NOTES ON CATEGORICAL SYSTEMS THEORY

MATTEO CAPUCCI
UNIVERSITY OF STRATHCLYDE

MARCH 15, 2023

Abstract. Categorical systems theory (CST) is a doubly-categorical framework for approaching and developing structural theories of systems and behaviour of any kind. These notes are meant to be (1) a basic reference for CST and (2) a source of examples and working developments. They're under active development, so quite far from camera-ready, and possibly never complete.

1. Introduction

Categorical Systems Theory (CST) is a categorical framework for the abstract study of systems irrespective of their contingent aspects. Instead of espousing a specific paradigm on the mathematical specification of systems, CST predicates upon the general features such paradigms should have. In doing so, it captures the essence of the notion of 'system'.

The object of study of CST is a *doctrine of systems*, which is defined in [Mye22] as way to answer the following questions:

- What does it mean to be a system? Does it have a notion of states, or of behaviours? Or is it a diagram describing the way some primitive parts are organized?
- What should the interface of a system be?
- How can interfaces be connected in composition patterns?
- How are systems composed through composition patterns between their interfaces.
- What is a map between systems, and how does it affect their interfaces?
- When can maps between systems be composed along the same composition patterns as the systems.

A doctrine of systems specializes in many different *theories of systems*. For instance there is a doctrine of open dynamical systems encompassing the theory of deterministic dynamical systems, the theory of stochastic dynamical systems, the theory of differential dynamical systems, and many more. Hence it's usually easier to start describing what a theory of systems is and then to say what does it mean for a doctrine to gather many of them in a single object.

1.1. References.

Categorical approaches to general systems theory have been around for a long time. The earliest is probably [Ros78], which deals with systems from an behavioural point of view. The work pioneers the idea of gathering systems in categories and considering their observable behaviour as input-output relations.

In the 90s, a series of papers by Katis, Sabadini, Walters and others established a theory of machines [SW93; BSW96; KSW97a; KSW97b; KSW99; KSW02] which anticipated some of the

ideas regarding functorial semantics of behaviour, finding categories of matrices are well-adapted to receive such functors.

The works on structured and decorated cospans [FS07; Fon15; BM20; BC20; BCV22] deals with the doctrine of port-plugging systems (circuits) in modern form, and started using some ideas from double category theory to talk about them. This also happens in [Ler18; CGKS20].

In [Spi13; LBPF21] and other related works the idea operads provide the syntax for composition of systems is introduced.

Thus we can say that albeit the current form of CST (as the yoga of doubly indexed category and 'copresheaves' over them) is due to David Jaz Myers, various pieces of this philosophy have been around for far longer, as well as specific incarnations of them.

In this sense, we might be at the boundary between pre-paradigmatic phase and a period of normal science in the categorical study of systems. This is conditional on the ability of CST to establish itself as the dominant paradigm. In fact, the subject is still in its infancy.

At the moment, most of CST lives in Myers' own book [Mye22], itself a longer version of the shorter preprint [Mye20c] (where the notion of *doctrine* wasn't yet developed). Moreover, Myers has given a few talks about the topic in the past years:

(1) D. J. Myers. *A general definition of open dynamical system*. 2020. URL: https://www.youtube.com/watch?v=8T-Km3taNko
(2) D. J. Myers. *Open dynamical systems, trajectories and hierarchical planning*. 2020. URL: https://www.youtube.com/watch?v=3FxeY5DbPn0
(3) D. J. Myers. *Double Categories of Open Dynamical Systems*. 2020. URL: https://www.youtube.com/watch?v=f9fjf9lo2_M
(4) D. J. Myers. *Paradigms of composition*. 2021. URL: https://www.youtube.com/watch?v=50s62D5Ah-M

I also gave a talk about CST and its extension to cybernetic systems:

(5) M. Capucci. *From categorical systems theory to categorical cybernetics*. 2022. URL: https://www.youtube.com/watch?v=wtgfyjFIHBQ

1.2. **A quick tour of CST.** Categorical systems theory is a conceptually simple, if mathematically sophisticated, framework. In a nutshell, it studies processes connecting systems, and the ways these behave. Processes are organized in (monoidal double) categories, which themselves index categories of systems, whose maps in (the behavioural theory of) sets are behaviours.

If one is not at ease with double categories, at a first approximation one can drop the horizontal direction and think of these as monoidal categories of processes. They index sets of systems which can be reindexed by processes. One can study behaviour by specifying the set of ways interfaces can be observed, the relations processes induce between observations on their interfaces, and the states systems can be in and the observables these expose.

However, none of the two dimensions in CST is ancillary to the other. The horizontal direction is often overlooked in pre-CST work, but it's extremely natural to consider: from a categorical standpoint, we study things (here, systems and processes) by looking at the way they map into each other.

Thus the first step in CST is to understand **processes organize in monoidal double categories**:

$$\mathbb{P} := \left\{ \begin{array}{ccc} \cdot & \xrightarrow{\text{map of interfaces}} & \cdot \\ \downarrow{\scriptstyle \text{process}} & \Rightarrow{\scriptstyle \text{map of processes}} & \downarrow{\scriptstyle \text{process}} \\ \cdot & \xrightarrow{\text{map of interfaces}} & \cdot \end{array} \right\} \tag{1.1}$$

Such processes are actually 'composition patterns' that can be used to weave systems together, i.e. the ways parts can come together to form wholes. These can be wiring diagrams, or bubble diagrams, or circuit diagrams, etc. Both ways of thinking about them can be useful.

Mathematically speaking, **processes index systems**, giving rise to doubly indexed categories called **systems theories**:

$$\mathbf{Sys} : \mathbb{P}^{\top} \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at} \tag{1.2}$$

Thus, and this is a fundamental idea in CST, systems and processes are formally distinguished, even though they might end up being quite similar. In fact, in many instances, systems are special instances of processes which are considered stateful. The categories of systems over a given interface are categories of structure-preserving morphisms of systems, which we call simulations here. These can be more or less rigid depending on the user's taste.

Finally, systems are as interesting as the things they do. The observations we can make of a system are its behaviour. Ways to observe systems in a given theory are **theories of behaviour**, which are maps into the 'behavioural theory':

$$\begin{array}{ccc} \mathbb{P}^{\top} & \xrightarrow{\mathbf{Sys}} & \\ {\scriptstyle B^{\top}}\downarrow & \Downarrow{\scriptstyle B^{\flat}} & \mathbb{C}\mathbf{at} \\ \mathbb{S}\mathbf{et}^{\top} & \xrightarrow{\mathbf{Set}/-} & \end{array} \tag{1.3}$$

Here $\mathbb{S}\mathbf{et}$ is the double category of spans in $\mathbf{Set}$ and $\mathbb{C}\mathbf{at}$ is the double category of functors and profunctors.

1.3. **Prerequisites.** CST is deeply rooted in double category theory. This is dictated by the structure of processes: they compose like morphisms but are also the subject of morphisms.

For many notions, we will reference [Gra19]. For a slow paced, well-motivated introduction of the minimum double category theory used in CST, we invite the reader to read along the main reference [Mye22].

1.4. **Acknowledgments.** Some of the wisdom collected in these notes is not mine, but comes from fruitful conversations. Chiefly, David Jaz Myers. Among them: Ezra Schoen (helped greatly with Example 3.3), Nima Motamed, and Nathaniel Virgo.

2. Processes

The starting point for defining a theory of systems is defining a double category of composition patterns such systems use to interact with one another. In practice, defining the systems and defining the composition patterns are activities that influence one another. Systems are made out

of composition patterns themselves, and it's usually them we have in mind when we approach a formalization problem. So one often starts by asking how the systems at hand could possibly be composed together.

Composition patterns usually form an operad.[1] Operads are indeed ways to specify how 'small things fit into larger things', i.e. they are *theories of composition*. Most importantly, they give meaning to various kinds of wiring diagrams [Spi13; VSL14; LBPF21].

One can also see composition patterns as *processes* that extend a given system with further dynamics, possibly gathering many systems in one. This point of view can be more natural from a 'European' point of view, more acquainted with string diagrams rather than wiring diagrams. In fact one can see double categories of composition patterns as higher-dimensional extensions of the process theories of Abramsky, Coecke, Gogioso [AC04; CK18].

Definition 2.1. A **process theory** is a symmetric monoidal double category with attitude.

Hence any monoidal double category can be a process theory if we have a convincing interpretation of it as such. In other words, at this level of generality there's no justification for limiting this definition further. Of course any specific doctrine of systems can make more opinionated choices on which monoidal double categories to admit, and we will see plenty of examples of this.

Remark 2.1.1. Rather than defining double categories of processes to be monoidal, one should arguably define them to be multicategories. Thus a nicer, if more exotic, definition of process theory would be that of a 'multicategory internal to categories'.This is the reason here, following the custom recently adopted by Myers, we often denote processes as having many inputs. With our definitions, one might interpret the notation $I_1, \dots, I_n$ as denoting a monoidal product, i.e. we denote with ',' the monoidal product on processes.

A process theory $\mathbb{P}$ thus unpacks as follows:

(1) There are a number of objects $I, J, K, \dots$ which are **interfaces** of the processes.
(2) There are horizontal maps $h : I \to J$ which are **maps of interfaces**, without dynamic content, they simply compare different interfaces with each other.[2]

$$I \xrightarrow{\ h\ } J \tag{2.1}$$

(3) There are vertical maps $p : I_1, \dots, I_n \nrightarrow K$ which are the **processes**, connecting interfaces in some way.[3]

$$\begin{array}{c} I_1, \dots, I_n \\ \Big\downarrow p \\ K \end{array} \tag{2.2}$$

[1] By 'operad' we mean 'coloured operad' which means 'multicategory'.

