composites of ranges of subgroups

Johan G. F. Belinfante
2012 January 24


```
In[1]:= SetDirectory["l:"]; << goedel.12jan24a

:Package Title: goedel.12jan24a              2012 January 24 at 11:40 a.m.

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

It is now:  2012 Jan 24 at 13:39

Loading Simplification Rules

TOOLS.M is now incorporated in the GOEDEL program as of 2010 September 3

weightlimit = 40

Loading completed.

It is now:  2012 Jan 24 at 13:52
```

---

summary

The **direct product** of two groups **x** and **y** is the group **direct[x, y] = (x ⊗ y) ∘ TWIST**. The range of a subgroup of a direct product of two groups is a relation. It is natural to ask whether constructions involving such relations yield ranges of other subgroups. One such result has already been derived. If **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **inverse[w]** is the range of a subgroup of the direct product of **y** and **x**.

```
In[2]:= implies[and[member[x, GROUPS], member[y, GROUPS],
          member[w, image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]],
         member[inverse[w], image[IMAGE[SECOND], intersection[GROUPS, P[direct[y, x]]]]]]

Out[2]= True
```

In this notebook it is shown that if **u** is the range of a subgroup of the direct product of groups **y** and **z**, and if **v** is the range of a subgroup of the direct product of groups **x** and **y**, then the composite relation **u ∘ v** is the range of a subgroup of the direct product of **x** and **z**. One can think of this result as a generalization of the fact that the composite of two binary homomorphisms is a homomorphism. Indeed, the set of binary homomorphisms from a group **x** to a group **y** is a subclass of the set of ranges of subgroups of the direct product of **x** and **y**.

```
In[3]:= implies[and[member[x, GROUPS], member[y, GROUPS]],
         subclass[binhom[x, y], image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]]

Out[3]= True
```

a general result

The following general result says that ranges of substructures of direct products are relations.

Theorem. The range **w** of a substructure of the direct product of **y** and **z** is a subclass of the cartesian product of the range of **y** and the range of **z**.

```
In[4]:= SubstTest[implies, and[member[w, u], subclass[u, v]], member[w, v],
          {u -> image[IMAGE[SECOND], intersection[x, P[direct[y, z]]]],
           v -> P[cart[range[y], range[z]]]}] // Reverse

Out[4]= or[not[
           member[w, image[IMAGE[SECOND], intersection[x, P[composite[cross[y, z], TWIST]]]]]],
          subclass[w, cart[range[y], range[z]]]] == True

In[5]:= or[not[member[w_,
             image[IMAGE[SECOND], intersection[x_, P[composite[cross[y_, z_], TWIST]]]]]],
          subclass[w_, cart[range[y_], range[z_]]]] := True
```

For the special case **x = GROUPS**, this theorem says that if **w** is the range of a subgroup of the direct product of **y** and **z**, then **w ⊂ range[y] × range[z]**.

binary closed

Theorem. If **w** is the range of a subgroup of **gp[x]**, then **w** is binary closed under **gp[x]**.

```
In[7]:= SubstTest[implies, and[member[w, u], subclass[u, v]],
          member[w, v], {u -> image[IMAGE[SECOND], intersection[GROUPS, P[gp[x]]]],
           v -> binclosed[gp[x]]}] // Reverse

Out[7]= or[not[member[w, image[IMAGE[SECOND], intersection[GROUPS, P[gp[x]]]]]],
          subclass[image[gp[x], cart[w, w]], w]] == True

In[8]:= or[not[member[w_, image[IMAGE[SECOND], intersection[GROUPS, P[gp[x_]]]]]],
          subclass[image[gp[x_], cart[w_, w_]], w_]] := True
```

Corollary. If **w** is the range of a subgroup of the direct product of **gp[x]** and **gp[y]**, then **w** is binary closed under **direct[gp[x], gp[y]]**.

```
In[9]:= SubstTest[implies, member[w, image[IMAGE[SECOND], intersection[GROUPS, P[gp[z]]]]],
          member[w, binclosed[gp[z]]], z → direct[gp[x], gp[y]]] // Reverse

Out[9]= or[not[member[w, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]]],
          subclass[composite[gp[y], cross[w, w], inverse[gp[x]]], w]] == True
```

```
In[11]:= or[not[member[w_, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x_], gp[y_]], TWIST]]]]]]],
          subclass[composite[gp[y_], cross[w_, w_], inverse[gp[x_]]], w_]] := True
```

The next step is to consider composites of ranges of subgroups of direct products.

