The hom functors

Definition. If $\mathcal{C}$ is a category, there are correspondences $\mathcal{C}(X,-)$ and $\mathcal{C}(-,Y)$ defined as follows:

- $\mathcal{C}({\color{red}X},-)$ comprises
  - a function sending each *object* $A \in \mathcal{C}_0$ to the *set* of all morphisms $f : {\color{red}X} \to A$.
  - a function sending each *morphism* $u : A \to A'$ in $\mathcal{C}$ to a function $\mathcal{C}(X,A) \to \mathcal{C}(X,A')$, defined taking $f : X \to A$ to $u \circ f : X \xrightarrow{f} A \xrightarrow{u} A'$.
  - The equalities $\mathcal{C}(X, g \circ f) = \mathcal{C}(X,g) \circ \mathcal{C}(X,f)$ and $\mathcal{C}(X, id_A) = id_{\mathcal{C}(X,A)}$ hold.
- $\mathcal{C}(-,{\color{blue}Y})$ comprises
  - a function sending each *object* $A \in \mathcal{C}_0$ to the *set* of all morphisms $f : A \to {\color{blue}Y}$.
  - a function sending each *morphism* $u : A \to A'$ in $\mathcal{C}$ to a function $\mathcal{C}(A',Y) \to \mathcal{C}(A,Y)$, defined taking $f : A' \to Y$ to $f \circ u : A \xrightarrow{u} A' \xrightarrow{f} Y$.
  - The equalities $\mathcal{C}(g \circ f, Y) = \mathcal{C}(f,Y) \circ \mathcal{C}(g,Y)$ and $\mathcal{C}(id_A, Y) = id_{\mathcal{C}(A,Y)}$ hold.

$\mathcal{C}(X,-)$ and $\mathcal{C}(-,Y)$ are the (covariant and contravariant) "hom functors" on the objects X and Y.

Some examples of $\mathcal{C}(X,-)$ and $\mathcal{C}(-,Y)$ in categories we know.

- In $\mathcal{C} = \mathrm{Set}_t$, let $X = \{0,1\}$; then $\mathrm{Set}_t(A, \{0,1\})$ is the set of all subsets of A, and $\mathrm{Set}_t(u, \{0,1\})$ is the function sending a subset $U \subseteq A'$ to the subset $u^{-1}A' := \{a \in A \mid u(a) \in U\}$.
- In $\mathcal{C} = \mathrm{Set}_p$, let $X = \{\mathtt{true}, \mathtt{false}, \bot\}$; then $\mathrm{Set}_p(A, X)$ is the set of predicates that might fail, defined on A.
- In $\mathcal{C} = \mathrm{Graph}$, fix a (small! Say, with three vertices) graph H and describe the set of graph homomorphisms $G \to H$.
- In $\mathcal{C} = (\mathbb{N}, +)$, there is only one object $X = \{\bullet\}$. Then, $(\mathbb{N},+)(\bullet, -)$ sends $\bullet$ to the set of morphisms $\bullet \to \bullet$, i.e. to $\mathbb{N} = \{0,1,2,3,\dots\}$ and a morphism n to the function $\_ + n : \mathbb{N} \to \mathbb{N}$.
- In $\mathcal{C} = (\mathbb{N}, \leq)$, let $n = 7$. Then $(\mathbb{N}, \leq)(7, -)$ sends $0, 1, \dots, 6$ to the set of morphisms $0, 1, \dots, 6, 7 \to 7$ (i.e., to a single morphism " `0,1,...,6,7 ≤ 7` "), and all number $8, 9, \dots$ to the empty set.

Isomorphism of functors $\iff$ isomorphism of objects

Since $\mathcal{C}(X,-)$ and $\mathcal{C}(-,Y)$ are defined in terms of *families* of sets

$$\{\mathcal{C}(A,Y) \mid A \in \mathcal{C}_0\} \qquad \{\mathcal{C}(X,B) \mid B \in \mathcal{C}_0\}$$

it follows that the correct notion of isomorphism between $\mathcal{C}(X,-)$ and $\mathcal{C}(X',-)$ must take them all into consideration, and thus consists of a family of isomorphisms (=bijective functions)

$$\mathcal{C}(X,A) \xrightarrow{\alpha_A} \mathcal{C}(X',A)$$

all of which are "polymorphic" in the "variable" A.