[2] One could call these 'algebraic maps of interfaces', as their role is to provide morphisms that *preserve* the interface structure, in order to compare them.

[3] Again, one might call these 'geometric maps of interfaces', as they *reflect* structure in many example we care about

(4) There are squares $\alpha : p \Rightarrow q$ which are **maps of processes** supported by given maps of interfaces:

$$\begin{array}{ccc} I_1, \ldots, I_n & \xrightarrow{h_1,\ldots,h_n} & J_1, \ldots, J_n \\ \downarrow{\scriptstyle p} & \xRightarrow{\alpha} & \downarrow{\scriptstyle q} \\ K & \xrightarrow[k]{} & L \end{array} \tag{2.3}$$

A very important takeway behind this definition is that it distinguishes morphisms *as* processes from morphisms *of* processes. In fact, the usual yoga of process theories (which amount to symmetric monoidal 'single' categories) only focuses on the compositional properties of processes (they compose like morphisms). But in category theory we know that if we want to study something, we need to study morphisms between them!

Example 2.1 (Alphabets)**.** Let $\mathbb{A}\mathbf{lph} = \mathbf{FinSet}^{\uparrow}$ be the double category of alphabets and alphabet reductions, and maps thereof. Its objects are finite sets of symbols we call 'alphabets'. Its horizontal maps are maps of finite sets. Its vertical maps are *alphabet reductions*, which are map of finite sets in the opposite direction:

$$p : \Sigma \bullet\!\!\to \Sigma' \in \mathbf{FinSet}(\Sigma', \Sigma) \tag{2.4}$$

Squares are commutative squares:

$$\begin{array}{ccc} \Sigma & \xrightarrow{h} & \Xi \\ {\scriptstyle p}\uparrow & & \uparrow{\scriptstyle q} \\ \Sigma' & \xrightarrow[k]{} & \Xi' \end{array} \tag{2.5}$$

Example 2.2 (Bidirectional processes)**.** Consider the following double category of *deterministic bidirectional processes* $\mathbb{L}\mathbf{ens}$:

(1) interfaces are given by pairs or sets $\binom{A^-}{A^+}$, whose monoidal product is given by componentwise product,
(2) processes $\binom{A^-}{A^+} \leftrightarrows \binom{C^-}{C^+}$ are given by **lenses** $\binom{f^\sharp}{f}$, comprised of a *get part* $f : A^+ \to C^+$ and a *put part* $f^\sharp : A^+ \times C^- \to A^-$,
(3) maps of interfaces $\binom{A^-}{A^+} \rightrightarrows \binom{B^-}{B^+}$ are given by **charts** $\binom{g^\flat}{g}$, comprised of a *states part* $g : A^+ \to B^+$ and a *directions part* $g^\flat : A^+ \times A^- \to B^-$,
(4) behaviours are given by arrangements

$$\begin{array}{ccc} \binom{A^-}{A^+} & \underset{g}{\overset{g^\flat}{\rightrightarrows}} & \binom{B^-}{B^+} \\ {\scriptstyle f}\downarrow\uparrow{\scriptstyle f^\sharp} & & {\scriptstyle k}\downarrow\uparrow{\scriptstyle k^\sharp} \\ \binom{C^-}{C^+} & \underset{h}{\overset{h^\flat}{\rightrightarrows}} & \binom{D^-}{D^+} \end{array} \tag{2.6}$$

such that for every $a^+ \in A^+$ and $c^- \in C^-$:

$$\begin{aligned} k(g(a^+)) &= h(f(a^+)), \\ g^\flat(a^+, f^\sharp(a^+, c^-)) &= k^\sharp(g(a^+), h^\flat(f(a^+), c^-)). \end{aligned} \tag{2.7}$$

The last conditions are hard to parse formally, but basically they say that both squares one can spot in (2.6) commute. Concretely, we have two bidirectional processes $\binom{f^\sharp}{f}$ and $\binom{k^\sharp}{k}$ and a way to map between their interfaces. We are then asking that their dynamics commute with such maps.

Remark 2.1.2. Viewed as composition patterns, lenses are algebras of the operad of wiring diagrams [Spi13]. Thus they represent ways to wire a number of boxes into a larger box:

Matteo: wiring diagram

Example 2.3. The previous example can be generalized greatly by employing F-lenses [Spi19]. These are lenses in which the backward part is dependent on the forward part in a way specified by an indexed category F. Intuitively, this correspond to a wiring pattern which can change dependening on the what flows in the wires (see [Spi20]).

The definitions of F-lenses and F-charts are substantially identical to the ones above, as well as that for the squares (except the 'commutativity condition' is now harder to eyeball). We gather some examples here:

	category	F
deterministic	$\mathcal{C}$ cartesian monoidal	$\mathbf{Set}/_{\mathsf{proj}}-$ (or $\mathbf{coKl}(-\times =)$)
possibilistic	$\mathcal{E}$ topos	$\mathbf{biKl}(-\times =, \mathcal{P})$ ($\mathcal{P}$ powerset monad)
probabilistic	**Msbl**	$\mathbf{biKl}(-\times =, \Delta)$ (Δ probability monad)
effectful	$\mathcal{C}$ cartesian monoidal	$\mathbf{biKl}(-\times =, M)$ (M commutative monad)
differential (Euclidean)	**Euc**	$\mathbf{Euc}/_{\mathsf{subm}}-$
differential (general)	**Smooth**	$\mathbf{Smooth}/_{\mathsf{subm}}-$

Example 2.4 (Behavioural theory)**.** Given any Cartesian category[4] spans & maps

3. Systems

Systems are the things processes link, or the things composition patterns compose.

Definition 3.1. A **doubly indexed category**, or **action of a double category**, is given, informally, by a unitary lax double functor $\mathbf{Sys} : \mathbb{P}^\top \to \mathbb{C}\mathbf{at}$. If $\mathbb{P}$ is monoidal, then we also ask **Sys** to be lax monoidal.

We recall $\mathbb{C}\mathbf{at}$ is the cartesian monoidal double category of categories, functors, profunctors and natural transformations. A unitary lax functor is a double functor preserving vertical identities strictly but not vertical composition. Hence given two maps of interfaces $h : I \to J$, $k : J \to K$, we have a natural transformation $\ell_{h,k} : \mathbf{Sys}(h) \otimes \mathbf{Sys}(k) \to \mathbf{Sys}(h \fatsemi k)$ between profunctors.

Definition 3.2. A **theory of systems over the process theory** $\mathbb{P}$ is a monoidal doubly indexed category $\mathbf{Sys} : \mathbb{P}^\top \to \mathbb{C}\mathbf{at}$ with attitude.

[4]By which we mean a category with a terminal object and all pullbacks.

Concretely, **Sys** maps interfaces to **categories of systems**, processes to **extension functors**, maps of interfaces to **mapping profunctors** and maps of processes to **extension transformations**.

Hence given an interface $I : \mathbb{P}$, we think of the objects of $\mathbf{Sys}(I)$ as systems of a certain kind while the maps are **simulations** between them, i.e. some notion of structure-preserving map between them.

$$\mathbf{Sys}(I) = \left\{ \mathsf{S} \xrightarrow{\varphi} \mathsf{T} \right\} \tag{3.1}$$

The functors induced by a process act by extending a system with that process. If we think of the process as a composition pattern instead, the functor assembles in a composite system:

$$\mathbf{Sys}(I \overset{p}{\bullet\!\!\to} K) : \mathbf{Sys}(I) \longrightarrow \mathbf{Sys}(K) \tag{3.2}$$

The profunctors induced by a map of interfaces give notions of simulations between systems on different interfaces:

$$\mathbf{Sys}(I \xrightarrow{h} J) : \mathbf{Sys}(I) \nrightarrow \mathbf{Sys}(J) \tag{3.3}$$

Hence an element $\ell \in \mathbf{Sys}(I \xrightarrow{h} J)(\mathsf{S}, \mathsf{T})$ is a *simulation of* S *in* T *mediated by the maps of interfaces* h.

Finally, squares in $\mathbb{P}$ induce squares witnessing the extension of a simulation of systems along a map of processes:

$$\mathbf{Sys}(p \overset{\alpha}{\Rightarrow} q) : \mathbf{Sys}(h) \Rightarrow (\mathbf{Sys}(p), \mathbf{Sys}(q))^* \mathbf{Sys}(k) \tag{3.4}$$

Example 3.1 (Closed dynamical systems)**.** The most basic model of dynamical systems in mathematics is simply endomorphisms $\delta : S \to S$ on some space S in a category of 'spaces' $\mathcal{S}$. These systems are closed: they expose nothing of their state, and their dynamics can't be influenced by external input: their process theory is trivial! Consequently, the systems theory of closed dynamical systems is given by a single category $\mathbf{DynSys}_{\mathcal{S}}$:

$$\mathbf{DynSys}_{\mathcal{S}} : 1^{\top} \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}. \tag{3.5}$$

In this category, the objects are endomorphisms and the maps are commuting squares of the form:

$$\begin{array}{ccc} S & \xrightarrow{\varphi} & R \\ {\scriptstyle\delta}\downarrow & & \downarrow{\scriptstyle\gamma} \\ S & \xrightarrow[\varphi]{} & R \end{array} \tag{3.6}$$

Clearly $\mathbf{DynSys}_{\mathcal{S}}$ is monoidal if $\mathcal{S}$ is. Thus given a category $\mathcal{S}$ of spaces, one gets a systems theory of **closed dynamical systems** in $\mathcal{S}$.

