rational multiplication by negatives

Johan G. F. Belinfante
2012 October 17

```
In[1]:= SetDirectory["l:"]; << goedel.12oct15a

:Package Title: goedel.12oct15a             2012 October 15 at 10:30 a.m.

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

It is now:  2012 Oct 17 at 3:36

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 Oct 17 at 3:52
```

summary

The **negative** of a rational number **rat[x]** is **APPLY[inv[RATADD], rat[x]] = rat[x] ∘ INVERSE**.

```
In[2]:= APPLY[inv[RATADD], rat[x]]

Out[2]= composite[rat[x], INVERSE]
```

In the notes accompanying the theorems, the negative of a rational number **r** will be denoted by **– r**. In a product, the minus signs can always be brought outside: **(– r) · s = r · (– s) = – (r · s)**. In this notebook, all rules involve rational numbers with explicit **rat** wrappers.

simplification rules

Rational numbers commute with **INVERSE**. The **GOEDEL** program has the following rewrite rule.

```
In[3]:= composite[INVERSE, rat[x]]

Out[3]= composite[rat[x], INVERSE]
```

Theorem. Double negative rule.

```
In[4]:= ApComp[inv[RATADD], inv[RATADD], rat[x]]

Out[4]= composite[rat[x], id[P[cart[V, V]]]] == rat[x]
```

```
In[5]:= composite[rat[x_], id[P[cart[V, V]]]] := rat[x]
```

There is also a dual rule.

Theorem. Dual simplification rule.

```
In[6]:= Assoc[id[P[cart[V, V]]], id[Z], rat[x]]

Out[6]= composite[id[P[cart[V, V]]], rat[x]] == rat[x]

In[7]:= composite[id[P[cart[V, V]]], rat[x_]] := rat[x]
```

multiplication by negatives

The negative of a rational number is its product with minus one: **(– 1) · r = r · (– 1) = – r**.

```
In[8]:= ratmul[composite[id[Z], INVERSE], rat[x]]

Out[8]= composite[rat[x], INVERSE]

In[9]:= ratmul[rat[x], composite[id[Z], INVERSE]]

Out[9]= composite[rat[x], INVERSE]
```

Theorem. A special case. **(r · s) · (– 1) = – (r · s)**.

```
In[10]:= SubstTest[ratmul, rat[t],
           composite[id[Z], INVERSE], t → ratmul[rat[x], rat[y]]] // Reverse

Out[10]= ratmul[ratmul[rat[x], rat[y]], composite[id[Z], INVERSE]] ==
          composite[ratmul[rat[x], rat[y]], INVERSE]

In[11]:= ratmul[ratmul[rat[x_], rat[y_]], composite[id[Z], INVERSE]] :=
          composite[ratmul[rat[x], rat[y]], INVERSE]
```

Theorem. Bringing the minus sign outside.

```
In[12]:= SubstTest[ratmul, rat[x], ratmul[rat[y], z], z -> composite[id[Z], INVERSE]] // Reverse

Out[12]= ratmul[rat[x], composite[rat[y], INVERSE]] == composite[ratmul[rat[x], rat[y]], INVERSE]

In[13]:= ratmul[rat[x_], composite[rat[y_], INVERSE]] :=
          composite[ratmul[rat[x], rat[y]], INVERSE]
```

Corollary. Dual rule.

```
In[14]:= SubstTest[ratmul, u, ratmul[v, w],
          {u -> composite[id[Z], INVERSE], v → rat[x], w → rat[y]}]

Out[14]= ratmul[composite[rat[x], INVERSE], rat[y]] == composite[ratmul[rat[x], rat[y]], INVERSE]
```

```
In[15]:= ratmul[composite[rat[x_], INVERSE], rat[y_]] :=
          composite[ratmul[rat[x], rat[y]], INVERSE]
```

Lemma. Simplification rule.

```
In[16]:= SubstTest[composite, rat[t], id[P[cart[V, V]]], t -> ratmul[rat[x], rat[y]]] // Reverse

Out[16]= composite[ratmul[rat[x], rat[y]], id[P[cart[V, V]]]] == ratmul[rat[x], rat[y]]

In[17]:= composite[ratmul[rat[x_], rat[y_]], id[P[cart[V, V]]]] := ratmul[rat[x], rat[y]]
```

Theorem. The product of two negatives: **(– r) · (– s) = r · s**.

```
In[18]:= SubstTest[ratmul, composite[rat[x], INVERSE],
           rat[t], t → composite[rat[y], INVERSE]] // Reverse

Out[18]= ratmul[composite[rat[x], INVERSE], composite[rat[y], INVERSE]] == ratmul[rat[x], rat[y]]

In[19]:= ratmul[composite[rat[x_], INVERSE], composite[rat[y_], INVERSE]] :=
          ratmul[rat[x], rat[y]]
```

---

variable-free rules

Theorem.

```
In[20]:= Assoc[RATMUL, composite[cross[Id, RATMUL], ASSOC], RIGHT[composite[id[Z], INVERSE]]]

Out[20]= composite[RATMUL, cross[Id, inv[RATADD]]] == composite[inv[RATADD], RATMUL]

In[21]:= composite[RATMUL, cross[Id, inv[RATADD]]] := composite[inv[RATADD], RATMUL]
```

Corollary. Dual rule.

```
In[22]:= Assoc[RATMUL, cross[Id, inv[RATADD]], SWAP]

Out[22]= composite[RATMUL, cross[inv[RATADD], Id]] == composite[inv[RATADD], RATMUL]

In[23]:= composite[RATMUL, cross[inv[RATADD], Id]] := composite[inv[RATADD], RATMUL]
```

Corollary. Rule for the product of two negatives.

```
In[24]:= Assoc[RATMUL, cross[inv[RATADD], Id], cross[Id, inv[RATADD]]]

Out[24]= composite[RATMUL, cross[inv[RATADD], inv[RATADD]]] == RATMUL

In[25]:= composite[RATMUL, cross[inv[RATADD], inv[RATADD]]] := RATMUL
```