inclusions for binclosed[x] and invar[x]

Johan G. F. Belinfante
2013 October 13

```
In[1]:= SetDirectory["l:"]; << goedel.13oct12a

:Package Title: goedel.13oct12a                 2013 October 12 at 7:20 p.m.

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

It is now:  2013 Oct 13 at 12:40

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 Oct 13 at 12:56
```

summary

Rewrite rules for inclusions involving **binclosed[x]** and **invar[x]** are derived.

introduction

If a class is invariant under both **x** and **y**, then it is also invariant under their composite **x ∘ y**. The following is an example.

Theorem.

```
In[5]:= SubstTest[subclass, intersection[invar[x], invar[y]],
          invar[composite[x, y]], {x → IMAGE[FIRST], y → IMAGE[SWAP]}] // Reverse

Out[5]= subclass[intersection[invar[IMAGE[FIRST]], invar[IMAGE[SWAP]]],
          invar[IMAGE[SECOND]]] == True

In[6]:= subclass[intersection[invar[IMAGE[FIRST]], invar[IMAGE[SWAP]]],
          invar[IMAGE[SECOND]]] := True
```

Similar results can be derived for classes of sets that are binary closed under given relations. In this notebook some basic rules are derived that can be used to show that sets that are binary closed or invariant under given relationare also binary closed under other relations that can be constructed from them.

invar[SINGLETON]

Binary closure under **x** implies invariance under **x ∘ DUP**. The following is an example.

Theorem. Binary closure under **PAIRSET** implies invariance under **SINGLETON**.

```
In[7]:= SubstTest[subclass, binclosed[x], invar[composite[x, DUP]], x → PAIRSET] // Reverse

Out[7]= subclass[binclosed[PAIRSET], invar[SINGLETON]] == True

In[8]:= subclass[binclosed[PAIRSET], invar[SINGLETON]] := True
```

binclosed[CUP]

If a class is invariant under **x** and binary closed under **y**, then it is binary closed under **x ∘ y**.

Theorem. If a set is invariant under **BIGCUP** and binary closed under **PAIRSET**, then it is binary closed under **CUP**.

```
In[9]:= SubstTest[subclass, intersection[invar[x], binclosed[y]],
          binclosed[composite[x, y]], {x → BIGCUP, y → PAIRSET}] // Reverse

Out[9]= subclass[intersection[binclosed[PAIRSET], invar[BIGCUP]], binclosed[CUP]] == True

In[10]:= subclass[intersection[binclosed[PAIRSET], invar[BIGCUP]], binclosed[CUP]] := True
```

binclosed is antitone

If **x ⊂ y**, then **binclosed[y] ⊂ binclosed[x]**.

Theorem.

```
In[11]:= SubstTest[implies, subclass[u, v],
           subclass[binclosed[v], binclosed[u]], {u → intersection[x, y], v → x}] // Reverse

Out[11]= subclass[binclosed[x], binclosed[intersection[x, y]]] == True

In[12]:= subclass[binclosed[x_], binclosed[intersection[x_, y_]]] := True
```

Theorem.

```
In[13]:= SubstTest[implies, subclass[u, v],
           subclass[binclosed[v], binclosed[u]], {u → composite[x, id[y]], v → x}] // Reverse

Out[13]= subclass[binclosed[x], binclosed[composite[x, id[y]]]] == True

In[14]:= subclass[binclosed[x_], binclosed[composite[x_, id[y_]]]] := True
```

binary closure under $x \circ (y \otimes z)$

Theorem.

```
In[15]:= Map[implies[and[subclass[image[y, s], u], subclass[image[z, t], v]], #] &, SubstTest[
           implies, and[subclass[p, q], subclass[image[x, q], w]], subclass[image[x, p], w],
           {p → cart[image[y, s], image[z, t]], q → cart[u, v]}]] // Reverse

Out[15]= or[not[subclass[image[x, cart[u, v]], w]],
           not[subclass[image[y, s], u]], not[subclass[image[z, t], v]],
           subclass[image[x, cart[image[y, s], image[z, t]]], w]] == True

In[16]:= or[not[subclass[image[x_, cart[u_, v_]], w_]],
           not[subclass[image[y_, s_], u_]], not[subclass[image[z_, t_], v_]],
           subclass[image[x_, cart[image[y_, s_], image[z_, t_]]], w_]] := True
```

