DRAFT

July 29, 2021

Categorical Systems Theory

David Jaz Myers

(Last updated: July 29, 2021)

This page intentionally left blank.

To André Joyal and Bill Lawvere

Preface

This book is a work in progress — including the acknowledgements below! Use at your own peril!

Acknowledgments

David Spivak has been a friend and mentor to me as I write this book and beyond. In many ways, I see this book as my take on David's research in lens based systems in recent years. David and I began writing a book together, of which this book was to be the first half and David's book on polynomial functors (now co-authored with Nelson Niu) was to be the second. But as we were writing, we realized that these weren't two halves of the same book, but rather two books in the same genre. It was a great pleasure writing with David during the summer of 2020, and I owe him endless thanks for ideas, suggestions, and great conversation. This book wouldn't exist without him.

Thanks go to Emily Riehl, tslil clingman, Sophie Libkind, John Baez, Geoff Cruttwell, Brendan Fong, Christian Williams.

Contents

Wiring together dynamical systems

1.1 Introduction

Here's a basic fact of life: *things change*. And how things change most often depends on how they currently are. This is the fundamental idea underlying all the various notions of *dynamical system* that we will see in this book.

> **Informal Definition 1.1.1.** A *dynamical system* consists of:
> - a notion of how things can be, called the *states*, and
> - a notion of how things will change given how they are, called the *dynamics*.
>
> The dynamics of a system might also depend on some free *parameters* or *inputs* that are imported from the environment, and we will often be interested in some particular *variables* of the state that are *exposed* or *output* to the environment.

    You and I are big, complicated dynamical systems. Our bodies and minds are in some particular configuration, and over time this configuration changes. We can sense things — seeing, touching, tasting — and what we sense affects how our bodies and mind changes. Seeing a scary snake can make me recoil and feel fear, but seeing a cute snake plushie can make me go over and start to pet it. Some parts of me are also put back into the environment, like the expression on my face. But not all of me is exposed in that way — some things just go on in my head.

    This is the basic model of a dynamical system we will be working with in this book. But to make the above informal definition precise, we need to answer a number of questions:

- What should a state be, really? Do we just have an abstract set of states, or could there be a continuum of states? Maybe there are some other structures that states can enter into which have to be respected by the dynamics, but aren't determined by them? Jaz: With this last sentence, I'm thinking of "states as polynomial comonad aka category". Not sure how to phrase it right.

- What does it mean to change? Do we want to know precisely which state will be next if we know how things are? Or, maybe we will only have a guess at which state will come next? Or, maybe we'll just say how a state is tending to change, but not where it will end up?
- Do we always take in the same sort of parameters, or does it depend on how our system is placed in its environment? Should the dynamics vary continuously (or linearly, or some other way) in the choice of parameters?

Different people have decided on different answers to these questions for different purposes. Here are some of the most widespread ways to answer those questions:

1. We'll assume the states form a discrete set, and that if we know the current state and our parameters, we know exactly what the next state will be. Such a system generally called a *Moore machine*.

2. We'll assume the states form a continuum, but that we only know how a state is tending to change, not what the "next" state will be. Such a system is generally called a *system of differential equations* — the differential equations tells us the derivatives of the state variables: the way they are tending.

3. We'll assume the states form a discrete set, but that we only have a guess at which state will follow from the current state. Such a system is generally called a *Markov process*, or a *Markov decision process*.

We will call a way of answering these questions the *doctrine* of dynamical systems we are working in.

Informal Definition 1.1.2. A *doctrine* of dynamical systems is a particular way to answer the following questions about what it means to be a dynamical system:

- What does it mean to be a state?
- How should the output vary with the state — discretely, continuously, linearly?
- Can the kinds of input a system takes in depend on what it's putting out, and how do they depend on it?
- What sorts of changes are possible in a given state?
- What does it mean for states to change.
- How should the way the state changes vary with the input?

Moore machines, differential equations, and Markov decision processes are each dynamical systems understood in a different doctrine.

1. A Moore machine is a dynamical system in a *discrete and deterministic* doctrine.
2. A system of differential equations is a dynamical system in a *differential* doctrine.
3. A Markov decision process is a dynamical system in a *stochastic* doctrine.

In most cases, mathematicians have assumed that that the kinds of parameters our systems take in never change — that our system will always interface with its environment in the same way. However, this assumption is quite restrictive; after all, I change the way I interface with my environment all the time. Every time I turn and face a new direction, I open myself up to new inputs. There are variations on all of the

above doctrines which allow for the kinds of input to depend on what the system is putting out, but in this book we will take a deep dive into the discrete and deterministic variant.

4. A *dependent system* is a dynamical system in a (discrete, deterministic, and) *dependent* doctrine.

The dynamical systems we will see in this book are *open* in the sense that they take in inputs from their environment and expose outputs back to their environment. Because of this, our systems can interact with eachother. One system can take what the other system outputs as part of its input, and the other can take what the first outputs as part of its input. For example, when we have a conversation, I take what I hear from you and use it to change how I feel, and from those feelings I generate some speech which I output to the world. You then take what I've said and do the same thing.

Jaz: Some wiring diagram of a conversation

We call this way of putting together dynamical systems to make more complex systems *composition*.

Informal Definition 1.1.3. *Composition* is the process by which some things are brought together to form bigger things.

Functions can be composed by $g \circ f(x) = g(f(x))$, and dynamical systems can be composed by plugging in the variables of the states of some into the parameters of others.

This book is all about composing dynamical systems. Jaz: We should also tell the story that categories for doing the mathematics of dependent systems — ie you need category theory to talk about dependent systems effectively. Because of this, we will use the abstract language of composition: *category theory*.

Informal Definition 1.1.4. *Category theory* is the abstract study of composition.

1.1.5 Category Theory

We'll be using the language of category theory quite freely in this book, and so we'll expect you to know the basics. These are the notions we will expect you to be familiar with:

- What a category is.
- What an isomorphism is.
- What a functor is.
- What a natural transformation is.
- What a terminal and an initial object are.
- What a product and a coproduct are.
- What a monad is, and it will help if you also know what a comonad is.

- What a monoidal category is.

Good introductions to category theory abound. One place to start is *An invitation to applied category theory* [FS19]. Another is *Notes on category theory* [Per21].

We will be using cartesian categories quite a bit in the first few chapters.

Definition 1.1.6. A category $\mathcal{C}$ is *cartesian* if every two objects A and B in $\mathcal{C}$ have a product $A \times B$, and $\mathcal{C}$ has a terminal object 1. Equivalently, $\mathcal{C}$ is cartesian if for any finite set I and I-indexed family $A_{(-)} : I \to \mathcal{C}$ of objects, there is a product $\prod_{i \in I} A_i$ in $\mathcal{C}$.

A functor $F : \mathcal{C} \to \mathcal{D}$ between cartesian categories is said to be *cartesian* if it preserves products and terminal objects, i.e. the map $(F\pi_A, F\pi_B) : F(A \times B) \to FA \times FB$ is an isomorphism for all A and B, and the terminal morphism $F1 \to 1$ is an isomorphism.

We will also use some more advanced category theory, like indexed categories, double categories, and toposes. However, you don't need to know them up front; we will introduce these concepts as we use them.

While we're at it, here's some notation we'll use repeatedly throughout the book. The nth ordinal is denoted n. It is defined to be the set

$$\mathsf{n} \coloneqq \{1, 2, \ldots, n\}.$$

So 0 is the empty set, 1 is a one-element set, etc.

1.2   Deterministic and differential doctrines

In this chapter, we will see how to wire together dynamical systems of all different sorts. First, however, we start with two exemplary doctrines:

1. First, systems which we will call *(discrete-time) deterministic systems*, which specify exactly which state the system will transition into given its current state and input parameters.
2. Second, systems which we will call *differential systems*, which do not specify a "next state" but rather specify exactly how the state is tending to change in the moment, given the current state and input parameters.

1.2.1   Deterministic systems

A paradigmatic example of this sort of dynamical system is a clock.
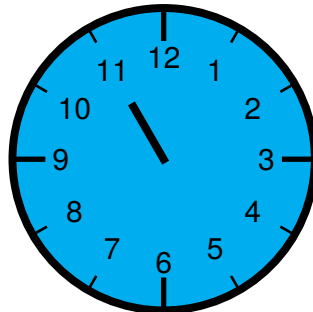
Suppose that our clock has just an hour hand for now. Then we may collect all the way things can be for the clock into a set of hours:

$$\text{Hour} := \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}.$$

This set Hour is the set of *states* of our clock system.

If we know what hour it is, we also know what hour is coming next. So, this system has the following dynamics:

$$\text{tick} : \text{Hour} \to \text{Hour} \tag{1.1}$$

$$t \mapsto \begin{cases} t + 1 & \text{if } t < 12 \\ 1 & \text{if } t = 12 \end{cases}$$

Here's a sample of the dynamics of the clock. Say we started at the 10 o'clock state:

$$10 \xmapsto{\text{tick}} 11 \xmapsto{\text{tick}} 12 \xmapsto{\text{tick}} 1 \xmapsto{\text{tick}} 2 \xmapsto{\text{tick}} \cdots$$

Ok, it's not the most dynamic of systems, but we have to start somewhere. If we want to refer to the whole system at once, we can box it up and draw it like this:

$$\boxed{\text{Clock}} - \text{Hour} \tag{1.2}$$

We imagine that the clock is going about its business inside the box, and that is shows the hour it is currently displaying on the outgoing wire. This outgoing wire constitutes the clock's exposed variable, but we'll explain that more later.

One issue with our clock is that it doesn't tell us whether it is morning or evening. Being morning or evening and going back and forth between them is another way that things might be and change, and hence we can see it as its own two-state dynamical system with states

$$\text{a.m./p.m.} = \{\text{a.m.}, \text{p.m.}\}.$$

However, rather than have this be an independent system, we want to consider it as a little addition to our clock system, one that reads a.m. or p.m.:

$$\tag{1.3}$$
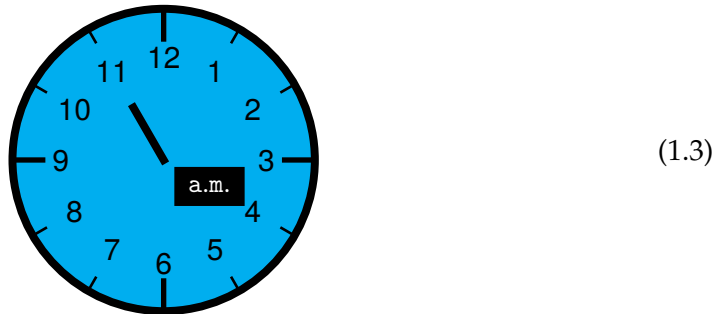
To connect the meridian to the clock means that the way the meridian changes should be based on the hour:

$$\text{next} : \text{a.m./p.m.} \times \text{Hour} \to \text{a.m./p.m.} \tag{1.4}$$

$$(\text{a.m.}, t) \mapsto \begin{cases} \text{p.m.} & \text{if } t = 11 \\ \text{a.m.} & \text{otherwise} \end{cases}$$

$$(\text{p.m.}, t) \mapsto \begin{cases} \text{a.m.} & \text{if } t = 11 \\ \text{p.m.} & \text{otherwise} \end{cases}$$

If it is a.m. and the clock reads 8, then it will still be a.m. at the next tick; but if it is a.m. and the clock reads 11, then the next tick will switch the meridian to p.m..

Again, the thing to note about the dynamics of the a.m./p.m. system is that they depend on what hour it is. The hour is imported as a *parameter* for the dynamics of the meridian system. We can draw the meridian system as a box like this:

$$(1.5)$$

We have the a.m./p.m. wire coming out, which carries the information of whether it is a.m. or p.m., just like the clock. But we also have a wire coming in, which carries the hour that we need as a parameter for our dynamics.

We can now express our whole clock (1.3) by wiring together our bare clock (1.2) and the a.m./p.m. system:

$$(1.6)$$
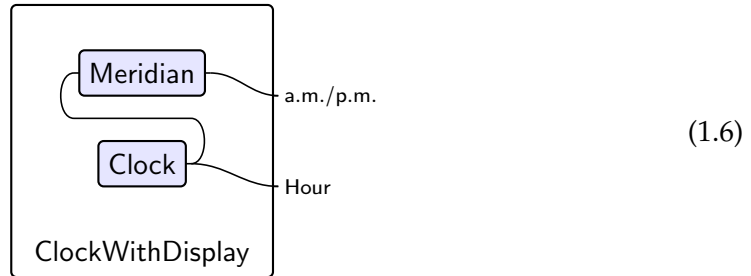
The resulting system has states

$$\text{HoursWithDisplay} := \text{Hour} \times \text{a.m.}/\text{p.m.}$$

each of which is a pair, e.g. (11, a.m.), consisting of an hour and a meridian reading. They update in a combined way, by using the hour shown on the clock face as the parameter we need for the Meridian system; this is expressed by having a wire from the output of Clock to the input of Meridian. In full, the dynamics looks like this:

$$\text{tick}' : \text{HoursWithDisplay} \to \text{HoursWithDisplay}$$
$$(t, m) \mapsto (\text{tick}(t), \text{next}(t, m))$$

where tick and next are as in (1.1) and (1.4).

Exercise 1.2.2.   Expand the definition of the combined system out in full, and check that it really does behave like the clock with a.m./p.m. display should.   ◊

Now that we have a working clock, we can use it for systems that need to know the time. For example, consider a diner that opens at 7a.m. and closes at 10p.m.. The states of this diner are

$$\mathsf{DinerState} = \{\mathtt{open}, \mathtt{closed}\}.$$

The diner's dynamics are then

$$\mathsf{dinerDynamics} : \mathsf{DinerState} \times \mathsf{HoursWithDisplay} \to \mathsf{DinerState}$$
$$(\mathtt{open}, (10, \mathrm{p.m.})) \mapsto \mathtt{closed}$$
$$(\mathtt{closed}, (7, \mathrm{a.m.})) \mapsto \mathtt{open}$$
$$(s, (t, m)) \mapsto s \quad \text{otherwise.}$$

Again, we can represent the diner by this box:

$$(1.7)$$
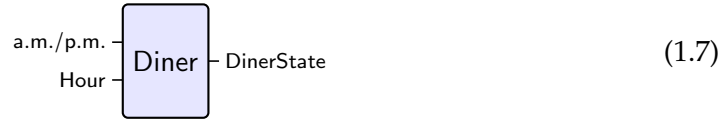
This time, we have two wires coming in, corresponding to the two parameters we need for the diner system: the hour and the meridien.

Assuming that the diner has a clock on its wall which it uses to decide whether to open or close, the full diner system would be given by wiring the clock with display into those input wires:

$$(1.8)$$
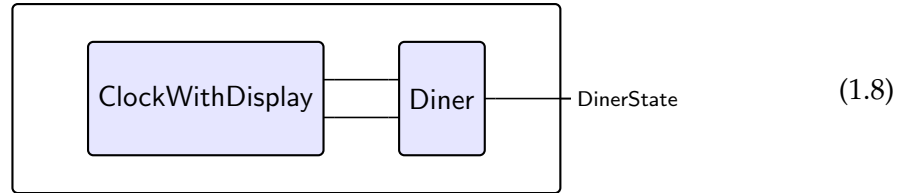
If we want to, we can peak into the clock with display and see that it is itself made out of a clock wired to a display:

$$(1.9)$$
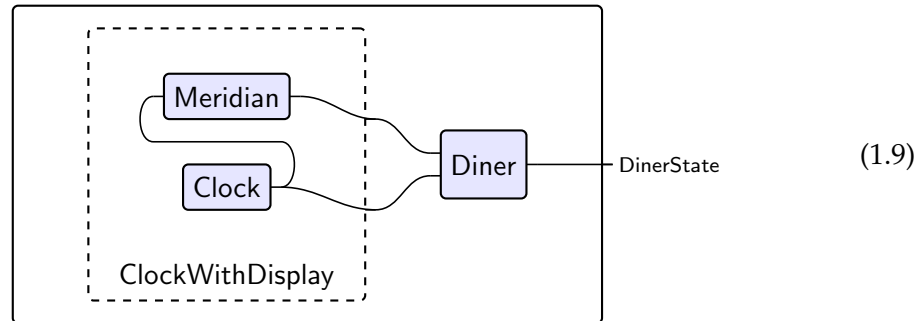
These examples are simple, but it doesn't take much more to get to some truly amazing phenomena. Consider this system: we have an infinite tape with a read-head at some integer position. On this infinite tape, we will write the symbols a, b, c, or d, or we will leave it blank: _. Together, the state of the tape and the position of the

read-head have states pairs (T, n) consisting of a function $\mathrm{T} : \mathbb{Z} \to \{a, b, c, d, \_\}$, telling us what symbol $\mathrm{T}(i)$ is found at position i of the tape, and a position n of the read-head:

$$\mathsf{Symbol} = \{a, b, c, d, \_\}$$
$$\mathsf{Tape} = \mathsf{Symbol}^{\mathbb{Z}}$$
$$\mathsf{Head} = \mathbb{Z}$$

The parameters that this system needs in order to change are a move-command and a write-command. The move-command will be either move left or move right, encoded as -1 or 1 respectively, and the write command will be one of the symbols that can be written on the tape:

$$\mathsf{Move} = \{-1, 1\} \qquad \text{and} \qquad \mathsf{Write} = \{a, b, c, d, \_\}.$$

The way this system changes is by writing the write command to the tape at the current position, and then moving according to the move command. As a function, this is:

$$\mathsf{execute} : \mathsf{Head} \times \mathsf{Tape} \times \mathsf{Move} \times \mathsf{Write} \to \mathsf{Head} \times \mathsf{Tape}$$

$$(n, i \mapsto \mathrm{T}(i), d, s) \mapsto \left(n + d, i \mapsto \begin{cases} \mathrm{T}(i) & \text{if } i \neq n \\ s & \text{if } i = n \end{cases} \right).$$

We can imagine that the system exposes the tape and the symbol under its read head. We can box this system up and draw it like so:

                                                       (1.10)
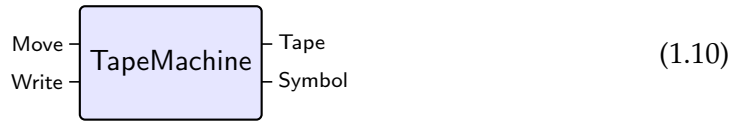
Now, we need one more simple ingredient to get our system going; a mysterious system of the form:

                                                       (1.11)
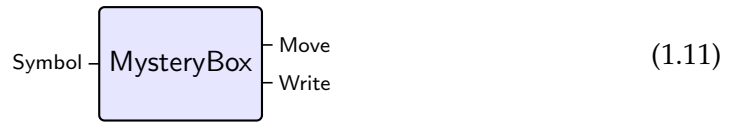
We can see that our mystery box will take in a symbol and put out a move command and a write command. The way our mystery box behaves is rather mysterious. It has six states $S \cong 6$, and its update rule is given by the following table, where the entry in the row i and the column s is written $(m, w) : s'$ to express the move command m, the write command w, and the next state s' that our mysterious system transitions to when input the symbol i in state s:

	1	2	3	4	5	6
a	(-1, b):1	(1, a):1	(-1, b):3	(1, b):2	(-1, b):6	(-1, b):4
b	(-1, a):1	(1, a):2	(-1, b):5	(1, a):4	(1, a):6	(1, a): 5
c	(1, d):2	(1, d):2	(-1, c):5	(1,d):4	(1, c):5	(1, a):1
d	(-1, c):1	(1, a):5	(-1, c):3	(1,d):5	(-1, b):3	end

(1.12)

Mysterious indeed. But when we wire the two together, magic happens!

$$(1.13)$$
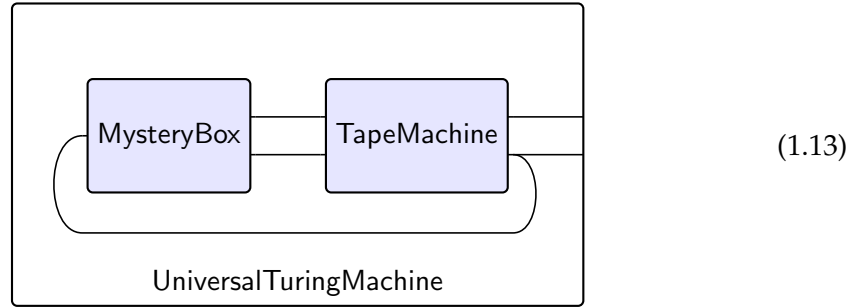
This is a universal Turing machine, i.e. when we encode everything into this strange alphabet, it is capable of arbitrarily complex calculation!

Even simple systems can have very interesting behavior when plugged in to the right environment.

That's a lot of informal definitions, we are ready for something precise:

Definition 1.2.3. A *deterministic system* S, also written as

$$\begin{pmatrix} \mathsf{update}_\mathsf{S} \\ \mathsf{expose}_\mathsf{S} \end{pmatrix} : \begin{pmatrix} \mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix},$$

consists of:

- a set $\mathsf{State}_\mathsf{S}$ of *states*;
- a set Out_S of *values for exposed variables*, or *outputs* for short;
- a set In_S of *parameter values*, or *inputs* for short;
- a function $\mathsf{expose}_\mathsf{S} : \mathsf{State}_\mathsf{S} \to \mathsf{Out}_\mathsf{S}$, the *exposed variable of state* or *expose* function, which takes a state to the output it yields; and
- a function $\mathsf{update}_\mathsf{S} : \mathsf{State}_\mathsf{S} \times \mathsf{In}_\mathsf{S} \to \mathsf{State}_\mathsf{S}$, the *dynamics* or *update* function which takes a state and a parameter and gives the next state.

We refer to the pair $\begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix}$ of exposed variable and parameter values as the *interface* of the system.

We can interpret this definition in any cartesian category $\mathcal{C}$; here, we have have used the category **Set** of sets.

Remark 1.2.4. Deterministic systems are also known as *Moore machines* in the literature. If the output set is taken to be {true, false}, then they are known as *deterministic automata*.

Often, these definitions also include a *start state* $s_0 \in \mathsf{State}_\mathsf{S}$ as part of the data. We don't do this.

Example 1.2.5. The Clock system can be seen as a deterministic system with:

$$\begin{pmatrix} \text{tick} \\ \text{id} \end{pmatrix} : \begin{pmatrix} \text{Hour} \\ \text{Hour} \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ \text{Hour} \end{pmatrix}.$$

In other words, it consists of

- State set $\text{State}_{\text{Clock}} = \text{Hour} = \{1, 2, \ldots, 12\}$.
- Output set $\text{Out}_{\text{Clock}} = \text{Hour}$.
- Input set $\text{In}_{\text{Clock}} = \{*\}$, a one element set.
- Readout function $\text{expose}_{\text{Clock}} = \text{id}_{\text{Hour}}$.
- update function $\text{update}_{\text{Clock}} : \text{hours} \times \{*\} \to \text{Hour}$ defined by $\text{update}_{\text{Clock}}(t, *) = \text{tick}(t)$.

Example 1.2.6. The term *Moore machine* is often used for the mathematical notion of deterministic system we've just presented, but it is also used for actual, real-life circuits which are designed on that principle.

For example, suppose that a wire carries the signals $\text{Wire} = \{\texttt{high}, \texttt{low}\}$. We can see a deterministic system M with input $\text{In}_{\text{M}} = \text{Wire}^n$ and $\text{Out}_{\text{M}} = \text{Wire}^k$ as a circuit with n incoming wires and k outgoing wires.[a] The state then describes the state of all the internal wires (and capacitors, etc.) in the circuit. We would wire up these systems by literally wiring them together.

Jaz: I would like to add an example of an implementation of a Moore machine into a circuit.

_____

[a]Of course, the notion of "incoming" and "outgoing" wires are ways we think about the circuit in design terms. Circuits aren't actually directed in this way. We'll think about undirected notions of system in **??**.

Note that when we say that a system doesn't have any parameters, as in Example 1.2.5, we don't take the parameter set to be empty but instead take it to have a single dummy value $\{*\}$, the one-element "hum of existence". In other words, having "no parameters" really means that the parameters are unchanging, or that there is no way to change the value of the parameters.

Also, we are just exposing the whole state with the system in Example 1.2.5. There is nothing preventing our systems from exposing their whole state, but often some aspects of the state are private, i.e. not exposed for use by other systems.

Exercise 1.2.7.   Write out the clock and meridian systems from (1.1) and (1.4) in terms of Definition 1.2.3. Really, this amounts to noticing which sets are the sets of states, which are the sets of inputs, and what (implicitly) are the sets of outputs.   ◊

Example 1.2.8 (SIR model). The set of states for a deterministic system doesn't need to be finite. The SIR model is an epimediological model used to study how a disease spreads through a population. "SIR" stands for "susceptible", "infected", and, rather ominously, "removed". This model is usually presented as a system of differential equations — what we will call a differential system — and we will see it in that form in **??**. But we can see a discrete approximation to this continuous model as a deterministic system.

A state of the SIR model is a choice of how many people are susceptible, how many are infected, and how many are removed. That is,

$$\text{State}_{\text{SIR}} = \left\{ \begin{bmatrix} s \\ i \\ r \end{bmatrix} \;\middle|\; s, i, r \in \mathbb{R} \right\} \cong \mathbb{R}^3.$$

is a 3-place vector of real numbers. We will again expose the whole state, so $\text{Out}_{\text{SIR}} = \text{State}_{\text{SIR}}$ and $\text{expose}_{\text{SIR}} = \text{id}$.

The idea behind the SIR model is that if a susceptible person comes in contact with an infected person, then they have a chance of becoming infected too. And, eventually, infected persons will be removed from the model, either by recovering (a gentler way to read the "R") or by dying. So we need two parameters: the rate a of infection and the rate b of removal:

$$\text{In}_{\text{SIR}} = \left\{ \begin{bmatrix} a \\ b \end{bmatrix} \;\middle|\; a, b \in \mathbb{R} \right\} = \mathbb{R}^2.$$

Now, we can show how a population will develop according to this model by defining the update function:

$$\text{update}_{\text{SIR}} : \text{State}_{\text{SIR}} \times \text{In}_{\text{SIR}} \to \text{State}_{\text{SIR}} \tag{1.14}$$

$$\left(\begin{bmatrix} s \\ i \\ r \end{bmatrix}, \begin{bmatrix} a \\ b \end{bmatrix} \right) \mapsto \begin{bmatrix} s - asi \\ i + asi - bi \\ r + bi \end{bmatrix} \tag{1.15}$$

Example 1.2.9. If a deterministic system has a small finite set of states, then we can draw it entirely as a *transition diagram*:
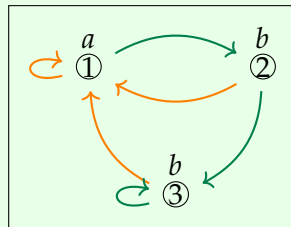
Note that every node has an orange and a green arrow emanating from it, but that there are no rules on how many arrows point to it.

This diagram describes the following system S:

$$\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \{1,2,3\} \\ \{1,2,3\} \end{pmatrix} \leftrightarrows \begin{pmatrix} \{\texttt{green, orange}\} \\ \{a,b\} \end{pmatrix}.$$

That is, we have

- $\text{States}_S = \{1,2,3\}$.
- $\text{In}_S = \{\texttt{green, orange}\}$,
- $\text{Out}_S = \{a,b\}$,
-

$$\text{expose}_S : \text{States}_S \rightarrow \text{Out}_S \qquad \text{update}_S : \text{States}_S \times \text{In}_S \rightarrow \text{In}_S$$

$$1 \mapsto a \qquad\qquad (1, \texttt{green}) \mapsto 2$$
$$2 \mapsto b \qquad\qquad (1, \texttt{orange}) \mapsto 1$$
$$3 \mapsto b \qquad\qquad (2, \texttt{green}) \mapsto 3$$
$$(2, \texttt{orange}) \mapsto 1$$
$$(3, \texttt{green}) \mapsto 3$$
$$(3, \texttt{orange}) \mapsto 1$$

To draw a transition diagram of a system S, we draw each state $s \in \text{States}_S$ as a bubble filled with the label $\text{expose}_S(s)$, and for each parameter $i \in \text{In}_S$ we draw an arrow from s to $\text{update}_S(s, i)$. For a diagram like this to be a transition diagram, every node must have an edge leaving it for each parameter.

Exercise 1.2.10.   Draw the Clock system (Example 1.2.5) as a transition diagram.   ◊

Example 1.2.11 (Deterministic Finite Automata). A *deterministic finite automaton* (DFA) is a simple model of computation. Given our definition of deterministic system, DFAs are easy enough to define: they are just the deterministic systems with finitely many states whose output values are either `true` or `false`.

This means that the exposed variable of state $\text{expose}_S : \text{States}_S \rightarrow \{\texttt{true, false}\}$ is a boolean valued function. We say a state s is an *accept state* if $\text{expose}_S(s) = \texttt{true}$, and a *reject state* if $\text{expose}_S(s) = \texttt{false}$.

The idea is that a DFA is a question answering machine. Given a starting state s_0 and a sequence of input values $i_1, \ldots, i_n$, we get a sequence of states by $s_{t+1} := \text{update}_S(s_t, i_t)$. The answer to the question is "yes" if s_n is an accept state, and "no" if s_n is a reject state.

There is an important special case of deterministic systems which appear very commonly in the literature: the *closed* systems. These are the systems which have no

parameters, and which expose no variables. They are closed off from their environment, and can't be wired into any other systems.

As mentioned after Example 1.2.5, when we say "no" in this way — no parameters, no variables — we should be careful with what we mean exactly. We mean that there is no *variation* in the parameters or variables, that they are trivial. That is, we make the following definition.

Definition 1.2.12. A deterministic system S is *closed* if both In_S and Out_S have only one element

$$\mathsf{In}_S \cong \{*\} \cong \mathsf{Out}_S.$$

Exercise 1.2.13. Show that to give a closed system

$$\begin{pmatrix} \mathsf{update}_S \\ \mathsf{expose}_S \end{pmatrix} : \begin{pmatrix} \mathsf{State}_S \\ \mathsf{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ \{*\} \end{pmatrix},$$

one just needs to choose a set State_S and an update function $\mathsf{update}_S : \mathsf{State}_S \to \mathsf{State}_S$.

◇

Given that we are mostly interested in how systems wire together, it may seem strange to draw attention to the closed systems that *can't* be wired into anything else. But we will often end up with a closed system as the result of wiring together some systems.

For example, suppose we have an Agent acting within a Environment. The agent will take an action, and the evironment will respond to that action. Depending on the action taken and response given, the agent and the environment will update their states. We can model this by the following wiring diagram:

Closed Agent/Environment Model
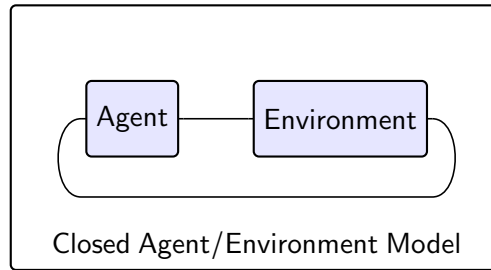
To have this be a closed model is to think — or pretend — that the our model of the Agent and the Environment includes all possible external parameters, that it is well isolated from its own environment.

Exercise 1.2.14. What would happen to a system S if its set of parameters or output values were actually empty sets? Let's find out.

1. Suppose $\mathsf{In}_S = \varnothing$. Explain the content of a deterministic system

$$\begin{pmatrix} \mathsf{update}_S \\ \mathsf{expose}_S \end{pmatrix} : \begin{pmatrix} \mathsf{State}_S \\ \mathsf{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \varnothing \\ \{*\} \end{pmatrix}.$$

2. Suppose $\mathsf{Out}_S = \varnothing$. Explain the content of a deterministic system

$$\begin{pmatrix} \mathsf{update}_S \\ \mathsf{expose}_S \end{pmatrix} : \begin{pmatrix} \mathsf{State}_S \\ \mathsf{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ \varnothing \end{pmatrix}. \qquad\qquad \Diamond$$

1.2.15 Differential systems

La nature ne fait jamais des sauts - Liebniz

A quirk of modeling dynamical systems as determinstic systems is that determinstic systems lurch from one state to the next. In life, there are no next moments. Time, at least at human scales and to a first approximation, flows continuously.

Instead of modelling the "next" state a system will be in, we can model *how the system is tending to change*, in the moment. In order to do this, we need to make concession in the way we model the states of our system: we must assume they form a continuum themselves.

For example, suppose we are studying a population of Rabbits. We can measure the rate at which rabbits are born, and the rate they die. Then the population changes according to these rates. We can express this dependency of the change in population on certain rates with a differential equation:

$$\frac{dr}{dt} = \mathsf{b}_{\mathsf{Rabbits}} \cdot r - \mathsf{d}_{\mathsf{Rabbits}} \cdot r$$

where $r \in \mathbb{R}$ is the population of rabbits (considered as a real number for convenience), and the rates $\mathsf{b}_{\mathsf{Rabbits}}$ and $\mathsf{d}_{\mathsf{Rabbits}}$. The state of our system of Rabbits is the current population of rabbits, so $\mathsf{State}_{\mathsf{Rabbits}} = \mathbb{R}$, while we take the birth and death rates as parameters, so that $\mathsf{In}_{\mathsf{Rabbits}} = \mathbb{R} \times \mathbb{R}$. Accordingly, we can box the rabbit system up like so:

$$(1.16)$$
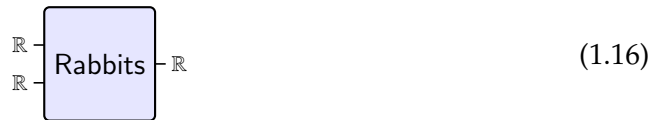
Now, rabbits are prey; they are eaten by other animals. That means that the rate at which rabbits die will depend on how often they are being eaten, and how often they are being eaten will depend on how many predators there are out there.

Now, the population of any predator will also change according to a birth rate and death rate. Suppose we have a similarly defined system of Foxes goverened whose population is governed by the differential equation

$$\frac{df}{dt} = \mathsf{b}_{\mathsf{Foxes}} \cdot f - \mathsf{d}_{\mathsf{Foxes}} \cdot f.$$

We can box up this system like so:

$$\begin{array}{c}\mathbb{R} \\ \mathbb{R}\end{array}\boxed{\text{Foxes}}\;\mathbb{R} \tag{1.17}$$

Now, we want the death rate of rabbits to depend on the number of foxes. But we also need the birth rate of the foxes to depend on the number of rabbits; after all, if a fox has nothing to eat, it has no time for hanky-panky. So we will add the following system of equations to the mix:

$$\begin{cases} \mathrm{d}_{\text{rabbits}} = c_1 f \\ \mathrm{b}_{\text{Foxes}} = c_2 r \end{cases}$$

Making these substitutions, we get the following system of differential equations:

$$\begin{cases} \frac{dr}{dt} = \mathrm{b}_{\text{Rabbits}} \cdot r - c_1 f r \\ \frac{df}{dt} = c_2 r f - \mathrm{d}_{\text{Foxes}} \cdot f \end{cases}$$

We are setting the parameters of the systems of Rabbits and Foxes according to the states of the other system. That is, we are wiring up the systems of Rabbits and Foxes:

$$(1.18)$$
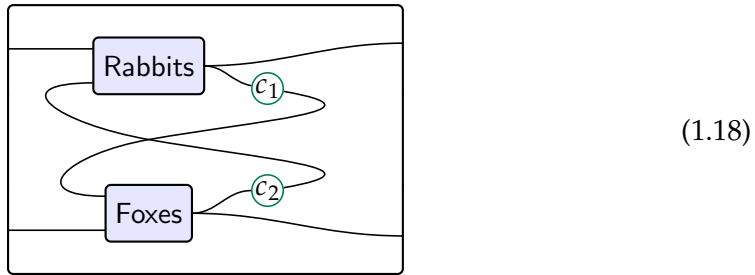
The resulting system is called the *Lotka-Volterra predator-prey model*, and it is a simple differential model of the ways that the population of a predator species depends on the population of a prey species, and vice-versa.

Where before our boxes were filled with deterministic systems, now they are filled with systems of (first order, ordinary) differential equations. We call these *differential systems*.

Definition 1.2.16. A *(first order, ordinary) differential system* S with n state variables, m parameters, and k exposed variables

$$\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \mathbb{R}^n \\ \mathbb{R}^n \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^m \\ \mathbb{R}^k \end{pmatrix}$$

consists of:
- An n-dimensional state space $\text{States}_S = \mathbb{R}^n$.
- An m-dimensional parameter space $\text{In}_S = \mathbb{R}^m$.
- A k-dimensional space of exposed variable values $\text{Out}_S = \mathbb{R}^k$.

- A smooth function $\text{update}_S : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ which gives us the derivative of each state variable at each time, so that the defining system of differential equations of S reads

$$\begin{cases} \frac{ds_1}{dt} & = \text{update}_{S1}(s, i) \\ & \vdots \\ \frac{ds_n}{dt} & = \text{update}_{Sn}(s, i). \end{cases}$$

- k exposed variables $\text{expose}_{Si} : \mathbb{R}^n \to \mathbb{R}$, which organize into a single smooth function $\text{expose}_S : \mathbb{R}^n \to \mathbb{R}^k$.

Remark 1.2.17. Definition 1.2.16 looks remarkably similar to Definition 1.2.3. As we mentioned, Definition 1.2.3 can be interpreted in any cartesian category, including the category **Euc** of Euclidean spaces and smooth maps (**??**). It appears that a differential system is the same thing as a deterministic system in the cartesian category **Euc**. But while the $\mathbb{R}^n$s occuring in $\text{update}_S : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ look the same, they are in fact playing very different roles. The $\mathbb{R}^n$ on the left is playing the role of the state space, while the $\mathbb{R}^n$ on the left is playing the role of the tangent space at s for some state $s \in \mathbb{R}^n$. The difference will be felt in **??** when we study behaviors of systems: the way a trajectory is defined for differential systems is different for differential systems and determinstic systems. For differential systems, a trajectory will be a solution to the system of differential equations, that is, a function $s : \mathbb{R} \to \mathbb{R}^n$ which satisfies

$$\frac{ds}{dt}(t) = \text{update}_S(s(t), i(t)).$$

for all choice of times t, while for a deterministic system a trajectory would be a sequence s_j of states so that $s_{j+1} = \text{update}_S(s_j, i_j)$.

We will see precisely how this difference is made manifest in the formal definition of a dynamical system doctrine as the choice of *section* in **??**.

Remark 1.2.18. There are other doctrines of differential systems that one can define (for example, allowing the state space to be a manifold), but in this book we will work with this simpler doctrine.

Example 1.2.19. The system of Rabbits has 1 state variable (the population of rabbits), 2 parameters (the birth and death rates of the rabbbits), and 1 exposed variable. It exposes its whole state, so that $\text{expose}_S = \text{id}$, and its update is given by

$$\text{update}_{\text{Rabbits}}(r, (b_{\text{Rabbits}}, d_{\text{Rabbits}})) = b_{\text{Rabbits}} \cdot r - d_{\text{Rabbits}} \cdot r.$$

The whole Lotka Voltera model of Eq. (1.18) has 2 state variables (the populations of rabbits and of foxes), 2 parameters (the birth rate of rabbits and the death rate of

foxes), and 2 exposed variables. It exposes its whole state, and its update is given by

$$\text{update}_{\text{LK}}\left(\begin{bmatrix} r \\ f \end{bmatrix}, (\text{b}_{\text{Rabbits}}, \text{d}_{\text{Foxes}})\right) = \begin{bmatrix} \text{b}_{\text{Rabbits}} \cdot r - c_1 r f \\ c_2 f r - \text{d}_{\text{Foxes}} \cdot f \end{bmatrix}$$

One might wonder why we said this system has 2 parameters when there are also the rate constants c_1 and c_2 involved — aren't they also parameters? We chose them to be *constant*, where our parameters might vary over time. We could have made them parameters instead — it was an arbitrary choice in how to make the model.

Example 1.2.20. The most basic epidemiological model is the SIR model. This models the spread of disease through a population. People are either *susceptible* (S), *infected* (I), *recovered* or more ominously *removed* (R) from the model. When a susceptible person comes in contact with an infected person, they have a chance to become infected; this means that the population of susceptible people tends downwards in proportion to the number of susceptible and the number of infected people, and the population of infected people tends up by the same amount. On the other hand, infected people will eventually be removed from the model, either by recovering or dieing; this means that the population of infected people tends downwards proportional to the current infected population, while the removed population tends upwards by the same amount. Said as a system of differential equations, this means:

$$\begin{cases} \frac{dS}{dt} = -\alpha SI \\ \frac{dI}{dt} = \alpha SI - \beta I \\ \frac{dR}{dt} = \beta I \end{cases} \tag{1.19}$$

The SIR model is a differential system with 3 state variables (S, I, and R) and 2 parameters (α and β). We will suppose that it exposes its whole state: $\text{expose}_{\text{SIR}} = \text{id}$. The update is given by

$$\text{update}_{\text{SIR}}\left(\begin{bmatrix} S \\ I \\ R \end{bmatrix}, (\alpha, \beta)\right) = \begin{bmatrix} -\alpha SI \\ \alpha SI - \beta I \\ \beta I \end{bmatrix}.$$

In order to model higher order systems of ordinary differential equations, we will resort to the standard trick of encoding them as larger systems of first order systems. For example, to encode a second order differential equation in n variables, we would set the state space to be $\mathbb{R}^{2n}$ with state variables $(s, \dot{s})$ (the first n being s, the second n being $\dot{s}$). We then add the equations $\frac{ds}{dt} = \text{update}_{S,s}((s, \dot{s}), i) := \dot{s}$ to the system $\frac{d\dot{s}}{dt} = \text{update}_{S,\dot{s}}((s, \dot{s}), i)$ of second order differential equations we were trying to model.

Often, we want to think of the state variables $\dot{s}$ as *hidden* technical tricks. For this reason, we will often only expose the "actual" state variables s. This is one use for the function expose_S.

Example 1.2.21. Consider a mass m on a spring with a spring constant of c, taking position $s(t)$ at time t. Newton's second law then says that the acceleration of the mass is proportional to the force exerted upon it:

$$m\frac{d^2s}{dt} = -cs. \tag{1.20}$$

We can express this as a differential system in the following way. We take the state variables to be s and $\dot{s}$: $\text{State}_{\text{Spring}} := \mathbb{R}^2$. We will suppose that the mass and the spring constant are constant, so that this system takes no parameters: $\text{In}_{\text{Spring}} := \mathbb{R}^0 = \{*\}$. We will only expose the position of the spring, and not its velocity: $\text{Out}_{\text{Spring}} := \mathbb{R}$ and $\text{expose}_{\text{Spring}}(s, \dot{s}) := s$. Finally, the dynamics of the system are given by:

$$\text{update}_{\text{Spring}}\left(\begin{bmatrix} s \\ \dot{s} \end{bmatrix}\right) := \begin{bmatrix} \dot{s} \\ -\frac{cs}{m} \end{bmatrix}.$$

This is a way of re-writing **??** as a system of first order differential equations:

$$\begin{cases} \frac{ds}{dt} &= \dot{s} \\ \frac{d\dot{s}}{dt} &= -\frac{cs}{m} \end{cases}$$

Before we go on, we should clarify the category that we are working in when we work with our differential systems.

