upward and downward directed partial orders

Johan G. F. Belinfante
2006 May 4

```
In[1]:= SetDirectory["l:"]; << goedel81.02a; << tools.m

        :Package Title: goedel81.02a          2006 May 2 at 11:55 a.m.

        It is now:  2006 May 4 at 13:24

        Loading Simplification Rules

        TOOLS.M                       Revised 2006 March 7

        weightlimit = 40
```

summary

A partial order **x** is **directed upward** if any pair of elements in **fix[x]** have an upper bound:

```
In[3]:= assert[forall[u, v,
          implies[and[member[u, fix[x]], member[v, fix[x]]], not[empty[ub[x, set[u, v]]]]]]]

Out[3]= subclass[cart[fix[x], fix[x]], composite[inverse[x], x]]
```

When **x** is a partial order, the reverse inclusion also holds, so one can strengthen this to an equation. Similarly, a partial order **x** is **directed downward** if **inverse[x]** is directed upward. In general, a partial order need not be directed either upward or downward, but both hold for lattices. In this notebook, it is shown that total orders and complete lattices are directed upward and downward.

results for total orders

For total orders, the simplest strategy is to use the **to** wrapper.

Lemma:

```
In[4]:= SubstTest[implies, subclass[v, w], subclass[composite[u, v], composite[u, w]],
         {u → to[x], v → id[fix[to[x]]], w → inverse[to[x]]}]

Out[4]= subclass[to[x], composite[to[x], inverse[to[x]]]] == True

In[5]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[6]:= SubstTest[subclass, inverse[u], inverse[v],
          {u → to[x], v -> composite[to[x], inverse[to[x]]]}]

Out[6]= subclass[inverse[to[x]], composite[to[x], inverse[to[x]]]] == True

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

Lemma.

```
In[8]:= SubstTest[subclass, union[u, v], w,
          {u → to[x], v → inverse[to[x]], w → composite[to[x], inverse[to[x]]]}]

Out[8]= subclass[cart[fix[to[x]], fix[to[x]]], composite[to[x], inverse[to[x]]]] == True

In[9]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem. Total orders are directed downward.

```
In[10]:= SubstTest[and, subclass[u, v], subclass[v, u], {u → composite[to[x], inverse[to[x]]],
            w → cart[fix[to[x]], fix[to[x]]], v → union[to[x], inverse[to[x]]]}]

Out[10]= True == equal[cart[fix[to[x]], fix[to[x]]], composite[to[x], inverse[to[x]]]]

In[11]:= composite[to[x_], inverse[to[x_]]] := cart[fix[to[x]], fix[to[x]]]
```

Corollary. Total orders are directed upward as well.

```
In[12]:= SubstTest[composite, to[y], inverse[to[y]], y → inverse[to[x]]]

Out[12]= composite[inverse[to[x]], to[x]] == cart[fix[to[x]], fix[to[x]]]

In[13]:= composite[inverse[to[x_]], to[x_]] := cart[fix[to[x]], fix[to[x]]]
```

The theorems derived above can be restated, replacing the **to** wrappers with **TOTALORDER** literals, as follows.

```
In[14]:= SubstTest[implies, equal[x, to[y]],
           equal[composite[x, inverse[x]], cart[fix[x], fix[x]]], y → x]

Out[14]= or[equal[cart[fix[x], fix[x]], composite[x, inverse[x]]], not[TOTALORDER[x]]] == True

In[15]:= or[equal[cart[fix[x_], fix[x_]], composite[x_, inverse[x_]]],
            not[TOTALORDER[x_]]] := True

In[16]:= SubstTest[implies, equal[x, to[y]],
           equal[composite[inverse[x], x], cart[fix[x], fix[x]]], y → x]

Out[16]= or[equal[cart[fix[x], fix[x]], composite[inverse[x], x]], not[TOTALORDER[x]]] == True

In[17]:= or[equal[cart[fix[x_], fix[x_]], composite[inverse[x_], x_]],
            not[TOTALORDER[x_]]] := True
```

results for complete lattices

Lemma.

```
In[18]:= image[inverse[PAIRSET], domain[UB[x]]] // RelnNormality
```

```
Out[18]= image[inverse[PAIRSET], domain[UB[x]]] == composite[inverse[x], x]
```

```
In[19]:= image[inverse[PAIRSET], domain[UB[x_]]] := composite[inverse[x], x]
```

Similarly, replacing **x** with **inverse[x]**, one has:

```
In[20]:= SubstTest[image, inverse[PAIRSET], domain[UB[y]], y → inverse[x]]
```

```
Out[20]= image[inverse[PAIRSET], domain[LB[x]]] == composite[x, inverse[x]]
```

```
In[21]:= image[inverse[PAIRSET], domain[LB[x_]]] := composite[x, inverse[x]]
```

Lemma.

```
In[22]:= Map[implies[#, subclass[cart[fix[x], fix[x]], composite[inverse[x], x]]] &,
           SubstTest[subclass, cart[fix[x], fix[x]],
            image[inverse[PAIRSET], y], y -> domain[UB[x]]]] // Reverse
```

```
Out[22]= or[not[subclass[image[PAIRSET, cart[fix[x], fix[x]]], domain[UB[x]]]],
           subclass[cart[fix[x], fix[x]], composite[inverse[x], x]]] == True
```

```
In[23]:= (% /. x → x_) /. Equal → SetDelayed
```

Thoerem. Complete lattices are directed upward.

```
In[24]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3],
           implies[p3, p4], implies[p4, p5], not[implies[p1, p5]], {p1 → member[x, CL],
            p2 -> equal[P[fix[x]], domain[LUB[x]]], p3 → subclass[P[fix[x]], domain[UB[x]]],
            p4 -> subclass[image[PAIRSET, cart[fix[x], fix[x]]], domain[UB[x]]],
            p5 -> subclass[cart[fix[x], fix[x]], composite[inverse[x], x]],
            p6 → subclass[composite[inverse[x], x], cart[fix[x], fix[x]]]}]]
```

```
Out[24]= or[not[member[x, CL]], subclass[cart[fix[x], fix[x]], composite[inverse[x], x]]] == True
```

```
In[25]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[26]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
           {p1 → member[x, CL], p2 → equal[fix[x], domain[x]], p3 → subclass[domain[x], fix[x]]}]]
```

```
Out[26]= or[not[member[x, CL]], subclass[domain[x], fix[x]]] == True
```

```
In[27]:= (% /. x → x_) /. Equal → SetDelayed
```

Corollary. A conditional equation.

```
In[28]:= SubstTest[and, implies[p, subclass[u, v]], implies[p, subclass[v, u]],
           {p → member[x, CL], u -> cart[fix[x], fix[x]], v -> composite[inverse[x], x]}] // Reverse

Out[28]= or[equal[cart[fix[x], fix[x]], composite[inverse[x], x]], not[member[x, CL]]] == True

In[29]:= or[equal[cart[fix[x_], fix[x_]], composite[inverse[x_], x_]],
           not[member[x_, CL]]] := True
```

Similarly, complete lattices are directed downward.

```
In[30]:= Map[implies[member[x, CL], #] &, SubstTest[implies, member[y, CL],
           equal[composite[inverse[y], y], cart[fix[y], fix[y]]], y → inverse[x]]]

Out[30]= or[equal[cart[fix[x], fix[x]], composite[x, inverse[x]]], not[member[x, CL]]] == True

In[31]:= or[equal[cart[fix[x_], fix[x_]], composite[x_, inverse[x_]]],
           not[member[x_, CL]]] := True
```