Automatisierte Logik und Programmierung II

Sommersemester 2009

Christoph Kreitz

Theoretische Informatik, Raum 1.18, Telephon 3060

kreitz@cs.uni-potsdam.de

http://www.cs.uni-potsdam.de/ti/lehre/alupII.htm

1. Lehrziele
2. Rückblick & Ausblick
3. Organisatorisches

Ziel der Veranstaltung

Computergestütztes logisches Schließen

- **Mathematische Beweisführung**
  - Aufdeckung und Korrektur von Fehlern (Beweisprüfung)
  - Automatische Suche nach neuen Beweisen (Theorembeweisen)
- **Unterstützung für Entwurf zuverlässiger Software**
  - Fehlersuche und Korrektheitsbeweise (Verifikation)
  - Verbesserung der Performanz (Optimierung)
  - Erzeugung aus Spezifikationen (Synthese)
- **Inferenzmaschine für KI-Systeme**
  - Problemlöser und Planer für Roboter, …

Themen des ersten Semesters

Inferenzkalküle für Mathematik & Programmierung

- **Beweisen $\hat{=}$ Anwendung formaler Regeln**
  - Umgeht Mehrdeutigkeiten der natürlichen Sprache
  - Erlaubt schematische Lösung mathematischer Probleme
- **Kernbestandteile:**
  - Formale Sprache (Syntax + Semantik)
  - Ableitungssystem (Axiome + Inferenzregeln)
  - Notwendige Eigenschaften: korrekt, vollständig, automatisierbar
  - Nützliche Eigenschaften: konstruktiv, ausdrucksstark, lesbar
- **Vorgestellte Kalküle**
  - Prädikatenlogik (Logisches Schließen)
  - λ-Kalkül (Programmierung)
  - Einfache Typentheorie (Programmeigenschaften)
  - **Konstruktive Typentheorie (CTT)** (Uniformer Kalkül)

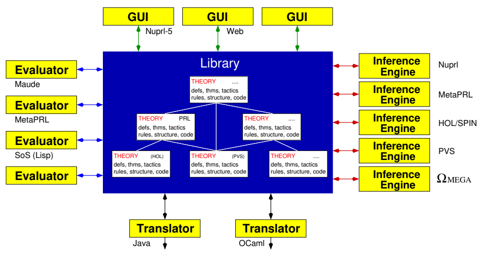
Eigenschaften der konstruktiven Typentheorie

- **Extrem ausdrucksstarkes Inferenzsystem**
  - Vereinheitlicht und erweitert Logik, λ-Kalkül & einfache Typentheorie
  - Direkte Darstellung der zentralen Konzepte (keine Simulation)
  - Formalisierung “natürlicher” Gesetze als Regeln
- **Sehr umfangreicher Formalismus**
  - Viele vordefinierte Basiskonstrukte, mehr als 150 Inferenzregeln
  - Programmkonstruktion durch konstruktive Beweisführung möglich
  - Abhängige Datentypen machen Wohlgeformtheit unentscheidbar
  - Gestützt auf eigenständige konstruktive semantische Theorie
- **Probleme bei der praktischen Anwendung**
  - Beweise erfordern viel Schreibarbeit $\longrightarrow$ interaktive Beweissysteme
  - Beweise sind unübersichtlich (komplexer Beweisbaum)
  - Beweise oft schwer zu finden (viele Regeln und Parameter)

    $\longrightarrow$ Automatisierung der Beweisführung

Themen des zweiten Teils

Konstruktion und Einsatz von Beweissystemen



- **Entwurf interaktiver Beweisassistenten**
  - Das NuPRL Logical Programming Environment
  - Das Isabelle Beweissystem unter ProofGeneral
- **Beweisautomatisierung**
  - Taktiken und Beweisplanung: benutzerdefinierte Beweisstrategien
  - Entscheidungsprozeduren: Lösung entscheidbarer Teilprobleme
- **Wissensverwaltung und Benutzerinteraktion**
- **Anwendungen & Demonstrationen**
  - Entwicklung formaler mathematischer Wissensbanken
  - Wissensbasierte Synthese von Programmen aus Spezifikationen
  - Korrektheitserhaltende Optimierung von Algorithmen

⋮

Organisatorisches

- **Zuordnung: theoretische/angewandte Informatik**
- **Veranstaltungsarten**
  - Vorlesung: Präsentation der zentralen Konzepte
  - Übung: Vertiefung und Anwendung theoretischer Aspekte
  - Optional: selbstgewähltes praktisches Beweiserprojekt
- **Veranstaltungstermine**
  - Do 10:15–11:45 – Vorlesung
  - Fr 12:30–14:00 – Vorlesung/Übung im Wechsel
- **Lehrmaterialien:**
  - Vorlesungsskript von 1995, Fachartikel und Manuals
- **Erfolgskriterien**
  - Abschlußprüfung (mündlich)

Aktive Teilnahme an Übungen wichtig