corrected version of Quaife's Theorem (O16)

Johan G. F. Belinfante
2002 October 14

```
<< goedel52.p89; << tools.m
```

```
:Package Title: goedel52.p89          2002 October 14 at 1:55 p.m.
```

```
It is now:  2002 Oct 14 at 16:50
```

```
Loading Simplification Rules
```

```
TOOLS.M                   Revised 2002 September 16
```

```
weightlimit = 40
```

■ summary

In this notebook a corrected version of Quaife's theorem **(O16)** is derived.

```
Art Quaife, Automated Development of Fundamental Mathematical Theories,
Kluwer Academic Publishers, Dordrecht, 1992. (See page 190.)
```

■ preliminaries

The following facts will be needed in the sequel.

```
Map[or[member[x, omega], #] &, SubstTest[implies, member[w, z], member[w, V],
    w -> natmul[x, y]]]
```

```
or[member[x, omega], not[member[natmul[x, y], z]]] == True
```

```
or[member[x_, omega], not[member[natmul[x_, y_], z_]]] := True
```

```
SubstTest[implies, and[member[y, z], subclass[z, w]],
 member[y, w], {z -> singleton[0], w -> omega}]
```

```
or[member[y, omega], not[equal[0, y]]] == True
```

```
or[member[y_, omega], not[equal[0, y_]]] := True
```

■ conversion to strict form

A weak version of Theorem **(O16)** was proved previously. Conversion to strict form adds extra hypotheses, which will shortly be eliminated:

```
basicidea = (SubstTest[member, pair[u, v], intersection[cart[w, w], complement[E]],
    {u -> natmul[x, y], v -> x, w -> omega}])

or[and[equal[0, x], member[x, omega], member[y, omega]],
  and[member[x, omega], member[y, omega], not[equal[0, y]]]] ==
 and[member[x, omega], member[y, omega], not[member[natmul[x, y], x]]]
```

Transpose everything to one side:

```
Map[implies[First[basicidea], #] &, basicidea] // Reverse // MapNotNot

or[equal[0, y], not[member[x, omega]],
  not[member[y, omega]], not[member[natmul[x, y], x]]] == True

or[equal[0, y_], not[member[x_, omega]],
  not[member[y_, omega]], not[member[natmul[x_, y], x_]]] := True

Map[implies[Last[basicidea], #] &, basicidea]

or[equal[0, x], member[natmul[x, y], x],
  not[equal[0, y]], not[member[x, omega]], not[member[y, omega]]] == True

or[equal[0, x_], member[natmul[x_, y_], x_],
  not[equal[0, y_]], not[member[x_, omega]], not[member[y_, omega]]] := True
```

Combining these two facts produces one direction of Quaife's theorem **(O16)**:

```
Map[not,
 SubstTest[and, implies[p1, or[not[p2], not[p3], p4]], implies[p1, p2], implies[p1, p3],
        not[implies[p1, p4]],
{p1 -> member[natmul[x, y], x],
   p2 -> member[x, omega], p3 -> member[y, omega], p4 -> equal[0, y]}]]

or[equal[0, y], not[member[natmul[x, y], x]]] == True

or[equal[0, y_], not[member[natmul[x_, y_], x_]]] := True
```

■ corrected converse to Quaife's theorem (O16).

Quaife's theorem as stated is not valid in the opposite direction; the example **x = y = 0** is a counterexample:

```
implies[equal[0, y], member[natmul[x, y], x]] /. {x -> 0, y -> 0}

False
```

The derivation of a corrected converse is not difficult.

```
Map[not, SubstTest[and, implies[and[not[p1], p2, p3], not[p6]],
  not[implies[and[not[p1], p2, p3], p5]], implies[p2, p4],
    implies[and[p3, p4, not[p6]], p5],
  {p1 -> equal[0, x], p2 -> equal[0, y], p3 -> member[x, omega], p4 -> member[y, omega],
    p5 -> member[natmul[x, y], x], p6 -> subclass[x, natmul[x, y]]}]]

or[equal[0, x], member[natmul[x, y], x], not[equal[0, y]], not[member[x, omega]]] == True

or[equal[0, x_], member[natmul[x_, y_], x_],
  not[equal[0, y_]], not[member[x_, omega]]] := True
```

■ final steps

The various facts that have been proved above just need to be combined by an application of double negation.

```
or[and[equal[0, y], member[x, omega], not[equal[0, x]]],
  not[member[natmul[x, y], x]]] // NotNotTest
```

```
or[and[equal[0, y], member[x, omega], not[equal[0, x]]],
  not[member[natmul[x, y], x]]] == True
```

```
or[and[equal[0, y_], member[x_, omega], not[equal[0, x_]]],
  not[member[natmul[x_, y_], x_]]] := True
```

At this point the **GOEDEL** program recognizes the truth of the following corrected version of Quaife's Theorem **(O16)**.

```
equiv[member[natmul[x, y], x], and[member[x, omega], not[equal[0, x]], equal[0, y]]]
```

```
True
```

The following rewrite rule may therefore be added:

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