partitions are partially ordered by refinement

Johan G. F. Belinfante
2010 January 7

```
In[1]:= SetDirectory["l:"]; << goedel.10jan07a; << tools.m

        :Package Title: goedel.10jan07a              2010 January 7 at 1:15 p.m.

        It is now:  2010 Jan 7 at 13:49

        Loading Simplification Rules

        TOOLS.M                                     Revised 2010 January 7

        weightlimit = 40
```

summary

The refinement relation **inverse[S] ∘ IMAGE[inverse[S]]** is a reflexive and transitive relation, but fails to be antisymmetric, and is therefore not a partial ordering. In this notebook it is shown that its restriction to partitions is a partial ordering.

Reference. (See exercise 5.10 on page 38.)

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

The partition wrapper **ptn[x]** is used to derive this result.

strategy

Suppose one partition **ptn[x]** is a refinement of another **ptn[y]** and vice versa. If **u ∈ ptn[x]** then there exists **v ∈ ptn[y]** with **u ⊂ v,** and there exists **w ∈ ptn[x]** with **v ⊂ w**. Then **u ⊂ w**, which means **u = w**, since both **u** and **w** belong to **ptn[x]**. Since **u ⊂ v ⊂ w**, then **u = v**. Thus every element of the one is equal to some element of the other and vice versa. ∴ **ptn[x] = ptn[y]**.

The argument involves a total of five variables. All five variables will eventually be eliminated to produce a variable-free statement.

derivation

A more general result will be derived first, dropping the hypothesis that **y** is a partition. Comment. The derivation goes faster if two proof steps are omitted:

implies[and[p2, p4, p7], p8] ***and*** **implies[and[p3, p8], p9]**

Lemma.

```
In[3]:= Map[not, SubstTest[and, implies[and[p2, p4], p6], implies[and[p1, p5, p6], p7],
          not[implies[and[p1, p2, p3, p4, p5], p9]], {p1 → member[u, ptn[x]],
           p2 → subclass[u, v], p3 → member[v, y], p4 → subclass[v, w], p5 → member[w, ptn[x]],
           p6 → subclass[u, w], p7 → equal[u, w], p8 → equal[u, v], p9 → member[u, y]}]] // Reverse

Out[3]= or[member[u, y], not[member[u, ptn[x]]], not[member[v, y]],
         not[member[w, ptn[x]]], not[subclass[u, v]], not[subclass[v, w]]] == True

In[4]:= (% /. {u → u_, v → v_, w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

The three variables **u**, **v** and **w** are eliminated from the statement in the preceding lemma.

Lemma.

```
In[5]:= Map[empty[domain[domain[composite[complement[#], id[cart[V, V]]]]]] &,
         SubstTest[class, pair[pair[u, v], w],
          or[member[u, y], not[member[u, t]], not[member[v, y]], not[member[w, t]],
           not[subclass[u, v]], not[subclass[v, w]]], t → ptn[x]]]

Out[5]= subclass[intersection[
          image[inverse[S], intersection[y, image[inverse[S], ptn[x]]]], ptn[x]], y] == True

In[6]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma. The statement in the preceding lemma is modified slightly to introduce the hypothesis that **ptn[x]** and **y** are refinements of each other.

```
In[7]:= SubstTest[implies, equal[t, image[inverse[S], ptn[x]]],
         subclass[intersection[image[inverse[S], intersection[y, t]], ptn[x]], y],
         t → image[inverse[S], y]] // Reverse

Out[7]= or[not[equal[image[inverse[S], y], image[inverse[S], ptn[x]]]],
         subclass[intersection[image[inverse[S], y], ptn[x]], y]] == True

In[8]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The following theorem just cleans up the conclusion in the preceding lemma.

Theorem. If a partition **x** is a refinement of a class **y**, and vice versa, then **x** ⊂ **y**.

