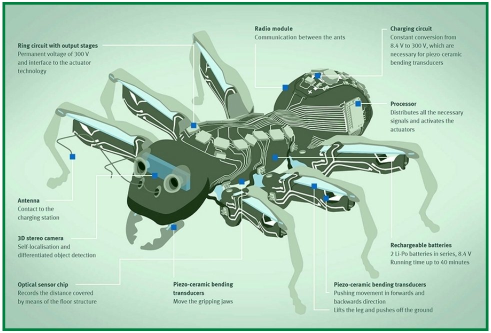
Category Theory and Systems

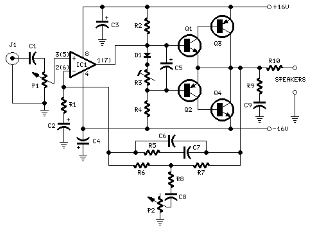


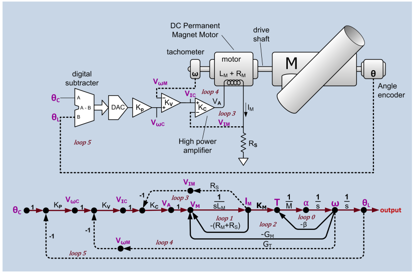
Ring circuit with output stages
Permanent voltage of 300 V and interface to the actuator technology
Radio module
Communication between the ants
Charging circuit
Constant conversion from 8.4 V to 300 V, which are necessary for piezo-ceramic bending transducers
Processor
Distributes all the necessary signals and activates the actuators
Antenna
Contact to the charging station
3D stereo camera
Self-localisation and differentiated object detection
Rechargeable batteries
2 Li-Po batteries in series, 8.4 V
Running time up to 40 minutes
Optical sensor chip
Records the distance covered by means of the floor structure
Piezo-ceramic bending transducers
Move the gripping jaws
Piezo-ceramic bending transducers
Pushing movement in forwards and backwards direction
Lifts the leg and pushes off the ground

John Baez, U. C. Riverside

Compositional Robotics: Mathematics and Tools
ICRA 2021

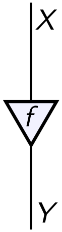
Compositional design requires a formalism for assembling networks of many kinds:





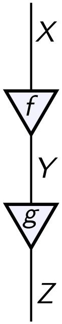
We can think of a network of any particular kind as a morphism in some "symmetric monoidal category".

Categories are great for describing networks. A network with input X and output Y is a **morphism** $f\colon X \to Y$, and we can draw it like this:



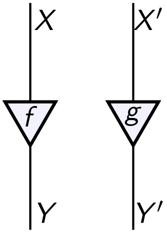
X and Y are **objects** of the category.

We can attach networks in series if the output of the first equals the input of the second:



Here we are **composing** morphisms $f\colon X \to Y$ and $g\colon Y \to Z$ to get a morphism $g \circ f\colon X \to Z$.

In a **monoidal** category, we can also put networks "in parallel":

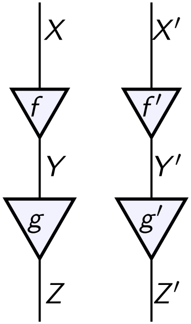


We say we are **tensoring** $f\colon X \to Y$ and $g\colon X' \to Y'$ to get the morphism $f \otimes g\colon X \otimes X' \to Y \otimes Y'$.

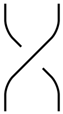
In a monoidal category, composition and tensoring must obey some laws, which all look obvious when drawn as diagrams. for example

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

says two ways of reading this diagram agree:

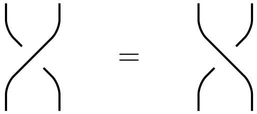


In a **braided monoidal category** we also have morphisms $B_{x,y} \colon x \otimes y \to y \otimes x$ called **braidings**:

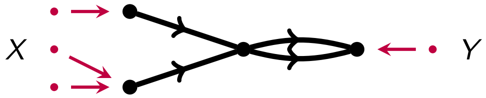


These let us draw diagrams where wires cross. Again, some obvious-looking laws must hold.

A braided monoidal category is **symmetric** if it doesn't matter which wire crosses over which:



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



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

- stuff (matter, energy, signals, etc.) can flow in or out;
- we can combine systems to form larger systems by *composition* and *tensoring*.

To use design open systems, we follow Lawvere's idea of "functorial semantics":

- Networks of some kind, with specified input and outputs, will be morphisms in some symmetric monoidal category $\mathbf{X}$.
- To describe what these networks *actually do*, we use a map $F\colon \mathbf{X} \to \mathbf{Y}$ that sends any network to its behavior. The *behavior* of a network is a morphism in $\mathbf{Y}$.

