Applied Compositional Thinking for Engineers (ACT4E)

Session 8 - Duality- Q&A
Questions & Answers

Q: can you really find all functors in the graph you have shown?

DS: Yes, you can have all trivial functors.

Q. In set theory, we usually forbid universal sets (which are sets that contain all objects, including themselves) as they lead to paradoxes (in particular Russell's paradox). Do we have a similar problem with categories of categories and, if not, why?

GZ: Yes, we do. This was not explicitly mentioned, but **Cat** is the category which has "small" categories as objects, functors as morphisms, and composition of functors as composition. A category is called "small" if it has a small set of objects and a small set of morphisms. What does small mean? The set of objects/morphisms is called small if it is a proper set, and not a "class" (collection of ;/ sets). We will come back at the end of the lecture and give a live answer to this.

Q: Why "natural" transformation?

AC: because it feels natural. "god given", says JL.
NM: I believe the history was that mathematicians would say something is natural and 'right' due to some instinct that came from years of mathematical practice. Natural transformations were an attempt to formalise and define what this instinct meant (I can't remember if I read this in Maclane's category theory for the working mathematician or if I heard this elsewhere)
Note: these two answers are the same thing, the intuition that people had was that certain things were "god given" and are sometimes still be described like that today

Q: Slide 8/30 elaborates a natural transformation as morphisms between objects, as in F(a) and G(b). But where is this "morphism between morpshims" thing in the slide?

GZ: The morphism between morphisms is alpha here (the natural transformation).
Author: but the alpha goes between objects (F(a) and G(b)), not between morphisms. No?
GZ**:** alpha “compares” the functors, and by doing so acts on the functors acting on objects and on the functors acting on morphisms.
Q: is equality of functors the same thing as natural isomorphism between them? Btw how do we define the equality between the mapped objects? Do we always have a notion of equality of functors?
GZ: Natural equivalence and natural isomorphism are in general defined differently. Which slide are you referring to? We will come back at this during the Q&A part of the lecture.

OP: Slides 11 and 12
NM: The equal sign here would mean the composition of 1-morphisms (functors in this case) map the pair F and G to an identity functor. It is possible that this equal may be upto natural isomorphism which would mean F;G = id says that the composition of F and G would give a functor that has a natural isomorphism to (and from) the identity functor. If this is exactly equal or only upto natural isomorphism is often when you see the terms strict, lax, weak appearing.

Q: In the case of functorial semantics what would be the (right?) adjoint of the functor F:syntax --> semantics? Does an adjoint always exist?

I was thinking of something simpler like regexp --> state machines or things like that

Q: In Geometric Algebra, can the duality (e.g. in 3D) between bivectors and vectors be described by adjoint functors?
NM: I think the hodge star (from differential geometry) is a more generalised example in any number of dimensions and I believe this is an equivalence of categories rather than only an adjunction between categories

OP: Thanks, so the hodge star would be an operation to get the dual? Your point with equivalence makes sense.

NM: I think you could construct the functors using the hodge star which would probably be done on object, and then you could work out how the morphisms change. The reason I’m not saying for certain if this is an adjunction or equivalence is that you can get some weird things for certain examples. Off the top of my head I think this is due to a difference between algebraic duals and topological duals, which are often the same but different for certain pathelogical cases.

Q: Is there a clear way to see adjunctions in terms of algorithmic optimisation?
KL: Like, finding the most general solution to a given problem is equivalent to finding the most specific problem that has a given solution?
KL: In engineering mathematics, one usually says that finding the approximate solution to a problem, is as good as finding the exact solution to an approximation of the problem. So instead of

approximating the solution to a differential equation (eg by a finite fourier sum), you can discretize the ODE to a matrix equation and solve exactly.

OP: The original question was motivated by the description on wikipedia of the pair of adjunction functors being the generalisation of the most efficient solution/most difficult problem pair. But part of what I was wondering was where the algorithms to perform variational optimisation fit into the categorical generalisation in order to help me parse the meaning better?

KL: So maybe a good example is to compare two kinds of solution strategies that we usually have in mathematics.
One idea is to transform a given problem to another but equivalent form, solve the transformed problem, and transform back. Since the transformation is (by definition) an equivalence, we have solved the original problem. The point of doing the (equivalent) transformation is that in the transformed domain the solution is obvious, or more transparent, or standard methods can be used. An example is to transform a differential equation to fourier domain, solve an algebraic problem (which is equivalent to the ode but in fourier space) and then transform back to time/space domain by inverse fourier transform.
Now think of a modification to the above strategy. Transform the given math problem to another form that is easier (easier in the strict mathematical sense). Solve the easy problem, and infer something about the original problem. Since the problems are not equivalent, we cannot hope that the solutions of the easy transformed problem are also solutions to the original problem, but we gain valuable information. This is often seen in the need to use implications (=>) instead of equivalences (<=>) when we transform an equation.
An example is to transform the problem of (global) minimization of a function to an equation about the derivative of the function. Of course not all zeros of the derivative are valid solutions to the minimization problem. We lost the equivalence, but we now have a finite set of candidates to check. This “close but no cigar, but good info” strategy is very often an example of the adjoint functor situation.

(PS: are these answers from the same person or two different people?)

Q: do opposite categories always exist? Or sometimes not? I imagine there might be cases where taking the opposite would just not make any semantic sense.

NM: The opposite category always exists (there is an opposite functor, though it is something called a contravariant functor), but the interpretation of the category does not always have an obvious meaning.