Interactive Theorem Proving in HOL4

Course 10: Miscellaneous

Dr Chun TIAN
chun.tian@anu.edu.au

3 ottobre 2024

Australian National University

Acknowledgement of Country

We acknowledge and celebrate the First Australians on whose traditional lands we meet, and pay our respect to the elders past and present.

More information about Acknowledgement of Country can be found here and here

Embedding HOL in LaTeX

Example: The Strong Law of Large Numbers (SLLN)

You write (in LaTeX documents):

```
\HOLthm{large_number.SLLN_IID_applied}
```

You get:

```
⊢ prob_space p ∧ (∀n. real_random_variable (X n) p) ∧
  pairwise_indep_vars p X (λn. Borel) 𝒰(:num) ∧
  identical_distribution p X Borel 𝒰(:num) ∧
  integrable p (X 0) ⇒
  ((λn x. ∑ (λi. X i x) (count1 n) / &SUC n) ⟶
   (λx. expectation p (X 0))) (almost_everywhere p)
```

Idea and User Workflow

The Idea

HOL translates special LaTeX commands to HOL terms/types/theorems in LaTeX notations for building the final documents (in PDF, etc.)

The User Workflow

1. (Optional) Write and build local (or user) HOL theories;
2. Build a special program called `munge.exe` from user HOL theories (by `Holmake`, with help of `Holmakefile`)
3. Write pre-LaTeX documents using commands from HOL's `holtex` package, etc.;
4. Translate the pre-LaTeX documents (by `munge.exe`) to (final) LaTeX documents;
5. Build PDF from (final) LaTeX documents as usual (by `pdflatex`, etc.)

Local HOL theory for paper writing

A small HOL script (usually called `paperScript.sml`) is very useful for the following purposes:

- The theory script lives in the same directory with your other LaTeX document files;
- It opens all theories of theorems to be generated in your LaTeX document;
- It may contain *local* definitions and theorems only needed in your LaTeX document;
- It's the only input theory for building `munge.exe`.

Sample code (`paperScript.sml`)

```
open HolKernel Parse boolLib bossLib;
open large_numberTheory; (* in examples/probability *)
val _ = new_theory "paper";
(* local scripts here *)
val _ = export_theory ();
```

Prepare the LaTeX sources for embedding HOL

- LaTeX sources containing embedded HOL code must be renamed from `*.tex` to `*.htex`;[1]
- Copy all `*.sty` files from `$(HOLDIR)/src/TeX` to your paper directory;
- Use at least the package `alltt` and `holindex` in the preamble of your LaTeX document.

Sample code (`report.htex`)

```
\documentclass{article}
\usepackage{alltt}
\usepackage{holindex}
\begin{document}
\begin{alltt}
\HOLthm{large_number.SLLN_IID_applied}
\end{alltt}
\end{document}
```

[1] In Emacs, putting a first line `%%% -*- Mode:  LaTeX -*-` into the `htex` files will force them in LaTeX mode.

Use Holmakefile for building everything

A `Holmakefile` in the same directory must be written for building `paperTheory` (from `paperScript.sml`), `munge.exe`, and the final LaTeX file (`*.tex`) from `*.htex` (LaTeX sources).

Sample code (`Holmakefile`)

```
INCLUDES = $(HOLDIR)/examples/probability
OPTIONS = QUIT_ON_FAILURE
MUNGE = ./munge.exe
MUNGE_DEPS = paperTheory
EXTRA_CLEANS = $(MUNGE)
all: $(MUNGE) report.tex
$(MUNGE): $(patsubst %,%.uo,$(MUNGE_DEPS))
    $(HOLDIR)/bin/mkmunge.exe $(MUNGE_DEPS)
report.tex: $(MUNGE) report.htex
    $(MUNGE) < report.htex > report.tex
.PHONY: all
```

LaTeX commands provided by HOL package

Embedding terms and types

- `\HOLtm{!x. P /\ Q \/ R}` translates to $\forall x.\ P \land Q \lor R$
- `\HOLty{:'a -> bool}` translates to $\alpha \to \mathtt{bool}$
- `\HOLtm[showtypes]{\x. P x}` translates to $\lambda (x : \alpha).\ (P : \alpha \to \beta)\ x$

Options when embedding theorems

- By default all output theorems are specialised (i.e. no universal quantifiers):

  `\HOLthm{pred_set.EXTENSION}`

  $\vdash s = t \iff \forall x.\ x \in s \iff x \in t$
- The default specialisation can be disabled, and type information can be shown:

  `\HOLthm[nosp,showtypes]{pred_set.EXTENSION}`

  $\vdash \forall (s : \alpha \to \mathtt{bool})\ (t : \alpha \to \mathtt{bool}).$
  $\quad s = t \iff \forall (x : \alpha).\ x \in s \iff x \in t$

The Definition package

Different types of definitions supported by Definition

- Abbreviations:
- Recursive functions;
- Mutually recursive functions.

