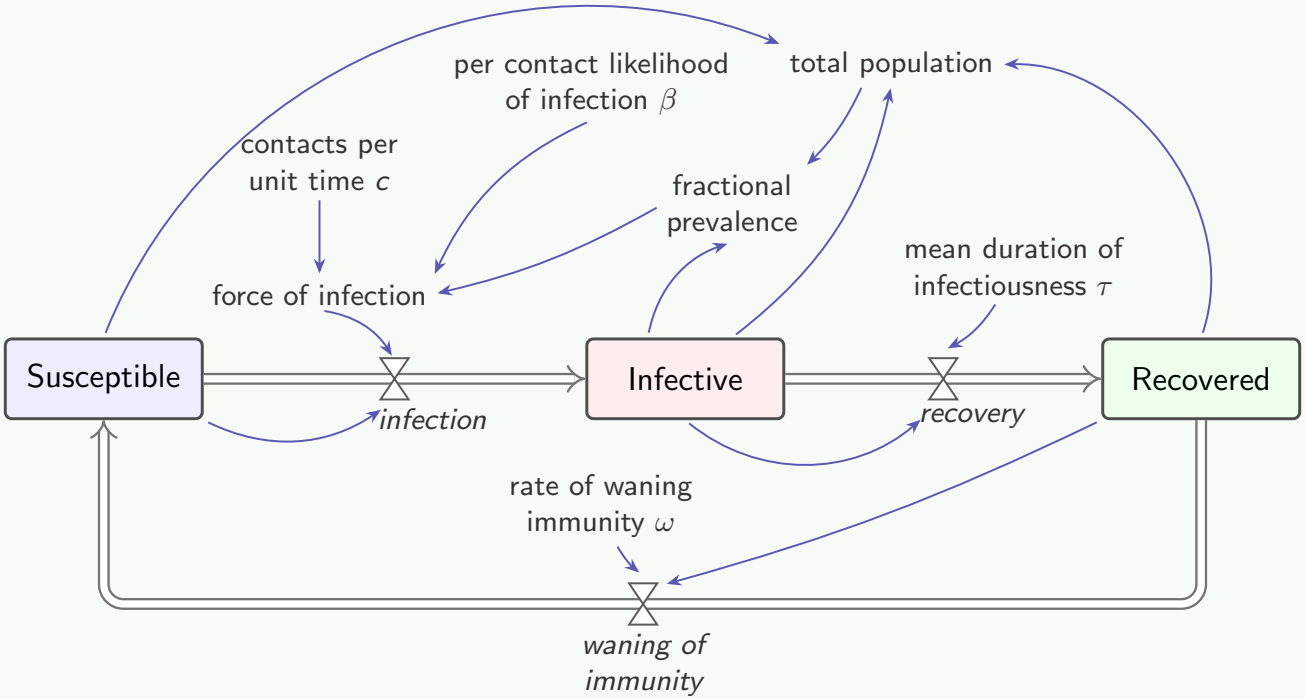
Applied Category Theory and The Mathematics of Networks – II

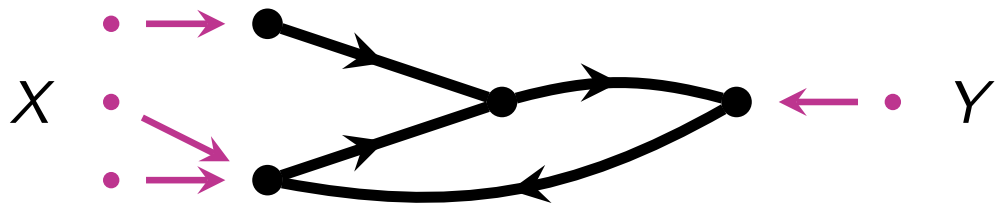


John Baez, University of Edinburgh

GeoCOW 2026

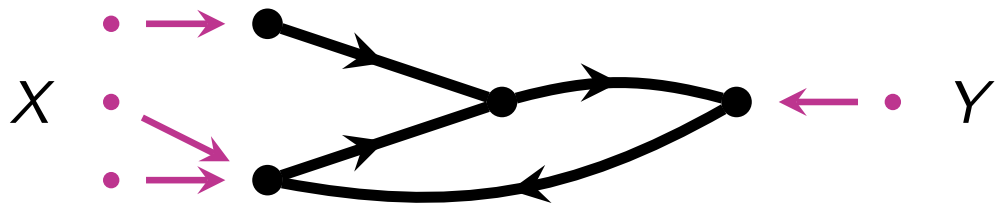
Why Monoidal Categories?

Networks of some particular kind, with specified inputs and outputs, can be seen as morphisms in some particular monoidal category:



Why Monoidal Categories?

Networks of some particular kind, with specified inputs and outputs, can be seen as morphisms in some particular monoidal category:



Such networks let us describe *open* systems, meaning systems where:

- stuff can flow in or out;
- we can combine systems to form larger systems by *composition* (sticking together networks in series) and *tensoring* (sticking them together in parallel).

