a wrapper for wellorderable sets

Johan G. F. Belinfante
2007 May 2

```
In[1]:= SetDirectory["l:"]; << goedel93.02a; << tools.m

        :Package Title: goedel93.02a      2007 May 2 at 3:05 p.m.

        It is now:  2007 May 2 at 16:18

        Loading Simplification Rules

        TOOLS.M                       Revised 2007 March 25

        weightlimit = 40
```

summary

In the **GOEDEL** program, the axiom of choice is not automatically assumed to hold. Consequently, a given set may or may not be in one to one correspondence with an ordinal. The class **image[Q, OMEGA]** holds those sets that can be put in one-to-one correspondence with an ordinal. Such sets are sometimes called **wellorderable**. In this notebook, a wrapper **wob[x]** for wellorderable sets is introduced.

```
In[2]:= Begin["Goedel`Private`"];

In[3]:= InfoMatch[class[w_, member[x_, HoldPattern[wob[y_]]]]]

Out[3]//TableForm=
        class[w_, member[x_, wob[y_]]] := class[w, and[member[x, y], member[y, image[Q, OMEGA]]
```

The wrapper **wob[x]** is useful for theorems about cardinality. The cardinality **card[x]** of a wellorderable set is the least ordinal with which **x** is in one-to-one correspondence. If **x** is not a wellorderable set, then **card[x] = V**. The function **CARD** assigns to each wellorderable set its cardinality. The domain of **CARD** is **image[Q, OMEGA]**.

normalizing the wrapper for wellorderable sets

The wrapper **wob[x]** is normalized as follows:

```
In[4]:= wob[x] // Normality // Reverse

Out[4]= intersection[x, image[V, intersection[OMEGA, image[Q, set[x]]]]] == wob[x]

In[5]:= intersection[x_, image[V, intersection[OMEGA, image[Q, set[x_]]]]] := wob[x]
```

```
In[6]:= SubstTest[member, intersection[x, image[V, intersection[w, image[Q, set[x]]]]],
          image[Q, w], w → OMEGA] // Reverse
```

```
Out[6]= member[wob[x], image[Q, OMEGA]] == True
```

```
In[7]:= member[wob[x_], image[Q, OMEGA]] := True
```

Corollary.

```
In[8]:= SubstTest[implies, member[t, w],
          member[t, V], {t -> wob[x], w → image[Q, OMEGA]}] // Reverse
```

```
Out[8]= member[wob[x], V] == True
```

```
In[9]:= member[wob[x_], V] := True
```

wrapper removal rule

Lemma.

```
In[10]:= equiv[or[equal[0, x], member[x, image[Q, OMEGA]]], member[x, image[Q, OMEGA]]]
```

```
Out[10]= True
```

```
In[11]:= or[equal[0, x_], member[x_, image[Q, OMEGA]]] := member[x, image[Q, OMEGA]]
```

Theorem. (Wrapper-removal rule.)

```
In[12]:= SubstTest[equal, x,
           intersection[x, image[V, intersection[w, image[Q, set[x]]]]], w → OMEGA] // Reverse
```

```
Out[12]= equal[x, wob[x]] == member[x, image[Q, OMEGA]]
```

```
In[13]:= equal[x_, wob[x_]] := member[x, image[Q, OMEGA]]
```

The following conditional rule is useful.

```
In[14]:= wob[x_] := x /; member[x, image[Q, OMEGA]]
```

equipollence rule

Theorem. (Wellorderable sets are equipollent if they have the same cardinality.)

```
In[15]:= SubstTest[member, pair[wob[x], wob[y]], intersection[Q, t], t → cartsq[image[Q, OMEGA]]]
```

```
Out[15]= member[pair[wob[x], wob[y]], Q] == equal[card[wob[x]], card[wob[y]]]
```

```
In[16]:= member[pair[wob[x_], wob[y_]], Q] := equal[card[wob[x]], card[wob[y]]]
```

From the standpoint of pattern matching, the equipollence statement is not symmetric with respect to its two arguments. This rewrite rule has the desirable feature that it replaces the equipollence statement with a logically equivalent equality statement which is symmetric with respect to the two variables **x** and **y**, since **equal** has the **Orderless** property.

---

reify rule

```
In[17]:= SubstTest[reify, x, intersection[f[x],
            image[V, intersection[w, image[Q, set[f[x]]]]], w → OMEGA] // Reverse

Out[17]= reify[x, wob[f[x]]] == composite[reify[x, f[x]],
           id[image[inverse[VERTSECT[reify[x, f[x]]], image[Q, OMEGA]]]]

In[18]:= reify[x_, wob[y_]] :=
          composite[reify[x, y], id[image[inverse[VERTSECT[reify[x, y]]], image[Q, OMEGA]]]]
```