monotone operations

Johan G. F. Belinfante
2010 August 2

```
In[1]:= SetDirectory["l:"]; << goedel.10jul29a; << tools.m

:Package Title: goedel.10jul29a              2010 July 29 at 1:45 p.m.

It is now:  2010 Aug 2 at 13:32

Loading Simplification Rules

TOOLS.M                                      Revised 2010 July 26

weightlimit = 40
```

summary

A monotone operation for a relation **x** is a function **t** satisfying **t ∘ x ∘ inverse[t] ⊂ x**. It is shown in this notebook that if **t** is a monotone operation for **x**, and if **z ⊂ fix[t]**, then the class **ub[x, z]** of upper bounds for **z** is invariant under **t**. A similar result holds for the class **lb[x, z]** of lower bounds. A remarkable feature of the derivation presented in this notebook is the use of the class **transvar[x, y]** about which two interesting new results are derived. Recall that the class **transvar[x, y]** holds all sets **t** satisfying **image[x, t] ⊂ image[y, t]**. One of the new results is that if **P[t] ⊂ transvar[x, y]**, then **image[x, t] ⊂ image[y, t]**. This result is of course trivial when **t** is a set, because in that case **t ∈ P[t]**. What is interesting however is that the same result also holds for any class **t**. The second new result is the derivation of a general lower bound for **transvar[x, y]** that generalizes the following familiar lower bound for **subvar[x]**.

```
In[2]:= subclass[P[fix[x]], subvar[x]]

Out[2]= True
```

a lower bound for transvar[x, y]

Lemma. Needed to cope with **complement**.

```
In[3]:= SubstTest[Uclosure, image[SINGLETON, t], t → complement[x]] // Reverse

Out[3]= Uclosure[intersection[complement[image[SINGLETON, x]], range[SINGLETON]]] ==
         P[complement[x]]

In[4]:= Uclosure[intersection[complement[image[SINGLETON, x_]], range[SINGLETON]]] :=
         P[complement[x]]
```

Lemma.

```
In[5]:= Map[subclass[#, transvar[x, y]] &,
          ImageComp[SINGLETON, inverse[SINGLETON], transvar[x, y]]] // Reverse

Out[5]= subclass[range[SINGLETON], union[
           image[SINGLETON, fix[composite[complement[inverse[y]], x]]], transvar[x, y]]] == True

In[6]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem. A lower bound for **transvar[x, y]**.

```
In[7]:= SubstTest[subclass, Uclosure[u], Uclosure[v],
          {u → image[SINGLETON, complement[fix[composite[complement[inverse[y]], x]]]],
           v → transvar[x, y]}] // Reverse

Out[7]= subclass[P[complement[fix[composite[complement[inverse[y]], x]]]], transvar[x, y]] ==
         True

In[8]:= subclass[P[complement[fix[composite[complement[inverse[y_]], x_]]]],
          transvar[x_, y_]] := True
```

Corollary. A lower bound for **invar[x]**.

```
In[9]:= SubstTest[subclass, P[complement[fix[composite[complement[inverse[y]], x]]]],
          transvar[x, y], y → Id] // Reverse

Out[9]= subclass[P[complement[fix[composite[Di, x]]]], invar[x]] == True

In[10]:= subclass[P[complement[fix[composite[Di, x_]]]], invar[x_]] := True
```

---

an example

The new lower bound for **invar[x]** is illustrated by an example in this section. The results obtained here are not used in later sections of this notebook.

Lemma.

```
In[11]:= SubstTest[subclass, P[complement[fix[composite[Di, x]]]],
           invar[x], x → inverse[ROT]] // Reverse

Out[11]= subclass[P[union[complement[cart[cart[V, V], V]], inverse[DUP]]], invar[ROT]] == True

In[12]:= % /. Equal → SetDelayed
```

Theorem.

```
In[13]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]], subclass[u, w],
           {u → P[inverse[DUP]], v → P[union[complement[cart[cart[V, V], V]], inverse[DUP]]],
            w -> invar[ROT]}] // Reverse

Out[13]= subclass[P[inverse[DUP]], invar[ROT]] == True
```

```
In[14]:= subclass[P[inverse[DUP]], invar[ROT]] := True
```

P[z] ⊂ transvar[x, y]

In this section it is shown that if **P[z] ⊂ transvar[x, y]**, then **image[x, z] ⊂ image[y, z]**.

Lemma.