Definition 1.2.22. The category **Euc** is the category of *Euclidean spaces* and smooth maps between them. The objects of **Euc** are $\mathbb{R}^n$ for all $n \in \mathbb{N}$, and a morphism $f : \mathbb{R}^n \to \mathbb{R}^m$ is a smooth map.

We note that **Euc** is a cartesian category with $\mathbb{R}^n \times \mathbb{R}^m = \mathbb{R}^{n+m}$ and $1 = \mathbb{R}^0$.

1.3   Wiring together systems with lenses

In the last section, we saw the formal definition of deterministic and differential systems and a few examples of them. In this section, we'll see how to wire systems together — as we did in Section 1.1 for the clock and the universal Turing machine, and in Section 1.2.15 for the Lotka-Volterra predator prey model — to make more complex systems. We will do this using an interesting notion coming from the world of function programming: a *lens*.

1.3.1 Lenses and lens composition

A lens is a framework for bi-directional information passing.

Definition 1.3.2. A *lens* $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ in a cartesian category $\mathcal{C}$ consists of:
- A *passforward* map $f : A^+ \to B^+$, and
- a *passback* map $f^\sharp : A^+ \times B^- \to A^-$.

The most useful thing about lenses is that they *compose*.

Definition 1.3.3. Let $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ and $\begin{pmatrix} g^\sharp \\ g \end{pmatrix} : \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} C^- \\ C^+ \end{pmatrix}$ be lenses in a cartesian category $\mathcal{C}$. We define their composite

$$\begin{pmatrix} g^\sharp \\ g \end{pmatrix} \circ \begin{pmatrix} f^\sharp \\ f \end{pmatrix}$$

to have passforward $g \circ f$ and passback

$$(a^+, c^-) \mapsto f^\sharp \left(a^+, g^\sharp(f(a^+), c^-) \right).$$

Here's a picture so that you can see the information flow for the composite of lenses:[1]

$$(1.21)$$
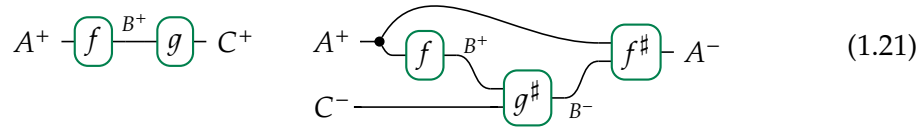
Remark 1.3.4. Even though our definition of lens was given in an arbitrary cartesian category $\mathcal{C}$, we felt comfortable defining it in terms of elements. Going forward, we will also reason with it using elements. This trick works for any cartesian category by using "generalized elements". We interpret an "element" x in an object X as a map $x : Z \to X$. If we do work with x to get a new element $f(x)$ of Y, then by the Yoneda lemma there is a map $f : X \to Y$ in the category which does that work by post-composition: $f(x) = f \circ x$.

The take-away is that even in a totally arbitrary cartesian category whose objects are not sets of any kind, we can still reason about them as if they were — at least when it comes to pairing elements and applying functions.

This gives us a category of lenses in any cartesian category $\mathcal{C}$.

---

[1]We draw this with a different style—green boxes, etc.—so that the reader will not confuse it with our usual wiring diagrams for systems. These are not dynamic in any way; everything below is just a set.

Definition 1.3.5. Let $\mathcal{C}$ be a cartesian category. Then the category $\mathbf{Lens}_{\mathcal{C}}$ has:

- as objects, the pairs $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ of objects in $\mathcal{C}$, which we will call *arenas*.

- as morphisms, the lenses $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$.

- The identity lens is $\begin{pmatrix} \pi_2 \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$, where $\pi_2 : A^+ \times A^- \to A^-$ is the projection.

Composition is given by lens composition as in **??**.

Jaz: Emily suggests giving a better motivation for lens. E.g. saying that a lens is a common generalization of a system and a wiring diagram which can be composed.

Remark 1.3.6. The category of lenses is special among categories because it is named for its *maps* (which are the lenses), rather than its objects (which are the arenas). This is because we will later meet another category, the *category of charts* (See **??**), whose objects are the arenas but whose maps are not lenses. Finally, in **??** we will meet a *double category*[2] $\mathbf{Arena}_{\mathcal{C}}$ which combines these two categories whose objects are arenas and which *is* named after its objects. In **??**, we will explain the name "arena" and its role in the theory of dynamical systems.

Exercise 1.3.7.

1. Draw the composite of two lenses in the style of (**??**).
2. Check that $\mathbf{Lens}_{\mathcal{C}}$ is actually a category. That is, check that lens composition is associative, and that the identity lens is an identity for it. (Hint: You can use your drawing for this. You can slide the function beads around on the strings; if you pull a function bead past a split in the string, you have to duplicate it (since that split represents the duplication function).)   ◊

Like any good categorical construction, $\mathbf{Lens}_{\mathcal{C}}$ varies functorially in its variable cartesian category $\mathcal{C}$.

Proposition 1.3.8 (Functoriality of **Lens**). Every cartesian functor $F : \mathcal{C} \to \mathcal{D}$ induces a functor $\begin{pmatrix} F \\ F \end{pmatrix} : \mathbf{Lens}_{\mathcal{C}} \to \mathbf{Lens}_{\mathcal{D}}$ given by

$$\begin{pmatrix} F \\ F \end{pmatrix} \begin{pmatrix} f^\sharp \\ f \end{pmatrix} = \begin{pmatrix} Ff^\sharp \circ \mu^{-1} \\ Ff \end{pmatrix}$$

where $\mu = (F\pi_1, F\pi_2) : F(X \times Y) \xrightarrow{\sim} FX \times FY$ is the isomorphism witnessing that F preserves products.

---

[2]A double category is like a category with two different kinds of morphisms and a way for them to commute. See **??** for the precise definition and the accompanying discussion.

Proof Sketch. Because lenses are defined just using the cartesian product, and F preserves these products, it commutes with everything in sight. ☐

Exercise 1.3.9.
1. What does the functor $\binom{F}{F}$: **Lens**$_\mathcal{C}$ → **Lens**$_\mathcal{D}$ do on objects?

2. Complete the proof of **??**, by showing that $\binom{F}{F}$ really is a functor. ◊

1.3.10 Deterministic and differential systems as lenses

The reason we are interested in lenses and lens composition is because dynamical systems of various sorts are themselves lenses. As written in Definition 1.2.3, a system S is a lens in the category of sets of the form

$$\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \text{State}_S \\ \text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}.$$

In fact, the deterministic systems are precisely the lenses whose input arena is of the form $\binom{S}{S}$. This means that we can compose a system S with a lens $\binom{f^\sharp}{f}$: $\binom{\text{In}_S}{\text{Out}_S} \leftrightarrows \binom{I}{O}$ to get a new dynamical system

$$\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} \mathbin{\overset{\circ}{,}} \begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} \text{State}_S \\ \text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} I \\ O \end{pmatrix}$$

with a new interface!

Similarly, a differential system is a lens in the category **Euc** of the form

$$\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \mathbb{R}^n \\ \mathbb{R}^n \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^m \\ \mathbb{R}^k \end{pmatrix}.$$

We can then compose this with lenses in **Euc** to get new differential systems!

We can use this observation to wire together different systems. We separate this into to phases: first we put two systems in parallel, then we wire them together using a lens. The first phase, combine two systems without having them interact, is achieved through what we call the *parallel product* and denote ⊗. Two put two arenas $\binom{A_1}{B_1}$ and $\binom{A_2}{B_2}$ in parallel we just take their product in our cartesian category $\mathcal{C}$:

$$\begin{pmatrix} A_1 \\ B_1 \end{pmatrix} \otimes \begin{pmatrix} A_2 \\ B_2 \end{pmatrix} := \begin{pmatrix} A_1 \times A_2 \\ B_1 \times B_2 \end{pmatrix}$$

In **??** we define parallel product for morphisms in **Lens**, i.e. for general lenses.

Definition 1.3.11. For lenses $\left(\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right) : \left(\begin{smallmatrix} A_1 \\ B_2 \end{smallmatrix}\right) \leftrightarrows \left(\begin{smallmatrix} C_1 \\ D_1 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} g^\sharp \\ g \end{smallmatrix}\right) : \left(\begin{smallmatrix} A_2 \\ B_2 \end{smallmatrix}\right) \leftrightarrows \left(\begin{smallmatrix} C_2 \\ D_2 \end{smallmatrix}\right)$, we define their *parallel product*

$$\begin{pmatrix} f^\sharp \\ f \end{pmatrix} \otimes \begin{pmatrix} g^\sharp \\ g \end{pmatrix} : \begin{pmatrix} A_1 \times A_2 \\ B_1 \times B_2 \end{pmatrix} \leftrightarrows \begin{pmatrix} C_1 \times C_2 \\ D_1 \times D_2 \end{pmatrix}$$

to have passforward $f \times g$ and passback

$$((b_1, b_2), (c_1, c_2)) \mapsto (f^\sharp(b_1, c_2), g^\sharp(b_2, c_2)).$$

In terms of morphisms, this is

$$(B_1 \times B_2) \times (C_1 \times C_2) \xrightarrow{\sim} (B_1 \times C_1) \times (B_2 \times C_2) \xrightarrow{f^\sharp \times g^\sharp} A_1 \times A_2.$$

Together with $\left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right)$, this gives $\mathbf{Lens}_\mathcal{C}$ the structure of a monoidal category.

Remark 1.3.12. We will show a slick way to prove that the parallel product does indeed make $\mathbf{Lens}_\mathcal{C}$ into a monoidal category in **??**.

Exercise 1.3.13.   Show the parallel product of morphisms as in **??** using the string diagram notation from (**??**).   ◊

Proposition 1.3.14. Let $F : \mathcal{C} \to \mathcal{D}$ be a cartesian functor. The induced functor $\left(\begin{smallmatrix} F \\ F \end{smallmatrix}\right) :$ $\mathbf{Lens}_\mathcal{C} \to \mathbf{Lens}_\mathcal{D}$ is strong monoidal with respect to the parallel product.

Proof. Since F preserves products, and Jaz: finish .   □

Given two dynamical systems S_1 and S_2, their parallel product $\mathsf{S}_1 \otimes \mathsf{S}_2$ is defined explicitly as follows:
- $\mathrm{States}_{\mathsf{S}_1 \otimes \mathsf{S}_2} := \mathrm{States}_{\mathsf{S}_1} \times \mathrm{States}_{\mathsf{S}_2}$.
- $\mathrm{Out}_{\mathsf{S}_1 \otimes \mathsf{S}_2} := \mathrm{Out}_{\mathsf{S}_1} \times \mathrm{Out}_{\mathsf{S}_2}$.
- $\mathrm{In}_{\mathsf{S}_1 \otimes \mathsf{S}_2} := \mathrm{In}_{\mathsf{S}_1} \times \mathrm{In}_{\mathsf{S}_2}$.
- $\mathrm{expose}_{\mathsf{S}_1 \otimes \mathsf{S}_2}((s_1, s_2)) = (\mathrm{expose}_{\mathsf{S}_1}(s_1), \mathrm{expose}_{\mathsf{S}_2}(s_2))$.
- $\mathrm{update}_{\mathsf{S}_1 \otimes \mathsf{S}_2}((s_1, s_2), (i_1, i_2)) = (\mathrm{update}_{\mathsf{S}_1}(s_1, i_1), \mathrm{update}_{\mathsf{S}_2}(s_2, i_2))$.

This can be expressed as the following wiring diagram:
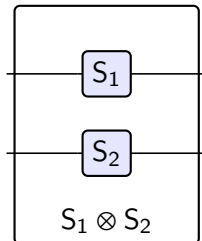
(1.22)

If we imagine physically wiring together our boxes, the first thing we would need to do is collect them together like this; then we can proceed to wire them. We will do exactly this with our systems: first we will take their parallel product, and then we compose it with a lens that represents the wiring diagram.

Example 1.3.15. We can describe the ClockWithDisplay system (reproduced below) as a composite of lenses.

(1.23)
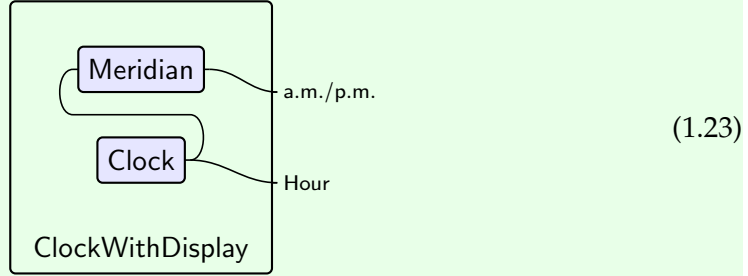
First, we take the parallel product of Meridian and Clock (see Exercise 1.2.7) to get the system

$$\text{Meridian} \otimes \text{Clock} : \begin{pmatrix} \text{a.m./p.m.} \times \text{Hour} \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix} \leftrightarrows \begin{pmatrix} 1 \times \text{Hour} \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix}.$$

Now, we will express the wiring pattern in **??** as a lens

$$\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} 1 \times \text{Hour} \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix} \leftrightarrows \begin{pmatrix} 1 \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix}.$$

We do this by setting

$$w(m, h) := (m, h), \text{ and}$$
$$w^\sharp((m, h), *) := (*, h).$$

Seen as a wiring diagram on its own, $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$ looks like this:

(1.24)
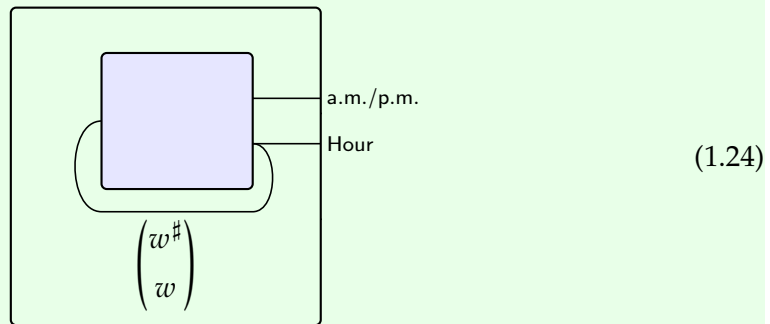
We can then see that

$$\text{ClockWithDisplay} = \begin{pmatrix} w^\sharp \\ w \end{pmatrix} \circ (\text{Meridian} \otimes \text{Clock})$$

just like we wanted! In terms of wiring diagrams, this looks like:

(1.25)
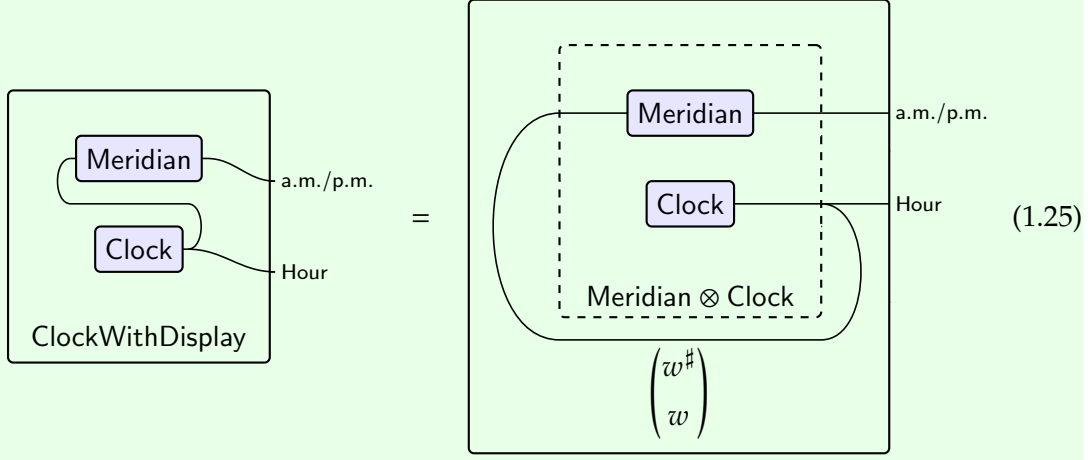
Example 1.3.16. We can describe the Lotka-Volterra predator prey model (reproduced below) as a composite of lenses.

(1.26)
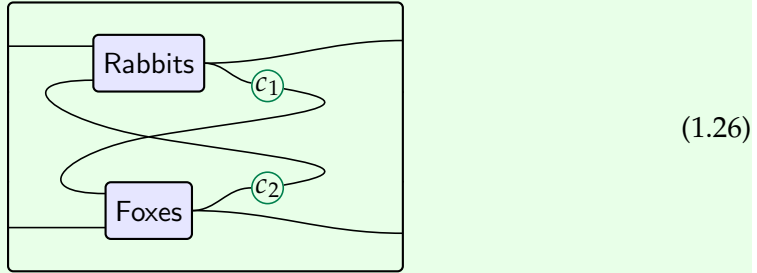
We can express the wiring pattern in **??** as a lens

$$\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R} \end{pmatrix} \otimes \begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R}^2 \end{pmatrix}.$$

We do this by setting

$$w(r, f) := (r, f)$$
$$w^\sharp((r, f), (a, b)) := (a, c_2 f, c_1 r, b)$$

We can draw $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$ as a wiring diagram on its own like this:
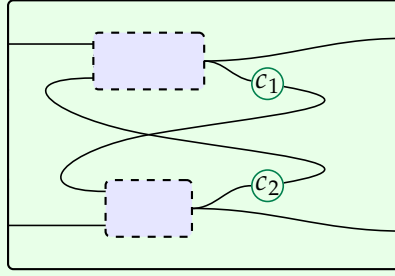
$$(1.27)$$

Filling those boxes with the systems of Rabbits and Foxes corresponds to taking the composite

$$(\mathsf{Rabbits} \otimes \mathsf{Foxes}) \ \mathring{,} \begin{pmatrix} w^\sharp \\ w \end{pmatrix}$$

of lenses.

Wiring together transition diagrams. When a deterministic system is presented as a transition diagram (See Example 1.2.9), its dynamics is given by reading the input and following the arrow with that label, and then output the label on the resulting node. When we wire together systems presented as transition diagrams, the dynamics then involve reading the input labels of all inner systems, moving along all the arrows with those labels, and then outputing the labels at each state, possible into the input of another system.

Exercise 1.3.17. Here are two systems, S_1 and S_2 presented in terms of transition diagrams. The task is calculate the transition diagram of a system made by wiring them together.

First, let $\mathsf{Colors} = \{\mathsf{red}, \mathsf{blue}, \mathsf{green}\}$ and let $\mathsf{Bool} = \{\mathsf{true}, \mathsf{false}\}$. Here is our first system S_1, which has interface $\begin{pmatrix} \mathsf{Bool} \\ \mathsf{Colors} \end{pmatrix}$:
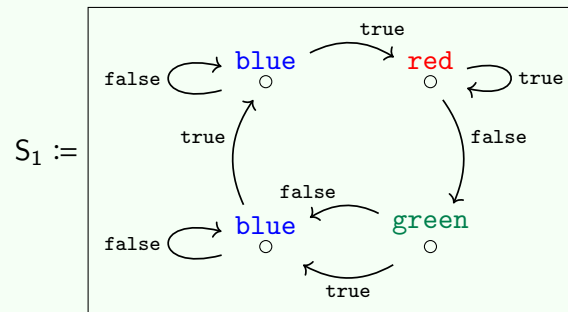
$$(1.28)$$

Our second system S_2 will have interface $\begin{pmatrix} \mathsf{Colors} \\ \mathsf{Bool} \end{pmatrix}$:

$$S_2 := \begin{array}{c}\begin{array}{ccc} \text{true} & & \text{false} \\ \circ & & \circ \\ & & \\ & \text{true} & \\ & \circ & \end{array}\end{array} \tag{1.29}$$
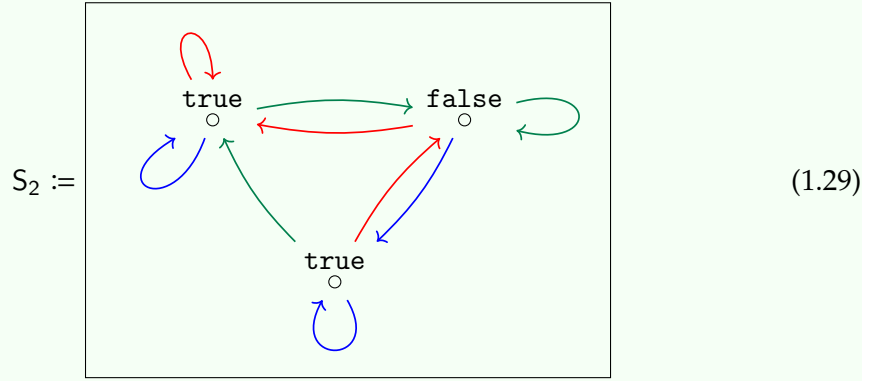
1. Write down the transition diagram of the system obtained by connecting the above systems according to the following wiring diagram:

$$S := \boxed{\; \boxed{S_1}\!-\!\boxed{S_2}\;}$$
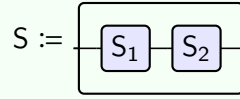
2. Explain how to understand the dynamics of this S in terms of the component systems S_1 and S_2.   ◊

Multi-city SIR models   In Example 1.2.8 and **??**, we saw deterministic and differential SIR models. Each models the spread of a disease through a single population. But what about a global pandemic where the disease is spreading through many local populations?

To model the spread of a disease through many different populations, we can use what is called a *multi-city SIR model*. We call each population a "city", and for now we will take flow of population between each city to be known constants. We can define a city as a differential system; then certain wiring diagrams of cities will correspond to multi-city models!

Definition 1.3.18. A City in a multi-city SIR model is a differential system

$$\begin{array}{c}\mathbb{R}^3 \\ \mathbb{R}^3\end{array}\!\!-\!\boxed{\text{City}}\!-\mathbb{R}^3 \tag{1.30}$$

A city is defined by:

- $\text{State}_{\text{City}} := \left\{ \begin{bmatrix} S \\ I \\ R \end{bmatrix} \;\middle|\; S, I, R \in \mathbb{R} \right\} = \mathbb{R}^3.$
- $\text{In}_{\text{City}} = \{(\texttt{inflow}, \texttt{outflow}) \mid \texttt{inflow}, \texttt{outflow} \in \mathbb{R}^3\} = \mathbb{R}^3 \times \mathbb{R}^3$
- $\text{Out}_{\text{City}} = \text{State}_{\text{City}} = \mathbb{R}^3.$
- $\text{expose}_S = \text{id}.$

-

$$\text{update}_S\left(\begin{bmatrix} S \\ I \\ R \end{bmatrix}, (\texttt{inflow}, \texttt{outflow})\right) := \begin{bmatrix} -k_1 SI + \texttt{inflow}_1 - \texttt{outflow}_1 \\ k_1 SI - k_2 I + \texttt{inflow}_2 - \texttt{outflow}_2 \\ k_1 I + \texttt{inflow}_3 - \texttt{outflow}_3 \end{bmatrix}$$

for some choice of constants k_1 and k_2.

That is, each city will run its own SIR model, and each of the three populations can flow between cities.

Now, to define a mutli-city SIR model, we need to know what cities we are dealing with and how population flows between them. We'll call this a *population flow graph*.

Definition 1.3.19. A *population-flow graph* (for a mutli-city SIR model) is a graph whose nodes are labeled by cities and whose edges $\text{City}_1 \rightarrow \text{City}_2$ are labeled by 3×3 real diagonal matrices $\texttt{Flow}_{1\rightarrow2}$ of the following form:

$$\begin{bmatrix} r_S & 0 & 0 \\ 0 & r_I & 0 \\ 0 & 0 & r_R \end{bmatrix}.$$

Example 1.3.20. Let's take a minute to understand **??**. Here is an example of a network of cities, represented in a graph:
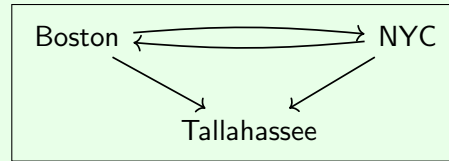
$$(1.31)$$

This map contains three cities, Boston, NYC, and Tallahassee. As we can see, Boston and NYC have restricted access to travellers from Tallahassee, but otherwise people can travel freely. Let's focus in on one of these ways to travel, say Boston → NYC. This is associated to a matrix

$$\texttt{Flow}_{\text{Boston}\rightarrow\text{NYC}} := \begin{bmatrix} r_S & 0 & 0 \\ 0 & r_I & 0 \\ 0 & 0 & r_R \end{bmatrix}.$$

per the definition of a population flow graph. Here's how to understand this matrix. If the current population of Boston (split into susceptible, infected, and removed) is

$$s = \begin{bmatrix} S \\ I \\ R \end{bmatrix}, \text{then}$$

$$\text{Flow}_{\text{Boston}\to\text{NYC}}\, s = \begin{bmatrix} r_S & 0 & 0 \\ 0 & r_I & 0 \\ 0 & 0 & r_R \end{bmatrix} \begin{bmatrix} S \\ I \\ R \end{bmatrix} = \begin{bmatrix} r_S S \\ r_I I \\ r_R R \end{bmatrix}$$

is the population that will leave Boston and arrive in NYC. Of course, this assumes that people do not become sick in transit, a temporary assumption that a more robust model would have to address.

Given a population flow graph, we can form a multi-city SIR model by wiring together the cities in a particular way. Namely, to every city we will first add sums to its inputs for every city it is flowing to and every that flows to it. That is, we will prepare each city like so:

(1.32)
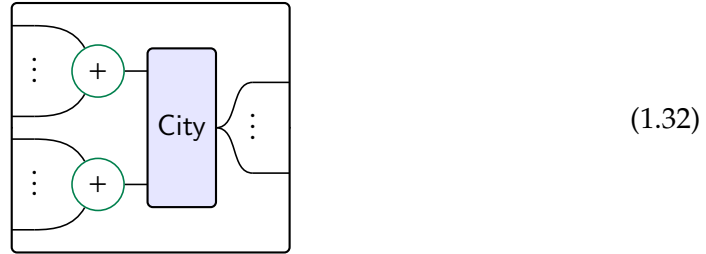
Specifically, we need to add together all the inflows from all other cities, and then record all the outflows to all other cities. We also need to copy the state enough times so that it can be passed to all other cities that our city flows to. So we need to add together inputs for all incoming edges in the population flow graph to the inflow port, and add together inputs for all outgoing edges in the population flow graph to the outflow port. And we also need to copy the output port to for all outgoing edges.

Example 1.3.21. For example, here is the preparation necessary for Boston in **??**:
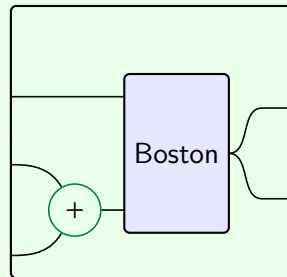
As you can see, there is only one incoming edge, and so the inflow input port doesn't need to anything to be added. But there are two outgoing edges, so we need to copy the output so they can be passed to NYC and Tallahassee and add together the two outflows into the outflow input port of Boston.

Exercise 1.3.22.   Prepare the cities of NYC and Tallahassee from **??** in the same way Boston was prepared in **??**.   ◊

Next, we wire together these prepared cities (from **??**). For each edge $City_1 \rightarrow City_2$ in our population flow graph, we will put the matrix $Flow_{City_1 \rightarrow City_2}$ on the wire leaving the prepared $City_1$ corresponding to the edge, then split the wire and plug one end into the corresponding outflow input port of $City_1$ and the corresponding inflow input port of $City_2$.

Example 1.3.23. Here is what it looks like to wire Boston to NYC along the edge Boston $\rightarrow$ NYC in the population flow graph **??**:

(1.33)
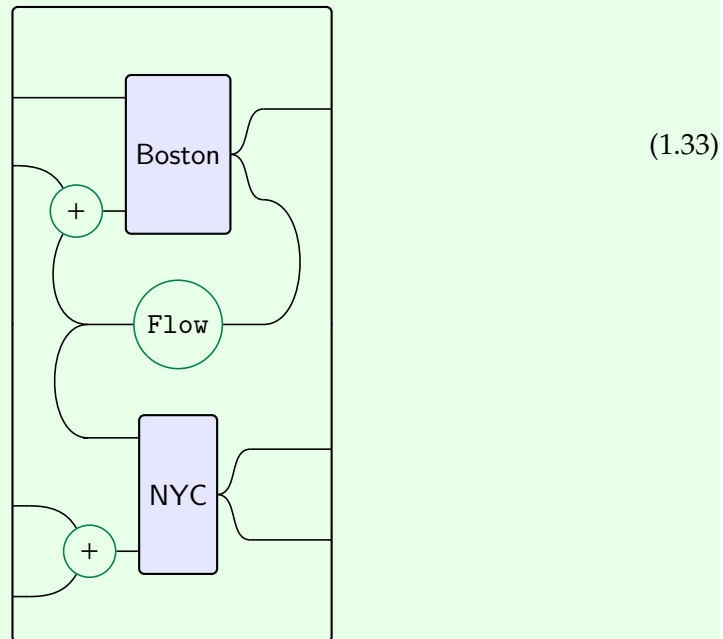
This wiring diagram says to take the population of Boston, take the proportion given by the flow rate $Flow_{Boston \rightarrow NYC}$, then set add this to the outflow parameter of Boston and the inflow parameter of NYC.

1.3.24   Wiring diagrams as lenses in categories of arities

We have been drawing a bunch of wiring diagrams so far, and we will continue to do so throughout the rest of the book. Its about time we explicitly described the rules one uses to draw these diagrams, and give a formal mathematical definition of them. The motto of this section is:

> *A wiring diagram is a lens in a free cartesian category — a category of* arities.

We'll begin by describing wiring diagrams and their category in informal terms. Then, we will see how diagrams relate to lenses in a particular category — which we

call the category of *arities* — and finally give a formal definition of the category of wiring diagrams.

Informal Definition 1.3.25. A *wiring diagram* is a diagram which consists of a number of *inner boxes*, each with some *input ports* and some *output ports*, that are wired together inside an *outer box*, which also has input and output ports. This gives four types of ports: inner (box) input (port), inner output, outer input, and outer output.

We can wire in the following ways:

1. Every outer output port is wired to exactly one inner output port.
2. Every inner input port is wired to exactly one inner output port or an outer input port.

The category of wiring diagrams has boxes as its objects and wiring diagrams as its morphisms. Wiring diagrams are composed by filling the inner boxes with other wiring diagrams, and then erasing the middle layer of boxes.

Jaz: This diagram is from D. Spivak.

Boxes	Wiring Diagrams	Composition by nesting

 (1.34)
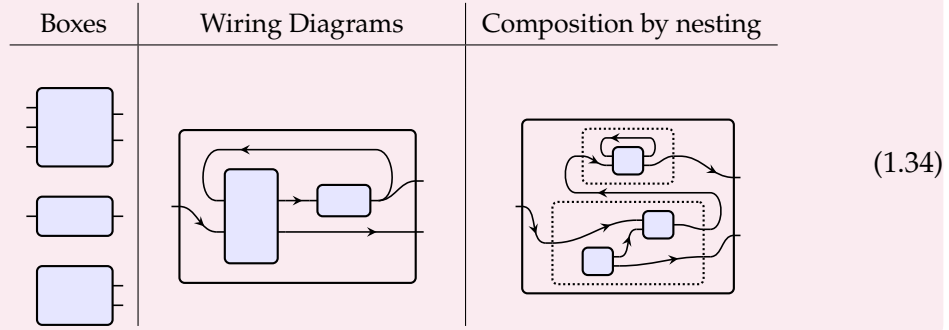
Wiring diagrams are designed to express the flow of variables through the system; how they are to be copied from one port to another, how they are to be shuffled about, and (though we haven't had need for this yet) how they are to be deleted or forgotton.

In order to capture this idea of copying, deleting, and shuffling around variables, we will work with the *category of arities* (and variations on it). The category of arities is extremely important since it captures precisely the algebra of copying, deleting, and shuffling around variables. In this section, we will interpret various sorts of wiring diagrams as lenses in categories of arities, which are the free cartesian categories.

Definition 1.3.26. The category **Arity** of arities is the free cartesian category generated by a single object X. That is, **Arity** constains an object X, called the *generic object*, and for any finite set I, there is an I-fold power X^I of X. The only maps are those that can be defined from the product structure by pairing and projection.

Explicitly, **Arity** is has:

- Objects $\{X^I \mid I \text{ a finite set}\}$.
- Maps $f^* : X^I \to X^J$ for any function $f : J \to I$.
- Composition defined by $g^* \circ f^* := (f \circ g)^*$ and id := id*.

The cartesian product in **Arity** is given, in terms of index sets, by the following familiar formula:

$$X^I \times X^J = X^{I+J}.$$

If you like opposite categories, this might clarify things a bit.

Proposition 1.3.27. **Arity** is isomorphic to the opposite of the category finite sets

$$\mathbf{Arity} \cong \mathbf{FinSet}^{\mathrm{op}}.$$

Now, X is just a formal object, so it doesn't have elements. But we can give a language for writing down the objects and arrows of **Arity** that makes it look like it does. Think of the elements of X^I as finite lists of variables $X^I = (x_i \mid i \in I)$ indexed by the set I. Then for any reindexing function $f : J \to I$, we can see f^* as telling us how J-variables are assigned I-variables. For example, consider the function $f : 3 \to 2$ given by $1 \mapsto 2$, $2 \mapsto 1$, and $3 \mapsto 2$

$$X^2 \longrightarrow X^3$$
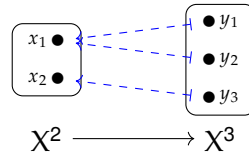
In other words, f says that the first slot of the resulting list will be filled by the second variable of the first, and the second slot will be filled by the first variable, and the third slot will be filled by the second variable.

Composition is of course just given by composing functions in the opposite direction. For example, given some $g : 4 \to 3$, we just compose to get our map $X^2 \to X^4$.

$$X^2 \longrightarrow X^3 \longrightarrow X^4$$
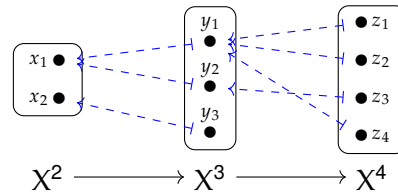
Exercise 1.3.28. Express the following morphisms in **Arity** in terms of lists of variables:

1. The terminal morphism $X^2 \to X^0$, given by the *initial* function $!' : 0 \to 2$ which includes empty set into the set with two elements (hint, there's *nothing* on one side).
2. The duplication morphism $!^* : X \to X^2$ given by $! : 2 \to 1$.
3. The swap morphisms $\mathrm{swap}^* : X^2 \to X^2$ given by $\mathrm{swap} : 2 \to 2$ defined by $0 \mapsto 1$ and $1 \mapsto 0$.
4. What map corresponds to the map $1 : 1 \to 2$ picking out $1 \in 2 = \{1, 2\}$? What about $2 : 1 \to 2$.
5. Convince yourself that *any* map $X^I \to X^J$ you can express with the universal property of products can be expressed by choosing an appropriate $f : J \to I$.

◊

Because **Arity** expresses the algebra of shuffling, copying, and deleting variables in the abstract, we can use it to define wiring diagrams. Recall from **??** the definition of lens in an arbitrary cartesian category.

Definition 1.3.29. The category **WD** of wiring diagrams is defined to be the category of lenses in the category of arities **Arity**.

$$\mathbf{WD} := \mathbf{Lens}_{\mathbf{Arity}}.$$

We consider **WD** as a monoidal category in the same way we consider **Lens**$_{\mathbf{Arity}}$ as a monoidal category.

This definition shows us that the wiring diagrams we have been using are precisely the lenses you can express if you only copy, delete, and shuffle around your variables. Here's how we interpret a lens $\begin{pmatrix} w^{\sharp*} \\ w^* \end{pmatrix} : \begin{pmatrix} X^{A^-} \\ X^{A^+} \end{pmatrix} \leftrightarrows \begin{pmatrix} X^{B^-} \\ X^{B^+} \end{pmatrix}$ in **Arity** as a wiring diagram:

- First, we interpret the index set A^- as the set of input ports of the inner boxes, and the set A^+ as the set of output ports of the inner boxes. Similarly, we see B^- as the set of input ports of the outer box, and B^+ as the set of output ports of the outer box.
- Then we remember that $w^* : X^{A^+} \to X^{B^+}$ comes from a reindexing function $w : B^+ \to A^+$), which we interpret as selecting for each outer output port $p \in B^+$, the unique inner output port $w(p)$ it will be wired to.
- Finally, we note that $w^{\sharp*} : X^{A^+} \times X^{B^-} \to X^{A^-}$ comes from a function $w^{\sharp} : A^- \to A^+ + B^-$ (because $X^{A^+} \times X^{B^-} = X^{A^+ + B^-}$), and we interpret this as selecting for each inner input port $p \in A^-$ either the inner output port $w^{\sharp}(p) \in A^+$ or the outer input port $w^{\sharp}(p) \in B^-$ which p will be wired to.

On the other hand, we can read any wiring diagram as a lens in **Arity** in the following way:

(1.35)
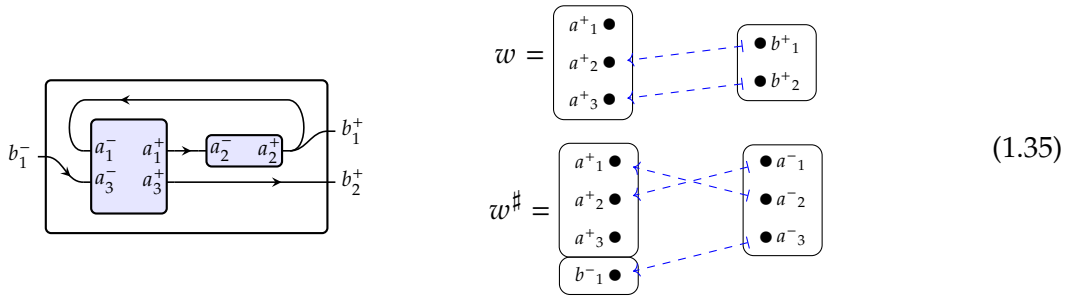
Exercise 1.3.30. Translate the following wiring diagrams into lenses in the category of arities, and vice versa:

1.
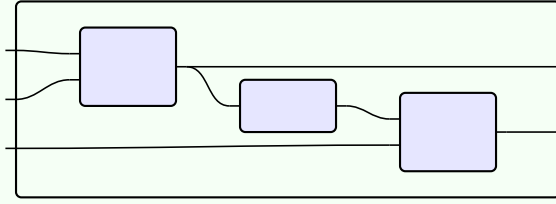
2. $\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} X^2 \times X^1 \times X^2 \\ X \times X \times X^2 \end{pmatrix} \leftrightarrows \begin{pmatrix} X^2 \\ X^1 \end{pmatrix}$

$\diamondsuit$
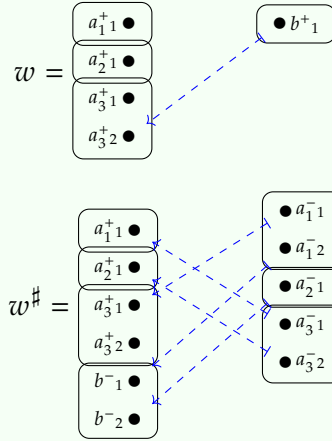
Ok, so the wiring diagrams correponds to the lenses in the category of arities. But do they compose in the same way? Composition of wiring diagrams is given by nesting: to compute $\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \, \mathring{,} \, \begin{pmatrix} u^\sharp \\ u \end{pmatrix}$, we fill in the inner box of $\begin{pmatrix} u^\sharp \\ u \end{pmatrix}$ with the outer box of $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$, and then remove this middle layer of boxes.

(1.36)
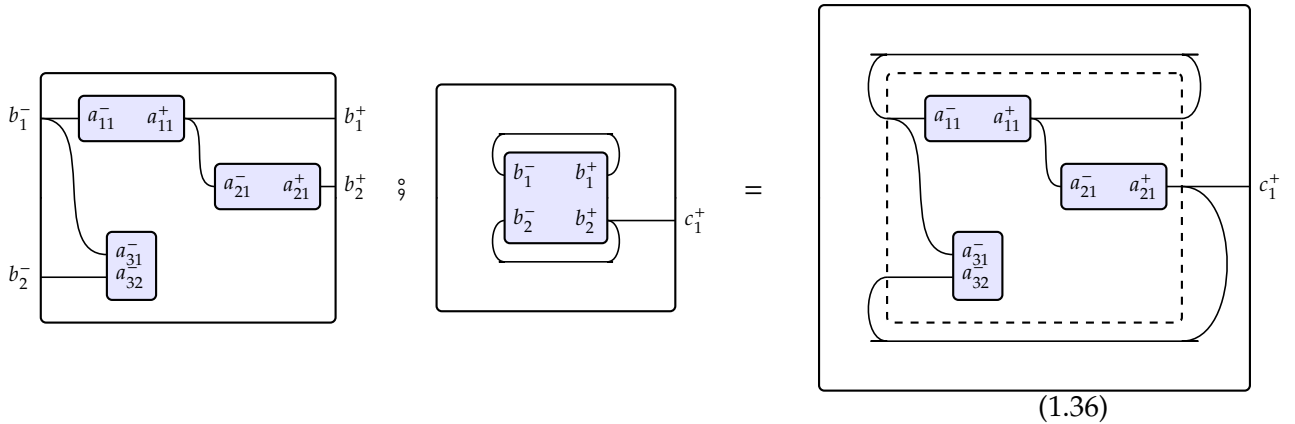
Let's say in prose how to compose two wiring diagrams. Then, we can check that this matches the formula given to us by lens composition in **Arity**.

- An outer output port is wired to a middle output port, and this middle output port is wired to an inner input port. So, to compose, we wire the outer output port to this inner output port.
- A inner input port is either wired to an inner input port or a middle input port. If it is wired to an inner input port, we leave it that way. Suppose that it was instead

wired to a middle input port. This middle input port is wired either to a middle output port or an outer input port. If it is wired to an outer input port, we then wire the inner input port to this outer input port. But if it was wired to a middle output port, we need to follow along to the inner output port that it is wired to; then we wire the inner input port to this inner output port.

Phew. After that block of text, I hope the mathematics will feel refreshingly crisp. Let's see what the lens composition looks like in **Arity**:
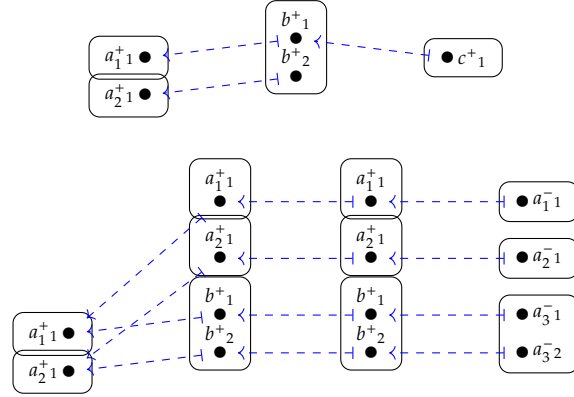
It's worth going through and seeing exactly how lens composition expresses the description we gave of nesting wiring diagrams above.

That **Arity** is the free cartesian category generated by a single object means that it satisfies a very useful universal property.

> **Proposition 1.3.31** (Universal property of **Arity**). For any cartesian category $\mathcal{C}$ and object $C \in \mathcal{C}$, there is a cartesian functor $\mathrm{ev}_C : \mathbf{Arity} \to \mathcal{C}$ which sends X to C. This functor is the unique such functor up to a unique natural isomorphism.