For example $\mathbf{X}$ could have circuit diagrams as morphisms, and $\mathbf{Y}$ could have differential equations as morphisms.

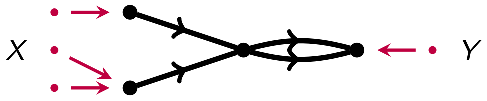
A map $F\colon \mathbf{X} \to \mathbf{Y}$ between symmetric monoidal categories is called a "symmetric monoidal functor".

We can also design using more layers of abstraction:

$$\mathbf{X} \xrightarrow{F} \mathbf{Y} \xrightarrow{G} \mathbf{Z} \to \cdots$$

How can we construct symmetric monoidal categories and functors to implement this strategy for designing open systems?

One way, pioneered by Brendan Fong, is to use "decorated cospans". For example, this:

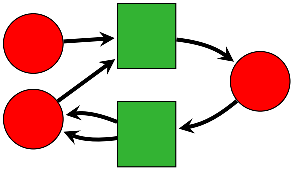


is really a cospan of finite sets:

$$\begin{tikzcd} & S & \\ X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \end{tikzcd}$$

where S is "decorated" with extra structure making it into the set of vertices of a graph.

Let's look at a more interesting example: Petri nets.

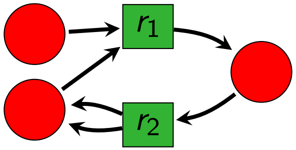


A **Petri net** is a bipartite graph. The two kinds of vertices are called **places** and **transitions**.

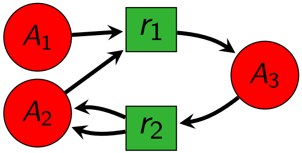
In computer science, Petri nets became popular as models of concurrency starting in the 1970s.

"Petri nets with rates" can be used to describe a large class of first-order differential equations with polynomial coefficients.

In a Petri net **with rates**, each transition is assigned a **rate constant**: a positive real number. We can then write down a **rate equation** describing dynamics. For example, this Petri net with rates:



In a Petri net **with rates**, each transition is assigned a **rate constant**: a positive real number. We can then write down a **rate equation** describing dynamics. For example, this Petri net with rates:



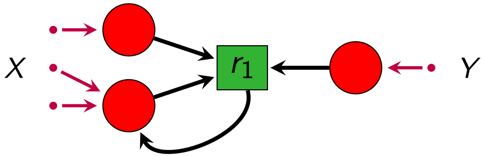
gives this rate equation:

$$\begin{aligned}
\frac{dA_1}{dt} &= -r_1\,A_1A_2 \\
\frac{dA_2}{dt} &= -r_1\,A_1A_2 + 2r_2\,A_3 \\
\frac{dA_3}{dt} &= r_1\,A_1A_2 - r_2\,A_3
\end{aligned}$$

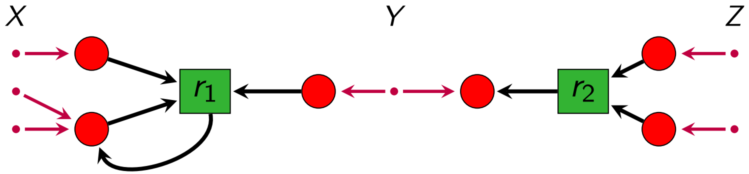
So far these Petri nets describe *closed* systems.

But there's a symmetric monoidal category of **open Petri nets with rates**, called **Petri**, where:

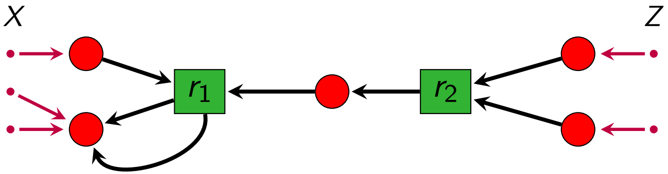
- an object is a finite set;
- a morphism $f\colon X \to Y$ is a Petri net with rates together with functions from X and Y to its set of places:



- To compose morphisms $f\colon X \to Y$ and $g\colon Y \to Z$:

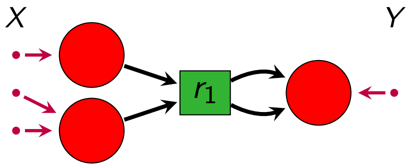


we put them in series, identifying outputs of f with inputs of g:

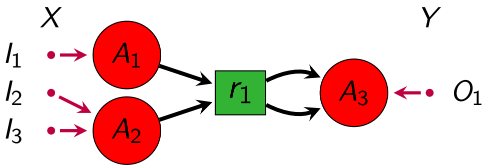


- To tensor morphisms, we put them in parallel.

