \rightarrow \psi[\overline{\beta}(v_i|a)]$. Damit gilt dann auch $\mathcal{M} \models \psi[\overline{\beta}(v_i|a)]$.

Die logische Gültigkeit von Formeln aus der Klasse (4) ist einfach zu sehen: Wegen Satz 3.2 gilt, wenn v_i in φ nicht frei vorkommt, für ein beliebiges Modell $\mathcal{M}$, für eine beliebige M–Belegung $\overline{\beta}$ und für ein beliebiges $a \in M$, $\mathcal{M} \models \varphi[\overline{\beta}]$ gdw. $\mathcal{M} \models \varphi[\overline{\beta}(v_i|a)]$. Also gilt dann auch $\mathcal{M} \models \varphi[\overline{\beta}]$ gdw. $\mathcal{M} \models \forall v_i \varphi[\overline{\beta}]$

Die logische Gültigkeit von Formeln aus der Klasse (5) ist trivial.

Sei nun ein Axiom aus Klasse (6) gegeben, d.h. etwa $v_i = v_j \rightarrow (\varphi \rightarrow \varphi')$, wobei φ atomar ist und φ' aus φ hervorgeht, indem an null oder mehr Stellen v_i durch v_j ersetzt wird. Sei etwa φ von der Gestalt $P\tau_0 \dots \tau_{n-1}$, wobei P n–stellig ist. Es genügt zu zeigen, dass aus $\mathcal{M} \models v_i = v_j[\overline{\beta}]$ und $\mathcal{M} \models \varphi[\overline{\beta}]$ folgt, dass $\mathcal{M} \models \varphi'[\overline{\beta}]$. Sei also β die durch $\overline{\beta}$ induzierte Ter-

minterpretation, und sei φ' gleich $P\tau'_0 \dots \tau'_{n-1}$ (so dass τ'_i aus τ_i hervorgeht, indem an null oder mehr Stellen v_i durch v_j ersetzt wird). Dann gilt offenbar $\mathcal{M} \models \varphi[\overline{\beta}]$ gdw. $(\beta(\tau_0), \dots, \beta(\tau_{n-1})) \in P^{\mathcal{M}}$ gdw. $(\beta(\tau'_0), \dots, \beta(\tau'_{n-1})) \in P^{\mathcal{M}}$ gdw. $\mathcal{M} \models \varphi'[\overline{\beta}]$. □

Wir zeigen nun, dass aus $\Gamma \vdash \varphi$ folgt, dass $\Gamma \models \varphi$. Der Gödelsche Vollständigkeitssatz wird sagen, dass sogar die Umkehrung gilt.

Satz 4.5 (Korrektheitssatz). *Sei $\Gamma \cup \{\varphi\}$ eine Menge von Formeln. Wenn $\Gamma \vdash \varphi$, dann gilt auch $\Gamma \models \varphi$.*

Beweis: Sei $\Gamma \vdash \varphi$ vorausgesetzt. Sei $(\varphi_0, \varphi_1, \dots, \varphi_{N-1})$ ein Beweis von φ aus Γ. Es ist leicht zu sehen, dass dann auch für jedes $n \leq N(\varphi_0, \dots, \varphi_{n-1})$ ein Beweis von φ aus Γ ist.

Sei $\mathcal{M}$ ein Modell, und sei $\overline{\beta}$ eine $\mathcal{M}$–Belegung. Wir müssen zeigen, dass aus $\mathcal{M} \models \Gamma[\overline{\beta}]$ folgt, dass $\mathcal{M} \models \varphi[\overline{\beta}]$. Wir zeigen durch Induktion nach $n \leq N$, dass $\mathcal{M} \models \varphi_{n-1}[\overline{\beta}]$.

Sei also $n > 0$ und gelte $\mathcal{M} \models \Gamma[\overline{\beta}]$, sowie $\mathcal{M} \models \varphi_i[\overline{\beta}]$ für alle $i < n-1$. Falls dann $\varphi_{n-1} \in \Gamma \cup \Lambda$, haben wir $\mathcal{M} \models \varphi_{n-1}[\overline{\beta}]$ (für $\varphi_{n-1} \in \Lambda$ benutzt dies den Gültigkeitssatz).

Andernfalls existieren $j, k < n-1$ mit $\varphi_j \equiv \varphi_k \to \varphi_{n-1}$. Dann folgt aber aus $\mathcal{M} \models \varphi_k \to \varphi_{n-1}[\overline{\beta}]$ und $\mathcal{M} \models \varphi_k[\overline{\beta}]$ sofort $\mathcal{M} \models \varphi_{n-1}[\overline{\beta}]$. □

Eine Menge Γ von Formeln heißt *(syntaktisch) konsistent* gdw. es keine Formel φ gibt, so dass Γ sowohl φ als auch $\neg\varphi$ beweist. Γ heißt *erfüllbar* (oder *semantisch konsistent*) gdw. ein Modell $\mathcal{M}$ und eine $\mathcal{M}$–Belegung $\overline{\beta}$ mit $\mathcal{M} \models \Gamma[\overline{\beta}]$ existieren. Der Korrektheitssatz besitzt folgendes

Korollar 4.6 *Sei Γ eine Menge von Formeln. Wenn Γ erfüllbar ist, dann ist Γ konsistent.*

Während, wie wir sehen werden, $\{\varphi : \emptyset \vdash \varphi\}$ nicht entscheidbar ist, gilt folgender

Satz 4.7 *Sei die Sprache $\mathcal{L}$ gegeben durch I, J, K, wobei $I \cup J \cup K$ höchstens abzählbar ist. Sei Γ eine rekursiv aufzählbare Menge von $\mathcal{L}$-Formeln. Dann ist auch $\{\varphi : \Gamma \vdash \varphi\}$ rekursiv aufzählbar.*

Beweis: Zunächst ist $\{\varphi : \varphi$ ist ein logisches Axiom$\}$ entscheidbar. Jedes logische Axiom ist nämlich Verallgemeinerung eines Axioms aus einer der Klassen (1) bis (6). Dass die Menge der Tautologien aus (1) entscheidbar ist,

kann mit den Methoden des 2. Kapitels gezeigt werden. Dass die Mengen von Axiomen, die zu den Klassen (2) bis (6) gehören, entscheidbar ist, zeigt man mit ähnlichen Mitteln.

Sei nun Γ rekursiv aufzählbar. Sei $\top_\Gamma$ eine Turing–Maschine, die dies bezeugt (d.h. $\top_\Gamma(\psi)\downarrow +$ gdw. $\psi \in \Gamma$). Wir bauen nun eine Turing–Maschine $\top$, die Folgendes leistet.

Es werde φ eingegeben. Hinter φ schreibt $\top$ in der n^{ten} Runde jeweils voneinander abgetrennt durch ein Trennungszeichen, etwa #, alle Ausdrücke der Sprache $\mathcal{L}$ der Länge $\leq n$ nebeneinander auf das Rechenband.[2] Diejenigen von ihnen werden markiert, die entweder ein logisches Axiom sind, oder die von $\top_\Gamma$ in $\leq n$ Rechenschritten als Element von Γ erkannt werden. Sodann schreibt $\top$ (immer noch in der n^{ten} Runde) hinter die entstandene Liste wieder jeweils abgetrennt voneinander alle Beweise

$$(\varphi_0, \varphi_1, \ldots, \varphi_{N-1})$$

mit $N \leq n$, wobei aber jedes φ_i entweder *eines der nun markierten* Axiome aus $\Lambda \cup \Gamma$ ist oder durch modus ponens aus früheren φ_j und φ_k hervorgeht. Falls dabei φ als eine Formel φ_{N-1} in einem solchen Beweis gesichtet wird, akzeptiert $\top$ die Eingabe φ. Andernfalls begibt sich $\top$ in die $(n+1)^{\text{te}}$ Runde.

Offensichtlich folgt aus $\top(\varphi)\downarrow +$, dass $\Gamma \vdash \varphi$. Wenn aber Γ die Formel φ beweist, dann existiert ein $n \in \mathbb{N}$, so dass $(\varphi_0, \varphi_1, \ldots, \varphi_{N-1})$ ein Beweis von φ aus Γ ist, für den gilt: $N \leq n$, jedes $\varphi_i \in \Lambda \cup \Gamma$ hat Länge $\leq n$, und für $\varphi_i \in \Gamma$ erkennt $\top_\Gamma$ in $\leq n$ Rechenschritten, dass φ_i in Γ ist; also gilt dann auch $\top(\varphi)\downarrow +$. $\square$

[2] Wir fassen dabei, analog zur Verfahrensweise hinsichtlich der Aussagenvariablen im 2. Kapitel, die Variable v_n etwa als eine Folge von $n+1$ Symbolen $*$ auf.

Kapitel 5

Der Gödelsche Vollständigkeitssatz

Wir beweisen zunächst einige Lemmata zur Beweisbarkeit.

Die Formeln $\varphi_0, \dots, \varphi_{n-1}$ *implizieren tautologisch* die Formel ψ gdw. die Formel

$$\varphi_0 \to (\varphi_1 \to (\dots(\varphi_{n-1} \to \psi)\dots))$$

eine Tautologie ist.

Lemma 5.1 (Tautologische Implikation) *Wenn die Formelmenge Γ die Formeln $\varphi_0, \dots, \varphi_{n-1}$ beweist und wenn $\varphi_0, \dots, \varphi_{n-1}$ tautologisch ψ implizieren, dann beweist Γ die Formel ψ.*

Beweis: $\varphi_0 \to (\varphi_1 \to (\dots(\varphi_{n-1} \to \psi)\dots))$ ist Tautologie, also aus Γ beweisbar. Da $\varphi_0, \dots, \varphi_{n-1}$ aus Γ beweisbar sind, liefert eine n–fache Anwendung des modus ponens, dass auch ψ beweisbar ist. □

Lemma 5.2 (Deduktion) *Wenn die Formelmenge $\Gamma \cup \{\varphi\}$ die Formel ψ beweist, dann beweist Γ die Formel $\varphi \to \psi$.*

Beweis: Sei $(\varphi_0, \dots, \varphi_{N-1})$ ein Beweis von ψ aus $\Gamma \cup \{\varphi\}$. Wir zeigen durch Induktion nach $i < N$, dass $\Gamma \vdash \varphi \to \varphi_i$.

1. Fall: $\varphi_i \equiv \varphi$. Dann ist $\varphi \to \varphi_i$ Tautologie und wird von Γ bewiesen.

2. Fall: $\varphi_i \in \Gamma \cup \Lambda$. Da φ_i tautologisch $\varphi \to \varphi_i$ impliziert, folgt mit dem Lemma zur tautologischen Implikation aus $\Gamma \vdash \varphi_i$, dass $\Gamma \vdash \varphi \to \varphi_i$.

3. Fall: φ_i geht durch modus ponens aus φ_j, φ_k hervor, d.h. $\varphi_k \equiv \varphi_j \to \varphi_i$, wobei $j, k < i$. Nach Induktionsannahme sind $\varphi \to \varphi_j$ und $\varphi \to \varphi_k$ (d.h. $\varphi \to (\varphi_j \to \varphi_i)$) aus Γ beweisbar. Da $\varphi \to \varphi_j$, $\varphi \to (\varphi_j \to \varphi_i)$ tautologisch $\varphi \to \varphi_i$ implizieren, folgt mit dem Lemma zur tautologischen Implikation aus $\Gamma \vdash \varphi \to \varphi_j$ und $\Gamma \vdash \varphi \to (\varphi_j \to \varphi_i)$, dass $\Gamma \vdash \varphi \to \varphi_i$. □

Eine Formelmenge Γ heit *inkonsistent* gdw. Γ nicht konsistent ist.

Lemma 5.3 (Widerspruchsbeweis) *Sei die Formelmenge $\Gamma \cup \{\varphi\}$ inkonsistent. Dann gilt $\Gamma \vdash \neg\varphi$.*

Beweis: Sei ψ so, dass $\Gamma \cup \{\varphi\} \vdash \psi$ und $\Gamma \cup \{\varphi\} \vdash \neg\psi$. Nach obigem Lemma zur Deduktion gilt dann $\Gamma \vdash \varphi \to \psi$ und auch $\Gamma \vdash \varphi \to \neg\psi$. Da aber $\varphi \to \psi$, $\varphi \to \neg\psi$ tautologisch $\neg\varphi$ implizieren, gilt dann auch $\Gamma \vdash \neg\varphi$ nach dem Lemma zur tautologischen Implikation. □

Lemma 5.4 (Verallgemeinerung 1) *Sei Γ eine Formelmenge, und sei $i \in \mathbb{N}$ so, dass v_i in keiner Formel aus Γ frei vorkommt. Sei φ eine Formel mit $\Gamma \vdash \varphi$. Dann gilt auch $\Gamma \vdash \forall v_i \varphi$.*

Beweis: Sei $(\varphi_0, \ldots, \varphi_{N-1})$ ein Beweis von φ aus Γ. Wir zeigen durch Induktion nach $j < N$, dass $\Gamma \vdash \forall v_i \varphi_j$.

1. Fall: $\varphi_j \in \Lambda$. Dann ist auch $\forall v_i \varphi_j \in \Lambda$, mithin $\Gamma \vdash \forall v_i \varphi_j$.

2. Fall: $\varphi_j \in \Gamma$. Da v_i nicht frei in φ_j vorkommt, ist $\varphi_j \to \forall v_i \varphi_j$ ein logisches Axiom der Klasse (4). Aus $\Gamma \vdash \varphi_j$ und $\Gamma \vdash \varphi_j \to \forall v_i \varphi_j$ ergibt sich dann $\Gamma \vdash \forall v_i \varphi_j$.

3. Fall: Es existieren $k, l < j$ mit $\varphi_l \equiv \varphi_k \to \varphi_j$. Nach Induktionsvoraussetzung gilt sowohl $\Gamma \vdash \forall v_i \varphi_k$ als auch $\Gamma \vdash \forall v_i \varphi_l$ (d.h. $\forall v_i(\varphi_k \to \varphi_j)$). Nun ist aber $\forall v_i(\varphi_k \to \varphi_j) \to (\forall v_i \varphi_k \to \forall v_i \varphi_j)$ ein logisches Axiom der Klasse (3). Damit ergibt sich nun leicht, dass $\Gamma \vdash \forall v_i \varphi_j$. □

Wir benötigen die folgenden Tatsachen zur Ersetzbarkeit. Hierbei ist φ eine Formel, τ ist ein Term, c ist eine Konstante, und $i, j, k \in \mathbb{N}$.

(a) v_i kann für v_i in φ eingesetzt werden.

(b) Wenn keine in φ vorkommende Variable in τ vorkommt, dann kann τ für v_i in φ eingesetzt werden.

(c) Wenn v_j nicht in φ vorkommt, dann kann v_i für v_j in $\varphi_{v_j}^{v_i}$ eingesetzt werden und es gilt $(\varphi_{v_j}^{v_i})_{v_i}^{v_j} \equiv \varphi$.

(d) Wenn $i \neq k$ und wenn τ für v_i in φ eingesetzt werden kann, dann kann τ für v_i in $\varphi_{v_k}^{v_i}$ eingesetzt werden.

(e) Wenn τ für v_i in φ eingesetzt werden kann und wenn v_j nicht in φ vorkommt, dann kann der Term $\tau_{v_j}^c$ für v_i in $\varphi_{v_j}^c$ eingesetzt werden. ($\tau_{v_j}^c$ bzw. $\varphi_{v_j}^c$ entsteht aus τ bzw. φ, indem überall c durch v_j ersetzt wird.)

Die folgenden beiden Lemmata lassen sich ähnlich zeigen wie obiges 1. Lemma zur Verallgemeinerung.

Lemma 5.5 (Verallgemeinerung 2) *Sei $\Gamma \cup \{\varphi\}$ eine Formelmenge, und sei c eine Konstante, die in keiner der Formeln aus Γ vorkommt. Sei $\Gamma \vdash \varphi$. Dann existiert ein $i \in \mathbb{N}$, so dass v_i nicht in φ vorkommt, $\Gamma \vdash \varphi_{v_i}^c$ und $\Gamma \vdash \forall v_i \varphi_{v_i}^c$, und es gibt Beweise sowohl von $\varphi_{v_i}^c$ als auch von $\forall v_i \varphi_{v_i}^c$ aus Γ, so dass c in keiner Formel dieser Beweise vorkommt.*

Lemma 5.6 (Verallgemeinerung 3) *Sei $\Gamma \cup \{\varphi\}$ eine Formelmenge, sei $i \in N$, und sei c eine Konstante, die in keiner Formel aus $\Gamma \cup \{\varphi\}$ vorkommt. Dann folgt aus $\Gamma \vdash \varphi_c^{v_i}$, dass $\Gamma \vdash \forall v_i \varphi$.*

Wir benötigen schlielich die folgenden weiteren Tatsachen, die leicht einzusehen sind.

Variablen–Umbenennung: Sei φ ein Formel, sei τ ein Term, und sei $i \in \mathbb{N}$. Dann existiert eine Formel φ', so dass τ für v_i in φ' eingesetzt werden kann, und es gilt sowohl $\vdash \varphi \to \varphi'$ als auch $\vdash \varphi' \to \varphi$. (Man erhält φ' durch Umbenennung der gebundenen, d.h. nicht freien Variablen.)

Tatsachen zur Gleichheit:

(a) $\vdash \forall v_i \; v_i = v_i$

(b) $\vdash \forall v_i \forall v_j (v_i = v_j \to v_j = v_i)$

(c) $\vdash \forall v_i \forall v_j \forall v_k (v_i = v_j \to (v_j = v_k \to v_i = v_k))$.
$(v_i = v_j \to (v_i = v_k \to v_j = v_k)$ ist ein logisches Axiom der Klasse (6).
$(v_i = v_j \to (v_i = v_k \to v_j = v_k)) \to (v_i = v_j \to (v_j = v_k \to v_i = v_k))$ ist ein logisches Axiom der Klasse (1). Dies liefert (c), mit Hilfe des modus ponens. Insbesondere ist aber auch $v_i = v_j \to (v_i = v_i \to v_j =$

v_i) logisches Axiom der Klasse (6). Da $v_i = v_i$ logisches Axiom der Klasse (5) ist und $v_i = v_i,\ v_i = v_j \to (v_i = v_i \to v_j = v_i)$ tautologisch $v_i = v_j \to v_j = v_i$ implizieren, sieht man dann (b).)

(d) Sei P ein n–stelliges Prädikatsymbol. Dann gilt

$$\vdash \forall v_{i_0} \forall v_{i'_0} \dots \forall v_{i_{n-1}} \forall v_{i'_{n-1}} (v_{i_0} = v_{i'_0} \to \\ (\dots (v_{i_{n-1}} = v_{i'_{n-1}} \to (P v_{i_0} \dots v_{i_{n-1}} \to P v_{i'_0} \dots v_{i'_{n-1}})) \dots)).$$

(e) Sei f ein n–stelliger Funktor. Dann gilt

$$\vdash \forall v_{i_0} \forall v_{i'_0} \dots \forall v_{i_{n-1}} \forall v_{i'_{n-1}} (v_{i_0} = v_{i'_0} \to \\ (\dots (v_{i_{n-1}} = v_{i'_{n-1}} \to f v_{i_0} \dots v_{i_{n-1}} = f v_{i'_0} \dots v_{i'_{n-1}}) \dots)).$$

Wir beweisen nun den GÖDELschen Vollständigkeitssatz. Dieser lautet wie folgt.

Satz 5.7 (Gödelscher Vollständigkeitssatz)

(*) Sei $\Gamma \cup \{\varphi\}$ ein Menge von Formeln. Wenn Γ (logisch) φ impliziert, dann ist φ aus Γ beweisbar. D.h. wenn $\Gamma \models \varphi$, dann gilt auch $\Gamma \vdash \varphi$.

(**) Jede (syntaktisch) konsistente Formelmenge ist erfüllbar (d.h. semantisch konsistent).

Zusammen mit dem Korrektheitssatz besagt dies also, dass $\Gamma \models \varphi$ gdw. $\Gamma \vdash \varphi$. Der Gödelsche Vollständigkeitssatz teilt mit, dass die Beweismethoden, die wir vorgestellt haben, vollständig sind.

Beweis des Vollständigkeitssatzes: Wir zeigen zunächst, dass (*) aus (**) folgt. Angenommen, $\Gamma \models \varphi$. Dann kann mit Hilfe von (**) $\Gamma \cup \{\neg\varphi\}$ nicht konsistent sein, da sonst $\Gamma \cup \{\neg\varphi\}$ erfüllbar wäre im Widerspruch zu $\Gamma \models \varphi$. Aus der Inkonsistenz von $\Gamma \cup \{\neg\varphi\}$ ergibt sich aber $\Gamma \vdash \neg\neg\varphi$ mit dem obigen Lemma zum Widerspruchsbeweis. Da aber $\neg\neg\varphi$ tautologisch φ impliziert, gilt dann auch $\Gamma \vdash \varphi$ wie gewünscht.

Es bleibt also, (*) zu beweisen. Sei Γ eine konsistente Menge von Formeln, wobei nun daran erinnert sei, dass diese Formeln $\mathcal{L}$–Formeln sind für eine Sprache $\mathcal{L}$ der Logik 1. Stufe. Der Beweis von (*) hat ähnlichkeiten mit dem Beweis des Kompaktheitssatzes der Aussagenlogik. Wir werden Γ zu einer Menge Δ von Formeln einer reicheren Sprache $\mathcal{L}^*$ vergröern, so dass Folgendes gilt:

(a) $\Gamma \subset \Delta$,

(b) Δ ist *maximal–konsistent*, d.h. Δ ist konsistent und für jede $\mathcal{L}^*$–Formel φ gilt $\varphi \in \Delta$ oder $\neg\varphi \in \Delta$, und

(c) Δ ist eine *Henkin–Menge*, d.h. für jede $\mathcal{L}^*$–Formel φ und für jedes $i \in \mathbb{N}$ existiert eine Konstante $c = c(\varphi, i)$, so dass $\neg\forall v_i\varphi \to \neg\varphi_c^{v_i}$ Element von Δ ist.[1]

Sodann verwenden wir Δ, um ein Modell $\mathfrak{M}$ und eine M–Belegung $\overline{\beta}$ mit $\mathfrak{M} \models \Delta[\overline{\beta}]$ zu konstruieren. Dies liefert dann auch ein Modell $\overline{\mathfrak{M}}$ (mit gleicher Trägermenge wie $\mathfrak{M}$), so dass $\overline{\mathfrak{M}} \models \Gamma[\overline{\beta}]$. $\overline{\mathfrak{M}}$ und $\overline{\beta}$ bezeugen also, dass Γ erfüllbar ist.

Für die Konstruktion von Δ benötigen wir die folgende Aussage.

Satz 5.8 (Hausdorffsches Maximalitätsprinzip) *Sei F eine beliebige Menge, und sei $\mathcal{F}$ eine Menge von Teilmengen F. Angenommen, es gilt Folgendes: für alle $\overline{\mathcal{F}} \subset \mathcal{F}$, so dass $X \subset Y$ oder $Y \subset X$ für beliebige $X, Y \in \overline{\mathcal{F}}$, ist auch $\bigcup\overline{\mathcal{F}} = \{a \in F : a \in X \text{ für ein } X \in \overline{\mathcal{F}}\}$ Element von $\mathcal{F}$. Dann existiert ein $X_{\max} \in \mathcal{F}$, so dass keine echte Obermenge von $X_{\max}$ ebenfalls ein Element von $\mathcal{F}$ ist.*

Das HAUSDORFFsche Maximalitätsprinzip ist äquivalent sowohl zum Zornschen Lemma als auch zum Auswahlaxiom.

Wir beginnen damit, $\mathcal{L}^*$ zu definieren. Wir setzen $\mathcal{L}_0 = \mathcal{L}$. Sei $\mathcal{L}_n, n \in \mathbb{N}$, gegeben. Für jedes Paar (i, φ), wobei $i \in \mathbb{N}$ und φ eine $\mathcal{L}_n$–Formel ist, wählen wir eine eigene neue Konstante $c = c(i, \varphi)$, die wir zur Sprache $\mathcal{L}_n$ hinzufügen; $\mathcal{L}_{n+1}$ ist die Sprache $\mathcal{L}_n$, erweitert genau um all diese Konstanten $c(i, \varphi)$. Schlielich sei $\mathcal{L}^*$ die "Vereinigung" aller Sprachen $\mathcal{L}_n, n \in \mathbb{N}$. Offensichtlich gilt für $\mathcal{L}^*$ Folgendes: für jedes Paar (i, φ), wobei $n \in \mathbb{N}$ und φ eine $\mathcal{L}^*$–Formel ist, existiert eine eigene Konstante $c(i, \varphi)$ der Sprache $\mathcal{L}^*$.

Wir fassen jetzt Γ als Menge von Formeln der erweiterten Sprache $\mathcal{L}^*$ auf. Als Menge von $\mathcal{L}$–Formeln war Γ als (syntaktisch) konsistent vorausgesetzt. Auch wenn es eine Subtilität darstellt, wir müssen uns nun überzeugen, dass Γ als Menge von $\mathcal{L}^*$–Formeln weiterhin (syntaktisch) konsistent ist.

Angenommen, es gibt eine $\mathcal{L}^*$–Formel φ, so dass sowohl $\Gamma \vdash \varphi$ als auch $\Gamma \vdash \neg\varphi$ gilt. In den in den Beweisen von φ und $\neg\varphi$ auftauchenden Formeln kommen nur endlich viele der neuen Konstanten der Form $c(i, \overline{\varphi})$ vor. Seien dies etwa k viele, $c_0, \ldots, c_{k-1}$. Auf Grund von k–facher

[1] Für $\varphi \equiv \neg\psi$ ist $\neg\forall v_i\varphi \to \neg\varphi_c^{v_i}$ im Wesentlichen die Formel $\exists v_i\psi \to \psi_c^{v_i}$.

Anwendung des 2. Lemmas zur Verallgemeinerung existieren dann Variablen $v_{i_0}, \ldots, v_{i_{k-1}}$, so dass Γ auch die Formeln $\varphi' \equiv (\ldots(\varphi^{c_0}_{v_{i_0}})\ldots)^{c_{k-1}}_{v_{i_{k-1}}}$ und $\neg\varphi' \equiv (\ldots(\neg\varphi^{c_0}_{v_{i_0}})\ldots)^{c_{k-1}}_{v_{i_{k-1}}}$ beweist. (Die Konstanten $c_0, \ldots, c_{k-1}$ kommen ja in den Formeln von Γ nicht vor.) Darüber hinaus gibt es Beweise dieser beiden Formeln, so dass $c_0, \ldots, c_{k-1}$ nicht mehr in den Formeln dieser Beweise vorkommen. Wir sehen, dass es Beweise dieser beiden Formeln φ' und $\neg\varphi'$ gibt, die nur aus $\mathcal{L}$–Formeln bestehen. Dies widerspricht der (syntaktischen) Konsistenz von Γ (als Menge von $\mathcal{L}$–Formeln).

Wir erweitern nun die Formelmenge Γ. Wir setzen $\Gamma_H = \Gamma \cup H$, wobei H alle Formeln der Gestalt $\neg\forall v_i\varphi \to \neg\varphi^{v_i}_{c(i,\varphi)}$ enthält, wobei $i \in \mathbb{N}$ und φ eine $\mathcal{L}^*$–Formel ist. Wir behaupten, dass Γ_H ebenfalls (syntaktisch) konsistent ist.

Wenn Γ_H nicht konsistent ist, dann existiert offensichtlich eine endliche Teilmenge $\overline{H}$ von H, so dass $\Gamma \cup \overline{H}$ inkonsistent ist. Sei $m_0 \in \mathbb{N}$ das kleinste m, so dass ein m–elementiges $\overline{H} \subset H$ existiert, so dass $\Gamma \cup \overline{H}$ inkonsistent ist. Da Γ konsistent ist, ist $m_0 > 0$. Sei $\neg\forall v_i\varphi \to \neg\varphi^{v_i}_{c(i,\varphi)}$ in $\overline{H}$, und sei $\overline{H}_0 = \overline{H} \setminus \{\neg\forall v_i\varphi \to \neg\varphi^{v_i}_{c(i,\varphi)}\}$. $\Gamma \cup \overline{H}_0$ ist also konsistent.

Schreibe $c = c(i,\varphi)$. Da $\Gamma \cup \overline{H} = \Gamma \cup \overline{H}_0 \cup \{\neg\forall v_i\varphi \to \neg\varphi^{v_i}_c\}$ inkonsistent ist, beweist nach dem Lemma zum Widerspruchsbeweis $\Gamma \cup \overline{H}_0$ die Formel $\neg(\neg\forall v_i\varphi \to \neg\varphi^{v_i}_c)$, d.h. die Formel $\neg\forall v_i\varphi \wedge \varphi^{v_i}_c$. Da diese Formel $\varphi^{v_i}_c$ tautologisch impliziert, beweist $\Gamma \cup \overline{H}_0$ also auch $\varphi^{v_i}_c$. Die Konstante c kommt aber in keiner Formel aus $\Gamma \cup \overline{H}_0$ vor, so dass nach dem 3. Lemma zur Verallgemeinerung $\Gamma \cup \overline{H}_0$ die Formel $\forall v_i\varphi$ beweist. Andererseits impliziert $\neg\forall v_i\varphi \wedge \varphi^{v_i}_c$ tautologisch $\neg\forall v_i\varphi$, so dass aus $\Gamma \cup \overline{H}_0 \vdash \neg\forall v_i\varphi \wedge \varphi^{v_i}_c$ auch $\Gamma \cup \overline{H}_0 \vdash \neg\forall v_i\varphi$ folgt. $\Gamma \cup \overline{H}_0$ beweist also sowohl $\neg\forall v_i\varphi$ als auch $\forall v_i\varphi$, ist also inkonsistent.

Dieser Widerspruch zeigt, dass Γ_H konsistent ist.

Wir wollen nun Γ_H zu einer maximal–konsistenten Menge Δ von $\mathcal{L}^*$–Formeln erweitern. Hierzu benutzen wir einfach das Hausdorffsche Maximalitätsprinzip. Sei $\mathcal{F}$ die Gesamtheit aller konsistenten Mengen X von $\mathcal{L}^*$–Formeln mit $X \supset \Gamma_H$. Da Beweise endlich sind, überzeugt man sich sehr leicht, dass $\mathcal{F}$ die Voraussetzung des Hausdorffschen Maximalitätsprinzips erfüllt.

Sei $X_{\max} \in \mathcal{F}$, so dass keine echte Obermenge von $X_{\max}$ auch in $\mathcal{F}$ ist. Schreibe $\Delta = X_{\max}$. Sei φ eine beliebige $\mathcal{L}^*$–Formel. Entweder $\Delta \cup \{\varphi\}$ oder $\Delta \cup \{\neg\varphi\}$ ist konsistent, da ansonsten Δ sowohl φ als auch $\neg\varphi$ bewiese, also inkonsistent wäre. Δ ist als Element von $\mathcal{F}$ aber konsistent. Da Δ keine konsistente echte Obermenge besitzt, ist also $\Delta \cup \{\varphi\} = \Delta$ (d.h. $\varphi \in \Delta$) oder $\Delta \cup \{\neg\varphi\} = \Delta$ (d.h. $\neg\varphi \in \Delta$). Δ ist also maximal–konsistent. Auerdem ist

$\Gamma \subset \Delta$, und Δ ist eine Henkin–Menge.

Wir konstruieren nun ein Modell von Δ.

Aufgrund der Maximalität von Δ sind für eine beliebige $\mathcal{L}^*$–Formel φ die folgenden Aussagen äquivalent: $\Delta \cup \{\varphi\}$ ist konsistent; $\varphi \in \Delta$; $\Delta \vdash \varphi$; $\Delta \cup \{\neg\varphi\}$ ist inkonsistent; $\neg\varphi \notin \Delta$; es gilt nicht $\Delta \vdash \neg\varphi$. Entsprechend sind für beliebige $\mathcal{L}^*$–Formeln die folgenden Aussagen äquivalent: wenn $\psi \in \Delta$, dann $\psi' \in \Delta$; $\psi \to \psi' \in \Delta$. Da Δ darüber hinaus eine Henkin–Menge ist, gilt für $i \in \mathbb{N}$ und eine beliebige $\mathcal{L}^*$-Formel: wenn $\neg\forall v_i\varphi \in \Delta$, dann existiert eine Konstante c (nämlich $c(i,\varphi)$), so dass $\neg\varphi_c^{v_i} \in \Delta$.

Wir konstruieren $\mathfrak{M}$ als "Termmodell".

