The OpenTheory Standard Theory Library

Joe Hurd

Galois, Inc.
joe@gilith.com

Galois Tech Talk
Tuesday 12 April 2011

galois

Theory Engineering

- Interactive theorem proving is growing up.
  - The FlySpeck project is driving the HOL Light theorem prover towards a formal proof of the Kepler sphere-packing conjecture.
  - The seL4 project recently completed a 20 man-year verification of an operating system kernel in the Isabelle theorem prover.
- There is a need for theory engineering techniques to support these major verification efforts.
  - Theory engineering is to proving as software engineering is to programming.
  - *"Proving in the large."*
  - *"Mixed language proving."*

OpenTheory Project

- In theory, mathematical proofs are immortal.
- In practice, proofs that depend on theorem prover implementations bit-rot at an alarming rate.
- **Idea:** Archive proofs as theory packages.
- The goal of the OpenTheory project is to transfer the benefits of package management to logical theories.[1]
- **Slogan:** *"Logic is an ABI for mathematics."*

[1] OpenTheory was initiated in 2004 with Rob Arthan.

OpenTheory Project Approach

- The initial case study for the project aims to port theories between three interactive theorem provers in the HOL family:
  - HOL4, HOL Light and ProofPower.
- What do they have **in common?**
  - Theorem provers in the LCF design.
  - Implement the same higher order logic: Church's simple theory of types, extended with Hindley-Milner style type variables.
- What is **different?**
  - Contain different theories (both opportunity and challenge).
  - Implement different proof tools (just challenge).

Theorem Provers in the LCF Design

- A theorem $\Gamma \vdash \phi$ states *"if all of the hypotheses Γ are true, then so is the conclusion ϕ"*.
- The novelty of Milner's Edinburgh LCF theorem prover was to make `theorem` an abstract ML type.
- Values of type `theorem` can only be created by a small logical kernel which implements the primitive inference rules of the logic.
- Soundness of the whole ML theorem prover thus reduces to soundness of the logical kernel.

HOL theorem prover $\sim$ the elephant
higher order logic $\sim$ the ball

The OpenTheory Logical Kernel

$$\frac{}{\vdash t = t}\ \text{refl}\ t \qquad \frac{}{\{\phi\} \vdash \phi}\ \text{assume}\ \phi \qquad \frac{\Gamma \vdash \phi = \psi \quad \Delta \vdash \phi}{\Gamma \cup \Delta \vdash \psi}\ \text{eqMp}$$

$$\frac{\Gamma \vdash t = u}{\Gamma \vdash (\lambda v.\ t) = (\lambda v.\ u)}\ \text{absThm}\ v \qquad \frac{\Gamma \vdash f = g \quad \Delta \vdash x = y}{\Gamma \cup \Delta \vdash f\ x = g\ y}\ \text{appThm}$$

$$\frac{\Gamma \vdash \phi \quad \Delta \vdash \psi}{(\Gamma - \{\psi\}) \cup (\Delta - \{\phi\}) \vdash \phi = \psi}\ \text{deductAntisym} \qquad \frac{\Gamma \vdash \phi}{\Gamma[\sigma] \vdash \phi[\sigma]}\ \text{subst}\ \sigma$$

$$\frac{}{\vdash (\lambda v.\ t)\ u = t[u/v]}\ \text{betaConv}\ ((\lambda v.\ t)\ u) \qquad \frac{}{\vdash c = t}\ \text{defineConst}\ c\ t$$

$$\frac{\vdash \phi\ t}{\vdash \mathit{abs}\ (\mathit{rep}\ a) = a \quad \vdash \phi\ r = (\mathit{abs}\ (\mathit{rep}\ r) = r)}\ \text{defineTypeOp}\ n\ \mathit{abs}\ \mathit{rep}\ \mathit{vs}$$

Current Practice: Porting Proof Scripts

Porting theories between theorem provers is typically carried out by manually porting proof scripts:

Code (Example HOL Light proof script)

```
let MODULAR_TO_NUM_DIV_BOUND = prove
  (`!x. modular_to_num x DIV modulus = 0`,
   GEN_TAC THEN
   MATCH_MP_TAC DIV_LT THEN
   REWRITE_TAC [MODULAR_TO_NUM_BOUND]);;
```

This is a labor-intensive process, and its success relies on the target system containing similar proof tools and dependent theories.

Alternative: Theory Packages

- **Idea:** Instead of packaging the source proof script, execute the script and record the generated primitive inference rules.
- Separates the concerns of proof search and proof storage:
  - Proof scripts often call proof tools that explore a search space.
  - Primitive inference proofs simply store the result of the search.
- **Benefit:** Primitive inference proofs do not rely on any proof tools, so are immune to bit-rot and can be read by any HOL theorem prover.
- **Drawback:** Primitive inference proofs are not human readable, so theories should be packaged only when they are stable enough to be archived and shared.

Semantic Embeddings

- Packaging theories as primitive inference rules solves the problem of differences in theorem prover proof tools.
- But how to deal with differences in the available theories?
- To successfully port a theory from theorem prover context A to B, we must find a semantic embedding $A \to B$ mapping type operators and constants in A to ones in B with properties that are at least as logically strong.
- We will need semantic embeddings from the core theories of each theorem prover in the HOL family to the core theories of the others.

Standard Theory Library

- Instead of maintaining pairwise semantic embeddings, we take the core theories and release a standard theory library of them in OpenTheory format.
- Distributes responsibility: each theorem prover maintains the semantic embeddings to and from the standard theory library.
- Serves as a published contract of interoperability:
  - *"If your theory uses only the standard theory library, we promise it will work on all of the supported theorem provers."*
- Permits dynamic linking of proofs: theorems proved in the standard theory library can be used by any theory.

