quotient relation for a quasiorder

Johan G. F. Belinfante
2009 December 29

```
In[1]:= SetDirectory["l:"]; << goedel.09dec28a; << tools.m

    :Package Title: goedel.09dec28a          2009 December 28 at 3:10 p.m.

    It is now:  2009 Dec 29 at 13:9

    Loading Simplification Rules

    TOOLS.M                                  Revised 2009 December 17

    weightlimit = 40
```

summary

A **quasiorder** (or preorder) is a reflexive and transitive relation. There is a canonical construction that associates to each quasiorder a partial order relation. All the results derived in this notebook are valid for any quasiorder, not just for sets. Among the named classes, there are ten quasiorders, mostly either equivalence relations or partial orders. But **INTDIV** and **ZN** are neither; the former is a set, while the latter is a proper class. The quasiorders **Q ∘ S** and **inverse[S] ∘ IMAGE[inverse[S]]** are other important examples of quasiorders that are proper classes and which are neither equivalence relations nor partial orders. To each quasiorder **q** there is an associated equivalence relation **e = q ⋂ inverse[q]**. In this notebook rewrite rules that connect a quasiorder and its associated equivalence relation are derived. The compound wrapper **trv[rfx[x]]** is especially useful for this purpose because the associated equivalence relation is simply **e = eqv[rfx[x]]**. The equivalence kernel for the canonical projection **p** associated with this equivalence relation **e** is its thin part: **inverse[p] ∘ p = thinpart[e]**. Rewrite rules about this thin part of **e** are derived. The canonical projection **p** can be used to construct a partial order **p ∘ q ∘ inverse[p]** sometimes described as the **quotient relation** of **q** modulo the equivalence **e**. If **e** is thin, one can recover the quaisorder **q** from its quotient partial order.

Reference. Problem 5.7 on page 38 in the following book.

```
In[2]:= "Karel Hrbacek and Thomas Jech, Introduction to Set Theory, Third
         Edition, Revised and Expanded, Marcel Dekker, Inc., New York, 1999.";
```

fix

The fixed-point classes of the quasiorder and its associated equivalence relation both simplify to **fix[x]**.

Theorem. Fixed-point class of the associated equivalence relation.

```
In[3]:= SubstTest[fix, intersection[t, inverse[t]], t → trv[rfx[x]]] // Reverse
Out[3]= fix[eqv[rfx[x]]] == fix[x]
In[4]:= fix[eqv[rfx[x_]]] := fix[x]
```

composites of q with q ⋂ inverse[q]

In this section four rewrite rules are derived about the composites of a quasiorder and its inverse with the associated equivalence relation.

Lemma.

```
In[5]:= SubstTest[implies, subclass[u, v], subclass[composite[t, u], composite[t, v]],
          {t → trv[rfx[x]], u → eqv[rfx[x]], v → trv[rfx[x]]}] // Reverse
Out[5]= subclass[composite[trv[rfx[x]], eqv[rfx[x]]], trv[rfx[x]]] == True
In[6]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[7]:= SubstTest[implies, subclass[u, v], subclass[composite[t, u], composite[t, v]],
          {t → trv[rfx[x]], u → id[fix[x]], v → eqv[rfx[x]]}] // Reverse
Out[7]= subclass[trv[rfx[x]], composite[trv[rfx[x]], eqv[rfx[x]]]] == True
In[8]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem. Composite of a quasiorder and its associated equivalence relation.

```
In[9]:= SubstTest[and, subclass[u, v], subclass[v, u],
         {u -> composite[trv[rfx[x]], eqv[rfx[x]]], v → trv[rfx[x]]}]
Out[9]= equal[composite[trv[rfx[x]], eqv[rfx[x]]], trv[rfx[x]]] == True
In[10]:= composite[trv[rfx[x_]], eqv[rfx[x_]]] := trv[rfx[x]]
```

Counterexample. One cannot omit the **rfx** wrapper.

```
In[11]:= equal[composite[trv[x], eqv[x]], trv[x]] /. x → PS
Out[11]= False
```

Other rewrite rules are easily obtained as corollaries.

Corollary. (Obtained by replacing **x** with its inverse.)

```
In[12]:= SubstTest[composite, trv[rfx[t]], eqv[rfx[t]], t → inverse[x]] // Reverse
Out[12]= composite[inverse[trv[rfx[x]]], eqv[rfx[x]]] == inverse[trv[rfx[x]]]
```

```
In[13]:= composite[inverse[trv[rfx[x_]]], eqv[rfx[x_]]] := inverse[trv[rfx[x]]]
```

Two more rewrite rules are obtained by using **DoubleInverse**.

Corollary.

