cliques of similar relations

Johan G. F. Belinfante
2013 November 10

```
In[1]:= SetDirectory["l:"]; << goedel.13nov07a

        :Package Title: goedel.13nov07a                2013 November 7 at 12:40 a.m.

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

        It is now:  2013 Nov 10 at 20:14

        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 Nov 10 at 20:31
```

summary

The cliques of similar relations are equipollent.

an equipollence theorem

Lemma. Application of an equipollence criterion to **cliques[x]**.

```
In[2]:= Map[implies[member[x, y], #] &,
          SubstTest[or, member[pair[y, image[IMAGE[oopart[t]], y]], Q], not[member[y, V]],
           not[subclass[U[y], domain[oopart[t]]]], y → cliques[x]]] // Reverse // MapNotNot

Out[2]= or[member[pair[cliques[x], image[IMAGE[oopart[t]], cliques[x]]], Q],
          not[member[x, y]], not[subclass[fix[x], domain[oopart[t]]]]] == True

In[3]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem.

```
In[4]:= Map[not, SubstTest[and, implies[and[p0, p2], p3],
          implies[and[p0, p1], p2], not[implies[and[p0, p1], p3]],
          {p0 → member[x, y], p1 → subclass[x, cartsq[domain[oopart[t]]]],
           p2 -> subclass[fix[x], domain[oopart[t]]],
           p3 -> member[pair[cliques[x], image[IMAGE[oopart[t]], cliques[x]]], Q]}]] // Reverse

Out[4]= or[member[pair[cliques[x], image[IMAGE[oopart[t]], cliques[x]]], Q], not[member[x, y]],
          not[subclass[x, cart[domain[oopart[t]], domain[oopart[t]]]]]] == True

In[5]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

The next step is to transfer sethood condition from **x** to **t**.

Lemma.

```
In[7]:= Map[not, SubstTest[and, implies[p1, p3], implies[and[p2, p3], p4],
          not[implies[and[p1, p2], p4]], {p1 → member[t, V], p2 → subclass[x, cartsq[domain[t]]],
           p3 -> member[domain[t], V], p4 → member[x, V]}]] // Reverse

Out[7]= or[member[x, V], not[member[t, V]], not[subclass[x, cart[domain[t], domain[t]]]]] == True

In[8]:= (% /. {x → x_, t → t_}) /. Equal → SetDelayed
```

Lemma. Transfer sethood condition from **x** to **t**.

```
In[11]:= Map[not, SubstTest[and, implies[and[p0, p1, p2], p3], implies[and[p0, p1], p2],
           not[implies[and[p0, p1], p3]], {p0 → member[oopart[t], V],
            p1 → subclass[x, cart[domain[oopart[t]], domain[oopart[t]]]], p2 → member[x, V],
            p3 → member[pair[cliques[x], image[IMAGE[oopart[t]], cliques[x]]], Q]}]] // Reverse

Out[11]= or[member[pair[cliques[x], image[IMAGE[oopart[t]], cliques[x]]], Q],
           not[member[oopart[t], V]],
           not[subclass[x, cart[domain[oopart[t]], domain[oopart[t]]]]]] == True

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

relating cliques of similar relations

In this section a simple formula relates **cliques[x]** to **cliques[y]** when **y** is related to **x** by a similarity transformation.

Lemma.

```
In[13]:= SubstTest[cliques,
           composite[inverse[funpart[w]], x, funpart[w]], w → oopart[t]] // Reverse

Out[13]= cliques[composite[inverse[oopart[t]], x, oopart[t]]] ==
          intersection[image[inverse[IMAGE[oopart[t]]], cliques[x]], P[domain[oopart[t]]]]

In[14]:= cliques[composite[inverse[oopart[t_]], x_, oopart[t_]]] :=
          intersection[image[inverse[IMAGE[oopart[t]]], cliques[x]], P[domain[oopart[t]]]]
```

The inverse image can be replaced with a direct image.

Theorem.

```
In[15]:= Map[image[IMAGE[oopart[t]], #] &,
          SubstTest[cliques, composite[inverse[oopart[t]], y, oopart[t]],
           y → composite[oopart[t], x, inverse[oopart[t]]]]]

Out[15]= cliques[composite[oopart[t], x, inverse[oopart[t]]]] ==
          image[IMAGE[oopart[t]], intersection[cliques[x], P[domain[oopart[t]]]]]

In[16]:= cliques[composite[oopart[t_], x_, inverse[oopart[t_]]]] :=
          image[IMAGE[oopart[t]], intersection[cliques[x], P[domain[oopart[t]]]]]
```

Lemma.

```
In[17]:= SubstTest[implies, equal[v, w], equal[image[u, v], image[u, w]], {u → IMAGE[oopart[t]],
            v → cliques[x], w -> intersection[cliques[x], P[domain[oopart[t]]]]}] // Reverse

Out[17]= or[equal[image[IMAGE[oopart[t]], cliques[x]],
            image[IMAGE[oopart[t]], intersection[cliques[x], P[domain[oopart[t]]]]]],
           not[subclass[fix[x], domain[oopart[t]]]]] == True

In[18]:= (% /. {x → x_, t → t_}) /. Equal → SetDelayed
```

Theorem. If **x** is related to **y** by a similarity transformation **oopart[t]**, then **cliques[y]** is the image of **cliques[x]** under **IMAGE[oopart[x]]**.

