Mathematische Logik, Grundlagen der Mathematik

Helmut Schwichtenberg
Mathematisches Institut, Ludwig-Maximilians-Universität
München

Vorlesungen

Mathematische Logik I, II

Beweistheorie
Rekursionstheorie
Mengenlehre
Modelltheorie

Wichtig für Diplom- (Staatsexamens-, Master-) Arbeit:
Übungen, Seminar

1. Formale Sprachen

Hier: über natürliche Zahlen.

Variablen:	x, y, z
Funktionssymbole:	$+$, $*$, S, 0
Terme:	$x \mid 0 \mid r + s \mid r * s \mid S(r)$

Ziffern sind spezielle Terme: für $a \in \mathbb{N}$ sei $\underline{a}$ definiert durch

$$\underline{0} := 0, \quad \underline{n+1} := S(\underline{n}).$$

Formeln: $r = s \mid A \wedge B \mid A \vee B \mid A \to B \mid \neg A \mid \forall x A \mid \exists x A$

Geschlossene Formel (Satz): Formel ohne freie Variable

Beispiele

$$
\begin{array}{lcl}
x < y & := & \exists z(z \neq 0 \wedge x + z = y) \\
y \mid x & := & \exists z(y * z = x) \\
x \text{ Primzahl} & := & 1 < x \wedge \forall y(y \mid x \rightarrow y = 1 \vee y = x)
\end{array}
$$

Es gibt unendlich viele Primzahlen:

$$\forall x \exists y(x < y \wedge y \text{ Primzahl})$$

Semantik

Sei $\mathcal{M} = (|\mathcal{M}|, 0^{\mathcal{M}}, S^{\mathcal{M}})$ zur Sprache passende Struktur.
Hier: $|\mathcal{M}| = \mathbb{N}, \quad 0^{\mathcal{M}} = 0, \quad S^{\mathcal{M}}(a) = a + 1.$

Wahrheitsbegriff von $\mathcal{M}$:

$$\mathsf{Th}(\mathcal{M}) := \{\, A \mid A \text{ geschlossene Formel mit } \mathcal{M} \models A \,\}$$

$R \subseteq \mathbb{N}$ definierbar:

$$\text{es gibt } A_R[z] \text{ mit } R = \{\, a \in \mathbb{N} \mid \mathcal{M} \models A_R[\underline{a}] \,\}$$

$R \subseteq \mathbb{N}^k$ definierbar: analog

Undefinierbarkeit des Wahrheitsbegriffs

Numerierung der Formeln: $A \mapsto \ulcorner A \urcorner$

Satz. (Tarski)

$$\ulcorner \mathsf{Th}(\mathcal{M}) \urcorner := \{ \ulcorner A \urcorner \mid A \text{ geschlossene Formel mit } \mathcal{M} \models A \}$$

ist nicht definierbar.

Fixpunktlemma.
Zu $B[z]$ findet man eine geschlossene Formel A mit

$$\mathcal{M} \models A \quad \text{gdw} \quad \mathcal{M} \models B[\ulcorner A \urcorner].$$

Beweis des Fixpunktlemmas

Zu $B[z]$ findet man eine geschlossene Formel A mit

$$\mathcal{M} \models A \quad \text{gdw} \quad \mathcal{M} \models B[\ulcorner A \urcorner].$$

Beweis. Bei geeig. Numerierung findet man $A_s[x_1, x_2, x_3]$:

$$\mathcal{M} \models \forall x \big(A_s[\ulcorner C \urcorner, k, x] \leftrightarrow x = \ulcorner C[\underline{k}] \urcorner \big)$$