```
In[9]:= SubstTest[implies, equal[t, image[inverse[S], y]],
          or[not[equal[t, image[inverse[S], ptn[x]]]], subclass[intersection[t, ptn[x]], y]],
          t -> image[inverse[S], ptn[x]]] // Reverse

Out[9]= or[not[equal[image[inverse[S], y], image[inverse[S], ptn[x]]]], subclass[ptn[x], y]] ==
         True

In[10]:= or[not[equal[image[inverse[S], y_], image[inverse[S], ptn[x_]]]],
           subclass[ptn[x_], y_]] := True
```

Corollary. If a partition **x** is a refinement of a partition **y**, and conversely, then **x = y**.

```
In[12]:= SubstTest[and, implies[p, subclass[ptn[x], ptn[y]]],
           implies[p, subclass[ptn[y], ptn[x]]],
           p → equal[image[inverse[S], ptn[x]], image[inverse[S], ptn[y]]]] // MapNotNot

Out[12]= or[equal[ptn[x], ptn[y]],
           not[equal[image[inverse[S], ptn[x]], image[inverse[S], ptn[y]]]]] == True

In[13]:= or[equal[ptn[x_], ptn[y_]],
           not[equal[image[inverse[S], ptn[x_]], image[inverse[S], ptn[y_]]]]] := True
```

Lemma. (The **ptn** wrappers are eliminated from the corollary to prepare for eliminating the variables **x** and **y**.)

```
In[14]:= SubstTest[implies, and[equal[x, ptn[u]], equal[y, ptn[v]]], or[equal[x, y],
            not[equal[image[inverse[S], x], image[inverse[S], y]]]], {u → x, v → y}] // Reverse

Out[14]= or[equal[x, y], member[0, x], member[0, y],
           not[equal[image[inverse[S], x], image[inverse[S], y]]],
           not[subclass[cart[x, x], union[DISJOINT, Id]]],
           not[subclass[cart[y, y], union[DISJOINT, Id]]]] == True

In[15]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The variables x and y are now eliminated, producing a variable-free statement.

Theorem. The restriction of the refinement relation to partitions is antisymmetric.

```
In[16]:= Map[empty[domain[complement[#]]] &,
          SubstTest[class, pair[x, y], implies[and[member[x, t], member[y, t],
             equal[image[inverse[S], x], image[inverse[S], y]]], equal[x, y]], t → PARTNS]]

Out[16]= FUNCTION[composite[id[PARTNS], inverse[IMAGE[inverse[S]]]]] == True

In[17]:= FUNCTION[composite[id[PARTNS], inverse[IMAGE[inverse[S]]]]] := True
```

Lemma. Any restriction of the refinement relation is reflexive.

```
In[18]:= SubstTest[REFLEXIVE, restrict[rfx[t], x, x],
           t -> composite[inverse[S], IMAGE[inverse[S]]]] // Reverse

Out[18]= REFLEXIVE[composite[id[x], inverse[S], IMAGE[inverse[S]], id[x]]] == True

In[19]:= REFLEXIVE[composite[id[x_], inverse[S], IMAGE[inverse[S]], id[x_]]] := True
```

Lemma. Any restriction of the refinement relation is transitive.

```
In[20]:= SubstTest[TRANSITIVE, restrict[trv[t], x, x],
           t -> composite[inverse[S], IMAGE[inverse[S]]]] // Reverse

Out[20]= TRANSITIVE[composite[id[x], inverse[S], IMAGE[inverse[S]], id[x]]] == True

In[21]:= TRANSITIVE[composite[id[x_], inverse[S], IMAGE[inverse[S]], id[x_]]] := True
```

Theorem. The restriction of the refinement relation to the class of partitions is a partial order relation.

```
In[22]:= SubstTest[and, QUASIORDER[t], ANTISYMMETRIC[t],
          t → restrict[composite[inverse[S], IMAGE[inverse[S]]], PARTNS, PARTNS]]

Out[22]= PARTIALORDER[composite[id[PARTNS], inverse[S], IMAGE[inverse[S]], id[PARTNS]]] == True

In[23]:= PARTIALORDER[composite[id[PARTNS], inverse[S], IMAGE[inverse[S]], id[PARTNS]]] := True
```