This means that they behave coherently according the possibility to define a function $\mathcal{C}(X,u) : \mathcal{C}(X,A) \to \mathcal{C}(X,A')$ out of a function $u : A \to A'$, and put together both in a square diagram

$$\begin{array}{ccc} \mathcal{C}(X,A) & \xrightarrow{\alpha_A} & \mathcal{C}(X',A) \\ {\scriptstyle \mathcal{C}(X,u)}\downarrow & & \downarrow{\scriptstyle \mathcal{C}(X',u)} \\ \mathcal{C}(X,A') & \xrightarrow[\alpha_{A'}]{} & \mathcal{C}(X',A') \end{array}$$

declaring that $\alpha_{A'} \circ \mathcal{C}(X,u) = \mathcal{C}(X',u) \circ \alpha_A$.

Polymorphism is (almost) always present when one defines a widget depending on a parameter, and wants it to behave well under change of that parameter.

For example: let `A={red,green,blue}` be a set; let `B={cabbage, goat, wolf}` be another set. Obviously $A \cong B$ (for example, under the map `{red -> cabbage, green -> goat, blue -> wolf}`). Then, recall how the `List<A>` type is defined in CRUST:

```
type List<A> = nil()
             | cons(A, List<A>)
```

If we want to express the fact that `List<A>` comes equipped with a function `h : A -> List<A>` sending `a : A` to `cons(a, nil())` for all `A`'s, polymorphically in `A` , we have to require that, for every way of transforming the type `A` on which `List<A>` depends, using `f : A -> B` , we obtain that

```
List<f> (h a) = h (f a)
```

which means `List<f> (cons(a, nil()) = cons(f(a), nil())` (which is true):

$$\begin{array}{ccc} A & \xrightarrow{h_A} & \text{List}\langle A\rangle \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \text{List}\langle f\rangle} \\ B & \xrightarrow[h_B]{} & \text{List}\langle B\rangle \end{array}$$

Polymorphism (or in its mathematical name, *naturality*) is a powerful generalization of this idea, that applies to all correspondences like $\mathcal{C}(X,-)$ and $\mathcal{C}(-,Y)$ and $\text{List}\langle -\rangle$.

The first important observation

> If X is isomorphic to Y, then C(-, X) is isomorphic to C(-, Y).

There is an easy way to generate an isomorphism $\mathcal{C}(-,Y)\cong\mathcal{C}(-,Y')$, in the above sense [=a "natural" isomorphism] from a pre-existing isomorphism $Y\cong Y'$ in $\mathcal{C}$, i.e. from a pair

$$u : Y \to Y', \quad v : Y' \to Y$$

such that $id_Y = Y \xrightarrow{u} Y' \xrightarrow{v} Y$ and $id_{Y'} = Y' \xrightarrow{v} Y \xrightarrow{u} Y'$. Indeed,

- the fact that the functors $\mathcal{C}(A,-)$ exist, yields for every A a function $\mathcal{C}(A,Y)\xrightarrow{\mathcal{C}(A,u)}\mathcal{C}(A,Y')$, and $\mathcal{C}(A,Y')\xrightarrow{\mathcal{C}(A,v)}\mathcal{C}(A,Y)$. It must then be that

$$\begin{aligned} \mathcal{C}(A,u)\circ\mathcal{C}(A,v) &= \mathcal{C}(A,u\circ v) \\ &= \mathcal{C}(A,id_{Y'}) \\ &= id_{\mathcal{C}(A,Y')} \\ \mathcal{C}(A,v)\circ\mathcal{C}(A,u) &= \mathcal{C}(A,v\circ u) \\ &= \mathcal{C}(A,id_Y) \\ &= id_{\mathcal{C}(A,Y)} \end{aligned}$$

  so that $\mathcal{C}(A,Y)$ and $\mathcal{C}(A,Y')$ are bijective functions, inverse to each other.
- Now, if $f : A \to A'$ is given, polymorphism/naturality for $\hat{u}_X := \mathcal{C}(X,u)$ is expressed by the fact that the square

$$\begin{array}{ccc} \mathcal{C}(A',Y) & \xrightarrow{\hat{u}_{A'}} & \mathcal{C}(A',Y') \\ {\scriptstyle \mathcal{C}(f,Y)}\downarrow & & \downarrow{\scriptstyle \mathcal{C}(f,Y')} \\ \mathcal{C}(A,Y) & \xrightarrow[\hat{u}_A]{} & \mathcal{C}(A,Y') \end{array}$$

  commutes, which means that

$$u\circ(\_\circ f) = (u\circ\_)\circ f.$$

  This is ensured by the fact that composition is associative! Similarly, for v,

$$\begin{array}{ccc} \mathcal{C}(A',Y') & \xrightarrow{\hat{v}_{A'}} & \mathcal{C}(A',Y) \\ {\scriptstyle \mathcal{C}(f,Y)}\downarrow & & \downarrow{\scriptstyle \mathcal{C}(f,Y')} \\ \mathcal{C}(A,Y') & \xrightarrow[\hat{v}_A]{} & \mathcal{C}(A,Y) \end{array}$$

