

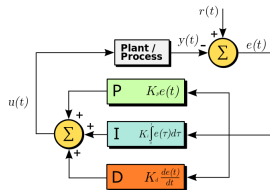
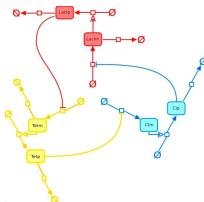
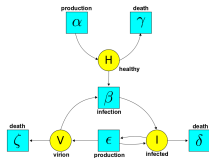
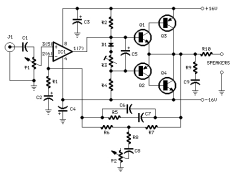
Compositionality in Network Theory



John Baez, U. C. Riverside

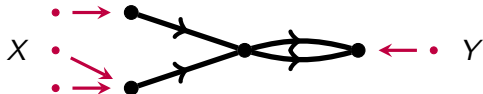
Simons Institute of Computing,
workshop on *Compositionality*, 6 December 2016

In many areas of science and engineering, people use *networks*, drawn as boxes connected by wires:



We need a good general theory of these!

Networks of some particular kind, with specified inputs and outputs, can be seen as morphisms in some symmetric 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* and *tensoring*.

To use networks as a 'syntax' for open systems, we follow the ideas of 'functorial semantics':

- ▶ Networks of some kind, with specified input and outputs, will be morphisms in some symmetric monoidal category \mathbf{X} .
- ▶ To 'interpret' these networks we use a symmetric monoidal functor $F: \mathbf{X} \rightarrow \mathbf{Y}$, where \mathbf{Y} is a symmetric monoidal category good for semantics, e.g. **Set** or **Rel**.

How can we construct symmetric monoidal categories with networks as morphisms?

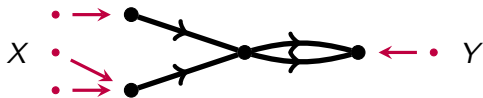
One way is to use generators and relations.

We can present a (strict) symmetric monoidal category by specifying:

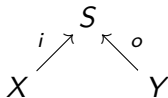
- ▶ a set of generating objects;
- ▶ a set of generating morphisms;
- ▶ a set of relations between morphisms.

We can then specify a (strict) symmetric monoidal functor by sending generators to generators in such a way that relations are preserved.

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

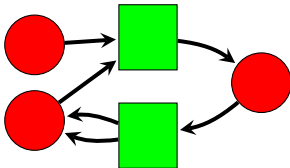


is really a cospan of finite sets:



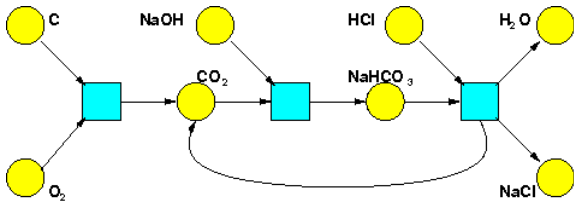
where S is ‘decorated’ with extra structure making it into the set of vertices of a graph: $E \begin{smallmatrix} \xrightarrow{s} \\ \xrightarrow{t} \end{smallmatrix} S$.

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

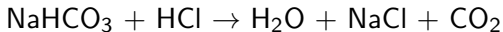
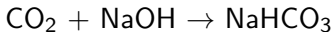
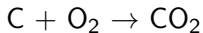


A **Petri net** is a bipartite graph. The two kinds of vertices are called **places** and **transitions**.

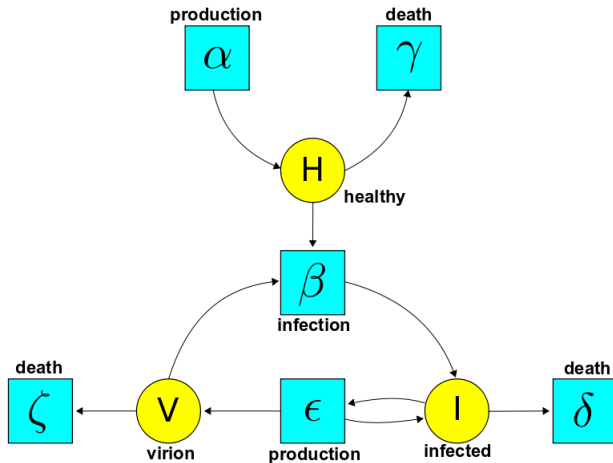
In computer science, Petri nets became popular as models of concurrency starting in the 1970s. But they were invented for chemistry in 1939:



