rotated associative relation

Johan G. F. Belinfante
2005 July 30

**summary**

Addition is related to subtraction by rotation. The associative law for addition implies a corresponding law for subtraction: \((x - y) - z = x - (y + z)\). In this notebook it is shown that a similar result holds for any associative relation.

**rotate and ASSOC**

Lemma.

\[
\text{In [12]} := \text{Assoc[rotate[inverse[rotate[x]]], id[composite[FIRST, rotate[x]]], id[cart[cart[V, V], V]]]} \text{// Reverse}
\]

\[
\text{Out [12]} = \text{composite[rotate[inverse[rotate[x]]], id[cart[cart[V, V], V]]]} := \text{rotate[inverse[rotate[x]]]}
\]

Theorem.

\[
\text{In [15]} := \text{composite[rotate[composite[x, ASSOC]], ASSOC]} \text{// VSTerNormality}
\]

\[
\text{Out [15]} = \text{composite[rotate[composite[x, ASSOC]], ASSOC]} = \text{rotate[inverse[rotate[x]]]}
\]

\[
\text{In [16]} := \text{composite[rotate[composite[x, ASSOC]], ASSOC]} := \text{rotate[inverse[rotate[x]]]}
\]
main theorem

Lemma.

\[ \text{In}[23]:= \ \text{composite[rotate[y], cross[x, Id]] // TripleRotate // Reverse} \]

\[ \text{Out}[23]= \ \text{rotate[composite[inverse[x], y]] := composite[rotate[y], cross[x, Id]]} \]

\[ \text{In}[24]:= \ \text{rotate[composite[inverse[x_], y_]] := composite[rotate[y], cross[x, Id]]} \]

The following observation is used in the next step.

\[ \text{In}[17]:= \ \text{abstract[x, composite[rotate[x], ASSOC]]} \]

\[ \text{Out}[17]= \ \text{composite[cross[inverse[AASSOC]], Id], ROT} \]

Theorem.

\[ \text{In}[25]:= \ \text{SubstTest[implies, equal[u, v], equal[image[w, u], image[w, v]],} \]
\[ \{u \rightarrow \text{composite[x, cross[x, Id]], v \rightarrow \text{composite[x, cross[Id, x], ASSOC]},} \]
\[ w \rightarrow \text{composite[cross[inverse[AASSOC], Id], ROT]}}) \]

\[ \text{Out}[25]= \ \text{or[equal[composite[rotate[x], cross[rotate[x], Id]]],} \]
\[ \text{composite[rotate[x], cross[Id, x], ASSOC]], not[equal[} \]
\[ \text{composite[x, cross[x, Id]], composite[x, cross[Id, x], ASSOC]]] \text{]} = \text{True} \]

\[ \text{In}[26]:= \ \text{or[equal[composite[rotate[x_], cross[rotate[x_], Id]]],} \]
\[ \text{composite[rotate[x_], cross[Id, x_], ASSOC]],} \]
\[ \text{not[equal[composite[x_, cross[x_, Id]],} \]
\[ \text{composite[x_, cross[Id, x_], ASSOC]]] \text{]} = \text{True} \]

Corollary.

\[ \text{In}[27]:= \ \text{Map[not, SubstTest[and, implies[p1, p2],} \]
\[ \text{implies[p2, p3], not[implies[p1, p3]],} \{p1 \rightarrow \text{associative[x]},} \]
\[ p2 \rightarrow \text{equal[composite[x, cross[x, Id]], composite[x, cross[Id, x], ASSOC]],} \]
\[ p3 \rightarrow \text{equal[composite[rotate[x], cross[rotate[x], Id]],} \]
\[ \text{composite[rotate[x], cross[Id, x], ASSOC]]] \text{]} \]

\[ \text{Out}[27]= \ \text{or[equal[composite[rotate[x], cross[rotate[x], Id]],} \]
\[ \text{composite[rotate[x], cross[Id, x], ASSOC]], not[associative[x]]] \text{]} = \text{True} \]

\[ \text{In}[28]:= \ \text{or[equal[composite[rotate[x_], cross[rotate[x_], Id]],} \]
\[ \text{composite[rotate[x_], cross[Id, x_], ASSOC]], not[associative[x_]]] \text{]} = \text{True} \]
An amusing application is presented in this section.

```
In[29]:= SubstTest[implies, associative[x],
      equal[composite[rotate[x]], cross[rotate[x], Id]],
      composite[rotate[x], cross[Id, x], ASSOC]], x \rightarrow inverse[DUP]]
```

```
Out[29]= equal[composite[SECOND, id[inverse[DUP]]],
      composite[FIRST, FIRST, id[inverse[DUP]]]] = True
```

```
In[30]:= composite[FIRST, FIRST, id[inverse[DUP]]] :=
      composite[SECOND, id[inverse[DUP]]]
```

The following formula is analogous.

```
In[31]:= composite[SECOND, id[inverse[DUP]]] // TripleRotate // Reverse
```

```
Out[31]= composite[SECOND, FIRST, id[inverse[DUP]]] :=
      composite[SECOND, id[inverse[DUP]]]
```

Two further corollaries:

```
In[33]:= Assoc[composite[FIRST, FIRST], id[inverse[DUP]], SWAP]
```

```
Out[33]= composite[FIRST, SECOND, id[DUP]] = composite[FIRST, id[DUP]]
```

```
In[34]:= Assoc[composite[SECOND, FIRST], id[inverse[DUP]], SWAP]
```

```
Out[34]= composite[SECOND, SECOND, id[DUP]] := composite[FIRST, id[DUP]]
```

```
In[35]:= composite[SECOND, SECOND, id[DUP]] := composite[FIRST, id[DUP]]
```