an exercise involving numerical dominance

Johan G. F. Belinfante and Tiffany D. Goble
2004 April 7

```
In[1]:= << goedel56.07a; << tools.m;

        :Package Title: goedel56.07a          2004 April 7 at 6:15 p.m.

        It is now:  2006 Jan 20 at 16:35

        Loading Simplification Rules

        TOOLS.M                      Revised 2004 April 18

        weightlimit = 40
```

summary

This notebook contains a solution to exercise 4 on page 69 of Jean Rubin's book, ***Set Theory for the Mathematician***. The problem is to show that for any (small) function, the power set of its range is numerically dominated by the power set of its domain. This theorem can be formulated as a variable−free assertion as follows:

```
In[2]:= assert[forall[x, implies[FUNCTION[x], member[pair[P[range[x]], P[domain[x]]], y]]]] /.
          y -> composite[Q, S]

Out[2]= subclass[image[DORA, FUNS], composite[inverse[POWER], Q, inverse[S], POWER]]
```

The basic strategy of the derivation is to exploit the fact that the inverse image of a subset of the range is a subset of the domain. For functions, the process of forming inverse images is a one−to−one correspondence from the power set of the range to a subset of the power set of the domain. To avoid some reasoning, one can take advantage of the observation that any small function can be written as **funpart[setpart[x]]**. Using **setpart** and **funpart** wrappers allows conditional rewrite rules to be applied automatically. The one−to−one correspondence between the power set of the range of a function and a subset of the domain of that function can be written as a restriction of an **IMAGE** function:

```
In[3]:= class[pair[u, v], and[subclass[u, range[z]], equal[v, image[inverse[z], u]]]] /.
          z -> funpart[x]

Out[3]= composite[IMAGE[inverse[funpart[x]]], id[P[range[funpart[x]]]]]
```

The **GOEDEL** program already contains a rule that this correspondence is one−to−one:

```
In[4]:= ONEONE[composite[IMAGE[inverse[funpart[x]]], id[P[range[funpart[x]]]]]]

Out[4]= True
```

As a consequence, all that is left to do is to rewrite this fact in terms of the equipotence relation **Q**.

some needed sethood facts

Since the equipotence relation only applies to sets, one needs to know that the one−to−one correspondence is a set. Consequently, some technical facts about sethood are needed:

```
In[5]:= member[composite[IMAGE[inverse[funpart[setpart[x]]]],
          id[P[range[funpart[setpart[x]]]]]], BIJ]

Out[5]= and[member[intersection[
           domain[VERTSECT[inverse[funpart[setpart[x]]]], range[funpart[setpart[x]]], V],
         member[range[IMAGE[inverse[funpart[setpart[x]]]], V]]
```

The thinness fact that is needed can be derived as follows:

```
In[6]:= domain[VERTSECT[inverse[funpart[setpart[x]]]]] // Normality

Out[6]= domain[VERTSECT[inverse[funpart[setpart[x]]]] == V

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

Since a subclass of a set is a set, one has:

```
In[8]:= SubstTest[implies, and[subclass[u, v], member[v, V]],
         member[u, V], {u → funpart[setpart[x]], v → setpart[x]}]

Out[8]= member[funpart[setpart[x]], V] == True

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

The final result that is needed is the observation that the range of **IMAGE[x]** is contained in the power class of the range of **x**.

```
In[10]:= subclass[range[IMAGE[x]], P[range[x]]]

Out[10]= True
```

Applying this result to the case at hand, one obtains:

```
In[11]:= SubstTest[implies, and[subclass[u, v], member[v, V]],
          member[u, V], {u -> range[IMAGE[inverse[funpart[setpart[x]]]]],
           v → P[range[inverse[funpart[setpart[x]]]]]}]

Out[11]= member[range[IMAGE[inverse[funpart[setpart[x]]]], V] == True

In[12]:= (% /. x → x_) /. Equal → SetDelayed
```

