when is composite[id[x], S] a complete lattice order ?

Johan G. F. Belinfante
2008 August 19

```
In[1]:= SetDirectory["l:"]; << goedel.08aug18a;<< tools.m

        :Package Title: goedel.08aug18a                2008 August 18 at 11:45 p.m.

        It is now:  2008 Aug 19 at 7:33

        Loading Simplification Rules

        TOOLS.M                                    Revised 2008 July 5

        weightlimit = 40
```

summary

The co-restriction of **S** to **x** is a complete lattice order if and only if **x** is a power set.

derivation

Lemma.

```
In[2]:= SubstTest[fix, composite[inverse[LB[t]], id[domain[t]],
          complement[composite[complement[t], id[domain[t]], LB[t]]]], t -> composite[id[x], S]]

Out[2]= domain[GLB[composite[id[x], S]]] ==
         union[intersection[image[V, intersection[x, set[U[x]]]], set[0]], intersection[
           image[inverse[BIGCAP], fix[composite[S, id[x], S, IMAGE[id[U[x]]]]]], P[x]]]

In[3]:= domain[GLB[composite[id[x_], S]]] :=
         union[intersection[image[V, intersection[x, set[U[x]]]], set[0]], intersection[
           image[inverse[BIGCAP], fix[composite[S, id[x], S, IMAGE[id[U[x]]]]]], P[x]]]
```

Lemma.

```
In[4]:= SubstTest[member, x, intersection[y, z],
         {y → fix[IMAGE[inverse[S]]], z -> fix[composite[inverse[E], IMAGE[inverse[BIGCUP]]]]}]

Out[4]= and[equal[x, image[inverse[S], x]], member[U[x], x]] ==
         and[equal[x, P[U[x]]], member[x, V]]

In[5]:= and[equal[x_, image[inverse[S], x_]], member[U[x_], x_]] :=
         and[equal[x, P[U[x]]], member[x, V]]
```

Lemma.

```
In[6]:= Map[equiv[#, and[equal[x, P[U[x]]], member[x, V]]] &,
         member[composite[id[x], S], CL] // AssertTest]

Out[6]= or[and[equal[x, P[U[x]]], member[x, V]], not[member[composite[id[x], S], CL]]] == True

In[7]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem.

```
In[8]:= equiv[member[composite[id[x], S], CL], and[equal[x, P[U[x]]], member[x, V]]]

Out[8]= True

In[9]:= member[composite[id[x_], S], CL] := and[equal[x, P[U[x]]], member[x, V]]
```

---

variable-free formulation

Theorem. (Variable-free reformulation.)

```
In[10]:= image[inverse[IMAGE[composite[id[S], inverse[SECOND]]]], CL] // Normality

Out[10]= image[inverse[IMAGE[composite[id[S], inverse[SECOND]]]], CL] == range[POWER]

In[11]:= image[inverse[IMAGE[composite[id[S], inverse[SECOND]]]], CL] := range[POWER]
```

Corollary.

```
In[12]:= ImageComp[IMAGE[composite[id[S], inverse[SECOND]]],
           inverse[IMAGE[composite[id[S], inverse[SECOND]]]], CL] // Reverse

Out[12]= image[IMAGE[composite[id[S], inverse[SECOND]]], range[POWER]] ==
          intersection[CL, image[INVERSE, RS[inverse[S]]]]

In[13]:= image[IMAGE[composite[id[S], inverse[SECOND]]], range[POWER]] :=
          intersection[CL, image[INVERSE, RS[inverse[S]]]]
```