Lemma.

```
In[12]:= (SubstTest[implies, subclass[p, q], subclass[composite[r, p, z], composite[r, q, z]],
            {r -> composite[gp[z], cross[u, u]], p -> id[cart[t, t]],
             q -> composite[inverse[gp[y]], gp[y]],
             z → composite[cross[v, v], inverse[gp[x]]]}] // Reverse) /. t → range[gp[y]]

Out[12]= subclass[composite[gp[z], cross[composite[u, id[range[gp[y]]], v],
            composite[u, id[range[gp[y]]], v]], inverse[gp[x]]], composite[gp[z],
           cross[u, u], inverse[gp[y]], gp[y], cross[v, v], inverse[gp[x]]]] == True

In[13]:= (% /. {u → u_ , v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma. (Eliminating the factor **id[range[gp[y]]]**.)

```
In[14]:= SubstTest[implies, equal[t, range[gp[y]]], subclass[composite[gp[z],
            cross[composite[u, id[t], v], composite[u, id[t], v]], inverse[gp[x]]],
           composite[gp[z], cross[u, u], inverse[gp[y]], gp[y], cross[v, v], inverse[gp[x]]]],
          t → union[domain[u], range[gp[y]]]] // Reverse

Out[14]= or[not[subclass[domain[u], range[gp[y]]]],
          subclass[composite[gp[z], cross[composite[u, v], composite[u, v]], inverse[gp[x]]],
           composite[gp[z], cross[u, u], inverse[gp[y]],
            gp[y], cross[v, v], inverse[gp[x]]]]] == True

In[15]:= (% /. {u → u_ , v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma.

```
In[16]:= Map[not,
          SubstTest[and, implies[and[p1, p2], p3], implies[p0, p4], implies[and[p2, p3, p4], p5],
           not[implies[and[p0, p1, p2], p5]], {p0 -> subclass[domain[u], range[gp[y]]],
            p1 -> subclass[composite[gp[z], cross[u, u], inverse[gp[y]]], u],
            p2 -> subclass[composite[gp[y], cross[v, v], inverse[gp[x]]], v],
            p3 -> subclass[composite[gp[z], cross[u, u], inverse[gp[y]],
               gp[y], cross[v, v], inverse[gp[x]]], composite[u, v]],
            p4 -> subclass[composite[gp[z], cross[composite[u, v], composite[u, v]],
               inverse[gp[x]]], composite[gp[z], cross[u, u],
               inverse[gp[y]], gp[y], cross[v, v], inverse[gp[x]]]],
            p5 -> subclass[composite[gp[z], cross[composite[u, v], composite[u, v]],
               inverse[gp[x]]], composite[u, v]]}]] // Reverse

Out[16]= or[not[subclass[composite[gp[y], cross[v, v], inverse[gp[x]]], v]],
          not[subclass[composite[gp[z], cross[u, u], inverse[gp[y]]], u]],
          not[subclass[domain[u], range[gp[y]]]], subclass[composite[gp[z],
            cross[composite[u, v], composite[u, v]], inverse[gp[x]]], composite[u, v]]] == True
```