Da Δ eine Henkin–Menge ist, haben wir genug Terme zur Verfügung. Da Δ maximal ist, ist die Theorie von $\mathfrak{M}$ vollständig festgelegt.

Die Trägermenge von $\mathfrak{M}$ besteht aus äquivalenzklassen von Termen. Für $\mathcal{L}^*$–Terme τ und σ schreiben wir $\tau{\sim}\sigma$ (τ und σ sind *äquivalent*) gdw. die Formel $\tau = \sigma$ zu Δ gehört.

Wir behaupten, dass $\sim$ eine äquivalenzrelation auf der Menge aller $\mathcal{L}^*$–Terme ist. Hierzu ist für beliebige $\mathcal{L}^*$–Terme τ, σ, ρ Folgendes zu zeigen:

(a) $\tau = \tau \in \Delta$.

(b) $\tau = \sigma \in \Delta \Rightarrow \sigma = \tau \in \Delta$

(c) $\tau = \sigma \in \Delta$ und $\sigma = \rho \in \Delta \Rightarrow \tau = \rho \in \Delta$

Betrachten wir etwa (b). Sei $\tau = \sigma \in \Delta$, d.h. $\Delta \vdash \tau = \sigma$. Die obige Tatsache (b) zur Gleichheit sagt, dass $\Delta \vdash \forall v_i \forall v_j (v_i = v_j \to v_j = v_i)$. Mit Hilfe der logischen Axiomenklasse (2) und zweifacher Anwendung des modus ponens sieht man dann

$$\Delta \vdash \tau = \sigma \to \sigma = \tau;$$

nochmalige Anwendung des modus ponens gibt schlielich $\Delta \vdash \sigma = \tau$, also

$$\sigma = \tau \in \Delta.$$

Analog beweist man (a) bzw. (c) mit Hilfe der Tatsachen (a) bzw. (c) zur Gleichheit.

Für einen $\mathcal{L}^*$–Term τ schreiben wir nun $[\tau]$ für $\{\sigma : \sigma{\sim}\tau\}$, d.h. für die zu τ gehörige äquivalenzklasse von Termen. Es gilt $\sigma{\sim}\tau$ gdw. $[\sigma] = [\tau]$.

Es sei dann M die Menge aller äquivalenzklassen $[\tau]$ für $\mathcal{L}^*$–Terme τ. M ist die Trägermenge unseres Modells $\mathfrak{M}$. Wir können nun $\mathfrak{M}$ definieren; hierzu müssen wir für jedes Prädikatsymbol P die Interpretation $P^{\mathfrak{M}}$, für jeden

Funktor f die Interpretation $f^{\mathfrak{M}}$ und für jede Konstante c die Interpretation $c^{\mathfrak{M}}$ angeben.

Sei zunächst c eine Konstante von $\mathcal{L}^*$. Wir setzen dann $c^{\mathfrak{M}} = [c]$. Sei f ein n–stelliger Funktor von $\mathcal{L}^*$ (d.h. von $\mathcal{L}$). Wir setzen dann $f^{\mathfrak{M}}([\tau_0], \dots, [\tau_{n-1}]) = [\sigma]$ gdw. die Formel $f\tau_0 \dots \tau_{n-1} = \sigma$ in Δ ist. Um zu zeigen, dass $f^{\mathfrak{M}}$ wohldefiniert ist, müssen wir zwei Dinge einsehen: $f^{\mathfrak{M}}$ ist "total", d.h. für beliebige Terme $\tau_0 \dots \tau_{n-1}$ existiert ein Term σ, so dass $f\tau_0 \dots \tau_{n-1} = \sigma$ in Δ ist; und: $f^{\mathfrak{M}}$ ist "funktional", d.h. für beliebige Terme $\tau_0 \dots \tau_{n-1}, \sigma, \tau'_0, \dots, \tau'_{n-1}, \sigma'$ folgt aus $\tau_0 \sim \tau'_0, \dots, \tau_{n-1} \sim \tau'_{n-1}$, $f\tau_0 \dots \tau_{n-1} \sim \sigma$ und $f\tau'_0 \dots \tau'_{n-1} \sim \sigma'$ so dass $\sigma \sim \sigma'$.

Die Totalität von $f^{\mathfrak{M}}$ ist trivial: man wähle einfach für σ den Term $f\tau_0 \dots \tau_{n-1}$. Die Funktionalität von $f^{\mathfrak{M}}$ ergibt sich wie folgt: die Annahme liefert mit Hilfe der Tatsache (e) zur Gleichheit, dass $\Delta \vdash f\tau_0 \dots \tau_{n-1} = f\tau'_0 \dots \tau'_{n-1}$, d.h. $f\tau_0 \dots \tau_{n-1} \sim f\tau'_0 \dots \tau'_{n-1}$; da $\sim$ äquivalenzrelation ist, ergibt die Annahme dann sofort $\sigma \sim \sigma'$.

Sei dann R ein n–stelliges Prädikatsymbol. Wir setzen dann $([\tau_0], \dots, [\tau_{n-1}]) \in R^{\mathfrak{M}}$ gdw. die Formel $R\tau_0 \dots \tau_{n-1}$ in Δ ist.

Mit Hilfe der Tatsache (d) zur Gleichheit sieht man dann wie im Falle eines Funktors, dass $R^{\mathfrak{M}}$ wohldefiniert ist. Wir haben damit $\mathfrak{M}$ definiert. Eine natürliche M–Belegung $\overline{\beta}$ ergibt sich wie folgt: $\overline{\beta}(v_i) = [v_i]$.

Wir wollen schlielich zeigen, dass für eine beliebige $\mathcal{L}^*$–Formel Folgendes gilt: $\mathfrak{M} \models \varphi[\overline{\beta}]$ gdw. $\varphi \in \Delta$. Damit gilt insbesondere $\mathfrak{M} \models \varphi[\overline{\beta}]$ für alle φ aus Γ. $\mathfrak{M}$ ist ein Modell von $\mathcal{L}^*$; wir erhalten daraus ein Modell $\overline{\mathfrak{M}}$ von $\mathcal{L}$, indem wir dasjenige "Redukt" von $\mathfrak{M}$ betrachten, das aus $\mathfrak{M}$ hervorgeht, indem man von den Interpretationen der neu hinzugefügten Konstanten $c(i, \overline{\varphi})$ absieht. Dann gilt auch $\overline{\mathfrak{M}} \models \varphi[\overline{\beta}]$ für alle $\varphi \in \Gamma$, und (**) ist bewiesen.

Sei β die durch $\overline{\beta}$ induzierte Terminterpretation. Man sieht leicht, dass $\beta(\tau) = [\tau]$ für alle Terme τ: für Variablen und Konstanten τ ergibt sich dies unmittelbar aus den Definitionen, und für ein τ von der Gestalt $f\tau_0 \dots \tau_{n-1}$ ergibt sich dies induktiv, da dann

$$\beta(f\tau_0 \dots \tau_{n-1}) = f^{\mathfrak{M}}(\beta(\tau_0), \dots, \beta(\tau_{n-1})) = f^{\mathfrak{M}}([\tau_0], \dots, [\tau_{n-1}]) = [f\tau_0 \dots \tau_{n-1}].$$

Wir beweisen jetzt die Aussage, dass $\mathfrak{M} \models \varphi[\overline{\beta}]$ gdw. $\varphi \in \Delta$ für alle $\mathcal{L}^*$–Formeln φ gilt, durch Induktion nach der Komplexität von φ.

Sei zuerst φ atomar. Sei etwa $\varphi \equiv \tau = \sigma$. Dann gilt $\mathfrak{M} \models \tau = \sigma[\overline{\beta}]$ gdw. $\beta(\tau) = \beta(\sigma)$ gdw. $[\tau] = [\sigma]$ gdw. $\tau \sim \sigma$ gdw. $\tau = \sigma \in \Delta$. Völlig analog argumentiert man, falls φ von der Gestalt $P\tau_0 \dots \tau_{n-1}$ für ein Prädikatsymbol P und Terme $\tau_0 \dots \tau_{n-1}$ ist.

Sei sodann $\varphi \equiv \neg\psi$. Dann gilt $\mathfrak{M} \models \neg\psi[\overline{\beta}]$ gdw. es nicht der Fall ist, dass $\mathfrak{M} \models \psi[\overline{\beta}]$ gdw. (nach Induktionsvoraussetzung) $\psi \notin \Delta$ gdw. $\neg\psi \in \Delta$.

Für $\psi \equiv \psi \to \psi'$ kann ähnlich argumentiert werden: $\mathfrak{M} \models \psi \to \psi'[\overline{\beta}]$ gdw. aus $\mathfrak{M} \models \psi[\overline{\beta}]$ folgt, dass $\mathfrak{M} \models \psi'[\overline{\beta}]$ gdw. (nach Induktionsvoraussetzung) aus $\psi \in \Delta$ folgt, dass $\psi' \in \Delta$ gdw. $\psi \to \psi' \in \Delta$.

Sei schlielich $\psi \equiv \forall v_i\psi$. Sei zuerst $\forall v_i\psi \in \Delta$ vorausgesetzt. Wir wollen sehen, dass $\mathfrak{M} \models \forall v_i\psi[\overline{\beta}]$, d.h. dass für alle $a \in M$, $\mathfrak{M} \models \psi[\overline{\beta}](v_i|a)]$. Sei $a \in M$ beliebig, $a = [\tau]$ für einen Term τ. Auf Grund der Möglichkeit der Variablen–Umbenennung existiert eine Formel ψ', so dass τ für v_i in ψ' eingesetzt werden kann und es gilt sowohl $\vdash \psi \to \psi'$ als auch $\vdash \psi' \to \psi$, auf Grund des Korrektheitssatzes also $\mathfrak{M} \models \psi[\overline{\beta}(v_i|a)]$ gdw. $\mathfrak{M} \models \psi'[\overline{\beta}(v_i|a)]$. Angenommen, $\mathfrak{M} \models \psi[\overline{\beta}(v_i|a)]$ gilt nicht; dann haben wir $\mathfrak{M} \models \neg\psi'[\overline{\beta}(v_i|[\tau])]$. Da $[\tau] = \beta(\tau)$, und da τ für v_i in ψ' eingesetzt werden kann, gilt dann wegen Lemma 4.3 $\mathfrak{M} \models \neg{\psi'}_\tau^{v_i}$. Nach Induktionsvoraussetzung ist dann $\neg{\psi'}_\tau^{v_i} \in \Delta$. Aus $\vdash \psi \to \psi'$ ergibt sich $\vdash \forall v_i(\psi \to \psi')$, und daraus mit Hilfe der logischen Axiome aus Klasse (3) $\forall v_i\psi \in \Delta$, und nach zweifacher Anwendung des modus ponens $\Delta \vdash \forall v_i\psi'$. Auf Grund eines Ersetzungsaxioms und einer weiteren Anwendung des modus ponens haben wir dann aber $\Delta \vdash \forall v_i\psi'$, also ${\psi'}_\tau^{v_i} \in \Delta$. Es ist also ${\psi'}_\tau^{v_i} \in \Delta$ und $\neg{\psi'}_\tau^{v_i} \in \Delta$ im Widerspruch zur Konsistenz von Δ.

Sei dann $\mathfrak{M} \models \forall v_i\psi[\overline{\beta}]$ vorausgesetzt. Wir müssen sehen, dass $\forall v_i\psi \in \Delta$. Hierfür benutzen wir, dass Δ eine Henkin–Menge ist. Angenommen, $\forall v_i\psi \notin \Delta$, also $\neg\forall v_i\psi \in \Delta$. Da für $c = c(i,\psi)$ gilt, dass $\neg\forall v_i\psi \to \neg\psi_c^{v_i} \in \Delta$, liefert der modus ponens $\Delta \vdash \neg\psi_c^{v_i}$, d.h. $\neg\psi_c^{v_i} \in \Delta$. Nach Induktionsvoraussetzung ist dann aber $\mathfrak{M} \models \neg\psi_c^{v_i}[\overline{\beta}]$, und eine Anwendung von Lemma 4.3 liefert $\mathfrak{M} \models \neg\psi[\overline{\beta}(v_i|\beta(c))]$. Dies widerspricht aber $\mathfrak{M} \models \forall v_i\psi[\overline{\beta}]$.

Damit ist der Vollständigkeitssatz bewiesen. □

Der Beweis des Vollständigkeitssatzes benutzt das Hausdorffsche Maximalitätsprinzip. Falls $\mathcal{L}$ höchstens abzählbar ist, wird das Maximalitätsprinzip nicht benötigt. $\mathcal{L}$ heit *höchstens abzählbar* gdw. $I \cup J \cup K$ höchstens abzählbar ist, d.h. gdw. es höchstens abzählbar viele Prädikatsymbole, Funktoren und Konstanten gibt. $\mathcal{L}^*$ entsteht dann aus $\mathcal{L}$ durch Hinzufügung abzählbar vieler Konstanten. Wir können uns dann (ähnlich wie im Beweis des Kompaktheitssatzes der Aussagenlogik) eine maximal–konsistente Menge $\Delta \supset \Gamma_H$ wie folgt konstruieren. Sei $(\varphi_n : n \in \mathbb{N})$ eine Aufzählung aller $\mathcal{L}^*$–Formeln. Sei $\Gamma_0 = \Gamma_H$. Sei $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$, falls $\Gamma_n \cup \{\varphi_n\}$ konsistent ist; andernfalls sei $\Gamma_{n+1} = \Gamma_n \cup \{\neg\varphi_1\}$. Schlielich sei $\Delta = \bigcup_{n\in\mathbb{N}} \Gamma_n$.

Mit Hilfe von Satz 4.7 hat der Vollständigkeitssatz trivialerweise die al-

lerdings erstaunliche Konsequenz, dass für eine rekursiv aufzählbare Formelmenge Γ auch $\{\varphi : \Gamma \models \varphi\}$ rekursiv aufzählbar ist.

Kapitel 6

Kompaktheit und Löwenheim–Skolem

Satz 6.1 *Sei* Γ *eine Menge von Formeln. Wenn jede endliche Teilmenge von* Γ *erfüllbar ist, dann ist auch* Γ *erfüllbar.*

Beweis: Der Kompaktheitssatz ist eine unmittelbare Konsequenz aus dem Vollständigkeitssatz. Angenommen, Γ ist nicht erfüllbar. Dann ist Γ inkonsistent, d.h. es existiert ein φ mit $\Gamma \vdash \varphi$ und $\Gamma \vdash \neg\varphi$. Dann gibt es natürlich eine endliche Teilmenge $\overline{\Gamma}$ von Γ mit $\overline{\Gamma} \vdash \varphi$ und $\overline{\Gamma} \vdash \neg\varphi$. $\overline{\Gamma}$ ist dann aber sicherlich nicht erfüllbar. □

Der Kompaktheitssatz hat erstaunliche Auswirkungen. Sei beispielsweise Γ eine Menge von Sätzen, so dass Γ beliebig große endliche Modelle besitzt, d.h. für jedes $n \in \mathbb{N}$ existiert ein $m \in \mathbb{N}, m \geq n$, und ein Modell $\mathfrak{M}$, dessen Trägermenge m Elemente besitzt, so dass $\mathfrak{M} \models \Gamma$. Dann besitzt Γ ein unendlich großes Modell $\mathfrak{M}$, d.h. es existiert ein $\mathfrak{M}$, dessen Trägermenge unendlich großist, so dass $\mathfrak{M} \models \Gamma$. Sei nämlich für $n \in \mathbb{N}$ φ_n der Satz

$$\exists v_0 \ldots \exists v_{n-1}(v_0 \neq v_1 \wedge v_0 \neq v_2 \wedge \ldots \wedge v_0 \neq v_{n-1} \wedge \\ v_1 \neq v_2 \wedge \ldots \wedge v_1 \neq v_{n-1} \wedge \ldots \wedge v_{n-2} \neq v_{n-1}),$$

der besagt, dass es mindestens n Dinge gibt. Nach Annahme ist jede endliche Teilmenge von $\Gamma \cup \{\varphi_n : n \in \mathbb{N}\}$ konsistent, so dass auch $\Gamma \cup \{\varphi_n : n \in \mathbb{N}\}$ konsistent ist.

Eine durch I, J, K, n gegebene Sprache der Logik 1. Stufe heißt höchstens abzählbar gdw. $I \cup J \cup K$ höchstens abzählbar ist.

Satz 6.2 (Löwenheim–Skolem) *Sei $\mathcal{L}$ eine höchstens abzählbare Sprache der Logik 1. Stufe. Sei Γ eine konsistente (erfüllbare) Menge von $\mathcal{L}$–Formeln. Dann besitzt Γ ein höchstens abzählbares Modelle, d.h. es existiert ein Modell $\mathfrak{M}$, dessen Trägermenge M höchstens abzählbar ist, und es existiert eine M–Belegung $\overline{\beta}$ mit $\mathfrak{M} \models \Gamma[\overline{\beta}]$.*

Beweis: Dies ergibt sich aus dem Beweis des Vollständigkeitssatzes. Mit $\mathcal{L}$ ist auch das dort definierte $\mathcal{L}^*$ höchstens abzählbar, so dass $\mathcal{L}^*$ abzählbar viele Terme enthält. Damit ist das dort konstruierte $M = \{[\tau] : \tau \text{ ist } \mathcal{L}^*\text{–Term}\}$ auch höchstens abzählbar. □

Wir werden weiter unten eine allgemeinere Version des Satzes von Löwenheim–Skolem formulieren.

Betrachten wir etwa das Modell

$$\mathcal{N} = (\mathbb{N}; 0, S, <, +, \cdot, E),$$

das wir bereits kurz in Kapitel 3 erwähnt haben. Zur Erinnerung: Dieses Modell besitzt das Universum $\mathbb{N}$ (= die Menge der natürlichen Zahlen = $\{0, 1, 2, \dots\}$). 0 ist die Null. S ist die Nachfolgeroperation, die definiert ist durch $S(n) = n + 1$ für $n \in \mathbb{N}$. $<$ ist die übliche strikte Ordnung auf $\mathbb{N}$. $+, \cdot$ und E bezeichnen die Addition, die Multiplikation und die Exponentiation.

Die zum Modell $\mathcal{N}$ gehörige Sprache der elementaren Zahlentheorie, die wir nun $\mathcal{L}_A$ nennen, enthält die Symbole 0 als Konstante, S als 1–stelligen Funktor, $<$ als 2–stelliges Relationssymbol, sowie $+, \cdot$ und E als 2–stellige Funktoren. Wir beabsichtigen natürlich $0^{\mathcal{N}} = 0, S^{\mathcal{N}} = S, <^{\mathcal{N}} = <, +^{\mathcal{N}} = +, \cdot^{\mathcal{N}} = \cdot$ und $E^{\mathcal{N}} = E$.

Hier ist eine Liste einiger Sätze φ der Sprache $\mathcal{L}_A$ mit $\mathcal{N} \models \varphi$:

(1) $\forall v_1 S(v_1) \neq 0$

(2) $\forall v_1 \forall v_2 (S(v_1) = S(v_2) \to v_1 = v_2)$

(3) $\forall v_1 \forall v_2 (v_1 < S(v_2) \leftrightarrow (v_1 < v_2 \vee v_1 = v_2))$

(4) $\forall v_1 \neg v_1 < 0$

(5) $\forall v_1 \forall v_2 (v_1 < v_2 \vee v_1 = v_2 \vee v_2 < v_1)$

(6) $\forall v_1 v_1 + 0 = v_1$

(7) $\forall v_1 \forall v_2 v_1 + S(v_2) = S(v_1 + v_2)$

(8) $\forall v_1 v_1 \cdot 0 = 0$

(9) $\forall v_1 \forall v_2 v_1 \cdot S(v_2) = v_1 \cdot v_2 + v_1$

(10) $\forall v_1 v_1 E 0 = S(0)$

(11) $\forall v_1 \forall v_2 v_2 E S(v_2) = (v_1 E v_2) \cdot v_1$

Wir wollen nun zeigen, dass $\mathcal{N}$ nicht das einzige Modell dieser Sätze ist. Genauer: Wir wollen zeigen, dass es Modelle dieser Sätze gibt, die *nicht isomorph* zu $\mathcal{N}$ sind.

Wir wollen nun zunächst zeigen, dass Modelle der obigen Sätze (1) bis (3) und (6) bis (11) eine "isomorphe Kopie" von $\mathcal{N}$ enthalten.

Sei

$$\mathcal{M} = (M; 0^{\mathcal{M}}, S^{\mathcal{M}}, <^{\mathcal{M}}, +^{\mathcal{M}}, \cdot^{\mathcal{M}}, E^{\mathcal{M}})$$

ein Modell der Sätze (1) bis (3) und (6) bis (11). Betrachten wir die Funktion $\pi_{\mathcal{M}} = \pi : \mathbb{N} \to M$, die wie folgt definiert ist:

$$\pi(n) = \underbrace{S^{\mathcal{M}}(S^{\mathcal{M}}(\dots S^{\mathcal{M}}}_{n\text{–viele } S^{\mathcal{M}}}(0^{\mathcal{M}})\dots)),$$

d.h., $\pi(0) = 0^{\mathcal{M}}$ und $\pi(n+1) = S^{\mathcal{M}}(\pi(n))$ für $n \in \mathbb{N}$.

Sei $\pi(m) = \pi(n)$ für $m \leq n \in \mathbb{N}$. Auf Grund von (2) ist dann $\pi(n-m) = 0^{\mathcal{M}}$, auf Grund von (1) also $n - m = 0$, d.h. $m = n$. Es ist also π injektiv. Wir werden später sehen, dass f nicht surjektiv sein muss.

Sei $m \in \mathbb{N}$. Mit Hilfe von (3) zeigt man dann leicht induktiv, dass $f(m) <^{\mathcal{M}} f(n)$ für alle $n > m$ gilt.

Ebenso leicht zeigt man mit Hilfe von (6) bis (11), dass für alle $m, n \in \mathbb{N}$ gilt: $\pi(m+n) = \pi(m) +^{\mathcal{M}} \pi(n), \pi(m \cdot n) = \pi(m) \cdot^{\mathcal{M}} \pi(n)$ und $\pi(m^n) = \pi(m) E^{\mathcal{M}} \pi(n)$.

Wir haben damit gesehen, dass π ein Monomorphismus, d.h. ein injektiver Homomorphismus, ist.

Nehmen wir nun zusätzlich an, dass $\mathcal{N}$ und $\mathcal{M}$ isomorph sind. Sei $\pi' : \mathbb{N} \to M$ ein Isomorphismus. Es ist leicht induktiv zu zeigen, dass dann $\pi'(n) = \pi_{\mathcal{M}}(n)$ für alle $n \in \mathbb{N}$ gelten muss. Es gilt also

Satz 6.3 *Wenn $\mathcal{M}$ ein Modell der Sätze (1) bis (3) und (6) bis (11) ist, dann ist $\mathcal{M}$ isomorph zu $\mathcal{N}$, gdw. $\pi_{\mathcal{M}}$ surjektiv ist.*

Sei nun $\mathcal{M}$ ein Modell der Sätze (1) bis (11). Mit Hilfe von (3) bis (5) ist leicht zu sehen, dass für alle $n \in \mathbb{N}$ und für alle x außerhalb des Wertebereichs von $\pi_{\mathcal{M}}$ gelten muss: $\pi_{\mathcal{M}}(n) <^{\mathcal{M}} x$. Ein solches x ist also "unendlich groß". Wir bezeichnen mit $\mathrm{Th}(\mathcal{N})$ die Menge aller Sätze φ der Sprache $\mathcal{L}_A$ mit $\mathcal{N} \models \varphi$. (Th steht für "Theorie".) Selbstverständlich gilt

$$\mathcal{N} \models \mathrm{Th}(\mathcal{N}),$$

d.h. $\mathcal{N} \models \varphi$ für alle $\varphi \in \mathrm{Th}(\mathcal{N})$. Wir wollen weitere Modelle $\mathcal{M}$ mit

$$\mathcal{M} \models \mathrm{Th}(\mathcal{N})$$

kennen lernen.

Satz 6.4 *Es existiert ein $\mathcal{M}$ mit abzählbarer Trägermenge und $\mathcal{M} \models \mathrm{Th}(\mathcal{N})$, so dass $\mathcal{M}$ und $\mathcal{N}$ nicht isomorph sind.*

Auf Grund dieses Satzes wird $\mathcal{N}$ als das *Nichtstandardmodell* von $\mathrm{Th}(\mathcal{N})$ bezeichnet. Jedes Modell von $\mathrm{Th}(\mathcal{N})$, das nicht zum Standardmodell isomorph ist, heißt ein *Nonstandardmodell*. Der Satz sagt also, dass es Nonstandardmodelle von $\mathrm{Th}(\mathcal{N})$ gibt.

Beweis: Der Beweis ist eine Anwendung des Kompaktheitssatzes. Wir erweitern die Sprache $\mathcal{L}_A$ durch Hinzunahme einer neuen Konstanten, c. Nennen wir die so erweiterte Sprache $\mathcal{L}_c$.

Sei $n \in \mathbb{N}$. Dann ist

$$\neg \underbrace{S(S(S \ldots (S}_{n \text{ viele } S}(0)) = c$$

ein Satz der Sprache $\mathcal{L}_c$. (Dieser Satz sagt, dass das durch c bezeichnete Objekt verschieden von n ist.) Nennen wir diesen Satz φ_n.

Wir betrachten nun die Menge

$$T = \mathrm{Th}(\mathcal{N}) \cup \{\varphi_n | n \in \mathbb{N}\}$$

von Sätzen der Sprache $\mathcal{L}_c$. T ist erfüllbar. Sei nämlich $\overline{T} \subset T$ endlich. Aufgrund des Kompaktheitssatzes genügt es zu zeigen, dass $\overline{T}$ erfüllbar ist.

Es existiert ein $n_0 \in \mathbb{N}$, so dass

$$\overline{T} \subset \mathrm{Th}(\mathcal{N}) \cup \{\varphi_n | n < n_0\}.$$

Es ist dann einfach zu sehen, dass

$$\mathcal{N}' = (\mathbb{N}; 0, S, <, +, \cdot, E, n_0) \models \overline{T}.$$

Hierbei beabsichtigen wir $0^{\mathcal{N}'} = 0, S^{\mathcal{N}'} = S, <^{\mathcal{N}'} = <, +^{\mathcal{N}'} = +, \cdot^{\mathcal{N}'} = \cdot, E^{\mathcal{N}'} = E$ und $c^{\mathcal{N}'} = n_0$.

Sei nun

$$\widetilde{\mathcal{M}} = (\widetilde{M}; 0^{\widetilde{\mathcal{M}}}, S^{\widetilde{\mathcal{M}}}, <^{\widetilde{\mathcal{M}}}, +^{\widetilde{\mathcal{M}}}, \cdot^{\widetilde{\mathcal{M}}}, E^{\widetilde{\mathcal{M}}}, c^{\widetilde{\mathcal{M}}}) \models T,$$

wobei $\widetilde{M}$ abzählbar ist. Ein solches $\widetilde{\mathcal{M}}$ existiert jetzt auf Grund des Satzes von Löwenheim–Skolem. Setze

$$\widetilde{\mathcal{M}} = (\widetilde{M}; 0^{\widetilde{\mathcal{M}}}, S^{\widetilde{\mathcal{M}}}, <^{\widetilde{\mathcal{M}}}, +^{\widetilde{\mathcal{M}}}, \cdot^{\widetilde{\mathcal{M}}}, E^{\widetilde{\mathcal{M}}}).$$

Selbstverständlich gilt

$$\widetilde{\mathcal{M}} \models \mathrm{Th}(\mathcal{N}).$$

Angenommen, $\pi = \pi_{\mathcal{M}}$ wäre surjektiv. Dann existiert ein $n \in \mathbb{N}$ mit $\pi(n) = c^{\widetilde{\mathcal{M}}}$. Es gilt $\widetilde{\mathcal{M}} \models \varphi_n$, d.h.

$$\widetilde{\mathcal{M}} \models \neg \underbrace{S(S(S \ldots (S}_{n \text{ viele } S}(0)) \ldots)) = c.$$

Dies besagt, dass

$$\pi(n) = \underbrace{S^{\mathcal{M}}(S^{\mathcal{M}}(\ldots S^{\mathcal{M}}}_{n \text{ viele } S^{\mathcal{M}}}(0^{\mathcal{M}}) \ldots)) \neq c^{\widetilde{\mathcal{M}}} = \pi(n).$$

Dies ist ein Widerspruch!

Der Monomorphismus $\pi_{\mathcal{M}}$ ist also nicht surjektiv, womit auf Grund von Satz 6.2 $\mathcal{M}$ nicht isomorph zu $\mathcal{N}$ ist. □

Sei M mit Trägermenge M ein Nonstandardmodell von $\mathrm{Th}(\mathcal{N})$. Jedes $x \in M$, das außerhalb des Wertebereichs von $\pi_{\mathcal{N}}$ liegt, heißt eine Nonstandardzahl. Ein $x \in M$ ist Nonstandardzahl gdw. $\pi_{\mathcal{M}}(n) <^{\mathcal{M}} x$ für alle $n \in \mathbb{N}$, d.h. gdw. x "unendlich groß" ist.

Wir wollen nun zeigen, dass es "sehr viele" paarweise nicht isomorphe abzählbare Modelle von $\mathrm{Th}(\mathcal{N})$ gibt. Um genau formulieren zu können, war "sehr viele" hier bedeutet, benötigen wir einen Begriff.

Sei M eine Menge. Wir sagen, dass M reellviele Elemente besitzt, falls eine Bijektion $f : M \to \mathcal{P}(\mathbb{N})$ existiert. Hierbei ist $\mathcal{P}(\mathbb{N})$ die Menge aller Teilmengen von $\mathbb{N}$. Insbesondere hat $\mathcal{P}(\mathbb{N})$ selbst reellviele Elemente. Wir sagen, dass die Menge M weniger als reellviele Elemente besitzt, falls keine Surjektion $f : M \to \mathcal{P}(\mathbb{N})$ existiert. Beispielsweise besitzt $\mathbb{N}$ weniger als reellviele Elemente: wenn $f : \mathbb{N} \to \mathcal{P}(\mathbb{N})$, dann ist

$$\{n \in \mathbb{N} | n \notin f(n)\} \in \mathcal{P}(\mathbb{N})$$

nicht im Wertebereich von f.

Die Aussage, dass jede überabzählbare Teilmenge von $\mathbb{R}$ reellviele Elemente besitzt, heißt *Kontinuumshypothese*. Diese Aussage wird in der Mengenlehre studiert.

Für den Beweis des folgenden Satzes benötigen wir eine Tatsache, die wir hier nicht zeigen können.

Tatsache: Sei $\mathcal{Z}$ eine Menge. Angenommen, $\mathcal{Z}$ besitzt weniger als reellviele Elemente und jedes $\mathcal{X} \in \mathcal{Z}$ ist eine abzählbare Menge. Dann ist auch

$$\bigcup \mathcal{Z} = \{X | \exists \mathcal{X} \in \mathcal{Z} X \in \mathcal{X}\}$$

eine Menge, die weniger als reellviele Elemente besitzt.

Es ist nicht schwierig zu zeigen, dass es höchstens reellviele nicht–isomorphe abzählbare Modelle von $\mathcal{L}_A$ geben kann. Der nachfolgende Satz sagt, dass es sogar mindestens so viele Modelle von $\mathrm{Th}(\mathcal{N})$ gibt.

Satz 6.5 *Es existieren reellviele paarweise nicht–isomorphe abzählbare Modelle $\mathcal{M}$ mit $\mathcal{M} \models \mathrm{Th}(\mathcal{N})$.*

Beweis: Wir erweitern wieder die Sprache $\mathcal{L}_A$ durch Hinzunahme einer neuen Konstanten c und nennen die so erweiterte Sprache $\mathcal{L}_c$.

Sei $p_0, p_1, p_2, \ldots$ die strikt monotone Aufzählung aller Primzahlen. D.h., $p_0 = 2, p_1 = 3, p_2 = 5$, etc.

