range of a binary homomorphism between groups

Johan G. F. Belinfante
2012 February 9

```
In[1]:= SetDirectory["l:"]; << goedel.12feb07a

        :Package Title: goedel.12feb07a               2012 February 7 at 3:35 p.m.

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

        It is now:  2012 Feb 9 at 14:38

        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 Feb 9 at 14:51
```

summary

In this notebook it is shown that the range of a binary homomorphism from a group **x** to a group **y** is the range of a subgroup of **y**.

derivation

Theorem. The range of any functor **t** from a group **x** to a group **y** satisfies **image[y, range[t] × range[t]] = range[t]**.

```
In[2]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p1, p3], implies[and[p1, p2, p3], p4], not[implies[p1, p4]],
           {p1 -> and[member[x, GROUPS], member[y, GROUPS], functor[t, x, y]],
            p2 → equal[domain[x], cart[range[x], range[x]]], p3 → category[y],
            p4 -> equal[image[y, cart[range[t], range[t]]], range[t]]}]] // Reverse

Out[2]= or[equal[image[y, cart[range[t], range[t]]], range[t]],
          not[functor[t, x, y]], not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[3]:= or[equal[image[y_, cart[range[t_], range[t_]]], range[t_]],
          not[functor[t_, x_, y_]], not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

The statement that **t** is a functor from a group **x** to a group **y** is equivalent to the statement that **t** is a binary homomorphism from **x** to **y**.

Corollary. The range of any binary homomorphism **t** from a group **x** to a group **y** satisfies **image[y, range[t] × range[t]] = range[t]**.

```
In[4]:= 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[t, binhom[x, y]]],
           p2 → functor[t, x, y], p3 → equal[image[y, cartsq[range[t]]], range[t]]}]] // Reverse

Out[4]= or[equal[image[y, cart[range[t], range[t]]], range[t]],
          not[member[t, binhom[x, y]]], not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[5]:= or[equal[image[y_, cart[range[t_], range[t_]]], range[t_]],
          not[member[t_, binhom[x_, y_]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Lemma. If **t** is a binary homomorphism from a group **x** to **y**, then **e[x] ∈ domain[t]**.

```
In[6]:= Map[not, SubstTest[and, implies[and[p1, p2], p3], implies[and[p1, p3], p4],
           not[implies[and[p1, p2], p4]], {p1 -> member[x, GROUPS], p2 -> member[t, binhom[x, y]],
            p3 -> equal[domain[t], range[x]], p4 → member[e[x], domain[t]]}]] // Reverse

Out[6]= or[member[e[x], domain[t]], not[member[t, binhom[x, y]]], not[member[x, GROUPS]]] == True

In[7]:= or[member[e[x_], domain[t_]],
          not[member[t_, binhom[x_, y_]]], not[member[x_, GROUPS]]] := True
```

Lemma. If **t** is a binary homomorphism from a group **x** to a group **y**, then **e[y] ∈ range[t]**.

```
In[8]:= Map[not, SubstTest[and, (*implies[p1,p2],implies[p1,p3],implies[p1,p4],*)
           implies[and[p2, p3, p4], p5], not[implies[p1, p5]],
           {p1 -> and[member[x, GROUPS], member[y, GROUPS], member[t, binhom[x, y]]],
            p2 -> equal[APPLY[t, e[x]], e[y]], p3 → member[e[x], domain[t]],
            p4 → FUNCTION[t], p5 → member[e[y], range[t]]}]] // Reverse

Out[8]= or[member[e[y], range[t]], not[member[t, binhom[x, y]]],
          not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[9]:= or[member[e[y_], range[t_]], not[member[t_, binhom[x_, y_]]],
          not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Lemma.

```
In[10]:= SubstTest[implies, equal[u, v], equal[range[u], range[v]],
           {u → composite[t, inv[x]], v -> composite[inv[y], t]}] // Reverse

Out[10]= or[equal[image[t, domain[inv[x]]], image[inv[y], range[t]]],
           not[equal[composite[t, inv[x]], composite[inv[y], t]]]] == True

In[11]:= (% /. {t → t_, x → x_, y → y_}) /. Equal→ SetDelayed
```

Theorem. If **t** is a binary homomorphism from a group **x** to a group **y**, then **range[t]** is invariant under the inversion function for **y**.

```
In[12]:= Map[not, SubstTest[and, implies[p1, p2],
            implies[p2, p3], implies[p3, p4], not[implies[p1, p4]],
            {p1 -> and[member[x, GROUPS], member[y, GROUPS], member[t, binhom[x, y]]],
             p2 -> equal[composite[t, inv[x]], composite[inv[y], t]],
             p3 → equal[image[t, domain[inv[x]]], image[inv[y], range[t]]],
             p4 → invariant[inv[y], range[t]]}]] // Reverse

Out[12]= or[not[member[t, binhom[x, y]]], not[member[x, GROUPS]],
           not[member[y, GROUPS]], subclass[image[inv[y], range[t]], range[t]]] == True

In[13]:= or[not[member[t_, binhom[x_, y_]]], not[member[x_, GROUPS]],
           not[member[y_, GROUPS]], subclass[image[inv[y_], range[t_]], range[t_]]] := True
```

Main Theorem. If **t** is a binary homomorphism from a group **x** to a group **y**, then **y ∘ id[range[t] × range[t]]** is a subgroup of **y**.