Setze $C := \exists x \big(B[x] \wedge A_s[z, z, x] \big)$ und $A := C[\underline{\ulcorner C \urcorner}]$, also

$$A = \exists x \big(B[x] \wedge A_s[\underline{\ulcorner C \urcorner}, \underline{\ulcorner C \urcorner}, x] \big).$$

$$\begin{aligned} \mathcal{M} \models A \quad &\text{gdw} \quad \exists a \in \mathbb{N} \big(\mathcal{M} \models B[a] \text{ und } a = \ulcorner C[\underline{\ulcorner C \urcorner}] \urcorner \big) \\ &\text{gdw} \quad \mathcal{M} \models B[\ulcorner A \urcorner]. \end{aligned}$$

Beweis des Undefinierbarkeitssatzes

Annahme: $\ulcorner \mathsf{Th}(\mathcal{M}) \urcorner$ ist definierbar, etwa durch $B_W[z]$.
Dann gilt für alle geschlossenen A

$$\mathcal{M} \models A \quad \text{gdw} \quad \mathcal{M} \models B_W[\ulcorner A \urcorner].$$

Betrachte die Formel $\neg B_W[z]$. Nach dem Fixpunktlemma hat man eine geschlossene Formel A mit

$$\mathcal{M} \models A \quad \text{gdw} \quad \mathcal{M} \models \neg B_W[\ulcorner A \urcorner].$$

Widerspruch.

Entscheidbarkeit, Aufzählbarkeit

$M \subseteq \mathbb{N}$ entscheidbar: es gibt einen Algorithmus, der bei Eingabe von a terminiert und feststellt, ob $a \in \mathbb{N}$ oder nicht.

Leicht: M entscheidbar $\Rightarrow$ M definierbar

Folgerung. $\ulcorner \mathsf{Th}(\mathcal{M}) \urcorner$ ist nicht entscheidbar.

$M \subseteq \mathbb{N}$ aufzählbar: es gibt einen Algorithmus, der bei Eingabe von a genau dann terminiert, wenn $a \in \mathbb{N}$.

Leicht: M aufzählbar $\Rightarrow$ M definierbar

Folgerung. $\ulcorner \mathsf{Th}(\mathcal{M}) \urcorner$ ist nicht aufzählbar.

2. Formales Schließen

Wahrheit $\mapsto$ Ableitbarkeit in einer formalen Theorie T.

Axiome: z.B. $A[0] \wedge \forall x(A[x] \rightarrow A[S(x)]) \rightarrow \forall x A[x]$
Schlußregeln: z.B. modus ponens.

Annahmen über T:

T ist axiomatisiert, d.h. $\mathsf{Bew}_T(n, m)$ ist entscheidbar. (1)

T ist widerspruchsfrei. (2)

T beweist die Axiome von Robinsons Q. (3)

Ziel: T is unvollständig.

Robinsons Q

$$S(x) \neq 0,$$

$$S(x) = S(y) \rightarrow x = y,$$

$$x + 0 = x,$$

$$x + S(y) = S(x + y),$$

$$x \cdot 0 = 0,$$

$$x \cdot S(y) = x \cdot y + x,$$

$$\exists z\,(x + S(z) = y) \vee x = y \vee \exists z\,(y + S(z) = x).$$

Unvollständigkeit

Satz (Gödel, Rosser). Man findet eine geschlossene Formel A mit $\nvdash_T A$ und $\nvdash_T \neg A$.

Beweis. Hilfsmittel: Jede entscheidbare Relation R ist in T repräsentierbar durch $B_R[\vec{x}]$.

Syntaktisches Fixpunktlemma: Zu $B[z]$ findet man eine geschlossene Formel A mit

$$\vdash_T A \leftrightarrow B[\ulcorner A \urcorner].$$

$\mathsf{Bew}_T(n, m)$ entscheidbar $\Rightarrow$ $\mathsf{Wdl}_T(n, m)$ entscheidbar.

