subgroups preserve inverses

Johan G. F. Belinfante
2009 July 3

```
In[1]:= SetDirectory["l:"]; << goedel.09jul02a; << tools.m
```

```
:Package Title: goedel.09jul02a              2009 July 2 at 2:50 p.m.

It is now:  2009 Jul 3 at 9:33

Loading Simplification Rules

TOOLS.M                                    Revised 2009 July 2

weightlimit = 40
```

summary

The inverse of an element of a subgroup of a group is the same as its inverse in the group. The function **inv[x]** assigns to each element of a group its inverse element. It is shown that if **x** is a subgroup of a group **y**, then **inv[x] ⊂ inv[y]**. The derivation initially uses the wrapper **gp[x]** which represents either a group or the empty set. To exclude the case of the empty set, it is occasionally necessary to include an explicit literal of the form **not[equal[0, gp[x]]]**. Such literals are automatically eliminated in the wrapper-free results derived as corollaries. In the present case, one such literal can be eliminated even for the wrapped version.

eliminating a literal

Observation.

```
In[2]:= inv[0]
```

```
Out[2]= 0
```

Lemma.

```
In[3]:= equiv[or[equal[0, x], subclass[inv[x], y]], subclass[inv[x], y]]
```

```
Out[3]= True
```

```
In[5]:= or[equal[0, x_], subclass[inv[x_], y_]] := subclass[inv[x], y]
```

derivation

Lemma.

```
In[6]:= SubstTest[implies, subclass[u, v], subclass[image[inverse[u], w], image[inverse[v], w]],
          {u → gp[x], v → gp[y], w → set[e[gp[y]]]}] // Reverse

Out[6]= or[not[subclass[gp[x], gp[y]]],
          subclass[composite[gp[x], RIGHT[e[gp[y]]], inv[gp[x]]], inv[gp[y]]] == True

In[7]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The identity element of a subgroup is the same as that of the group. Using this fact permits one to clean up the statement in the lemma.

Theorem.

```
In[8]:= Map[not,
          SubstTest[and, implies[p1, p2], implies[and[p0, p1], p3], implies[and[p2, p3], p4],
           not[implies[and[p0, p1], p4]], {p0 → not[empty[gp[x]]], p1 → subclass[gp[x], gp[y]],
            p2 -> subclass[composite[gp[x], RIGHT[e[gp[y]]], inv[gp[x]]], inv[gp[y]]],
            p3 → equal[e[gp[x]], e[gp[y]]], p4 ->
             subclass[composite[gp[x], RIGHT[e[gp[x]]], inv[gp[x]]], inv[gp[y]]]}]] // Reverse

Out[8]= or[not[subclass[gp[x], gp[y]]], subclass[inv[gp[x]], inv[gp[y]]] == True

In[9]:= or[not[subclass[gp[x_], gp[y_]]], subclass[inv[gp[x_]], inv[gp[y_]]]] := True
```

The **gp[x]** wrappers can be eliminated.

Theorem.

```
In[10]:= Map[implies[and[member[x, GROUPS], member[y, GROUPS]], #] &,
           SubstTest[implies, and[equal[x, gp[u]], equal[y, gp[v]]], or[not[subclass[x, y]],
             subclass[inv[x], inv[y]]], {u → x, v → y}]] // Reverse // MapNotNot

Out[10]= or[not[member[x, GROUPS]], not[member[y, GROUPS]],
           not[subclass[x, y]], subclass[inv[x], inv[y]]] == True

In[11]:= or[not[member[x_, GROUPS]], not[member[y_, GROUPS]],
           not[subclass[x_, y_]], subclass[inv[x_], inv[y_]]] := True
```