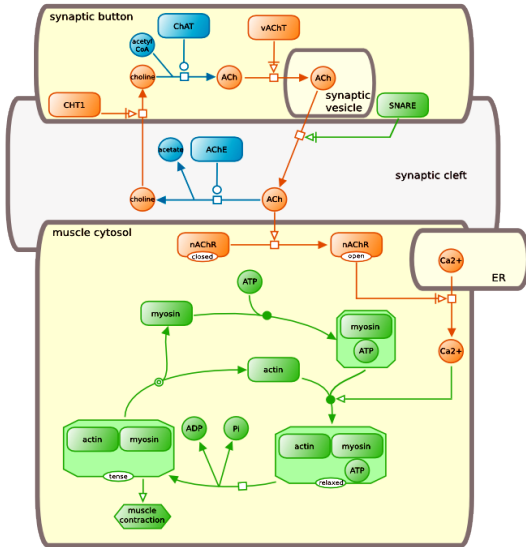
as an alternative to the more familiar [reaction networks](#):



Now they're used in epidemiology...

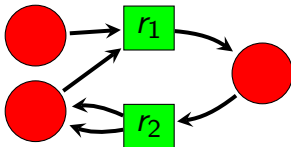


...systems biology...

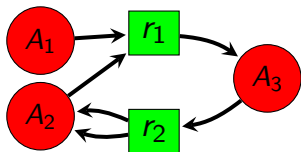


... and many other fields.

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:

$$\frac{dA_1}{dt} = -r_1 A_1 A_2$$

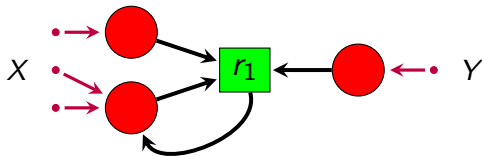
$$\frac{dA_2}{dt} = -r_1 A_1 A_2 + 2r_2 A_3$$

$$\frac{dA_3}{dt} = r_1 A_1 A_2 - r_2 A_3$$

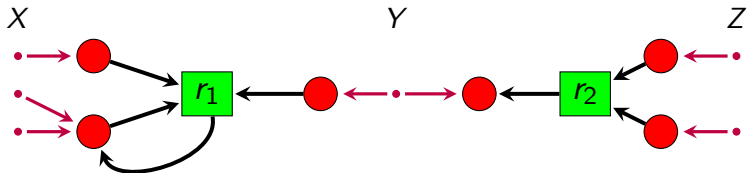
So far these Petri nets describe *closed* systems.

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

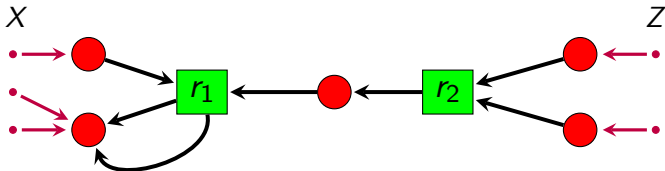
- ▶ an object is a finite set;
- ▶ a morphism $f: X \rightarrow Y$ is a Petri net with rates together with functions from X and Y to its set of places:



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

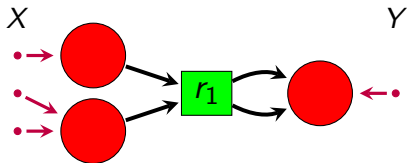


we put them in series, identifying outputs of f with inputs of g :

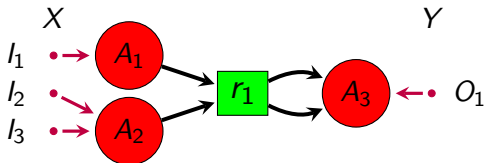


- To tensor morphisms, we put them in parallel.

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



gives:

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

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

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

So: open Petri nets with rates serve as a ‘syntax’, with open dynamical systems providing one possible ‘semantics’.

