PLUS as an isomorphism

Johan G. F. Belinfante
2003 July 5

In[1]:= << goedel52.s43; << tools.m

:Package Title: goedel52.s43 2003 July 5 at 6:25 a.m.

It is now: 2003 Jul 8 at 11:39

Loading Simplification Rules

TOOLS.M Revised 2003 July 8

weightlimit = 40

■ summary

One can view the function PLUS as an isomorphism relating the associative operations NATADD and COMPOSE. The formulas are derived twice. First, an efficient derivation is given, which leads directly to the desired result. Then the same result is rederived in a more roundabout fashion, which was done mainly to discover useful rewrite rules that simplify expressions involving the function PLUS. The derivation of these simplification rules is speeded up dramatically by turning off the simplify flag, which controls certain conditional rewrite rules. Not only does turning off these conditional rules speed things up, but in the final step of the derivation, a better result is obtained with the flag off.

■ the efficient derivation

The only drawback of the quick derivation presented in this section is that the result to be derived must already be known. This is therefore a verification, not a discovery.

In[2]:= symdif[composite[PLUS, NATADD], composite[COMPOSE, cross[PLUS, PLUS]]] // VSTriNormality

composite[complement[PLUS], NATADD]], id[cart[V, V]]],
composite[intersection[composite[PLUS, NATADD],
composite[complement[COMPOSE], cross[PLUS, PLUS]]], id[cart[V, V]]]] == 0

In[3]:= % /. Equal -> SetDelayed

This verifies the main result in this notebook:

In[4]:= SubstTest[equal, 0, composite[symdif[u, v], id[cart[V, V]]],
{u -> composite[PLUS, NATADD], v -> composite[COMPOSE, cross[PLUS, PLUS]]}]


A rewrite rule for this result is not added at this time to allow further exploration to take place, with the aim of deriving new simplification rules that would permit results such as this to be discovered even if the result were not already known.
### simplification rules

In[5] :=  simplify = False;

Comment: The derivations go faster when one turns off the flag `simplify`. If this is not done, all but the last step still goes through, albeit much more slowly.


Out[6] =  composite[complement[E], SWAP, inverse[rotate[NATADD]]] ==
          composite[complement[S], PLUS]

In[7] :=  composite[complement[E], SWAP, inverse[rotate[NATADD]]] :=
          composite[complement[S], PLUS]

In[8] :=  composite[id[cart[V, omega]],
          inverse[rotate[composite[inverse[power[SUCC]], SWAP]]]] // DoubleInverse

Out[8] =  composite[id[cart[V, omega]],
          inverse[rotate[composite[inverse[power[SUCC]], SWAP]]]] ==
          composite[SWAP, inverse[rotate[NATADD]]]

In[9] :=  composite[id[cart[V, omega]],
          inverse[rotate[composite[inverse[power[SUCC]], SWAP]]]] :=
          composite[SWAP, inverse[rotate[NATADD]]]


Out[10] =  composite[INVERSE, inverse[UB[rotate[NATADD]]], id[omega]] ==
          composite[inverse[S], PLUS]

In[11] :=  composite[INVERSE, inverse[UB[rotate[NATADD]]], id[omega]] :=
          composite[INVERSE, inverse[UB[rotate[NATADD]]], id[omega]]

In[12] :=  image[inverse[PLUS], singleton[0]] // Normality

Out[12] =  image[inverse[PLUS], singleton[0]] == 0

In[13] :=  image[inverse[PLUS], singleton[0]] := 0

Temporary rule:

In[14] :=  PLUS // VSNormality // Reverse

           composite[S, PLUS]], id[omega]] == PLUS

In[15] :=  % /. Equal -> SetDelayed

Better result:

In[16] :=  intersection[composite[INVERSE, inverse[UB[rotate[NATADD]]]],
            composite[S, PLUS]] // VSNormality

Out[16] =  intersection[
            composite[INVERSE, inverse[UB[rotate[NATADD]]]], composite[S, PLUS]] == PLUS

In[17] :=  intersection[
            composite[INVERSE, inverse[UB[rotate[NATADD]]]], composite[S, PLUS]] := PLUS
Now the isomorphism theorem could be automatically discovered:

\[
\text{In[18]:=} \text{composite}[\text{COMPOSE}, \text{cross}[\text{PLUS}, \text{PLUS}]] \text{ // VSTriNormality}
\]

\[
\text{Out[18]=} \text{composite}[\text{COMPOSE}, \text{cross}[\text{PLUS}, \text{PLUS}]] \text{ == composite[PLUS, NATADD]}
\]

It is not clear how best to orient this equation as a rewrite rule. The following choice is tentative, based on analogy with other similar rules. The idea in favor of this choice is that pattern matching would be rendered more difficult were the equation to be flipped.

\[
\text{In[19]:=} \text{composite}[\text{PLUS}, \text{NATADD}] \text{ := composite[COMPOSE, cross[PLUS, PLUS]]}
\]

\section*{comment}

If the \texttt{simplify} flag were not turned off, the final step of the derivation presented in the preceding section does not go through. In the course of investigating this matter, it was discovered that the failure was due to the existence of yet another identity:

\[
\text{In[20]:=} \text{composite}[\text{inverse[rotate[composite[inverse[power[SUCC]], SWAP]]], id[omega]]} \text{ // inverse} \text{ // TripleRotate}
\]

\[
\text{Out[20]=} \text{composite[\text{id[omega]}, \text{rotate[composite[inverse[power[SUCC]], SWAP]]]]} \text{ == composite[rotate[NATADD], SWAP]}
\]

\[
\text{In[21]:=} \text{composite[\text{id[omega]}, \text{rotate[composite[inverse[power[SUCC]], SWAP]]]]} \text{ := composite[rotate[NATADD], SWAP]}
\]

The inverse of this gets wrapped with \texttt{VERTSECT}, and then simplification is prevented because the \texttt{id[omega]} factor slips outside. This is what happens:

\[
\text{In[22]:=} \text{composite[inverse[rotate[composite[inverse[power[SUCC]], SWAP]]], id[omega]]} \text{ // DoubleInverse}
\]

\[
\text{Out[22]=} \text{composite[\text{inverse[rotate[composite[inverse[power[SUCC]], SWAP]]], id[omega]]} \text{ == composite[SWAP, inverse[rotate[NATADD]]]}
\]

\[
\text{In[23]:=} \text{composite[\text{inverse[rotate[composite[inverse[power[SUCC]], SWAP]]], id[omega]]} \text{ := composite[\text{SWAP, inverse[rotate[NATADD]]]}}
\]

The following formula is the strange result that one finds when the \texttt{simplify} flag is not turned off:

\[
\text{In[24]:=} \text{Map[composite[\#, NATADD] \&, \text{SubstTest[VERTSECT, composite[x, id[omega]], x \rightarrow inverse[rotate[composite[inverse[power[SUCC]], SWAP]]]]]}}
\]

\[
\text{Out[24]=} \text{composite[\text{COMPOSE, cross[PLUS, PLUS]] \text{ == composite[\text{VERTSECT[\text{inverse[rotate[composite[inverse[power[SUCC]], SWAP]]]}}]}, NATADD]}}
\]