Twitter Thread by Jonathan Beardsley

Jonathan Beardsley
@JBeardsleyMath

In light of my tweet thread about the category of finite sets and commutative monoids (https://t.co/jnY0wZZbxq), I thought I might try to say what the analogue is for braided monoidal things (although much of this is still somewhat hypothetical).

> So okay, here's a thread on the category of finite sets and a way in which it controls algebraic structure in symmetric monoidal categories. I think it's some really pretty stuff.
>
> — Jonathan Beardsley (@JBeardsleyMath) December 6, 2020

It's also just kind of a cool combinatorial structure! I've been talking to @CreeepyJoe about this lately, as well as @grassmannian.

The first thing you have to know is that, in a braided monoidal category you can still have commutative monoids. Since a braided monoidal category C has a "twist" map for every object $\beta(x):x\otimes x\to x\otimes x$, if x is a monoid you can ask for the following diagram to commute:

Remember that being symmetric monoidal just means that if you take the twist map above and do it twice, you get the identity map, but braided monoidal doesn't mean that. But it's okay! You can still define commutative monoids here.

But so anyway, we can talk about commutative monoids in braided monoidal categories.

So we can talk about "the free braided monoidal category containing a commutative monoid," which I'll denote V. One way to say precisely what that means is that for any other braided monoidal category C, the category of braided monoidal functors BrFun(V,C) is equivalent to the…

…category of commutative monoids in C. This should be compared to the fact that symmetric monoidal functors from FinSet (with coproduct as "tensor product") into a symmetric monoidal category are equivalent to commutative monoids in that category.

So we want the same thing for braided monoidal categories. It turns out that this is NOT just FinSet again. A simple reason why is to think about what a braided monoidal category need.

One thing it needs is the following: given any object x in a braided monoidal category, there should be actions of the braid group on n strands, B_n, on the n-fold tensor product x■.

Now, FinSet has this property since there are maps B_n→S_n that take a braid's "underlying permutation," but think about what it means to be "freely generated."

Recall, for instance that the free group on 2 generators {a,b} just takes all possible words on a and b, e.g. ab, abb, abababababaaaa, aaaba, etc. So if we want to freely generate a braided monoidal category then we need to freely install all these braid group actions.

I.e. for each n-fold tensor product x■ we should just kind of freely produce a new object βx■ for each β∈B_n (and then mod out by the relevant relations so, like, σ(β(x■))=σβx■, or whatever. And this isn't what happens in FinSet.

In FinSet there are lots of braids that act the same way on n-fold tensor products, since we've only remembered the underlying permutation of the braid. So then what IS the free braided monoidal category containing a commutative monoid?

It's interesting to me because the answer to this question doesn't seem to be nearly as well known as the answer to the question "What is the free symmetric monoidal category containing a commutative monoid?" (i.e. FinSet).

The answer is the category of so-called "vines," which is why I'm calling it V. Honestly I feel a bit weird about saying the word "vines" so much. It feels a bit "cranky," like I'm going to go to my grave screaming about this bizarre structure I'm heavily invested in. Anyway...

Let's talk about what this category looks like. First of all, its objects are the natural numbers (which obviously include zero). The morphisms are a bit trickier, and probably easiest to describe somewhat geometrically.

Given two natural numbers n and m, to describe a morphism n→m I want to start by fixing n points on the line (x,0,1) in ■³ and m points on the line (x,0,0).

For convenience you can imagine them stacked right above one another, so for 2→3 you can take {(1,0,1), (2,0,1)} and {(1,0,0),(2,0,0),(3,0,0)}, if you like, although ultimately it won't matter.

Then a morphism from n to m is a collection of n piecewise-linear arcs starting at the top points and moving at unit speed to the bottom points satisfying the condition that IF two arcs collide at time t then they are identical until they get to a bottom point.

The picture of a "vine" is something like this (where I've "smoothed" the piecewise-linearity because it's not really important):

It's also important that I do something akin to taking "isotopy classes" of vines. I don't want to differentiate between two vines if one is just the other wiggled around, so long as the strands don't violate the rules. Something like this:

A couple things to notice: I can move the time that two strands collide up and down however I want. Also notice that a bottom point need not have anything "incoming." We'll see a consequence of that when we get to thinking about composition.

T.G.

Lavers studied the set of such things in the case that n=m and showed that it had a monoid structure in this paper: https://t.co/DvhCj4449T

He also showed that every vine decomposes as a braid followed by a planar tree.