Sei $n \in \mathbb{N}$. Dann bezeichnen wir mit φ_n den Satz

$$\exists v_1 \underbrace{S(S(S \ldots (S}_{p_n \text{ viele } S}(0)) \ldots)) \cdot v_1 = c$$

der Sprache $\mathcal{L}_c$. Dieser Satz sagt, dass p_n (die Interpretation von) c teilt. Mit $\overline{\varphi}_n$ bezeichnen die Negation von φ_n, d.h. den Satz

$$\neg\exists v_1 \underbrace{S(S(S \ldots (S}_{p_n \text{ viele } S}(0)) \ldots)) \cdot v_1 = c.$$

Sei nun $X \subset \mathbb{N}$. Wir bezeichnen dann mit T_X die Menge

$$\mathrm{Th}(\mathcal{N}) \cup \{\varphi_n | n \in X\} \cup \{\overline{\varphi}_n | n \notin X\}$$

von Sätzen der Sprache $\mathcal{L}_c$. Für jedes $X \subset \mathbb{N}$ ist T_X ist erfüllbar. Sei nämlich $\overline{T} \subset T_X$ endlich. Auf Grund des Kompaktheitssatzes genügt es zu zeigen, dass $\overline{T}$ erfüllbar ist.

Es existiert ein $n_0 \in \mathbb{N}$, so dass

$$\overline{T} \subset \mathrm{Th}(\mathcal{N}) \cup \{\varphi_n | n < n_0 \wedge n \in X\} \cup \{\overline{\varphi}_n | n < n_0 \wedge n \notin X\}.$$

Setze

$$q = \prod_{n<n_0, n\in X} p_n.$$

Es ist dann einfach zu sehen, dass

$$\mathcal{N}' = (\mathbb{N}; 0, S, <, +, \cdot, E, q) \models \overline{T}.$$

Hierbei beabsichtigen wir $0^{\mathcal{N}'} = 0, S^{\mathcal{N}'} = S, <^{\mathcal{N}'} = <, +^{\mathcal{N}'} = +, \cdot^{\mathcal{N}'} = \cdot, E^{\mathcal{N}'} = E$ und $c^{\mathcal{N}'} = q$. Für $X \subset \mathbb{N}$ sei nun

$$\widetilde{\mathcal{M}} = (M_X; 0^{\widetilde{\mathcal{M}}_X}, S^{\widetilde{\mathcal{M}}_X}, <^{\widetilde{\mathcal{M}}_X}, +^{\widetilde{\mathcal{M}}_X}, \cdot^{\widetilde{\mathcal{M}}_X}, E^{\widetilde{\mathcal{M}}_X}, c^{\widetilde{\mathcal{M}}_X}) \models T_{\mathcal{X}},$$

wobei M_X abzählbar ist. Ein solches $\widetilde{\mathcal{M}}_X$ existiert auf Grund der Behauptung 1 und des Satzes von Löwenheim–Skolem. Setze

$$\mathcal{M}_X = (M_X; 0^{\widetilde{\mathcal{M}}_X}, S^{\widetilde{\mathcal{M}}_X}, <^{\widetilde{\mathcal{M}}_X}, +^{\widetilde{\mathcal{M}}_X}, \cdot^{\widetilde{\mathcal{M}}_X}, E^{\widetilde{\mathcal{M}}_X}).$$

Selbstverständlich gilt

$$\mathcal{M}_X \models \mathrm{Th}(\mathcal{N}).$$

Für $X, Y \subset \mathbb{N}$ ist es möglich, dass $\mathcal{M}_X$ und $\mathcal{M}_Y$ isomorph sind. Wir wollen jedoch zeigen, dass eine Teilmenge von $\{\mathcal{M}_X | X \subset \mathbb{N}\}$ mit reellvielen Elementen existiert, die aus paarweise nicht–isomorphen Modellen besteht. Sei

$$\mathcal{M} = (M; 0^{\mathcal{M}}, S^{\mathcal{M}}, <^{\mathcal{M}}, +^{\mathcal{M}}, \cdot^{\mathcal{M}}, E^{\mathcal{M}}) \models \mathrm{Th}(\mathcal{N}).$$

Für jedes $x \in M$ sei

$$X_x^{\mathcal{M}} = \{n \in \mathbb{N} : \mathcal{M} \models \exists v_1 \underbrace{S(S(S \ldots (S}_{p_n \text{ viele } S}(0)) \ldots)) \cdot v_1 = v_2[\bar{\beta}] \text{ mit } \bar{\beta}(v_2) = x\}.$$

Jedes solche $X_x^{\mathcal{M}}$ ist eine Menge natürlicher Zahlen, die durch das Modell $\mathcal{M}$ und $x \in M$ "kodiert" wird. Sei

$$\mathcal{X}^{\mathcal{M}} = \{X_x^{\mathcal{M}} : x \in M\}$$

die Menge aller Teilmengen von $\mathbb{N}$, die durch $\mathcal{M}$ und seine Elemente in diesem Sinne "kodiert" werden. Offensichtlich ist $\mathcal{X}^{\mathcal{M}}$ eine abzählbare Menge

von Teilmengen von $\mathbb{N}$, wenn M abzählbar ist. Für jedes $X \subset \mathbb{N}$ gilt offensichtlich

$$X = X^{\mathcal{M}_X}_{c^{\mathcal{M}_X}} \in \mathcal{X}^{\mathcal{M}_X}.$$

Wir zeigen nun: Seien $\mathcal{M}$ und $\mathcal{P}$ isomorphe Modelle von $\mathrm{Th}(\mathcal{N})$. Dann gilt $\mathcal{X}^{\mathcal{M}} = \mathcal{X}^{\mathcal{P}}$. Sei nämlich $\pi : M \to P$ ein Isomorphismus. Sei $X \in \mathcal{X}^{\mathcal{M}}$. Sei $x \in M$ so, dass $X = X^{\mathcal{M}}_x$. Sei $\bar{\beta} : \{v_1, v_2, \dots\} \to M$, und sei $\bar{\beta}' : \{v_1, v_2, \dots\} \to P$ definiert durch $\bar{\beta}'(v_k) = \pi(\bar{\beta}(v_k))$. Für jedes $n \in \mathbb{N}$ gilt dann auf Grund von Satz 6.1

$$\mathcal{M} \models \exists v_1 \underbrace{S(S(S \dots (S}_{p_n \text{ viele } S}(0)) \dots)) \cdot v_1 = v_2[\bar{\beta}] \Leftrightarrow$$
$$\mathcal{P} \models \exists v_1 \underbrace{S(S(S \dots (S}_{p_n \text{ viele } S}(0)) \dots)) \cdot v_1 = v_2[\bar{\beta}].$$

Damit haben wir $X = X^{\mathcal{P}}_{\pi(x)}$, also $X \in \mathcal{X}^{\mathcal{P}}$.

Wir haben gezeigt, dass $\mathcal{X}^{\mathcal{M}} \subset \mathcal{X}^{\mathcal{P}}$. Aus Symmetriegründen gilt ebenso $\mathcal{X}^{\mathcal{P}} \subset \mathcal{X}^{\mathcal{M}}$. Damit ist $\mathcal{X}^{\mathcal{M}} = \mathcal{X}^{\mathcal{P}}$.

Kehren wir nun zu den Modellen $\mathcal{M}_X$ für $X \subset \mathbb{N}$ zurück. Für $X \subset \mathbb{N}$ sei

$$[\mathcal{M}_X] = \{\mathcal{M}_Y | Y \subset \mathbb{N} \text{ und } \mathcal{M}_Y \text{ ist isomorph mit } \mathcal{M}_X\}$$

die äquivalenzklasse aller zu $\mathcal{M}_X$ isomorphen Modelle $\mathcal{M}_Y$. Wir definieren

$$\mathcal{X}^{[\mathcal{M}_X]} = \mathcal{X}^{\mathcal{M}_X}.$$

Auf Grund der Behauptung 2 folgt aus $[\mathcal{M}_X] = [\mathcal{M}_Y]$, dass $\mathcal{X}^{\mathcal{M}_X} = \mathcal{X}^{\mathcal{M}_Y}$. Auf Grund dieser Unabhängigkeit von der Wahl des Repräsentanten ist $\mathcal{X}^{[\mathcal{M}_X]}$ wohldefiniert. Jedes $\mathcal{X}^{[\mathcal{M}_X]}$ abzählbar.

Nehmen wir nun an, es gäbe weniger als reellviele äquivalenzklassen $[\mathcal{M}_X]$. Dann wäre auf Grund der oben unbewiesen mitgeteilten Tatsache

$$\mathcal{Z} = \bigcup \{\mathcal{X}^{[\mathcal{M}_X]} | X \subset \mathbb{N}\}$$

eine Menge, die weniger als reellviele Elemente besitzt.

Auf der anderen Seite gilt für ein beliebiges $X \subset \mathbb{N}$, dass $X \in \mathcal{X}^{\mathcal{M}_X} = \mathcal{X}^{[\mathcal{M}_X]} \subset \mathcal{Z}$. Also enthält $\mathcal{Z}$ *alle* Teilmengen von $\mathbb{N}$, besteht also aus reellvielen Elementen. Widerspruch! □

Wir formulieren nun eine allgemeinere Version des Satzes von Löwenheim–Skolem.

Definition 6.6 *Sei $\mathcal{L}$ eine Sprache der Logik 1. Stufe, und seien $\mathcal{M}$ und $\mathcal{N}$ Modelle von $\mathcal{L}$. Dann heißt $\mathcal{M}$* elementar äquivalentes Submodell *von $\mathcal{N}$, geschrieben $\mathcal{M} \prec \mathcal{N}$ gdw. für alle $\mathcal{L}$–Formeln φ und für alle M–Belegungen $\bar{\beta}$ (wobei M die Trägermenge von $\mathcal{M}$ ist) gilt:*

$$\mathcal{M} \models \varphi[\bar{\beta}] \text{ gdw. } \mathcal{N} \models \varphi[\bar{\beta}].$$

Satz 6.7 (Allgemeiner Löwenheim–Skolem) *Sei $\mathcal{L}$ eine Sprache der Logik 1. Stufe, die durch I, J, K, n gegeben ist. Sei Γ eine Menge von Formeln, so dass $\mathcal{M}$ und $\overline{\beta}$ existieren, wobei $\mathcal{M} \models \Gamma[\overline{\beta}]$ und die Trägermenge von $\mathcal{M}$ unendlich großist. Sei X eine beliebige unendliche Menge, wobei eine Injektion $f : I \cup J \cup K \to X$ existiert. Dann existieren $\mathcal{N}$ und $\overline{\beta}'$, wobei $\mathcal{M} \models \Gamma[\overline{\beta}']$ und X die Trägermenge von $\mathcal{M}$ ist. Ebenfalls existiert ein $\mathcal{N}$, so dass $\mathcal{N} \prec \mathcal{M}$ oder $\mathcal{M} \prec \mathcal{N}$ und so dass es eine Bijektion von X mit der Trägermenge von $\mathcal{N}$ gibt.*

Insbesondere besitzen konsistente Formelmengen beliebig große Modelle.

Beweis: Der Allgemeine Löwenheim–Skolem–Satz ergibt sich schnell aus dem Beweis des Vollständigkeitssatzes. Wir erweitern zunächst $\mathcal{L}$ zu $\mathcal{L}'$ durch Hinzufügung einer eigenen Konstante c_x für jedes $x \in X$. Sodann erweitern wir Γ zu Γ' durch Hinzufügung genau aller $\mathcal{L}'$–Sätze $c_x \neq c_y$ für $x \neq y, x, y \in X$. Γ' ist dann konsistent. Wir produzieren nun wie im Beweis des Vollständigkeitssatzes $\mathcal{M}$ und $\overline{\beta}$ (ausgehend von $\mathcal{L}'$ und Γ' an Stelle von $\mathcal{L}$ und Γ). Es lässt sich zeigen, dass eine Bijektion zwischen X und der Trägermenge M von $\mathcal{M}$ existiert. Wir können dann die Elemente von M durch X ersetzen und erhalten $\mathcal{M}, \overline{\beta}$ wie gewünscht. Der Rest ist einfach.□

Insbesondere existiert z.B. ein Modell von $\mathrm{Th}(\mathcal{N})$, dessen Trägermenge die Menge $\mathbb{R}$ aller reellen Zahlen ist!

Kapitel 7

Das Halteproblem und der 1. Gödelsche Unvollständigkeitssatz

Wir haben gesehen, dass für ein rekursiv aufzählbares Γ die Menge $\{\varphi : \Gamma \models \varphi\} = \{\varphi : \Gamma \vdash \varphi\}$ auch rekursiv aufzählbar ist. Es stellt sich die Frage, ob für ein entscheidbares Γ die Menge $\{\varphi : \Gamma \vdash \varphi\}$ entscheidbar ist. Da mathematische Axiomensysteme üblicherweise entscheidbar sind, würde dies im positiven Falle bedeuten, dass ein Computer prinzipiell entscheiden könnte, ob eine gegebene Aussage φ unter Zugrundelegung des Axiomensystems Γ beweisbar ist oder nicht. Wir werden nun sehen, dass jedoch im Allgemeinen $\{\varphi : \Gamma \vdash \varphi\}$ nicht entscheidbar ist. Unser Schlüssel zu dieser Frage ist das *Halteproblem*, dem wir uns nun zunächst zuwenden. Das Problem lautet: lässt sich entscheiden, ob eine gegebene Turing–Maschine $\top$ bei einer gegebenen Eingabe w hält oder nicht, d.h. ob $\top(w)\downarrow$ oder $\top(w)\uparrow$?[1]

Wir wollen zunächst eine *universelle Turing–Maschine* bauen. Eine solche ist eine Turing–Maschine, die so programmiert ist, dass sie alle Berechnungen beliebiger Turing–Maschinen simulieren kann. Wir erinnern uns daran, dass eine Turing–Maschine $\top$ durch endlich viele Zustände Q, ein endliches Alphabet Γ (wovon ein nichtleeres $\Sigma \subsetneqq \Gamma$ das Eingabealphabet ist) und eine (dann ebenfalls endliche) übergangsfunktion δ gegeben ist. Es sei etwa

$$Q = \{q_0, q_+, q_-, q_1, q_2, \dots, q_{n-1}\}$$

[1] Hierbei schreiben wir $\top(w)\downarrow$ für $(\top(w)\downarrow + \text{ oder } (\top(w)\downarrow -)$.

und

$$\Gamma = \{\sqcup, x_0, x_1, \ldots, x_{m-1}\}.$$

Wir wollen $\top$ einen *Namen* $\#\top$ zuordnen. Die universelle Turing–Maschine U wird dann bei Eingabe von $\#\top, w$ den Rechenvorgang von $\top$ bei Eingabe von w simulieren. Dies wird zeigen, dass $\{w : \top(w)\downarrow\}$ zumindest rekursiv aufzählbar ist.

Wir wollen den Zustand q_+ von $\top$ mit $*, q_-$ mit $**$ und q_i, für $i < n$, mit der Folge $*\ldots*$ von $i+3$ Sternen identifizieren. Sodann wollen wir das Leerzeichen $\sqcup$ von $\top$ mit $|$ und das Symbol x_i, für $i < m$, mit der Folge $|\ldots|$ von $i+2$ Strichen identifizieren. Für "links" und "rechts" reservieren wir uns die Symbole L und R. Die übergangsfunktion δ von $\top$ können wir nun durch Auflistung ihres Graphen, d.h. aller 5–Tupel $(q, \gamma, q', \gamma', x)$ mit $\delta(q, \gamma) = (q', \gamma', x)$, mitteilen. Es gibt $l = (n+2)\cdot(m+1)$ derartige Tupel. Diese Auflistung können wir einfach in der Form

$$(*, |, q', \gamma', x)(**, |, q'', \gamma'', x')\ldots(*\ldots*, |\ldots|, q^{(l)}, \gamma^{(l)}, x^{(l-1)})$$

mitteilen. Diese Auflistung verwendet die 7 Symbole $(,)$, das Komma, $*, |, L$ und R; wir können sie einer Turing–Maschine eingeben. Wir wollen sie den *Namen von* $\top$, in Zeichen $\#\top$, nennen.

Unsere universelle Turing–Maschine U besitzt das Eingabealphabet $\Sigma_U = \{(,)$, das Komma, $*, |, L, R, \sqcup', ;\}$. U arbeitet sinnvoll bei einer Eingabe der Gestalt

$$\#\top(***\sqcup'\ldots\sqcup')w.$$

Hierbei ist $\#\top$ der Name einer Turing–Maschine $\top, \sqcup'\ldots\sqcup'$ ist eine Folge von $n-1$ "Leerzeichen" $\sqcup'$, und w ist (ein Code für) eine Eingabe, so dass U bei obiger Eingabe die Berechnung von $\top$ bei Eingabe von w simulieren wird. Hierbei können wir eine solche Eingabe w etwa in folgender Art und Weise kodieren. Das Eingabealphabet von $\top$ ist eine Teilmenge von $\{x_0, \ldots, x_{m-1}\}$, so dass w eine Folge von $x_i, i < m$, ist. Da wir $x_i, i < m$, mit einer Folge $|\ldots|$ von $i+2$ Strichen identifizieren, können wir w als Folge derartiger Strichfolgen, voneinander etwa durch das Komma, abgetrennt, ansehen. w sei also von der Gestalt[2]

$$;|\ldots|,|\ldots|,\ldots,|\ldots|.$$

Der Teil $(***\sqcup'\ldots\sqcup')$ der Eingabe dient als Platzhalter, so dass U festhalten kann, in welchem Zustand die simulierte Turing–Maschine $\top$ sich jeweils

[2]Die Rolle des ; am Anfang wird später erläutert.

befindet. Beim Start von U befindet sich $\top$ im Anfangszustand q_0 von $\top$, d.h. $***$.

U kann unschwer so gestaltet werden, dass die drei Teile der Eingabe, $\#\top$ (zur Mitteilung des "Programms" von $\top$), $(***\sqcup'\ldots\sqcup')$ (als Platzhalter für den Zustand von $\top$) und w (die Eingabe für $\top$) voneinander getrennt werden können.[3] Das Alphabet Γ_U von U sei Σ_U, zusammen mit dem Zeichen $\sqcup$.

Werde nun U mit der Eingabe

$$\#\top(***\sqcup'\ldots\sqcup')w$$

gefüttert, die so aussieht, wie soeben beschrieben. Zu einem späteren Zeitpunkt der Berechnung von U wird auf dem Band Folgendes niedergeschrieben sein:

$$\#\top(*\ldots*\sqcup'\ldots\sqcup')w'.$$

Hierbei ist $*\ldots*$ eine Folge von i Sternen, $1 \leq i < n+3$, und $\sqcup'\ldots\sqcup'$ ist eine Folge von $n+2-i$ "Leerzeichen"; dies teilt mit, dass die simulierte Turing–Maschine $\top$ gerade im Zustand q_+ (für $i=1$), q_- (für $i=z$), bzw. q_{i-3} (für $2<i<n+3$) ist. w' ist von der Gestalt

$$,|\ldots|,|\ldots|,\ldots;|\ldots|,\ldots,|\ldots|.$$

w' steht für einen Bandinhalt, den die simulierte Turing–Maschine $\top$ gerade auf ihrem Band vorfindet. ähnlich wie w oben stellt w' eine Folge von Symbolen aus $\{\sqcup, x_0, \ldots, x_{m-1}\}$ dar, die voneinander durch das Komma und ; abgetrennt sind. Wie in w so soll auch in w' das ; genau an einer Stelle vorkommen; dadurch wird markiert, dass der Kopf der simulierten Maschine $\top$ gerade auf dem Zeichen steht, das durch die dahinter stehende Strichfolge $|\ldots|$ kodiert wird.

Die Turing–Maschine U wird nun das durch ; markierte Zeichen, sowie den aktuellen Zustand von $\top$ (an Hand von $(*\ldots*\sqcup'\ldots\sqcup')$) einlesen, und sodann mit Hilfe von $\#\top$ (d.h. mit Hilfe der eingelesenen übergangsfunktion δ von $\top$) das markierte Zeichen durch ein neues Zeichen ersetzen, das Markierungszeichen ; nach links oder rechts versetzen (bzw. an derselben Stelle belassen, falls der Kopf von $\top$ als am linken Bandende stehend erkannt wird) und den neuen Zustand notieren. Dieser Vorgang, der einem Rechenschritt von $\top$ entspricht, wird einige Rechenschritte von U erfordern.

Schlielich soll U die ursprüngliche Eingabe $\#\top(***\sqcup'\ldots\sqcup')w$ akzeptieren/verwerfen, gdw. U als Zustand der simulierten Maschine $\top$ jemals $*$

[3] Eine Markierung des Bandanfangs ist übrigens nicht nötig. Wenn U beim Nach–Links–Gehen zweimal hintereinander (liest, dann ist das linke Bandende erreicht.

(d.h. q_+ von $\top$)/$**$ (d.h. q_- von $\top$) vorfindet. Falls dies niemals der Fall ist, so wird U unendlich lange weiterrechnen.

Wir können schlielich U auch so bauen, dass diese Maschine, falls sie nicht mit einer Eingabe der Form

$$\#\top(***\sqcup'\ldots\sqcup')w$$

wie beschrieben gefüttert wird, ebenfalls unendlich lange rechnet (U kann entscheiden, ob die Eingabe "vernünftig" ist und im gegenseitigen Fall von da ab einfach mit dem Kopf immer einen Schritt nach rechts gehen).

Unsere Turing–Maschine U hat also die folgende Eigenschaft: U akzeptiert/verwirft eine Eingabe gdw. diese von der Gestalt $\#\top(***\sqcup'\ldots\sqcup')w$ ist, wobei $\top(w)\downarrow +$ bzw. $\top(w)\downarrow -$. Wir haben damit folgenden Satz bewiesen.

Satz 7.1 *Die Mengen* $\{(\#\top, w) : \top(w)\downarrow +\}$, $\{(\#\top, w) : \top(w)\downarrow -\}$ *und* $\{(\#\top, w) : \top(w)\downarrow\}$ *sind rekursiv aufzählbar.*

Wir hatten $\{(\#\top, w) : \top(w)\downarrow\}$ das Halteproblem genannt. Wir zeigen nun mit Hilfe eines "Diagonalargumentes":

Satz 7.2 *Das Halteproblem* $\{(\#\top, w) : \top(w)\downarrow\}$ *ist nicht entscheidbar.*

Beweis: Sei H eine Turing–Maschine, für die Folgendes gilt: $H(\overline{w})\downarrow +$ gdw. $\overline{w}$ von der Gestalt $(\#\top, w)$ ist, wobei $\top$ eine Turing–Maschine ist mit $\top(w)\downarrow$; $H(\overline{w})\downarrow -$ gdw. $\overline{w}$ von der Gestalt $(\#\top, w)$ ist, wobei $\top$ eine Turing–Maschine ist mit $\top(w)\uparrow$, oder $\overline{w}$ ist nicht von der Gestalt $(\#\top, w)$. Wir können dann sehr einfach eine Turing–Maschine D bauen, für die gilt:

$$D(\#\top)\downarrow + \text{ gdw. } H((\#\top, \#\top))\downarrow -$$

und[4]

$$D(\#\top)\downarrow - \text{ gdw. } H((\#\top, \#\top))\downarrow +.$$

Dann gilt aber

$$D(\#D)\downarrow + \text{ gdw. } H((\#D, \#D))\downarrow - \text{ gdw. } D(\#D)\uparrow .$$

Dies ist ein Widerspruch! □

Wir wollen jetzt zeigen, dass die Nichtentscheidbarkeit des Halteproblems die

[4]Wir benötigen dies nur für Maschinen $\top$ in die $\#\top$ eingegeben werden kann.

Nichtentscheidbarkeit der elementaren Zahlentheorie liefert. Daraus werden wir schlielich den 1. Gödelschen Unvollständigkeitssatz ableiten können.

Die elementare Zahlentheorie, A_E, ist durch die Axiome (1) bis (11) aus dem letzten Kapitel gegeben. Eine Formel φ der Sprache der elementaren Zahlentheorie heit Σ_1 bzw. Π_1 gdw. φ eine Existenz– bzw. Allaussage ist.

Wir werden sehen, dass jede wahre Existenzaussage, d.h. jeder wahre[5] Σ_1–Satz, in A_E beweisbar ist. Wir werden sodann beobachten, dass der Sachverhalt $\top(w)\downarrow +$ als Σ_1–Satz $\varphi_{\top,w}$ der Sprache der elementaren Zahlentheorie geschrieben werden kann. Damit gilt also $\top(w)\downarrow +$ gdw. $A_E \vdash \varphi_{\top,w}$. Da aber nun $\{(\#\top, w) : \top(w)\downarrow +\}$ nicht entscheidbar ist, kann also die Menge der wahren (= in A_E beweisbaren) Σ_1–Sätze der Sprache der elementaren Zahlentheorie nicht entscheidbar sein. Wenn dann $A \supset A_E$ eine rekursiv aufzählbare Menge von wahren Aussagen der elementaren Zahlentheorie ist, dann muss ein wahrer Π_1–Satz φ existieren, so dass A den Satz φ nicht beweist. (Andernfalls wäre sowohl $\{\varphi : \varphi$ ist wahrer Σ_1–Satz$\}$ und $\{\varphi : \varphi$ ist wahrer Π_1–Satz$\}$ rekursiv aufzählbar, also beide Mengen auch entscheidbar nach Lemma 2.1.) Dasselbe Argument zeigt: wenn $A \supset A_E$ eine konsistente rekursiv aufzählbare Menge von Formeln der elementaren Zahlentheorie ist, dann existiert ein Π_1–Satz φ der Sprache der elementaren Zahlentheorie, so dass weder φ noch $\neg\varphi$ in A beweisbar ist. Dies ist die Aussage des 1. Gödelschen Unvollständigkeitssatzes. Der 2. Gödelsche Unvollständigkeitssatz sagt, dass ein hinreichend starkes rekursiv aufzählbares Axiomensystem A genau dann seine eigene Konsistenz beweist, wenn A *in*konsistent ist.

Wir bezeichnen mit $\mathcal{L}_A$ die Sprache der elementaren Zahlentheorie. Eine $\mathcal{L}_A$–Formel heit *beschränkt* gdw. sie zu allen Mengen S von $\mathcal{L}_A$–Formeln gehört, die Folgendes erfüllen:

(a) Jede atomare $\mathcal{L}_A$–Formel gehört zu S,

(b) mit φ und ψ gehören auch $\neg\varphi$ und $\varphi \to \psi$ zu S, und

(c) wenn φ zu S gehört, wenn τ ein $\mathcal{L}_A$–Term ist, in dem v_i nicht vorkommt, und wenn $i \in \mathbb{N}$, dann gehören auch $\forall v_i(v_i < \tau \to \varphi)$ und $\exists v_i(v_i < \tau \wedge \varphi)$ zu S.

Wir schreiben $\forall v_i(v_i < \tau \to \varphi)$ auch als $\forall v_i < \tau\varphi$ und $\exists v_i(v_i < \tau \wedge \varphi)$ auch als $\exists v_i < \tau\varphi$. Eine beschränkte Formel enthält keine "unbeschränkten" Quantoren. Eine beschränkte Formel heit auch Σ_0.

[5] Ein Satz φ der Sprache der elementaren Zahlentheorie heit *wahr* gdw. $\mathcal{N} \models \varphi$.

Eine $\mathcal{L}_A$–Formel φ heit Σ_1 gdw. $i_0, \dots, i_{n-1} \in \mathbb{N}$ und eine beschränkte $\mathcal{L}_A$–Formel ψ existieren, so dass

$$\varphi \equiv \exists v_{i_0} \dots \exists v_{i_{n-1}} \psi.$$

Eine $\mathcal{L}_A$–Formel φ heit Π_1 gdw. $i_0, \dots, i_{n-1} \in \mathbb{N}$ und eine beschränkte $\mathcal{L}_A$–Formel ψ existieren, so dass

$$\varphi \equiv \forall v_{i_0} \dots \forall v_{i_{n-1}} \psi.$$

Wir wollen nun zeigen, dass jeder wahre Σ_1–Satz in A_E beweisbar ist.

Für $n \in \mathbb{N}$ und einen Term τ schreiben wir $S^{(n)}(\tau)$ für den Term

$$\underbrace{SS \dots S}_{n \text{ viele } S} \tau.$$

Lemma 7.3 *Für jeden $\mathcal{L}_A$–Term τ, der keine Variablen enthält, existiert ein $n \in \mathbb{N}$, so dass $A_E \vdash \tau = S^{(n)}(0)$.*

Beweis durch Induktion nach der "Komplexität" von τ: Für $\tau \equiv 0$ ist die Aussage mit $n = 0$ trivial. Wenn $A_E \vdash \tau = S^{(n)}(0)$, dann gilt $A_E \vdash S\tau = S^{(n+1)}(0)$.

Seien nun τ, σ gegeben mit $A_E \vdash \tau = S^{(n)}(0)$ und $A_E \vdash \sigma = S^{(m)}(0)$.

Wir wollen sehen, dass $A_E \vdash \tau + \sigma = S^{(n+m)}(0)$. Zunächst ist $A_E \vdash \tau + \sigma = S^{(n)}(0) + S^{(m)}(0)$. Wir bekommen aber durch m–fache Anwendung von (7) von A_E der Reihe nach

$$A_E \vdash S^{(n)}(0) + S^{(m)}(0) = S(S^{(n)}(0) + S^{(m-1)}(0)), \dots,$$
$$A_E \vdash S^{(n)}(0) + S^{(m)}(0) = S^{(m)}(S^{(n)}(0) + 0),$$

mit Hilfe von (6) von A_E also

$$A_E \vdash S^{(n)}(0) + S^{(m)}(0) = S^{(m)}(S^{(n)}(0)),$$

d.h. $A_E \vdash \tau + \sigma = S^{(n+m)}(0)$.

Ganz analog zeigt man mit Hilfe von (9) und (8) von A_E, dass $A_E \vdash \tau \cdot \sigma = S^{(n \cdot m)}(0)$; mit Hilfe von (11) und (10) von A_E zeigt man, dass $A_E \vdash \tau E \sigma = S^{(n^m)}(0)$. □

Satz 7.4 *Sei φ ein wahrer Σ_1–Satz von $\mathcal{L}_1$. Dann gilt $A_E \vdash \varphi$.*

Beweis durch Induktion nach der "Komplexität" von φ. Wir müssen allerdings zunächst φ in "fast negationsfreie" Form bringen.
Eine $\mathcal{L}_A$–Formel heit *beschränkt fast negationsfrei* gdw. sie zu allen Mengen S von $\mathcal{L}_A$–Formeln gehört, die Folgendes erfüllen:

(a) Jede atomare $\mathcal{L}_A$–Formel gehört zu S,

(b) Mit einer atomaren $\mathcal{L}_A$–Formel φ gehört auch $\neg\varphi$ zu S,

(c) mit φ und ψ gehören auch $\varphi \wedge \psi$ und $\varphi \vee \psi$ zu S, und

(d) wenn φ zu S gehört, τ ein $\mathcal{L}_A$–Term ist und wenn $i \in \mathbb{N}$, dann gehören auch $\forall v_i < \tau\varphi$ und $\exists v_i < \tau\varphi$ zu S.

Man sieht leicht, dass für jede beschränkte Formel φ eine beschränkte fast negationsfreie Formel φ' existiert mit $\vdash \varphi \leftrightarrow \varphi'$.

Wir zeigen nun zuerst durch Induktion nach der "Komplexität" von φ, dass der Satz für jede beschränkte fast negationsfreie Formel φ gilt.

Sei zunächst φ atomar. Dann ist $\varphi \equiv \tau = \sigma$ oder $\varphi \equiv \tau < \sigma$, wobei in den Termen τ, σ keine Variablen vorkommen. Es existieren $n, m \in \mathbb{N}$ mit $A_E \vdash \tau = S^{(n)}(0)$ und $A_E \vdash \sigma = S^{(m)}(0)$. Sei zuerst $\varphi \equiv \tau = \sigma$. Da $\tau = \sigma$ wahr ist, muss $n = m$ gelten. Dann folgt aber $A_E \vdash \tau = \sigma$. Sei jetzt $\varphi \equiv \tau < \sigma$. Da $\tau < \sigma$ wahr ist, muss $n < m$ gelten. Dann sieht man mit Hilfe von (3) von A_E, dass $A_E \vdash \tau < \sigma$.

Sei nun $\varphi \equiv \neg\psi$, wobei ψ atomar ist. Mit Hilfe von (2) von A_E zeigt man dann einfach das Gewünschte.

Sei jetzt $\varphi \equiv \psi \wedge \psi'$ bzw. $\varphi \equiv \psi \vee \psi'$. Dann folgt die Aussage für φ sehr leicht daraus, dass die Aussage für ψ und ψ' gilt.

