Automatic Deduction — LV 703522
Introduction to Isabelle

Clemens Ballarin

Universität Innsbruck

Intro

Organisatorials

Web Page

`http://cl-informatik.uibk.ac.at/teaching/ss08`

Installation

- Installed on `zid-gpl`, type `Isabelle &`
- We will use Isabelle 2007.

General Schedule

- Lectures
- Homework
- Exercise sessions where homework will be discussed

What You Will Learn

- How to use a theorem prover
- Background, how it works
- How to prove and specify

Health Warning

Theorem Proving is addictive

Contents

- Intro & motivation, getting started with Isabelle
- Foundations & Principles
  - Lambda Calculus
  - Types & Classes
  - Natural Deduction
  - Term rewriting
- Proof & Specification Techniques
  - Isar: mathematics style proofs
  - Inductively defined sets, rule induction
  - Datatypes, structural induction
  - Recursive functions & code generation

Contents

- Intro & motivation, getting started with Isabelle
- Foundations & Principles
  - Lambda Calculus
  - Types & Classes
  - Natural Deduction
  - Term rewriting
- Proof & Specification Techniques
  - Isar: mathematics style proofs
  - Inductively defined sets, rule induction
  - Datatypes, structural induction
  - Recursive functions & code generation

Schedule

6 Mar	Introduction		
13 Mar	λ-calculus	3 April	Exercises
10 April	Higher-Order Logic	17 April	"
24 April	Rewriting	8 May	"
15 May	ISAR	29 May	"
5 June	Sets and inductive definitions	12 June	"
19 June	HOL as programming language	26 June	"
3 July	Exam		

Introduction

What is a Proof?

To prove (Merriam-Webster)

1. from Latin probare (test, approve, prove)
2. to learn or find out by experience (archaic)
3. to establish the existence, truth, or validity of (by evidence or logic)
   Prove a theorem; the charges were never proved in court

Pops up everywhere

- politics (weapons of mass destruction)
- courts (beyond reasonable doubt)
- religion (god exists)
- science (cold fusion works)

What is a Mathematical Proof?

In mathematics, a proof is a demonstration that, given certain axioms, some statement of interest is necessarily true.

(Wikipedia)

Example: $\sqrt{2}$ is not rational.

Proof. Assume there is $r \in \mathbb{Q}$ such that $r^2 = 2$.
Hence there are mutually prime p and q with $r = \frac{p}{q}$.
Thus $2q^2 = p^2$, i.e. p^2 is divisible by 2.
2 is prime, hence it also divides p, i.e. $p = 2s$.
Substituting this into $2q^2 = p^2$ and dividing by 2 gives $q^2 = 2s^2$.
Hence, q is also divisible by 2. Contradiction. Qed.

Nice, but...

- Still not rigourous enough for some
  - What are the axioms?
  - What are the rules?
  - How big can the steps be?
  - What is obvious or trivial?
- Informal language, easy to get wrong,
- easy to miss something, easy to cheat.

Theorem. A cat has nine tails.

Proof. No cat has eight tails.
One cat has one more tail than no cat.
Hence it must have nine tails.

What is a Formal Proof?

A derivation in a formal calculus

Example

$A \wedge B \longrightarrow B \wedge A$ derivable in the following system:

$$\frac{X \in S}{S \vdash X}\ \text{(assumption)} \qquad \frac{S \cup \{X\} \vdash Y}{S \vdash X \longrightarrow Y}\ \text{(impI)}$$

$$\frac{S \vdash X \qquad S \vdash Y}{S \vdash X \wedge Y}\ \text{(conjI)} \qquad \frac{S \cup \{X, Y\} \vdash Z}{S \cup \{X \wedge Y\} \vdash Z}\ \text{(conjE)}$$

Proof

1.	$\{A, B\} \vdash B$	(by assumption)
2.	$\{A, B\} \vdash A$	(by assumption)
3.	$\{A, B\} \vdash B \wedge A$	(by conjI with 1 and 2)
4.	$\{A \wedge B\} \vdash B \wedge A$	(by conjE with 3)
5.	$\{\} \vdash A \wedge B \longrightarrow B \wedge A$	(by impI with 4)

What is a Theorem Prover?

