the relation inverse[RATIO] ∘ RATIO

Johan G. F. Belinfante
2012 July 12

```
In[1]:= SetDirectory["l:"]; << goedel.12jul08a

        :Package Title: goedel.12jul08a                    2012 July 8 at 7:30 a.m.

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

        It is now:  2012 Jul 12 at 13:33

        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 Jul 12 at 13:51
```

summary

A rewrite rule is derived for the equi-ratio relation **inverse[RATIO] ∘ RATIO**. A formula for the canonical map for this equivalence relation is also derived.

introduction

Theorem. The equivalence relation **inverse[RATIO] ∘ RATIO** is a set.

```
In[2]:= member[composite[inverse[RATIO], RATIO], V] // AssertTest

Out[2]= member[composite[inverse[RATIO], RATIO], V] == True

In[3]:= member[composite[inverse[RATIO], RATIO], V] := True
```

Lemma. An inclusion.

```
In[4]:= SubstTest[implies, subclass[u, v], subclass[composite[u, w], composite[v, w]],
          {u → inverse[RATIO], v → inverse[E], w → RATIO}] // Reverse

Out[4]= subclass[composite[inverse[RATIO], RATIO],
          composite[SWAP, twist[composite[inverse[INTMUL], INTMUL]]]] == True

In[5]:= subclass[composite[inverse[RATIO], RATIO],
          composite[SWAP, twist[composite[inverse[INTMUL], INTMUL]]]] := True
```

It will be shown in the sequel that **inverse[RATIO] ∘ RATIO** is the restriction of **SWAP ∘ twist[inverse[INTMUL ∘ INTMUL]]** to **domain[RATIO]**.

derivation

The following key theorem would take just under one minute if no steps were omitted, but the execution time drops by a factor of ten when the equality substitution step, commented out with **(* ⋯ *)**, is omitted.

Theorem. Equi-ratio theorem.

```
In[6]:= Map[not, SubstTest[and, implies[p3, p4],
           implies[p1, p5], implies[p2, p6], (* implies[and[p4,p5,p6],p7], *)
           not[implies[and[p1, p2, p3], p7]], {p1 → member[u, dif[Z, set[id[omega]]]],
            p2 -> member[x, dif[Z, set[id[omega]]]], p3 → equal[intmul[u, y], intmul[v, x]],
            p4 -> equal[composite[inverse[inttimes[intmul[u, x]]], inttimes[intmul[u, y]]],
              composite[inverse[inttimes[intmul[u, x]]], inttimes[intmul[v, x]]]],
            p5 → equal[composite[inverse[inttimes[intmul[u, x]]], inttimes[intmul[u, y]]],
              composite[inverse[inttimes[x]], inttimes[y]]],
            p6 → equal[composite[inverse[inttimes[intmul[u, x]]], inttimes[intmul[v, x]]],
              composite[inverse[inttimes[u]], inttimes[v]]],
            p7 → equal[composite[inverse[inttimes[u]], inttimes[v]],
              composite[inverse[inttimes[x]], inttimes[y]]]}]] // Reverse

Out[6]= or[equal[u, id[omega]], equal[x, id[omega]],
          equal[composite[inverse[inttimes[u]], inttimes[v]],
           composite[inverse[inttimes[x]], inttimes[y]]],
          not[equal[intmul[u, y], intmul[v, x]]], not[member[u, Z]], not[member[x, Z]] == True

In[7]:= or[equal[composite[inverse[inttimes[u_]], inttimes[v_]],
            composite[inverse[inttimes[x_]], inttimes[y_]]], equal[id[omega], u_],
          equal[id[omega], x_], not[equal[intmul[u_, y_], intmul[v_, x_]]],
          not[member[u_, Z]], not[member[x_, Z]]] := True
```

All four variables can be eliminated from this statement in one step.

