Theorems from the SC, A and PC groups

Johan G. F. Belinfante and Tiffany D. Goble
2004 March 4

```
In[1]:= << goedel54.28a; << tools.m

        :Package Title: goedel54.28a     2004 February 28 at 5:40 a.m.

        It is now:  2004 Mar 5 at 13:10

        Loading Simplification Rules

        TOOLS.M                       Revised 2004 February 21

        weightlimit = 40
```

summary

Some theorems about sum class, power class and unary intersection that were proved using **Otter** are rederived here to make them available in the **GOEDEL** program.

Theorem SC−C

The following is equivalent to Theorem **SC−C** proved 1998 February 21 using **Otter**.

```
In[2]:= SubstTest[U, union[x, y], y -> complement[x]] // Reverse

Out[2]= union[U[x], U[complement[x]]] == V

In[3]:= union[U[x_], U[complement[x_]]] := V
```

Theorem A−SC−DJ

Theorem **A−SC−DJ** was proved 1999 April 23 using McCune's automated theorem proving program **Otter**. The derivation can be done all in one step:

```
In[4]:= Map[assert[forall[z, not[#]]] &,
          SubstTest[and, implies[p1, p3], implies[p2, p4], implies[and[p3, p4], p5],
                 not[implies[and[p1, p2], p5]],
            {p1 -> member[z, x], p2 -> member[z, y],
           p3 -> subclass[A[x], z], p4 -> subclass[z, U[y]], p5 -> subclass[A[x], U[y]]}]]

Out[4]= or[equal[0, intersection[x, y]], subclass[A[x], U[y]]] == True

In[5]:= or[equal[0, intersection[x_, y_]], subclass[A[x_], U[y_]]] := True
```

Theorem PC−A−SU

Theorem **PC−A−SU** was proved 1999 December 29 using **Otter**.

```
In[6]:= SubstTest[implies, and[subclass[u, v], subclass[v, y]],
          subclass[u, y], {u -> A[x], v -> U[x]}]

Out[6]= or[equal[0, x], not[subclass[U[x], y]], subclass[A[x], y]] == True
```

```
In[7]:= or[equal[0, x_], not[subclass[U[x_], y_]], subclass[A[x_], y_]] := True
```

Theorem PC−SC−4

Theorem **PC−SC−4** was proved 1997 September 7 using **Otter**.

```
In[8]:= SubstTest[implies, subclass[u, y], subclass[U[u], U[y]], u → P[x]]

Out[8]= or[not[subclass[P[x], y]], subclass[x, U[y]]] == True
```

```
In[9]:= or[not[subclass[P[x_], y_]], subclass[x_, U[y_]]] := True
```

The following companion result holds:

```
In[10]:= Map[assert, SubstTest[implies, subclass[u, x], subclass[A[x], A[u]], u -> P[y]]]

Out[10]= or[equal[0, A[x]], not[subclass[P[y], x]]] == True
```

```
In[11]:= or[equal[0, A[x_]], not[subclass[P[y_], x_]]] := True
```