domain and range of subgroups of direct products

Johan G. F. Belinfante
2012 January 25

```
In[1]:= SetDirectory["l:"]; << goedel.12jan24b

        :Package Title: goedel.12jan24b              2012 January 24 at 4:20 p.m.

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

        It is now:  2012 Jan 25 at 13:30

        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 25 at 13:43
```

summary

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

two general inclusions

In this section two general inclusions are derived. (Only the special case **x = GROUPS** will be used in the remainder of this notebook.)

Theorem.

```
In[2]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
           {p1 → member[w, image[IMAGE[SECOND], intersection[x, P[direct[y, z]]]]],
            p2 -> subclass[w, cart[range[y], range[z]]],
            p3 → subclass[domain[w], range[y]]}]] // Reverse

Out[2]= or[not[
           member[w, image[IMAGE[SECOND], intersection[x, P[composite[cross[y, z], TWIST]]]]]],
          subclass[domain[w], range[y]]] == True

In[3]:= or[not[member[w_,
            image[IMAGE[SECOND], intersection[x_, P[composite[cross[y_, z_], TWIST]]]]]],
          subclass[domain[w_], range[y_]]] := True
```

Theorem.

```
In[4]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
          {p1 → member[w, image[IMAGE[SECOND], intersection[x, P[direct[y, z]]]]],
           p2 -> subclass[w, cart[range[y], range[z]]],
           p3 → subclass[range[w], range[z]]}]] // Reverse

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

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

The above two theorems imply in particular that if **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **domain[w] ⊂ range[x]** and **range[w] ⊂ range[y]**.

---

ranges of subgroups

A set **x** is a **subgroup** of a group **y** if **x** is a group and **x ⊂ y**. Subgroups are determined by their ranges. If a set **w** is the range of a subgroup **x** of **y**, then **x = y ∘ id[w × w]**. In this section a familiar criterion for a set **w** to be the range of a subgroup of a group **x** is obtained by simply introducing a new variable **w** in an available rewrite rule.

Lemma. (This has a redundant sethood literal.)

```
In[6]:= Map[not, SubstTest[and, implies[p1, p2], implies[and[p2, p3], p4],
          not[implies[and[p1, p3], p4]], {p1 → member[x, GROUPS],
           p2 → equal[intersection[binclosed[x], fix[IMAGE[inv[x]]], image[E, set[e[x]]]],
             image[IMAGE[SECOND], intersection[GROUPS, P[x]]]],
           p3 → member[w, intersection[binclosed[x], fix[IMAGE[inv[x]]], image[E, set[e[x]]]]],
           p4 → member[w, image[IMAGE[SECOND], intersection[GROUPS, P[x]]]]}]] // Reverse

Out[6]= or[member[w, image[IMAGE[SECOND], intersection[GROUPS, P[x]]]],
        not[equal[w, image[inv[x], w]]], not[member[w, V]], not[member[x, GROUPS]],
        not[member[e[x], w]], not[subclass[image[x, cart[w, w]], w]]] == True

In[7]:= (% /. {w → w_, x → x_}) /. Equal → SetDelayed
```

Lemma. (Needed to deal with the sethood literal.)

```
In[8]:= SubstTest[implies, equal[x, gp[t]],
          implies[equal[w, image[inv[x], w]], member[w, V]], t → x] // Reverse // MapNotNot

Out[8]= or[member[w, V], not[equal[w, image[inv[x], w]]], not[member[x, GROUPS]]] == True

In[9]:= (% /. {w → w_, x → x_}) /. Equal → SetDelayed
```

Theorem. If **w** is binary closed under a group **x**, holds the neutral element **e[x]**, and is fixed under imaging with **inv[x]**, then **w** is the range of a subgroup of **x**.

```
In[10]:= Map[not, SubstTest[and, implies[p1, p2], implies[and[p2, p3, p4], p5],
           not[implies[and[p1, p3], p5]], {p1 → member[x, GROUPS],
            p2 → equal[intersection[binclosed[x], fix[IMAGE[inv[x]]], image[E, set[e[x]]]],
              image[IMAGE[SECOND], intersection[GROUPS, P[x]]]], p3 → and[
              equal[w, image[inv[x], w]], member[e[x], w], subclass[image[x, cart[w, w]], w]],
            p4 → member[w, V], p5 → member[w, image[IMAGE[SECOND],
               intersection[GROUPS, P[x]]]]}]] // Reverse

Out[10]= or[member[w, image[IMAGE[SECOND], intersection[GROUPS, P[x]]]],
          not[equal[w, image[inv[x], w]]], not[member[x, GROUPS]],
          not[member[e[x], w]], not[subclass[image[x, cart[w, w]], w]]] == True

In[11]:= or[member[w_, image[IMAGE[SECOND], intersection[GROUPS, P[x_]]]],
          not[equal[image[inv[x_], w_], w_]], not[member[e[x_], w_]],
          not[member[x_, GROUPS]], not[subclass[image[x_, cart[w_, w_]], w_]]] := True
```

---

binary closed

In this section it is shown that if **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **range[w]** is binary closed under **y**.

Lemma.