Lemma. (Eliminating four variables.)

```
In[8]:= Map[composite[id[cart[V, V]], complement[#], id[cart[V, V]]] &,
          SubstTest[class, pair[pair[u, v], pair[x, y]],
           implies[member[pair[pair[u, v], pair[x, y]], intersection[r, s]], member[
             pair[pair[u, v], pair[x, y]], composite[inverse[t], t]]], {r → cartsq[domain[RATIO]],
           s -> composite[SWAP, twist[composite[inverse[INTMUL], INTMUL]]],
           t -> composite[COMPOSE, cross[composite[INVERSE, INTTIMES], INTTIMES]]}]]

Out[8]= composite[id[cart[intersection[Z, complement[set[id[omega]]]], Z]],
          intersection[composite[SWAP, twist[composite[inverse[INTMUL], INTMUL]]],
           composite[cross[composite[inverse[INTTIMES], INVERSE], inverse[INTTIMES]],
            complement[inverse[COMPOSE]], COMPOSE,
            cross[composite[INVERSE, INTTIMES], INTTIMES]]],
          id[cart[intersection[Z, complement[set[id[omega]]]], Z]]] == 0

In[9]:= % /. Equal→ SetDelayed
```

The above variable-free statement will now be simplified.

Lemma. A simplification rule.

```
In[10]:= composite[SWAP, id[cart[Z, intersection[Z, x]]],
           twist[composite[inverse[INTMUL], INTMUL]]] // DoubleInverse

Out[10]= composite[SWAP, id[cart[Z, intersection[x, Z]]],
           twist[composite[inverse[INTMUL], INTMUL]]] ==
          composite[SWAP, id[cart[V, x]], twist[composite[inverse[INTMUL], INTMUL]]]

In[11]:= composite[SWAP, id[cart[Z, intersection[x_, Z]]],
           twist[composite[inverse[INTMUL], INTMUL]]] :=
          composite[SWAP, id[cart[V, x]], twist[composite[inverse[INTMUL], INTMUL]]]
```

Lemma. An inclusion.

```
In[12]:= SubstTest[empty, intersection[dif[s, composite[inverse[t], t]], cartsq[r]],
           {r → domain[RATIO], s → composite[SWAP, twist[composite[inverse[INTMUL], INTMUL]]],
           t -> composite[COMPOSE, cross[composite[INVERSE, INTTIMES], INTTIMES]]}]

Out[12]= subclass[composite[SWAP, id[cart[V, complement[set[id[omega]]]]], twist[
              composite[inverse[INTMUL], INTMUL]], id[cart[complement[set[id[omega]]], V]]],
            composite[cross[composite[inverse[INTTIMES], INVERSE], inverse[INTTIMES]],
             inverse[COMPOSE], COMPOSE, cross[composite[INVERSE, INTTIMES], INTTIMES]]] == True

In[13]:= % /. Equal→ SetDelayed
```

Lemma. Simplification rule.

```
In[14]:= composite[cross[composite[id[complement[set[id[omega]]]], inverse[INTTIMES], INVERSE],
            inverse[INTTIMES]], inverse[COMPOSE]] // DoubleInverse

Out[14]= composite[cross[composite[id[complement[set[id[omega]]]], inverse[INTTIMES], INVERSE],
            inverse[INTTIMES]], inverse[COMPOSE]] == inverse[RATIO]
```

```
In[15]:= composite[cross[composite[id[complement[set[id[omega]]]], inverse[INTTIMES], INVERSE],
           inverse[INTTIMES]], inverse[COMPOSE]] := inverse[RATIO]
```

Lemma. An inclusion.

```
In[16]:= SubstTest[subclass, t, intersection[u, v],
           {t -> composite[SWAP, id[cart[V, complement[set[id[omega]]]]], twist[
               composite[inverse[INTMUL], INTMUL]], id[cart[complement[set[id[omega]]], V]]],
            u -> composite[cross[composite[inverse[INTTIMES], INVERSE], inverse[INTTIMES]],
              inverse[COMPOSE], COMPOSE, cross[composite[INVERSE, INTTIMES], INTTIMES]],
            v → cartsq[domain[RATIO]]}] // Reverse
```

```
Out[16]= subclass[composite[SWAP, id[cart[V, complement[set[id[omega]]]]],
           twist[composite[inverse[INTMUL], INTMUL]],
           id[cart[complement[set[id[omega]]], V]]], composite[inverse[RATIO], RATIO]] == True
```

```
In[17]:= % /. Equal→ SetDelayed
```

