addition and subtraction

Johan G. F. Belinfante
2002 October 31

```
<< goedel52.q12; << tools.m

:Package Title: goedel52.q12          2002 October 30 at 5:52 a.m.

It is now:  2002 Oct 31 at 12:19

Loading Simplification Rules

TOOLS.M                      Revised 2002 October 29

weightlimit = 40
```

■ summary

For natural numbers, unlike for integers, addition and subtraction do not quite commute. For example, **(0 + 1) −1** is not the same as **(0 −1) + 1** because the latter involves the term **0 −1** which is not a natural number. The functors **natadd** and **natsub** are defined to be equal to **V** when applied to arguments that do not produce natural numbers. A formula is derived in this notebook which expresses the fact that addition and subtraction of natural numbers do commute when the result is a natural number, and takes care of the unnatural cases as well. A variable−freeversion of this law is also derived. The proposed orientation of these formulas as rewrite rules is tentative.

■ a lemma

To simplify the final formulas, a temporary lemma is useful:

```
image[V, intersection[complement[natadd[x, z]], natadd[y, z]]] // Normality

image[V, intersection[complement[natadd[x, z]], natadd[y, z]]] ==
 union[intersection[complement[image[V, intersection[omega, singleton[y]]]], image[V,
    intersection[omega, singleton[x]]], image[V, intersection[omega, singleton[z]]]],
  intersection[image[V, intersection[omega, singleton[x]]], image[V,
    intersection[omega, singleton[z]]], image[V, intersection[y, complement[x]]]]]

image[V, intersection[complement[natadd[x_, z_]], natadd[y_, z_]]] :=
 union[intersection[complement[image[V, intersection[omega, singleton[y]]]], image[V,
    intersection[omega, singleton[x]]], image[V, intersection[omega, singleton[z]]]],
  intersection[image[V, intersection[omega, singleton[x]]], image[V,
    intersection[omega, singleton[z]]], image[V, intersection[y, complement[x]]]]]
```

■ the basic result

The main formula can be derived in a single step:

```
Map[A[singleton[#]] &, SubstTest[natadd, natsub[u, v], natsub[v, w],
    {u -> natadd[y, x], v -> natadd[z, x], w -> z}]]
```

```
natadd[x, natsub[y, z]] ==
 union[image[V, intersection[z, complement[y]]], natsub[natadd[x, y], z]]
```

The following orientation of this formula as a rewrite rule is tentative:

```
natadd[x_, natsub[y_, z_]] :=
 union[image[V, intersection[z, complement[y]]], natsub[natadd[x, y], z]]
```

■ a variable−freeversion

A quick way to derive a variable−freeversion of this result uses **symdif** and **SubstTest**.

```
symdif[composite[rotate[NATADD], cross[NATADD, Id], id[composite[inverse[S], SECOND]]],
  composite[NATADD, cross[Id, rotate[NATADD]], ASSOC]] // VSTerNormality
```

```
union[composite[intersection[composite[complement[rotate[NATADD]], cross[NATADD, Id]],
    composite[NATADD, cross[Id, rotate[NATADD]], ASSOC]],
   id[composite[inverse[S], SECOND]]],
  composite[intersection[composite[rotate[NATADD], cross[NATADD, Id]],
    composite[complement[NATADD], cross[Id, rotate[NATADD]], ASSOC]],
   id[composite[inverse[S], SECOND]]]] == 0
```

```
union[composite[intersection[composite[complement[rotate[NATADD]], cross[NATADD, Id]],
    composite[NATADD, cross[Id, rotate[NATADD]], ASSOC]],
   id[composite[inverse[S], SECOND]]],
  composite[intersection[composite[rotate[NATADD], cross[NATADD, Id]],
    composite[complement[NATADD], cross[Id, rotate[NATADD]], ASSOC]],
   id[composite[inverse[S], SECOND]]]] := 0
```

It is not clear how to orient the following formula as a rewrite rule, but we tentatively opt to orient it the same way as the formula for variables.

```
SubstTest[equal, 0, symdif[u, v],
    {u ->
   composite[rotate[NATADD], cross[NATADD, Id], id[composite[inverse[S], SECOND]]],
  v -> composite[NATADD, cross[Id, rotate[NATADD]], ASSOC]}]
```

```
True == equal[composite[NATADD, cross[Id, rotate[NATADD]], ASSOC],
  composite[rotate[NATADD], cross[NATADD, Id], id[composite[inverse[S], SECOND]]]]
```

Before making this into a rewrite rule it is useful to move the **ASSOC** to the other side.

```
Map[composite[#, inverse[ASSOC]] &,
 composite[NATADD, cross[Id, rotate[NATADD]], ASSOC] ==
  composite[rotate[NATADD], cross[NATADD, Id], id[composite[inverse[S], SECOND]]]]
```

```
composite[NATADD, cross[Id, rotate[NATADD]]] == composite[rotate[NATADD],
  cross[NATADD, Id], id[composite[inverse[S], SECOND]], inverse[ASSOC]]
```

```
composite[NATADD, cross[Id, rotate[NATADD]]] := composite[rotate[NATADD],
  cross[NATADD, Id], id[composite[inverse[S], SECOND]], inverse[ASSOC]]
```

A flipped version of this can be derived as well:

```
Assoc[NATADD, cross[Id, rotate[NATADD]], SWAP]

composite[NATADD, cross[rotate[NATADD], Id]] ==
 composite[rotate[NATADD], cross[NATADD, Id], id[composite[inverse[S], SECOND]], ROT]

composite[NATADD, cross[rotate[NATADD], Id]] :=
 composite[rotate[NATADD], cross[NATADD, Id], id[composite[inverse[S], SECOND]], ROT]
```