# a rewrite rule for function application

*Johan G. F. Belinfante*
*2004 October 29*

```
In[1]:= SetDirectory["i:"]; << goedel62.28b; << tools.m

        :Package Title: goedel62.28b         2004 October 28 at 11:50 p.m.

        It is now:  2004 Oct 29 at 10:16

        Loading Simplification Rules

        TOOLS.M                       Revised 2004 October 28

        weightlimit = 40
```

## summary

A rewrite rule is derived that allows one to obtain statements about function application from membership statements.

## FUNCTION rule

Lemma using a **funpart** wrapper.

```
In[2]:= Map[implies[#, equal[y, APPLY[funpart[z], x]]] &,
          SubstTest[member, y, image[w, singleton[x]], w → funpart[z]]] // Reverse

Out[2]= or[equal[y, APPLY[funpart[z], x]],
          not[equal[image[z, singleton[x]], singleton[y]]],
          not[member[x, V]], not[member[y, V]]] == True

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

Main result.

```
In[4]:= SubstTest[implies, and[equal[w, funpart[z]], member[pair[x, y], w]],
         equal[y, APPLY[w, x]], w → z]

Out[4]= or[equal[y, APPLY[z, x]], not[FUNCTION[z]], not[member[pair[x, y], z]]] == True

In[5]:= or[equal[y_, APPLY[z_, x_]],
          not[FUNCTION[z_]], not[member[pair[x_, y_], z_]]] := True
```

The following conditional rewrite rule allows one to use the main result with less effort.

```
In[6]:= or[equal[y_, APPLY[z_, x_]], not[member[pair[x_, y_], z_]]] :=
         True /; FUNCTION[z]
```

---

## improving a lemma

The lemma used in the preceding section can be improved upon. The hypothesis **member[x,V]** can be eliminated.

```
In[7]:= implies[and[equal[image[z, singleton[x]], singleton[y]], member[y, w]],
          member[x, V]] // AssertTest

Out[7]= or[member[x, V],
          not[equal[image[z, singleton[x]], singleton[y]]], not[member[y, w]]] == True

In[8]:= or[member[x_, V], not[equal[image[z_, singleton[x_]], singleton[y_]]],
          not[member[y_, w_]]] := True
```

This is an improved version of the **funpart** lemma in the preceding section.

```
In[9]:= Map[not, SubstTest[and, implies[and[p1, p2], p3], implies[p1, p4],
          implies[and[p2, p3, p4], p5], not[implies[and[p1, p2], p5]],
          {p1 → member[y, w], p2 -> equal[image[z, singleton[x]], singleton[y]],
           p3 → member[x, V], p4 -> member[y, V], p5 → equal[y, APPLY[funpart[z], x]]}]]

Out[9]= or[equal[y, APPLY[funpart[z], x]],
          not[equal[image[z, singleton[x]], singleton[y]]], not[member[y, w]]] == True

In[10]:= or[equal[y_, APPLY[funpart[z_], x_]],
           not[equal[image[z_, singleton[x_]], singleton[y_]]],
           not[member[y_, w_]]] := True
```