Proof Sketch. The functor ev_C can be defined by "just substitute C for X". Namely, we send

$$X^I \mapsto C^I$$

and for every function $f^* : X^I \to X^J$, we send it to $f^* : C^I \to C^J$ defined by the universal property of the product in $\mathcal{C}$. This is cartesian because $C^{I+J} \cong C^I \times C^J$ in any cartesian category. It is unique up to a unique natural isomorphism because X^I is the I-fold product of X, and so if $X \mapsto C$, then universal comparison maps between the image of X^I and C^I must be isomorphisms. □

We can think of the functor $\mathrm{ev}_C : \mathbf{Arity} \to \mathcal{C}$ as the functor which tells us how to interpret the abstract variables in **Arity** as variables of type C. For example, the functor $\mathrm{ev}_\mathbb{R} : \mathbf{Arity} \to \mathbf{Set}$ tells us how to interpret the abstract variables $(x_i \mid i \in I)$ in **Set** as variable real numbers $\{x_i \in \mathbb{R} \mid i \in I\}$. Under ev_C, the map of arities $(x_1, x_2, x_3 \mapsto x_2, x_2)$ gets sent to the actual map $C^3 \to C^2$ given by sending (c_1, c_2, c_3) to (c_2, c_2).

By the functoriality of the lens construction, this means that given an object $C \in \mathcal{C}$ of a cartesian category of "values that should be flowing on our wires", we can interpret a wiring diagram as a lens in $\mathcal{C}$! We record this observation in the following proposition.

Proposition 1.3.32. Let $C \in \mathcal{C}$ be an object of a cartesian category. Then there is a strong monoidal functor

$$\begin{pmatrix} \mathrm{ev}_C \\ \mathrm{ev}_C \end{pmatrix} : \mathbf{WD} \to \mathbf{Lens}_{\mathcal{C}}$$

which interprets a wiring diagram as a lens in $\mathcal{C}$ with values in C flowing along its wires.

Proof. This is just **??** (and **??**) applied to $\mathrm{ev}_C : \mathbf{Arity} \to \mathcal{C}$ from **??**. $\square$

This is how we can interpret a wiring diagram as a lens in whatever cartesian category we are working in. There is, however, a slight issue; in most of our previous examples, there have been many different types of signals flowing along the wires. We can fix this by using *typed arities*. We will keep track of what type of signal is flowing along each wire, and only allow ourselves to connect wires that carry the same type of signal.

Definition 1.3.33. Let $\mathcal{T}$ be a set, elements of which we call *types*. The category $\mathbf{Arity}_{\mathcal{T}}$ is the free cartesian category generated by objects X_τ for each type $\tau \in \mathcal{T}$. Explicitly, $\mathbf{Arity}_{\mathcal{T}}$ has:
- Objects $\prod_{i \in I} X_{\tau_i}$ for any finite set I and *typing function* $\tau_{(-)} : I \to \mathcal{T}$. We interpret $\tau_i \in \mathcal{T}$ as the type of element $i \in I$.
- Maps $f^* : \prod_{i \in I} X_{\tau_i} \to \prod_{j \in J} X_{\tau_j}$ for any function $f : I \to J$ which preserves the typing: $\tau_{fi} = \tau_i$.
- Composition is given by $g^* \circ f^* = (f \circ g)^*$ and the identity is given by $\mathrm{id} := \mathrm{id}^*$.

That is, $\mathbf{Arity}_{\mathcal{T}} \cong (\mathbf{Fin} \downarrow \mathcal{T})^{\mathrm{op}}$ is dual to the category $\mathbf{Fin} \downarrow \mathcal{T}$ of $\mathcal{T}$-*typed finite sets*, the slice category (a.k.a. comma category) of the inclusion $\mathbf{Fin} \hookrightarrow \mathbf{Set}$ over the set $\mathcal{T}$ of types.

Exercise 1.3.34. We blew through that isomorphism $\mathbf{Arity}_{\mathcal{T}} \cong (\mathbf{Fin} \downarrow \mathcal{T})^{\mathrm{op}}$ quickly, but its not entirely trivial. The category $\mathbf{Fin} \downarrow \mathcal{T}$ has objects functions $\tau : I \to \mathcal{T}$ where I is a finite set, and a morphism is a commuting triangle like this:

$$
\begin{array}{ccc}
I & \dashrightarrow & J \\
& \searrow_{\tau} \quad \swarrow_{\tau} & \\
& \mathcal{T} &
\end{array}
$$

Expand the isomorphism out in full and check that you understand it.   ◊

Note that **Arity** = **Arity**$_1$ is the special case where we have a single type. Just as we wrote the morphisms in **Arity** as $(x_1, x_2 \mapsto x_2, x_1, x_2)$, we can write the morphisms in **Arity**$_{\mathcal{T}}$ as

$$(x_1 : \tau_1, x_2 : \tau_2, x_3 : \tau_2 \mapsto x_2 : \tau_2, x_1 : \tau_1, x_2 : \tau_1)$$

where $\tau_1, \tau_2, \tau_3 \in \mathcal{T}$ are all (fixed, not variable) types.

We check that **Arity**$_{\mathcal{T}}$ as we defined it does indeed have the correct universal property.

Proposition 1.3.35. For any $\mathcal{T}$-indexed family of elements $C_{(-)} : \mathcal{T} \to \mathcal{C}$ in a cartesian category $\mathcal{C}$, there is a cartesian functor $ev_C : \mathbf{Arity}_{\mathcal{T}} \to \mathcal{C}$ sending X_τ to C_τ. The functor ev_C is the unique such functor up to a unique natural isomorphism.

Proof Sketch. Just like in **??**, we can take

$$ev_C \left(\prod_{i \in I} X_{\tau_i} \right) := \prod_{i \in I} C_{\tau_i}.$$

□

Exercise 1.3.36.   Complete the proof of **??**.   ◊

As before, we note that the map

$$(x_1 : \tau_1, x_2 : \tau_2, x_3 : \tau_2 \mapsto x_2 : \tau_2, x_1 : \tau_1, x_2 : \tau_1)$$

Spiz: redraw gets sent by ev_C to the function $C_{\tau_1} \times C_{\tau_2} \times C_{\tau_3} \to C_{\tau_2} \times C_{\tau_1} \times C_{\tau_2}$ which sends (c_1, c_2, c_3) to (c_2, c_1, c_2)

Corollary 1.3.37. For any function $f : \mathcal{T} \to \mathcal{T}'$, there is a change of type functor $ev_{X_f} : \mathbf{Arity}_{\mathcal{T}} \to \mathbf{Arity}_{\mathcal{T}}$.

Proof. We apply **??** to the family $X_{f(-)} : \mathcal{T} \to \mathbf{Arity}_{\mathcal{T}'}$ of objects of $\mathbf{Arity}_{\mathcal{T}'}$. That is, we send

$$\prod_{i \in I} X_{\tau_i} \mapsto \prod_{i \in I} X_{\tau(f(i))}.$$ □

We can now define the category of typed wiring diagrams to be the category of lenses in the category of typed arities.

Definition 1.3.38. For a set $\mathcal{T}$ of types, the category $\mathbf{WD}_{\mathcal{T}}$ of $\mathcal{T}$-*typed wiring diagrams* is

the category of lenses in the category of $\mathcal{T}$-typed arities:

$$\mathbf{WD}_{\mathcal{T}} := \mathbf{Lens}_{\mathcal{T}}.$$

As with the singly-typed case, we can interpret any typed wiring diagram as a lens in a cartesian category of our choosing.

Proposition 1.3.39. For any family $C_{(-)} : \mathcal{T} \to \mathcal{C}$ of objects in a cartesian category $\mathcal{C}$, indexed by a set $\mathcal{T}$ of types, there is a strong monoidal functor

$$\begin{pmatrix} \mathrm{ev}_C \\ \mathrm{ev}_C \end{pmatrix} : \mathbf{WD}_{\mathcal{T}} \to \mathbf{Lens}_{\mathcal{C}}$$

which interprets a typed wiring diagram as a lens in $\mathcal{C}$ with appropriately typed values flowing along its wires.

Remark 1.3.40. Because the action of ev_C is so simple, we will often just equate the typed wiring diagram with the lens it gives when interpreted in our category of choice.

Example 1.3.41. We can describe the wiring diagram

$$(1.37)$$
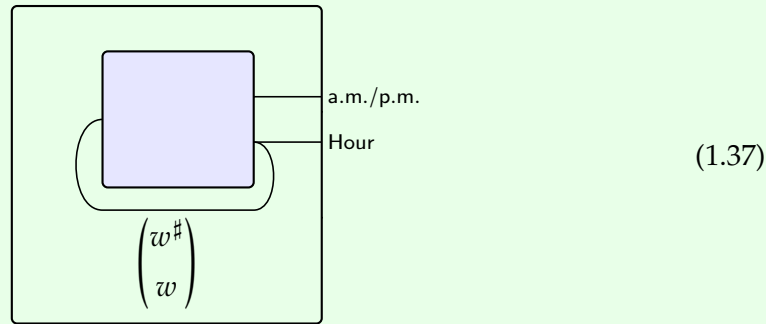
from **??** as a lens in a category of typed arities in the following way. We have two types: a.m./p.m. and Hour. So, $\mathcal{T} = \{\mathsf{a.m./p.m.}, \mathsf{Hour}\}$. Then

$$w = (t : \mathsf{Hour},\, m : \mathsf{a.m./p.m.} \mapsto t : \mathsf{Hour},\, m : \mathsf{a.m./p.m.})$$
$$w^{\sharp} = (t : \mathsf{Hour},\, m : \mathsf{a.m./p.m.} \mapsto t : \mathsf{Hour})$$

giving us a wiring diagram in $\mathbf{WD}_{\mathcal{T}}$. We can then interepret this as the lens from **??** as the image of this wiring diagram which interprets the types a.m./p.m. and Hour as the actual sets $\{\mathsf{a.m.}, \mathsf{p.m.}\}$ and $\{1, 2, \ldots, 12\}$.

1.3.42 Wiring diagrams with operations as lenses in Lawvere theories

The wiring diagrams we have described as lenses in categories of arities are pure wiring diagrams. But in **??**, we used a wiring diagram (**??**) with little green beads representing

multiplication by a constant scalar, and in **??** we used a wiring diagram with little green beads representing multiplication by a matrix (**??**). It is very useful to be able to perform operations on the exposed variables we are passing to parameters.

In this section, we will see that if we have an *algebraic theory* of the kinds of operations we want to perform on our variables while we wire them, we can describe wiring diagrams with green beads representing those adjustments as lenses in the *Lawvere theory* of that algebraic theory.

Algebraic theories are theories of operations that are subject to certain equational laws.

Informal Definition 1.3.43. A *algebraic theory* $\mathbb{T}$ consists:
- A set $\mathbb{T}_n$ of n-ary operations for each $n \in \mathbb{N}$.
- A set of *laws* setting some composites of operations equal to others.

Example 1.3.44. The algebraic theory of real vector spaces can be described like this:
- There is a binary operation $(-) + (-)$ of vector addition, and for every $r \in \mathbb{R}$ a unary operation $r \cdot (-)$ of scalar multiplication, and a nullary operation (a.k.a. constant) 0.
- These satisfy the laws that make $+$ and 0 into an abelian group with addition inverses given by $-1 \cdot (-)$, and which satisfy associativity and distributivity with regards to scalar multiplication.

$$(a + b) + c = a + (b + c) \qquad r \cdot (s \cdot a) = (rs) \cdot a$$
$$0 + a = a \qquad (r + s) \cdot a = r \cdot a + s \cdot a$$
$$a + b = b + a \qquad 1 \cdot a = a$$
$$a + (-1 \cdot a) = 0 \qquad 0 \cdot a = 0$$

We can use an algebraic theory to organize the sorts of operations we are willing or able to perform on the values flowing through the wires of our wiring diagrams.

Informal Definition 1.3.45. A wiring diagram with operations from an algebraic theory $\mathbb{T}$ is a wiring diagram where operations from the theory $\mathbb{T}$ can be drawn in little green beads on the wires.

Example 1.3.46. The wiring diagram **??** (reproduced below) is a wiring diagram in the algebraic theory of real vector spaces. The little green beads have scalar multiplications

drawn in them.
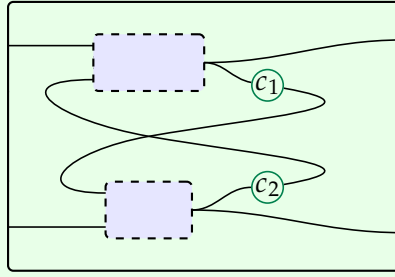
We want to make these informal definitions precise. Ultimately, we want to be able to say that "wiring diagrams with operations from $\mathbb{T}$ are lenses in such and such cartesian category". We can do this with the notion of Lawvere theory.

Lawvere introduced his theories in his 1963 thesis "Functorial Semantics of Algebraic Theories" [] as the invariant concepts of algebraic theories, freed from any particular presentation by symbols and their relations. In **??**, we presented the algebraic theory of real vector spaces in a particular way; but we could have done it differently, say by avoiding the vector 0 entirely and adding the law $(0 \cdot a) + b = b$. Lawvere wanted to avoid these petty differences in presentation. He focuses instead on the cartesian category freely containing the operations of the theory (satisfying their laws). This gives an invariant of the concept of real vector space that is independent of how that concept is presented.

A Lawvere theory is, in some sense, a category of arities "with extra maps". We think of these extra maps as coming from the operations of some theory.

Definition 1.3.47. A $\mathcal{T}$-sorted Lawvere theory $\mathcal{L}$ is a cartesian category equipped with a bijective-on-objects functor $\mathbf{Arity}_{\mathcal{T}} \hookrightarrow \mathcal{L}$.

If $\mathcal{T}$ has a single element, we refer to this as a *single sorted* Lawvere theory.

Where we wrote the objects of **Arity** as X^I to suggest the genericness of the generating object X, we will see that the objects of Lawvere theories are often A^I for some "actual" object A in some cartesian category.

Example 1.3.48. The single sorted Lawvere theory **Vect** of real vector spaces can be defined as follows:
- For every finite set I, it has an object $\mathbb{R}^I \in$ **Vect**.
- A map $f : \mathbb{R}^I \to \mathbb{R}^J$ is a linear map, or equivalently a $J \times I$ matrix.
- The cartesian product $\mathbb{R}^I \times \mathbb{R}^J$ is $\mathbb{R}^{I+J}$.

Since **Vect** is a cartesian category, it admits a functor $X \mapsto \mathbb{R}$ from **Arity**. By construction, this functor is bijective on objects; we just need to show that it is faithful. If $g^*, f^* : X^I \to X^J$ are such that $g^* = f^*$ as maps $\mathbb{R}^I \to \mathbb{R}^J$, then in particular $g^*(e_i) = f^*(e_i)$

for all standard basis vectors e_i defined by

$$e_i(j) := \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

But $g^*(e_i)(j) := e_i(g(j))$ and $f^*(e_i)(j) := e_i(f(j))$, so by varying i we can test that $g(j) = f(j)$ for all $j \in J$, and therefore that $g^* = f^*$ as maps $X^I \to X^J$.

How do we know that the extra maps in a Lawvere theory really do come from the operations of an algebraic theory? We show that the Lawvere theory satisfies a certain universal property: cartesian functors out of it correpond to *models* of the theory. If this is the case, we say that the Lawvere theory is *presented* by the algebraic theory.

Informal Definition 1.3.49. Let $\mathbb{T}$ be an algebraic theory. A *model* of $\mathbb{T}$ in a cartesian category $\mathcal{C}$ is an object $C \in \mathcal{C}$ together with maps $m(f) : C^n \to C$ for each n-ary operation $f \in \mathbb{T}_n$ such that the maps $m(f)$ satisfy the laws of the theory.

Definition 1.3.50. A *model* of a Lawvere theory $\mathcal{L}$ in a cartesian category $\mathcal{C}$ is a cartesian functor $M : \mathcal{L} \to \mathcal{C}$.

We say that a Lawvere theory is presented by an algebraic theory if they have the same models in any cartesian category. We can show that our Lawvere theory **Vect** of vector spaces is presented by the theory of vector spaces of **??**.

Proposition 1.3.51. Let $\mathcal{C}$ be a cartesian category. Then for every real vector space in $\mathcal{C}$, by which we mean an object $V \in \mathcal{C}$ with a binary addition $+ : V^2 \to V$, a unary scalar multiplication $r\cdot : V \to V$ for each $r \in \mathbb{R}$, and a nullary $0 : 1 \to V$ which satisfy the laws of a vector space, there is a cartesian functor $\hat{V} : \textbf{Vect} \to \mathcal{C}$ sending $\mathbb{R}$ to V. Moreover, this functor is unique up to a unique isomorphism among functors sending $\mathbb{R}$ to V.

Proof Sketch. We define the functor $\hat{V}$ by sending $\mathbb{R}^I$ to V^I, and sending the operations $+ : \mathbb{R}^2 \to \mathbb{R}, r\cdot : \mathbb{R} \to \mathbb{R}$, and $0 : \mathbb{R}^0 \to \mathbb{R}$ to the corresponding operations on V. Given a general linear map $f : \mathbb{R}^I \to \mathbb{R}^J$, f can be expressed as a composite of these operations; therefore, we can define $\hat{V}(f)$ to be the corresponding composite of the operations on V. □

Definition 1.3.52. Let $\mathcal{L}$ be a Lawvere theory. The category $\textbf{WD}^{\mathcal{L}}$ of wiring diagrams with operations from $\mathcal{L}$ is the category of lenses in $\mathcal{L}$:

$$\textbf{WD}_{\mathcal{L}} := \textbf{Lens}_{\mathcal{L}}.$$

Remark 1.3.53. The bijective-on-objects functor $\mathbf{Arity}_{\mathcal{T}} \to \mathcal{L}$ lets us interpret every $\mathcal{T}$-typed wiring diagram as a wiring diagram with operations from $\mathcal{L}$ by **??**.

In order to interpret a wiring diagram with operations from $\mathcal{L}$ as a lens in a cartesian category $\mathcal{C}$, we need a cartesian functor $\mathcal{L} \to \mathcal{C}$. These are precisely the models of the Lawvere theory. So, if our interpretation of the wires of our diagrams are models of our Lawvere theory $\mathcal{L}$, we can interpret diagrams with operations from $\mathcal{L}$.

Example 1.3.54. The wiring diagram **??** is a wiring diagram with operations from **Vect**, the theory of vector spaces. This is why we are able to put matrices in the beads.

1.4 Summary and Futher Reading

In this first chapter, we introduced deterministic and differential systems and saw how they could be composed using wiring diagrams. The trick is that both systems and wiring diagrams are examples of lenses — systems are lenses with a special domain, and wiring diagrams are lenses in free cartesian categories.

We will build on these ideas through the rest of the book. Most directly, in **??**, we will see how non-deterministic systems can be seen as a variant of lenses: *monadic lenses*.

The idea of composing dynamical systems — deterministic and differential — using lenses can be found in [VSL15]. Further exploration of this idea for differential systems can be found in [**Bakirtsiz-Vasilakopolou-Fleming:Compositional.Cyber.Physical.Systems.Modelling**].

Non-deterministic doctrines

So far, we have seen how deterministic systems of the discrete- and continuous-time variety can be wired together. But modelling a system deterministically can be a bit hubristic: it assumes we have taken account of all variables that act on the state of the system, so that we can know exactly what will happen next or exactly how the system is tending to change. Often we know that the way we've modeled state is incomplete, and so knowing the state in our model might not tell us exactly what will happen next.

As an example, consider a person typing out an email. We know that the output of this system over time will be a stream of ASCII characters, and we won't model the various sorts of inputs that might be affecting the person as they write the email. The particular email written will depend on the person's state, but this state is extraordinarily complex and modelling it to the point that we would know exactly which email they will write is nigh impossible.

So, instead, we could use what we know about how emails that this person writes tend to look to predict what the next character will be. This would give us a *stochastic* model of the email-writer system.

In this section, we will see a variety of non-deterministic (discrete-time) doctrines. The kind of non-deterministism — possibilistic, stochastic, etc. — will be encoded in a *commutative monad* (**??**).

2.1 Possibilistic systems

Suppose that we are observing a deterministic system S from the outside. We can choose what input $i \in \mathsf{In_S}$ to put into the system, and we observe from that what output $o \in \mathsf{Out_S}$ comes out as a result. Can we understand how the system works from knowing this alone? In other words, can we construct a new system S' just from knowing how inputs relate to outputs in S?

In full generality, the answer is of course "no"; if there was only one possible output, for example, we have no chance to understand what's going on inside the system. But if we do observe a bunch of different changing outputs, we can give it a shot.

As a first guess, we might try to model how an input $i \in \mathsf{In_S}$ changes the output $o \in \mathsf{Out_S}$ that we are seeing. That is, we might try and make $\mathsf{States'} = \mathsf{Out_S}$, and then define the new dynamics $\mathrm{update}_{S'}(o, i)$ be the new output S gives when fed input i while it is exposing output o. There's just one problem with this idea: we won't always get the same output when we feed i in to S while it's exposing o.

For example, consider the following transition system:

$$
\mathsf{S} :=
$$

(2.1)
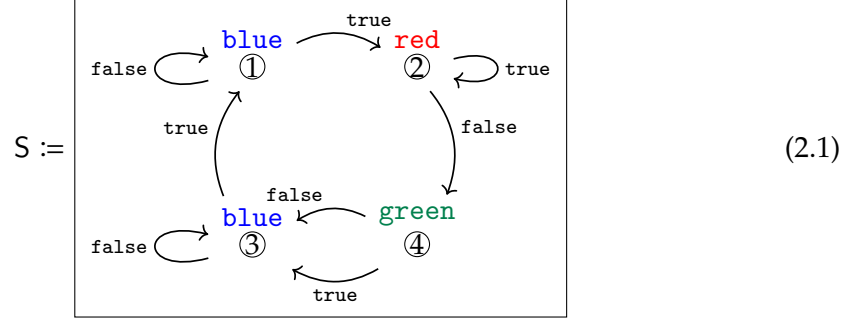
The inputs to this system are from the set $\mathsf{In_S} = \{\texttt{true}, \texttt{false}\}$, and the outputs are from the set $\mathsf{Out_S} = \{\texttt{red}, \texttt{blue}, \texttt{green}\}$. Suppose that we can only see what the system is outputing, and that it is outputing blue. If we feed the system false, we will see blue. But, if we feed the system true, what happens depends on whether the system was in state 1 or state 3; if we were in state 1, then we will see red, but if were were in state 3, we will see blue. So, the next output is not uniquely determined by the current output and current input — there are many possibilities. We are tempted to say that blue will transition to *either* red or blue in our model $\mathsf{S'}$ of the system S. That is, we want the update of $\mathsf{S'}$ to tell us what is *possible*, since we can't know just from the outputs of S what is *determined* to happen. We can do that by having the update of $\mathsf{S'}$ give us the set of possibilities:

$$
\mathrm{update}_{S'}(\texttt{blue}, \texttt{true}) = \{\texttt{blue}, \texttt{red}\}.
$$

In this section, we will see two dynamical system doctrines which, instead of telling us the next state, tell us which states are possible or which are probable. Both are examples of *non-deterministic* doctrines, since the current state doesn't determine precisely the next state.

Definition 2.1.1. A *possibilistic system* S, also written as

$$
\begin{pmatrix} \mathrm{update_S} \\ \mathrm{expose_S} \end{pmatrix} : \begin{pmatrix} \mathsf{States} \\ \mathsf{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathsf{In_S} \\ \mathsf{Out_S} \end{pmatrix},
$$

consists of:
- a set States of *states*;
- a set $\mathsf{Out_S}$ of *values for exposed variables*, or *outputs* for short;
- a set $\mathsf{In_S}$ of *parameter values*, or *inputs* for short;

- a function $\text{expose}_S : \text{States}_S \to \text{Out}_S$, the *exposed variable of state* or *expose* function, which takes a state to the output it yields; and
- a function $\text{update}_S : \text{States}_S \times \text{In}_S \to \text{PStates}_S$, where PStates_S is the set of subsets of States_S. This is the *dynamics* or *update* function which takes a state and a parameter and gives the set of possible next states.

Remark 2.1.2. While Definition 1.2.3 can be interepreted in any cartesian category because it only used maps and the cartesian product, **??** makes use of the *power set* operation P which sends a set to its set of subsets. This can't be interpreted in any cartesian category — we need something resembling P in order for it to make sense.

Example 2.1.3. A possibilistic automaton can be presented as a transition diagram as well. Consider, for example, the following diagram:

$$S' := \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (2.2)$$
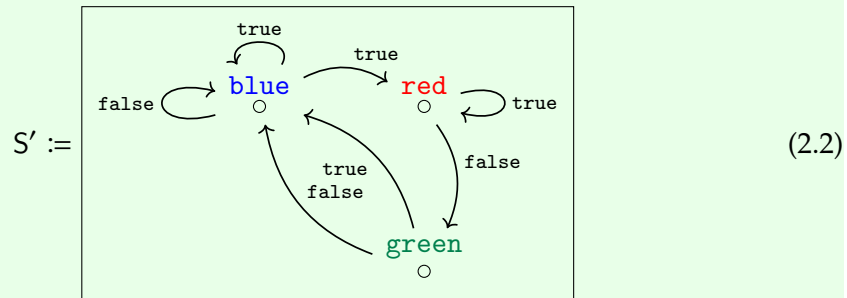
This system resembles system S of **??**, except that it has a single state for each output. We can tell that this transition diagram represents a possibilistic system because there are two arrows leaving `blue` both labeled `true`. Since the dynamics of a transition diagram are given by following the arrow labeled by the input along to a new state, we see that here we will end up at a set of states:

$$\text{update}_{S'}(\texttt{blue}, \texttt{true}) = \{\texttt{blue}, \texttt{red}\}.$$

Example 2.1.4. In Example 1.2.11, we saw that deterministic finite automata (DFAs) are examples of deterministic systems. There is another common notion in automata theory: *non-deterministic* finite automata (NFAs).

An NFA is a possibilistic system S with finitely many states whose output values are Booleans: $\text{Out}_S = \{\texttt{true}, \texttt{false}\}$. As with DFAs, the exposed variable $\text{expose}_S : \text{States}_S \to \{\texttt{true}, \texttt{false}\}$ tells us whether or not a state is an accept state.

Again, NFAs are question answering machines. But this time, since they are non-deterministic, we ask whether or not it is *possible* to accept a given sequence of inputs. Suppose we have a sequence of inputs $i_0, \dots, i_n$, and we start in a state s_0. Now, because an NFA is possibilistic, we don't have a "next state" s_1. Rather, we have a set of states $S_1 := \text{update}_S(s_0, i_0)$. Now, we need to interatively define the next evolution: S_2 should

be the set of states that are possible to get to from *any* state in S_1. Generally,

$$S_{j+1} := \{s' \mid s \in S_j, s' \in \text{update}_S(s, i_j)\} = \bigcup_{s \in S_j} \text{update}_S(s, i_j)$$

We then say that the machine accepts the input sequence if there is any accept state in S_n.

?? contains an answer to an interesting question: how do we iterate the behavior of a possibilistic system? For a deterministic system whose update has the signature $\text{update}_S : \text{States} \times \text{In}_S \to \text{States}$, we can compose to get

$$\text{States} \times \text{In}_S \times \text{In}_S \xrightarrow{\text{update}_S \times \text{In}_S} \text{States} \times \text{In}_S \xrightarrow{\text{update}_S} \text{States}$$

which sends $(s, (i_0, i_1))$ to $\text{update}_S(\text{update}_S(s, i_0), i_1)$. We can do this as many times as we like to apply an entire sequence of inputs to a state.

But for a possibilistic system, the update has signature $\text{update}_S : \text{States} \times \text{In}_S \to \text{PStates}$. Now we can't just compose, if we tried the trick above we would go from $\text{States} \times \text{In}_S \times \text{In}_S \to \text{PStates} \times \text{In}_S$, and we're stuck.

But from $\text{update}_S : \text{States} \times \text{In}_S \to \text{PStates}$ we can define a function $U : \text{PStates} \times \text{In}_S \to \text{PStates}$ by

$$U(S, i) = \{s' \mid s \in S, s' \in \text{update}_S(s, i)\} = \bigcup_{s \in S} \text{update}_S(s, i)$$

Then we can define the iterated action of the system to be the composite

$$\text{States} \times \text{In}_S \times \text{In}_S \xrightarrow{\text{update}_S} \text{PStates} \text{In}_S \xrightarrow{U} \text{PStates}.$$

This process of lifting a function $A \times B \to \text{PC}$ to a function $\text{PA} \times B \to \text{PC}$ is fundamental, and worthy of abstraction. This operation comes from the fact that P is a *commutative monad*.

Definition 2.1.5. Let $\mathcal{C}$ be a cartesian category. A *monad* (M, η) on $\mathcal{C}$ consists of:
- An assignment of an object MA to every object $A \in \mathcal{C}$.
- For every object $A \in \mathcal{A}$, a map $\eta_A : A \to MA$.
- For every map $f : A \to MB$, a *lift* $f^M : MA \to MA$.

This data is required to satisfy the following laws:
- (Unit) For any object A,
$$\eta_A^M = \text{id}_{MA}.$$

- (Identity) For any map $f : A \to MB$,
$$f^M \circ \eta_A = f.$$

- (Composition) For any $f : A \to MB$ and $g : B \to MC$,
$$g^M \circ f^M = (g^M \circ f)^M.$$

From this data, we note that we can extend M into a functor $M : \mathcal{C} \to \mathcal{C}$ by sending $f : A \to B$ to $Mf := (\eta_B \circ f)^M : MA \to MB$. Then $\eta : A \to MA$ is natural in A, and we get another natural transformation $\mu : MMA \to MA$ defined by lifting the identity: $\mu := \mathrm{id}^M$. In fact, a monad may be equivalently defined as a functor $M : \mathcal{C} \to \mathcal{C}$ with natural transformations $\eta : A \to MA$ and $\mu : M^2A \to MA$ for which the following diagrams commutes:

$$
\begin{array}{ccccc}
MA & \xrightarrow{\ \eta\ } & M^2A & \xleftarrow{\ M\eta\ } & MA \\
 & \searrow & \downarrow{\scriptstyle \mu} & \swarrow & \\
 & & MA & &
\end{array}
\qquad\qquad
\begin{array}{ccc}
M^3A & \xrightarrow{\ \mu\ } & M^2A \\
{\scriptstyle M\mu}\downarrow & & \downarrow{\scriptstyle \mu} \\
M^2A & \xrightarrow{\ \mu\ } & MA
\end{array}
\qquad (2.3)
$$

For $f : A \to MB$, we can recover $f^M : MA \to MB$ from this definition of the monad M as $MA \xrightarrow{\ Mf\ } M^2B \xrightarrow{\ \mu\ } MB$.

A monad M is said to be *commutative* if there is a natural transformation $\sigma : MA \times MB \to M(A \times B)$ for which the following diagrams commute:

-
$$
\begin{array}{ccc}
MA \times 1 & \xrightarrow{(MA\times\eta)\,\S\,\sigma_{A,1}} & M(A \times 1) \\
 & {\scriptstyle \pi_1}\searrow & \downarrow{\scriptstyle M\pi_1} \\
 & & MA
\end{array}
\qquad (2.4)
$$

-
$$
\begin{array}{ccc}
1 \times MA & \xrightarrow{(\eta\times MA)\,\S\,\sigma_{1,A}} & M(1 \times A) \\
 & {\scriptstyle \pi_2}\searrow & \downarrow{\scriptstyle M\pi_2} \\
 & & MA
\end{array}
\qquad (2.5)
$$

-
$$
\begin{array}{ccc}
MA \times MB \times MC & \xrightarrow{MA\times\sigma_{B,C}} & MA \times M(B \times C) \\
{\scriptstyle \sigma_{A,B}\times MC}\downarrow & & \downarrow{\scriptstyle \sigma_{A,B\times C}} \\
M(A \times B) \times MC & \xrightarrow{\ \sigma_{A\times B,C}\ } & M(A \times B \times C)
\end{array}
\qquad (2.6)
$$

-
$$
\begin{array}{ccc}
A \times B & \xrightarrow{\ \eta\times\eta\ } & MA \times MB \\
 & {\scriptstyle \eta}\searrow & \downarrow{\scriptstyle \sigma} \\
 & & M(A \times B)
\end{array}
\qquad (2.7)
$$

-
$$
\begin{array}{ccc}
M^2A \times M^2B & \xrightarrow{\ \sigma\,\S\,M\sigma\ } & M^2(A \times B) \\
{\scriptstyle \mu\times\mu}\downarrow & & \downarrow{\scriptstyle \mu} \\
MA \times MB & \xrightarrow{\ \sigma\ } & M(A \times B)
\end{array}
\qquad (2.8)
$$

Remark 2.1.6. If you are familiar with the programming language Haskell, you will likely be familiar with the notion of monad. What we have called η_A here (which is

traditional in the category theory literature) is called `return` or `pure`. What we have called f^M for $f : A \rightarrow MB$ would, in haskell, be called `lift` and be defined by

```
lift = swap (>>=)
```

What we have called μ is called `join`. A monad in haskell is commutative if the following two programs have the same results:

```
f :: Monad m => m a -> m b -> m (a, b)
f ma ma = do
            a <- ma
            b <- mb
            return (a, b)

g :: Monad m => m a -> m b -> m (a, b)
g ma ma = do
            b <- mb
            a <- ma
            return (a, b)
```

That is, a monad is commutative when its order of execution doesn't matter.

Proposition 2.1.7. The powerset P is a commutative monad on the category of sets, with the following data:
- $\eta : A \rightarrow PA$ sends $a \in A$ to the singleton set $\{a\}$.
- For $f : A \rightarrow PB$, $f^P : PA \rightarrow PB$ is defined by

$$f^P(X) = \bigcup_{a \in X} f(a).$$

- $\sigma_{A,B} : PA \times PB \rightarrow P(A \times B)$ is defind by

$$\sigma_{A,B}(X, Y) = \{(a, b) \mid a \in X, b \in Y\}.$$

Proof. We just need to check the laws.
- The function η_A^P takes a set $X \in PA$ and yields $\bigcup_{x \in X}\{x\}$, which is equal to X.
- Let $f : A \rightarrow PB$ be a function. Then $f^P(\{a\}) = \bigcup_{a' \in \{a\}} f(a') = f(a)$ for any element $a \in A$.
- Let $f : A \rightarrow PB$ and $g : B \rightarrow PC$. For $X \in PA$, we have

$$g^P \circ f^P(X) = \bigcup_{b \in f^P(X)} g(b)$$

$$= \bigcup_{b \in \bigcup_{a \in X} f(a)} g(b)$$

$$= \bigcup_{a \in X} \bigcup_{b \in f(a)} g(b)$$

$$= (g^P \circ f)^P.$$

It remains to show that the powerset monad is commutative. We note that P acts as a functor on $f : A \to B$ by

$$Pf(X) = (\eta_B \circ f)^P(X) = \bigcup_{a \in X}\{f(a)\} = f[X].$$

sending a subset of A to its image in B. We also note that $\mu : P^2 A \to PA$ defined by $\mu = \mathrm{id}^P_{PA}$ sends a set S of subsets of S to its union $\bigcup s \in Ss$.

- (??) Beginning with $(X, *) \in PA \times 1$ (taking $1 \cong \{*\}$), we need to show that $P\pi_1 \circ \sigma_{A,1}(X, \{*\}) = X$. Now, $\sigma_{A,1}(X, \{*\}) = \{(a, b) \mid a \in X, b \in \{*\}\}$; since there is just one $b \in \{*\}$, every $a \in X$ is paired with some b, so projecting out the first component gives us all of X.
- (??) This is the same as the above, but on the other side.
- (??) If we have $(X, Y, Z) \in PA \times PB \times PC$, both sides of this diagram will give us $\{(a, b, c) \mid a \in X, b \in Y, c \in Z\}$.
- (??) For $(a, b) \in A \times B$, we have $\eta(a, b) = \{a, b\}$, and $\sigma(\eta(a), \eta(b)) = \{(x, y) \mid x \in \{a\}, y \in \{b\}\}$.
- (??) Let S be a set of subsets of A and T a set of subsets of B. The bottom path gives us

$$\sigma(\mu(S), \mu(T)) = \left\{(x, y) \,\middle|\, x \in \bigcup_{s \in S} s, \ y \in \bigcup_{t \in T} t\right\}$$

while taking the top path, we first get $\sigma(S, T) = \{(s, t) \mid s \in S, t \in T\}$ and then $M\sigma$ of that to get

$$\sigma\left[\{(s, t) \mid s \in S, t \in T\}\right] = \{\{(x, y) \mid x \in s, y \in t\} \mid s \in S, t \in T\}.$$

Finally, we take the union over this to get

$$\mu(P\sigma(\sigma(S, T))) = \bigcup_{s \in S, t \in T} \{(x, y) \mid x \in s, y \in t\}.$$

These two paths are easily seen to give the same result.

$\square$

Using the commutative monad structure of P, we can see that $U : PStates_S \times Ins_S \to PStates_S$ is the composite

$$PStates_S \times Ins_S \xrightarrow{id \times \eta} PStates_S \times PIns_S \xrightarrow{\sigma} P(States_S \times Ins_S) \xrightarrow{update^P_S} PStates_S.$$

This lets us iteratively apply the update function to a starting state or set of states.

It also lets us get the exposed variable out at the end. If we've been iteratively running a possibilistic system , then we won't know which state we are in but instead have a set $S \in \mathrm{PState_S}$ of states we could possibly be in. Because of this, we can't directly apply $\mathrm{expose_S} : \mathrm{State_S} \to \mathrm{Out_S}$, since it takes in a single state. But the monad structure of P gives us a function $\mathrm{Pexpose_S} : \mathrm{PState_S} \to \mathrm{POut_S}$. Applying this to our current set of possible states gives us a set of possible outputs, which is the best we could hope to know.

Do Notation   If we have a function $f : X \to Y$, we can think of this as mapping x in X to $f(x)$ in Y using "generalized elements" (see **??**). The *do notation* extends this way of writing morphisms in a cartesian category to include the action of a commutative monad M. The do notation is based on this simple equation for $f : X \to MY$:

$$\boxed{\begin{array}{l} \mathbf{do} \\ \quad x \leftarrow m \\ \quad f(x) \end{array}} := f^M(m) \tag{2.9}$$

where m is an element of MX and $f : X \to MY$. For $M = $ D, we can understand the do notation in this way: m is a subset of X, $f^M(m)$ is the subset $\{f(x) \in Y \mid x \in m\}$. We see this reflected in the do notation; we can read it as saying "get an element x from m, and then apply $f(x)$ to it; join together all the results." As we see more monads, we will see that a similar story can be told about them using the do notation.

There are a few rules for do notation which correspond to the laws for a monad. We can discover them by using **??** to expand out a few terms. First of all, since $\eta^M = \mathrm{id}_{MX}$, if m is an element of MX, then

$$\boxed{\begin{array}{l} \mathbf{do} \\ \quad x \leftarrow m \\ \quad \eta(x) \end{array}} = m$$

Next, since $\eta \,\mathring{,}\, f^M = f$, we find that

$$\boxed{\begin{array}{l} \mathbf{do} \\ \quad x' \leftarrow \eta(x) \\ \quad f(x') \end{array}} = f(x)$$

Finally, since $f^M \,\mathring{,}\, g^M = (f \,\mathring{,}\, g^M)^M$, we find that

$$\boxed{\begin{array}{l} \mathbf{do} \\ \quad y \leftarrow \boxed{\begin{array}{l} \mathbf{do} \\ \quad x \leftarrow m \\ \quad f(x) \end{array}} \\ \quad g(y) \end{array}} = \boxed{\begin{array}{l} \mathbf{do} \\ \quad x \leftarrow m \\ \quad \boxed{\begin{array}{l} \mathbf{do} \\ \quad y \leftarrow f(x) \\ \quad g(y) \end{array}} \end{array}}$$

Because these two expressions with nested do's are equal, we can simplify our notation by writing them as:

$$\boxed{\begin{aligned} &\textbf{do} \\ &\quad x \leftarrow m \\ &\quad y \leftarrow f(x) \\ &\quad g(y) \end{aligned}}$$

So far, we haven't used any pairs (x, y) in our do notation. To use pairs, we need our monad to be commutative. We can write down two expressions, assuming m_1 is an element of MX and m_2 is an element of MY. A monad is commutative precisely when these two expressions are equal:

$$\boxed{\begin{aligned} &\textbf{do} \\ &\quad x \leftarrow m_1 \\ &\quad y \leftarrow m_2 \\ &\quad \eta(x, y) \end{aligned}} = \boxed{\begin{aligned} &\textbf{do} \\ &\quad y \leftarrow m_2 \\ &\quad x \leftarrow m_1 \\ &\quad \eta(x, y) \end{aligned}}$$

When they are both equal, they are $\sigma(m_1, m_2)$, where $\sigma : MX \times MY \to M(X \times Y)$ is from the definition of a commutative monad. This lets us describe morphisms quite nicely. For example, given $f : X \to MY, g : Z \to MW$, and $h : Y \times W \to MQ$, we may define

$$\boxed{\begin{aligned} &\textbf{do} \\ &\quad y \leftarrow f(x) \\ &\quad w \leftarrow g(z) \\ &\quad h(y, w) \end{aligned}}$$

which desugars to the composite

$$X \times Z \xrightarrow{f \times g} MY \times MW \xrightarrow{\sigma} M(Y \times W) \xrightarrow{h^M} MQ.$$

In particular, to iterate a system S with update $\text{update}_S : \text{State}_S \times \text{In}_S \to D\text{State}_S$, we can define

$$U(S, i) := \boxed{\begin{aligned} &\textbf{do} \\ &\quad s \leftarrow S \\ &\quad \text{update}_S(s, i) \end{aligned}}$$

2.2 Stochastic systems

Possibility is not the only kind of non-determinism. When studying how things change in the world, we often notice that we can only predict how likely some change will be, and not precisely which change will occur. If instead of asking whether a change is possible, we ask how *probable* it is, we arrive at a notion of probabilistic or *stochastic* system.

The notion of a stochastic system is based on the idea that there should be a probability of a given change occuring, conditioned upon the current state. A useful way

to formulate the notion of conditional probability is the notion of *stochastic map*. A stochastic map from A to B is a function which takes an $a \in A$ and yields a probability distribution $p(- \mid a)$ on elements of B which we think of as likelyhoods conditioned on a. We can make this more precise using the notion of monad.

Definition 2.2.1. For a set A, the set DA is the set of finitely supported probability distributions on A. A probability distribution on A is a function $p : A \to [0, 1]$ which takes non-zero values at only finitely many elements of A, and for which

$$\sum_{a \in A} p(a) = 1.$$

This sum makes sense because only finitely many elements of A give non-zero $p(a)$.

The elements of DA can be identified with *formal convex combinations* of elements of A. A formal convex combination

$$\sum_{a \in X} \lambda_a a$$

of elements of A consists of a finite and inhabited[a] subset $X \subseteq A$ of elements together with a function $\lambda_{(-)} : X \to (0, 1]$ assigning each $a \in X$ a coefficient λ_a such that $\sum_{a \in X} \lambda_a = 1$.

$$DA = \left\{ \sum_{a \in X} \lambda_a a \;\middle|\; X \subseteq A, \ X \text{ finite and inhabited}, \ \lambda_{(-)} : X \to (0, 1], \ \sum_{a \in X} \lambda_a = 1 \right\}.$$

[a]That is, there is some $a \in X$.