```
In[20]:= Map[not, SubstTest[and, implies[and[p1, p2], p5], implies[p2, p6],
            implies[and[p2, p6], p7], implies[and[p5, p7], p8], not[implies[and[p1, p2], p8]],
            {p1 -> equal[y, composite[oopart[t], x, inverse[oopart[t]]]],
             p2 → subclass[x, cartsq[domain[oopart[t]]]], p5 → equal[cliques[y],
               image[IMAGE[oopart[t]], intersection[cliques[x], P[domain[oopart[t]]]]]],
             p6 -> subclass[fix[x], domain[oopart[t]]],
             p7 -> equal[image[IMAGE[oopart[t]], cliques[x]],
               image[IMAGE[oopart[t]], intersection[cliques[x], P[domain[oopart[t]]]]]],
             p8 → equal[cliques[y], image[IMAGE[oopart[t]], cliques[x]]]}]] // Reverse

Out[20]= or[equal[cliques[y], image[IMAGE[oopart[t]], cliques[x]]],
           not[equal[y, composite[oopart[t], x, inverse[oopart[t]]]]],
           not[subclass[x, cart[domain[oopart[t]], domain[oopart[t]]]]]] == True

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

The next step is to combine this result with the equipollence theorem of the preceding section.

Theorem. If **x** and **y** are related by a similarity transformation **t**, then **cliques[x]** and **cliques[y]** are equipollent.

```
In[23]:= Map[not, SubstTest[and, implies[and[p1, p2, p3], p4],
            implies[and[p1, p2, p3], p5], implies[and[p4, p5], p6],
            not[implies[and[p1, p2, p3], p6]], {p1 → member[oopart[t], V],
             p2 → equal[y, composite[oopart[t], x, inverse[oopart[t]]]],
             p3 → subclass[x, cart[domain[oopart[t]], domain[oopart[t]]]],
             p4 → member[pair[cliques[x], image[IMAGE[oopart[t]], cliques[x]]], Q],
             p5 → equal[cliques[y], image[IMAGE[oopart[t]], cliques[x]]],
             p6 → member[pair[cliques[x], cliques[y]], Q]}]] // Reverse

Out[23]= or[member[pair[cliques[x], cliques[y]], Q],
          not[equal[y, composite[oopart[t], x, inverse[oopart[t]]]]],
          not[member[oopart[t], V]],
          not[subclass[x, cart[domain[oopart[t]], domain[oopart[t]]]]]] == True

In[24]:= (% /. {t → t_, x → x_, y → y_}) /. Equal → SetDelayed
```

Corollary. (Eliminate **oopart** wrapper.)

```
In[26]:= SubstTest[implies, equal[t, oopart[w]], or[member[pair[cliques[x], cliques[y]], Q],
            not[equal[y, composite[t, x, inverse[t]]]], not[member[t, V]],
            not[subclass[x, cart[domain[t], domain[t]]]]], w → t] // Reverse

Out[26]= or[member[pair[cliques[x], cliques[y]], Q], not[equal[y, composite[t, x, inverse[t]]]],
          not[FUNCTION[t]], not[FUNCTION[inverse[t]]], not[member[t, V]],
          not[subclass[x, cart[domain[t], domain[t]]]]] == True

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

---

eliminating variables

In this section a variable-free restatement is obtained.

Corollary. Introduce **setpart** wrapper.

```
In[30]:= SubstTest[or, member[pair[cliques[x], cliques[y]], Q],
          not[equal[y, composite[w, x, inverse[w]]]], not[member[w, BIJ]],
          not[subclass[x, cart[domain[w], domain[w]]]], w → setpart[t]] // Reverse

Out[30]= or[member[pair[cliques[x], cliques[y]], Q],
          not[equal[y, composite[setpart[t], x, inverse[setpart[t]]]]],
          not[FUNCTION[inverse[setpart[t]]]], not[FUNCTION[setpart[t]]],
          not[subclass[x, cart[domain[setpart[t]], domain[setpart[t]]]]]] == True

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

Theorem. (Variable elimination.)

```
In[32]:= Map[empty[composite[complement[#], inverse[SECOND]]] &,
          SubstTest[class, pair[pair[t, x], y],
           implies[member[pair[pair[setpart[t], setpart[x]], setpart[y]], u],
            member[pair[setpart[x], setpart[y]], v]],
           {u → composite[IMG, cross[composite[CROSS, DUP, id[BIJ]], Id],
              id[composite[inverse[S], CART, DUP, IMAGE[FIRST]]]],
            v → composite[inverse[CLIQUES], Q, CLIQUES]}]]

Out[32]= subclass[composite[CLIQUES, SIMILAR, inverse[CLIQUES]], Q] == True

In[33]:= subclass[composite[CLIQUES, SIMILAR, inverse[CLIQUES]], Q] := True
```

Corollary.

```
In[34]:= subclass[SIMILAR, composite[inverse[CLIQUES], Q, CLIQUES]] // AssertTest

Out[34]= subclass[SIMILAR, composite[inverse[CLIQUES], Q, CLIQUES]] == True

In[35]:= subclass[SIMILAR, composite[inverse[CLIQUES], Q, CLIQUES]] := True
```

Theorem. If **x** and **y** are similar relations, then **cliques[x]** and **cliques[y]** are equipollent.

```
In[36]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]],
            subclass[u, w], {u → set[PAIR[x, y]], v → SIMILAR,
             w -> composite[inverse[CLIQUES], Q, CLIQUES]}] // Reverse // MapNotNot

Out[36]= or[member[pair[cliques[x], cliques[y]], Q], not[member[pair[x, y], SIMILAR]]] == True

In[37]:= or[member[pair[cliques[x_], cliques[y_]], Q],
           not[member[pair[x_, y_], SIMILAR]]] := True
```