An open Petri net with rates $f\colon X \to Y$ gives an **open rate equation** involving flows in and out, which can be arbitrary smooth functions of time. For example this:



An open Petri net with rates $f\colon X \to Y$ gives an **open rate equation** involving flows in and out, which can be arbitrary smooth functions of time. For example this:



gives:

$$\frac{dA_1}{dt} = -r_1\, A_1 A_2 + I_1(t)$$

$$\frac{dA_2}{dt} = -r_1\, A_1 A_2 + I_2(t) + I_3(t)$$

$$\frac{dA_3}{dt} = 2r_1\, A_1 A_2 - O_1(t)$$

Let's understand this using functorial semantics! We'll get a symmetric monoidal functor

$$\square \colon \mathbf{Petri} \to \mathbf{Dynam}$$

Other choices of semantics correspond to other symmetric monoidal functors.

There is a symmetric monoidal category **Dynam** where:

- an object is a finite set;
- a morphism $f\colon X \to Y$ is an **open dynamical system**, meaning a cospan of finite sets

$$\begin{tikzcd} & S & \\ X \arrow[ur, hook, "i"] & & Y \arrow[ul, hook', "o"'] \end{tikzcd}$$

  equipped with a smooth vector field v on $\mathbb{R}^S$.

Given time-dependent inputs and outputs $I\colon \mathbb{R} \to \mathbb{R}^X$, $O\colon \mathbb{R} \to \mathbb{R}^Y$, an open dynamical system gives an open rate equation as in the example.

Theorem (JB–Blake Pollard)

There is a symmetric monoidal functor $\square\colon \mathbf{Petri} \to \mathbf{Dynam}$ *sending any open Petri net with rates to its open dynamical system.*

This is a statement of *compositionality*: we can determine the rate equation of a Petri net with rates by breaking it down into a composite and/or tensor product of simpler *open* Petri nets with rates, and repeatedly using:

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

$$\square(f \otimes g) = \square(f) \otimes \square(g).$$

How do we show this? We use Fong's theory of decorated cospans, later refined by Kenny Courser, Christina Vasilakopolou and myself.

Let **FinSet** be the category of finite sets and functions. Given cospans in **FinSet** like this:

$$\begin{tikzcd}
 & S & & & S' & \\
X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] & Y \arrow[ur, "i'"] & & Z \arrow[ul, "o'"']
\end{tikzcd}$$

we can compose them by taking a pushout:

$$\begin{tikzcd}
 & & S +_Y S' & & \\
 & S \arrow[ur] & & S' \arrow[ul] & \\
X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \arrow[ur, "i'"] & & Z \arrow[ul, "o'"']
\end{tikzcd}$$

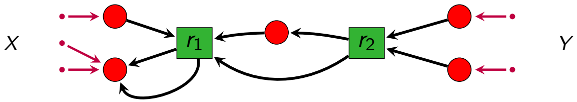
Here the **pushout** $S +_Y S'$ is the disjoint union $S + S'$ modulo the smallest equivalence relation such that $o(y) \sim i'(y)$ for all $y \in Y$.

Next, if we choose a functor $F\colon \mathbf{FinSet} \to \mathbf{Cat}$, we can try to build a category where a morphism is a **decorated cospan**: a cospan of finite sets

$$\begin{tikzcd} & S & \\ X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \end{tikzcd}$$

with a **decoration** $d \in F(S)$.

To build **Petri**, we take $F(S)$ to be the category of all Petri nets with rates having S as their set of places. Then a decorated cospan looks like this:



But how do we compose decorated cospans?

Given decorated cospans like this:

$$\begin{tikzcd} & S & \\ X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \end{tikzcd} \qquad \begin{tikzcd} & S' & \\ Y \arrow[ur, "i"] & & Z \arrow[ul, "o"'] \end{tikzcd}$$

$$d \in F(S) \qquad\qquad d' \in F(S')$$

we compose the cospans by taking a pushout as before.

We compose the decorations by taking $(d, d') \in F(S) \times F(S')$ and applying the composite function

$$F(S) \times F(S') \longrightarrow F(S + S') \longrightarrow F(S +_Y S')$$

where the first step comes from F being "lax monoidal".

Theorem (JB, K. Courser, C. Vasilakopolou)

Suppose that

$$F\colon (\mathbf{FinSet}, +) \longrightarrow (\mathbf{Cat}, \times)$$

is a lax symmetric monoidal functor. Then there is a symmetric monoidal category of ***F*-decorated cospans**, $F\mathbf{Cospan}$, *where:*

- *an object is a finite set;*
- *a morphism from X to Y is a cospan of finite sets*

$$\begin{tikzcd} & S & \\ X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \end{tikzcd}$$

together with a **decoration** $d \in F(S)$.

