isomorphism is an equivalence relation

Johan G. F. Belinfante
2009 August 16

```
In[1]:= SetDirectory["l:"]; << goedel.09aug15a;<< tools.m

        :Package Title: goedel.09aug15a              2009 August 15 at 5:25 a.m.

        It is now:  2009 Aug 16 at 10:27

        Loading Simplification Rules

        TOOLS.M                                      Revised 2009 July 2

        weightlimit = 40
```

summary

An **isomorphism** in an (arrows-only) category **cat[x]** is an invertible morphism, that is, a morphism **u** for which there exists another morphism **v** such that both **u · v** and **v · u** are identity morphisms. The class of isomorphisms **domain[inv[cat[x]]]** is a subclass of the class **range[cat[x]]** of all morphisms. In an arrows-only category **cat[x]**, the role of objects is played by identity morphisms. The class **ids[cat[x]]** of all identity morphisms is a subclass of the class of isomorphisms.

Recall that the relation **domain[hom[cat[x]]]** consists of all pairs of identities **u**, **v** for which there is a morphism from **u** to **v**. This transitive relation is a subclass of the cartesian square of the class **ids[cat[x]]**. The focus in this notebook is a certain subrelation of the relaton **domain[hom[cat[x]]]**.

Two identity morphisms **u** and **v** are **isomorphic** if there exists an invertible morphism from **u** to **v**. This isomorphism relation on the class **ids[cat[x]]** is given by the expression **image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]**. It will be shown below that this isomorphism relation on identity morphisms is an equivalence relation. An interesting feature of the derivation is that it can be accomplished entirely without introducing variables for morphisms.

derivation

The class of **domain[inv[cat[x]]** of isomorphisms is binary-closed under the composition law **cat[x]**. This fact can be strengthened to an equation and made into a rewrite rule:

Theorem. The class of products of isomorphisms is equal to the class of isomorphisms.

```
In[2]:= SubstTest[implies, functor[id[t], cat[w], cat[x]],
          equal[range[cat[w]], t], {t → domain[inv[cat[x]]],
           w → composite[cat[x], id[cartsq[domain[inv[cat[x]]]]]]}] // Reverse

Out[2]= equal[domain[inv[cat[x]]],
          image[cat[x], cart[domain[inv[cat[x]]], domain[inv[cat[x]]]]]] == True

In[3]:= image[cat[x_], cart[domain[inv[cat[x_]]], domain[inv[cat[x_]]]]] := domain[inv[cat[x]]]
```

domain[hom[cat[x]]] is transitive

Lemma.

```
In[5]:= Map[composite[#, cross[DUP, DUP]] &,
         (composite[SWAP, RIF, cross[cross[u, v], cross[u, v]]] // ReifTriNormality) /.
          {u → dom[cat[x]], v → cod[cat[x]]}]

Out[5]= composite[SWAP, RIF, cross[inverse[hom[cat[x]]], inverse[hom[cat[x]]]]] ==
         composite[SWAP, cross[cod[cat[x]], dom[cat[x]]], id[domain[cat[x]]]]

In[6]:= composite[SWAP, RIF, cross[inverse[hom[cat[x_]]], inverse[hom[cat[x_]]]]] :=
         composite[SWAP, cross[cod[cat[x]], dom[cat[x]]], id[domain[cat[x]]]]
```

Lemma. Simplification rule.

```
In[7]:= ImageComp[composite[SWAP, RIF],
         cross[inverse[hom[cat[x]]], inverse[hom[cat[x]]]], cartsq[domain[inv[cat[x]]]]]

Out[7]= composite[cod[cat[x]], id[domain[inv[cat[x]]]],
          inverse[domain[cat[x]]], id[domain[inv[cat[x]]]], inverse[dom[cat[x]]]] ==
         composite[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]],
          image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]]

In[8]:= composite[cod[cat[x_]], id[domain[inv[cat[x_]]]],
          inverse[domain[cat[x_]]], id[domain[inv[cat[x_]]]], inverse[dom[cat[x_]]]] :=
         composite[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]],
          image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]]
```

Theorem. The isomorphism relation is idempotent.

```
In[9]:= ImageComp[inverse[hom[cat[x]]], cat[x], cartsq[domain[inv[cat[x]]]]]

Out[9]= composite[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]],
          image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]] ==
         image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]

In[10]:= composite[image[inverse[hom[cat[x_]]], domain[inv[cat[x_]]]],
           image[inverse[hom[cat[x_]]], domain[inv[cat[x_]]]]] :=
          image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]
```

Lemma. Simplification rule.

```
In[11]:= equal[composite[Id, image[inverse[hom[x]], y]], image[inverse[hom[x]], y]]

Out[11]= True

In[12]:= composite[Id, image[inverse[hom[x_]], y_]] := image[inverse[hom[x]], y]
```

Corollary. The isomorphism relation is transitive.

```
In[13]:= SubstTest[subclass, composite[t, t], t,
          t → image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]]

Out[13]= TRANSITIVE[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]] == True

In[14]:= TRANSITIVE[image[inverse[hom[cat[x_]]], domain[inv[cat[x_]]]]] := True
```

symmetry

The **cod** of the inverse of an invertible morphism is the **dom** of the morphism.

