universität innsbruck

Summer Term 2019

Interactive Theorem Proving using Isabelle/HOL

Session 8

Christian Sternagel

Department of Computer Science

Topics

calculational reasoning, case analysis, code generation, computation induction, data type invariants, document preparation, finding theorems, first steps, functional programming in HOL, higher-order logic, history and motivation, induction, inductive definitions, Isabelle basics, Isabelle/Isar, Isabelle/ML, IsaFoR/CeTA, locales, manual termination proofs, multisets, natural deduction, notation, proof methods, PSL: a high-level proof strategy language, rule induction, rule inversion, session management, sets, simplification, sledgehammer, structural induction, structured proof, The Archive of Formal Proofs, the certification approach, total recursive functions, type classes, type definitions, well-foundedness

Overview

- Session Management
- Document Preparation
- Sets
- Exercises

Isabelle Sessions

- session is "project" consisting of collection of theory files
- sessions are defined in `ROOT` files
- processing sessions may take considerable time
- possible to capture state of sessions in persistent heap image/session image

Session Specifications

$$
\begin{array}{rcl}
\mathit{session} & ::= & \textbf{session}\ \mathit{name} = \mathit{name} + \mathit{description}^?\ \mathit{options}^?\ \mathit{sessions}^?\ \mathit{theories}^* \\
\mathit{description} & ::= & \textbf{description}\ \langle \mathit{text} \rangle \\
\mathit{options} & ::= & \textbf{options}\ [(\mathit{key}{=}\mathit{value}\ \mid\ \mathit{key})^+] \\
\mathit{sessions} & ::= & \textbf{sessions}\ \mathit{name}^+ \\
\mathit{theories} & ::= & \textbf{theories}\ ([(\mathit{key}{=}\mathit{value}\ \mid\ \mathit{key})^+])^?\ \mathit{name}
\end{array}
$$

Example Session

```
session Test_Session = HOL +
  description ‹Test Session›
  options [names_short]
  theories
    Test
```

Invoking the Build Process

use '`$ isabelle build [OPTIONS] S1 ... SN`' to run sessions `S1` to `SN` with `OPTIONS`:

`-d`	*dir*	search for `ROOT` files in *dir*
`-D`	*dir*	search for `ROOT` file in *dir* an select all its sessions
`-b`		build heap image
`-o`	*option*	override Isabelle option (syntax: *name*=*val* or *name*)
`-v`		be verbose
`-n`		no build; test dependencies only

Some Available Options

- `browser_info` – output HTML browser info (default: `false`)
- `document=pdf` – output PDF document
- `document_output=dir` – specify alternative directory `dir` for generated output
- `quick_and_dirty` – accept proof by `sorry`
- `names_short` – do not use qualified names in output
- `show_question_marks` – control printing of question marks for schematic variables

Sectioning and Structuring

- `chapter`, `section`, `subsection`, ... – different levels of sectioning
- ■ – for itemizations
- ► – for enumerations
- `text` ‹ ... › – plain text and LaTeX code

Isabelle Symbols – Lists

symbol	internal	abbreviation
■	`\<^item>`	\ i t e m
►	`\<^enum>`	\ e n u m

General Structure of Document Antiquotations

$$
\begin{array}{rcl}
\mathit{antiquotation} & ::= & @\{\mathit{name}\ \mathit{options}^{?}\ \mathit{arguments}\} \\
& | & \backslash\!<\!\hat{}\,\mathit{name}\!> \mathit{cartouche} \\
& | & \mathit{cartouche} \\
\mathit{options} & ::= & [\,] \mid [\ \mathit{option}\ (,\mathit{option})^{*}\ ] \\
\mathit{option} & ::= & \mathit{name} \mid \mathit{name} = \mathit{name}
\end{array}
$$

Antiquotations

- `text` – uninterpreted inner syntax
- `theory_text` – uninterpreted outer syntax
- `theory` – session-qualified theory name
- `thm` fact^{*} – theorem statements
- `prop` ϕ – well-typed proposition ϕ
- `lemma` ϕ `by` *method* –
- `term` t – well-typed term t

Antiquotations (cont'd)

- `value` t – result of evaluating t
- `term_type` t – well-typed term t together with its type
- `typeof` t – type of well-typed term t
- `const` c – constant c
- `typ` τ – well-formed type τ
- `type` κ – type constructor κ
- `method` m – proof method m
- `datatype` τ – data type specification of τ
- `verbatim` – uninterpreted text in typewriter font
- ...

Setting Up a Session Root Directory

- use '`$ isabelle mkroot dir`' to set up directory `dir` (can be `.`) as session root
- results in:
  - `dir/ROOT` – session setup for document preparation
    (note the `document_files` section)
  - `dir/document/root.tex` – LaTeX setup
- for BibTeX (together with `cite` antiquotation) create file `document/root.bib` and add `root.bib` to `document_files` section in `ROOT` file

Sets in Isabelle

- type ‘`'a set`’ for sets with elements of type `'a`

Set Basics

- `x ∈ A` – membership
- `A ∩ B` – intersection
- `A ∪ B` – union
- `-A` – complement
- `A - B` – difference
- `A ⊆ B` – subset
- `{}` – empty set
- `UNIV` – universal set of specific type
- `{x}` – singleton set
- `insert x A` – insertion of single elements (`insert x A = {x} ∪ A`)
- `f ` A` – image of function with respect to set (“map `f` over elements of `A`”)

Example Proof – session08/Demo08.thy

```
lemma "A ∩ (B ∪ C) ⊆ (A ∩ B) ∪ (A ∩ C)"
sorry
```

Cantor's Theorem

Let f be a mapping from set A to its power set $\mathscr{P}(A)$. Then $f : A \to \mathscr{P}(A)$ is not surjective.

Consequence

since injective $g : A \to \mathscr{P}(A)$ exists, cardinality of $\mathscr{P}(A)$ is strictly greater than cardinality of A

Example – Proof of Cantor's Theorem

```
lemma
  assumes "f ` A ⊆ Pow A" — ‹a mapping from A to its power set›
  shows "¬ (Pow A ⊆ f ` A)"
sorry
```

Hints:

- `Pow :: 'a set ⇒ 'a set set` computes power set
- `let ?X = "t"` introduces `?X` as input abbreviation for term `t` inside proof

Exercises (start from `Exercises08.thy`)

URL

http://cl-informatik.uibk.ac.at/teaching/ss19/itp/thys/Exercises08.thy

Further Reading


Makarius Wenzel.
The Isabelle System Manual.
Isabelle documentation, 2018.