review of theorems in the Otter ID groups

Johan G. F. Belinfante and Tiffany D. Goble
2004 April 20

```
In[1]:= << goedel56.18d ; << tools.m;

        :Package Title: goedel56.18d          2004 April 18 at 10:00 p.m.

        It is now:  2004 Apr 21 at 9:2

        Loading Simplification Rules

        TOOLS.M                       Revised 2004 April 18

        weightlimit = 40
```

summary

Some theorems in the **ID** groups proved using **Otter** are reviewed in this notebook, especially those that the **GOEDEL** program currently does not immediately recognize as true. Frequently the **GOEDEL** program rewrites the theorems that were proved with **Otter**, so what is proved here may be transformed versions of the original theorems, or a generalization of them. The goal is not so much to reproduce the **Otter** results, but rather to increase the power of the **GOEDEL** program by filling in gaps. Some of the results obtained in the **Otter** work are merely lemmas that lack sufficient interest to warrant being kept as permanent rules, but interesting facts are sometimes discovered in the course of the work.

a theorem in the ID−CO group

The only result in the **ID−CO** group that is not immediately recognized by the **GOEDEL** program is Theorem **ID−CO−SU**. In this section a rewrite rule is derived which implies Theorem **ID−CO−SU**. First a lemma:

```
In[2]:= SubstTest[implies, subclass[u, v],
          subclass[composite[w, z, u], composite[w, z, v]], {u -> Id, v -> composite[x, y]}]

Out[2]= or[not[equal[V, fix[composite[x, y]]]],
          subclass[composite[w, z], composite[w, z, x, y]]] == True

In[3]:= (% /. {w -> w_, x -> x_, y -> y_, z -> z_}) /. Equal -> SetDelayed
```

Theorem **ID−CO−SU** in the **ID−CO** group amounts to this result:

```
In[4]:= Map[not, SubstTest[and, implies[p1, p4], implies[p2, p5],
          implies[p3, p6], implies[and[p4, p5], p7], implies[and[p6, p7], p8],
          not[implies[and[p1, p2, p3], p8]], {p1 -> equal[V, fix[composite[x, y]]],
           p2 -> subclass[composite[z, x], composite[u, v]],
                 p3 -> subclass[composite[w, u], Id],
                 p4 -> subclass[composite[w, z], composite[w, z, x, y]],
           p5 -> subclass[composite[w, z, x, y], composite[w, u, v, y]],
                 p6 -> subclass[composite[w, u, v, y], composite[v, y]],
           p7 -> subclass[composite[w, z], composite[w, u, v, y]],
                 p8 -> subclass[composite[w, z], composite[v, y]]}]]

Out[4]= or[not[equal[V, fix[composite[x, y]]]],
          not[subclass[composite[w, u], Id]], not[subclass[composite[z, x], composite[u, v]]],
          subclass[composite[w, z], composite[v, y]]] == True

In[5]:= or[not[equal[V, fix[composite[x_, y_]]]], not[subclass[composite[w_, u_], Id]],
          not[subclass[composite[z_, x_], composite[u_, v_]]],
          subclass[composite[w_, z_], composite[v_, y_]]] := True
```

Since there are 6 free variables in this result, one can derive numerous corollaries by specializing these. One cannot remove all 6 variables by specializing them to sets because the condition **equal[V,fix[composite[x,y]]]** can not be satisfied when **x** and **y** are sets. Some attempts were made to remove some of the other variables, but nothing of interest was obtained.

theorems in the ID2 group

There are 4 theorems in the **ID2** group not currently recognized by the **GOEDEL** program. The first of these is Lemma **CO−4−LEM** which was used in **Otter**'s proof of Theorem **CO−4**. Since the **GOEDEL** program recognizes the theorem, the easiest thing here is to deduce the lemma from the theorem! But first one needs another lemma:

```
In[6]:= SubstTest[implies, and[member[u, v], subclass[v, w]], member[u, w],
         {u -> pair[x, x], v -> id[domain[z]], w -> composite[inverse[z], z]}]

Out[6]= or[member[pair[x, x], composite[inverse[z], z]], not[member[x, domain[z]]]] == True

In[7]:= (% /. {x -> x_, z -> z_}) /. Equal -> SetDelayed
```

