LUB[id[cliques[x]] ∘ S]

Johan G. F. Belinfante
2009 December 30

```
In[1]:= SetDirectory["l:"]; << goedel.09dec29a; << tools.m

        :Package Title: goedel.09dec29a              2009 December 29 at 6:35 p.m.

        It is now:  2009 Dec 30 at 12:35

        Loading Simplification Rules

        TOOLS.M                                    Revised 2009 December 17

        weightlimit = 40
```

summary

A formula is derived for the **LUB** function for the restriction of the subset relation **S** to any clique. The formula was suggested by exercise 5.9 on page 38 of the following reference. This exercise deals with the special case of a class of functions.

```
In[2]:= "Karel Hrbacek and Thomas Jech, Introduction to Set Theory, Third
         Edition, Revised and Expanded, Marcel Dekker, Inc., New York, 1999.";
```

Because the **GOEDEL** program can deal with proper classes, it is not necessary to restrict the class of functions to those contained in a cartesian product of two sets, and for convenience this has not been done here.

derivation

Lemma. The function hull of a class.

```
In[3]:= SubstTest[hull, cliques[t], x,
          t → intersection[cartsq[cartsq[V]], complement[cross[Id, Di]]]] // Reverse

Out[3]= hull[FUNS, x] == union[x, complement[image[V, intersection[FUNS, set[x]]]]]

In[4]:= hull[FUNS, x_] := union[x, complement[image[V, intersection[FUNS, set[x]]]]]
```

Theorem.

```
In[5]:= Map[equal[intersection[FUNS, set[U[x]]], #] &,
         Map[complement[complement[#]] &, (lub[composite[id[t], S], x] // Normality) /. t → FUNS]]

Out[5]= equal[intersection[FUNS, set[U[x]]], lub[composite[id[FUNS], S], x]] == True
```

```
In[6]:= lub[composite[id[FUNS], S], x_] := intersection[FUNS, set[U[x]]]
```

Theorem. Simplification rule.

```
In[7]:= equal[composite[BIGCUP, id[image[inverse[BIGCUP], x]]],
          composite[id[x], BIGCUP]] // AssertTest
Out[7]= equal[composite[BIGCUP, id[image[inverse[BIGCUP], x]]], composite[id[x], BIGCUP]] == True

In[8]:= composite[BIGCUP, id[image[inverse[BIGCUP], x_]]] := composite[id[x], BIGCUP]
```

Theorem.

```
In[9]:= SubstTest[reify, x, lub[t, x], t -> composite[id[FUNS], S]]
Out[9]= LUB[composite[id[FUNS], S]] == composite[id[FUNS], BIGCUP]

In[10]:= LUB[composite[id[FUNS], S]] := composite[id[FUNS], BIGCUP]
```

---

cliques case

The same techniques that were used for the class of functions can be used for the more general case of any class of the form **cliques[x]**. This generalization is considered in this section.

Lemma.

```
In[11]:= Map[equal[intersection[cliques[x], set[U[y]]], #] &, Map[complement[complement[#]] &,
           (lub[composite[id[t], S], y] // Normality) /. t → cliques[x]]]
Out[11]= equal[intersection[cliques[x], set[U[y]]], lub[composite[id[cliques[x]], S], y]] == True

In[12]:= lub[composite[id[cliques[x_]], S], y_] := intersection[cliques[x], set[U[y]]]
```

Theorem.

```
In[13]:= SubstTest[reify, y, lub[t, y], t -> composite[id[cliques[x]], S]]
Out[13]= LUB[composite[id[cliques[x]], S]] == composite[id[cliques[x]], BIGCUP]

In[14]:= LUB[composite[id[cliques[x_]], S]] := composite[id[cliques[x]], BIGCUP]
```

Corollary. (The special case considered in exercise 5.9 in the book by Hrbacek and Jech.)

```
In[15]:= SubstTest[LUB, composite[id[cliques[t]], S],
           t -> intersection[cartsq[cart[x, y]], complement[cross[Id, Di]]]] // Reverse
Out[15]= LUB[composite[id[intersection[FUNS, P[cart[x, y]]]], S]] ==
          composite[id[intersection[FUNS, P[cart[x, y]]]], BIGCUP]

In[16]:= LUB[composite[id[intersection[FUNS, P[cart[x_, y_]]]], S]] :=
          composite[id[intersection[FUNS, P[cart[x, y]]]], BIGCUP]
```