Sei dann $\varphi \equiv \forall v_i < \tau\psi$. Da φ ein Satz ist, können in τ keine Variablen vorkommen, so dass ein $n \in \mathbb{N}$ existiert mit $A_E \vdash \tau = S^{(n)}(0)$. Es gilt dann

$$A_E \vdash \forall v_i < \tau\psi \leftrightarrow (\psi_0^{v_i} \wedge \psi_{S^{(1)}(0)}^{v_i} \wedge \ldots \wedge \psi_{S^{(n-1)}(0)}^{v_i})$$

mit Hilfe von (3) und (4) von A_E. Da $\forall v_i < \tau\psi$ wahr ist, sind $\psi_0^{v_i}, \ldots, \psi_{S^{(1)}(0)}^{v_i}$ auf Grund der Induktionsvoraussetzung in A_E beweisbar, so dass $A_E \vdash \forall v_i < \tau\psi$ gilt.

Für $\varphi \equiv \exists v_i < \tau\psi$ zeigt man analog, dass

$$A_E \vdash \exists v_i\psi \leftrightarrow (\psi_0^{v_i} \vee \psi_{S^{(1)}(0)}^{v_i} \vee \ldots \vee \psi_{S^{(n-1)}(0)}^{v_i})$$

wobei $A_E \vdash \tau = S^{(n)}(0)$, so dass $A_E \vdash \exists v_i < \tau\psi$.

Damit gilt die zu zeigende Aussage für alle beschränkten Sätze.
Sei nun endlich

$$\varphi \equiv \exists v_{i_0} \dots \exists v_{i_{n-1}} \psi,$$

wobei ψ Σ_0 ist. Da φ wahr ist, existieren $m_{i_0}, \dots, m_{i_{n-1}} \in \mathbb{N}$, so dass $\psi' \equiv (\dots(\psi^{v_{i_0}}_{m_{i_0}})\dots)^{v_{i_{n-1}}}_{m_{i_{n-1}}}$ wahr ist. Dann folgt aber aus $A_E \vdash \psi'$, dass $A_E \vdash \varphi$.$\square$

Wir wollen jetzt sehen, dass es für jede Turing–Maschine $\top$ eine Σ_1–Formel $\chi_\top$ mit freier Variable v_0 gibt, so dass $\mathcal{N} \models \chi_\top(w)$ gdw. $\top(w)\downarrow$. Nun, "$\mathcal{N} \models \chi_\top(w)$" ist zunächst nicht sinnvoll, da w als Eingabe für $\top$ eine Symbolfolge, jedoch keine natürliche Zahl ist. Wir können aber Symbolfolgen mit Zahlen "identifizieren"; eine solche Identifizierung heit *"Gödelisierung"*.

Sei $\widetilde{\Gamma}$ ein endliches Alphabet, etwa $\widetilde{\Gamma} = \{\gamma_0, \dots, \gamma_{n-1}\}$. Wir können dann zuerst, für $i < n$, γ_i mit der natürlichen Zahl $i+1$ identifizieren. Sei $p_0 = 2, p_1 = 3, p_2 = 5, \dots$ die natürliche Aufzählung aller Primzahlen. Wir können dann ein Wort $w \in \widetilde{\Gamma}^*$, d.h. eine endliche Folge $\gamma_{i_0}\gamma_{i_1}\dots\gamma_{i_{m-1}}$ von Symbolen aus $\widetilde{\Gamma}$ der Länge $m \in \mathbb{N}$ mit der natürlichen Zahl

$$p_0^{i_0} \cdot p_1^{i_1} \cdot \ldots \cdot p_{m-1}^{i_{m-1}}$$

identifizieren. Jedes Wort entspricht damit einer Zahl und jeder Zahl der Gestalt

$$p_0^{i_0} \cdot p_1^{i_1} \cdot \ldots \cdot p_{m-1}^{i_{m-1}},$$

wobei $m \in \mathbb{N}$ und $i_k < n$ für alle $k < m$, entspricht ein Wort. Wir bezeichnen die dem Wort $w \in \widetilde{\Gamma}^*$ zugeordnete Zahl als die *Gödelnummer von* w, in Zeichen $\ulcorner w \urcorner$. Es gilt sodann:

Satz 7.5 *Für jede Turing–Maschine $\top$ existiert eine Σ_1–Formel $\chi_\top$ von $\mathcal{L}_A$ mit freier Variable v_0, so dass*

$$\mathcal{N} \models \chi_\top(\ulcorner w \urcorner) \text{ gdw. } \top(w)\downarrow$$

für alle Worte w im Eingabealphabet von $\top$.

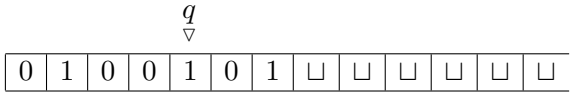
Beweis: Sei $\top$ gegeben durch Q, Γ (und Σ), sowie δ. Sei etwa

$$Q = \{q_0, q_+, q_-, q_1, q_2, \dots, q_{n-1}\}$$

und

$$\Gamma = \{\sqcup, x_0, x_1, \dots, x_{m-1}\}.$$

Wir setzen $\widetilde{\Gamma} = Q \cup \Gamma$.[6] Eine Konfiguration im Rahmen eines Rechenvorgangs von $\top$ wollen wir durch das aktuell auf dem Band niedergeschriebene Wort kodieren, wobei wir vor die Stelle, auf der der Kopf steht, den aktuellen Maschinenzustand einfügen. Beispielsweise kodieren wir die Konfiguration



durch die Folge

$$0\,1\,0\,0\,q\,1\,0\,1,$$

oder auch durch die Folge der Form

$$0\,1\,0\,0\,q\,1\,0\,1\,\sqcup \ldots \sqcup .$$

Ein Rechenvorgang von $\top$, der ja eine Folge von Konfigurationen ist, kann dann als Folge derartiger Folgen, oder mit Hilfe von einfacher Buchhaltung nur als Folge von Symbolen aus $\widetilde{\Gamma}$ angesehen werden.

Sei w eine Eingabe mit $\top(w)\downarrow$. Sei $l \in \mathbb{N}$ die Laufzeit von $\top$ bei Eingabe von w, d.h. der Rechenvorgang von $\top$ bei Eingabe von w besteht aus l Konfigurationen $K_r, r < l$, und bricht dann ab, da $\top$ in den Zustand q_+ oder q_- geraten ist. Sei $k \in \mathbb{N}$ minimal, so dass die ersten k Zellen bei diesem Rechenvorgang benötigt werden, d.h. so dass die $(k{+}1)^{\text{te}}$, $(k{+}2)^{\text{te}}$ etc. Zelle in keinem Rechenschritt besucht werden. Wir können dann den Rechenvorgang von $\top$ bei Eingabe von w sehr leicht durch eine Folge $s_0, s_1 \ldots s_{l(k+1)-1}$ der Länge $l(k+1)$ von Symbolen aus $\widetilde{\Gamma}$ kodieren: für $r < l$ kodiert der r^{te} Block $s_{r(k+1)} s_{r(k+1)+1} \ldots s_{r(k+1)+k}$ die r^{te} Konfiguration K_r. Offensichtlich können wir jetzt $\top(w)\downarrow$ wie folgt ausdrücken:

Es gibt $l, k \in \mathbb{N}$ und eine Folge $s_0, s_1 \ldots s_{l(k+1)-1}$ der Länge $l(k+1)$ von Symbolen aus $\widetilde{\Gamma}$, wobei gilt:

(a) Der nullte Block $s_0 s_1 \ldots s_{k-1}$ ist von der Gestalt q_0, gefolgt von w, gefolgt von Leerzeichen $\sqcup$.

(b) Für $r+1 < l$ ergibt sich der $(r+1)^{\text{te}}$ Block $s_{(r+1)(k+1)} \ldots s_{(r+1)(k+1)+k}$ aus dem r^{ten} Block $s_{r(k+1)} \ldots s_{r(k+1)+k}$ durch Anwendung der übergangsfunktion δ.

[6] Wir nehmen hierbei o.B.d.A. an, dass Q und Γ disjunkt sind.

(c) Der $(l-1)^{\text{te}}$ Block $s_{(l-1)(k+1)} \dots s_{(l-1)(k+1)+k}$ enthält eines der Symbole q_+ oder q_-.

Mit Hilfe der oben vorgenommenen Gödelisierung können wir dies wiederum wie folgt formulieren:

Es gibt $l, k, f \in \mathbb{N}$, wobei f von der Gestalt

$$p_0^{i_0} \cdot p_1^{i_1} \cdot \ldots \cdot p_{l(k+1)-1}^{i_{l(k+1)-1}}$$

ist mit[7] $i_j < n + m + 3$ und es gilt:

(a) $i_0 = q_0$, für ein $t < k + 1$ ist $i_1 i_2 \dots i_t$ das Wort w (des Eingabealphabets), wobei $v_0 = \ulcorner w \urcorner$, und $i_{t+1} = \ldots = i_k = \sqcup$.

(b) Für $r + 1 < l$ ergibt sich der $(r+1)^{\text{te}}$ Block $i_{(r+1)(k+1)} \dots i_{(r+1)(k+1)+k}$ aus dem r^{ten} Block $i_{r(k+1)} \dots i_{r(k+1)+k}$ durch Anwendung von δ.

(c) $i_u = q_+$ oder $i_u = q_-$ für ein u mit $(l-1)(k+1) \leq u < l(k+1)$.

Eine nähere Inspektion, die im Detail langwierig aber nicht schwierig zu bewerkstelligen ist, zeigt, dass wir schlielich eine Σ_1–Formel $\chi_\top$ finden, die das Gewünschte leistet. □

Korollar 7.6 *Die Menge aller wahren Σ_1–Sätze von $\mathcal{L}_A$ ist nicht entscheidbar.*

Beweis: Wir erinnern uns daran, dass unsere universelle Turing–Maschine U die folgende Eigenschaft hat:

$$U((\#\top, w)) \downarrow \text{ gdw. } \top(w) \downarrow .$$

Nach dem obigen Satz existiert eine Σ_1–Formel φ_U mit freier Variable v_0, so dass

$$\mathcal{N} \models \varphi_U(\ulcorner(\#\top, w)\urcorner) \text{ gdw. } U((\#\top, w)) \downarrow,$$

d.h., so dass

$$\mathcal{N} \models \varphi_U(\ulcorner(\#\top, w)\urcorner) \text{ gdw. } \top(w) \downarrow .$$

Nun gilt aber $\mathcal{N} \models \varphi_U(\ulcorner(\#\top, w)\urcorner)$ gdw. $\mathcal{N} \models (\varphi_U)^{v_0}_{S^{(l)}(0)}$ für $l = \ulcorner(\#\top, w)\urcorner$. Wäre die Menge aller wahren Σ_1–Sätze von $\mathcal{L}_A$ entscheidbar, dann wäre auch die Menge aller $(\varphi_U)^{v_0}_{S^{(l)}(0)}$, wobei $l = \ulcorner(\#\top, w)\urcorner$ für eine Turing–Maschine $\top$ und ein Eingabewort w, entscheidbar. Dann wäre aber auch $\{(\top, w) : \top(w) \downarrow\}$ entscheidbar. Widerspruch! □

[7] $\widetilde{\Gamma}$ enthält $n + m + 3$ viele Symbole, die wir mit $0, 1, \ldots, n + m + 2$ identifizieren.

Satz 7.7 (Erster Gödelscher Unvollständigkeitssatz) *Sei $A \supset A_E$ eine rekursiv aufzählbare konsistente Menge von $\mathcal{L}_A$–Formeln. Dann existiert ein Π_1–Satz φ von $\mathcal{L}_A$, so dass A weder φ noch $\neg\varphi$ beweist.*

Beweis: Da $A \supset A_E$, beweist A jeden wahren Σ_1–Satz von $\mathcal{L}_A$. Nehmen wir nun an, für alle Π_1–Sätze φ von $\mathcal{L}_A$ gilt $A \vdash \varphi$ oder $A \vdash \neg\varphi$. Dann wäre insbesondere die Menge aller Σ_1–Sätze φ mit $A \vdash \varphi$ entscheidbar, da nach Annahme A einen solchen Satz φ *nicht* beweist gdw. A den Satz $\neg\varphi$ beweist. Dann wäre nun aber wie im Beweis von Korollar 7.6 die Menge aller wahren Σ_1–Sätze der Form $(\varphi_U)^{v_0}_{S^{(l)}(0)}$, wobei $l = \ulcorner(\#\top, w)\urcorner$ für eine Turing–Maschine $\top$ und ein Eingabewort w, entscheidbar. Widerspruch! □

Sei $A \supset A_E$ eine Menge von $\mathcal{L}_A$–Sätzen, die nur aus wahren Sätzen besteht. Wenn dann φ ein Π_1–Satz ist, so dass A weder φ noch $\neg\varphi$ beweist, dann muss φ *wahr* sein. Wäre nämlich $\neg\varphi$ wahr, dann wäre (da $\neg\varphi$ logisch äquivalent zu einem Σ_1–Satz ist) $\neg\varphi$ in A_E, also auch in A, beweisbar.

Korollar 7.8 *Sei $\Gamma \supset A_E$ eine rekursiv aufzählbare konsistente Menge von $\mathcal{L}_A$–Formeln. Dann gilt es reellviele konsistente Mengen $\widetilde{\Gamma} \supset \Gamma$ von $\mathcal{L}_A$–Formeln, die sich paarweise widersprechen, d.h. für zwei verschiedene derartige Mengen $\widetilde{\Gamma}$ und $\widetilde{\Gamma}'$ gibt es eine $\mathcal{L}_A$–Formel φ mit $\varphi \in \widetilde{\Gamma}$ und $\neg\varphi \in \widetilde{\Gamma}'$.*

Beweis: Wir definieren zunächst rekursiv nach der Länge von s für jede endliche 0-1–Folge $s \in \{0,1\}^*$ eine Menge $\Gamma_s \supset \Gamma$. Sei $(\varphi_n : n \in \mathbb{N})$ eine Aufzählung aller $\mathcal{L}_A$–Formeln. Wir definieren jedes Γ_s so, dass Γ_s aus Γ durch Hinzunahme endlich vieler Formeln entsteht; insbesondere ist jedes Γ_s immer noch rekursiv aufzählbar.

Wir setzen $\Gamma_\emptyset = \Gamma$. Sei nun Γ_s definiert. Auf Grund des ersten Gödelschen Unvollständigkeitssates gibt es ein kleinstes $k = k(s)$, so dass Γ_s weder φ_k noch $\neg\varphi_k$ beweist. Wir setzen sodann $\Gamma_{s0} = \Gamma_s \cup \{\varphi_k\}$ und $\Gamma_{s1} = \Gamma_s \cup \{\neg\varphi_k\}$.[8]

Für jede unendliche 0-1–Folge f definieren wir $\Gamma_f = \bigcup\{\Gamma_s : s \subset f\}$ und $\widetilde{\Gamma}_f = \{\varphi : \Gamma_f \vdash \varphi\}$. Offenbar ist jedes Γ_f konsistent und die Familie aller $\widetilde{\Gamma}_f$ ist wie gewünscht. □

Wir wollen nun eine alternativen Beweis einer Variante des 1. Gödelschen Unvollständigkeitssatzes kennen lernen. Dieser Beweis wurde von S. Kripke

[8] Hierbei ist sh, für $h = 0, 1$, das Resultat des Anfügens des neuen Folgengliedes h an die Folge s.

gefunden. Er ist weniger indirekt als der oben angegebene Beweis: wir sehen tatsächlich eine Aussage, die weder beweisbar noch widerlegbar ist!

Im Folgenden sei $\mathcal{N}$ wieder das Modell $(\mathbb{N}; 0, 1, <, +, \cdot, E)$ und $\mathcal{L}_A$ sei die zugehörige Sprache der elementaren Zahlentheorie Stufe.

Sei s eine endliche oder unendliche (echt) aufsteigende Folge natürlicher Zahlen. Wir bezeichnen mit $(s)_k$ das k^{te} Element von s (falls $k < lh(s) =$ Länge von s; sonst ist $(s)_k$ nicht definiert). D.h. $s = (s)_0, (s)_1, \ldots, (s)_{lh(s)-1}$ mit $(s)_0 < (s)_1 < \ldots < (s)_{lh(s)-1}$ falls $lh(s) < \infty$ und $s = (s)_0, (s)_1, \ldots$ mit $(s)_0 < (s)_1 < \ldots$ falls $lh(s) = \infty$.

Sei ein solches s mit $lh(s) \geq 2$ gegeben. Sei $\varphi(v, w)$ eine Formel der Sprache $\mathcal{L}_A$. Wir betrachten das folgende Spiel $\mathcal{G}(s, \varphi)$ zwischen den Spielern I und II. I spielt zunächst ein $k < lh(s) - 1$ und eine natürliche Zahl $m_0 < (s)_k$. II spielt daraufhin eine natürliche Zahl $m_1 < (s)_{k+1}$:

I	k	m_0	
II			m_1

II gewinnt $\mathcal{G}(s, \varphi)$ gdw. $\mathcal{N} \models \varphi(m_0, m_1)$. Wir sagen, dass II eine Gewinnstrategie für $\mathcal{G}(s, \varphi)$ besitzt, oder auch: dass s φ erfüllt, gdw.

$$\forall k < lh(s) - 1 \forall m_0 < (s)_k \exists m_1 < (s)_{k+1} \varphi(m_0, m_1).$$

Sei nun s gegeben mit $lh(s) \geq 2n (n > 0)$. Sei $\varphi(v_0, \ldots, v_{2n-1})$ eine Formel der Sprache $\mathcal{L}_A$. Dann bezeichnet $\mathcal{G}(s, \varphi)$ das folgende Spiel zwischen den Spielern I und II.

I	k_0	m_0		k_2	m_2		$\ldots$		k_{2n-2}	m_{2n-2}	
II			m_1			m_3		$\ldots$			m_{2n-1}

Regeln: $k_0 < lh(s) - 1, m_0 < (s)_{k_0}, m_1 < (s)_{k_0+1}$, und $k_{2i} < k_{2i+2} < lh(s) - 1$, $m_{2i+2} < (s)_{k_{2i+3}}, m_{2i+3} < (s)_{k_{2i+3}}$ für $i < n - 1$. II gewinnt $\mathcal{G}(s, \varphi)$ gdw. $\mathcal{N} \models \varphi(m_0, \ldots, m_{2n-1})$. Man definiert "$II$ hat eine Gewinnstrategie für $\mathcal{G}(s, \varphi)$" (= "s erfüllt φ") völlig analog zum obigen Spezialfall.

Wenn s *un*endlich und $\varphi(v_0, \ldots, v_{2n-1})$ beliebig ist, dann gilt: Wenn s die Formel $\varphi(v_0, \ldots, v_{2n-1})$ erfüllt, dann gilt

$$\mathcal{N} \models \forall m_0 \exists m_1 \ldots \forall m_{2n-2} \exists m_{2n-1} \varphi(m_0, \ldots, m_{2n-1}),$$

Da $\lim_{k \to \infty} (s)_k = \infty$.

Eine endliche Folge s heit gut, gdw. $(s)_0 > lh(s)$ und für $0 < i < lh(s)$ ist $(s)_i > ((s)_{i-1})^2$. Eine Formel $\varphi(v_0, \ldots, v_{2n-1})$ heit m–erfüllbar (für $m \geq 2n$) gdw. es eine gute Folge s der Länge m gibt, so dass s φ erfüllt.

Wenn

$$\mathcal{N} \models \forall m_0 \exists m_1 \ldots \forall m_{2n-2} \exists m_{2n-1} \varphi(m_0, \ldots, m_{2n-1}),$$

dann ist φ m–erfüllbar für alle $m \geq 2n$.

Die Peano–Arithmetik (kurz: PA) ist durch die folgenden Axiome gegeben: Kommutativität, Assoziativität und Distributivität für $+$ und $\cdot, x+0 = x,\ x \cdot 0 = 0,\ x \cdot 1 = x$, Transitivität, Assymmetrie und Vergleichbarkeit für $<, x < y \to x+z < y+z,\ 0 < z \wedge x < y \to x \cdot z < y \cdot z, x < y \to \exists z x+z = y$, $0 < 1, 0 < x \to x = 1 \vee 1 < x, x = 0 \vee 0 < x$ sowie dem Induktionsschema

$$(\varphi(0) \wedge \forall x(\varphi(x) \to \varphi(x+1))) \to \forall x \varphi(x)$$

für alle Formeln φ.

Im Folgenden sei $(\varphi_m | m \in \mathbb{N})$ eine beliebige feste rekursive Aufzählung der Axiome von PA. Für $m \in \mathbb{N}$ bezeichnen wir mit Φ_m die Formel $\varphi_0 \wedge \ldots \wedge \varphi_m$. Wir setzen im Folgenden voraus, dass $(\varphi_m | m \in \mathbb{N})$ so gewählt wurde, dass Φ_m logisch äquivalent zu einer Formel der folgenden Gestalt ist:

$$\forall v_0 \exists v_1 \ldots \forall v_{2m-1} \Psi(v_0, v_1, \ldots, v_{2m-2}, v_{2m-1}).$$

wobei Ψ keine Quantoren enthält.

Sei $m \in \mathbb{N}$. Es gibt dann eine unendliche (echt aufsteigende) Folge s, die Ψ_m erfüllt. Es gilt auch: Ψ_m ist $2m$–erfüllbar.

Wir bezeichnen mit σ den Satz: für alle m ist Ψ_m $2m$–erfüllbar. σ ist von der Gestalt $\forall v_0 \ldots \forall v_k \exists v_{k+1} \ldots \exists v_1 \psi(v_0, \ldots, v_l)$, wobei $\psi(v_0, \ldots, v_m)$ keine *unbeschränkten* Quantoren besitzt. (D.h. σ ist Π_2.) Es gilt $\mathcal{N} \models \sigma$.

Wir wollen nun ein Modell von PA konstruieren, in dem σ falsch ist.

Sei Φ ein Satz der Gestalt

$$\forall v_0 \exists v_1 \ldots \forall v_{2m-1} \Psi(v_0, v_1, \ldots, v_{2m-2}, v_{2m-1}),$$

wobei Ψ keine Quantoren enthält. Dann gilt

$$\mathrm{PA} \vdash \Phi \to \Psi \text{ ist } 2m\text{–erfüllbar.}$$

Es gilt also in PA auch, dass Ψ_m $2m$–erfüllbar ist. Die Aussage

$$\exists m \Psi_m \text{ ist nicht } 2m\text{–erfüllbar}$$

(d.h. $\neg\sigma$!) ist *nicht* in PA beweisbar (falls alle in PA beweisbaren Sätze wahr sind). Wir wollen zeigen, dass σ nicht beweisbar ist.

Sei $\mathfrak{M}$ ein Nichtstandardmodell von PA. Da für fede Standard–Zahl $n \in |\mathfrak{M}|$ gilt, dass $\mathfrak{M} \models \Psi_n$ ist $2n$–erfüllbar, gibt es eine Nichtstandard–Zahl $N \in |\mathfrak{M}|$ mit

$$\mathfrak{M} \models \Psi_N \text{ ist } 2N\text{–erfüllbar.}$$

Es gibt also ein $S \in |\mathfrak{M}|$, so dass $\mathfrak{M} \models S$ erfüllt Ψ_N. Seien o.B.d.A. $0, 1, 2 \ldots \in |\mathfrak{M}|$ die Standard–Zahlen in $\mathfrak{M}$. Bezeichne $\vec{S}$ die unendliche Folge $(S)_0, (S)_1, \ldots$ Jedes echte Anfangsstück dieser Folge existiert in $\mathfrak{M}$, aber $\vec{S}$ existiert nicht in $\mathfrak{M}$ (da die Menge der Standard–Zahlen nicht über $\mathfrak{M}$ definierbar ist).

Sei $n \in |\mathfrak{M}|$ eine Standard–Zahl. Dann gilt

$$(\mathfrak{M}, \vec{S}) \text{ erfüllt } \Psi_n.$$

Sei nun H dasjenige Submodell von $\mathfrak{M}$, das genau alle Zahlen aus $|\mathfrak{M}|$ enthält, die (in $\mathfrak{M}$) kleiner als ein Element der Folge $\vec{S}$ sind. Da $\vec{S}$ gut ist, ist H abgeschlossen unter $+$ und $\cdot$.

Es gilt $H \models$ PA. [Sei $n \in |\mathfrak{M}|$ eine Standard–Zahl. $(\mathfrak{M}, \vec{S}) \models \vec{S}$ erfüllt Ψ_n. $\vec{S}$ ist "kofinal in" H. Damit gilt dann Ψ_n in H.]

Wir nehmen nun o.B.d.A. an, dass S (in $\mathfrak{M}$) minimal gewählt war, so dass S die Länge $2N$ hat und $\mathfrak{M} \models S$ erfüllt Ψ_N gilt. Dann gilt $S \notin H$, und $H \models P_N$ ist nicht $2N$–erfüllbar. [Angenommen, $H \models \overline{S}$ erfüllt P_N. Dann gilt $\mathfrak{M} \models \overline{S}$ erfüllt P_N. Aber $\overline{S}$ ist kleiner als S in $\mathfrak{M}$.]

Damit gilt $\neg\sigma$ in H!

Wir haben damit die folgende Variante des Ersten Gödelschen Unvollständigkeitssatzes gezeigt:

Satz 7.9 *Der oben angegebene* Π_2*–Satz* σ *der Sprache der Zahlentheorie, ist in* PA *weder beweisbar noch widerlegbar.*

Kapitel 8

Reduktionen und das Postsche Korrespondenzproblem

Eine Reduktion ist eine effiziente Art und Weise, ein gegebenes Problem auf ein anderes zurückzuführen, so dass eine Lösung des zweiten Problems eine Lösung des ursprünglichen Problems liefert (falls das zweite Problem lösbar ist).

Wir wollen nun diese Methode an einem Beispiel betrachten, dem POSTschen Korrespondenzproblem.

Wir beginnen mit einer Menge von Dominosteinen. Ein typischer Dominostein hat die Gestalt

$$\begin{array}{|c|}\hline a \\ \hline b\ a \\ \hline \end{array}$$

und eine typische Menge von Dominosteinen ist etwa

$$\left\{ \begin{array}{|c|}\hline b \\ \hline c\ a \\ \hline \end{array}, \begin{array}{|c|}\hline a \\ \hline a\ b \\ \hline \end{array}, \begin{array}{|c|}\hline c\ a \\ \hline a \\ \hline \end{array}, \begin{array}{|c|}\hline a\ b\ c \\ \hline c \\ \hline \end{array} \right\}$$

Die Aufgabe ist es, aus einer solchen Menge von Dominosteinen ein *Match* zu gestalten. Dabei wird jeder der Menge entnommene Stein sofort durch einen gleichen Stein ersetzt, so dass also im Match Wiederholungen erlaubt sind. Ein Match ist eine Folge von Dominosteinen, bei der das in der oberen Zeile entstehende Wort mit dem in der unteren Zeile entstehenden Wort übereinstimmt.

Beispielsweise lässt sich aus der obigen Menge wie folgt ein Match bilden:

$$\left\{\begin{array}{|c|}\hline a\\\hline a\ b\\\hline\end{array}\right\},\ \left\{\begin{array}{|c|}\hline b\\\hline c\ a\\\hline\end{array}\right\},\ \left\{\begin{array}{|c|}\hline c\ a\\\hline a\\\hline\end{array}\right\},\ \left\{\begin{array}{|c|}\hline a\\\hline a\ b\\\hline\end{array}\right\},\ \left\{\begin{array}{|c|}\hline a\ b\ c\\\hline c\\\hline\end{array}\right\}$$

Aus anderen Mengen lässt sich kein Match bilden, z.B. nicht aus

$$\left\{\left\{\begin{array}{|c|}\hline a\\\hline b\ c\\\hline\end{array}\right\},\ \left\{\begin{array}{|c|}\hline a\ b\\\hline c\ b\\\hline\end{array}\right\},\ \left\{\begin{array}{|c|}\hline c\ a\\\hline b\\\hline\end{array}\right\}\right\}$$

da jeder Dominostein aus dieser Menge oben ein a und unten kein a enthält.

Das *Postsche Korrespondenzproblem* lautet:

Gegeben sei eine (endliche!) Menge M von Dominosteinen. Lässt sich aus M ein Match gestalten?

Satz 8.1 *Das Postsche Korrespondenzproblem ist nicht entscheidbar.*

Beweis: Wir liefern eine Reduktion des Halteproblems auf das Korrespondenzproblem. Wäre das Korrespondenzproblem entscheidbar, dann wäre auch das Halteproblem entscheidbar.

Unsere Aufgabe ist es also, eine Anfrage an das Halteproblem mit Hilfe von Dominosteinen umzuformulieren.

Das *modifizierte Korrespondenzproblem* lautet:

Gegeben sei eine (endliche!) Menge M von Dominosteinen, wobei ein Stein aus M als "erster" Stein ausgezeichnet sei. Lässt sich aus M ein Match gestalten, welches mit dem ersten Stein von M beginnt?

Wir zeigen zunächst, dass das Halteproblem entscheidbar wäre, wenn das modifizierte Korrespondenzproblem entscheidbar wäre.

Sei $\top$ eine Turingmaschine, die durch Q, Γ (und Σ), sowie δ gegeben ist. Wir wollen eine Menge $M = M_\top^W$ von Dominosteinen konstruieren, so dass einem Match ein Rechenvorgang von $\top$ bei Eingabe von $w \in \Sigma^*$ einspricht.

Sei $w = \gamma_0 \dots \gamma_{n-1}$, wobei $\gamma_0, \dots, \gamma_{n-1} \in \Sigma$. Dann sei

$$\begin{array}{|l|}\hline \#\\\hline \#q_0\gamma_0 \dots \gamma_{n-1}\#\\\hline\end{array}$$

ein Dominostein von M, der als "erster" Stein ausgezeichnet werden soll.[1] Diesem Stein entspricht in offensichtlicher Weise die nullte Konfiguration des Rechenvorgangs von $\top$ bei Eingabe von w. Die nächsten Dominosteine stehen für Anwendungen der übergangsfunktion δ.

[1] Hierbei sei # ein Symbol, das weder in Q noch in Γ enthalten ist.

Seien $q, q' \in Q$ und $\gamma_0, \gamma_1 \in \Gamma$. Wenn $\delta(q, \gamma_0) = (q', \gamma_1, R)$, dann sei

$$\begin{array}{|c|}\hline q\ \gamma_0 \\ \hline \gamma_1\ q' \\ \hline \end{array}$$

ein Dominostein aus M. Wenn $\delta(q, \gamma_0) = (q', \gamma_1, L)$, dann sei für jedes $\gamma_2 \in \Gamma$

$$\begin{array}{|c|}\hline \gamma_2\ q\ \gamma_0 \\ \hline q'\ \gamma_2\ \gamma_1 \\ \hline \end{array}$$

ein Dominostein aus M.

Weiterhin sei für jedes $\gamma \in \Gamma \cup \{\#\}$

$$\begin{array}{|c|}\hline \gamma \\ \hline \gamma \\ \hline \end{array}$$

ein Dominostein von M, und es sei

$$\begin{array}{|c|}\hline \# \\ \hline \sqcup\ \# \\ \hline \end{array}$$

ein Dominostein von M.

Betrachten wir, bevor wir weitergehen, ein Beispiel. Sei etwa $\top$ die Maschine aus Kapitel 2, die entscheidet, ob die Eingabe eine gerade Anzahl von Nullen enthält. Den Rechenvorgang von $\top$ bei Eingabe von 000 können wir dann wie folgt durch eine Folge von Dominosteinen aus M wiedergeben (vgl. Seite 12):

$$\begin{array}{|c|}\hline \# \\ \hline \#q_0000\# \\ \hline \end{array}\ \begin{array}{|c|}\hline q_0\ 0 \\ \hline \sqcup\ q_1 \\ \hline \end{array}\ \begin{array}{|c|}\hline 0 \\ \hline 0 \\ \hline \end{array}\ \begin{array}{|c|}\hline 0 \\ \hline 0 \\ \hline \end{array}$$

$$\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\ \begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\ \begin{array}{|c|}\hline q_1\ 0 \\ \hline \sqcup\ q_0 \\ \hline \end{array}\ \begin{array}{|c|}\hline 0 \\ \hline 0 \\ \hline \end{array}$$

$$\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\ \begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\ \begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\ \begin{array}{|c|}\hline q_0\ 0 \\ \hline \sqcup\ q_1 \\ \hline \end{array}$$

$$\begin{array}{|c|}\hline \# \\ \hline \sqcup\ \# \\ \hline \end{array}\ \begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\ \begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\ \begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\ \begin{array}{|c|}\hline q_1\ \sqcup \\ \hline \sqcup\ q_- \\ \hline \end{array}$$

Dies ist natürlich kein Match. Die obere, bzw. untere Zeile lautet

$$\#q_0000\#\sqcup q_100\#\sqcup\sqcup q_00\#\sqcup\sqcup\sqcup q_1\sqcup$$
$$\#q_0000\#\sqcup q_100\#\sqcup\sqcup q_00\#\sqcup\sqcup\sqcup q_1\sqcup\#\sqcup\sqcup\sqcup\sqcup q_-$$

Mit Hilfe von weiteren Dominosteinen, die das Halten von $\top$ wiederspiegeln, können wir aber die obige Folge zu einem Match fortsetzen.

