Kurt Goedel - Leben und Werk

Kurs auf der CdE-Winteraka 2023/2024

Merlin Carl*

4. Januar 2024

*Mitschrift von Maximilian Keßler

Inhaltsverzeichnis

1 Einleitung **3**

2 Semantik **3**

2.1 Ableitungsregeln: Tableau-Kalkül 4

3 Der Unvollständigkeitssatz **7**

3.1 Turing-Maschinen 8

3.2 Axiome der Arithmetik: Peano-Arithmetik 11

1 Einleitung

Steckbrief von Kurt Friedrich Gödel

- Geboren am 28.06.1906 (Brünn, Mähren (Tschechien, damals Österreich-Ungarn))
- Vater Rudolf Gödel (Manager einer Textilfabrik), Mutter Marianne Gödel (geb. Handschuh), Bruder ? Gödel)
- Ab 1924: Studium in Wien (Physik, Mathematik, Philosophie)
- 1929: Österreichische Staatsbürgerschaft;
  Promotion (Beweis des Vollständigkeitssatzes)
- 1930: Unvollständigkeitssätze
- 1935: Relative Konsistenz des Auswahlaxioms
  Heirat mit Adele Nimbursky (geb. Porkert)
- 1937: Relative Konsistenz der Kontinuumshypothese
- 1940: Emigration in die USA (Princeton), Mitglied des IAS
  Arbeit zu „Geschlossene zeitartige Kurven“
- 1941: Ontologischer Gottesbeweise
- 1948: Amerikanische Staatsbürgerschaft
- 1951: Albert Einstein Award; 1957 National Medal of Science etc.
- 1953: Professur am IAS
- 1956: Emeritierung
- Diverse philosophische Aufsätze, ontologischer Gottesbeweis, Überlegungen zur Kontinuumshypothese
- Gestorben 14.01.1978 (Princeton)

2 Semantik

Definition 1 (Sprache)**.** Eine **Sprache** $\mathcal{S}$ besteht aus

1. Prädikatensymbolen P, Q, R, alle mit fester Arität (auf wie viele Terme man das Prädikat anwendet)
2. Funktionszeichen f, g, h, ebenfalls jeweils mit Arität.

3. Konstantenzeichen a, b, c, ...

Definition 2 (S-Struktur)**.** Eine **S**-Struktur für eine Sprache $\mathcal{S}$ besteht aus

- einer Domäne (Menge) M
- Für jedes Prädikatenzeichen von $\mathcal{S}$ eine Relation über M
- Für jedes Funktionszeichen und jede Konstante von $\mathcal{S}$ eine Funktion bzw. ein Element aus M.

Definition 3. Sei $\mathcal{S}$ eine Sprache und $\mathcal{A}$ ein Axiomensystem, d.h. eine Menge an Formeln über $\mathcal{S}$. Für eine Formel φ schreiben wir $\mathcal{A} \models \varphi$, falls aus $\mathcal{A}$ **semantisch** die Formel φ folgt, d.h. falls für alle $\mathcal{S}$-Strukturen, in denen $\mathcal{A}$ wahr ist, auch φ wahr ist.

2.1 Ableitungsregeln: Tableau-Kalkül

Definition 4 (Tableu-Kalkül)**.** Eine **Herleitung** für eine Formel φ im **Tableau-Kalkül** besteht immer daraus, mit der Formel $\neg\varphi$ anzufangen und dann durch Anwendung von Schlussregeln einen Baum zu erzeugen. Ein Ast des Baumes gilt als **abgeschlossen**, wenn auf ihm sowohl ψ als auch $\neg\psi$ stehen, sich also ein Widerspruch ergeben hat. Eine Herleitung von φ ist es, alle Äste des Baumes, an dessen Wurzel $\neg\varphi$ steht, zu schließen.