Beweis des Unvollständigkeitssatzes

$$T \vdash \forall x\big(x < \underline{n} \to x = \underline{0} \vee \cdots \vee x = \underline{n-1}\big), \tag{4}$$

$$T \vdash \forall x\big(x = \underline{0} \vee \cdots \vee x = \underline{n} \vee \underline{n} < x\big). \tag{5}$$

Seien $B_{\mathsf{Bew}_T}[x_1, x_2]$ und $B_{\mathsf{Wdl}_T}[x_1, x_2]$ repräsentierende Formeln zu Bew_T und Wdl_T. Fixpunktlemma: Satz A mit

$$T \vdash A \leftrightarrow \forall x\big(B_{\mathsf{Bew}_T}[x, \underline{\ulcorner A \urcorner}] \to \exists y(y < x \wedge B_{\mathsf{Wdl}_T}[y, \underline{\ulcorner A \urcorner}])\big).$$

A besagt die eigene Unbeweisbarkeit: „Zu jedem Beweis von mir gibt es einen kürzeren Beweis meiner Negation“.

Wir zeigen $(*)$ $T \nvdash A$ und $(**)$ $T \nvdash \neg A$.

Beweis des Unvollständigkeitssatzes (Forts.)

Zu $(*)$. Gelte $T \vdash A$. Wähle n mit

$$\mathsf{Bew}_T(n, \ulcorner A \urcorner).$$

Dann gilt auch

$$\text{not } \mathsf{Wdl}_T(m, \ulcorner A \urcorner) \qquad \text{für alle } m,$$

da T widerspruchsfrei ist. Es folgt (Repräsentierbarkeit)

$$T \vdash B_{\mathsf{Bew}_T}[\underline{n}, \underline{\ulcorner A \urcorner}],$$
$$T \vdash \neg B_{\mathsf{Wdl}_T}[\underline{m}, \underline{\ulcorner A \urcorner}] \quad \text{für alle } m.$$

Beweis des Unvollständigkeitssatzes (Forts.)

Wegen (4) folgt

$$T \vdash B_{\mathsf{Bew}_T}[\underline{n}, \underline{\ulcorner A \urcorner}] \wedge \forall y \big(y < \underline{n} \rightarrow \neg B_{\mathsf{Wdl}_T}[y, \underline{\ulcorner A \urcorner}] \big).$$

Also

$$T \vdash \exists x \big(B_{\mathsf{Bew}_T}[x, \underline{\ulcorner A \urcorner}] \wedge \forall y (y < x \rightarrow \neg B_{\mathsf{Wdl}_T}[y, \underline{\ulcorner A \urcorner}]) \big),$$
$$T \vdash \neg A.$$

Dies widerspricht der angenommenen Widerspruchsfreiheit von T.

Beweis des Unvollständigkeitssatzes (Forts.)

Zu $(**)$. Gelte $T \vdash \neg A$. Wähle n mit

$$\mathsf{Wdl}_T(n, \ulcorner A \urcorner).$$

Dann gilt auch

$$\text{nicht } \mathsf{Bew}_T(m, \ulcorner A \urcorner) \quad \text{für alle } m,$$

da T widerspruchsfrei ist. Es folgt (Repräsentierbarkeit)

$$T \vdash B_{\mathsf{Wdl}_T}[\underline{n}, \underline{\ulcorner A \urcorner}],$$
$$T \vdash \neg B_{\mathsf{Bew}_T}[\underline{m}, \underline{\ulcorner A \urcorner}] \quad \text{für alle } m.$$

Beweis des Unvollständigkeitssatzes (Forts.)

Mit (5) folgt

$$T \vdash \forall x \big(B_{\mathsf{Bew}_T}[x, \underline{\ulcorner A \urcorner}] \to \exists y (y < x \wedge B_{\mathsf{Wdl}_T}[y, \underline{\ulcorner A \urcorner}]) \big),$$

durch Fallunterscheidung nach x, also

$$T \vdash A.$$

Dies widerspricht der angenommenen Widerspruchsfreiheit von T.