In[17]:=
```
(% /. {u → u_ , v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Theorem. If **u** is the range of a subgroup of **direct[gp[y], gp[z]]** and **v** is the range of a subgroup of **direct[gp[x], gp[y]]**, then **u ∘ v** is binary closed under **direct[gp[x], gp[z]]**.

In[18]:=
```
Map[not, SubstTest[and, (*implies[p1,p3],implies[p2,p4], implies[p1,p5],*)
   implies[p5, p6], implies[and[p3, p4, p6], p7],
   not[implies[and[p1, p2], p7]], {p1 -> member[u, image[IMAGE[SECOND],
       intersection[GROUPS, P[composite[cross[gp[y], gp[z]], TWIST]]]]],
 p2 -> member[v, image[IMAGE[SECOND], intersection[GROUPS,
        P[composite[cross[gp[x], gp[y]], TWIST]]]]],
    p3 -> subclass[composite[gp[z], cross[u, u], inverse[gp[y]]], u],
    p4 -> subclass[composite[gp[y], cross[v, v], inverse[gp[x]]], v],
    p5 → subclass[u, cart[range[gp[y]], range[gp[z]]]],
    p6 -> subclass[domain[u], range[gp[y]]],
    p7 -> subclass[composite[gp[z], cross[composite[u, v], composite[u, v]],
       inverse[gp[x]]], composite[u, v]]}]] // Reverse
```

Out[18]=
```
or[not[member[u, image[IMAGE[SECOND],
     intersection[GROUPS, P[composite[cross[gp[y], gp[z]], TWIST]]]]]],
  not[member[v, image[IMAGE[SECOND], intersection[GROUPS,
     P[composite[cross[gp[x], gp[y]], TWIST]]]]]],
  subclass[composite[gp[z], cross[composite[u, v], composite[u, v]], inverse[gp[x]]],
   composite[u, v]]] == True
```

In[19]:=
```
(% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

images under inversion

Lemma. If **t** is the range of a subgroup of **gp[x]**, then **image[inv[gp[x]], t] = t**.

In[20]:=
```
SubstTest[implies, and[member[t, u], subclass[u, v]],
  member[t, v], {u -> image[IMAGE[SECOND], intersection[GROUPS, P[gp[x]]]],
   v → fix[IMAGE[inv[gp[x]]]]}] // Reverse
```

Out[20]=
```
or[equal[t, image[inv[gp[x]], t]],
  not[member[t, image[IMAGE[SECOND], intersection[GROUPS, P[gp[x]]]]]]] == True
```

In[21]:=
```
or[equal[t_, image[inv[gp[x_]], t_]],
  not[member[t_, image[IMAGE[SECOND], intersection[GROUPS, P[gp[x_]]]]]]] := True
```

Theorem. If **w** is the range of a subgroup of **direct[gp[x], gp[y]]**, then **w** is fixed under imaging with the inversion function for **direct[gp[x], gp[y]]**.

In[22]:=
```
SubstTest[implies, member[w, image[IMAGE[SECOND], intersection[GROUPS, P[gp[z]]]]],
  member[w, fix[IMAGE[inv[gp[z]]]]], z → direct[gp[x], gp[y]]] // Reverse
```

Out[22]=
```
or[equal[w, composite[inv[gp[y]], w, inv[gp[x]]]], not[member[w, image[IMAGE[SECOND],
     intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]]]] == True
```

```
In[23]:= or[equal[w_, composite[inv[gp[y_]], w_, inv[gp[x_]]]],
           not[member[w_, image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[gp[x_], gp[y_]], TWIST]]]]]]] := True
```

Lemma.

```
In[24]:= SubstTest[implies, and[equal[u, p], equal[v, q]],
           equal[composite[u, v], composite[p, q]], {p -> composite[inv[gp[z]], u, inv[gp[y]]],
            q -> composite[inv[gp[y]], v, inv[gp[x]]]}] // Reverse

Out[24]= or[equal[composite[u, v], composite[inv[gp[z]], u, id[range[gp[y]]], v, inv[gp[x]]]],
           not[equal[u, composite[inv[gp[z]], u, inv[gp[y]]]]],
           not[equal[v, composite[inv[gp[y]], v, inv[gp[x]]]]]] == True

In[25]:= (% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma. (Eliminating **id[range[gp[y]]]**.)