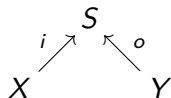
Let’s understand this using functorial semantics! We’ll get a symmetric monoidal functor

$$\square: \mathbf{Petri} \rightarrow \mathbf{Dynam}$$

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 $f: X \rightarrow Y$ is an **open dynamical system**, meaning a cospan of finite sets



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

Given input and output flows $I(t) \in \mathbb{R}^X$, $O(t) \in \mathbb{R}^Y$, an open dynamical system describes how a point $A(t) \in \mathbb{R}^S$ changes with time:

$$\frac{d}{dt}A(t) = v(A(t)) + i_*(I(t)) - o_*(O(t))$$

where i_* , o_* push forward \mathbb{R} -valued functions from X , Y to S .

Theorem (Baez–Pollard)

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

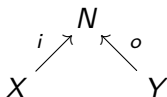
This is a statement of *compositionality*: we can determine the rate equation of a Petri net with rates by breaking it down into a composite and/or tensor product of simpler *open* Petri nets with rates, and repeatedly using:

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

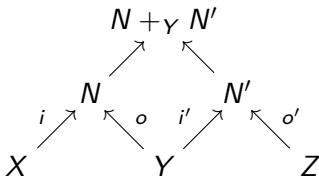
$$\square(f \otimes g) = \square(f) \otimes \square(g).$$

How do we prove this theorem? We use [Fong's theory of decorated cospan](#)s.

Say we start with a category \mathbf{C} with finite colimits: in our example, $\mathbf{C} = \mathbf{FinSet}$. We can build a bicategory where morphisms are cospan in \mathbf{C} :



and composition is done by pushout:



Pushouts are defined only up to isomorphism, which is why we get a bicategory. But there is a category **Cospan**(**C**) where morphisms are *isomorphism classes* of cospans in **C**.

Next, if we choose a functor $F: \mathbf{C} \rightarrow \mathbf{Set}$, we can try to build a category where a morphism is an isomorphism class of cospans

$$\begin{array}{ccc} & S & \\ i \nearrow & & \nwarrow o \\ X & & Y \end{array}$$

with S 'decorated' by an element of $F(S)$.

In the case of **Petri**, $F(S)$ is the set of all ways of making S into the set of places in some Petri net with rates.

But how do we 'compose the decorations' when we compose cospans?

Given composable morphisms

$$\begin{array}{ccc} & S & \\ i \nearrow & & \nwarrow o \\ X & & Y \end{array} \quad d \in F(S)$$

$$\begin{array}{ccc} & S' & \\ i \nearrow & & \nwarrow o \\ Y & & Z \end{array} \quad d' \in F(S')$$

we compose the cospans by taking a pushout. 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 a *lax* monoidal functor.

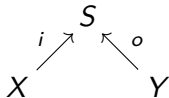
Theorem (Fong)

Suppose that \mathbf{C} has finite colimits and

$$F: (\mathbf{C}, +) \longrightarrow (\mathbf{Set}, \times)$$

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

- ▶ an object is an object of \mathbf{C} ;
- ▶ a morphism from X to Y is a cospan



together with a **decoration** $d \in F(S)$. (Just kidding: actually, a morphism is an isomorphism class of these!)

Corollary (Baez–Pollard)

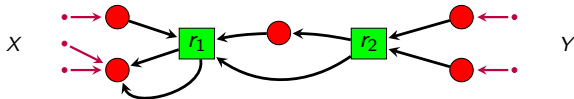
There is a symmetric monoidal category **Petri** where:

- ▶ an object is a finite set;
- ▶ a morphism $f: X \rightarrow Y$ is a cospan of finite sets

$$\begin{array}{ccc} & S & \\ i \nearrow & & \nwarrow o \\ X & & Y \end{array}$$

together with a Petri net with rates having S as its set of places. (Just kidding: actually an isomorphism class of such cospans!)

So, a morphism looks like this:



Corollary (Baez–Pollard)

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

- ▶ an object is a finite set;
- ▶ a morphism $f : X \rightarrow Y$ is a cospan of finite sets

$$\begin{array}{ccc} & S & \\ i \nearrow & & \nwarrow o \\ X & & Y \end{array}$$

together with a smooth vector field on \mathbb{R}^S . (Just kidding: actually an isomorphism class of these!)

