image[DORA,BINOPS]

Johan G. F. Belinfante
2006 July 9

```
In[1]:= SetDirectory["l:"]; << goedel83.09a; << tools.m

        :Package Title: goedel83.09a         2006 July 9 at 8:15 p.m.

        It is now:  2006 Jul 9 at 22:57

        Loading Simplification Rules

        TOOLS.M                       Revised 2006 July 8

        weightlimit = 40
```

summary

For unary operations, one has:

```
In[2]:= image[DORA, UNOPS]

Out[2]= union[cart[set[0], set[0]], composite[id[complement[set[0]]], inverse[S]]]
```

A similar result holds for binary operations, and can be deduced as a consequence of the above formula. The key idea is to associate to each unary operation an appropriate binary operation, and thereby obtain a lower bound on **image[DORA,-BINOPS]**.

an upper bound

Lemma.

```
In[3]:= composite[id[complement[set[0]]], inverse[S],
          inverse[DUP], inverse[CART], id[complement[set[0]]]] // VSNormality

Out[3]= composite[id[complement[set[0]]], inverse[S],
          inverse[DUP], inverse[CART], id[complement[set[0]]]] ==
         composite[id[complement[set[0]]], inverse[S], inverse[DUP], inverse[CART]]

In[4]:= % /. Equal → SetDelayed
```

Theorem.

```
In[5]:= SubstTest[subclass, image[DORA, BINOPS], intersection[u, v],
          {u -> union[cart[complement[set[0]], complement[set[0]]], cart[set[0], set[0]]],
           v -> composite[inverse[S], inverse[DUP], inverse[CART]]}]

Out[5]= subclass[image[DORA, BINOPS], union[cart[set[0], set[0]],
          composite[id[complement[set[0]]], inverse[S], inverse[DUP], inverse[CART]]]] == True

In[6]:= % /. Equal → SetDelayed
```

---

lower bound

```
In[7]:= Map[not, SubstTest[and, implies[p2, p5], implies[p2, p6],
           implies[and[p6, p2], p8], implies[p2, p9], implies[p2, q1],
           implies[and[p5, p8, p9, q1], q2], implies[q2, q3], not[implies[p2, q3]],
           {p2 → and[member[x, V], equal[y, composite[x, FIRST, id[cart[domain[x], domain[x]]]]],
              FUNCTION[x], equal[z, domain[x]], subclass[range[x], z]],
            p5 → equal[domain[y], cart[z, z]], p6 → equal[range[y], range[x]],
            p8 → subclass[range[y], z], p9 → FUNCTION[y], q1 → member[y, V],
            q2 -> member[y, map[cart[z, z], z]], q3 → member[y, BINOPS]}]] /.
         y -> composite[x, FIRST, id[cart[domain[x], domain[x]]]]

Out[7]= or[member[composite[x, FIRST, id[cart[domain[x], domain[x]]]], BINOPS],
          not[equal[z, domain[x]]], not[FUNCTION[x]],
          not[member[x, V]], not[subclass[range[x], z]]] == True

In[8]:= (% /. {x → x_, z → z_}) /. Equal → SetDelayed

In[9]:= Map[not, SubstTest[and, implies[p1, p3], implies[p1, p4],
           implies[p1, p5], implies[p1, p6], implies[and[p2, p3, p4, p5, p6], p7],
           not[implies[and[p1, p2], p7]], {p1 → member[x, map[z, z]],
            p2 → equal[y, composite[x, FIRST, id[cart[domain[x], domain[x]]]]],
            p3 → FUNCTION[x], p4 → equal[z, domain[x]], p5 → member[x, V],
            p6 → subclass[range[x], z], p7 → member[y, BINOPS]}]] /.
         {y -> composite[x, FIRST, id[cart[domain[x], domain[x]]]], z → domain[x]}

Out[9]= or[member[composite[x, FIRST, id[cart[domain[x], domain[x]]]], BINOPS],
          not[member[x, UNOPS]]] == True

In[10]:= or[member[composite[x_, FIRST, id[cart[domain[x_], domain[x_]]]], BINOPS],
           not[member[x_, UNOPS]]] := True

In[11]:= SubstTest[implies, member[u, v], member[pair[domain[u], range[u]], image[DORA, v]],
          u -> composite[x, FIRST, id[cart[domain[x], domain[x]]]]]

Out[11]= or[member[pair[cart[domain[x], domain[x]], range[x]], image[DORA, v]],
           not[member[composite[x, FIRST, id[cart[domain[x], domain[x]]]], v]]] == True

In[12]:= (% /. {x → x_, v → v_}) /. Equal → SetDelayed
```

```
In[13]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p2, p3], not[implies[p1, p3]], {p1 → member[x, UNOPS],
            p2 -> member[composite[x, FIRST, id[cart[domain[x], domain[x]]]], BINOPS],
            p3 → member[pair[cart[domain[x], domain[x]], range[x]], image[DORA, BINOPS]]}]]

Out[13]= or[member[pair[cart[domain[x], domain[x]], range[x]], image[DORA, BINOPS]],
           not[member[x, UNOPS]]] == True

In[14]:= or[member[pair[cart[domain[x_], domain[x_]], range[x_]], image[DORA, BINOPS]],
           not[member[x_, UNOPS]]] := True
```

