funpart[po[x]]

Johan G. F. Belinfante
2009 April 28

```
In[1]:= SetDirectory["l:"]; << goedel.09apr26a; << tools.m

        :Package Title: goedel.09apr26a            2009 April 26 at 11:15 p.m.

        It is now:  2009 Apr 28 at 13:17

        Loading Simplification Rules

        TOOLS.M                                    Revised 2009 April 6

        weightlimit = 40
```

summary

The function **funpart[po[x]]** is the class of all **pair[u, v] ∈ po[x]** for which **pair[u, w] ∈ po[x]** implies **v = w**. Since partial orders are reflexive, this implies that **u = v**, and hence **funpart[po[x]]** is an identity function, a fact that is available in the **GOEDEL** program.

```
In[2]:= subclass[funpart[po[x]], Id]

Out[2]= True
```

new rules

Theorem. If **pair[u,v] ∈ funpart[po[x]]** then **u = v**.

```
In[3]:= SubstTest[implies, and[member[t, y], subclass[y, z]], member[t, z],
           {t → pair[u, v], y → funpart[po[x]], z → Id}] // Reverse // MapNotNot

Out[3]= or[equal[u, v], not[member[pair[u, v], funpart[po[x]]]]] == True

In[4]:= or[equal[u_, v_], not[member[pair[u_, v_], funpart[po[x_]]]]] := True
```

An element **u** is a maximal element of **fix[po[x]]** if the vertical section of **po[x]** at **u** is **set[u]**.

```
In[5]:= member[u, maximal[po[x], fix[po[x]]]]

Out[5]= and[equal[image[po[x], set[u]], set[u]], member[u, V]]
```

Theorem. The function **funpart[po[x]]** is the identity function on the class of maximal elements of **fix[po[x]]**.

```
In[6]:= SubstTest[implies, member[pair[u, v], composite[Id, t]],
           member[u, domain[t]], t → funpart[po[x]]] // MapNotNot // Reverse

Out[6]= or[equal[image[po[x], set[u]], set[u]], not[member[pair[u, v], funpart[po[x]]]]] == True

In[7]:= or[equal[image[po[x_], set[u_]], set[u_]],
          not[member[pair[u_, v_], funpart[po[x_]]]]] := True
```

Corollary.

```
In[8]:= SubstTest[implies, equal[x, po[t]], subclass[funpart[x], Id], t → x] // Reverse

Out[8]= or[not[PARTIALORDER[x]], subclass[funpart[x], Id]] == True

In[9]:= or[not[PARTIALORDER[x_]], subclass[funpart[x_], Id]] := True
```

Comment. A variable-free statement is already available in the **GOEDEL** program:

```
In[10]:= image[FUNPART, PO]

Out[10]= P[Id]
```

Theorem. A rewrite rule corresponding to the fact that the funpart of a partial order is an identity function.

```
In[11]:= equal[id[fix[funpart[po[x]]]], funpart[po[x]]]

Out[11]= True

In[12]:= id[fix[funpart[po[x_]]]] := funpart[po[x]]
```

A variable-free restatement of this rewrite rule will now be derived.

Lemma. A simplification rule.

```
In[13]:= equal[intersection[PO, complement[image[inverse[IMAGE[id[cart[V, V]]]], PO]]], 0]

Out[13]= True

In[14]:= intersection[PO, complement[image[inverse[IMAGE[id[cart[V, V]]]], PO]]] := 0
```

Theorem. (Variable-free restatement.)

```
In[15]:= Map[composite[VERTSECT[#], id[PO]] &,
          SubstTest[reify, x, id[fix[funpart[f[x]]]], f → po]]

Out[15]= composite[IMAGE[id[Id]], FUNPART, id[PO]] == composite[FUNPART, id[PO]]

In[16]:= composite[IMAGE[id[Id]], FUNPART, id[PO]] := composite[FUNPART, id[PO]]
```

The rewrite rule for **id[fix[funpart[po[x]]]]** causes some familiar facts to be rewritten.

