

Heg: A Coq library for Heterogeneous Equality

Chung-Kil Hur

09 / Jul / 2010

@ Coq Workshop 2010

Dependent types

- One of the main strengths of Coq
 - : Dependent types
- However, sometimes dealing with dependent types is a big overhead due to intentional type theory.
- We develop Heq to reduce such an overhead.

Motivating example

- Inductive vector : nat \rightarrow Type :=
 - | vnil : vector 0
 - | vcons (hd:nat) n (tl:vector n) : vector (S n)
- ++ : $\forall n m, \text{vector } n \rightarrow \text{vector } m \rightarrow \text{vector } (n+m)$

Problem with equality

$$? \quad \forall n, (v: \text{vector } n), \quad v ++ \text{vnil} = v$$

\downarrow
 \downarrow
vector (n+0)
vector n

$$? \quad \forall n, (v: \text{vector } n), \quad v ++ \text{vnil} = \text{eq_rect_vector } v \text{ (plus } n)$$

\downarrow
 vector n

\downarrow
 proof of
 $n+0 = n$

$$? \quad \forall n, (v: \text{vector } n), \quad \exists \text{Meq } (v ++ \text{vnil}) v$$

$$\approx (\text{vector } n, v) = (\text{vector } (n+0), v ++ \text{vnil}) : \prod_{ty \in \text{Type}} ty$$

$$? \quad \forall n, (v: \text{vector } n), \quad \text{eq_dep } \underline{\text{vector}} (v ++ \text{vnil}) v$$

$$\approx (n, v) = (n+0, v ++ \text{vnil}) : \prod_{n \in \mathbb{N}} (\text{vector } n)$$

Heg: equality

We use dependent product.

Notation: $\{\{ \text{vector } \# v == v' \}\}$

def $(n, v) = (n', v') : \prod_{n \in \mathbb{N}} \text{vector } n$

? $\forall n, v : \text{vector } n, \{\{ \text{vector } \# v ++ vnil == v \}\}$

this can be inferable using type classes

"Matthieu Sozeau"

Heg: casts

Notation: $\langle\langle \text{vector} \# C \rangle\rangle v \stackrel{\text{def}}{=} \text{eq_rect_vector } v _ C$

E.g.) Program Fixpoint $\text{vrev } n \ (v: \text{vector } n) : \text{vector } n$

= match v with

| $v\text{nil}$ \Rightarrow $v\text{nil}$

| $v\text{cons } hd \ tl$ \Rightarrow $\langle\langle \text{vector} \# _ \rangle\rangle \text{rev } tl ++ (hd :: v\text{nil})$

end.

Next obligation. ---. Defined.

rewriting of heterogeneous equality

$$\{\{ \text{vector} \# v ++ \text{vnil} == v \}\}$$

- Inductive Case

$$\vdots$$
$$\text{IH}_v : \{\{ \text{vector} \# v ++ \text{vnil} == v \}\}$$

$$\{\{ \text{vector} \# \text{hd} :: (v ++ \text{vnil}) == \text{hd} :: v \}\}$$

Demonstration

Algorithm

$$\{ \{ v \text{ ++ } v_{\text{nil}} == v \} \} \quad v \text{ ++ } v_{\text{nil}} = \langle \langle \text{pf} \rangle \rangle v \quad : \text{vector } (n+0)$$

$$\begin{array}{ccc} \swarrow & & \searrow \\ \text{pf} : n+0 = n & & \end{array} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} \text{rewrite}$$

$$\text{hd} :: (v \text{ ++ } v_{\text{nil}}) == \text{hd} :: v \quad \rightarrow \quad \text{hd} :: \langle \langle \text{pf} \rangle \rangle v == \text{hd} :: v$$

↓ generalize

$$\forall c : n+0 = n, \quad \text{hd} :: \langle \langle c \rangle \rangle v == \text{hd} :: v$$

↓ rewrite $n+0 = n$

$$\forall c : n = n, \quad \text{hd} :: \langle \langle c \rangle \rangle v == \text{hd} :: v$$

||
v

elimination
of casts

Elimination & relocation of casts

Demonstration

hpattern

$n, m : \text{nat}$

$p : \forall n m l, \text{vector } n \rightarrow \text{vector } m \rightarrow \text{Prop}$

$H : S \ n = m$

$\forall (v : \text{vector } (S \ n)) (w : \text{vector } n), P \ (S \ n) \ (S \ n) \ (S \ n) \ V \ (0 :: w)$

$\text{vector}(S \ n) \ \text{vector}(S \ n)$

rewrite $H \rightsquigarrow$ fail

pattern $(S \ n) \rightsquigarrow$ fail

pattern $n \ (S \ n) \ \text{at } 1, 2 \rightarrow$ succeed

hrewrite $H \quad \text{at } 1 \rightarrow$ succeed
or hpattern $(S \ n) \ \text{at } 1$

\rightarrow sound & complete &
linear & easy

hpattern : algorithm

hpattern (S n) at 1

$\forall (v:\text{vector } (S\ n)) (w:\text{vector } n), P\ (S\ n)\ (S\ n)\ (S\ n)\ v\ (0::w)$

↓

$\forall (v:\text{vector } \mathbb{R}) (w:\text{vector } n), P\ _ _ _ v\ (0::w)$

↓ type check : fail

cannot infer

↓

$\forall (v:\text{vector } \mathbb{R}) (w:\text{vector } n), P\ _ _ _ v\ (0::w)$

↓ type check

↑ fill (S n)

$\forall (v:\text{vector } \mathbb{R}) (w:\text{vector } n), P\ \mathbb{R}\ (S\ n)\ (S\ n)\ v\ (0::w)$

↑ infer ↑ infer ↑ I provided

Summary

- Heq :- simple notations for heterogeneous equality & casts
- elimination, relocation of casts \rightarrow rewriting of heq
 - hpattern
 - iterated dependent pairs

See <http://www.pps.jussieu.fr/~gil/Heq> for more details