```
In[14]:= 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[t, binhom[x, y]]],
             p2 → subclass[image[y, cart[range[t], range[t]]], range[t]],
             p3 → member[e[y], range[t]], p4 → invariant[inv[y], range[t]],
             p5 -> member[composite[y, id[cart[range[t], range[t]]]], GROUPS]}]] // Reverse

Out[14]= or[member[composite[y, id[cart[range[t], range[t]]]], GROUPS],
           not[member[t, binhom[x, y]]], not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[15]:= or[member[composite[y_, id[cart[range[t_], range[t_]]]], GROUPS],
           not[member[t_, binhom[x_, y_]]],
           not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Lemma.

```
In[16]:= SubstTest[implies, and[member[t, GROUPS], subclass[t, x]],
           member[range[t], image[IMAGE[SECOND], intersection[GROUPS, P[x]]]],
           t → composite[x, id[cart[y, y]]]] // Reverse

Out[16]= or[member[image[x, cart[y, y]], image[IMAGE[SECOND], intersection[GROUPS, P[x]]]],
           not[member[composite[x, id[cart[y, y]]], GROUPS]]] == True

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

Theorem. The range of a binary homomorphism from a group **x** to a group **y** is the range of a subgroup of **y**.

```
In[18]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3],
           implies[p2, p4], (*implies[and[p3,p4],p5],*) not[implies[p1, p5]],
           {p1 -> and[member[x, GROUPS], member[y, GROUPS], member[t, binhom[x, y]]],
            p2 -> member[composite[y, id[cart[range[t], range[t]]]], GROUPS],
            p3 → equal[image[y, cartsq[range[t]]], range[t]],
            p4 -> member[image[y, cart[range[t], range[t]]],
              image[IMAGE[SECOND], intersection[GROUPS, P[y]]]], p5 →
             member[range[t], image[IMAGE[SECOND], intersection[GROUPS, P[y]]]]}]] // Reverse

Out[18]= or[member[range[t], image[IMAGE[SECOND], intersection[GROUPS, P[y]]]],
           not[member[t, binhom[x, y]]], not[member[x, GROUPS]], not[member[y, GROUPS]]] == True

In[19]:= or[member[range[t_], image[IMAGE[SECOND], intersection[GROUPS, P[y_]]]],
           not[member[t_, binhom[x_, y_]]],
           not[member[x_, GROUPS]], not[member[y_, GROUPS]]] := True
```

Corollary. (Eliminate the variable **t**.)

```
In[20]:= Map[equal[V, domain[#]] &,
          SubstTest[reify, t, case[or[member[range[t], v], not[member[t, u]],
             not[member[x, GROUPS]], not[member[y, GROUPS]]]],
            {u → binhom[x, y], v -> image[IMAGE[SECOND], intersection[GROUPS, P[y]]]}]]

Out[20]= or[not[member[x, GROUPS]], not[member[y, GROUPS]],
           subclass[image[IMAGE[SECOND], binhom[x, y]],
            image[IMAGE[SECOND], intersection[GROUPS, P[y]]]]] == True

In[21]:= or[not[member[x_, GROUPS]], not[member[y_, GROUPS]],
           subclass[image[IMAGE[SECOND], binhom[x_, y_]],
            image[IMAGE[SECOND], intersection[GROUPS, P[y_]]]]] := True
```

---

an example

The following is an example of the final corollary derived in the last section.

```
In[22]:= SubstTest[implies, and[member[x, GROUPS], member[y, GROUPS]],
           subclass[image[IMAGE[SECOND], binhom[x, y]], image[IMAGE[SECOND],
             intersection[GROUPS, P[y]]]], {x → INTADD, y → INTADD}] // Reverse

Out[22]= subclass[image[IMAGE[SECOND], binhom[INTADD, INTADD]],
           image[VERTSECT[INTDIV], Z]] == True
```

A slightly better result can be obtained.

Lemma.

```
In[23]:= composite[IMAGE[SECOND], INTTIMES] // FastReifNormality

Out[23]= composite[IMAGE[SECOND], INTTIMES] == composite[VERTSECT[INTDIV], id[Z]]
```

```
In[24]:= composite[IMAGE[SECOND], INTTIMES] := composite[VERTSECT[INTDIV], id[Z]]
```

Theorem.

```
In[25]:= ImageComp[IMAGE[SECOND], INTTIMES, V] // Reverse
Out[25]= image[IMAGE[SECOND], binhom[INTADD, INTADD]] == image[VERTSECT[INTDIV], Z]
```

```
In[26]:= image[IMAGE[SECOND], binhom[INTADD, INTADD]] := image[VERTSECT[INTDIV], Z]
```

Two additional rewrite rules of a similar nature can be derived in the same fashion.

Lemma

```
In[32]:= composite[DORA, INTTIMES] // FastReifNormality
Out[32]= composite[DORA, INTTIMES] == composite[LEFT[Z], VERTSECT[INTDIV], id[Z]]
```

```
In[33]:= composite[DORA, INTTIMES] := composite[LEFT[Z], VERTSECT[INTDIV], id[Z]]
```

Theorem.

```
In[35]:= ImageComp[DORA, INTTIMES, V] // Reverse
Out[35]= image[DORA, binhom[INTADD, INTADD]] == cart[set[Z], image[VERTSECT[INTDIV], Z]]
```

```
In[36]:= image[DORA, binhom[INTADD, INTADD]] := cart[set[Z], image[VERTSECT[INTDIV], Z]]
```