symmetric restriction

Johan G. F. Belinfante
2007 October 21

```
In[1]:= SetDirectory["l:"]; << goedel98.19a; << tools.m

        :Package Title: goedel98.19a         2007 October 19 at 10:05 p.m.

        It is now:  2007 Oct 21 at 9:10

        Loading Simplification Rules

        TOOLS.M                                        Revised 2007 September 19

        weightlimit = 40
```

summary

The term **symmetric restriction** in this notebook refers to the process of cutting down a relation by intersecting it with a cartesian square, that is, by simultaneously restricting both domain and range to the same set. The erasure relation for symmetric restriction is given by the expression:

```
In[2]:= class[pair[x, y],
          exists[t, and[member[t, image[CART, Id]], equal[y, intersection[t, x]]]]]

Out[2]= composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]
```

Another expression for this erasure relation is derived below by a suitable sequence of rewrite rules.

rewrite rules

Theorem.

```
In[3]:= composite[inverse[E], CLIQUES, DISJOINT] // RelnNormality // Reverse

Out[3]= composite[inverse[POWER], DISJOINT, IMAGE[PAIRSET]] ==
          composite[inverse[E], CLIQUES, DISJOINT]

In[4]:= composite[inverse[POWER], DISJOINT, IMAGE[PAIRSET]] :=
          composite[inverse[E], CLIQUES, DISJOINT]
```

Theorem.

```
In[5]:= composite[CAP, id[cart[V, image[CART, Id]]],
          inverse[FIRST], IMAGE[id[cart[V, V]]]] // ReifNormality

Out[5]= composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST], IMAGE[id[cart[V, V]]]] ==
         composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]

In[6]:= composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST], IMAGE[id[cart[V, V]]]] :=
         composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]
```

Theorem.

```
In[7]:= composite[id[P[x]], intersection[y, composite[inverse[S], IMAGE[id[x]]]]] //
         ReifNormality

Out[7]= composite[id[P[x]], intersection[y, composite[inverse[S], IMAGE[id[x]]]]] ==
         intersection[y, composite[inverse[S], IMAGE[id[x]]]]

In[8]:= composite[id[P[x_]], intersection[y_, composite[inverse[S], IMAGE[id[x_]]]]] :=
         intersection[y, composite[inverse[S], IMAGE[id[x]]]]
```

Theorem.

```
In[9]:= Assoc[composite[CAP, id[cart[V, y]], inverse[FIRST]], id[P[x]],
          intersection[composite[S, IMAGE[id[x]]], inverse[S]]] // Reverse

Out[9]= composite[CAP, id[cart[P[x], y]], inverse[FIRST],
          intersection[composite[S, IMAGE[id[x]]], inverse[S]]] ==
         composite[CAP, id[cart[V, y]], inverse[FIRST], IMAGE[id[x]]]

In[10]:= composite[CAP, id[cart[P[x_], y_]], inverse[FIRST],
           intersection[composite[S, IMAGE[id[x_]]], inverse[S]]] :=
          composite[CAP, id[cart[V, y]], inverse[FIRST], IMAGE[id[x]]]
```

main theorem

Technical lemma. (This takes a while.)

```
In[12]:= Map[fix, Map[inverse, composite[inverse[SECOND], inverse[S], id[image[CART, Id]],
             inverse[IMAGE[id[cart[V, V]]]], complement[composite[FIRST,
               intersection[composite[inverse[SECOND], IMAGE[id[cart[V, V]]], SECOND],
                composite[inverse[DIF], E, inverse[E], FIRST]]]]] // TriRenormality]]

Out[12]= fix[composite[complement[composite[
              intersection[composite[inverse[SECOND], inverse[IMAGE[id[cart[V, V]]]], SECOND],
               composite[inverse[FIRST], E, inverse[E], DIF]], inverse[FIRST]]],
            IMAGE[id[cart[V, V]]], id[image[CART, Id]], S, SECOND]] ==
          composite[id[P[cart[V, V]]], fix[composite[inverse[DIF], DISJOINT,
             id[image[CART, Id]], S, IMAGE[id[cart[V, V]]], SECOND]]]

In[13]:= % /. Equal → SetDelayed
```

Main Theorem. (This also takes quite a while.)

```
In[14]:= (composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST], IMAGE[id[cart[x, V]]]] //
             VSNormality // Reverse) /. x → V

Out[14]= intersection[composite[inverse[S], IMAGE[id[cart[V, V]]]], fix[composite[inverse[DIF],
             DISJOINT, id[image[CART, Id]], S, IMAGE[id[cart[V, V]]], SECOND]]] ==
          composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]

In[15]:= intersection[composite[inverse[S], IMAGE[id[cart[V, V]]]], fix[composite[inverse[DIF],
             DISJOINT, id[image[CART, Id]], S, IMAGE[id[cart[V, V]]], SECOND]]] :=
          composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]
```

Comment. The following similar rewrite rule without the **IMAGE[id[cart[V,V]]]** function was already available in the **GOEDEL** program:

```
In[16]:= intersection[inverse[S],
          fix[composite[inverse[DIF], DISJOINT, id[image[CART, Id]], S, SECOND]]]

Out[16]= composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]
```

The technical lemma was needed to reduce the expression with the extra **IMAGE[id[cart[V, V]]]** to the one without it.

---

the symmetric erasure relation

Intersecting a class **x** with a cartesian square **cart[u, u]** is equivalent to composing it with **id[u]** on both sides:

```
In[17]:= intersection[x, cart[u, u]]

Out[17]= composite[id[u], x, id[u]]
```

The new rewrite rules enable one to express the erasure relation for symmetric restriction in the following natural way:

```
In[18]:= class[pair[x, y], exists[u, equal[y, composite[id[u], x, id[u]]]]]

Out[18]= composite[CAP, id[cart[V, image[CART, Id]]], inverse[FIRST]]
```