## Applied Compositional Thinking for Engineers (ACT4E)



## Session 5 - Choosing

## Questions \& Answers

Q: First you mentioned unions, and then went to how CT generalizes disjoint unions. Does category theory also have a way to generalize union, not just disjoint union?
DS: Why yes! In the category of subsets of a set X , with morphisms being inclusions, the coproduct is the regular old union. The disjoint union is the coproduct in the category Set of sets and functions.

Q: Should the arrows in the battery example from $A / B$ to $A+B$ be reversed to be consistent with the general diagram?
DS: Here's the right form, whatever was said:

$A+B$
NM: No, the direction of the arrows was correct, one thing to keep in mind is that the arrows between $A / B$ and $A+B$ (or $A x B$ ) should be the same direction as between $A / B$ and the extra object $C$ (in this case the set of prices). A very crude way of thinking about it is that the existence of a product/co-product is saying that if it kind of looks like C could replace $A+B$ (or $A x B$ ) then we can instead pass through $A+B$ (or $A x B$ ) when moving between $\mathrm{A} / \mathrm{B}$ and C .
To try to give an intuitive meaning of this, I would say that the product/co-product contains the smallest/largest amount of stuff required.
Q. In the example of the category whose morphisms are $\leq$ and objects are sets, what are the $\boldsymbol{i}_{\mathbf{\prime}} \mathbf{a}$ and $\mathrm{i}_{-} b$ morphisms (the injections)?

NM: The morphisms/injections would have to still be a $\leq$, though what exactly do you mean by $\leq$, do you mean set inclusions?
OP: Yes, maybe you can clarify: in that example that Giole just showed, are the objects of the category sets, in which case they correspond essentially to tuples in the relation, or propositional statements? or are they sets of sets so that the morphism $\leq$ is a relation?
NM: This is the category of ordered sets? Or the union of subsets?
OP: I don't know. What exactly is the category in Gioele's example?
NM: Do you have the page of the slides that it was on?
OP: Slide 21/30.
NM: So union of subsets, in this case the objects are subsets of a fixed set X . Then the morphisms are set inclusions
OP: I see. So in this kind of category, the morphisms are really trivial things, because (in set theoretic terms) they are just individual tuples, right?
NM: I guess the morphisms can be as trivial or complicated as you like, I'm not familiar with formal set theory OP: Thanks! I think I get it.

JL: I'm not sure if l'm understanding which example is meant, but $i$ think its the one where we are looking at the real numbers and the relation $\leq$. Here the objects of the category in question are the elements of the set of real numbers. That is, objects are numbers. Does that help?
(So here objects are real numbers, and morphisms are the relationships of less-than-or-equal between them) OP. Sorry for the confusion. Was referring to subset not LTE...

## Q. What are good examples of monoid objects in monoidal categories where the monoidal product is a categorical coproduct? In (Set,+) it seems a monoid is just a pair of functions (not interesting). Are there any less trivial examples?

NM: If the question is about monoidal categories that use a co-product structure as the monoidal product then this is called a co-cartisian monoidal category (https://en.wikipedia.org/wiki/Cartesian_monoidal_category) which includes the categories of: abelian groups, vector spaces, and of R-modules. What you might be picking up on is that if this category is also closed (every Hom-set is itself an object) then the category is equivalent to the terminal category and therefore non-interesting
(https://ncatlab.org/nlab/show/cocartesian+closed+category)
NM: For monoidal objects, any object of the co-cartisian monoidal category would be an example (if you can find valid morphisms for multiplication and units)
OP: Yepp! Correct! My question was wrong :) What I meant to say was what interesting comonoid objects there are. A comonoid object in (Set,+) is just a pair of functions (boring). The nlab link above was a good answer. Thanks
NM: I think you may have mixed up terms here, for a monoidal category you can have both monoids, and comonoids (Off the top of my head I think algebras and co-algebras are examples of monoidal objects and comonoidal objects in the monoidal category of vector spaces)
Other: There are "special" monoidal categories, where the monoids and comonoids behave trivially. So in a cartesian monoidal category (where the "tensor product" is the cartesian product) the comonoid objects are unique and trivial (the diagonal copy) and the monoid objects are the ordinary monoids. In a cocartesian monoidal category (where the "tensor product" is the coproduct) the monoid objects are unique and trivial (forget the index) and the comonoid objects are a partition of an object in 2 parts

In Sets $A+A=\{<a, 1>\mid a$ in $A\} \cup\{<a, 2>\mid a$ in $A\}$
Each element of $A$ occurs 2 times in $A+A$, and we keep track of them by indexing them with a parameter 1 or 2. The only way to build a monoid $m: A+A->A$ is to map an element of $A+A$ to its obvious element of $A$ by forgeting the index.
The comonoids are the maps d:A->A+A that make an arbitrary choice whether a in A shall get the index 1 or get the index 2. So the comonoids are a the ways to partition A into A_1 and A_2

Q: What are some examples of where someone would use coproducts of graphs? It looks reminiscent of steps in Feynman diagrams
A forest is a coproduct of trees right?

Q: Can you give an example (of a coproduct) in a category where objects are not set-like, i.e., they don't have elements?

JL : there were several examples of this sort in the lecture:

- real numbers as objects, and "less-than-or-equal" relations as morphisms
- natural numbers as objects and "divides" relationships as morphisms

OR: thank you!

Q: In programming we have product types and sum types (i.e. struct and union in C), I guess these can be modeled with products and coproducts?
http://brendanfong.com/programmingcats.html
https://bartoszmilewski.com/2014/10/28/category-theory-for-programmers-the-preface/

