Physics, Topology, Logic and Computation:  
a Rosetta Stone

John Baez and Mike Stay

April 9, 2010  
California State University, Fresno

<table>
<thead>
<tr>
<th>Categories</th>
<th>Physics</th>
<th>Topology</th>
<th>Logic</th>
<th>Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>system</td>
<td>manifold</td>
<td>proposition</td>
<td>data type</td>
</tr>
<tr>
<td>morphism</td>
<td>process</td>
<td>cobordism</td>
<td>proof</td>
<td>program</td>
</tr>
</tbody>
</table>

\[
X \& Y \vdash Z \quad \lambda x. \lambda y. \, x(y)
\]
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 $f : X \to Y$.

We require that:

- Every $X \in C$ has an identity morphism $1_X : X \to X$.
- Given $f : X \to Y$ and $g : Y \to Z$, there is a composite morphism $g f : X \to Z$.
- The unit laws hold: if $f : X \to Y$, then $f 1_X = f = 1_Y f$.
- Composition is associative: $(h g) f = h (g f)$.
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 $f: X \to Y$ and $g: Y \to Z$ like this:
Then the associative law is implicit:

$$X \xrightarrow{f} Y \xrightarrow{g} Z \xrightarrow{h} W$$

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

$$X \xrightarrow{1_X} X$$

the unit laws are implicit too!
For theories with at least 1 dimension of space, we need *monoidal* categories.

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

$$f \otimes f' : X \otimes X' \rightarrow Y \otimes Y'$$

We use this to describe parallel processes:

$$\begin{array}{c}
\begin{array}{c}
X \\
\uparrow f \\
\hline
Y
\end{array}
\begin{array}{c}
X' \\
\uparrow f' \\
\hline
Y'
\end{array}
\end{array} = \begin{array}{c}
\begin{array}{c}
X \otimes X' \\
\uparrow f \otimes f' \\
\hline
Y \otimes Y'
\end{array}
\end{array}$$

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

$$f : X_1 \otimes \cdots \otimes X_n \rightarrow Y_1 \otimes \cdots \otimes Y_m$$

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:

\[ X \leftrightarrow Y \]

It has an inverse, drawn like this:

\[ Y \leftrightarrow X \]
Then we have:

\[
\begin{align*}
X & \quad Y \\
\downarrow & \quad \downarrow \\
\quad & \quad \\
\swarrow & \quad \swarrow \\
X & \quad Y \\
\end{align*}
\]

= \quad \quad X \quad Y

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

\[
\begin{align*}
X & \quad Y \\
\downarrow & \quad \downarrow \\
\quad & \quad \\
\swarrow & \quad \swarrow \\
X & \quad Y \\
\end{align*}
\]

= \quad \quad X \quad Y

The most familiar braided monoidal categories are symmetric:

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

  \[ B_{X,Y} : X \times Y \to Y \times X \]
  \[ (x, y) \mapsto (y, x) \]

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

  \[ B_{X,Y} : X \otimes Y \to Y \otimes X \]
  \[ x \otimes y \mapsto y \otimes x \]
However, in thin films there can be ‘anyons’. These are particle-like excitations described by braided monoidal categories that are \textit{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

<table>
<thead>
<tr>
<th>Categories</th>
<th>Physics</th>
<th>Topology</th>
<th>Logic</th>
<th>Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>system</td>
<td>manifold</td>
<td>proposition</td>
<td>data type</td>
</tr>
<tr>
<td>morphism</td>
<td>process</td>
<td>cobordism</td>
<td>proof</td>
<td>program</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
X & \& Y + Z \\
\lambda x \lambda y . x(y)
\end{align*}
\]
In topology, there is a category $n$Cob where:
- objects are $(n - 1)$-dimensional manifolds;
- morphisms are cobordisms.

A cobordism $f : X \to Y$ is an $n$-dimensional manifold whose boundary is the disjoint union of $X$ and $Y$. For example, when $n = 2$:
We compose cobordisms by gluing the ‘output’ of one to the ‘input’ of the other:
$n$Cob is a monoidal category. We tensor cobordisms by taking their disjoint union:
In fact, \( n\text{Cob} \) is a symmetric monoidal category:
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 spacetime in general relativity!

‘Weird’ properties of quantum theory correspond to unsurprising properties of spacetime.
<table>
<thead>
<tr>
<th></th>
<th>object</th>
<th>morphism</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>•</td>
<td>• → •</td>
</tr>
<tr>
<td><strong>SET THEORY</strong></td>
<td>set</td>
<td>function between</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sets</td>
</tr>
<tr>
<td><strong>QUANTUM THEORY</strong></td>
<td>Hilbert space</td>
<td>operator between</td>
</tr>
<tr>
<td></td>
<td>(state)</td>
<td>Hilbert spaces</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(process)</td>
</tr>
<tr>
<td><strong>GENERAL RELATIVITY</strong></td>
<td>manifold</td>
<td>cobordism between</td>
</tr>
<tr>
<td></td>
<td>(space)</td>
<td>manifolds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(spacetime)</td>
</tr>
</tbody>
</table>
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*:

$$\Lambda_X : X \to X \otimes X$$

In Set we can duplicate as follows:

$$\Delta_X : X \to X \times X$$

$$x \mapsto (x, x)$$
In Hilb we cannot duplicate: the function
\[ X \rightarrow X \otimes X \]
\[ x \mapsto x \otimes x \]

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

Similarly, in \( n \text{Cob} \) there is no duplication, despite this misleading picture for \( n = 2 \):

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:

\[
X \vdash X
\]
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 $f : X \to Y$.

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 $f : X \to Y$, $f' : X' \to Y'$, we can write a program $f \otimes f'$ that does these two jobs in parallel:

\[
\begin{array}{ccc}
X & X' \\
\downarrow f & \downarrow f' \\
Y & Y'
\end{array}
= \begin{array}{c}
X \otimes X' \\
\downarrow f \otimes f' \\
Y \otimes Y'
\end{array}
\]
These categories are typically symmetric monoidal:

\[
\begin{array}{c}
X \\
\downarrow \\
\uparrow \\
Y
\end{array}
\]

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

\[\Delta_X : X \to X \otimes X\]

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 on the arXiv.

<table>
<thead>
<tr>
<th>Categories</th>
<th>Physics</th>
<th>Topology</th>
<th>Logic</th>
<th>Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>system</td>
<td>manifold</td>
<td>proposition</td>
<td>data type</td>
</tr>
<tr>
<td>morphism</td>
<td>process</td>
<td>cobordism</td>
<td>proof</td>
<td>program</td>
</tr>
</tbody>
</table>

\[
X & Y \vdash Z \quad \lambda x \lambda y . x(y)
\]