Theorem. If a set is binary closed under **x** and invariant under **y** and **z**, then it is binary closed under **x ∘ (y ⊗ z)**.

```
In[17]:= Map[equal[V, domain[#]] &,
          SubstTest[reify, t, case[implies[member[t, u], member[t, v]]],
           {u → intersection[binclosed[x], invar[y], invar[z]],
            v → binclosed[composite[x, cross[y, z]]]}]]

Out[17]= subclass[intersection[binclosed[x], invar[y], invar[z]],
           binclosed[composite[x, cross[y, z]]]] == True

In[18]:= subclass[intersection[binclosed[x_], invar[y_], invar[z_]],
           binclosed[composite[x_, cross[y_, z_]]]] := True
```

Theorem.

```
In[28]:= SubstTest[subclass, intersection[binclosed[x], invar[y], invar[z]],
           binclosed[composite[x, cross[y, z]]],
           {x → CUP, y → SINGLETON, z → SINGLETON}] // Reverse

Out[28]= subclass[intersection[binclosed[CUP], invar[SINGLETON]], binclosed[PAIRSET]] == True

In[29]:= subclass[intersection[binclosed[CUP], invar[SINGLETON]], binclosed[PAIRSET]] := True
```

binary closure under $x \circ (y \otimes z) \circ DUP$

Lemma.

```
In[19]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
           {p1 -> and[subclass[u, v], subclass[v, w]], p2 → subclass[u, w],
            p3 → implies[subclass[image[x, w], t], subclass[image[x, u], t]]}]] // Reverse

Out[19]= or[not[subclass[u, v]], not[subclass[v, w]],
           not[subclass[image[x, w], t]], subclass[image[x, u], t]] == True

In[20]:= or[not[subclass[image[x_, w_], t_]], not[subclass[u_, v_]],
           not[subclass[v_, w_]], subclass[image[x_, u_], t_]] := True
```

Theorem.

```
In[21]:= Map[implies[and[subclass[image[x, cart[t, t]], t],
             subclass[image[y, cart[t, t]], t], subclass[image[z, cart[t, t]], t]], #] &,
           SubstTest[implies, and[subclass[u, v], subclass[v, w], subclass[image[x, w], t]],
            subclass[image[x, u], t], {u -> composite[z, id[cart[t, t]], inverse[y]],
             v -> cart[image[y, cart[t, t]], image[z, cart[t, t]]], w → cart[t, t]}]] // Reverse

Out[21]= or[not[subclass[image[x, cart[t, t]], t]],
           not[subclass[image[y, cart[t, t]], t]], not[subclass[image[z, cart[t, t]], t]],
           subclass[image[x, composite[z, id[cart[t, t]], inverse[y]]], t]] == True

In[22]:= (% /. {t → t_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Theorem. If a set is binary closed under **x**, **y** and **z**, then it is also binary closed under $\mathbf{x} \circ (\mathbf{y} \otimes \mathbf{z}) \circ \mathbf{DUP}$.

```
In[23]:= Map[equal[V, #] &, intersection[binclosed[x], binclosed[y], binclosed[z],
            complement[binclosed[composite[x, intersection[composite[inverse[FIRST], y],
                composite[inverse[SECOND], z]]]]]] // complement // Normality]

Out[23]= subclass[
           intersection[binclosed[x], binclosed[y], binclosed[z]], binclosed[composite[x,
            intersection[composite[inverse[FIRST], y], composite[inverse[SECOND], z]]]]] == True

In[24]:= subclass[intersection[binclosed[x_], binclosed[y_], binclosed[z_]],
           binclosed[composite[x_, intersection[
              composite[inverse[FIRST], y_], composite[inverse[SECOND], z_]]]]] := True
```

Theorem. (An application to **SYMDIF**.)

```
In[25]:= SubstTest[subclass, intersection[binclosed[x], binclosed[y], binclosed[z]],
           binclosed[composite[x, intersection[composite[inverse[FIRST], y],
              composite[inverse[SECOND], z]]]], {x → CUP, y → DIF, z → flip[DIF]}] // Reverse

Out[25]= subclass[intersection[binclosed[CUP], binclosed[DIF]], binclosed[SYMDIF]] == True

In[26]:= subclass[intersection[binclosed[CUP], binclosed[DIF]], binclosed[SYMDIF]] := True
```