membership of first and second in domain and range

Johan G. F. Belinfante and Tiffany D. Goble
2004 February 26

```
In[1]:= << goedel54.25a; << tools.m

        :Package Title: goedel54.25a     2004 February 25 at 8:11 p.m.

        It is now:  2004 Feb 27 at 10:34

        Loading Simplification Rules

        TOOLS.M                       Revised 2004 February 21

        weightlimit = 40
```

summary

In this notebook, basic results are derived that relate first and second elements of an ordered pair to domain and range. These results are analogous to, but technically differ from theorems that had been proved using **Otter**. The results do not mention ordered pairs explicitly, the hypothesis that **x** being tranformed as follows:

```
In[2]:= member[x, cart[V, V]]

Out[2]= member[first[x], V]
```

To facilitate pattern matching, the hypothesis that **x** is an ordered pair is replaced by a more general form in which **V** is replaced by an arbitrary variable **y**.

a result analogous to Theorem DO−4B in the DO2 group

The following lemma still explicitly mentions ordered pairs.

```
In[3]:= Map[implies[member[first[x], z], #] &,
          SubstTest[or, member[u, domain[y]], not[member[u, V]], not[member[v, V]],
            not[member[pair[u, v], y]], {u → first[x], v → second[x]}]]

Out[3]= or[member[first[x], domain[y]], not[member[first[x], z]],
          not[member[pair[first[x], second[x]], y]]] == True

In[4]:= (% /. {x_ -> x_, y -> y_, z -> z_}) /. Equal -> SetDelayed
```

Theorem.

```
In[5]:= Map[not, SubstTest[and, implies[p1, p3], implies[and[p2, p3], p4],
            implies[and[p1, p4], p5], not[implies[and[p1, p2], p5]],
            {p1 → member[first[x], z], p2 → member[x, y], p3 → equal[x, pair[first[x], second[x]]],
             p4 → member[pair[first[x], second[x]], y], p5 → member[first[x], domain[y]]}]]

Out[5]= or[member[first[x], domain[y]], not[member[x, y]], not[member[first[x], z]]] == True
```

```
In[6]:= or[member[first[x_], domain[y_]],
           not[member[x_, y_]], not[member[first[x_], z_]]] := True
```

a result similar to Theorem RA−4X in the RA2 group

As in the preceding section, an initial lemma is derived in which ordered pairs appear explicitly.

```
In[7]:= SubstTest[or, member[v, range[y]], not[member[u, V]],
          not[member[v, V]], not[member[pair[u, v], y]], {u → first[x], v → second[x]}]

Out[7]= or[member[second[x], range[y]], not[member[first[x], V]],
           not[member[pair[first[x], second[x]], y]]] == True

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

The derivation is completed in the same manner:

```
In[9]:= Map[implies[member[first[x], z], not[#]] &, SubstTest[and, implies[p1, p3],
           implies[and[p2, p3], p4], implies[and[p1, p4], p5], not[implies[and[p1, p2], p5]],
           {p1 → member[first[x], V], p2 → member[x, y], p3 → equal[x, pair[first[x], second[x]]],
            p4 → member[pair[first[x], second[x]], y], p5 -> member[second[x], range[y]]}]]

Out[9]= or[member[second[x], range[y]], not[member[x, y]], not[member[first[x], z]]] == True

In[10]:= or[member[second[x_], range[y_]],
            not[member[x_, y_]], not[member[first[x_], z_]]] := True
```

two additional results of a similar nature

The results derived in this section are similar, except that the hypothesis that **x** is an ordered pair is given in terms of **second** rather than **first**.

```
In[11]:= Map[not, SubstTest[and, implies[p1, p2], implies[and[p2, p3], p4],
           not[implies[and[p1, p3], p4]], {p1 -> member[second[x], z], p2 -> member[first[x], V],
                 p3 -> member[x, y], p4 -> member[first[x], domain[y]]}]]

Out[11]= or[member[first[x], domain[y]], not[member[x, y]], not[member[second[x], z]]] == True

In[12]:= or[member[first[x_], domain[y_]],
            not[member[x_, y_]], not[member[second[x_], z_]]] := True

In[13]:= Map[not, SubstTest[and, implies[p1, p2], implies[and[p2, p3], p4],
           not[implies[and[p1, p3], p4]], {p1 -> member[second[x], z], p2 -> member[first[x], V],
                 p3 -> member[x, y], p4 -> member[second[x], range[y]]}]]

Out[13]= or[member[second[x], range[y]], not[member[x, y]], not[member[second[x], z]]] == True

In[14]:= or[member[second[x_], range[y_]],
            not[member[x_, y_]], not[member[second[x_], z_]]] := True
```

rules for inverse relations

Simpler rules can be derived for inverse relations because members of inverse relations are automatically ordered pairs.

```
In[15]:= implies[member[x, inverse[y]], member[first[x], V]] // AssertTest

Out[15]= or[member[first[x], V], not[member[x, inverse[y]]]] == True

In[16]:= or[member[first[x_], V], not[member[x_, inverse[y_]]]] := True
```

From the result of the first section, one obtains:

```
In[17]:= SubstTest[implies, and[member[first[x], V], member[x, z]],
          member[first[x], domain[z]], z -> inverse[y]]

Out[17]= or[member[first[x], range[y]],
           not[member[x, inverse[y]]], not[member[first[x], V]]] == True

In[18]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

This can be improved:

```
In[19]:= Map[not, SubstTest[and, implies[p1, p2],
            implies[and[p1, p2], p3], not[implies[p1, p3]], {p1 -> member[x, inverse[y]],
             p2 -> member[first[x], V], p3 -> member[first[x], range[y]]}]]

Out[19]= or[member[first[x], range[y]], not[member[x, inverse[y]]]] == True

In[20]:= or[member[first[x_], range[y_]], not[member[x_, inverse[y_]]]] := True
```

Similarly, from the result of the second section, one obtains:

```
In[21]:= SubstTest[implies, and[member[first[x], V], member[x, z]],
          member[second[x], range[z]], z -> inverse[y]]

Out[21]= or[member[second[x], domain[y]],
           not[member[x, inverse[y]]], not[member[first[x], V]]] == True

In[22]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

This too can be improved:

```
In[23]:= Map[not, SubstTest[and, implies[p1, p2],
            implies[and[p1, p2], p3], not[implies[p1, p3]], {p1 -> member[x, inverse[y]],
             p2 -> member[first[x], V], p3 -> member[second[x], domain[y]]}]]

Out[23]= or[member[second[x], domain[y]], not[member[x, inverse[y]]]] == True

In[24]:= or[member[second[x_], domain[y_]], not[member[x_, inverse[y_]]]] := True
```