inverses of powers of group elements

Johan G. F. Belinfante
2013 January 29

```
In[1]:= SetDirectory["l:"]; << goedel.13jan28a

:Package Title: goedel.13jan28a             2013 January 28 at 10:15 a.m.

Loading takes about sixteen minutes, half that time due to builtin pauses.

It is now:  2013 Jan 29 at 14:38

Loading Simplification Rules

TOOLS.M is now incorporated in the GOEDEL program as of 2010 September 3

weightlimit = 40

Loading completed.

It is now:  2013 Jan 29 at 14:54
```

summary

The composite of a functor from one monoid to another with the power list of an element of the range of a monoid is the list of powers of a transformed element of the range of the second monoid. Since **inv[gp[x]]** is a functor from **gp[x]** to **gp[x]** ∘ **SWAP**, one can show that powers of inverses are inverses of powers.

functors for groups

For groups, functors are the same as binary homomorphisms.

Lemma.

```
In[2]:= SubstTest[implies, and[member[u, GROUPS], member[v, GROUPS], member[t, binhom[u, v]]],
          functor[t, u, v], {u → gp[x], v → gp[y]}] // Reverse

Out[2]= or[equal[0, gp[x]], equal[0, gp[y]],
          functor[t, gp[x], gp[y]], not[member[t, binhom[gp[x], gp[y]]]]] == True

In[3]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma. (Eliminate redundant literals.)

```
In[4]:= SubstTest[and, implies[p, q], or[p, q], {p -> or[equal[0, gp[x]], equal[0, gp[y]]],
           q → or[functor[t, gp[x], gp[y]], not[member[t, binhom[gp[x], gp[y]]]]]}] // MapNotNot

Out[4]= or[functor[t, gp[x], gp[y]], not[member[t, binhom[gp[x], gp[y]]]]] == True

In[5]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem.

```
In[6]:= equiv[functor[t, gp[x], gp[y]], member[t, binhom[gp[x], gp[y]]]]

Out[6]= True

In[7]:= functor[t_, gp[x_], gp[y_]] := member[t, binhom[gp[x], gp[y]]]
```

Corollary.

```
In[8]:= Map[domain[reify[t, #]] &, SubstTest[case, member[t, w], w → func[gp[x], gp[y]]]]

Out[8]= func[gp[x], gp[y]] == binhom[gp[x], gp[y]]

In[9]:= func[gp[x_], gp[y_]] := binhom[gp[x], gp[y]]
```

main theorem

Theorem. The composite of a group homomorphism and a list of powers of a group element is the list of powers of the transformed element.

```
In[10]:= SubstTest[implies, and[functor[t, monoid[u], monoid[v]], member[z, range[monoid[u]]]],
           equal[composite[t, iterate[composite[monoid[u], LEFT[z]], set[e[monoid[u]]]]],
            iterate[composite[monoid[v], LEFT[APPLY[t, z]]], set[e[monoid[v]]]]],
           {u → gp[x], v → gp[y]}] // Reverse

Out[10]= or[equal[composite[t, iterate[composite[gp[x], LEFT[z]], set[e[gp[x]]]]],
            iterate[composite[gp[y], LEFT[APPLY[t, z]]], set[e[gp[y]]]]],
           not[member[t, binhom[gp[x], gp[y]]]], not[member[z, range[gp[x]]]]] == True

In[11]:= or[equal[composite[t_, iterate[composite[gp[x_], LEFT[z_]], set[e[gp[x_]]]]],
            iterate[composite[gp[y_], LEFT[APPLY[t_, z_]]], set[e[gp[y_]]]]],
           not[member[t_, binhom[gp[x_], gp[y_]]]], not[member[z_, range[gp[x_]]]]] := True
```

In the special case of **inv[gp[x]]**, one has this corollary.