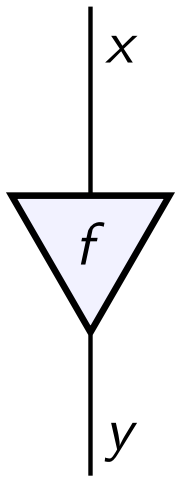
Definition of a Category

A **category** $\mathbf{C}$ consists of:

1. a set $\mathrm{Ob}(\mathbf{C})$ of **objects**.

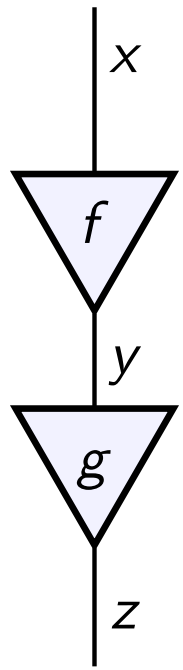
2. for any pair of objects $x, y \in \mathrm{Ob}(\mathbf{C})$, a set $\mathbf{C}(x, y)$ whose elements are called **morphisms** $f : x \to y$.

We can draw a morphism $f : x \to y$ like this:



Furthermore, in a category

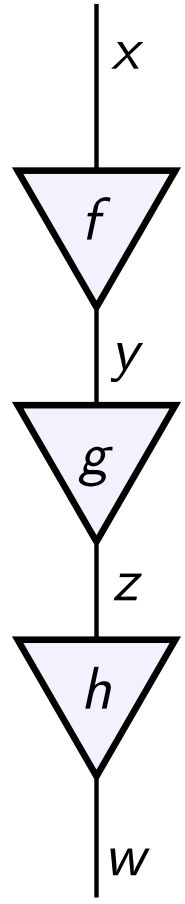
3. we can **compose** a morphism $f : x \to y$ and a morphism $g : y \to z$ to get a morphism $g \circ f : x \to z$.



4. Composition is associative:

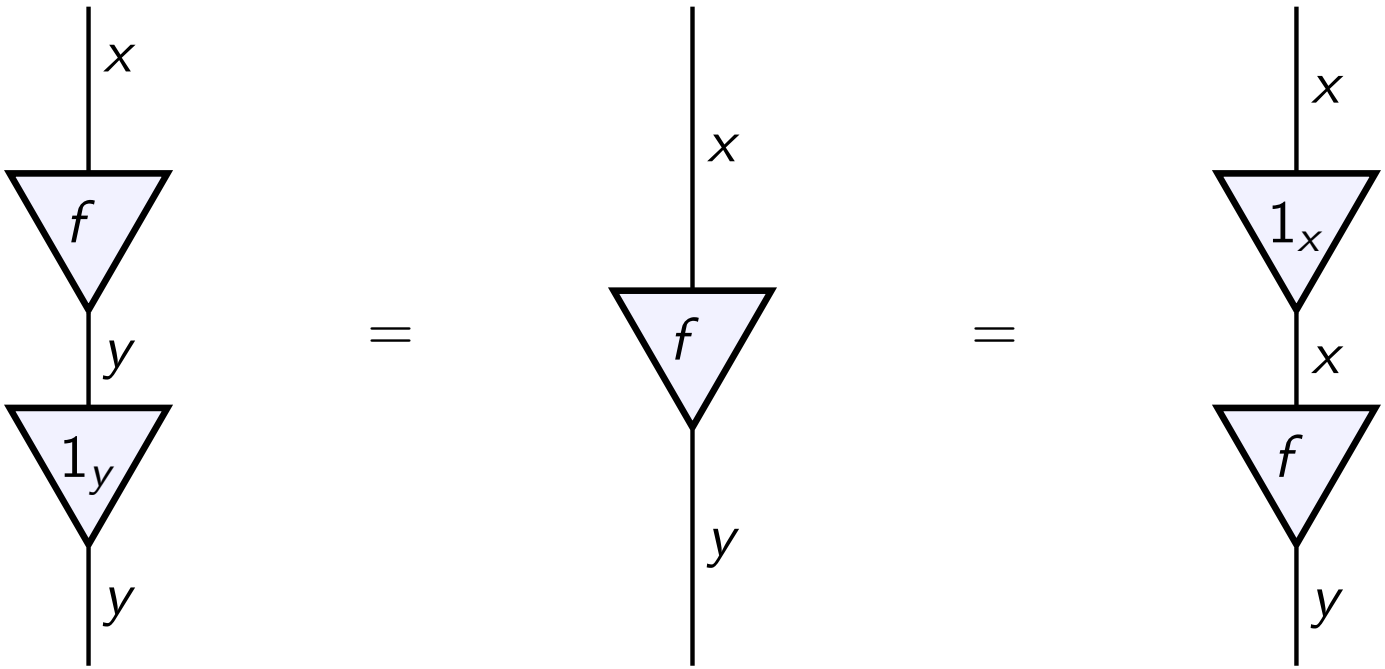
$$h \circ (g \circ f) = (h \circ g) \circ f$$

so the two ways of reading this diagram agree:



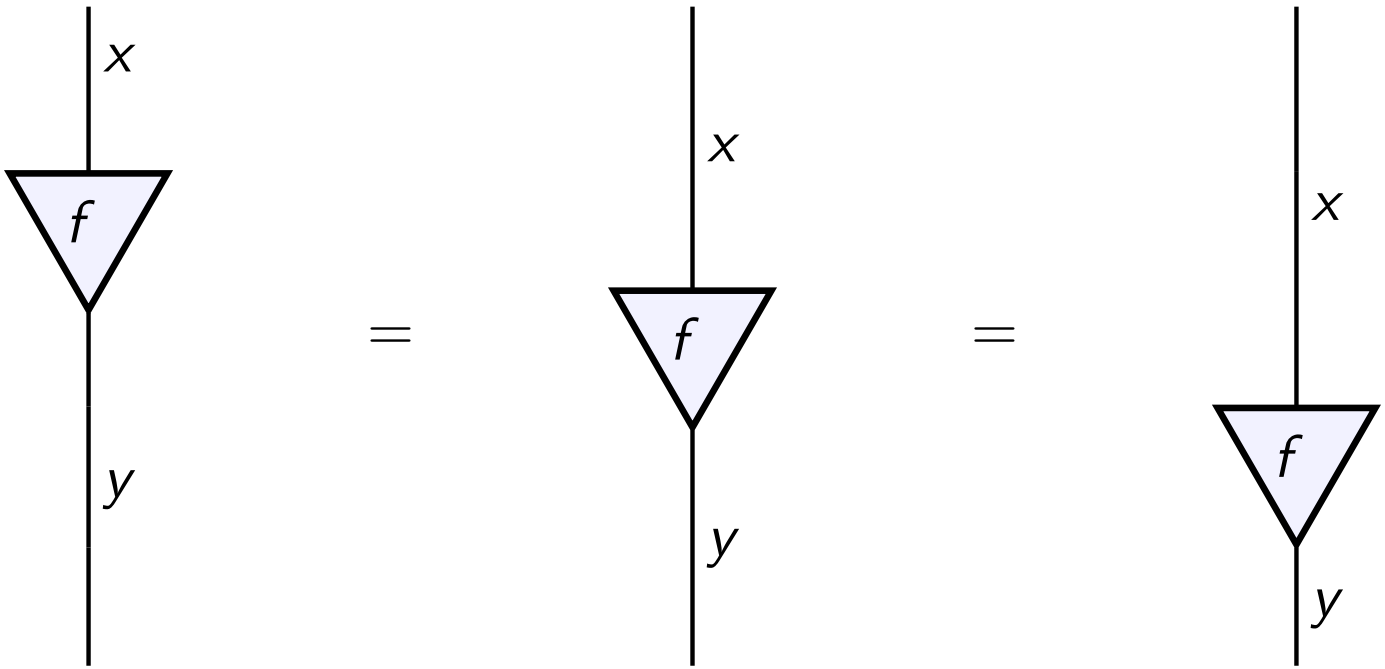
5. For any object $x \in \mathrm{Ob}(\mathbf{C})$, there is a so-called **identity morphism** $1_x : x \to x$ such that for any $f : x \to y$ we have

$$1_y \circ f = f = f \circ 1_x$$



5. For any object $x \in \mathrm{Ob}(\mathbf{C})$, there is a so-called **identity morphism** $1_x : x \to x$ such that for any $f : x \to y$ we have

$$1_y \circ f = f = f \circ 1_x$$

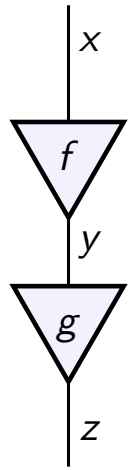


The most important category for traditional math is **Set**. Here the objects are *sets* and the morphisms are *functions*.

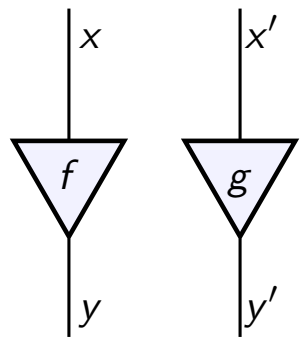
Another very important category is **Vect**. Here the objects are *vector spaces* and the morphisms are *linear maps*.

There are thousands more, but the categories we're interested in are all 'monoidal'.

In a category, we can do processes 'in series'. This is just composition of morphisms:



In a monoidal category, we can also do processes 'in parallel':



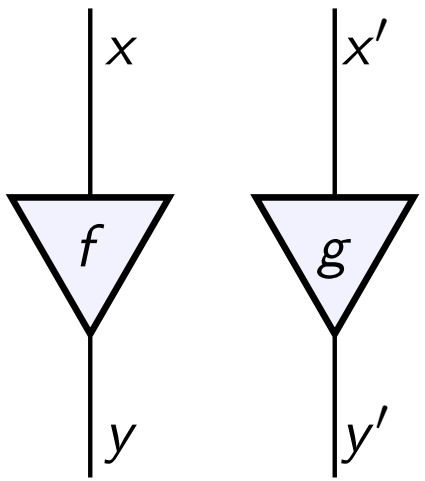
We say we are **tensoring** $f : x \to y$ and $g : x' \to y'$ to get a morphism $f \otimes g : x \otimes x' \to y \otimes y'$.