Talk Plan

1. Identifying Standard Theories
2. Extracting Proofs
3. Building the Standard Theory Library
4. Summary

galois

Identifying Core Theories

- By looking at the system documentation and source code for HOL Light, HOL4 and ProofPower, we can identify a core set of theories present in each theorem prover.
- For the core theories, the semantic embeddings between the theorem provers are just renamings of the type operators and constants.
- OpenTheory implements hierarchical namespaces for type operators and constants to help avoid name clashes.

Standard Theories

Version 1.0 of the OpenTheory standard theory library contains the following set of theories:

1. Data.Bool – A theory of the boolean type
2. Data.List – A theory of list types
3. Data.Option – A theory of option types
4. Data.Pair – A theory of product types
5. Data.Sum – A theory of sum types
6. Data.Unit – A theory of the unit type
7. Function – A theory of functions
8. Number.Natural – A theory of natural numbers
9. Number.Numeral – A theory of natural number numerals
10. Relation – A theory of relations

Sourcing

- It would be possible to formalize the standard theory library from scratch using the OpenTheory primitive inference rules.
- But by definition the standard theory library is already present in each of the theorem provers.
- We can extract theories by implementing a semantic embedding from a theorem prover to OpenTheory.
- We chose HOL Light for this, having the simplest logical kernel to instrument.

Standardization

We used the following methods to standardize HOL Light theories:

1. Mapping HOL Light names of type operators and constants into the OpenTheory standard namespace.
2. Compiling HOL Light primitive inference rules to OpenTheory.
   - i.e., expressing TRANS in terms of refl, appThm and eqMp.
3. Removing HOL Light term tags.
   - e.g., Renaming *'numeral zero'* to 0.
   - e.g., Post-processing proofs to rewrite NUMERAL $t \rightarrow t$.
   - With both the above changes, natural number numerals are just bit0 and bit1 operators terminated by 0.

Such methods need to be invertible to implement a semantic embedding back from OpenTheory to HOL Light.

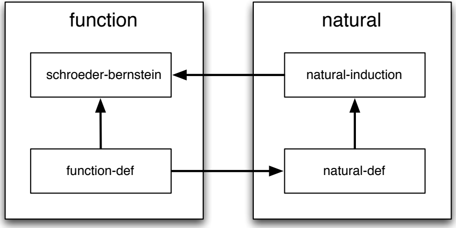
Construction Technique

We use the following procedure for converting the proofs extracted from HOL Light into the standard theory library:

1. Create a basic theory package wrapping each emitted proof.
2. Create compilation theory packages for higher-level topics, such as `bool` or `list`.
3. Create a theory package called `base` which is a compilation of the highest-level theory packages.

Compilation Theory Packages

- Compilation theory packages function as a union of theories, but do not introduce dependency cycles:



- During design we expected compilation theories to be essential for constucting a standard theory library.
- However, experimentation revealed a natural theory order:

$$\texttt{bool} < \texttt{unit} < \texttt{function} < \texttt{pair} < \texttt{natural}$$
$$< \texttt{relation} < \texttt{sum} < \texttt{option} < \texttt{list}$$

Axioms

- It is standard practice in the higher order logic theorem proving community to avoid axioms.
- An exception is made for a small set of standard axioms that are used to set up the basic theories of higher order logic.
- The OpenTheory standard theory library uses the following three axioms:
  1. **Extensionality:** $\vdash \forall t.\ (\lambda x.\ t\ x) = t$
  2. **Choice:** $\vdash \forall P, x.\ P\ x \implies P\ (\mathrm{select}\ P)$
  3. **Infinity:** $\vdash \exists f : \mathrm{ind} \rightarrow \mathrm{ind}.\ \mathrm{injective}\ f \land \neg \mathrm{surjective}\ f$

galois

Standard Theory Library

- 139 theory packages
  - = 102 basic theory packages
  - + 36 higher-level theory packages
  - + the top-level `base` theory package
- 3 axioms
- 6 defined type operators
- 64 defined constants
- 450 theorems
- `http://opentheory.gilith.com/?pkg=base-1.0`

Profiling the Standard Theory Library

Profiling the 139 theory packages of the standard theory library:

Primitive Inference	**Count**
eqMp	55,209
subst	45,651
appThm	44,130
deductAntisym	28,625
refl	17,388
betaConv	8,035
absThm	7,765
assume	2,455
axiom	1,672
defineConst	119
defineTypeOp	9
Total	**211,058**

Profiling the Standard Theory Library

What if we compress the 139 theory packages into one giant proof?

Primitive Inference	Count
eqMp	55,209
subst	45,651
appThm	44,130
deductAntisym	28,625
refl	17,388
betaConv	8,035
absThm	7,765
assume	2,455
axiom	1,672
defineConst	119
defineTypeOp	9
Total	**211,058**

Primitive Inference	Count
eqMp	32,386
subst	27,949
appThm	27,796
deductAntisym	17,300
refl	9,332
absThm	6,313
betaConv	3,646
assume	1,169
defineConst	85
defineTypeOp	7
axiom	3
Total	**125,986**

Summary

- Developing a standard theory library is essential for porting theories between theorem provers.
- It is feasible to construct a standard theory library from standardized theories extracted from a theorem prover.
- The next challenge is to package theories so that they can be exported beyond the HOL family of theorem provers.
- The project web page:

  `http://gilith.com/research/opentheory`