```
In[12]:= SubstTest[implies, subclass[t, w], subclass[range[t], range[w]],
          t → composite[y, cross[w, w], inverse[binop[x]]]] // Reverse

Out[12]= or[not[subclass[composite[y, cross[w, w], inverse[binop[x]]], w]], subclass[
           image[y, cart[image[w, fix[domain[binop[x]]]], image[w, fix[domain[binop[x]]]]]],
           range[w]]] == True

In[13]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem. If **x** is a binary operation, if **w** is binary closed under **direct[x, y]** and if **domain[w] ⊂ fix[domain[x]]**, then **range[w]** is binary closed under **y**.

```
In[14]:= SubstTest[implies, equal[t, fix[domain[binop[x]]]],
          or[not[subclass[composite[y, cross[w, w], inverse[binop[x]]], w]],
           subclass[image[y, cart[image[w, t], image[w, t]]], range[w]]],
          t → union[fix[domain[binop[x]]], domain[w]]] // Reverse

Out[14]= or[not[subclass[composite[y, cross[w, w], inverse[binop[x]]], w]],
          not[subclass[domain[w], fix[domain[binop[x]]]]],
          subclass[image[y, cart[range[w], range[w]]], range[w]]] == True

In[15]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

Corollary. (Eliminate the **binop** wrapper.)

```
In[16]:= SubstTest[implies, equal[x, binop[t]],
           or[not[subclass[composite[y, cross[w, w], inverse[x]], w]],
            not[subclass[domain[w], fix[domain[x]]]],
            subclass[image[y, cart[range[w], range[w]]], range[w]]], t → x] // Reverse

Out[16]= or[not[member[x, BINOPS]], not[subclass[composite[y, cross[w, w], inverse[x]], w]],
           not[subclass[domain[w], fix[domain[x]]]],
           subclass[image[y, cart[range[w], range[w]]], range[w]]] == True

In[17]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

This result will now be specialized to the case of groups.

Lemma.

```
In[18]:= SubstTest[implies,
           and[member[t, GROUPS], member[w, image[IMAGE[SECOND], intersection[GROUPS, P[t]]]]],
           subclass[image[t, cart[w, w]], w], t → direct[x, y]] // Reverse

Out[18]= or[not[member[w,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
           not[member[composite[cross[x, y], TWIST], GROUPS]],
           subclass[composite[y, cross[w, w], inverse[x]], w]] == True

In[19]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem. The range of a subgroup of the direct product of two groups is binary closed under the direct product.

```
In[20]:= Map[not, SubstTest[and, implies[p1, p2], implies[and[p1, p2], p3],
            not[implies[p1, p3]], {p1 → and[member[x, GROUPS], member[y, GROUPS],
               member[w, image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]],
             p2 -> member[composite[cross[x, y], TWIST], GROUPS],
             p3 -> subclass[composite[y, cross[w, w], inverse[x]], w]}]] // Reverse

Out[20]= or[not[member[w,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
           not[member[x, GROUPS]], not[member[y, GROUPS]],
           subclass[composite[y, cross[w, w], inverse[x]], w]] == True

In[21]:= or[not[member[w_,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
           not[member[x_, GROUPS]], not[member[y_, GROUPS]],
           subclass[composite[y_, cross[w_, w_], inverse[x_]], w_]] := True
```

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

```
In[22]:= Map[not,
           SubstTest[and, (*implies[p1,p2], implies[p1,p4],implies[p1,p5],implies[p1,p6],*)
            implies[p6, p7], implies[and[p2, p4, p5, p7], p8],
            not[implies[p1, p8]], {p1 → and[member[x, GROUPS], member[y, GROUPS],
               member[w, image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]],
             p2 → member[x, BINOPS], p4 -> subclass[composite[y, cross[w, w], inverse[x]], w],
             p5 → equal[fix[domain[x]], range[x]],
             p6 → subclass[w, cart[range[x], range[y]]], p7 → subclass[domain[w], range[x]],
             p8 -> subclass[image[y, cart[range[w], range[w]]], range[w]]}]] // Reverse

Out[22]= or[not[member[w,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
           not[member[x, GROUPS]], not[member[y, GROUPS]],
           subclass[image[y, cart[range[w], range[w]]], range[w]]] == True

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

images under inversion

The first theorem is obtained from an available rewrite rule by just removing the **gp** wrappers.