Theorem **CO−4−LEM** in the **ID−2** group follows:

```
In[8]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
              {p1 -> member[pair[x, y], composite[Id, z]],
           p2 -> member[x, domain[z]], p3 -> member[pair[x, x], composite[inverse[z], z]]}]]

Out[8]= or[member[pair[x, x], composite[inverse[z], z]],
          not[member[x, V]], not[member[y, V]], not[member[pair[x, y], z]]] == True

In[9]:= (% /. {x -> x_, y -> y_, z -> z_}) /. Equal -> SetDelayed
```

The second item on the agenda is Theorem **ID−5D**. For this theorem it is convenient to derive the following lemma, which is actually better than the theorem and is worth saving.

```
In[10]:= SubstTest[subclass, x, intersection[x, y], y -> Id]

Out[10]= subclass[x, id[fix[x]]] == subclass[x, Id]

In[11]:= subclass[x_, id[fix[x_]]] := subclass[x, Id]
```

Theorem **ID−5D** from the **ID2** group follows from this:

```
In[12]:= SubstTest[implies, and[subclass[x, y], subclass[y, z]],
          subclass[x, z], {y -> id[fix[x]], z -> id[domain[x]]}]

Out[12]= or[not[subclass[x, Id]], subclass[x, id[domain[x]]]] == True

In[13]:= (% /. x -> x_) /. Equal -> SetDelayed

In[14]:= equiv[subclass[x, Id], subclass[x, id[domain[x]]]]

Out[14]= True

In[15]:= subclass[x_, id[domain[x_]]] := subclass[x, Id]
```

The following result is analogous:

```
In[16]:= SubstTest[implies, and[subclass[x, y], subclass[y, z]],
          subclass[x, z], {y -> id[fix[x]], z -> id[range[x]]}]

Out[16]= or[not[subclass[x, Id]], subclass[x, id[range[x]]]] == True

In[17]:= (% /. x -> x_) /. Equal -> SetDelayed

In[18]:= equiv[subclass[x, id[range[x]]], subclass[x, Id]]

Out[18]= True

In[19]:= subclass[x_, id[range[x_]]] := subclass[x, Id]
```

Next we turn to Theorem **ID−5F**., for which two lemmas will be used. The first is:

```
In[20]:= SubstTest[implies, equal[u, v], equal[domain[u], domain[v]],
          {u -> intersection[x, y], v -> id[intersection[domain[x], domain[y]]]}]

Out[20]= or[equal[fix[composite[inverse[x], y]], intersection[domain[x], domain[y]]],
           not[equal[id[intersection[domain[x], domain[y]]], intersection[x, y]]]] == True

In[21]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

The second lemma is:

```
In[22]:= SubstTest[implies, and[equal[u, x], equal[v, y]],
          equal[intersection[u, v], intersection[x, y]],
             {u -> id[domain[x]], v -> id[domain[y]]}]

Out[22]= or[equal[id[intersection[domain[x], domain[y]]], intersection[x, y]],
           not[subclass[x, Id]], not[subclass[y, Id]]] == True

In[23]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

Theorem **ID−5F** in the **ID−2** group now follows:

```
In[24]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3],
           not[implies[p1, p3]], {p1 -> and[subclass[x, Id], subclass[y, Id]],
          p2 -> equal[id[intersection[domain[x], domain[y]]], intersection[x, y]],
             p3 -> equal[fix[composite[inverse[x], y]], intersection[domain[x], domain[y]]]}]]

Out[24]= or[equal[fix[composite[inverse[x], y]], intersection[domain[x], domain[y]]],
           not[subclass[x, Id]], not[subclass[y, Id]]] == True
```

```
In[25]:= or[equal[fix[composite[inverse[x_], y_]], intersection[domain[x_], domain[y_]]],
           not[subclass[x_, Id]], not[subclass[y_, Id]]] := True
```