Folgende formale Regeln erlauben wir für logische Schlussfolgerungen:

- Steht $\neg\neg\varphi$ auf einem Ast, füge φ hinzu.
- Steht $\alpha_1 \wedge \alpha_2$ auf einem Ast, füge α_1 und α_2 hinzu
- Steht $\alpha_1 \vee \alpha_2$ auf einem Ast, spalte in einen Ast mit α_1 und einen mit α_2
- Steht $\alpha_1 \to \alpha_2$ auf einem Ast, spalte in $\neg\alpha_1$ und α_2
- Ergänze $\neg(\varphi \to \psi)$ durch φ und $\neg\psi$
- Ergänze $\neg(\varphi\varphi \vee \psi)$ durch $\neg\varphi$ und $\neg\psi$
- Spalte $\neg(\varphi \wedge \psi)$ in $\neg\varphi$ und $\neg\psi$

Folgende Regeln sind für Quantoren erlaubt:

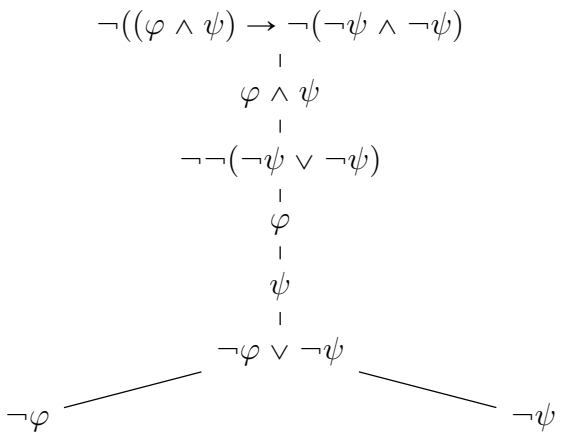
- Ergänze $\exists x\varphi$ durch $\varphi^{[x\leftarrow a]}$ für ein neues Konstantenzeichen a. Hierbei steht $\varphi^{[x\leftarrow a]}$ für die Formel, in der wir x durch a ersetzen
- Ergänze $\neg\forall x\varphi$ durch $\neg\varphi^{[x\leftarrow a]}$ für ein neues Konstantenzeichen a
- Ergänze $\forall x\varphi$ durch $\varphi[x \leftarrow l]$ für einen Term t
- Ergänze $\neg\exists x\varphi$ durch $\neg\varphi^{[x\leftarrow t]}$ für einen Term t

Remark 4.1. Man sollte sich den entstehenden Baum so vorstellen, dass man auf jedem Ast wissen sammelt. Jede neue Zeile fügt neues Wissen hinzu, man kann also jederzeit Wissen des eigenen Astes verwenden. Wenn wir einen Ast in 2 neue Aufteilen, entspricht das einer üblichen Fallunterscheidung.

Wenn wir alle Äste schließen, haben wir alle Fälle zum Widerspruch geführt.

Exercise. *Man mache sich für alle obigen formalen Schlussfolgerungen Gedanken, was diese* intuitiv *sagen. Das heißt, welches Wissen wird hier in welches neue Wissen verwandelt, und warum macht das Sinn, bzw. warum würden wir erwarten, dass dies eine sinnvolle logische Schlussregel ist.*

Example 5 (Logische Folgerungen)**.** Wir wollen die Aussage $(\varphi \wedge \psi) \rightarrow \neg(\neg\varphi \wedge \neg\psi)$ zeigen.



Example 6 (Quantoren)**.** Sei $L(x, y)$ die Aussage „x liebt y“. Wir wollen zeigen: $\exists x\forall y L(x, y) \rightarrow \forall y\exists x L(x, y)$. In gesprochener Sprache: Wenn es eine Person (x) gibt, die alle liebt, dann wird jede Person von jemandem geliebt.

Eine Herleitung dieser Formel ist die Folgende:

$$\begin{gathered}
\neg(\exists x \forall y\ L(x,y) \to \forall y \exists x\ L(x,y)) \\
\exists x \forall y\ L(x,y) \\
\neg \forall y \exists x\ L(x,y) \\
\forall y\ L(a,y) \\
\neg \exists x\ L(x,b) \\
L(a,b) \\
\neg L(a,b)
\end{gathered}$$

Notation 6.1. Für eine Sprache $\mathcal{S}$ und ein Axiomensystem $\mathcal{A}$ schreiben wir $A \vdash \varphi$, falls sich aus A und $\neg\varphi$ eine geschlossenes Tableau herleiten lässt.