Example 3.2. There's many possible variations on the definition of $\mathbf{DynSys}_{\mathcal{S}}$. Two that encompass many interesting examples are as follows.

(1) One can choose an endofunctor $F : \mathcal{S} \to \mathcal{S}$ and consider F-coalgebras instead of mere endomorphisms as the dynamical systems. In this way one can get, e.g. non-deterministic closed systems. We denote this category $\mathbf{Coalg}(F)$. The basic case is recovered for the choice $F = 1_{\mathcal{S}}$.

(2) One can choose a monoid of 'time' T and consider T-actions instead of mere endomorphisms. Notice one can pick $T : \mathbf{Mon}(\mathcal{S})$ but also $T : \mathbf{Mon}(\mathbf{Set})$, and then consider T-actions to be functors $BT \to \mathcal{S}$. In this way one can get, e.g. continuous time dynamical systems by choosing $T = \mathbb{R}$. We denote this category $\mathbf{TimeSys}(T)$. The basic case is recovered for the choice $T = \mathbb{N}$.

These two examples also admit a more interesting theory of processes, as we are going to see shortly.

Example 3.3. We can think of a coalgebra $A \to FA$ as system with states A and interface F. Now, natural transformations $\alpha : F \Rightarrow F'$ are 'lenses' and one gets an indexed category

$$\mathbf{Coalg} : \mathrm{End}(\mathcal{C}) \to \mathbf{Cat} \tag{3.7}$$

Now, if $\mathcal{C}$ is additionally finitely complete, we can go further and add another dimension. In fact, in this case, $\mathrm{End}(\mathcal{C})$ is fibred over $\mathcal{C}$ by evaluation at the terminal object:

$$- (1) : \mathrm{End}(\mathcal{C}) \to C \tag{3.8}$$

The cartesian lift of a given arrow $f : A \to G(1)$ is given by a natural transformation $f_G : f^*G \Rightarrow G$ obtained from the pullback square

$$\begin{array}{ccc} f^*GX & \xrightarrow{f_{G,X}} & GX \\ \downarrow & \lrcorner & \downarrow{\scriptstyle G!} \\ A & \xrightarrow[f]{} & G1 \end{array} \tag{3.9}$$

that simultaneously defines f^*G (on morphisms is defined by pullback again) and f_G.

The fibred subcategory of polynomial functors is what gives 'dependent' lenses, whose opposite is the codomain fibration, i.e. 'dependent' charts [Spi19]. This suggests that taking the opposite fibration of $-(1)$ gives us a fibration of 'generalized charts'.

We can explicitly construct these things if we work out the cartesian factorization system induced by $-(1)$ on $\mathrm{End}(\mathcal{C})$. This is given by

(1) Cartesian maps are given by natural transformations whose naturality is witness by pullback squares, as suggested by the definition of f^*G above: which we make explicit here:

$$\begin{array}{ccc} FX & \xrightarrow{\alpha_X} & GX \\ {\scriptstyle Ff}\downarrow & \lrcorner & \downarrow{\scriptstyle Gf} \\ FY & \xrightarrow[\alpha_Y]{} & GY \end{array} \tag{3.10}$$

(2) Vertical maps are given by natural transformations whose component at 1 is an isomorphism (think: the identity)

We define a *generalized chart* $\binom{k^\flat}{k} : F \rightrightarrows G$ to be a span in $\mathrm{End}(\mathcal{C})$ whose left leg is vertical and whose right leg is cartesian. Hence they look like this:

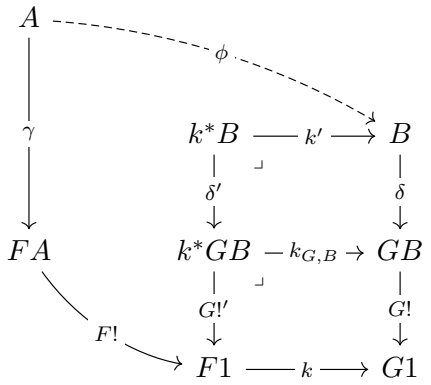
$$\begin{array}{ccc} \mathrm{End}(\mathcal{C}) & & F \xleftarrow{k^\flat} f^*G \xrightarrow{k_G} G \\ {\scriptstyle -(1)}\downarrow & & \\ \mathcal{C} & & F1 = F1 \xrightarrow[k]{} G1 \end{array} \tag{3.11}$$

These might look like lenses (because lenses are obtained by opping a fibration) but they are actually charts. We can verify this by looking at the case in which F and G are polynomial (over **Set**), to see if this construction recovers the usual one. We see that $k^\flat$ lives in

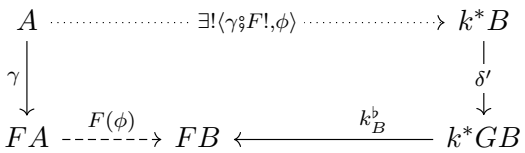
$$\mathrm{Nat}\left(\sum_{i\in F1} y^{F_i}, \sum_{j\in F1} y^{G_{k(j)}}\right) \cong \prod_{i\in F1}\sum_{j\in F1} \mathbf{Set}(F_i, G_{k(j)}) \tag{3.12}$$

and since we know this is a vertical map, meaning it lives over the identity map of $F1$, we know $i = j$ on the right hand side. Thus we see $\binom{k^\flat}{k}$ encodes the data of a chart (notice how F and G swapped places: charts are lenses 'relative to lenses').

This allows us to extend the indexed category **Coalg** defined previously to have a profunctorial action. So a given generalized chart $\binom{k^\flat}{k} : F \rightrightarrows G$ is mapped to a profunctor $\mathbf{Coalg}\binom{k^\flat}{k} : \mathbf{Coalg}(F) \nrightarrow \mathbf{Coalg}(G)$. This has a rather complex definition: it maps two coalgebras $\gamma : A \to FA$ and $\delta : B \to GB$ to the set of $\phi : A \to B$ that make the following commute:



(3.13)



(3.14)

Specifically, we are asking for the following:

$$\begin{aligned} &\forall a \in A, \quad G!(\delta(\phi(a))) = k(F!(\gamma(a))) \\ &\forall a \in A, \quad k^\flat_B(\delta'(F!(\gamma(a)), \phi(a))) = F(\phi)(\gamma(a)) \end{aligned} \tag{3.15}$$

Matteo: The double category End...

All in all, this gives us the **theory of coalgebras**:

$$\mathbf{Coalg} : \mathbb{End}^\top \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}. \tag{3.16}$$

Example 3.4 (Monoid actions)**.** The categories $\mathbf{TimeSys}(T)$ naturally gather in a doubly indexed category where the indexing double category is $\mathbf{Mon}(\mathcal{S})^{\uparrow}$. This is the double category of monoids in $\mathcal{S}$ and commutative squares thereof, except we take the opposite of the vertical direction. Hence a vertical morphism $p : M \nrightarrow N$ corresponds to a morphism of monoids $N \to M$.

This double category is a process theory for the theory of dynamical systems 'with time' described above. In fact a vertical morphism $p : M \nrightarrow N$ maps to a functor $\mathbf{TimeSys}(M) \to \mathbf{TimeSys}(N)$ given by 'restriction of scalars':

$$M \times S \xrightarrow{\delta} S \qquad \longmapsto \qquad N \times S \xrightarrow{p \times S} M \times S \xrightarrow{\delta} S, \tag{3.17}$$

and maps of monoids $h : M \to N$ also map to profunctors $\mathbf{TimeSys}(M) \nrightarrow \mathbf{TimeSys}(N)$ that sends a pair of dynamical systems $S : \mathbf{TimeSys}(M)$, $R : \mathbf{TimeSys}(N)$ to the set of squares

$$\begin{array}{ccc} M \times S & \xrightarrow{h \times \varphi} & N \times R \\ {\scriptstyle\delta}\downarrow & & \downarrow{\scriptstyle\gamma} \\ S & \xrightarrow[\varphi]{} & R \end{array} \tag{3.18}$$

Example 3.5 (Labelled transition systems)**.** We can use the double category $\mathbb{A}\mathbf{lph}$ of alphabets defined in Example 2.1 to index labelled transition systems. When $T = \Sigma^*$ (the free monoid on a finite set Σ), then $\mathbf{DynSys}_{\mathcal{S}}(\Sigma^*)$ is the category of transition systems labelled with Σ. Hence we define $\mathbf{LabTransSys} : \mathbb{A}\mathbf{lpha}^{\top} \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}$ by restricting $\mathbf{DynSys}_{\mathbf{Set}}$ along the double functor $(-)^* : \mathbf{Mon}^{\uparrow} \to \mathbb{A}\mathbf{lph}$ given by taking free monoids.

Example 3.6 (Finite state machines)**.** A finite states machine with alphabet $\Sigma : \mathbf{FinSet}$ is a finite set S of *states* equipped with a *transition function* $\delta : S \times \Sigma \to S$ and a subset $S^* \subseteq S$ of *accepting states*. In other words, it's a labelled transition system with alphabet Σ equipped with a predicate S^* over S. These form a systems theory over $\mathbb{A}\mathbf{lph}$, which mostly behaves like **LabTransSys**. The only substiantal difference is that morphisms of finite states machines are given by commutative squares like

$$\begin{array}{ccc} S \times \Sigma & \xrightarrow{\varphi} & R \times \Sigma \\ {\scriptstyle\delta}\downarrow & & \downarrow{\scriptstyle\gamma} \\ S & \xrightarrow[\varphi]{} & R \\ \cup\!| & & \cup\!| \\ S^* & \subseteq & R^* \end{array} \tag{3.19}$$

where the commutativity of the bottom square means $\varphi(S^*) \subseteq R^*$.