Implementation of a formal logic on a computer

- Fully automated (propositional logic)
- Automated, but not necessarily terminating (first-order logic)
- With automation, but mainly interactive (higher-order logic)

- Based on rules and axioms
- Can deliver proofs

There are other (algorithmic) verification tools:

- Model checking, static analysis, ...
- Usually do not deliver proofs

Why Theorem Proving?

- Analysing systems/programs thoroughly
- Finding design and specification errors early
- High assurance (machine checked)
- Can communicate proof for checking by others
- It's not always easy
- It's fun

Isabelle

A generic interactive proof assistant

- Generic:
  not specialised to one particular logic
  (two large developments: HOL and ZF, will use HOL)
- Interactive:
  more than just yes/no, you can interactively guide the system
- Proof assistant:
  helps to explore, find, and maintain proofs

The Heads behind Isabelle

Larry Paulson

Tobias Nipkow

Markus Wenzel

Why Isabelle?

- Free
- Widely used systems
- Active development
- High expressiveness and automation
- Reasonably easy to use
- and because we know it best ;-)

If I prove it on the computer, it is correct, right?

If I Prove It on the Computer, It Is Correct, Right?

No, because:

1. Hardware could be faulty
2. Operating system could be faulty
3. Implementation runtime system could be faulty
4. Compiler could be faulty
5. Implementation of prover could be faulty
6. Logic could be inconsistent
7. Theorem could mean something else

If I Prove It on the Computer, It Is Correct, Right?

No, but:

Probability for

- 1 and 2 reduced by using different systems
- 3 and 4 reduced by using different compilers
- Faulty implementation reduced by right architecture
- Inconsistent logic reduced by implementing & analysing it
- Wrong theorem reduced by expressive/intuitive logics

No guarantees, but assurance immensely higher than with manual proof.

If I Prove It on the Computer, It Is Correct, Right?

Soundness architectures

Careful implementation	PVS
LCF approach, small proof kernel	HOL4 Isabelle
Explicit proofs + proof checker	Coq Twelf Isabelle

Meta Logic

Meta language

The language used to talk about another language.

Examples

German in a Spanish class, English in an English class.

Meta logic

The logic used to formalize another logic.

Example

Mathematics used to formalize derivations in formal logic.

Meta Logic — Example

Syntax

Formulae: $F ::= V \mid F \longrightarrow F \mid F \wedge F \mid$ False

$V ::= [A - Z]$

Derivability: $S \vdash X$ where X a formula, S a set of formulae

Logic vs. meta logic

$$\frac{X \in S}{S \vdash X} \qquad \frac{S \cup \{X\} \vdash Y}{S \vdash X \longrightarrow Y}$$

$$\frac{S \vdash X \qquad S \vdash Y}{S \vdash X \wedge Y} \qquad \frac{S \cup \{X, Y\} \vdash Z}{S \cup \{X \wedge Y\} \vdash Z}$$

Isabelle's Meta Logic

$$\bigwedge \qquad \Longrightarrow \qquad \lambda \qquad \equiv$$

$\bigwedge$

Syntax	$\bigwedge x.\ F$	(F another meta level formula)
in ASCII	`!!x. F`	

- Universal quantifier at the meta level
- Used to denote parameters
- Example and more later

$\Longrightarrow$

Syntax $A \Longrightarrow B$ (A, B other meta level formulae)

in ASCII `A ==> B`

Binds to the right

$$A \Longrightarrow B \Longrightarrow C \quad = \quad A \Longrightarrow (B \Longrightarrow C)$$

Abbreviation

$$[\![A; B]\!] \Longrightarrow C \quad = \quad A \Longrightarrow B \Longrightarrow C$$

