map[x,x] is closed under composition

Johan G. F. Belinfante
2006 June 20

```
In[1]:= SetDirectory["l:"]; << goedel82.18a; << tools.m

        :Package Title: goedel82.18a         2006 June 18 at 1:00 p.m.

        It is now:  2006 Jun 20 at 16:3

        Loading Simplification Rules

        TOOLS.M                       Revised 2006 June 6

        weightlimit = 40
```

summary

If **x** and **y** are a unary operations on a set **z**,then so is **composite[x, y]**. That is, the set **map[z,z]** is closed under composition. An inclusion is already known:

```
In[2]:= subclass[image[COMPOSE, cart[map[x, x], map[x, x]]], map[x, x]]

Out[2]= True
```

In this notebook, an equation is derived.

derivation

Lemma.

```
In[3]:= member[id[x], map[x, x]] // AssertTest

Out[3]= member[id[x], map[x, x]] == member[x, V]

In[4]:= member[id[x_], map[x_, x_]] := member[x, V]
```

Lemma.

```
In[5]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
          {p1 → member[w, map[x, y]], p2 → subclass[w, cart[x, y]], p3 → subclass[w, cart[x, V]]}]]

Out[5]= or[not[member[w, map[x, y]]], subclass[w, cart[x, V]]] == True

In[6]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

If **w** is a member of **map[x, x]**, one can write **w = composite[w, id[x]]**. One can rewrite **composite** in terms of **COMPOSE**. A lemma is needed:

```
In[7]:= SubstTest[implies, subclass[u, v], subclass[image[w, u], image[w, v]],
          {u -> cart[set[y], set[id[x]]], v → cart[map[x, x], map[x, x]], w → COMPOSE}]

Out[7]= or[member[composite[y, id[x]], image[COMPOSE, cart[map[x, x], map[x, x]]]],
          not[member[x, V]], not[member[y, map[x, x]]], not[member[image[y, x], V]]] == True

In[8]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The following is a bit tricky in that rewrite rules are being circumvented by equality substitutions:

```
In[9]:= Map[not, SubstTest[and, implies[p2, p3],
            implies[and[p1, p2, p3], p4], implies[and[p4, p5, p6], p7],
            implies[and[p2, p5], p6], not[implies[and[p1, p2, p5], p7]],
            {p1 → member[x, V], p2 → member[y, map[x, x]], p3 → member[image[y, x], V],
             p4 -> member[composite[y, id[x]], image[COMPOSE, cart[map[x, x], map[x, x]]]],
             p5 → equal[z, composite[y, id[x]]], p6 → equal[y, z],
             p7 -> member[y, image[COMPOSE, cart[map[x, x], map[x, x]]]]}]] /.
          z → composite[y, id[x]]

Out[9]= or[member[y, image[COMPOSE, cart[map[x, x], map[x, x]]]],
          not[member[x, V]], not[member[y, map[x, x]]]] == True

In[10]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The variable **y** is eliminated in a standard fashion.

```
In[11]:= Map[equal[V, #] &,
           SubstTest[class, y, or[member[y, s], not[member[x, V]], not[member[y, t]]],
            {s → image[COMPOSE, cart[map[x, x], map[x, x]]], t → map[x, x]}]] // Reverse

Out[11]= or[not[member[x, V]],
           subclass[map[x, x], image[COMPOSE, cart[map[x, x], map[x, x]]]]] == True

In[12]:= (% /. x → x_) /. Equal → SetDelayed
```

The same conclusion holds when **x** is not a set.

```
In[13]:= SubstTest[implies, equal[0, z], subclass[z, image[COMPOSE, cart[z, z]]], z -> map[x, x]]

Out[13]= or[member[x, V], subclass[map[x, x], image[COMPOSE, cart[map[x, x], map[x, x]]]]] == True

In[14]:= (% /. x → x_) /. Equal → SetDelayed
```

Therefore the literal **member[x,V]** is not needed.

```
In[15]:= SubstTest[and, implies[p1, p2], or[p1, p2], {p1 -> member[x, V],
             p2 -> subclass[map[x, x], image[COMPOSE, cart[map[x, x], map[x, x]]]]}] // Reverse

Out[15]= subclass[map[x, x], image[COMPOSE, cart[map[x, x], map[x, x]]]] == True
```

```
In[16]:= (% /. x → x_) /. Equal → SetDelayed
```

Combine this with the reverse inclusion.

```
In[17]:= SubstTest[and, subclass[u, v], subclass[v, u],
           {u -> map[x, x], v -> image[COMPOSE, cart[map[x, x], map[x, x]]]}]

Out[17]= True == equal[image[COMPOSE, cart[map[x, x], map[x, x]]], map[x, x]]

In[18]:= image[COMPOSE, cart[map[x_, x_], map[x_, x_]]] := map[x, x]
```

One can now use **reify** to eliminate the variable **x**.

```
In[19]:= Map[VERTSECT,
           SubstTest[reify, x, image[COMPOSE, cart[f[x, x], f[x, x]]], f → map]] // Reverse

Out[19]= composite[IMAGE[COMPOSE], CART, DUP, MAP, DUP] == composite[MAP, DUP]

In[20]:= composite[IMAGE[COMPOSE], CART, DUP, MAP, DUP] := composite[MAP, DUP]
```

Corollary.

```
In[21]:= ImageComp[composite[IMAGE[COMPOSE], CART, DUP], composite[MAP, DUP], V] // Reverse

Out[21]= image[IMAGE[COMPOSE], image[CART, id[image[MAP, Id]]]] == image[MAP, Id]

In[22]:= image[IMAGE[COMPOSE], image[CART, id[image[MAP, Id]]]] := image[MAP, Id]
```

Another formulation of this:

```
In[25]:= Map[equal[0, #] &,
           SubstTest[reify, x, dif[set[map[x, x]], y], y -> binclosed[COMPOSE]]] // Reverse

Out[25]= subclass[image[MAP, Id], binclosed[COMPOSE]] == True

In[26]:= subclass[image[MAP, Id], binclosed[COMPOSE]] := True
```