inverse elements in groups, part 4

Johan G. F. Belinfante
2009 February 11

```
In[1]:= SetDirectory["l:"]; << goedel.09feb11a;<< tools.m

        :Package Title: goedel.09feb11a            2009 February 11 at 12:15 noon

        It is now:  2009 Feb 11 at 14:19

        Loading Simplification Rules

        TOOLS.M                                    Revised 2009 February 9

        weightlimit = 40
```

summary

An arsenal of tools, including quasigroup theory, duality, reification, proof by rotation and **gp** wrappers are called upon in this notebook to quickly derive rewrite rules for the fact that the inverse of a product of elements **u**, **v ∈ range[gp[x]]** of a group **gp[x]** is the product of their inverses in the reverse order. The various laws about inverses in group theory can be succinctly summarized using the language of category theory: the function **inv[gp[x]]** is a functor from the group **gp[x]** to the opposite group **flip[gp[x]]**.

regular representation rules

On account of the associative law, products of elements of a group can be related to composition of the corresponding left-multiplication functions.

Theorem. The regular representation of a group.

```
In[2]:= Assoc[composite[gp[x], cross[Id, gp[x]]], ASSOC, LEFT[PAIR[u, v]]]

Out[2]= composite[gp[x], LEFT[u], gp[x], LEFT[v]] ==
         composite[gp[x], LEFT[APPLY[gp[x], PAIR[u, v]]]]
```

```
In[3]:= composite[gp[x_], LEFT[u_], gp[x_], LEFT[v_]] :=
         composite[gp[x], LEFT[APPLY[gp[x], PAIR[u, v]]]]
```

Theorem. A dual result.

```
In[4]:= Assoc[composite[gp[x], cross[gp[x], Id]], inverse[ASSOC], RIGHT[PAIR[v, u]]]

Out[4]= composite[gp[x], RIGHT[u], gp[x], RIGHT[v]] ==
         composite[gp[x], RIGHT[APPLY[gp[x], PAIR[v, u]]]]
```

```
In[5]:= composite[gp[x_], RIGHT[u_], gp[x_], RIGHT[v_]] :=
         composite[gp[x], RIGHT[APPLY[gp[x], PAIR[v, u]]]]
```

divisibility relations

Groups are defined as nonempty associative quasigroups. The domain of a quasigroup is the same as the domains of its rotations, the latter being the left and right divisibility relations. The rules in this section amount to the statement that any element of a group is both left and right divisible by any other element. These rewrite rules will be needed to simplify expressions encountered in the next section.

Theorem. The left-divisibility relation for a group.

```
In[6]:= SubstTest[composite, quasigp[t], inverse[FIRST], t → gp[x]] // Reverse
Out[6]= composite[gp[x], inverse[FIRST]] == domain[gp[x]]

In[7]:= composite[gp[x_], inverse[FIRST]] := domain[gp[x]]
```

Corollary.

```
In[8]:= composite[FIRST, inverse[gp[x]]] // DoubleInverse
Out[8]= composite[FIRST, inverse[gp[x]]] == domain[gp[x]]

In[9]:= composite[FIRST, inverse[gp[x_]]] := domain[gp[x]]
```

Theorem. The right-divisibility relation for a group.

```
In[10]:= SubstTest[composite, quasigp[t], inverse[SECOND], t → gp[x]] // Reverse
Out[10]= composite[gp[x], inverse[SECOND]] == domain[gp[x]]

In[11]:= composite[gp[x_], inverse[SECOND]] := domain[gp[x]]
```

Corollary.

```
In[12]:= composite[SECOND, inverse[gp[x]]] // DoubleInverse
Out[12]= composite[SECOND, inverse[gp[x]]] == domain[gp[x]]

In[13]:= composite[SECOND, inverse[gp[x_]]] := domain[gp[x]]
```

rotation rule

Lemma. Simplification rule.

```
In[14]:= Assoc[gp[x], id[cart[V, range[gp[x]]]], cross[y, Id]]
Out[14]= composite[gp[x], cross[y, id[range[gp[x]]]]] == composite[gp[x], cross[y, Id]]
```

```
In[15]:= composite[gp[x_], cross[y_, id[range[gp[x_]]]]] := composite[gp[x], cross[y, Id]]
```

Theorem. A rotation rule for **gp[x]**.

```
In[16]:= Map[flip[rotate[inverse[reify[u, #]]]] &,
           Assoc[composite[gp[x], LEFT[APPLY[inv[gp[x]], u]]],
            composite[gp[x], LEFT[u]], inverse[composite[gp[x], LEFT[u]]]]] // Reverse

Out[16]= rotate[gp[x]] == composite[gp[x], SWAP, cross[Id, inv[gp[x]]]]

In[17]:= rotate[gp[x_]] := composite[gp[x], SWAP, cross[Id, inv[gp[x]]]]
```

In a quasigroup, left and right multiplications are one-to-one. For groups a more explicit statement can be made.