Lemma. An inclusion in the opposite direction.

```
In[18]:= SubstTest[subclass, u, intersection[v, w], {u -> composite[inverse[RATIO], RATIO],
           v -> composite[SWAP, twist[composite[inverse[INTMUL], INTMUL]],
             id[cart[complement[set[id[omega]]], V]]], w → cart[V, domain[RATIO]]}] // Reverse
```

```
Out[18]= subclass[composite[inverse[RATIO], RATIO],
           composite[SWAP, id[cart[V, complement[set[id[omega]]]]],
            twist[composite[inverse[INTMUL], INTMUL]]]] == True
```

```
In[19]:= % /. Equal→ SetDelayed
```

Main Theorem. An equation for the equi-ratio relation.

```
In[20]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> composite[SWAP, id[cart[V, complement[set[id[omega]]]]],
             twist[composite[inverse[INTMUL], INTMUL]],
             id[cart[complement[set[id[omega]]], V]]], v -> composite[inverse[RATIO], RATIO]}]
```

```
Out[20]= equal[composite[inverse[RATIO], RATIO], composite[SWAP,
           id[cart[V, complement[set[id[omega]]]]], twist[composite[inverse[INTMUL], INTMUL]],
           id[cart[complement[set[id[omega]]], V]]]] == True
```

```
In[21]:= composite[SWAP, id[cart[V, complement[set[id[omega]]]]],
           twist[composite[inverse[INTMUL], INTMUL]],
           id[cart[complement[set[id[omega]]], V]]] := composite[inverse[RATIO], RATIO]
```

removing the origin

Rational numbers are straight lines through the origin in the plane **Z × Z**. To obtain a pairwise disjoint collection from the class **RATS** of all rational numbers, one needs to remove the point at the origin of **Z × Z** from each rational number. The function **IMAGE[id[id[{id[ω]}]']]** removes this point at the origin from each member of any family of sets.

Theorem. Removing the vertical axis is equivalent to removing the origin.

```
In[22]:= Map[composite[IMAGE[id[complement[cart[set[id[omega]], set[id[omega]]]]]], #, RATIO] &,
          SubstTest[composite, IMAGE[id[U[x]]], id[x], x → RATS]]

Out[22]= composite[IMAGE[id[complement[cart[set[id[omega]], set[id[omega]]]]]], RATIO] ==
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]

In[23]:= composite[IMAGE[id[complement[cart[set[id[omega]], set[id[omega]]]]]], RATIO] :=
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]
```

Corollary. A forumula for the class of sets obtained by removing the point at the origin from each rational number.

```
In[24]:= ImageComp[IMAGE[id[complement[cart[set[id[omega]], set[id[omega]]]]]], RATIO, V] //
          Reverse

Out[24]= image[IMAGE[id[complement[cart[set[id[omega]], set[id[omega]]]]]], RATS] ==
          image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]

In[25]:= image[IMAGE[id[complement[cart[set[id[omega]], set[id[omega]]]]]], RATS] :=
          image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]
```

