natmod[x,y] < x

Johan G. F. Belinfante
2007 March 17

```
In[1]:= SetDirectory["l:"]; << goedel91.16a; << tools.m

        :Package Title: goedel91.16a        2007 March 16 at 7:05 p.m.

        It is now:  2007 Mar 17 at 11:46

        Loading Simplification Rules

        TOOLS.M                       Revised 2007 March 3

        weightlimit = 40
```

summary

A rewrite rule is derived for the condition **member[natmod[x,y], x]**.

numberhood lemmas

Lemma 1.

```
In[2]:= Map[or[member[x, omega], #] &,
          SubstTest[implies, member[t, z], member[t, V], t → natmod[x, y]]] // Reverse

Out[2]= or[member[x, omega], not[member[natmod[x, y], z]]] == True

In[3]:= or[member[x_, omega], not[member[natmod[x_, y_], z_]]] := True
```

Lemma 2.

```
In[4]:= Map[or[member[y, omega], #] &,
          SubstTest[implies, member[t, z], member[t, V], t → natmod[x, y]]] // Reverse

Out[4]= or[member[y, omega], not[member[natmod[x, y], z]]] == True

In[5]:= or[member[y_, omega], not[member[natmod[x_, y_], z_]]] := True
```

the condition natmod[x,y] < x

A natural number cannot be less than itself.

```
In[6]:= SubstTest[implies, and[member[u, omega], equal[u, v]],
          not[member[u, v]], {u → natmod[x, y], v → x}] // MapNotNot // Reverse

Out[6]= or[not[member[x, y]], not[member[y, omega]], not[member[natmod[x, y], x]]] == True

In[7]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The numberhood literal is redundant.

```
In[8]:= Map[not,
          SubstTest[and, implies[p1, p2], not[implies[p1, p3]], {p1 -> member[natmod[x, y], x],
            p2 → member[y, omega], p3 → not[member[x, y]]}] // Reverse

Out[8]= or[not[member[x, y]], not[member[natmod[x, y], x]]] == True

In[9]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The following temporary lemma, using **nat** wrappers will be subsumed by the main theorem.

```
In[10]:= SubstTest[and, subclass[nat[t], nat[x]],
           not[equal[nat[t], nat[x]]], t → natmod[nat[x], nat[y]]]

Out[10]= member[natmod[nat[x], nat[y]], nat[x]] ==
           and[not[equal[0, nat[y]]], not[member[nat[x], nat[y]]]]

In[11]:= member[natmod[nat[x_], nat[y_]], nat[x_]] :=
           and[not[equal[0, nat[y]]], not[member[nat[x], nat[y]]]]
```

Eliminating the **nat** wrappers yields:

```
In[12]:= SubstTest[implies, and[equal[x, nat[u]], equal[y, nat[v]]],
           or[equal[0, y], member[x, y], member[natmod[x, y], x]], {u → x, v → y}] // Reverse

Out[12]= or[equal[0, y], member[x, y], member[natmod[x, y], x],
           not[member[x, omega]], not[member[y, omega]]] == True

In[13]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Main theorem.

```
In[14]:= equiv[member[natmod[x, y], x], and[member[x, omega],
             member[y, omega], not[equal[0, y]], not[member[x, y]]]] // not // not

Out[14]= True

In[15]:= member[natmod[x_, y_], x_] :=
           and[member[x, omega], member[y, omega], not[equal[0, y]], not[member[x, y]]]
```