APPLY[rotate[x],y]

Johan G. F. Belinfante
2008 November 27

```
In[1]:= SetDirectory["l:"]; << goedel.08nov26b;<< tools.m

        :Package Title: goedel.08nov26b          2008 November 26 at 9:15 p.m.

        It is now:  2008 Nov 27 at 18:18

        Loading Simplification Rules

        TOOLS.M                                    Revised 2008 October 21

        weightlimit = 40
```

summary

The **APPLY** rule for **rotate[x]** is removed now and will be replaced with two rewrite rules in the opposite direction.

```
In[2]:= APPLY[rotate[x_], y_] =.
```

As an application, six basic equations in the theory of quasigroups are derived.

replacement rules

Theorem.

```
In[3]:= SubstTest[A, image[t, set[PAIR[y, z]]], t → rotate[inverse[x]]] // Reverse

Out[3]= APPLY[image[x, set[y]], z] == APPLY[rotate[inverse[x]], PAIR[y, z]]

In[4]:= APPLY[image[x_, set[y_]], z_] := APPLY[rotate[inverse[x]], PAIR[y, z]]
```

Theorem.

```
In[5]:= SubstTest[APPLY, image[t, set[y]], z, t → composite[SWAP, x]] // Reverse

Out[5]= APPLY[inverse[image[x, set[y]]], z] ==
         APPLY[rotate[composite[inverse[x], SWAP]], PAIR[y, z]]

In[6]:= APPLY[inverse[image[x_, set[y_]]], z_] :=
         APPLY[rotate[composite[inverse[x], SWAP]], PAIR[y, z]]
```

NATADD rule

Theorem.

```
In[7]:= SubstTest[APPLY, image[inverse[t], set[x]], y, t → NATADD]

Out[7]= APPLY[rotate[NATADD], PAIR[x, y]] == natsub[x, y]

In[8]:= APPLY[rotate[NATADD], PAIR[x_, y_]] := natsub[x, y]
```

quasigp equations

If elements **a**, **b**, **c** of a quasigroup satisfy **c = a · b = c**, then any two of these determines the third. One writes: **a = c / b** and **b = a \ c**. The following abbreviations help to compare with the formulas in books on quasigroups.

```
In[9]:= into[x_] := flip[rotate[x]]

In[10]:= over[x_] := rotate[flip[x]]

        General::spell1 :
         Possible spelling error: new symbol name "over" is similar to existing symbol "Over". More…
```

If **q** is a quasigroup binary operation, then **q**, **into[q]** and **over[q]** are also quasigroup binary operations. The notation **a · b = c** is equivalent to the statement **member[pair[pair[a, b], c], q]**. Similarly, the notations **a \ c = b** and **c / b = a** are equivalent to the statements:

```
In[11]:= member[pair[pair[a, c], b], into[q]]

Out[11]= member[pair[pair[a, b], c], q]

In[12]:= member[pair[pair[c, b], a], over[q]]

Out[12]= member[pair[pair[a, b], c], q]
```

One can use each of these three equations to eliminate one of the three variables in the other two equations. This yields a total of six equations:

(IL) **a \ (a · b) = b**
(IR) **a = (a · b) / b**
(SL) **a · (a \ c) = c**
(SR) **c = (c / b) · b**
(DL) **c / (a \ c) = a**
(DR) **b = (c / b) \ c**

See, for example, page 3 in the following reference for the first four of these six equations. The remaining two are on page 6.

```
In[13]:=  "Jonathan D. H. Smith, An Introduction to
           Quasigroups and Their Representations,  Chapman and Hall/CRC,
           Boca Raton, Florida, 2007.  QA 181.5 .S65.  ISBN 1-58488-537-8";
```

The letters used in the abbreviations used for these equations are shorthand for **I** = injective, **S** = surjective, **D** = division and **L** = left,
R = right. Each one of these six equations corresponds to a rewrite rule in the **GOEDEL** program for the wrapper **quasigp[x]**. Only the first equation needs to be derived; the other five can be obtained using **flip** and **rotate**.