```
In[14]:= composite[eqv[rfx[x]], trv[rfx[x]]] // DoubleInverse

Out[14]= composite[eqv[rfx[x]], trv[rfx[x]]] == trv[rfx[x]]

In[15]:= composite[eqv[rfx[x_]], trv[rfx[x_]]] := trv[rfx[x]]
```

Corollary.

```
In[16]:= composite[eqv[rfx[x]], inverse[trv[rfx[x]]]] // DoubleInverse

Out[16]= composite[eqv[rfx[x]], inverse[trv[rfx[x]]]] == inverse[trv[rfx[x]]]

In[17]:= composite[eqv[rfx[x_]], inverse[trv[rfx[x_]]]] := inverse[trv[rfx[x]]]
```

thin part simplification rules

Lemma.

```
In[18]:= SubstTest[subclass, fix[thinpart[t]], fix[t], t → eqv[rfx[x]]] // Reverse

Out[18]= subclass[fix[thinpart[eqv[rfx[x]]]], fix[x]] == True

In[19]:= subclass[fix[thinpart[eqv[rfx[x_]]]], fix[x_]] := True
```

Theorem. Simplification rule.

```
In[20]:= equal[composite[thinpart[eqv[rfx[x]]], id[fix[x]]], thinpart[eqv[rfx[x]]]]

Out[20]= True

In[21]:= composite[thinpart[eqv[rfx[x_]]], id[fix[x_]]] := thinpart[eqv[rfx[x]]]
```

Corollary. (Composite in the reverse order.)

```
In[22]:= composite[id[fix[x]], thinpart[eqv[rfx[x]]]] // DoubleInverse

Out[22]= composite[id[fix[x]], thinpart[eqv[rfx[x]]]] == thinpart[eqv[rfx[x]]]

In[23]:= composite[id[fix[x_]], thinpart[eqv[rfx[x_]]]] := thinpart[eqv[rfx[x]]]
```

the canonical projection

The canonical projection **p** associated with the equivalence relation **e = eqv[rfx[x]]** simplifies to

```
In[24]:= composite[id[complement[set[0]]], VERTSECT[e]] /. e → eqv[rfx[x]]

Out[24]= composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]]
```

The equivalence kernel of the canonical projection is the thin part of **e**.

```
In[25]:= composite[inverse[p], p] /. p -> composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]]

Out[25]= thinpart[eqv[rfx[x]]]
```

composites of q with thinpart[e]

The four rewrite rules about composites of a quasiorder and its inverse with the thin part of its associated equivalence relation derived in this section resemble those derived earlier for the equivalence relation itself. These rewrite rules are needed to show that the quotient relation of a quasiorder is antisymmetric.

Theorem. A simplification rule.

```
In[26]:= Assoc[trv[rfx[x]], eqv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]]]

Out[26]= composite[trv[rfx[x]], thinpart[eqv[rfx[x]]]] ==
          composite[trv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]]]

In[27]:= composite[trv[rfx[x_]], thinpart[eqv[rfx[x_]]]] :=
          composite[trv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]]]
```

Corollary. A similar rule with **x** replaced by its inverse.

```
In[28]:= SubstTest[composite, trv[rfx[t]], thinpart[eqv[rfx[t]]], t → inverse[x]] // Reverse

Out[28]= composite[inverse[trv[rfx[x]]], thinpart[eqv[rfx[x]]]] ==
          composite[inverse[trv[rfx[x]]], id[domain[VERTSECT[eqv[rfx[x]]]]]]

In[29]:= composite[inverse[trv[rfx[x_]]], thinpart[eqv[rfx[x_]]]] :=
          composite[inverse[trv[rfx[x]]], id[domain[VERTSECT[eqv[rfx[x]]]]]]
```

Two more such rules are derived using **DoubleInverse**.

Corollary.

```
In[30]:= composite[thinpart[eqv[rfx[x]]], inverse[trv[rfx[x]]]] // DoubleInverse

Out[30]= composite[thinpart[eqv[rfx[x]]], inverse[trv[rfx[x]]]] ==
          composite[id[domain[VERTSECT[eqv[rfx[x]]]]], inverse[trv[rfx[x]]]]

In[31]:= composite[thinpart[eqv[rfx[x_]]], inverse[trv[rfx[x_]]]] :=
          composite[id[domain[VERTSECT[eqv[rfx[x]]]]], inverse[trv[rfx[x]]]]
```

Corollary.