Definition 7 (Korrektheit)**.** Wir sagen, dass eine Kombination aus Sprache und Schlussregeln **korrekt** ist, wenn alle Formeln, die sich herleiten lassen (die also syntaktisch folgerbar sind), auch sematisch folgebar sind.

Kompakter geschrieben sprechen wir von Korrektheit, wenn aus $\mathcal{A} \vdash \varphi$ auch stets $\mathcal{A} \models \varphi$ folgt.

Remark 7.1. Das zu beweisen ist letztendlich nicht schwer. Man muss sich „nur“ davon überzeugen, dass jede einzelne der syntaktischen Schlussregeln so gestaltet ist, dass sich auch in S-Strukturen die entsprechenden Schritte durchführen lassen.

Definition 8 (Vollständigkeit)**.** Ähnlich sprechen wir von **Vollständigkeit**, wenn jede Formel, die sich semantisch herleiten lässt, auch syntaktisch herleiten lässt, d.h. wenn aus $A \models \varphi$ auch schon $A \vdash \varphi$ folgt.

Theorem 9 (Gödel'scher Vollständigkeitssatz)**.** Das Tableau-Kalkül ist korrekt und vollständig für jede Sprache $\mathcal{S}$.

Remark 9.1. Vollständigkeit zu beweisen ist weitaus schwieriger: Wir müssen über *alle* $\mathcal{S}$-Strukturen etwas aussagen, haben aber nur eine fixe Menge an syntaktischen Regeln zur Verfügung.

Corollary 10. Ist jede *endliche* Teilmenge eines Axiomensystems konsistent, so ist auch das Axiomensystem selbst konsistent.

Beweis. Sei $\mathcal{A}$ dieses System und nimm an, dass man $\mathcal{A}$ nicht interpretieren kann, ich also keine $\mathcal{S}$-Struktur finde, in der $\mathcal{A}$ gilt. Dann gilt insbesondere $\mathcal{A} \models \bot$, wobei $\bot$ irgendeine falsche Aussage sei, denn in jeder $\mathcal{S}$-Struktur, in der $\mathcal{A}$ gilt (es gibt keine solchen), gilt ja auch $\bot$. Nach dem Vollständigkeitssatz gibt es dann auch schon eine Herleitung $\mathcal{A} \vdash \bot$. Eine Herleitung besteht allerdings aus endlich vielen Schritten, also gibt es auch schon eine endliche Teilmenge $\mathcal{B} \subseteq \mathcal{A}$, sodass $\mathcal{B} \vdash \bot$. Nach Korrektheit ist dann auch $\mathcal{B} \models \bot$, also ist $\mathcal{B}$ inkonsistent, ein Widerspruch zu unserer Annahme. □

3 Der Unvollständigkeitssatz

Wir wollen nun verstehen, wie man einfache Arithmetik (d.h. Rechnen mit natürlichen Zahlen) in Prädikatenlogik formulieren kann.

Definition 11. Sei $\mathcal{L}_A$ die Sprache mit Relationszeichen $=$, $<$, Funktionszeichen $+$, $\cdot$, sowie Konstantenzeichen 0, 1.

Example 12. Wir können zum Beispiel folgende Übersetzungen treffen:

$$\begin{aligned} a \text{ ist ein Teiler von } b &\leftrightarrow \exists x : a \cdot x = b \\ p \text{ ist keine Primzahl} &\leftrightarrow \exists a \exists b \colon (a \neq 1 \wedge b \neq 1 \wedge a \cdot b = p) \\ p \text{ ist eine Primzahl} &\leftrightarrow \neg(\exists a \exists b \colon a \neq 1 \wedge b \neq 1 \wedge a \cdot b = p) \\ &\leftrightarrow \forall a \forall b \colon a \cdot b = p \to (a = 1 \vee b = 1) \\ \text{Die Zahl „Zwei“} &\leftrightarrow 1 + 1 \end{aligned}$$