Example 2.2.2. Let's see what DA looks like for a few different sets A:

1. If $A = \{a\}$ has a single element, then there is only one inhabited subset $X \subseteq A$ (namely $X = A$) and since the coefficients of any convex linear combination must sum to 1, the coefficient of the single element must be 1. So $D\{a\} = \{1 \cdot a\}$ contains a single element.

2. If $A = \{a, b\}$, things get more interesting. Now there are three possible subsets X: $\{a\}$, $\{b\}$, and $\{a, b\}$. A convex combination with a single element must have coefficient 1, so we at least have the convex combinations $1 \cdot a$ and $1 \cdot b$. But for the set $\{a, b\}$, we have the convex combination $\lambda_a a + \lambda_b b$ where $\lambda_a + \lambda_b = 1$ and $\lambda_a, \lambda_b > 0$. If we make the association of $1 \cdot a$ with $1 \cdot a + 0 \cdot b$, and similarly for $1 \cdot b$, then we can see that

$$D\{a, b\} = \{\lambda a + (1 - \lambda)b \mid \lambda \in [0, 1]\}$$

which is bijective with the closed interval $[0, 1]$.

3. In general, if A is a finite set with n elements, then DA can be identified with the *standard n-simplex*, that is, the set of solutions to the equation $\sum_{i=1}^{n} \lambda_i = 1$ for

$\lambda_i \in [0, 1]$.

$$Dn \cong \{(\lambda_1, \ldots, \lambda_n) \in [0, 1]^n \mid \sum_{i=1}^{n} \lambda_i = 1\}.$$

Definition 2.2.3. A *stochastic map* from a set A to a set B is a function $f : A \to DB$, assigning each $a \in A$ to a probability distribution $f(a)$ on B.

   If the sets A and B are finite, then we can write a stochastic map $f : A \to DB$ as a *stochastic matrix*. This is an $B \times A$ matrix whose ba-entry is $f(a)(b)$. Any matrix of positive entries where every column sums to 1 arises as the stochastic matrix of a stochastic map.

   We think of a stochastic map $f : A \to DB$ as giving a bunch of conditional probabilities

$$p(b \mid a) := f(a)(b).$$

Example 2.2.4. If I see someone enter the office soaking wet, it is likely to have been raining. If they are dry, it may be less likely that it was raining; but, if they have an umbrella, then they might be dry but it is still more likely that it was raining. We can express these various conditional probabilities as a stochastic function

$$\{\mathtt{wet}, \mathtt{dry}\} \times \{\mathtt{umbrella}, \mathtt{no-umbrella}\} \to D\{\mathtt{raining}, \mathtt{not-raining}\}.$$

We can describe this stochastic function in full by giving its stochastic matrix:

	(wet,umbrella)	(wet,no-umbrella)	(dry,umbrella)	(dry,no-umbrella)
raining	.9	.9	.5	.3
not-raining	.1	.1	.5	.7

   A stochastic system is a system whose dynamics is given by a stochastic map.

Definition 2.2.5. A *stochastic system* S, also written as

$$\begin{pmatrix} \mathrm{update}_S \\ \mathrm{expose}_S \end{pmatrix} : \begin{pmatrix} \mathrm{States} \\ \mathrm{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathrm{In}_S \\ \mathrm{Out}_S \end{pmatrix},$$

consists of:
  - a set States of *states*;
  - a set Out_S of *values for exposed variables*, or *outputs* for short;
  - a set In_S of *parameter values*, or *inputs* for short;
  - a function $\mathrm{expose}_S : \mathrm{States} \to \mathrm{Out}_S$, the *exposed variable of state* or *expose* function, which takes a state to the output it yields; and

- a function $\mathrm{update}_S : \mathrm{State}_S \times \mathrm{In}_S \to \mathrm{PState}_S$, where DState_S is the set of subsets of State_S. This is the *dynamics* or *update* function which takes a state and a parameter and gives the set of possible next states.

Remark 2.2.6. A stochastic system is often called a *Markov process*.

Example 2.2.7. A simple, but entertaining example of a stochastic system is a text generator. Suppose we have a big pile of text — say, the speeches of a famous person — and we want to generate some text that looks like it was written by the same author. There are many sophisticated ways to do this, but here's a very bone-headed approach. We will look at the text in 5-character length sequences, and ask: how likely is for a given character to follow this 5-character sequence.

For example, if our text is

$$\text{To be or not to be, that is the question.}$$

Then we can see that there is a 50% chance that " " and a 50% chance that "," follows the 5-character sequence "to be". Of course, such a small sample woulnd't give us very useful statistics, but if we use the combined works of Shakespeare, we might get a better sense of what is likely to occur next.

Now we build a stochastic system S which will generate text. We take State_S to be length 5 sequences of characters from our alphabet $\mathrm{Alphabet}$: $\mathrm{State}_S = \mathrm{Alphabet}^5$. We will expose the first character in the sequence: $\mathrm{Out}_S = \mathrm{Alphabet}$ and $\mathrm{expose}_S(s) = s_1$. We don't need any input to the system: $\mathrm{In}_S = \{*\}$. Now, $\mathrm{update}_S(s)$ will assign to a sequence (s_2, s_3, s_4, s_5, c) the probability that the character c follows the sequence $s = (s_1, s_2, s_3, s_4, s_5)$ in our sample text, and assign all other sequences the probability 0.

If we run our stochastic text generator over time, it will produce a stream of characters that have the statistical properties of our sample text. As simple minded as this approach is, it can produce some fun results:

HAMLET

Whose image even but now appear'd again!

HORATIO

From top to toe?

FRANCISCO

Bernardo, on their promise, as it is a course to any moment leisure, but to persever Than the cock, that this believe Those friend on Denmark Do not dull thy name with a defeated him yesternight.

Example 2.2.8. A *stochastic source process* is a stochastic system S with no inputs $\text{In}_S = 1$. Such a stochastic system would be boxed up like this:
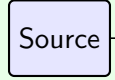
$$\boxed{\text{Source}}$$

These are means by which random streams of outputs can be generated. In **??**, we described a stochastic source process that produced Shakespearean writing (of a stunted sort). In his seminal paper "A mathematical theory of communication", Claude Shannon imagined communicators as stochastic source processes sending somewhat random language through various communication channels. This point of view is still used today to model communications that have some complicated structure which, not knowing how that structure is generated in particular, are best modeled as somewhat random processes.

Example 2.2.9. We can model a faulty wire as a stochastic system of the following sort:

$$\text{Bit} - \boxed{\text{FaultyWire}} - \text{Bit}$$
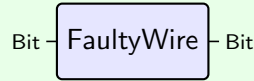
We will define FaultyWire as follows:

- A faulty wire will either have `good` contact, `partial` contact, or `missing` contact, and it will be carrying a `high` or `low` charge:

$$\text{State}_{\text{FaultyWire}} := \{\texttt{high}, \texttt{low}\} \times \{\texttt{good}, \texttt{partial}, \texttt{missing}\}.$$

- The faulty wire will take in either a `high` or `low`:

$$\text{In}_{\text{FaultyWire}} = \text{Out}_{\text{FaultyWire}} = \text{Bit} = \{\texttt{high}, \texttt{low}\}.$$

- The faulty wire exposes its current charge:

$$\text{expose}_{\text{FaultyWire}}(b, s) = b.$$

- The faulty wire will try and set its charge to the charge on the incoming wire, but if it is has bad contact, this won't succeed and it will have low charge. It's contact also has a small chance to decay.

$$\text{update}_{\text{FaultyWire}}((b, \texttt{good}), i) = .99(i, \texttt{good}) + .01(i, \texttt{partial}),$$

$$\text{update}_{\text{FaultyWire}}((b, \texttt{partial}), i) = .50(i, \texttt{partial}) + .49(\texttt{low}, \texttt{partial}) + .01(\texttt{low}, \texttt{missing}),$$

$$\text{update}_{\text{FaultyWire}}((b, \texttt{missing}), i) = (\texttt{low}, \texttt{no}).$$

When wiring up our systems, if we put a faulty wire in between, we will introduce the probability of the failure of this wire to communicate into the model.

Example 2.2.10. We can draw transition diagrams for stochastic systems, just like we do for deterministic and possibilistic systems. This time, we will label each transition with the probability that it occurs. We just have to make sure that the probability labels on all the outgoing transitions with the same input label on any state sum to 1.

For example, here is a stochastic system drawn as a transition diagram:
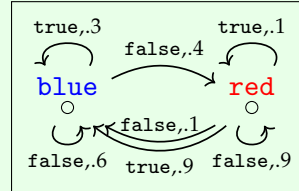
$$\begin{array}{c}
\text{true,.3} \qquad\qquad \text{true,.1} \\
\curvearrowleft \quad \text{false,.4} \quad \curvearrowleft \\
\textcolor{blue}{\text{blue}} \qquad\longrightarrow\qquad \textcolor{red}{\text{red}} \\
\circ \qquad\qquad\qquad \circ \\
\curvearrowright \overset{\text{false,.1}}{\longleftarrow} \curvearrowright \\
\text{false,.6} \quad \text{true,.9} \quad \text{false,.9}
\end{array} \qquad (2.10)$$

The set D of probability distributions is a commutative monad, like the powerset P monad.

Proposition 2.2.11. The assignment of a set A to its set DA of probability distributions is a commutative monad with the data:

- $\eta_A : A \to DA$ sends every element a to its *Dirac delta distribution* $\eta_A(a) = 1 \cdot a$ which assigns probability 1 to a and probability 0 to everything else. As a convex linear combination, it looks like this:

$$\eta_A(a) = \sum_{a' \in \{a\}} 1 \cdot a'$$

- Given a stochastic map $f : A \to DB$ sending $a \in A$ to $f(a) = \sum_{b \in Y_a} \rho_{ba} b$, we can *push forward* a probability distribution $p = \sum_{a \in X} \lambda_a a$ on A to a probability distribution

$$f^D(p) := \sum_{b \in \bigcup_{a \in X} Y_a} \left(\sum_{a \in X} \rho_{ba} \lambda_a \right) b = \sum_{a \in X} \sum_{b \in Y_a} \rho_{ba} \lambda_a b$$

  on B. In classical terms, this says that given conditional probabilities $p(b \mid a) := f(a)(b)$ and any prior distribution $p(a) := \lambda_a$, we can form a posterior distribution $p(b) := \sum_{a \in A} p(b \mid a) p(a)$.

- Given a probability distribution $\sum_{a \in X} \lambda_a a$ on A and $\sum_{b \in Y} \mu_b b$ on B, we can form their joint distribution

$$\sum_{(a,b) \in X \times Y} \lambda_a \mu_b (a, b)$$

  on $A \times B$. This gives us $\sigma : DA \times DB \to D(A \times B)$. In classical terms, this says that the probability of two independent events is the product of their probabilities: $p(a, b) = p(a) p(b)$.

Proof. We check the laws:

- If we push forward a distribution $p = \sum_{a \in X} \lambda_a a$ along $\eta_A : A \to DA$, we get

$$\eta_A^D(p) = \sum_{a \in X} \sum_{a' \in \{a\}} 1 \cdot \lambda_{a'} a' = \sum_{a \in X} \lambda_a a.$$

- For a stochastic map $f : A \to DB$, we aim to show that pushing forward the Dirac delta distribution $\eta_A(a)$ along f gives $f(a) = \sum_{b \in Y_a} \lambda_{ba} b$. The definition of push forward gives us

$$f^D(\eta_A(a)) = \sum_{a' \in \{a\}} \sum_{b \in Y_{a'}} \lambda_{ba} \cdot 1 \cdot b = \sum_{b \in Y_a} \lambda_{ba} b.$$

- Given stochastic functions $f : A \to DB$ and $g : B \to DC$, we need to show that $g^D(f^D(p)) = (g^D \circ f)^D(p)$. Let

$$p = \sum_{a \in X} \lambda_a,$$

$$f(a) = \sum_{b \in Y_a} \rho_{ba} b,$$

$$g(b) = \sum_{c \in Z_b} \gamma_{cb} c.$$

Then we see that

$$g^D(f(a)) = \sum_{c \in \bigcup_{b \in \bigcup_{a \in X} Y_a}} \gamma_{cb} \rho_{ba} c$$

so that, finally

$$g^D(f^D(p)) = g^D \left(\sum_{a \in X} \sum_{b \in Y_a} \rho_{ba} \lambda_a c \right)$$

$$= \sum_{a \in X} \sum_{b \in Y_a} \sum_{c \in Z_b} \gamma_{cb} \rho_{ba} \lambda_a c$$

$$= (g^D \circ f)^D(p).$$

Next, we check that the laws of a commutative monad hold. We note that for a function $f : A \to B$, the function $Df = (\eta_B \circ f)^D$ is defined by

$$Df \left(\sum_{a \in X} \lambda_a a \right) = \sum_{a \in X} \sum_{b \in \{f(a)\}} \lambda_a b = \sum_{a \in X} \lambda_a f(a).$$

Furthermore, $\mu : D^2 A \to DA$ sends a formal convex combination $\sum_i \lambda_i p_i$ of probability distributions to the *actual* convex combination of those probability distributions, namely the distribution

$$\mu \left(\sum_i \lambda_i p_i \right)(a) := \sum_i \lambda_i p_i(a).$$

- (**??**) The unit on $1 \cong \{*\}$ sends $*$ to the distribution $1 \cdot *$. So, $\sigma(p, 1) = \sum_{(a,*) \in X \times 1} \lambda_a \cdot 1 \cdot (a, *)$, and projecting out again gives us $p = \sum_{a \in X} \lambda_a a$.
- (**??**) The same, but on the other side.
- (**??**) Suppose that we have

$$p = \sum_{a \in X} p_a a,$$

$$q = \sum_{b \in Y} q_b b,$$

$$r = \sum_{c \in Z} r_c c.$$

The both paths of **??** give us the distribution

$$\sum_{(a,b,c) \in X \times Y \times Z} p_a q_b r_c (a, b, c).$$

- (**??**) This is asking whether $\delta_{(a,b)} = \delta_a \delta_b$ as distributions on $A \times B$, which they are.

- (??) Let $\sum_i \lambda_i p_i$ be an element of DDA, and similarly let $\sum_j \rho_j q_j$ be an element of DDB. Following the bottom path around, we get

$$\sigma\left(\mu\left(\sum_i \lambda_i p_i\right), \mu\left(\sum_j \rho_j q_j\right)\right)(a, b) = \left(\sum_i \lambda_i p_i(a)\right)\left(\sum_j \rho_j q_j(b)\right) = \sum_i \sum_j \lambda_i \rho_j p_i(a) q_j(b).$$

Meanwhile,

$$\sigma\left(\sum_i \lambda_i p_i, \sum_j \rho_j q_j\right) = \sum_i \sum_j \lambda_i \rho_j (p_i, q_j).$$

and taking Dσ of that gives

$$\sum_i \sum_j \lambda_i \rho_j p_i q_j$$

which means that finally

$$\mu\left(\text{D}\sigma\left(\sigma\left(\sum_i \lambda_i p_i, \sum_j \rho_j q_j\right)\right)\right)(a, b) = \sum_i \sum_j \rho_j p_i(a) q_j(b).$$

$\square$

Exercise 2.2.12. Let $f : \text{n} \to \text{D}\text{m}$ and $g : \text{m} \to \text{D}\text{k}$ be stochastic maps. Note that we can interpret f as an $m \times n$ stochastic matrix F, and similarly g as a $k \times m$ stochastic matrix G. Show that the stochastic map $g^{\text{D}} \circ f$ is associated to the stochastic matrix GF. ◊

Just as the commutative monad structure of P helped us iterate possibilistic systems and get the set of possible output values from them, so the commutative monad structure of D helps us iterate stochastic systems and get a probability distribution of likely output values from them.

Given a stochastic system S, we have $\text{update}_\text{S} : \text{States}_\text{S} \times \text{Ins}_\text{S} \to \text{DStates}_\text{S}$. From this, we can get a stochastic map:

$$\text{DStates}_\text{S} \times \text{Ins}_\text{S} \xrightarrow{\text{id} \times \eta} \text{DStates}_\text{S} \times \text{DIns}_\text{S} \xrightarrow{\sigma} \text{D}(\text{States}_\text{S} \times \text{Ins}_\text{S}) \xrightarrow{\text{update}_\text{S}^\text{D}} \text{DStates}_\text{S}$$

which will let us iterate. We can see that this sends a probability distribution p on states and an input i to the distribution

$$s \mapsto \sum_{s' \in \text{States}_\text{S}} p(s')\text{update}_\text{S}(s', i)(s).$$

2.3   Monadic doctrines and the Kleisli category

We have now seen two sorts of non-determinism expressed by commutative monads. To each of these we associated a doctrine:

- To the powerset monad P, we associated the doctrine of possibilistic systems. This is because a map $f : A \to PB$ is a *possibilistic map* — it assigns a set of *possible images* to each element $a \in A$.
- To the probability distribution monad D, we associated the doctrine of stochastic system. This is because a map $f : A \to DB$ is a stochastic map.

In general, for any commutative monad M we call a map of the form $f : A \to MB$ a *Kleisli map*. The structure of a monad on M lets us compose Kleisli maps, giving us the *Kleisli category* of the monad. The commutativity then makes the Kleisli category into a symmetric monoidal category.

Definition 2.3.1. Let $M : \mathcal{C} \to \mathcal{C}$ be a commutative monad on a cartesian category. The *Kleisli category* **Kl**(M) is defined as follows:

- The objects of **Kl**(M) are the same as those of $\mathcal{C}$.
- A map $f : A \rightsquigarrow B$ in **Kl**(M) is a map $f : A \to MB$ in $\mathcal{C}$.
- The identity $\mathrm{id}_A : A \rightsquigarrow A$ is $\eta_A : A \to MA$.
- For $f : A \rightsquigarrow B$ and $g : B \rightsquigarrow C$, their composite is $f \mathbin{\S} g^M : A \to MC$. In do notation, the Kleisli composite is given by

$$(f \mathbin{\S} g)(a) := \boxed{\begin{array}{l} \textbf{do} \\ \quad b \leftarrow f(a) \\ \quad g(b) \end{array}}.$$

Since $g^M = Mg \mathbin{\S} \mu$, the Kleisli composite may be equivalently defined as $f \mathbin{\S} Mg \mathbin{\S} \mu$. The Kleisli category of M becomes a symmetric monoidal structure with with the tensor $A \times B$ and 1. Note that although $A \times B$ is cartesian in $\mathcal{C}$, it will rarely be cartesian in **Kl**(M).

We can understand Kleisli composition a bit better if we introduce a graphical language for monads.[1] This will also help us later in **??** when we learn about *biKleisli* composition. We will draw an object of our category $X \in \mathcal{C}$ as a string:

and a map $f : X \to Y$ as a bead:

Composition is drawn by connecting strings, and the identity map on X is represented by the same string which represents X. We will draw our monad $M : \mathcal{C} \to \mathcal{C}$ as a red string:

We can draw the natural transformations $\eta : \mathrm{id}_{\mathcal{C}} \Rightarrow M$ and $\mu : M^2 \Rightarrow M$ as

and

---

[1] If you know of it, this is just the usual string diagram language for 2-categories.

respectively. The laws **??** can be written as:
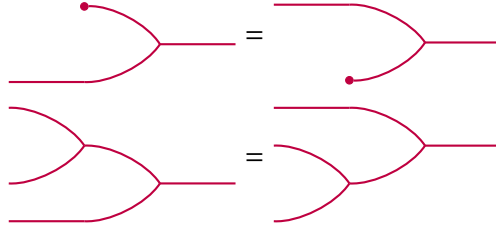
The map $Mf : MX \to MY$ on objects is written:

Note that functoriality is baked in to this string diagram notation; the following diagram could either be interpreted as $Mf \mathbin{\fatsemi} Mg$ or $M(f \mathbin{\fatsemi} g)$, which are equal by the functoriality of M:

The naturality of η and μ is also baked into this notation; it just means we can move them independently of the beads representing functions:
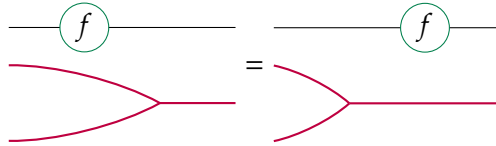
With these conventions in hand, we can now represent a Kleisli map $f : X \to MY$ as

The unit $\eta : X \to MX$ is written

The composition of Kleisli maps $f : X \to MY$ and $g : Y \to MZ$ is then given by
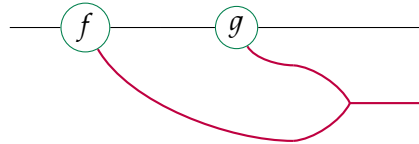
We can use these string diagrams to easily check that **Kl**(M) is actually a category. We use the monad laws **??**:
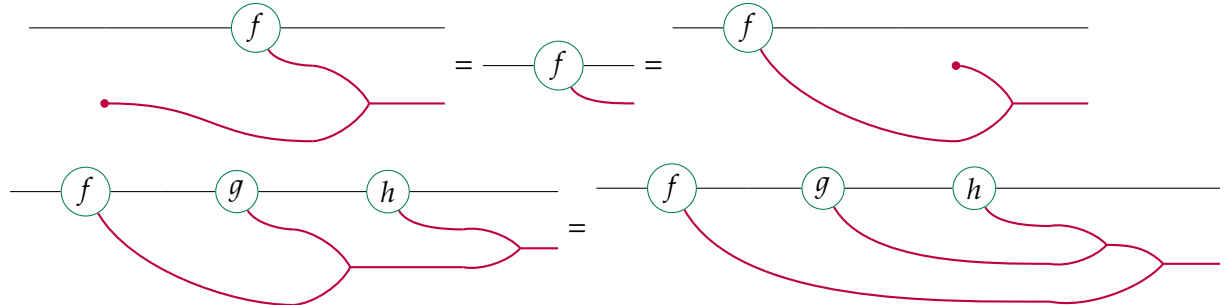
Example 2.3.2. The Kleisli category **Kl**(P) of the powerset monad P is the category of *multi-valued maps*. A Kleisli map $f : A \to PB$ assigns to each $a \in A$ a subset $f(a) \subseteq B$ of possible images of a. Given another Kleisli map $g : B \to PC$, their composite in the Kleisli category $g^P \circ f : A \to PC$ sends $a \in A$ to the union $\bigcup_{b \in f(a)} g(b)$. In other words, a possible image of $g \circ f$ is any possible image of g of any possible image of f.

Example 2.3.3. The Kleisli category **Kl**(D) of the probability monad D is the category of *stochastic maps*. A Kleisli map $f : A \to DB$ assigns to each $a \in A$ a probability distribution $f(a)$ on B. Given another Kleisli map $g : B \to DC$, their composite $g^D \circ f : A \to DC$ in the Kleisli category sends a to the probability distribution $c \mapsto \sum_{b \in B} f(a)(b) \cdot g(b)(c)$. That is, since c is the image of a under $g \circ f$ if there is a b which is the image of a under f and c is the image of b under g, the probability that c is the image of a is the probability of their being such a b.

Thinking of stochastic maps as conditional probabilities, where $f : A \to DB$ expresses the conditional probability $p(b \mid a) = f(a)(b)$, then we see that $p(c \mid a) = \sum_{b \in B} p(b \mid a)p(c \mid b)$ as we expect from conditional probabilities.

Now we encompass all our non-deterministic examples in a single definition.

Definition 2.3.4. Let $M : \mathcal{C} \to \mathcal{C}$ be a commutative monad. A (discrete-time) M-system S, also written as

$$\begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} : \begin{pmatrix} \mathsf{States} \\ \mathsf{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathsf{In_S} \\ \mathsf{Out_S} \end{pmatrix},$$

is a system whose dynamics is given by a Kleisli map for M. It consists of:
- an object States of *states*;
- an object $\mathsf{Out_S}$ of *values for exposed variables*, or *outputs* for short;
- an object $\mathsf{In_S}$ of *parameter values*, or *inputs* for short;
- a map $\mathsf{expose_S} : \mathsf{States} \to \mathsf{Out_S}$, the *exposed variable of state* or *expose* map, which takes a state to the output it yields; and
- a Kleisli map $\mathsf{update_S} : \mathsf{States} \times \mathsf{In_S} \to M\mathsf{States}$. This is the *dynamics* or *update* map which takes a state and a parameter and gives the next state in a non-deterministic way determined by M.

This will let us more swiftly describe new non-deterministic doctrines.

For example, suppose that our system is free to choose which state it transitions to next, but there's a catch. For any state s and input parameter i, there will be a cost $\mathsf{update_S}(s, i)(s') \in [0, \infty]$ associated to each other state s' — the cost of transitioning from s to s' given the parameter i. A cost of 0 means that this transition is free; a cost of ∞ means it is prohibitively expensive, or impossible.

Definition 2.3.5. We will define a monad Cost on the category of sets. We think of a Kleisli map $f : A \to \text{Cost}(B)$ as assiging the best-case cost of producing a $b \in B$ from a given $a \in A$. For practical reasons, we assume that only finitely many $b \in B$ are possible (that is, have finite cost) to produce from an $a \in A$.

- For a set A,

$$\text{Cost}(A) := \{c : A \to [0, \infty] \mid \{a \in A \mid c(a) < \infty\} \text{ is finite}\}$$

  is the set of cost functions $c : A \to [0, \infty]$ which assign finite values to only finitely many elements of A.

- For a set A, $\eta^{\text{Cost}} : A \to \text{Cost}(A)$ assumes that we can only produce what we have, but that if we already have it, it's free. Formally:

$$\eta^{\text{Cost}}(a)(a') := \begin{cases} 0 & \text{if } a = a' \\ \infty & \text{otherwise} \end{cases}$$

- For a map with cost $f : A \to \text{Cost}(B)$, we define $f^{\text{Cost}} : \text{Cost}(A) \to \text{Cost}(B)$ by

$$f^{\text{Cost}}(c)(b) := \min_{a \in A} c(a) + f(a)(b).$$

  That is, given costs on elements of A and conditional costs on elements of B given by f, the cost of an element of B is the cost of getting an $a \in A$ together with the cost of producing b from that a. So, the best case cost of such a b is the minimum over all $a \in A$ of the total cost of producing b from a. We note that the minimum is achieved because only finitely many of the costs are finite.

- Given sets A and B, the cost of having an element of A and an element of B is the sum of their costs.

$$\sigma(c, c')(a, b) := c(a) + c'(b).$$

Remark 2.3.6. We will prove that **??** does indeed give a commutative monad in the upcoming **??**.

Now we can quickly define our new sort of non-determinism.

Definition 2.3.7. A (discrete-time) system with costs is a Cost-system.

Example 2.3.8. Suppose we are trying to complete a project Proj that involves a number of steps. Let Steps be the set of steps involved. The state of our project at any given time is the set of steps we have completed so far: $\text{State}_{\text{Proj}} := \text{PSteps}$. Now, we may not want to show everyone exactly how our project is going, just that it has hit certain milestones. So we can let $\text{Out}_{\text{Proj}} := \text{Milestones}$ be our set of milestones and $\text{expose}_{\text{Proj}} : \text{State}_{\text{Proj}} \to \text{Out}_{\text{Proj}}$ send each project state to the most recent milestone

completed.

Now, in any project, there are some external conditions to be dealt with. Let $\mathsf{In}_{\mathsf{Proj}}$ = Externalities be the set of these externalities. We can assume that there is a cost associated to choosing a next step to take which depends not only on what steps have been completed so far but also on the current external conditions: that is, we can assume we have a function cost : $\mathsf{State}_{\mathsf{Proj}} \times \mathsf{In}_{\mathsf{Proj}} \to \mathsf{Cost(Steps)}$, and that $\mathrm{cost}(s, i)(x) = 0$ whenever $x \in s$ is a step we have already completed.[a] Given this, we can define the update of our project system as

$$\mathrm{update}_{\mathsf{Proj}}(s, i)(s') := \sum_{x \in s'} \mathrm{cost}(s, i)(x).$$

This tells us that the cost moving from having completed the steps s to having completed the steps s' given external conditions i is the sum of the cost of completing each step in s' which is not in s.

The crucial question we want to ask of this model is: how much will the project cost in the best case scenario, given a sequence of external conditions? That is, we will iterate the action of the system through the sequence of paramters starting at $\emptyset \in \mathsf{State}_{\mathsf{Proj}}$, and then ask the cost of $\mathsf{Steps} \in \mathsf{State}_{\mathsf{Proj}}$ at the end.

----

[a]Although one could imagine this instead as a "maintenance" cost of maintaining the completion of that step.

We took Cost to be the monad of best case costs. Let's show that there is also a monad $\mathrm{Cost}^{\max}$ of worst case costs. Everything will be the same, but instead of

$$f^{\mathrm{Cost}}(c)(b) := \min_{a \in A} c(a) + f(a)(b),$$

we will have

$$f^{\mathrm{Cost}^{\max}}(c)(b) := \max_{a \in A} c(a) + f(a)(b).$$

It is worth noting that this formula has a formal similarity to the following formula:

$$f^{R}(c)(b) := \sum_{a \in A} c(a) \cdot f(a)(b).$$

which resembles matrix multiplication. This is indeed the case; for any sort of (commutative) scalars, we get a monad that reproduces matrix arithmetic with those scalars. An appropriate set of scalars is called a *commutative rig*.

Definition 2.3.9. A *commutative rig* (for "**ri**ng without **n**e**g**atives"[a]) is a set R equipped with an abelian group structure $(R, +, 0)$ and a commutative monoid structure $(R, \cdot, 1)$ such that

$$a \cdot (b + c) = a \cdot b + a \cdot c$$

for all a, b, $c \in R$.

---

[a]Rigs are also sometimes referred to as "semirings".

Example 2.3.10. The following are important examples of rigs:
1. The natural numbers $\mathbb{N}$ with their usual addition and multiplication form a rig. Similarly, the non-negative rationals and reals form rigs with their usual addition and multiplication.
2. Any ring is a rig. In particular, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ are all rigs with their usual addition and multiplication.
3. The *tropical rigs* are rigs where "addition" is actually minimum or maximum, and "multiplication" is actually addition. In particular, the *rig of best-case costs* $[0, \infty]$ is a rig with min as its addition and $+$ as its multiplication. In this rig, distributivity looks like

$$a + \min\{b, c\} = \min\{a + b, a + c\},$$

and a linear combination looks like

$$\min_{i \in I} c_i + x_i.$$

The additive unit is ∞, and the multiplicative unit is 0.
Similarly, there is a *rig of worst-case costs* on $[0, \infty]$ with max as addition and $+$ as multiplication. This rig is remarkable in that its additive and multiplicative unit are the same; they are both 0.
4. In fact, any ordered commutative monoid $(M, +, 0, \leq)$ (where if $a \leq b$, then $c + a \leq c + b$) which admits joins $a \vee b$ (that is, least upper bounds) can be made into a commutative rig with addition given by $\vee$ and multiplication given by $+$.

Proposition 2.3.11. For any commutative rig R, there is a commutative monad $R \otimes - :$ **Set** $\to$ **Set** defined by
- $R \otimes X$ is the set of R-linear combinations of elements of X.
- $\eta : X \to R \otimes X$ sends x to the linear combination $\cdot x$.
- For $f : X \to R \otimes Y$, we have $f^R : R \otimes X \to R \otimes Y$ defined by

$$f^R \left(\sum_i r_i x_i \right) = \sum_i r_i f(x_i).$$

- For sets X and Y, we have $\sigma : (R \otimes X) \times (R \otimes Y) \to R \otimes (X \times Y)$ defined by

$$\sigma \left(\sum_i r_i x_i, \sum_j s_j y_j \right) = \sum_i \sum_j r_i s_j (x_i, y_j).$$

2.4 Adding rewards to non-deterministic systems

A common way to think of a discrete-time system is as a *decision process*. We think of the system A as an *agent* who needs to make a decision. The agent can choose an *action*, an element of In_A, and will then transition into a new state — although it may not know precisely which. We then ask the question: what is the best action for the agent to take in a given situation?

Clearly, an answer to this question will depend on what it means for one action to be better than another. The most common way to model this is by associating each action with a real number *reward*. The bigger the reward, the better the action (and negative rewards are harmful actions). If the agent is going to take a sequence of actions, we want the rewards to accumulate so that the total reward of a sequence of actions is the sum of each reward.

We can handle this accumulation of rewards, even in a deterministic system, with a commutative monad.

Definition 2.4.1. Let $(R, +, 0)$ be a commutative monoid (such as the real numbers). The *R-valued reward monad* or *monad of R-actions* is defined by the following data:
- To each set A, we associate the set $R \times A$ of pairs of a reward and an element of A.
- For each set A, we have $\eta_A : A \to R \times A$ given by yielding no reward: $\eta_A(a) = (0, a)$.
- For a function $f : A \to R \times B$ which yields an element of B and a reward, we give the function
$$f^R : R \times A \to R \times B$$
  defined by $f^R(r, a) = (r + \pi_1 f(a), \pi_2 f(a))$. This accumulates the reward $\pi_1 f(a)$ from applying f to a onto a current reward r
- For sets A and B, we have
$$\sigma : (R \times A) \times (R \times B) \to R \times (A \times B)$$
  given by $\sigma((r, a), (r', b)) = (r + r', (a, b))$. The reward for doing two actions simultaneously is the sum of their rewards.

We remark that this works not only in the category of sets, but in any cartesian category.

Exercise 2.4.2.   Show that the monad of R-valued rewards is really a commuativative monad. That is, show that the above data satisfies all each of the laws in **??**. Do you see where the commutativity comes into the mix?   ◊

We can then describe a system with reward as having an update $\mathsf{update}_S : \mathsf{States}_S \times \mathsf{In}_S \to R \times \mathsf{States}_S$ which sends the current state and action to the next state together with the reward for taking that action (in that state).

Definition 2.4.3. A deterministic system with R-valued rewards is an $(R \times -)$-system in the sense of **??**

We would really like to mix our rewards with non-determinism. In particular, when thinking of a system as an agent making decisions with imperfect information of its environment, we would like to use stochastic systems to model this lack of perfect information. The agent doesn't know exactly what will happen when it performs an action, but it has a good idea of what will *probably* happen.

The reward our agent gets should depend on what state the agent actually ends up in, and not just the action it takes. Therefore, we want to know the probability of transitioning to a next state *and* getting a certain reward. This has signature

$$\mathsf{State_S} \times \mathsf{In_S} \to \mathsf{D}(\mathbb{R} \times \mathsf{State_S}).$$

We will show that the assignment $A \mapsto \mathsf{D}(\mathbb{R} \times A)$ forms a commutative monad. We will show that more generally, if M is *any* commutative monad and R any commutative monoid, then $M(R \times -)$ is a commutative monad again. We say that we can "put the rewards R into the monad M". We can do this explicitly using the map $\lambda : R \times MA \to M(R \times A)$ defined to be the composite

$$\lambda := R \times MA \xrightarrow{\eta^M \times \mathrm{id}} MR \times MA \xrightarrow{\sigma^M} M(R \times A)$$

Intuitively, this takes a reward $r \in R$ and a non-deterministic $a \in MA$ and gives us the non-determinstic pair (r, a).

Proposition 2.4.4. Let M be a commutative monad and $(R, +, 0)$ a commutative monoid. Then the assignment $A \mapsto M(R \times A)$ is a commutative monad with the following structure:

- $\eta^{M(R \times -)} : A \to M(R \times A)$ is the composite $A \xrightarrow{\eta^R} R \times A \xrightarrow{\eta^M} M(R \times A)$.
- Given $f : A \to M(R \times B)$, we define $f^{M(R \times -)}$ to be the following composite:

$$M(R \times A) \xrightarrow{M(R \times f)} M(R \times M(R \times B)) \xrightarrow{M\lambda} MM(R \times R \times B)$$
$$\xrightarrow{\mu^M} M(R \times R \times B) \xrightarrow{M\mu^R} M(R \times B).$$

  Intuitively, this takes a non-deterministic pair (r, a) and, gets the non-deterministic pair $f(a) = (f_1(a), f_2(a))$, and then returns the non-deterministic pair $(r + f_1(a), f_2(a))$.
- Given sets A and B, we define $\sigma^{M(R \times -)} : M(R \times A) \to M(R \times B)$ to be the composite

$$M(R \times A) \xrightarrow{M} (R \times B) \xrightarrow{\sigma^M} M((R \times A) \times (R \times B)) \xrightarrow{M\sigma^R} M(R \times A \times B).$$

Proof. It is not obvious that this will satsify the monad laws, but it is a rather straight-forward check using the laws of M and $R \times -$. We will not prove this result explicitly. However, we will give a slick proof for experts.

A monad structure on $M(R \times A)$ arising via a distributive law such as $\lambda : R \times MA \to M(R \times A)$ is equivalent to a lift of the monad M to the category of $R \times -$ algebras — that is, the category of R-actions. But $M : \mathcal{C} \to \mathcal{C}$ is a commutative monad, and so in particular it is a symmetric monoidal functor; therefore, it preserves commutative monoids and their actions. For this reason, M extends to the category of $(R \times -)$-algebras, giving us the desired monad structure on $M(R \times -)$. This is again commutative as it is the composite of monoidal functors and so also monoidal.   □

Example 2.4.5. Let's see what this general theorem looks like in the case that $R = \mathbb{R}$ and $M = D$. In this case, $\lambda : \mathbb{R} \times DA \to D(\mathbb{R} \times A)$ sends the pair (r, p) of a reward and a probability distribution and yields the probability distribution $\delta_r p$. Let's see how this lets us iterate the dynamics of a $D(\mathbb{R} \times -)$-system S. We have $\text{update}_S : \text{State}_S \times \text{In}_S \to D(\mathbb{R} \times \text{State}_S)$, giving us a probabilities $\text{update}_S(s, i)(r, s')$ of transitioning from state s on action i into state s' and receiving reward r. To iterate this, we form the composite

$$D(\mathbb{R} \times \text{State}_S) \times \text{In}_S \xrightarrow{\sigma \circ (\text{id} \times \eta)} D(\mathbb{R} \times \text{State}_S \times \text{In}_S) \xrightarrow{\text{update}_S^{D(\mathbb{R} \times -)}} D(\mathbb{R} \times \text{State}_S)$$

which sends a pair (p, i) of a prior probability distribution on states and an an action to the distribution $(r, s) \mapsto \sum_{s' \in \text{State}_S} p(s') \text{update}_S(s', i)(r, s)$ which gives the probability of receiving the reward r and transitioning into the state s conditioned upon the prior p. To iterate, we can continually apply this map to many inputs; let's just do i and j. Then we end up with the distribution

$$(r, s) \mapsto \sum_{s'' \in \text{State}_S} \sum_{s' \in \text{State}_S} \sum_{r'' + r' = r} p(s) \cdot \text{update}_S(s'', i)(r'', s') \cdot \text{update}_S(s', j)(r', s)$$

which is the probability that we transition to s in two steps and receive a cumulative reward of r.

2.5   Changing the flavor of non-determinism: Monad maps

In the same way that 0 is a number, determinstic systems are non-deterministic systems, just with a trivial sort of non-determinism. Deterministic systems are M-systems for the *identity monad* $\text{id}(X) = X$. No matter what kind of non-determinism we are considering, we can always consider a deterministic system as a non-deterministic system, because we can take the update : State $\times$ In $\to$ State and post compose by η : State $\to M$State. This operation of turning a deterministic system into an M-system has a few nice properties; for example, if we iterate the system and then turn it into an M-system, we get the same result as if we had iterated it as an M-system.

In general, if we have a commutative monad morphism $M \to N$, then we can turn M-systems into N-systems.

Definition 2.5.1. A commutative monad map $\phi : M \to N$ is a natural transformation for which the following diagrams commute:

- $$
\begin{array}{ccc}
A & \xrightarrow{\eta^M} & MA \\
 & {\eta^N}\searrow & \downarrow{\phi} \\
 & & NA
\end{array}
\tag{2.11}
$$

- $$
\begin{array}{ccc}
M^2A & \xrightarrow{M\phi\,\mathring{}\,\phi} & N^2A \\
{\mu^M}\downarrow & & \downarrow{\mu^N} \\
MA & \xrightarrow{\phi} & NA
\end{array}
\tag{2.12}
$$

- $$
\begin{array}{ccc}
MA \times MB & \xrightarrow{\phi \times \phi} & NA \times NB \\
{\sigma^M}\downarrow & & \downarrow{\sigma^N} \\
M(A \times B) & \xrightarrow{\phi} & N(A \times B)
\end{array}
\tag{2.13}
$$

Proposition 2.5.2. There is a unique commutative monad map $\mathrm{id} \to M$, and it is given by η^M.

Proof. Let ϕ be such a map. Then condition **??** says precisely that $\phi = \eta^M$. So it just remains to check that η is a commutative monad map. Now, **??** commutes trivially, and **??** is in this case one of the diagrams defining M from **??**. Finally, **??** is in this case **??**. $\qquad\square$

We can then turn any deterministic system S into an M-system by defining its new update to be $\eta^M \circ \mathrm{update}_\mathsf{S}$. For possibilistic systems, this says that only the state that S actually transitions into is possible. For stochastic systems, this says that the probability that the system transitions into the state it actually transitions into is 1.

Intuitively, stochastic non-determinism is a refinement of possibilistic non-determinism: it not only tells us what is possible, but how likely it is. We can package this intuition into a commutative monad morphism $\phi : \mathsf{D} \to \mathsf{P}$.

Proposition 2.5.3. There is a commutative monad morphism $\phi : \mathsf{D} \to \mathsf{P}$ given by sending a probability distribution to the set of elements with non-zero probability:

$$
\phi(p) = \{a \in A \mid p(a) \neq 0\}.
$$

Proof. We check that this satisfies the laws.
- (**??**) The only element which δ_a assigns a non-zero probability is a.

- (??) Given a formal convex combination $\sum_i \lambda_i p_i$ of probability distributions $p_i \in \mathrm{D}A$, we see that

$$\phi \mu^{\mathrm{D}} \left(\sum_i \lambda_i p_i \right) = \{ a \in A \mid \sum_i \lambda_i p_i(a) \neq 0 \},$$

while

$$\mathrm{D}\phi \left(\sum_i \lambda_i p_i \right) = \sum_i \lambda_i \{ a \in A \mid p_i(a) \neq 0 \}$$

and so taking ϕ of that yields

$$\{ \{ a \in A \mid p_i(a) \neq 0 \} \mid \lambda_i \neq 0 \}$$

so, finally

$$\mu^{\mathrm{P}} \left(\phi \mathrm{D}\phi \left(\sum_i \lambda_i p_i \right) \right) = \bigcup_{\lambda_i \neq 0} \{ a \in A \mid p_i(a) \neq 0 \}.$$

Both paths around the square are equal since all of the λ_i and $p_i(a)$ are positive.

- (??) Let p be a probability distribution on A and q a probability distribution on B. Then

$$\phi(\sigma(p, q)) = \{ (a, b) \mid p(a)q(b) \neq 0 \}$$

while

$$\sigma(\phi(p), \phi(q)) = \{ (a, b) \mid p(a) \neq 0 \, q(b) \neq 0 \}.$$

These are equal since $p(a)q(b) \neq 0$ if and only if both $p(a)$ and $q(b)$ are not 0.

