Translating proofs from an existing library to Logipedia

Chantal Keller

January, 22nd 2019 – Logipedia meeting

UNIVERSITÉ PARIS SUD
FACULTÉ DES SCIENCES D'ORSAY
université PARIS-SACLAY

Context

Exploratory work: import the HOL Light library in Logipedia

Get for free the existing targets: Coq, Matita, Lean, PVS, (OpenTheory), . . . and future

Context

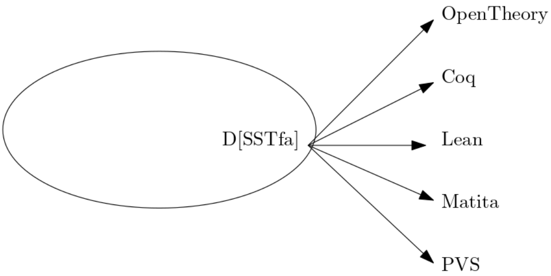
Background on HOL Light: import into Coq 10 years ago

- + "concept alignment": HOL Light constants directly mapped to their Coq counterparts
- + extensible
- • relying on proof objects older than OpenTheory
- - bad memory management

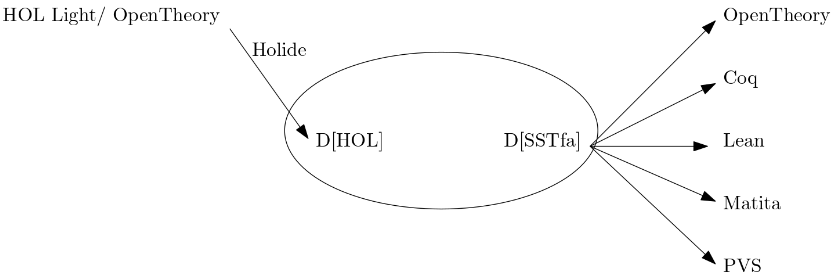
Background on dedukti/Logipedia: follower

↪ "semi-naive" point of view on importing an existing library

Starting point



Starting point



Outline

1 Introduction

2 From D[HOL] to D[STTfa]

3 Questions

Reminder: D[STT]

$$
\begin{array}{rcl}
\mathit{type} & : & \mathit{Type} \\
o & : & \mathit{type} \\
\mathit{arrow} & : & \mathit{type} \to \mathit{type} \to \mathit{type} \\
\eta & : & \mathit{type} \to \mathit{Type} \\
\Rightarrow & : & (\eta\ (\mathit{arrow}\ o\ (\mathit{arrow}\ o\ o))) \\
\forall & : & \Pi a : \mathit{type}.\ \eta\ (\mathit{arrow}\ (\mathit{arrow}\ a\ o)\ o) \\
\varepsilon & : & (\eta\ o) \to \mathit{Type}
\end{array}
$$

$$
\begin{array}{rcl}
\eta\ (\mathit{arrow}\ x\ y) & \longrightarrow & (\eta\ x) \to (\eta\ y) \\
\varepsilon\ (\Rightarrow x\ y) & \longrightarrow & (\varepsilon\ x) \to (\varepsilon\ y) \\
\varepsilon\ (\forall a\ P) & \longrightarrow & \Pi x : (\eta\ a).\ \varepsilon\ (P\ x)
\end{array}
$$

Reminder: D[STTfa] – explicit prenex type polymorphism

$$\begin{array}{rcl}
\mathit{type} & : & \mathit{Type} \\
o & : & \mathit{type} \\
\mathit{arrow} & : & \mathit{type} \to \mathit{type} \to \mathit{type} \\
\mathit{ptype} & : & \mathit{Type} \\
p & : & \mathit{type} \to \mathit{ptype} \\
\eta & : & \mathit{ptype} \to \mathit{Type} \\
\Rightarrow & : & (\eta\ (p\ (\mathit{arrow}\ o\ (\mathit{arrow}\ o\ o)))) \\
\forall & : & \Pi a : \mathit{type}.\ \eta\ (p\ (\mathit{arrow}\ (\mathit{arrow}\ a\ o)\ o)) \\
\varepsilon & : & (\eta\ (p\ o)) \to \mathit{Type} \\
\forall_t & : & (\mathit{type} \to \mathit{ptype}) \to \mathit{ptype} \\
\forall_o & : & (\mathit{type} \to \eta\ (p\ o)) \to \eta\ (p\ o)
\end{array}$$

$$\begin{array}{rcl}
\eta\ (p\ (\mathit{arrow}\ x\ y)) & \longrightarrow & (\eta\ (p\ x)) \to (\eta\ (p\ y)) \\
\varepsilon\ (\Rightarrow x\ y) & \longrightarrow & (\varepsilon\ x) \to (\varepsilon\ y) \\
\varepsilon\ (\forall a\ P) & \longrightarrow & \Pi x : (\eta\ (p\ a)).\ \varepsilon\ (P\ x) \\
\eta\ (\forall_t\ f) & \longrightarrow & \Pi a : \mathit{type}.\ \eta\ (f\ a) \\
\varepsilon\ (\forall_o\ f) & \longrightarrow & \Pi a : \mathit{type}.\ \varepsilon\ (f\ a)
\end{array}$$

D[HOL]

Differences:

- implicit prenex type polymorphism (relies on dedukti's Π)
- Q_0: connectives defined in terms of $=$ instead of $(\forall, \Rightarrow)$
- ~ 10 deduction rules, no conversion
- functional and propositional extensionality, Hilbert's choice operator
- all objects of type *type* are inhabited

Work in progress: small prototype

Translation for the subset included in STTfa:

- implicit → explicit polymorphism
- connectives: =, translated into Leibniz equality expressed in STTfa
- ∼ half of the deduction rules
- tested on a handmade example

My approach:

- realize D[HOL] using D[STTfa]
- back-port it as an AST manipulation (could we avoid it?)

To do

- Q_0: instrument OpenTheory to rely on "standard" (and intuitionistic) natural deduction rules (and conversion?)
- functional and propositional extensionality, Hilbert's choice operator: when unavoidable, depend on axioms (that the target system may or may not realize)
- think about the fact that all objects of type *type* are inhabited

$\hookrightarrow$ support any HOL Light development

Practical implementation

Logipedia's code:

- short and clean
- a bit more documentation would be helpful

Relies on dedukti's code:

- very well documented

Running everything:

- requires quite a bit of installation (MongoDB, PHP, . . .)

STTfa:

- well suited for (most of) HOL as presented in HOL Light/OpenTheory
- missing: type aliases (currently unfolded)

Outline

1 Introduction

2 From D[HOL] to D[STTfa]

3 Questions

1. Related to HOL Light/OpenTheory

Which libraries to import? Flyspeck, others? (reals?)

Classical version of connectives? What about other non-computational aspects?

Use some translation instead?

2. General

Some engineering to require only a realization of the source system using STTfa?

Which logic/systems/libraries could be useful for Logipedia?

Is the process (source → dedukti → STTfa → target) well suited to import a sufficient subset?

Interoperability inside the same system?