axch and image[DORA,FUNS]

Johan G. F. Belinfante
2005 November 20

```
In[1]:= SetDirectory["l:"]; << goedel75.19a; << tools.m

        :Package Title: goedel75.19a            2005 November 19 at 9:20 p.m.

        It is now:  2005 Nov 20 at 18:10

        Loading Simplification Rules

        TOOLS.M                      Revised 2005 October 25

        weightlimit = 40
```

summary

When the axiom of choice is assumed, the relation **image[DORA,FUNS]** can be expressed in terms of the equipollence relation **Q**. When the axiom of choice is not assumed, only an inclusion for the relation **image[DORA,FUNS]** can be derived.

a lower bound for image[DORA, FUNS]

Lemma.

```
In[2]:= or[FUNCTION[union[z, cart[intersection[x, complement[domain[z]]], set[y]]]],
          not[FUNCTION[z]], not[member[y, range[z]]], not[subclass[domain[z], x]]] // AssertTest

Out[2]= or[FUNCTION[union[z, cart[intersection[x, complement[domain[z]]], set[y]]]],
          not[FUNCTION[z]], not[member[y, range[z]]], not[subclass[domain[z], x]]] == True

In[3]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

The main argument is this: one can always extend any nonempty function to one with the same range and any bigger domain by assigning to each new point of the extended domain one of the values of the original function. That is, the extended function is the union of the old function and a constant function, with the constant part using an old value so that the range remains unchanged.

```
In[4]:= Map[not, SubstTest[and, implies[and[p0, p4], p8], implies[and[p1, p2, p3, p4], p5],
          implies[and[p1, p2, p3, p4], p6], implies[and[p1, p2, p3, p4], p7],
          implies[and[p5, p6, p7, p8], p9], not[implies[and[p0, p1, p2, p3, p4], p9]],
          {p0 → and[member[x, V], member[z, V]], p1 → FUNCTION[z], p2 → subclass[domain[z], x],
           p3 -> member[y, range[z]], p4 → equal[w, union[z, cart[dif[x, domain[z]], set[y]]]],
           p5 → FUNCTION[w], p6 → equal[domain[w], x], p7 → equal[range[w], range[z]],
           p8 → member[w, V], p9 → member[pair[x, range[z]], image[DORA, FUNS]]}]] /.
        w -> union[z, cart[dif[x, domain[z]], set[y]]]

Out[4]= or[member[pair[x, range[z]], image[DORA, FUNS]], not[FUNCTION[z]], not[member[x, V]],
          not[member[y, range[z]]], not[member[z, V]], not[subclass[domain[z], x]]] == True

In[5]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

All three variables can be eliminated simultaneously:

```
In[6]:= Map[equal[0, composite[complement[#], id[cart[V, V]]]] &,
          SubstTest[class, pair[pair[x, y], z], or[member[pair[x, range[z]], w],
            not[FUNCTION[z]], not[member[x, V]], not[member[y, range[z]]], not[member[z, V]],
            not[subclass[domain[z], x]]], w → image[DORA, FUNS]]] // Reverse

Out[6]= subclass[fix[composite[complement[image[DORA, FUNS]], S, inverse[image[DORA, FUNS]]]],
          set[0]] == True

In[7]:= subclass[fix[composite[complement[image[DORA, FUNS]], S, inverse[image[DORA, FUNS]]]],
          set[0]] := True
```

Restatement:

```
In[8]:= (subclass[composite[id[complement[set[0]]], w, inverse[S]], w] // AssertTest) /.
         w → image[DORA, FUNS]

Out[8]= subclass[composite[id[complement[set[0]]], image[DORA, FUNS], inverse[S]],
          image[DORA, FUNS]] == True

In[9]:= subclass[composite[id[complement[set[0]]], image[DORA, FUNS], inverse[S]],
          image[DORA, FUNS]] := True
```

Corollary.

```
In[10]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]], subclass[u, w],
          {u -> composite[id[complement[set[0]]], Q, inverse[S]],
           v -> composite[id[complement[set[0]]], image[DORA, FUNS], inverse[S]],
           w → image[DORA, FUNS]}]

Out[10]= subclass[composite[id[complement[set[0]]], Q, inverse[S]], image[DORA, FUNS]] == True

In[11]:= subclass[composite[id[complement[set[0]]], Q, inverse[S]], image[DORA, FUNS]] := True
```

This inclusion is valid whether or not one assumes the axiom of choice.

