# binary homomorphisms, part 6

Johan G. F. Belinfante
2006 August 5

```
In[1]:= SetDirectory["l:"]; << goedel84.05a; << tools.m

        :Package Title: goedel84.05a        2006 August 5 at 4:30 a.m.

        It is now:  2006 Aug 5 at 17:23

        Loading Simplification Rules

        TOOLS.M                       Revised 2006 July 18

        weightlimit = 40
```

## summary

This sixth notebook on binary homomorphisms is concerned with sums of endomorphisms of **NATADD** and **INTADD**. If **f** and **g** are functions, then their sum **h = f + g** is defined by **h(x) = f(x) + g(x)**. This commonly used notation is suggestive, but the meaning of the plus sign is different for the two equations, and moreover, there is no way to determine whether the addition is to be interpreted in the arithmetic of natural numbers or that of integers. In this notebook, a more clumsy but less ambiguous notation is used. It is shown that the sum of endomorphisms of **NATADD** is an endomorphism of **NATADD**, and similarly for **INTADD**. The result could be generalized to an arbitrary commutative associative binary operation.

## sums of endomorphisms of NATADD

The **sum** of endomorphisms **x** and **y** of **NATADD** is defined to be **composite[NATADD, cross[x,y], DUP]**. The rewrite rules in the **GOEDEL** program transforms this as follows:

```
In[2]:= composite[NATADD, cross[x, y], DUP]

Out[2]= composite[NATADD,
         intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]
```

Any endomorphism of **NATADD** has the form **times[x]** for some natural number **x**, so the theorem to be derived amounts to the following observation:

```
In[3]:= composite[NATADD, intersection[
          composite[inverse[FIRST], times[x]], composite[inverse[SECOND], times[y]]]]

Out[3]= times[natadd[x, y]]
```

Lemma.

```
In[4]:= SubstTest[implies, and[member[x, map[y, z]], member[u, y]],
         member[APPLY[x, u], z], {u → set[0], y → omega, z → omega}]

Out[4]= or[member[APPLY[x, set[0]], omega], not[member[x, map[omega, omega]]]] == True

In[5]:= (% /. x → x_) /. Equal → SetDelayed
```

The following corollary is needed for the proof.

```
In[6]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3],
          not[implies[p1, p3]], {p1 → member[x, binhom[NATADD, NATADD]],
           p2 → member[x, map[omega, omega]], p3 -> member[APPLY[x, set[0]], omega]}]]

Out[6]= or[member[APPLY[x, set[0]], omega], not[member[x, binhom[NATADD, NATADD]]]] == True

In[7]:= or[member[APPLY[x_, set[0]], omega], not[member[x_, binhom[NATADD, NATADD]]]] := True
```

The observation made above can be turned into a formal proof as follows:

```
In[8]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3],
          implies[and[p2, p3], p4], implies[p1, p5], implies[p1, p6],
          implies[and[p5, p6], p7], implies[and[p4, p7], p8], not[implies[p1, p8]],
          {p1 → and[member[x, binhom[NATADD, NATADD]], member[y, binhom[NATADD, NATADD]]],
           p2 → equal[x, times[APPLY[x, set[0]]]],
           p3 -> equal[y, times[APPLY[y, set[0]]]], p4 → equal[composite[NATADD,
              intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]],
             times[natadd[APPLY[x, set[0]], APPLY[y, set[0]]]]],
           p5 → member[APPLY[x, set[0]], omega], p6 → member[APPLY[y, set[0]], omega],
           p7 → member[natadd[APPLY[x, set[0]], APPLY[y, set[0]]], omega],
           p8 → member[composite[NATADD, intersection[composite[inverse[FIRST], x],
               composite[inverse[SECOND], y]]], binhom[NATADD, NATADD]]}]]

Out[8]= or[member[composite[NATADD,
            intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]],
           binhom[NATADD, NATADD]], not[member[x, binhom[NATADD, NATADD]]],
          not[member[y, binhom[NATADD, NATADD]]]] == True

In[9]:= or[member[composite[NATADD,
             intersection[composite[inverse[FIRST], x_], composite[inverse[SECOND], y_]]],
            binhom[NATADD, NATADD]], not[member[x_, binhom[NATADD, NATADD]]],
           not[member[y_, binhom[NATADD, NATADD]]]] := True
```

