cutting with rectangles

Johan G. F. Belinfante
2007 October 19

```
In[1]:= SetDirectory["l:"]; << goedel98.18a; << tools.m

        :Package Title: goedel98.18a         2007 October 18 at 5:35 p.m.

        It is now:  2007 Oct 19 at 20:46

        Loading Simplification Rules

        TOOLS.M                                      Revised 2007 September 19

        weightlimit = 40
```

summary

In this notebook, it is shown that the following erasure relations are equal:

```
In[2]:= class[pair[x, y], assert[exists[u, v, equal[y, composite[id[u], x, id[v]]]]]]

Out[2]= composite[id[P[cart[V, V]]], intersection[inverse[S],
          UB[union[E, composite[inverse[DORA], inverse[CART], complement[E]]]]]]

In[3]:= class[pair[x, y], exists[t, and[member[t, range[CART]], equal[y, intersection[t, x]]]]]

Out[3]= composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]
```

Each of these is a different description of the process of cutting down a relation by intersecting it with a cartesian product, that is, by separately restricting both its domain and range.

vertical sections

The main idea that will be used to prove the equality of the expressions for the rectangular-cut erasure relation is to show that their vertical sections are equal. One needs to show that the following two expressions are equal.

```
In[8]:= class[y, assert[exists[t, and[member[t, range[CART]], equal[y, intersection[t, x]]]]]]

Out[8]= image[IMAGE[id[x]], range[CART]]

In[9]:= class[y, assert[exists[u, v, equal[y, composite[id[u], x, id[v]]]]]]

Out[9]= intersection[fix[composite[S, IMAGE[id[x]], CART, DORA]], P[composite[Id, x]]]
```

An inclusion in one direction is immediate.

```
In[10]:= SubstTest[subclass, fix[t], range[t],
           t -> composite[IMAGE[id[x]], CART, DORA]] // Reverse

Out[10]= subclass[fix[composite[IMAGE[id[x]], CART, DORA]],
           image[IMAGE[id[x]], range[CART]]] == True

In[11]:= (% /. x → x_) /. Equal → SetDelayed
```

For the reverse inclusion, the main lemma one needs is this:

```
In[12]:= SubstTest[composite, id[range[t]], t,
           id[domain[t]], t → composite[id[u], x, id[v]]] // Reverse

Out[12]= composite[id[intersection[u, image[x, v]]], x,
           id[intersection[v, image[inverse[x], u]]]] == composite[id[u], x, id[v]]

In[13]:= composite[id[intersection[u_, image[x_, v_]]], x_,
           id[intersection[v_, image[inverse[x_], u_]]]] := composite[id[u], x, id[v]]
```

The following temporary lemma will be later be removed and replaced with a better one.

```
In[14]:= fix[composite[IMAGE[id[x]], CART, DORA]] // Renormality // Reverse

Out[14]= intersection[fix[composite[S, IMAGE[id[x]], CART, DORA]], P[composite[Id, x]]] ==
          fix[composite[IMAGE[id[x]], CART, DORA]]

In[15]:= intersection[fix[composite[S, IMAGE[id[x_]], CART, DORA]], P[composite[Id, x_]]] :=
          fix[composite[IMAGE[id[x]], CART, DORA]]
```

The variable **y** can now be eliminated as follows.

```
In[16]:= Map[implies[and[member[u, V], member[v, V]], equal[V, #]] &,
           SubstTest[class, y, or[equal[y, composite[id[range[y]], x, id[domain[y]]]],
             not[equal[y, t]]], t -> composite[id[u], x, id[v]]]]

Out[16]= or[member[composite[id[u], x, id[v]], fix[composite[IMAGE[id[x]], CART, DORA]]],
           not[member[u, V]], not[member[v, V]]] == True

In[17]:= (% /. {u → u_, v → v_}) /. Equal → SetDelayed
```

