sethood rules for GLB[x] and LUB[x]

Johan G. F. Belinfante
2005 September 27

```
In[1]:= SetDirectory["l:"]; << goedel73.27a; << tools.m

        :Package Title: goedel73.27a            2005 September 27 at 12:30 noon

        It is now:  2005 Sep 27 at 13:45

        Loading Simplification Rules

        TOOLS.M                      Revised 2005 September 18

        weightlimit = 40
```

summary

It is shown that if **x** is a set, then both **GLB[x]** and **LUB[x]** also are sets.

upper bounds for the ranges of GLB and LUB

```
In[2]:= Map[subclass[#, domain[x]] &, ImageComp[id[domain[x]],
          intersection[complement[composite[complement[x], id[domain[x]], LB[x]]], LB[x]], V]]

Out[2]= subclass[range[GLB[x]], domain[x]] == True

In[3]:= subclass[range[GLB[x_]], domain[x_]] := True

In[4]:= SubstTest[subclass, range[GLB[y]], domain[y], y → inverse[x]]

Out[4]= subclass[range[LUB[x]], range[x]] == True

In[5]:= subclass[range[LUB[x_]], range[x_]] := True
```

Corollaries:

```
In[6]:= composite[id[domain[x]], GLB[x]] // VSNormality

Out[6]= composite[id[domain[x]], GLB[x]] == GLB[x]

In[7]:= composite[id[domain[x_]], GLB[x_]] := GLB[x]

In[8]:= composite[id[range[x]], LUB[x]] // VSNormality

Out[8]= composite[id[range[x]], LUB[x]] == LUB[x]
```

```
In[9]:= composite[id[range[x_]], LUB[x_]] := LUB[x]
```

upper bounds for the domains of GLB and LUB

Lemma.

```
In[10]:= SubstTest[implies, subclass[u, v], subclass[composite[w, u], composite[w, v]],
          {u -> intersection[complement[composite[complement[inverse[x]], id[range[x]], UB[x]]],
             UB[x]], v → UB[x], w → id[range[x]]}]
Out[10]= subclass[composite[LUB[x], E], x] == True

In[11]:= subclass[composite[LUB[x_], E], x_] := True
```

Corollary 1.

```
In[12]:= subclass[domain[LUB[x]], domain[UB[x]]]
Out[12]= True
```

Corollary 2.

```
In[13]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]],
           subclass[u, w], {u → domain[LUB[x]], v → domain[UB[x]], w → P[domain[x]]}]
Out[13]= subclass[U[domain[LUB[x]]], domain[x]] == True

In[14]:= subclass[U[domain[LUB[x_]]], domain[x_]] := True
```

The corresponding result for **GLB** is less pretty, but is completely analogous.

```
In[15]:= SubstTest[subclass, composite[LUB[y], E], y, y → inverse[x]]
Out[15]= subclass[composite[inverse[E], inverse[GLB[x]]], x] == True

In[16]:= subclass[composite[inverse[E], inverse[GLB[x_]]], x_] := True
```

Corollary 1.

```
In[17]:= subclass[domain[GLB[x]], domain[LB[x]]]
Out[17]= True
```

Corollary 2.

```
In[18]:= SubstTest[subclass, U[domain[LUB[y]]], domain[y], y → inverse[x]]
Out[18]= subclass[U[domain[GLB[x]]], range[x]] == True

In[19]:= subclass[U[domain[GLB[x_]]], range[x_]] := True
```

sethood rules

The sethood lemma is first derived using a **setpart** wrapper.

```
In[20]:= SubstTest[implies, and[subclass[u, v], member[v, V]], member[u, V],
           {u → LUB[y], v -> composite[id[range[y]], UB[y]]}] /. y → setpart[x]

Out[20]= member[LUB[setpart[x]], V] == True

In[21]:= member[LUB[setpart[x_]], V] := True
```

Removing the wrapper yields:

```
In[22]:= Map[implies[member[x, y], #] &,
          SubstTest[implies, equal[z, setpart[x]], member[LUB[z], V], z → x]]

Out[22]= or[member[LUB[x], V], not[member[x, y]]] == True

In[23]:= or[member[LUB[x_], V], not[member[x_, y_]]] := True
```

Similar results hold for **GLB**.

```
In[24]:= SubstTest[implies, and[subclass[u, v], member[v, V]], member[u, V],
           {u → GLB[y], v -> composite[id[domain[y]], LB[y]]}] /. y → setpart[x]

Out[24]= member[GLB[setpart[x]], V] == True

In[25]:= member[GLB[setpart[x_]], V] := True

In[26]:= Map[implies[member[x, y], #] &,
          SubstTest[implies, equal[z, setpart[x]], member[GLB[z], V], z → x]]

Out[26]= or[member[GLB[x], V], not[member[x, y]]] == True

In[27]:= or[member[GLB[x_], V], not[member[x_, y_]]] := True
```