Definition of a Monoidal Category

A **(strict) monoidal category** $(\mathbf{C}, \otimes)$ is a category $\mathbf{C}$ where:

1. for any pair of objects $x, y \in \mathrm{Ob}(\mathbf{C})$ there is an object $x \otimes y$ called their **tensor product**;

2. for any pair of morphisms $f : x \to y$ and $g : x' \to y'$ there is a morphism $f \otimes g : x \otimes x' \to y \otimes y'$ called their **tensor product**.



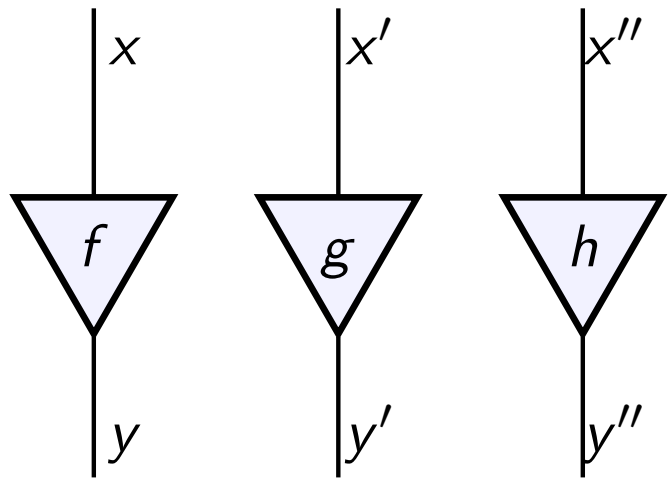
4. the tensor product of objects is associative:

$$(x \otimes y) \otimes z = x \otimes (y \otimes z)$$

and the tensor product of morphisms is associative:

$$(f \otimes g) \otimes h = f \otimes (g \otimes h)$$

so the two ways of reading this diagram agree:



4. there is a **unit object** $I \in \mathrm{Ob}(\mathbf{C})$ such that

$$I \otimes x = x = x \otimes I$$

for all $x \in \mathrm{Ob}(\mathbf{C})$ and

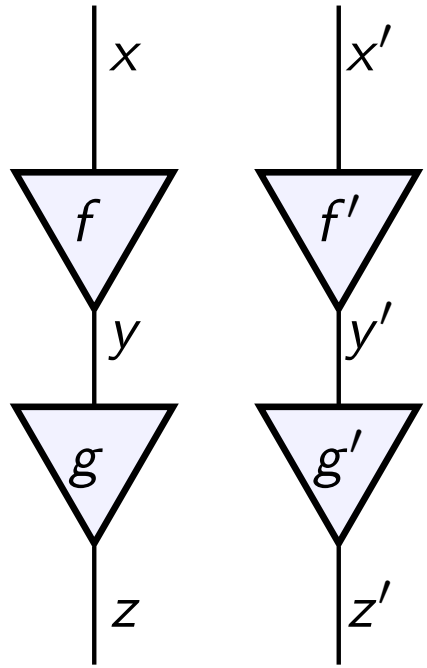
$$1_I \otimes f = f = f \otimes 1_I$$

for every morphism $f : x \to y$;

5. finally, the **interchange law** holds for all $f : x \to y$, $g : y \to z$, $f' : x' \to y'$, $g' : y' \to z'$:

$$(g \circ f) \otimes (g' \circ f') = (g \otimes g') \circ (f \otimes f')$$

This says that the two ways of reading this diagram agree:



There is a monoidal category $(\mathbf{Set}, \times)$ where $\times$ is the cartesian product of sets and the unit object is a fixed 1-element set.

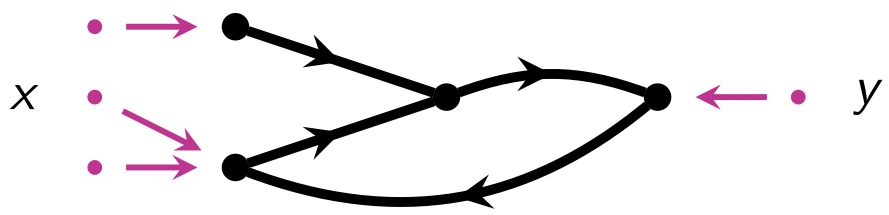
There is also a monoidal category $(\mathbf{Vect}, \otimes)$ where $\otimes$ is the usual tensor product of vector spaces and the unit object is a fixed 1-dimensional vector space.

But the monoidal categories we're interested in *now* have 'networks' as morphisms — different categories for different kinds of networks.

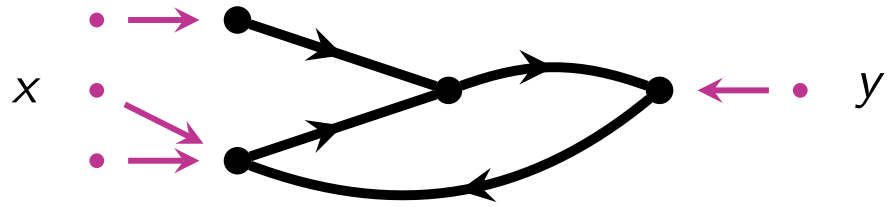
For example:

Define a **graph** (or quiver) to be a set N of **nodes**, a set E of **edges** and maps $s, t \colon E \to N$.