The final result in the **ID2** group that will be reviewed is Theorem **ID−5G**. Several lemmas are needed:

```
In[26]:= SubstTest[implies, and[subclass[u, v], equal[u, x]],
          subclass[x, v], {u → id[domain[x]], v → id[domain[y]]}]

Out[26]= or[not[subclass[x, Id]],
           not[subclass[domain[x], domain[y]]], subclass[x, id[domain[y]]]] == True

In[27]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

Lemma:

```
In[28]:= SubstTest[implies, and[subclass[x, v], equal[v, y]], subclass[x, y], v → id[domain[y]]]

Out[28]= or[not[subclass[x, id[domain[y]]]], not[subclass[y, Id]], subclass[x, y]] == True

In[29]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

Theorem **ID−5G** in the **ID−2** group.

```
In[30]:= Map[not, SubstTest[and, implies[and[p1, p3], p4], implies[and[p2, p4], p5],
           not[implies[and[p1, p2, p3], p5]], {p1 → subclass[x, Id],
          p2 → subclass[y, Id], p3 → subclass[domain[x], domain[y]],
             p4 → subclass[x, id[domain[y]]], p5 → subclass[x, y]}]]

Out[30]= or[not[subclass[x, Id]], not[subclass[y, Id]],
           not[subclass[domain[x], domain[y]]], subclass[x, y]] == True

In[31]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

Theorem **ID−5G** has a converse which requires no formal proof, and one can combine the theorem with this converse:

```
In[32]:= equiv[and[subclass[x, Id], subclass[y, Id], subclass[domain[x], domain[y]]],
            and[subclass[x, y], subclass[y, Id]]] // not // not

Out[32]= True
```

This justifies the following rule:

```
In[33]:= and[subclass[x_, Id], subclass[y_, Id], subclass[domain[x_], domain[y_]]] :=
          and[subclass[x, y], subclass[y, Id]]
```

The negative version of this is:

```
In[34]:= or[not[subclass[x, Id]], not[subclass[y, Id]],
           not[subclass[domain[x], domain[y]]]] // NotNotTest

Out[34]= or[not[subclass[x, Id]], not[subclass[y, Id]], not[subclass[domain[x], domain[y]]]] ==
          or[not[subclass[x, y]], not[subclass[y, Id]]]

In[35]:= or[not[subclass[x_, Id]], not[subclass[y_, Id]],
           not[subclass[domain[x_], domain[y_]]]] :=
          or[not[subclass[x, y]], not[subclass[y, Id]]]
```

theorems from the ID3 group

Four results from the **ID3** group will be addressed in this section. The first, Theorem **ID−5H**, is an immediate corollary of **ID−5G**.

```
In[36]:= Map[not, SubstTest[and, implies[p2, and[p3, p4]],
           implies[and[p1, p3], p5], implies[and[p1, p4], p6], implies[and[p5, p6], p7],
           not[implies[and[p1, p2], p7]], {p1 → and[subclass[x, Id], subclass[y, Id]],
         p2 → equal[domain[x], domain[y]], p3 → subclass[domain[x], domain[y]],
            p4 → subclass[domain[y], domain[x]], p5 → subclass[x, y],
            p6 → subclass[y, x], p7 → equal[x, y]}]]

Out[36]= or[equal[x, y], not[equal[domain[x], domain[y]]],
           not[subclass[x, Id]], not[subclass[y, Id]]] == True

In[37]:= or[equal[x_, y_], not[equal[domain[x_], domain[y_]]],
           not[subclass[x_, Id]], not[subclass[y_, Id]]] := True
```

The key to deriving Theorem **ID−IM−SS** is the following formula:

```
In[38]:= class[z, subclass[composite[u, z], v]]

Out[38]= P[complement[composite[inverse[u], complement[v]]]]
```

Using this, one derives the following result, which is actually better than Theorem **ID−IM−SS** in the **ID3** group in that the hypothesis that **y** be a set has been removed.

