generic derivations for functions

Johan G. F. Belinfante
2004 May 17

In[1]:= <<goedel57.16a; << tools.m

Package Title: goedel57.16a 2004 May 16 at 10:05 p.m.

It is now: 2004 May 17 at 14:53

Loading Simplification Rules

TOOLS.M Revised 2004 May 14

weightlimit = 40

summary

In this notebook a general method is presented that can serve as a template for proofs that certain membership rules yield functions. The idea is based on the technique used recently for the specific case of the function \( \text{EQUIV}=\lambda x.f[x] \), but the method should work for any unary constructor \( f[x] \). The key new rewrite rule that makes the present method work is only recently added to the GOEDEL program:

\[
\text{In}[2]:= \text{member}[\text{pair}[x, y], \text{composite}[\text{Di}, z]]
\]
\[
\text{Out}[2]= \text{and}[\text{member}[y, V], \text{not}[\text{subclass}[\text{image}[z, \text{singleton}[x]], \text{singleton}[y]]]]
\]

a generic definition for functions

Given a functor \( f \), the following membership rule can be used to define the function \( L[f] = \lambda x. f[x] \):

\[
\text{In}[3]:= \text{member}[x_, L[f_]] := \text{and}[\text{member}[\text{first}[x], V], \text{equal}[	ext{second}[x], f[\text{first}[x]]]]
\]

In practice if may sometimes be useful to replace the first literal on the right side with the more general condition \( \text{member}[\text{first}[x], y] \), which has little effect on the overall procedure. The first step is to derive that fact that \( L[f] \) is a relation. This can be done in two steps as follows:

\[
\text{In}[4]:= \text{Map}[	ext{equal}[0, \#] \&, \text{dif}[L[f], \text{cart}[V, V]]] // \text{Normality}
\]
\[
\text{Out}[4]= \text{subclass}[L[f], \text{cart}[V, V]] = \text{True}
\]

\[
\text{In}[5]:= \text{subclass}[L[f_], \text{cart}[V, V]] := \text{True}
\]

\[
\text{In}[6]:= \text{equal}[	ext{composite}[\text{Id}, L[f]], L[f]]
\]
\[
\text{Out}[6]= \text{True}
\]

\[
\text{In}[7]:= \text{composite}[\text{Id}, L[f_]] := L[f]
\]
The second step is to derive a vertical section rule:

\[ \text{In[8]} := \text{image}[L[f], \text{singleton}[x]] \quad /\quad \text{Normality} \]

\[ \text{Out[8]} = \text{image}[L[f], \text{singleton}[x]] = \text{intersection}[\text{image}[V, \text{singleton}[x]], \text{singleton}[f[\text{union}[x, \text{complement}[\text{image}[V, \text{singleton}[x]]]]]]] \]

\[ \text{In[9]} := \text{image}[L[f_], \text{singleton}[x_]] := \text{intersection}[\text{image}[V, \text{singleton}[x]], \text{singleton}[f[\text{union}[x, \text{complement}[\text{image}[V, \text{singleton}[x]]]]]]] \]

With these rules in place one quickly derives the fact that \( L[f] \) is a function:

\[ \text{In[10]} := \text{Map}[[\text{equal}[0, #] \&, \text{dif}[L[f], \text{funpart}[L[f]]]] /\quad \text{RelnNormality}] \]

\[ \text{Out[10]} = \text{FUNCTION}[L[f]] = \text{True} \]

\[ \text{In[11]} := \text{FUNCTION}[L[f_]] := \text{True} \]

The reification of \( f \) is the class

\[ \text{In[16]} := \text{class}[\text{pair}[x_, y_], \text{member}[y_, f_[x_]]] := R[f] \]

The function \( L[f] \) can be identified as its VERTSECT.

\[ \text{In[23]} := L[f] /\quad \text{VSNormality} /\quad \text{Reverse} \]

\[ \text{Out[23]} = \text{VERTSECT}[R[f]] = L[f] \]

\[ \text{In[24]} := \text{VERTSECT}[R[f_]] := L[f] \]

\[ \text{In[27]} := \text{SubstTest}[\text{fix}, \text{composite}[\text{LB}[x], \text{VERTSECT}[x]], \text{x} \rightarrow R[f]] \]

\[ \text{Out[27]} = \text{fix}[\text{composite}[\text{LB}[R[f]], L[f]]] = \text{domain}[L[f]] \]

\[ \text{In[28]} := \text{fix}[\text{composite}[\text{LB}[R[f_]], L[f_]]] := \text{domain}[L[f]] \]

The domain of the function \( L[f] \) can be computed as the class of all sets \( x \) for which \( f[x] \) is a set:

\[ \text{In[29]} := \text{class}[x, \text{member}[f[x], V]] \]

\[ \text{Out[29]} = \text{domain}[L[f]] \]

---

**an example**

In this section, the general theory is illustrated with the special case \( f = \text{trv} \).

\[ \text{In[19]} := \text{FUNCTION}[L[\text{trv}]] \]

\[ \text{Out[19]} = \text{True} \]

The reification rules suffice to compute \( R[\text{trv}] \).
In[36]:= reify[x, trv[x]]

Out[36]= \text{composite}[\text{inverse}[E], \text{HULL}[TRV], \text{inverse}[S]]

This result can be independently verified:

In[22]:= \text{SubstTest}[\text{class}, \text{pair}[x, y], \text{member}[y, f[x]], f \rightarrow \text{trv}] // \text{Reverse}

Out[22]= R[trv] = \text{composite}[\text{inverse}[E], \text{HULL}[TRV], \text{inverse}[S]]

In[30]:= R[trv] := \text{composite}[\text{inverse}[E], \text{HULL}[TRV], \text{inverse}[S]]

In[32]:= \text{SubstTest}[\text{VERTSECT}, R[f], f \rightarrow \text{trv}] // \text{Reverse}

Out[32]= L[trv] = \text{composite}[\text{HULL}[TRV], \text{IMAGE}[\text{id}[\text{cart}[V, V]]]]

In[33]:= L[trv] := \text{composite}[\text{HULL}[TRV], \text{IMAGE}[\text{id}[\text{cart}[V, V]]]]

In this case, the domain is already known to be $V$.

In[35]:= \text{class}[x, \text{member}[\text{trv}[x], V]] = \text{domain}[L[trv]]

Out[35]= \text{True}