```
In[15]:= SubstTest[implies, and[member[u, v], subclass[v, w]],
           member[u, w], {v → P[z], w → transvar[x, y]}] // Reverse

Out[15]= or[not[member[u, V]], not[subclass[u, z]],
           not[subclass[P[z], transvar[x, y]]], subclass[image[x, u], image[y, u]]] == True

In[16]:= (% /. {u → u_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Corollary. (Replacing **image[y, u]** with **image[y, z]**.)

```
In[17]:= Map[not, SubstTest[and, implies[and[p1, p2], p3],
            implies[and[p1, p3], p4], not[implies[and[p1, p2], p4]], {p1 → member[u, P[z]],
             p2 → subclass[P[z], transvar[x, y]], p3 -> subclass[image[x, u], image[y, u]],
             p4 → subclass[image[x, u], image[y, z]]}]] // Reverse

Out[17]= or[not[member[u, V]], not[subclass[u, z]],
           not[subclass[P[z], transvar[x, y]]], subclass[image[x, u], image[y, z]]] == True

In[18]:= (% /. {u → u_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

The new result is obtained by eliminating the variable **u**.

Theorem. If **P[z] ⊂ transvar[x, y]**, then **transvariant[x, y, z]**.

```
In[19]:= Map[equal[V, #] &, SubstTest[class, u, or[not[member[u, t]],
            not[subclass[P[z], transvar[x, y]]], subclass[image[x, u], image[y, z]]], t → P[z]]]

Out[19]= or[not[subclass[P[z], transvar[x, y]]], subclass[image[x, z], image[y, z]]] == True

In[20]:= or[not[subclass[P[z_], transvar[x_, y_]]],
           subclass[image[x_, z_], image[y_, z_]]] := True
```

Restatement.

```
In[24]:= implies[subclass[P[z], transvar[x, y]], transvariant[x, y, z]]

Out[24]= True
```

Corollary. The special case of the class **invar[x]** of invariant sets.

```
In[23]:= SubstTest[implies, subclass[P[y], transvar[x, t]],
           transvariant[x, t, y], t → Id] // Reverse

Out[23]= or[not[subclass[P[y], invar[x]]], subclass[image[x, y], y]] == True

In[25]:= or[not[subclass[P[y_], invar[x_]]], subclass[image[x_, y_], y_]] := True
```

application to monotone operations

Lemma.

```
In[34]:= SubstTest[subclass,
           P[complement[fix[composite[complement[inverse[v]], u]]]], transvar[u, v],
           {u → composite[inverse[funpart[t]], complement[y]], v → complement[x]}] // Reverse

Out[34]= subclass[
           P[complement[fix[composite[inverse[x], inverse[funpart[t]], complement[y]]]]],
           transvar[composite[inverse[funpart[t]], complement[y]], complement[x]]] == True

In[35]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[36]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]],
           subclass[u, w], {u -> P[fix[funpart[t]]], v ->
             P[complement[fix[composite[inverse[x], inverse[funpart[t]], complement[y]]]]], w ->
             transvar[composite[inverse[funpart[t]], complement[y]], complement[x]]}] // Reverse

Out[36]= or[not[subclass[composite[funpart[t], x, id[fix[funpart[t]]]], y]],
           subclass[P[fix[funpart[t]]],
            transvar[composite[inverse[funpart[t]], complement[y]], complement[x]]]] == True

In[37]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem.

```
In[38]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
            {p1 → subclass[composite[funpart[t], x, inverse[funpart[t]]], y],
             p2 -> subclass[composite[funpart[t], x, id[fix[funpart[t]]]], y],
             p3 -> subclass[P[fix[funpart[t]]], transvar[
                composite[inverse[funpart[t]], complement[y]], complement[x]]]}]] // Reverse

Out[38]= or[not[subclass[composite[funpart[t], x, inverse[funpart[t]]], y]],
           subclass[P[fix[funpart[t]]],
            transvar[composite[inverse[funpart[t]], complement[y]], complement[x]]]] == True

In[39]:= or[not[subclass[composite[funpart[t_], x_, inverse[funpart[t_]]], y_]],
           subclass[P[fix[funpart[t_]]],
            transvar[composite[inverse[funpart[t_]], complement[y_]], complement[x_]]]] := True
```

The result obtained becomes clearer if a new variable is introduced.

