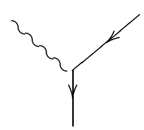
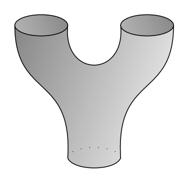
Physics, Topology, Logic and Computation: a Rosetta Stone

John Baez and Mike Stay

April 9, 2010
California State University, Fresno

Categories	Physics	Topology	Logic	Computation
object	system	manifold	proposition	data type
morphism	process	cobordism	proof	program

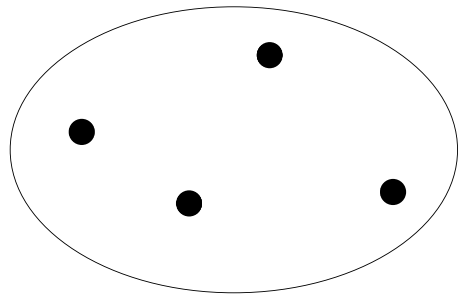




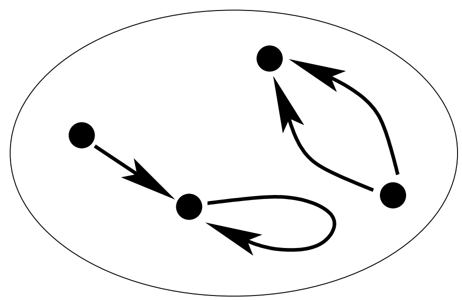
$X \& Y \vdash Z$ $\qquad$ $\lambda x \lambda y \, . \, x(y)$

The Big Idea

Once upon a time, mathematics was all about *sets*:



In 1945, Eilenberg and Mac Lane introduced *categories*:



These put *processes* on an equal footing with *things*.

In physics, we often use categories where:

- **objects represent physical *systems*;**
- **morphisms represent physical *processes*.**

In classical physics we often use the category Set, where:

- **an object is a *set***
- **a morphism is a *function***

In quantum physics we often use Hilb, where:

- **an object is a *Hilbert space***
- **a morphism is a *linear operator***

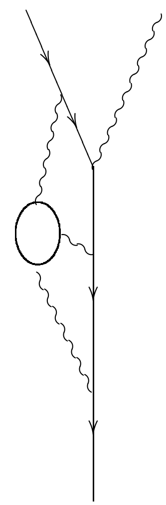
A category C consists of:

- **A collection of *objects*. If X is an object of C we write $X \in C$.**
- **For any $X, Y \in C$, a set of *morphisms* $f\colon X \to Y$.**

We require that:

- **Every $X \in C$ has an *identity* morphism $1_X\colon X \to X$.**
- **Given $f\colon X \to Y$ and $g\colon Y \to Z$, there is a *composite* morphism $gf\colon X \to Z$.**
- **The *unit laws* hold: if $f\colon X \to Y$, then $f1_X = f = 1_Y f$.**
- **Composition is *associative*: $(hg)f = h(gf)$.**

Feynman used diagrams to describe processes in quantum physics:

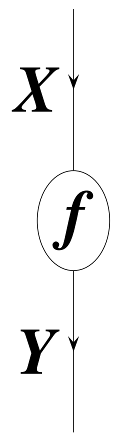


Now we know that these are pictures of *morphisms* — so we can use these diagrams in other contexts!

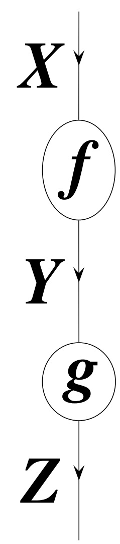
We can draw a morphism

$$f: X \to Y$$

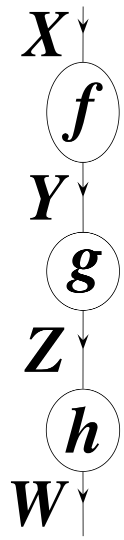
like this:



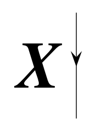
We draw the composite of $f\colon X \to Y$ and $g\colon Y \to Z$ like this:



