Formale Methoden der Softwaretechnik
Vorlesung vom 30.05.11: Logik Höherer Stufe

Till Mossakowski & Christoph Lüth

Universität Bremen

Sommersemester 2011

Fahrplan

- Alles über Logik höherer Stufe (Higher-order Logic, HOL):
  - Typen und Terme
  - Die Basis-Axiome
  - Definierte Operatoren

Fahrplan

- Aussagenlogik
- Prädikatenlogik
- Isabelle I: Grundlagen
  - Aussagenlogik und natürliches Schließen
  - Prädikatenlogik und Quantoren
  - Logik höherer Stufe
  - Definitionen und konservative Erweiterung
  - Automatische Beweisprozeduren
- Isabelle II: Anwendungen

Logik höherer Stufe

- Ziel: Formalisierung von Mathematik
  - "Logik für Erwachsene"
- Problem: Mögliche Inkonsistenz (Russel's Paradox)
- Lösung: Restriktion vs. Ausdrucksstärke
- Alternative Grundlagen:
  - Andere Typtheorien (Martin-Löf, Calculus of Constructions)
  - Ungetypte Mengenlehre (ZFC)
- HOL: guter Kompromiss, weit verbreitet.
  - Klassische Logik höherer Stufe nach Church
  - Schwächer als ZFC, stärker als Typtheorien

Warum Logik höherer Stufe?

- Aussagenlogik: keine Quantoren
- Logik 1. Stufe: Quantoren über Terme

$$\forall x\, y.\, x = y \longrightarrow y = x$$

- Logik 2. Stufe: Quantoren über Prädikaten und Funktionen

$$\forall P.(P\,0 \wedge \forall x.P\,x \longrightarrow P\,(S\,x)) \longrightarrow \forall x.P\,x$$

- Logik 3. Stufe: Quantoren über Argumenten von Prädikaten
- Logik höherer Stufe (HOL): alle endlichen Quantoren
  - Keine wesentlichen Vorteile von Logik 2. Ordnung

Vermeidung von Inkonsistenzen

- Russell's Paradox
  - $R = \{X \mid X \notin X\}$
  - Abhilfe: Typen
- Gödel's 2. Unvollständigkeitssatz:
  - Jede Logik, die ihre eigene Konsistenz beweist, ist inkonsistent.
- Unterscheidung zwischen Termen und Aussagen
  - Dadurch **in** HOL keine Aussage **über** HOL

Typen

- Typen $\mathcal{T}ype$ gegeben durch
  - Typkonstanten: $c \in \mathcal{C}_{Type}$ (Menge $\mathcal{C}_{Type}$ durch Signatur gegeben)
    - $Prop, Bool \in \mathcal{C}_{Type}$: $Prop$ alle Terme, $Bool$ alle Aussagen
  - Typvariablen: $\alpha \in \mathcal{V}_{Type}$ (Menge $\mathcal{V}_{Type}$ fest)
  - Funktionen: $s, t \in \mathcal{T}ype$ dann $s \Rightarrow t$ in $\mathcal{T}ype$
- Konvention: Funktionsraum nach rechts geklammert
  $\alpha \Rightarrow \beta \Rightarrow \gamma$ für $\alpha \Rightarrow (\beta \Rightarrow \gamma)$

Terme

- Terme $\mathcal{T}erm$ gegeben durch
  - Konstanten: $c \in \mathcal{C}$ (Menge $\mathcal{C}$ durch Signatur gegeben)
  - Variablen: $v \in \mathcal{V}$
  - Applikation: $s, t \in \mathcal{T}erm$ dann $s\,t \in \mathcal{T}erm$
  - Abstraktion: $x \in \mathcal{V}, t \in \mathcal{T}erm$ dann $\lambda x.\, t \in \mathcal{T}erm$
- Konventionen: Applikation links geklammert, mehrfache Abstraktion
  $\lambda x\, y\, z.\, f\, x\, y\, z$ für $\lambda x.\, \lambda y.\, \lambda z.\, ((f\, x)\, y)\, z$

Basis-Syntax

$$
\begin{array}{ll}
= & :: \alpha \Rightarrow \alpha \Rightarrow \mathit{Bool} \\
\longrightarrow & :: \mathit{Bool} \Rightarrow \mathit{Bool} \Rightarrow \mathit{Bool} \\
\iota & :: (\alpha \Rightarrow \mathit{Bool}) \Rightarrow \alpha \\
\\
\neg & :: \mathit{Bool} \Rightarrow \mathit{Bool} \\
\mathit{true} & :: \mathit{Bool} \\
\mathit{false} & :: \mathit{Bool} \\
\mathit{if} & :: \mathit{Bool} \Rightarrow \alpha \Rightarrow \alpha \Rightarrow \alpha \\
\forall & :: (\alpha \Rightarrow \mathit{Bool}) \Rightarrow \mathit{Bool} \\
\exists & :: (\alpha \Rightarrow \mathit{Bool}) \Rightarrow \mathit{Bool} \\
\wedge & :: \mathit{Bool} \Rightarrow \mathit{Bool} \Rightarrow \mathit{Bool} \\
\vee & :: \mathit{Bool} \Rightarrow \mathit{Bool} \Rightarrow \mathit{Bool}
\end{array}
$$