$\square$

This lets us turn a stochastic system into a possibilistic system, saying that a transition is possible if it has non-zero probability.

Exercise 2.5.4.   Show that $\mathrm{D}\eta^{\mathbb{R}} : \mathrm{D}A \to \mathrm{D}(\mathbb{R} \times A)$ is a commutative monad morphism. That is, show that the following diagrams commute:

1.

$$\begin{array}{ccc}
A & \xrightarrow{\eta^{\mathrm{D}}} & \mathrm{D}A \\
& \searrow{\scriptstyle \eta^{\mathrm{D}(\mathbb{R}\times-)}} & \downarrow{\scriptstyle \mathrm{D}(\eta^{\mathbb{R}})} \\
& & \mathrm{D}(\mathbb{R} \times A)
\end{array} \tag{2.14}$$

2.

$$\begin{array}{ccc}
\mathrm{D}^2 A & \xrightarrow{\mathrm{DD}\eta^{\mathbb{R}} \,\text{\textredchi}\, \mathrm{D}\eta^{\mathbb{R}}} & \mathrm{D}(\mathbb{R} \times \mathrm{D}(\mathbb{R} \times A)) \\
{\scriptstyle \mu^{\mathrm{D}}} \downarrow & & \downarrow {\scriptstyle \mu^{\mathrm{D}(\mathbb{R}\times-)}} \\
\mathrm{D}A & \xrightarrow[\mathrm{D}\eta^{\mathbb{R}}]{} & \mathrm{D}(\mathbb{R} \times A)
\end{array} \tag{2.15}$$

3.

$$DA \times DB \xrightarrow{D\eta^{\mathbb{R}} \times D\eta^{\mathbb{R}}} D(\mathbb{R} \times A) \times D(\mathbb{R} \times B)$$

$$\sigma^D \downarrow \qquad\qquad \downarrow \sigma^{D(\mathbb{R} \times -)}$$

$$D(A \times B) \xrightarrow[\quad D\eta^{\mathbb{R}} \quad]{} D(\mathbb{R} \times A \times B)$$

(2.16)

This shows that we can always consider a stochastic system as a stochastic system with rewards by assigning every transition the reward 0. ◊

The reason we need all the laws for the monad morphism and not just an arbitrary family of maps $\phi : MA \to NA$ is that with these laws, we get functors $\mathbf{Kl}(M) \to \mathbf{Kl}(N)$ which tell us that iterating and then changing our non-determinism is the same as changing our non-determinism and then iterating. We begin with a useful lemma.

Lemma 2.5.5. In the definition of a commutative monad map $\phi : M \to N$, the commutativity of diagram **??** can be replaced by the commutativity of the following diagram for any $f : A \to MB$:

$$MA \xrightarrow{\phi} NA$$

$$f^M \downarrow \qquad\qquad \downarrow (f \mathbin{\,\hat{}\,} \phi)^N$$

$$MB \xrightarrow[\phi]{} NB$$

(2.17)

That is,

$$f^M \mathbin{\,\hat{}\,} \phi = \phi \mathbin{\,\hat{}\,} (f \mathbin{\,\hat{}\,} \phi)^N.$$

In do notation, this reads

$$\phi \left(\begin{array}{|l|} \hline \mathbf{do} \\ \qquad x \leftarrow m \\ \qquad f(x) \\ \hline \end{array} \right) = \begin{array}{|l|} \hline \mathbf{do} \\ \qquad x \leftarrow \phi(m) \\ \qquad \phi(f(x)) \\ \hline \end{array}$$

Proof. Before we begin, we note that, by the naturality of ϕ, $M\phi \mathbin{\,\hat{}\,} \phi = \phi \mathbin{\,\hat{}\,} N\phi$:

$$M^2 A \xrightarrow{\phi_{MA}} NMA$$

$$M\phi_A \downarrow \qquad\qquad \downarrow N\phi_A$$

$$MNA \xrightarrow[\phi_{NA}]{} N^2 A$$

That is, we can take the top of **??** to be $\phi \mathbin{\,\hat{}\,} N\phi$ rather than $M\phi \mathbin{\,\hat{}\,} \phi$.

We recall that $f^M = Mf \mathbin{\,\hat{}\,} \mu^M$, and similarly $(f \mathbin{\,\hat{}\,} \phi)^N = N(f \mathbin{\,\hat{}\,} \phi) \mathbin{\,\hat{}\,} \mu^N$. So we may

rewrite **??** as the solid outer diagram in

$$
\begin{array}{ccc}
MA & \xrightarrow{\phi} & NA \\
{\scriptstyle Mf}\big\downarrow & & \big\downarrow{\scriptstyle Nf\,\mathring{,}\,N\phi)} \\
M^2B & \xdashrightarrow{\phi\,\mathring{,}\,N\phi} & N^2B \\
{\scriptstyle \mu^M}\big\downarrow & & \big\downarrow{\scriptstyle \mu^N} \\
MB & \xrightarrow{\phi} & NB
\end{array}
\tag{2.18}
$$

Now we are ready to prove our lemma. We note that the top square in this diagram always commutes by the naturality of ϕ. **??** is the lower square in this diagram; so, if it commutes, then the outer square (which is **??**) commutes. On the other hand, if **??** commutes for all $f : A \to MB$, we may take $f = \mathrm{id} : MA \to MA$ to find that the outer square of **??** becomes just **??**. □

> **Proposition 2.5.6.** Let $\phi : M \to N$ be a commutative monad morhpisms. Then there is a strict symmetric monoidal functor
>
> $$\phi_* : \mathbf{Kl}(M) \to \mathbf{Kl}(N)$$
>
> acting as the identity on objects and sending the Kleisli map $f : A \to MB$ to the composite
>
> $$\phi_* f := A \xrightarrow{f} MB \xrightarrow{\phi} NB.$$

Proof. We will check that this is a functor; that it is strictly symmetric monoidal follows from this and from the fact that it acts as the identity on objects. The identity $\eta^M : A \to MA$ in $\mathbf{Kl}(M)$ gets sent to $\phi_* \eta^M = \eta^M \,\mathring{,}\, \phi$. This equals $\eta^N : A \to NA$ by **??**.

Given $f : A \to MB$ and $g : B \to MC$, their composite is $f \,\mathring{,}\, g^M : A \to MC$, so that

$$
\begin{aligned}
\phi_*(f \,\mathring{,}\, g^M) &:= f \,\mathring{,}\, g^M \,\mathring{,}\, \phi \\
&= f \,\mathring{,}\, \phi \,\mathring{,}\, (g \,\mathring{,}\, \phi)^N && \text{by \textbf{??}} \\
&= (\phi_* f)(\phi_* g)^N.
\end{aligned}
$$

□

We can also check that ϕ_* is a functor using our string diagram notation for monads. In that notation, $\phi : M \to N$ is written as
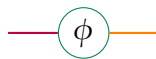
and would satisfy the laws:
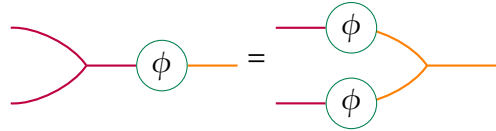
(As before, these diagrams are not really equipped to describe the commutativity of monads, and so we are only using the laws concerning the unit and multiplication.) The action of ϕ_* on a Kleisli map $f : X \to MY$ is then written as
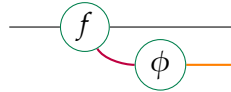
We can check that ϕ_* is functorial quickly and diagrammatically:
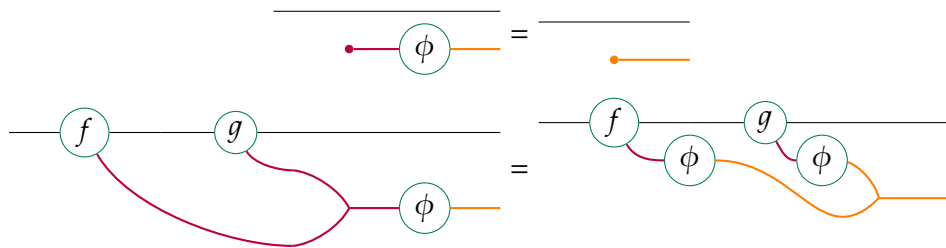
2.6 Wiring together non-deterministic systems: the generalized lens construction

Consider a stochastic source process
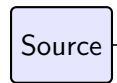
We can imagine, as Claude Shannon did, that this source is an interlocutor communicating over a wire. Suppose we have another interlocutor who will read the signal generated by our source and generate their own signal in repsonse:

Having these two models, we can form a new stochastic source by considering them together:
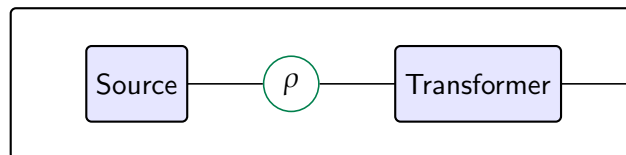
We imagine that the Transformer listens to the signal generated by the Source, but with noise ρ on the wire. This wiring diagram
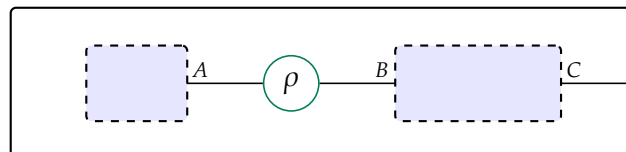
can be described as a *monadic lens* $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} B \\ A \times C \end{pmatrix} \leftrightarrows \begin{pmatrix} 1 \\ C \end{pmatrix}$:

- $f : A \times C \to C$ is the projection π_2.
- $f^\sharp : A \times C \times 1 \to DB$ is $\rho \circ \pi_1$ where $\rho : A \to DB$ is the stochastic function describing the noise on the wire.

This new notion of monadic lens which lets us wire together non-deterministic systems will be the focus of this section.

In **??**, we saw how to wire together systems deterministically and with functions from an algebraic theory on the wire. This worked because wiring diagrams could be interpreted as lenses, and deterministic and differential systems were also lenses; then we could just compose them.

But non-deterministic systems are not lenses in a cartesian category; they have that monad sitting over the states in the codomain of update:

$$\mathsf{update}_S : \mathsf{State}_S \times \mathsf{In}_S \to M\mathsf{State}_S.$$

It may appear that we could consider this as a map in the Kleisli category, and just take lenses in the Kleisli category. But in the Kleisli category, the operation $\times$ is rarely a cartesian product, and we can only describe lenses in cartesian categories. The reason we can only describe lenses in cartesian categories is because in the formula for the passback of a composite of lenses, we use a variable twice; that is, we use the diagonal map $\Delta : A^+ \to A^+ \times A^+$, a feature of cartesian categories.

We will need a new perspective on lenses and lens composition which suggests how to change the passback of the lenses. It is worth noting that we only need to duplicate in the passforward direction; we should be free to change the passback direction.

In this section, we will give a new perspective on the category of lenses using the *Grothendieck construction*. This perspective constructs the category of lenses out of an *indexed category* $\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ of objects of the cartesian category $\mathcal{C}$ *in context*. This construction works for *any* indxed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$, which lets us define a notion of $\mathcal{A}$-lenses out of any indexed category. By choosing an appropriate indexed category, we will arrive at the notion of M-lenses for a commutative monad M; this will give us the wiring diagram calculus for non-deterministic systems that we wanted.

First, we introduce the abstract categorical notions of *indexed category* and the *Grothendieck construction*.

2.6.1   Indexed categories and the Grothendieck construction

An indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ is a family of categories $\mathcal{A}(C)$ that varies functorially with an object $C \in \mathcal{C}$ of the *base category* $\mathcal{C}$ (see **??** for the full definition). We will intepret the base category $\mathcal{C}$ as the category of passforward maps, and the categories $\mathcal{A}(C^+)$ as the categories of passback maps that take C^+ as an extra argument.

Definition 2.6.2. A *strict indexed category* $\mathcal{A} : \mathcal{C}^{\text{op}} \to \mathbf{Cat}$ is a contravariant functor[a]. We call the category $\mathcal{C}$ the *base* of the indexed category $\mathcal{A}$. Explicitly, an indexed category $\mathcal{A}$ has:

- A base category $\mathcal{C}$.
- For every object $C \in \mathcal{C}$ of the base, a category $\mathcal{A}(C)$.
- For every map $f : C \to C'$ in the base, a *pullback* functor $f^* : \mathcal{A}(C') \to \mathcal{A}(C)$, which we think of as "reindexing" the objects of $\mathcal{A}(C')$ so that they live over $\mathcal{A}(C)$.
- Reindexing is functorial: $(f \mathbin{\fatsemi} g)^* = g^* \mathbin{\fatsemi} f^*$ and $\text{id}^* = \text{id}$.

---

[a] A pseudo-functor is like a functor, but it only satisfies the functoriality conditions up to isomorphism, and these isomorphisms must satisfy some laws that make them "cohere". The indexed categories we will see in Chapter 1 and **????** will all be actual functors, however.

Remark 2.6.3. We have given the definition of a *strict* indexed category. A general indexed category is a *pseudo*-functor $\mathcal{A} : \mathcal{C}^{\text{op}} \to \mathbf{Cat}$, which is like a functor but functoriality only holds up to coherent isomorphism. As in the case of monoidal categories, the coherences in the isomorphisms are often just bookkeeping trivialities.

However, the theory of strict indexed categories is noticeably easier, and most of our examples will be strict. Since we will mostly be using strict indexed categories, we will often refer to them simply as "indexed categories".

Indexed categories are quite common throughout mathematics. We will construct a particular example for our own purposes in **??**, and more throughout the book.

Example 2.6.4. Recall that a *dependent set* is a function $X : A \to \mathbf{Set}$ from a set into the category of sets. We have an indexed category of dependent sets

$$\mathbf{Set}^{(-)} : \mathbf{Set}^{\text{op}} \to \mathbf{Cat}$$

which is defined as follows:

- To each set A, we assign the category $\mathbf{Set}^A$ of sets indexed by A. The objects of $\mathbf{Set}^A$ are the sets $X : A \to \mathbf{Set}$ indexed by A, and a map $f : X \to Y$ is a family of maps $f_a : X_a \to Y_a$ indexed by the elements $a \in A$. Composition is given componentwise: $(g \circ f)_a = g_a \circ f_a$.
- To every function $f : A' \to A$, we get a reindexing functor

$$f^* : \mathbf{Set}^A \to \mathbf{Set}^{A'}$$

  Given by precomposition: $X \mapsto X \circ f$. The indexed set $X \circ f : A' \to \mathbf{Set}$ is the set $X_{f(a')}$ on the index $a' \in A'$. The families of functions get reindexed the same way.
- Since our reindexing is just given by precomposition, it is clearly functorial.

We will return to this example in much greater detail in **??**.

If we have an family of sets $A : I \to$ **Set** indexed by a set I, we can form the disjoint union $\sum_{i \in I} A_i$, together with the projection $\pi : \sum_{i \in I} A_i \to I$ sending each $a \in A_i$ to i. The Grothendieck construction is a generalization of this construction to indexed categories. Namely, we will take an indexed category $\mathcal{A} : \mathcal{C} \to$ **Cat** and form a new category

$$\int^{C:\mathcal{C}} \mathcal{A}(C)$$

which we think of as a "union" off all the categories $\mathcal{A}(C)$. But this "union" will not be disjoint since there will be morphisms from objects in $\mathcal{A}(C)$ to objects in $\mathcal{A}(C')$. This is why we use the integral notation; we want to suggest that the Grothendieck construction is a sort of sum.[2]

Definition 2.6.5. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to$ **Cat** be an indexed category. The *Grothendieck construction* of $\mathcal{A}$

$$\int^{C:\mathcal{C}} \mathcal{A}(C)$$

is the category with:

- Objects pairs $\begin{pmatrix} A \\ C \end{pmatrix}$ of objects $C \in \mathcal{C}$ and $A \in \mathcal{A}(C)$. We say that A "sits over" C.
- Maps $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A \\ C \end{pmatrix} \Rightarrow \begin{pmatrix} A' \\ C' \end{pmatrix}$ pairs of $f : C \to C'$ in $\mathcal{C}$ and $f_\flat : A \to f^*A'$ in $\mathcal{A}(C)$.
- Given $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A \\ C \end{pmatrix} \Rightarrow \begin{pmatrix} A' \\ C' \end{pmatrix}$ and $\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} A' \\ C' \end{pmatrix} \Rightarrow \begin{pmatrix} A'' \\ C'' \end{pmatrix}$, their composite is given by

$$\begin{pmatrix} g_\flat \\ g \end{pmatrix} \circ \begin{pmatrix} f_\flat \\ f \end{pmatrix} := \begin{pmatrix} f^*g_\flat \circ f_\flat \\ g \circ f \end{pmatrix}$$

  Written with the signatures, this looks like

$$\begin{pmatrix} A \xrightarrow{f_\flat} f^*A' \xrightarrow{f^*g_\flat} f^*g^*A'' == (g \circ f)^*A'' \\ C \xrightarrow{f} C' \xrightarrow{g} C'' \end{pmatrix}$$

- The identity is given by $\begin{pmatrix} \mathrm{id}_A \\ \mathrm{id}_C \end{pmatrix} : \begin{pmatrix} A \\ C \end{pmatrix} \Rightarrow \begin{pmatrix} A \\ C \end{pmatrix}$

Exercise 2.6.6.   Check that **??** does indeed make $\int^{C:\mathcal{C}} \mathcal{A}(C)$ into a category. That is, check that composition as defined above is associative and unital.   ◊

Pure and cartesian maps.   A map in a Grothendieck construction is a pair $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$: $\begin{pmatrix} A \\ C \end{pmatrix} \Rightarrow \begin{pmatrix} A' \\ C' \end{pmatrix}$ of maps $f : C \to C'$ and $f_\flat : A \to f^*A'$. It is not too hard to see that a map

---

[2]The Grothendieck construction is an example of a *lax colimit* in 2-category theory, another sense in which it is a 'sort of sum'.

is an isomorphism in a Grothendieck construction if and only if both its constituent maps are isomorphisms in their respective categories.

Proposition 2.6.7. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be an indexed category and let $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} A \\ C \end{smallmatrix} \right) \rightrightarrows \left(\begin{smallmatrix} A' \\ C' \end{smallmatrix} \right)$ be a map in its Grothendieck construction. Then $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ is an isomorphism if and only if f is an isomorphism in $\mathcal{C}$ and $f_\flat$ is an isomorphism in $\mathcal{A}(C)$.

Proof. First, let's show that if both f and $f_\flat$ are isomorphisms, then $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ is an isomorphism. We then have $f^{-1} : C' \to C$ and $f_\flat^{-1} : f^*A' \to A$. From $f_\flat^{-1}$, we can form $(f^{-1})^*(f_\flat^{-1}) : (f^{-1})^* f^* A' \to (f^{-1})^* A$, which we can pre-compose with some of the coherences to have the signature $A' \to (f^{-1})^* A$:

$$A' = (f \circ f^{-1})^* A' = (f^{-1})^* f^* A' \xrightarrow{(f^{-1})^*(f_\flat^{-1})} (f^{-1})^* A.$$

Now, consider the map $\left(\begin{smallmatrix} (f^{-1})^* f_\flat^{-1} \\ f^{-1} \end{smallmatrix} \right) : \left(\begin{smallmatrix} A' \\ C' \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} A \\ C \end{smallmatrix} \right)$. We'll show that this is an inverse to $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$. Certainly, the bottom components will work out; we just need to worry about the top. That is, we need to show that $f^*((f^{-1})^* f_\flat^{-1}) \circ f_\flat = \mathrm{id}$ and $(f^{-1})^*(f_\flat) \circ (f^{-1})^*(f_\flat^{-1}) = \mathrm{id}$. Both of these follow quickly by functoriality.

On the other hand, suppose that $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ is an isomorphism with inverse $\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix} \right)$. Then $gf = \mathrm{id}$ and $fg = \mathrm{id}$, so f is an isomorphism. We can focus on $f_\flat$. We know that $f^* g_\flat \circ f_\flat = \mathrm{id}$ and $g^* f_\flat \circ g_\flat = \mathrm{id}$. Applying f^* to the second equation, we find that $f_\flat \circ f^* g_\flat = \mathrm{id}$, so that $f_\flat$ is an isomorphism with inverse $f^* g_\flat$. $\square$

Remark 2.6.8. **??** gives a general solution to **??**, since the category of charts is a Grothendieck construction.

This proposition suggests two interesting classes of maps in a Grothendieck construction: the maps $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ for which f is an isomorphism, and those for which $f_\flat$ is an isomorphism.

Definition 2.6.9. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be an indexed category and let $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ be a map in its Grothendieck construction. We say that $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ is

- *pure* if f is an isomorphism, and
- *cartesian* if $f_\flat$ is an isomorphism.

The pure maps correspond essentially to the maps in the categories $\mathcal{A}(C)$ at a given index C, while the cartesian maps correspond essentially to the maps in $\mathcal{C}$.

Remark 2.6.10. The name "pure" is non-standard. The usual name is "vertical". But we are about to talk about "vertical" maps in a technical sense when we come to double categories, so we've renamed the concept here to avoid confusion later.

Example 2.6.11. We have often seen systems that expose their entire state, like Time of **??**. Considered as lenses, these are *pure* in the sense that their expose function is an isomorphism.

Exercise 2.6.12.   Let $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ and $\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix} \right)$ be composable maps in a Grothendieck construction,

1. Suppose that $\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix} \right)$ is cartesian. Show that $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ is cartesian if and only if their composite is cartesian. Is the same true for pure maps?
2. Suppose that $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ is pure. Show that $\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix} \right)$ is pure if and only if their composite is pure. Is the same true for cartesian maps?

$\diamond$

2.6.13   Maps with context and lenses

In this section, we'll see the category **Lens**$_\mathcal{C}$ of lenses in a cartesian category $\mathcal{C}$ can be described using the Grothendieck construction. To do this, we need some other categories named after their maps (rather than their objects): *category of maps with context C* for some a given $C \in \mathcal{C}$.

Definition 2.6.14. Let $\mathcal{C}$ be a cartesian category and let $C \in \mathcal{C}$. The *category **Ctx**$_C$ of maps with context C* is the category defined by:
  - Objects are the objects of $\mathcal{C}$.
  - Maps $f : X \rightsquigarrow Y$ are maps $f : C \times X \to Y$.
  - The composite $g \circ f$ of $f : X \rightsquigarrow Y$ and $g : Y \rightsquigarrow Z$ is the map

    $$(c, x) \mapsto g(c, f(c, x)) : C \times X \to Z.$$

    Diagrammatically, this is the composite:

    $$C \times X \xrightarrow{\Delta_C \times X} C \times C \times X \xrightarrow{C \times f} C \times Y \xrightarrow{g} Z.$$

  - The identity id : $X \rightsquigarrow X$ is the second projection $\pi_2 : C \times X \to X$.

We can prove that **Ctx**$_C$ is a category using a similar string diagrams to those we used in **??**. We have a functor $X \mapsto C \times X : \mathcal{C} \to \mathcal{C}$ which we can draw as a blue string:

If we represent $X \in \mathcal{C}$ by the string _____ then we represent $C \times X$ as

We can therefore represent a morphism $f : C \times X \to Y$ in the context of C as a bead like this:
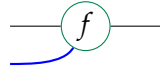
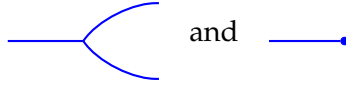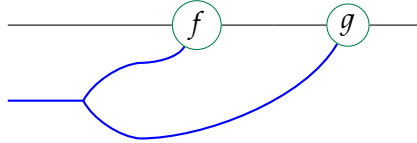To compose maps in context, we need the diagonal map $\Delta_C \times X : C \times X \to C \times C \times X$ and the second projection $\pi_2 : C \times X \to X$. Since these maps are natural in X, we can draw them as

 and

Then the composition in $\mathbf{Ctx}_C$ of maps in context $f : C \times X \to Y$ and $g : C \times Y \to Z$ is drawn as:

and the second projection $\pi_2 : C \times X \to X$ is drawn

This is exactly dual to the story about Kleisli composition we saw in **??**! To show that $\mathbf{Ctx}_C$ is a category, we need to note that the following equations hold:
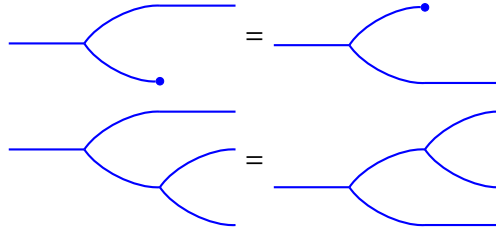
These say that

$$(\Delta_C \times X) \mathbin{\fatsemi} (\pi_2 \times X) = \mathrm{id}_{C \times X} = (\Delta_C \times X) \mathbin{\fatsemi} (C \times \pi_2)$$

and

$$(\Delta_C \times X) \mathbin{\fatsemi} (C \times \Delta_C \times X) = (\Delta_C \times X) \mathbin{\fatsemi} (\Delta_C \times X \times C).$$

These hold by some simple work in the cartesian category C (see **??**).

With these laws in hand, we can prove associativity and identity of composition in $\mathbf{Ctx}_C$ by appealing to the following diagrams:
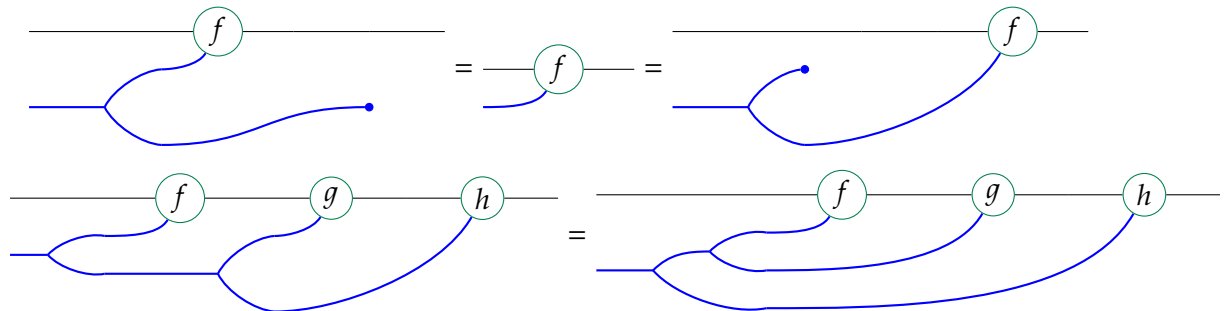
Exercise 2.6.15.   Show that the following composites are equal in any cartesian category:
1.
$$(\Delta_C \times X) \mathbin{\fatsemi} (\pi_2 \times X) = (\Delta_C \times X) \mathbin{\fatsemi} (C \times \pi_2)$$

These are both maps $C \times X \to C \times C \times C \times X$.
2.
$$(\Delta_C \times X) \mathbin{\fatsemi} (C \times \Delta_C \times X) = \mathrm{id}_{C \times X} = (\Delta_C \times X) \mathbin{\fatsemi} (\Delta_C \times X \times C).$$

These are all maps $C \times X \to C \times X$.

◇

Exercise 2.6.16.   Show that **Ctx**$_1$ is equivalent to the underlying cartesian category $\mathcal{C}$. In other words, maps in the context 1 have "no context".   ◇

Together, we can arrange the categories of maps with context into an indexed category.

Definition 2.6.17. The *indexed category of maps with context*

$$\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$$

is defined by:
- For $C \in \mathcal{C}$, we have the category **Ctx**$_C$ of maps with context C.
- For a map $r : C' \to C$, we get a reindexing functor

$$r^* : \mathbf{Ctx}_C \to \mathbf{Ctx}_{C'}$$

  given by sending each object to itself, but each morphism $f : C \times X \to Y$ in **Ctx**$_C$ to the map $r^* f := f \circ (r \times X)$:
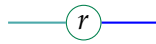
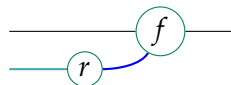$$C' \times X \xrightarrow{r \times X} C \times X \xrightarrow{f} Y.$$

  On elements,
$$r^* f(c', x) := f(r(c'), x).$$
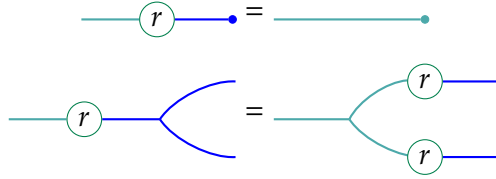
We note that this is evidently functorial.

To see that to every $r : C' \to C$ we get a functor $r^* : \mathbf{Ctx}_C \to \mathbf{Ctx}_{C'}$, we can use string diagrams. We can draw r as
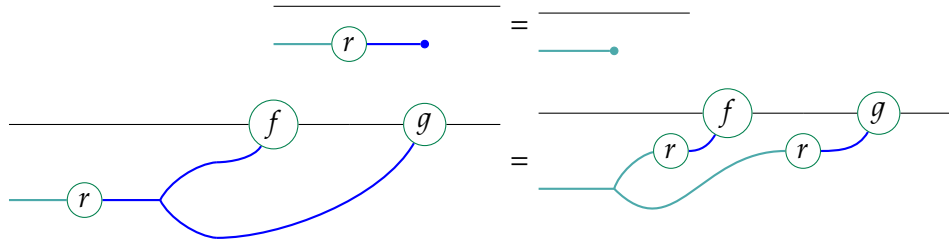

so that the action of r^* is given by

If we note that r satisfies the following laws:

we can then prove that r^* is a functor graphically:

Proposition 2.6.18. The category $\mathbf{Lens}_{\mathcal{C}}$ of lenses in $\mathcal{C}$ is the Grothendieck construction of the indexed category of *opposites* of the categories of maps with context:

$$\mathbf{Lens}_{\mathcal{C}} = \int^{C \in \mathcal{C}} \mathbf{Ctx}_C^{\mathrm{op}}.$$

Proof. We will expand the definition of the right hand side and see that it is precisely the category of lenses.

The objects of $\int^{C \in \mathbf{Set}} \mathbf{Ctx}_C^{\mathrm{op}}$ are pairs $\binom{A^-}{A^+}$ of objects of $\mathcal{C}$. All good so far.

A map in $\int^{C \in \mathcal{C}} \mathbf{Ctx}_C^{\mathrm{op}}$ is a pair $\binom{f^\sharp}{f}$ with $f : A^+ \to B^+$ and $f^\sharp : A^- \rightsquigarrow f^* B^-$ in $\mathbf{Ctx}_{A^+}^{\mathrm{op}}$. Now, $f^* B^- = B^-$ so $f^\sharp$ has signature $A^- \rightsquigarrow B^-$ in $\mathbf{Ctx}_{A^+}^{\mathrm{op}}$, which means $f^\sharp$ has signature $B^- \rightsquigarrow A^-$ in $\mathbf{Ctx}_{A^+}$, which means that $f^\sharp$ is a really a function $A^+ \times B^- \to A^-$. In other words, a map in $\int^{C \in \mathbf{Set}} \mathbf{Ctx}_C^{\mathrm{op}}$ is precisely a lens. We note that the identity map is the identity lens.

Finally, we need to check that composition in $\int^{C \in \mathcal{C}} \mathbf{Ctx}_C^{\mathrm{op}}$ is lens composition. Suppose that $\binom{f^\sharp}{f} : \binom{A^-}{A^+} \leftrightarrows \binom{B^-}{B^+}$ and $\binom{g^\sharp}{g} : \binom{B^-}{B^+} \leftrightarrows \binom{C^-}{C^+}$ are lenses. In $\int^{C \in \mathbf{Set}} \mathbf{Ctx}_C^{\mathrm{op}}$, their composite is

$$\binom{f^* g^\sharp \circ f^\sharp}{g \circ f}.$$

The bottom is all good, we just need to check that the top — which, remember, lives in $\mathbf{Ctx}_{A^+}^{\mathrm{op}}$ — is correct. Since the composite up top is in the opposite, we are really calculating $f^\sharp \circ f^* g^\sharp$ in $\mathbf{Ctx}_{A^+}$. By definition, this is

$$(a^+, c^-) \mapsto f^\sharp(a^+, g^\sharp(f(a^+), c^-))$$

which is precisely their composite as lenses! $\square$

Exercise 2.6.19.   Make sure you *really* understand **??**.   ◊

We take **??** as paradigmatic of the notion of lens, and use this idea to define lenses from any indexed category.

Definition 2.6.20. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be an indexed category. The category of $\mathcal{A}$-lenses is the Grothendieck construction of $\mathcal{A}^{\mathrm{op}}$:

$$\mathbf{Lens}_{\mathcal{A}} = \int^{C \in \mathcal{C}} \mathcal{A}(C)^{\mathrm{op}}.$$

Example 2.6.21. Recall the indexed category $\mathbf{Set}^{(-)} : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Cat}$ of dependent sets from **??**. A $\mathbf{Set}^{(-)}$-lens $\begin{pmatrix} f^{\sharp} \\ f \end{pmatrix} : \begin{pmatrix} A_a^- \\ a \in A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} B_b^- \\ b \in B^+ \end{pmatrix}$ consists of
- A passforward function $f : A^+ \to B^+$, and
- A family of passback functions $f_a^{\sharp} : B^- \to A^-$ for every $a \in A^+$.

We call these *dependent lenses*, and they will be a major character in the later chapters of this book.

2.6.22   Monoidal indexed categories and the product of lenses

To describe wiring diagrams, it is not enough just to have the category of lenses; we also need the monoidal product

$$\begin{pmatrix} A^- \\ A^+ \end{pmatrix} \otimes \begin{pmatrix} B^- \\ B^+ \end{pmatrix} := \begin{pmatrix} A^- \times B^- \\ A^+ \times B^+ \end{pmatrix}.$$

We need this product to put systems together before wiring them. In order to wire together non-deterministic systems, we will need to generalize this product of lenses to generalized lenses. For this, we will need the notion of an *monoidal indexed category* and the associated *monoidal Grothendieck construction* as defined in [**MoellerVasilakopolou**].

Definition 2.6.23. A *monoidal strict indexed category* $(\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, \otimes, 1, \boxtimes, \Bbbk)$ consists of:
- A strict indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$,
- A monoidal structure $(\otimes, 1)$ on $\mathcal{C}$,
- A natural family of functors $\boxtimes : \mathcal{A}(C) \times \mathcal{A}(C') \to \mathcal{A}(C \otimes C')$ and $\Bbbk \in \mathcal{A}(1)$ with natural isomorphisms

$$A_1 \boxtimes (A_2 \boxtimes A_3) \cong (A_1 \boxtimes A_2) \boxtimes A_3,$$

$$\Bbbk \boxtimes A \cong A \cong A \boxtimes \Bbbk.$$

These natural isomorphisms are required to satisfy coherences reminiscent of

those of a monoidal category.

Theorem 2.6.24 ([**MoellerVasilakopolou**]). Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be a monoidal indexed category. Then the Grothendieck construction $\int^{C:\mathcal{C}} \mathcal{A}(C)$ may be equipped with a monoidal structure

$$\begin{pmatrix} A^- \\ A^+ \end{pmatrix} \otimes \begin{pmatrix} B^- \\ B^+ \end{pmatrix} := \begin{pmatrix} A^- \boxtimes B^- \\ A^+ \otimes B^+ \end{pmatrix}.$$

If the base of indexing $\mathcal{C}$ is cartesian, then there is a simpler way to describe a monoidal structure on an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$.

Theorem 2.6.25 ([**ShulmanMonoidal**]). Let $\mathcal{C}$ be a cartesian category. Then a monoidal structures on a strict indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ whose underlying monoidal structure on $\mathcal{C}$ is given by the cartesian product may be equivalently given by the data:
- A monoidal structure $\otimes : \mathcal{A}(C) \times \mathcal{A}(C) \to \mathcal{A}(C)$ and $1 \in \mathcal{A}(C)$ for each $C \in \mathcal{C}$,
- A lax structure on each reindexing $r^* : \mathcal{A}(C) \to \mathcal{A}(C')$ for each $r : C' \to C$, so that the lax structure on $(r_2 \circ r_1)^*$ is the composite of the lax structures on r_2 and r_1.

Proof Sketch. We define the product $\otimes : \mathcal{A}(C) \times \mathcal{A}(C) \to \mathcal{A}(C)$ as $\boxtimes \circ \Delta^*$ where $\Delta : C \to C \times C$ is the diagonal. We similarly define $1 \in \mathcal{A}(C)$ as $!^*(\Bbbk)$. □

We use **??** and **??** to recover the product of lenses.

Lemma 2.6.26. Let $\mathcal{C}$ be a cartesian category and let $C \in \mathcal{C}$. The category $\mathbf{Ctx}_C$ has a monoidal structure given by $X \otimes Y := X \times Y$, $1 := 1$, and

$$f \otimes g := C \times X \times Y \xrightarrow{\Delta} C \times C \times X \times Y \xrightarrow{\sim} C \times X \times C \times Y \xrightarrow{f \times g} X' \times Y'.$$

In terms of elements,
$$(f \otimes g)(c, x, y) := (f(c, x), g(c, y)).$$

Proof. We begin by showing that $\otimes$ is functorial:

$$\begin{aligned}
((f' \circ f) \otimes (g' \circ g))(c, x, y) &= ((f' \circ f)(c, x), (g' \circ g)(c, y)) \\
&= (f'(c, f(c, x)), g'(c, g'(c, y))) \\
&= (f' \otimes g')(f(c, x), g(c, y)) \\
&= (f' \otimes g') \circ (f \otimes g)(x, y).
\end{aligned}$$

Next, we need associators $X \otimes (Y \otimes Z) \cong (X \otimes Y) \otimes Z$ and unitors $1 \otimes X \cong X \cong X \otimes 1$. We may get these by applying $!^* : \mathcal{C} \to \mathbf{Ctx}_C$ (which sends $f : X \to Y$ to $f \circ \pi_1 : C \times X \to Y$) to the associators and unitors of $\mathcal{C}$. It is straightforward to see that these are natural with respect to maps in $\mathbf{Ctx}_C$. □

Proposition 2.6.27. Let $\mathcal{C}$ be a cartesian category. Then $\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ may be endowed with a monoidal structure so that the induced monoidal structure on the Grothendieck construction is the product of lenses

$$\begin{pmatrix} A^- \\ A^+ \end{pmatrix} \otimes \begin{pmatrix} B^- \\ B^+ \end{pmatrix} := \begin{pmatrix} A^- \times B^- \\ A^+ \times B^+ \end{pmatrix}.$$

Proof. By **??**, there is monoidal structure on each $\mathbf{Ctx}_C$. We note that by definition, each reindexing $r^* : \mathbf{Ctx}_C \to \mathbf{Ctx}_{C'}$ along $r : C' \to C$ preserves this monoidal structure strictly.

$$\begin{aligned} r^*(f \otimes g)(c', (x, y)) &= (f \otimes g)(r(c'), (x, y)) \\ &= (f(r(c'), x), g(r(c'), y)) \\ &= (r^* f \otimes r^* g)(c, (x, y)). \end{aligned}$$

The rest then follows by **??** and **??**. □

2.6.28 Monadic lenses as generalized lenses

Now we are ready to define monadic lenses. We have a formula: we just need to find the right indexed category. We will do this by modifying the definition of $\mathbf{Ctx}_C$ so that a map is of the form $C \times X \to MY$. If the resulting categories $\mathbf{Ctx}_C^M$ remain indexed over $\mathcal{C}$, we have a ready made notion of monadic lens and monadic lens composition given by the Grothendieck construction!

We will be able to define composition in the categories $\mathbf{Ctx}_C^M$ by making use of the natural map

$$\lambda : C \times MX \xrightarrow{\eta \times MX} MC \times MX \xrightarrow{\sigma} M(C \times X)$$

Using string diagrams, we may draw this map as
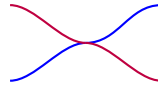
Using do notation, we may describe this map as

$$(c, m) \mapsto \boxed{\begin{array}{l} \mathbf{do} \\ \quad c' \leftarrow \eta(c) \\ \quad x \leftarrow m \\ \quad \eta(c', x) \end{array}} = \boxed{\begin{array}{l} \mathbf{do} \\ \quad x \leftarrow m \\ \quad \eta(c, x) \end{array}}$$

Definition 2.6.29. Let $\mathcal{C}$ be a cartesian category and $M : \mathcal{C} \to \mathcal{C}$ a commutative monad. For an object $C \in \mathcal{C}$, there is a category $\mathbf{Ctx}_C^M$ with:
- Objects the objects of $\mathcal{C}$.
- Map $f : X \rightsquigarrow Y$ are maps $f : C \times X \to MY$ in $\mathcal{C}$.

- The identity $X \rightsquigarrow X$ is $\pi_2 \,\mathring{,}\, \eta$.
- The composite $f \,\mathring{,}\, g$ of $f : X \rightsquigarrow Y$ and $g : Y \rightsquigarrow Z$ is given by

$$f \,\mathring{,}\, g := (\Delta_C \times X) \,\mathring{,}\, (C \times f) \,\mathring{,}\, \lambda \,\mathring{,}\, Mg \,\mathring{,}\, \mu.$$

$$C \times X \to C \times C \times X \to C \times MY \to M(C \times Y) \to M^2 Z \to MZ$$

Here, $\lambda := (\eta \times MX) \,\mathring{,}\, \sigma$. Using do notation, we may describe the composite $f \,\mathring{,}\, g$ as

$$(c, m) \mapsto \boxed{\begin{array}{l} \textbf{do} \\ \quad x \leftarrow m \\ \quad y \leftarrow f(c, x) \\ \quad g(c, y) \end{array}}$$

We can show that $\textbf{Ctx}_C^M$ is indeed a category using string diagrams. In string diagrams, a map $f : C \times X \to MY$ in $\textbf{Ctx}_C^M$ is drawn
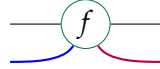
and composition is drawn
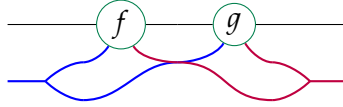
The identity is drawn

In order to show that this composition is unital and associative, we will need to show that the following four laws hold relating λ to the structure of M and of $C \times (-)$:

$$(2.19)$$

$$(2.20)$$

$$(2.21)$$

$$(2.22)$$

We will prove these laws in the upcoming **??**.[3] Using them, we can see that composition in $\mathbf{Ctx}_C^M$ is unital and associative.
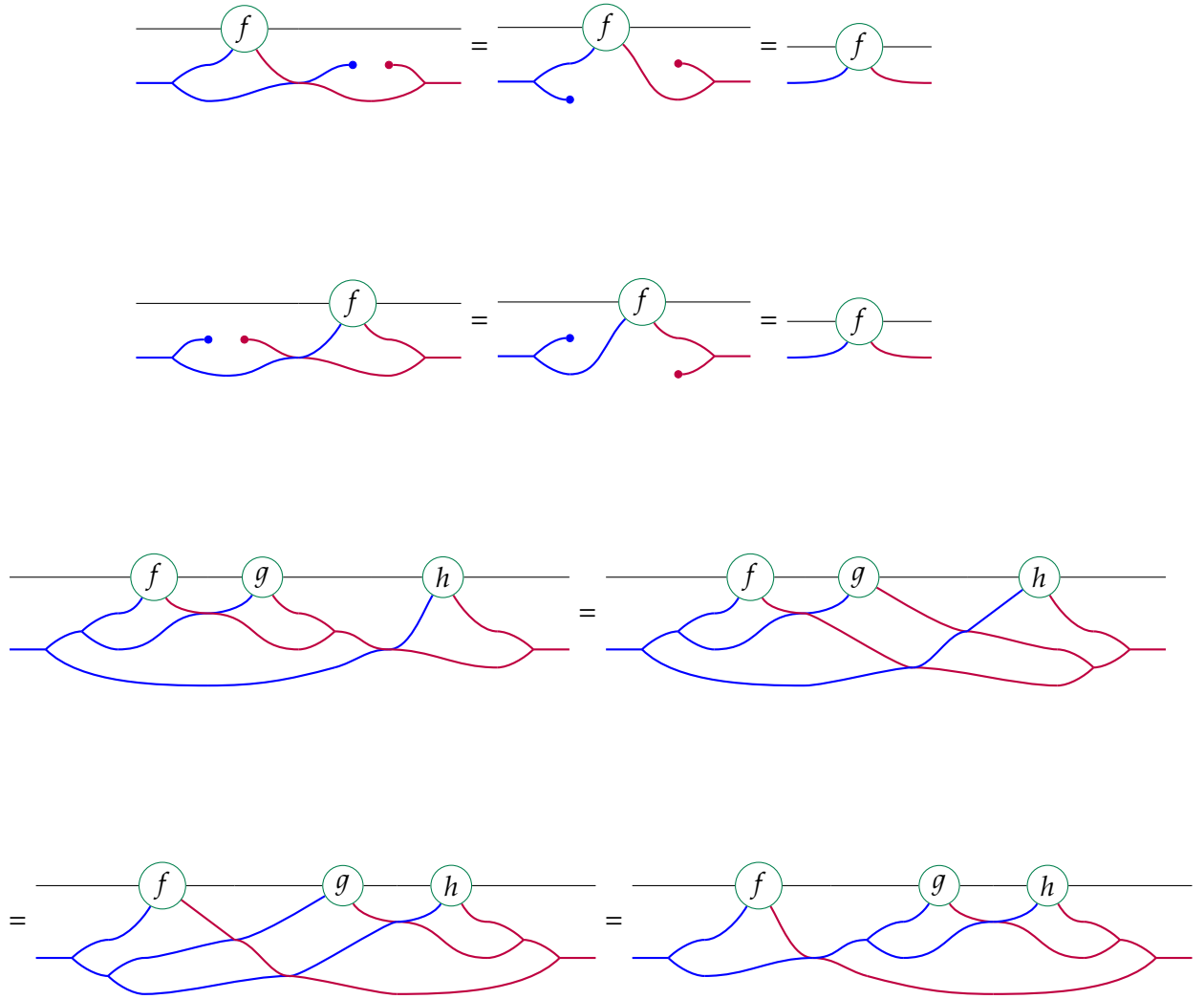
This shows that $\mathbf{Ctx}_C^M$ is a category. We now prove the crucial laws which undergird the above graphical arguments.