Then the associative law is implicit:



If we draw the identity morphism $1_X \colon X \to X$ like this:



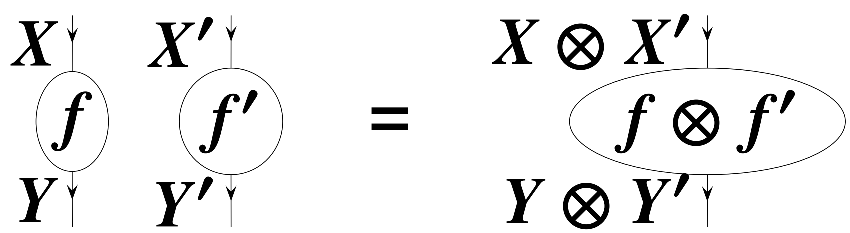
the unit laws are implicit too!

For theories with at least 1 dimension of space, we need *monoidal* categories.

Here any pair of morphisms $f\colon X \to Y$, $f'\colon X' \to Y'$ has a tensor product

$$f \otimes f'\colon X \otimes X' \to Y \otimes Y'$$

We use this to describe parallel processes:



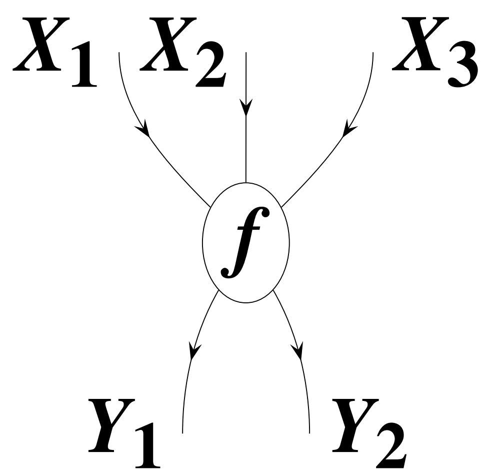
Examples:

- The category Hilb, with its usual tensor product $\otimes$.
- The category Set, with the cartesian product $\times$.

More generally, we can draw any morphism

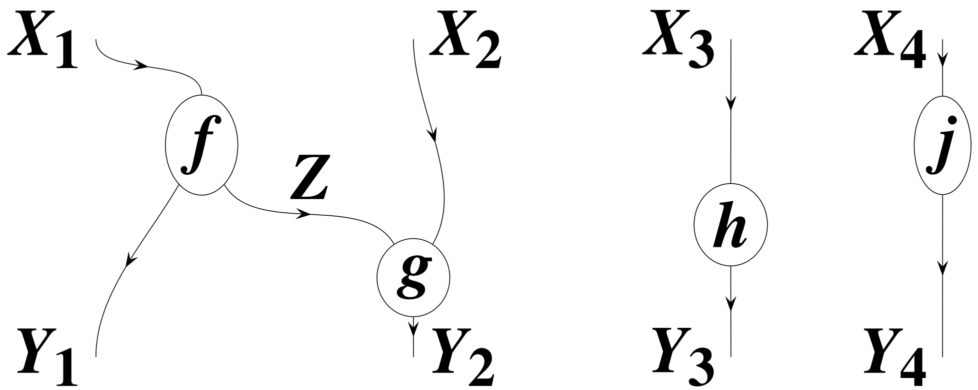
$$f\colon X_1 \otimes \cdots \otimes X_n \to Y_1 \otimes \cdots \otimes Y_m$$

like this:

