Addition of natural numbers: rewrite rules and attributes for natadd.

Johan G. F. Belinfante  
2002 July 28

In this notebook, basic rewrite rules involving natadd[x,y] are derived. Old rules related to the associative law of addition and the additive law of exponents are replaced. Justification is given for setting Flat and Orderless attributes for natadd.

### Normality for natadd[x,y]

The Normality rule is needed for just about everything else, so it is a natural place to start.

```
natadd[x, y] // Normality // Reverse
A[image[NATADD, cart[singleton[x], singleton[y]]]] == natadd[x, y]
A[image[NATADD, cart[singleton[x_], singleton[y_]]]] := natadd[x, y]
```

From this it follows that natadd[x,y] is equal to V unless x and y are both natural numbers:

```
SubstTest[equal, V, A[image[NATADD, cart[u, v]]], {u -> singleton[x], v -> singleton[y]}]
equal[V, natadd[x, y]] == or[not(member[x, omega]], not(member[y, omega]])
```

This is added as a rewrite rule, used below to derive the commutative property of natadd.

```
equal[V, natadd[x_, y_]] := or[not(member[x, omega]], not(member[y, omega]])
```
Commutativity and the Orderless attribute

The commutativity of \texttt{NATADD} is used to derive the corresponding property of \texttt{natadd}.

\[
\begin{align*}
\text{SubstTest}[\text{implies, equal}[u, v], \text{equal}[A[u], A[v]],
\{u \rightarrow \text{image}[\text{NATADD}, \text{cart}[\text{singleton}[x], \text{singleton}[y]]],
\quad v \rightarrow \text{image}[\text{NATADD}, \text{cart}[\text{singleton}[y], \text{singleton}[x]]]\}]
\text{equal}[\text{natadd}[x, y], \text{natadd}[y, x]] &= \text{True}
\end{align*}
\]

This commutative property justifies adding the attribute \texttt{Orderless} to \texttt{natadd}.

\texttt{SetAttributes[natadd, Orderless]}

Now \textit{Mathematica} recognizes that \texttt{natadd} is commutative without needing to add any rewrite rules.

\[
\text{equal}[\text{natadd}[x, y], \text{natadd}[y, x]]
\]

\text{True}

Vertical section rule for NATADD

A vertical section rule for the binary function \texttt{NATADD} is deduced by using rules for \texttt{PAIR}:

\[
\begin{align*}
\text{SubstTest}[\text{implies, FUNCTION}[z],
\text{equal}[\text{image}[z, \text{singleton}[w]], \text{singleton}[A[\text{image}[z, \text{singleton}[w]]]],
\quad \{z \rightarrow \text{NATADD}, w \rightarrow \text{PAIR}[x, y]\}]
\text{equal}[\text{image}[\text{NATADD}, \text{cart}[\text{singleton}[x], \text{singleton}[y]]], \text{singleton}[\text{natadd}[x, y]]] &= \text{True}
\end{align*}
\]

This justifies adding the following rewrite rule:

\[
\text{image}[\text{NATADD}, \text{cart}[\text{singleton}[x_\_], \text{singleton}[y_\_]]] := \text{singleton}[\text{natadd}[x, y]]
\]

Sethood rule

From the vertical section rule one easily deduces a sethood rule:

\[
\text{Map}[\text{not, SubstTest}[\text{equal, 0}, \text{image}[\text{NATADD}, \text{singleton}[z]], z \rightarrow \text{PAIR}[x, y]]]
\text{member}[\text{natadd}[x, y], V] &= \text{and}[\text{member}[x, \omega], \text{member}[y, \omega]]
\text{member}[\text{natadd}[x_\_, y_\_], V] := \text{and}[\text{member}[x, \omega], \text{member}[y, \omega]]
\]

The following corollary is worth noting.
image[V, singleton[natadd[x, y]]] // Normality

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

image[V, singleton[natadd[x_, y_]]] :=
intersection[image[V, intersection[omega, singleton[x]]],
image[V, intersection[omega, singleton[y]]]]

This rewrite rule causes problems unless it is supplemented an additional rule. The **GOEDEL** program recognizes this truth:

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

True

On account of this, one is justified in adding a corresponding rewrite rule:

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

- **A consequence of the associative law**

There is an old rewrite rule that follows from the associative law:

composite[NATADD, RIGHT[x], NATADD, RIGHT[y]]

composite[NATADD, RIGHT[natadd[x, y]]]

This rule can be generalized. First, the old rule is removed:

composite[NATADD, RIGHT[x_], NATADD, RIGHT[y_]] =.

Instead of restoring it, we deduce a more general rewrite rule:

Assoc[composite[NATADD, cross[Id, NATADD]], ASSOC, RIGHT[x]] // Reverse

composite[NATADD, RIGHT[x], NATADD] ==
composite[NATADD, cross[Id, composite[NATADD, RIGHT[x]]]]

composite[NATADD, RIGHT[x_], NATADD] :=
composite[NATADD, cross[Id, composite[NATADD, RIGHT[x]]]]

The old rule follows from this new rule as a special case:

composite[NATADD, RIGHT[x], NATADD, RIGHT[y]]

composite[NATADD, RIGHT[natadd[x, y]]]

- **New additive law of exponents**

The **GOEDEL** program currently contains the following additive law of exponents which involves **iterate** and **SUCC**: 
We remove this old rule preparatory to deriving a better replacement rule.

\[
\text{composite}[\text{image}[\text{power}[x], u], \text{image}[\text{power}[x], v]]
\]

\[
\text{image}[\text{power}[x], \text{image}[\text{iterate}[\text{SUCC}, v], u]]
\]

