Understanding Natural Transformations

Brendan Good

2018-04-11 Wed 19:44

Natural transformations are the gatekeeper to the rest of category theory. Once you understand them, adjoint functors and even the (in)famous monads unveil themselves. In fact, the "correct" notion of two categories being equivalent (unsurprisingly called an equivalence of categories) relies on natural transformations.

Despite this notion being critical, I've only really heard two explanations of them (both unenlightening):

- A natural transformations are "coordinate free transformations" (what does that even mean)
- A natural transformation is a "homomorphism of functors" (a bit too high-level to make sense "out of the box")

As is often the case for me, the best approach for me was stare at the diagram and figure out the intuition for myself.

In this post, I'll talk about my intuition for natural transformations and, roughly, how I came to it. Although the understanding I ultimately arrived at is pretty much identical to the "homomorphism of functors" viewpoint, this post unwinds that analogy a bit more.

What is a homomorphism, really?

At its core, a homomorphism is just a map which preserves the structure of its argument. For example, a group homomorphism is a map which sends groups to groups, a "category homomorphism" (functor) sends categories to categories, etc. In fact, in a general category C, if, for two objects $X, Y \in ob(C)$[1], there is a morphism $f : X \to Y$ this means that Y has

[1] Here I take $ob(C)$ to mean the objects of the category C.

the same structure as X. Even though that last point sounds like a needless abstraction, that insight will be useful later.

With that in mind, what is the structure of a functor? It's a strange question with a straightforward answer: you apply it. You give me an object from a category C, I'll give you an object from category a D. Similarly, you give me a morphism from C, I'll give you a morphism from D.

Structure of morphisms

Before we get into natural transformations, it will be useful to discuss when we consider two morphisms to have the same structure. Consider two morphisms in a category C, $f : X \to Y$ and $g : Z \to W$. What is a reasonable criterion to say that g has the same structure as f? A morphism consists of 2 things, a domain and a codomain.

Recall that in a category C, object Y "has the same structure" as X if there exists a morphism $f : X \to Y$. With that in mind, we might say that a reasonable definition would be that two morphisms have the same structure such that the domains and the codomains have the same structure. In other words, we want a diagram that looks like this (not necessarily commutative):

$$\begin{array}{ccc} X & \xrightarrow{j} & Z \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ Y & \xrightarrow{k} & W \end{array}$$

Diagram A

I want to re-emphasize that, at this point, we do **not** require that this diagram be commutative (that is do say $j \circ g$ does not necessarily equal $f \circ k$).

This looks fine at first glance, since we have that the domain and the codomain have the same structure, but consider the following (silly) example:

$$\begin{array}{ccc} \mathbb{Z} & \xrightarrow{x \mapsto x} & \mathbb{Z} \\ {\scriptstyle x \mapsto x \mod 2}\downarrow & & \downarrow{\scriptstyle x \mapsto x \mod 3} \\ \mathbb{Z}/2\mathbb{Z} & \xrightarrow{0} & \mathbb{Z}/3\mathbb{Z} \end{array}$$

If we compose the top arrow with the right arrow, we get the morphism $x \mapsto x \mod 3$. However, when we compose the left arrow with the bottom

arrow, we get the morphism $x \mapsto 0$. Because these resulting morphisms are different, it is possible to recover whether we mapped over via the domain or the codomain depending on what the resulting morphism is. This seems to indicate that the domain and codomain somehow have a different structure.

That is to say, we want to map from either the domain or the codomain and achieve the same morphism. Therefore, we say that g has the same structure as f if diagram A (above) is commutative.

Natural transformations

The definition

A natural transformation between two functors $F, G : C \to D$ associates to each object $X \in ob(C)$ a morphism $\eta_X : F(X) \to G(X)$ such that the following diagram commutes (which is a 90 degree rotation from the one listed on Wikipedia).

$$\begin{array}{ccc} F(X) & \xrightarrow{\eta_X} & G(X) \\ {\scriptstyle F(f)}\downarrow & & \downarrow{\scriptstyle G(f)} \\ F(Y) & \xrightarrow{\eta_Y} & G(Y) \end{array}$$

Where $f \in mor(X, Y)$[2]

The intuition

How does this construction have those properties I talked about earlier? If I have two functors F and G, I can apply it them to an object X and get two objects from D, the top arrow says that we require a morphism from $\eta_X : F(X) \to G(X)$, which is to say that we need $F(X)$ and $G(X)$ to have the same structure (whatever that means in our particular category).