Für jedes $\gamma \in \Gamma$ seien

$$\left[\begin{array}{|c|}\hline \gamma\ q_+ \\ \hline q_+ \\ \hline \end{array}\right], \left[\begin{array}{|c|}\hline q_+\ \gamma \\ \hline q_+ \\ \hline \end{array}\right], \left[\begin{array}{|c|}\hline \gamma\ q_- \\ \hline q_- \\ \hline \end{array}\right] \text{ und } \left[\begin{array}{|c|}\hline q_-\ \gamma \\ \hline q_- \\ \hline \end{array}\right]$$

Dominosteine von M. Schließlich seien auch

$$\left[\begin{array}{|c|}\hline q_+\ \#\ \# \\ \hline \# \\ \hline \end{array}\right] \text{ und } \left[\begin{array}{|c|}\hline q_-\ \#\ \# \\ \hline \# \\ \hline \end{array}\right]$$

Dominosteine von M.

In unserem obigen Beispiel entsteht dann ein Match, wenn wir die Folge so fortsetzen:

$$\left[\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup\ q_- \\ \hline q_- \\ \hline \end{array}\right]$$

$$\left[\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup q_- \\ \hline q_- \\ \hline \end{array}\right]$$

$$\left[\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup \\ \hline \sqcup \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup\ q_- \\ \hline q_- \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline \sqcup\ q_- \\ \hline q_- \\ \hline \end{array}\right]$$

$$\left[\begin{array}{|c|}\hline \# \\ \hline \# \\ \hline \end{array}\right]\left[\begin{array}{|c|}\hline q_-\ \#\ \# \\ \hline \# \\ \hline \end{array}\right]$$

Die obere, bzw. untere Zeile dieser Fortsetzung lautet:

$$\begin{array}{r} \#\sqcup\sqcup\sqcup\sqcup q_-\#\sqcup\sqcup\sqcup q_-\#\sqcup\sqcup q_-\#\sqcup q_-\#q_-\#\# \\ \#\sqcup\sqcup\sqcup q_-\#\sqcup\sqcup q_-\#\sqcup q_-\#q_-\#\# \end{array}$$

Ein Dominostein möge nun zur Menge $M = M_\top^w$ gehören, gdw. er dies gemäßeiner der obigen Bestimmungen sein muss.

Nennen wir ad hoc eine Turingmaschine $\top$ *vernünftig*, falls bei keiner Eingabe von w Folgendes passiert: während der Berechnung steht der Kopf irgendwann am linken Ende des Bandes und δ sagt, dass der Kopf im nächsten Rechenschritt nach links gehen soll (so dass er also am linken Bandende stehen bleibt).

Falls $\top$ nicht vernünftig ist, dann haben wir Schwierigkeiten, mit Hilfe der obigen Dominosteine Berechnungen von $\top$ zu simulieren. Folgendes ist aber richtig: wenn $\top$ vernünftig ist, dann lässt sich aus dem oben definierten

$M_\top^w$ ein Match gestalten, welches mit dem "ersten" Stein von $M_\top^w$ beginnt, genau dann, wenn $\top(w)\downarrow$.

Mit dem Problem der Existenz nicht–vernünftiger Turingmaschinen gehen wir wie folgt um.

Zu jeder Turingmaschine $\top$ gibt es eine vernünftige Turingmaschine $\top^v$, so dass $\top(w)\downarrow$ gdw. $\top^v(w)\downarrow$ für alle Eingaben w gilt. Damit haben sir das Halteproblem auf das modifizierte Korrespondenzproblem reduziert: es gilt $\top(w)\downarrow$ gdw. sich aus $M_{\top^v}^w$ in Match gestalten lässt, welches mit dem "ersten" Stein beginnt.

Wir wollen nun $M = M_\top^w$ so zu einer Menge $\widetilde{M} = \widetilde{M}_\top^{\,w}$ umgestalten, dass jedes Match aus $\widetilde{M}$ einem Match aus M entspricht, welches mit dem ersten Stein beginnt.

Für eine Folge $w = \gamma_0 \dots \gamma_{k-1}$ schreiben wir

$*w$ für $*\gamma_0 \dots * \gamma_{k-1}$,

w für $\gamma_0 * \dots \gamma_{k-1}*$ und

$*w*$ für $*\gamma_0 * \dots * \gamma_{k-1}*$.

Bestehe nun $M = M_\top^w$ aus den Dominosteinen

$$\begin{array}{|c|}\hline w_i \\ \hline w_i' \\ \hline \end{array}$$

für $i < N$, wobei

$$\begin{array}{|c|}\hline w_0 \\ \hline w_0' \\ \hline \end{array}$$

der erste Dominostein von M sei. Dann enthalte $\widetilde{M} = \widetilde{M}_\top^{\,w}$ die folgenden $N + 2$ Dominosteine:

$$\begin{array}{|l|}\hline * \, w_0 \\ \hline * \, w_0' \, * \\ \hline \end{array}, \quad \begin{array}{|l|}\hline * \diamond \\ \hline \diamond \\ \hline \end{array},$$

sowie

$$\begin{array}{|l|}\hline * \, w_i \\ \hline w_i' \, * \\ \hline \end{array}$$

für alle $i < N$.[2] Es ist unschwer zu erkennen, dass jedes Match von Steinen aus $\widetilde{M}$ mit

$$\begin{array}{|l|}\hline * \, w_0 \\ \hline * \, w_0' \, * \\ \hline \end{array}$$

[2]Hierbei seien $*$ und $\diamond$ neue Symbole.

beginnen und mit

enden muss. Darüber hinaus entspricht jedes Match aus $\widetilde{M}$ genau einem Match aus M, das mit dem ersten Stein beginnt, und umgekehrt. Es gilt also: $\top(w)\downarrow$ gdw. sich aus $\widetilde{M}^{w}_{\top v}$ ein Match gestalten lässt. □

Wir wollen schließlich den Begriff der Reduktion formal definieren.

Definition 8.2 *Eine Funktion $f : \Sigma^* \to \Sigma^*$ heißt* (Turing–) berechenbar *gdw. es eine Turing–Maschine $\top$ gibt, so dass für alle $w \in \Sigma^*$ $\top$ bei Eingabe von w hält, wobei am Ende $f(w)$ auf dem Rechenband geschrieben ist.*

Definition 8.3 *Seien $L_0, L_1 \subset \Sigma^*$ Sprachen. L_0 heißt* (Turing–) reduzierbar auf L_1*, kurz $L_0 \leq_\top L_1$, gdw. es eine berechenbare Funktion $f : \Sigma^* \to \Sigma^*$ gibt, so dass für alle $w \in \Sigma^*$ gilt:*

$$w \in L_0 \text{ gdw. } f(w) \in L_1.$$

Der obige Beweis zeigt, dass das Halteproblem auf das Postsche Korrespondenzproblem reduzierbar ist, im eben definierten Sinne. Allgemein gilt:

Lemma 8.4 *Seien $L_0, L_1 \subset \Sigma^*$ Sprachen mit $L_0, \leq_\top L_1$. Wenn L_1 rekursiv aufzählbar ist, dann ist auch L_0 rekursiv aufzählbar. Wenn L_1 entscheidbar ist, dann ist auch L_0 entscheidbar.*

Beweis: Sei $w \in L_0$ gdw. $f(w) \in L_1$ für alle $w \in \Sigma^*$, wobei f berechenbar ist. Wenn L_1 rekursiv aufzählbar ist, dann existiert eine Turing–Maschine, die bei eingegebenem $w \in \Sigma^*$ zunächst $f(w)$ berechnet und sodann $f(w)$ akzeptiert, gdw. $f(w) \in L_1$; diese Turing–Maschine bezeugt dann, dass L_0 rekursiv aufzählbar ist. Der zweite Teil ergibt sich dann mit Lemma 2.1. □

Kapitel 9

Kontextfreie Sprachen

Eine kontextfreie Grammatik sieht beispielsweise so aus:

$$\begin{array}{lcl} x & \mapsto & 0x1 \\ x & \mapsto & y \\ y & \mapsto & \# \end{array}$$

Diese Regeln sind so zu verstehen, dass mit ihrer Hilfe Symbolfolgen generiert werden. Ein einzelnes Symbol (x bzw. y) darf durch die angegebene Symbolfolge (also $0x1$ oder y bzw. $\#$) ersetzt werden. Dabei starten wir mit einem einzelnen Symbol. Wenn wir mit x starten, so können wir etwa 000#111 folgendermaßen generieren:

$$\begin{array}{l} x \Rightarrow 0x1 \Rightarrow 00x11 \Rightarrow \\ 000x111 \Rightarrow 000y111 \Rightarrow \\ 000\#111 \end{array}$$

Formal besteht eine *kontextfreie Grammatik* aus einer endlichen Menge V von Variablen, einer endlichen Menge Σ von Terminalen ($\Sigma \cap V = \emptyset$), einer endlichen Menge R von Regeln, die einzelnen Variablen Worte aus $(\Sigma \cup V)^*$ zuweisen, und einer Startvariablen $s \in V$. Im obigen Beispiel ist etwa $V = \{x, y\}$, $\Sigma = \{0, 1, \#\}$, R wird durch die obigen drei Zeilen angegeben, und $s = x$. Durch wiederholte Anwendung der Regeln können wir ausgehend von s verschiedene Symbolfolgen generieren.

Wir wollen nun, für ein Symbol γ, γ^n für die Symbolfolge

$$\underbrace{\gamma\gamma\ldots\gamma}_{n \text{ viele}}$$

schreiben. Im obigen Beispiel können wir dann die Folgen $0^n x 1^n$, $0^n y 1^n$ und $0^n \# 1^n$ für $n \in \mathbb{N}$ generieren. Ausgezeichnet seien dabei die Symbolfolgen aus Σ^*.

Eine Sprache $L \subset \Sigma^*$ heißt *kontextfrei* gdw. es eine kontextfreie Grammatik gibt, die aus V, Σ, R, s besteht, so dass die Symbolfolgen aus Σ^*, die diese Grammatik generiert, genau diejenigen sind, die zu L gehören. Demnach ist also die Sprache $\{0^n \# 1^n : n \in \mathbb{N}\}$ kontextfrei.

Die leere Folge $\emptyset$ ist auch eine Folge.

Betrachten wir $V = \{x\}, \Sigma = \{0\}, s = x$ und Regeln R, die wie folgt gegeben sind:

$$x \mapsto 0x1$$
$$x \mapsto \emptyset$$

Die Folgen aus Σ^*, die sich mit Hilfe dieser Grammatik erzeugen lassen, sind genau diejenigen der Form $0^n 1^n$ für $n \in \mathbb{N}$. Die Sprache $\{0^n 1^n : n \in \mathbb{N}\}$ ist also kontextfrei; dies sieht man leicht, indem man die obigen Regeln sanft variiert.

Wie sieht es mit der Sprache $\{0^{2^n} : n \in \mathbb{N}\}$ aus? Wir wollen zeigen, dass diese *nicht* kontextfrei ist.

Lemma 9.1 (Pumping–Lemma). *Sei $L \subset \Sigma^*$ eine kontextfreie Sprache. Dann existiert ein $p \in \mathbb{N}$, so dass für alle $w \in L$, deren Länge mindestens p ist, eine Zerlegung von w in fünf Teile $w \equiv abcde$ existiert, so dass gilt:*

(a) für alle $n \in \mathbb{N}$ ist $ab^n cd^n e$ in L,[1]

(b) bd hat positive Länge, und

(c) $b \subset d$ hat höchstens die Länge p.

Die Zahl p heißt die "Pump–Länge".

Aus diesem Lemma ergibt sich sofort, dass $\{0^{2^n} : n \in \mathbb{N}\}$ nicht kontextfrei ist. Andernfalls sei p die Pump–Länge. Sei $2^n \geq p$, und sei $w \equiv abcde$ eine Zerlegung des Wortes $w \equiv 0^{2^n}$ wie im Pumping–Lemma. Wenn $m > 0$ die Länge von bd ist, dann wären alle Zahlen der Gestalt $2^n - m + km = 2^n + (k-1)m$ Zweierpotenzen. Das ist aber Unsinn.

Beweis des Pumping–Lemmas: Sei die durch V, Σ, R, s gegebene kontextfreie Grammatik Zeuge dafür, dass $L \subset \Sigma^*$ eine kontextfreie Sprache ist. Die Worte $w \in L$ sind also genau diejenigen $w \in \Sigma^*$, die sich aus s mit Hilfe

[1] Wir schreiben hier, für Folgen f, f^n für die n–fache Wiederholung der Folge f.

von R generieren lassen, d.h. so dass eine Generierung $s = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1} = w$ existiert, wobei in jedem Schritt für jedes $\gamma \in V$, das in w_i vorkommt, die eine der entsprechenden Regeln aus R auf γ angewandt wird. Wenn $\gamma \in V$ und γ in w_i vorkommt (für ein $i < N-1$), dann können wir zusehen, was im weiteren Verlauf der Generierung (von w aus s) aus γ wird. Wir werden im nächsten Schritt eine Regel aus R auf γ anwenden (da γ kein Terminal ist), wodurch ein Wort entsteht, so dass auf die Variablen in diesem Wort wiederum später Regeln aus R angewandt werden, etc. Wenn also γ in w_i vorkommt, so gehört zu γ für jedes $j \geq i$ mit $j < N$ dasjenige Teilwort $w_{i,j}^{\gamma}$ von w_j, das aus γ im weiteren Verlauf der Generierung aus γ hervorgeht:

$$\begin{array}{c} s \\ \vdots \\ w_i \equiv a\gamma b \\ \vdots \\ w_j \equiv a' w_{i,j}^{\gamma} b' \\ \vdots \\ w_{N-1} \equiv a'' w_{i,N-1}^{\gamma} b''. \end{array}$$

Wir dürfen annehmen, dass es beliebig lange Worte w aus L gibt, da sonst die zu beweisende Aussage trivial richtig ist.

Sei $c \geq 2$ so dass jede Anwendung einer Regel ein Symbol aus Σ durch eine Folge aus $(\Sigma \cup V)^*$ der Länge höchstens c ersetzt. Sei dann

$$s = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1} = w$$

so, dass die Länge von w mindestens $p = c^{l+2}$ ist, wobei l die Zahl der Elemente von V ist. Dann gilt sicherlich $N \geq l + 2$.

Wir wählen nun für jedes $w \in L$ der Länge $\geq p$ eine Art und Weise

$$s = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1} = w$$

der Generierung, so dass möglichst wenige Regeln aus R angewandt werden müssen. Für jede solche Generierung definieren wir $(i, \gamma) < (j, \gamma')$ durch: $i < j < N-1$, γ und γ' sind beide aus V, und γ' kommt in $w_{i,j}^{\gamma}$ vor. Da w_{N-2} Variablen enthält, existieren $\gamma_0, \gamma_1, \ldots, \gamma_{N-2}$ mit $(0, \gamma_0) < (1, \gamma_1) < \ldots < (N-2, \gamma_{N-2})$. Da $N - 1 \geq l + 1$, existiert ein $\gamma \in V$ mit $\gamma_i = \gamma = \gamma_j$ für $i < j < N - 1$.

Da auf die Generierung möglichst wenige Regeln angewandt werden, kann $w_{i,j}^{\gamma}$ nicht nur aus dem Symbol γ bestehen, hat also Länge ≥ 2. Wir erhalten damit folgendes Bild:

$$\begin{array}{c} s \\ \vdots \\ w_i \equiv a\gamma b \\ \vdots \\ w_j \equiv a'c\gamma db' \\ \vdots \\ w_{N-1} \equiv a''c'w_{j,N-1}^{\gamma}d'b'', \end{array}$$

wobei $c\gamma d \equiv w_{i,j}^{\gamma}$ und *cd nicht* die leere Folge $\emptyset$ ist.

Durch Ineinanderschachteln sieht man dann aber, dass auch

$$\begin{array}{l} a'cw_{i,j}^{\gamma}db' \equiv a'c^2\gamma d^2b', \\ a'c^2w_{i,j}^{\gamma}d^2b' \equiv a'c^3\gamma d^3b', \text{ etc.}, \end{array}$$

also auch

$$\begin{array}{l} a''c'^2w_{j,N-1}^{\gamma}d'^2b'', \\ a''c^3w_{j,N-1}^{\gamma}d'^2b'', \text{ etc.} \end{array}$$

generiert werden können. Dies zeigt die Aussagen (a) und (b) des Pumping–Lemmas. Man überzeugt sich dann leicht, dass (c) auch arrangiert werden kann. □

Eine kontextfreie Grammatik ist in CHOMSKY*scher Normalform* gdw. jede Regel eine der drei folgenden Formen besitzt:

$$\begin{array}{lcl} s & \mapsto & \emptyset \\ v & \mapsto & uu' \\ v & \mapsto & t \end{array}$$

Hierbei ist s die Startvariable, v, u, u' sind Variablen und t ist ein Terminal.

Lemma 9.2 *Sei $L \subset \Sigma^*$ eine kontextfreie Sprache. Dann existiert eine kontextfreie Grammatik in Chomskyscher Normalform, so dass die Symbolfolgen aus Σ^*, die diese Grammatik generiert, genau diejenigen sind, die zu L gehören.*

Beweis: Werde L durch die kontextfreie Grammatik V, Σ, R, s generiert. Wir fügen zu V neue Variablen hinzu und zerbrechen einzelne Regelanwendungen in Anwendungen mehrerer Regeln.

Zunächst führen wir eine neue Variable s_0 als neues Startsymbol ein und wir nehmen

$$s_0 \mapsto s$$

als neue Regel hinzu.

Betrachten wir nun Regeln aus R der Form

$$v \mapsto \emptyset,$$

wobei $v \not\equiv s$. Für jede solche Regel betrachten wir alle Regeln

$$v' \mapsto w$$

aus R, wobei v' eine Variable und w ein Wort aus $(\Sigma \cup V)^*$ ist, in dem v vorkommt. Wir fügen dann für jede Menge von Vorkommnissen der Variablen v in w die Regel

$$v' \mapsto \overline{w}$$

hinzu, wobei $\overline{w}$ aus w hervorgeht, indem alle Vorkommnisse von v in w aus dieser Menge gelöscht werden. (Wenn also v in w n–fach vorkommt, dann fügen wir 2^n Regeln hinzu.) Anschließend streichen wir die Regel

$$v \mapsto \emptyset.$$

Wir wiederholen all dies solange, bis wir keine Regel der Form mit $v \not\equiv s$ mehr vorliegen haben.

Auf ähnliche Art und Weise können wir mit jeder Regel der Form

$$v \mapsto v',$$

wobei v und v' Variablen sind, umgehen.

Betrachten wir schließlich eine Regel der Form

$$v \mapsto w \equiv v_0 v_1 \dots v_{i-1},$$

wobei $v, v_0, v_1, \dots, v_{i-1}$ Variablen sind. Wir ersetzen diese eine Regel durch die folgende Liste von $i-1$–Regeln:

$$\begin{aligned} v &\mapsto v_0 x_0 \\ x_0 &\mapsto v_1 x_1 \\ &\vdots \\ x_{i-4} &\mapsto v_{i-3} x_{i-3} \\ x_{i-3} &\mapsto v_{i-2} v_{i-1} \end{aligned}$$

Hierbei sind $x_0, x_1, \ldots, x_{i-3}$ eigens für diesen Zweck neu eingeführte Variablen.

Es ist leicht zu sehen, dass die neue Grammatik dieselbe Sprache generiert wie die alte. Offensichtlich ist die neue Sprache in Chomskyscher Normalform. □

Kapitel 10

Die Komplexitätsklassen P und NP

Entscheidbare Probleme müssen nicht "einfach" sein. Wir werden im nächsten Kapitel sehen, in welchem Sinne das in Kapitel 2 betrachtete SAT–Probleme sehr kompliziert ist. In diesem Kapitel wollen wir *einfache* Probleme studieren.

Probleme können schwierig sein, da ihre Lösung viel Zeit oder "Papier" (Speicherplatz, bzw. Platz auf dem Rechenband) in Anspruch nimmt. Wir wollen uns mit Zeitkomplexität beschäftigen.

Betrachten wir die drei Turing–Maschinen, die wir in Kapitel 2 gebaut haben. Die erste Maschine entscheidet die Menge aller Folgen der Form

$$\underbrace{00\ldots0}_{n \text{ viele}},$$

wobei n gerade ist. Bei einer Eingabe der Länge n benötigt die Maschine $n+1$ Rechenschritte, um eine Antwort zu finden.

Die zweite Maschine entscheidet

$$\underbrace{00\ldots0}_{2^n \text{ viele}},$$

wobei $n \in \mathbb{N}$. Bei einer Eingabe der Länge 2^n benötigt die Maschine

$$(2n+1) \cdot 2^n + 1$$

Rechenschritte; entsprechend benötigt sie bei einer Eingabe kürzerer Länge höchstens $(2n+1)2^n+1$ Rechenschritte. Da wir nicht allzu genau sein müssen,

können wir also sagen, dass diese Maschine bei einer Eingabe der Länge n zirka

$$2 \cdot \log_2 n \cdot n$$

(also im Allgemeinen weniger als $2 \cdot n^2$) Rechenschritte benötigt.

Betrachten wir schließlich die Turing–Maschine, die SAT entscheidet. Hier dauert die Entscheidung wesentlich länger. Wenn die Eingabe die Länge n hat, dann kann die Eingabe "größenordnungsmäßig" n verschiedene Aussagenvariablen enthalten, so dass 2^n Belegungen dieser Aussagenvariablen betrachtet werden müssen.

Definition 10.1 *Seien $f : \mathbb{N} \to \mathbb{R}^+$, $g : \mathbb{N} \to \mathbb{R}^+$ Funktionen.*[1] *Wir schreiben $f \leq O(g)$ gdw. ein $c \in \mathbb{R}^+$ existiert mit*

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} \leq c.$$

Wenn $f \leq O(g)$, dann sagt man auch, dass f asymptotisch durch g beschränkt *wird. Beispielsweise wird $n \mapsto 2 \cdot \log_2 n \cdot n$ asymptotisch durch $n \mapsto n^2$ beschränkt.*

Definition 10.2 *Seien $f : \mathbb{N} \to \mathbb{R}^+$, $g : \mathbb{N} \to \mathbb{R}^+$ Funktionen. Wir schreiben $f \leq o(g)$ gdw.*

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Offensichtlich gilt $f \leq O(f)$ für jedes f und $f \leq o(f)$ für kein f. Aus $f \leq o(g)$ folgt $f \leq O(g)$. Für $f(n) = 2 \cdot \log_2 n \cdot n$ und $g(n) = n^2$ gilt $f \leq o(g)$.

Mit Hilfe der O–Notation können wir nun die Laufzeitkomplexitätsklassen definieren.

Definition 10.3 *Sei $\top$ eine Turing–Maschine, die bei allen Eingaben hält. Die* Laufzeit von $\top$ bei Eingabe von *w ist die Zahl der Rechenschritte, die $\top$ bei Eingabe von w durchführt, bis sie hält. Die* Laufzeit von $\top$ *ist die Funktion $f : \mathbb{N} \to \mathbb{N}$, so dass für alle $n \in \mathbb{N}$ gilt: $f(n)$ ist die maximale Laufzeit von $\top$ bei Eingabe eines Wortes w der Länge n.*

[1] Hierbei ist $\mathbb{R}^+$ die Menge der positiven reellen Zahlen.

Definition 10.4 *Sei $f : \mathbb{N} \to \mathbb{N}$ eine Funktion. Die* zu f gehörige Laufzeitkomplexitätsklasse *ist die Menge aller entscheidbaren Sprachen L, für die Folgendes gilt: es gibt eine Turing–Maschine $\top$ mit Laufzeit $g : \mathbb{N} \to \mathbb{N}$, so dass $g \leq O(f)$ und $\top(w)\downarrow + $ gdw. $w \in L$ und $\top(w)\downarrow -$ gdw. $w \notin L$ für alle Worte w des Eingabealphabets.*

So gehört nun beispielsweise die Sprache

$$\{\underbrace{0\ldots0}_{n \text{ viele}} : n \text{ gerade}\}$$

zur zur Identität $n \mapsto n$ gehörigen Laufzeitkomplexitätsklasse. Die Sprache

$$\{\underbrace{0\ldots0}_{2^n \text{ viele}} : n \in \mathbb{N}\}$$

gehört zur zur Funktion $n \mapsto n^2$ gehörigen Laufzeitkomplexitätsklasse.

SAT gehört zur zur Funktion $n \mapsto 2^n$ gehörigen Laufzeitkomplexitätsklasse. Dies scheint zu besagen, dass SAT "schwieriger" ist als die beiden erstgenannten Probleme.

Für eine Funktion $f : \mathbb{N} \to \mathbb{N}$ bezeichnen wir im Folgenden die zu f gehörige Laufzeitkomplexitätsklasse mit $K(f)$.

Betrachten wir die Sprache

$$L = \{\underbrace{0\ldots0}_{n \text{ viele}}\underbrace{1\ldots1}_{n \text{ viele}} : n \in \mathbb{N}\}.$$

Es ist nicht schwer einzusehen, dass L zu $K(n \mapsto n^2)$ gehört. Etwas mehr Arbeit ist erforderlich, um zu zeigen, dass L zu $K(n \mapsto \log_2 n \cdot n)$ gehört. Dies wird durch die folgende Turing–Maschine $\top$ geleistet.

Sei $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, x, \sqcup\}$. Sei $w \in \Sigma^*$ gegeben. $\top$ sieht zunächst nach, ob in w eine 0 rechts von einer 1 vorkommt. In diesem Falle verwirft $\top$ die Eingabe w. Andernfalls geht der Kopf zum linken Bandende zurück und $\top$ durchläuft die folgende Schleife: Es werde nachgesehen, ob eine gerade Anzahl an Zeichen auf dem Band nicht durchgestrichen (d.h. durch ein x ersetzt) sind — andernfalls verwerfe man die Eingabe; sodann streiche man jede zweite 0 und sodann jede zweite 1. Falls am Ende nur noch Nullen oder nur noch Einsen auf dem Band übrig bleiben, verwerfe man die Eingabe. Falls am Ende Nichts übrig bleibt, akzeptiere man die Eingabe.

Wir definieren nun die Komplexitätsklasse P

Definition 10.5 *P ist die Menge aller Sprachen, die in einer zu einer Polynomfunktion gehörigen Laufzeitkomplexitätsklasse liegt.*

Für $f(n) = a_k n^k + a_{k-1} n^{k-1} + \ldots + a_0$ und $g(n) = n^k$ gilt $g \leq 0(f)$. Wir können also schreiben

$$P = \bigcup_{k \in \mathbb{N}} K(n \mapsto n^k).$$

Beispielsweise liegen also die Sprachen

$$\{\underbrace{0 \ldots 0}_{n \text{ viele}} : n \text{ gerade}\},$$
$$\{\underbrace{0 \ldots 0}_{2^n \text{ viele}} : n \in \mathbb{N}\} \text{ und}$$
$$\{\underbrace{0 \ldots 0}_{n \text{ viele}} \underbrace{1 \ldots 1}_{n \text{ viele}} : n \in \mathbb{N}\}$$

in P. Für die erste und die letzte dieser Sprachen ergibt sich dies auch aus der folgenden Aussage:

Lemma 10.6 *Sei $L \subset \Sigma^*$ eine kontextfreie Sprache. Dann ist L in P.*

Beweis: Wir zeigen zunächst, dass jede kontextfreie Sprache entscheidbar ist. Sei $L \subset \Sigma^*$ kontextfrei. Auf Grund von Lemma 9.2 existiert dann eine kontextfreie Grammatik in Chomskyscher Normalform, so dass die Symbolfolgen aus Σ^*, die diese Grammatik generiert, genau diejenigen sind, die zu L gehören.

Sei $w \in \Sigma^*$. Wir können dann wie folgt entscheiden, ob $w \in L$ oder nicht. Angenommen, $w \in L$. Sei dann

$$s \equiv w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1} \equiv w$$

eine Generierung von w, bei der in jedem Schritt genau eine Regel auf eine vorkommende Variable angewandt wird. Sei $w \not\equiv \emptyset$. Wenn dann $n > 0$ die Länge von w ist, dann muss bei obiger Generierung $n - 1$–fach eine Regel der Form

$$v \mapsto uu'$$

(wobei v, u, u' Variablen sind) und n–fach eine Regel der Form

$$v \mapsto t$$

(wobei v Variable und t Terminal ist) angewandt werden. Es gilt also $N-1 = 2n-1$, d.h. $N = 2n$. Dies zeigt: wenn $w \in L, w \not\equiv \emptyset$, die Länge $n > 0$ hat, dann gilt für jede Generierung

$$s \equiv w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1} \equiv w$$

von w, bei der in jedem Schritt genau eine Regel auf eine vorkommende Variable angewandt wird, dass $N = 2n$.

Sei nun $w \in \Sigma^*$ beliebig. Sei $w \not\equiv \emptyset$, und sei $n > 0$ die Länge von w. Wenn dann k die Anzahl der Regeln der Grammatik ist, dann gibt es höchstens $(k \cdot \frac{N}{2})^{N-1}$ Generierungen

$$s \equiv w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1},$$

bei denen in jedem Schritt genau eine Regel auf eine vorkommende Variable angewandt wird. Wir können alle diese Generierungen für $N = 2n$ durchgehen und sehen, ob mittels einer von diesen das vorgelegte w generieren wird.

Dies zeigt, dass L entscheidbar ist, aber es zeigt offensichtlich noch nicht, dass L in P ist. Hierzu brauchen wir "dynamisches Programmieren". Wir sagen, dass $w \in \Sigma^*$ aus der Variablen $v \in V$ (wobei v nicht notwendigerweise die Startvariable sein muss) generiert werden kann gdw. eine Generierung

$$v \equiv w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_{N-1} \equiv w$$

mit Hilfe der vorliegenden kontextfreien Grammatik existiert.

Wir können die Frage, ob ein gegebenes Wort $w \in \Sigma^*$ der Länge $n \geq 2$ aus $v \in V$ generiert werden kann, wie folgt auf die Frage, ob gewisse Worte der Länge $< n$ aus Variablen v_0, v_1 generiert werden können, zurückführen. Es gibt $n-1$ Möglichkeiten, w in zwei nichttriviale Teile w_0 und w_1 aufzuspalten, d.h. zu schreiben

$$w \equiv w_0 w_1,$$

wobei $w_0, w_1 \in \Sigma^*$ und w_0 und w_1 beide positive Längen haben. Für jede solche Aufspaltung und für jede Regel

$$v \mapsto v_0 v_1$$

der Grammatik gilt offensichtlich: wenn w_0 aus v_0 und wenn w_1 aus v_1 generiert werden kann, dann kann w aus v generiert werden.

Wir können nun die Frage, ob ein gegebenes $w \in \Sigma^*$ in L ist, schnell folgendermaßen entscheiden. Sei $n \in \mathbb{N}$ die Länge von w,

$$w \equiv t_0 t_1 \dots t_{n-1},$$

wobei $t_0, \dots, t_{n-1}$ Terminale sind. Wir konstruieren dann eine $n \times n$ Matrix

$$M = \begin{pmatrix} a_{00} & \dots & a_{0n-1} \\ \vdots & & \vdots \\ a_{n-1\,0} & \dots & a_{n-1n-1} \end{pmatrix}$$

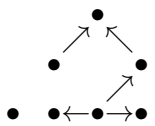
wie folgt: für jedes $i \leq j < n$ sei a_{ij} die Menge aller Variablen $v \in V$, so dass das Teilwort

$$t_i t_{i+1} \dots t_{j-1} t_j$$

von w aus v generiert werden kann. Für $i > j$ sei $a_{ij} = \emptyset$. Mit Hilfe des oben dargestellten Verfahrens können wir sehr leicht durch Rekursion nach $j - i$ die Matrixeintragungen a_{ij} bestimmen. w kann dann (aus der Startvariablen s) generiert werden gdw. $s \in s_{0n-1}$. Man überzeugt sich, dass dieses Verfahren beweist, dass L in P ist. □

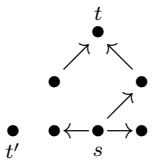
Wir werden nun einige andere Beispiele von Sprachen, die in P liegen, kennen lernen. Wenn eine Sprache L in P liegt, dann ist in gewisser Weise "schnell" entscheidbar, ob ein gegebenes w in L liegt oder nicht. Unsere Turing–Maschine, die das SAT–Problem entscheidet, ist *nicht* schnell in diesem Sinne. Wir werden uns später der Frage zuwenden, ob $P \in SAT$ oder nicht. Die Klasse P entspricht einer gängigen Auffassung nach der Klasse aller Probleme, die mit Hilfe von Computern im wahren Leben tatsächlich gelöst werden können.