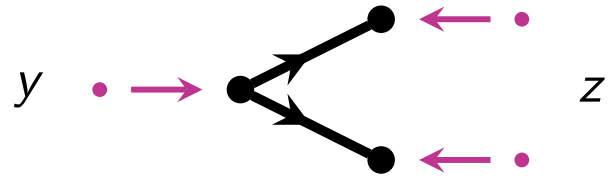
There's a category **OpenGraph** where an object is a finite set and a morphism $f : x \to y$ is an **open graph from x to y**: a graph together with maps from the sets x, y to its set of nodes:



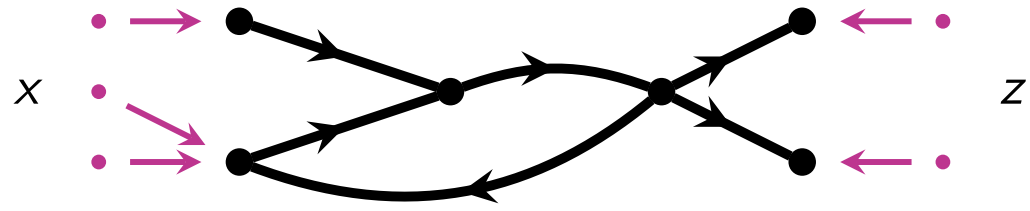
We can **compose** open graphs $f : x \to y$



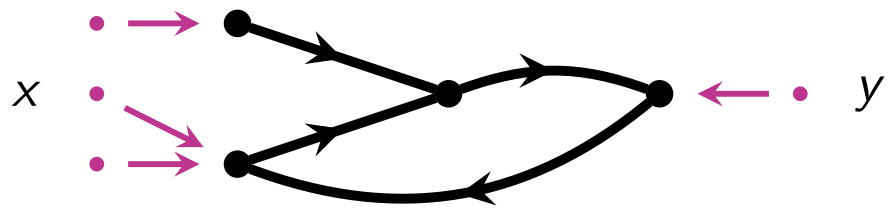
and $g : y \to z$



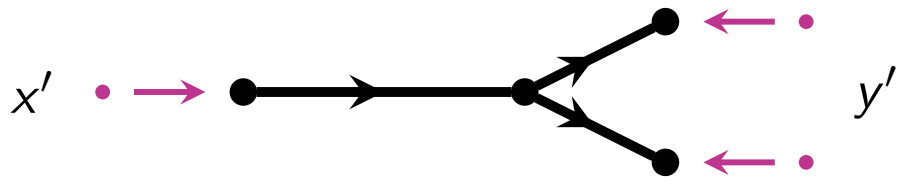
by gluing them together along y, obtaining an open graph $g \circ f : x \to z$



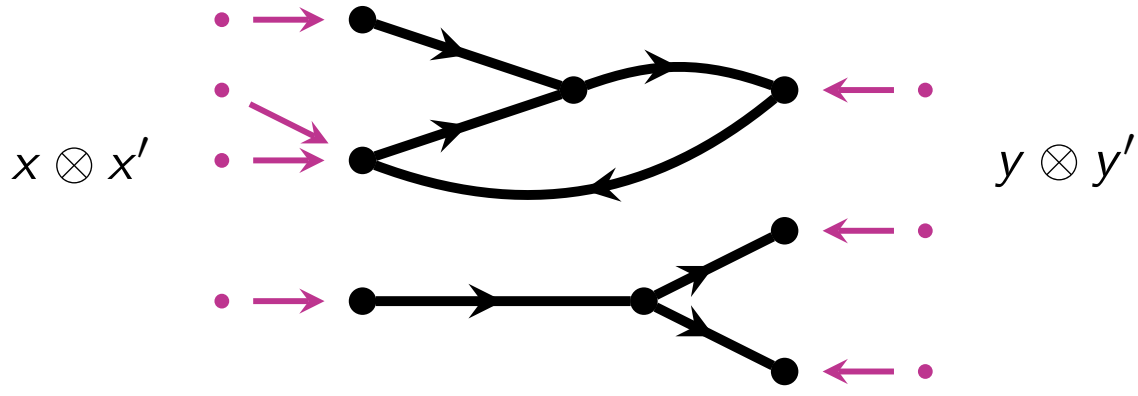
The category **OpenGraph** is monoidal: we can **tensor** open graphs $f : x \to y$