Um uns Notation zu vereinfachen, führen wir $a \mid b$ sowie $\text{prim}(p)$ mit den obigen Bedeutungen ein, ebenfalls können wir jede *feste* natürliche Zahle als $1 + 1 + \ldots b + 1$ kodieren. Damit meinen wir nicht, dass wir die Sprache tatsächlich erweitern (um solche Relationssymbole), sondern einfach, dass man jede Vorkommnis von diesen Relationen rein formal durch ihre obige Langschreibweise ersetzen kann.

Jetzt lässt sich die (starke) Goldbach'sche Vermutung formulieren als

$$\forall z((z > 2 \wedge 2 \mid z) \to \exists a, b(z = a + b \wedge \text{prim}(a) \wedge \text{prim}(b))).$$

Etwas schwieriger wird es, zu übersetzen, dass es unendlich viele Primzahlen

gibt:

$$\forall x \exists p(x < p \wedge \text{prim}(p)).$$

Eigentlich steht hier eher etwas wie „Es gibt beliebig große Primzahlen“, allerdings ist das (in den natürlichen Zahlen) das gleiche Konzept.

Definition 13. Ein **Axiomensystem** ist eine Menge von Sätzen (Formeln).

Question 13.1. Was sind wünschenswerte Eigenschaften eines Axiomensystems?

- **Widerspruchsfreiheit**, d.h. es gibt keine Aussage φ, für die sowohl φ als auch $\neg\varphi$ bewiesen werden können
- **Vollständigkeit**, d.h. für jeden Satz φ der Sprache gilt $A \vdash \varphi$ oder $A \vdash \neg\varphi$.

Remark 13.2. Dieser Begriff der *Vollständigkeit* ist ein anderer, wie wir ihn bereits gesehen haben.

Ein Kalkül kann vollständig sein, wenn $A \models \varphi \iff A \vdash \varphi$, Herleitungen also vollständig die semantische wahren Aussagen widerspiegeln.

Ein Axiomensystem kann vollständig sein, indem $A \vdash \varphi \vee A \vdash \neg\varphi$ für jede Formel φ.

Remark 13.3. Beide Eigenschaften für sich sind natürlich einzeln leicht realisierbar: Das leere Axiomensystem ist natürliche widerspruchsfrei, denn man kann gar nichts ableiten, und widersprüchliche Axiomensysteme sind vollständig, weil man jede Aussage zeigen kann.

Es gibt auch widerspruchsfreie und vollständige Systeme, allerdings können diese nicht über Arithmetik reden. Das wird Inhalt von Gödels Unvollständigkeitssatz sein.

3.1 Turing-Maschinen

Turing-Maschinen sind ein theoretisches Computermodell, das von Alan Turing eingeführt wurde.

Definition 14. Eine **Turingmaschine** besteht aus einem (nach rechts hin) unendlichen Speicherband, auf dem an jeder Position 0 oder 1 steht, einem Schreib-/Lesekopf, der sich stets an einer Position findet.

Das Programm einer Turingmaschine besteht aus einer Liste an Zuständen,

wobei jeder Zustand in Abhängigkeit vom Wert des Bands am Lesekopf angibt:

- welchen Wert der Schreibkopf auf die aktuelle Position schreiben soll
- in welche Richtung (links/rechts/stehen bleiben) sich der Schreibkopf als nächstes bewegt
- was der nächste Programmzustand der Turingmaschine ist

Das Programm lässt sich also Beschreiben über eine Zustandsmenge $\mathcal{S}$ und eine Funktion

$$\mathcal{S} \times \{0,1\} \to \{0,1\} \times \{\text{links}, \text{rechts}, \text{stehen bleiben}\} \times (\mathcal{S} \cup \{\text{end}\}),$$

wobei end beschreibt, dass das Programm terminiert hat. Eine Turing-Maschine lässt sich ausgehend von einem Zustands des Speicherbands und einem Zustand $s \in \mathcal{S}$ sukzessive ausführen.