```
In[39]:= Map[implies[member[pair[x, y], z], #] &,
          SubstTest[implies, and[subclass[r, s], subclass[s, t]],
           subclass[r, t], {r -> singleton[PAIR[x, y]], s -> z,
            t -> complement[composite[inverse[u], complement[v]]]}]]

Out[39]= or[not[member[x, V]], not[member[pair[x, y], z]], not[subclass[composite[u, z], v]],
           subclass[image[u, singleton[y]], image[v, singleton[x]]]] == True

In[40]:= or[not[member[pair[x_, y_], z_]],
           not[member[x_, V]], not[subclass[composite[u_, z_], v_]],
           subclass[image[u_, singleton[y_]], image[v_, singleton[x_]]]] := True
```

Theorem **ID−CO−DO** is an immediate consequence of the following more general result:

```
In[41]:= SubstTest[implies, equal[x, z],
          equal[image[inverse[x], y], image[inverse[z], y]], z -> id[domain[x]]]

Out[41]= or[equal[image[inverse[x], y], intersection[y, domain[x]]],
           not[subclass[x, Id]]] == True

In[42]:= or[equal[image[inverse[x_], y_], intersection[y_, domain[x_]]],
           not[subclass[x_, Id]]] := True
```

We verify that Theorem **ID−CO−DO** is a consequence of this:

```
In[43]:= or[not[subclass[x, Id]], not[subclass[y, Id]],
          equal[domain[composite[x, y]], intersection[domain[x], domain[y]]]]

Out[43]= True
```

A better result can be obtained by replacing **domain** with **fix** :

```
In[44]:= SubstTest[implies, equal[x, z],
          equal[image[inverse[x], y], image[inverse[z], y]], z -> id[fix[x]]]

Out[44]= or[equal[image[inverse[x], y], intersection[y, fix[x]]], not[subclass[x, Id]]] == True

In[45]:= or[equal[image[inverse[x_], y_], intersection[y_, fix[x_]]],
           not[subclass[x_, Id]]] := True
```

One can also derive a similar result for direct images:

```
In[46]:= SubstTest[implies, equal[x, z], equal[image[x, y], image[z, y]], z -> id[fix[x]]]

Out[46]= or[equal[image[x, y], intersection[y, fix[x]]], not[subclass[x, Id]]] == True

In[47]:= or[equal[image[x_, y_], intersection[y_, fix[x_]]], not[subclass[x_, Id]]] := True
```

The final task in this section is to derive Theorem **ID−CO−I**. These lemmas will be used:

```
In[48]:= SubstTest[implies, and[equal[x, u], equal[y, v]],
          equal[composite[x, y], composite[u, v]], {u -> id[fix[x]], v -> id[fix[y]]}]

Out[48]= or[equal[composite[x, y], id[intersection[fix[x], fix[y]]]],
           not[subclass[x, Id]], not[subclass[y, Id]]] == True

In[49]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed

In[50]:= SubstTest[implies, and[equal[x, u], equal[y, v]],
          equal[intersection[x, y], intersection[u, v]], {u -> id[fix[x]], v -> id[fix[y]]}]

Out[50]= or[equal[id[intersection[fix[x], fix[y]]], intersection[x, y]],
           not[subclass[x, Id]], not[subclass[y, Id]]] == True

In[51]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

Theorem **ID−CO−DO** follows:

```
In[52]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p1, p3], implies[and[p2, p3], p4], not[implies[p1, p4]],
                 {p1 -> subclass[union[x, y], Id],
            p2 -> equal[id[intersection[fix[x], fix[y]]], intersection[x, y]],
                      p3 -> equal[id[intersection[fix[x], fix[y]]], composite[x, y]],
            p4 -> equal[composite[x, y], intersection[x, y]]}]]

Out[52]= or[equal[composite[x, y], intersection[x, y]],
           not[subclass[x, Id]], not[subclass[y, Id]]] == True

In[53]:= or[equal[composite[x_, y_], intersection[x_, y_]],
           not[subclass[x_, Id]], not[subclass[y_, Id]]] := True
```