This verifies that the one−to−one correspondence is a set:

```
In[13]:= member[composite[IMAGE[inverse[funpart[setpart[x]]]],
           id[P[range[funpart[setpart[x]]]]]], BIJ]

Out[13]= True
```

the connection with Q

The connection with the equipotence relation **Q** is obtained from the following fact which follows directly from the (wrapped) membership rule which defines **Q**.

```
In[14]:= implies[member[x, BIJ], member[pair[domain[x], range[x]], Q]]

Out[14]= True
```

Specializing this to the case at hand yields:

```
In[15]:= SubstTest[implies, member[z, BIJ], member[pair[domain[z], range[z]], Q], z ->
           composite[IMAGE[inverse[funpart[setpart[x]]]], id[P[range[funpart[setpart[x]]]]]]]

Out[15]= member[pair[P[range[funpart[setpart[x]]]],
            range[IMAGE[inverse[funpart[setpart[x]]]]], Q] == True

In[16]:= (% /. x → x_) /. Equal → SetDelayed
```

To complete the derivation, all one needs to use are the following observations:

```
In[17]:= subclass[range[IMAGE[inverse[funpart[setpart[x]]]], P[domain[funpart[setpart[x]]]]]

Out[17]= True

In[18]:= member[P[domain[funpart[setpart[x]]]], V]

Out[18]= True
```

lemmas

The numerical dominance relation is **composite[Q,S]**. Since the relations **Q** and **S** commute, there are rewrite rules that cause the inverse of this relation to be transformed as follows:

```
In[19]:= inverse[composite[Q, S]]

Out[19]= composite[Q, inverse[S]]
```

The following lemma is useful to rewrite the facts derived above in terms of the inverse of the numerical dominance relation:

```
In[20]:= SubstTest[implies,
           and[member[pair[u, v], composite[Id, y]], member[pair[v, w], composite[Id, x]]],
           member[pair[u, w], composite[x, y]], {x → inverse[z], y → inverse[S]}] /.
          {u -> P[domain[funpart[setpart[x]]]], v -> range[IMAGE[inverse[funpart[setpart[x]]]],
           w -> P[range[funpart[setpart[x]]]], z -> Q}

Out[20]= member[pair[P[domain[funpart[setpart[x]]]], P[range[funpart[setpart[x]]]],
           composite[Q, inverse[S]]] == True

In[21]:= (% /. x -> x_) /. Equal → SetDelayed
```

eliminating the variable x

The remaining task is to eliminate the variable **x** that occurs in the statement derived in the preceding section. To do so, it is convenient to sequester as much as possible in that statement. The two power class constructors that appear in the ordered pair can be removed by using the following lemma:

```
In[22]:= member[pair[x, y], composite[inverse[POWER], z]] // AssertTest

Out[22]= member[pair[x, y], composite[inverse[POWER], z]] ==
          and[member[x, V], member[y, V], member[pair[x, P[y]], z]]

In[23]:= member[pair[x_, y_], composite[inverse[POWER], z_]] :=
          and[member[x, V], member[y, V], member[pair[x, P[y]], z]]
```

The following companion fact is already known:

```
In[24]:= member[pair[x, y], composite[z, POWER]]

Out[24]= and[member[x, V], member[y, V], member[pair[P[x], y], z]]
```

The domain and range in the ordered pair are removed using the following rule:

```
In[25]:= member[x, image[inverse[DORA], z]]

Out[25]= and[member[x, V], member[pair[domain[x], range[x]], z]]
```

This allows one to rewrite the assertion derived in the preceding section as follows:

```
In[26]:= member[funpart[setpart[x]],
          image[inverse[DORA], composite[inverse[POWER], Q, inverse[S], POWER]]]

Out[26]= True
```

The variable **x** can now be eliminated as follows:

```
In[27]:= Map[subclass[V, #] &, SubstTest[class, z, member[funpart[setpart[z]], w], w →
             image[inverse[DORA], composite[inverse[POWER], Q, inverse[S], POWER]]]] // Reverse

Out[27]= subclass[image[DORA, FUNS], composite[inverse[POWER], Q, inverse[S], POWER]] == True

In[28]:= subclass[image[DORA, FUNS], composite[inverse[POWER], Q, inverse[S], POWER]] := True
```