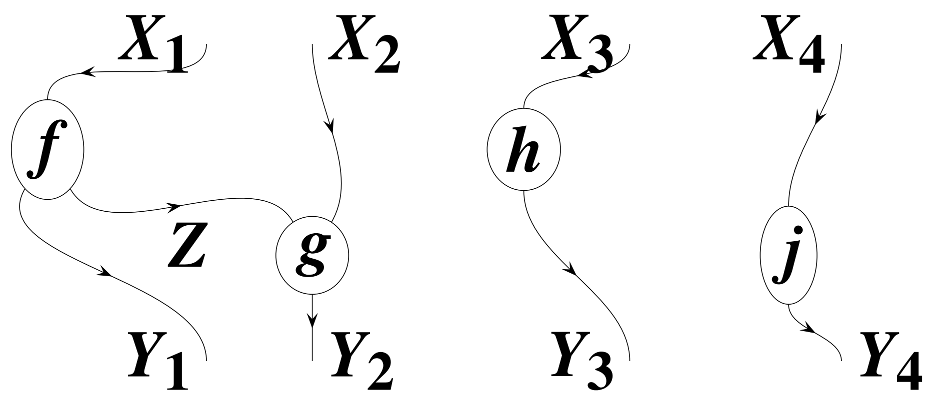


In physics we use this to depict an interaction between particles.

By composing and tensoring, we can build up bigger diagrams:



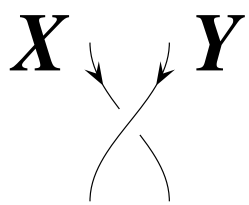
The monoidal category axioms let us deform the picture without changing the morphism:



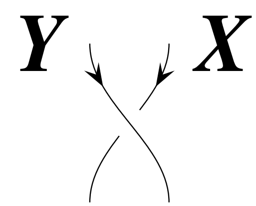
In theories with least 2 dimensions of space, we use *braided* monoidal categories. We can draw the braiding

$$B_{X,Y} : X \otimes Y \to Y \otimes X$$

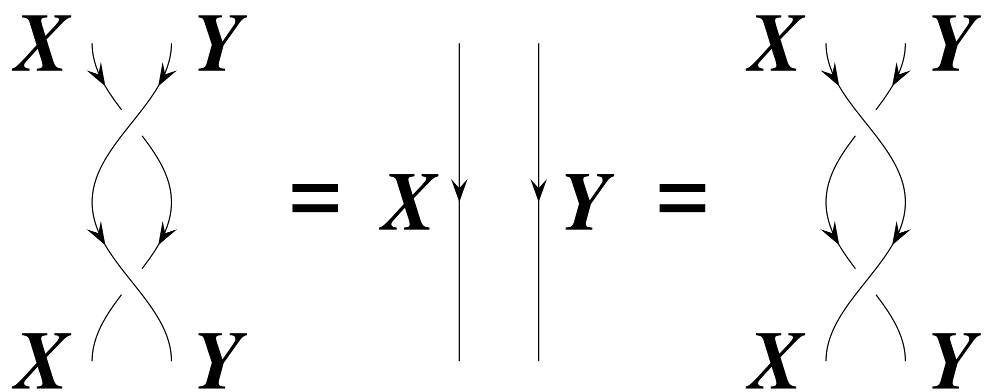
like this:



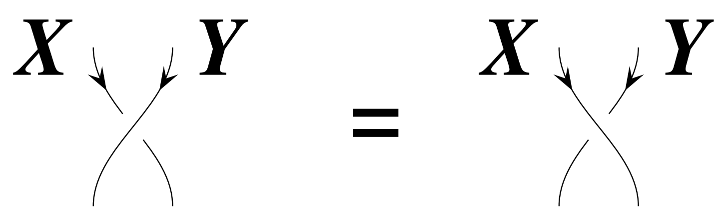
It has an inverse, drawn like this:



Then we have:



In theories with at least 3 dimensions of space, we use *symmetric* monoidal categories, where:



The most familiar braided monoidal categories are symmetric:

- **In Set with its cartesian product, the standard braiding is:**

$$\begin{array}{rcl} B_{X,Y} : X \times Y & \to & Y \times X \\ (x, y) & \mapsto & (y, x) \end{array}$$

- **In Hilb with its usual tensor product, the standard braiding is:**

$$\begin{array}{rcl} B_{X,Y} : X \otimes Y & \to & Y \otimes X \\ x \otimes y & \mapsto & y \otimes x \end{array}$$

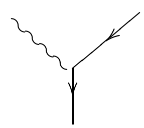
However, in thin films there can be ‘anyons’. These are particle-like excitations described by braided monoidal categories that are *not* symmetric!

