Uclosure intersection rule

Johan G. F. Belinfante
2001 October 18

```
<< goedel52.k85; << tests.m

:Package Title: GOEDEL52.K85      2001 October 18 at 8:00 a.m.

It is now:  2001 Oct 18 at 11:42

Loading Simplification Rules

TESTS.M                   Revised 2001 October 18

weightlimit = 30


Context switch to ‘Goedel‘Private is needed for ReplaceTest

Just ignore the error message about Unterminated use of BeginPackage

Get::bebal : Unterminated uses of BeginPackage or Begin in << tests.m.
```

■ Introduction

What do we need to show to prove the following conjecture?

```
equal[Uclosure[intersection[x, P[y]]], intersection[Uclosure[x], P[y]]] // assert

and[subclass[intersection[P[y], Uclosure[x]], Uclosure[intersection[x, P[y]]]],
 subclass[Uclosure[intersection[x, P[y]]], Uclosure[x]]]
```

The second condition can be easily disposed of:

```
SubstTest[implies, subclass[u, x], subclass[Uclosure[u], Uclosure[x]],
u -> intersection[x, P[y]]]

subclass[Uclosure[intersection[x, P[y]]], Uclosure[x]] == True


subclass[Uclosure[intersection[x_, P[y_]]], Uclosure[x_]] := True
```

The first condition took more ingenuity, but after much experimentation I hit on the following proof.

```
SubstTest[implies, subclass[u, v], subclass[range[u], range[v]],
    {u -> composite[id[P[y]], BIGCUP, id[P[x]]],
        v -> composite[BIGCUP, id[P[intersection[x, P[y]]]]]}]

subclass[intersection[P[y], Uclosure[x]], Uclosure[intersection[x, P[y]]]] == True


subclass[intersection[P[y_], Uclosure[x_]], Uclosure[intersection[x_, P[y_]]]] := True
```

With these two lemmas out of the way, we can prove the main conjecture:

```
equal[Uclosure[intersection[x, P[y]]], intersection[Uclosure[x], P[y]]] // AssertTest
```

```
equal[Uclosure[intersection[x, P[y]]], intersection[P[y], Uclosure[x]]] == True
```

The following rule will be made permanent:

```
Uclosure[intersection[x_, P[y_]]] := intersection[P[y], Uclosure[x]]
```

■ Serenditipity

The following interesting rule was discovered as a bonus:

```
Map[U, SubstTest[Uclosure, intersection[y, P[x]], y -> FUNS]]
```

```
U[intersection[FUNS, P[x]]] == composite[Id, x]
```

Can this rule be proved directly? I think so.

```
Map[U, AssInt[FUNS, P[cart[V, V]], P[x]]]
```

```
U[intersection[FUNS, P[composite[Id, x]]]] == U[intersection[FUNS, P[x]]]
```

```
U[intersection[FUNS, P[composite[Id, x_]]]] := U[intersection[FUNS, P[x]]]
```

```
SubstTest[subclass, U[intersection[u, P[v]]], v,
    {u -> FUNS, v -> composite[Id, x]}]
```

```
subclass[U[intersection[FUNS, P[x]]], cart[V, V]] == True
```

```
subclass[U[intersection[FUNS, P[x_]]], cart[V, V]] := True
```

```
or[not[subclass[u, v]], subclass[intersection[u, P[x]], v]] // AssertTest
```

```
or[not[subclass[u, v]], subclass[intersection[u, P[x]], v]] == True
```

```
or[not[subclass[u_, v_]], subclass[intersection[u_, P[x_]], v_]] := True
```

```
SubstTest[implies, subclass[u, v],
    subclass[intersection[u, P[x]], intersection[v, P[x]]],
    {u -> image[SINGLETON, cart[V, V]], v -> FUNS}]
```

```
subclass[image[SINGLETON, composite[Id, x]], FUNS] == True
```

```
subclass[image[SINGLETON, composite[Id, x]], FUNS] := True
```

```
SubstTest[implies, subclass[u, v], subclass[U[u], U[v]],
    {u -> image[SINGLETON, composite[Id, x]], v -> intersection[FUNS, P[x]]}]
```

```
subclass[composite[Id, x], U[intersection[FUNS, P[x]]]] == True
```

```
subclass[composite[Id, x_], U[intersection[FUNS, P[x_]]]] := True
```

```
equal[U[intersection[FUNS, P[x]]], composite[Id, x]] // AssertTest
```

```
equal[U[intersection[FUNS, P[x]]], composite[Id, x]] == True
```

```
U[intersection[FUNS, P[x_]]] := composite[Id, x]
```

```
CORE[FUNS] // VSNormality
```

```
CORE[FUNS] == IMAGE[id[cart[V, V]]]
```

```
CORE[FUNS] := IMAGE[id[cart[V, V]]]
```