- Read: A and B implies C
- Used to write rules, theorems, and proof states

Example: a Theorem

Mathematics	if $x < 0$ and $y < 0$, then $x + y < 0$
Formal logic	$\vdash\ x < 0 \wedge y < 0 \longrightarrow x + y < 0$
variation	$\{x < 0; y < 0\} \vdash\ x + y < 0$
Isabelle	**lemma** "$x < 0 \wedge y < 0 \longrightarrow x + y < 0$"
variation	**lemma** "$[\![x < 0; y < 0]\!] \Longrightarrow x + y < 0$"
variation	**lemma** **assumes** "$x < 0$" **and** "$y < 0$" **shows** "$x + y < 0$"

Example: a Rule

Logic $$\frac{X \qquad Y}{X \land Y}$$

variation $$\frac{S \vdash X \qquad S \vdash Y}{S \vdash X \land Y}$$

Isabelle $$[\![X; Y]\!] \Longrightarrow X \land Y$$

Example: a Rule with Nested Implication

Logic

$$\frac{X \vee Y \qquad \begin{matrix} X \\ \vdots \\ Z \end{matrix} \qquad \begin{matrix} Y \\ \vdots \\ Z \end{matrix}}{Z}$$

variation

$$\frac{S \cup \{X\} \vdash Z \qquad S \cup \{Y\} \vdash Z}{S \cup \{X \vee Y\} \vdash Z}$$

Isabelle

$$[\![X \vee Y; X \Longrightarrow Z; Y \Longrightarrow Z]\!] \Longrightarrow Z$$

λ

Syntax $\lambda x.\ F$ (F another meta level formula)
in ASCII `%x. F`

- Lambda abstraction
- Used for functions in object logics
- Used to encode bound variables in object logics
- More about this in the next lecture

System Architecture

Proof General – user interface

HOL, ZF – object-logics

Isabelle – generic, interactive theorem prover

Standard ML – logic implemented as ADT

User can access all layers!

System Requirements

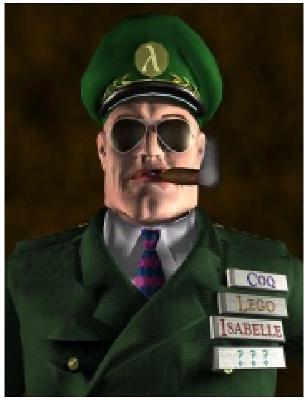
- **Linux**, **FreeBSD**, **MacOS X** or **Solaris**
- **Standard ML**
  (PolyML fastest, SML/NJ supports more platforms)
- **XEmacs** or **Emacs**
  (for ProofGeneral)

Documentation

Available from `http://isabelle.in.tum.de`

- Learning Isabelle
  - Tutorial on Isabelle/HOL (LNCS 2283)
  - Tutorial on Isar
  - Tutorials for various packages
- Reference Manuals
  - Isabelle/Isar Reference Manual
  - Isabelle Reference Manual
  - Isabelle System Manual
- Reference Manuals for Object-Logics

ProofGeneral



- User interface for Isabelle
- Runs under XEmacs or Emacs
- Isabelle process in background

Interaction via

- Basic editing in XEmacs (with highlighting etc)
- Buttons (tool bar)
- Key bindings
- ProofGeneral Menu (lots of options, try them)

X-Symbol Cheat Sheet

Input of funny symbols in ProofGeneral

- via menu ("X-Symbol")
- via ASCII encoding (similar to LaTeX): \<and>, \<or>, ...
- via abbreviation: /\\, \\/, -->, ...
- via *rotate*: l C-. $= \lambda$ (cycles through variations of letter)

	$\forall$	$\exists$	λ	$\neg$	$\wedge$	$\vee$	$\longrightarrow$	$\Rightarrow$
1.	\<forall>	\<exists>	\<lambda>	\<not>	/\\	\\/	-->	=>
2.	ALL	EX	%	~	&	|		

1. converted to X-Symbol
2. stays ASCII

For more symbols, see LNCS 2283, Appendix 1.

Demo