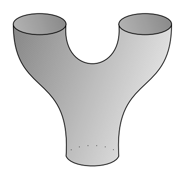
- **Superconducting films: the quantum Hall effect.**
- **Graphene (single-layer graphite): fractional-charge anyons are possible, not yet seen.**

But there's a lot more to this story...

THE ROSETTA STONE

Categories	Physics	Topology	Logic	Computation
object	system	manifold	proposition	data type
morphism	process	cobordism	proof	program





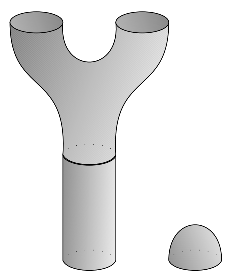
$X \& Y \vdash Z$

$\lambda x \lambda y \,.\, x(y)$

In topology, there is a category nCob where:

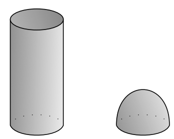
- **objects are $(n - 1)$-dimensional *manifolds*;**
- **morphisms are *cobordisms*.**

A *cobordism* $f\colon X \to Y$ is an n-dimensional manifold whose boundary is the disjoint union of X and Y. For example, when $n = 2$:

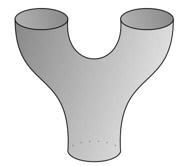


$$\begin{tikzcd} X \arrow[d, "f"'] \\ Y \end{tikzcd}$$

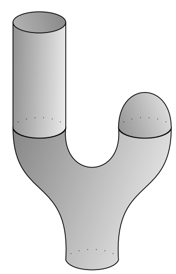
We compose cobordisms by gluing the ‘output’ of one to the ‘input’ of the other:



$X \xrightarrow{f} Y$

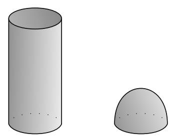


$Y \xrightarrow{g} Z$

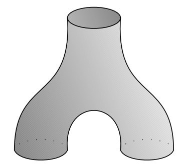


$X \xrightarrow{gf} Z$

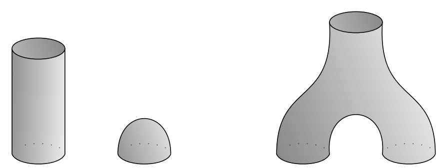
***n*Cob is a monoidal category. We tensor cobordisms by taking their disjoint union:**



$$\begin{tikzcd} X \arrow[d, "f"'] \\ Y \end{tikzcd}$$

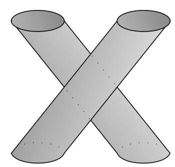


$$\begin{tikzcd} X' \arrow[d, "g"'] \\ Y' \end{tikzcd}$$



$$\begin{tikzcd} X \otimes X' \arrow[d, "f \otimes g"'] \\ Y \otimes Y' \end{tikzcd}$$

In fact, nCob is a symmetric monoidal category:



$$\begin{array}{c} X \otimes Y \\ \Big\downarrow {\scriptstyle B_{X,Y}} \\ Y \otimes X \end{array}$$

In general relativity, objects in *n*Cob describe choices of *space*, while morphisms describe choices of *space-time*. I believe that:

Quantum theory will eventually make more sense, as part of a theory of quantum gravity — but this can only be understood using categories.

Why? The weird features of quantum theory come from the ways that Hilb is less like Set than *n*Cob. But *n*Cob is what we use to describe space and space-time in general relativity!

'Weird' properties of quantum theory correspond to unsurprising properties of spacetime.

	object $\bullet$	morphism $\bullet \to \bullet$
SET THEORY	set	function between sets
QUANTUM THEORY	Hilbert space (state)	operator between Hilbert spaces (process)
GENERAL RELATIVITY	manifold (space)	cobordism between manifolds (spacetime)

For example: Set is 'cartesian', while *n*Cob and Hilb are not.

