Automatisierte Logik und Programmierung II

Sommersemester 2004

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. Ziele
2. Rückblick Wintersemester 2003/04
3. Ausblick Sommersemester 2004
4. Organisatorisches

Ziel: 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 (Winter 2003/04)

Inferenzkalküle für Mathematik und 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)
  - Intuitionistische/Konstruktive Typentheorie (Uniformer Kalkül)

Eigenschaften der konstruktiven Typentheorie

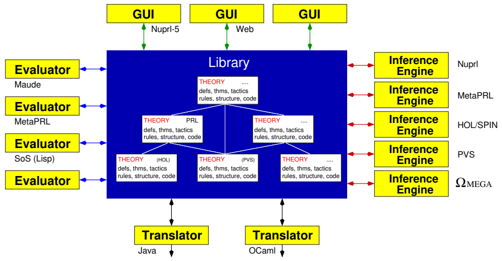
- **Extrem ausdrucksstarkes Inferenzsystem**
  – Direkte Darstellung der zentralen Konzepte (keine Simulation)
  – Formalisierung "natürlicher" Gesetze als Regeln
  – Sehr umfangreiche Theorie
    · Viele Basiskonstrukte, mehr als 150 Inferenzregeln
    · Programmkonstruktion durch konstruktive Beweisführung möglich
    · Abhängige Datentypen machen Wohlgeformtheit unentscheidbar
    · Gestützt auf konstruktive semantische Theorie

- **Praktische Probleme**
  – Beweise erfordern viel Schreibarbeit $\longrightarrow$ *interaktive Beweissysteme*
  – Beweise sind unübersichtlich (komplexer Beweisbaum)
  – Beweise manchmal schwer zu finden (viele Regeln und Parameter)

$\longrightarrow$ *Automatisierung der Beweisführung*

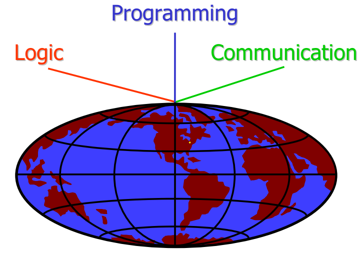
Themen des zweiten Teils (Sommer 2004)

- **Aufbau von Beweissystemen**
  - Implementierung interaktiver Beweisassistenten
  - Das Nuprl Logical Programming Environment



- **Beweisautomatisierung**
  - Taktisches Beweisen
  - Entscheidungsprozeduren
  - Integration externer Systeme

- **Anwendungen & Demonstrationen**
  - Entwicklung formaler Theorien
  - Programmsynthese
  - Optimierung des Kommunikationssytems Ensemble



Secure software infrastructure

⋮

Organisatorisches

- **Zuordnung: theoretische/angewandte Informatik**
- **Veranstaltungsarten**
  - Vorlesung: Präsentation der zentralen Konzepte
  - Übung: Vertiefung und Anwendung theoretischer Aspekte
  - Praktikum (optional): selbstgewähltes praktisches Beweiserprojekt
- **Kreditpunkte: 6–9** (Verbindliche Anmeldung bis 6. Mai)
- **Veranstaltungstermine**
  - Mo 13:30–15:00 – Vorlesung
  - Di 13:30–15:00 – Vorlesung/Übung im Wechsel
  - Do 11:00–12:30 – Praktikum
- **Lehrmaterialien:**
  - Vorlesungsskript von 1995, Fachartikel und Manuals
- **Erfolgskriterien**
  - Aktive Teilnahme an Übungen
  - Abschlußprüfung (mündlich) und praktische Projektaufgabe (je 50%)