

cartesian squares

Johan G. F. Belinfante
2003 May 3

```
In[1]:= << goedel52.r58; << tools.m

:Package Title: goedel52.r58      2003 April 30 at 10:15 p.m.

It is now: 2003 May 6 at 9:10

Loading Simplification Rules

TOOLS.M                          Revised 2003 April 1

weightlimit = 40
```

■ summary

There are many equivalent descriptions of the class of cartesian squares. For example:

```
In[2]:= class[x, exists[y, equal[x, cart[y, y]]]]
Out[2]= image[CART, Id]

In[3]:= class[x, equal[x, cart[fix[x], fix[x]]]]
Out[3]= image[CART, Id]
```

In this notebook, two further characterizations are derived in which **fix** is replaced with **domain** or **range**, respectively.

■ two fix rules involving IMAGE[FIRST]

Without any additional simplification rules, the description involving **domain** looks complicated:

```
In[4]:= class[x, equal[x, cart[domain[x], domain[x]]]]
Out[4]= intersection[fix[composite[S, CART, DUP, IMAGE[FIRST]]],
  image[inverse[DORA], inverse[S]], P[cart[V, V]]]
```

The first order of business is to clean this up using **Renormality**.

```
In[5]:= fix[composite[inverse[S], CART, DUP, IMAGE[FIRST]]] // Renormality
Out[5]= fix[composite[inverse[S], CART, DUP, IMAGE[FIRST]]] ==
  intersection[image[inverse[DORA], inverse[S]], P[cart[V, V]]]

In[6]:= fix[composite[inverse[S], CART, DUP, IMAGE[FIRST]]] :=
  intersection[image[inverse[DORA], inverse[S]], P[cart[V, V]]]
```

```
In[7]:= fix[composite[CART, DUP, IMAGE[FIRST]]] // Renormality // Reverse
```

```
Out[7]= intersection[fix[composite[S, CART, DUP, IMAGE[FIRST]]],
  image[inverse[DORA], inverse[S]], P[cart[V, V]]] ==
  fix[composite[CART, DUP, IMAGE[FIRST]]]
```

```
In[8]:= intersection[fix[composite[S, CART, DUP, IMAGE[FIRST]]],
  image[inverse[DORA], inverse[S]], P[cart[V, V]]] :=
  fix[composite[CART, DUP, IMAGE[FIRST]]]
```

These new rewrite rules help:

```
In[9]:= class[x, equal[x, cart[domain[x], domain[x]]]]
```

```
Out[9]= fix[composite[CART, DUP, IMAGE[FIRST]]]
```

■ connecting domain and fix

The connection between the **fix** formula and the **domain** formula is studied in this section.

```
In[10]:= SubstTest[implies, equal[x, y], equal[domain[x], domain[y]],
  y -> cart[fix[x], fix[x]]]
```

```
Out[10]= or[equal[domain[x], fix[x]], not[equal[x, cart[fix[x], fix[x]]]]] == True
```

```
In[11]:= or[equal[domain[x_], fix[x_]], not[equal[x_, cart[fix[x_], fix[x_]]]]] := True
```

```
In[12]:= SubstTest[implies, and[equal[x, y], equal[y, z]], equal[x, z],
  {y -> cart[fix[x], fix[x]], z -> cart[domain[x], domain[x]]}]
```

```
Out[12]= or[equal[x, cart[domain[x], domain[x]]],
  not[equal[x, cart[fix[x], fix[x]]]], not[equal[domain[x], fix[x]]]] == True
```

```
In[13]:= or[equal[x_, cart[domain[x_], domain[x_]]],
  not[equal[x_, cart[fix[x_], fix[x_]]]], not[equal[domain[x_], fix[x_]]]] := True
```

Some elementary reasoning is needed to combine these results:

```
In[14]:= Map[not,
  SubstTest[and, implies[p1, p2], implies[and[p1, p2], p3], not[implies[p1, p3]],
    {p1 -> equal[x, cart[fix[x], fix[x]]],
     p2 -> equal[domain[x], fix[x]], p3 -> equal[x, cart[domain[x], domain[x]]}]]]
```

```
Out[14]= or[equal[x, cart[domain[x], domain[x]]], not[equal[x, cart[fix[x], fix[x]]]]] == True
```

The variable **x** can be eliminated as follows. This takes a while:

```
In[15]:= Map[equal[V, class[x, #]] &, %]
```

```
Out[15]= subclass[image[CART, Id], fix[composite[CART, DUP, IMAGE[FIRST]]]] == True
```

```
In[16]:= subclass[image[CART, Id], fix[composite[CART, DUP, IMAGE[FIRST]]]] := True
```

■ inclusion in the other direction

```
In[17]:= Map[equal[V, class[x, #]] &, SubstTest[implies, equal[u, v], equal[range[u], range[v]],
  {u -> cart[domain[x], domain[x]], v -> x}]]
```

```
Out[17]= subclass[image[DORA, fix[composite[CART, DUP, IMAGE[FIRST]]]], Id] == True
```

```
In[18]:= subclass[image[DORA, fix[composite[CART, DUP, IMAGE[FIRST]]]], Id] := True
```