Man könnte meinen, dass Turing-Maschinen nicht sehr viel können, allerdings gilt:

Proposition 15 (Church-Turing-These)**.** Jede Funktion auf endlichen Zeichenfolgen, die überhaupt berechenbar ist, ist durch eine Turing-Maschine berechenbar.

Remark 15.1. Der Begriff „überhaupt berechenbar“ ist hierbei natürlich mathematisch unpräzise, und dementsprechend gibt es auch keinen formalen Beweis. Wir meinen hiermit Funktionen, die ein idealisierter Mensch mit hinreichend viel Zeit durch Ausführung eines Schemas berechnen kann.

Zum Beispiel lässt sich jede moderne Programmiersprache systematisch in eine Turingmaschine übersetzen und es ist bis dato keine Funktion bekannt, welche *intuitiv berechenbar* ist, allerdings nicht von einer Turingmaschine.

Die Church-Turing-These motiviert die folgende Definition:

Definition 16 (Berechenbarkeit)**.** Eine Funktion heißt **berechenbar**, wenn eine Turing-Maschine gibt, die sie berechnet, d.h. eine Turing-Maschine, die bei Kodierung der Eingabe auf dem Band und Ausführung der Maschine stets nach endlich vielen Schritten terminiert und eine korrekte Ausgabe auf das Band schreibt.

Corollary 17. Es gibt ein Turing-Programm, das von einer Zeichenfolge S feststellt, ob sie ein korrekter Tableau-Beweis für eine Aussage φ aus den Axiomen $\mathcal{A}$ ist.

Wir wollen nun ansehen, wie Gödel zu seinem Unvollständigkeitssatz kam:

Schritt 1: Axiomatisierung Zu einem gegebenen Turingprogramm P funde eine arithmetische Formel φ_P so, dass $\varphi_P(a, b)$ wahr ist genau dann, wenn P bei Eingabe a Ausgabe b liefert.

Die Zustände seien mit $\{S_1, \ldots, S_n\}$ bezeichnet, wir können S_i durch die natürliche Zahl i kodieren. Die Kopfposition ist ebenfalls eine natürliche Zahl, genauso wie der Bandinhalt. Die gesamte Konfiguration einer Turing-Maschine, die gerade ein Programm ausführt, ist also durch ein Tripel natürlicher Zahlen beschreibbar.

Die Formel $\varphi^P_{\text{suc}}(z_0, b_0, k_0, z_1, b_1, k_1)$ soll nun beschreiben, dass bei Ausführung des States (z_0, b_0, k_0) sich im nächsten Schritt (z_1, b_1, k_1) ergibt.

Die Formel $\text{digit}(z, i, j)$ ist erfüllt genau dann, wenn j die i-te Ziffer von z ist (wobei i-te Ziffer einer Zahl von hinten gezählt wird, die 1-te Ziffer ist also die Einerziffer).

Dann können wir zum Beispiel die Regel

$$(s_1, 0) \mapsto (s_2, 0, \text{rechts})$$

durch die Formel

$$(z_0 = 1 \wedge \text{digit}(b_0, k_0, 0)) \rightarrow (z_1 = 2 \wedge k_1 = k_0 + 1 \wedge b_1 = b_0)$$

ausdrücken. Um Zustandsänderungen des Bandes auszudrücken, benötigen wir noch eine Formel $\text{digitChange}(z_0, i, z_1)$, die ausdrückt, dass sich z_0 und z_1 genau in der i-ten Ziffer unterscheiden. Die Regel

$$(s_1, 1) \mapsto (s_0, 0, \text{links})$$

können wir dann als

$$(z_0 = 1 \wedge \text{digit}(b_0, k_0, 1) \rightarrow (z_1 = 9 \wedge k_1 = k_0 - 1 \wedge \text{digitChange}(b_0, k_0, b_1))$$

realisieren.

Das Prädikat φ^P_{suc} lässt sich nun als großes „Und“ all dieser Zustandsimplikationen.