Next the variables **u** and **v** are eliminated, yielding an inclusion in the reverse direction.

```
In[18]:= Map[empty[composite[Id, complement[#]]] &, SubstTest[class, pair[v, u],
           implies[member[pair[u, v], z], member[composite[id[u], x, id[v]], t]],
           {t -> fix[composite[IMAGE[id[x]], CART, DORA]], z → cart[V, V]}]]

Out[18]= subclass[image[IMAGE[id[x]], range[CART]],
           fix[composite[IMAGE[id[x]], CART, DORA]]] == True

In[19]:= (% /. x → x_) /. Equal → SetDelayed
```

The two inclusions can be combined into an equation for vertical sections.

```
In[20]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> image[IMAGE[id[x]], range[CART]], v -> fix[composite[IMAGE[id[x]], CART, DORA]]}]

Out[20]= equal[fix[composite[IMAGE[id[x]], CART, DORA]],
           image[IMAGE[id[x]], range[CART]]] == True

In[21]:= fix[composite[IMAGE[id[x_]], CART, DORA]] := image[IMAGE[id[x]], range[CART]]
```

Corollary. Since its range and fixed point sets are equal, the function **composite[IMAGE[id[x]], CART, DORA]** is idempotent.

```
In[22]:= SubstTest[implies, and[FUNCTION[t], equal[range[t], fix[t]]],
          idempotent[t], t -> composite[IMAGE[id[x]], CART, DORA]]

Out[22]= True == equal[composite[IMAGE[id[x]], CART, DORA],
           composite[IMAGE[id[x]], CART, DORA, IMAGE[id[x]], CART, DORA]]

In[23]:= composite[IMAGE[id[x_]], CART, DORA, IMAGE[id[x_]], CART, DORA] :=
          composite[IMAGE[id[x]], CART, DORA]
```

replacement for the temporary lemma

The temporary lemma must first be removed before it can be replaced.

```
In[25]:= intersection[fix[composite[S, IMAGE[id[x_]], CART, DORA]], P[composite[Id, x_]]] =.
```

Lemma.

```
In[26]:= fix[composite[inverse[S], IMAGE[id[x]], CART, DORA]] // Renormality

Out[26]= fix[composite[inverse[S], IMAGE[id[x]], CART, DORA]] == P[composite[Id, x]]

In[27]:= fix[composite[inverse[S], IMAGE[id[x_]], CART, DORA]] := P[composite[Id, x]]
```

Theorem. (Replacement for the temporary lemma.)

```
In[28]:= SubstTest[intersection, fix[composite[S, funpart[t]]],
           fix[composite[inverse[S], funpart[t]]],
           t -> composite[IMAGE[id[x]], CART, DORA]] // Reverse

Out[28]= intersection[fix[composite[S, IMAGE[id[x]], CART, DORA]], P[composite[Id, x]]] ==
          image[IMAGE[id[x]], range[CART]]

In[29]:= intersection[fix[composite[S, IMAGE[id[x_]], CART, DORA]], P[composite[Id, x_]]] :=
          image[IMAGE[id[x]], range[CART]]
```

a simplification rule

Lemma.

```
In[30]:= SubstTest[implies, equal[x, composite[Id, t]],
           subclass[x, cart[V, range[x]]], t → x] // Reverse

Out[30]= or[not[subclass[x, cart[V, V]]], subclass[x, cart[V, range[x]]]] == True

In[31]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem.

```
In[32]:= equiv[subclass[x, cart[V, range[x]]], subclass[x, cart[V, V]]]

Out[32]= True

In[33]:= subclass[x_, cart[V, range[x_]]] := subclass[x, cart[V, V]]
```

Comment. A similar rule for **domain[x]** is already in the **GOEDEL** program.

rectangular cuts

It is an unavoidable feature of Gödel's algorithm that equations are broken up into two inclusions, which are then subject to further rewrite rules that could prevent the two inclusions from being reunited into an equation. The following lemma is needed to reunite two inclusions for the equation that says that **y** is obtained from **x** by intersecting with a rectangle.