```
In[26]:= SubstTest[implies, equal[t, range[gp[y]]],
           or[equal[composite[u, v], composite[inv[gp[z]], u, id[t], v, inv[gp[x]]]],
            not[equal[u, composite[inv[gp[z]], u, inv[gp[y]]]]],
            not[equal[v, composite[inv[gp[y]], v, inv[gp[x]]]]]],
           t → union[domain[u], range[gp[y]]]] // Reverse

Out[26]= or[equal[composite[u, v], composite[inv[gp[z]], u, v, inv[gp[x]]]],
           not[equal[u, composite[inv[gp[z]], u, inv[gp[y]]]]],
           not[equal[v, composite[inv[gp[y]], v, inv[gp[x]]]]],
           not[subclass[domain[u], range[gp[y]]]]] == True

In[27]:= (% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Theorem. If **u** is the range of a subgroup of **direct[gp[y], gp[z]]** and **v** is the range of a subgroup of **direct[gp[x], gp[y]]**, then **u ∘ v** is fixed under imaging with the inversion function for **direct[gp[x], gp[z]]**.

```
In[28]:= Map[not, SubstTest[and, (*implies[p1,p3],implies[p2,p4], implies[p1,p5],*)
            implies[p5, p6], implies[and[p3, p4, p6], p7],
            not[implies[and[p1, p2], p7]], {p1 -> member[u, image[IMAGE[SECOND],
                intersection[GROUPS, P[composite[cross[gp[y], gp[z]], TWIST]]]]],
         p2 -> member[v, image[IMAGE[SECOND], intersection[GROUPS,
                P[composite[cross[gp[x], gp[y]], TWIST]]]]],
             p3 -> equal[u, composite[inv[gp[z]], u, inv[gp[y]]]],
             p4 -> equal[v, composite[inv[gp[y]], v, inv[gp[x]]]],
             p5 → subclass[u, cart[range[gp[y]], range[gp[z]]]],
             p6 -> subclass[domain[u], range[gp[y]]],
             p7 -> equal[composite[u, v], composite[inv[gp[z]], u, v, inv[gp[x]]]]}]] // Reverse

Out[28]= or[equal[composite[u, v], composite[inv[gp[z]], u, v, inv[gp[x]]]],
           not[member[u, image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[gp[y], gp[z]], TWIST]]]]]],
           not[member[v, image[IMAGE[SECOND], intersection[GROUPS,
               P[composite[cross[gp[x], gp[y]], TWIST]]]]]]] == True

In[29]:= (% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma.

```
In[30]:= SubstTest[subclass, t, intersection[u, v],
          {u -> binclosed[direct[gp[x], gp[y]]], v -> fix[IMAGE[inv[direct[gp[x], gp[y]]]]]}]

Out[30]= and[subclass[t, binclosed[composite[cross[gp[x], gp[y]], TWIST]]],
           subclass[t, fix[IMAGE[cross[inv[gp[x]], inv[gp[y]]]]]]] ==
          subclass[t, union[image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]], set[0]]]

In[31]:= and[subclass[t_, binclosed[composite[cross[gp[x_], gp[y_]], TWIST]]],
           subclass[t_, fix[IMAGE[cross[inv[gp[x_]], inv[gp[y_]]]]]]] :=
          subclass[t, union[image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]], set[0]]]
```

Lemma.

