Transforming Proof

AI4FM 2011

Iain Whiteside

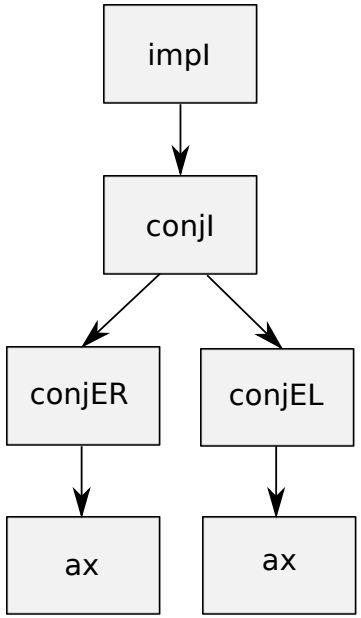
DReaM Group

April 26, 2011

A 'low-level' proof

Proof tree of $\vdash P \wedge Q \to Q \wedge P$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{Q \vdash Q}\,ax}{P \wedge Q \vdash Q}\,conjE_R \qquad \dfrac{\dfrac{}{P \vdash P}\,ax}{P \wedge Q \vdash P}\,conjE_L}{P \wedge Q \vdash Q \wedge P}\,conjI}{\vdash P \wedge Q \to Q \wedge P}\,impI$$



We can describe syntactically as:

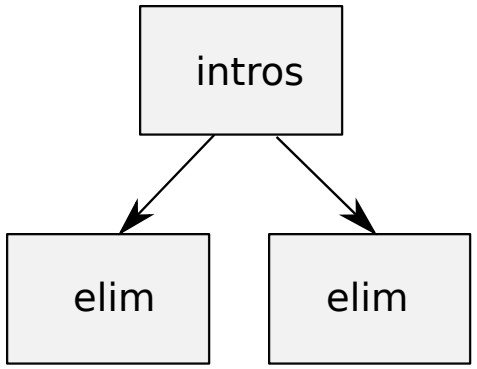
$$impI \;;\; conjI \;;\; (conjE_R \;;\; ax) \otimes (conjE_L \;;\; ax)$$

Tactics

$$intros := REPEAT(impI \mid conjI)$$

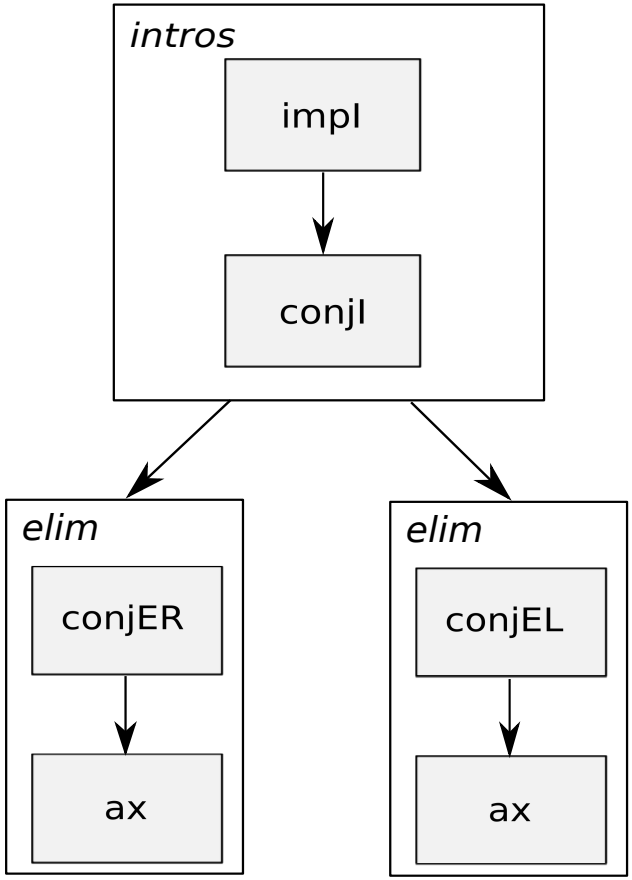
$$elim := (conjE_L \mid conjE_R)\ ;\ ax$$

$$\dfrac{\dfrac{}{P \wedge Q \vdash Q}\, elim \qquad \dfrac{}{P \wedge Q \vdash P}\, elim}{\vdash P \wedge Q \rightarrow Q \wedge P}\, intros$$



Same underlying proof – higher level view.

