binary isomorphisms

Johan G. F. Belinfante
2006 December 29

In [1]:= SetDirectory["1:"]; << goedel88.26a; << tools.m

:Package Title: goedel88.26a 2006 December 26 at 3:50 p.m.

It is now: 2006 Dec 29 at 12:54

Loading Simplification Rules

TOOLS.M Revised 2006 December 17

weightlimit = 40

summary

If a binary homomorphism of binary operations is one-to-one and onto, then its inverse is also a binary homomorphism.

derivation

The main idea is pretty simple, but some cleaning up will be required.

In [2]:= SubstTest[implies, equal[u, v],
    equal[composite[t, u, cross[t, t]], composite[t, v, cross[t, t]]],
    {u -> composite[oopart[w], x], v -> composite[y, cross[oopart[w], oopart[w]]],
    t -> inverse[oopart[w]]}] // Reverse

Out[2]= or[equal[
    composite[id[domain[oopart[w]]], x, cross[inverse[oopart[w]], inverse[oopart[w]]]],
    composite[inverse[oopart[w]], y, id[cart[range[oopart[w]], range[oopart[w]]]]],
    not[equal[composite[y, cross[oopart[w], oopart[w]]], composite[oopart[w], x]]] = True

In [3]:= (% /. {w \[RightTarrow] w_, x \[RightTarrow] x_, y \[RightTarrow] y_}) /. Equal \[RightTarrow] SetDelayed

The oopart wrapper can be removed.
The factor \texttt{id}[\texttt{domain}[\texttt{w}]] can now be removed:

\begin{verbatim}
In[8]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3],
                 implies[p1, p6], implies[and[p2, p3, p4], p5],
                 implies[and[p1, p2], p5]],
       (p1 -> member[\texttt{w}, binhom[\texttt{x}, \texttt{y}]], member[\texttt{x}, BINOPS],
        p2 -> FUNCTION[\texttt{w}], p3 -> equal[\texttt{domain}[\texttt{w}], \texttt{x}, \texttt{y}],
        p4 -> SUBCLASS[range[\texttt{x}], \texttt{cross}[\texttt{x}, \texttt{y}]],
        p5 -> SUBCLASS[\texttt{domain}[\texttt{w}], \texttt{domain}[\texttt{w}]]] // Reverse
Out[8]= or[equal[\texttt{composite}[\texttt{x}, \texttt{cross}[\texttt{inverse}[\texttt{w}], \texttt{inverse}[\texttt{w}]])],
       \texttt{composite}[\texttt{inverse}[\texttt{w}], \texttt{y}, \texttt{id}[\texttt{cart}[\texttt{range}[\texttt{w}], \texttt{range}[\texttt{w}]])],
       \texttt{not}[\texttt{FUNCTION}[\texttt{inverse}[\texttt{w}]]],
       \texttt{not}[\texttt{member}[\texttt{w}, \texttt{binhom}[\texttt{x}, \texttt{y}]], \texttt{member}[\texttt{x}, \texttt{BINOPS}]]

In[9]:= (% /\{\texttt{w} -> \texttt{w}_\texttt{m}, \texttt{x} -> \texttt{x}_\texttt{m}, \texttt{y} -> \texttt{y}_\texttt{m}\}) / Equal \rightarrow \texttt{SetDelayed}
\end{verbatim}

Main theorem. If a binary homomorphism of binary operations is one-to-one and onto, then its inverse is also a binary homomorphism. Comment. The execution time of this derivation was cut in half by omitting the following steps of the complete proof: implies[p1, p4], implies[and[p1, p4], p5], implies[p1, p6], implies[and[p3, p7], p8]. The rewrite rules of the GOEDEL program automatically supply these omitted steps.
In[10]:= Map[not, SubstTest[and, implies[p1, p2],
        implies[and[p1, p2], p3], implies[and[p1, p5, p6], p7], not[implies[p1, p8]],
        {p1 -> member[x, BINOPS], member[y, BINOPS], member[w, binhom[x, y]],
         FUNCTION[inverse[w]], equal[range[w], fix[domain[y]]]},
        p2 -> member[w, map[fix[domain[x]], fix[domain[y]]]],
        p3 -> member[inverse[w], map[fix[domain[y]], fix[domain[x]]]],
        p4 -> equal[composite[w, x], composite[y, cross[w, w]]],
        p5 -> equal[composite[x, cross[inverse[w], inverse[w]]],
         composite[inverse[w], y, id[cart[range[w], range[w]]]]],
        p6 -> equal[domain[y], cartsq[fix[domain[y]]]],
        p7 ->
        equal[composite[x, cross[inverse[w], inverse[w]]], composite[inverse[w], y]],
        p8 -> member[inverse[w], binhom[y, x]]]]] // Reverse

Out[10]= or[member[inverse[w], binhom[y, x]], not[equal[fix[domain[y]], range[w]]],
    not[FUNCTION[inverse[w]]], not[member[w, binhom[x, y]]],
    not[member[x, BINOPS]], not[member[y, BINOPS]]] = True

In[12]:= or[member[inverse[w], binhom[y, x]], not[equal[fix[domain[y]], range[w]]],
    not[FUNCTION[inverse[w]]], not[member[w, binhom[x, y]]],
    not[member[x, BINOPS]], not[member[y, BINOPS]]] = True