using the axiom of choice

In this section it is shown that if the axiom of choice is assumed, an upper bound for **image[DORA, FUNS]** holds.

```
In[12]:= 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 → Q, y → inverse[S]}]

Out[12]= or[member[pair[u, w], composite[Q, inverse[S]]],
           not[member[u, V]], not[member[pair[v, w], Q]], not[subclass[v, u]]] == True

In[13]:= (% /. {u → u_, v → v_, w → w_}) /. Equal → SetDelayed

In[14]:= Map[not, SubstTest[and, implies[and[p1, p2], p4],
           implies[and[p1, p4], p5], implies[p1, p6], implies[p3, p7], implies[p5, p8],
           implies[and[p6, p7, p8], p9], not[implies[and[p1, p2, p3], p9]],
           {p1 → member[y, X[inverse[x]]], p2 → FUNCTION[x], p3 → member[x, V], p4 → ONEONE[y],
            p5 → member[pair[range[x], range[y]], Q], p6 → subclass[range[y], domain[x]],
            p7 → member[domain[x], V], p8 -> member[pair[range[y], range[x]], Q],
            p9 → member[pair[domain[x], range[x]], composite[Q, inverse[S]]]}]]

Out[14]= or[member[pair[domain[x], range[x]], composite[Q, inverse[S]]],
           not[equal[domain[y], range[x]]], not[FUNCTION[x]], not[FUNCTION[y]],
           not[member[x, V]], not[member[y, V]], not[subclass[y, inverse[x]]]] == True

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

In[16]:= Map[equal[V, #] &, SubstTest[class, y,
            implies[and[member[x, u], member[y, v]], member[pair[domain[x], range[x]], w]],
            {u → FUNS, v → X[inverse[x]], w → composite[Q, inverse[S]]}]] // Reverse

Out[16]= or[equal[0, X[inverse[x]]], member[range[x], image[Q, P[domain[x]]]],
           not[FUNCTION[x]], not[member[x, V]]] == True

In[17]:= (% /. x → x_) /. Equal → SetDelayed
```

Restatement.

```
In[18]:= implies[and[member[x, FUNS], member[inverse[x], SELECT]],
          member[range[x], image[Q, P[domain[x]]]]]

Out[18]= True
```

One can now eliminate the variable **x**.

```
In[19]:= Map[equal[V, #] &, SubstTest[class, x, implies[and[member[x, u], member[inverse[x], v]],
             member[range[x], image[w, P[domain[x]]]]],
            {u → FUNS, v → SELECT, w → Q}]] // Reverse

Out[19]= subclass[image[DORA, intersection[FUNS, image[inverse[IMAGE[SWAP]], SELECT]]],
           composite[Q, inverse[S]]] == True
```

```
In[20]:= subclass[image[DORA, intersection[FUNS, image[inverse[IMAGE[SWAP]], SELECT]]],
           composite[Q, inverse[S]]] := True
```

Corollary.

```
In[21]:= SubstTest[implies, equal[SELECT, v],
          subclass[image[DORA, intersection[FUNS, image[inverse[IMAGE[SWAP]], v]]],
           composite[Q, inverse[S]]], v → V]

Out[21]= or[not[axch], subclass[image[DORA, FUNS], composite[Q, inverse[S]]]] == True

In[22]:= % /. Equal → SetDelayed
```

---

a conditional rewrite rule

In this section it is shown that if the axiom of choice is assumed, an equation expressing **image[DORA, FUNS]** in terms of **Q** can be derived

Lemma.

```
In[23]:= SubstTest[composite, union[u, v], image[DORA, FUNS],
           {u → id[set[0]], v → id[complement[set[0]]]}] // Reverse

Out[23]= union[cart[set[0], set[0]], composite[id[complement[set[0]]], image[DORA, FUNS]]] ==
          image[DORA, FUNS]

In[24]:= % /. Equal → SetDelayed
```

Lemma.

```
In[25]:= Map[not, SubstTest[and, implies[p1, p2],
             implies[p2, p3], implies[p3, p4], not[implies[p1, p4]],
             {p1 → axch, p2 -> subclass[image[DORA, FUNS], composite[Q, inverse[S]]],
              p3 -> subclass[composite[u, image[DORA, FUNS]], composite[u, Q, inverse[S]]],
              p4 -> subclass[composite[u, image[DORA, FUNS]],
                union[id[set[0]], composite[u, Q, inverse[S]]]]}]] /. u → id[complement[set[0]]]

Out[25]= or[not[axch], subclass[composite[id[complement[set[0]]], image[DORA, FUNS]], union[
            cart[set[0], set[0]], composite[id[complement[set[0]]], Q, inverse[S]]]]] == True

In[26]:= % /. Equal → SetDelayed
```

Lemma.

```
In[27]:= Map[implies[axch, #] &, SubstTest[subclass, union[x, z], union[y, z],
           {z -> cart[set[0], set[0]], x -> composite[id[complement[set[0]]], image[DORA, FUNS]],
            y -> composite[id[complement[set[0]]], Q, inverse[S]]}]]

Out[27]= or[not[axch], subclass[image[DORA, FUNS], union[cart[set[0], set[0]],
            composite[id[complement[set[0]]], Q, inverse[S]]]]] == True
```

```
In[28]:= % /. Equal → SetDelayed
```

Theorem. An equation expressing **image[DORA, FUNS]** in terms of **Q** when **axch** holds.

```
In[29]:= SubstTest[and, implies[axch, subclass[u, v]], subclass[v, u], {u -> image[DORA, FUNS],
           v → union[id[set[0]], composite[id[complement[set[0]]], Q, inverse[S]]]}] // Reverse

Out[29]= or[equal[image[DORA, FUNS], union[cart[set[0], set[0]],
            composite[id[complement[set[0]]], Q, inverse[S]]]], not[axch]] == True

In[30]:= or[equal[image[DORA, FUNS], union[cart[set[0], set[0]],
            composite[id[complement[set[0]]], Q, inverse[S]]]], not[axch]] := True
```

A conditional rewrite rule may be more useful if one does plan to assume that **axch** holds.

```
In[31]:= image[DORA, FUNS] :=
          union[cart[set[0], set[0]], composite[id[complement[set[0]]], Q, inverse[S]]] /; axch
```