3. Konstruktive Mathematik

Hermann Weyl (Über die neue Grundlagenkrise der Mathematik, Mathematische Zeitschrift 1921):

Ein Existentialsatz – etwa „es gibt eine gerade Zahl“ – ist überhaupt kein Urteil im eigentlichen Sinne, das einen Sachverhalt behauptet; Existentialsachverhalte sind eine leere Erfindung der Logiker.

„2 ist eine gerade Zahl“, das ist ein wirkliches, einem Sachverhalt Ausdruck gebendes Urteil; „es gibt eine gerade Zahl“ ist nur ein aus diesem Urteil gewonnenes Urteilsabstrakt.

Weyl 1921 (Forts.)

Bezeichne ich Erkenntnis als einen wertvollen Schatz, so ist das Urteilsabstrakt ein Papier, welches das Vorhandensein eines Schatzes anzeigt, ohne jedoch zu verraten, an welchem Ort. Sein einziger Wert kann darin liegen, daß es mich antreibt, nach dem Schatze zu suchen.

Konstruktive Mathematik

Weyl, Brouwer, Kolmogorov, Bishop: ohne Formalisierung.

Heyting, Feferman, Beeson, Troelstra:
formale Systeme der konstruktiven Mathematik.

Kreisel: Was weiß man mehr, wenn ein Satz konstruktiv bewiesen ist, als wenn man nur weiß, daß er wahr ist?

Beispiel eines nicht-konstruktiven Beweises

Es gibt irrationale Zahlen a, b mit a^b rational.

Beweis durch Fallunterscheidung.

Fall 1: $\sqrt{2}^{\sqrt{2}}$ ist rational. Wähle $a = \sqrt{2}$ und $b = \sqrt{2}$.
Dann sind a, b irrational, und nach Annahme ist a^b rational.

Fall 2: $\sqrt{2}^{\sqrt{2}}$ ist irrational. Wähle $a = \sqrt{2}^{\sqrt{2}}$ und $b = \sqrt{2}$.
Dann sind nach Annahme a, b irrational, und

$$a^b = \left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \left(\sqrt{2}\right)^2 = 2 \quad \text{ist rational.}$$

Programme aus konstruktiven Beweisen

Unentscheidbar, ob Programm seine Spezifikation erfüllt.
Formaler Beweis: Korrektheit leicht zu prüfen.

Beweis = Programm mit hinreichend vielen Kommentaren
(genauer: Programm extrahierbar).

Vision: Verwende mathematische Kultur zum Organisieren komplexer Strukturen, zwecks Programmextraktion
(Zum Beispiel: Steuerprogramme im Anlagenbau oder in der Telekommunikation).

Einwände

1. Algorithmus-Idee ist *vor* einem konstruktiven Beweis vorhanden.
(Richtig, aber: (a) Programm korrekt nach Konstruktion,
(b) Programmentwicklung wird möglich)

2. Komplexität des extrahierten Programms.
(Man erhält in polynomialer Zeit berechenbare Funktionen, bei geeigneter Einschränkung der Beweismittel).

3. Klassische Beweise verwendbar?
(Ja, aber man muß genauer hinsehen)

Beispiel: Zwischenwertsatz

Seien $a < b$ rational. Ist $f\colon [a,b] \to \mathbb{R}$ Lipschitz-stetig mit $f(a) \leq 0 \leq f(b)$, so findet man $x \in [a,b]$ mit $f(x) = 0$.

Beweisskizze. (1) ApproxSplit. Seien x, y, z gegeben mit $x < y$. Dann ist $z \leq y$ oder $x \leq z$.

(2) IVTAux. Gelte $a \leq c < d \leq b$, etwa $2^{-n} < d - c$, und $f(c) \leq 0 \leq f(d)$. Konstruiere c_1, d_1 mit $d_1 - c_1 = \frac{2}{3}(d-c)$, so daß $a \leq c \leq c_1 < d_1 \leq d \leq b$ und $f(c_1) \leq 0 \leq f(d_1)$.