and $g : x' \to y'$

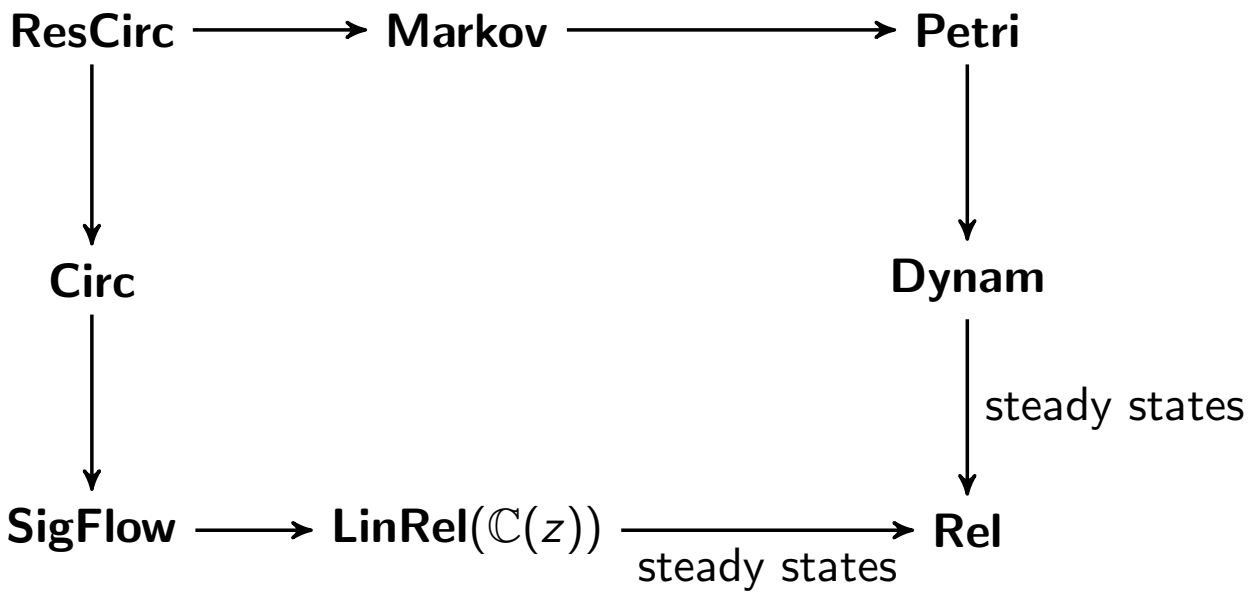


by setting them side by side, obtaining $f \otimes g : x \otimes x' \to y \otimes y'$



To use sets well, we need functions between sets. To use vector spaces well, we need linear maps. Similarly, to use categories we need "functors", which are maps between categories.

I gave lots of examples last time, without details.



Definition of a Functor

Given categories $\mathbf{C}$ and $\mathbf{D}$, a **functor** $F\colon \mathbf{C} \to \mathbf{D}$ consists of:

1. a map sending objects of $\mathbf{C}$ to objects of $\mathbf{D}$, called $F\colon \mathrm{Ob}(\mathbf{C}) \to \mathrm{Ob}(\mathbf{D})$,

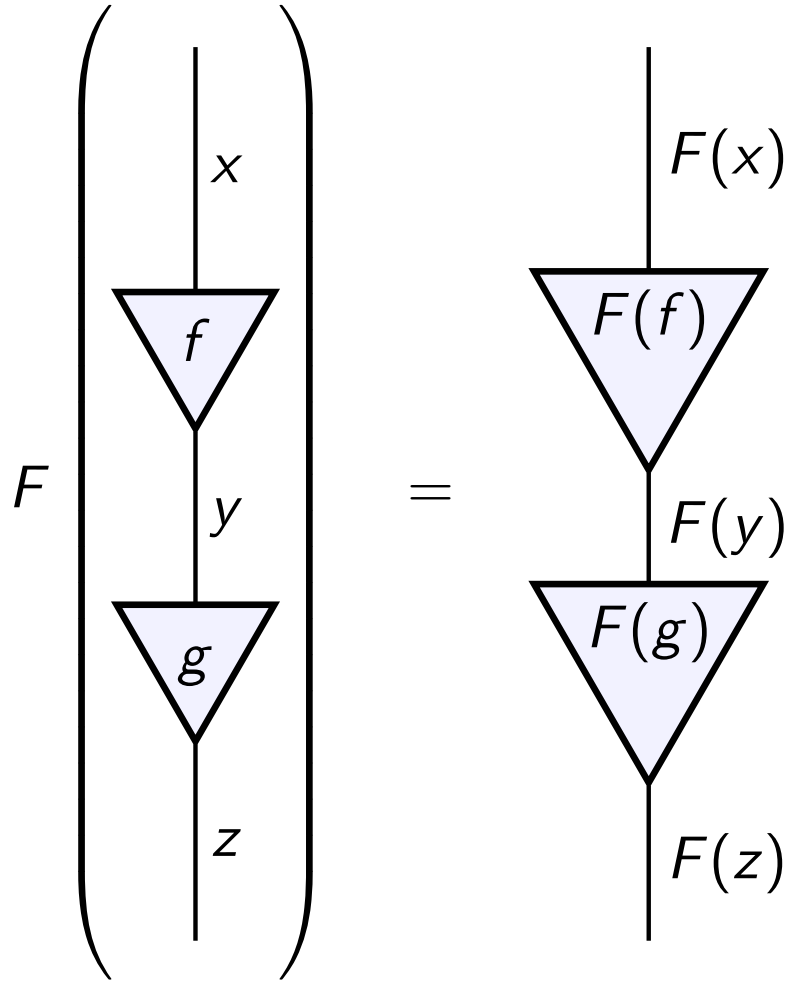
2. a map from morphisms of $\mathbf{C}$ to morphisms of $\mathbf{D}$, also called F, sending any morphism $g\colon x \to y$ in $\mathbf{C}$ to a morphism $F(g)\colon F(x) \to F(y)$,

such that:

3. F preserves composition: $F(g \circ f) = F(g) \circ F(f)$ for any composable morphisms f, g in $\mathbf{C}$;

4. F preserves identities: $F(1_x) = 1_{F(x)}$ for every $x \in \mathrm{Ob}(\mathbf{C})$.

F preserves composition, in pictures:



Definition of a Monoidal Functor

Given monoidal categories $(\mathbf{C}, \otimes)$ and $(\mathbf{D}, \otimes)$, a functor $F : \mathbf{C} \to \mathbf{D}$ is **(strict) monoidal** if:

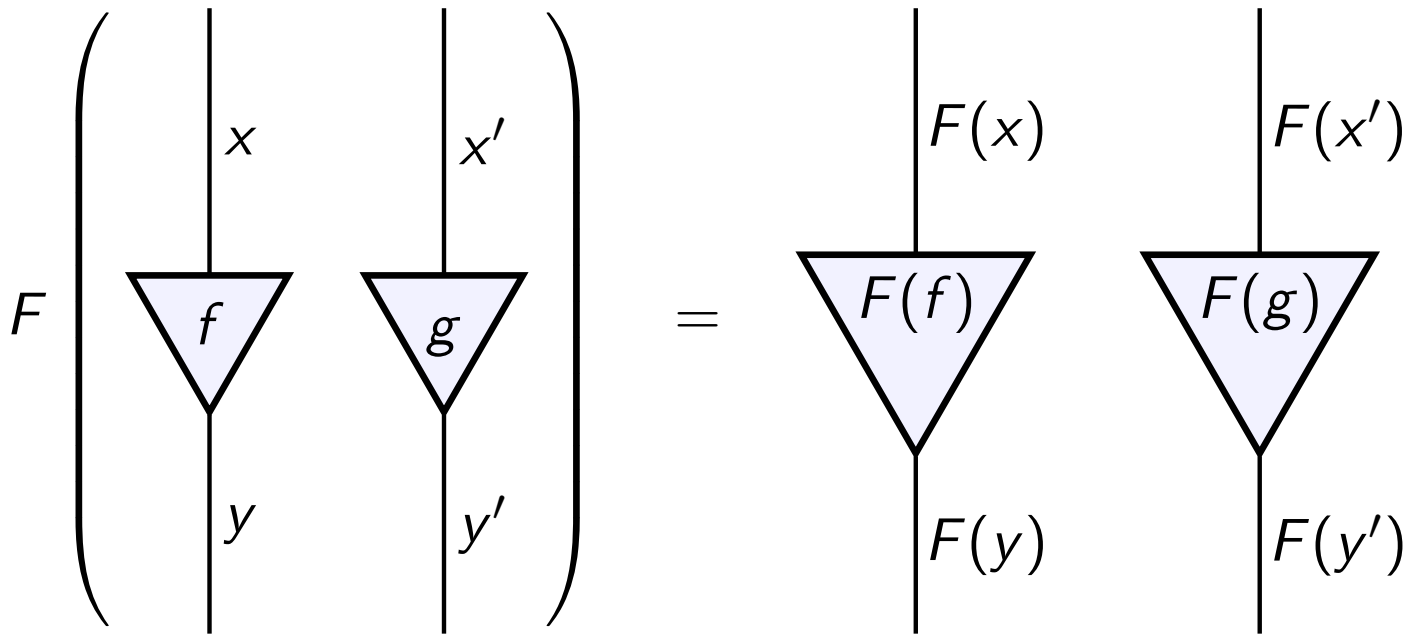
1. F preserves the tensor product of objects: for all $x, y \in \mathrm{Ob}(C)$ we have

$$F(x \otimes y) = F(x) \otimes F(y)$$

2. F preserves the tensor product of morphisms: for all $f : x \to y$ and $g : x' \to y'$ in $\mathbf{C}$ we have

$$F(f \otimes g) = F(f) \otimes F(g)$$

F preserves tensor products, in pictures:



3. F preserves the unit object:

$$F(I) = I$$

where at left I the unit object of $\mathbf{C}$, and at right it's the unit object of $\mathbf{D}$.

An easy example of a monoidal functor:

Let $F\colon (\mathbf{Set}, \times) \to (\mathbf{Vect}, \otimes)$ send any set S to the vector space $F(S)$ of formal linear combinations of elements of S: this has S as basis.

Given $f\colon S \to T$ in **Set**, let $F(f)\colon F(S) \to F(T)$ be the linear map that sends basis vectors to basis vectors, and equals f on basis vectors.

This has

$$F(g \circ f) = F(g) \circ F(f)$$

and

$$F(f \otimes g) = F(f) \otimes F(g)$$

Researchers have defined and studied monoidal categories where morphisms are open networks of many kinds, and monoidal functors between these:

- electrical circuits [5, 7, 9]
- Markov processes [2, 3, 6, 10]
- signal flow diagrams [8, 13, 16, 17]
- stock and flow diagrams [11, 12]
- bond graphs [15, 16]
- gene regulatory networks [1, 4, 12]
- classical mechanical systems [14, 18]

and more.