This implies:

```
In[19]:= subclass[fix[composite[CART, DUP, IMAGE[FIRST]]], image[inverse[DORA], Id]]
```

```
Out[19]= True
```

Note that the following inclusion also holds:

```
In[20]:= subclass[fix[composite[CART, DUP, IMAGE[FIRST]]], range[CART]]
```

```
Out[20]= True
```

The intersection of these two classes is:

```
In[21]:= intersection[range[CART], image[inverse[DORA], Id]]
```

```
Out[21]= image[CART, Id]
```

Therefore we obtain the inclusion:

```
In[22]:= SubstTest[subclass, u, intersection[v, w],
  {u -> fix[composite[CART, DUP, IMAGE[FIRST]]],
   v -> image[inverse[DORA], Id], w -> range[CART]}}
```

```
Out[22]= subclass[fix[composite[CART, DUP, IMAGE[FIRST]]], image[CART, Id]] == True
```

```
In[23]:= subclass[fix[composite[CART, DUP, IMAGE[FIRST]]], image[CART, Id]] := True
```

■ combining the two inclusions into an equation

```
In[24]:= SubstTest[and, subclass[u, v], subclass[v, u],
  {u -> image[CART, Id],
   v -> fix[composite[CART, DUP, IMAGE[FIRST]]]} // Reverse
```

```
Out[24]= equal[fix[composite[CART, DUP, IMAGE[FIRST]]], image[CART, Id]] == True
```

```
In[25]:= fix[composite[CART, DUP, IMAGE[FIRST]]] := image[CART, Id]
```

At this point, the **domain** formula for the class of cartesian squares simplifies as desired:

```
In[26]:= class[x, equal[x, cart[domain[x], domain[x]]]]
```

```
Out[26]= image[CART, Id]
```

■ analogous rules involving IMAGE[SECOND]

To obtain an analogous formula involving **range** instead of **domain**, we start as before:

```
In[27]:= fix[composite[inverse[S], CART, DUP, IMAGE[SECOND]]] // Renormality
```

```
Out[27]= fix[composite[inverse[S], CART, DUP, IMAGE[SECOND]]] ==  
intersection[image[inverse[DORA], S], P[cart[V, V]]]
```

```
In[28]:= fix[composite[inverse[S], CART, DUP, IMAGE[SECOND]]] :=  
intersection[image[inverse[DORA], S], P[cart[V, V]]]
```

The use of **Renormality** for the second rule did not work, but this alternate method does:

```
In[29]:= SubstTest[intersection, fix[composite[S, funpart[x]]],  
fix[composite[inverse[S], funpart[x]]],  
x -> composite[CART, DUP, IMAGE[SECOND]]]
```

```
Out[29]= intersection[fix[composite[S, CART, DUP, IMAGE[SECOND]]],  
image[inverse[DORA], S], P[cart[V, V]]] == fix[composite[CART, DUP, IMAGE[SECOND]]]
```

```
In[30]:= intersection[fix[composite[S, CART, DUP, IMAGE[SECOND]]],  
image[inverse[DORA], S], P[cart[V, V]]] := fix[composite[CART, DUP, IMAGE[SECOND]]]
```

At this juncture, the result obtained is:

```
In[31]:= class[x, equal[x, cart[range[x], range[x]]]]
```

```
Out[31]= fix[composite[CART, DUP, IMAGE[SECOND]]]
```

Instead of repeating all of the previous steps, the idea is to transform this result directly to the desired final form by applying the function **INVERSE**.

■ rules involving CART

The following two general results about the function **CART** will be needed.

```
In[32]:= SubstTest[fix, composite[id[y], CART, x], y -> P[cart[V, V]]] // Reverse
```

```
Out[32]= intersection[fix[composite[CART, x]], P[cart[V, V]]] == fix[composite[CART, x]]
```

```
In[33]:= intersection[fix[composite[CART, x_]], P[cart[V, V]]] := fix[composite[CART, x]]
```

```
In[34]:= ImageComp[INVERSE, CART, x] // Reverse
```

```
Out[34]= image[INVERSE, image[CART, x]] == image[CART, inverse[x]]
```

```
In[35]:= image[INVERSE, image[CART, x_]] := image[CART, inverse[x]]
```

■ range result

The characterization of the class of cartesian squares involving **range** now follows immediately:

```
In[36]:= SubstTest[fix, composite[INVERSE, x, INVERSE], x -> composite[CART, DUP, IMAGE[FIRST]]]
```

```
Out[36]= fix[composite[CART, DUP, IMAGE[SECOND]]] == image[CART, Id]
```

```
In[37]:= fix[composite[CART, DUP, IMAGE[SECOND]]] := image[CART, Id]
```

The **range** formula for the class of cartesian squares now simplifies to the same expression as the **domain** formula does:

```
In[38]:= class[x, equal[x, cart[range[x], range[x]]]]
```

```
Out[38]= image[CART, Id]
```