Theorembeweiser und ihre Anwendungen

Prof. Dr.-Ing. Gregor Snelting
Dipl.-Inf. Univ. Daniel Wasserrab

Lehrstuhl Programmierparadigmen
IPD Snelting
Universität Karlsruhe (TH)

Teil II

Die Theorembeweiser dieser Welt

Der ideale Theorembeweiser!...

Der ideale Theorembeweiser!...

existiert nicht! (analog zu Programmiersprachen)

Der ideale Theorembeweiser!...

existiert nicht! (analog zu Programmiersprachen)

- jeder Beweiser hat Stärken und Schwächen
- für jedes Verifikationsziel gibt es geeignete Beweiser und weniger gut geeignete
- zu wissen, welcher Beweiser was warum gut kann, ist essentiell

10 unterschiedliche Theorembeweiser vorgestellt
Bekanntere ausgewählt, jedoch Liste nicht vollständig (und subjektiv)

F. Wiedijk (ed.).
The Seventeen Provers of the World.
Volume 3600 of *LNCS*. Springer, 2006.
`www.cs.ru.nl/~freek/comparison/comparison.pdf`

ACL2

A Computational Logic for Applicative Common LISP

Autoren: Matt Kaufmann, J. Strother Moore, Robert S. Boyer

Land: USA

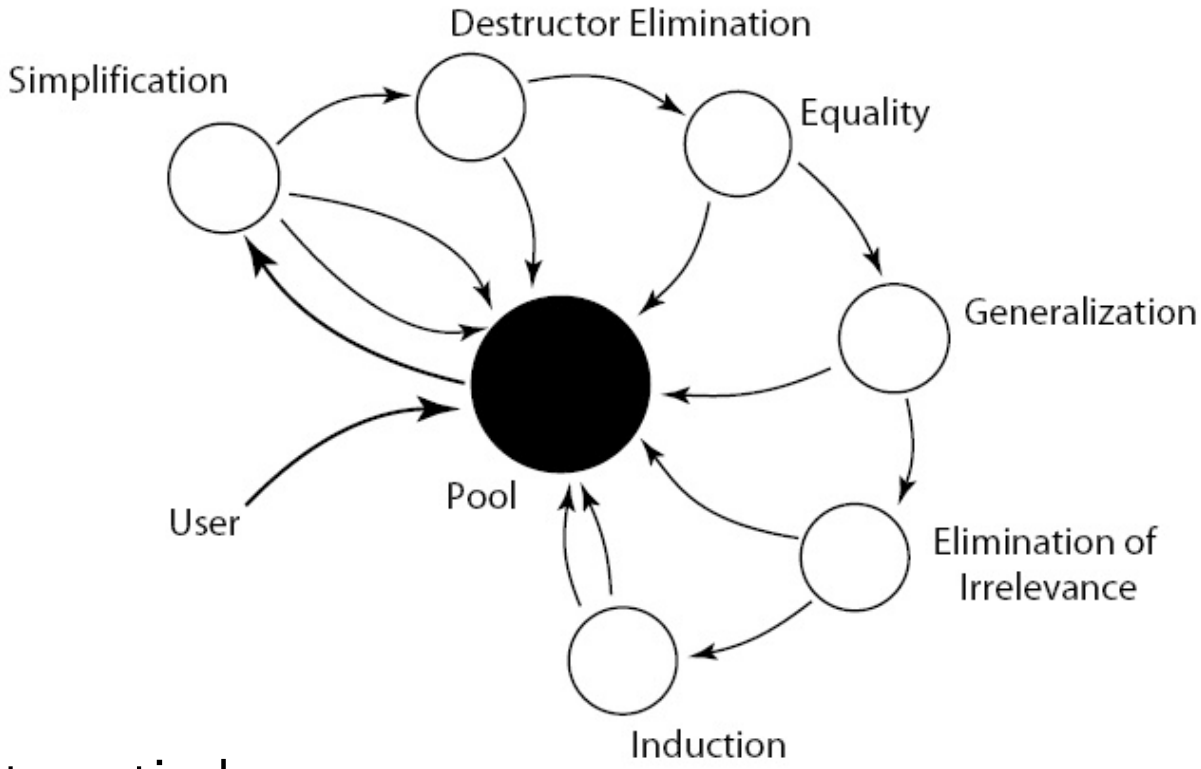
Sprache: applikative Teilmenge von *Common LISP* (nur Bootstrap, Rest in ACL2 geschrieben)