Thus there is a theory of systems, the **theory of finite states machines**

$$\mathbf{FSM} : \mathbb{A}\mathbf{lph}^{\top} \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}. \tag{3.20}$$

Remark 3.2.1. Often FSMs are assumed to be equipped with an initial state too. Adding this data to the previous definition is straightforward, but we think doing so is less elegant.

Remark 3.2.2. In **Set** (or really, any topos $\mathcal{E}$) FSMs over an alphabet Σ can be seen, alternatively, as coalgebras for endofunctor $2 \times (-)^{\Sigma}$. Thus one can instantiate the machinery of Example 3.3

to get categories of systems. There is a subtle problem with that though: morphisms of FSMs naturally make use of the order structure on 2, which is not really accounted for in the 'vanilla' coalgebraic framework. Hence we believe it's closer to the nature of FSMs to consider them as transition systems equipped with a subobject of states.

Example 3.7 (Moore machines)**.** Moore machines are open dynamical systems. Myers spends a long time developing their theory in [Mye22]. In fact a very large class of systems is captured by Moore machines, especially in the extended form Myers considers.

'Classical' Moore machines have a set of states S, a set of inputs I, a set of outputs O, and two maps $\mathsf{expose} : S \to O$ and $\mathsf{update} : S \times I \to S$. Hence they are given by lenses of the form $\binom{S}{S} \leftrightarrows \binom{I}{O}$. The special form of the left boundary is what makes them system-y: we guarantee statefulness of by guaranteeing the two 'S' on the left are the same. We do so by defining maps of Moore machines (over a fixed interface $\binom{I}{O}$) to be squares in $\mathbb{L}\mathbf{ens}$ (Example 2.2) of the form:

$$\begin{array}{ccc} \binom{S}{S} & \overset{\pi_2 \fatsemi \varphi}{\underset{\varphi}{\rightrightarrows}} & \binom{R}{R} \\ {\scriptstyle \mathsf{update}_S} \uparrow \downarrow {\scriptstyle \mathsf{expose}_S} & & {\scriptstyle \mathsf{update}_R} \uparrow \downarrow {\scriptstyle \mathsf{expose}_R} \\ \binom{I}{O} & = & \binom{I}{O} \end{array} \tag{3.21}$$

Notice the special form of the top chart: the map $\varphi : S \to R$ is used both on the bottom and on top. The fact maps of Moore machines cannot distinguish between top and bottom is the reason that boundary behaves as a stateful interface.

The 'generalized' Moore machines of Myers are obtained by passing from lenses to F-lenses. This testifies an important conceptual shift.

Fix a category $\mathcal{C}$ and an indexed category $F : \mathcal{C}^{\mathsf{op}} \to \mathbf{Cat}$. First of all, F-lenses are now bidirectional morphisms that have some kind of mode-dependence: an object $\binom{I}{O}$ in $\mathbb{L}\mathbf{ens}_F$ is given by an object $O : \mathcal{C}$ and an object $I : F(O)$. This can be thought as a type dependent on O or as a bundle over O. The important thing is that we introduce a more elaborate (and thus expressive) model of bidirectionality: stuff happens in the forward/bottom part (given by a morphism in $\mathcal{C}$), and then, *depending on what happened there*, something else happens in the backward/top part. The difference can be substantial, to the point of having completely different types involved.

In generalized Moore machines, we use this added dependency to distinguish between *states* and *state changes*. We thus think of the two Ss in $\binom{S}{S}$ as playing different roles: the bottom one is a proper object of states, whereas the top one is actually more accurately represented by something like $TS : F(S)$, an object 'over' S (hence depending on S, varying with it) accounting for the possible changes in state.

Hence whereas in a classical Moore machines all states are potentially reachable from any other states, we now contemplate machines in which the way we change state can be more complex. For example, we might choose a distribution of new states, or allow transitions only to states 'nearby' in some sense (thus introducing geometric structure on S).

Still, we want to be coherent in our meaning of 'change'. Hence we parametrize theories of generalized Moore machines by a section $T : \mathcal{C} \to \int F$ picking out uniformly, for each $S : \mathcal{C}$, an object of changes $\binom{TS}{S}$ over it. Importantly, T also maps morphisms $\varphi : S \to R$ to morphisms of changes $\binom{T\varphi}{\varphi} : \binom{TS}{S} \to \binom{TR}{R}$ (which we can think of as some kind of derivative). In this way we preserve the statefulness of the boundary.

Thus a theory of Moore machines is constructed as follows. Chosen $\mathcal{C} : \mathbf{Cat}$ monoidal, $F : \mathcal{C}^{\mathsf{op}} \to \mathbf{Cat}$ lax monoidal and $T : \mathcal{C} \to \int F$ lax monoidal section, we first build the process theory $\mathbb{L}\mathbf{ens}_F$ of F-lenses and F-charts. Then, we define, for each $\binom{I}{O} : \mathbb{L}\mathbf{ens}_F$, the categories

$$\mathbf{Moore}_{(F,T)}\binom{I}{O} = \left\{ \begin{array}{ccc} \binom{TS}{S} & \overset{T\varphi}{\underset{g}{\rightrightarrows}} & \binom{R}{R} \\ \mathsf{update_S} \uparrow \downarrow \mathsf{expose_S} & & \mathsf{update_R} \uparrow \downarrow \mathsf{expose_R} \\ \binom{I}{O} & = & \binom{I}{O} \end{array} \right\} \tag{3.22}$$

Given an F-lens $\binom{p^\sharp}{p} : \binom{I}{O} \rightrightarrows \binom{J}{Q}$ we get a functor:

$$\mathbf{Moore}_{(F,T)}\binom{p^\sharp}{p} : \mathbf{Moore}_{(F,T)}\binom{I}{O} \longrightarrow \mathbf{Moore}_{(F,T)}\binom{J}{Q}$$

$$\begin{array}{c} \binom{TS}{S} \\ \mathsf{expose_S} \downarrow \uparrow \mathsf{update_S} \\ \binom{I}{O} \end{array} \longmapsto \begin{array}{c} \binom{TS}{S} \\ \mathsf{expose_S} \downarrow \uparrow \mathsf{update_S} \\ \binom{I}{O} \\ p \downarrow \uparrow p^\sharp \\ \binom{J}{Q} \end{array} \tag{3.23}$$

Given an F-chart $\binom{h^\flat}{h} : \binom{I}{O} \rightrightarrows \binom{J}{Q}$, the induced profunctor is

$$\mathbf{Moore}_{(F,T)}\binom{I}{O}^{\mathsf{op}} \times \mathbf{Moore}_{(F,T)}\binom{J}{Q} \xrightarrow{\mathbf{Moore}_{(F,T)}\binom{h^\flat}{h}} \mathbf{Set}$$

$$\begin{array}{c} \binom{TS}{S} \\ \mathsf{update_S} \uparrow \downarrow \mathsf{expose_S} \\ \binom{I}{O} \end{array}, \begin{array}{c} \binom{TR}{R} \\ \mathsf{update_R} \uparrow \downarrow \mathsf{expose_R} \\ \binom{J}{Q} \end{array} \longmapsto \left\{ \begin{array}{ccc} \binom{TS}{S} & \overset{T\varphi}{\underset{\varphi}{\dashrightarrow}} & \binom{TR}{R} \\ \mathsf{update_S} \uparrow \downarrow \mathsf{expose_S} & & \mathsf{update_R} \uparrow \downarrow \mathsf{expose_R} \\ \binom{I}{O} & \overset{h^\flat}{\underset{h}{\rightrightarrows}} & \binom{J}{Q} \end{array} \right\} \tag{3.24}$$

Finally, given a square

$$\begin{array}{ccc} \begin{pmatrix} I \\ O \end{pmatrix} & \overset{h^\flat}{\underset{h}{\rightrightarrows}} & \begin{pmatrix} J \\ Q \end{pmatrix} \\ {\scriptstyle p^\sharp}\uparrow\downarrow{\scriptstyle p} & & {\scriptstyle q^\sharp}\uparrow\downarrow{\scriptstyle q} \\ \begin{pmatrix} I' \\ O' \end{pmatrix} & \overset{k^\flat}{\underset{k}{\rightrightarrows}} & \begin{pmatrix} J' \\ Q' \end{pmatrix} \end{array} \tag{3.25}$$

in $\mathbb{L}\mathbf{ens}_F$ we get a square in $\mathbb{C}\mathbf{at}$

$$\begin{array}{ccc} \mathbf{Moore}\begin{pmatrix} I \\ O \end{pmatrix} & \xrightarrow{\mathbf{Moore}\begin{pmatrix} p^\sharp \\ p \end{pmatrix}} & \mathbf{Moore}\begin{pmatrix} I' \\ O' \end{pmatrix} \\ {\scriptstyle \mathbf{Moore}\begin{pmatrix} h^\flat \\ h \end{pmatrix}}\downarrow & \xRightarrow{\mathbf{Moore}(\square)} & \downarrow{\scriptstyle \mathbf{Moore}\begin{pmatrix} k^\flat \\ k \end{pmatrix}} \\ \mathbf{Moore}\begin{pmatrix} J \\ Q \end{pmatrix} & \xrightarrow[\mathbf{Moore}\begin{pmatrix} q^\sharp \\ q \end{pmatrix}]{} & \mathbf{Moore}\begin{pmatrix} J' \\ Q' \end{pmatrix} \end{array} \tag{3.26}$$

given by stacking squares:

$$\mathbf{Moore}\begin{pmatrix} h^\flat \\ h \end{pmatrix}(\mathsf{S},\mathsf{R}) \xrightarrow{\mathbf{Moore}(\square)_{\mathsf{S},\mathsf{R}}} \mathbf{Moore}\begin{pmatrix} h^\flat \\ h \end{pmatrix}\left(\mathbf{Moore}\begin{pmatrix} p^\sharp \\ p \end{pmatrix}(\mathsf{S}), \mathbf{Moore}\begin{pmatrix} q^\sharp \\ q \end{pmatrix}(\mathsf{R})\right)$$

$$\begin{array}{ccc} \begin{pmatrix} TS \\ S \end{pmatrix} & \overset{T\varphi}{\underset{\varphi}{\dashrightarrow}} & \begin{pmatrix} TR \\ R \end{pmatrix} \\ {\scriptstyle \mathsf{update}_\mathsf{S}}\uparrow\downarrow{\scriptstyle \mathsf{expose}_\mathsf{S}} & & {\scriptstyle \mathsf{update}_\mathsf{R}}\uparrow\downarrow{\scriptstyle \mathsf{expose}_\mathsf{R}} \\ \begin{pmatrix} I \\ O \end{pmatrix} & \overset{h^\flat}{\underset{h}{\rightrightarrows}} & \begin{pmatrix} J \\ Q \end{pmatrix} \end{array} \longmapsto \begin{array}{ccc} \begin{pmatrix} TS \\ S \end{pmatrix} & \overset{T\varphi}{\underset{\varphi}{\dashrightarrow}} & \begin{pmatrix} TR \\ R \end{pmatrix} \\ {\scriptstyle \mathsf{expose}_\mathsf{S}}\uparrow\downarrow{\scriptstyle \mathsf{update}_\mathsf{S}} & & {\scriptstyle \mathsf{expose}_\mathsf{R}}\uparrow\downarrow{\scriptstyle \mathsf{update}_\mathsf{R}} \\ \begin{pmatrix} I \\ O \end{pmatrix} & \overset{h^\flat}{\underset{h}{\rightrightarrows}} & \begin{pmatrix} J \\ Q \end{pmatrix} \\ {\scriptstyle p^\sharp}\uparrow\downarrow{\scriptstyle p} & & {\scriptstyle q^\sharp}\uparrow\downarrow{\scriptstyle q} \\ \begin{pmatrix} I' \\ O' \end{pmatrix} & \overset{k^\flat}{\underset{k}{\rightrightarrows}} & \begin{pmatrix} J' \\ Q' \end{pmatrix} \end{array} \tag{3.27}$$

We thus defined the **theory of (F,T)-generalized Moore machines**:

$$\mathbf{Moore}_{(F,T)} : \mathbb{L}\mathbf{ens}_F^\top \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}. \tag{3.28}$$

Example 3.8. A notable example of generalized Moore machines is given by differential open dynamical systems. These are Moore machines in **Smooth**, the category of smooth manifolds.[5]

[5]Sometimes it's useful to consider suitable generalizations, like diffeological or smooth spaces, to get better properties out of the theory. We don't go beyond undergraduate differential geometry here.

We consider bundles given by submersions,[6] since these are pullback-stable maps:

$$\mathbf{Smooth}/_{\mathsf{subm}} - : \mathbf{Smooth}^{\mathsf{op}} \longrightarrow \mathbf{Cat}$$

$$\begin{array}{c} M \\ \downarrow{\scriptstyle p} \\ N \end{array} \longmapsto \begin{array}{c} \mathbf{Smooth}/_{\mathsf{subm}} M \\ \uparrow{\scriptstyle p^*} \\ \mathbf{Smooth}/_{\mathsf{subm}} N \end{array} \qquad \begin{array}{ccc} M & \longleftarrow & p^*E \\ {\scriptstyle p}\downarrow & \lrcorner & \downarrow \\ N & \underset{\pi}{\longleftarrow} & E \end{array} \tag{3.29}$$

We then choose the section $T : \mathbf{Smooth} \to \int \mathbf{Smooth}/_{\mathsf{subm}}$ to be the one picking out, for each smooth manifold S, its tangent bundle $\pi_S : TS \to S$. This extends to smooth maps $\varphi : S \to R$ by mapping them to their differential, which is indeed a map of tangent bundles $TS \to TR$.

Then a Moore machine $\binom{\mathsf{update}}{\mathsf{expose}} : \binom{TS}{S} \leftrightarrows \binom{I}{O}$ is a pair of maps

$$\mathsf{expose} : S \to O, \quad \mathsf{update} : (s : S) \times (i : I(\mathsf{expose}(s))) \to T_s S \tag{3.30}$$

which describe an observable on S and a non-autonomous vector field on it:

$$\mathrm{d}s = \mathsf{update}(s, i) \tag{3.31}$$

Notably, a system of this kind with a trivial interface is simply a vector field (i.e. a section of the tangent bundle).

Example 3.9 (Mealy machines)**.** Similar to Moore machines are Mealy machines: they too are stateful open dynamical systems over a bidirectional interface $\binom{I}{O}$, but their output is also dependent on their input:

$$\mathsf{expose} : S \times I \to O, \qquad \mathsf{update} : S \times I \to S \tag{3.32}$$

Therefore they are usually given as a single map

$$\delta : S \times I \longrightarrow S \times O \tag{3.33}$$

in some cartesian monoidal category $(\mathcal{C}, 1, \times)$.

This seemingly inconsequential difference makes Mealy machines quite different. In fact, contrary to Moore machines, such systems cannot be reindexed by lenses anymore $\binom{p^\sharp}{p} : \binom{I}{O} \leftrightarrows \binom{J}{Q}$ like Moore machines, the problem arising when reindexing the input:

$$\mathbf{Mealy}_{\mathcal{C}}\binom{p^\sharp}{p} : \delta \;\mapsto\; S \times J \xrightarrow{?} S \times I \xrightarrow{\delta} S \times O \xrightarrow{p} S \times Q \tag{3.34}$$

In fact in order to use $p^\sharp$ to convert a J to an I we need to have O in scope, but O can be *produced* only if I is available! So we run into a vicious cycle.

There's at least three ways to break this *empasse*. The first is to remove the dependency on O in $p^\sharp$. Hence instead of lenses we'll be looking at adapters, which are simply pairs of maps going in opposite directions.

The second is to solve the circularity by using a trace (or a trace-like) operator, thus asking for the ambient category $\mathcal{C}$ to be a traced or a feedback category (as in [KSW97a; DGR+21]).

[6] A *submersion* of smooth manifolds is a surjective smooth map whose differential is also surjective. It means we are mapping to a manifold of equal or smaller dimension, and all the fibers are manifolds themselves (hence excluding critial points or singular submanifolds).

Then Mealy machines can be reindexed by morphisms in $\mathbf{Int}(\mathcal{C})$, which are indeed quite similar to Mealy machines:

$$p : \binom{I}{O} \to \binom{J}{Q} \equiv J \times O \to I \times Q. \tag{3.35}$$

These reindex δ by using the trace to get rid of extra O:

$$\mathbf{Mealy}_{\mathcal{C}}(p) : \delta \mapsto \mathsf{trace}_O(S \times J \times O \xrightarrow{S \times p} S \times I \times Q \xrightarrow{\delta \times Q} S \times O \times Q). \tag{3.36}$$

They compose with each other similarly.

The third option is to turn lenses 'upside down'. Let a *colens* in $\mathcal{C}$ $\binom{I}{O} \leftrightarrows \binom{J}{Q}$ be a pair of maps $p : J \to I$ and $p_\sharp : J \times O \to Q$. These compose like lenses and can reindex Mealy machines:[7]

$$\mathbf{Mealy}_{\mathcal{C}}\binom{p}{p_\sharp} : \delta \mapsto S \times J \xrightarrow{\langle J, S \times p \rangle} J \times S \times I \xrightarrow{\delta \times J} J \times S \times O \xrightarrow{\langle S, p_\sharp \rangle} S \times Q \tag{3.37}$$

All these options (when well-defined) admit suitably dual, chart-like morphisms that fit into double categories of 'commutative squares', like the one in Example 2.2. This makes $\mathbf{Mealy}_{\mathcal{C}}$ a theory of systems:

$$\mathbf{Mealy}_{\mathcal{C}} : \mathbb{P}^{\top} \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at} \tag{3.38}$$

for $\mathbb{P} = \mathbb{A}\mathbf{dap}(\mathcal{C})$ (adapters & pairs of morphisms), $\mathbb{C}\mathbf{olens}_{\mathcal{C}}$ (colenses & 'cocharts'), $\mathbb{I}\mathbf{nt}(\mathcal{C})$ ($\mathbf{Int}$-morphisms & their duals). Clearly the latter is admissibile only when $\mathcal{C}$ is traced.