Next, how do we get our symmetric monoidal functor

$$\square: \mathbf{Petri} \rightarrow \mathbf{Dynam} ?$$

Theorem (Fong)

Suppose \mathbf{C} has finite colimits,

$$F, G: (\mathbf{C}, +) \longrightarrow (\mathbf{Set}, \times)$$

are lax symmetric monoidal functors, and

$$\theta: F \Rightarrow G$$

is a monoidal natural transformation. Then we obtain a symmetric monoidal functor

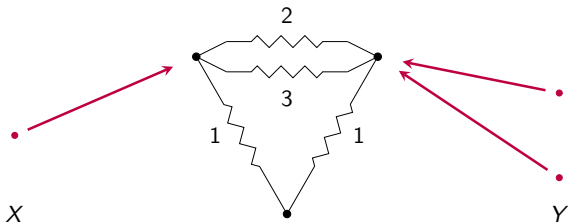
$$T_\theta: F\mathbf{Cospan} \rightarrow G\mathbf{Cospan}.$$

Corollary (Baez–Pollard)

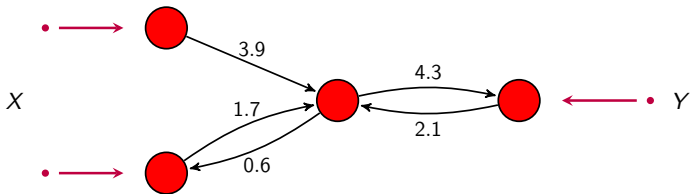
There is a symmetric monoidal functor $\square: \mathbf{Petri} \rightarrow \mathbf{Dynam}$ sending any open Petri net with rates to the corresponding open dynamical system.

The same methods, and also the ‘generators and relations’ approach, let us study many categories of networks — and how they’re connected by functors.

- **Electrical circuits:** Baez and Fong, [A compositional framework for passive linear networks](#).



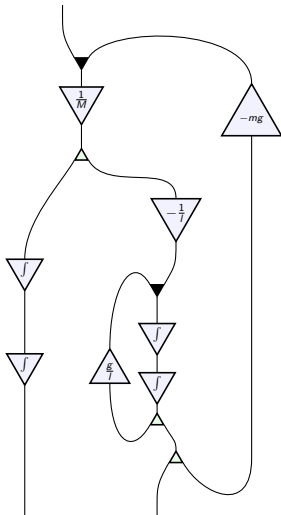
- **Markov processes:** Baez, Fong and Pollard, [A compositional framework for Markov processes](#).



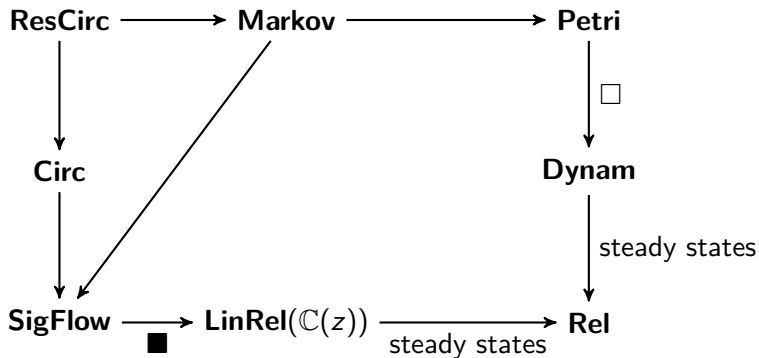
► **Signal-flow graphs in control theory:**

Erbele, *Categories in Control: Applied PROPs*.

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



We find a 'network of network languages': interesting symmetric monoidal categories connected by symmetric monoidal functors:



There is also more to say about decorated cospans! For example:

Theorem (Courser)

Suppose that \mathbf{C} has finite colimits and $F: (\mathbf{C}, +) \longrightarrow (\mathbf{Set}, \times)$ is a lax symmetric monoidal functor. Then there is a symmetric monoidal *bicategory* where:

- ▶ an object is an object of \mathbf{C} ;

- ▶ a morphism from X to Y is a cospan $X \begin{array}{c} \nearrow^i \\ S \\ \nwarrow_o \end{array} Y$ together with a decoration $d \in F(S)$;

- ▶ a 2-morphism is a map of cospans $X \begin{array}{ccc} & S & \\ i \nearrow & \downarrow f & \nwarrow o \\ & S' & \\ i' \searrow & & \swarrow o' \end{array} Y$ such that

$F(f)$ maps $d \in F(S)$ to the decoration $d' \in F(S')$.