pairwise disjointness of cosets

Johan G. F. Belinfante
2011 June 30

```
In[1]:= SetDirectory["l:"]; << goedel.11jun29a

        :Package Title: goedel.11jun29a               2011 June 29 at 11:55 a.m.

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

        It is now:  2011 Jun 30 at 14:39

        Loading Simplification Rules

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

        weightlimit = 40

        Loading completed.

        It is now:  2011 Jun 30 at 14:49
```

summary

The left cosets of a subgroup of a group are pairwise disjoint. The same goes for right cosets. The left and right cosets for the identity element are both equal to the range of the subgroup.

pairwise disjointness of cosets

The **GOEDEL** program has a rewrite rule for a vertical section of a relation to be disjoint from a given class. To keep this rule from interfering with the derivation of the main theorem, the following temporary lemma suffices to cope with this situation.

Temporary Lemma.

```
In[4]:= Map[not, SubstTest[disjoint, image[t, set[y]],
          image[t, set[z]], t -> composite[x, id[cart[V, w]], inverse[FIRST]]]]

Out[4]= member[pair[z, y],
          composite[FIRST, id[cart[V, w]], inverse[x], x, id[cart[V, w]], inverse[FIRST]]] ==
         not[equal[0, intersection[image[x, cart[set[y], w]], image[x, cart[set[z], w]]]]]

In[5]:= member[pair[z_, y_],
          composite[FIRST, id[cart[V, w_]], inverse[x_], x_, id[cart[V, w_]], inverse[FIRST]]] :=
         not[equal[0, intersection[image[x, cart[set[y], w]], image[x, cart[set[z], w]]]]]
```

Lemma. A condition for left cosets to be pairwise disjoint.

```
In[6]:= SubstTest[implies, EQUIVALENCE[t], or[equal[image[t, set[u]], image[t, set[v]]],
          disjoint[image[t, set[u]], image[t, set[v]]]],
         t → composite[gp[y], id[cart[V, range[gp[x]]]], inverse[FIRST]]] // Reverse

Out[6]= or[equal[0, intersection[image[gp[y], cart[set[u], range[gp[x]]]],
           image[gp[y], cart[set[v], range[gp[x]]]]], equal[
          image[gp[y], cart[set[u], range[gp[x]]]], image[gp[y], cart[set[v], range[gp[x]]]]],
         not[EQUIVALENCE[composite[gp[y], id[cart[V, range[gp[x]]]], inverse[FIRST]]]] == True

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

Theorem. If **gp[x] ⊂ gp[y]**, then the left cosets of **gp[x]** are pairwise disjoint.

```
In[8]:= Map[not, SubstTest[and, implies[p1, p2],
          implies[p2, p3], not[implies[p1, p3]], {p1 → subclass[gp[x], gp[y]],
           p2 -> EQUIVALENCE[composite[gp[y], id[cart[V, range[gp[x]]]], inverse[FIRST]]],
           p3 -> or[equal[0, intersection[image[gp[y], cart[set[u], range[gp[x]]]],
               image[gp[y], cart[set[v], range[gp[x]]]]]], equal[image[gp[y], cart[set[u],
            range[gp[x]]]], image[gp[y], cart[set[v], range[gp[x]]]]]}]] // Reverse

Out[8]= or[equal[0, intersection[image[gp[y], cart[set[u], range[gp[x]]]],
           image[gp[y], cart[set[v], range[gp[x]]]]],
          equal[image[gp[y], cart[set[u], range[gp[x]]]],
           image[gp[y], cart[set[v], range[gp[x]]]]], not[subclass[gp[x], gp[y]]]] == True

In[9]:= or[equal[0, intersection[image[gp[y_], cart[set[u_], range[gp[x_]]]],
            image[gp[y_], cart[set[v_], range[gp[x_]]]]]],
          equal[image[gp[y_], cart[set[u_], range[gp[x_]]]],
           image[gp[y_], cart[set[v_], range[gp[x_]]]], not[subclass[gp[x_], gp[y_]]]] := True
```

Corollary. Restatement without **gp** wrappers.

```
In[10]:= SubstTest[implies, and[equal[x, gp[s]], equal[y, gp[t]]], or[equal[0,
             intersection[image[y, cart[set[u], range[x]]], image[y, cart[set[v], range[x]]]]],
            equal[image[y, cart[set[u], range[x]]], image[y, cart[set[v], range[x]]]],
            not[subclass[x, y]]], {s → x, t → y}] // MapNotNot // Reverse

Out[10]= or[equal[0,
            intersection[image[y, cart[set[u], range[x]]], image[y, cart[set[v], range[x]]]],
           equal[image[y, cart[set[u], range[x]]], image[y, cart[set[v], range[x]]]],
           not[member[x, GROUPS]], not[member[y, GROUPS]], not[subclass[x, y]]] == True

In[11]:= or[equal[0, intersection[
             image[y_, cart[set[u_], range[x_]]], image[y_, cart[set[v_], range[x_]]]],
           equal[image[y_, cart[set[u_], range[x_]]], image[y_, cart[set[v_], range[x_]]]],
           not[member[x_, GROUPS]], not[member[y_, GROUPS]], not[subclass[x_, y_]]] := True
```

right cosets

In this section, duality is used to derive the corresponding facts about right cosets.

Theorem. If **gp[x] ⊂ gp[y]**, then the right cosets of **gp[x]** are pairwise disjoint.

