**IMAGE[x] = IMAGE[y]**

Johan G. F. Belinfante  
2004 February 28

---

**summary**

The constructor \texttt{VERTSECT} is not one-to-one. For example \texttt{VERTSECT[E] = VERTSECT[S] = 0}, but \texttt{E} is obviously not equal to \texttt{S}. If \texttt{VERTSECT[x] = VERTSECT[y]}, then \texttt{thinpert[x] = thinpert[y]}, but a counterexample shows that the converse does not hold. On the other hand, \texttt{VERTSECT[x] and IMAGE[x]} determine one another, and therefore \texttt{IMAGE[x] = IMAGE[y]} is equivalent to \texttt{VERTSECT[x] = VERTSECT[y]}.

---

**a thinpart counterexample**

The **thinpart** of \texttt{x} is defined as follows:

\[
\text{In[2]} := \text{thinpert[x]}
\]

\[
\text{Out[2]} := \text{composite[x, id[domain[VERTSECT[x]]]]}
\]

This is identical:

\[
\text{In[3]} := \text{composite[inverse[E], VERTSECT[x]]}
\]

\[
\text{Out[3]} := \text{composite[x, id[domain[VERTSECT[x]]]]}
\]

Equality of \texttt{VERTSECT}'s implies equality of **thinpart**'s:

\[
\text{In[4]} := \text{SubstTest[implies, equal[u, v], equal[composite[w, u], composite[w, v]],} \\
\{u \rightarrow \text{VERTSECT[x]}, v \rightarrow \text{VERTSECT[y]}, w \rightarrow \text{inverse[E]}\}
\]

\[
\text{Out[4]} := \text{or[equal[composite[x, id[domain[VERTSECT[x]]]]], composite[y, id[domain[VERTSECT[y]]]]],} \\
\text{not[equal[VERTSECT[x], VERTSECT[y]]]] = True}
\]

The converse is not true; the following counterexample shows that the equality of **thinpart**'s of a relation does not imply equality of their **VERTSECT**'s.
The somewhat tedious details of the proof of the assertion made in the preceding section are taken care of in this section.

The converse also holds because one construct `IMAGE[x]` out of `VERTSECT[x]` as follows:

The somewhat tedious details of the proof of the assertion made in the preceding section are taken care of in this section.

The converse also holds because one construct `IMAGE[x]` out of `VERTSECT[x]` as follows:
a new rewrite rule

The two implications derived above can be combined into a new rewrite rule.

\[
\text{In[13]} := \text{equiv[equal[\text{IMAGE}[x], \text{IMAGE}[y]], equal[\text{VERTSECT}[x], \text{VERTSECT}[y]]]}
\]
\[
\text{Out[13]} = \text{True}
\]

The orientation of the rewrite rule is tentatively chosen as follows:

\[
\text{In[14]} := \text{equal[\text{VERTSECT}[x\_], \text{VERTSECT}[y\_]]} := \text{equal[\text{IMAGE}[x], \text{IMAGE}[y]]}
\]

This is done to preserve an existing rewrite rule that deals with this special case:

\[
\text{In[15]} := \text{equal[\text{IMAGE}[\text{id}[x]], \text{IMAGE}[\text{id}[y]]]}
\]
\[
\text{Out[15]} = \text{equal}[x, y]
\]

The equality of \text{thinpert}'s derived in an earlier section is transformed by the new rule to the following:

\[
\text{In[16]} := \text{SubstTest[\text{implies, equal}[u, v], equal[\text{composite}[w, u], \text{composite}[w, v]],}
\]
\[
\{u \rightarrow \text{VERTSECT}[x], v \rightarrow \text{VERTSECT}[y], w \rightarrow \text{inverse}[E]\}]
\]
\[
\text{Out[16]} = \text{or[equal[\text{composite}[x, \text{id[domain[VERTSECT][x][\_]]}],}
\]
\[
\text{composite}[y, \text{id[domain[VERTSECT][y][\_]]}], \text{not[equal[\text{IMAGE}[x], \text{IMAGE}[y]]]]} = \text{True}
\]
\[
\text{In[17]} := \text{or[equal[\text{composite}[x\_, \text{id[domain[VERTSECT][x\_[\_]]]]},}
\]
\[
\text{composite}[y\_, \text{id[domain[VERTSECT][y\_[\_]]]]}], \text{not[equal[\text{IMAGE}[x\_], \text{IMAGE}[y\_[\_]]]]} := \text{True}
\]

The equality substitution rule for \text{VERTSECT} now follows from that for \text{IMAGE}, and will therefore be removed from the \text{GOEDEL} program.