## a lemma

An endomorphism **x** of **INTADD** is a mapping from **Z** to **Z** that satisfies **composite[x, INTADD] = composite[INTADD, cross[x,x]]**. The **sum** of endomorphisms **x** and **y** of **INTADD** is defined to be **composite[INTADD, cross[x,y], DUP]**. In this section it will be shown that the sum of two mappings from **Z** to **Z** is another one. The following temporary abbreviation will be used:

```
In[10]:= intsum[x_, y_] := composite[INTADD,
           intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]
```

Lemma 1.

```
In[11]:= or[member[composite[INTADD,
             intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]], V],
           not[member[x, u]], not[member[y, v]]] // AssertTest

Out[11]= or[member[composite[INTADD,
             intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]], V],
           not[member[x, u]], not[member[y, v]]] == True

In[12]:= (% /. {u → u_, v → v_, x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma 2.

```
In[13]:= SubstTest[implies, and[equal[x, funpart[u]], equal[y, funpart[v]]],
          FUNCTION[composite[INTADD, intersection[composite[inverse[FIRST], x],
            composite[inverse[SECOND], y]]]], {u → x, v → y}]

Out[13]= or[FUNCTION[composite[INTADD,
             intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]],
           not[FUNCTION[x]], not[FUNCTION[y]]] == True

In[14]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Corollary.

```
In[15]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3], implies[and[p2, p3], p4],
           not[implies[p1, p4]], {p1 → and[member[x, map[Z, Z]], member[y, map[Z, Z]]],
            p2 → FUNCTION[x], p3 → FUNCTION[y], p4 → FUNCTION[composite[INTADD,
              intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]]}]]

Out[15]= or[FUNCTION[composite[INTADD,
             intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]],
           not[member[x, map[Z, Z]]], not[member[y, map[Z, Z]]]] == True

In[16]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[17]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p1, p3], implies[and[p2, p3], p4], not[implies[p1, p4]],
           {p1 -> member[x, map[Z, Z]], p2 -> equal[domain[x], Z],
            p3 -> subclass[range[x], Z], p4 -> equal[domain[x], image[inverse[x], Z]]}]]

Out[17]= or[equal[domain[x], image[inverse[x], Z]], not[member[x, map[Z, Z]]]] == True

In[18]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[19]:= Map[not, SubstTest[and, implies[p1, p4],
           implies[p2, p5], implies[and[p1, p4], p6], implies[and[p2, p5], p7],
           implies[and[p3, p4, p5, p6, p7], p8], not[implies[and[p1, p2, p3], p8]],
           {p1 → member[x, map[Z, Z]], p2 → member[y, map[Z, Z]], p3 → equal[z, composite[INTADD,
               intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]],
            p4 → equal[domain[x], Z], p5 → equal[domain[y], Z],
            p6 → equal[domain[x], image[inverse[x], Z]],
            p7 → equal[domain[y], image[inverse[y], Z]], p8 → equal[domain[z], Z]}]] /.
         z -> composite[INTADD, intersection[composite[inverse[FIRST], x],
            composite[inverse[SECOND], y]]]

Out[19]= or[equal[Z, intersection[image[inverse[x], Z], image[inverse[y], Z]]],
           not[member[x, map[Z, Z]]], not[member[y, map[Z, Z]]]] == True

In[20]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem. The set **map[Z, Z]** is closed under the sum operation:

```
In[21]:= Map[not, SubstTest[and, implies[and[p1, p2], p3], implies[and[p1, p2], p4],
           implies[and[p1, p2], p5], implies[and[p1, p2], p6], implies[and[p3, p4, p5, p6], p7],
           not[implies[and[p1, p2], p7]], {p1 → and[member[x, map[Z, Z]], member[y, map[Z, Z]]],
            p2 -> equal[z, composite[INTADD,
               intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]],
            p3 → member[z, V], p4 → FUNCTION[z], p5 → equal[Z, domain[z]],
            p6 → subclass[range[z], Z], p7 → member[z, map[Z, Z]]}]] /. z -> composite[INTADD,
           intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]

Out[21]= or[member[composite[INTADD,
            intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]],
           map[Z, Z]], not[member[x, map[Z, Z]]], not[member[y, map[Z, Z]]]] == True

In[22]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

## sums of endomorphisms of INTADD

The commutative and associative laws for **INTADD** imply the following rewrite rule which forms the basis of the derivation in this section.

```
In[23]:= composite[INTADD, cross[INTADD, INTADD], TWIST]

Out[23]= composite[INTADD, cross[INTADD, INTADD]]
```