Theorem. If **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **w** is fixed under imaging with respect to the inversion function for the direct product.

```
In[24]:= SubstTest[implies, and[equal[x, gp[u]], equal[y, gp[v]]], implies[member[w,
              image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]],
             equal[w, composite[inv[y], w, inv[x]]]], {u → x, v → y}] // Reverse // MapNotNot

Out[24]= or[equal[w, composite[inv[y], w, inv[x]]], not[member[w,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
           not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[25]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[26]:= SubstTest[implies, equal[w, t], equal[range[w], range[t]],
           t → composite[inv[gp[y]], w, inv[gp[x]]]] // Reverse

Out[26]= or[equal[image[inv[gp[y]], image[w, range[gp[x]]]], range[w]],
           not[equal[w, composite[inv[gp[y]], w, inv[gp[x]]]]]] == True

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

Lemma.

```
In[28]:= SubstTest[implies,
           and[equal[x, gp[u]], equal[y, gp[v]], equal[w, composite[inv[y], w, inv[x]]]],
           equal[image[inv[y], image[w, range[x]]], range[w]],
           {u → x, v → y}] // Reverse // MapNotNot

Out[28]= or[equal[image[inv[y], image[w, range[x]]], range[w]],
          not[equal[w, composite[inv[y], w, inv[x]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[29]:= (% /. {w → w_, 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 **range[w]** is fixed under imaging with respect to the inversion function **inv[y]**.

```
In[30]:= Map[not, SubstTest[and, (*implies[p1,p2],*) implies[and[p1, p2], p3],
           (*implies[p1,p4],*) implies[p4, p5], (*implies[and[p3,p5],p6],*)
           not[implies[p1, p6]], {p1 -> and[member[x, GROUPS], member[y, GROUPS],
              member[w, image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]],
            p2 -> equal[w, composite[inv[y], w, inv[x]]],
            p3 -> equal[image[inv[y], image[w, range[x]]], range[w]],
            p4 → subclass[domain[w], range[x]], p5 → equal[image[w, range[x]], range[w]],
            p6 -> equal[image[inv[y], range[w]], range[w]]}]] // Reverse

Out[30]= or[equal[image[inv[y], range[w]], 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[31]:= or[equal[image[inv[y_], range[w_]], 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 **w** is the range of a subgroup of the direct product of groups **x** and **y**, then **range[w]** is the range of a subgroup of **y**.

```
In[32]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p1, p3], implies[p1, p4], (*implies[and[p1,p2,p3,p4],p5],*)
           not[implies[p1, p5]], {p1 -> and[member[x, GROUPS], member[y, GROUPS],
              member[w, image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]],
            p2 → subclass[image[y, cart[range[w], range[w]]], range[w]],
            p3 -> equal[image[inv[y], range[w]], range[w]], p4 → member[e[y], range[w]], p5 →
             member[range[w], image[IMAGE[SECOND], intersection[GROUPS, P[y]]]]}]] // Reverse

Out[32]= or[member[range[w], image[IMAGE[SECOND], intersection[GROUPS, P[y]]]], not[member[w,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[33]:= or[
          member[range[w_], image[IMAGE[SECOND], intersection[GROUPS, P[y_]]]], not[member[w_,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

A dual result will now be derived.

Lemma.

```
In[34]:= SubstTest[or,
          member[range[t], image[IMAGE[SECOND], intersection[GROUPS, P[y]]]], not[member[t,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]], t → inverse[w]] // Reverse

Out[34]= or[member[domain[w], image[IMAGE[SECOND], intersection[GROUPS, P[y]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]],
          not[member[inverse[w], image[IMAGE[SECOND],
             intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]]] == True

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

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

```
In[36]:= Map[not, SubstTest[and, implies[p1, p2], implies[and[p1, p2], p3],
           not[implies[p1, p3]], {p1 -> and[member[x, GROUPS], member[y, GROUPS],
              member[w, image[IMAGE[SECOND], intersection[GROUPS, P[direct[x, y]]]]]],
            p2 -> member[inverse[w], image[IMAGE[SECOND],
               intersection[GROUPS, P[composite[cross[y, x], TWIST]]]]], p3 ->
             member[domain[w], image[IMAGE[SECOND], intersection[GROUPS, P[x]]]]}]] // Reverse

Out[36]= or[member[domain[w], image[IMAGE[SECOND], intersection[GROUPS, P[x]]]], not[member[w,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x, y], TWIST]]]]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[37]:= or[member[domain[w_],
           image[IMAGE[SECOND], intersection[GROUPS, P[x_]]]], not[member[w_,
            image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[x_, y_], TWIST]]]]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

an application to arithmetic

It has been proposed to define the rational number **n/d** as **inverse[inttimes[d]] ∘ inttimes[n]**. If this were done, then a rational number would be a certain special type of subgroup of the direct product **direct[INTADD, INTADD]**. It then would follow that the domain and range of a rational number are both ranges of subgroups of **INTADD**. The theorem below addresses this special case. Recall that the range of any subgroup of **INTADD** is the set of integer multiples of some integer. Accordingly, the set of ranges of subgroups of **INTADD** is the set **image[VERTSECT[INTDIV], Z]**.