Sample code (mutually recursive functions)

```
Definition MY_EVEN_ODD[simp] :
   (EVEN 0 = T) /\
   (EVEN (SUC n) = ODD n) /\
   (ODD 0 = F) /\
   (ODD (SUC n) = EVEN n)
End
```

Prove termination of recursive functions

By default, HOL's `Definition` package understands termination of natural numbers and lists, etc. Sometimes user must explicitly prove the termination of recursive functions.

Sample code (from `primeTheory`)

```
Definition count_up_def:
   count_up n m k = if m = 0 then 0
   else if n <= m then k else count_up n (2 * m) (SUC k)
Termination
   WF_REL_TAC ‘measure (\(n, m, k). n - m)‘
End
```

Frequently used termination arguments: `WF_REL_TAC ‘measure FST‘`, `WF_REL_TAC ‘measure SND‘` (when the first/second argument is natural number).

Defining Inductive Relations (1)

One can define new predicates by specifying the relation of its values on inductively given values:

Sample code

```
Inductive MY_EVEN_ODD :
   EVEN 0 /\
   (!n. ODD n ==> EVEN (n + 1)) /\
   (!n. EVEN n ==> ODD (n + 1))
End

Inductive MY_RTC :
   (!x. MY_RTC R x x) /\
   (!x z. (?y. R x y /\ MY_RTC R y z) ==> MY_RTC R x z)
End
```

Defining Inductive Relations (2)

Each successful invocation of the `Inductive` package returns 3 theorems fully capturing the properties of the newly defined constant:

1. The *rules* theorem (the original inputs):

   `[MY_RTC_rules]`

   $\vdash (\forall x.\ \texttt{MY\_RTC}\ R\ x\ x) \land$
   $\forall x\ z.\ (\exists y.\ R\ x\ y \land \texttt{MY\_RTC}\ R\ y\ z) \Rightarrow \texttt{MY\_RTC}\ R\ x\ z$

2. The *cases* theorem (the fixed point):

   `[MY_RTC_cases]`

   $\vdash \texttt{MY\_RTC}\ R\ a_0\ a_1 \iff a_1 = a_0 \lor \exists y.\ R\ a_0\ y \land \texttt{MY\_RTC}\ R\ y\ a_1$

3. The *ind* (induction) theorem (the smallest fixed point):

   `[MY_RTC_ind]`

   $\vdash (\forall x.\ MY\_RTC'\ x\ x) \land$
   $(\forall x\ z.\ (\exists y.\ R\ x\ y \land MY\_RTC'\ y\ z) \Rightarrow MY\_RTC'\ x\ z) \Rightarrow$
   $\forall a_0\ a_1.\ \texttt{MY\_RTC}\ R\ a_0\ a_1 \Rightarrow MY\_RTC'\ a_0\ a_1$

The `relation` theory

Selective definitions and theorems

\HOLthm[def]{relation.reflexive_def}\hfill[reflexive_def]

$\texttt{reflexive}\ R \stackrel{\text{def}}{=} \forall x.\ R\ x\ x$ [reflexive_def]

$\texttt{symmetric}\ R \stackrel{\text{def}}{=} \forall x\ y.\ R\ x\ y \iff R\ y\ x$ [symmetric_def]

$\texttt{transitive}\ R \stackrel{\text{def}}{=} \forall x\ y\ z.\ R\ x\ y \land R\ y\ z \Rightarrow R\ x\ z$ [transitive_def]

$\texttt{equivalence}\ R \stackrel{\text{def}}{=} \texttt{reflexive}\ R \land \texttt{symmetric}\ R \land \texttt{transitive}\ R$ [equivalence_def]

$R_1 \subseteq_r R_2 \stackrel{\text{def}}{=} \forall x\ y.\ R_1\ x\ y \Rightarrow R_2\ x\ y$ [RSUBSET]

$(R_1 \cup_r R_2)\ x\ y \stackrel{\text{def}}{=} R_1\ x\ y \lor R_2\ x\ y$ [RUNION]

$(R_1 \cap_r R_2)\ x\ y \stackrel{\text{def}}{=} R_1\ x\ y \land R_2\ x\ y$ [RINTER]

$(R_1 \circ_r R_2)\ x\ z \stackrel{\text{def}}{=} \exists y.\ R_2\ x\ y \land R_1\ y\ z$ [O_DEF]

$\vdash R_1 \circ_r R_2 \circ_r R_3 = (R_1 \circ_r R_2) \circ_r R_3$ [O_ASSOC]

$\vdash (\forall x.\ R^*\ x\ x) \land \forall x\ y\ z.\ R\ x\ y \land R^*\ y\ z \Rightarrow R^*\ x\ z$ [RTC_RULES]

$\vdash (\forall x.\ R^*\ x\ x) \land \forall x\ y\ z.\ R^*\ x\ y \land R\ y\ z \Rightarrow R^*\ x\ z$ [RTC_RULES_RIGHT1]

$\vdash R^*\ x\ y \Rightarrow \forall z.\ R^*\ y\ z \Rightarrow R^*\ x\ z$ [RTC_RTC]

Homework

Start writing your project report in LaTeX, with (at least) *title*, *abstract* and some of your currently finished HOL theorems embedded.