(3) IVTcds. Iteriere den Schritt $c, d \mapsto c_1, d_1$ in IVTAux. Seien $x = (c_n)_n$ und $y = (d_n)_n$. Da f stetig ist, gilt $f(x) = 0 = f(y)$ für die reelle Zahl $x = y$.

IVTFinal

```
(set-goal
 (pf "all f,l,k0,k1.Cont f ->
     f f doml<<=0 -> 0<<=f f domr ->
     (1#exp 2 k0)<=f domr-f doml ->
     f domr-f doml<=exp 2 k1 ->
     (all c,d,m.f doml<=c -> d<=f domr ->
       (1#exp 2 m)<=d-c -> realLess(f c)(f d)(m+l)) ->
     ex z.Real z & f z===0"))
(assume "f" "l" "k0" "k1" "Cont f" "f a<<=0" "0<<=f b"
        "a <_k0 b" "b-a<=exp 2 k1" "HypSlope")
...
```

Beispiel

```
(define a-sq-minus-two
  (pt "contConstr 1 2([a0,n1]a0*a0-2)([n0]1)S"))

(define sqrt-two-approx
  (nt (apply mk-term-in-app-form
             (list
              (proof-to-extracted-term
               (theorem-name-to-proof "IVTApprox"))
              a-sq-minus-two
              (pt "1") (pt "1") (pt "1") (pt "2")))))
```

```
[n0]
 l[let cd1
 ((cDC rat@@rat)(1@2)
 ([n1]
   (cId rat@@rat=>rat@@rat)
   ([cd3]
     [let cd4
       ((2#3)*l cd3+(1#3)*r cd3@(1#3)*l cd3+(2#3)*r cd3)
       [if (0<=(l cd4*l cd4-2+(r cd4*r cd4-2))/2)
        (l cd3@r cd4)
        (l cd4@r cd3)]]))
 (PosPred(SZero(S(S n0)))))
```

```
[let cd2
 ((2#3)*l cd1+(1#3)*r cd1@(1#3)*l cd1+(2#3)*r cd1)
 [if (0<=(l cd2*l cd2-2+(r cd2*r cd2-2))/2)
  (l cd1@r cd2)
  (l cd2@r cd1)]]]
```

```
(lambda (n0)
  (car (let ([cd1
    (((cdc (cons 1 2))
      (lambda (n1)
        (lambda (cd3)
          (let ([cd4
                 (cons (+ (* 2/3 (car cd3)) (* 1/3 (cdr cd3)))
                       (+ (* 1/3 (car cd3)) (* 2/3 (cdr cd3))))
            (if (<= 0
                    (/ (+ (- (* (car cd4) (car cd4)) 2)
                          (- (* (cdr cd4) (cdr cd4)) 2))
                       2))
                (cons (car cd3) (cdr cd4))
```

```
                  (cons (car cd4) (cdr cd3))))))))
      (pospred (* (+ (+ n0 1) 1) 2)))])
(let ([cd2
       (cons (+ (* 2/3 (car cd1)) (* 1/3 (cdr cd1)))
             (+ (* 1/3 (car cd1)) (* 2/3 (cdr cd1))))])
           (if (<= 0
          (/ (+ (- (* (car cd2) (car cd2)) 2)
                (- (* (cdr cd2) (cdr cd2)) 2))
             2))
      (cons (car cd1) (cdr cd2))
      (cons (car cd2) (cdr cd1))))))
```

Experimente

Das Ergebnis (mit Fehlerschranke 2^{-20}) ist (in 2600 ms)

$$\frac{46422545185920140 4985}{3282569673945370776 27}$$

also

$$1.41421355\ldots$$

Übersetzt man die internen Terme in externen Code, so reichen 20 ms. Mit Fehlerschranke 2^{-80} (in 200 ms)

$$\frac{834217924535838176640617893668869675754670281954447525095818024665335844956569}{589881151426658740854227725580736348849310352832644300781946246613899173590427}$$