Lemma.

```
In[24]:= Map[
           equal[composite[INTADD, intersection[composite[inverse[FIRST], INTADD, cross[x, x]],
               composite[inverse[SECOND], INTADD, cross[y, y]]]], composite[#, DUP]] &,
           Assoc[composite[INTADD, cross[INTADD, INTADD]], composite[TWIST,
             cross[cross[x, y], cross[x, y]]], TWIST]] // Reverse

Out[24]= equal[
           composite[INTADD, cross[composite[INTADD, intersection[composite[inverse[FIRST], x],
               composite[inverse[SECOND], y]]], composite[INTADD,
              intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]]],
           composite[INTADD, intersection[composite[inverse[FIRST], INTADD, cross[x, x]],
             composite[inverse[SECOND], INTADD, cross[y, y]]]]] == True

In[25]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem.

```
In[26]:= Map[not, SubstTest[and, implies[and[p1, p2, p3], p4],
            implies[p4, p5], not[implies[and[p1, p2, p3], p5]],
            {p1 -> equal[composite[x, INTADD], composite[INTADD, cross[x, x]]],
             p2 -> equal[composite[y, INTADD], composite[INTADD, cross[y, y]]],
             p3 → equal[w, composite[INTADD, cross[x, y], DUP, INTADD]],
             p4 → equal[w,
               composite[INTADD, cross[INTADD, INTADD], cross[cross[x, x], cross[y, y]], DUP]],
             p5 -> equal[w, composite[INTADD, cross[INTADD, INTADD],
                cross[cross[x, y], cross[x, y]], cross[DUP, DUP]]]}]] /.
          w -> composite[INTADD, cross[x, y], DUP, INTADD]

Out[26]= or[equal[composite[INTADD,
              cross[composite[INTADD, intersection[composite[inverse[FIRST], x],
                 composite[inverse[SECOND], y]]], composite[INTADD,
                intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]]],
             composite[INTADD, intersection[composite[inverse[FIRST], x],
               composite[inverse[SECOND], y]], INTADD]],
           not[equal[composite[INTADD, cross[x, x]], composite[x, INTADD]]],
           not[equal[composite[INTADD, cross[y, y]], composite[y, INTADD]]]] == True

In[27]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[28]:= SubstTest[implies, and[member[x, map[fix[domain[y]], fix[domain[y]]]],
           equal[composite[x, y], composite[y, cross[x, x]]]],
          member[x, binhom[y, y]], y → INTADD]

Out[28]= or[member[x, binhom[INTADD, INTADD]],
           not[equal[composite[INTADD, cross[x, x]], composite[x, INTADD]]],
           not[member[x, map[Z, Z]]]] == True

In[29]:= or[member[x_, binhom[INTADD, INTADD]],
           not[equal[composite[INTADD, cross[x_, x_]], composite[x_, INTADD]]],
           not[member[x_, map[Z, Z]]]] := True
```

Theorem. The sum of endomorphisms of **INTADD** is an endomorphism.

```
In[30]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3],
           implies[p1, p4], implies[p1, p5], implies[and[p2, p3], p6],
           implies[and[p4, p5], p7], implies[and[p6, p7], p8], not[implies[p1, p8]],
           {p1 → and[member[x, binhom[INTADD, INTADD]], member[y, binhom[INTADD, INTADD]]],
            p2 → member[x, map[Z, Z]], p3 → member[y, map[Z, Z]],
            p4 -> equal[composite[x, INTADD], composite[INTADD, cross[x, x]]],
            p5 -> equal[composite[y, INTADD], composite[INTADD, cross[y, y]]],
            p6 → member[intsum[x, y], map[Z, Z]],
            p7 -> equal[composite[intsum[x, y], INTADD],
              composite[INTADD, cross[intsum[x, y], intsum[x, y]]]],
            p8 → member[intsum[x, y], binhom[INTADD, INTADD]]}]]

Out[30]= or[member[composite[INTADD,
             intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]],
            binhom[INTADD, INTADD]], not[member[x, binhom[INTADD, INTADD]]],
           not[member[y, binhom[INTADD, INTADD]]]] == True

In[31]:= or[member[composite[INTADD,
             intersection[composite[inverse[FIRST], x_], composite[inverse[SECOND], y_]]],
            binhom[INTADD, INTADD]], not[member[x_, binhom[INTADD, INTADD]]],
           not[member[y_, binhom[INTADD, INTADD]]]] := True
```