Website: `http://www.cs.utexas.edu/users/moore/acl2/`

automatischer Beweiser (mit interaktivem Add-on)
basiert auf Theorie rekursiver Funktionen
(Logik 1. Stufe mit Induktion für Inferenzregeln)
ACL2 ist auch Programmiersprache
Nachfolger des Boyer-Moore-Theorembeweiser *Nqthm*

ACL2

Wasserfallprinzip:



Falls nicht automatisch:

- Hilfslemmata zeigen lassen, werden *rewrite-rules*
- Hinweise (*hints*) geben zur Steuerung des Beweisprozesses

Coq

Autoren: T. Coquand, G. Huet, C. Paulin, C. Murthy, J.C. Filliâtre, B. Barras, H. Herbelin

Land: Frankreich

Sprache: Objective Caml (*OCaml*)

Website: `http://coq.inria.fr/`

prozeduraler Beweiser
basiert auf intuitionistischer Typtheorie:

- jede Aussage hat einen Typ, Beweisen entspricht Typcheck
- Curry-Howard-Isomorphismus: "*Beweise sind Programme*"
- verbindet Deduktion und getypten Lambda-Kalkül

kann aus Beweisen ausführbare Programme (proof terms) generieren

Einschub: Intuitionistische Logik

Idee: Beweis einer Aussage benötigt einen *Zeugen* für diese Aussage

Einschub: Intuitionistische Logik

Idee: Beweis einer Aussage benötigt einen *Zeugen* für diese Aussage

Deshalb

- "Gesetz des ausgeschlossenen Dritten" $P \vee \neg P$ (*tertium non datur*) nicht gültig! also keine Fallunterscheidung!
- jeder Existenzbeweis nur gültig mit Repräsentant!

Einschub: Intuitionistische Logik

Idee: Beweis einer Aussage benötigt einen *Zeugen* für diese Aussage

Deshalb

- "Gesetz des ausgeschlossenen Dritten" $P \vee \neg P$ (*tertium non datur*) nicht gültig! also keine Fallunterscheidung!
- jeder Existenzbeweis nur gültig mit Repräsentant!

weitere Aussagen, die **nicht** gelten:

- Dualität $\exists$ und $\forall$: $\exists x.P\ x = \neg\forall x.\neg P\ x$ bzw. $\forall x.P\ x = \neg\exists x.\neg P\ x$
- ein De Morgan-Gesetz: $\neg(P \vee Q) = \neg P \wedge \neg Q$
- Gesetz der doppelten Negation: $\neg\neg P = P$
- Auswahlaxiom: $\forall X.\exists f.\forall x \in X.f(x) \in x\ (x \neq \emptyset)$

Einschub: Intuitionistische Logik

Idee: Beweis einer Aussage benötigt einen *Zeugen* für diese Aussage

Deshalb

- "Gesetz des ausgeschlossenen Dritten" $P \vee \neg P$ (*tertium non datur*) nicht gültig! also keine Fallunterscheidung!
- jeder Existenzbeweis nur gültig mit Repräsentant!

weitere Aussagen, die **nicht** gelten:

- Dualität $\exists$ und $\forall$: $\exists x.P\ x = \neg\forall x.\neg P\ x$ bzw. $\forall x.P\ x = \neg\exists x.\neg P\ x$
- ein De Morgan-Gesetz: $\neg(P \vee Q) = \neg P \wedge \neg Q$
- Gesetz der doppelten Negation: $\neg\neg P = P$
- Auswahlaxiom: $\forall X.\exists f.\forall x \in X.f(x) \in x\ (x \neq \emptyset)$

Anmerkung: alle in der 1. Übung kennengelernten Isabelle/HOL Regeln sind intuitionistisch **außer** `classical` (*klassische* Logik)

Einschub: Intuitionistische Logik

Beispiel eines nicht-intuitionistischen Beweises:
Aussage: *"Es gibt irrationale Zahlen a und b, so dass a^b rational ist"*

Einschub: Intuitionistische Logik

Beispiel eines nicht-intuitionistischen Beweises:
Aussage: *"Es gibt irrationale Zahlen a und b, so dass a^b rational ist"*
Beweis durch Fallunterscheidung: ist $\sqrt{2}^{\sqrt{2}}$ rational?