Example 3.10 (Behavioural theories)**.** Any behavioural theory of processes $\mathbf{Span}(\mathcal{C})$ (**??**) supports a behavioural theory of systems (recall $\mathcal{C}$ is assumed to be cartesian, hence admits all pullbacks):

$$\mathbf{BSys}_{\mathcal{C}} : \mathbf{Span}(\mathcal{C})^{\top} \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at} \tag{3.40}$$

$$\mathbf{BSys}_{\mathcal{C}}(I) = \left\{ \begin{array}{ccc} S & \xrightarrow{h} & R \\ {\scriptstyle \mathsf{observe_S}}\downarrow & & \downarrow{\scriptstyle \mathsf{observe_R}} \\ I & = & I \end{array} \right\} = \mathcal{C}/I \tag{3.41}$$

In this theory, a system S over the interface $I : \mathcal{C}$ is simply a state space $S : \mathcal{C}$ together with an observation $\mathsf{observe_S} : S \to I$.[8]

[7] Both lenses and colenses embed in $\mathbf{Int}(\mathcal{C})$ when the latter exists, and both trivialize the tracing because they present only non-trivial circularity. All of adapters, lenses, colenses and $\mathbf{Int}(\mathcal{C})$ are categories of optics fit into a diamond:

$$\begin{array}{ccccc} & & \mathbf{Int}(\mathcal{C}) & & \\ & \nearrow & & \nwarrow & \\ \mathbf{Colens}(\mathcal{C}) & & & & \mathbf{Lens}(\mathcal{C}) \\ & \nwarrow & & \nearrow & \\ & & \mathbf{Adap}(\mathcal{C}) & & \end{array} \tag{3.39}$$

Adapters, lenses and colenses are all instances of optics [CEG+20] and recently $\mathbf{Int}(\mathcal{C})$ was also shown to be a theory of optics (this was first noted in [Hed23], and then expanded upon in private communication between Hedges and Milewski). Thus we conjecture this diamond arises from relations between the actions generating the optics [Rom20].

[8] One can see maps $S \to I$ as spans $S = S \to I$, thereby fitting this example into the more general pattern of 'systems are processes with a special left boundary'.

There's two ways to understand this map. The first is to think of it as a literal observable. Hence S is to be considered as a space of states while I as a space fo quantities, and $\mathsf{observe} : S \to I$ maps a state to its corresponding observable.

The second is to think of these maps as display maps for some kind of type dependency of S over I. From this point of view we are more interested in the fibers of such map: they give us, for any observation $i : I$ at the interface, a set $s : S(i) = \mathsf{observe}^{-1}(i)$ of states compatible with it.

There's many reasons to prefer this second interpretation. The first is given by the way we are going to define the reindexing operations of **BSys**[9], which treat $\mathsf{observe}$ as a display map. The second is by the way **BSys** is used in the context of categorical systems theory, namely as a 'semantical theory': we are going to map other systems theory to behavioural theories to model the act of substantiating a systems' specification with a an actual observable behaviour (see Section 4). The examples then suggest $\mathsf{observe}$ is indeed a display map.

Thus, let's now look at how $\mathbf{Span}(\mathcal{C})$ indexes observations.

Given an observation $\mathsf{observe}_S : S \to I$, we can reindex it functorially along a span $I \overset{p_\ell}{\leftarrow} X \overset{p_r}{\to} I'$ by pulling back along p_ℓ first and then composing with p_r (this is called the pull-push action of spans):

$$\mathbf{BSys}(I \overset{p_\ell}{\leftarrow} X \overset{p_r}{\to} I') : \mathbf{BSys}(I) \longrightarrow \mathbf{BSys}(I')$$

$$\begin{array}{ccc} S & & p_\ell^* S \\ \downarrow{\scriptstyle \mathsf{observe}_S} & \longmapsto & \downarrow{\scriptstyle p_\ell^* \mathsf{observe}_S} \\ I & & X \\ & & \downarrow{\scriptstyle p_r} \\ & & I' \end{array} \tag{3.42}$$

We can visualize this better by arranging an appropriate diagram:

$$\begin{array}{ccccc} S & \longleftarrow & p_\ell^* S & = & p_\ell^* S \\ \downarrow{\scriptstyle \mathsf{observe}_S} & & \downarrow{\scriptstyle p_\ell^* \mathsf{observe}_S} & & \downarrow{\scriptstyle \mathbf{BSys}(p)(\mathsf{observe}_S)} \\ I & \underset{p_\ell}{\longleftarrow} & X & \underset{p_r}{\longrightarrow} & I' \end{array} \tag{3.43}$$

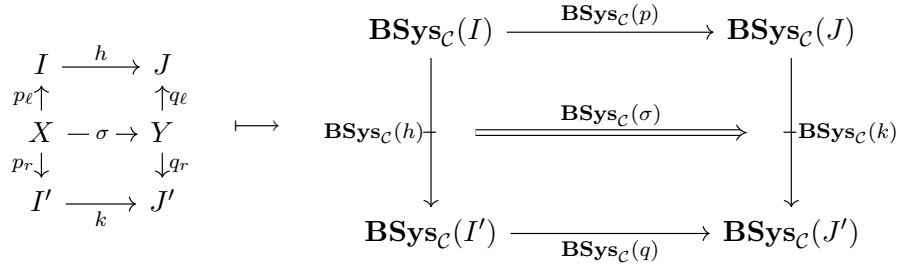
Given a map $I \overset{h}{\to} J$, we define a profunctor that maps a pair of observation maps to the set of maps between their domains that make the evident square commute:

$$\mathbf{BSys}_{\mathcal{C}}(I)^{\mathsf{op}} \quad \times \quad \mathbf{BSys}_{\mathcal{C}}(J) \xrightarrow{\mathbf{BSys}_{\mathcal{C}}(h)} \mathbf{Set}$$

$$\begin{array}{ccc} S \\ \downarrow{\scriptstyle \mathsf{observe}_S} \\ I \end{array} \qquad \begin{array}{ccc} R \\ \downarrow{\scriptstyle \mathsf{observe}_R} \\ J \end{array} \qquad \longmapsto \qquad \left\{ \begin{array}{ccc} S & \dashrightarrow & R \\ \mathsf{observe}_S \downarrow & & \downarrow \mathsf{observe}_R \\ I & \overset{h}{\to} & J \end{array} \right\} \tag{3.44}$$

[9]This might seem circular but the definition of **BSys** is actually mathematically motivated. So we use the mathematics as a suggestion of which intepretation fits it better, as one should do.

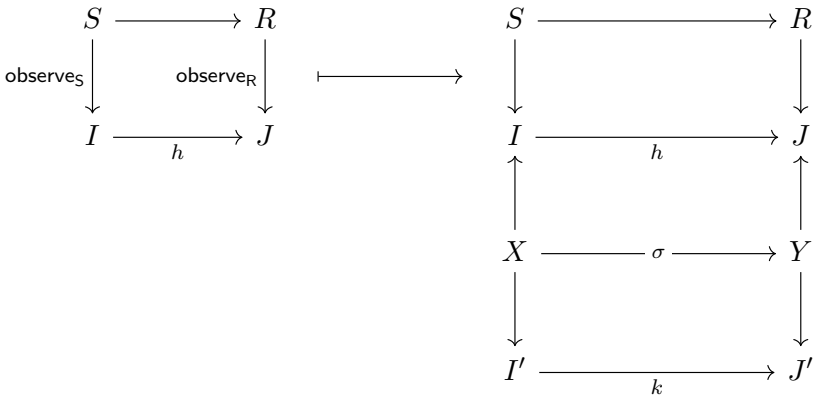
Finally, we have a map on squares:



(3.45)

given, again, by stacking:

$$\mathbf{BSys}_\mathcal{C}(h)(\mathsf{S},\mathsf{R}) \xrightarrow{\mathbf{BSys}_\mathcal{C}(\sigma)_{\mathsf{S},\mathsf{R}}} \mathbf{BSys}_\mathcal{C}(k)(\mathbf{BSys}_\mathcal{C}(f)(\mathsf{S}),\mathbf{BSys}_\mathcal{C}(g)(\mathsf{R}))$$



(3.46)

Thus we have a systems theory:

$$\mathbf{BSys}_\mathcal{C} : \mathbf{Span}(\mathcal{C})^\top \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at} \tag{3.47}$$

In particular, when $\mathcal{C} = \mathbf{Set}$, then $\mathbb{S}\mathbf{et} := \mathbf{Span}(\mathbf{Set})$ and we define:

$$\mathbf{BSys} := \mathbf{BSys}(\mathbf{Set}) : \mathbb{S}\mathbf{et}^\top \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}. \tag{3.48}$$

Notably, in this case, $I \mapsto \mathbf{Set}/I \cong \mathbf{Set}^I$ (where the latter sends a 'display map' to the indexed set of its fibers).

Example 3.11 (Structured cospans)**.**

Matteo: TODO

3.1. **Doctrines.** A doctrine of systems is a uniform way to specify theories of systems given some data. Most of the theories we described above are actually doctrines, since we defined them parametric on some data:

(1) a behavioural theory is defined for each Cartesian category $\mathcal{C}$,
(2) a theory of Moore machines is defined for every fibration $\pi : \mathcal{E} \to \mathcal{C}$ with a section T,
(3) a theory of coalgebras is defined for every category $\mathcal{C}$ with pullbacks,

and so on.

Definition 3.3. A **doctrine of systems** is a 2-functor into $\mathbb{S}\mathbf{ys}\mathbb{T}\mathbf{h}$.

The 2-category $\mathbb{S}\mathbf{ys}\mathbb{T}\mathbf{h}$ has system theories as objects and the following as 1-cells:

Definition 3.4. A **map of system theories** is a pair $(F, F^\flat) : \mathbf{Sys}_1 \to \mathbf{Sys}_2$ where F is a lax double functor while $F^\flat$ is a vertical lax-natural transformation.

$$\begin{array}{ccc} \mathbb{P}_1^\top & \xrightarrow{\mathbf{Sys}_1} & \\ {\scriptstyle F^\top}\downarrow & \Downarrow {\scriptstyle F^\flat} & \mathbb{C}\mathbf{at} \\ \mathbb{P}_2^\top & \xrightarrow{\mathbf{Sys}_2} & \end{array} \tag{3.49}$$

The 2-cells in $\mathbb{S}\mathbf{ys}\mathbb{T}\mathbf{h}$ are pairs of an horizontal natural transformation and a modification [Gra19].