There are systematic ways to handle many of these examples: "structured and decorated cospans".

However, these is much left to understand!

Next time I'll explain the monoidal category of open stock-flow diagrams, and some category-based software that handles stock-flow diagrams and Petri nets — both of which are used in epidemiology.

But first, an overview of the project to develop this math and software.

Some Institutions and Software

- The Topos Institute (in Berkeley) and Topos Research UK (in Oxford)
- AlgebraicJulia, a category-based software platform in Julia, developed by researchers at Topos, Georgia Tech, etc.
- StockFlow, a tool for creating, composing and running stock and flow models, based on AlgebraicJulia.
- ModelCollab, a web-based tool for creating and composing stock and flow diagrams, relying on StockFlow.jl.
- CatColab, a new category-based software platform in Rust, developed at Topos and Topos UK.
- ARIA's £59m Safeguarded AI program, which funded Topos UK and many UK category theorists.
- A general framework for open systems, developed for the Safeguarded AI program: "the thesis".

References

[1] R. Aduddell, J. Fairbanks, A. Kumar, P. S. Ocal, E. Patterson and B. T. Shapiro, A compositional account of motifs, mechanisms, and dynamics in biochemical regulatory networks, *Compositionality* **6** (2024). Also available as arXiv:2301.01445.

[2] L. de Francesco Albasini, N. Sabadini and R. F. C. Walters, The compositional construction of Markov processes, *Appl. Categ. Str.* **19** (2011), 425–437. Also available as arXiv:0901.2434.

[3] L. de Francesco Albasini, N. Sabadini, and R. F. C. Walters, The compositional construction of Markov processes II, in *RAIRO Theor. Inform. Appl.* **45** (2011), 117–142. Also available as arXiv:1005.0949.

[4] J. C. Baez and A. Chaudhuri, Graphs with polarities. Available as arXiv:2506.23375.

[5] J. C. Baez and K. Courser, Structured cospans, *Theory Appl. Categ.* **35** (2020), 1771–1822. Also available as arXiv:1911.04630.

[6] J. C. Baez and K. Courser, Coarse-graining open Markov processes, *Theory Appl. Categ.* **33** (2018), 1223–1268. Also available as arXiv:1710.11343.

[7] J. C. Baez, B. Coya and F. Rebro, Props in circuit theory, *Theory Appl. Categ.* **33** (2018), 727–783. Available as arXiv:1707.08321.

[8] J. C. Baez and J. Erbele, Categories in control, *Theory Appl. Categ.* **30** (2015), 836–881. Also available as arXiv:1405.6881.

[9] J. C. Baez and B. Fong, A compositional framework for passive linear networks, *Theory Appl. Categ.* **33** (2018), 1158–1222. Also available as arXiv:1504.05625.

[10] J. C. Baez, B. Fong and B. S. Pollard, A compositional framework for Markov processes, *Jour. Math. Phys.* **57** (2016), 033301. Also available as arXiv:1508.06448.

[11] J. C. Baez, X. Li, S. Libkind, N. D. Osgood and E. Redekopp, A categorical framework for modeling with stock and flow diagrams, in *Mathematics of Public Health: Mathematical Modelling from the Next Generation*, eds. J. David and J. Wu, Springer, 2023, pp. 175–207. Also available as arXiv:2211.01290.

[12] J. C. Baez, X. Li, S. Libkind, N. D. Osgood and E. Patterson, Compositional modeling with stock and flow diagrams, *Electron. Proc. Theor. Comput. Sci.* **380** (2023), 77–96. Also available as arXiv:2205.08373.

[13] F. Bonchi, P. Sobociński and F. Zanasi, A categorical semantics of signal flow graphs, in *CONCUR 2014–Concurrency Theory*, eds. P. Baldan and D. Gorla, Lecture Notes in Computer Science **8704**, Springer, Berlin, 2014, pp. 435–450. Also available at http://users.ecs.soton.ac.uk/ps/papers/sfg.pdf.

[14] J. C. Baez, D. Weisbart and A. M. Yassine, Open systems in classical mechanics, *Jour. Math. Phys.* **62** (2021), 042902. Also available as arXiv:1710.11392.

[15] B. Coya, A compositional framework for bond graphs. Available as arXiv:1710.00098.

[16] B. Coya, *Circuits, Bond Graphs, and Signal-Flow Diagrams: A Categorical Perspective*, Ph.D. thesis, Department of Mathematics, U. C. Riverside, 2018. Available as arXiv:1805.08290.

[17] B. Fong, P. Rapisarda and P. Sobocinski, A categorical approach to open and interconnected dynamical systems, in *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, IEEE, New York, 2016, pp. 1–10. Available as arXiv:1510.05076.

[18] O. Lynch, *Relational Composition of Physical Systems: A Categorical Approach*, M.Sc. Thesis, Department of Physics, Universiteit Utrecht, 2022. Available at arXiv:2310.06088.