Einschub: Intuitionistische Logik

Beispiel eines nicht-intuitionistischen Beweises:
Aussage: *"Es gibt irrationale Zahlen a und b, so dass a^b rational ist"*
Beweis durch Fallunterscheidung: ist $\sqrt{2}^{\sqrt{2}}$ rational?

Ja! dann wähle $a = \sqrt{2}$ und $b = \sqrt{2}$, damit a^b rational

Einschub: Intuitionistische Logik

Beispiel eines nicht-intuitionistischen Beweises:
Aussage: *"Es gibt irrationale Zahlen a und b, so dass a^b rational ist"*

Beweis durch Fallunterscheidung: ist $\sqrt{2}^{\sqrt{2}}$ rational?

Ja! dann wähle $a = \sqrt{2}$ und $b = \sqrt{2}$, damit a^b rational

Nein! dann wähle $a = \sqrt{2}^{\sqrt{2}}$ und $b = \sqrt{2}$, damit $a^b = 2$ und damit rational

□

Einschub: Intuitionistische Logik

Beispiel eines nicht-intuitionistischen Beweises:
Aussage: *"Es gibt irrationale Zahlen a und b, so dass a^b rational ist"*
Beweis durch Fallunterscheidung: ist $\sqrt{2}^{\sqrt{2}}$ rational?

Ja! dann wähle $a = \sqrt{2}$ und $b = \sqrt{2}$, damit a^b rational

Nein! dann wähle $a = \sqrt{2}^{\sqrt{2}}$ und $b = \sqrt{2}$, damit $a^b = 2$ und damit rational

□

Nur: ist jetzt $\sqrt{2}^{\sqrt{2}}$ rational oder nicht?
Problem: Beweis macht Existenzaussage, ohne Zeugen zu liefern!
Wird von Intuitionistikern als nicht aussagekräftig abgelehnt

Einschub: Intuitionistische Logik

Beispiel eines nicht-intuitionistischen Beweises:
Aussage: *"Es gibt irrationale Zahlen a und b, so dass a^b rational ist"*

Beweis durch Fallunterscheidung: ist $\sqrt{2}^{\sqrt{2}}$ rational?

Ja! dann wähle $a = \sqrt{2}$ und $b = \sqrt{2}$, damit a^b rational

Nein! dann wähle $a = \sqrt{2}^{\sqrt{2}}$ und $b = \sqrt{2}$, damit $a^b = 2$ und damit rational

□

Nur: ist jetzt $\sqrt{2}^{\sqrt{2}}$ rational oder nicht?
Problem: Beweis macht Existenzaussage, ohne Zeugen zu liefern!
Wird von Intuitionistikern als nicht aussagekräftig abgelehnt

P.S.: Intuitionistischer Beweis existiert, ist jedoch viel schwerer...

HOL

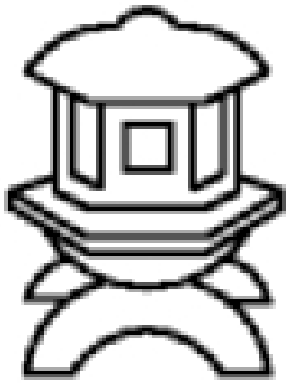
Theorem Proving in Higher Order Logic

Autoren: M. J. C. Gordon, T. F. Melham, Konrad Slind

Land: Großbritannien

Sprache: Standard ML (*SML*, funktional)