Corollary.

```
In[12]:= SubstTest[implies, and[member[t, binhom[gp[u], gp[v]]], member[y, range[gp[u]]]],
           equal[composite[t, iterate[composite[gp[u], LEFT[y]], set[e[gp[u]]]]],
            iterate[composite[gp[v], LEFT[APPLY[t, y]]], set[e[gp[v]]]]],
           {t → inv[gp[x]], u → gp[x], v → flip[gp[x]]}] // Reverse

Out[12]= or[equal[composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]],
            iterate[composite[gp[x], RIGHT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]]],
           not[member[y, range[gp[x]]]]] == True

In[13]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

One can replace right multiplication with left multiplication in the present situation.

Lemma.

```
In[14]:= SubstTest[implies, and[implies[p, equal[u, v]], equal[v, w]],
           implies[p, equal[u, w]], {p -> member[y, range[gp[x]]],
            u → composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]],
            v -> iterate[composite[gp[x], RIGHT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]], w ->
             iterate[composite[gp[x], LEFT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]]}] // Reverse

Out[14]= or[equal[composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]],
            iterate[composite[gp[x], LEFT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]]],
           not[member[y, range[gp[x]]]]] == True

In[15]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

There is a redundant literal.

Lemma.

```
In[16]:= SubstTest[implies, and[empty[u], empty[v]],
           or[equal[composite[inv[gp[x]], iterate[u, set[e[gp[x]]]]],
             iterate[v, set[e[gp[x]]]]]], {u -> composite[gp[x], LEFT[y]],
            v -> composite[gp[x], LEFT[APPLY[inv[gp[x]], y]]]}] // Reverse

Out[16]= or[equal[composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]],
            iterate[composite[gp[x], LEFT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]]],
           member[y, range[gp[x]]]] == True

In[17]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

It is not clear how best to orient the following. The choice was made to eliminate **APPLY**.

Main Theorem. Inverses of powers are powers of inverses for group elements.

```
In[18]:= SubstTest[and, implies[p, q], or[p, q], {p -> member[y, range[gp[x]]],
           q -> equal[composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]],
             iterate[composite[gp[x], LEFT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]]]}]

Out[18]= equal[composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]],
           iterate[composite[gp[x], LEFT[APPLY[inv[gp[x]], y]]], set[e[gp[x]]]]] == True
```

```
In[19]:= iterate[composite[gp[x_], LEFT[APPLY[inv[gp[x_]], y_]]], set[e[gp[x_]]]] :=
          composite[inv[gp[x]], iterate[composite[gp[x], LEFT[y]], set[e[gp[x]]]]]
```

Corollary.

```
In[20]:= SubstTest[iterate, composite[gp[u], LEFT[APPLY[inv[gp[u]], v]]],
           set[e[gp[u]]], {u → INTADD, v → int[x]}] // Reverse

Out[20]= iterate[inverse[intplus[int[x]]], set[id[omega]]] ==
          composite[INVERSE, MIXMUL, LEFT[int[x]]]

In[21]:= iterate[inverse[intplus[int[x_]]], set[id[omega]]] :=
          composite[INVERSE, MIXMUL, LEFT[int[x]]]
```

The orientation of the following was suggested by examining the special case that **int[x]** is **plus[x]**.

Theorem.

```
In[22]:= SubstTest[iterate, intplus[int[t]], set[id[omega]], t → inverse[int[x]]]

Out[22]= composite[MIXMUL, LEFT[inverse[int[x]]]] == composite[INVERSE, MIXMUL, LEFT[int[x]]]
```

A more general result can be derived.

Theorem.

```
In[23]:= Assoc[MIXMUL, cross[INVERSE, Id], LEFT[composite[Id, x]]]

Out[23]= composite[MIXMUL, LEFT[inverse[x]]] ==
          composite[INVERSE, MIXMUL, LEFT[composite[Id, x]]]

In[24]:= composite[MIXMUL, LEFT[inverse[x_]]] :=
          composite[INVERSE, MIXMUL, LEFT[composite[Id, x]]]
```