Hiproofs



- Hierarchical representation
- Zoom in/out
- Tactic applications introduce nesting
- Labels described syntactically as:
  $([intros]impI\ ;\ conjI)\ \ ;$
  $([elim]conjE_R\ ;\ ax)\ \otimes$
  $([elim]conjE_L\ ;\ ax)$
- Can be constructed using Hitac tactics.

A 'declarative' lemma

lemma $conj\_comm: \ \vdash P \wedge Q \rightarrow Q \wedge P$
proof($impI$**)**
 show $\{P \wedge Q\} \vdash Q \wedge P$
 proof($conjI$**)**
  show $q: \ \{P \wedge Q\} \ \vdash \ Q$ **by** $(conjE_R \ ; \ ax)$
  show $p: \ \{P \wedge Q\} \ \vdash \ P$ **by** $(conjE_L \ ; \ ax)$
 qed
qed

A ‘declarative’ lemma with tactics

lemma $conj\_comm : \ \vdash P \wedge Q \rightarrow Q \wedge P$
proof($intros$**)**
 show $q : \ \{P \wedge Q\} \ \vdash \ Q$ **by** $elim$
 show $p : \ \{P \wedge Q\} \ \vdash \ P$ **by** $elim$
qed

A ‘procedural’ lemma

lemma $conj\_comm : \vdash P \wedge Q \rightarrow Q \wedge P$
by $intros\ ;\ elim\ \otimes\ elim$

- In HOL Light or Coq: *intros* THEN [*elim, elim*]
- Gradually higher level syntactic proofs.

A 'forward' lemma

lemma $conj\_comm: \ \vdash P \wedge Q \longrightarrow Q \wedge P$
proof($impI$**)**
 have $q: \ \{P \wedge Q\} \ \vdash \ Q$ **by** $(conjE \ ; \ ax)$
 have $p: \ \{P \wedge Q\} \ \vdash \ P$ **by** $(conjE \ ; \ ax)$
 from $q \ p$ **show** $\{P \wedge Q\} \vdash Q \wedge P$ **by** $conjI$
qed

Proof construction and understanding

- Procedural proofs are easier to write, but difficult to understand statically;
- Declarative proofs take longer, but are standalone;
- Backward vs. forward depends on context and user;
- For different readers, different parts are interesting/boring.

Use Hiproofs as an underlying representation to aid syntactic transformations.

A proof language: syntax

A fragment:

$$
\begin{array}{lcl}
prf & ::= & \textbf{proof(}\ rule\ \textbf{)} \\
 & & \quad stmt^* \\
 & & \textbf{qed} \\
 & & \\
rule & ::= & t
\end{array}
\qquad
\begin{array}{lcl}
stmt & ::= & \langle\rangle \\
 & | & \textbf{show}\ (name :)^?\ goal\ prf \\
 & | & \textbf{have}\ name :\ goal\ prf \\
 & | & \textbf{from}\ name^*\ \textbf{show}\ goal\ \textbf{by}\ rule
\end{array}
$$

t is the tactic language seen earlier:

$$t\ ::=\ \ldots\ \mid\ t\,;\ t\ \mid\ t \otimes t\ \mid\ \ldots$$

Tactics act on lists of goals and return lists of goals and Hiproofs.

Semantics

Acts on lists of goals g and constructs Hiproofs s, using a *context* of tactics/lemmas:

$$\langle g, prf \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s \rangle.$$

Example rules:

$$\frac{\langle [\gamma], t \rangle \Downarrow^{t}_{(\mathcal{T},\mathcal{L})} \langle s_1, g \rangle \qquad \langle g, stmts \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s_2 \rangle}{\langle [\gamma], \textbf{proof(}t\textbf{)}\ stmts\ \textbf{qed} \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s_1\ ;\ s_2 \rangle} \quad \text{(B-Prf-Block)}$$

$$\frac{\langle [\gamma], prf \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s_1 \rangle \qquad \langle g, stmts \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s_2 \rangle}{\langle \gamma :: g, \textbf{show}\ name\text{:}\ \gamma\ prf\ stmts \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s_1 \otimes s_2 \rangle} \quad \text{(B-Prf-Show)}$$

$$\frac{\langle [\gamma], prf \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s_1 \rangle \qquad \langle g, stmts \rangle \Downarrow_{(\mathcal{T},\mathcal{L}[name \longmapsto (\gamma, s_1)])} \langle s \rangle}{\langle g, \textbf{have}\ name\text{:}\ \gamma\ prf\ stmts \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s \rangle} \quad \text{(B-Prf-Have)}$$

