Uclosure rules

Johan G. F. Belinfante
2001 October 16

<< goedel52.k83; << tests.m

:Package Title: GOEDEL52.K83 2001 October 16 at 10:15 p.m.

It is now: 2001 Oct 17 at 21:44

Loading Simplification Rules

TESTS.M Revised 2001 October 15

weightlimit = 30

Context switch to 'Goedel'Private is needed for ReplaceTest

Just ignore the error message about Untermined use of BeginPackage

Get::bepal : Untermined uses of BeginPackage or Begin in << tests.m.

Introduction

In this notebook we derive a rule for Uclosure that had resisted a direct attack. The main tools used in the proof are AssertTest and SubstTest. The latter test just compares two different orders of evaluation for some expression.

?? assert

assert[p] is a statement equivalent to p obtained by applying Goedel's algorithm to class[w,p]. Applying assert repeatedly sometimes simplifies a statement.

assert[p_] := Module[{w = Unique[]}, equal[V, class[w, p]]]

?? AssertTest

Goedel'Private'AssertTest

AssertTest[x_] := x == assert[x]

?? SubstTest

Goedel'Private'SubstTest

SubstTest[f_, x___, r_] := f @@@ ({x} /. r) == (f @@@ {x} /. r)

A second ingredient in the proof, of a more minor nature, is the use of a modified ordered pair:

PAIR[x, y]

A[cart[singleton[x], singleton[y]]]
This agrees with `pair[x,y]` when \( x \) and \( y \) are both sets.

```latex
equal[PAIR[a, b], pair[a, b]] // assert
True
```

When either or both of \( x \) and \( y \) is a proper class, the modified ordered pair reduces to \( V \).

```latex
{PAIR[p, y], PAIR[x, p]} 
{V, V}
```

### A general theorem

```latex
theorem[x_, y_, z_, w_] := 
implies[and[member[x, z], member[PAIR[x, y], w]], member[y, image[w, z]]]
```

We first prove the theorem when \( x \) is a set:

```latex
forall[x, theorem[x, y, z, w]] // assert
True
```

Next we consider the case that \( x \) is a proper class:

```latex
theorem[p, y, z, w]
True
```

We introduce a temporary rule based on the theorem just proved:

```latex
theorem[x_, y_, z_, w_] == True
or[member[y_, image[w_, z_]], 
not[member[A[cart[singleton[x_], singleton[y_]]], w_]], not[member[x_, z_]]] == True
or[member[y_, image[w_, z_]], 
not[member[A[cart[singleton[x_], singleton[y_]]], w_]], not[member[x_, z_]]] := True
```

### A corollary for image under BIGCUP.

We need a membership rule for \( \text{BIGCUP} \) that applies to the modified ordered pair.

```latex
member[PAIR[x, y], BIGCUP] // AssertTest 
member[A[cart[singleton[x], singleton[y]]], BIGCUP] == and[equal[y, U[x]], member[y, V]]
member[A[cart[singleton[x_], singleton[y_]]], BIGCUP] := 
and[equal[y, U[x]], member[y, V]]
```

With this in place, we derive a corollary of our theorem that we actually need to use.
This rule is simple enough that it could be made permanent:

\[ \text{or} \left[ \text{member} \left[ U[x], \text{image} \left[ \text{BIGCUP}, z \right] \right], \text{not} \left[ \text{member} \left[ x, z \right] \right] \right] : = \text{True} \]

\section*{A corollary for Uclosure.}

\[ \text{or} \left[ \text{member} \left[ U[x\_], \text{image} \left[ \text{BIGCUP}, z\_ \right] \right], \text{not} \left[ \text{member} \left[ x\_, z\_ \right] \right] \right] : = \text{True} \]

\section*{Derivation of the rule of interest.}

First we prove a little lemma:

\[ \text{image} \left[ \text{inverse} \left[ \text{SINGLETON} \right], \text{image} \left[ \text{SINGLETON}, x \right] \right] \] // Normality

\[ \text{image} \left[ \text{inverse} \left[ \text{SINGLETON} \right], \text{image} \left[ \text{SINGLETON}, x \right] \right] : = x \]

This rule should probably be made permanent:

\[ \text{image} \left[ \text{inverse} \left[ \text{SINGLETON} \right], \text{image} \left[ \text{SINGLETON}, x\_ \right] \right] : = x \]

Next we apply \textbf{SubstTest} to get the difficult half of the result we want.

\[ \text{subclass} \left[ \text{P} \left[ z\_ \right], \text{Uclosure} \left[ \text{image} \left[ \text{SINGLETON}, z\_ \right] \right] \right] : = \text{True} \]

We add this as a temporary rule:

\[ \text{subclass} \left[ \text{P} \left[ z\_ \right], \text{Uclosure} \left[ \text{image} \left[ \text{SINGLETON}, z\_ \right] \right] \right] : = \text{True} \]

The final step is to strengthen this to equality:

\[ \text{equal} \left[ \text{Uclosure} \left[ \text{image} \left[ \text{SINGLETON}, x \right] \right], P \left[ x \right] \right] \] // AssertTest

\[ \text{equal} \left[ \text{Uclosure} \left[ \text{image} \left[ \text{SINGLETON}, x \right] \right], P \left[ x \right] \right] : = \text{True} \]

This is the first rule we sought:

\[ \text{Uclosure} \left[ \text{image} \left[ \text{SINGLETON}, x\_ \right] \right] : = P \left[ x \right] \]
Derivation of a second rule.

We use the first rule to derive a second, more interesting, rule. First a temporary rule, which does all the hard work.

```
SubstTest[implies, subclass[u, v], subclass[Uclosure[u], Uclosure[v]],
{u -> image[SINGLETON, U[x]], v -> image[inverse[S], x]}]
subclass[P[U[x]], Uclosure[image[inverse[S], x]]] == True
subclass[P[U[x_]], Uclosure[image[inverse[S], x_]]] := True
```

Again, it turns out to be easy to strengthen this to equality.

```
equal[Uclosure[image[inverse[S], x]], P[U[x]]] // AssertTest
equal[Uclosure[image[inverse[S], x]], P[U[x]]] == True
Uclosure[image[inverse[S], x_]] := P[U[x]]
```