---

[3]And we will re-express them as commutative diagrams there.

Lemma 2.6.30. Let $M : \mathcal{C} \to \mathcal{C}$ be a commutative monad on a cartesian category $\mathcal{C}$. Then the map $\lambda : C \times MX \to M(C \times X)$ defined by

$$\lambda := (\eta \times MX) \, \mathring{,} \, \sigma$$

is natural in both X and C. Furthermore, the following four diagrams commute:

$$
\begin{array}{ccc}
C \times X & \xrightarrow{\;C \times \eta\;} & C \times MX \\
 & \eta \searrow & \downarrow \lambda \\
 & & M(C \times X)
\end{array}
\tag{2.23}
$$

$$
\begin{array}{ccc}
C \times MX & \xrightarrow{\;\lambda\;} & M(C \times X) \\
 & \pi_2 \searrow & \downarrow M\pi_2 \\
 & & MX
\end{array}
\tag{2.24}
$$

$$
\begin{array}{ccc}
C \times M^2 X & \xrightarrow{\;C \times \mu\;} & C \times MX \\
\lambda \downarrow & & \downarrow \lambda \\
M(C \times MX) \xrightarrow{M\lambda} M^2(C \times X) \xrightarrow{\mu} & & M(C \times X)
\end{array}
\tag{2.25}
$$

$$
\begin{array}{ccc}
C \times MX & \xrightarrow{\;\lambda\;} & M(C \times X) \\
\Delta_C \times MX \downarrow & & \downarrow M(\Delta_C \times X) \\
C \times C \times MX \xrightarrow{C \times \lambda} C \times M(C \times X) \xrightarrow{\lambda} & & M(C \times C \times X)
\end{array}
\tag{2.26}
$$

Exercise 2.6.31.   Prove **??** by showing that the diagrams commute. This uses the properties of the commutativity σ and naturality. You may find the do notation helpful.

◊

In order to wire diagrams together, we also need the monoidal product on lenses.

Lemma 2.6.32. Let $\mathcal{C}$ be a cartesian category and let $M : \mathcal{C} \to \mathcal{C}$ be a commutative monad. Then for any $C \in \mathcal{C}$, there is a symmetric monoidal structure on $\mathbf{Ctx}_C^M$ given by $X \otimes Y := X \times Y$, with unit 1, and

$$f \otimes g := C \times X \times Y \xrightarrow{\Delta} C \times C \times X \times Y \xrightarrow{\sim} C \times X \times C \times Y \xrightarrow{f \times g} MX \times MY \xrightarrow{\sigma} M(X \times Y).$$

With the do notation, $f \otimes g$ may be defined as

$$(c, x, y) \mapsto \boxed{\begin{array}{l} \mathbf{do} \\ \quad z \leftarrow f(c, x) \\ \quad w \leftarrow g(c, y) \\ \quad \eta(z, w) \end{array}}$$

Proof. We will use the do notation to argue this. The proofs in the do notation can, with some care, be extended out into diagram chases if the reader desires to do so.

We will show that $\otimes$ is functorial. Let $f : X_1 \to Y_1$, $g : X_2 \to Y_2$, $f' : Y_1 \to Z_1$ and $g' : Y_2 \to Z_2$. Then

$$(f \otimes g) \mathbin{\fatsemi} (f' \otimes g') = (c, x_1, x_2) \mapsto \boxed{\begin{array}{l} \mathbf{do} \\ \quad y_1 \leftarrow f(c, x_1) \\ \quad y_2 \leftarrow g(c, x_2) \\ \quad z_1 \leftarrow f'(c, y_1) \\ \quad z_2 \leftarrow g'(c, y_2) \\ \quad \eta(z_1, z_2) \end{array}}$$

$$= (c, x_1, x_2) \mapsto \boxed{\begin{array}{l} \mathbf{do} \\ \quad y_1 \leftarrow f(c, x_1) \\ \quad z_1 \leftarrow f'(c, y_1) \\ \quad y_2 \leftarrow g(c, x_2) \\ \quad z_2 \leftarrow g'(c, y_2) \\ \quad \eta(z_1, z_2) \end{array}}$$

$$= (f \mathbin{\fatsemi} f') \otimes (g \mathbin{\fatsemi} g')$$

Note the use of commutativity.

Next, we need to give associators $\alpha : (X \otimes Y) \otimes Z \to X \otimes (Y \otimes Z)$ and unitors $\ell : 1 \otimes X \to X$ and $r : X \otimes 1 \to X$.

$$\alpha(c, (x, y), z) := \eta(x, (y, z)).$$
$$\ell(c, (*, x)) := \eta(x)$$
$$r(c, (x, *)) := \eta(x)$$

These can easily be seen to satisfy the required coherences, and they are just defined by shuffling the parentheses about. $\square$

From this, we may finally prove the following theorem.

Theorem 2.6.33. Let $M : \mathcal{C} \to \mathcal{C}$ be a commutative monad on a cartesian category. Then there is a monoidal strict indexed category

$$\mathbf{Ctx}_-^M : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$$

which sends an object $C \in \mathcal{C}$ to the category $\mathbf{Ctx}_C^M$ and which sends a map $r : C' \to C$ to the functor

$$r^* : \mathbf{Ctx}_C^M \to \mathbf{Ctx}_{C'}^M$$

which acts as the identity on objects and which sends a morphism $f : C \times X \to MY$ to the composite $C' \times X \xrightarrow{r \times X} C \times X \xrightarrow{f} MY$.

Proof. All that remains to be proven is functoriality in C. Letting $r : C' \to C$, we get a functor $r^* : \mathbf{Ctx}_C^M \to \mathbf{Ctx}_{C'}^M$ given by sending $f : C \times X \to MY$ to $f \circ (r \times X) : C' \times X \to MY$. In terms of elements, this means

$$r^* f(c', x) := f(r(c'), x)$$

Using the do notation, we can quickly show that this is functorial:

$$r^*(g \circ f)(c', x) = (g \circ f)(r(c'), x)$$

$$= \begin{array}{|l|} \hline \mathbf{do} \\ \quad y \leftarrow f(r(c'), x) \\ \quad g(r(c'), y) \\ \hline \end{array}$$

$$= \begin{array}{|l|} \hline \mathbf{do} \\ \quad y \leftarrow r^* f(c', x) \\ \quad r^* g(c', y) \\ \hline \end{array}$$

$$= (r^* g \circ r^* f)(c', x)$$

To show that it is monoidal, we may also use the do notation:

$$r^*(f \otimes g)(c', x, y) = (f \otimes g)(r(c'), x, y)$$

$$= \begin{array}{|l|} \hline \mathbf{do} \\ \quad z \leftarrow f(r(c'), x) \\ \quad w \leftarrow g(r(c'), y) \\ \quad \eta(z, w) \\ \hline \end{array}$$

$$= \begin{array}{|l|} \hline \mathbf{do} \\ \quad z \leftarrow r^* f(c', x) \\ \quad w \leftarrow r^* g(c', y) \\ \quad \eta(z, w) \\ \hline \end{array}$$

$$= (r^* f) \otimes (r^* g)(c', x, y)$$

□

With **??** in hand, we may now define the category of monadic lenses.

Definition 2.6.34. For a commutative monad $M : \mathcal{C} \to \mathcal{C}$ on a cartesian category, we define the symmetric monoidal category of M-lenses to be the symmetric monoidal category of $\mathbf{Ctx}_-^M$-lenses:

$$\mathbf{Lens}_{\mathcal{C}}^M := \int^{C:\mathcal{C}} \mathbf{Ctx}_{C}^{M\,\mathrm{op}}$$

Exercise 2.6.35.   Show that the category of M-lenses may be described as follows:
- Its objects are pairs $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ of objects of $\mathcal{C}$.
- Its maps are M-lenses $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ where $f : A^+ \to B^+$ and $f^\sharp : A^+ \times B^- \to MA^-$.
- The identity is $\begin{pmatrix} \eta \circ \pi_2 \\ \mathrm{id} \end{pmatrix}$.
- Composition is defined by

$$\begin{pmatrix} f^\sharp \\ f \end{pmatrix} \,\mathring{\,}\, \begin{pmatrix} g^\sharp \\ g \end{pmatrix} := \begin{pmatrix} h \\ f \,\mathring{\,}\, g \end{pmatrix}$$

where h is defined in the do notation as

$$h(a^+, c^-) := \boxed{\begin{aligned} &\mathbf{do} \\ &\quad b^- \leftarrow g^\sharp(f(a^+), c^-) \\ &\quad f^\sharp(a^+, b^-) \end{aligned}}$$

◊

2.7   Changing the Flavor of Non-determinism

In **??**, we saw how commutative monad maps $\phi : M \to N$ let us change the flavor of non-determinism. In particular, since the unit $\eta : \mathrm{id} \to M$ is always a commutative monad map, we can always interpret a deterministic system as a non-deterministic system.

In this section, we'll show that any commutative monad morphism $\phi : M \to N$ induces a symmetric monoidal functor $\mathbf{Lens}_{\mathcal{C}}^M \to \mathbf{Lens}_{\mathcal{C}}^N$. We will do this using the functoriality of the Grothendieck construction: any *indexed functor* induces a functor on the Grothendieck constructions.

Definition 2.7.1. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ and $\mathcal{B} : \mathcal{D}^{\mathrm{op}} \to \mathbf{Cat}$ be strict indexed categories. A *strict indexed functor* $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$ is a pair consisting of
- A functor $F : \mathcal{C} \to \mathcal{D}$, and

- A natural transformation $\overline{F} : \mathcal{A} \to \mathcal{B} \circ F^{\mathrm{op}}$. Explicitly, this is a family of functors $\overline{F}_C : \mathcal{A}(C) \to \mathcal{B}(FC)$ so that for any $r : C' \to C$, we have that $\overline{F} \circ r^* = (Fr)^* \circ \overline{F}$.

If $\mathcal{A}$ and $\mathcal{B}$ are monoidal strict indexed categories, then an indexed functor $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$ is strict monoidal if $F(C_1 \otimes C_2) = FC_1 \otimes FC_2$ and $\overline{F}(A_1 \boxtimes A_2) = \overline{F}(A_1) \boxtimes \overline{F}(A_2)$, and F and $\overline{F}$ send associators to associators and unitors to unitors.

Proposition 2.7.2. Let $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$ be a strict indexed functor. Then there is a functor

$$\binom{\overline{F}}{F} : \int^{C : \mathcal{C}} \mathcal{A}(C) \to \int^{D : \mathcal{D}} \mathcal{B}(D)$$

given by

$$\binom{\overline{F}}{F}\binom{f_\flat}{f} := \binom{\overline{F}f_\flat}{Ff}.$$

If furthermore $(F, \overline{F})$ is strictly monoidal, then so is $\binom{\overline{F}}{F}$.

Proof. We will show that this assignment is functorial. Recall that

$$\binom{f_\flat}{f} \mathbin{\mathring{\,}}_{\mathring{\,}} \binom{g_\flat}{g} := \binom{f_\flat \mathbin{\mathring{\,}}_{\mathring{\,}} f^* g_\flat}{f \mathbin{\mathring{\,}}_{\mathring{\,}} g}.$$

We may therefore calculate:

$$\binom{\overline{F}}{F}\left(\binom{f_\flat}{f} \mathbin{\mathring{\,}}_{\mathring{\,}} \binom{g_\flat}{g}\right) = \binom{\overline{F}(f_\flat \mathbin{\mathring{\,}}_{\mathring{\,}} f^* g_\flat)}{F(f \mathbin{\mathring{\,}}_{\mathring{\,}} g)}$$

$$= \binom{\overline{F}f_\flat \mathbin{\mathring{\,}}_{\mathring{\,}} \overline{F}(f^* g_\flat)}{Ff \mathbin{\mathring{\,}}_{\mathring{\,}} Fg}$$

$$= \binom{\overline{F}f_\flat \mathbin{\mathring{\,}}_{\mathring{\,}} (Ff)^*(\overline{F}g_\flat)}{Ff \mathbin{\mathring{\,}}_{\mathring{\,}} Fg}$$

$$= \binom{\overline{F}}{F}\binom{f_\flat}{f} \mathbin{\mathring{\,}}_{\mathring{\,}} \binom{\overline{F}}{F}\binom{g_\flat}{g}$$

We end by noting that $\binom{\overline{F}}{F}\binom{\mathrm{id}}{\mathrm{id}} = \binom{\mathrm{id}}{\mathrm{id}}$ by functoriality of F and $\overline{F}$.

If $(F, \overline{F})$ is strictly monoidal, then so is $\binom{\overline{F}}{F}$ because the monoidal structure of the Grothendieck constructions are defined by pairing the monoidal structures of the base. $\square$

Proposition 2.7.3. Let $\phi : M \to N$ be a commutative monad morphism. Then there is a strict monoidal indexed functor

$$(\mathrm{id}, \phi_*) : \mathbf{Ctx}_-^M \to \mathbf{Ctx}_-^N.$$

Proof. We need to give a family of strict monoidal functors $\phi_* : \mathbf{Ctx}_C^M \to \mathbf{Ctx}_C^N$, natural in C. We take ϕ_* to act as the identity on objects, and for $f : C \times X \to MY$, we define

$$\phi_* f := f \,\mathbin{\mathring{\,}}\, \phi.$$

We now show that this is functorial using the do notation:

$$\phi_*(f \mathbin{\mathring{\,}} g) = f \mathbin{\mathring{\,}} g \mathbin{\mathring{\,}} \phi$$

$$= (c, x) \mapsto \phi \left(\begin{array}{|l|} \hline \mathbf{do} \\ \quad y \leftarrow f(c, x) \\ \quad g(c, y) \\ \hline \end{array} \right)$$

$$= (c, x) \mapsto \begin{array}{|l|} \hline \mathbf{do} \\ \quad y \leftarrow \phi(f(c, x)) \\ \quad \phi(g(c, y)) \\ \hline \end{array} \qquad \text{by ??}$$

$$= \phi_* f \mathbin{\mathring{\,}} \phi_* g.$$

We also note that $\phi_*\mathrm{id} = \mathrm{id}$ since

$$\phi_*(\mathrm{id}) = \pi_2 \mathbin{\mathring{\,}} \eta^M \mathbin{\mathring{\,}} \phi$$

$$= \pi_2 \mathbin{\mathring{\,}} \eta^N$$

$$= \mathrm{id}$$

We may also use the do notation to prove strict monoidal-ness. We begin by noting that the functor is strictly monoidal on objects since it is identity on objects and the monoidal structures are defined identically.

$$\phi_*(f \otimes g) = (c, x_1, x_2) \mapsto \phi \left(\begin{array}{|l|} \hline \mathbf{do} \\ \quad y_1 \leftarrow f(c, x_1) \\ \quad y_2 \leftarrow g(c, x_2) \\ \quad \eta^M(y_1, y_2) \\ \hline \end{array} \right)$$

$$= \begin{array}{|l|} \hline \mathbf{do} \\ \quad y_1 \leftarrow \phi(f(c, x_1)) \\ \quad y_2 \leftarrow \phi(g(c, x_2)) \\ \quad \phi\eta^M(y_1, y_2) \\ \hline \end{array}$$

$$= \begin{array}{|l|} \hline \mathbf{do} \\ \quad y_1 \leftarrow \phi(f(c, x_1)) \\ \quad y_2 \leftarrow \phi(g(c, x_2)) \\ \quad \eta^N(y_1, y_2) \\ \hline \end{array}$$

$$= \phi_* f \otimes \phi_* g.$$

$\square$

Corollary 2.7.4. Let $\phi : M \to N$ be a commutative monad morphism. Then there is a strict monoidal functor

$$\phi_* : \mathbf{Lens}_{\mathcal{C}}^M \to \mathbf{Lens}_{\mathcal{C}}^N$$

Given by

$$\phi_* \begin{pmatrix} f^\sharp \\ f \end{pmatrix} := \begin{pmatrix} f^\sharp \, \mathbin{\mathring{\,}} \phi \\ f \end{pmatrix}.$$

Proof. We may apply **??** to **??** (or, more precisely, to the pointwise opposite $(\mathrm{id}, \phi_*^{\mathrm{op}})$). $\square$

The theorem has a useful corollary: we can always wire together non-deterministic systems with wiring diagrams.

Corollary 2.7.5. For any commutative monad $M : \mathcal{C} \to \mathcal{C}$, there is a strictly monoidal functor

$$\eta_* : \mathbf{Lens}_{\mathcal{C}} \to \mathbf{Lens}_{\mathcal{C}}^M.$$

Example 2.7.6. Suppose we have two people S_1 and S_2 flipping coins. S_1 flips a single fair coin and exposes its value:
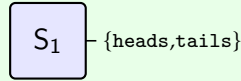
That is, $\mathsf{State}_{S_1} = \{\texttt{heads}, \texttt{tails}\}$

$$\mathrm{update}_{S_1}(\_) = \frac{1}{2}\texttt{heads} + \frac{1}{2}\texttt{tails}$$

$$\mathrm{expose}_{S_1} = \mathrm{id}.$$

On the other hand, S_2 will flip either a left coin or a right coin, and expose the resulting value. But these coins are biased in different ways The coin that S_2 flips is determined by whether it sees `heads` or `tails`.
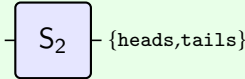
That is, $\mathsf{State}_{S_1} = \{\texttt{heads}, \texttt{tails}\}$ and

$$\mathrm{update}_{S_2}(\_, \texttt{heads}) = \frac{1}{4}\texttt{heads} + \frac{3}{4}\texttt{tails}$$

$$\mathrm{update}_{S_2}(\_, \texttt{tails}) = \frac{3}{4}\texttt{heads} + \frac{1}{4}\texttt{tails}$$

$$\text{expose}_{S_2} = \text{id}.$$

We can now imagine that S_1 sends the result of their coin flip over a channel to S_2. But this channel has noise given by

$$\rho(\texttt{heads}) = \frac{9}{10}\texttt{heads} + \frac{1}{10}\texttt{tails}$$

$$\rho(\texttt{tails}) = \frac{1}{10}\texttt{heads} + \frac{9}{10}\texttt{tails}$$

Explictly, we will compose with the wiring diagram:
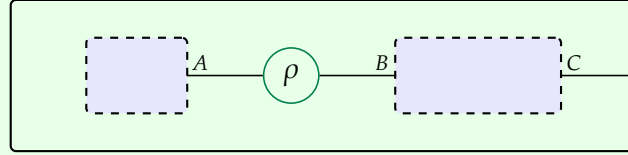
We can describe this as a D-lens

$$\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} \mathsf{In}_{S_1} \times \mathsf{In}_{S_2} \\ \mathsf{Out}_{S_1} \times \mathsf{Out}_{S_2} \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ \mathsf{Out}_{S_2} \end{pmatrix}$$

- $w : \mathsf{Out}_{S_1} \times \mathsf{Out}_{S_2} \to \mathsf{Out}_{S_2}$ is the projection π_2.
- $w^\sharp : \mathsf{Out}_{S_1} \times \mathsf{Out}_{S_2} \times \{*\} \to D(\mathsf{In}_{S_1} \times \mathsf{In}_{S_2})$ is given by

$$w^\sharp(x, y, *) = (*, \rho(x)).$$

We may now form the composite system:
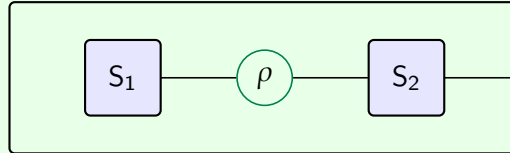
This has states $\mathsf{States}_{S_1} \times \mathsf{States}_{S_2}$, exposes just the state of States_{S_2}, and updates in the following way:

$$\text{update}((\_,\_),*) = \begin{array}{l} \frac{1}{2}\left(\frac{9}{10}\frac{1}{4} + \frac{1}{10}\frac{3}{4}\right)(\texttt{heads},\texttt{heads}) \\[2ex] + \frac{1}{2}\left(\frac{1}{10}\frac{3}{4} + \frac{9}{10}\frac{1}{4}\right)(\texttt{tails},\texttt{heads}) \\[2ex] + \frac{1}{2}\left(\frac{9}{10}\frac{3}{4} + \frac{1}{10}\frac{1}{4}\right)(\texttt{heads},\texttt{tails}) \\[2ex] + \frac{1}{2}\left(\frac{1}{10}\frac{3}{4} + \frac{9}{10}\frac{1}{4}\right)(\texttt{tails},\texttt{tails}) \end{array}$$

2.8 Summary and Further Reading

In this chapter, we extended the notion of lens to monadic lenses to accomodate non-deterministic systems. We saw how any commutative monad gave rise to a doctrine of non-determinism, from possibilistic to probabilistic to costs and rewards. One nice thing about monads is that you can play with them in Haskell. There are plenty of places to learn about monads in Haskell (perhaps too many), so I won't make any specific recommendations. For more about monads in category theory, check out Chapter 5 of [Per21].

We then saw how the notion of lens could be generalized to any indexed category. This notion of generalized lens is due to Spivak in [Spi19]. This generalization of lens will underly our formal notion of dynamical system doctrine, which will be introduced in the next chapter.

Chapter 3

How systems behave

3.1 Introduction

So far, we have seen how to wire up dynamical systems. But we haven't seen our dynamical systems actually *do anything*. In this section, we will begin to study the behavior of our dynamical systems. We will see particular kinds of behaviors our systems can have: trajectories, steady states, and periodic orbits.

Informal Definition 3.1.1. A *behavior* of a dynamical system is a particular way its states can change according to its dynamics.

There are different *kinds of behavior* corresponding to the different sorts of ways that the states of a system could evolve. Perhaps they eventually repeat, or they stay the same despite changing conditions.

In **??**, we will give a formal definition of behavior of dynamical system. We will see that the different kinds of behaviors — trajectories, steady states, periodic orbits, etc. — can each be packaged up into a single system[1] that *represents* that kind of behavior. This system will behave in exactly that kind of way, and do nothing else. Maps from it to a system of interest will exhibit that sort of behavior in the system of interest.

We will then investigate the definition of behaviors in terms of a *double category* which merges together the category of lenses with a category of *charts* (which are important for defining behaviors). We will see that behaviors are certain squares in this double category, and see what using this doublec category can tell us about how behaviors of component systems relate to the behaviors of composed systems.

---

[1]Or family of systems.

3.2   Kinds of behavior

3.2.1   Trajectories

A trajectory is the simplest and freest sort of behavior a system can have. A trajectory
is just "what a state does". In this section, we will see what trajectories look like in the
deterministic and differential doctrines.

Trajectories in the deterministic doctrine

In the introduction, we saw that the Clock system Eq. (1.2) has behaves in this way if it
starts at 3 o'clock:

$$3 \xmapsto{\text{tick}} 4 \xmapsto{\text{tick}} 5 \xmapsto{\text{tick}} 6 \xmapsto{\text{tick}} \cdots$$

This sequence of states of the clock system, each following from the last by the
dynamics of the system, is called a *trajectory*. When our systems have input parameters,
we will need to choose a sequence of input parameters to feed the system in order for
the states to change.

Definition 3.2.2. Let

$$S = \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \text{States}_S \\ \text{States}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$$

be a deterministic system. Suppose that $p : \mathbb{N} \to \text{In}_S$ is a sequence of parameters for S.
Then a *p-trajectory* of S is a sequence $s : \mathbb{N} \to \text{States}_S$ of states so that

$$\text{update}_S(s_i, p_i) = s_{i+1}$$

for all $i \in \mathbb{N}$.
    If additionally $v : \mathbb{N} \to \text{Out}_S$ is a sequence of output values for S, then a $\begin{pmatrix} p \\ v \end{pmatrix}$-*trajectory*
is a sequence of states $s : \mathbb{N} \to \text{States}_S$ so that

$$\text{update}_S(s_i, p_i) = s_{i+1}$$
$$\text{expose}_S(s_i) = v_i$$

for all $i \in \mathbb{N}$. We call the pair $\begin{pmatrix} p \\ v \end{pmatrix}$ the *chart* of the trajectory s.

Its worth noting that a trajectory $s : \mathbb{N} \to \text{States}_S$ in a deterministic system is
determined entirely by its start state s_0. This is what makes deterministic systems
deterministic; if you know the dynamics and you know what state the system is in, you
know how it will continue to behave. We'll relax this condition later in **??**.

Example 3.2.3. Consider the SIR model of Example 1.2.8. Suppose that we let our
parameters $(a, b) : \mathbb{N} \to \text{In}_{\text{SIR}}$ be constant at .2 and .3 respectively: that is, $a_t = .2$

and $b_t = .3$ for all t. Then a trajectory for SIR with parameters (a, b) is a sequence of populations $(s, i, r) : \mathbb{N} \to \text{State}_{\text{SIR}}$ such that

$$\begin{bmatrix} s_{t+1} \\ i_{t+1} \\ r_{t+1} \end{bmatrix} = \begin{bmatrix} s_t - .2 s_t i_t \\ i_t + .2 s_t i_t - .3 i_t \\ r_t + .3 i_t \end{bmatrix}$$

Here is an example of such a trajectory with a 1000 total people and one infected person to start, that is $(s_0, i_0, r_0) = (999, 1, 0)$.
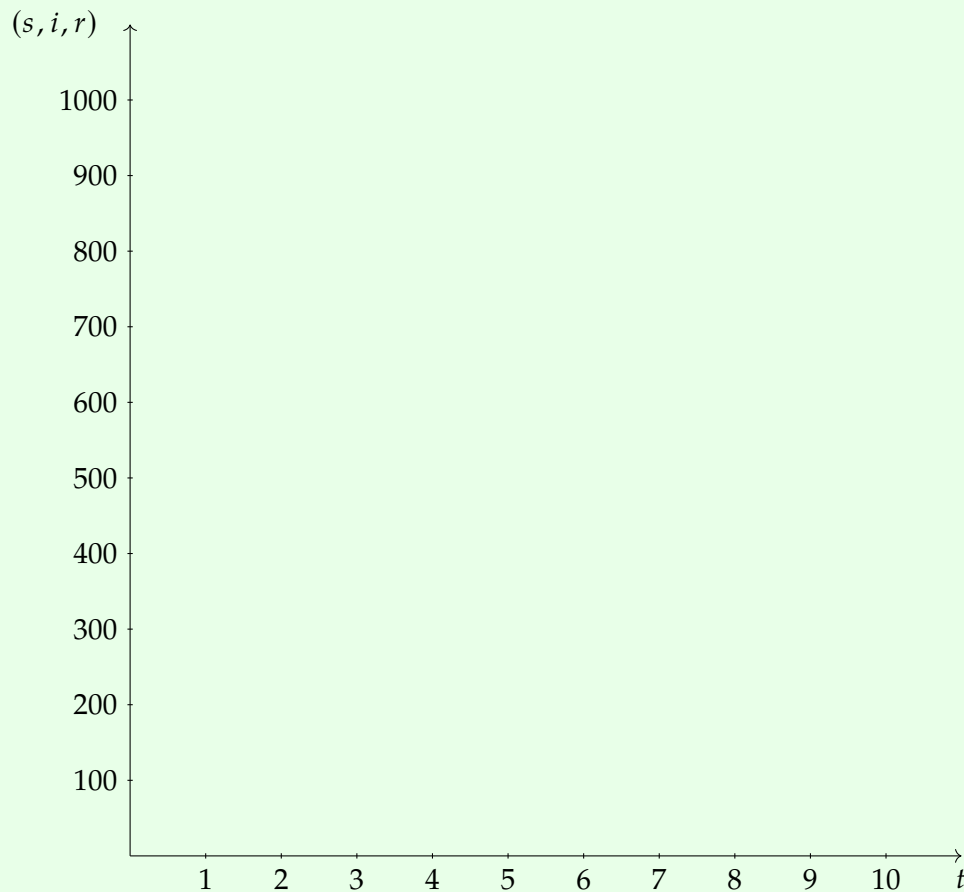
Jaz: I don't know how to actually plot this...

Example 3.2.4. If a deterministic system is written as a transition diagram, then the trajectories in the system are paths through the diagram. Recall from Example 1.2.9

this system:
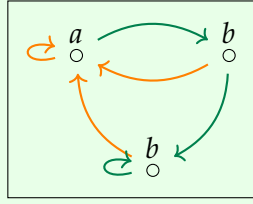
Suppose that $p : \mathbb{N} \to \{\texttt{green}, \texttt{orange}\}$ alternates between green and orange. Then starting at the top right state, a trajectory quickly settles into alternating between the top two states:
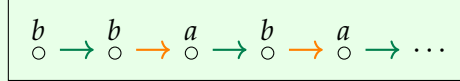
Knowing about trajectories can show us another important role that deterministic systems play: they are *stream transformers*. From a stream $p : \mathbb{N} \to \text{In}_\text{S}$ of inputs and a start state $s_0 \in \text{States}_\text{S}$, we get a trajectory $s : \mathbb{N} \to \text{States}_\text{S}$ given recursively by

$$s_{t+1} := \text{update}_\text{S}(s_t, p_t).$$

We then get a stream $v : \mathbb{N} \to \text{Out}_\text{S}$ of output values by defining

$$v_t := \text{expose}_\text{S}(s_t).$$

The system S is a way of transforming streams of input parameters into streams of output values.

Proposition 3.2.5 (Deterministic systems as stream transformers). Let

$$S = \begin{pmatrix} \text{update}_\text{S} \\ \text{expose}_\text{S} \end{pmatrix} : \begin{pmatrix} \text{States} \\ \text{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_\text{S} \\ \text{Out}_\text{S} \end{pmatrix}$$

be a deterministic system. Then for every $s_0 \in \text{States}_\text{S}$, we get a stream transformation function

$$\text{transform}_\text{S} : \text{In}_\text{S}^\mathbb{N} \to \text{Out}_\text{S}^\mathbb{N}$$

Given by

$$\text{transform}_\text{S}(p)_0 = \text{expose}_\text{S}(s_0)$$
$$\text{transform}_\text{S}(p)_{t+1} = \text{expose}_\text{S}(\text{update}_\text{S}(s_t, p_t))$$

where $s_{t+1} = \text{update}_\text{S}(s_t, p_t)$ is the trajectory given by s_0.

Exercise 3.2.6.   Say how the system of **??** acts as a stream transformer on the following streams:

1. $p_{2t} = \texttt{green}$ and $p_{2t+1} = \texttt{orange}$.

2. $p_t =$ green.

3. $p_0 =$ green and $p_t =$ orange for all $t > 0$. ◊

Later, in **??**, we will see that given trajectories of component systems, we get a trajectory of a whole wired system. Even better, every trajectory of the whole wired system can be calculated this way.

Trajectories in the differential doctrine

In a differential system, there is no "next" state after a given state. All we know is how each state is tending to change. So to define a trajectory in the differential doctrine, we can't just pick a state and see how it updates; instead, we are going to pick a state s_t for every time $t \in \mathbb{R}$ which are changing in the way described by the system.

Definition 3.2.7. Let

$$S = \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \text{State}_S \\ \text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$$

be a differential system. Suppose that $p : \mathbb{R} \to \text{In}_S$ is a differentiable choice of parameter for all times $t \in \mathbb{R}$. Then a *p-trajectory* is a differentiable function $s : \mathbb{R} \to \text{State}_S$ so that

$$\text{update}_S(s_t, p_t) = \frac{ds}{dt}(t).$$

for all $t \in \mathbb{R}$. Here, $\frac{ds}{dt}$ is the vector of derivatives $\frac{ds_i}{dt}$ for $i \in \{1, \ldots, n\}$ where n is the number of state variables.

If, additionally, $v : \mathbb{R} \to \text{Out}_S$ is a differentiable choice of outputs, then a $\begin{pmatrix} p \\ v \end{pmatrix}$-*trajectory* is a differentiable function $s : \mathbb{R} \to \text{State}_S$ so that

$$\text{update}_S(s_t, p_t) = \frac{ds}{dt}(t)$$

$$\text{expose}_S(s_t) = v_t$$

for all $t \in \mathbb{R}$. We call the pair $\begin{pmatrix} p \\ v \end{pmatrix}$ the *chart* of the trajectory s.

Remark 3.2.8. A *p*-trajectory of a differential system is also referred to as a *solution* of the differential equation it represents which choice of parameters p.

The definition of trajectory is what makes our differential systems actually describe differential equations. Consider the Lotka-Volterra preditor prey model from Section 1.2.15:

$$\begin{cases} \frac{dr}{dt} = \mathsf{b}_{\text{Rabbits}} \cdot r - c_1 f r \\ \frac{df}{dt} = c_2 r f - \mathsf{d}_{\text{Foxes}} \cdot f \end{cases} \tag{3.1}$$

Strictly speaking, this is not how we represent the system of differential equations as a differential system. Instead, we would describe its update function $\text{update}_{\text{LK}}$:

$\mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}^3$ as

$$\text{update}_{\text{LK}}\left(\begin{bmatrix} S \\ I \\ R \end{bmatrix}, \begin{bmatrix} \text{b}_{\text{Rabbits}} \\ \text{d}_{\text{Foxes}} \end{bmatrix}\right) := \begin{bmatrix} \text{b}_{\text{Rabbits}} \cdot r - c_1 f r \\ c_2 r f - \text{d}_{\text{Foxes}} \cdot f \end{bmatrix}$$

The differential equations **??** are the defining equations which make the function

$$t \mapsto \begin{bmatrix} S(t) \\ I(t) \\ R(t) \end{bmatrix} : \mathbb{R} \to \mathbb{R}^3$$

a $\begin{bmatrix} \text{b}_{\text{Rabbits}} \\ \text{d}_{\text{Foxes}} \end{bmatrix}$-trajectory. That is, we interpret a differential system $\left(\begin{smallmatrix} \text{update}_\text{S} \\ \text{expose}_\text{S} \end{smallmatrix}\right)$ as a system of differential equations by considering the equations which define what it means for a $s : \mathbb{R} \to \text{States}$ to be a trajectory.

Unlike deterministic systems, it is not necessarily the case that a state uniquely determines a trajectory through it for differentiable systems. This is the case, however, if the differential equations are linear.

Example 3.2.9. Consider the following variant of an SIR model proposed by Normal Bailey in [**BaileySIR**]:

$$\begin{cases} \frac{dS}{dt} &= \frac{-bSI}{S+I} \\ \frac{dI}{dt} &= \frac{bSI}{S+I} - bI \\ \frac{dR}{dt} &= bI \end{cases}$$

That is,

$$\text{update}_{\text{SIR}}\left(\begin{bmatrix} S \\ I \\ R \end{bmatrix}\right) = \begin{bmatrix} \frac{-bSI}{S+I} \\ \frac{bSI}{S+I} - bI \\ bI \end{bmatrix}.$$

We note that the total population $N = S + I + R$ will always be constant. Suppose, for simplicity, that b is a constant. Suppose that S_0 and I_0 are initial values for susceptible and infected populations respectively, and let $\kappa := \frac{I_0}{S_0}$. Then the function

$$\begin{bmatrix} S(t) \\ I(i) \\ R(t) \end{bmatrix} := \begin{bmatrix} S_0 e^{-\frac{b\kappa t}{1+\kappa}} \\ I_0 e^{-\frac{b\kappa t}{1+\kappa}} \\ N - (S_0 + I_0)e^{-\frac{b\kappa t}{1+\kappa}} \end{bmatrix}$$

will be a b-trajectory for SIR. This can be solved in greater generality, for variable parameter b and for two separate parameters governing the transition from susceptible to infected and infected to removed; see [**SIR_solved**].

Example 3.2.10. In this example, we will consider a simple RL-circuit:
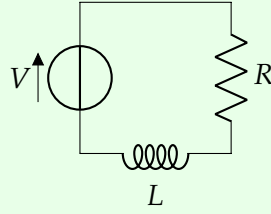
The voltage across the resistor is $V_R = IR$ while the voltage across the inductor is $V_L = L\frac{dI}{dt}$. By Kirchoff's voltage law, the total voltage differences, summed in an oriented manner, must be 0. Therefore, $-V + V_R + V_L = 0$, or, in terms of $\frac{dI}{dt}$:

$$\frac{dI}{dt} = \frac{V - RI}{L}.$$

We can express this RL-circuit as a differential system

$$\begin{pmatrix} \text{update}_{RL} \\ \text{id} \end{pmatrix} : \begin{pmatrix} \mathbb{R} \\ \mathbb{R} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^2 \times \mathbb{R}^* \\ \mathbb{R} \end{pmatrix}$$

where

$$\text{update}_{RL}\left(I, \begin{bmatrix} V \\ R \\ L \end{bmatrix} \right) := \frac{V - RI}{L}.$$

We can then see that $I : \mathbb{R} \to \mathbb{R}$ defined by

$$I(t) = \frac{V}{R}(1 - e^{-\frac{R}{L}t})$$

gives a $\begin{bmatrix} V \\ R \\ L \end{bmatrix}$-trajectory for the RL system.

3.2.11 Steady states

A steady state of a system is a state which does not change. Steady states are important because they are guarantees of stability: a vase in a steady state is doing great, a heart in a steady state is in need of attention.

Steady states in the deterministic doctrine

A steady state in the deterministic doctrine is a state which transitions to itself.

Definition 3.2.12. Let

$$S = \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \text{States}_S \\ \text{States}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$$

be a deterministic system. For input parameter $i \in \text{In}_S$ and output value $o \in \text{Out}_S$, an $\begin{pmatrix} i \\ o \end{pmatrix}$-*steady state* is a state $s \in \text{States}_S$ such that

$$\text{update}_S(s, i) = s,$$
$$\text{expose}_S(s) = o.$$

We call the pair $\begin{pmatrix} i \\ o \end{pmatrix}$ the *chart* of the steady state.

Remark 3.2.13. Its important to note that a steady state is relative to the input parameter chosen. For example, in Example 1.2.9, the top left state is steady for the input parameter orange but not for the input parameter green.

Unlike with trajectories, a system need not necessarily have *any* steady states. For example, the Clock has no steady states; it always keeps ticking to the next hour.

In the transition diagram of a finite deterministic system, steady states will be loops that begin and end at the same node. Since the system is finite, we can arrange the steady states by their chart into a $\text{In}_S \times \text{Out}_S$ matrix. For example, in Example 1.2.9, we get the following $\{\text{green}, \text{orange}\} \times \{a, b\}$ matrix:

$$\begin{matrix} & \text{green} & \text{orange} \\ a & \begin{bmatrix} \emptyset & \left\{ \rightleftarrows \begin{smallmatrix} a \\ \circ \end{smallmatrix} \right\} \\ b & \left\{ \rightleftarrows \begin{smallmatrix} b \\ \circ \end{smallmatrix} \right\} & \emptyset \end{bmatrix} \end{matrix} \tag{3.2}$$

This is a "matrix of sets", in that the entries are the actual sets of steady states. If we just counted how many steady states there were for each input-output pair, we would get this matrix:

$$\begin{matrix} & \text{green} & \text{orange} \\ a & \begin{bmatrix} 0 & 1 \\ b & 1 & 0 \end{bmatrix} \end{matrix} \tag{3.3}$$

In **??**, we'll see that each wiring diagram gives a formula for calculating the matrix of steady states of the composite system from the matrices of steady states of the inner systems.

Exercise 3.2.14.   What are the steady state matrices of systems S_1 and S_2 from **???** What about the combined system S?   ◊

Steady-looking trajectories. The reason we are interested in steady states is that they are highly predictable; if we know we are in a steady state, then we know we are always going to get the same results. But it is possible for us to always get the same outputs for the same input even though the internal state keeps changing. These are special trajectories, and we call them *steady-looking* trajectories.

Definition 3.2.15. For $i \in \text{In}_S$ and $o \in \text{Out}_S$ of a system S, a $\binom{i}{o}$-*steady looking trajectory* is a sequence of states $s : \mathbb{N} \to \text{States}_S$ such that

$$\text{update}_S(s_t, i) = s_{t+1}$$
$$\text{expose}_S(s_t) = o$$

for all $t \in \mathbb{N}$. We call the pair $\binom{i}{o}$ the *chart* of the steady-looking trajectory s.

Remark 3.2.16. While the steady states of a wired together system can be calculated from those of its components, this is not true for steady-looking trajectories. Intuitively, this is because the internal systems can be exposing changing outputs between eachother even while the eventual external output remains unchanged.

Exercise 3.2.17. Consider the wiring diagram:

$$\text{S} := \boxed{\;\boxed{S_1}\!-\!\boxed{S_2}\;}$$
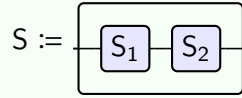
Find systems S_1 and S_2 and a steady-looking trajectory of the wired system S which is not steady-looking on the component systems. ◊

Steady states in the differential doctrine

A steady state in the differential doctrine is a state which has no tendency to change.