Website: `http://hol.sourceforge.net/`



prozeduraler Beweiser
basiert auf Typtheorie, folgend dem LCF Ansatz

HOL Light

Autoren: John Harrison

Land: USA

Sprache: *OCaml*

Website: `http://www.cl.cam.ac.uk/~jrh13/hol-light/`

prozeduraler Beweiser
basiert auf Typtheorie, wahrscheinlich "reinster" Ansatz zu LCF
einfache logische Regeln, Kern nur 433(!) Zeilen OCaml

geplant als experimentelle "Referenz" von HOL,
inzwischen in größeren Projekten verwendet

Einschub: LCF Ansatz

Logic for Computable Functions

wahrscheinlich erster Ansatz für maschinenunterstützte Beweiskonstruktion
"Vater": Robin Milner, 70er Jahre, Edinburgh (Edinburgh LCF)
weiterentwickelt durch Larry Paulson, Cambridge (Cambridge LCF)

Einschub: LCF Ansatz

Logic for Computable Functions

wahrscheinlich erster Ansatz für maschinenunterstützte Beweiskonstruktion
"Vater": Robin Milner, 70er Jahre, Edinburgh (Edinburgh LCF)
weiterentwickelt durch Larry Paulson, Cambridge (Cambridge LCF)

ML ursprüglich designt als Beweismanagement Sprache für HOL,
jetzt funktionale Programmiersprache (*Standard ML*)
1991 Turing Award für Milner u.a. für LCF und ML

Einschub: LCF Ansatz

Ideen:

- abstrakter Datentyp `theorem`
  keine Speicherung von Beweisen nötig
- Theoreme durch Inferenzregeln für Primitive (Operatoren etc.),
  garantiert durch ML Typsystem
- ML ermöglicht Erstellen von Taktiken
- Behauptung in *subgoals* aufteilen,
  wenn *subgoals* korrekt, Behauptung korrekt
- Methodik anwendbar auf andere Logiken als HOL

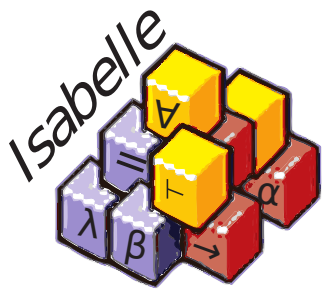
Isabelle

Autoren: Larry Paulson, Tobias Nipkow, Markus (Makarius) Wenzel

Land: Großbritannien, Deutschland

Sprache: *SML*

Website: `isabelle.in.tum.de`



prozeduraler/deklarativer Beweiser

generisch, d.h. instantiierbar z.B. mit Typ- (HOL)
oder Mengentheorie (Zermelo-Fraenkel) (als *Objektlogiken*)

Beweiserstellung

prozedural mittels unstrukturierten Taktikskripten ("*apply-Skripten*")

deklarativ mittels strukturierten Isar Beweisskripten (analog zu Mizar, nahe an üblicher mathematischer Notation)

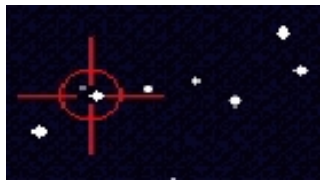
Mizar

Autoren: Andrzej Trybulec

Land: Polen (Japan, Kanada)

Sprache: *Pascal*

Website: `http://www.mizar.org/`



deklarativer Beweiser
basiert auf Mengentheorie
Schwerpunkt: mathematische Beweise
große mathematische Bibliothek, *Mizar Mathematical Library* (MML)
kaum Dokumentation

Mizar

Erstellung eines Mizar Beweises in 3 Stufen:

- Beweistext in ASCII Editor schreiben
- Text durch Accomodator laufen lassen
  erstellt Umgebung aus verfügbarer Datenbank
- Verifier prüft Text
  Ausgabe meldet nicht akzeptierte Fragmente des Beweises

Mizar

Erstellung eines Mizar Beweises in 3 Stufen:

- Beweistext in ASCII Editor schreiben
- Text durch Accomodator laufen lassen
  erstellt Umgebung aus verfügbarer Datenbank
- Verifier prüft Text
  Ausgabe meldet nicht akzeptierte Fragmente des Beweises