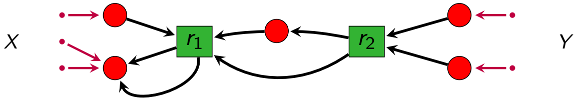
As a consequence, there is a symmetric monoidal category **Petri** where:

- an object is a finite set;
- a morphism $f\colon X \to Y$ is a cospan of finite sets

$$\begin{tikzcd} & S & \\ X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \end{tikzcd}$$

  together with a Petri net with rates having S as its set of places.

So, a morphism in **Petri** looks like this:



Also as a consequence, there is a symmetric monoidal category **Dynam** where:

- an object is a finite set;
- a morphism $f\colon X \to Y$ is a cospan of finite sets

$$\begin{tikzcd} & S & \\ X \arrow[ur, "i"] & & Y \arrow[ul, "o"'] \end{tikzcd}$$

  together with a smooth vector field on $\mathbb{R}^S$.

The theory also lets us build maps between decorated cospan categories, as needed in functorial semantics:

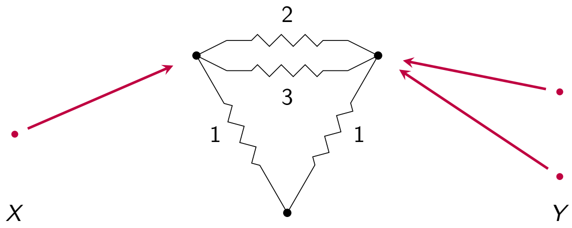
Theorem (JB, K. Courser, B. Pollard, C. Vasilakopolou)
There is a symmetric monoidal functor $\Box \colon \mathbf{Petri} \to \mathbf{Dynam}$ *sending any open Petri net with rates to the corresponding open dynamical system.*

Note: this was just an example chosen to illustrate ideas — not necessarily of most importance to robotics!

Decorated cospans and other methods let us work with many symmetric monoidal categories of networks.

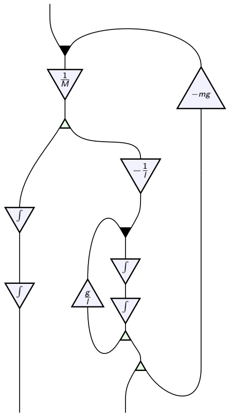
- **Electrical circuits**

  • JB and B. Fong, A compositional framework for passive linear networks.
  • JB, Brandon Coya and Franciscus Rebro, Props in network theory.



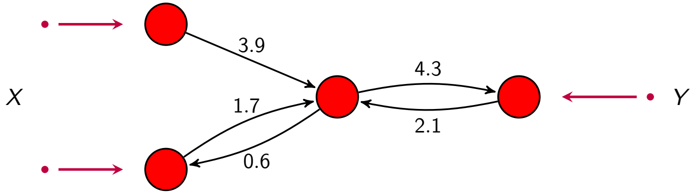
- **Signal-flow graphs in control theory**

  • JB and Jason Erbele, Categories in control.
  • Fillipo Bonchi, Pawel Sobocinski and Fabio Zanasi, The calculus of signal flow diagrams I.



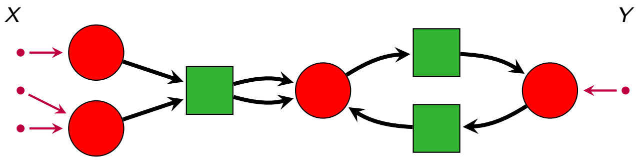
- **Markov processes**

  - JB, B. Fong and B. Pollard, A compositional framework for Markov processes.
  - JB and K. Courser, Coarse-graining open Markov processes.



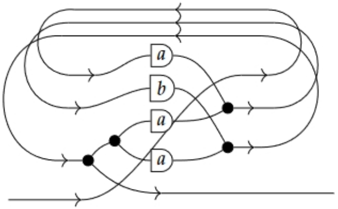
- **Petri nets**

  - JB and Jade Master, Open Petri nets.



- **Finite state machines**

  - Robin Piedeleu and Fabio Zanasi, A string diagrammatic axiomatisation of finite-state automata.



All these frameworks are — or will be — unified by a "network of network languages". In other words, useful symmetric monoidal categories are connected by symmetric monoidal functors:

$$\begin{tikzcd}
\mathbf{ResCirc} \arrow[r] \arrow[d] & \mathbf{Markov} \arrow[r] \arrow[ddl] & \mathbf{Petri} \arrow[d, "\square"] \\
\mathbf{Circ} \arrow[d] & & \mathbf{Dynam} \arrow[d, "\text{steady states}"] \\
\mathbf{SigFlow} \arrow[r, "\blacksquare"'] & \mathbf{LinRel}(\mathbb{C}(z)) \arrow[r, "\text{steady states}"'] & \mathbf{Rel}
\end{tikzcd}$$

For the overall philosophy, read:

- Brendan Fong, *The Algebra of Open and Interconnected Systems*.