4. Behaviours

Behaviours in CST are simply maps into an behavioural theory. By default, we consider behaviours valued in $\mathbf{BSys}(\mathbf{Set})$, but one can consider 'structured behaviours' into $\mathbf{BSys}(\mathcal{C})$ when one wants to keep track of extra structure on the set of behaviours of a system.

Definition 4.1. A **theory of behaviour** for a systems theory **Sys** valued in the Cartesian category $\mathcal{C}$ is a map of system theories $B : \mathbf{Sys} \to \mathbf{BSys}(\mathcal{C})$:

$$\begin{array}{ccc} \mathbb{P}^\top & \xrightarrow{\mathbf{Sys}} & \\ {\scriptstyle B^\top}\downarrow & \Downarrow {\scriptstyle B^\flat} & \mathbb{C}\mathbf{at} \\ \mathbf{Span}(\mathcal{C})^\top & \xrightarrow{\mathcal{C}/-} & \end{array} \tag{4.1}$$

Example 4.1 (States)**.** One of the simplest notion of behaviour for a system is given by the idea of 'state space'. This is the very classical stance that what can be observed about a system is its permanence in a state which varies in a prescribed space. So suppose $\mathcal{C}$ is some category of spaces[10] and let's consider the theory of coalgebras on $\mathcal{C}$: $\mathbf{Coalg}_\mathcal{C} : \mathbb{E}\mathbf{nd}^\top \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}$.

Then there is a theory of behaviour:

$$\begin{array}{ccc} \mathbb{E}\mathbf{nd}^\top & \xrightarrow{\mathbf{Coalg}_\mathcal{C}} & \\ {\scriptstyle 1^\top}\downarrow & \Downarrow \mathsf{states} & \mathbb{C}\mathbf{at} \\ \mathbf{Span}(\mathcal{C})^\top & \xrightarrow{\mathcal{C}/-} & \end{array} \tag{4.2}$$

which is trivial on the interfaces and maps coalgebras to their carriers:

$$\begin{aligned} \mathsf{states}_F : \mathbf{Coalg}_\mathcal{C}(F) &\longrightarrow \mathcal{C} \\ (S, S \xrightarrow{\delta} FS) &\longmapsto S. \end{aligned} \tag{4.3}$$

[10] Following Lawvere, we consider 'cartesian extensive' to be a pretty good off-the-shelf notion of *category of spaces*.

Clearly such a theory of behaviour can also be defined for Moore machines. Let $\mathbf{Moore}_{(F,T)} : \mathbb{L}\mathbf{ens}^\top \xrightarrow{\text{unitary lax}} \mathbb{C}\mathbf{at}$ be a theory of Moore machines, then there is again a theory of behaviour

$$\begin{array}{ccc} \mathbb{L}\mathbf{ens}^\top & \xrightarrow{\mathbf{Moore}_{(F,T)}} & \\ {\scriptstyle 1^\top}\downarrow & \Downarrow \mathsf{states} & \mathbb{C}\mathbf{at} \\ \mathbf{Span}(\mathcal{C})^\top & \xrightarrow{\mathcal{C}/-} & \end{array} \tag{4.4}$$

which is trivial on the interfaces and maps Moore machines to their state spaces:

$$\begin{aligned} &\mathsf{states}_{\binom{I}{O}} : \mathbf{Moore}_{(F,T)}\binom{I}{O} \longrightarrow \mathcal{C} \\ &(S, \binom{\mathsf{update}}{\mathsf{expose}} : \binom{TS}{S} \leftrightarrows \binom{I}{O}) \longmapsto S. \end{aligned} \tag{4.5}$$

Example 4.2 (Fixpoints)**.**

Matteo: TODO

Example 4.3 (Trajectories)**.**

Matteo: TODO

Example 4.4 (Languages)**.** One of the first notions of 'behaviour of a system' one encounters in computer science is that of *language of a finite states machines*. These, indeed, give a theory of behaviours for the theory of finite states machine we described in Example 3.6.

Given an FSM $\mathsf{S} = (S, \Sigma, \delta, S^*)$, its language is the subset of Σ^* given by those words which, when fed to S, leave it in an accepting state (one in S^*). More precisely, we first have to pick an initial state $s_0 \in S$, then we can start using δ, inputting the given word one character at a time, until we are done. Thus the language of an S is really a family of subsets of Σ^*, given by the recursively-defined map

$$\begin{aligned} L_{\mathsf{S}} : S &\longrightarrow P\Sigma^* \\ s_0 &\longmapsto \{\sigma_1 \cdots \sigma_n \,|\, \sigma_2 \cdots \sigma_n \in L_{\mathsf{S}}(\delta(s_0, \sigma_1))\}. \end{aligned} \tag{4.6}$$

Notice this family of subsets can also be defined as a 'bundle of sets' (a map we use for its fibers)

$$\Lambda(\mathsf{S}) := \{(s_0, w) \mid\in w \in L_{\mathsf{S}}(s_0)\} \rightarrowtail S \times \Sigma^* \xrightarrow{\pi_{\Sigma^*}} \Sigma^* \in \mathbf{Set}/\Sigma^*. \tag{4.7}$$

Thus we see there is a theory of behaviour

$$\begin{array}{ccc} \mathbb{A}\mathbf{lph}^\top & \xrightarrow{\mathbf{FSM}} & \\ {\scriptstyle (-)^{*\top}}\downarrow & \Downarrow \Lambda & \mathbb{C}\mathbf{at} \\ \mathbf{Span}(\mathbf{Set})^\top & \xrightarrow{\mathbf{Set}/-} & \end{array} \tag{4.8}$$

where

(1) $(-)^* : \mathbb{A}\mathbf{lph} \to \mathbf{Span}(\mathbf{Set})$ sends a finite alphabet Σ to the free monoid Σ^*, an alphabet mapping $h : \Sigma \to \Sigma'$ to the morphism of monoids $h^* : \Sigma^* \to \Sigma'^*$, an alphabet reduction $p : \Sigma \leftarrow \Xi$ to the span $\Sigma^* \xleftarrow{p^*} \Xi^* = \Xi^*$, and a commutative square to an obvious morphism of spans.

(2) The component at $\Sigma : \mathbb{A}\mathbf{lph}$ of Λ is given by

$$\begin{aligned} L_\Sigma : \mathbf{FSM}(\Sigma) &\longrightarrow \mathbf{Set}/\Sigma^* \\ (S, \Sigma, \delta, S^*) &\longmapsto \Lambda(\mathsf{S}) : \{(s_0, w) \mid\in w \in L_{\mathsf{S}}(s_0)\} \to \Sigma^* \end{aligned} \tag{4.9}$$

The functoriality properties of these assignments can be proven by making the following observation. For each $n \in \mathbb{N}$ (including 0), there are finite states machines $\mathsf{Fin\,n} = (\mathrm{Fin}\, n + 1, \mathrm{Fin}\, n, \_ + 1, \{n\})$. Explicitly, these have $n + 1$ states and are defined over an alphabet with n symbols. The transition map is

$$\begin{aligned} \_ + 1 : \mathrm{Fin}\, n + 1 \ \times\ \mathrm{Fin}\, n &\longrightarrow \mathrm{Fin}\, n + 1 \\ (k, i) &\longmapsto k + 1 \end{aligned} \tag{4.10}$$

and the only accepting state is $n \in \mathrm{Fin}\, n + 1$.

A map from this machine to another like S, mediated by a map of alphabets $\sigma : \mathrm{Fin}\, n \to \Sigma$, looks like this:

$$\begin{array}{ccc} \mathrm{Fin}\, n + 1 \times \mathrm{Fin}\, n & \xrightarrow{s \times \sigma} & S \times \Sigma \\ {\scriptstyle \_+1}\downarrow & & \downarrow{\scriptstyle \delta} \\ \mathrm{Fin}\, n + 1 & \xrightarrow{\quad s \quad} & S \\ \cup\mid & & \cup\mid \\ \{n\} & \subseteq & S^* \end{array} \tag{4.11}$$

Concretely, this is a map $s : \mathrm{Fin}\, n + 1 \to S$ selecting a sequence of $n + 1$ states $s_0, \ldots, s_n$ such that

$$\forall 0 \leq k \leq n - 1,\ s_{k+1} = \delta(s_k, \sigma_k) \quad \text{and} \quad s_n \in S^*. \tag{4.12}$$

In other words, this data amounts to a word $\sigma_1 \cdots \sigma_n \in \Sigma^*$ and a state $s_0 \in S$ such that S accepts $\sigma_1 \cdots \sigma_n$ in n steps when starting from s_0.

Notice this also works for the empty word: a map from $\mathsf{Fin\,0}$ is given by just a choice of state $s_0 \in S^*$, meaning S accepts the empty word when starting from s_0.

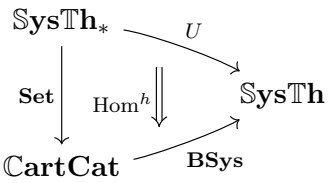
Therefore we see $((-)^*, \Lambda)$ is a sum of corepresentable behaviours, namely $\sum_{n \geq 0} \mathbf{FSM}(\mathsf{Fin\,n}, -)$. Observe the symmetry restored: $(-)^*$ is also the functor $\sum_{n \geq 0} \mathbf{Set}(\mathrm{Fin}\, n, -)$!