Mizar

Erstellung eines Mizar Beweises in 3 Stufen:

- Beweistext in ASCII Editor schreiben
- Text durch Accomodator laufen lassen
  erstellt Umgebung aus verfügbarer Datenbank
- Verifier prüft Text
  Ausgabe meldet nicht akzeptierte Fragmente des Beweises

Mizar

Erstellung eines Mizar Beweises in 3 Stufen:

- Beweistext in ASCII Editor schreiben
- Text durch Accomodator laufen lassen
  erstellt Umgebung aus verfügbarer Datenbank
- Verifier prüft Text
  Ausgabe meldet nicht akzeptierte Fragmente des Beweises

Mizar

```
theorem Th2:
 ex x, y st x is irrational & y is irrational &
  x.^.y is rational
proof
 set w = sqrt 2;
A1: w is irrational by INT_2:44,Th1;
 w>0 by AXIOMS:22,SQUARE_1:84;
 then
A2: (w.^.w).^.w = w.^.(w * w) by POWER:38
  .= w.^.(w^2) by SQUARE_1:def 3
  .= w.^.2 by SQUARE_1:def 4
  .= w^2 by POWER:53
  .= 2 by SQUARE_1:def 4;
 per cases;
 suppose
A3: w.^.w is rational;
  take w, w;
  thus thesis by A1,A3;
 suppose
A4: w.^.w is irrational;
  take w.^.w, w;
  thus thesis by A1,A2,A4,RAT_1:7;
end;
```

$$\exists x\ y.\ x \text{ irrational} \wedge y \text{ irrational} \wedge x^y \text{ rational}$$

Mizar

```
theorem Th2:
 ex x, y st x is irrational & y is irrational &
  x.^.y is rational
proof
 set w = sqrt 2;
A1: w is irrational by INT_2:44,Th1;
 w>0 by AXIOMS:22,SQUARE_1:84;
 then
A2: (w.^.w).^.w = w.^.(w * w) by POWER:38
  .= w.^.(w^2) by SQUARE_1:def 3
  .= w.^.2 by SQUARE_1:def 4
  .= w^2 by POWER:53
  .= 2 by SQUARE_1:def 4;
 per cases;
 suppose
A3: w.^.w is rational;
  take w, w;
  thus thesis by A1,A3;
 suppose
A4: w.^.w is irrational;
  take w.^.w, w;
  thus thesis by A1,A2,A4,RAT_1:7;
end;
```

$\exists x\ y.\ x$ irrational $\wedge$
y irrational $\wedge\ x^y$ rational

Label : *Aussage* by *Beweis*

Mizar

```
theorem Th2:
 ex x, y st x is irrational & y is irrational &
  x.^.y is rational
proof
 set w = sqrt 2;
A1: w is irrational by INT_2:44,Th1;
 w>0 by AXIOMS:22,SQUARE_1:84;
 then
A2: (w.^.w).^.w = w.^.(w * w) by POWER:38
  .= w.^.(w^2) by SQUARE_1:def 3
  .= w.^.2 by SQUARE_1:def 4
  .= w^2 by POWER:53
  .= 2 by SQUARE_1:def 4;
 per cases;
 suppose
A3: w.^.w is rational;
  take w, w;
  thus thesis by A1,A3;
 suppose
A4: w.^.w is irrational;
  take w.^.w, w;
  thus thesis by A1,A2,A4,RAT_1:7;
end;
```

$\exists x\ y.\ x$ irrational $\wedge$
y irrational $\wedge\ x^y$ rational

Label : *Aussage* by *Beweis*

Fallunterscheidung

Mizar

```
theorem Th2:
 ex x, y st x is irrational & y is irrational &
  x.^.y is rational
proof
 set w = sqrt 2;
A1: w is irrational by INT_2:44,Th1;
 w>0 by AXIOMS:22,SQUARE_1:84;
 then
A2: (w.^.w).^.w = w.^.(w * w) by POWER:38
  .= w.^.(w^2) by SQUARE_1:def 3
  .= w.^.2 by SQUARE_1:def 4
  .= w^2 by POWER:53
  .= 2 by SQUARE_1:def 4;
 per cases;
 suppose
A3: w.^.w is rational;
  take w, w;
  thus thesis by A1,A3;
 suppose
A4: w.^.w is irrational;
  take w.^.w, w;
  thus thesis by A1,A2,A4,RAT_1:7;
end;
```

$$\exists x\ y.\ x \text{ irrational} \wedge y \text{ irrational} \wedge x^y \text{ rational}$$