If a symmetric monoidal category is cartesian, you can do various things including *duplication*:

$$\Delta_X \colon X \to X \otimes X$$

In Set we can duplicate as follows:

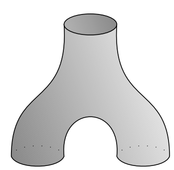
$$\begin{array}{rccc} \Delta_X \colon & X & \to & X \times X \\ & x & \mapsto & (x, x) \end{array}$$

In Hilb we cannot duplicate: the function

$$\begin{aligned} X &\to X \otimes X \\ x &\mapsto x \otimes x \end{aligned}$$

is not linear! It's not a morphism in Hilb.
So: we 'cannot clone a quantum state'.

Similarly, in nCob there is no duplication, despite this misleading picture for $n = 2$:



$$\begin{array}{c} X \\ \downarrow \\ X \otimes X \end{array}$$

When $n = 1$ there's typically no cobordism from a manifold X to $X \otimes X$, and similarly for $n = 4$.

What about logic and computer science? These too study categories of things and processes:

In proof theory, we use categories where:

- **an object is a *proposition***
- **a morphism is a *proof***

In computer science, we use categories where:

- **an object is a *data type***
- **a morphism is a *program***

In proof theory $X \vdash Y$ means *assuming* X, *we can prove* Y. But we can also let it mean *the set of proofs leading from assumption* X *to conclusion* Y.

Since proofs are morphisms, we can compose them:

$$\frac{X \vdash Y \quad Y \vdash Z}{X \vdash Z}$$

The identity morphism:

$$\frac{}{X \vdash X}$$

Logic uses *monoidal* categories where the tensor product is 'and'. We can tensor propositions, and tensor proofs:

$$\frac{W \vdash X \quad Y \vdash Z}{W \& Y \vdash X \& Z}$$

In fact, logic uses symmetric monoidal categories:

$$\frac{X \vdash Y \& Z}{X \vdash Z \& Y}$$

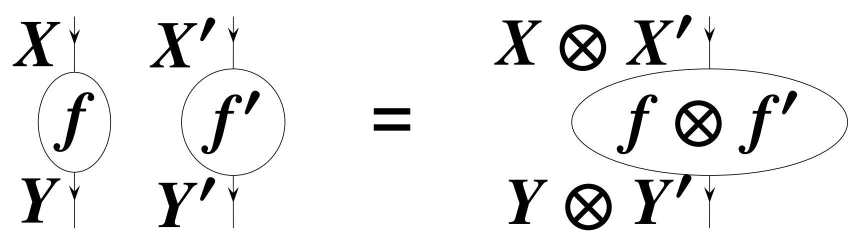
Classical logic is cartesian, so it permits duplication:

$$\frac{X \vdash Y}{X \vdash Y \& Y}$$

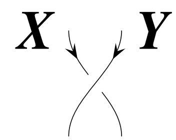
Linear logic does not!

A program that takes data of type X as input and returns data of type Y can be seen as a morphism $f\colon X \to Y$.

Categories of data types and programs are monoidal. Given data types X and X' there is a data type $X \otimes X'$. And given programs $f\colon X \to Y$, $f\colon X' \to Y'$, we can write a program $f \otimes f'$ that does these two jobs in parallel:



These categories are are typically symmetric monoidal:



They're also cartesian. For example, we can write programs that duplicate data:

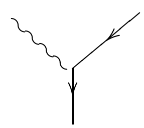
$$\Delta_X \colon X \to X \otimes X$$

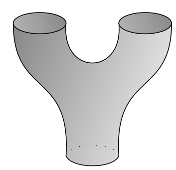
But for quantum computation, we need programming languages that apply to *noncartesian* categories — because you can't duplicate quantum data!

And in quantum computation using anyons, the relevant categories are *braided!*

For more detail, read our paper in Bob Coecke's forthcoming book *New Structures in Physics*. You can find it now . You can find it now on the arXiv.

Categories	Physics	Topology	Logic	Computation
object	system	manifold	proposition	data type
morphism	process	cobordism	proof	program





$X \& Y \vdash Z$

$\lambda x \lambda y \, . \, x(y)$