```
In[32]:= Map[implies[and[not[empty[t]], #], member[t, image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]]] &,
          SubstTest[member, t, intersection[u, v], {u -> binclosed[direct[gp[x], gp[y]]],
            v -> fix[IMAGE[inv[direct[gp[x], gp[y]]]]]}]]

Out[32]= or[equal[0, t], member[t, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]],
           not[equal[t, composite[inv[gp[y]], t, inv[gp[x]]]]], not[member[t, V]],
           not[subclass[composite[gp[y], cross[t, t], inverse[gp[x]]], t]]] == True

In[33]:= (% /. {t → t_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma.

```
In[34]:= SubstTest[or, equal[0, t], member[t, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]],
           not[equal[t, composite[inv[gp[y]], t, inv[gp[x]]]]], not[member[t, V]],
           not[subclass[composite[gp[y], cross[t, t], inverse[gp[x]]], t]],
           t → composite[u, v]] // Reverse

Out[34]= or[equal[0, intersection[domain[u], range[v]]],
           member[composite[u, v], image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]],
           not[equal[composite[u, v], composite[inv[gp[y]], u, v, inv[gp[x]]]]],
           not[member[composite[u, v], V]], not[subclass[composite[gp[y],
              cross[composite[u, v], composite[u, v]], inverse[gp[x]]], composite[u, v]]]] == True

In[35]:= (% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma.

```
In[36]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3],
            implies[and[p0, p2, p3], p4], not[implies[and[p0, p1], p4]],
            {p0 → and[member[composite[u, v], V], not[disjoint[domain[u], range[v]]]],
             p1 → and[member[u, image[IMAGE[SECOND], intersection[GROUPS,
                  P[composite[cross[gp[y], gp[z]], TWIST]]]]], member[v, image[IMAGE[SECOND],
                 intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]]],
             p2 -> subclass[composite[gp[z], cross[composite[u, v], composite[u, v]],
                inverse[gp[x]]], composite[u, v]],
             p3 -> equal[composite[u, v], composite[inv[gp[z]], u, v, inv[gp[x]]]],
             p4 → member[composite[u, v], image[IMAGE[SECOND],
                intersection[GROUPS, P[composite[cross[gp[x], gp[z]], TWIST]]]]]}]] // Reverse

Out[36]= or[equal[0, intersection[domain[u], range[v]]],
          member[composite[u, v], image[IMAGE[SECOND],
            intersection[GROUPS, P[composite[cross[gp[x], gp[z]], TWIST]]]]],
          not[member[u, image[IMAGE[SECOND], intersection[GROUPS,
              P[composite[cross[gp[y], gp[z]], TWIST]]]]]], not[member[v, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]]]] == True

In[37]:= (% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma. (Remove the wrappers.)

```
In[38]:= SubstTest[implies, and[equal[x, gp[r]], equal[y, gp[s]], equal[z, gp[t]]],
            or[equal[0, intersection[domain[u], range[v]]], member[composite[u, v],
              image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, z], TWIST]]]]],
             not[member[u, image[IMAGE[SECOND], intersection[GROUPS,
                 P[composite[cross[y, z], TWIST]]]]]], not[member[v,
               image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]]],
            {r → x, s → y, t → z}] // MapNotNot // Reverse

Out[38]= or[equal[0, intersection[domain[u], range[v]]], member[composite[u, v],
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, z], TWIST]]]]],
          not[member[u, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[y, z], TWIST]]]]]], not[member[v,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]], not[member[z, GROUPS]]] == True

In[39]:= (% /. {u → u_, v → v_, x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma.

```
In[40]:= SubstTest[implies, and[member[t, image[IMAGE[SECOND], intersection[GROUPS, P[z]]]],
            member[z, GROUPS]], member[e[z], t], z → direct[gp[x], gp[y]]] // Reverse

Out[40]= or[equal[0, gp[x]], equal[0, gp[y]],
          member[pair[e[gp[x]], e[gp[y]]], t], not[member[t, image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[gp[x], gp[y]], TWIST]]]]]]] == True

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

Theorem. If **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **pair[e[x], e[y]] ∈ w**.

```
In[43]:= SubstTest[implies, and[equal[x, gp[u]], equal[y, gp[v]]],
           or[equal[0, x], equal[0, y], member[pair[e[x], e[y]], w], not[member[w,
              image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]]],
           {u → x, v → y}] // Reverse // MapNotNot

Out[43]= or[member[pair[e[x], e[y]], w], not[member[w,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[44]:= or[member[pair[e[x_], e[y_]], w_], not[member[w_,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Corollary 1. If **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **e[x] ∈ domain[w]**.