Betrachten wir das *Pfadproblem*. Ein *gerichteter Graph* ist eine endliche Menge von Punkten zusammen mit einer Menge von Pfeilen, die von Punkten zu anderen Punkten zeigen. Beispiel:



Formal ist ein gerichteter Graph einfach eine Teilmenge G von $M \times M$, wobei M eine endliche Menge (von Punkten) ist. Dabei bedeutet $(p, q) \in G$, dass ein Pfeil von p nach q zeigt.

Sei nun ein gerichteter Graph $G \subset M \times M$ gegeben, und seien zwei Punkte s und t aus M als *Start–* und *Zielpunkt* ausgezeichnet. Das Pfadproblem lautet: gibt es durch G einen Pfad von s nach t? Beispiel:



Hier gibt es einen Pfad von s nach t, nicht aber von s nach t'.

Formal ist ein Pfad durch $G \subset M \times M$ von s nach t eine Folge $s_0, s_1, \ldots, s_{N-1}$ (für ein $N \in \mathbb{N}$), so dass $s_0 = s, s_{N-1} = t$ und $(s_i, s_{i+1}) \in G$ für alle $i < N-2$.

Die Frage, ob es durch G einen Pfad von s nach t gibt, lässt sich folgendermaßen beantworten: Offensichtlich kann es nicht sinnvoll, bzw. nötig sein, auf einen Pfad von s nach t ein und denselben Punkt mehrmals aufzusuchen. Wenn M nun n Elemente hat, dann gibt es $(n-1)!$ "potentielle" Pfade der Länge $n-1$, die s als Startpunkt besitzen und die keinen Punkt mehrmals aufsuchen, d.h. $(n-1)!$ Folgen der Form

$$s_0 = s, s_1, \ldots, s_{n-1},$$

wobei $s_i \neq s_j$ für $i \neq j$. Wenn $t = s_k$ (für $k < n$), dann können wir nachsehen, ob für alle $i < k$ gilt, dass $(s_i, s_{i-1}) \in G$.

Wenn wir das Pfadproblem mit diesem Ansatz lösen, dann ist die Laufzeit der entsprechenden Turing–Maschine sehr groß, nämlich "größenordnungsmäßig" $n \to (n-1)!$.

Es gibt aber eine schnellere Methode. Seien G, s, t gegeben. Im nullten Durchgang markieren wir den Punkt s. Im $(k-1)^{\text{ten}}$ Durchgang markieren wir alle Punkte, die durch einen Pfeil von bereits markierten Punkten aus erreichbar sind; falls allerdings keine neuen Punkte markiert würden, brechen wir dieses Verfahren ab. Offensichtlich kann es, wenn n die Zahl der Punkte aus M ist, höchstens n derartige Durchgänge geben. Nun gibt es offenbar einen Pfad durch G von s nach t gdw. am Ende der Punkt t markiert ist.

Es ist nicht schwer zu sehen, dass mit Hilfe von dieser Idee gezeigt werden kann, dass das Pfadproblem in $K(n \mapsto n^2)$, also insbesondere in P liegt.

Ein weiteres Problem, das nicht auf den ersten Blick in P liegt, lautet: sind zwei gegebene natürliche Zahlen n und m relativ prim?[2] Ein nahe liegender Lösungsansatz untersucht zu jeder natürlichen Zahl $q \leq \min(n, m)$

[2] n, m sind relativ prim gdw. 1 die größte Zahl ist, die sowohl n als auch m teilt.

(oder $q \leq \sqrt{\min(n,m)}$) ob q sowohl n als auch m teilt. Wenn nun die Zahlen n und m etwa in Dualdarstellung in eine Turing–Maschine eingegeben werden, dann ist die Länge der Eingabe zirka $\log_2(n) + \log_2(m)$; die Rechenzeit aber ist größenordnungsmäßig gleich $\min(n,m)$ (oder $\sqrt{\min(n,m)}$). Bei einer Eingabe der Länge k ist die Rechenzeit dann größenordnungsmäßig gleich 2^k.

Das Problem, ob zwei gegebene natürliche Zahlen relativ prim sind, ist dennoch in P. Man zeigt dies mit Hilfe des Euklidschen Algorithmus.

Wir wollen uns nun Problemen zuwenden, die scheinbar schwierig sind. Wir führen zunächst eine Variante des Begriffs "Turing–Maschine" ein, nämlich den der "nichtdeterministischen Turing–Maschine".

Während eine Turing–Maschine stur gemäßihres Programms (d.h. gemäßihrer übergangsfunktion δ) rechnet, also deterministisch arbeitet, können nichtdeterministische Turing–Maschinen im Rahmen ihrer Möglichkeiten Entscheidungen treffen.

Eine *nichtdeterministische Turing–Maschine* $\top$ besteht formal aus den folgenden Objekten:

(1) Einer endlichen Menge Q von *Zuständen*, wobei es drei ausgezeichnete Zustände gibt: den *Anfangszustand* $q_0 \in Q$, den *positiven Endzustand* q_+ und den *negativen Endzustand* q_-.

(2) Ein endliches *Alphabet* Γ, d.h. eine Menge von Symbolen, die das Leerzeichen $\sqcup$ enthalten soll; eine nichtleere Teilmenge Σ von Γ, die nicht $\sqcup$ enthält, ist als *Eingabealphabet* ausgezeichnet.

(3) Eine *nichtdeterministische übergangsfunktion* $\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, R\}) \backslash \emptyset$, die Paaren (q, γ) mit $q \in Q$ und $\gamma \in \Gamma$ eine nichtleere Menge von Tripeln (q', b, x) mit $q' \in Q, b \in \Gamma$ und $x \in \{L, R\}$ zuordnet.

(1) und (2) sind also genau wie im Falle der (klassischen, deterministischen) Turing–Maschine. In (3) wurde $Q \times \Gamma \times \{L, R\}$ durch die Potenzmenge $\mathcal{P}(Q \times \Gamma \times \{L, R\})$ von $Q \times \Gamma \times \{L, R\}$ (d.h. der Menge aller Teilmengen von $Q \times \Gamma \times \{L, R\}$) als Bildbereich von δ ersetzt. Während jeder konkreten Berechnung wählt die nichtdeterministische Turing–Maschine ein Element aus $\delta(q, \gamma)$ (für das aktuelle Paar (q, γ)) aus, um nach ihm zu verfahren.

Ein *Rechenvorgang* der nichtdeterministische Turing–Maschine $\top$ ist eine (endliche oder unendliche) Folge von *Konfigurationen* K_n. Dabei sieht jede Konfiguration genau so aus wie im Falle (klassischer) Turing–Maschinen. Sei q der aktuelle Zustand, und sei γ das Symbol, das aktuell gelesen wird.

Dann ergibt sich K_{n+1} aus K_n, indem $\top$ zunächst ein Tripel $(q', b, x) \in \delta(q, \gamma)$ auswählt und sodann verfährt wie im Falle der klassischen Turing–Maschinen.

Offensichtlich ist jede (klassische) Turing–Maschine auch eine nichtdeterministische Turing–Maschine, aber nicht umgekehrt.

Wir wollen die Funktionsweise nichtdeterministischer Turing–Maschinen zunächst an einem Beispiel illustrieren, nämlich einer Variante der Turing–Maschine $\top_{SAT}$, die im 2. Kapitel zur Lösung des SAT–Problems gebaut wurde. Diese Maschine produziert nach der Eingabe einer aussagenlogischen Formel φ auf dem Rechenband zunächst das Wort

$$\varphi\#\varphi\#\#\underbrace{0\ldots0}_{n \text{ viele}},$$

wobei n die Länge von φ ist. Zu einem späteren Zeitpunkt sieht es auf dem Band so aus:

$$\varphi\#\varphi\#\#d,$$

wobei d die Dualdarstellung einer Zahl zwischen 0 und $2^n - 1$ ist. Die Zahl d wird zur Buchführung benutzt und diktiert eine Belegung der in φ vorkommenden Aussagenvariablen durch Nullen und Einsen. $\top_{SAT}$ testet für eine solche Belegung, ob sie φ erfüllt. (Dabei wird mit dem Vorkommnis von φ zwischen # und ## gearbeitet.)

Es ist nicht schwer zu sehen, dass die Laufzeit von $\top_{SAT}$ "größenordnungsmäßig" $n \mapsto 2^n \cdot n^2$ insbesondere also nicht polynomiell ist, so dass $\top_{SAT}$ bezeugt, dass $SAT \in K(2^n \cdot n^2)$.

Wir können nun $\top_{SAT}$ sehr leicht in eine nichtdeterministische Turing–Maschine $\top^n_{SAT}$ umbauen, die eine erheblich geringere Laufzeit besitzt. Die Maschine $\top^n_{SAT}$ soll dabei nach der Eingabe einer aussagenlogischen Formel φ der Länge n wie folgt vorgehen. $\top^n_{SAT}$ "rät" zunächst eine Folge d von Nullen und Einsen der Länge n und schreibt diese, abgetrennt von φ durch #, auf das Rechenband, so dass dort

$$\varphi\#d$$

geschrieben steht. Sodann wird d analog wie im Falle von $\top_{SAT}$ benutzt, um eine Belegung der in φ vorkommenden Aussagenvariablen zu diktieren. Für diese eine Belegung testet $\top^n_{SAT}$ schließlich, ob sie φ erfüllt.

Das "Raten" von d ist der einzige Teil der Berechnung, der nichtdeterministisch stattfindet. Formal geht dies etwa folgendermaßen vor sich. Bei

Eingabe von φ produziert $\top^n_{SAT}$ zuallererst das Wort

$$\varphi\#\varphi,$$

indem # hinter φ geschrieben wird und dahinter wiederum φ kopiert wird. Sodann begibt sich der Kopf zum ersten Symbol des rechten Vorkommnisses von φ und $\top^n_{SAT}$ begibt sich in einen ausgezeichneten "Rate"–Zustand q_R. Die übergangsfunktion δ sei nun so beschaffen, dass

$$\delta(q_R, \gamma) = \{(q_R, 0, R), (q_R, 1, R)\}$$

für alle $\gamma \in \Sigma$ (dem Eingabealphabet). Dies bedeutet, dass der Kopf nun solange nach rechts läuft, bis $\sqcup$ erscheint, und in jedem Schritt das aktuell gelesene Symbol durch 0 oder 1 ersetzt wird.

Definition 10.7 *Sei $\top$ eine nichtdeterministische Turing–Maschine, so dass jeder Rechenvorgang bei jeder Eingabe endlich ist.*[3] *Die Laufzeit von $\top$ ist die Funktion $f : \mathbb{N} \to \mathbb{N}$, so dass für alle $n \in \mathbb{N}$ gilt: $f(n)$ ist die maximale Laufzeit von $\top$ bei Eingabe eines Wortes w, welches höchstens die Länge n besitzt.*

Im obigen Beispiel ist dies nicht der Fall, aber im Allgemeinen kann natürlich die Länge der Berechnung bei Eingabe von w davon abhängen, welche Auswahlen im Verlauf der Berechnung getroffen werden. So existiert im Falle von nichtdeterministischen Turing–Maschinen $\top$ nicht notwendig "die" Laufzeit von $\top$ bei Eingabe von w.

Die Laufzeit von $\top^n_{SAT}$ ist "größenordnungsmäßig" $n \mapsto n^2$. Das SAT–Problem ist mit Hilfe von nichtdeterministischen Turing–Maschinen also wesentlich schneller lösbar als mit Hilfe von (klassischen, deterministischen) Turing–Maschinen.

Definition 10.8 *Sei $f : \mathbb{N} \to \mathbb{N}$ eine Funktion. Die* zu f gehörige nichtdeterministische Laufzeitkomplexitätsklasse *ist die Menge aller Sprachen L, für die Folgendes gilt: es gibt eine nichtdeterministische Turing–Maschine $\top$ mit Laufzeit $g : \mathbb{N} \to \mathbb{N}$, so dass $g \leq O(f), w \in L$ gdw. es einen Rechenvorgang von $\top$ bei Eingabe von w gibt, der w akzeptiert, und $w \notin L$ gdw. alle Rechenvorgänge von $\top$ bei Eingabe von w das Wort w verwerfen.*[4]

[3] Dies bedeutet, dass $\top$ bei jeder Eingabe hält, unabhängig davon, welche Auswahlen im Verlauf der Berechnung getroffen werden.

[4] Ein Rechenvorgang akzeptiert/verwirft w gdw. dieser Rechenvorgang mit einer Konfiguration endet, bei der die Maschine sich am Schluss im Zustand q_+/q_- befindet.

Beispielsweise bezeugt $\top^n_{SAT}$, dass SAT in der zu $n \mapsto n^2$ gehörigen nichtdeterministischen Komplexitätsklasse liegt.

Für eine Funktion $f : \mathbb{N} \to \mathbb{N}$ schreiben wir im Folgenden $K^n(f)$ für die zu f gehörige nichtdeterministische Komplexitätsklasse.

Definition 10.9 *NP ist die Menge aller Sprachen, die in einer zu einer Polynomfunktion gehörigen nichtdeterministischen Laufzeitkomplexitätsklasse liegen.*

Die obigen überlegungen zeigen, dass SAT in NP liegt. Da jede (klassische) Turing–Maschine auch eine nichtdeterministische Turing–Maschine ist, gilt trivialerweise

$$P \subset NP.$$

Es wird vermutet, dass NP verschieden ist von P. Hierfür gibt es aber keinen Beweis. Wir werden später sehen, dass $P \neq NP$ gdw. $SAT \notin P$.

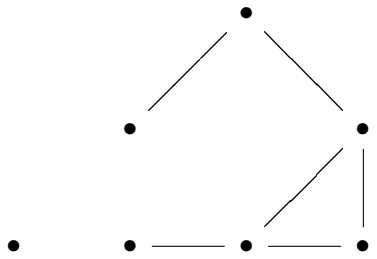
Hier sind zwei weitere Beispiele für Probleme in NP.

Sei G ein gerichteter Graph. Ein *Hamiltonscher Weg* durch G ist ein Weg durch G, der jeden Knoten genau einmal besucht. Genauer können wir dies wie folgt definieren. Sei $G \subset M \times M$, wobei M endlich ist. Dann ist ein Hamiltonscher Weg durch G von $s \in M$ nach $z \in M$ eine Folge $(s_0, s_1), (s_1, s_2), \ldots, (s_{N-2}, s_{N-1})$, wobei $s_0 = s_1, s_{N-1} = z, (s_i, s_{i+1}) \in G$ für alle $i < N - 1$ und $s_i \neq s_j$ für alle $i < j < N$. (In diesem Falle muss offensichtlich N die Zahl der Elemente von M sein.)

Das *Problem Hamiltonscher Wege* lautet wie folgt. Gegeben seien ein gerichteter Graph $G \subset M \times M$ und $s, z \in M$. Existiert dann ein ein Hamiltonscher Weg durch G von s nach z? Man überzeugt sich leicht davon, dass dieses Problem in NP liegt. Wir können eine nichtdeterministische Turing–Maschine zunächst einen Weg durch G raten lassen und sie sodann entscheiden lassen, ob dieser Weg ein Hamiltonscher Weg von s nach z ist.

Unser nächstes Beispiel ist das CLIQUE–Problem. Ein *ungerichteter Graph* ist eine endliche Menge von Punkten, zusammen mit einer Menge von Verbin-

dungsstrecken, die gewisse Punkte mit anderen Punkten verbinden. Beispiel:



Formal ist ein ungerichteter Graph eine Teilmenge G von $[M]^2$, wobei M eine endliche Menge (von Punkten) ist. Hierbei ist $[M]^2$ die Menge aller zweielementigen Teilmengen von M und $\{s,t\} \in G$ bedeutet, dass eine Verbindungsstrecke von s mit t existiert.

Sei $k \in \mathbb{N}$. Eine *k–Clique in G* ist eine k–elementige Teilmenge X von M, so dass $[X]^2 \subset G$, d.h. so dass je zwei Punkte aus X durch eine Verbindungsstrecke verbunden sind.

Im obigen Beispiel gibt es etwa genau eine 3–Clique, aber keine 4–Clique.

Das CLIQUE–Problem lautet wie folgt. Gegeben seien ein ungerichteter Graph G und eine natürliche Zahl k. Gibt es eine k–Clique in G?

Wir hätten NP auch mit Hilfe von "Verifikatoren" anstelle von nichtdeterministischen Turing–Maschinen definieren können. Sei $L \subset \Sigma^*$ eine Sprache. Dann heißt L *verifizierbar* gdw. eine (klassische) Turing–Maschine $\top$ existiert, so dass Folgendes gilt. Wenn $w \in L$, dann gibt es ein(en Zeugen) $z \in \Sigma^*$, so dass $\top(w\#z) \downarrow +$; wenn $w \notin L$, so gilt für alle $z \in \Sigma^*$, dass $\top(w\#z) \downarrow -$.[5]

Lemma 10.10 *Sei $L \subset \Sigma^*$ eine Sprache. Die folgenden Aussagen sind äquivalent.*

(1) L ist verifizierbar.

(2) Es gibt eine nichtdeterministische Turing–Maschine $\top$, so dass jeder Rechenvorgang bei jeder Eingabe endlich ist und so dass $w \in L$ gdw. es einen Rechenvorgang bei Eingabe von w gibt, der w akzeptiert.

Lemma 10.11 *Eine Sprache $L \subset \Sigma^*$ ist in NP gdw. L verifizierbar ist, wobei dies durch eine Turing–Maschine $\top$ bezeugt wird, deren Laufzeit polynomiell in der ersten Komponente der Eingabe ist.*

[5]Hierbei sei # ein neues (Trennungs–)Symbol.

Kapitel 11

Der Satz von Cook und Levin

Für Sprachen $L_0, L_1 \subset \Sigma^*$ hatten wir in Definition 8.3 den Begriff der "(Turing–)Reduzibilität" definiert. Wir interessieren uns nun für eine Verschärfung dieses Begriffs.

Definition 12.1 *Eine Funktion $f : \Sigma^* \to \Sigma^*$ heißt in polynomieller Zeit berechenbar gdw. es eine Turing–Maschine $\top$ mit polynomieller Laufzeit gibt, so dass für alle $w \in \Sigma^*\top$ bei Eingabe von w hält, wobei am Ende $f(w)$ auf dem Rechenband geschrieben ist.*

Lemma 12.2 *Seien $L_0, L_1 \subset \Sigma^*$ Sprachen mit $L_0 \leq_p L_1$. Dann folgt aus $L_1 \in P$, dass auch $L_0 \in P$.*

Beweis: Der Beweis ist analog zum Beweis von Lemma 8.4, wobei wir allerdings auf Laufzeiten zu achten haben. Sei $w \in L_0$ gdw. $f(w) \in L_1$ für alle $w \in \Sigma^*$, wobei f in polynomieller Zeit berechenbar ist. Werde dies etwa durch die Turing–Maschine $\top$ bezeugt, deren Laufzeit $\leq 0(n \mapsto n^k)$ sei. Sei $\top'$ eine Turing–Maschine mit Laufzeit $\leq 0(n \mapsto n^m)$, so dass $\top'$ ein Wort $w' \in \Sigma^*$ akzeptiert/verwirft gdw. $w' \in L_1 / w' \notin L_1$. Sei U diejenige Turing–Maschine, die bei Eingabe von w zunächst das Programm von $\top$ laufen lässt und $f(w)$ berechnet und sodann das Programm von $\top'$ laufen lässt und entscheidet, ob $f(w) \in L_1$ oder nicht. Für hinreichend großes $n \in \mathbb{N}$ existieren dann Konstanten $c_0, c_1 \in \mathbb{R}^+$, so dass für alle w der Länge n Folgendes gilt. Da die Laufzeit von $\top$ bei Eingabe von w durch $c_0 \cdot n^k$ beschränkt ist, ist die Länge von $f(w)$ höchstens $c_0 \cdot n^k$, so dass die Laufzeit von $\top'$ bei Eingabe von $f(w)$ durch $c_1 \cdot (c_0 \cdot n^k)^m = c_1 \cdot c_0^m \cdot n^{km}$ beschränkt ist. Damit ist die Laufzeit von $U \leq 0(n \mapsto n^{km})$. Dies zeigt, dass L_0 in P ist.

Wir werden später Beispiele für $\leq_p$ kennen lernen. Wir wollen zunächst die allgemeine Theorie vorantreiben.

Definition 12.3 *Sei $L \subset \Sigma^*$ eine Sprache. L heißt NP–vollständig gdw. $L \in NP$ und für alle $L_0 \subset \Sigma^*$ mit $L_0 \in NP$ gilt $L_0 \leq_p L$.*

Satz 12.4 *Sei $L \subset \Sigma^*$ NP–vollständig. Dann gilt $P = NP$ gdw. $L \in P$.*

Beweis: Wenn $P = NP$, dann folgt $L \in P$ aus $L \in NP$. Sei umgekehrt $L \in P$. Sei $L_0 \in NP$. Dann folgt $L_0 \in P$ aus $L \in P$ mit Hilfe des obigen Lemmas.

Die Frage, ob $P = NP$ oder nicht, kann also entschieden werden, indem für ein konkretes NP–vollständiges Problem gezeigt wird, dass dieses in P bzw. nicht in P liegt. Wir werden sogleich mehrere NP–vollständige Probleme kennen lernen.

Lemma 12.5 *Sei $L \subset \Sigma^*$ NP–vollständig, und sei $L' \subset \Sigma^*$ in NP, wobei $L \leq_p L'$. Dann ist auch L' NP–vollständig.*

Beweis: Der Beweis von Lemma 12.2 zeigt auch, dass die Relation $\leq_p$ transitiv ist. Sei $L_0 \subset \Sigma^*$ in NP. Dann folgt aus $L_0 \leq_p L$ und $L \leq_p L'$, dass $L_0 \leq_p L'$. L' ist also NP–vollständig.

Satz 12.6 (Cook, Levin) *SAT ist NP–vollständig.*

Beweis: Wir haben im letzten Kapitel gesehen, dass $SAT \in NP$. Es bleibt also zu zeigen, dass jedes $L \in NP$ in polynomieller Zeit auf SAT reduzierbar ist.

Sei $L \subset \Sigma^*$ in NP. Sei $\top$ eine nichtdeterministische Turing–Maschine mit polynomieller Laufzeit, so dass gilt: $w \in L$ gdw. es einen Rechenvorgang von $\top$ bei Eingabe von w gibt, der w akzeptiert. Sei $k \in \mathbb{N}$ so, dass die Laufzeit von $\top \leq 0(n \mapsto n^k)$ ist. Wir nehmen im Folgenden tatsächlich an, dass die Laufzeit von $\top$ bei Eingabe eines Wortes w der Länge n höchstens $n^k - 3$ ist. (Andernfalls müssen im Folgenden nur die Notationen etwas geändert werden.)

Sei nun $w \in \Sigma^*$, und habe w die Länge n. Ein Rechenvorgang von $\top$ bei Eingabe von w ist dann eine Folge $(K_i : i < l)$ von Konfigurationen, wobei K_{l-1} den Zustand q_+ oder q_- beinhaltet. Es gilt $l \leq n^k$. Wir können uns einen solchen Rechenvorgang durch eine $n^k \times n^k$ Matrix (die wir jetzt auch "Tableau" nennen wollen) wie folgt veranschaulichen. Dabei sei $w \equiv \gamma_0 \dots \gamma_{n-1}$, wobei $\gamma_j \in \Sigma$ für alle $j < n$.

0	#	q_0	γ_0	γ_1	γ_2	$\cdots$	γ_{n-1}	$\sqcup$	$\cdots$	$\sqcup$	#
1	#										#
2	#										#
	#										#
	#										#
	#										#
	#										#
n^k-1	#										#
	0	1	2								

Dabei stellen wir uns vor, dass der Rechenvorgang ähnlich wie in den Beweisen von Satz 7.5 und Satz 8.1 mitgeteilt wird. Sie i^{te} Zeile dieses Tableaus steht für die i^{te} Konfiguration der betrachteten Berechnung, wobei wir wieder etwa durch $\gamma'_0, \ldots \gamma'_{r-1} q \gamma'_r \ldots \gamma'_s \sqcup \ldots \sqcup$ die Tatsache mitteilen, dass $\top$ aktuell im Zustand q ist, der Kopf auf der (mit γ'_r beschrifteten) r^{ten} Zeile steht und sich auf dem Rechenband die Inschrift $\gamma'_0 \ldots \gamma'_{r-1} \gamma'_r \ldots \gamma'_s \sqcup \ldots$ befindet.

Wir werden sogleich den übergang von K_i zu K_{i+1} ähnlich wie im Beweis von Satz 8.1 erfassen.

Unsere Aufgabe ist es, eine Turing–Maschine R mit polynomieller Laufzeit zu bauen, so dass R die Eingabe w in eine aussagenlogische Formel φ "übersetzt", so dass $w \in L$ gdw. φ erfüllbar ist. Die Tatsache, dass $w \in L$ gilt, ist aber äquivalent damit, dass es ein Tableau wie oben gibt, so dass q_+ eine der Tableaueintragungen ist und dieses Tableau für einen Rechenvorgang von $\top$ bei Eingabe von w steht.

Betrachten wir $Q \cup \Gamma \cup \{\#\}$. Hierbei ist Q die Menge der Zustände von $\top$, Γ (disjunkt von $\top$) ist das Schreibalphabet von $\top$ und # kommt nicht in $Q \cup \Gamma$ vor. Wir können sehr leicht in eindeutiger Art und Weise jeden Tripel $(i, j, s) \in \{0, \ldots, n^k-1\} \times \{0, \ldots, n^k-1\} \times (Q \cup \Gamma \cup \{\#\})$ eine Aussagenvariante aus $\{A_0, A_1, A_2, \ldots\}$ zuordnen.[1] Die (i, j, s) zugeordnete Aussagenvariable wollen wir einfach mit $A_{i,j,s}$ bezeichnen. Sei $\overline{\beta} : \{A_0, A_1, \ldots\} \to \{0, 1\}$. Dann soll $\overline{\beta}(A_{i,j,s}) = 1$ für die Tatsache stehen, dass die Tableauzeile mit Index (i, j) die Eintragung s enthält. Entsprechend steht $\overline{\beta}(A_{i,j,s}) = 0$ dafür, dass diese Tableauzeile nicht die Eintragung s enthält.

Wir konstruieren nun die aussagenlogische Formel φ. Die Maschine R wird dann ähnlich wie wir bei gegebenen w die Formel φ konstruieren. Es ist leicht sich zu überzeugen, dass R eine polynomielle Laufzeit hat.

[1] Wir könnten z.B. dem Tripel (i, j, s) die Aussagenvariable $A_{2^i \cdot 3^j \cdot 5^a}$ zuordnen, wobei s das a^{te} Element von $Q \cup \Gamma \cup \{\#\}$ gemäßeiner fixierten Aufzählung von $Q \cup \Gamma \cup \{\#\}$ ist.

Die Formel φ besitzt die Gestalt $\varphi_0 \wedge \varphi_1 \wedge \varphi_2 \wedge \varphi_3$.

Die Teilformel φ_0 sagt, dass jede Tableauzeile genau ein Symbol $s \in Q \cup \Gamma \cup \{\#\}$ enthält:

$$\varphi_0 \equiv \bigwedge_{0 \le i,j < n^k} \Big[\bigvee_s A_{i,j,s} \wedge \bigwedge_{s \neq t} (\neg A_{i,j,s} \vee \neg A_{i,j,t}) \Big].$$

Hierbei steht $\bigwedge$ für Konjunktionen und $\bigvee$ für Disjunktionen. Z.B. ist $\bigvee_s A_{i,j,s}$ die (endliche!) Disjunktion aller $A_{i,j,s}$, wobei $s \in Q \cup \Gamma \cup \{\#\}$.

Die Teilformel φ_1 beschreibt vollständig die oberste Tableauzeile:

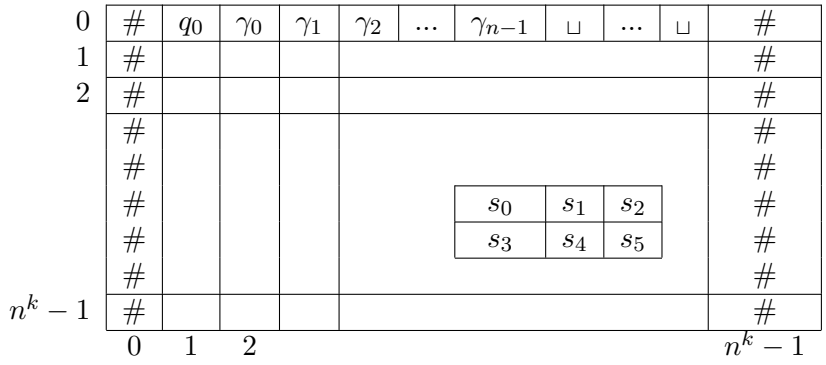
$$\varphi_1 \equiv A_{0,0,\#} \wedge A_{0,1,q_0} \wedge A_{0,2,\gamma_0} \wedge \ldots \wedge A_{0,n+1,\gamma_{n-1}} \wedge A_{0,n+2,\sqcup} \wedge \ldots \wedge A_{0,n^k-2,\sqcup} \wedge A_{0,n^k-1,\#}$$

Die Teilformel φ_2 besagt, dass die Eingabe w akzeptiert wird:

$$\varphi_2 \equiv \bigvee_{0 \le i,j < n^k} A_{i,j,q_+}.$$

Die Teilformel φ_3 schließlich drückt aus, dass das Tableau einen Rechenvorgang von $\top$ bei Eingabe von w entspricht. Hierzu müssen wir mitteilen, dass die $i+1^{\text{te}}$ Tableauzeile aus der i^{ten} Tableauzeile gemäßder übergangsfunktion δ hervorgeht. (Falls die i^{te} Tableauzeile bereits q_+ enthält, dann soll die $i+1^{\text{te}}$ Zeile einfach eine Wiederholung der i^{ten} Zeile sein.)

Für diesen Zweck betrachten wir 2×3–Fenster des Tableaus:



Ein solches Fenster besteht, für $i < n^k - 1$ und $j < n^k - 2$, aus den Tableauzeilen mit Index $(i,j), (i,j+1), (i,j+2), (i+1,j), (i+1,j+1)$ und $(i+1,j+2)$. Für die Tatsache, dass das Tableau einen Rechenvorgang wiedergibt, ist notwendig und hinreichend, dass jedes im Tableau vorkommende 2×3–Fenster im folgenden Sinne *legal* ist.

Sei $(q', b, L) \in \delta(q, a)$. Dann ist für alle $\gamma \in \Gamma$

γ	q	a
q'	γ	b

ein legales Fenster. Weiterhin seien für alle $\gamma, \gamma'' \in \Gamma$ und $\gamma' \in \Gamma \cup \{\#\}$

q	a	γ'
γ	b	γ'

,

a	γ''	γ'
b	γ''	γ'

,

γ'	γ	q
γ'	q'	γ

,

und

γ'	γ'	γ
γ'	γ'	q'

legale Fenster. Darüber hinaus seien auch

$\#$	q	a
$\#$	q'	b

und

q	a	γ
q'	b	γ

legale Fenster. (Letztere werden für die Situation benötigt, dass der Kopf am linden Bandende steht und aufgefordert wird, nach links zu gehen.)

Sei nun $(q', b, R) \in \delta(q, a)$. Dann sei für alle $\gamma \in \Gamma$

q	a	γ
b	q'	γ

ein legales Fenster. Weiterhin seien für alle $\gamma \in \Gamma$ und $\gamma' \in \Gamma \cup \{\#\}$

γ'	q	a
γ'	b	q'

,

γ'	γ	q
γ'	γ	b

und

a	γ	γ'
q'	γ	γ'

legale Fenster.

