summary

This is the fifth notebook in a series about binary homomorphisms. In an earlier notebook of this series, it was shown that every endomorphism of $\text{NATADD}$ is a function of the form $\text{times}[x]$ where $x$ is a natural number. A similar result holds for integers, but at this point in the development of integer arithmetic using the GOEDEL program, multiplication of integers has not yet even been defined. Indeed, one way to define integer multiplication would be in terms of endomorphisms of $\text{INTADD}$. In this notebook, some examples and elementary properties of endomorphisms of $\text{INTADD}$ are derived. Among other things, it is shown that the only constant endomorphism of $\text{INTADD}$ is the function $\text{cart}[Z, \text{set}[\text{id[omega]]}]$ which assigns the integer zero to every integer. (Recall that $\text{id[omega]}$ is the integer zero.) This result is analogous to a corresponding theorem about constant endomorphisms of $\text{NATADD}$, which was derived using the fact that endomorphisms of $\text{NATADD}$ are of the form $\text{times}[x]$. For the case of integers, the theorem is instead derived from the basic observation that since all endomorphisms of $\text{INTADD}$ take zero to zero, a constant endomorphism must take every integer to zero.

examples of endomorphisms of $\text{INTADD}$

The identity endomorphism:

\[
\text{In}[2]:= \text{SubstTest[implies, member[x, BINOPS],}
\text{ member[id[fix[domain[x]]], binhom[x, x]], x \rightarrow INTADD]}
\]

\[
\text{Out}[2]= \text{member[id[Z], binhom[INTADD, INTADD]] = True}
\]

\[
\text{In}[3]:= \text{member[id[Z], binhom[INTADD, INTADD]] := True}
\]

Simplification rules:
some properties of endomorphisms of INTADD

Mapping property.

Domains:

```plaintext
In[16]:= SubstTest[implies, member[x, binhom[w, y]],
   equal[domain[x], fix[domain[w]]], w \rightarrow INTADD]
Out[16]= or[equal[Z, domain[x]], not[member[x, binhom[INTADD, y]]]] := True
In[17]:= or[equal[Z, domain[x_]], not[member[x_, binhom[INTADD, y_]]]] := True
```
Corollary.

\[ \text{In}[18] := \text{Map}[	ext{not}, \text{SubstTest}[\text{implies}[p1, p2], \text{implies}[p2, p3], \text{not}[\text{implies}[p1, p3]],
\text{\{p1} \rightarrow \text{member}\{x, \text{binhom}[\text{INTADD}, y]\},
\text{p2} \rightarrow \text{equal}\{Z, \text{domain}\{x\}\}, \text{p3} \rightarrow \text{member}[\text{id}[\text{omega}], \text{domain}\{x\}\}]]
\]

\[ \text{Out}[18] = \text{or}[\text{member}[\text{id}[\text{omega}], \text{domain}\{x\}], \text{not}[\text{member}\{x, \text{binhom}[\text{INTADD}, y]\}]] = \text{True} \]

\[ \text{In}[19] := \text{or}[\text{member}[\text{id}[\text{omega}], \text{domain}\{x\}], \text{not}[\text{member}\{x, \text{binhom}[\text{INTADD}, y]\}]] := \text{True} \]

Ranges:

\[ \text{In}[20] := \text{SubstTest}[\text{implies}, \text{member}\{x, \text{binhom}[y, w]\},
\text{subclass}[\text{range}\{x\}, \text{fix}[\text{domain}\{w\}]], w \rightarrow \text{INTADD}]
\]

\[ \text{Out}[20] = \text{or}[\text{not}[\text{member}\{x, \text{binhom}[y, \text{INTADD}]\}], \text{subclass}[\text{range}\{x\}, Z] = \text{True} \]

\[ \text{In}[21] := \text{or}[\text{not}[\text{member}\{x, \text{binhom}[y, \text{INTADD}]\}], \text{subclass}[\text{range}\{x\}, Z] := \text{True} \]

---

**endomorphisms preserve intadd**

Lemma.

\[ \text{In}[22] := \text{SubstTest}[\text{implies}, \text{equal}\{x, s\},
\text{equal}[\text{APPLY}\{x, t\}, \text{APPLY}\{s, t\}], \{r \rightarrow \text{composite}[\text{funpart}\{x\}, \text{INTADD}],
\text{s} \rightarrow \text{composite}[\text{INTADD}, \text{cross}[\text{funpart}\{x\}, \text{funpart}\{x\}]], t \rightarrow \text{PAIR}[u, v]\}]
\]

\[ \text{Out}[22] = \text{or}[\text{equal}[\text{APPLY}[\text{funpart}\{x\}, \text{intadd}\{u, v\}],
\text{intadd}[\text{APPLY}[\text{funpart}\{x\}, u], \text{APPLY}[\text{funpart}\{x\}, v]],
\text{not}[\text{equal}[\text{composite}[\text{INTADD}, \text{cross}[\text{funpart}\{x\}, \text{funpart}\{x\}]],
\text{composite}[\text{funpart}\{x\}, \text{INTADD}]]] = \text{True} \]

\[ \text{In}[23] := (\% / . \{x \rightarrow x, u \rightarrow u, v \rightarrow v\}) /. \text{Equal} \rightarrow \text{SetDelayed} \]

Removing the \text{funpart} wrapper yields:

\[ \text{In}[24] := \text{SubstTest}[\text{implies}, \text{equal}\{x, \text{funpart}[y]\},
\text{or}[\text{equal}[\text{APPLY}[x, \text{intadd}\{u, v\}], \text{intadd}[\text{APPLY}[x, u], \text{APPLY}[x, v]]],
\text{not}[\text{equal}[\text{composite}[\text{INTADD}, \text{cross}[x, x]], \text{composite}[x, \text{INTADD}]]], y \rightarrow x]
\]

\[ \text{Out}[24] = \text{or}[\text{equal}[\text{APPLY}[x, \text{intadd}\{u, v\}], \text{intadd}[\text{APPLY}[x, u], \text{APPLY}[x, v]]],
\text{not}[\text{equal}[\text{composite}[\text{INTADD}, \text{cross}[x, x]], \text{composite}[x, \text{INTADD}]]],
\text{not}[\text{FUNCTION}[x]] = \text{True} \]

\[ \text{In}[25] := (\% / . \{x \rightarrow x, u \rightarrow u, v \rightarrow v\}) /. \text{Equal} \rightarrow \text{SetDelayed} \]

The \text{FUNCTION} literal is superfluous, and can be omitted:
endomorphisms of INTADD take zero to zero

This can be cleaned up:

The following lemma is needed due to the limitation of the equality substitution rule to atomic expressions.

The following corollary of the APPLY rule follows.
Theorem. The only constant endomorphism of \textbf{INTADD} is the zero endomorphism.

Corollary. Eliminating the variable \texttt{x} yields:

Eliminating the variable \texttt{x} yields:

Corollary.