(observe that one can "glue" together the two squares

$$\begin{array}{ccccc} \mathcal{C}(A',Y) & \xrightarrow{\hat{u}_{A'}} & \mathcal{C}(A',Y') & \xrightarrow{\hat{v}_{A'}} & \mathcal{C}(A',Y) \\ {\scriptstyle \mathcal{C}(f,Y)}\downarrow & & \downarrow{\scriptstyle \mathcal{C}(f,Y')} & & \downarrow{\scriptstyle \mathcal{C}(f,Y)} \\ \mathcal{C}(A,Y) & \xrightarrow[\hat{u}_A]{} & \mathcal{C}(A,Y') & \xrightarrow[\hat{v}_A]{} & \mathcal{C}(A,Y) \end{array}$$

to get the identities as horizontal compositions.)

The second (more) important observation

> C(-, X) iso to C(-, Y) implies that X is iso to Y.

Or in other words, *all* polymorphic/natural isomorphisms $\alpha_A : \mathcal{C}(A, X) \cong \mathcal{C}(A, Y)$ originates from an isomorphism $X \cong Y$ in $\mathcal{C}$.

This is important because

- it allows to derive isomorphism of objects from isomorphisms of functors using those objects as parameters
- it allows to formally reason about isomorphism: in order to prove that $X \cong Y$ (abstract objects that you can touch), fix an object A and look at the sets $\mathcal{C}(A, X), \mathcal{C}(A, Y)$ (concrete sets that you can enumerate): if these sets are *polymorphically* isomorphic, then X must be isomorphic to Y.

The proof will be very fun.

Suppose there is a polymorphic isomorphism $\alpha_A : \mathcal{C}(A, X) \cong \mathcal{C}(A, Y)$. Spelled out: we have α given as a family of bijections $\alpha_A : \mathcal{C}(A, X) \to \mathcal{C}(A, Y)$ with inverse $\alpha_A^{-1} : \mathcal{C}(A, Y) \to \mathcal{C}(A, X)$, and such that for every $f : A \to A'$ the squares

$$
\begin{array}{ccccccc}
\mathcal{C}(A',X) & \xrightarrow{\alpha_{A'}} & \mathcal{C}(A',Y) & & \mathcal{C}(A',X) & \xleftarrow{\alpha_{A'}^{-1}} & \mathcal{C}(A',Y) \\
\downarrow & & \downarrow & & \downarrow & & \downarrow \\
\mathcal{C}(A,X) & \xrightarrow[\alpha_A]{} & \mathcal{C}(A,Y) & & \mathcal{C}(A,X) & \xleftarrow[\alpha_A^{-1}]{} & \mathcal{C}(A,Y)
\end{array}
$$

commute. This means:

- $\forall f^{A \to A'} g^{A' \to X} \ : \ \alpha_A(g \circ f) = \alpha_{A'}(g) \circ f$
- $\forall f^{A \to A'} h^{A' \to Y} \ : \ \alpha_A^{-1}(h \circ f) = \alpha_{A'}^{-1}(h) \circ f$

Then,

- look at the component $\alpha_X : \mathcal{C}(X, X) \to \mathcal{C}(X, Y)$. The set $\mathcal{C}(X, X)$ is not empty, so $\mathcal{C}(X, Y)$ contains at least one element, $\alpha_X(id_X) =: u$.
- look at the component $\alpha_Y^{-1} : \mathcal{C}(Y, Y) \to \mathcal{C}(Y, X)$. The set $\mathcal{C}(Y, Y)$ is not empty, so $\mathcal{C}(Y, X)$ contains at least one element, $\alpha_Y^{-1}(id_Y) =: v$.

Now,

$$
\begin{aligned}
u \circ v &= \alpha_X(id_X) \circ \alpha_Y^{-1}(id_Y) \\
&= \alpha_X(id_X) \circ f \\
&= \alpha_X(id_X) \circ f \\
&= \alpha_Y(id_X \circ \alpha_Y^{-1}(id_Y)) \\
&= \alpha_Y(\alpha_Y^{-1}(id_Y)) \\
&= id_Y \\
v \circ u &= \alpha_Y^{-1}(id_Y) \circ \alpha_X(id_X) \\
&= \alpha_Y^{-1}(id_Y) \circ f \\
&= \alpha_X^{-1}(id_Y \circ f) \\
&= \alpha_X^{-1}(id_Y \circ \alpha_X(id_X)) \\
&= \alpha_X^{-1}(\alpha_X(id_X)) \\
&= id_X
\end{aligned}
$$