rewrite rules for COARSER

Johan G. F. Belinfante
2004 October 9

```
In[1]:= SetDirectory["i:"]; << goedel62.08a; << tools.m
        :Package Title: goedel62.08a          2004 October 8 at 9:55 a.m.
        It is now:  2004 Oct 9 at 11:0
        Loading Simplification Rules
        TOOLS.M                      Revised 2004 September 25
        weightlimit = 40
```

summary

Some rewrite rules for **image[COARSER, x]** and **image[inverse[COARSER], x]** are derived to clear the way for applications to compactness. Note that the following characterization of compactness involves both such expressions:

```
In[2]:= subclass[image[inverse[COARSER], singleton[x]], image[COARSER, FINITE]]
Out[2]= or[member[x, COMPACT], not[member[x, V]]]
```

image[COARSER, x]

Rule 1.

```
In[3]:= member[x, image[COARSER, y]] // AssertTest // Reverse
Out[3]= member[U[x], image[inverse[S], image[BIGCUP, intersection[y, P[x]]]]] ==
         member[x, image[COARSER, y]]

In[4]:= member[U[x_], image[inverse[S], image[BIGCUP, intersection[y_, P[x_]]]]] :=
         member[x, image[COARSER, y]]
```

Rule 2.

```
In[5]:= SubstTest[image, intersection[u, v], x,
          {u → S, v → composite[inverse[S], POWER, BIGCUP]}] // Reverse
```

```
Out[5]= fix[composite[S, id[x], inverse[BIGCUP], inverse[POWER], S]] ==
         image[COARSER, x]
```

```
In[6]:= fix[composite[S, id[x_], inverse[BIGCUP], inverse[POWER], S]] :=
         image[COARSER, x]
```

---

sethood result

Lemma 1.

```
In[7]:= SubstTest[implies, and[thin[t], member[x, V]],
         member[image[t, x], V], t → COARSER]
```

```
Out[7]= or[member[image[COARSER, x], V], not[member[x, V]]] == True
```

```
In[8]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma 2.

```
In[9]:= SubstTest[implies, and[subclass[x, y], member[y, V]],
         member[x, V], y → image[COARSER, x]]
```

```
Out[9]= or[member[x, V], not[member[image[COARSER, x], V]]] == True
```

```
In[10]:= (% /. x → x_) /. Equal → SetDelayed
```

Combining these lemmas, one obtains:

```
In[11]:= equiv[member[image[COARSER, x], V], member[x, V]]
```

```
Out[11]= True
```

This justifies the following rewrite rule:

```
In[12]:= member[image[COARSER, x_], V] := member[x, V]
```

---

image[inverse[COARSER], x]

Rule 1.

```
In[13]:= fix[composite[inverse[S], id[x], inverse[BIGCUP], BIGCUP]] // Normality

Out[13]= fix[composite[inverse[S], id[x], inverse[BIGCUP], BIGCUP]] ==
          image[inverse[COARSER], x]

In[14]:= fix[composite[inverse[S], id[x_], inverse[BIGCUP], BIGCUP]] :=
          image[inverse[COARSER], x]
```

Rule 2.

```
In[15]:= SubstTest[fix, composite[inverse[S], POWER, BIGCUP, id[x],
           intersection[S, y]], y → composite[inverse[BIGCUP], BIGCUP]]

Out[15]= fix[composite[inverse[S], POWER, BIGCUP, id[x], COARSER]] ==
          image[inverse[COARSER], x]

In[16]:= fix[composite[inverse[S], POWER, BIGCUP, id[x_], COARSER]] :=
          image[inverse[COARSER], x]
```

Rule 3.

```
In[17]:= image[inverse[COARSER], singleton[x]] // Normality // Reverse

Out[17]= intersection[image[inverse[BIGCUP], image[S, singleton[U[x]]]], P[x]] ==
          image[inverse[COARSER], singleton[x]]

In[18]:= intersection[image[inverse[BIGCUP], image[S, singleton[U[x_]]]], P[x_]] :=
          image[inverse[COARSER], singleton[x]]
```

---

sethood results

Lemma 1.

```
In[19]:= SubstTest[implies, and[thin[t], member[x, V]],
          member[image[t, x], V], t → inverse[COARSER]]

Out[19]= or[member[image[inverse[COARSER], x], V], not[member[x, V]]] == True

In[20]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma 2.

```
In[21]:= SubstTest[implies, and[subclass[x, y], member[y, V]],
          member[x, V], y → image[inverse[COARSER], x]]

Out[21]= or[member[x, V], not[member[image[inverse[COARSER], x], V]]] == True

In[22]:= (% /. x → x_) /. Equal → SetDelayed
```

Combining these lemmas, one obtains:

```
In[23]:= equiv[member[image[inverse[COARSER], x], V], member[x, V]]
Out[23]= True
```

This justifies the following rewrite rule:

```
In[24]:= member[image[inverse[COARSER], x_], V] := member[x, V]
```