Lemma.

```
In[40]:= SubstTest[implies, subclass[P[w], transvar[u, v]], subclass[image[u, w], image[v, w]],
           {u -> composite[inverse[funpart[t]], complement[y]], v -> complement[x]}] // Reverse

Out[40]= or[not[subclass[P[w],
             transvar[composite[inverse[funpart[t]], complement[y]], complement[x]]]],
           subclass[cart[w, image[funpart[t], ub[x, w]]], y]] == True

In[41]:= (% /. {t → t_, x → x_, y → y_, w → w_}) /. Equal → SetDelayed
```

Theorem. A theorem about invariance of upper bounds of classes of fixed points for monotone operations.

```
In[42]:= Map[not, SubstTest[and, implies[and[p0, p2], p3], implies[p1, p2], implies[p3, p4],
            not[implies[and[p0, p1], p4]], {p0 → subclass[z, fix[funpart[t]]],
             p1 -> subclass[composite[funpart[t], x, inverse[funpart[t]]], y],
             p2 -> subclass[P[fix[funpart[t]]],
               transvar[composite[inverse[funpart[t]], complement[y]], complement[x]]],
             p3 -> subclass[P[z], transvar[composite[inverse[funpart[t]],
                 complement[y]], complement[x]]],
             p4 -> subclass[cart[z, image[funpart[t], ub[x, z]]], y]}]] // Reverse

Out[42]= or[not[subclass[z, fix[funpart[t]]]],
           not[subclass[composite[funpart[t], x, inverse[funpart[t]]], y]],
           subclass[cart[z, image[funpart[t], ub[x, z]]], y]] == True

In[43]:= or[not[subclass[composite[funpart[t_], x_, inverse[funpart[t_]]], y_]],
           not[subclass[z_, fix[funpart[t_]]]],
           subclass[cart[z_, image[funpart[t_], ub[x_, z_]]], y_]] := True
```

Restatement: If **t** is a monotone operation for **x**, and if **z ⊂ fix[t]**, then **ub[x, z]** is invariant under **t**.

```
In[44]:= implies[and[subclass[z, fix[funpart[t]]],
           subclass[composite[funpart[t], x, inverse[funpart[t]]], x]],
          invariant[funpart[t], ub[x, z]]]

Out[44]= True
```

Replacing **x** and **y** with their inverses yields an analogous result for lower bounds.

Corollary.

```
In[49]:= SubstTest[implies, and[subclass[z, fix[funpart[t]]],
            subclass[composite[funpart[t], u, inverse[funpart[t]]], v]],
           conduct[funpart[t], ub[u, z], ub[v, z]], {u → inverse[x], v → inverse[y]}] // Reverse

Out[49]= or[not[subclass[z, fix[funpart[t]]]],
           not[subclass[composite[funpart[t], x, inverse[funpart[t]]], y]],
           subclass[cart[image[funpart[t], lb[x, z]], z], y]] == True

In[50]:= or[not[subclass[z_, fix[funpart[t_]]]],
           not[subclass[composite[funpart[t_], x_, inverse[funpart[t_]]], y_]],
           subclass[cart[image[funpart[t_], lb[x_, z_]], z_], y_]] := True
```

Restatement.

```
In[51]:= implies[and[subclass[z, fix[funpart[t]]],
           subclass[composite[funpart[t], x, inverse[funpart[t]]], x]],
          invariant[funpart[t], lb[x, z]]]

Out[51]= True
```

serendipity

The following result was discovered in the course of deriving the new lower bound for **transvar[x, y]**.

Theorem.

```
In[52]:= fix[composite[inverse[VERTSECT[x]], S, VERTSECT[y]]] // Normality

Out[52]= fix[composite[inverse[VERTSECT[x]], S, VERTSECT[y]]] == intersection[
           complement[fix[composite[complement[inverse[x]], y]]], domain[VERTSECT[x]]]

In[53]:= fix[composite[inverse[VERTSECT[x_]], S, VERTSECT[y_]]] := intersection[
           complement[fix[composite[complement[inverse[x]], y]]], domain[VERTSECT[x]]]
```