Schließlich seien alle Fenster der Form

x	y	z
x	y	z

mit $x, y, z \in \Gamma \cup \{\#, q_+\}$ legal. Alle übrigen Fenster gelten als illegal.

Wir setzen nun

$$\varphi_3 = \bigwedge_{\substack{i < n^k - 1 \\ j < n^k - 2}} \bigvee_{\substack{s_0, \ldots, s_5 \\ \text{ist legal}}} (A_{i,j,s_0} \wedge A_{i,j,s_1} \wedge A_{i,j+2,s_2} \wedge A_{i+1,j,s_3} \wedge A_{i+1,j+1,s_1} \wedge A_{i+1,j+2,s_5}).$$

Hierbei schreiben wir "$s_0, \ldots, s_5$ ist legal" für die Tatsache, dass da 2×3–Fenster

s_0	s_1	s_2
s_3	s_4	s_5

legal ist.

Die Formel $\varphi \equiv \varphi_0 \wedge \varphi_1 \wedge \varphi_2 \wedge \varphi_3$ ist erfüllbar gdw. $w \in L$. □

Die folgende nur scheinbar einfachere Variante von SAT, die wir SAT_3 nennen, ist ebenfalls NP–vollständig.

Eine aussagenlogische Formel φ ist ein konjunktiver Normalform gdw. φ die Gestalt

$$\bigwedge_i \bigvee_j \varphi_{i,j}$$

besitzt, wobei jedes $\varphi_{i,j}$ entweder atomar oder die Negation einer atomaren Formel ist. Eine aussagenlogische Formel ist in 3–*beschränkter konjunktiver Normalform* gdw. φ die Gestalt

$$\bigwedge_i (\varphi_{i,0} \vee \varphi_{i,1} \vee \varphi_{i,2})$$

besitzt, wobei jedes $\varphi_{i,j}$ entweder atomar oder die Negation einer atomaren Formel ist.

Das Problem SAT_3 lautet wie folgt. Sei φ eine aussagenlogische Formel in 3–beschränkter konjunktiver Normalform. Ist φ erfüllbar?

Satz 12.7 *SAT_3 ist NP–vollständig.*

Beweis: Diese Aussage ergibt sich aus dem Beweis des Satzes von Cook und Levin. Die zum vorgelegten w produzierte Formel φ kann in polynomieller Zeit in eine aussagenlogisch äquivalente Formel in 3–beschränkter konjunktiver Normalform umgewandelt werden.

Wir zeigen nun, dass da CLIQUE–Problem NP–vollständig ist. Dies ergibt sich jetzt sofort aus dem folgenden

Lemma 12.8 *$SAT_3 \leq_P$ CLIQUE.*

Beweis: Sei φ eine aussagenlogische Formel in 3–beschränkter konjunktiver Normalform. Sei etwa

$$\varphi \equiv \bigwedge_{i<n} (\varphi_{i,0} \vee \varphi_{i,1} \vee \varphi_{i,2}),$$

wobei jedes $\varphi_{i,j}$ entweder atomar oder Negation einer atomaren Aussage ist. Wir setzen $M = \{0, \dots, n-1\} \times \{0,1,2\}$ und ordnen φ den ungerichteten Graphen $G \subset [M]^2$ zu, wobei $\{(i,j),(i',j')\} \subset G$ gdw. $i \neq i'$ und es nicht der Fall ist, dass $\varphi_{i,j}$ die Negation von $\varphi_{i',j'}$ oder umgekehrt $\varphi_{i',j'}$ die Negation von $\varphi_{i,j}$ ist. Zwei Punkte $(i,j),(i',j')$ aus M sind also verbunden gdw. $i \neq i'$ und $\varphi_{i,j}$ nicht $\varphi_{i',j'}$ widerspricht. Jedes $\overline{\beta} : \{A_0, \dots\} \to \{0,1\}$, das die Erfüllbarkeit von φ bezeugt, entspricht einer n–CLIQUE in G und umgekehrt.

Auch das Problem Hamiltonscher Wege ist NP–vollständig:

Lemma 12.9 *SAT_3 ist in polynomieller Zeit auf das Problem Hamiltonscher Wege reduzierbar.*

Beweis: Sei wieder

$$\varphi \equiv \bigwedge_{i<n} (\varphi_{i,0} \vee \varphi_{i,1} \vee \varphi_{i,2})$$

gegeben, wobei jedes $\varphi_{i,j}$ entweder atomar oder Negation einer atomaren Formel ist.

Seien o.B.d.A. $A_0, A_1, \dots, A_{l-1}$ die Aussagenvariablen, die in φ vorkommen. (Es gibt also $l \leq 3n$.)

Wir setzen $M = \{s_0, s_1, \dots, s_l\} \cup \{0,1,\dots,n-1\} \cup \{(i,j) : i < l \wedge j < 3n+1\}$ [2] Hierbei stehen $0,1,\dots,n-1$ für die Konjunktionsglieder von φ, die Punkte (i,j) für $j < 3n+1$ stehen in gewisser Weise für die Aussagenvariable

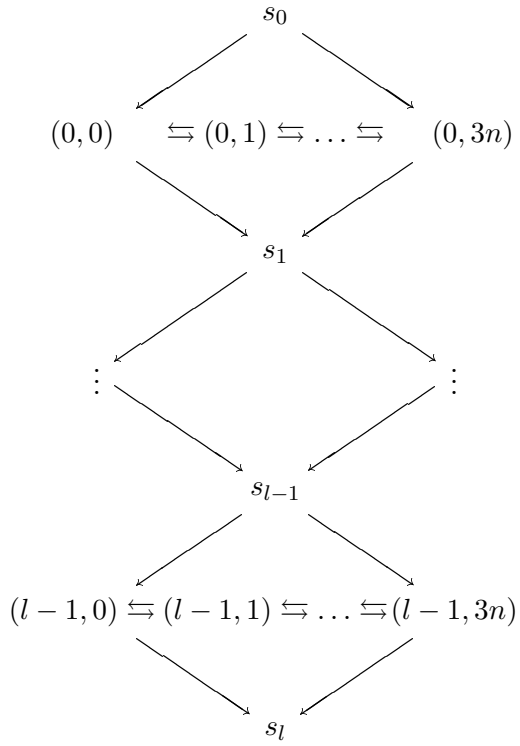
[2] Wir setzen voraus, dass die drei Mengen, deren Vereinigung M ist, paarweise disjunkt sind.

A_i, und die s_k sind "übergangspunkte". s_0 soll der Start und s_l der Zielpunkt werden.

Wir definieren nun $G \subset M \times M$ wie folgt.

Für alle $i < l$ und $j < 3n$ seien $((i,j),(i,j+1)) \in G$ und $((i,j+1),(i,j)) \in G$. Außerdem seien für alle $i < l$ sowohl $(s_i,(i,0)) \in G$ als auch $(s_i,(i,3n)) \in G$. Ferner seien für alle $i < l-1$ sowohl $((i,0),s_{i+1}) \in G$ als auch $((i,3n),s_{i+1}) \in G$.

Bislang ergibt sich damit das folgende Bild:



Die Punkte $0,1,\ldots,n-1$ sind noch nicht beteiligt. Im Moment gibt es zwei Hamiltonsche Wege durch $G \subset M \backslash \{0,1,\ldots,n-1\}$ von s_0 nach s_l.

Wir verbinden nun die Punkte $0,1,\ldots,n-1$ so mit Punkten (i,j), dass jeder Hamiltonsche Weg von s_0 nach s_l in konsistenter Weise für jedes Konjunktionsglied $\varphi_{i,0} \vee \varphi_{i,2}$ von φ entscheidet, ob $\varphi_{i,0}, \varphi_{i,1}$ oder $\varphi_{i,2}$ als wahr bewertet werden soll oder nicht. Dabei ist jedes $\varphi_{i,j}$ eines der $A_0, A_1, \ldots, A_{l-1}$ oder Negation eines der $A_0, A_1, \ldots, A_{l-1}$.

Betrachten wir das i^{te} Konjunktionsglied

$$\varphi_{i,0} \vee \varphi_{i,1} \vee \varphi_{i,2}$$

von φ. Sei, für $j < 3, A_{f(i,j)}$ so, dass $\varphi_{i,j} \equiv A_{f(i,j)}$, oder $\varphi_{i,j} \equiv \neg A_{f(i,j)}$. Wir verbinden dann wie folgt den Punkt $i \in M$ mit anderen Punkten aus M.

Es seien $(i, 3 \cdot f(i,j) + 2) \in G$ und $(3 \cdot f(i,j) + 3, i) \in G$, falls $\varphi_{i,j} \equiv A_{f(i,j)}$. Und es seien $(i, 3 \cdot f(i,j) + 3) \in G$ und $(3 \cdot f(i,j) + 2, i) \in G$, falls $\varphi_{i,j} \equiv \neg A_{f(i,j)}$. Dies beendet die Beschreibung von G.

Jeder Hamiltonsche Weg durch G von s_0 nach s_l basiert auf einem der beiden Hamiltonschen Wege durch die Einschränkung von G auf $M \backslash \{0, 1, \dots, n-1\}$. Allerdings müssen Umwege so eingeflochten werden, dass auch die Punkte aus $\{0, 1, \dots, n-1\}$ längs des Weges besucht werden. Ein Hamiltonscher Weg durch G von s_0 nach s_l liefert dann ein $\overline{\beta} : \{A_0, \dots\} \to \{0, 1\}$, das die Erfüllbarkeit von φ zeigt, wie folgt: wir setzen $\overline{\beta}(A_i) = 0$ bzw. 1 gdw. wir auf dem Hamiltonschen Weg von s_i aus nach $(i, 0)$ bzw. nach $(i, 3n)$ gehen. Umgekehrt liefert auch jedes $\overline{\beta}$, das die Erfüllbarkeit von φ zeigt, einen Hamiltonschen Weg durch G von s_0 nach s_l.

Kapitel 13

Peano–Arithmetik und Gödels zweiter Unvollständigkeitssatz

Die Sprache der PEANO–Arithmetik ist $\mathcal{L}_A$[1], d.h., die Sprache der elementaren Zahlentheorie. Diese Sprache besitzt die Konstante 0, die Funktoren $S, +, \cdot$ und E, sowie das Relationssymbol $<$. Das Standardmodell von $\mathcal{L}_A$ ist $\mathcal{N} = (\mathbb{N}; 0, <, S, +, \cdot, E)$.[2]

Die *Peano–Arithmetik*, kurz PA, besitzt die folgenden Axiome. Zunächst sind alle Axiome von A_E aus Kapitel 6 Axiome von PA, d.h.

(1) $\forall v_1 S(v_1) \neq 0$

(2) $\forall v_1 \forall v_2 (S(v_1) = S(v_2) \to v_1 = v_2)$

(3) $\forall v_1 \forall v_2 (v_1 < S(v_2) \leftrightarrow (v_1 < v_2 \vee v_1 = v_2))$

(4) $\forall v_1 \neg v_1 < 0$

(5) $\forall v_1 \forall v_2 (v_1 < v_2 \vee v_1 = v_2 \vee v_2 < v_1)$

(6) $\forall v_1 v_1 + 0 = v_1$

(7) $\forall v_1 \forall v_2 v_1 + S(v_2) = S(v_1 + v_2)$

(8) $\forall v_1 v_1 \cdot 0 = 0$

(9) $\forall v_1 \forall v_2 v_1 \cdot S(v_2) = v_1 \cdot v_2 + v_1$

[1] Das A steht für "Arithmetik".

[2] Das Symbol 0 steht also für die Null, d.h. für 0, etc. Zur Differenzierung hätten wir vielleicht $\dot{0}$ für das Symbol und 0 für das Objekt Null schreiben sollen.

(10) $\forall v_1 v_1 E0 = S(0)$

(11) $\forall v_1 \forall v_2 v_2 ES(v_2) = (v_1 E v_2) \cdot v_1$

Sodann enthält PA unendlich viele weitere Axiome. Sei φ eine $\mathcal{L}_A$–Formel, in der die Variablen $v_{i_0}, v_{i_1}, \ldots, v_{i_n}$ frei vorkommen. Dann ist das zu φ gehörige *Induktionsaxiom*

$$(12)_\varphi \quad \forall v_{i_1} \ldots \forall v_{i_n} (\varphi_0^{v_{i_0}} \wedge \forall v_{i_0} (\varphi \to \varphi_{Sv_{i_0}}^{v_{i_0}}) \to \forall v_{i_0} \varphi)$$

ebenfalls zu PA. Die Axiome von PA sind die $\mathcal{L}_A$–Sätze, die in der Menge

$$A_E \cup \{(12)_\varphi : \varphi \text{ ist } \mathcal{L}_A\text{–Formel}\}$$

liegen. Satz 7.4 besagt, dass A_E alle wahren Σ_1–Sätze von $\mathcal{L}_A$ beweist. Wir hatten diese Aussage benutzt, um den 1. Gödelschen Unvollständigkeitssatz zu zeigen. Die Induktionsaxiome können benutzt werden, um wahre Π_1–Sätze von $\mathcal{L}_A$ zu beweisen. Insbesondere werden wir sehen, dass PA den Satz

"alle wahren Σ_1–Sätze von $\mathcal{L}_A$ sind in A_E beweisbar",

bzw. die Übersetzung dieses Satzes in die Sprache $\mathcal{L}_A$ beweist. Dies wird sodann benutzt, um den 2. Gödelschen Unvollständigkeitssatz zu zeigen. Dieser besagt Folgendes: wenn $\top$ eine (u.U. triviale) rekursiv aufzählbare konsistente Erweiterung von PA ist, dann beweist $\top$ nicht die Konsistenz von $\top$. Da PA offensichtlich rekursiv aufzählbar ist und da (wie wir annehmen dürfen) PA konsistent ist, beweist also insbesondere PA nicht die Konsistenz von PA. Nach den Gödelschen Unvollständigkeitssätzen gibt es also ein (Nichtstandard–) Modell von PA $\cup$ {PA ist inkonsistent}.

Der Beweis des 2. Gödelschen Unvollständigkeitssatzes erfordert einige Vorbereitung. Insbesondere muss z.B. geklärt werden, wie "$\top$ ist konsistent" in der Sprache $\mathcal{L}_A$ ausdrückbar ist. Die Formalisierung des Satzes "$\top$ ist konsistent" wird ein Π_1–Satz sein, so dass für $\top \supset$ PA aus dem 2. Gödelschen Unvollständigkeitssatz der 1. Unvollständigkeitssatz folgt.

Zuerst müssen wir $\mathcal{L}_A$ "gödelisieren". Derartiges wurde bereits in Kapitel 7 getan. Wir wollen diesen Vorgang hier wiederholen und gründlich darstellen.

$\mathcal{L}_A$ besitzt die folgenden Symbole: $0, <, S, +, \cdot, E,$ sowie $(,), \neg, \to, \forall, =$ und $v_0, v_1, v_2, \ldots$ $\mathcal{L}_A$–Ausdrücke sind endliche Folgen derartiger Symbole. Seien $\gamma_0, \gamma_1, \ldots, \gamma_{n-1}$ Symbole. Wir wollen dem Ausdruck

$$a \equiv \gamma_0 \gamma_1 \ldots \gamma_{n-1}$$

eine "Gödelnummer" $\ulcorner a \urcorner$ zuordnen. Hierzu nehmen wir zunächst die folgenden Identifizierungen vor:

0	0
$<$	1
S	2
$+$	3
$\cdot$	4
E	5
$($	6
$)$	7
$\neg$	8
$\rightarrow$	9
$\forall$	10
$=$	11
v_0	12
v_1	13
v_2	14
$\vdots$	$\vdots$

Die Folge

$$a \equiv \gamma_0 \gamma_1 \dots \gamma_{n-1}$$

wird damit zu einer Folge natürlicher Zahlen. Die "Gödelnummer" $\ulcorner a \urcorner$ von a sei dann die Zahl

$$\ulcorner a \urcorner = p_0^{\gamma_0+1} \cdot p_1^{\gamma_1+1} \cdot \ldots \cdot p_{n-1}^{\gamma_{n-1}+1}.$$

Hierbei sei $p_0, p_1, \dots$ die natürliche Aufzählung aller Primzahlen. Offensichtlich können wir aus der Gödelnummer $\ulcorner a \urcorner$ die Folge (d.h. den Ausdruck) a "ablesen".

Bevor wir nun sagen können, was zu tun ist, um den 2. Unvollständigkeitssatz zu beweisen, wollen wir einige Konventionen vereinbaren.

Sei $n \in \mathbb{N}$. Wir wollen dann in Zukunft

$$\underbrace{SS \dots S0}_{n\text{–viele}}$$

als den Standardnamen von n bezeichnen; wir schreiben auch $\check{n}$ für diesen Standardnamen. Sei φ eine Formel, in der genau die Variablen $v_{i_0}, \dots, v_{i_{k-1}}$ mit $i_0 < \ldots < i_{k-1}$ frei vorkommen. Seien $m_0, \dots, m_{k-1} \in \mathbb{N}$. Wir schreiben dann

$$\varphi(m_0, \dots, m_{k-1})$$

für die Formel

$$\left(\dots\left(\varphi^{v_{i_0}}_{\check{m}_0}\right)\dots\right)^{v_{i_{k-1}}}_{\check{m}_{k-1}},$$

die aus φ hervorgeht, indem die Variable v_i (an ihren freien Vorkommnissen) durch den Standardnamen $\check{m}_i$ ersetzt wird, für $i = i_0, \dots, i_{k-1}$.[3] Sei nun $\top \supset$ PA eine rekursiv aufzählbare Menge von Axiomen (d.h. von $\mathcal{L}_A$–Formeln). Der technische Teil des Beweises des 2. Unvollständigkeitssatzes besteht in der Produktion einer Σ_1–Formel, für die wir $Bew_\top$ schreiben. $Bew_\top$ besitzt v_0 als einzige freie Variable, und es gelten die folgenden Aussagen.

Satz 12.1 (Fixpunktsatz) *Es gibt einen $\mathcal{L}_A$–Satz γ mit*

$$\top \vdash \gamma \leftrightarrow \neg Bew_\top(\ulcorner\gamma\urcorner).$$

In der Tat gibt es für jede $\mathcal{L}_A$–Formel ψ mit freier Variable v_i einen $\mathcal{L}_A$–Satz γ_ψ mit

$$\top \vdash \gamma_\psi \leftrightarrow \psi(\ulcorner\gamma_\psi\urcorner).$$

Satz 12.2 *Seien φ, ψ beliebige $\mathcal{L}_A$–Formeln. Dann gilt:*

(a) Wenn $\top \vdash \varphi$, dann $\top \vdash Bew_\top(\ulcorner\varphi\urcorner)$.

(b) $\top \vdash Bew_\top(\ulcorner\varphi \to \psi\urcorner) \to (Bew_\top(\ulcorner\varphi\urcorner) \to Bew_\top(\ulcorner\psi\urcorner))$.

(c) $\top \vdash Bew_\top(\ulcorner\varphi\urcorner) \to Bew_\top(\ulcorner Bew_\top(\ulcorner\varphi\urcorner)\urcorner)$.

Die Teilaussage (a) von Satz 12.2 wird sich daraus ergeben, dass für alle $\mathcal{L}_A$–Formeln φ gelten wird:

(*) $\top \vdash \varphi$ gdw. $\mathfrak{N} \models Bew_\top(\ulcorner\varphi\urcorner)$. (Da $Bew_\top\Sigma_1$ ist, folgt dann aus $\top \vdash \varphi$ sogar $A_E \vdash Bew_\top(\ulcorner\varphi\urcorner)$.) Insbesondere können wir sagen, dass $Bew_\top$ das "Beweisbarkeitsprädikat" formalisiert. Darüber hinaus folgt aus (*):

(**) $\top$ ist inkonsistent gdw.

$$\mathfrak{N} \vdash Bew_\top(\ulcorner 0 \neq 0\urcorner).$$

$Bew_\top(\ulcorner 0 \neq 0\urcorner)$ ist also eine faire Formalisierung der Tatsache, dass $\top$ inkonsistent ist

Aus den beiden obigen Sätzen ergibt sich nun:

[3] Eine frühere Konvention erklärte, was wir z.B. unter $\mathfrak{N} \models \varphi(m_0, \dots, m_{k-1})$ verstehen. Die beiden Konventionen stehen offenbar nicht in Konflikt miteinander.

Zweiter Gödelscher Unvollständigkeitssatz.

Sei $\top \supset \mathrm{PA}$ rekursiv aufzählbar und konsistent. Dann gilt

$$\top \nvdash \neg Bew_\top(\ulcorner 0 \neq 0 \urcorner).$$

Falls $\top$ inkonsistent ist, dann beweist $\top$ natürlich auch $\neg Bew_\top(\ulcorner 0 \neq 0 \urcorner)$, so dass $\top$ seine eigene Konsistenz genau dann beweist, wenn $\top$ inskonsistent ist.

Beweis des 2. Unvollständigkeitssatzes: Sei zunächst γ wie in Satz 12.1, d.h.

$$\top \vdash \gamma \leftrightarrow \neg Bew_\top(\ulcorner \gamma \urcorner).$$

Angenommen, $\top \vdash \gamma$. Dann gilt $\top \vdash Bew_\top(\ulcorner \gamma \urcorner)$ nach Satz 12.2 (a) und damit $\top \vdash \neg\gamma$ wegen der Eigenschaft von γ. Also beweist $\top$ sowohl γ als auch $\neg\gamma$, so dass $\top$ inkonsistent ist.

Dies zeigt: aus der Konsistenz von $\top$ (die wir voraussetzen) folgt $\top \nvdash \gamma$.

Wir betrachten nun den Satz $\gamma \to (\neg\gamma \to 0 \neq 0)$. Da dieser eine Tautologie ist, haben wir $\top \vdash \gamma \to (\neg\gamma \to 0 \neq 0)$. Wegen $\top \vdash \neg\gamma \leftrightarrow Bew_\top(\ulcorner \gamma \urcorner)$ folgt aber dann $\top \vdash \gamma \to (Bew_\top(\ulcorner \gamma \urcorner) \to 0 \neq 0)$. Satz 12.2 (a) liefert daraus $\top \vdash Bew_\top(\ulcorner \gamma \to (Bew_\top(\ulcorner \gamma \urcorner) \to 0 \neq 0) \urcorner)$. Zweimalige Anwendung von Satz 12.2 (b) ergibt dann

$$\top \vdash Bew_\top(\ulcorner \gamma \urcorner) \to (Bew_\top(\ulcorner Bew_\top(\ulcorner \gamma \urcorner) \urcorner) \to Bew_\top(\ulcorner 0 \neq 0 \urcorner)).$$

Satz 12.2 (c) besagt aber, dass

$$\top \vdash Bew_\top(\ulcorner \gamma \urcorner) \to Bew_\top(\ulcorner Bew_\top(\ulcorner \gamma \urcorner) \urcorner),$$

so dass also

$$\top \vdash Bew_\top(\ulcorner \gamma \urcorner) \to Bew_\top(\ulcorner 0 \neq 0 \urcorner),$$

und damit

$$\top \vdash \neg Bew_\top(\ulcorner 0 \neq 0 \urcorner) \to \neg Bew_\top(\ulcorner \gamma \urcorner).$$

Mit Hilfe der Eigenschaft von γ ergibt sich dann

$$\top \vdash \neg Bew_\top(\ulcorner 0 \neq 0 \urcorner) \to \gamma.$$

Hätten wir jetzt $\top \vdash \neg Bew_\top(\ulcorner 0 \neq 0 \urcorner)$, dann folgte $\top \vdash \gamma$, was nicht der Fall ist.

Wir haben also gezeigt, dass $\top \nvdash \neg Bew_\top(\ulcorner 0 \neq 0 \urcorner)$.

Leider ergeben sich oft die Dinge, die zu Gödels Zeiten in der Logik bewiesen wurden, durch ein derartiges Herumschieben von Symbolen.

Übrigens ergibt sich aus Satz 12.1 ein weiteres schönes Resultat, nämlich Tarskis Satz zur Undefinierbarkeit der Wahrheit:

Satz 12.3 (Tarski). *Es gibt keine $\mathcal{L}_A$–Formel φ mit freier Variable v_0, so dass*

$$\mathfrak{N} \models \varphi(\ulcorner \psi \urcorner) \text{ gdw. } \mathfrak{N} \models \psi$$

für alle $\mathcal{L}_A$–Sätze ψ.

Beweis: Andernfalls betrachte γ mit $\mathrm{PA} \vdash \gamma \leftrightarrow \neg\varphi(\ulcorner \gamma \urcorner)$. Dann folgt aus $\mathfrak{N} \models \gamma$, dass $\mathfrak{N} \models \neg\varphi(\ulcorner \gamma \urcorner)$, also $\mathfrak{N} \models \neg\gamma$. Widerspruch! □

Mit Hilfe des Beweises des Fixpunktsatzes ergibt sich weiterhin auch, dass $\top \vdash Bew_\top(\ulcorner \varphi \urcorner) \to \varphi$ nicht für alle Σ_1–Sätze φ gelten kann, da γ mit $\top \vdash \gamma \leftrightarrow \neg Bew_\top(\ulcorner \varphi \urcorner)$ Σ_1 sein wird und $\top \vdash \gamma \to Bew_\top(\ulcorner \gamma \urcorner)$.

Wei wird der obige Fixpunktsatz gezeigt? Hierzu benötigen wir eine $\mathcal{L}_A$–Formel, Sub, in der genau die Variablen v_0, v_1, v_2, v_3 frei vorkommen und die ausdrückt, dass "v_0 aus v_1 hervorgeht, indem jedes freie Vorkommnis der v_2–ten Variable durch den Standardnamen von v_3 ersetzt wird". Um dies genauer zu formulieren, vereinbaren wir eine Verallgemeinerung der oben getroffenen Konvention.

Sei φ eine Formel, in der genau die Variablen $v_{i_0}, \ldots, v_{i_{k-1}}$ mit $i_0 < \ldots < i_{k-1}$ frei vorkommen. Seien $m_0, \ldots, m_{k-1} \in \mathbb{N}$. Wir schreiben dann z.B.

$$\varphi(v_{i_0}, \ldots, v_{i_{l-1}}, m_l, v_{i_{l+1}}, \ldots, v_{i_{k-1}})$$

für die Formel

$$\varphi_{\check{m}_l}^{v_{i_l}}.$$

Analog dürfte klar sein, was wir mit

$$\varphi(v_{i_0}, \dots, v_{i_{l-1}}, m_l, v_{i_{l+1}}, \dots, v_{i_{j-1}}, m_j, v_{i_{j+1}}, \dots, v_{i_{k-1}})$$

meinen, usw.

Wir wollen nun Folgendes.

Lemma 12.4 *Für alle $\mathcal{L}_A$–Formeln φ und für alle $i, n \in \mathbb{N}$ gilt*

$$\top \vdash \forall v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, i, n) \leftrightarrow v_0 = \ulcorner\varphi_{\check{n}}^{v_i}\urcorner).$$

Mit Hilfe dieses Lemmas zeigt sich der Fixpunktsatz wie folgt. Sei ψ eine $\mathcal{L}_A$–Formel mit freier Variable v_i. Sei o.B.d.A. $i = 0$. Wir definieren φ mit freier Variable v_3 durch

$$\varphi \equiv \exists v_0 \exists v_1(\mathrm{Sub}(v_0, v_1, 3, v_3) \wedge v_1 = v_2 \wedge \psi).$$

Sei dann

$$\gamma \equiv \gamma_\psi \equiv \varphi^{v_3}_{\ulcorner\varphi\urcorner v} \equiv \varphi(\ulcorner\varphi\urcorner).$$

Es gilt dann auf Grund des obigen Lemmas

$$\top \vdash \forall v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, \ulcorner\varphi\urcorner) \leftrightarrow v_0 = \ulcorner\gamma\urcorner).$$

Nun haben wir also

$$\top \vdash \psi(\ulcorner\gamma\urcorner) \leftrightarrow \exists v_0(v_0 = \ulcorner\gamma\urcorner \wedge \psi),$$

und damit

$$\top \vdash \psi(\ulcorner\gamma\urcorner) \leftrightarrow \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, \ulcorner\varphi\urcorner) \wedge \psi),$$

also auch

$$\top \vdash \psi(\ulcorner\gamma\urcorner) \leftrightarrow \exists v_0 \exists v_1(\mathrm{Sub}(v_0, v_1, 3, \ulcorner\varphi\urcorner) \wedge v_1 = \ulcorner\varphi\urcorner \wedge \psi),$$

d.h.

$$\top \vdash \psi(\ulcorner\gamma\urcorner) \leftrightarrow \varphi(\ulcorner\varphi\urcorner),$$

und damit

$$\top \vdash \psi(\ulcorner\gamma\urcorner) \leftrightarrow \gamma.$$

Es bleibt uns also, die $\mathcal{L}_A$–Formeln Sub und $Bew_\top$ so zu konstruieren, dass Lemma 12.1 und Satz 12.2 für sie gelten.

Da wir uns nicht so sehr für das Aussehen von Sub und $Bew_\top$ als vielmehr für die Existenz dieser beiden Formeln interessieren, können wir auf früher Geleistetes zurückgreifen.

Wir setzen voraus, dass $\top$ rekursiv aufzählbar ist. Nach Kapitel 4, ist dann auch

$$\{\varphi : \varphi \text{ ist } \mathcal{L}_A\text{–Formel und } \top \vdash \varphi\}$$

rekursiv aufzählbar. Sei $\top_0$ eine Turing–Maschine, die φ akzeptiert gdw. φ eine $\mathcal{L}_A$–Formel ist mit $\top \vdash \varphi$. Auf Grund von Satz 7.5 existiert eine Σ_1–Formel $\chi_{\top_0}$ (mit freier Variable v_0), so dass $\mathfrak{N} \models \chi_{\top_0}(\ulcorner\varphi\urcorner)$ gdw. $\top_0(\ulcorner\varphi\urcorner)\downarrow$, d.h. $\mathfrak{N} \models \chi_{\top_0}(\ulcorner\varphi\urcorner)$ gdw. φ eine $\mathcal{L}_A$–Formel mit $\top \vdash \varphi$ ist. Wir können also $Bew_\top \equiv \chi_{\top_0}$ wählen.

Die Menge aller $(\ulcorner\psi\urcorner, \ulcorner\varphi\urcorner, i, n)$, so dass ψ aus φ hervorgeht, indem jedes freie Vorkommnis von v_i durch $\check{n}$ ersetzt wird, ist sogar Turing–entscheidbar. Damit gibt es eine Σ_1–Formel Sub und eine Π_1–Formel Sub$'$, so dass $\mathfrak{N} \models \mathrm{Sub}(\ulcorner\psi\urcorner, \ulcorner\varphi\urcorner, i, n)$ gdw. $\mathfrak{N} \models \mathrm{Sub}'(\ulcorner\psi\urcorner, \ulcorner\varphi\urcorner, i, n)$ gdw. ψ aus φ hervorgeht, indem jedes freie Vorkommnis von v_i durch $\check{n}$ ersetzt wird.

Wir betrachten nun Satz 12.2 und Lemma 12.1. Wir beginnen mit Satz 12.2. Sei zunächst $\top \vdash \varphi$ für eine $\mathcal{L}_A$–Formel φ vorausgesetzt. Dann gilt auch $\mathfrak{N} \models Bew_\top(\ulcorner\varphi\urcorner)$ auf Grund der Wahl von $Bew_\top$. Da aber $Bew_\top$ Σ_1 ist, folgt damit aus Satz 7.4, dass $A_E \vdash Bew_\top(\ulcorner\varphi\urcorner)$, also erst recht $\top \vdash Bew_\top(\ulcorner\varphi\urcorner)$. Dies zeigt Satz 12.2 (a).

Die Aussage von Satz 12.2 (b) ist die formale Variante der Tatsache, dass aus $\top\varphi \to \psi$ und $\top \vdash \varphi$ auch $\top \vdash \psi$ folgt (welches sich wiederum daraus ergibt, dass man die Beweise von $\varphi \to \psi$ und φ aus $\top$ neben–/untereinander schreibt und sodann einmal den modus ponens anwendet). Satz 12.2 (b) ergibt sich somit leicht aus der Konstruktion von $Bew_\top$.

Anstelle von Satz 12.2 (c) zeigen wir die folgende allgemeine Aussage:

Lemma 12.5 *Sei ψ eine Σ_1–Formel von $\mathcal{L}_A$. Dann gilt*

$$\mathrm{PA} \vdash \psi \to Bew_\top(\ulcorner\psi\urcorner).$$

Die Aussage dieses Lemmas kann als formalisierte Variante von Satz 7.4 angesehen werden.