Curried binary operations

Theorem. A general result for Curried binary operations.

```
In[26]:= Map[composite[INVERSE, #] &,
           SubstTest[VERTSECT, composite[inverse[E], t], t -> composite[COMPOSE, cross[
               composite[INVERSE, APPLY[CURRY, binop[y]]], APPLY[CURRY, binop[x]]]]] // Reverse

Out[26]= VERTSECT[twist[composite[inverse[binop[x]], binop[y]]]] ==
          union[cart[complement[cart[fix[domain[binop[y]]], fix[domain[binop[x]]]]], set[0]],
           composite[COMPOSE, SWAP,
            cross[APPLY[CURRY, binop[y]], composite[INVERSE, APPLY[CURRY, binop[x]]]]]]

In[27]:= VERTSECT[twist[composite[inverse[binop[x_]], binop[y_]]]] :=
          union[cart[complement[cart[fix[domain[binop[y]]], fix[domain[binop[x]]]]], set[0]],
           composite[COMPOSE, SWAP,
            cross[APPLY[CURRY, binop[y]], composite[INVERSE, APPLY[CURRY, binop[x]]]]]]
```

Corollary. A special case of interest.

```
In[28]:= SubstTest[VERTSECT,
           twist[composite[inverse[binop[x]], binop[x]]], x → INTMUL] // Reverse

Out[28]= VERTSECT[twist[composite[inverse[INTMUL], INTMUL]]] ==
          union[cart[complement[cart[Z, Z]], set[0]],
           composite[COMPOSE, SWAP, cross[INTTIMES, composite[INVERSE, INTTIMES]]]]
```

```
In[29]:= VERTSECT[twist[composite[inverse[INTMUL], INTMUL]]] :=
          union[cart[complement[cart[Z, Z]], set[0]],
           composite[COMPOSE, SWAP, cross[INTTIMES, composite[INVERSE, INTTIMES]]]]
```

---

the canonical map

Lemma. Simplification rule.

```
In[30]:= composite[SECOND, id[cart[complement[set[id[omega]]], V]], inverse[INTMUL]] //
          DoubleInverse

Out[30]= composite[SECOND, id[cart[complement[set[id[omega]]], V]], inverse[INTMUL]] ==
          union[cart[set[id[omega]], set[id[omega]]],
           composite[inverse[INTDIV], id[complement[set[id[omega]]]]]]

In[31]:= % /. Equal→ SetDelayed
```

Theorem. A formula for **INVERSE** ∘ **RATIO**.

```
In[35]:= Assoc[INVERSE, INVERSE,
            composite[COMPOSE, SWAP, cross[composite[INTTIMES, id[complement[set[id[omega]]]]],
              composite[INVERSE, INTTIMES]]] // Reverse

Out[35]= composite[COMPOSE, SWAP, cross[composite[INTTIMES, id[complement[set[id[omega]]]]],
            composite[INVERSE, INTTIMES]]] == composite[INVERSE, RATIO]

In[36]:= composite[COMPOSE, SWAP, cross[composite[INTTIMES, id[complement[set[id[omega]]]]],
            composite[INVERSE, INTTIMES]]] := composite[INVERSE, RATIO]
```

Lemma. A simplification rule.

```
In[37]:= Assoc[composite[id[P[cart[V, V]]], IMAGE[id[x]]], IMAGE[id[cart[V, V]]], RATIO]

Out[37]= composite[id[P[cart[V, V]]], IMAGE[id[x]], RATIO] == composite[IMAGE[id[x]], RATIO]

In[38]:= composite[id[P[cart[V, V]]], IMAGE[id[x_]], RATIO] := composite[IMAGE[id[x]], RATIO]
```

Lemma. A simplification rule.

```
In[50]:= Assoc[IMAGE[id[cart[x, V]]], composite[COMPOSE,
            cross[composite[INVERSE, INTTIMES], INTTIMES]], id[domain[RATIO]]] // Reverse

Out[50]= composite[COMPOSE, cross[composite[INVERSE, INTTIMES, id[complement[set[id[omega]]]]],
            composite[IMAGE[id[cart[x, V]]], INTTIMES]]] ==
          composite[IMAGE[id[cart[x, V]]], RATIO]

In[51]:= composite[COMPOSE, cross[composite[INVERSE, INTTIMES, id[complement[set[id[omega]]]]],
            composite[IMAGE[id[cart[x_, V]]], INTTIMES]]] :=
          composite[IMAGE[id[cart[x, V]]], RATIO]
```