Label : *Aussage* by *Beweis*

Fallunterscheidung

Repräsentanten, erhalten durch Existenzquantor

Nuprl

nu (= new) *Proof Refinement Logic*
(gesprochen: "new pearl")

Autoren: Robert L. Constable

Land: USA

Sprache: *Common Lisp* und *Edinburgh ML*

Website: `http://www.cs.cornell.edu/Info/Projects/NuPRL/nuprl.html`

prozeduraler Beweiser, folgend dem LCF Ansatz
basiert auf konstruktiver Typtheorie
Beweisbaumeditor: Taktiken laufen auf Knoten

Otter

Organized Techniques for Theorem-proving and Effective Research

Autoren: William McCune, Larry Wos

Land: USA

Sprache: *ANSI C*

Website: `http://www-unix.mcs.anl.gov/AR/otter/`

automatischer Beweiser
basiert auf ungetypter Logik 1. Stufe
kombiniert mit Mace, sucht nach endlichen Modellen und Gegenbeispielen
nicht mehr gepflegt, Nachfolger Prover9

PVS

Prototype Verification System

Autoren: Sam Owre, Natarajan Shankar, John Rushby

Land: USA

Sprache: *Common Lisp*

Website: `http://pvs.csl.sri.com/`

prozeduraler Beweiser
basiert auf Typtheorie
Kombination mit Spezifikationssprache und Typchecker

PVS

Beweisen in PVS in folgenden Schritten:

- Benutzer erstellt Spezifikationsdateien
- Typüberprüfung dieser Dateien, generiert *TCC*s (type-correctness conditions), Beweisverpflichtungen
- interaktives Beweisen von Formeln

PVS

Beweisen in PVS in folgenden Schritten:

- Benutzer erstellt Spezifikationsdateien
- Typüberprüfung dieser Dateien, generiert *TCC*s (type-correctness conditions), Beweisverpflichtungen
- interaktives Beweisen von Formeln

PVS

Beweisen in PVS in folgenden Schritten:

- Benutzer erstellt Spezifikationsdateien
- Typüberprüfung dieser Dateien, generiert *TCC*s (type-correctness conditions), Beweisverpflichtungen
- interaktives Beweisen von Formeln

PVS

Beweisen in PVS in folgenden Schritten:

- Benutzer erstellt Spezifikationsdateien
- Typüberprüfung dieser Dateien, generiert *TCC*s (type-correctness conditions), Beweisverpflichtungen
- interaktives Beweisen von Formeln

PVS

Beweisen in PVS in folgenden Schritten:

- Benutzer erstellt Spezifikationsdateien
- Typüberprüfung dieser Dateien, generiert *TCC*s (type-correctness conditions), Beweisverpflichtungen
- interaktives Beweisen von Formeln

PVS besitzt Subtypen und *dependent types* (Typen abhängig von Werten), auf BDDs (*binary decision diagram*) basierenden Model Checker

Twelf

Autoren: Frank Pfenning, Carsten Schuermann

Land: USA

Sprache: *SML*

Website: `http://twelf.plparty.org/wiki/Main_Page`

eigentlich Metalogik-Umgebung für deduktive Systeme
beinhaltet deklarativen Beweiser (*induktiver Metatheorem-Beweiser*)
basierend auf Typtheorie, genauer: LF Logical Framework
(abhängig getypter Lambda-Kalkül)
baut auf Elf Constraint Logic Programmiersprache auf

Fazit

Viel Auswahl, viele Möglichkeiten
passenden Beweiser für passendes Problem

Fazit

Viel Auswahl, viele Möglichkeiten
passenden Beweiser für passendes Problem

Choose your weapon!