# a substitute for Quaife's Theorem AP−11

Johan G. F. Belinfante and Tiffany Goble
2004 January 22

```
In[1]:= << goedel53.22b; << tools.m

        :Package Title: goedel53.22b      2004 January 22 at 4:30 p.m.

        It is now:  2004 Jan 23 at 10:14

        Loading Simplification Rules

        TOOLS.M                       Revised 2004 January 3

        weightlimit = 40
```

## summary

A substitute is derived for Quaife's theorem **AP−11** with **APPLY** in place of **apply**. Recall the definitions:

```
In[2]:= apply[x, y] == U[image[x, singleton[y]]]

Out[2]= True

In[3]:= APPLY[x, y] == A[image[x, singleton[y]]]

Out[3]= True
```

The literal about the **domain** in Quaife's theorem is not needed for the substitute theorem.

## derivation

Lemma 1.

```
In[4]:= SubstTest[implies, equal[w, y], equal[composite[x, w], composite[x, y]], w → funpart[y]]

Out[4]= or[equal[composite[x, y], composite[x, funpart[y]]], not[FUNCTION[y]]] == True

In[5]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma 2.

```
In[6]:= SubstTest[implies, equal[u, v], equal[APPLY[u, z], APPLY[v, z]],
         {u -> composite[x, y], v -> composite[x, funpart[y]]}]

Out[6]= or[equal[A[image[x, image[y, singleton[z]]]], APPLY[x, APPLY[funpart[y], z]]],
          not[equal[composite[x, y], composite[x, funpart[y]]]]] == True

In[7]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma 3.

```
In[8]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
          {p1 → FUNCTION[y], p2 → equal[composite[x, y], composite[x, funpart[y]]],
           p3 → equal[APPLY[composite[x, y], z], APPLY[x, APPLY[funpart[y], z]]]}]]

Out[8]= or[equal[A[image[x, image[y, singleton[z]]]], APPLY[x, APPLY[funpart[y], z]]],
          not[FUNCTION[y]]] == True

In[9]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

The substitute theorem:

```
In[10]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3], implies[p3, p4],
           implies[and[p2, p4], p5], not[implies[p1, p5]], {p1 → FUNCTION[y],
            p2 -> equal[APPLY[composite[x, funpart[y]], z], APPLY[composite[x, y], z]],
            p3 → equal[APPLY[funpart[y], z], APPLY[y, z]],
            p4 → equal[APPLY[x, APPLY[funpart[y], z]], APPLY[x, APPLY[y, z]]],
            p5 -> equal[APPLY[composite[x, y], z], APPLY[x, APPLY[y, z]]]}]]

Out[10]= or[equal[A[image[x, image[y, singleton[z]]]], APPLY[x, APPLY[y, z]]],
           not[FUNCTION[y]]] == True

In[11]:= or[equal[A[image[x_, image[y_, singleton[z_]]]], APPLY[x_, APPLY[y_, z_]]],
           not[FUNCTION[y_]]] := True
```

Restatement:

```
In[12]:= or[not[FUNCTION[y]], equal[APPLY[composite[x, y], z], APPLY[x, APPLY[y, z]]]]

Out[12]= True
```