---

main result

Lemma.

```
In[15]:= AssInt[P[cart[V, V]], FUNS, BINOPS]

Out[15]= intersection[BINOPS, P[cart[V, V]]] == BINOPS

In[16]:= intersection[BINOPS, P[cart[V, V]]] := BINOPS
```

Lemma.

```
In[17]:= AssInt[P[complement[cart[V, V]]], P[cart[V, V]], BINOPS]

Out[17]= intersection[BINOPS, P[complement[cart[V, V]]]] == set[0]

In[18]:= % /. Equal → SetDelayed
```

Lemma.

```
In[19]:= member[pair[0, 0], image[DORA, BINOPS]] // AssertTest

Out[19]= member[pair[0, 0], image[DORA, BINOPS]] == True

In[20]:= % /. Equal → SetDelayed
```

The variable **x** is eliminated from the result derived in the preceding section.

```
In[21]:= Map[equal[V, #] &, SubstTest[class, x,
            or[member[pair[cart[domain[x], domain[x]], range[x]], v], not[member[x, u]]],
            {u → UNOPS, v → image[DORA, BINOPS]}]] // Reverse

Out[21]= subclass[composite[id[complement[set[0]]], inverse[S]],
           composite[image[DORA, BINOPS], CART, DUP]] == True

In[22]:= % /. Equal → SetDelayed
```

The result is solved for **image[DORA,BINOPS]**.

```
In[23]:= SubstTest[implies, subclass[u, v], subclass[composite[u, w], composite[v, w]],
          {u -> composite[id[complement[set[0]]], inverse[S]],
           v -> composite[image[DORA, BINOPS], CART, DUP],
           w → composite[inverse[DUP], inverse[CART]]}]

Out[23]= subclass[composite[id[complement[set[0]]], inverse[S], inverse[DUP], inverse[CART]],
           image[DORA, BINOPS]] == True

In[24]:= % /. Equal → SetDelayed
```

Main result.

```
In[25]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> image[DORA, BINOPS], v -> union[cart[set[0], set[0]],
             composite[id[complement[set[0]]], inverse[S], inverse[DUP], inverse[CART]]]}]

Out[25]= True == equal[image[DORA, BINOPS], union[cart[set[0], set[0]],
            composite[id[complement[set[0]]], inverse[S], inverse[DUP], inverse[CART]]]]

In[26]:= image[DORA, BINOPS] := union[cart[set[0], set[0]],
           composite[id[complement[set[0]]], inverse[S], inverse[DUP], inverse[CART]]]
```

corollaries

Lemma.

```
In[32]:= SubstTest[image, CART, union[u, v],
           {u → id[complement[set[0]]], v → id[set[0]]}] // Reverse

Out[32]= union[image[CART, id[complement[set[0]]], set[0]] == image[CART, Id]

In[34]:= union[image[CART, id[complement[set[0]]], set[0]] := image[CART, Id]
```

Corollary.

```
In[35]:= ImageComp[FIRST, DORA, BINOPS]

Out[35]= image[IMAGE[FIRST], BINOPS] == image[CART, Id]

In[36]:= image[IMAGE[FIRST], BINOPS] := image[CART, Id]
```

Corollary.

```
In[39]:= ImageComp[SECOND, DORA, BINOPS]

Out[39]= image[IMAGE[SECOND], BINOPS] == V

In[40]:= image[IMAGE[SECOND], BINOPS] := V
```