Theorem. Equation **(IL)**. This equation is derived using **ReifNormality**.

```
In[14]:= Map[APPLY[#, PAIR[y, z]] &, composite[rotate[quasigp[x]],
            cross[composite[quasigp[x], SWAP], Id], ROT, cross[DUP, Id]]] // ReifNormality

Out[14]= APPLY[rotate[quasigp[x]], PAIR[APPLY[quasigp[x], PAIR[y, z]], y]] ==
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]

In[15]:= APPLY[rotate[quasigp[x_]], PAIR[APPLY[quasigp[x_], PAIR[y_, z_]], y_]] :=
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]
```

Corollary. Equation **(IR)**.

```
In[16]:= SubstTest[APPLY, rotate[quasigp[t]],
           PAIR[APPLY[quasigp[t], PAIR[y, z]], y], t → flip[quasigp[x]]] // Reverse

Out[16]= APPLY[rotate[composite[quasigp[x], SWAP]], PAIR[APPLY[quasigp[x], PAIR[z, y]], y]] ==
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]

In[17]:= APPLY[rotate[composite[quasigp[x_], SWAP]],
           PAIR[APPLY[quasigp[x_], PAIR[z_, y_]], y_]] :=
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]
```

Corollary. Equation **(SL)**.

```
In[18]:= SubstTest[APPLY, rotate[quasigp[t]],
           PAIR[APPLY[quasigp[t], PAIR[y, z]], y], t → flip[rotate[quasigp[x]]]] // Reverse

Out[18]= APPLY[quasigp[x], PAIR[y, APPLY[rotate[quasigp[x]], PAIR[z, y]]]] ==
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]

In[19]:= APPLY[quasigp[x_], PAIR[y_, APPLY[rotate[quasigp[x_]], PAIR[z_, y_]]]] :=
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]
```

Corollary. Equation **(SR)**.

```
In[20]:= SubstTest[APPLY, rotate[quasigp[t]],
           PAIR[APPLY[quasigp[t], PAIR[y, z]], y], t → rotate[rotate[quasigp[x]]]] // Reverse

Out[20]= APPLY[quasigp[x], PAIR[APPLY[rotate[composite[quasigp[x], SWAP]], PAIR[z, y]], y]] ==
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]

In[21]:= APPLY[quasigp[x_],
           PAIR[APPLY[rotate[composite[quasigp[x_], SWAP]], PAIR[z_, y_]], y_]] :=
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]
```

Corollary. Equation **(DL)**.

```
In[22]:= SubstTest[APPLY, rotate[quasigp[t]],
           PAIR[APPLY[quasigp[t], PAIR[y, z]], y], t → rotate[quasigp[x]]] // Reverse

Out[22]= APPLY[rotate[composite[quasigp[x], SWAP]],
           PAIR[y, APPLY[rotate[quasigp[x]], PAIR[y, z]]]] ==
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]

In[23]:= APPLY[rotate[composite[quasigp[x_], SWAP]],
           PAIR[y_, APPLY[rotate[quasigp[x_]], PAIR[y_, z_]]]] :=
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]
```

Corollary. Equation **(DR)**.

```
In[24]:= SubstTest[APPLY, rotate[quasigp[t]],
           PAIR[APPLY[quasigp[t], PAIR[y, z]], y], t → rotate[flip[quasigp[x]]]] // Reverse

Out[24]= APPLY[rotate[quasigp[x]],
           PAIR[y, APPLY[rotate[composite[quasigp[x], SWAP]], PAIR[y, z]]]] ==
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]

In[25]:= APPLY[rotate[quasigp[x_]],
           PAIR[y_, APPLY[rotate[composite[quasigp[x_], SWAP]], PAIR[y_, z_]]]] :=
          union[z, complement[image[V, intersection[range[quasigp[x]], set[y]]]],
           complement[image[V, intersection[range[quasigp[x]], set[z]]]]]
```