NATMUL is a function

Johan G. F. Belinfante
2002 June 30

```海湾goedel52.o82;海湾tools.m
:Package Title: goedel52.o82 2002 June 30 at 11:30 a.m.
It is now: 2002 Jun 30 at 16:35
Loading Simplification Rules
TOOLS.M Revised 2002 June 12
weightlimit = 40

■ Introduction

In this notebook an old definition of natural multiplication is resurrected, and some of its properties are derived.

■ a thin normality rule

The following special Normality rule is needed to deal with the thin expressions arising in the theory of natural multiplication.

```iterate[th, singleton[y]] // Normality // Reverse
```

```U[intersection[P[cart[omega, V]]], subvar[
union[cart[cart[singleton[0], singleton[y]]], cart[singleton[0], singleton[y]]],
cross[SUCC, th]])]] == iterate[th, singleton[y]]
```

```U[intersection[P[cart[omega, V]]], subvar[
union[cart[cart[singleton[0], singleton[y]], singleton[y]], cart[singleton[0], singleton[y_]]]],
cross[SUCC, x_] ]]]] := iterate[x, singleton[y]] // thin[x]
```

■ Definition of NATMUL

We explore again the idea that multiplication of natural numbers be based on the following membership rule:

```member[x_, NATMUL] := and[member[first[first[x]], omega],
member[pair[second[first[x]], second[x]],
iterate[iterate[SUCC, singleton[first[first[x]]]], singleton[0]]]]
```

The key to further progress is this formula:
composite[NATMUL, LEFT[x]] // VSNormality

composite[NATMUL, LEFT[x]] == composite[id[image[V, intersection[omega, singleton[x]]]], iterate[iterate[SUCC, singleton[x]], singleton[0]]]

composite[NATMUL, LEFT[x_]] :=
composite[id[image[V, intersection[omega, singleton[x]]]], iterate[iterate[SUCC, singleton[x]], singleton[0]]]

A second key formula is obtained this way:

ImageComp[NATMUL, LEFT[x], y] // Reverse

image[NATMUL, cart[singleton[x], y]] ==
intersection[image[V, intersection[omega, singleton[x]]], image[iterate[iterate[SUCC, singleton[x]], singleton[0]], y]]

image[NATMUL, cart[singleton[x], y_]] :=
intersection[image[V, intersection[omega, singleton[x]]], image[iterate[iterate[SUCC, singleton[x]], singleton[0]], y]]

### Corollaries

Some corollaries that do not require separate rules:

```plaintext
composite[NATMUL, LEFT[0]]
cart[omega, singleton[0]]

composite[NATMUL, LEFT[singleton[0]]]
id[omega]

FUNCTION[composite[NATMUL, LEFT[x]]]
True
```

### NATMUL is a binary function

Later on a more careful investigation of the domain and range of NATMUL will be undertaken. For now, all one needs is rather basic information:

```plaintext
Map[equal[0, #] &, symdif[NATMUL, composite[NATMUL, id[cart[V, V]]]]] // Normality
subclass[NATMUL, cart[cart[V, V], V]] == True

subclass[NATMUL, cart[cart[V, V], V]] := True

equal[composite[NATMUL, id[cart[V, V]]], NATMUL] // AssertTest
equal[NATMUL, composite[NATMUL, id[cart[V, V]]]] == True

composite[NATMUL, id[cart[V, V]]] := NATMUL
```
Evidence of commutativity

The following formulas are similar to their \texttt{LEFT} counterparts.

\begin{verbatim}
composite[NATMUL, RIGHT[0]] // VSNormality
composite[NATMUL, RIGHT[0]] == cart[omega, singleton[0]]

composite[NATMUL, RIGHT[0]] := cart[omega, singleton[0]]

composite[NATMUL, RIGHT[singleton[0]]] // VSNormality
composite[NATMUL, RIGHT[singleton[0]]] == id[omega]

composite[NATMUL, RIGHT[singleton[0]]] := id[omega]
\end{verbatim}