Theorem. If **x** and **y** are integers, then the inverse image under **inttimes[x]** of the set of multiples of **y** is the set of multiples of some integer.

```
In[38]:= SubstTest[or,
           member[range[w], image[IMAGE[SECOND], intersection[GROUPS, P[v]]]], not[member[w,
             image[IMAGE[SECOND], intersection[GROUPS, P[composite[cross[u, v], TWIST]]]]]],
           not[member[u, GROUPS]], not[member[v, GROUPS]],
           {w → composite[inverse[inttimes[x]], inttimes[y]], u → INTADD, v → INTADD}] // Reverse

Out[38]= or[member[image[inverse[inttimes[x]], image[INTDIV, set[y]]],
            image[VERTSECT[INTDIV], Z]], not[member[x, Z]], not[member[y, Z]]] == True

In[39]:= or[member[image[inverse[inttimes[x_]], image[INTDIV, set[y_]]],
            image[VERTSECT[INTDIV], Z]], not[member[x_, Z]], not[member[y_, Z]]] := True
```

The variable **y** can be eliminated.

Corollary. If **x** is an integer, then **image[VERTSECT[INTDIV], Z]** is invariant under **IMAGE[inverse[inttimes[x]]]**.

```
In[40]:= Map[equal[V, domain[#]] &, SubstTest[reify, y,
           case[or[member[image[inverse[inttimes[x]], image[INTDIV, set[y]]],
              image[VERTSECT[INTDIV], z]], not[member[x, z]], not[member[y, z]]]], z → Z]]

Out[40]= or[not[member[x, Z]],
           subclass[image[IMAGE[inverse[inttimes[x]]], image[VERTSECT[INTDIV], Z]],
            image[VERTSECT[INTDIV], Z]]] == True

In[41]:= or[not[member[x_, Z]],
           subclass[image[IMAGE[inverse[inttimes[x_]]], image[VERTSECT[INTDIV], Z]],
            image[VERTSECT[INTDIV], Z]]] := True
```

It is also possible to eliminate the remaining variable **x** in the same fashion. Instead, the special case **x = –1** will be considered.

Lemma.

```
In[42]:= SubstTest[implies, member[x, Z],
           subclass[image[IMAGE[inverse[inttimes[x]]], image[VERTSECT[INTDIV], Z]],
            image[VERTSECT[INTDIV], Z]], x → inverse[plus[set[0]]]] // Reverse

Out[42]= subclass[image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]],
           image[VERTSECT[INTDIV], Z]] == True

In[43]:= % /. Equal → SetDelayed
```

Lemma.

```
In[44]:= composite[IMAGE[INVERSE], IMAGE[INVERSE]] // FastReifNormality

Out[44]= composite[IMAGE[INVERSE], IMAGE[INVERSE]] == IMAGE[id[P[cart[V, V]]]]

In[45]:= composite[IMAGE[INVERSE], IMAGE[INVERSE]] := IMAGE[id[P[cart[V, V]]]]
```

Lemma. Simplification rule.

```
In[46]:= ImageComp[IMAGE[INVERSE], IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]] // Reverse

Out[46]= image[IMAGE[INVERSE], image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]]] ==
          image[VERTSECT[INTDIV], Z]

In[47]:= % /. Equal → SetDelayed
```

Lemma.

```
In[48]:= SubstTest[implies, subclass[u, v], subclass[image[t, u], image[t, v]],
           {t → IMAGE[INVERSE], u -> image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]],
            v -> image[VERTSECT[INTDIV], Z]}] // Reverse

Out[48]= subclass[image[VERTSECT[INTDIV], Z],
           image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]]] == True

In[49]:= % /. Equal → SetDelayed
```

Theorem. A simplification rule.

```
In[50]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]],
           v -> image[VERTSECT[INTDIV], Z]}]

Out[50]= equal[image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]],
           image[VERTSECT[INTDIV], Z]] == True

In[51]:= image[IMAGE[INVERSE], image[VERTSECT[INTDIV], Z]] := image[VERTSECT[INTDIV], Z]
```

The above result is not to be confused with the following related fact.

Theorem. The set of multiples of an integer is the same as the set of multiples of its negative.

```
In[52]:= ImageComp[INTDIV, INVERSE, set[int[x]]] // Reverse

Out[52]= image[INTDIV, set[inverse[int[x]]]] == image[INTDIV, set[int[x]]]

In[53]:= image[INTDIV, set[inverse[int[x_]]]] := image[INTDIV, set[int[x]]]
```