Wir wollen nun noch Zustandsfolgen betrachten, um über den gesamten Berechnungsverlauf der Turingmaschine reden zu können. Hierzu kodieren wir Listen $(x_1, x_2, \ldots,)$ von natürlichen Zahlen als das Produkt $2^{x_1} \cdot 3^{x_2} \cdot \cdots \cdot p_k^{x_k}$, wobei p_k die k-te Primzahl sein.

Jetzt können wir die ursprüngliche Formel $\varphi_P(a, b)$ ausdrücken als

- Es gibt eine Zahl $m = p_1^{x_1} \cdot p_2^{x_2} \ldots p_n^{x_n}$, sodas

- x_1 kodiert die Konfiguration $(1, 1, a)$.
- x_n kodiert die Konfiguration der Form $(-, -, b)$.
- Für alle $2 \leqslant i \leqslant n$ gilt: $\varphi^P_{\text{suc}}(x_{i-1}, x_i)$.

Damit haben wir also Berechenbarkeit und somit auch Beweisbarkeit formalisiert.

Corollary 18. Es existiert eine arithmetische Formel $\text{bew}(a, b)$, die genau dann wahr ist, wenn a einen Tableaubeweis für die von b kodierte Formel kodiert.

Goal. *Konstruiere einen arithmetischen Satz φ, für den folgendes beweisbar ist:*

$$\varphi \leftrightarrow \neg\exists a\, \text{bew}(a, \lceil\varphi\rceil).$$

Es wird noch etwas dauern, dass wir genau solch ein φ konstruieren können, weil wir noch besser $\lceil\varphi\rceil$ verstehen müssen, um es in φ selbst „einzubetten".

3.2 Axiome der Arithmetik: Peano-Arithmetik

Die **Peano-Arithmetik** beschreibt (in Prädikatenlogik erster Stufe) ein Axiomensystem zur Formalisierung der Arithmetik der natürlichen Zahlen.

Folgendes sind die Axiome von PA^-:

$$\begin{aligned}
a + 0 &= a && (1)\\
a + b &= b + a && (2)\\
a + (b + c) &= (a + b) + c && (3)\\
1 \cdot a &= a && (4)\\
a \cdot b &= b \cdot a && (5)\\
a \cdot (b \cdot c) &= (a \cdot b) \cdot c && (6)\\
a \cdot (b + c) &= a \cdot b + a \cdot c && (7)\\
0 &< 1 && (8)\\
a < b &\to a + c < b + c && (9)\\
(a < b \wedge c > 0) &\to c \cdot a < c \cdot b && (10)\\
a < b \mathbin{\dot\vee} b < a &\mathbin{\dot\vee} a = b && (11)\\
a < b \wedge b < c &\to a < c && (12)
\end{aligned}$$

Hierbei ist $\dot\vee$ ein „exclusives oder", d.h. *genau* einer der beiden soll wahr sein.

In der Peano-Arithmetik fordern wir zusätzlich noch, dass **Induktion** gilt, d.h. für jede Formel $\varphi(x)$ gilt das **Induktionsaxiom**

$$\text{Ind}_\varphi := [\varphi(0) \wedge \forall n\, (\varphi(n) \to \varphi(n + 1))] \to \forall n\, \varphi(n). \qquad (13)$$

Notation 18.1. Wir führen das Prädikat $\text{beweisbar}(\lceil\varphi\rceil) := \exists a\, \text{bew}(a, \lceil\varphi\rceil)$ ein.

Lemma 19 (Gödel'sches Diagonallemma)**.** Sei $\psi(x)$ eine arithmetische Formel mit einer freien Variable x. Dann existiert eine Formel φ, sodass

$$\text{PA} \vdash \varphi \leftrightarrow \psi(\lceil\varphi\rceil).$$

Etwas salopp könnte man sagen, dass die Aussage von φ ist, dass φ die Eigenschaft ψ besitzt.

Beweis. $\text{Subst}(\lceil\varphi\rceil, a, \lceil\chi\rceil)$ bedeute: Wenn man für alle freien Variablen von φ die Zahl a einsetzt, so erhält man χ. Wegen der Church-Turing These und unseren vorherigen Überlegungen gibt es ein solches Prädikat in PA.