To derive a good variable−free formulation of this result requires somewhat greater effort than one might expect. First turn off some flags.

```
In[54]:= simplify = False; cond = False;
```

Next a membership rule is derived. This step takes a surprisingly long time even with the flags cleared:

```
In[55]:= member[pair[x, y], fix[composite[inverse[CAP], COMPOSE]]] // AssertTest

Out[55]= member[pair[x, y], fix[composite[inverse[CAP], COMPOSE]]] ==
          and[equal[composite[x, y], intersection[x, y]], member[x, V], member[y, V]]
```

```
In[56]:= member[pair[x_, y_], fix[composite[inverse[CAP], COMPOSE]]] :=
          and[equal[composite[x, y], intersection[x, y]], member[x, V], member[y, V]]
```

A negative version of this is needed:

```
In[57]:= member[pair[x, y],
            dif[cart[P[Id], P[Id]], fix[composite[inverse[CAP], COMPOSE]]]] // NotNotTest

Out[57]= and[member[x, V], member[y, V], not[equal[composite[x, y], intersection[x, y]]],
            subclass[x, Id], subclass[y, Id]] == False

In[58]:= (% /. {x -> x_, y -> y_}) /. Equal -> SetDelayed
```

The final step is easy:

```
In[59]:= Map[equal[0, #] &, SubstTest[class, pair[x, y], member[pair[x, y], z],
             z -> dif[cart[P[Id], P[Id]], fix[composite[inverse[CAP], COMPOSE]]]]] // Reverse

Out[59]= subclass[cart[P[Id], P[Id]], fix[composite[inverse[CAP], COMPOSE]]] == True

In[60]:= subclass[cart[P[Id], P[Id]], fix[composite[inverse[CAP], COMPOSE]]] := True
```

theorems from the ID4 group

Four results from the **ID4** group will be discussed in this section. The first three of these are Theorems **ID−9C**, **ID−9F** and **ID−9G**. Each of these readily succumbs to double negation.

```
In[61]:= or[and[not[equal[0, y]], not[member[y, range[SINGLETON]]]],
            equal[x, z], not[member[x, y]], not[member[z, y]]] // NotNotTest

Out[61]= or[and[not[equal[0, y]], not[member[y, range[SINGLETON]]]],
            equal[x, z], not[member[x, y]], not[member[z, y]]] == True

In[62]:= or[and[not[equal[0, x]], not[member[x, range[SINGLETON]]]], member[x, V]] //
           NotNotTest

Out[62]= or[and[not[equal[0, x]], not[member[x, range[SINGLETON]]]], member[x, V]] == True

In[63]:= or[and[not[equal[0, x]], not[equal[0, y]], not[equal[x, y]]],
            and[not[equal[0, x]], not[equal[0, y]], not[member[x, range[SINGLETON]]]],
            member[x, V], member[y, V]] // NotNotTest

Out[63]= or[and[not[equal[0, x]], not[equal[0, y]], not[equal[x, y]]],
            and[not[equal[0, x]], not[equal[0, y]], not[member[x, range[SINGLETON]]]],
            member[x, V], member[y, V]] == True
```

The final result is Theorem **ID−10**, which follows directly from Theorem **SC−4B** of the **SC1** group. The derivation of the latter requires a little bit of reasoning:

```
In[64]:= Map[not, SubstTest[and, implies[p1, or[p2, p3]],
            implies[p2, or[p3, p4]], not[implies[p1, or[p3, p4]]],
              {p1 -> subclass[x, singleton[y]], p2 -> equal[x, singleton[y]],
               p3 -> equal[0, x], p4 -> member[x, range[SINGLETON]]}]]

Out[64]= or[equal[0, x], member[x, range[SINGLETON]], not[subclass[x, singleton[y]]]] == True

In[65]:= or[equal[0, x_], member[x_, range[SINGLETON]], not[subclass[x_, singleton[y_]]]] := True
```