Definition 3.2.18. Let

$$S = \binom{\text{update}_S}{\text{expose}_S} : \binom{\text{States}_S}{\text{States}_S} \leftrightarrows \binom{\text{In}_S}{\text{Out}_S}$$

be a differential system. For input parameter $i \in \text{In}_S$ and output value $o \in \text{Out}_S$, an $\binom{i}{o}$-*steady state* is a state $s \in \text{States}_S$ such that

$$\text{update}_S(s, i) = 0,$$
$$\text{expose}_S(s) = o.$$

We call the pair $\left(\begin{smallmatrix} i \\ o \end{smallmatrix}\right)$ the *chart* of the steady state.

Example 3.2.19. Let's see if there are any steady states of the Lotka-Volterra predator prey model:

$$\text{update}_{\text{LK}}\left(\begin{bmatrix} r \\ f \end{bmatrix}, \begin{bmatrix} \text{b}_{\text{Rabbits}} \\ \text{d}_{\text{Foxes}} \end{bmatrix}\right) := \begin{bmatrix} \text{b}_{\text{Rabbits}} \cdot r - c_1 f r \\ c_2 r f - \text{d}_{\text{Foxes}} \cdot f \end{bmatrix}$$

We are looking for a state $\begin{bmatrix} r \\ f \end{bmatrix}$ whose update is 0. That is, we want to solve the system of equations

$$\begin{cases} 0 & = \text{b}_{\text{Rabbits}} \cdot r - c_1 r f \\ 0 & = c_2 r f - \text{d}_{\text{Foxes}} \cdot f \end{cases}$$

If the parameters $\begin{bmatrix} \text{b}_{\text{Rabbits}} \\ \text{d}_{\text{Foxes}} \end{bmatrix}$ are both zero, then any state is a steady state. Clearly, $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ is a steady state for any choice of parameters; this steady state could be called "extinction". But if the populations and parameters are non-zero, then

$$\begin{bmatrix} r \\ f \end{bmatrix} = \begin{bmatrix} \frac{\text{d}_{\text{Foxes}}}{c_2} \\ \frac{\text{b}_{\text{Rabbits}}}{c_1} \end{bmatrix}$$

is a steady state.

Example 3.2.20. Recall the RL circuit from **??**:

$$\text{update}_{\text{RL}}\left(I, \begin{bmatrix} V \\ R \\ L \end{bmatrix}\right) := \frac{V - RI}{L}.$$

We can see that $I := \frac{V}{R}$ is a steady state for this system given the parameters V and R.

3.2.21 Periodic orbits

Even if the behavior of a system isn't perfectly steady, it may continually repeat. To a reasonable approximation, the position of the earth around the sun follows a cycle that repeats every year. Using this as a paradigmatic example, we call these behaviors that repeat *periodic orbits*.

Periodic orbits in the deterministic doctrine

Definition 3.2.22 (Periodic orbit). A $\binom{p}{v}$-trajectory $s : \mathbb{N} \to \text{States}_S$ is *periodic* if there exists a time $t_0 \in \mathbb{N}_{\geq 1}$, called the *period*, such that $s_{t_0} = s_0$. If the sequence of parameters $p : \mathbb{N} \to \text{Ins}_S$ is also periodic with the same period (in that $p_{t_0} = p_0$ as well), then we say that s has *periodic parameters*.

Remark 3.2.23. Note that when we say that a periodic orbit has periodic parameters, we assume that they are periodic with the same period. This has important but subtle consequences for our theorems concerning the composition of behaviors in **??**. We explain the difference between a periodic orbit and a periodic orbit with periodic parameters in a more precise manner in **??**.

Remark 3.2.24. Note that a steady state is a periodic orbit (with periodic parameters) that has a period of 1.

Exercise 3.2.25. Describe a periodic orbit with period 1 that does not have periodic parameters; how are they different from steady states? Are there any of these in systems S_1 and S_2 of **???** ◊

Example 3.2.26. The Clock system is an exemplary periodic system with a period of 12. The ClockWithDisplay of Eq. (1.6) has period 24.

Exercise 3.2.27. What are the periodic orbits in the systems S_1 and S_2 of **??** with periodic parameters, and what are their periods? What about the combined system S? ◊

Exercise 3.2.28. Can you think of any periodic orbits in S_1 and S_2 of **??** which don't have periodic parameters? ◊

Periodic orbits in the differential doctrine

Definition 3.2.29. A p-trajectory $s : \mathbb{R} \to \text{States}_S$ for a differential system S is a *periodic orbit* if there is a number k such that

$$s(t) = s(t + k)$$

for all $t \in \mathbb{R}$. We refer to k as the *period* of the orbit s. If p is periodic of period k as well (that is, $p(t) = p(t + k)$ for all t), then we say that s has *periodic parameters*.

Example 3.2.30. Recall the Lotka-Volterra predator prey model of Section 1.2.15:

$$\begin{cases} \frac{dr}{dt} = a \cdot r - bfr \\ \frac{df}{dt} = crf - df \end{cases}$$

We may take the Jacobian of this system to get the "community matrix"

$$J(r, f) = \begin{pmatrix} a - bf & -br \\ cf & \\ cr - d & \end{pmatrix}.$$

We may investigate the stability of the steady states (from **??**) by looking at the Jacobian. In particular, we find that

$$J\left(\frac{d}{c}, \frac{a}{b}\right) = \begin{pmatrix} 0 & -\frac{bd}{c} \\ \frac{ac}{b} & 0 \end{pmatrix}$$

whose eigenvalues are $\pm i\sqrt{ad}$. Since the eigenvalues are purely imaginary and conjugate, this steady state is elliptic. Therefore the trajectories around this steady state are ellipses, which is to say, periodic.

Eventually Periodic Orbits   A trajectory might not get back to where it started, but may still end up being periodic. We call these trajectories *eventually* periodic orbits, since they eventually end up in a repeating cycle of states.

Definition 3.2.31 (Eventually periodic orbit). A $\binom{p}{v}$-trajectory $s : \mathbb{N} \to \text{State}_S$ is *eventually periodic* if there are times $t_0 < t_1 \in \mathbb{N}$ such that $s_{t_0+t} = s_{t_1+t}$ for all $t \in \mathbb{N}$. If the sequence of parameters $p : \mathbb{N} \to \text{In}_S$ is also eventually periodic with the same period (in that $p_{t_0+t} = p_{t_1+t}$ for all t), then we say that s has *eventually periodic parameters*.

   The *period* of an eventually periodic trajectory is the smallest difference $t_1 - t_0$ between times such that $s_{t_0} = s_{t_1}$.

Exercise 3.2.32.   Formulate an analogous definition of eventually periodic orbit in the differential doctrine.   ◊

3.3   Behaviors of systems in the deterministic doctrine

In the previous **??????**, we saw a number of different kinds of behaviors of dynamical systems. Not only were there a lot of definitions in those sections, each of those definitions had slight variants (like periodic orbits versus periodic orbits with periodic parameters, or steady states versus steady-looking trajectories). In this section, we'll define a general notion of behavior and see that we can package each of the above

sorts of behavior into a single system[2] in its own right, one that *represents* that sort of behavior. The representative system of a certain kind of behavior behaves in exactly that way, and does nothing else.

We will begin, for concreteness, with the deterministic doctrine. We will then return in the next section to see how we may formulate a general definition which also encompasses the differential doctrine.

We begin with a general definition of *chart*. A behavior is defined relative to its chart, which is the choice of parameters and the values of the variables it will expose. For example, the chart of a steady state was a parameter and an output value so that the state is steady with that parameter and it exposes that output value.

Definition 3.3.1. A *chart* $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \rightrightarrows \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ in a cartesian category $\mathcal{C}$ is a pair of maps $f : A^+ \to B^+$ and $f_\flat : A^+ \times A^- \to B^-$. Note that *this is not a lens*.

Exercise 3.3.2.
1. How many lenses are there $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} 3 \\ 2 \end{pmatrix} \leftrightarrows \begin{pmatrix} 4 \\ 3 \end{pmatrix}$?
2. How many charts are there $\begin{pmatrix} f^\flat \\ f \end{pmatrix} : \begin{pmatrix} 3 \\ 2 \end{pmatrix} \rightrightarrows \begin{pmatrix} 4 \\ 3 \end{pmatrix}$? ◊

Exercise 3.3.3.
1. Show that a chart $\begin{pmatrix} 1 \\ 1 \end{pmatrix} \rightrightarrows \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ is given by the data of a pair of elements $a^- \in A^-$ and $a^+ \in A^+$. Compare this to the notion of chart used in the definition of steady state (**??**).
2. Show that a chart $\begin{pmatrix} 1 \\ \mathbb{N} \end{pmatrix} \rightrightarrows \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ is given by the data of a sequence $a^- : \mathbb{N} \to A^+$ and a sequence $a^+ : \mathbb{N} \to A^+$. Compare this to the notion of chart used in the definition of trajectory (**??**) ◊

Definition 3.3.4 (Behavior of deterministic systems). Let T and S be deterministic systems. Given a chart of interfaces $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \rightrightarrows \begin{pmatrix} \mathsf{In_S} \\ \mathsf{Out_S} \end{pmatrix}$, a $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$-*behavior of shape* T *in* S, written $\phi : \mathsf{T} \to \mathsf{S}$, is a function $\phi : \mathsf{State_T} \to \mathsf{State_S}$ sending states of T to states of S which preserves the dynamics and exposed variables by satisfying the following equations:

$$\mathsf{expose_S}(\phi(t)) = f(\mathsf{expose_T}(t)),$$
$$\mathsf{update_S}(\phi(t), f_\flat(\mathsf{expose_T}(t), i)) = \phi(\mathsf{update_T}(t, i)) \tag{3.4}$$

for all $t \in \mathsf{State_T}$ and $i \in \mathsf{In_T}$. We say that $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$ is the chart of the behavior ϕ.

---

[2]Or a family of systems.

Remark 3.3.5. If you prefer commutative diagrams to systems of equations, don't fret. We'll reinterpret **??** in terms of commutative diagrams in **??**
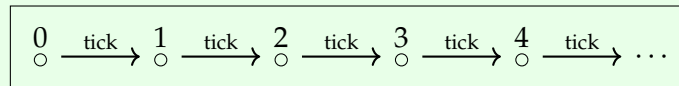
Remark 3.3.6. Suppose that we have transition diagrams for systems T and S. Then a behavior of shape T in S will correspond to part of the transition diagram of S which is shaped like the transition diagram of T. See the upcoming examples to see how this looks in practice.

Let's make this definition feel real with a few examples.

Example 3.3.7. Let Time be the system $\left(\begin{smallmatrix} t \mapsto t+1 \\ \mathrm{id} \end{smallmatrix} \right) : \left(\begin{smallmatrix} \mathbb{N} \\ \mathbb{N} \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} \{\text{tick}\} \\ \mathbb{N} \end{smallmatrix} \right)$, i.e. with

- $\mathsf{State}_{\mathsf{Time}} := \mathbb{N}$,
- $\mathsf{Out}_{\mathsf{Time}} := \mathbb{N}$,
- $\mathsf{In}_{\mathsf{Time}} := \{\text{tick}\}$,
- $\mathrm{expose}_{\mathsf{Time}} = \mathrm{id}$,
- $\mathrm{update}_{\mathsf{Time}}(t, *) = t + 1$.

As a transition diagram, Time looks like this:

$$0 \xrightarrow{\ \text{tick}\ } 1 \xrightarrow{\ \text{tick}\ } 2 \xrightarrow{\ \text{tick}\ } 3 \xrightarrow{\ \text{tick}\ } 4 \xrightarrow{\ \text{tick}\ } \cdots$$

Let's see what a behavior of shape Time in an arbitrary system S will be. We will expect the shape of Time to appear in the transition diagram of S, like this:
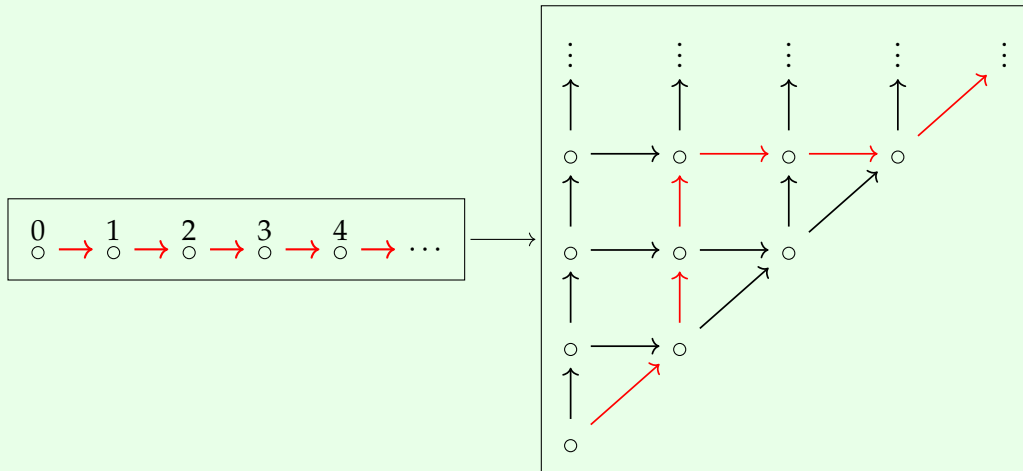
First, we need to know what a chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} \mathsf{In}_{\mathsf{Time}} \\ \mathsf{Out}_{\mathsf{Time}} \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} \mathsf{In}_{\mathsf{S}} \\ \mathsf{Out}_{\mathsf{S}} \end{smallmatrix} \right)$ is like. Since $\mathsf{Out}_{\mathsf{Time}} = \mathbb{N}$ and $\mathsf{In}_{\mathsf{Time}} \cong 1$, this means $f : \mathbb{N} \to \mathsf{Out}_{\mathsf{S}}$ is a sequence of outputs, and $f_\flat : \mathbb{N} \times 1 \to \mathsf{In}_{\mathsf{S}}$ is a sequence of input parameters. We might as well instead call f our sequence of exposed values v, and $f_\flat$ our sequence of input parameters p, so that we have a chart $\left(\begin{smallmatrix} p \\ v \end{smallmatrix} \right) : \left(\begin{smallmatrix} 1 \\ \mathbb{N} \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} \mathsf{In}_{\mathsf{S}} \\ \mathsf{Out}_{\mathsf{S}} \end{smallmatrix} \right)$.

Now, let's see what a $\binom{p}{v}$-behavior $\gamma : \text{Time} \to \text{S}$ is. It is a function $\gamma : \text{State}_{\text{Time}} \to \text{State}_{\text{S}}$ satsifying some properties. But $\text{State}_{\text{Time}} = \mathbb{N}$, so $\gamma : \mathbb{N} \to \text{State}_{\text{S}}$ is a sequence of states in S. Now, **??** becomes the equations:
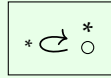
$$\text{expose}_{\text{S}}(\gamma(t)) = v(t)$$

$$\text{update}_{\text{S}}(\gamma(t), p(t)) = \gamma(t + 1).$$

which are exactly the equations defining a $\binom{p}{v}$-trajectory from **??**!

Example 3.3.8. Consider the simple system Fix with:
- $\text{State}_{\text{Fix}} = \{*\}$.
- $\text{Out}_{\text{Fix}} = \{*\}$.
- $\text{In}_{\text{Fix}} = \{*\}$.
- $\text{expose}_{\text{Fix}} = \text{id}$.
- $\text{update}_{\text{Fix}}(*, *) = *$.

As a transition diagram, this looks like:

A behavior $s : \text{Fix} \to \text{S}$ in an arbitrary system S should be a loop of this shape within the transition diagram of S: a steady state.

Let's check that this works. First, we need to know what a chart $\binom{f_\flat}{f} : \binom{\text{In}_{\text{Fix}}}{\text{Out}_{\text{Fix}}} \rightrightarrows \binom{\text{In}_{\text{S}}}{\text{Out}_{\text{S}}}$ is. Since $\text{Out}_{\text{Fix}} = \text{In}_{\text{Fix}} = \{*\}$, we have that $f : \{*\} \to \text{Out}_{\text{S}}$ is simply an output value of S and $f_\flat : \{*\} \times \{*\} \to \text{In}_{\text{S}}$ is simply an input parameter. Therefore, we might as well write o for f and i for $f_\flat$, to see that a chart $\binom{i}{o} : \binom{\{*\}}{\{*\}} \rightrightarrows \binom{\text{In}_{\text{S}}}{\text{Out}_{\text{S}}}$ is a pair of elements $i \in \text{In}_{\text{S}}$ and $o \in \text{Out}_{\text{S}}$.

Now, let's see what a $\binom{i}{o}$-behavior $s : \text{Fix} \to \text{S}$ is. It is a function $s : \text{State}_{\text{Fix}} \to \text{State}_{\text{S}}$ satisfying a few properties. But $\text{State}_{\text{S}} = \{*\}$ so $s : \{*\} \to \text{State}_{\text{S}}$ is a single state of S.

Then, **??** becomes the equations

$$\text{expose}_S(s) = o$$
$$\text{update}_S(s, i) = s$$

which are precisely the equations defining a $\begin{pmatrix} i \\ o \end{pmatrix}$-steady state from **??**.

Example 3.3.9. Let $0 < n \in \mathbb{N}$ be a positive natural number, and consider the system Clock_n having:

- $\text{State}_{\text{Clock}_n} = n = \{1, \ldots, n\}$.
- $\text{Out}_{\text{Clock}_n} = n$.
- $\text{In}_{\text{Clock}_n} = \{*\}$.
- $\text{expose}_{\text{Clock}_n} = \text{id}$.
- $\text{update}_{\text{Clock}_n}(t, *) = \begin{cases} t + 1 & \text{if } t < n \\ 1 & \text{if } t = n \end{cases}$.

This is the clock with n hours. Our example system Clock from Example 1.2.5 is Clock_{12}, a clock with 12 hours. Here's what Clock_4 looks like as a transition diagram:
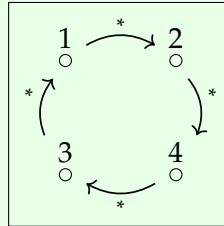
A behavior $\gamma : \text{Clock}_n \to S$ should be a cycle like this in the transition diagram of S: a periodic orbit. We can see the Clock_4-behavior inside the system shown right:
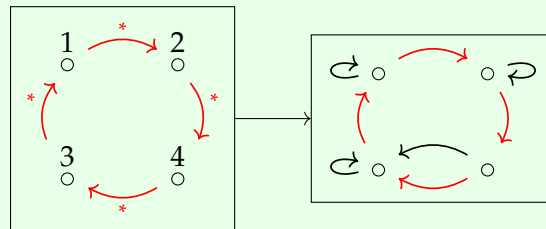
Let's check that this works. First, we need to know what a chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} \text{In}_{\text{Clock}_n} \\ \text{Out}_{\text{Clock}_n} \end{pmatrix} \rightrightarrows$ $\begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$ is. Since $\text{Out}_{\text{Clock}_n} = n$ and $\text{In}_{\text{Clock}_n} = \{*\}$, $f : n \to \text{Out}_S$ is a sequence of n exposed values of S while $f_\flat : n \times \{*\} \to \text{In}_S$ is a sequence of n parameters. Therefore, we might as well write v for f and p for $f_\flat$ to find that a chart $\begin{pmatrix} p \\ v \end{pmatrix} : \begin{pmatrix} \{*\} \\ n \end{pmatrix} \rightrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$ consists of an n-length sequence of parameters and an n-length sequence of exposed values.

A $\begin{pmatrix} p \\ v \end{pmatrix}$-behavior $\gamma : \mathsf{Clock}_n \to \mathsf{S}$, then, is a function $\gamma : \mathsf{State}_{\mathsf{Clock}_n} \to \mathsf{State}_{\mathsf{S}}$ satisfying a few properties. Since $\mathsf{State}_{\mathsf{Clock}_n} = n$, $\gamma : n \to \mathsf{State}_{\mathsf{S}}$ is a n-length sequence of states of S, and **??** become the equations

$$\mathrm{expose}_{\mathsf{S}}(\gamma(t)) = v(t)$$

$$\mathrm{update}_{\mathsf{S}}(\gamma(t), p(t)) = \begin{cases} \gamma(t+1) & \text{if } t < n \\ \gamma(1) & \text{if } t = n \end{cases}.$$

As we can see, this determines a sequence of length n of states of S which repeats when it gets to the end. In other words, this is a periodic orbit with periodic parameters as in **??**!

If we have a certain kind of behavior in mind, and we find a system T so that behaviors of shape T are precisely this kind of behavior, then we say that T *represents* that behavior. For example, we have just seen that:

- The system $\mathsf{Time} = \begin{pmatrix} +1 \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} \mathbb{N} \\ \mathbb{N} \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ \mathbb{N} \end{pmatrix}$ represents trajectories.
- The system $\mathsf{Fix} = \begin{pmatrix} \pi_2 \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} \{*\} \\ \{*\} \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ \{*\} \end{pmatrix}$ represents steady states.
- The systems $\mathsf{Clock}_n = \begin{pmatrix} +1 \bmod n \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} n \\ n \end{pmatrix} \leftrightarrows \begin{pmatrix} \{*\} \\ n \end{pmatrix}$ represents periodic orbits with periodic parameters whose period divides n.

Note that there is always a particularly simple behavior on a system: the identity behaviors id : $\mathsf{State}_\mathsf{T} \to \mathsf{State}_\mathsf{T}$. This says that every system behaves as itself. In particular, Time has a trajectory behavior given by id : Time $\to$ Time (namely, the trajectory $s_t = t$), and Fix has a steady state behavior given by id : Fix $\to$ Fix (namely, the steady state $*$), etc. We refer to the identity behavior of T as the *generic* behavior of type T.

Exercise 3.3.10.   Find a representative system for the following kinds of behavior.
1. An eventually periodic orbit (see **??**) that takes n steps to get to a period of size m.
2. A steady-looking trajectory (see **??**).
3. A periodic orbit of period at most n whose parameters aren't necessarily also periodic (see **??**).
4. An trajectory which yields the same output value at every 10th step.   ◊

Remark 3.3.11. As **??** shows, the difference between a periodic orbit and a periodic orbit with periodic parameters can be surmised precisely by noting that they are represented by systems with different interfaces. The dynamics of the systems are the same, but the interfaces (and accordingly, the exposed variable) are different; this explains how the difference between a periodic orbit and a periodic orbit with periodic parameters is all in the chart.

Exercise 3.3.12.   What kind of behaviors do the following systems represent? First, figure out what kind of charts they have, and then see what a behavior with a given chart is. Describe in your own words.

1. The system Plus with:
   - $\text{State}_{\text{Plus}} = \mathbb{N}$.
   - $\text{Out}_{\text{Plus}} = \mathbb{N}$.
   - $\text{In}_{\text{Plus}} = \mathbb{N}$.
   - $\text{expose}_{\text{Plus}} = \text{id}$.
   - $\text{update}_{\text{Plus}}(t, j) = t + j$.

2. The system T_n with:
   - $\text{State}_{\text{T}_n} = \mathbb{N}$.
   - $\text{Out}_{\text{T}_n} = \{0, \dots, n - 1\}$.
   - $\text{In}_{\text{T}_n} = \{*\}$.
   - $\text{expose}_{\text{T}_n}(t) = t \mod n$.
   - $\text{update}_{\text{T}_n}(t, *) = t + 1$.

3. The system XOR with:
   - $\text{State}_{\text{XOR}} = \text{Bool} = \{\texttt{true}, \texttt{false}\}$.
   - $\text{Out}_{\text{XOR}} = \text{Bool}$.
   - $\text{In}_{\text{XOR}} = \text{Bool}$.
   - $\text{expose}_{\text{XOR}} = \text{id}$.
   -
     $$\begin{aligned}
     \text{update}_{\text{XOR}}(\texttt{true}, \texttt{true}) &= \texttt{false}, \\
     \text{update}_{\text{XOR}}(\texttt{false}, \texttt{true}) &= \texttt{true}, \\
     \text{update}_{\text{XOR}}(\texttt{true}, \texttt{false}) &= \texttt{true}, \\
     \text{update}_{\text{XOR}}(\texttt{false}, \texttt{false}) &= \texttt{false}.
     \end{aligned}$$

4. The system List_C for a set of *choices* C with:
   - $\text{State}_{\text{List}_C} = \text{List}_C$ is the set of lists of elements in C.
   - $\text{Out}_{\text{List}_C} = \text{List}_C$.
   - $\text{In}_{\text{List}_C} = C$.
   - $\text{expose}_{\text{List}_C} = \text{id}$.
   - $\text{update}_{\text{List}_C}(\ell, c) = c :: \ell$, that is, we update a list by appending the character $c \in C$ to the start.

$\diamond$

While every system T represents some kind of behavior — just take the kind of behavior to be exactly described by behaviors $\text{T} \to \text{S}$ — we are most interested in those simple systems T whose behavior we can fully understand.

We have written a behavior of shape T in S with an arrow $\phi : \text{T} \to \text{S}$. This suggests that there is a category with deterministic systems as its objects and behaviors as its morphisms; and there is!

Definition 3.3.13. The category $\mathbf{Chart}_{\mathcal{C}}$ of charts in $\mathcal{C}$ has

- Objects the *arenas* $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$, pairs of objects in $\mathcal{C}$.

- Maps the *charts* $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$.

- Composition the composite of a chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ with a chart $\begin{pmatrix} g_\flat \\ g \end{pmatrix} :$
  $\begin{pmatrix} B^- \\ B^+ \end{pmatrix} \Rightarrow \begin{pmatrix} C^- \\ C^+ \end{pmatrix}$ is

$$
\begin{pmatrix} g_\flat \\ g \end{pmatrix} \circ \begin{pmatrix} f_\flat \\ f \end{pmatrix} := \begin{pmatrix} (a^+, a^-) \mapsto g_\flat(f(a^+), f_\flat(a^+, a^-)) \\ g \circ f \end{pmatrix}.
$$

- The identity chart is $\begin{pmatrix} \pi_2 \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$.

Exercise 3.3.14. Check that $\mathbf{Chart}_{\mathcal{C}}$ is indeed a category. That is,

1. For charts $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$, $\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \Rightarrow \begin{pmatrix} C^- \\ C^+ \end{pmatrix}$, and $\begin{pmatrix} h_\flat \\ h \end{pmatrix} : \begin{pmatrix} C^- \\ C^+ \end{pmatrix} \Rightarrow \begin{pmatrix} D^- \\ D^+ \end{pmatrix}$,
   show that

$$
\begin{pmatrix} h_\flat \\ h \end{pmatrix} \circ \left(\begin{pmatrix} g_\flat \\ g \end{pmatrix} \circ \begin{pmatrix} f_\flat \\ f \end{pmatrix} \right) = \left(\begin{pmatrix} h_\flat \\ h \end{pmatrix} \circ \begin{pmatrix} g_\flat \\ g \end{pmatrix} \right) \circ \begin{pmatrix} f_\flat \\ f \end{pmatrix}.
$$

2. For a chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$, show that

$$
\begin{pmatrix} \pi_2 \\ \mathrm{id} \end{pmatrix} \circ \begin{pmatrix} f_\flat \\ f \end{pmatrix} = \begin{pmatrix} f_\flat \\ f \end{pmatrix} = \begin{pmatrix} f_\flat \\ f \end{pmatrix} \circ \begin{pmatrix} \pi_2 \\ \mathrm{id} \end{pmatrix}.
$$

◇

Exercise 3.3.15. What are the charts of the following forms in simpler terms?

1. $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} 1 \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$.

2. $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

3. $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} 1 \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$.

◇

Proposition 3.3.16. There is a category $\mathbf{Sys}$ with deterministic systems as its objects and where a map $\mathsf{T} \to \mathsf{S}$ is a pair consisting of a chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix} \Rightarrow \begin{pmatrix} \mathrm{In}_\mathsf{S} \\ \mathrm{Out}_\mathsf{S} \end{pmatrix}$ and a $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$-behavior $\phi : \mathsf{T} \to \mathsf{S}$. Composition is given by composing both the charts and the functions on states, and identities are given by the generic behaviors: the identity chart with the identity function $\mathrm{id} : \mathrm{State}_\mathsf{T} \to \mathrm{State}_\mathsf{T}$.

Proof. We just need to check that the composite $\psi \circ \phi$ of two behaviors $\phi : \mathsf{T} \to \mathsf{S}$ and $\psi : \mathsf{S} \to \mathsf{U}$ with charts $\binom{f_\flat}{f} : \binom{\mathsf{In}_\mathsf{T}}{\mathsf{Out}_\mathsf{T}} \Rightarrow \binom{\mathsf{In}_\mathsf{S}}{\mathsf{Out}_\mathsf{S}}$ and $\binom{g_\flat}{g} : \binom{\mathsf{In}_\mathsf{S}}{\mathsf{Out}_\mathsf{S}} \Rightarrow \binom{\mathsf{In}_\mathsf{U}}{\mathsf{Out}_\mathsf{U}}$ is a behavior with chart $\binom{g_\flat}{g} \circ \binom{f_\flat}{f}$. That is, we need to check that **??** is satisfied for $\psi \circ \phi$. We can do this using the fact that it is satisfied for both ψ and ϕ.

$$\mathrm{expose}_\mathsf{U}(\psi(\phi(t))) = \psi(\mathrm{expose}_\mathsf{S}(\phi(t)))$$
$$= \psi(\phi(\mathrm{expose}_\mathsf{T}(t))).$$

$$\mathrm{update}_\mathsf{U}(\psi(\phi(t)), g_\flat(f(\mathrm{expose}_\mathsf{T}(t)), f_\flat(\mathrm{expose}_\mathsf{T}(t), i)))$$
$$= \mathrm{update}_\mathsf{U}(\psi(\phi(t)), g_\flat(\mathrm{expose}_\mathsf{S}(\phi(t)), f_\flat(\mathrm{expose}_\mathsf{T}(t), i)))$$
$$= \psi(\mathrm{update}_\mathsf{S}(\phi(t), f_\flat(\mathrm{expose}_\mathsf{T}(t), i)))$$
$$= \psi(\phi(\mathrm{update}_\mathsf{T}(t, i))). \qquad \qquad \square$$

There are two different ways to understand what composition of behaviors means: one based on post-composition, and the other based on pre-composition.

- We see that any behavior $\mathsf{S} \to \mathsf{U}$ gives a way of turning T-shaped behaviors in S to T-shaped behaviors in U.
- We see that any behavior $\mathsf{T} \to \mathsf{S}$ gives a way of turning S-shaped behaviors in U into T-shaped behaviors in U.

Example 3.3.17. Any steady state s can be seen as a particularly simple trajectory: $s_t = s$ for all t. We have seen in **??** that steady states are Fix-shaped behaviors. We can use composition of behaviors to understand how steady states give rise to trajectories.

The generic steady state $*$ of Fix (that is, the identity behavior of Fix) generates a trajectory $s : \mathbb{N} \to \mathsf{State}_{\mathsf{Fix}}$ with input parameters $p_t = *$ and $s_t = *$. This gives us a behavior $s : \mathsf{Time} \to \mathsf{Fix}$.

Now, for every steady state $\gamma : \mathsf{Fix} \to \mathsf{S}$, we may compose to get a trajectory $\gamma \circ s : \mathsf{Time} \to \mathsf{S}$.

Exercise 3.3.18.   Adapt the argument of **??** to show that
1. Any eventually periodic orbit gives rise to a trajectory.
2. If n divides m, then any orbit of period at most n gives rise to an orbit of period of most m.   ◊

Isomorphisms of Systems   Now that we have a category of systems and behaviors, category theory supplies us with a definition of isomorphism for systems.

Definition 3.3.19. An *isomorphism* of a system T with a system S is a a behavior $\phi :$ $\mathsf{T} \to \mathsf{S}$ for which there is another behavior $\phi^{-1} : \mathsf{S} \to \mathsf{T}$ such that $\phi \circ \phi^{-1} = \mathrm{id}_{\mathsf{S}}$ and $\phi^{-1} \circ \phi = \mathrm{id}_{\mathsf{T}}$.

Let's see that this is indeed a good notion of sameness for systems.

Proposition 3.3.20. A behavior $\phi : \mathsf{T} \to \mathsf{S}$ is an isomorphism if and only if the following conditions hold:
1. The map $\phi : \mathrm{State}_{\mathsf{T}} \to \mathrm{State}_{\mathsf{S}}$ is an isomorphism of sets — a bijection.
2. The chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} \mathrm{In}_{\mathsf{T}} \\ \mathrm{Out}_{\mathsf{T}} \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} \mathrm{In}_{\mathsf{S}} \\ \mathrm{Out}_{\mathsf{S}} \end{smallmatrix} \right)$ of ϕ is an isomorphism in **Chart$_{\mathbf{Set}}$**. That is, $f : \mathrm{Out}_{\mathsf{T}} \to \mathrm{Out}_{\mathsf{S}}$ is a bijection and there is a bijection $f'_\flat : \mathrm{In}_{\mathsf{T}} \to \mathrm{Out}_{\mathsf{T}}$ such that $f_\flat = f'_\flat \circ \pi_2$.

Proof. Since composition in the category of systems and behaviors is given by composition of the underlying charts and maps, ϕ is an isomorphism of systems if and only if its action on states is a bijection and its chart is an isomorphism in the category of charts. It just remains to see that our description of isomorphism of charts is accurate, which we leave to **??**. $\square$

Exercise 3.3.21. Show that a chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)$ is an isomorphism if and only if f is an isomorphisms and there is an isomorphism $f'_\flat : A^- \to B^-$ such that $f_\flat = f'_\flat \circ \pi_2$. $\diamond$

3.3.22 Simulations

While we will often be interested in behaviors of systems that change the interface in the sense of having non-trivial charts, we will also be interested in behaviors of systems that do not changed the exposed variables at all. These behaviors play a very different role in the theory of dynamical systems than behaviors like trajectories and steady states. Because they don't change observable behavior (since they have identity chart), they say more about how we *model* the observable behavior than what that behavior is itself. For that reason, we will call behaviors with identity chart *simulations*.

Definition 3.3.23. Let $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ be an arena. The category

$$\mathbf{Sys} \begin{pmatrix} I \\ O \end{pmatrix}$$

of deterministic $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$-systems has as objects the systems $\left(\begin{smallmatrix} \mathrm{update}_{\mathsf{S}} \\ \mathrm{expose}_{\mathsf{S}} \end{smallmatrix} \right) : \left(\begin{smallmatrix} \mathrm{State}_{\mathsf{S}} \\ \mathrm{State}_{\mathsf{S}} \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ with interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ and as maps the *simulations* $\phi : \mathsf{T} \to \mathsf{S}$, those behaviors whose chart

is the identity chart on $\begin{pmatrix} I \\ O \end{pmatrix}$.

Example 3.3.24. Recall the $\begin{pmatrix} \{\texttt{green},\texttt{orange}\} \\ \{a,b\} \end{pmatrix}$-system S from Example 1.2.9:
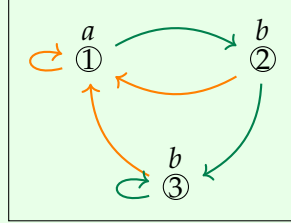
If we had built this system as a model of some relationships between input colors and output letters we were seeing in the wild, then we have made this system a bit redundant. If the output is a, and we feed it green, the output will be b; if we feed it orange, the output will be a. Similarly, if the output is b — no matter which of states 2 or 3 the system is actually in — and we feed it green, the output will again be b, and if we feed it orange, the output will be a. And there really isn't much else going on in the system.

We can package this observation into a behavior in $\mathbf{Sys}\left(\begin{smallmatrix} \{\texttt{green},\texttt{orange}\} \\ \{a,b\} \end{smallmatrix} \right)$. Let U be the system
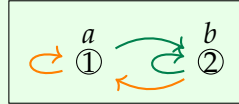
We can give a behavior $q : \mathsf{S} \to \mathsf{U}$ with identity chart as follows defined by

$$q(1) = 1$$
$$q(2) = 2$$
$$q(3) = 2$$

We can check, by cases, that this is indeed a behavior. That it is a behavior in $\mathbf{Sys}\left(\begin{smallmatrix} \{\texttt{green},\texttt{orange}\} \\ \{a,b\} \end{smallmatrix} \right)$ means that it doesn't change the observable behavior.

?? also gives us an example of an important relation between systems: *bisimulation*. We saw what it means for two systems to be isomorphic: it means they have isomorphic states and the same dynamics and output relative to those isomorphisms. But this is sometimes too strong a notion of sameness for systems; we want to know when two systems *look the same* on the outside.

Let's see what this notion looks like for deterministic systems; then we will describe it in a doctrinal way.

Definition 3.3.25. In the deterministic doctrine, a *bisimulation* $\sim$ between $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-systems S and U is a relation $\sim$: $\mathsf{States} \times \mathsf{State}_U \to \{\texttt{true}, \texttt{false}\}$ between states of these systems such that $s \sim u$ only when s and u have related dynamics:

$$s \sim u \text{ implies expose}_S(s) = \text{expose}_U(u)$$
$$s \sim u \text{ implies update}_S(s, i) \sim \text{update}_U(u, i) \text{ for all } i \in I.$$

If $\sim$ is a bisimulation, we say that s and u are *bisimilar* when $s \sim u$.

A bisimulation $\sim$ is said to be *total* if every $s \in \mathsf{States}$ is bisimilar to some $u \in \mathsf{State}_U$ and vice-versa.

Bisimilarity is a strong relation between states of systems. For deterministic systems, this implies that they act the same on any input.

Proposition 3.3.26. Let S and U be deterministic $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-systems, and let $\sim$ be a bisimulation between them. If $s_0 \sim u_0$ are bisimilar, then they induce the same transformation on streams of inputs into streams of outputs:

$$\text{transform}_S^{s_0} = \text{transform}_U^{u_0}.$$

Proof. Let $i : \mathbb{N} \to I$ be a stream of inputs. Let $s : \mathbb{N} \to \mathsf{States}$ be the stream of states generated by s_0 and similarly, let $u : \mathbb{N} \to \mathsf{State}_U$ be the stream of states generated by u_0.

We first show that $s_n \sim u_n$ for all n. Our base case holds by hypothesis; now suppose that $s_n \sim u_n$ seeking $s_{n+1} \sim u_{n+1}$. Well,

$$s_{n+1} = \text{update}_S(s_n, i_n) \sim \text{update}_U(u_n, i_n) = u_{n+1}$$

because $\sim$ is a bisimulation.

Finally,

$$\text{transform}_S(i)_n = \text{expose}_S(s_n) = \text{expose}_U(u_n) = \text{transform}_U(i)_n$$

because $s_n \sim u_n$. $\qquad\square$

We can talk about bisimilar states without reference to the particular bisimulation between the systems they are a part of because, as it turns out, being bisimilar is independent of the particular bisimulation. To see this, we need to introduce an interesting system: the system of trees.

Definition 3.3.27. Let $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ be an arena in the deterministic doctrine. An $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-tree τ (or a O-labeled, I-branching tree) consists of:

- A *root* $\text{root}(\tau) \in O$.
- For each parameter $i \in I$, a *child* tree $\text{child}(\tau, i)$.

Definition 3.3.28. Let $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ be an arena in the deterministic doctrine. The $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-system $\text{Tree}_{\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)}$ of $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-trees has

- $\text{State}_{\text{Tree}}$ is the set of $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-trees.
- Each tree exposes its root: $\text{expose}_{\text{Tree}}(\tau) = \text{root}(\tau)$.
- The system updates by following a tree down the ith branch: $\text{update}_{\text{Tree}}(\tau, i) = \text{child}(\tau, i)$

We can think of an $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-tree as a stream of possible outputs of an $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-system. In the current state, we see the root of the tree. When we transition to the next state with parameter i, we will see the rest of the output. This observation suggests a universal characterization of the system of $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-trees.

Proposition 3.3.29. The $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-system $\text{Tree}_{\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)}$ of $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-trees is terminal in the category of $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-systems.

Proof. We will show that there is a unique simulation $!_S : S \to \text{Tree}_{\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)}$ for any $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-system S. For any $s \in \text{State}_S$, we will define a tree $!_S(s)$ of outputs visible from the state s. We define this as follows:

- The root of $!_S(s)$ is the variable exposed by S:

$$\text{root}(!_S(s)) = \text{expose}_S(s).$$

- The ith child of $!_S(s)$ is the tree of outputs visible from the next state $\text{update}_S(s, i)$:

$$\text{child}(!_S(s), i) = !_S(\text{update}_S(s, i)).$$

Now, we can show that this is a simulation *and* that it is the unique such similation by noticing that this definition is precisely what is required to satisfy the defining laws of a simulation.   □

Now we can express the idea that bisimilarity of states is independent of any particular bisimulation between their systems with the following theorem.

Theorem 3.3.30. Let S and U be $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-systems. A state $s \in \text{State}_S$ is bisimilar to a state $u \in \text{State}_U$ for some bisimulation $\sim$ between S and U if and only if $!_S(s) = !_U(u)$.

Proof. First, let's show that if s is bisimilar to u via some bisimulation $\sim$, then $!_S(s) = !_U(u)$. Now, to show that two trees are equal, we need to show that they have equal roots and equal children.

- The root of $!_S(s)$ is $\text{expose}_S(s)$, and the root of $!_U(u)$ is $\text{expose}_U(u)$. But since $s \sim u$ by hypothesis, these are equal.
- Similarly, the ith child of $!_S(s)$ is $!_S(\text{update}_S(s, i))$, while the ith child of $!_U(u)$ is $!_U(\text{update}_U(u, i))$. But since $\sim$ is a bisimulation, we have that $\text{update}_S(s, i) \sim \text{update}_U(u, i)$, and so by the same argument we are giving, we will find that $!_S(\text{update}_S(s, i)) = !_U(\text{update}_U(u, i))$.[3]

On the other hand, suppose that $!_S(s) = !_U(u)$. We now need to defined a bisimulation $\sim$ between S and U for which $s \sim u$. For any sequence of inputs $i : \mathsf{n} \to I$, we can evolve a system in state s by the entire sequence i to yield a state $\text{update}_S^*(s, i)$ in the following way:

- If $n = 0$, then $\text{update}_S^*(s, i) = s$.
- For $n + 1$, then $\text{update}_S^*(s, i) = \text{update}_S(\text{update}_S^*(s, i|_\mathsf{n}), i_{n+1})$.

We may then define $\sim$ in the following way.

$$x \sim y \text{ if and only if there is an } n \in \mathbb{N} \text{ and } i : \mathsf{n} \to I \text{ with } x = \text{update}_S^*(s, i) \text{ and}$$
$$y = \text{update}_U^*(u, i).$$

It remains to show that this is a bisimulation.

For any $\binom{I}{O}$-tree τ and any n-length sequence $i : \mathsf{n} \to I$ of parameter (for any $n \in \mathbb{N}$), we can follow the path i through the tree τ to get a new tree $\text{subtree}(\tau, i)$:

- If $n = 0$, then $\text{subtree}(\tau, i) = \tau$.
- For $n + 1$, $\text{subtree}(\tau, i) = \text{child}(\text{subtree}(\tau, i|_\mathsf{n}))$ is the ith child of the tree found by following i for the first n steps.

Note that $_S(\text{update}_S^*(s, i)) = \text{subtree}(!_S(s), i)$ by a quick inductive argument. Now we can show that $\sim$ is a bisimulation.