Definition 4.2. A **doctrine of behaviour** for the doctrine $\mathbb{D}$**octrine** is map of doctrines of systems into the behavioural doctrine **BSys**:

$$\begin{array}{ccc} \mathbb{D}\mathbf{ata} & \xrightarrow{\quad\mathbb{D}\mathbf{octrine}\quad} & \\ {\scriptstyle \mathbb{B}^\top}\downarrow & \overset{\mathbb{B}^\flat}{\Downarrow} & \mathbb{S}\mathbf{ys}\mathbb{T}\mathbf{h} \\ \mathbb{C}\mathbf{artCat} & \xrightarrow{\quad\mathbf{BSys}\quad} & \end{array} \tag{4.13}$$

Remark 4.2.1. The laxity of $B^\flat$ relates the behaviours of the parts of a system to the behaviour of the whole system. The non-invertibility of such a map witnesses emergent behaviours. In [Mye22, Theorem 5.3.3.1], Myers proves that a large class of behaviours for the doctrine of Moore machines does not, in fact, exhibit emergence, by showing such the laxity of $B^\flat$ is invertible.

Example 4.5. A large class of behaviours are corepresentables, i.e. defined by simulations of an archetypal system exhibiting that behaviour. Thus there is a **doctrine of corepresentable behaviour** on the doctrine of pointed theories:



(4.14)

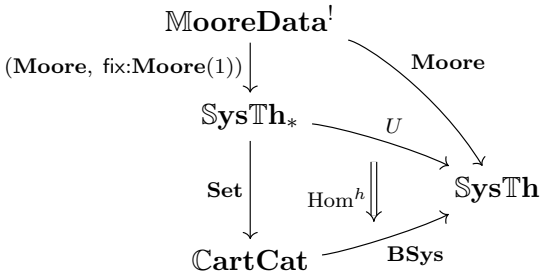
An object of $\mathbb{S}\mathbf{ys}\mathbb{T}\mathbf{h}_*$ is a systems theory equipped with a distinguished system (hence a pair of an interface and a system over it) which is used as archetype for a certain kind of behaviour.

The transformation Hom^h at a pointed theory $(\mathbf{Sys} : \mathbb{P}^\top \to \mathbb{C}\mathbf{at}, \mathsf{B} : \mathbf{Sys}(I))$ is the map of system theories $\mathbb{P}^h(\mathsf{B}, -) : \mathbf{Sys} \to \mathbf{BSys}(\mathbf{Set})$ defined as follows. Its two components are the horizontal hom-functor $\mathbb{P}^h(I, -) : \mathbb{P} \to \mathbb{S}\mathbf{et}$ and the similar fiberwise hom-functor

$$\mathbb{P}^h(\mathsf{B}, -)^\flat : \mathbf{Sys} \Rightarrow \mathbf{Set}/\mathbb{P}^h(I, -). \tag{4.15}$$

This latter functor sends a system $\mathsf{S} : \mathbf{Sys}(J)$ to the $\mathbb{P}^h(I, J)$-indexed family of sets sending a map of interfaces $k : I \to J$ to the set $\mathbf{Sys}(k)(\mathsf{B}, \mathsf{S})$ of maps of systems mediated by k.

Example 4.6. Let $\mathbb{M}\mathbf{ooreData}^!$ be the 2-category of fibrations of categories with a terminal object and a section thereof. These amount to a fibration $p : \mathcal{E} \to \mathcal{C}$ such that $p(1_\mathcal{E}) = 1_\mathcal{C}$, and a section $T : \mathcal{C} \to \mathcal{E}$ such that $T(1_\mathcal{C}) = 1_\mathcal{E}$. In this situation, one can build the Moore machine $\mathsf{fix} : \binom{T1}{1} \Longrightarrow \binom{1}{1}$ which 'does nothing'. Hence we get a *doctrine of fixpoints* by using such a machine as the archetype for the behaviour of a still system:



(4.16)

References

[AC04] S. Abramsky and B. Coecke. "A categorical semantics of quantum protocols". In: *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004*. IEEE. 2004, pp. 415–425.

[BC20] J. C. Baez and K. Courser. "Structured cospans". In: *Theory and Applications of Categories* 35.48 (2020), pp. 1771–1822.

[BCV22] J. C. Baez, K. Courser, and C. Vasilakopoulou. "Structured versus Decorated Cospans". In: *Compositionality* 4 (3 Sept. 2022). ISSN: 2631-4444. DOI: 10.32408/compositionality-4-3. URL: https://doi.org/10.32408/compositionality-4-3.

[BM20] J. C. Baez and J. Master. "Open petri nets". In: *Mathematical Structures in Computer Science* 30.3 (2020), pp. 314–341.

[BSW96] S. L. Bloom, N. Sabadini, and R. F. Walters. "Matrices, machines and behaviors". In: *Applied Categorical Structures* 4.4 (1996). Publisher: Springer, pp. 343–360.

[Cap22] M. Capucci. *From categorical systems theory to categorical cybernetics*. 2022. URL: https://www.youtube.com/watch?v=wtgfyjFIHBQ.

[CEG+20] B. Clarke et al. "Profunctor optics, a categorical update". In: *arXiv preprint arXiv:2001.07488* (2020).

[CGKS20] J. Culbertson et al. "Formal composition of hybrid systems". In: *Theory and Applications of Categories* 35.45 (2020), pp. 1634–1682.

[CK18] B. Coecke and A. Kissinger. "Picturing quantum processes". In: *International Conference on Theory and Application of Diagrams*. Springer. 2018, pp. 28–31.

[DGR+21] E. Di Lavore et al. "A canonical algebra of open transition systems". In: *International Conference on Formal Aspects of Component Software*. Springer. 2021, pp. 63–81.

[Fon15] B. Fong. "Decorated Cospans". In: *Theory & Applications of Categories* 30 (2015).

[FS07] J. L. Fiadeiro and V. Schmitt. "Structured co-spans: an algebra of interaction protocols". In: *International Conference on Algebra and Coalgebra in Computer Science*. Springer. 2007, pp. 194–208.

[Gra19] M. Grandis. *Higher Dimensional Categories: From Double to Multiple Categories*. World Scientific, 2019.

[Hed23] J. Hedges. *Geometry of interaction is the optic for copointed functors*. en-GB. Jan. 2023. URL: https://julesh.com/2023/01/28/geometry-of-interaction-is-the-optic-for-copointed-functors/ (visited on 03/03/2023).

[KSW02] P. Katis, N. Sabadini, and R. F. Walters. "Feedback, trace and fixed-point semantics". In: *RAIRO-Theoretical Informatics and Applications* 36.2 (2002), pp. 181–194.

[KSW97a] P. Katis, N. Sabadini, and R. F. Walters. "Bicategories of processes". In: *Journal of Pure and Applied Algebra* 115.2 (1997), pp. 141–178.

[KSW97b] P. Katis, N. Sabadini, and R. F. Walters. "Span(Graph): A categorical algebra of transition systems". In: *International Conference on Algebraic Methodology and Software Technology*. Springer. 1997, pp. 307–321.

[KSW99] P. Katis, N. Sabadini, and R. F. Walters. *On the algebra of feedback and systems with boundary*. Citeseer, 1999.

[LBPF21] S. Libkind et al. "Operadic modeling of dynamical systems: mathematics and computation". In: (2021). Available at https://arxiv.org/abs/2105.12282.

[Ler18] E. Lerman. "Networks of open systems". In: *Journal of Geometry and Physics* 130 (2018), pp. 81–112.

[Mye20a] D. J. Myers. *A general definition of open dynamical system*. 2020. URL: https://www.youtube.com/watch?v=8T-Km3taNko.

[Mye20b] D. J. Myers. *Double Categories of Open Dynamical Systems*. 2020. URL: https://www.youtube.com/watch?v=f9fjf9lo2_M.

[Mye20c] D. J. Myers. "Double categories of open dynamical systems". In: *arXiv preprint arXiv:2005.05956* (2020).

[Mye20d] D. J. Myers. *Open dynamical systems, trajectories and hierarchical planning*. 2020. URL: https://www.youtube.com/watch?v=3FxeY5DbPn0.

[Mye21] D. J. Myers. *Paradigms of composition*. 2021. URL: https://www.youtube.com/watch?v=50s62D5Ah-M.

[Mye22] D. J. Myers. *Categorical Systems Theory*. 2022.

[Rom20] M. Román. "Profunctor optics and traversals". In: *arXiv preprint arXiv:2001.08045* (2020).

[Ros78] R. Rosen. *Fundamentals of measurement and representation of natural systems*. Vol. 1. Elsevier Science Limited, 1978.

[Spi13] D. I. Spivak. "The operad of wiring diagrams: formalizing a graphical language for databases, recursion, and plug-and-play circuits". In: (2013). Available at https://arxiv.org/abs/1305.0297.

[Spi19] D. I. Spivak. "Generalized Lens Categories via functors $F : \mathcal{C}^{\mathrm{op}} \to \mathsf{Cat}$". 2019.

[Spi20] D. I. Spivak. *Poly: An abundant categorical setting for mode-dependent dynamics*. arXiv:2005.01894 [math]. June 2020. DOI: 10.48550/arXiv.2005.01894. URL: http://arxiv.org/abs/2005.01894 (visited on 03/03/2023).

[SW93] N. Sabadini and R. F. Walters. "On functions and processors: an automata-theoretic approach to concurrency through distributive categories". In: *School of Mathematics and Statistics Research Reports* 7 (93 Nov. 1993).

[VSL14] D. Vagner, D. I. Spivak, and E. Lerman. "Algebras of open dynamical systems on the operad of wiring diagrams". In: (2014). Available at https://arxiv.org/abs/1408.1598.