```
In[34]:= equal[composite[id[range[y]], x, id[domain[y]]], y] // AssertTest // Reverse

Out[34]= and[subclass[y, x], subclass[y, cart[V, V]],
           subclass[composite[id[range[y]], x, id[domain[y]]], y]] ==
         equal[y, composite[id[range[y]], x, id[domain[y]]]]

In[35]:= and[subclass[composite[id[range[y_]], x_, id[domain[y_]]], y_],
           subclass[y_, cart[V, V]], subclass[y_, x_]] :=
         equal[composite[id[range[y]], x, id[domain[y]]], y]
```

Eliminating the variables **x** and **y** yields this equation for the corresponding erasure relation:

```
In[36]:= SubstTest[class, pair[x, y],
          member[y, fix[composite[IMAGE[id[x]], t]]], t → composite[CART, DORA]]

Out[36]= intersection[fix[
           composite[inverse[DIF], DISJOINT, intersection[S, composite[CART, DORA]], SECOND]],
          inverse[S]] == composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]

In[37]:= intersection[fix[composite[inverse[DIF], DISJOINT,
             intersection[S, composite[CART, DORA]], SECOND]], inverse[S]] :=
          composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]
```

a membership relation for DORA

Theorem.

```
In[38]:= member[pair[x, y], composite[z, DORA]] // AssertTest

Out[38]= member[pair[x, y], composite[z, DORA]] ==
          and[member[x, V], member[y, V], member[pair[pair[domain[x], range[x]], y], z]]

In[39]:= member[pair[x_, y_], composite[z_, DORA]] :=
          and[member[x, V], member[y, V], member[pair[pair[domain[x], range[x]], y], z]]
```

standardization rules

Lemma. The following rewrite rule helps avoid slight variants of a certain **UB** expression encountered below.

```
In[40]:= composite[id[P[x]], intersection[inverse[S], UB[y]]] // RelnNormality // Reverse

Out[40]= intersection[composite[inverse[S], IMAGE[id[x]]], UB[y]] ==
          composite[id[P[x]], intersection[inverse[S], UB[y]]]

In[41]:= intersection[composite[inverse[S], IMAGE[id[x_]]], UB[y_]] :=
          composite[id[P[x]], intersection[inverse[S], UB[y]]]
```

Theorem.

```
In[42]:= fix[composite[inverse[DIF], DISJOINT,
              intersection[S, composite[CART, DORA]], SECOND]] // RelnRenormality // Reverse

Out[42]= composite[id[P[cart[V, V]]],
           UB[union[E, composite[inverse[DORA], inverse[CART], complement[E]]]]] ==
          fix[composite[inverse[DIF], DISJOINT, intersection[S, composite[CART, DORA]], SECOND]]

In[43]:= composite[id[P[cart[V, V]]],
           UB[union[E, composite[inverse[DORA], inverse[CART], complement[E]]]]] :=
          fix[composite[inverse[DIF], DISJOINT, intersection[S, composite[CART, DORA]], SECOND]]
```

Main Theorem.

```
In[44]:= SubstTest[intersection, inverse[S], composite[id[P[cart[V, V]]], t],
          t -> UB[union[E, composite[inverse[DORA], inverse[CART], complement[E]]]]]

Out[44]= composite[id[P[cart[V, V]]], intersection[inverse[S],
           UB[union[E, composite[inverse[DORA], inverse[CART], complement[E]]]]]] ==
          composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]

In[45]:= composite[id[P[cart[V, V]]], intersection[inverse[S],
           UB[union[E, composite[inverse[DORA], inverse[CART], complement[E]]]]]] :=
          composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]
```

The complicated expression encountered earlier now does not appear in the following erasure relation:

```
In[46]:= class[pair[x, y], assert[exists[u, v, equal[y, composite[id[u], x, id[v]]]]]]

Out[46]= composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]
```

reify result

Using **reify**, one finds yet another formula for the rectangular-cut erasure relation.

```
In[47]:= SubstTest[reify, x, fix[composite[IMAGE[id[x]], t]], t → composite[CART, DORA]]

Out[47]= composite[intersection[IMG, composite[inverse[DORA], inverse[CART], SECOND]],
           inverse[FIRST], IMAGE[DUP]] ==
         composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]

In[48]:= composite[intersection[IMG, composite[inverse[DORA], inverse[CART], SECOND]],
           inverse[FIRST], IMAGE[DUP]] :=
         composite[CAP, id[cart[V, range[CART]]], inverse[FIRST]]
```