Umass Amherst

Fun with Cats

An Introduction to Archery

Ben Kushigian

What is Category Theory?

Category theory, lovingly[1] referred to as abstract nonsense, can initially be thought of as the "study of abstract algebras of functions". Category theory isn't interested in the values being mapped by functions so much as the general structure induced by the functions themselves.

For example, the set theorist would write injectivity of a function $f : A \to B$ as

$$\forall xy \in A : f(x) = f(y) \implies x = y$$

while the category theorist would say that f is injective[2] if for all $g, h : C \to A$, $fg = fh$ entails $g = h$,

[1] and accurately

[2] This construction is called a *monomorphism* in category theory land, and by reversing the arrows

$$D \underset{j}{\overset{i}{\leftleftarrows}} B \xleftarrow{f} A \,.$$

we get an *epimorphism* that is equivalent to surjectivity of functions. This 'reversing the arrows' trick is called the dual relation and is prevalent in mathematics: union is dual to intersection, addition is dual to multiplication, logical or is dual to logical and, etc...

$$C \underset{h}{\overset{g}{\rightrightarrows}} A \xrightarrow{f} B \,.$$

This is non-constructive, but allows us to easily generalize our definition. Why, you ask? Well, we haven't explicitly used any elements and, in fact, we can restate this without mentioning that we are using sets and functions at all: all we have used are some names A, B, C, some arrows $\xrightarrow{f}$, $\xrightarrow{g}$, and $\xrightarrow{h}$, and equality, as well as related them in some way ($C \xrightarrow{g} A$, ...).

While this may seem an arbitrary distinction at first, it turns out that this abstraction allows us to apply definitions to many different classes of objects since we never had to reference elements or types directly. What we have done is captured a relationship between 'objects' and 'arrows' that can be applied to many different mathematical types, from sets and functions to groups and group homomorphisms to topological spaces and continuous functions to logical statements and logical entailment to types and subtyping to lambdas and evaluation. This is the power of category theory. The computer scientist should immediately be reminded of parametric polymorphism.

To make this generalization work, however, we need to relax our definition slightly. Instead of talking about *functions between sets* we instead talk about *arrows*[3] *between objects*. At first blush arrows and objects can be thought of as strange names for functions and sets — this provides good intuition and ends up being correct for sets.

A collection of arrows and objects satisfying some rather modest properties is called a *category*, and these can be thought of as a *type* of mathematical object. As we can see, we are really just talking about PL!

This Reading Group

This summer we will be working through Steve Awodey's "Category Theory" which provides a lovely introduction to the subject with relatively few mathematical prerequisites. Mathematical constructions will be presented as needed[4] and a large

[3] arrows are also referred to as *morphisms*, as in group/ring/graph homomorphisms

[4] In particular, Awodey develops to structures that are very familiar to computer scientists: the monoid and the poset. Monoids are things that can be added and that have an identity. Strings under concatenation are your basic "free monoid", though the natural numbers under addition, $\mathbb{Z}_n$, and matrix multiplication also offer examples of monoids.

number of examples are presented from a variety of subjects, including mathematics, mathematical logic, and programming languages. From the preface:

> *Why write a new textbook on Category Theory, when we already have Mac Lane's* Categories for the Working Mathematician?. *Simply put, because Mac Lane's book is for the working (and aspiring) mathematician. What is needed now, after 30 years of spreading into various other disciplines and places in the curriculum, is a book for everyone else.*
>
> *[. . .]*
>
> *The students in my courses often have little background in Mathematics beyond a course in Discrete Math and some Calculus or Linear Algebra or a course or two in Logic. Nonetheless, eventually, as researchers in Computer Science or Logic, many will need to be familiar with the basic notions of Category Theory, without the benefit of much further mathematical training. [. . .] Most of my students do not know what a free group is (yet), and so they are not illuminated to learn that it is an example of an adjoint.*
>
> *This, then, is intended as a text and reference book on Category Theory, not only for students of Mathematics, but also for researchers and students in Computer Science, Logic, Linguistics, Cognitive Science, Philosophy, and any of the other fields that now make use of it.*

In a perfect world we would cover natural transformations, adjoints, and the Yoneda lemma, and wrap up by covering monads, as these mark the end of "elementary category theory." However, this is perhaps too ambitious and I will be satisfied if we come away with enough material and background to finish the text on our own time. Regardless, there is a lot of material so we will try to set a brisk pace, but since this will be somewhat a case of the blind leading the blind (I have only worked through the first few chapters myself, as well as skimming through later sections) we will stop and take time as needed.

In my experience it is very easy to passively read a math text without learning anything and the remedy for this is to do problems. To this end I will be picking out problems for us to work through on our own, with the idea of either presenting

---

Posets, or partially ordered sets, are another common construction in computer science. A DAG (directed acyclic graph), for example, induces a poset under the relation $u \leq b$ if $u = v$ or $REACH(u, v)$ (this is just the transitive closure) and is used, for example, in dataflow analysis.

these once a week or swapping them to be 'graded'. These are of course optional (since this is not for credit) but I will urge people to at least attempt some problems every week.

Meetings and Administratia

I'm not sure when/where we will meet or how long or often. I would be open to starting off with some lectures to give us a bit of a head start and then either continue with the lecture format or switch to a round table discussion format. We can talk about this when we have a first meeting.

Joining

Anyone who is interested in joining or being affiliated in any way can reach out on Slack (which I check annually) or email me at `bkushigian@cs.umass.edu`.