```
In[46]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p1, p3], implies[and[p2, p3], p4], not[implies[p1, p4]],
           {p1 → and[member[x, GROUPS], member[y, GROUPS], member[w, image[IMAGE[SECOND],
                intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
            p2 -> member[pair[e[x], e[y]], w], p3 → subclass[w, cart[range[x], range[y]]],
            p4 → member[e[x], domain[w]]}]] // Reverse

Out[46]= or[member[e[x], domain[w]], not[member[w,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[47]:= or[member[e[x_], domain[w_]], not[member[w_,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Corollary 2. If **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **e[y] ∈ range[w]**.

```
In[48]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p1, p3], implies[and[p2, p3], p4], not[implies[p1, p4]],
           {p1 → and[member[x, GROUPS], member[y, GROUPS], member[w, image[IMAGE[SECOND],
                intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
            p2 -> member[pair[e[x], e[y]], w], p3 → subclass[w, cart[range[x], range[y]]],
            p4 → member[e[y], range[w]]}]] // Reverse

Out[48]= or[member[e[y], range[w]], not[member[w,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[49]:= or[member[e[y_], range[w_]], not[member[w_,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Main Theorem. If **u** is the range of a subgroup of the direct product of groups **y** and **z**, and if **v** is the range of a subgroup of the direct product of groups **x** and **y**, then the composite relation **u ∘ v** is the range of a subgroup of the direct product of **x** and **z**.

```
In[50]:= Map[not, SubstTest[and, implies[p1, p2],
            implies[p1, p3], implies[and[p2, p3], p4], implies[and[p1, p4], p5],
            not[implies[p1, p5]], {p1 → and[member[u, image[IMAGE[SECOND],
                 intersection[GROUPS, P[composite[cross[y, z], TWIST]]]]], member[v,
                image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]],
              member[x, GROUPS], member[y, GROUPS], member[z, GROUPS]],
          p2 → member[e[y], domain[u]], p3 → member[e[y], range[v]],
          p4 → not[disjoint[domain[u], range[v]]],
          p5 -> member[composite[u, v], image[IMAGE[SECOND],
                intersection[GROUPS, P[composite[cross[x, z], TWIST]]]]]}]] // Reverse

Out[50]= or[member[composite[u, v],
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, z], TWIST]]]]],
          not[member[u, image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[y, z], TWIST]]]]]], not[member[v,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]], not[member[z, GROUPS]]] == True

In[51]:= or[member[composite[u_, v_],
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, z_], TWIST]]]]],
          not[member[u_, image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[y_, z_], TWIST]]]]]], not[member[v_,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]], not[member[z_, GROUPS]]] := True
```

eliminating the variables u and v

In this section, the variables **u** and **v** are eliminated.

Theorem.

```
In[52]:= Map[empty[composite[Id, complement[#]]] &,
          SubstTest[class, pair[u, v], implies[and[member[x, r], member[y, r],
              member[z, r], member[pair[u, v], s]], member[pair[u, v], t]], {r → GROUPS, s ->
              cart[image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[y, z], TWIST]]]],
               image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]],
            t -> image[inverse[COMPOSE], image[IMAGE[SECOND],
                intersection[GROUPS, P[composite[cross[x, z], TWIST]]]]]}]]

Out[52]= or[not[member[x, GROUPS]], not[member[y, GROUPS]],
          not[member[z, GROUPS]], subclass[image[COMPOSE,
              cart[image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[y, z], TWIST]]]],
               image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
            image[IMAGE[SECOND], intersection[GROUPS,
               P[composite[cross[x, z], TWIST]]]]]] == True
```

