divisibility relations for cat[x]

Johan G. F. Belinfante
2009 January 7

```
In[1]:= SetDirectory["l:"]; << goedel.09jan06a;<< tools.m

        :Package Title: goedel.09jan06a         2009 January 6 at 3:45 p.m.

        It is now:  2009 Jan 7 at 11:0

        Loading Simplification Rules

        TOOLS.M                                      Revised 2008 December 26

        weightlimit = 40
```

summary

The left and right divisibility relations for **cat[x]** are reflexive and transitive.

left-divisibility

Lemma.

```
In[5]:= SubstTest[implies, subclass[u, v], subclass[domain[u], domain[v]],
          {u -> dom[cat[x]], v -> fix[composite[inverse[FIRST], cat[x]]]}] // Reverse

Out[5]= subclass[range[cat[x]], fix[composite[cat[x], inverse[FIRST]]]] == True

In[6]:= (% /. x → x_) /. Equal→ SetDelayed
```

Theorem.

```
In[7]:= SubstTest[subclass, id[udora[t]], t, t -> composite[cat[x], inverse[FIRST]]]

Out[7]= REFLEXIVE[composite[cat[x], inverse[FIRST]]] == True

In[8]:= REFLEXIVE[composite[cat[x_], inverse[FIRST]]] := True
```

Corollary. Strengthening of the lemma to an equation.

```
In[9]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> fix[composite[cat[x], inverse[FIRST]]], v → range[cat[x]]}]

Out[9]= equal[fix[composite[cat[x], inverse[FIRST]]], range[cat[x]]] == True
```

```
In[10]:= fix[composite[cat[x_], inverse[FIRST]]] := range[cat[x]]
```

Theorem.

```
In[15]:= SubstTest[TRANSITIVE, composite[assoc[t], inverse[FIRST]], t → cat[x]] // Reverse

Out[15]= TRANSITIVE[composite[cat[x], inverse[FIRST]]] == True
```

```
In[16]:= TRANSITIVE[composite[cat[x_], inverse[FIRST]]] := True
```

Corollary.

```
In[17]:= SubstTest[composite, trv[rfx[t]], trv[rfx[t]],
           t -> composite[cat[x], inverse[FIRST]]] // Reverse

Out[17]= composite[cat[x], inverse[FIRST], cat[x], inverse[FIRST]] ==
          composite[cat[x], inverse[FIRST]]
```

```
In[18]:= composite[cat[x_], inverse[FIRST], cat[x_], inverse[FIRST]] :=
          composite[cat[x], inverse[FIRST]]
```

---

right-divisibility

Corollary.

```
In[20]:= SubstTest[REFLEXIVE, composite[cat[t], inverse[FIRST]], t → flip[cat[x]]] // Reverse

Out[20]= REFLEXIVE[composite[cat[x], inverse[SECOND]]] == True
```

```
In[21]:= REFLEXIVE[composite[cat[x_], inverse[SECOND]]] := True
```

Corollary.

```
In[22]:= SubstTest[TRANSITIVE, composite[cat[t], inverse[FIRST]], t → flip[cat[x]]] // Reverse

Out[22]= TRANSITIVE[composite[cat[x], inverse[SECOND]]] == True
```

```
In[23]:= TRANSITIVE[composite[cat[x_], inverse[SECOND]]] := True
```

Corollary.

```
In[24]:= SubstTest[fix, composite[cat[t], inverse[FIRST]], t → flip[cat[x]]] // Reverse

Out[24]= fix[composite[cat[x], inverse[SECOND]]] == range[cat[x]]
```

```
In[25]:= fix[composite[cat[x_], inverse[SECOND]]] := range[cat[x]]
```

Corollary.

In[26]:=
```
SubstTest[composite, trv[rfx[t]], trv[rfx[t]],
  t -> composite[cat[x], inverse[SECOND]]] // Reverse
```

Out[26]=
```
composite[cat[x], inverse[SECOND], cat[x], inverse[SECOND]] ==
 composite[cat[x], inverse[SECOND]]
```

In[27]:=
```
composite[cat[x_], inverse[SECOND], cat[x_], inverse[SECOND]] :=
 composite[cat[x], inverse[SECOND]]
```