multiples of a rational number

Johan G. F. Belinfante
2012 December 1

```
In[1]:= SetDirectory["l:"]; << goedel.12nov30a
```

```
:Package Title: goedel.12nov30a              2012 November 30 at 9:35 p.m.

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

It is now:  2012 Dec 1 at 17:16

Loading Simplification Rules

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

weightlimit = 40

Loading completed.

It is now:  2012 Dec 1 at 17:31
```

summary

Every binary homomorphism **x** from **NATADD** to **RATADD** is of the form **rattimes[APPLY[x, {0}]] ∘ INTTIMES ∘ PLUS**.

derivation

Temporary definition.

```
In[2]:= funadd[x_, y_] := composite[INTADD,
          intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]
```

Lemma. Temporary rewrite rule.

```
In[3]:= (SubstTest[hull, RATS, funadd[rat[u], rat[v]],
            {u -> inttimes[plus[nat[t]]], v -> inttimes[plus[nat[x]]]}] /. t → set[0]) // Reverse
```

```
Out[3]= inttimes[composite[SUCC, plus[nat[x]]]] == ratadd[id[Z], inttimes[plus[nat[x]]]]
```

```
In[4]:= inttimes[composite[SUCC, plus[nat[x_]]]] := ratadd[id[Z], inttimes[plus[nat[x]]]]
```

Lemma. Temporary rewrite rule.

```
In[5]:= Map[composite[VERTSECT[#], id[omega]] &, SubstTest[reify, t, APPLY[w, nat[t]],
          w -> composite[rattimes[rat[x]], INTTIMES, PLUS, SUCC]] // Reverse

Out[5]= composite[ratplus[rat[x]], rattimes[rat[x]], INTTIMES, PLUS] ==
         composite[rattimes[rat[x]], INTTIMES, PLUS, SUCC]

In[6]:= composite[ratplus[rat[x_]], rattimes[rat[x_]], INTTIMES, PLUS] :=
         composite[rattimes[rat[x]], INTTIMES, PLUS, SUCC]
```

Theorem. (An application of the uniqueness theorem for **iterate**.)

```
In[7]:= SubstTest[implies,
          and[equal[composite[u, w], composite[w, SUCC]], equal[image[w, set[0]], v]],
          equal[composite[w, id[omega]], iterate[u, v]], {u → ratplus[rat[x]],
           v → set[e[RATADD]], w → composite[rattimes[rat[x]], INTTIMES, PLUS]}] // Reverse

Out[7]= equal[composite[rattimes[rat[x]], INTTIMES, PLUS],
          iterate[ratplus[rat[x]], set[cart[Z, set[id[omega]]]]]] == True

In[8]:= iterate[ratplus[rat[x_]], set[cart[Z, set[id[omega]]]]] :=
         composite[rattimes[rat[x]], INTTIMES, PLUS]
```

Corollary. (Eliminate the **rat** wrapper.)

```
In[9]:= SubstTest[implies, equal[x, rat[t]], equal[composite[rattimes[x], INTTIMES, PLUS],
          iterate[ratplus[x], set[cart[Z, set[id[omega]]]]]], t → x] // Reverse

Out[9]= or[equal[composite[rattimes[x], INTTIMES, PLUS],
          iterate[ratplus[x], set[cart[Z, set[id[omega]]]]], not[member[x, RATS]]] == True

In[10]:= or[equal[composite[rattimes[x_], INTTIMES, PLUS],
           iterate[ratplus[x_], set[cart[Z, set[id[omega]]]]], not[member[x_, RATS]]] := True
```

Any binary homomorphism from **NATADD** to a group is the power sequence of its value at **1 = {0}**. For the group **RATADD**, this yields:

Theorem.

```
In[11]:= SubstTest[or, equal[x, iterate[composite[y, LEFT[APPLY[x, set[0]]]], set[e[y]]]],
          not[member[x, binhom[NATADD, y]]], not[member[y, GROUPS]], y → RATADD] // Reverse

Out[11]= or[equal[x, iterate[ratplus[APPLY[x, set[0]]], set[cart[Z, set[id[omega]]]]]],
          not[member[x, binhom[NATADD, RATADD]]]] == True

In[12]:= or[equal[x_, iterate[ratplus[APPLY[x_, set[0]]], set[cart[Z, set[id[omega]]]]]],
          not[member[x_, binhom[NATADD, RATADD]]]] := True
```

Theorem.

```
In[13]:= SubstTest[implies, member[x, binhom[u, v]],
          member[x, map[fix[domain[u]], fix[domain[v]]]], {u → NATADD, v → RATADD}] // Reverse

Out[13]= or[member[x, map[omega, RATS]], not[member[x, binhom[NATADD, RATADD]]]] == True
```

```
In[14]:= or[member[x_, map[omega, RATS]], not[member[x_, binhom[NATADD, RATADD]]]] := True
```

Corollary.

```
In[15]:= Map[equal[V, #] &,
          dif[binhom[NATADD, RATADD], map[omega, RATS]] // complement // Normality]

Out[15]= subclass[binhom[NATADD, RATADD], map[omega, RATS]] == True

In[16]:= subclass[binhom[NATADD, RATADD], map[omega, RATS]] := True
```

Theorem.

```
In[17]:= SubstTest[implies, and[member[x, map[w, y]], member[s, w]],
           member[APPLY[x, s], y], {w → omega, s → set[0]}] // Reverse

Out[17]= or[member[APPLY[x, set[0]], y], not[member[x, map[omega, y]]]] == True

In[18]:= or[member[APPLY[x_, set[0]], y_], not[member[x_, map[omega, y_]]]] := True
```

Theorem.

```
In[19]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3],
             not[implies[p1, p3]], {p1 -> member[x, binhom[NATADD, RATADD]],
              p2 -> member[x, map[omega, RATS]], p3 -> member[APPLY[x, set[0]], RATS]}]] // Reverse

Out[19]= or[member[APPLY[x, set[0]], RATS], not[member[x, binhom[NATADD, RATADD]]]] == True

In[20]:= or[member[APPLY[x_, set[0]], RATS], not[member[x_, binhom[NATADD, RATADD]]]] := True
```

Main Theorem. If **x** is a binary homomorphism from **NATADD** to **RATADD**, then **x = rattimes[APPLY[x, {0}]] ∘ INTTIMES ∘ PLUS**.

```
In[21]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3],
             implies[p2, p4], implies[and[p3, p4], p5], not[implies[p1, p5]],
             {p1 → member[x, binhom[NATADD, RATADD]], p2 -> member[APPLY[x, set[0]], RATS],
              p3 → equal[x, iterate[ratplus[APPLY[x, set[0]]], set[cart[Z, set[id[omega]]]]]],
              p4 → equal[iterate[ratplus[APPLY[x, set[0]]], set[cart[Z, set[id[omega]]]]],
                composite[rattimes[APPLY[x, set[0]]], INTTIMES, PLUS]],
              p5 → equal[x, composite[rattimes[APPLY[x, set[0]]], INTTIMES, PLUS]]}]] // Reverse

Out[21]= or[equal[x, composite[rattimes[APPLY[x, set[0]]], INTTIMES, PLUS]],
           not[member[x, binhom[NATADD, RATADD]]]] == True

In[22]:= or[equal[x_, composite[rattimes[APPLY[x_, set[0]]], INTTIMES, PLUS]],
           not[member[x_, binhom[NATADD, RATADD]]]] := True
```