```
In[53]:= or[not[member[x_, GROUPS]], not[member[y_, GROUPS]],
           not[member[z_, GROUPS]], subclass[image[COMPOSE, cart[
              image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[y_, z_], TWIST]]]],
              image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
            image[IMAGE[SECOND], intersection[GROUPS,
              P[composite[cross[x_, z_], TWIST]]]]]]] := True
```

fractions as subgroups

Although rational numbers have not yet been formally defined in the **GOEDEL** program, it is perhaps apt to contemplate a possible (but non-standard) definition that would give the rationals some nice algebraic properties. The concentional definition of integers as equivalence classes of a certain equivalence relation amounts to making rational numbers be linear functions in the plane **Z × Z** with the point at the origin forcibly removed. Restoring that deleted point would make better algebraic sense.

If **n** and **d** are integers (with nonzero **d**), it is natural to think of the fraction **r = n/d** as a linear function that satisfies **n = r(d)**. The graph of a linear function is the range of a subgroup of the direct product of **INTADD** with itself. Thus the following lemma would then amount to the statement that one could consider any rational number as a certain subgroup of **direct[INTADD, INTADD]**. A natural candidate for **n/d** is obtained by rewriting **n/d = (1/d) ∘ (n/1)**, and identifying **n/1** with **inttimes[n]** and **1/d** with the inverse of **d/1**. Thus:

n/d = inverse[inttimes[d]] ∘ inttimes[n].

This proposed definition of rational numbers is also suggested by analogy with the construction of integers in terms of natural numbers. In the latter case, an integer is a function in the plane $\omega \times \omega$ whose graph is a straight line with slope 1. Such functions have the form **inverse[plus[a]] ∘ plus[b]**, where **a** and **b** are natural numbers. These lines are parallel to one another and are the equivalence classes of the equivalence relaton **EQUIDIFF**.

Lemma. If **x** and **y** are integers, then **inverse[inttimes[x]] ∘ inttimes[y]** is the range of a subgroup of the direct product of **INTADD** with itself.

```
In[57]:= SubstTest[or, member[composite[u, v],
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[r, t], TWIST]]]]],
           not[member[u, image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[s, t], TWIST]]]]]], not[member[v,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[r, s], TWIST]]]]]],
           not[member[r, GROUPS]], not[member[s, GROUPS]], not[member[t, GROUPS]],
           {r → INTADD, s → INTADD, t → INTADD,
            u → inverse[inttimes[x]], v → inttimes[y]}] // Reverse

Out[57]= or[member[composite[inverse[inttimes[x]], inttimes[y]], image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[INTADD, INTADD], TWIST]]]]],
           not[member[x, Z]], not[member[y, Z]]] == True

        (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma. The converse statement.

```
In[65]:= SubstTest[or, not[empty[w]], not[member[w,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[u, v], TWIST]]]]]],
           not[member[u, GROUPS]], not[member[v, GROUPS]],
           {u → INTADD, v → INTADD, w -> composite[inverse[inttimes[x]], inttimes[y]]}] // Reverse

Out[65]= or[and[member[x, Z], member[y, Z]],
           not[member[composite[inverse[inttimes[x]], inttimes[y]], image[IMAGE[SECOND],
              intersection[GROUPS, P[composite[cross[INTADD, INTADD], TWIST]]]]]]] == True

In[66]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem. The relation **inverse[inttimes[x]] ∘ inttimes[y]** is the range of a subgroup of the direct product of **INTADD** with itself if and only if **x** and **y** are both integers.

```
In[67]:= equiv[member[composite[inverse[inttimes[x]], inttimes[y]], image[IMAGE[SECOND],
            intersection[GROUPS, P[composite[cross[INTADD, INTADD], TWIST]]]]],
          and[member[x, Z], member[y, Z]]]

Out[67]= True

In[69]:= member[composite[inverse[inttimes[x_]], inttimes[y_]], image[IMAGE[SECOND],
            intersection[GROUPS, P[composite[cross[INTADD, INTADD], TWIST]]]]] :=
          and[member[x, Z], member[y, Z]]
```