Betrachte nun die Formel

$$\exists n(\text{Subst}(\lceil\varphi\rceil, \lceil\varphi\rceil, n) \wedge \psi(n)).$$

Man könnte diese verbalisieren als „ψ trifft auf die Selbsteinsetzung von φ zu“. Diese Formel hat selbst eine freie Variable und eine Gödelnummer g. Wir können also folgende Aussage betrachten:

$$G := \exists n(\text{Subst}(g, g, n) \wedge \psi(n)).$$

Die Aussage G ist nun, was wir suchen:

Angenommen, G ist wahr, dann gibt es ein n, sodass $\text{Subst}(g, g, n) \wedge \psi(n)$ gilt. Nach der Definition von Subst ist dann n die Gödelnummer von G (denn G ist gerade dadurch entstanden, in die Formel mit Nummer g die Zahl g für alle freien Variablen einzusetzen). Es gilt also $\psi(n) = \psi(\lceil G\rceil)$. Das zeigt $G \to \psi(\lceil G\rceil)$.

Angenommen, $\psi(\lceil G\rceil)$ ist wahr: Dann gibt es ein n, nämlich $\lceil G\rceil$, sodass $\text{Subst}(g, g, n) \wedge \psi(n)$ erfüllt ist, also haben wir G gefolgert.

Es gilt also $G \leftrightarrow \psi(\lceil G\rceil)$ wie gewünscht. □

Jetzt können wir das Diagonallemma auf die Aussage

$$\psi(\lceil\varphi\rceil) := \neg\exists a\, \text{bew}(a, \lceil\varphi\rceil) = \neg\, \text{beweisbar}(\lceil\varphi\rceil)$$

anwenden und erhalten die gewünschte Formel φ, sodass

$$\text{PA} \vdash \varphi \leftrightarrow \neg\, \text{beweisbar}(\lceil\varphi\rceil)$$

gilt. Das beweist den Unvollständigkeitssatz.

Remark 19.1. In unserer aktuellen Formulierung müssen wir als Prämisse für den Unvollständigkeitssatz noch verwenden, dass das System keine falschen Aussagen beweist. Unser Ziel ist es, dies darauf zu reduzieren, dass das System „nur“ widerspruchsfrei ist.

Wähle nun die Aussage

$$\psi(\lceil\varphi\rceil) \coloneqq \forall a\,[\mathrm{bew}(a, \lceil\varphi\rceil) \to \exists b(b < a \wedge \mathrm{bew}(b, \lceil\neg\varphi\rceil))]$$

und wende wieder das Diagonallemma an.

Falls $\mathrm{PA} \vdash \varphi$, existiert ein a mit $\mathrm{bew}(a, \lceil\varphi\rceil)$, dann aber auch ein b mit $\mathrm{bew}(b, \lceil\neg\varphi\rceil)$ und somit $\mathrm{PA} \vdash \neg\varphi$, ein Widerspruch.

Angenommen, $\mathrm{PA} \vdash \neg\varphi$, also

$$\mathrm{PA} \vdash \neg\forall a\,[\mathrm{bew}(a, \lceil\varphi\rceil) \to \exists b(b < a \wedge \mathrm{bew}(b, \lceil\neg\varphi\rceil))]\,.$$

Also gibt es ein a mit $\mathrm{bew}(a, \lceil\varphi\rceil)$ und somit $\mathrm{PA} \vdash \varphi$, ein Widerspruch.

Es folgt also:

Kein Axiomensystem, das PA erweitert und widerspruchsfrei ist, ist vollständig

Jetzt stellen wir aber auch fest, dass

$$\mathrm{PA} \vdash \mathrm{Con}(\mathrm{PA}) \to \varphi,$$

wobei Con(PA) dafür steht, dass PA konsistent ist. Also

$$\neg(\mathrm{PA} \vdash \mathrm{Con}(\mathrm{PA})).$$