Example proof

lemma $conj\_comm : \vdash P \wedge Q \rightarrow Q \wedge P$
proof($intros$**)**
 show $q : \{P \wedge Q\} \vdash Q$ **by** $elim$
 show $p : \{P \wedge Q\} \vdash P$ **by** $elim$
qed

- The **by** *rule* is syntactic sugar for **proof(**$rule$**)** $\langle\rangle$ **qed**;
- We can see that the semantics constructs the Hiproof shown earlier;
- Hiproof: $([intros]impI \ ; \ conjI) \ ; \ ([elim]conjE_R \ ; \ ax) \otimes ([elim]conjE_L \ ; \ ax)$
- Haven't shown tactic evaluation semantics.

Proof Transformations

A *proof transformation*, P is a map such that if

$$\langle g, prf \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s \rangle$$

then

$$\langle g, P(prf) \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s \rangle.$$

Actually, this is a little too specific, we would rather have:

$$\langle g, P(prf) \rangle \Downarrow_{(\mathcal{T},\mathcal{L})} \langle s' \rangle \quad where \quad s \equiv s'$$

for some equivalence relation.

Backward to Forward

proof(t **)**
  show $goal1$: γ_1 prf_1
  $\vdots$
  show $goaln$: γ_n prf_n
qed

proof
  have $goal1$: γ_1 prf_1
  $\vdots$
  have $goaln$: γ_n prf_n
  from $goal1$ $\ldots$ $goaln$ **show** γ **by** t
qed

$$\frac{stmts \xrightarrow{b2f} stmts' \qquad [n_1, \ldots, n_n] = shows(stmts)}{\textbf{proof(}t\textbf{)}\ stmts\ \textbf{qed} \xrightarrow{b2f} \textbf{proof}\ stmts'\ \textbf{from}\ n_1\ \ldots\ n_n\ \textbf{show}\ \gamma\ \textbf{by}\ t\ \textbf{qed}}$$

$$\frac{stmts \xrightarrow{b2f} stmts'}{\textbf{show}\ name\text{:}\ \gamma\ prf\ \ stmts \xrightarrow{b2f} \textbf{have}\ name\text{:}\ \gamma\ prf\ \ stmts'}$$

Declarative to Procedural

We map a proof to a tactic. For instance:

$$\frac{stmts \xrightarrow{d2p} stmts'}{\textbf{proof(}t\textbf{)}\ stmts\ \textbf{qed} \xrightarrow{d2p} t\ ;\ stmts'}$$

$$\frac{stmts \xrightarrow{d2p} stmts' \qquad prf \xrightarrow{d2p} t}{\textbf{show}\ name\colon \gamma\ prf\ \ stmts \xrightarrow{d2p}\ t \otimes ID\ \ ;\ stmts'}$$

- As **show** only works on the head of a list of open goals, *ID* is used to skip the rest.

- We can then normalise the tactic to remove the identities.

Procedural to Declarative

• Less straightforward: we need to execute the tactic and transform the resulting Hiproof (because of alternation, repetition);

• We can transform the tactic into a form t ; $(t_1 \otimes \ldots \otimes t_n)$, which maps to:

proof(t **)**
  show $goal1$: γ_1 **by** t_1
  ⋮
  show $goaln$: γ_n **by** t_n
qed

Further transformations

There are many more transformations possible:

- Extract a subproof: create a named lemma from a proof block; replace the original block with a call to lemma;

- Fold tactic: take a portion of a proof and create a new named lemma from it;

- Remove assumptions: replace calls to that application with *gaps*.

Conclusions

- Motivated proof transformations;
- Described Hiproofs as a proof representation;
- Described how to turn a backward-style proof forward;
- Folding ‘boring’ parts of a proof;
- Unfolding ‘confusing parts’.