DORA and RESTRICT

Johan G. F. Belinfante
2013 July 13

```
In[1]:= SetDirectory["l:"]; << goedel.13jul12a

        :Package Title: goedel.13jul12a                    2013 July 12 at 4:05 p.m.

        Loading takes about seventeen minutes, half that time due to builtin pauses.

        It is now:  2013 Jul 13 at 14:51

        Loading Simplification Rules

        TOOLS.M is now incorporated in the GOEDEL program as of 2010 September 3

        weightlimit = 40

        Loading completed.

        It is now:  2013 Jul 13 at 15:8
```

summary

The function **IPD** takes any set **x** to the restriction of **IMAGE[x]** to the power set of the domain of **x**. Although the function **IMAGE[x]** itself is a proper class, this restriction is a set when **x** is a set.

```
In[3]:= APPLY[IPD, setpart[x]]

Out[3]= composite[IMAGE[setpart[x]], id[P[domain[setpart[x]]]]]
```

An important application of **IPD** is the power functor in category theory. A succinct formula for the function **IPD** is derived in this notebook, as well as some rewrite rules involving composites with **IPD**.

derivation

The key observation leading to the main result is this:

```
In[4]:= image[DORA, RS[x]]

Out[4]= composite[IMAGE[x], id[P[domain[x]]]]
```

The variable **x** can be eliminated using **reify**. The orientation of this rewrite rule is suggested by a desire to move functions to the right in composites.

Theorem. (Result obtained by eliminating the variable **x**.)

```
In[5]:= SubstTest[reify, x, image[t, RS[x]], t → DORA]
Out[5]= composite[DORA, RESTRICT] == composite[inverse[E], IPD]

In[6]:= composite[DORA, RESTRICT] := composite[inverse[E], IPD]
```

Corollary. A simple formula for **IPD**.

```
In[7]:= SubstTest[VERTSECT, composite[funpart[t], RESTRICT], t → DORA]
Out[7]= composite[IMAGE[DORA], VERTSECT[RESTRICT]] == IPD

In[8]:= composite[IMAGE[DORA], VERTSECT[RESTRICT]] := IPD
```

---

various rewrite rules involving composites of IPD

Theorem.

```
In[9]:= Assoc[DORA, RESTRICT, RESTRICT] // Reverse
Out[9]= composite[inverse[E], IPD, RESTRICT] == composite[inverse[E], IPD]

In[10]:= composite[inverse[E], IPD, RESTRICT] := composite[inverse[E], IPD]
```

Corollary.

```
In[11]:= SubstTest[VERTSECT, composite[inverse[E], IPD, t], t → RESTRICT]
Out[11]= composite[BIGCUP, IMAGE[IPD], VERTSECT[RESTRICT]] == IPD

In[12]:= composite[BIGCUP, IMAGE[IPD], VERTSECT[RESTRICT]] := IPD
```

Lemma. Simplification rule.

```
In[13]:= composite[DORA, inverse[S], IMAGE[id[cart[V, V]]]] // FastReifNormality
Out[13]= composite[DORA, inverse[S], IMAGE[id[cart[V, V]]]] == composite[DORA, inverse[S]]

In[14]:= composite[DORA, inverse[S], IMAGE[id[cart[V, V]]]] := composite[DORA, inverse[S]]
```

Theorem.

```
In[15]:= Assoc[DORA, RESTRICT, inverse[S]] // Reverse
Out[15]= composite[inverse[E], IPD, inverse[S]] == composite[DORA, inverse[S]]

In[16]:= composite[inverse[E], IPD, inverse[S]] := composite[DORA, inverse[S]]
```

Corollary.

```
In[17]:= SubstTest[VERTSECT, composite[inverse[E], IPD, t], t → inverse[S]]
Out[17]= composite[BIGCUP, IMAGE[IPD], POWER] == composite[IMAGE[DORA], POWER]

In[18]:= composite[BIGCUP, IMAGE[IPD], POWER] := composite[IMAGE[DORA], POWER]
```

Theorem. A simplification rule.

```
In[30]:= composite[IMAGE[DORA], POWER, IMAGE[id[cart[V, V]]]] // FastReifNormality
Out[30]= composite[IMAGE[DORA], POWER, IMAGE[id[cart[V, V]]]] == composite[IMAGE[DORA], POWER]

In[31]:= composite[IMAGE[DORA], POWER, IMAGE[id[cart[V, V]]]] := composite[IMAGE[DORA], POWER]
```

Theorem.

```
In[20]:= Assoc[DORA, RESTRICT, FUNPART]
Out[20]= composite[DORA, inverse[S], FUNPART] == composite[inverse[E], IPD, FUNPART]

In[21]:= composite[DORA, inverse[S], FUNPART] := composite[inverse[E], IPD, FUNPART]
```

Corollary.

```
In[32]:= Assoc[composite[DORA, inverse[S]], FUNPART, OOPART]
Out[32]= composite[DORA, inverse[S], OOPART] == composite[inverse[E], IPD, OOPART]

In[33]:= composite[DORA, inverse[S], OOPART] := composite[inverse[E], IPD, OOPART]
```

Corollary.

```
In[34]:= Assoc[composite[DORA, inverse[S]], FUNPART, VS]
Out[34]= composite[DORA, inverse[S], VS] == composite[inverse[E], IPD, VS]

In[35]:= composite[DORA, inverse[S], VS] := composite[inverse[E], IPD, VS]
```

Theorem.

```
In[38]:= Assoc[composite[IMAGE[DORA], POWER], FUNPART, OOPART]
Out[38]= composite[IMAGE[DORA], POWER, OOPART] == composite[IPD, OOPART]

In[39]:= composite[IMAGE[DORA], POWER, OOPART] := composite[IPD, OOPART]
```

Theorem.

```
In[40]:= Assoc[composite[IMAGE[DORA], POWER], FUNPART, VS]
Out[40]= composite[IMAGE[DORA], POWER, VS] == composite[IPD, VS]

In[41]:= composite[IMAGE[DORA], POWER, VS] := composite[IPD, VS]
```

Theorem.

```
In[43]:= Assoc[DORA, RESTRICT, COMPOSE]
Out[43]= composite[DORA, COMPOSE, cross[Id, RESTRICT]] == composite[inverse[E], IPD, COMPOSE]

In[44]:= composite[DORA, COMPOSE, cross[Id, RESTRICT]] := composite[inverse[E], IPD, COMPOSE]
```

Theorem.

```
In[48]:= Map[VERTSECT, Assoc[DORA, RESTRICT, inverse[E]]]
Out[48]= composite[IMAGE[DORA], IMAGE[RESTRICT]] == composite[BIGCUP, IMAGE[IPD]]

In[49]:= composite[IMAGE[DORA], IMAGE[RESTRICT]] := composite[BIGCUP, IMAGE[IPD]]
```