Tutorial #1. Goedel's primitives.

Johan G. F. Belinfante
2001 January 11

```
<< goedel52.a3; << tests.m
```

```
:Package Title: GOEDEL52.A3   2001 January 2 at 8:20 a.m.

It is now:  2001 Jan 11 at 7:28

Loading Simplification Rules

TESTS.M         Revised 2000 December 30

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.
```

■ GOEDEL's primitives

The following examples explain the meaning of Goedel's primitives. We begin with the universal class **V** and the membership relation **E**.

```
class[x, True]
```

```
V
```

```
class[pair[x, y], member[x, y]]
```

```
E
```

There are four primitive unary constructors:

```
class[w, not[member[w, x]]]
```

```
complement[x]
```

```
class[u, exists[v, member[pair[u, v], x]]]
```

```
domain[x]
```

```
class[pair[pair[u, v], w], member[pair[pair[v, u], w], x]] == flip[x]
```

```
True
```

```
class[pair[pair[u, v], w], member[pair[pair[v, w], u], x]]
```

```
rotate[x]
```

There are three primitive binary constructors:

```
equal[class[w, or[equal[w, x], equal[w, y]]], pairset[x, y]] // assert
```

```
True
```

```
class[w, and[member[w, x], member[w, y]]]
```

```
intersection[x, y]
```

```
class[pair[u, v], and[member[u, x], member[v, y]]]
```

```
cart[x, y]
```

In the **GOEDEL** program, the constructors **flip** and **pairset** have been de−emphasized. We rewrite **flip** as a composite with the function **SWAP**.

```
flip[x]
```

```
composite[x, SWAP]
```

The function **SWAP** interchanges the two members of an ordered pair:

```
class[pair[pair[u, v], pair[x, y]], and[equal[u, y], equal[v, x]]]
```

```
SWAP
```

The constructor **pairset** has been largely replaced as a primitive by **singleton**.

```
pairset[x, x]
```

```
singleton[x]
```

The general connection between **pairset** and **singleton** is revealed by a **Normality** test:

```
pairset[x, y] // Normality
```

```
pairset[x, y] == union[singleton[x], singleton[y]]
```

Goedel also introduced **inverse** as a primitive, but it can be expressed in terms of his other primitives:

```
domain[flip[cart[x, V]]]
```

```
inverse[x]
```

The meaning of **inverse** is better understood from its conventional description:

```
class[pair[u, v], member[pair[v, u], x]]
```

```
inverse[x]
```

■ The rotate constructor.

The constructor **rotate** is not familiar to most mathematicians, and it could in fact be eliminated in favor of more familiar quantities. To do so, we introduce the constructor **composite**, which could be defined in terms of the primitives as:

```
domain[intersection[rotate[flip[cart[x, V]]], flip[rotate[cart[y, V]]]]]
```

```
composite[x, y]
```

The meaning of **composite** is more clearly explained by its more conventional characterization:

```
class[pair[u, w], exists[v, and[member[pair[u, v], y], member[pair[v, w], x]]]]
```

```
composite[x, y]
```

The subclass relation **S** and the identity relation **Id** are definable in terms of the membership relation **E** using **composite**:

```
intersection[cart[V, V], complement[composite[complement[E], inverse[E]]]]
```

```
S
```

```
intersection[S, inverse[S]]
```

```
Id
```

More understandable descriptions of these relations are:

```
class[pair[x, y], subclass[x, y]]
```

```
S
```

```
class[pair[x, y], equal[x, y]]
```

```
Id
```

The function **SWAP** is definable in terms of Goedel's primitives as

```
flip[Id]
```

```
SWAP
```

It is also convenient to introduce the functions **FIRST** and **SECOND**, which can be defined in terms of the primitives in several ways. One can for example use either one of the following as a starting point:

```
domain[rotate[Id]]
```

```
FIRST
```

```
rotate[cart[Id, V]]
```

```
SECOND
```

These functions are related to each other by **flip**:

```
flip[FIRST]
```

```
SECOND
```

```
flip[SECOND]
```

```
FIRST
```

The meaning of **FIRST** and **SECOND** is made tranparent by the following descriptions:

```
class[pair[pair[u, v], w], equal[w, u]]
```

```
FIRST
```

```
class[pair[pair[u, v], w], equal[w, v]]
```

```
SECOND
```

Goedel introduced both **flip** and **rotate** for the algorithm on which the **GOEDEL** program is based. For this application one does not actually need to apply **rotate** to an arbitrary ternary relation; one mainly needs to apply **rotate** to ternary relations of the form **cart[x,V]**, where **x** is a binary relation.

```
rotate[cart[x, V]]
```

```
composite[x, SECOND]
```

Using **FIRST** and **SECOND**, one can in principle eliminate **rotate** altogether:

```
rotate[x] == composite[SECOND, intersection[composite[inverse[x], FIRST],
             composite[inverse[FIRST], SECOND]]]
```

```
True
```

The discovery of this formula was one of the first applications of the **GOEDEL** program. Recently we have found that **rotate** is actually quite useful, so we have stopped using the above formula to eliminate it.