classes of squares

Johan G. F. Belinfante  
2011 May 25

In[1]:= SetDirectory["l:"]; << goedel.11may24a

:Package Title: goedel.11may24a 2011 May 24 at 10:00 a.m.
Loading takes about ten minutes, half that time due to builtin pauses.
It is now: 2011 May 25 at 14:24

Loading Simplification Rules

TOOLS.M is now incorporated in the GOEDEL program as of 2010 September 3
weightlimit = 40
Loading completed.
It is now: 2011 May 25 at 14:35

summary

Rewrite rules are derived for classes of the form \texttt{image[CART, id[x]]}. These are classes whose members are cartesian squares.

\begin{verbatim}
In[3]:= equiv[member[s, image[CART, id[x]]],
        and[equal[s, cart[fix[s], fix[s]]], member[fix[s], x]]]
\end{verbatim}

\texttt{Out[3]= True}

Several rewrite rules involving \texttt{fix[image[inverse[CART], x]]} remain valid when this expression is replaced with the closely related expression \texttt{image[IMAGE[inverse[DUP]], x]}. As an application, a rewrite rule is derived for the class of domains of sub-binary operations of a given one.

derivation

Theorem. An inclusion.

\begin{verbatim}
In[6]:= SubstTest[implies, subclass[u, v], subclass[image[u, x], image[v, x]],
                 {u \rightarrow inverse[composite[CART, DUP]], v \rightarrow FIX}] // Reverse
\end{verbatim}

\texttt{Out[6]= subclass[fix[image[inverse[CART], x]], image[IMAGE[inverse[DUP]], x]] := True}

\begin{verbatim}
In[7]:= subclass[fix[image[inverse[CART], x_]], image[IMAGE[inverse[DUP]], x_]] := True
\end{verbatim}
Theorem. An alternate formula for the class of sides of a class of squares.

\[% \text{subclass}[x_, \text{image}\left[\text{IMAGE}\left[\text{inverse}\left[\text{DUP}\right]\right]\right], x_, y_] := \text{True} \]

Lemma. A temporary rewrite rule.

\[% \text{subclass}[x_, \text{image}[\text{CART}, \text{id}], x_, y_] := \text{subclass}[x, \text{image}[\text{CART}, \text{id}]] \]

Theorem. A simplification rule.

\[% \text{equal}[x, \text{image}[\text{CART}, \text{id}], x] := \text{subclass}[x, \text{image}[\text{CART}, \text{id}]] \]

Theorem. An alternate formula for the class of sides of a class of squares.

\[% \text{subclass}[\text{image}[\text{CART}, \text{id}], x, y] := \text{subclass}[\text{image}[\text{CART}, \text{id}], x, y] \]

Lemma. A temporary rewrite rule.

\[% \text{equal}[x, \text{image}[\text{CART}, \text{id}], x] := \text{subclass}[x, \text{image}[\text{CART}, \text{id}]] \]
In[18]:= equiv[equal[x, image[CART, id[image[IMAGE[Inverse[DUP]], x]]]], subclass[x, image[CART, Id]]]

Out[18]= True

In[19]:= equal[x_, image[CART, id[image[IMAGE[Inverse[DUP]], x_]]]] := subclass[x, image[CART, Id]]

---

an application

An application to the theory of binary operations is made in this section.

Theorem. The domain of any binary operation is a cartesian square.

In[20]:= SubstTest[implies, subclass[u, v], subclass[image[t, u], image[t, v]], (t -> IMAGE[FIRST], u -> Intersection[BINOPS, x], v -> BINOPS)] // Reverse

Out[20]= subclass[image[IMAGE[FIRST], Intersection[BINOPS, x]], image[CART, Id]] = True

In[21]:= subclass[image[IMAGE[FIRST], Intersection[BINOPS, x_]], image[CART, Id]] := True

A more precise result is derived below for the special case that x is replaced with P[binop[x]].

Lemma. A temporary rewrite rule.

In[22]:= Map[equal[V, #] & , union[complement[binclosed[binop[x]]]], image[Inverse[IMAGE[id[fix[domain[binop[x]]]]]]], binclosed[binop[x]]]]] // Normality

Out[22]= subclass[image[IMAGE[id[fix[domain[binop[x]]]]]], binclosed[binop[x]]], binclosed[binop[x]]]] = True

In[23]:= (% /. x -> x_) /. Equal -> SetDelayed

Theorem. A better rewrite rule.

In[24]:= equal[image[IMAGE[id[fix[domain[binop[x]]]]]], binclosed[binop[x]]], intersection[P[fix[domain[binop[x]]]], binclosed[binop[x]]]]]

Out[24]= True

In[25]:= image[IMAGE[id[fix[domain[binop[x]]]]]], binclosed[binop[x_]]]] := intersection[binclosed[binop[x]], P[fix[domain[binop[x]]]]]]

Theorem. (A temporary rule subsumed by another rewrite rule derived below.)

In[26]:= Map[image[FIX, #] & , ImageComp[IMAGE[FIRST], IMAGE[composite[id[binop[x]], inverse[FIRST]]], image[CART, id[binop[x]]]]]] // Reverse

Out[26]= image[IMAGE[Inverse[DUP]], image[IMAGE[FIRST], intersection[BINOPS, P[binop[x]]]]]] = intersection[binclosed[binop[x]], P[fix[domain[binop[x]]]]]]
Lemma. A simplification rule.

Theorem. The class of domains of sub-binary operations of a given binary operation is the class of cartesian squares of subsets of the fixed-point set of the domain of the given binary operation and sets that are binary closed under that binary operation.

It is not clear how best to orient the following rewrite rule. The present choice should be regarded as tentative.

The choice of orientation of the rewrite rule made above has the advantage that it avoids having to add a separate rule for the following fact.

In addition, the above choice has the advantage that it makes it unnecessary to retain the rewrite rule derived above for the more complicated quantity `image[IMAGE[inverse[DUP]], image[IMAGE[FIRST], intersection[BINOPS, P[binop[x]]]]]`. So the net effect is that one can get by with fewer rewrite rules if the chosen orientation is adopted.