It's worth pointing out that our choice of X is arbitrary; it could have been any object in $ob(C)$, so do we really need to specify the bottom arrow since we already have a mapping of arbitrary objects?

Consider the following (silly) example for groups. Here I take $F, G : Grp \to Grp$ to be the identity functors. I take $\eta_{\mathbb{Z}}$ to be the identity function but I take $\eta_{\mathbb{Z}/2\mathbb{Z}}$ to be the zero function. If we didn't specify the bottom arrow, the resulting non-commutative diagram would represent a natural transformation.

[2]Here I take $mor(X, Y)$ to mean the morphisms between the objects X and Y in the category C.

$$\begin{array}{ccc} \mathbb{Z} & \xrightarrow{x \mapsto x} & \mathbb{Z} \\ {\scriptstyle x \mapsto x \mod 2}\downarrow & & \downarrow{\scriptstyle x \mapsto x \mod 2} \\ \mathbb{Z}/2\mathbb{Z} & \xrightarrow{0} & \mathbb{Z}/2\mathbb{Z} \end{array}$$

Explicitly stating that the bottom arrow makes the diagram commute is surprisingly deep. The bottom arrow is the condition that our functors change morphisms and objects in a way that are compatible with each other! Something is the same regardless of whether you started in the domain or codomain.

An odd example that helped it click for me

At a Haskell meetup, the organizer (Gershom Bazerman) said "When people ask me for references on category theory, I say 'you always learn category theory from the second book you end up reading'", and I suspect that there may be an element of that here. Nonetheless, even though the example I'm about to provide is not a natural transformation, it helped me further understand "structure preserving functions".

Chain complexes

A chain complex $C_\bullet$ (or just C) is a collection of $R-$modules[3] C_i along with module homomorphisms $d_i : C_i \to C_{i-1}$ such that $d_{i-1} \circ d_i = 0$

$$\ldots \xrightarrow{d_{n+2}} C_{n+1} \xrightarrow{d_{n+1}} C_n \xrightarrow{d_n} C_{n-1} \xrightarrow{d_{n-1}} \ldots$$

In particular, note that the kernel of d_{i-1} is a submodule of the image of d_i (since $d_{i-1} \circ d_i = 0$), one of the things that we're interested in when we study chain complexes is the kernel of d_{i-1} (called cycles) and the image of d_i (called boundaries). In particular, we care about how much bigger the kernel is than the image; in other words, we want to know the extent to which this diagram is not exact.

From now on, I'll simply refer to the morphisms d_i as simply d.

[3] If you aren't familiar with modules, replace all instances with "R-module" with "vector space" or "abelian group".

Morphisms of chain complexes

A morphism of chain complexes from $C_\bullet \to D_\bullet$ is a collection of morphisms $u_i : C_i \to D_i$ such that the following diagram commutes:

$$\begin{array}{ccccccccc}
\ldots & \xrightarrow{d} & C_{n+1} & \xrightarrow{d} & C_n & \xrightarrow{d} & C_{n-1} & \xrightarrow{d} & \ldots \\
 & & \downarrow{\scriptstyle u_{n+1}} & & \downarrow{\scriptstyle u_n} & & \downarrow{\scriptstyle u_{n-1}} & & \\
\ldots & \xrightarrow{d} & D_{n+1} & \xrightarrow{d} & D_n & \xrightarrow{d} & D_{n-1} & \xrightarrow{d} & \ldots
\end{array}$$

It can be proven via diagram chasing that u sends cycles to cycles and boundaries to boundaries, which is to say, chain complex morphisms preserve precisely the structure that we're interested in.

Even though natural transformations may be intimidating at first, it is simply a way to say that two functors have the same structure; I hope this explanation has been helpful!