FUL-REG3

Johan G. F. Belinfante
2004 December 30

```
In[1]:= SetDirectory["i:"]; << goedel64.28a; << tools.m

        :Package Title: goedel64.28a              2004 December 28 at 8:50 a.m.

        It is now:  2004 Dec 30 at 10:2

        Loading Simplification Rules

        TOOLS.M                       Revised 2004 December 21

        weightlimit = 40
```

summary

Theorem **FUL-REG3**, proved using **Otter** on 1998 March 14, says that any nonempty full regular set holds the empty set. A more general theorem holds; one can replace set with class. This more general theorem is derived in this notebook, imitating the proof found by **Otter**. The **Otter** proof contains a Skolem function which was denoted **notsub(x, image(E, x)) = n$(x)**. The derivation presented below introduces a new variable **y** in place of this Skolem function.

imitating Otter's proof

The variable **y** is introduced to replace the Skolem function produced by a non-subvariance statement.

```
In[2]:= assert[exists[y, and[member[y, x], not[member[y, image[E, x]]]]]] ==
          not[subvariant[E, x]]

Out[2]= True
```

Most of the proof produced by **Otter** amounts to the following argument:

```
In[3]:= Map[not, SubstTest[and, implies[and[p1, p2], p4], implies[and[p3, p4], p5],
          implies[and[p2, p5], p6], not[implies[and[p1, p2, p3], p6]],
          {p1 → subclass[U[x], x], p2 → member[y, x], p3 → equal[0, intersection[x, y]],
           p4 → subclass[y, x], p5 → equal[0, y], p6 → member[0, x]}]]

Out[3]= or[member[0, x], not[equal[0, intersection[x, y]]],
          not[member[y, x]], not[subclass[U[x], x]]] == True

In[4]:= or[member[0, x_], not[equal[0, intersection[x_, y_]]],
          not[member[y_, x_]], not[subclass[U[x_], x_]]] := True
```

The variable **y** is eliminated in the usual way.

```
In[5]:= Map[equal[V, #] &,
          SubstTest[class, y, or[member[z, x], not[equal[0, intersection[x, y]]],
            not[full[x]], not[member[y, x]], z → 0]] // Reverse

Out[5]= or[equal[0, intersection[x, P[complement[x]]]],
          member[0, x], not[subclass[U[x], x]]] == True

In[6]:= (% /. x → x_) /. Equal → SetDelayed
```

The remaining step in **Otter**'s proof is the following:

```
In[7]:= Map[not, SubstTest[and, implies[and[p2, p3], p4], implies[and[p1, p4], p5],
          not[implies[and[p1, p2, p3], p5]], {p1 -> member[x, REGULAR], p2 → full[x],
           p3 → not[member[0, x]], p4 → subvariant[E, x], p5 → equal[0, x]}]]

Out[7]= or[equal[0, x], member[0, x],
          not[member[x, REGULAR]], not[subclass[U[x], x]]] == True
```

This is Theorem **FUL-REG3**.

```
In[8]:= or[equal[0, x_], member[0, x_],
          not[member[x_, REGULAR]], not[subclass[U[x_], x_]]] := True
```

To generalize this, a lemma is needed:

```
In[9]:= SubstTest[or, equal[V, y], not[equal[V, union[y, REGULAR]]],
         not[subclass[P[y], y]], {y → complement[x]}]

Out[9]= or[equal[0, x], not[equal[0, intersection[x, P[complement[x]]]]],
          not[subclass[x, REGULAR]]] == True

In[10]:= or[equal[0, x_], not[equal[0, intersection[x_, P[complement[x_]]]]],
           not[subclass[x_, REGULAR]]] := True
```

The last step in **Otter**'s proof is repeated, replacing **member[x, REGULAR]** with **subclass[x, REGULAR]**.

```
In[11]:= Map[not, SubstTest[and, implies[and[p1, p2, p3], p4], implies[and[p1, p4], p5],
           not[implies[and[p1, p2, p3], p5]], {p1 → subclass[x, REGULAR], p2 → full[x],
            p3 → not[member[0, x]], p4 → subvariant[E, x], p5 → equal[0, x]}]]

Out[11]= or[equal[0, x], member[0, x],
           not[subclass[x, REGULAR]], not[subclass[U[x], x]]] == True
```

This generalizes Theorem **FUL-REG3**.

```
In[12]:= or[equal[0, x_], member[0, x_],
           not[subclass[x_, REGULAR]], not[subclass[U[x_], x_]]] := True
```

---

variable-free reformulation

When **x** is a set, this variable can be eliminated to produce a variable-free reformulation of Theorem **FUL-REG3**.

```
In[13]:= Map[equal[V, #] &,
           SubstTest[class, x, implies[member[x, z], or[equal[0, x], member[0, x]]],
            z → intersection[FULL, REGULAR]]] // Reverse

Out[13]= subclass[intersection[FULL,
            P[intersection[REGULAR, complement[singleton[0]]]]], singleton[0]] == True

In[14]:= % /. Equal → SetDelayed
```

This inclusion can be sharpened to an equation.

```
In[15]:= equal[intersection[FULL, P[intersection[REGULAR, complement[singleton[0]]]]],
           singleton[0]] // AssertTest

Out[15]= equal[intersection[FULL, P[intersection[REGULAR, complement[singleton[0]]]]],
           singleton[0]] == True

In[16]:= intersection[FULL, P[intersection[REGULAR, complement[singleton[0]]]]] :=
          singleton[0]
```

Corollary.

```
In[17]:= SubstTest[U, intersection[FULL, P[intersection[REGULAR, x]]],
           x -> complement[singleton[0]]] // Reverse

Out[17]= intersection[REGULAR, H[complement[singleton[0]]]] == 0
```

```
In[18]:= intersection[REGULAR, H[complement[singleton[0]]]] := 0
```

If the axiom of regularity is assumed, then this implies that **H[complement[singleton[0]]]** is empty:

```
In[19]:= SubstTest[implies, equal[v, REGULAR],
          disjoint[v, H[complement[singleton[0]]]], v → V]

Out[19]= or[equal[0, H[complement[singleton[0]]]], not[equal[REGULAR, V]]] == True
```