Wir zeigen Lemma 12.5 zunächst für Σ_0–Formeln ψ. Sei also ψ eine Σ_0–Formel von $\mathcal{L}_A$. Wir wollen annehmen, dass ψ sogar ein Satz ist. Nun gilt sicherlich $\mathfrak{N} \models \psi \to Bew_\top(\ulcorner\psi\urcorner)$ auf Grund von Satz 7.4, also $\mathfrak{N} \models \neg\psi$ oder $\mathfrak{N} \models Bew_\top(\ulcorner\psi\urcorner)$. Falls $\mathfrak{N} \models \neg\psi$, dann ist, da $\neg\psi$ Σ_0 ist, $\neg\psi$ in A_E beweisbar,

d.h. $A_E \vdash \neg\psi$. Falls $\mathfrak{N} \models Bew_\top(\ulcorner\psi\urcorner)$, dann ist $Bew_\top(\ulcorner\psi\urcorner)$. In jedem Falle ergibt sich also $A_E \vdash \neg\psi \vee Bew_\top(\ulcorner\psi\urcorner)$, also auch $\top \vdash \psi \to Bew_\top(\ulcorner\psi\urcorner)$.

Dieses Argument funktioniert nicht mehr, falls ψ ein Σ_1–Satz ist, da dann $\neg\psi$ Π_1 ist. Sei also nun ψ ein Σ_1–Satz von $\mathcal{L}_A$.

Sei $\psi \equiv \exists \vee_3 \varphi$, wobei φ Σ_0 ist. Der Schlüssel ist das folgende

Lemma 12.6 $\top \vdash \forall v_3[\varphi \to \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, v_3) \wedge Bew_\top(v_0))]$.

Mit Hilfe von Lemma 12.6 zeigt sich Lemma 12.5 wie folgt. Es gilt

$$\top \vdash \exists v_3 \exists v_0[\varphi \to \exists v_3 \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, v_3) \wedge Bew_\top(v_0))].$$

Also

$$\top \vdash \exists v_3 \varphi \to Bew_\top(\ulcorner\exists v_3 \varphi\urcorner)$$

wie gewünscht.

Der Beweis von Lemma 12.6 benutzt $\top \supset \mathrm{PA}$.
In $\top$ wird $\forall v_3[\varphi \to \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, v_3) \wedge Bew_\top(v_0))]$ durch Induktion gezeigt. Die zu zeigende Aussage reduziert sich also auf die beiden Aussagen (1) und (2).

(1) $\top \vdash \varphi(0) \to \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, 0) \wedge Bew_\top(v_0))$.
Dies ergibt sich sehr leicht, indem wir wieder beachten, dass φ Σ_0 ist.

(2) $$\begin{aligned}\top \vdash &(\varphi \to \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, v_3) \wedge Bew_\top(v_0))) \\ &\to (\varphi^{v_3}_{Sv_3} \to \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, Sv_3) \wedge Bew_\top(v_0)).\end{aligned}$$

Hierzu stellt sich aber heraus, dass, da φ Σ_0 ist, sogar

$$\top \vdash \varphi^{v_3}_{Sv_3} \to \exists v_0(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, 3, Sv_3) \wedge Bew_\top(v_0))$$

gilt.

Dies zeigt Lemma 12.6 und beendet somit den Beweis von Satz 12.2.

Der Beweis von Lemma 12.4 verläuft in gewisser Weise analog. Zu zeigen ist

(1) $\top \vdash (\mathrm{Sub}(\ulcorner\varphi^{v_i}_{\dot{n}}\urcorner, \ulcorner\varphi\urcorner, i, n)$, was einfach ist, und

(2) $\top \vdash \forall v_0 \forall v_0'(\mathrm{Sub}(v_0, \ulcorner\varphi\urcorner, i, n) \wedge \mathrm{Sub}(v_0', \ulcorner\varphi\urcorner, i, n) \to v_0 = v_0')$.

Zu Letzterem zeigt man induktiv in $\top$, dass Sub in der nullten Komponente funktional ist. Wir übergehen die Details.

Ultraprodukte

Wir wollen in diesem Kapitel einen alternativen Beweis des Kompaktheitssatzes kennen lernen. Dieser Beweis verwendet die Methode der Konstruktion neuer Modelle durch Ultraprodukte.

Definition 12.7 *Sei $I \neq \emptyset$ eine beliebige Menge. Eine Menge F von Teilmengen von I heißt ein* Filter auf I *gdw.*

(a) wenn $X \in F$ und $Y \in F$, dann ist auch $X \cap Y \in F$,

(b) wenn $X \in F$ und $Y \supset X$, wobei $Y \subset I$, dann ist auch $Y \in F$,

(c) $I \in F$, und

(d) $\emptyset \notin F$.

Sei etwa u eine nichtleere Teilmenge von I. Dann ist

$$\{X \subset I : u \subset X\}$$

ein Filter auf I. Dieser heißt der *von u erzeugte prinzipale Filter*. Sei I unendlich. Dann ist

$$\{X \subset I : I \backslash X \text{ ist endlich}\}$$

ein Filter auf I, der aus allen *koendlichen* Teilmengen von I besteht. Dieser heißt der Frechét–Filter auf I.

Definition 12.8 *Sei $I \neq \emptyset$, und sei F ein Filter auf I. Dann heißt F* Ultrafilter auf I *gdw für jedes $X \in I$ gilt: $X \in F$ oder $I \backslash X \in F$. wenn F Ultrafilter auf I ist, dann gilt für jedes $X \subset I$ genau eine der beiden Aussagen $X \in F, I \backslash X \in F$. Mit Hilfe des Hausdorffschen Maximalitätsprinzip (siehe Satz 5.8) zeigt man den folgenden*

Satz 12.9 (Tarski) *Sei $I \neq \emptyset$, und sei F ein Filter auf I. Dann existiert ein Ultrafilter U auf I, der F fortsetzt, d.h. so dass $U \supset F$.*

Sei nun $\mathcal{L}$ eine Sprache der Logik erster Stufe. Sei $I \neq \emptyset$, und sei für jedes $i \in I$ ein $\mathcal{L}$–Modell $\mathfrak{M}_i$ gegeben. Sei weiterhin U ein Ultrafilter auf I. Wir wollen dann das Ultraprodukt der Modelle $\mathfrak{M}_i$ mittels U, in Zeichen

$$\prod_{i \in I} \mathfrak{M}_i / U,$$

definieren. Für diesen Zweck wollen wir voraussetzen, dass $\mathcal{L}$ weder Konstanten noch Funktoren und ein einziges Relationssymbol, nämlich das zweistellige Relationssymbol R, besitzt.[4] Alles, was im Folgenden entwickelt wird, lässt sich aber in offensichtlicher Weise für beliebige Sprachen verallgemeinern.

Wir definieren zunächst die Trägermenge des Ultraproduktes. Mit $|\mathfrak{M}_i|$ bezeichnen wir die Trägermenge des Modells $\mathfrak{M}_i$. Es sei F^* die Menge aller Funktionen f mit Definitionsbereich I, so dass $f(i) \in |\mathfrak{M}_i|$ für alle $i \in I$. Für $f, g \in F^*$ schreiben wir $f \sim g$ gdw. $\{i \in I : f(i) = g(i)\} \in U$.

Lemma 12.10 *$\sim$ ist eine äquivalenzrelation.*

Beweis: Lediglich die Transitivität von $\sim$ ist nicht trivial. Aus $f \sim g$ und $g \sim h$, d.h. $\{i \in I : f(i) = g(i)\} \in U$ und $\{i \in I : g(i) = h(i)\} \in U$ folgt aber wegen $\{i \in I : f(i) = h(i)\} \supset \{i \in I : f(i) = g(i)\} \cap \{i \in I : g(i) = h(i)\}$, dass $\{i \in I : f(i) = h(i)\} \in U$, also $f \sim h$. □

Wir schreiben $[f]$ für die äquivalenzklasse von $f \in F^*$, d.h. $[f] = \{g \in F^* : g \sim f\}$. Wir schreiben auch $\mathfrak{F}$ für die Menge aller $[f]$ mit $f \in F^*$. $\mathfrak{F}$ wird die Trägermenge des Ultraproduktes sein.

Jedes der Modelle $\mathfrak{M}_i$ interpretiert R, d.h. $R^{\mathfrak{M}_i} \subset |\mathfrak{M}_i| \times |\mathfrak{M}_i|$. Wir wollen nun ein $\tilde{R} \subset \mathfrak{F} \times \mathfrak{F}$ definieren. Wir setzen $([f], [g]) \in \tilde{R}$ gdw. $\{i \in I : (f(i), g(i)) \in R^{|\mathfrak{M}_i|}\} \in U$. Man sieht leicht, d.h. für $f \sim f'$ und $g \sim g'$ gilt $\{i \in I : (f(i), g(i)) \in R^{|\mathfrak{M}_i|}\} \in U$ gdw. $\{i \in I : (f'(i), g'(i)) \in R^{|\mathfrak{M}_i|}\} \in U$. Wir haben damit ein $\mathcal{L}$–Modell konstruiert, nämlich $(\mathfrak{F}; \tilde{R})$. Dieses Modell besitzt die Trägermenge $\mathfrak{F}$ und interpretiert F durch $\tilde{R}$. Dieses so konstruierte Modell bezeichnen wir als das *Ultraprodukt der Modelle $\mathfrak{M}_i$ mittels U*.

Die folgende Aussage ist von zentraler Bedeutung.

[4] Die Sprache $\mathcal{L}_\in$ der Mengenlehre besitzt beispielsweise diese Gestalt.

Satz 12.11 (Łoś) *Seien für $i \in I$ β_i eine $\mathfrak{M}_i$–Belegung, und sei die $\prod_{i\in I}\mathfrak{M}_i/u$–Belegung β wie folgt definiert: $\beta(v_k) = [f]$, wobei $f(i) = \beta_i(v_k)$ für alle $i \in I$. Dann gilt für alle Formeln φ*

$$\prod_{i\in I}\mathfrak{M}_i/u \models \varphi[\beta] \text{ gdw. } \{i \in I : \mathfrak{M}_i \models \varphi[\beta]\} \in U.$$

Beweis durch Induktion nach der Komplexität von φ. Wir schreiben $\tilde{\mathfrak{M}}$ für $\prod_{i\in I}\mathfrak{M}/U$.

Sei zunächst φ atomar. Sei etwa $\varphi \equiv v_k = v_l$. Dann gilt $\tilde{\mathfrak{M}} \models v_k = v_l[\beta]$ gdw. $\beta(v_k) = \beta(v_l)$ gdw. $[f] = [g]$, wobei $f(i) = \beta_i(v_k)$ und $g(i) = \beta_i(v_l)$ für alle $i \in I$, gdw. $\{i \in I : \beta_i(v_k) = \beta_i(v_l)\} \in U$ gdw. $\{i \in I : \mathfrak{M}_i \models v_k = v_l[\beta_i]\} \in U$. Für $\varphi \equiv Rv_kv_l$ ist das Argument völlig analog.

Sei nun $\varphi \equiv \neg\psi$. Hier benutzt der Induktionsschritt die Tatsache, dass U ein Ultrafilter (und nicht etwa nur ein Filter) auf I ist. Es gilt $\mathfrak{M} \models \neg\psi[\beta]$ gdw. $\mathfrak{M} \not\models \psi[\beta]$ gdw. $\{i \in I : \mathfrak{M}_i \models \psi[\beta_i]\} \notin U$ nach Induktionsvoraussetzung, gdw. $\{i \in I : \mathfrak{M}_i \not\models \psi[\beta_i]\} \in U$, da U ein Ultrafilter ist, gdw. $\{i \in I : \mathfrak{M}_i \models \neg\psi[\beta_i]\} \in U$.

Der Induktionsschritt für $\varphi \equiv \psi \to \psi'$ ist sehr einfach.

Wir betrachten schließlich den Fall, dass $\varphi \equiv \forall v_k\psi$.

Setzen wir zunächst voraus, dass $\{i \in I : \mathfrak{M}_i \models \forall v_k\psi[\beta_i]\} \in U$. Sei $[f] \in |\tilde{\mathfrak{M}}| = \mathfrak{F}$ beliebig. Dann gilt $\{i \in I : \mathfrak{M}_i \models \psi[\beta_i(v_k|f(i))]\} \supset \{i \in I : \mathfrak{M}_i \models \forall v_k\psi[\beta_i]\}$, also $\{i \in I : \mathfrak{M}_i \models \psi[\beta_i(v_k|f(i))]\} \in U$. Die Induktionsvoraussetzung liefert dann $\tilde{\mathfrak{M}} \models \psi[\beta(v_k|[f])]$. Da $[f]$ beliebig war, haben wir also $\mathfrak{M} \models \forall v_k\psi[\beta]$ gezeigt.

Setzen wir nun voraus, dass $\{i \in I : \mathfrak{M}_i \models \forall v_k\psi[\beta_i]\} \notin U$, d.h. $\{i \in I : \mathfrak{M}_i \models \exists v_k\neg\psi[\beta_i]\} \in U$. Wir wollen zeigen, dass $\tilde{\mathfrak{M}} \not\models \forall v_k\psi[\beta]$, d.h. $\tilde{\mathfrak{M}} \models \exists v_k\neg\psi[\beta]$. Mit Hilfe des Auswahlaxioms finden wir ein $f \in F^*$, so dass $\mathfrak{M}_i \models \neg\psi[\beta_i(v_k|f(i))]$ für alle $i \in I$, für die ein $a \in |\mathfrak{M}_i|$ mit $\mathfrak{M}_i \models \neg\psi[\beta_i(v_k|a)]$ existiert. Dann gilt offensichtlich

$$\{i \in I : \mathfrak{M}_i \models \neg\psi[\beta_i(v_k|f(i))]\} = \{i \in I : \mathfrak{M}_i \models \exists v_k\psi[\beta_i]\} \subset U,$$

und damit mit Hilfe der Induktionsvoraussetzung $\mathfrak{M} \models \neg\psi[\beta(v_k|[f])]$, wie gewünscht. □

Mit Hilfe dieser Methode zeigt sich nun der Kompaktheitssatz sehr leicht wie folgt. Sei Σ eine endlich erfüllbare Menge von $\mathcal{L}$–Formeln, d.h. für jedes endliche $\sigma \subset \Sigma$ existiert ein $\mathcal{L}$–Modell $\mathfrak{M}_\sigma$ und eine $\mathfrak{M}_\sigma$–Belegung β_σ mit

$$\mathfrak{M}_\sigma \models \sigma[\beta_\sigma].$$

Gesucht ist ein $\mathcal{L}$–Modell $\tilde{\mathfrak{M}}$ und eine $\tilde{\mathfrak{M}}$–Belegung β mit $\tilde{\mathfrak{M}} \models \Sigma[\beta]$. Sei I die Menge aller endlichen Teilmengen von Σ. Für $\sigma \in I$ sei

$$X_\sigma = \{\tau \in I : \tau \supset \sigma\}.$$

Die Menge X_σ genieren einen Filter F auf I wie folgt: Wir setzen

$$X \in F \text{ gdw. } \exists \sigma \in I X \supset X_\sigma.$$

Lemma 12.12 *F ist ein Filter auf I.*

Beweis: (a): Seien $X, Y \in F$. Seien $\sigma, \tau \in I$ so, dass $X \supset X_\sigma$ und $Y \supset X_\tau$. Dann gilt $X \cap Y \supset X_\sigma \cap X_\tau = X_{\sigma \cup \tau}$, also auch $X \cap Y \in F$. (b) ist trivial. (c): Da $\emptyset \in I$, gilt $I \in F$. (d): Da $\sigma \in X_\sigma$ für alle $\sigma \in I$, gilt $\emptyset \notin F$. □

Sei nun (mit Hilfe des Satzes von Tarski) U ein Ultrafilter auf I, der F fortsetzt. Wir setzen dann $\tilde{\mathfrak{M}} = \prod_{\sigma \in I} \mathfrak{M}_\sigma / U$. Wir definieren eine $\tilde{\mathfrak{M}}$–Belegung β durch: $\beta(v_k) = [f]$, wobei $f(\sigma) = \beta_\sigma(v_k)$ für alle $\sigma \in I$. Es genügt nun zu zeigen, dass $\mathfrak{M} \models \Sigma[\beta]$. Sei also $\varphi \in \Sigma$. Dann folgt aus $\varphi \in \sigma$ (d.h. $\{\varphi\} \subset \sigma\}$, dass $\mathfrak{M}_\sigma \models \varphi[\beta_\sigma]$, also $\{\sigma : \mathfrak{M}_\sigma \varphi[\beta_\sigma]\} \supset X_{\{\varphi\}} \in UK$, und damit auch $\{\sigma : \mathfrak{M}_\sigma \models \varphi[\beta_\sigma]\} \in U$. Auf Grund des Satzes von Łoś gilt dann $\tilde{\mathfrak{M}} \models \varphi[\beta]$ wie gewünscht.

Ein Spezialfall von Ultraprodukten ist die Ultrapotenz. Sei $I \neq \emptyset$, und sei U ein Ultrafilter auf I. Sei $\mathfrak{M}$ ein $\mathcal{L}$–Modell, und sei $\mathfrak{M}_i = \mathfrak{M}$ für alle $i \in I$. Dann schreiben wir:

$$Ult(\mathfrak{M}; U)$$

für $\prod_{i \in I} \mathfrak{M}_i / U$ und nennen dies die *Ultrapotenz von $\mathfrak{M}$ mittels U*.

Der Satz von Łoś ergibt sofort das folgende

Korollar 12.13 *Sei $\bar{\beta}$ eine $\mathfrak{M}$–Belegung, und sei die $Ult(\mathfrak{M}; U)$–Belegung β definiert durch $\beta(v_k) = [c_{\bar{\beta}(v_k)}]$, wobei $c_{\bar{\beta}(v_k)}$ die konstante Funktion mit Wert $\bar{\beta}(v_k)$ (und Definitionsbereich I) ist. Dann gilt*

$$Ult(\mathfrak{M}; U) \models \varphi[\beta] \text{ gdw. } \mathfrak{M} \models \varphi[\bar{\beta}]$$

für alle Formeln φ.

Es existiert auch eine natürliche "Einbettung" e von $\mathfrak{M}$ nach $Ult(\mathfrak{M}; U)$. Für $a \in |\mathfrak{M}|$ setzen wir $e(a) = [c_a]$, wobei c_a die konstante Funktion mit Wert a (und Definitionsbereich I) ist. Es gilt dann (unter Verwendung der Schreibweise aus Kapitel 3):

Korollar 12.14 *Es gilt*

$$Ult(\mathfrak{M}; U) \models \varphi(e(a_0), \dots, e(a_{j-1})) \text{ gdw. } \mathfrak{M} \models \varphi(a_0, \dots, a_{j-1})$$

für alle Formeln φ *und* $a_0, \dots, a_{j-1} \in |\mathfrak{M}|$.

Auf diese Art und Weise lassen sich sehr leicht "Nichtstandardmodelle" konstruieren. Sei etwa

$$\mathfrak{N} = (\mathbb{N}; 0, S, <, +, \cdots, E),$$

und sei U ein Ultrafilter auf $\mathbb{N}$, der den Frechét–Filter auf $\mathbb{N}$ fortsetzt. Dann ist $Ult(\mathfrak{N}; U)$ nicht isomorph zu $\mathfrak{N}$ (d.h. die Einbettung e ist nicht surjektiv). Sei

$$\mathcal{R} = (\mathbb{R}; 0, 1, <, +, \cdot, E),$$

und sei wieder U ein Ultrafilter auf $\mathbb{N}$, der den Frechét–Filter auf $\mathbb{N}$ fortsetzt. Dann enthält $Ult(\mathcal{R}; U)$ "infinitesimale" Zahlen als auch "unendlich große" Zahlen.

Kapitel 13

Arithmetik und Mengenlehre

Wir wollen nun die natürlichen Zahlen benutzen, um ein Modell der Mengenlehre zu konstruieren. In der Mengenlehre kann mit weniger Kodierungsaufwand über mehr Dinge gesprochen werden.

Wir fassen natürliche Zahlen wie folgt als "Mengen" auf. Sei $n \in \mathbb{N}$. Schreibe n in Dualdarstellung, d.h. $n = \Sigma m_j \cdot 2^i$, wobei $m_i \in \{0, 1\}$ für alle i. Wir fassen dann n als die Menge aller i auf, so dass $m_i = 1$. Anders gesagt: wir definieren eine zweistellige Relation $E \subset \mathbb{N} \times \mathbb{N}$ auf $\mathbb{N}$ wie folgt. Seien $k, n \in \mathbb{N}$. Dann gelte kEn gdw. $m_k = 1$, wobei $n = \Sigma m_i \cdot 2^i$ die Dualdarstellung von n ist.

Die Sprache $\mathcal{L}_\in$ der Mengenlehre besitzt weder Konstanten noch Funktoren und als einziges Relationssymbol das zweistellige $\in$ für "ist Element von". Wenn $M \neq \emptyset$ und $R \subset M \times M$, so ist $(M; R)$ Modell von $\mathcal{L}_\in$. Insbesondere is $(\mathbb{N}; E)$ Modell von $\mathcal{L}_\in$. Wir wollen nun untersuchen, ob $(\mathbb{N}; E)$ ein "sinnvolles" Modell von $\mathcal{L}_\in$ ist, d.h. ob $(\mathbb{N}; E)$ Modell eines hinreichend großen Fragments der Standardaxiomatisierung der Mengenlehre ist. Hierzu müssen wir letztere kennen lernen.

Das erste Axiom, das *Extensionalitätsaxiom*, besagt, dass zwei Mengen gleich sind gdw. sie dieselben Elemente besitzen.

(Ext) $\forall x \forall y (x = y \leftrightarrow \forall z (z \in x \leftrightarrow z \in y))$.

Offensichtlich gilt $(\mathbb{N}; E) \models (Ext)$.

Das nächste Axiom, das *Fundierungsaxiom*, besagt, dass jede nichtleere Menge ein $\in$–minimales Element besitzt.

$(Fund)$ $\forall x (\exists y\, y \in x \rightarrow \exists y (y \in x \wedge \neg \exists z (z \in y \wedge z \in x)))$.

Mit Hilfe von Abkürzungen schreibt sich dies besser. Wir schreiben $x = \emptyset$ für $\neg\exists y\ y \in x, x \neq \emptyset$ für $\neg x = \emptyset, x \cap y = \emptyset$ für $\neg\exists z(z \in x \wedge z \in y)$. Dann liest sich $(Fund)$ wie folgt:

$(Fund)\ \forall x(x \neq \emptyset \to \exists y \in x\ y \cap x = \emptyset)$.

Seien $k, n \in \mathbb{N}$. Dann folgt aus $(\mathbb{N}; E) \models k \in n$, dass $k < n$: dies folgt einfach daraus, dass $2^k > k$ für alle $k \in \mathbb{N}$. Sei also $n \in \mathbb{N}$ so, dass $(\mathbb{N}; E) \models n \neq \emptyset$ d.h. $n \neq 0$!). Sei $k < n$ das kleinste k' mit $(\mathbb{N}; E) \models k \in n$. Dann gilt $(\mathbb{N}; E) \models k \cap n = \emptyset$. Wir haben $(\mathbb{N}; E) \models Fund$ gezeigt.

Wir schreiben $x = \{y, z\}$ für

$$y \in x \wedge z \in x \wedge \forall u \in x(u = y \vee u = z).$$

Das *Paarmengenaxiom* lautet

$(Paar)\ \forall x \forall y \exists z\ z = \{x, y\}$.

$(\mathbb{N}; E) \models Paar$ zeigt sich wie folgt.

Seien $n, m \in \mathbb{N}$. Sei

$$q = \begin{cases} 2^n + 2^m, \text{ falls } n \neq m \\ 2^n \qquad\quad , \text{ falls } n = m. \end{cases}$$

Dann gilt offensichtlich $(\mathbb{N}; E) \models q = \{n, m\}$.

Wir schreiben $x = \bigcup y$ für

$$\forall z(z \in x \leftrightarrow \exists u \in y\ z \in u).$$

Das *Vereinigungsaxiom* lautet

$(Ver)\ \forall x \exists y\ y = \bigcup x$.

Wir zeigen $(\mathbb{N}; E) \models Ver$ folgendermaßen. Sei $n \in \mathbb{N}$. Sei $n = \Sigma m_i \cdot 2^i$ die Dualdarstellung von n, und sei für $m_i = 1$ $i = \Sigma m^i_k \cdot 2^k$ die Dualdarstellung von i. Wir setzen dann

$$m = \sum_{\substack{m^i_k = i \\ \text{für ein } i \\ \text{mit } m_i = 1}} 2^k.$$

Offensichtlich gilt $(\mathbb{N}; E) \models m = \bigcup n$.

Wir schreiben $x \subset y$ für $\forall z \in x\ z \in y$ und $x = \mathcal{P}(y)$ für $\forall z(z \in x \leftrightarrow z \subset y)$. Das *Potenzmengenaxiom* besagt

$(Pot)\ \forall x \exists y\ y = \mathcal{P}(x)$.

$(\mathbb{N}; E) \models Pot$ zeigt man mit Hilfe derselben Methode, die auch $(\mathbb{N}; E) \models Ver$ zeigte. Sei $n \in \mathbb{N}$. Sei $n = \Sigma m_i \cdot 2^i$ die Dualdarstellung von n. Für ein $m \in \mathbb{N}$ mit Dualdarstellung $\Sigma m_i' \cdot 2^i$ gilt offenbar $(\mathbb{N}; E) \models m \subset n$ gdw. für alle $i, m_i' = 1 \Rightarrow m_i = 1$. Sei also I die (endliche!) Menge aller i mit $m_i = 1$ und sei P die Menge aller Teilmengen von I. Für $I^* \in P$ sei

$$n_{I^*} = \Sigma_{i \in I^*} 2^i.$$

(Offensichtlich ist $I^* \mapsto n_{I^*}$ injektiv.) Schließlich sei

$$m = \sum_{I^* \subset I} 2^{n_{I^*}}.$$

Es ist leicht zu sehen, dass $(\mathbb{N}; E) \models m = \mathcal{P}(n)S$.

Die Aussonderungsaxiome werden benötigt, um die Existenz definierbarer Teilmengen von einer gegebenen Menge zu zeigen.

Sei φ eine $\mathcal{L}_\in$–Formel, in der (o.B.d.A.) die Variablen $x, v_1, \ldots, v_p$ frei vorkommen. Das zu φ gehörige *Aussonderungsaxiom* lautet:

$(Aus_\varphi)\ \forall v_1 \ldots v_p \forall a \exists b \forall x (x \in b \leftrightarrow x \in a \wedge \varphi)$.

Die (unendliche!) Menge aller Aussonderungsaxiome wird auch als *Aussonderungsschema* bezeichnet. Man beweist leicht, dass $(\mathbb{N}; E)$ Modell des Aussonderungsschemas ist. Wir gehen einen Umweg.

Sei φ eine $\mathcal{L}_\in$–Formel, in der (o.B.d.A.) die Variablen $x, y, v_1, \ldots, v_p$ frei vorkommen. Das zu φ gehörige *Ersetzungsaxiom* lautet[1]

$(Ers_\varphi)\ \forall v_1 \ldots \forall v_p (\forall x \in a \forall y \forall y' (\varphi \wedge \varphi^y_{y'} \rightarrow y = y') \rightarrow \exists b \forall y (y \in b \leftrightarrow \exists x \in a \varphi))$.

Die (wiederum unendliche!) Menge aller Ersetzungsaxiome wird auch als *Ersetzungsschema* bezeichnet. Aus dem Ersetzungsschema zeigt sich das Aussonderungsschema sehr leicht wie folgt. Sei φ wie im zu φ gehörigen Aussonderungsaxiom gegeben. Setze $\psi \equiv \varphi \wedge x = y$. Dann gilt $Ers_\psi \vdash Aus_\varphi$.

Wir zeigen $(\mathbb{N}; E) \models Ers_\varphi$ für ein beliebiges φ wie folgt. Sei φ wie im zu φ gehörigen Ersetzungsaxiom gegeben. Seien $n_1, \ldots, n_p \in \mathbb{N}$ und sei $a \in \mathbb{N}$. Wir setzen voraus, dass

$$(\mathbb{N}; E) \models \forall x \in a \forall y \forall y' (\varphi(x, y, n_1, \ldots, n_p) \wedge \varphi(x, y', n_1, \ldots, n_p) \rightarrow y = y').$$

[1] o.B.d.A. komme y' in φ gar nicht vor.

Sei $(\mathbb{N}; E) \models k \in a$, d.h. für die Dualdarstellung $a = \Sigma m_i \cdot 2^i$ von a gilt $m_k = 1$. Falls ein l mit

$$(\mathbb{N}; E) \models \varphi(k, l, n_1, \ldots, n_p)$$

existiert, so sei $l(k)$ das eindeutige derartige l. Andernfalls sei $l(k)$ nicht definiert.

Setze dann

$$b = \sum_{\substack{(\mathbb{N}; E) \models k \in a \\ \text{und } l(k) \text{ ist} \\ \text{definiert.}}} 2^{l(k)}.$$

Es ist leicht zu sehen, dass

$$(\mathbb{N}; E) \models \forall y(y \in b \leftrightarrow \exists x \in a\varphi).$$

Aus dem Aussonderungsschema folgt die Existenz der leeren Menge, indem $\varphi \equiv x \neq x$ gewählt wird. Mit Hilfe des Paarmengen– und des Vereinigungsmengenaxioms zeigt sich dann leicht die Existenz der Mengen

$$\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}, \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\},$$

usw. Das *Auswahlaxiom* lautet

(AC) $\forall x(x \neq \emptyset \wedge \forall y \in x \forall y' \in x(y \cap y' \neq \emptyset \leftrightarrow y = y') \rightarrow (\exists z \forall y \in x \exists u z \cap y = \{u\})$.

AC besagt also, dass jede nichtleere Menge, die aus paarweise disjunkten Mengen besteht, eine "Auswahlmenge" besitzt. Wir zeigen $(\mathbb{N}; E) \models AC$ wie folgt.

Sei $n \in \mathbb{N}$. Sei $(\mathbb{N}; E) \models n \neq \emptyset$, d.h. für die Dualdarstellung $n = \Sigma m_i \cdot 2^i$ von n gilt, dass ein i mit $m_i = 1$ existiert. Für jedes i mit $m_i = 1$ sei $i = \Sigma m_k^i \cdot 2^k$ die Dualdarstellung von i. Unter der Voraussetzung

$$(\mathbb{N}; E) \models \forall y \in n \forall y' \in n(y \cap y' \neq \emptyset \leftrightarrow y = y')$$

gilt dann, dass aus $i \neq j$ mit $m_i = 1 = m_j$ folgt: $m_k^i \neq m_k^j$ für alle k. Außerdem ist für alle i mit $m_i = 1$ eines der m_k^i gleich 1. Sei für $m_i = 1$ $k(i)$ das kleinste k mit $m_k^i = 1$. Es ist dann leicht zu sehen, dass

$$\sum_{m_i = 1} 2^{k(i)}$$

bezeugt, dass im Sinne von $(\mathbb{N}; E)$ n das Auswahlaxiom erfüllt.

Wir bezeichnen die Menge der Aussagen Ext, $Fund$, $Paar$, Ver, Pot, Ers_φ (für beliebige φ) und AC als $ZFC^{-\infty}$. Hierbei steht ZFC für Zermelo-Fraenkel, C für "choice" (Auswahl) und "$-\infty$" für die Abwesenheit des Unendlichkeitsaxioms. Letzterem wollen wir uns nun zuwenden.

Wir schreiben $x = y \cup z$ für $x = \bigcup\{y, z\}$. Wir schreiben $y = x + 1$ für $y = x \cup \{x\}$.

Eine Menge x heißt *induktiv* gdw. $\emptyset \in x \wedge \forall y \in x \; y + 1 \in x$. Das *Unendlichkeitsaxiom* besagt:

(∞) $\exists x$ (x ist induktiv).

Das System ZFC (Zermelo–Fraenkel, mit Auswahlaxiom) entsteht aus $ZFC^{-\infty}$ durch Hinzunahme des Unendlichkeitsaxioms.

Sei $(\mathbb{N}; E) \models m = n + 1$, d.h. $(\mathbb{N}; E) \models m = n \cup \{n\}$. Dann gilt insbesondere $(\mathbb{N}; E) \models n \in m$, also ist $n < 2^n \leq m$. Damit kann $(\mathbb{N}; E)$ *nicht* das Unendlichkeitsaxiom erfüllen.