Theorem. Image rule.

```
In[18]:= SubstTest[image, id[t], y, t → fix[funpart[po[x]]]] // Reverse

Out[18]= image[funpart[po[x]], y] == intersection[y, fix[funpart[po[x]]]]

In[19]:= image[funpart[po[x_]], y_] := intersection[y, fix[funpart[po[x]]]]
```

Theorem. Rule for inverse.

```
In[20]:= SubstTest[inverse, id[t], t → fix[funpart[po[x]]]] // Reverse

Out[20]= inverse[funpart[po[x]]] == funpart[po[x]]

In[21]:= inverse[funpart[po[x_]]] := funpart[po[x]]
```

Theorem. Restriction of **po[x]** to the class of maximal elements.

```
In[22]:= SubstTest[composite, t, id[fix[funpart[t]]], t → po[x]] // Reverse

Out[22]= composite[po[x], funpart[po[x]]] == funpart[po[x]]

In[23]:= composite[po[x_], funpart[po[x_]]] := funpart[po[x]]
```

Theorem. Idempotence.

```
In[24]:= SubstTest[composite, funpart[t], id[fix[funpart[t]]], t → po[x]] // Reverse

Out[24]= composite[funpart[po[x]], funpart[po[x]]] == funpart[po[x]]

In[25]:= composite[funpart[po[x_]], funpart[po[x_]]] := funpart[po[x]]
```

Corollary.

```
In[26]:= SubstTest[TRANSITIVE, id[t], t → fix[funpart[po[x]]]] // Reverse

Out[26]= TRANSITIVE[funpart[po[x]]] == True

In[27]:= TRANSITIVE[funpart[po[x_]]] := True
```

Theorem. Reflexivity.

```
In[28]:= SubstTest[REFLEXIVE, id[t], t → fix[funpart[po[x]]]] // Reverse

Out[28]= REFLEXIVE[funpart[po[x]]] == True

In[29]:= REFLEXIVE[funpart[po[x_]]] := True
```

Theorem.

```
In[30]:= SubstTest[PARTIALORDER, id[t], t → fix[funpart[po[x]]]] // Reverse

Out[30]= PARTIALORDER[funpart[po[x]]] == True

In[31]:= PARTIALORDER[funpart[po[x_]]] := True
```

examples

Since every partial order is isomorphic to a restriction of the subset relation **S**, the natural place to look for examples to illustrate and to find counterexamples for conjectures about **funpart[po[x]]** is for restrictions of **S** to low-rank sets, using **ens** to generate candidates. Here only the simplest case is considered, the restriction of **S** to a power class.

Lemma.

```
In[32]:= Map[domain, SubstTest[funpart, restrict[S, t, t], t → P[x]]] // Reverse

Out[32]= domain[funpart[composite[id[P[x]], S]]] == set[x]

In[33]:= domain[funpart[composite[id[P[x_]], S]]] := set[x]
```

Theorem. A better rewrite rule.

```
In[34]:= SubstTest[id, fix[funpart[po[t]]], t → restrict[S, P[x], P[x]]]

Out[34]= funpart[composite[id[P[x]], S]] == cart[set[x], set[x]]

In[35]:= funpart[composite[id[P[x_]], S]] := cart[set[x], set[x]]
```

Corollary. The restriction of **S** to the natural number **2 = succ[set[0]]**.

```
In[36]:= SubstTest[funpart, restrict[S, P[x], P[x]], x → set[0]] // Reverse

Out[36]= funpart[composite[id[succ[set[0]]], S]] == cart[set[set[0]], set[set[0]]]

In[37]:= funpart[composite[id[succ[set[0]]], S]] := cart[set[set[0]], set[set[0]]]
```

serendipity

Theorem.

```
In[40]:= SubstTest[image, inverse[PS], P[x], x → set[0]] // Reverse

Out[40]= image[inverse[PS], succ[set[0]]] == set[0]

In[41]:= image[inverse[PS], succ[set[0]]] := set[0]
```

Zorn’s lemma

The function **funpart[po[x]]** is the identity on the set of maximal elements of the class **fix[po[x]]**. This class enters into the statement of Zorn’s lemma:

```
In[38]:= SubstTest[implies, and[axch, not[empty[t]], member[t, y], PARTIALORDER[t],
            subclass[chains[t], domain[UB[t]]]], not[empty[funpart[t]]], t → po[x]] // Reverse

Out[38]= or[equal[0, po[x]], not[axch], not[equal[0, funpart[po[x]]]],
           not[member[po[x], y]], not[subclass[chains[po[x]], domain[UB[po[x]]]]]] == True

In[39]:= or[equal[0, po[x_]], not[axch], not[equal[0, funpart[po[x_]]]],
           not[member[po[x_], y_]], not[subclass[chains[po[x_]], domain[UB[po[x_]]]]]] := True
```