APPLY[CURRY, RATMUL] = RATTIMES

Johan G. F. Belinfante
2012 November 15

```
In[1]:= SetDirectory["l:"]; << goedel.12nov13a
```

```
:Package Title: goedel.12nov13a              2012 November 13 at 5:00 a.m.

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

It is now:  2012 Nov 15 at 11:24

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 Nov 15 at 11:40
```

summary

The function **RATTIMES** is defined, and some of its properties are derived.

definition of RATTIMES

Definition.

```
In[2]:= APPLY[CURRY, RATMUL] := RATTIMES
```

basic properties

Theorem.

```
In[3]:= SubstTest[FUNCTION, APPLY[CURRY, binop[x]], x → RATMUL] // Reverse
```

```
Out[3]= FUNCTION[RATTIMES] == True
```

```
In[4]:= FUNCTION[RATTIMES] := True
```

Theorem.

```
In[5]:= SubstTest[domain, APPLY[CURRY, binop[x]], x → RATMUL] // Reverse

Out[5]= domain[RATTIMES] == RATS

In[6]:= domain[RATTIMES] := RATS
```

Theorem.

```
In[7]:= SubstTest[member, APPLY[CURRY, binop[x]], map[fix[domain[binop[x]]],
          map[fix[domain[binop[x]]], fix[domain[binop[x]]]]], x → RATMUL] // Reverse

Out[7]= member[RATTIMES, map[RATS, map[RATS, RATS]]] == True

In[8]:= member[RATTIMES, map[RATS, map[RATS, RATS]]] := True
```

APPLY and vertical section rules

Theorem. Function application rule.

```
In[9]:= SubstTest[APPLY, APPLY[CURRY, binop[t]], x, t → RATMUL] // Reverse

Out[9]= APPLY[RATTIMES, x] ==
         union[complement[image[V, intersection[RATS, set[x]]]], rattimes[x]]

In[10]:= APPLY[RATTIMES, x_] :=
          union[complement[image[V, intersection[RATS, set[x]]]], rattimes[x]]
```

Theorem. Vertical section rule.

```
In[11]:= SubstTest[image, funpart[t], set[x], t → RATTIMES] // Reverse

Out[11]= image[RATTIMES, set[x]] ==
          intersection[image[V, intersection[RATS, set[x]]], set[rattimes[x]]]

In[12]:= image[RATTIMES, set[x_]] :=
          intersection[image[V, intersection[RATS, set[x]]], set[rattimes[x]]]
```

Theorem. Inverse image rule.

```
In[13]:= (member[x, image[inverse[funpart[t]], y]] // AssertTest) /. t → RATTIMES

Out[13]= member[x, image[inverse[RATTIMES], y]] == and[member[x, RATS], member[rattimes[x], y]]

In[14]:= member[x_, image[inverse[RATTIMES], y_]] := and[member[x, RATS], member[rattimes[x], y]]
```

eval and FUNPART rules

Theorem. Evaluation rule.

```
In[15]:= SubstTest[composite, eval[x], APPLY[CURRY, binop[t]], t → RATMUL] // Reverse
Out[15]= composite[eval[x], RATTIMES] == rattimes[x]

In[16]:= composite[eval[x_], RATTIMES] := rattimes[x]
```

Corollary.

```
In[22]:= Map[flip[rotate[inverse[#]]] &,
          SubstTest[reify, x, composite[eval[x], t], t → RATTIMES]]
Out[22]= composite[inverse[SINGLETON], IMG, cross[RATTIMES, SINGLETON]] == RATMUL

In[23]:= composite[inverse[SINGLETON], IMG, cross[RATTIMES, SINGLETON]] := RATMUL
```

Theorem. Simplification rule.

```
In[17]:= SubstTest[composite, FUNPART, APPLY[CURRY, binop[x]], x → RATMUL] // Reverse
Out[17]= composite[FUNPART, RATTIMES] == RATTIMES

In[18]:= composite[FUNPART, RATTIMES] := RATTIMES
```

Theorem. Simplification rule.

```
In[24]:= ImageComp[FUNPART, RATTIMES, x] // Reverse
Out[24]= image[FUNPART, image[RATTIMES, x]] == image[RATTIMES, x]

In[25]:= image[FUNPART, image[RATTIMES, x_]] := image[RATTIMES, x]
```

Corollary. Simplification rule.

```
In[26]:= Assoc[id[P[cart[V, V]]], FUNPART, RATTIMES]
Out[26]= composite[id[P[cart[V, V]]], RATTIMES] == RATTIMES

In[27]:= composite[id[P[cart[V, V]]], RATTIMES] := RATTIMES
```

Corollary.

```
In[28]:= Assoc[IMAGE[SWAP], id[P[cart[V, V]]], RATTIMES]
Out[28]= composite[IMAGE[SWAP], RATTIMES] == composite[INVERSE, RATTIMES]

In[29]:= composite[IMAGE[SWAP], RATTIMES] := composite[INVERSE, RATTIMES]
```

Corollary.

```
In[30]:= ImageComp[id[P[cart[V, V]]], RATTIMES, x] // Reverse
Out[30]= intersection[image[RATTIMES, x], P[cart[V, V]]] == image[RATTIMES, x]

In[31]:= intersection[image[RATTIMES, x_], P[cart[V, V]]] := image[RATTIMES, x]
```

normalization rules

Theorem. Normalization rule.

```
In[32]:= SubstTest[VERTSECT, inverse[rotate[binop[x]]], x → RATMUL] // Reverse

Out[32]= VERTSECT[inverse[rotate[RATMUL]]] ==
          union[cart[complement[RATS], set[0]], composite[INVERSE, RATTIMES]]

In[33]:= VERTSECT[inverse[rotate[RATMUL]]] :=
          union[cart[complement[RATS], set[0]], composite[INVERSE, RATTIMES]]
```

Corollary.

```
In[38]:= Map[composite[SWAP, #] &, SubstTest[composite,
            inverse[E], VERTSECT[t], t -> inverse[rotate[RATMUL]]]] // Reverse

Out[38]= composite[inverse[E], RATTIMES] == composite[SWAP, inverse[rotate[RATMUL]]]

In[40]:= composite[inverse[E], RATTIMES] := composite[SWAP, inverse[rotate[RATMUL]]]
```

Theorem. Serendipity.

```
In[49]:= image[inverse[RATTIMES], set[0]] // Renormality

Out[49]= image[inverse[RATTIMES], set[0]] == 0

In[50]:= image[inverse[RATTIMES], set[0]] := 0
```