summary

By definition, a binary homomorphism \( t \in \text{binhom}[x, y] \) is a mapping from \( \text{fix}[\text{domain}[x]] \) to \( \text{fix}[\text{domain}[y]] \) which satisfies the equation \( t \circ x = y \circ (t \otimes t) \). In this notebook it is shown that if \( x \) and \( y \) are binary operations, this equation follows from the (seemingly weaker) inclusion \( t \circ x \subseteq y \circ (t \otimes t) \).

derivation

For simplicity, the derivation is first given using \texttt{funpart} and \texttt{binop} wrappers, which are subsequently eliminated.

Lemma. This derivation contains all the essential steps. Two temporary variables \( u \) and \( v \) are introduced, and eliminated from the final result. The key idea is that if \( u \subseteq v \) and \( v \) is a function, and if \( \text{domain}[u] = \text{domain}[v] \), then \( u = v \).
Theorem. Here the funpart wrapper is removed, and the constructor binhom is introduced.

Corollary. If \( x \) and \( y \) are binary operations and if the mapping \( t: \text{fix}[\text{domain}[x]] \to \text{fix}[\text{domain}[y]] \) satisfies the condition \( t \circ x \subseteq y \circ (t \otimes t) \), then \( t \) is a binary homomorphism: \( t \in \text{binhom}[x, y] \). (The corollary follows from the theorem by removing the \text{binop} wrappers in a standard fashion.)
In[6]:= SubstTest[implies, and[equal[x, binop[u]],
equal[y, binop[v]], member[t, map[fix[domain[x]], fix[domain[y]]]],
subclass[composite[t, x], composite[y, cross[t, t]]],
member[t, binhom[x, y]], {u \rightarrow x, v \rightarrow y}] // Reverse

Out[6]= or[member[t, binhom[x, y]], not[member[t, map[fix[domain[x]], fix[domain[y]]]]],
not[member[x, BINOPS]], not[member[y, BINOPS]],
not[subclass[composite[t, x], composite[y, cross[t, t]]]]] = True

In[7]:= or[member[t_, binhom[x_, y_]], not[member[t_, map[fix[domain[x_]], fix[domain[y_]]]]],
not[member[x_, BINOPS]], not[member[y_, BINOPS]],
not[subclass[composite[t_, x_], composite[y_, cross[t_, t_]]]]] := True