iterate and thinness

Johan G. F. Belinfante
2002 May 19

```
<< goedel52.n84; << tools.m
```

```
:Package Title: GOEDEL52.n84          2002 May 19 at 2:50 p.m.

It is now:  2002 May 19 at 18:37

Loading Simplification Rules

TOOLS.M                   Revised 2002 May 17

weightlimit = 40
```

■ Prerequisites

```
Map[implies[#, member[image[x, y], V]] &,
 SubstTest[member, y, image[inverse[IMAGE[x]], z], z -> V]]
```

```
or[member[image[x, y], V], not[member[y, V]],
  not[subclass[y, domain[VERTSECT[x]]]]] == True
```

```
or[member[image[x_, y_], V], not[member[y_, V]],
  not[subclass[y_, domain[VERTSECT[x_]]]]] := True
```

This can be restated thus:

```
implies[member[y, domain[IMAGE[x]]], member[image[x, y], V]]
```

```
True
```

■ Introduction

We wish to show by an induction on the domain that if **x** is thin, and **y** is a set, then **iterate[x,y]** is a set. Since the domain of **iterate[x,y]** is a set, it would suffice to show that **iterate[x,y]** is thin. The induction step is this:

```
Map[equal[V, #] &, SubstTest[class, w, implies[thin[x], implies[member[w, z],
    member[succ[w], z]]], z -> domain[VERTSECT[iterate[x, y]]]]] // Reverse
```

```
or[not[equal[V, domain[VERTSECT[x]]]],
  subclass[image[SUCC, domain[VERTSECT[iterate[x, y]]]],
   domain[VERTSECT[iterate[x, y]]]]] == True
```

```
or[not[equal[V, domain[VERTSECT[x_]]]],
  subclass[image[SUCC, domain[VERTSECT[iterate[x_, y_]]]],
   domain[VERTSECT[iterate[x_, y_]]]]] := True
```

We now use induction:

```
SubstTest[implies, and[member[0, w], subclass[image[SUCC, w], w]],
 subclass[omega, w], w -> domain[VERTSECT[iterate[x, y]]]]

or[not[member[y, V]], not[subclass[
    image[SUCC, domain[VERTSECT[iterate[x, y]]]], domain[VERTSECT[iterate[x, y]]]],
  subclass[omega, domain[VERTSECT[iterate[x, y]]]]] == True

or[not[member[y_, V]], not[subclass[
    image[SUCC, domain[VERTSECT[iterate[x_, y_]]]], domain[VERTSECT[iterate[x_, y_]]]],
  subclass[omega, domain[VERTSECT[iterate[x_, y_]]]]] := True
```

■ The rest of the story.

```
Map[not, SubstTest[and, implies[p1, p3], implies[and[p2, p3], p4],
        not[implies[and[p1, p2], p4]], {p1 -> thin[x], p2 -> member[y, V],
    p3 -> subclass[
     image[SUCC, domain[VERTSECT[iterate[x, y]]]], domain[VERTSECT[iterate[x, y]]]],
    p4 -> subclass[omega, domain[VERTSECT[iterate[x, y]]]]}]]

or[not[equal[V, domain[VERTSECT[x]]]], not[member[y, V]],
  subclass[omega, domain[VERTSECT[iterate[x, y]]]]] == True

or[not[equal[V, domain[VERTSECT[x_]]]], not[member[y_, V]],
  subclass[omega, domain[VERTSECT[iterate[x_, y_]]]]] := True

SubstTest[member, composite[Id, w], V, w -> iterate[x, y]] // Reverse

member[range[iterate[x, y]], V] == member[iterate[x, y], V]

member[range[iterate[x_, y_]], V] := member[iterate[x, y], V]

SubstTest[implies, member[v, P[domain[VERTSECT[u]]]], member[image[u, v], V],
    {v -> omega, u -> iterate[x, y]}]

or[member[iterate[x, y], V],
  not[subclass[omega, domain[VERTSECT[iterate[x, y]]]]]] == True

or[member[iterate[x_, y_], V],
  not[subclass[omega, domain[VERTSECT[iterate[x_, y_]]]]]] := True

Map[not, SubstTest[and, implies[and[p1, p2], p3], implies[p3, p4],
        not[implies[and[p1, p2], p4]],
{p1 -> thin[x], p2 -> member[y, V],
   p3 -> subclass[omega, domain[VERTSECT[iterate[x, y]]]],
    p4 -> member[iterate[x, y], V]}]]

or[member[iterate[x, y], V],
  not[equal[V, domain[VERTSECT[x]]]], not[member[y, V]]] == True

or[member[iterate[x_, y_], V],
  not[equal[V, domain[VERTSECT[x_]]]], not[member[y_, V]]] := True
```