```
In[32]:= composite[thinpart[eqv[rfx[x]]], trv[rfx[x]]] // DoubleInverse

Out[32]= composite[thinpart[eqv[rfx[x]]], trv[rfx[x]]] ==
          composite[id[domain[VERTSECT[eqv[rfx[x]]]]], trv[rfx[x]]]

In[33]:= composite[thinpart[eqv[rfx[x_]]], trv[rfx[x_]]] :=
          composite[id[domain[VERTSECT[eqv[rfx[x]]]]], trv[rfx[x]]]
```

the quotient relation

Observation. The quotient relation **p ∘ q ∘ inverse[p]** simplifies as follows:

```
In[34]:= composite[p, q, inverse[p]] /.
          {p -> composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]], q → trv[rfx[x]]}

Out[34]= composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]
```

In this section it is shown that this quotient relation is transitive and reflexive.

Lemma. Simplification rule.

```
In[35]:= equal[image[trv[rfx[x]], complement[fix[x]]], 0]

Out[35]= True

In[36]:= image[trv[rfx[x_]], complement[fix[x_]]] := 0
```

Lemma.

```
In[37]:= SubstTest[implies, subclass[u, v], subclass[composite[t, u, w], composite[t, v, w]],
           {t -> composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]]],
            u → thinpart[eqv[rfx[x]]], v → eqv[rfx[x]],
            w → composite[trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]}] // Reverse

Out[37]= subclass[
           composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]],
            trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]], composite[
            VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]] == True

In[38]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem.

```
In[39]:= SubstTest[subclass, composite[t, t], t,
          t -> composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]]

Out[39]= TRANSITIVE[composite[VERTSECT[eqv[rfx[x]]],
            trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]] == True

In[40]:= TRANSITIVE[composite[VERTSECT[eqv[rfx[x_]]],
            trv[rfx[x_]], inverse[VERTSECT[eqv[rfx[x_]]]]]] := True
```

To derive the reflexive property a formula is needed for the fixed point class of the quotient relation.

Lemma.

```
In[41]:= (SubstTest[implies, subclass[u, v], subclass[image[t, u], image[t, v]],
           {t → intersection[composite[w, FIRST], composite[w, SECOND]],
            u → id[fix[x]], v → trv[rfx[x]]}] /. w → VERTSECT[eqv[rfx[x]]]) // Reverse

Out[41]= subclass[image[VERTSECT[eqv[rfx[x]]], fix[x]], fix[composite[
           VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]]] == True

In[42]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[43]:= SubstTest[subclass, fix[composite[u, v]],
          range[u], {u → composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]],
           v -> composite[trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]}] // Reverse

Out[43]= subclass[
          fix[composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]],
          image[VERTSECT[eqv[rfx[x]]], fix[x]]] == True

In[44]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem. Simplification rule for the fixed point class of the quotient relation.

```
In[45]:= SubstTest[and, subclass[u, v], subclass[v, u], {u ->
           fix[composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]],
          v -> image[VERTSECT[eqv[rfx[x]]], fix[x]]}]

Out[45]= equal[
          fix[composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]],
          image[VERTSECT[eqv[rfx[x]]], fix[x]]] == True

In[46]:= fix[composite[VERTSECT[eqv[rfx[x_]]], trv[rfx[x_]],
           inverse[VERTSECT[eqv[rfx[x_]]]]]] := image[VERTSECT[eqv[rfx[x]]], fix[x]]
```

Corollary. The quotient relation is reflexive.

```
In[47]:= SubstTest[subclass, t, cartsq[fix[t]],
          t -> composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]]

Out[47]= REFLEXIVE[composite[VERTSECT[eqv[rfx[x]]],
           trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]] == True

In[48]:= REFLEXIVE[composite[VERTSECT[eqv[rfx[x_]]],
           trv[rfx[x_]], inverse[VERTSECT[eqv[rfx[x_]]]]]] := True
```

antisymmetry

To derive the antisymmetry property of the quotient relation, various simplification rules need to be established.

Lemma. Simplification rule.

```
In[49]:= equal[image[eqv[rfx[x]], complement[fix[x]]], 0]

Out[49]= True

In[50]:= image[eqv[rfx[x_]], complement[fix[x_]]] := 0
```

Lemma. Simplification rule.

```
In[51]:= equal[image[inverse[trv[rfx[x]]], complement[fix[x]]], 0]

Out[51]= True

In[52]:= image[inverse[trv[rfx[x_]]], complement[fix[x_]]] := 0
```

Theorem. A general simplification rule.

```
In[53]:= SubstTest[intersection, composite[id[t], u], composite[id[t], v],
          {t → range[x], u → composite[x, y], v → composite[x, z]}]

Out[53]= composite[id[range[x]], intersection[composite[x, y], composite[x, z]]] ==
          intersection[composite[x, y], composite[x, z]]

In[54]:= composite[id[range[x_]], intersection[composite[x_, y_], composite[x_, z_]]] :=
          intersection[composite[x, y], composite[x, z]]
```

Theorem. The quotient relation is antisymmetric.