```
In[12]:= SubstTest[implies, subclass[gp[s], gp[t]],
           or[equal[0, intersection[image[gp[t], cart[set[u], range[gp[s]]]],
              image[gp[t], cart[set[v], range[gp[s]]]]]], equal[image[gp[t],
              cart[set[u], range[gp[s]]]], image[gp[t], cart[set[v], range[gp[s]]]]],
           {s → flip[gp[x]], t → flip[gp[y]]}] // Reverse

Out[12]= or[equal[0, intersection[image[gp[y], cart[range[gp[x]], set[u]]],
             image[gp[y], cart[range[gp[x]], set[v]]]],
           equal[image[gp[y], cart[range[gp[x]], set[u]]],
            image[gp[y], cart[range[gp[x]], set[v]]]], not[subclass[gp[x], gp[y]]] == True

In[13]:= or[equal[0, intersection[image[gp[y_], cart[range[gp[x_]], set[u_]]],
             image[gp[y_], cart[range[gp[x_]], set[v_]]]],
           equal[image[gp[y_], cart[range[gp[x_]], set[u_]]],
            image[gp[y_], cart[range[gp[x_]], set[v_]]], not[subclass[gp[x_], gp[y_]]] := True
```

Corollary.

```
In[14]:= SubstTest[implies, and[equal[x, gp[s]], equal[y, gp[t]]], or[equal[0,
             intersection[image[y, cart[range[x], set[u]]], image[y, cart[range[x], set[v]]]]],
           equal[image[y, cart[range[x], set[u]]], image[y, cart[range[x], set[v]]]],
           not[subclass[x, y]]], {s → x, t → y}] // MapNotNot // Reverse

Out[14]= or[equal[0,
            intersection[image[y, cart[range[x], set[u]]], image[y, cart[range[x], set[v]]]],
           equal[image[y, cart[range[x], set[u]]], image[y, cart[range[x], set[v]]]],
           not[member[x, GROUPS]], not[member[y, GROUPS]], not[subclass[x, y]]] == True

In[15]:= or[equal[0, intersection[
             image[y_, cart[range[x_], set[u_]]], image[y_, cart[range[x_], set[v_]]]]],
           equal[image[y_, cart[range[x_], set[u_]]], image[y_, cart[range[x_], set[v_]]]],
           not[member[x_, GROUPS]], not[member[y_, GROUPS]], not[subclass[x_, y_]]] := True
```

cosets of the identity element

The left and right cosets of the identity element are both equal to the range of the subgroup. This result is already available in the following form which uses the **gp** wrapper.

```
In[16]:= implies[subclass[gp[x], gp[y]],
          equal[image[gp[y], cart[set[e[gp[y]]], range[gp[x]]]], range[gp[x]]]]

Out[16]= True
```

A wrapper-free restatement of this fact can be derived using the standard method for eliminating wrappers.

Theorem. If **x** is a subgroup of a group **y**, then the left coset of the identity element **e[y]** is the range of the subgroup.

```
In[17]:= SubstTest[implies, and[equal[x, gp[u]], equal[y, gp[v]]],
           implies[subclass[x, y], equal[image[y, cart[set[e[y]], range[x]]], range[x]]],
           {u → x, v → y}] // Reverse // MapNotNot

Out[17]= or[equal[image[y, cart[set[e[y]], range[x]]], range[x]],
          not[member[x, GROUPS]], not[member[y, GROUPS]], not[subclass[x, y]]] == True

In[18]:= or[equal[image[y_, cart[set[e[y_]], range[x_]]], range[x_]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]], not[subclass[x_, y_]]] := True
```

Lemma. (The dual of an available rewrite rule.)

```
In[20]:= image[gp[x], cart[y, set[e[gp[x]]]]] // Renormality

Out[20]= image[gp[x], cart[y, set[e[gp[x]]]]] == intersection[y, range[gp[x]]]

In[21]:= image[gp[x_], cart[y_, set[e[gp[x_]]]]] := intersection[y, range[gp[x]]]
```

Dual theorem. If **x** is a subgroup of a group **y**, then the right coset of the identity element **e[y]** is the range of the subgroup.

```
In[22]:= SubstTest[implies, and[equal[x, gp[u]], equal[y, gp[v]]],
           implies[subclass[x, y], equal[image[y, cart[range[x], set[e[y]]]], range[x]]],
           {u → x, v → y}] // Reverse // MapNotNot

Out[22]= or[equal[image[y, cart[range[x], set[e[y]]]], range[x]],
          not[member[x, GROUPS]], not[member[y, GROUPS]], not[subclass[x, y]]] == True

In[23]:= or[equal[image[y_, cart[range[x_], set[e[y_]]]], range[x_]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]], not[subclass[x_, y_]]] := True
```