Lemma.

```
In[62]:= Assoc[IMAGE[id[cart[x, y]]], IMAGE[SWAP], INTTIMES]

Out[62]= composite[IMAGE[id[cart[x, y]]], INVERSE, INTTIMES] ==
          composite[INVERSE, IMAGE[id[cart[y, x]]], INTTIMES]

In[63]:= composite[IMAGE[id[cart[x_, y_]]], INVERSE, INTTIMES] :=
          composite[INVERSE, IMAGE[id[cart[y, x]]], INTTIMES]
```

Theorem. A temporary rewrite rule.

```
In[65]:= Map[composite[#, id[cart[intersection[Z, complement[set[id[omega]]]], Z]]] &,
           SubstTest[VERTSECT,
            composite[id[t], SWAP, twist[composite[inverse[INTMUL], INTMUL]], id[t]],
            t -> cart[complement[set[id[omega]]], V]]] // Reverse

Out[65]= composite[VERTSECT[inverse[RATIO]], RATIO] ==
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]

In[66]:= % /. Equal→ SetDelayed
```

Corollary.

```
In[67]:= Assoc[VERTSECT[inverse[RATIO]], RATIO, inverse[RATIO]]

Out[67]= composite[VERTSECT[inverse[RATIO]], id[RATS]] ==
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], id[RATS]]

In[68]:= % /. Equal→ SetDelayed
```

Theorem. A simplification rule.

```
In[69]:= SubstTest[union, cart[complement[domain[x]], set[0]],
          composite[VERTSECT[x], id[domain[x]]], x → inverse[RATIO]]

Out[69]= VERTSECT[inverse[RATIO]] == union[cart[complement[RATS], set[0]],
           composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], id[RATS]]]

In[70]:= VERTSECT[inverse[RATIO]] := union[cart[complement[RATS], set[0]],
           composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], id[RATS]]]
```

Observation. The canonical map for the equi-ratio relation is:

```
In[71]:= APPLY[VS, composite[inverse[RATIO], RATIO]]

Out[71]= composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]
```

Lemma.

```
In[72]:= member[composite[IMAGE[id[x]], RATIO], V] // AssertTest

Out[72]= member[composite[IMAGE[id[x]], RATIO], V] == True

In[73]:= member[composite[IMAGE[id[x_]], RATIO], V] := True
```

Theorem. An equation for the canonical map.

```
In[74]:= SubstTest[implies, and[member[x, y], member[x, domain[funpart[t]]]],
           member[APPLY[funpart[t], x], image[funpart[t], y]],
           {x -> composite[inverse[RATIO], RATIO], t → VS, y → EQV}] // Reverse

Out[74]= equal[composite[id[image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]], E],
           composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]] == True

In[75]:= Equal[composite[id[image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]], E],
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]]

Out[75]= composite[id[image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]], E] ==
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]

In[76]:= composite[id[image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]], E] :=
          composite[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATIO]
```

Corollary. The class obtained by removing the origin from each rational number is pairwise disjoint.

```
In[77]:= SubstTest[FUNCTION, composite[id[t], E],
          t -> image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]]

Out[77]= subclass[cart[image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS], image[
             IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]], union[DISJOINT, Id]] == True

In[78]:= subclass[cart[image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS],
            image[IMAGE[id[cart[complement[set[id[omega]]], V]]], RATS]],
           union[DISJOINT, Id]] := True
```

Theorem. Another simplification rule.

```
In[79]:= SubstTest[composite, inverse[E], VERTSECT[t], t → inverse[RATIO]] // Reverse

Out[79]= composite[inverse[E], IMAGE[id[cart[complement[set[id[omega]]], V]]], id[RATS]] ==
          inverse[RATIO]

In[80]:= composite[inverse[E], IMAGE[id[cart[complement[set[id[omega]]], V]]], id[RATS]] :=
          inverse[RATIO]
```