```
In[55]:= Map[inverse,
          Map[composite[VERTSECT[eqv[rfx[x]]], #, inverse[VERTSECT[eqv[rfx[x]]]]] &,
           SubstTest[intersection,
            composite[inverse[VERTSECT[t]], VERTSECT[t], u, inverse[VERTSECT[t]], VERTSECT[t]],
            composite[inverse[VERTSECT[t]], VERTSECT[t], v, inverse[VERTSECT[t]], VERTSECT[t]],
            {t → eqv[rfx[x]], u → trv[rfx[x]], v → inverse[trv[rfx[x]]]}]]]

Out[55]= intersection[composite[VERTSECT[eqv[rfx[x]]],
            inverse[trv[rfx[x]]], inverse[VERTSECT[eqv[rfx[x]]]]],
           composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]] ==
          id[image[VERTSECT[eqv[rfx[x]]], fix[x]]]

In[56]:= intersection[composite[VERTSECT[eqv[rfx[x_]]],
            inverse[trv[rfx[x_]]], inverse[VERTSECT[eqv[rfx[x_]]]]],
           composite[VERTSECT[eqv[rfx[x_]]], trv[rfx[x_]], inverse[VERTSECT[eqv[rfx[x_]]]]]] :=
          id[image[VERTSECT[eqv[rfx[x]]], fix[x]]]
```

Main Theorem. The quotient relation of a quasiorder modulo its associated equivalence relation is a partial order.

```
In[57]:= SubstTest[and, REFLEXIVE[t], ANTISYMMETRIC[t], TRANSITIVE[t],
          t -> composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]]

Out[57]= PARTIALORDER[composite[VERTSECT[eqv[rfx[x]]],
            trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]] == True

In[58]:= PARTIALORDER[composite[VERTSECT[eqv[rfx[x_]]],
            trv[rfx[x_]], inverse[VERTSECT[eqv[rfx[x_]]]]]] := True
```

---

recovery of a quasiorder from its quotient

Although the quotent relation for any quasiorder is always a partial order, this is mainly useful when the associated equivalence relation is thin. When **e** is thin, one can recover **q** from its quotient partial order **r = p ◦ q ◦ inverse[p]**. Additional rewrite rules are needed to simplify certain expressions that arise in this recovery process.

Lemma.

```
In[59]:= Assoc[trv[rfx[x]], inverse[composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]]],
           composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]]] // Reverse

Out[59]= composite[trv[rfx[x]], id[intersection[domain[VERTSECT[eqv[rfx[x]]]], fix[x]]]] ==
          composite[trv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]]]

In[60]:= composite[trv[rfx[x_]], id[intersection[domain[VERTSECT[eqv[rfx[x_]]]], fix[x_]]]] :=
          composite[trv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]]]
```

Theorem.

```
In[61]:= SubstTest[composite, inverse[E],
           id[image[VERTSECT[eqv[t]], fix[eqv[t]]]], t → rfx[x]] // Reverse

Out[61]= composite[inverse[E], id[image[VERTSECT[eqv[rfx[x]]], fix[x]]]] ==
          composite[id[fix[x]], inverse[VERTSECT[eqv[rfx[x]]]]]

In[62]:= composite[inverse[E], id[image[VERTSECT[eqv[rfx[x_]]], fix[x_]]]] :=
          composite[id[fix[x]], inverse[VERTSECT[eqv[rfx[x]]]]]
```

Theorem.

```
In[63]:= SubstTest[composite, id[image[VERTSECT[eqv[t]], fix[eqv[t]]]], E, t → rfx[x]] // Reverse

Out[63]= composite[id[image[VERTSECT[eqv[rfx[x]]], fix[x]]], E] ==
          composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]]

In[64]:= composite[id[image[VERTSECT[eqv[rfx[x_]]], fix[x_]]], E] :=
          composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]]
```

The recovery of the quasiorder **q** from its quotient partial order **r** succeeds when the associated equivalence relation **e** is thin; in that case **domain[VERTSECT[eqv[rfx[x]]] = V**. This is the case, for example, then **q** is a set.

```
In[65]:= composite[inverse[p], r, p] /. {p -> composite[VERTSECT[eqv[rfx[x]]], id[fix[x]]],
           r -> composite[VERTSECT[eqv[rfx[x]]], trv[rfx[x]], inverse[VERTSECT[eqv[rfx[x]]]]]}

Out[65]= composite[id[domain[VERTSECT[eqv[rfx[x]]]]],
          trv[rfx[x]], id[domain[VERTSECT[eqv[rfx[x]]]]]]
```