- Einbettung (wird weggelassen) $\mathit{trueprop} :: \mathit{Bool} \Rightarrow \mathit{Prop}$
- Basis-Operatoren: $=, \longrightarrow, \iota$
- Syntaktische Konventionen:
  - Bindende Operatoren: $\forall$, $\exists$, ι
    $\forall x.P \equiv \forall(\lambda x.P)$
  - Infix-Operatoren: $\wedge, \vee, \longrightarrow, =$
  - Mixfix-Operator:
    $\mathit{if}\ b\ \mathit{then}\ p\ \mathit{else}\ q \equiv \mathit{if}\ b\ p\ q$

Basis-Axiome I: Gleichheit

- Reflexivität:

$$\frac{}{t = t}\ \mathrm{refl}$$

- Substitutivität:

$$\frac{s = t \quad P(s)}{P(t)}\ \mathrm{subst}$$

- Extensionalität:

$$\frac{\forall x. fx = gx}{(\lambda x. fx) = (\lambda x. gx)}\ \mathrm{ext}$$

- Einführungsregel:

$$\frac{}{(P \longrightarrow Q) \longrightarrow (Q \longrightarrow P) \longrightarrow (P = Q)}\ \mathrm{iff}$$

Basis-Axiome II: Implikation und Auswahl

- Einführungsregel Implikation:

$$\frac{\begin{array}{c}[P]\\ \vdots \\ Q\end{array}}{P \longrightarrow Q}\ \mathrm{impl}$$

- Eliminationsregel Implikation:

$$\frac{P \longrightarrow Q \quad P}{Q}\ \mathrm{mp}$$

- Eliminationsregel Auswahloperator:

$$\frac{}{(\iota x. x = a) = a}\ \mathrm{the\_eq}$$

- HOL ist klassisch:

$$\frac{}{(P = \mathit{true}) \vee (P = \mathit{false})}\ \mathrm{true\_or\_false}$$

Die Basis-Axiome (Isabelle-Syntax)

$$
\begin{array}{rl}
\mathrm{refl}: & t = t \\
\mathrm{subst}: & [\![s = t; P(s)]\!] \Longrightarrow P(t) \\
\mathrm{ext}: & [\![\bigwedge x. fx = gx]\!] \Longrightarrow (\lambda x. fx) = (\lambda x. gx) \\
\mathrm{impl}: & [\![P \Longrightarrow Q]\!] \Longrightarrow P \longrightarrow Q \\
\mathrm{mp}: & [\![P \longrightarrow Q; P]\!] \Longrightarrow Q \\
\mathrm{iff}: & (P \longrightarrow Q) \longrightarrow (Q \longrightarrow P) \longrightarrow (P = Q) \\
\mathrm{the\_eq}: & (\iota x. x = a) = a \\
\mathrm{true\_or\_false}: & (P = \mathit{true}) \vee (P = \mathit{false})
\end{array}
$$

Abgeleitete Operatoren

$$
\begin{array}{rcl}
\mathit{true} & \equiv & (\lambda x. x) = (\lambda x. x) \\
\forall P & \equiv & (P = \lambda x. \mathit{true}) \\
\exists P & \equiv & \forall Q. (\forall x. Px \longrightarrow Q) \longrightarrow Q \\
\mathit{false} & \equiv & \forall P. P \\
\neg P & \equiv & P \longrightarrow \mathit{false} \\
P \wedge Q & \equiv & \forall R. (P \longrightarrow Q \longrightarrow R) \longrightarrow R \\
P \vee Q & \equiv & \forall R. (P \longrightarrow R) \longrightarrow (Q \longrightarrow R) \longrightarrow R \\
\mathit{if}\ P\ \mathit{then}\ x\ \mathit{else}\ y & \equiv & \iota z. (P = \mathit{true} \longrightarrow z = x) \wedge (P = \mathit{false} \longrightarrow z = y)
\end{array}
$$

Erweiterungen

- Weitere Operatoren
- Weitere Typen: natürliche Zahlen, Datentypen
- Axiomatisch (vgl. Peano/Presburger in FOL)
  - Mögliche Inkonsistenzen
- Konservative Erweiterung
  - Logik konsistentzerhaltend erweitern

Zusammenfassung

Logik höherer Stufe (HOL):

- Syntax basiert auf dem einfach getypten λ-Kalkül
- Drei Basis-Operatoren, acht Basis-Axiome
- Rest folgt durch konservative Erweiterung — nächstes Mal