Theorem. The inverse of left-multiplication by an element is left-multiplication by its inverse.

```
In[18]:= Map[equal[#, inverse[composite[gp[x], LEFT[u]]]] &,
          Map[inverse, SubstTest[composite, rotate[t], RIGHT[APPLY[inv[gp[x]], u]], t → gp[x]]]]

Out[18]= equal[composite[gp[x], LEFT[APPLY[inv[gp[x]], u]]],
           composite[inverse[LEFT[u]], inverse[gp[x]]]] == True

In[19]:= composite[inverse[LEFT[u_]], inverse[gp[x_]]] :=
          composite[gp[x], LEFT[APPLY[inv[gp[x]], u]]]
```

The function **inv[gp[x]]** can be expressed as the vertical section of **inverse[gp[x]]** at the identity element. More generally, a similar result holds for all vertical sections.

Theorem. Vertical sections of **inverse[gp[x]]**.

```
In[20]:= SubstTest[composite, rotate[t], LEFT[u], t → gp[x]]

Out[20]= image[inverse[gp[x]], set[u]] == composite[gp[x], RIGHT[u], inv[gp[x]]]

In[21]:= image[inverse[gp[x_]], set[u_]] := composite[gp[x], RIGHT[u], inv[gp[x]]]
```

Observation. A special case of this is the function **inv[gp[x]]**.

```
In[22]:= image[inverse[gp[x]], set[e[gp[x]]]]

Out[22]= inv[gp[x]]
```

a dual rotation theorem

Theorem. A rotation formula for the opposite group.

```
In[23]:= SubstTest[rotate, gp[t], t → flip[gp[x]]] // Reverse

Out[23]= rotate[composite[gp[x], SWAP]] == composite[gp[x], cross[Id, inv[gp[x]]]]

In[24]:= rotate[composite[gp[x_], SWAP]] := composite[gp[x], cross[Id, inv[gp[x]]]]
```

Corollary. The inverse of right-multiplication by an element is right-multiplication by the inverse element.

```
In[25]:= SubstTest[composite, rotate[t], RIGHT[u], t → flip[gp[x]]]

Out[25]= composite[inverse[RIGHT[u]], inverse[gp[x]]] ==
          composite[gp[x], RIGHT[APPLY[inv[gp[x]], u]]]

In[26]:= composite[inverse[RIGHT[u_]], inverse[gp[x_]]] :=
          composite[gp[x], RIGHT[APPLY[inv[gp[x]], u]]]
```

inverse of a product

In this final section reification is used to derive clean forms of the rule about inverses of products.

Theorem. Rule for inverse of a product without variables for group elements.

```
In[27]:= Map[composite[inverse[#], cross[inv[gp[x]], Id]] &,
           SubstTest[reify, u, composite[inverse[RIGHT[u]], t], t -> inverse[gp[x]]]] // Reverse

Out[27]= composite[inv[gp[x]], gp[x]] == composite[gp[x], SWAP, cross[inv[gp[x]], inv[gp[x]]]]

In[28]:= composite[inv[gp[x_]], gp[x_]] := composite[gp[x], SWAP, cross[inv[gp[x]], inv[gp[x]]]]
```

Corollary. A clean **APPLY** rewrite rule for the inverse of a product, using the **gp[x]** wrapper. Note that one need not even assume that the variables **u** and **v** belong to **range[gp[x]]**. If either variable is not an element of the group then the equation simply reduces to the true statement **V = V**.

```
In[29]:= ApComp[inv[gp[x]], gp[x], PAIR[u, v]]

Out[29]= APPLY[inv[gp[x]], APPLY[gp[x], PAIR[u, v]]] ==
          APPLY[gp[x], PAIR[APPLY[inv[gp[x]], v], APPLY[inv[gp[x]], u]]]

In[30]:= APPLY[inv[gp[x_]], APPLY[gp[x_], PAIR[u_, v_]]] :=
          APPLY[gp[x], PAIR[APPLY[inv[gp[x]], v], APPLY[inv[gp[x]], u]]]
```

Since the function **inv[gp[x]]** also preserves the identity element, a concise functorial interpretation of the rules for inverses follows.

Corollary. The function **inv[gp[x]]** is a functor from **gp[x]** to its opposite.

```
In[31]:= functor[inv[gp[x]], gp[x], composite[gp[x], SWAP]] // AssertTest

Out[31]= functor[inv[gp[x]], gp[x], composite[gp[x], SWAP]] == True

In[32]:= functor[inv[gp[x_]], gp[x_], composite[gp[x_], SWAP]] := True
```