This is pretty non-trivial, but if you get out some shoelaces and mess around you can probably convince yourself it's true. I think of the process of "isotoping" a vine into this form as "combing" it, in the same way that you "comb" a braid.

Here's another funky consequence of these axioms. These two vines are identified:

More generally, a vine has a sort of "canonical" form where it's a braid stacked on top of a planar tree such that if we only look at the strands that hit a chosen bottom point then the braid on those strands is the trivial braid.

If you go back to that picture up there it's a little tricky to see, but starting at the bottom you can kind of push the strands around (since they're allowed to approach that bottom point from any angle in ■^{3}) and slowly unwind the braid. That's going to be important later.

Now how do we compose a vine n→m with a vine m→k? The short answer is that we "stack them" like we stack braids in the braid group. You can probably visualize how this works, but there's potentially some confusion about "loose ends."

Think about the following composition of a vine 3→3 with another 3→2 :

The composite vine has that funny little hair sticking out of the bottom right vertex that doesn't come from a top vertex anymore. What we do to make sense of this is just "contract" that hair down to nothing. We ignore it, i.e.:

I get that this may seem a bit arbitrary, but Lavers in the reference I cited above proves that it works out (at least when n=m, which gives you a monoid structure), and Weber gets into it for the more general case in section 6.3 of this: https://t.co/U9uSMBJkFS

I don't know if anyone says this in any of those references, but I imagine you can sort add an arbitrary number of "hairs" to any vine and they just kind of contract down. Like you could plug that into the definition somehow, but anyway I guess it doesn't matter.

So these are the morphisms of this category V. Notice that the braid group embeds into the endomorphism monoid V(n,n) (where the monoid structure comes from composition).

This category is also braided monoidal. The monoidal structure is given by addition of natural numbers. The braiding should be a symmetry n+m→m+n, and it's the same braiding you have in the free braided monoidal category (a braid as a vine). Here it is for 3+2→2+3:

So why does this category contain a commutative monoid? Just like FinSet, the commutative monoid is going to be the object 1. There's a vine that looks like a Y that goes 1⊗1=2→1 and that's the multiplication. Why is it commutative?

Well, the twist map in this case is the braid 2→2 which just crosses the left strand over the right strand. And we want to ask that that diagram way up there (the first one in this thread) commutes, which just means that the two following braids should be the same vine:

But of course... that's exactly the funny condition we had way up there about isotoping vines. But I do think this is super cool, that there's this confluence of a sort-of "geometric" structure and a "categorical" structure.

And so if you take a braided monoidal functor out of this into a braided monoidal category it has to preserve the multiplication (the Y vine) and the braiding (the, you know, braid), and it has to preserve that commutative diagram. So it has to take 1 to a commutative monoid.

And just like with FinSet, that's ALL it does. It picks out a commutative diagram. So this isn't a proof at all, but hopefully you can sort of see why this category V is the free braided monoidal category containing a commutative monoid.

Now here's where things get "hypothetical" in the sense that I haven't written the proofs.

I want to say that braided monoidal pseudofunctors out of V into the 2-category of categories (where the "braided" monoidal structure is just Cartesian product) are equivalent to small braided monoidal categories.

I think that proving this is basically a matter of checking a bunch of technical details, but I haven't done it yet.

Here's the rough idea, and it goes back to the diagram that I've now drawn like three times, the braiding composed with the multiplication vs. the multiplication alone. In V these two morphisms are equal.

If you'll let me write X for the braid 2→2 that we've been discussing, then in V we have that YX=Y. But since we're now talking about pseudofunctors, we don't need composition to be preserved strictly.

In other words, all we need is an invertible natural transformation from F(YX) to F(Y). And that's EXACTLY the salient data of a braided monoidal category.

So I think this is pretty cool that this funny kind of combinatorial category is precisely the thing that "parameterizes" braided monoidal categories, and that the really important stuff about it has actually been floating around since the mid 90's!

Ultimately I think I just want people to think a bit more about this category. In some sense this category V is "as important" as these other really fundamental categories like FinSet and Δ, but people just never seem to have heard of it.

By the way, if you're, I dunno, some rich person reading this, you can give me money for explaining these fun things to you: https://t.co/S78AAkpD5j (as always if you're a grad student or postdoc who gives me money I'll find you and beat you up, so don't do it).

Credits: thread composed using @chirrapp, that commutative diagram was made using @q_uiver_app, and I'm about to produce a single document containing this thread by asking @threadreaderapp to unroll it.