The RIF version of the additive law of exponents is still available:

\[
\text{composite}[\text{RIF}, \text{cross}[\text{power}[x], \text{power}[x]]]
\]

\[
\text{composite}[\text{SWAP}, \text{power}[x], \text{NATADD}]
\]

From it we can deduce a replacement for the removed additive law of exponents:

\[
\text{ImageComp}[\text{composite}[\text{SWAP}, \text{RIF}], \text{cross}[\text{power}[x], \text{power}[x]], \text{cart}[u, v]] // \text{Reverse}
\]

\[
\text{composite}[\text{image}[\text{power}[x], u], \text{image}[\text{power}[x], v]] == \text{image}[\text{power}[x], \text{image}[\text{NATADD}, \text{cart}[u, v]]]
\]

\[
\text{composite}[\text{image}[\text{power}[x_], u_], \text{image}[\text{power}[x_], v_]] := \text{image}[\text{power}[x], \text{image}[\text{NATADD}, \text{cart}[u, v]]]
\]

The special case of greatest interest is the following, which involves \text{natadd}.

\[
\text{composite}[\text{image}[\text{power}[x], \text{singleton}[u]], \text{image}[\text{power}[x], \text{singleton}[v]]]
\]

\[
\text{image}[\text{power}[x], \text{singleton}[\text{natadd}[u, v]]]
\]

\[\textbf{The sum of natural numbers is a natural number}\]

We start with this result:

\[
\text{ImageComp}[\text{id}[\text{omega}], \text{NATADD}, \text{cart}[\text{singleton}[x], \text{singleton}[y]]] // \text{Reverse}
\]

\[
\text{intersection}[\text{omega}, \text{singleton}[\text{natadd}[x, y]]] == \text{singleton}[\text{natadd}[x, y]]
\]

\[
\text{intersection}[\text{omega}, \text{singleton}[\text{natadd}[x_], y_]] := \text{singleton}[\text{natadd}[x, y]]
\]

From it we deduce a temporary rule:

\[
\text{SubstTest}[\text{equal}, \text{intersection}[\text{omega}, \text{singleton}[w]],
\text{singleton}[w], w \rightarrow \text{natadd}[x, y]] // \text{Reverse}
\]

\[
\text{or}[\text{member}[\text{natadd}[x, y], \text{omega}], \text{not}[\text{member}[x, \text{omega}]], \text{not}[\text{member}[y, \text{omega}]]] == \text{True}
\]

\[
\text{or}[\text{member}[\text{natadd}[x_], y_], \text{omega}], \text{not}[\text{member}[x_], \text{omega}]], \text{not}[\text{member}[y_], \text{omega}]]] := \text{True}
\]

The converse also holds:

\[
\text{or}[\text{and}[\text{member}[x, \text{omega}], \text{member}[y, \text{omega}]],
\text{not}[\text{member}[\text{natadd}[x, y], \text{omega}]]] // \text{AssertTest}
\]

\[
\text{or}[\text{and}[\text{member}[x, \text{omega}], \text{member}[y, \text{omega}]], \text{not}[\text{member}[\text{natadd}[x, y], \text{omega}]]] == \text{True}
\]
These two results can be combined into a single rule:

\[
equiv[\text{member}[\text{nata}\text{dd}[x, y, \omega]], \text{and}][\text{member}[x, \omega], \text{member}[y, \omega]]] := \text{True}
\]

Various related facts are automatically recognized as consequences, and do not require additional rules. For example:

\[
\text{or}[\text{equal}][\text{nata}\text{dd}[x, y, V], \text{member}[\text{nata}\text{dd}[x, y, \omega]]]
\]

\[
\text{True}
\]

### Associativity and the Flat attribute

The **GOEDEL** program recognizes this truth:

\[
\text{equal}[
\text{union}[
\text{complement}][\text{image}[V, \text{intersection}][\omega, \text{singleton}[y]]], \text{nata}\text{dd}[x, \text{nata}\text{dd}[y, z]]],
\text{nata}\text{dd}[x, \text{nata}\text{dd}[y, z]]]
\]

\[
\text{True}
\]

Consequently one is justified in adding a corresponding rewrite rule:

\[
\text{union}[
\text{complement}][\text{image}[V, \text{intersection}][\omega, \text{singleton}[y]]],
\text{nata}\text{dd}[x, \text{nata}\text{dd}[y, z]]] :=
\text{nata}\text{dd}[x, \text{nata}\text{dd}[y, z]]
\]

With this rule in place, one deduces that **nata**dd is associative.

\[
\text{Map}[
\text{ImageComp}][\text{composite}[\text{NATADD}, \text{RIGHT}[x]], \text{composite}[\text{NATADD}, \text{RIGHT}[y]], \text{singleton}[z]]
\text{nata}\text{dd}[z, \text{nata}\text{dd}[x, y]]] == \text{nata}\text{dd}[x, \text{nata}\text{dd}[y, z]]
\]

This associative property justifies adding the attribute **Flat** to **nata**dd.

\[
\text{SetAttributes}[
\text{nata}\text{dd}, \text{Flat}]
\]

Now **Mathematica** recognizes that **nata**dd is associative without needing to add any rewrite rules.

\[
\text{nata}\text{dd}[z, \text{nata}\text{dd}[x, y]]] == \text{nata}\text{dd}[x, \text{nata}\text{dd}[y, z]]
\]

\[
\text{True}
\]