composite of a unary operation with itself

Johan G. F. Belinfante and Lee Martie
2006 June 27

```
In[1]:= SetDirectory["l:"]; << goedel82.22a; << tools.m

        :Package Title: goedel82.22a         2006 June 22 at 10:30 p.m.

        It is now:  2006 Jun 27 at 16:46

        Loading Simplification Rules

        TOOLS.M                       Revised 2006 June 21

        weightlimit = 40
```

summary

If **x** is a unary operation, then so is **composite[x, x]**. The derivation is a good illustration of the processes of eliminating and introducing variables.

derivation

The starting point is a result with variables:

```
In[2]:= SubstTest[implies, and[member[x, map[y, y]], member[z, map[y, y]]],
          member[composite[x, z], map[y, y]], z → x]

Out[2]= or[member[composite[x, x], map[y, y]], not[member[x, map[y, y]]]] == True

In[3]:= or[member[composite[x_, x_], map[y_, y_]], not[member[x_, map[y_, y_]]]] := True
```

The variable **x** is eliminated first.

```
In[4]:= Map[equal[V, #] &, SubstTest[class, x,
           or[member[composite[x, x], z], not[member[x, z]]], z → map[y, y]]] // Reverse

Out[4]= subclass[map[y, y], fix[image[inverse[COMPOSE], map[y, y]]]] == True

In[5]:= (% /. y → y_) /. Equal → SetDelayed
```

One of the **map[y,y]** sets is easily replaced by the larger class **UNOPS**.

```
In[6]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]], subclass[u, w],
          {u → map[y, y], v → fix[image[inverse[COMPOSE], map[y, y]]],
           w → fix[image[inverse[COMPOSE], UNOPS]]}]

Out[6]= subclass[map[y, y], fix[image[inverse[COMPOSE], UNOPS]]] == True

In[7]:= (% /. y → y_) /. Equal → SetDelayed
```

The remaining **y** variable is eliminated using **reify**.

```
In[8]:= Map[equal[0, range[#]] &, SubstTest[reify, y,
            dif[map[y, y], fix[image[inverse[z], UNOPS]]], z → COMPOSE]] // Reverse

Out[8]= subclass[UNOPS, fix[image[inverse[COMPOSE], UNOPS]]] == True

In[9]:= subclass[UNOPS, fix[image[inverse[COMPOSE], UNOPS]]] := True
```

The variable **x** is reintroduced to help make the final result more understandable.

```
In[10]:= SubstTest[implies, and[member[u, v], subclass[v, w]], member[u, w],
           {u → pair[x, x], v → id[UNOPS], w → image[inverse[COMPOSE], UNOPS]}]

Out[10]= or[member[composite[x, x], UNOPS], not[member[x, UNOPS]]] == True

In[11]:= or[member[composite[x_, x_], UNOPS], not[member[x_, UNOPS]]] := True
```

invariance under composite[COMPOSE, DUP]

A related result is derived in this section. Again **reify** is used to remove the variable **y**.

```
In[12]:= Map[equal[0, #] &, SubstTest[reify, y,
             dif[map[y, y], fix[image[inverse[z], map[y, y]]]], z → COMPOSE]] // Reverse

Out[12]= subclass[image[MAP, Id], invar[composite[COMPOSE, DUP]]] == True

In[13]:= subclass[image[MAP, Id], invar[composite[COMPOSE, DUP]]] := True
```

The union of a collection of invariant subsets is itself invariant.

```
In[14]:= SubstTest[implies, subclass[u, invar[v]], invariant[v, U[u]],
           {u -> image[MAP, Id], v -> composite[COMPOSE, DUP]}]

Out[14]= subclass[image[COMPOSE, id[UNOPS]], UNOPS] == True

In[15]:= subclass[image[COMPOSE, id[UNOPS]], UNOPS] := True
```

This result is equivalent to the variable-free statement **subclass[UNOPS, fix[image[inverse[COMPOSE], UNOPS]]]** derived in the preceding section, but using direct images in place of inverse images.