Lemma.

```
In[15]:= SubstTest[implies, subclass[u, v], subclass[composite[t, u], composite[t, v]],
           {t → cod[cat[x]], u → inv[cat[x]], v → domain[cat[x]]}] // Reverse

Out[15]= subclass[composite[cod[cat[x]], inv[cat[x]]], dom[cat[x]]] == True

In[16]:= (% /. x → x_) /. Equal → SetDelayed
```

Since these are all functions, one can replace this with an equation.

Theorem. Equation for the **cod** of an inverse.

```
In[19]:= SubstTest[equal, v, composite[funpart[u], id[domain[v]]],
           {u → dom[cat[x]], v -> composite[cod[cat[x]], inv[cat[x]]]}] // Reverse

Out[19]= equal[composite[cod[cat[x]], inv[cat[x]]],
           composite[dom[cat[x]], id[domain[inv[cat[x]]]]]] == True

In[21]:= composite[cod[cat[x_]], inv[cat[x_]]] :=
          composite[dom[cat[x]], id[domain[inv[cat[x]]]]]
```

An application of duality provides the following corollary.

Corollary. Equation for the **dom** of an inverse.

```
In[23]:= SubstTest[composite, cod[cat[t]], inv[cat[t]], t → flip[cat[x]]] // Reverse

Out[23]= composite[dom[cat[x]], inv[cat[x]]] == composite[cod[cat[x]], id[domain[inv[cat[x]]]]]

In[24]:= composite[dom[cat[x_]], inv[cat[x_]]] :=
          composite[cod[cat[x]], id[domain[inv[cat[x]]]]]
```

Theorem. The function **inv[cat[x]]** is an involution.

```
In[27]:= SubstTest[composite, funpart[t], inverse[funpart[t]], t -> inv[cat[x]]] // Reverse

Out[27]= composite[inv[cat[x]], inv[cat[x]]] == id[domain[inv[cat[x]]]]

In[28]:= composite[inv[cat[x_]], inv[cat[x_]]] := id[domain[inv[cat[x]]]]
```

Theorem. An equation for the composite of **inv[cat[x]]** and **hom[cat[x]]**.

```
In[30]:= Map[composite[inv[cat[x]], inverse[#], SWAP] &, Assoc[
           cross[dom[cat[x]], cod[cat[x]]], cross[inv[cat[x]], inv[cat[x]]], DUP]] // Reverse

Out[30]= composite[inv[cat[x]], hom[cat[x]]] ==
          composite[id[domain[inv[cat[x]]]], hom[cat[x]], SWAP]

In[31]:= composite[inv[cat[x_]], hom[cat[x_]]] :=
          composite[id[domain[inv[cat[x]]]], hom[cat[x]], SWAP]
```

Theorem. Isomorphism is a symmetric relation.

```
In[32]:= IminComp[inv[cat[x]], hom[cat[x]], V]

Out[32]= inverse[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]] ==
          image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]

In[33]:= inverse[image[inverse[hom[cat[x_]]], domain[inv[cat[x_]]]]] :=
          image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]
```

Corollary. Isomorphism is an equivalence relation.

```
In[34]:= SubstTest[and, SYMMETRIC[t], TRANSITIVE[t],
          t -> image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]]

Out[34]= EQUIVALENCE[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]] == True

In[35]:= EQUIVALENCE[image[inverse[hom[cat[x_]]], domain[inv[cat[x_]]]]] := True
```

fix point class

Lemma.

```
In[38]:= SubstTest[implies, subclass[u, v], subclass[fix[u], fix[v]],
           {u -> image[inverse[hom[cat[x]]], domain[inv[cat[x]]]],
            v -> cart[ids[cat[x]], ids[cat[x]]]}] // Reverse

Out[38]= subclass[fix[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]], ids[cat[x]]] == True

In[39]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[44]:= SubstTest[implies, subclass[u, v], subclass[image[t, u], image[t, v]],
           {t → inverse[hom[cat[x]]], u → ids[cat[x]], v → domain[inv[cat[x]]]}] // Reverse

Out[44]= subclass[ids[cat[x]], fix[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]]] == True

In[45]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem. Every identity morphism is isomorphic to itself.

```
In[46]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> fix[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]], v → ids[cat[x]]}]

Out[46]= equal[fix[image[inverse[hom[cat[x]]], domain[inv[cat[x]]]]], ids[cat[x]]] == True

In[48]:= fix[image[inverse[hom[cat[x_]]], domain[inv[cat[x_]]]]] := ids[cat[x]]
```