- Suppose that $x \sim y$, seeking to show that $\text{expose}_S(x) = \text{expose}_U(y)$. By hypothesis, $x = \text{update}_S^*(s, i)$ and $y = \text{update}_U^*(u, i)$. But then

$$\begin{aligned}
\text{expose}_S(x) &= \text{root}(!_S(x)) \\
&= \text{root}(\text{subtree}(!_S(s), i)) \\
&= \text{root}(\text{subtree}(!_U(u), i)) \\
&= \text{root}(!_U(y)) \\
&= \text{expose}_U(y).
\end{aligned}$$

- Suppose that $x \sim y$, seeking to show that $\text{update}_S(x, j) \sim \text{update}_U(y, j)$. By hypothesis, $x = \text{update}_S^*(s, i)$ and same for $y = \text{update}_U^*(u, i)$. Then letting i' :

---

[3]This style of proof is called *proof by co-induction*. Where induction assumes a base case and then breaks apart the next step into a smaller step, co-induction shows that the proof can always be continued in a manner which covers all possible options.

n + 1 → $\mathbb{N}$ be defined by $i'_{n+1} = j$ and $i'_k = i_k$ otherwise, we see that $\text{update}_S(x, j) = \text{update}^*_S(s, i')$ and $\text{update}_U(y, j) = \text{update}^*_U(y, i')$, so that by definition they are related by ∼.

<div align="right">□</div>

3.4   Dealing with two kinds of composition: Double categories

In this section, we will introduce the notion of *double category* to help us deal with our two kinds of composition: the composition of systems, and the composition of behaviors. By revealing that **??** can be expressed as a square in a double category of *arenas*, we will find a generalization of this definition of behavior which applies to the differential doctrine as well. It is at this point that we will introduce the formal definition of a *dynamical system doctrine*.

Definition 3.4.1. A *double category* $\mathcal{D}$ has:
- A class ob$\mathcal{D}$ of *objects*.
- A *horizontal* category $h\mathcal{D}$ whose objects are those of $\mathcal{D}$. We call the maps in $h\mathcal{D}$ the *horizontal* maps of $\mathcal{D}$.
- A *vertical* category $v\mathcal{D}$ whose objects are those of $\mathcal{D}$. We call the maps in $v\mathcal{D}$ the *vertical* maps of $\mathcal{D}$.
- For vertical maps $j : A \to B$ and $k : C \to D$ and horizontal maps $f : A \to C$ and $g : B \to D$, there is a set of *squares*

$$
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
{\scriptstyle j}\downarrow & \alpha & \downarrow{\scriptstyle k} \\
C & \xrightarrow[\ g\ ]{} & D
\end{array}
$$

- Squares can be composed both horizontally and vertically:

$$
\begin{array}{ccccc}
A_1 & \xrightarrow{\ f_1\ } & A_2 & \xrightarrow{\ f_2\ } & A_3 \\
{\scriptstyle j}\downarrow & \alpha & {\scriptstyle k}\downarrow & \beta & \downarrow{\scriptstyle \ell} \\
B_1 & \xrightarrow[\ g_1\ ]{} & B_2 & \xrightarrow[\ f_2\ ]{} & B_3
\end{array}
\quad\mapsto\quad
\begin{array}{ccc}
A_1 & \xrightarrow{\ f_2 f_1\ } & A_3 \\
{\scriptstyle j}\downarrow & \alpha \mid \beta & \downarrow{\scriptstyle \ell} \\
B_1 & \xrightarrow[\ g_2 g_1\ ]{} & B_3
\end{array}
$$

$$
\begin{array}{ccc}
A_1 & \xrightarrow{\;f\;} & A_2 \\
j_1 \downarrow & \alpha & \downarrow k_1 \\
B_1 & \xrightarrow{\;g\;} & B_2 \\
j_1 \downarrow & \beta & \downarrow k_2 \\
C_1 & \xrightarrow{\;h\;} & C_2
\end{array}
\quad \mapsto \quad
\begin{array}{ccc}
A_1 & \xrightarrow{\;f\;} & A_3 \\
j_2 j_1 \downarrow & \frac{\alpha}{\beta} & \downarrow k_2 k_1 \\
C_1 & \xrightarrow{\;h\;} & B_3
\end{array}
$$

- For every vertical map $j : A \to B$, there is an identity square

$$
\begin{array}{ccc}
A & === & A \\
j \downarrow & j & \downarrow j \\
B & === & B
\end{array}
$$

  which we will also refer to as j, for convenience. Similarly, for every horizontal map $f : A \to B$, there is an identity square

$$
\begin{array}{ccc}
A & \xrightarrow{\;f\;} & A \\
\| & f & \| \\
B & \xrightarrow{\;f\;} & B
\end{array}
$$

  which we will also refer to as f, for convenience.
- Vertical and horizontal composition is associative and unital, and the *interchange law* holds. That is:
  - For horizontally composable squares α, β, and γ,

  $$(\alpha \mid \beta) \mid \gamma = \alpha \mid (\beta \mid \gamma).$$

  - For vertically composable squares α, β, and γ,[a]

  $$\left(\frac{\frac{\alpha}{\beta}}{\gamma} \right) = \left(\frac{\alpha}{\frac{\beta}{\gamma}} \right)$$

  - For a square α with left and right vertical edges j and k respectively,

  $$j \mid \alpha = \alpha = \alpha \mid k.$$

– For a square α with top and bottom horizontal edges f and g,

$$\frac{f}{\alpha} = \alpha = \frac{\alpha}{g}.^b$$

– For four appropriately composable squares α, β, γ, and δ, the following interchange law holds:

$$\frac{\alpha \mid \beta}{\gamma \mid \delta} = \frac{\alpha}{\beta} \bigg| \frac{\gamma}{\delta} \ .$$

---

[a]If you're seeing this and feeling worried about fractions, you can put your mind at ease; we promise there will be no fractions. Only squares next to squares.

[b]There aren't any fractions here either.

Phew, that was quite the definition! The reason the definition of a double category is so much more involved than the definition of a category is that there is more than twice the data: there's the vertical category and the horizontal category, but also how they interact through the squares.

Remark 3.4.2. Just like we notate the identity square on a vertical morphism j by j and the identity square on a horizontal morphism f by f, we will often denote composition of vertical morphisms by $\frac{f}{g}$ and of horizontal morphisms by $j \mid k$. This notation agrees with the composition of their respective identity squares, and will be much more pleasant to look at when writing equations.

Let's see a few important examples of double categories.

3.4.3   The double category of arenas in the deterministic doctrine

Finally, we are ready to meet the double category of arenas in the deterministic doctrine. This is where our dynamical systems live, and where they behave.

Definition 3.4.4. The *double category of arenas* in the deterministic doctrine is a double category which has:

- Its objects are the *arenas*, pairs of sets $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$.
- Its horizontal category is the category of charts.
- Its vertical category is the category of lenses.

- There is a square of the following form

$$
\begin{pmatrix} A^- \\ A^+ \end{pmatrix} \xrightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} B^- \\ B^+ \end{pmatrix}
$$

$$
\begin{pmatrix} j^\sharp \\ j \end{pmatrix} \Updownarrow \qquad\qquad \Downarrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix}
$$

$$
\begin{pmatrix} C^- \\ C^+ \end{pmatrix} \xrightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} \begin{pmatrix} D^- \\ D^+ \end{pmatrix}
\tag{3.5}
$$

if and only if the following equations hold:

$$
g(j(a^+)) = k(f(a^+)) \tag{3.6}
$$

$$
k^\sharp(f(a^+), g_\flat(j(a^+), c^-)) = f_\flat(a^+, j^\sharp(a^+, c^-)) \tag{3.7}
$$

for all $a^+ \in A^+$ and $c^- \in C^-$.

It's not obvious from this definition that we actually get a double category with this definition. It's not even clear that we have defined a way to compose the squares vertically and horizontally.

It turns out we don't need to know anything else to know that we can compose these squares, at least in principle. This is because there is at most one square filling any two charts and two lenses that line up as in **??**; to compose these squares just means that if we have two such squares lining up, the defining equations **??** hold also for the appropriate composites. We call double categories with this property *thin*.

Definition 3.4.5. A double category is *thin* if there is at most one square of any signature.

So long as composition is well defined in a thin double category, the laws of associativity and interchange for square composition come for free; there is at most one square of the appropriate signature, so any two you can write down are already equal. We do still have to show that composition is well defined in this way, which we'll do a bit more generally in **??**

Remark 3.4.6. While the definition of double category we gave treated both horizontal and vertical directions the same, we will often want to see a square

$$
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
\downarrow{\scriptstyle j} & \quad\alpha & \downarrow{\scriptstyle k} \\
C & \xrightarrow[\ g\ ]{} & D
\end{array}
$$

as a sort of map $\alpha : j \to k$ from its left to its right side, or a map $\alpha : f \to g$ from its top to its bottom side. For example, the systems themselves are certain lenses (vertical maps), and the behaviors are squares between them. On the other hand, we can also see a square as a way of wiring together charts.

Example 3.4.7. A square

$$
\begin{array}{ccc}
\begin{pmatrix} A^- \\ A^+ \end{pmatrix} & \xRightarrow{\begin{pmatrix} f_b \\ f \end{pmatrix}} & \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\
\begin{pmatrix} j^\sharp \\ j \end{pmatrix}\Updownarrow\Big\uparrow & & \Updownarrow\Big\uparrow\begin{pmatrix} k^\sharp \\ k \end{pmatrix} \\
\begin{pmatrix} C^- \\ C^+ \end{pmatrix} & \xRightarrow[\begin{pmatrix} g^\sharp \\ g \end{pmatrix}]{} & \begin{pmatrix} D^- \\ D^+ \end{pmatrix}
\end{array}
\tag{3.8}
$$

can be seen as a chart between lenses, that is, two charts which are compatible according to the wiring pattern the lenses describe. For example, consider a square of the following form where $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$ is a wiring diagram:

$$
\begin{array}{ccc}
\begin{pmatrix} 1 \\ 1 \end{pmatrix} & \xRightarrow{\begin{pmatrix} b^- \\ b^+ \end{pmatrix}} & \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\
\Big\|\Big\| & & \Updownarrow\Big\uparrow\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \\
\begin{pmatrix} 1 \\ 1 \end{pmatrix} & \xRightarrow[\begin{pmatrix} d^- \\ d^+ \end{pmatrix}]{} & \begin{pmatrix} D^- \\ D^+ \end{pmatrix}
\end{array}
$$

By **??**, we know that the charts in this diagram are pairs of elements $\begin{pmatrix} b^- \\ b^+ \end{pmatrix}$ and $\begin{pmatrix} d^- \\ d^+ \end{pmatrix}$ in the arenas $\begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ and $\begin{pmatrix} D^- \\ D^+ \end{pmatrix}$ respectively. The square then says that $\begin{pmatrix} d^- \\ d^+ \end{pmatrix}$ are the values you would get if you passed $\begin{pmatrix} b^- \\ b^+ \end{pmatrix}$ along the wires in the wiring diagram $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$.

Taking for granted that the double category of arenas is indeed a double category, what does this mean for systems? Well, behaviors are particular squares in the double category of arenas.

Proposition 3.4.8. Let T and S be dynamical systems. A behavior $\phi : \mathsf{T} \to \mathsf{S}$ is equivalently a square of the following form in the double category of arenas:

$$
\begin{pmatrix} \mathsf{State}_\mathsf{T} \\ \mathsf{State}_\mathsf{T} \end{pmatrix} \xRightarrow{\begin{pmatrix} \phi\circ\pi_2 \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{pmatrix}
$$
$$
\begin{pmatrix} \mathsf{update}_\mathsf{T} \\ \mathsf{expose}_\mathsf{T} \end{pmatrix}\Big\Updownarrow\uparrow \qquad\qquad \Updownarrow\downarrow\begin{pmatrix} \mathsf{update}_\mathsf{S} \\ \mathsf{expose}_\mathsf{S} \end{pmatrix}
$$
$$
\begin{pmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{pmatrix} \xRightarrow{\begin{pmatrix} f^\sharp \\ f \end{pmatrix}} \begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix}
\tag{3.9}
$$

Proof. This is a simple matter of checking the definitions against eachother. The defining equations of **??** specialize to the defining equations of **??**. □

This re-expression of the notion of behavior in terms of the double category of arenas will let us generalize from the deterministic doctrine to other doctrines.

3.4.9 The double category of sets, functions, and matrices

Now we turn to our second double category of interest, the double category of sets, functions, and matrices of sets.

Definition 3.4.10. The double category **Matrix** of sets, functions, and matrices of sets is defined by:
- Its objects are sets.
- Its horizontal category is the category of sets and functions.
- Its vertical category is the category of sets and matrices of sets, where composition is given by matrix multiplication. We write $M : A \to B$ to say that M is a $B \times A$ matrix.
- For functions $f : A \to B$ and $g : C \to D$ and matrices $M : A \to C$ and $N : B \to D$, a square

$$
\begin{CD}
A @>f>> B \\
@VMVV @VVNV \\
C @>>g> D
\end{CD}
$$

with α in the center,

is a family of functions $\alpha_{ba} : M_{ba} \to N_{g(b)f(a)}$ for all $a \in A$ and $b \in B$.
- Horizontal composition of squares is given by composition of the families:

$$
(\alpha|\beta)_{ba} = \beta_{g(b)f(a)} \circ \alpha_{ba}.
$$

- Vertical composition of squares

$$
\begin{array}{ccc}
A_1 & \xrightarrow{\ f\ } & B_1 \\
M^1 \downarrow & \quad \alpha \quad & \downarrow N^1 \\
A_2 & \xrightarrow[\ g\ ]{} & B_2 \\
M^2 \downarrow & \quad \beta \quad & \downarrow N^2 \\
A_3 & \xrightarrow[\ h\ ]{} & B_3
\end{array}
$$

is given by

$$
\left(\frac{\alpha}{\beta}\right)_{ac} : \sum_{b_1 \in B_1} M^2_{cb_1} \times M^1_{b_1 a} \to \sum_{b_2 \in B_2} N^2_{h(c)b_2} \times N^1_{b_2 f(a)}
$$

$$
(b_1, m_2, m_1) \mapsto (g(b_1), \beta(m_2), \alpha(m_1)).
$$

Exercise 3.4.11.   We can see that horizontal composition of squares is associative and unital since it is basically just function composition. Show that **Matrix** is a double category by checking that

1. Vertical composition of squares is associative and unital (up to isomorphism).
2. The interchange law holds.

◇

There is another useful way to express the double category of matrices in terms of pullbacks: spans. A span of sets from A to B is a diagram

$$
\begin{array}{ccc}
 & S & \\
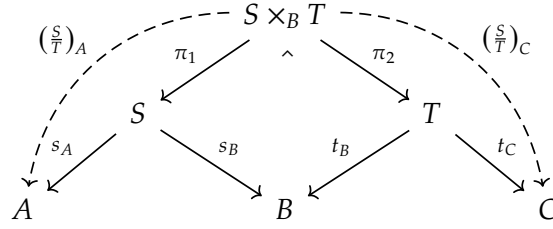{}^{s_A}\swarrow & & \searrow^{s_B} \\
A & & B
\end{array}
$$

We can think of this as a matrix of sets by sending any $a \in A$ and $b \in B$ to the set $S_{ba} = \{x \in S \mid a = s_A(x) \text{ and } s_B = b\}$. And to any $(B \times A)$-matrix M of sets, we can associate the span

$$
\begin{array}{ccc}
 & \sum_{(a,b) \in A \times B} M_{ba} & \\
{}^{\pi_1}\swarrow & & \searrow^{\pi_2} \\
A & & B
\end{array}
$$

with the disjoint union of all M_{ba} at the top, with the two maps begin the projections onto A and B respectively.

The composition of matrices can be represented in terms of spans as well. Given the spans S from A to B and T from B to C, we can define their composite span $\frac{S}{T}$ from

A to *C* by taking the pullback:

$$
\begin{array}{c}
S \times_B T
\end{array}
$$

with maps $\left(\frac{S}{T}\right)_A$, π_1, π_2, $\left(\frac{S}{T}\right)_C$, S, T, s_A, s_B, t_B, t_C to objects A, B, C.
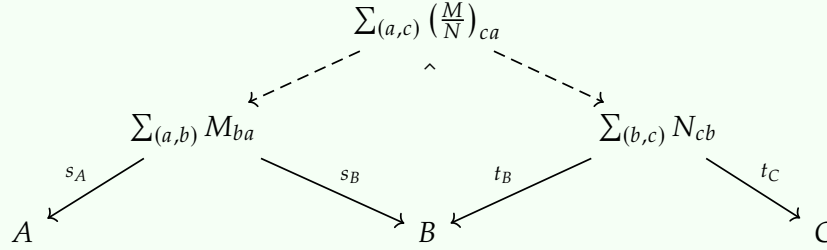
where

$$
S \times_B T = \{(x, y) \in S \times T \mid s_B(x) = t_B(y)\}.
$$

A bit of thinking shows that this corresponds to the composite of matrices.

Exercise 3.4.12. Let M be an $(A \times B)$-matrix and N be a $(B \times C)$-matrix. Consider the following diagram:

$$
\sum_{(a,c)} \left(\tfrac{M}{N}\right)_{ca}
$$

with maps to $\sum_{(a,b)} M_{ba}$ and $\sum_{(b,c)} N_{cb}$, and s_A, s_B, t_B, t_C to objects A, B, C.

Show that there are dashed maps as in the above diagram so that the square is a pullback. This shows that the composition of matrices corresponds to the composition of spans. ◇

One nice feature that spans have over matrices is that they work for things other than sets. We can take spans in any category with pullbacks. We'll record the double category of spans here.

Definition 3.4.13. Let $\mathcal{C}$ be a category with pullbacks. The double category **Span**($\mathcal{C}$) is defined by:
- Its objects are the objects of $\mathcal{C}$.
- Its horizontal category is $\mathcal{C}$.
- Its vertical category has as morphisms $S : A \to B$ the spans

$$
\begin{array}{ccc}
 & S & \\
s_A \swarrow & & \searrow s_B \\
A & & B
\end{array}
$$
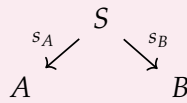
and these are composed by pullback.

- A square

$$
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
M \downarrow & \alpha & \downarrow N \\
C & \xrightarrow[g]{} & D
\end{array}
\qquad \text{is a map } \alpha \text{ so that} \qquad
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
{}_{s_A}\uparrow & & \uparrow {}^{t_B} \\
S & \dashrightarrow{}^{\alpha} & T \\
{}_{s_C}\downarrow & & \downarrow {}^{t_D} \\
C & \xrightarrow[g]{} & D
\end{array}
\qquad \text{commutes.}
$$

- Horizontal composition of squares is by composing in $\mathcal{C}$.
- Vertical composition of squares follows from the functoriality of the pullback:

$$
\begin{array}{ccc}
A_1 & \xrightarrow{\ f_1\ } & B_1 \\
\uparrow & & \uparrow \\
S_1 & \xrightarrow{\ \alpha\ } & T_1 \\
\downarrow & & \downarrow \\
A_2 & \xrightarrow[f_2]{} & B_2 \\
\uparrow & & \uparrow \\
S_2 & \xrightarrow{\ \beta\ } & T_2 \\
\downarrow & & \downarrow \\
A_3 & \longrightarrow & B_3
\end{array}
\qquad \mapsto \qquad
\begin{array}{ccc}
A_1 & \xrightarrow{\qquad f_1 \qquad} & B_1 \\
\uparrow & & \uparrow \\
S_1 \times_{A_2} S_2 & \dashrightarrow{}^{\frac{\alpha}{\beta}} & T_1 \times_{B_2} T_2 \\
\downarrow & & \downarrow \\
A_3 & \xrightarrow[f_3]{} & B_3
\end{array}
$$

3.4.14   The double category of categories, profunctors, and functors

Now we come to the primordial double category: the double category of categories, *profunctors*, and functors. This is an important double category because it is in some sense the setting in which all category theory takes place. Before we describe this double category, let's define the notion of profunctor and their category.

Definition 3.4.15. A *profunctor* $P : \mathcal{A} \nrightarrow \mathcal{B}$ is a functor $P : \mathcal{A}^{\mathrm{op}} \times \mathcal{B} \to \mathbf{Set}$. Given objects $A \in \mathcal{A}$ and $B \in \mathcal{B}$, we write an element $p \in P(A, B)$ as $p : A \nrightarrow B$.

In terms of this, the functoriality of P can be seen as letting us compose $p : A \nrightarrow B$ on the left and right by $f : A' \to A$ and $g : B \to B'$ to get $fpg : A' \nrightarrow B'$. In other words, we can interpret a diagram of this form

$$
A' \xrightarrow{f} A \xrightarrow{p} B \xrightarrow{g} B'
$$

as an element of $P(A', B')$.

If we call maps $f : A' \to A$ in a category $\mathcal{A}$ *homomorphisms* because they go between objects of the same form, we could call elements $p : A \xrightarrow{B} $ — that is, $p \in P(A, B)$ — as *heteromorphisms*, maps going between objects of different forms.

We can't necessarily compose these heteromorphisms, which we can see right away from their signature: for $p : A \nrightarrow B$, there is always an object of $\mathcal{A}$ on the left and an object of $\mathcal{B}$ on the right, so we'll never be able to line two of them up. However,

if we have another profunctor $Q : \mathscr{B} \nrightarrow \mathcal{C}$ — another notion of heteromorphism — then we can "compose" heteromorphisms $A \xrightarrow{p} B$ in P with $B \xrightarrow{q} C$ in Q to get a heteromorphism $A \xrightarrow{p} B \xrightarrow{q} C$ in a new profunctor $P \odot Q : \mathscr{A} \nrightarrow \mathcal{C}$.

Definition 3.4.16. The composite $P \odot Q$ of a profunctor $P : \mathscr{A} \nrightarrow \mathscr{B}$ with a profunctor $Q : \mathscr{B} \nrightarrow \mathcal{C}$ is defined to be the following quotient:

$$(P \odot Q)(A, C) := \frac{\sum_{B \in \mathscr{B}} P(A, B) \times Q(B, C)}{(pf, q) \sim (p, fq)} \tag{3.10}$$

We write an element $[(p, q)] \in (P \odot Q)(A, C)$ as $A \xrightarrow{p} B \xrightarrow{q} C$, so that the relation we quotient by says that

$$A \xrightarrow{p} B \xrightarrow{f} B' \xrightarrow{q} C$$

has a unique interpretation as an element of $P \odot Q$.

The identity profunctor $\mathscr{A} : \mathscr{A} \nrightarrow \mathscr{A}$ is the hom-functor sending A and A' to the set $\mathscr{A}(A, A')$ of maps $A \to A'$.

We can see that composition of profunctors is associative (up to isomorphism) because the objects of $P \odot (Q \odot R)$ and $(P \odot Q) \odot R$ can both be written as

$$A \xrightarrow{p} B \xrightarrow{q} C \xrightarrow{r} D.$$

The reason the hom profunctor $\mathscr{A} : \mathscr{A} \nrightarrow \mathscr{A}$ is the identity profunctor is because the elements of $\mathscr{A} \odot P$ would be written as

$$A' \xrightarrow{f} A \xrightarrow{p} B$$

but by the functoriality of P, this is already an element of $P(A', B)$, which is to say more precisely that every equivalence class $[(f, p)] \in (\mathscr{A} \odot P)(A', B)$ is equally presented as $[(\mathrm{id}_{A'}, fp)]$.

Exercise 3.4.17.   Let $P : \mathscr{A} \nrightarrow \mathscr{B}$ be a profunctor.
1. Show that there is a natural transformation $\mathscr{A} \odot P \to P$ given by the naturality of P on the left.
2. Show that there is a natural transformation $P \odot \mathscr{B} \to P$ given by the naturality of P on the right.
3. Show that both of these natural transformations are isomorphisms.

$\diamond$

Example 3.4.18. A profunctor $1 \nrightarrow \mathscr{A}$ is the same thing as a functor $\mathscr{A} \to \mathbf{Set}$, and a profunctor $\mathscr{A} \nrightarrow 1$ is the same thing as a functor $\mathscr{A}^{\mathrm{op}} \to \mathbf{Set}$. Profunctors are therefore

intimately related with presheaves.

Now, we are ready to put functors and profunctors together into a double category.

Definition 3.4.19. The double category **Cat** of categories, profunctors, and functors has

- Objects the categories.
- Horizontal category the category of categories and profunctors.
- Vertical category the category of categories and functors between them.
- A square

$$
\begin{array}{ccc}
\mathcal{A} & \xrightarrow{\;P\;} & \mathcal{B} \\
F \downarrow & \alpha & \downarrow G \\
\mathcal{C} & \xrightarrow[\;Q\;]{} & \mathcal{D}
\end{array}
$$

  Is a natural transformation $\alpha : P \to Q(F, G)$, where $Q(F, G)$ is the profunctor $\mathcal{A}^{\mathrm{op}} \times \mathcal{B} \xrightarrow{F^{\mathrm{op}} \times G} \mathcal{C} \times \mathcal{D} \xrightarrow{Q} \mathbf{Set}$. For $p : A \nrightarrow B$, we have $\alpha(p) : FA \nrightarrow GB$, and naturality says that $\alpha(fpg) = (Ff)\alpha(p)(Gg)$.
- Vertical composition of squares is given by composing the natural transformations.
- Given squares $\alpha : P_1 \to Q_1(F_1, F_2)$ and $\beta : P_2 \to Q_2(F_2, F_3)$, we define their horizontal composite $\alpha \mid \beta : P_1 \cdot P_2 \to (Q_1 \cdot Q_2)(F_1, F_3)$ by

$$
(\alpha \mid \beta)(A_1 \xrightarrow{p_1} A_2 \xrightarrow{p_2} A_3) = F_1 A_1 \xrightarrow{\alpha(p_1)} F_2 A_2 \xrightarrow{\beta(p_2)} F_3 A_3
$$

  and checking that this descends correctly to the quotient.

Remark 3.4.20. We are using "**Cat**" to refer to the category of categories and functors and to the double category of categories, profunctors, and functors. The one we mean will be clear from context, and the category of categories and functors is the vertical category of the double category of categories.

Remark 3.4.21. We omit full proofs of associativity and unitality for profunctor composition because they are best done with the *coend calculus*, and this would take us quite far afield.

However, we will note that there is always a unique coherent isomorphism between any two sequences of profunctors which would be equal if unity and associativity held on the nose. We will do an example, since the general principle is always the same.

Consider $P : \mathcal{A} \nrightarrow \mathcal{B}$ and $Q : \mathcal{B} \nrightarrow \mathcal{C}$. We will give the canonical isomorphism $(\mathcal{A} \odot P) \odot (Q \odot \mathcal{C}) \xrightarrow{\sim} P \odot (\mathcal{B} \odot (\mathcal{B} \odot Q))$.

First, we begin with an isomorphism $(\mathcal{A} \odot P) \odot (Q \odot \mathcal{C}) \xrightarrow{\sim} P \odot Q$ and then an isomorphism $P \odot Q \xrightarrow{\sim} P \odot (\mathcal{B} \odot (\mathcal{B} \odot Q))$. The first will be given by naturality,

composition and re-associating; the second by inserting appropriate identities and re-associating.

An element of $(\mathcal{A} \odot P) \odot (Q \odot \mathcal{C})(A, C)$ is an equivalence class $[((f, p), (q, g))]$. We may therefore use the naturality of P and Q to give the class $[(f \cdot p, q \cdot g)] \in (P \odot Q)(A, C)$. It is routine to check that this is indeed an isomorphism. It is hopefully clear how to do this in general.

Now, we go the other way. An element of $(P \odot Q)(A, C)$ is an equivalence class $[(p, q)]$. We may then insert identities to give the class $[(p, (\mathrm{id}, (\mathrm{id}, q)))] \in P \odot (\mathcal{B} \odot (\mathcal{B} \odot Q))(A, C)$.

A crucial point about canonical isomorphisms constructed in this manner is that they compose: the composite of a canonical isomorphism is the canonical isomorphism of that signature.

Exercise 3.4.22. Describe the canonical isomorphisms between the following composites of profunctors. First, flatten them out by removing all hom profunctors using naturality; then expand them again by inserting identities. Let $P : \mathcal{A} \nrightarrow \mathcal{B}, Q : \mathcal{B} \nrightarrow \mathcal{C}$, and $R : \mathcal{C} \nrightarrow \mathcal{D}$.

1. $(P \odot \mathcal{B}) \odot (\mathcal{B} \odot Q) \xrightarrow{\sim} \mathcal{A} \odot ((P \odot \mathcal{B}) \odot \mathcal{C})$.
2. $P \odot ((Q \odot \mathcal{C}) \odot (\mathcal{C} \odot R)) \xrightarrow{\sim} ((P \odot Q) \odot \mathcal{C}) \odot (R \odot \mathcal{D})$.

$\diamond$

Remark 3.4.23. We will often need equalities between squares in the double category $\mathcal{C}at$ whose boundaries are not precisely equal, but which are canonically isomorphic. The coming **??** is an example of this common scenario.

It would clutter already intricate proofs to keep track of the canonical isomorphisms which are being introduced and cancelled at each step. For this reason, we'll introduce notation for "equal up to canonical isomorphism on the boundary". We will write

$$\alpha \doteq \beta$$

to mean that although α and β have different boundaries, these boundaries are canonically isomorphic and whenever they are made to be the same by any canonical isomorphism (pre- or post-composing α and β as necessary), the resulting squares will be honestly equal. We will see our first example in **??**.

Before we move on from the double category **Cat**, let's record an important relationship between its squares (natural transformations between profunctors) and natural transformations between functors. We will show that natural transformations are the same thing as squares in **Cat** whose top and bottom sides are hom profunctors.

Proposition 3.4.24. Let F and $G : \mathcal{A} \to \mathcal{B}$ be functors. Then there is a (natural) bijection

$$\{\text{Natural transformations } F \Rightarrow G\} \cong \left\{ \text{Squares } \begin{array}{ccc} \mathcal{A} & = & \mathcal{A} \\ F\downarrow & \alpha & \downarrow G \\ \mathcal{B} & = & \mathcal{B} \end{array} \right\}$$

given by sending the natural transformation $\alpha : F \Rightarrow G$ to the transformation $\bar{\alpha} :$ $\mathcal{A}(X,Y) \Rightarrow \mathcal{B}(FX, GY)$ that sends any $f : X \to Y$ the diagonal α_f of the naturality square:

$$\begin{array}{ccc} FX & \xrightarrow{\alpha_X} & GX \\ Ff\downarrow & \overset{\alpha_f}{\searrow} & \downarrow Gf \\ FY & \xrightarrow{\alpha_Y} & GY \end{array}$$

Proof. First, let's check that the transformation $\bar{\alpha}(f) = \alpha_f$ is natural. If $x : X' \to X$ and $y : Y \to Y'$, then we can form the following commutative diagram of naturality squares:

$$\begin{array}{ccc} FX' & \xrightarrow{\alpha_{X'}} & GX' \\ Fx\downarrow & & \downarrow Gx \\ FX & \xrightarrow{\alpha_X} & GX \\ Ff\downarrow & \overset{\alpha_f}{\searrow} & \downarrow Gf \\ FY & \xrightarrow{\alpha_Y} & GY \\ Fy\downarrow & & \downarrow Gy \\ FY' & \xrightarrow{\alpha_{Y'}} & GY' \end{array}$$

The diagonal of the outer square is by definition $\bar{\alpha}(x \,\mathring{,}\, f \,\mathring{,}\, y)$, but we can see from the commutativity of the diagram that it equals $Fx \,\mathring{,}\, \alpha_f \,\mathring{,}\, Gy$.

It remains to show that any natural transformation $\beta : \mathcal{A}(X,Y) \Rightarrow \mathcal{B}(FX, GY)$ arises uniquely as $\bar{\alpha}$ for a natural transformation $\alpha : F \Rightarrow G$. Given such a β, define $\alpha_X := \beta(\text{id}_X)$. We need to prove the naturality of α by showing that any solid square

$$\begin{array}{ccc} FX & \xrightarrow{\alpha_X} & GX \\ Ff\downarrow & \overset{\beta(f)}{\searrow} & \downarrow Gf \\ FY & \xrightarrow{\alpha_Y} & GY \end{array}$$

commutes. But note that if we put in the dashed $\beta(f)$, we can see that both triangles commute by the naturality of $\beta(f)$:

$$\beta(f) = \beta(\text{id}_X \,\mathring{,}\, f) = \beta(\text{id}_X) \,\mathring{,}\, Gf = \alpha_X \,\mathring{,}\, Gf.$$

$$\beta(f) = \beta(f \,\mathring{,}\, \text{id}_Y) = Ff \,\mathring{,}\, \beta(\text{id}_Y) = Ff \,\mathring{,}\, \alpha_Y.$$

This also shows that $\beta(f) = \bar{\alpha}(f)$, which completes the proof. $\qquad\square$

There are two ways to compose natural transformations: vertically, and horizontally. The above bijection respects both of these compositions. In the following lemmas we take the notation from **??**.

Lemma 3.4.25. Let $\alpha : F \Rightarrow G$ and $\beta : G \Rightarrow H$. Then

$$\overline{\alpha \,\mathring{,}\, \beta} \doteq \bar{\alpha} \mid \bar{\beta}.$$

Proof. Here we are using the symbol "$\doteq$" from **??** for the first time; this is because the two sides do not have equal signature, only isomorphic signature. To correctly compare them, we must conjugate by the appropriate isomorphisms. Here, with signature included, is the actual equality we will prove:

$$
\begin{array}{ccc}
\mathscr{A} \;=\; \mathscr{A} \;=\; \mathscr{A} & & \mathscr{A} \;=\; \mathscr{A} \;=\; \mathscr{A} \\
\Big\| \quad \cong \quad \Big\| & & {\scriptstyle F}\Big\downarrow \quad \bar{\alpha} \quad {\scriptstyle G}\Big\downarrow \quad \bar{\beta} \quad \Big\downarrow{\scriptstyle H} \\
\mathscr{A} \;=\; \mathscr{A} & \;=\; & \mathscr{B} \;=\; \mathscr{B} \;=\; \mathscr{B} \\
{\scriptstyle F}\Big\downarrow \quad \overline{\alpha\,\mathring{,}\,\beta} \quad \Big\downarrow{\scriptstyle H} & & \Big\| \quad \cong \quad \Big\| \\
\mathscr{B} \;=\; \mathscr{B} & & \mathscr{B} \;=\; \mathscr{B}
\end{array}
$$

We leave the canonical isomorphisms without names. They can be described by the process outlined in **??**. We note that both of these canonical isomorphisms are given by composing two arrows, so in order to prove the equality above we will show that given $f : X \to Y$ and $g : Y \to Z$,

$$(\alpha \,\mathring{,}\, \beta)_{f\,\mathring{,}\,g} = \alpha_f \,\mathring{,}\, \beta_g.$$

We will do this by contemplating the following diagram:

$$
\begin{array}{ccccc}
FX & \xrightarrow{\ \alpha_X\ } & GX & \xrightarrow{\ \beta_X\ } & HX \\
{\scriptstyle Ff}\Big\downarrow & {\scriptstyle \alpha_f}\searrow & \Big\downarrow{\scriptstyle Gf} & & \Big\downarrow{\scriptstyle Hf} \\
FY & \xrightarrow{\ \alpha_Y\ } & GY & \xrightarrow{\ \beta_Y\ } & HY \\
{\scriptstyle Fg}\Big\downarrow & & \Big\downarrow{\scriptstyle Gg} & {\scriptstyle \beta_g}\searrow & \Big\downarrow{\scriptstyle Hg} \\
FZ & \xrightarrow{\ \alpha_Z\ } & GZ & \xrightarrow{\ \beta_Z\ } & HZ
\end{array}
$$

The naturality square for the composite $f \,\mathring{,}\, g$ under the composite $\alpha \,\mathring{,}\, \beta$ is the outer square, and therefore its diagonal $(\alpha \,\mathring{,}\, \beta)_{f\,\mathring{,}\,g}$ is the composite of the diagonals in the diagram, which is $\alpha_f \,\mathring{,}\, \beta_g$. $\qquad\square$

Lemma 3.4.26. Let F_1, $G_1 : \mathcal{A} \to \mathcal{B}$ and F_2, $G_2 : \mathcal{B} \to \mathcal{C}$ be functors, and let $\alpha : F_1 \Rightarrow G_1$ and $\beta : F_2 \Rightarrow G_2$ be natural transformations. We may define their composite $\alpha * \beta$ by

$$(\alpha * \beta)_X := F_2\alpha_X \,\fatsemi\, \beta_{G_1 X}.$$

With this definition, we have

$$\overline{\alpha * \beta} = \frac{\bar{\alpha}}{\bar{\beta}}$$

Remark 3.4.27. Note that the equality claimed here is a bona-fide equality, and not an "equality up to canonical isomorphism" ($\doteq$). This is because the two squares involved have the exact same boundary, not merely a canonically isomorphic boundaries.

Proof. This time, we may prove the equality as stated. It comes down to showing that

$$(\alpha * \beta)_f = \beta_{\alpha_f}$$

for any $f : X \to Y$. Consider the following diagram:
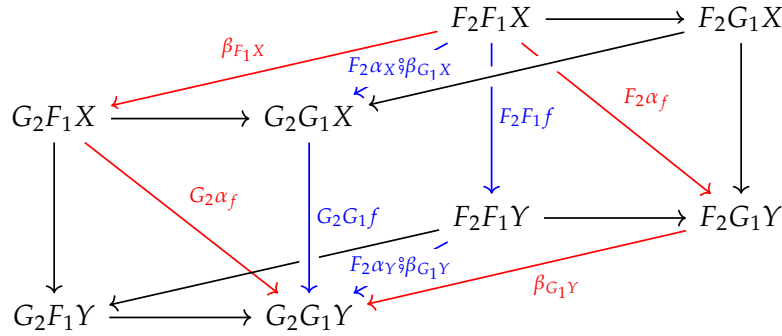
The back and front faces are the α naturality square of f pushed through F_2 and G_2 respectively. The β naturality square of α_f is in the middle, colored in red. The $\alpha * \beta$ naturality square of f is in the middle, colored in blue. We note that the diagonal of both these squares is the diagonal of the whole cube $F_2 F_1 X \to G_2 G_1 Y$, which means that they are equal. But this is what we were trying to show.

$\square$

3.5   Dynamical System Doctrines

In **??**, we saw how from the data of an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ we could define a category of $\mathcal{A}$-lenses via the Grothendieck construction:

$$\mathbf{Lens}_{\mathcal{A}} := \int^{C:\mathcal{C}} \mathcal{A}(C)^{\mathrm{op}}.$$

From this, we learned we could wire non-deterministic systems together because a system could be expressed as a monadic lens of the form $\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} \text{State}_S \\ \text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$.

Now, the form $\begin{pmatrix} S \\ S \end{pmatrix} \leftrightarrows \begin{pmatrix} I \\ O \end{pmatrix}$ is *not* something that be expressed for a general $\mathcal{A}$-lens because in an $\mathcal{A}$-lens $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$, $A^+ \in \mathcal{C}$ while $A^- \in \mathcal{A}(C)$. In general, $\mathcal{C}$ and $\mathcal{A}(C)$ might have different objects. This suggests that we need a way to assign an object $TC \in \mathcal{A}(C)$ to each object of C, so that we can define a system, in general, to be an $\mathcal{A}$-lens of the form

$$\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} T\text{States}_S \\ \text{States}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$$

At this point, your categorical nose should be twitching. We've given a assignment on objects; how is this assignment functorial? We can discover what sort of functoriality we need from considering the expression in **??** of behaviors as squares of arenas in the deterministic doctrine:

$$
\begin{pmatrix} \text{State}_T \\ \text{State}_T \end{pmatrix} \xrightarrow{\begin{pmatrix} \phi \circ \pi_2 \\ \phi \end{pmatrix}} \begin{pmatrix} \text{States}_S \\ \text{States}_S \end{pmatrix}
$$

$$
\begin{pmatrix} \text{update}_T \\ \text{expose}_T \end{pmatrix} \Updownarrow\uparrow \qquad \Updownarrow\uparrow \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix}
$$

$$
\begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix} \xrightarrow{\begin{pmatrix} f^\sharp \\ f \end{pmatrix}} \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}
$$

To express this square, we did not just use the fact that we could find States_S in both the base $\mathcal{C}$ and in the category $\mathcal{A}(\text{States}_S)$ (recall that here, $\mathcal{A} = \mathbf{Ctx}_- : \mathcal{C}^{\text{op}} \to \mathbf{Cat}$). We also used the fact that to any map $\phi : \text{State}_T \to \text{States}_S$ we can build a chart $\begin{pmatrix} \phi \circ \pi_2 \\ \phi \end{pmatrix} : \begin{pmatrix} State T \\ \text{State}_T \end{pmatrix} \to \begin{pmatrix} \text{States}_S \\ \text{States}_S \end{pmatrix}$. This is the sort of functoriality we need to define the notion of behavior in general.

Definition 3.5.1. Let $\mathcal{A} : \mathcal{C}^{\text{op}} \to \mathbf{Cat}$ be a strict indexed category. A *section T* of $\mathcal{A}$ consists of the following assignments:

- To every object $C \in \mathcal{C}$, an object $TC \in \mathcal{A}(C)$.
- To every $\phi : C' \to C$, a map $T\phi : TC' \to \phi^*C$.

These are required to satisfy the following laws:

- For any $C \in \mathcal{C}$, $T\text{id}_C = \text{id}_{TC}$.
- For $\phi : C' \to C$ and $\psi : C'' \to C'$,

$$T\psi \,\mathring{,}\, \psi^*(T\phi) = T(\psi \,\mathring{,}\, \phi).$$

We can express a section of an indexed category in terms of a functor into its Grothendieck construction.

Proposition 3.5.2. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be a strict indexed category. A section T of $\mathcal{A}$ is equivalently given by the data of a functor $\hat{T} : \mathcal{C} \to \int^{C:\mathcal{C}} \mathcal{A}(C)$ for which the composite $\mathcal{C} \xrightarrow{\hat{T}} \int^{C:\mathcal{C}} \mathcal{A}(C) \xrightarrow{\text{proj}} \mathcal{C}$ with the projection is the identity on $\mathcal{C}$.

Given a section T, we may more suggestively refer to $\hat{T}$ by $\left(\begin{smallmatrix} T(-) \\ (-) \end{smallmatrix} \right)$.

Proof. Given a section T, we can form the functor

$$ C \mapsto \begin{pmatrix} TC \\ C \end{pmatrix} : \mathcal{C} \to \int^{C:\mathcal{C}} \mathcal{A}(C) $$

sending ϕ to $\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix} \right)$. The laws of the section show that this is a functor.

On the other hand, given a $\hat{T} : \mathcal{C} \to \int^{C:\mathcal{C}} \mathcal{A}(C)$ whose composite with the projection is the identity, we see that $\hat{T}(C)$ must be of the form $\left(\begin{smallmatrix} TC \\ C \end{smallmatrix} \right)$ and that $\hat{T}(\phi)$ must be of the form $\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix} \right)$, where TC and $T\phi$ are defined to be the components of $\hat{T}$ which live in the categories $\mathcal{A}(C)$. It is straightforward to check that functoriality implies the laws of a section. $\qquad\square$

We can see that the assignmen $\phi \mapsto \phi \circ \pi_2$ is a section of $\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$.

Proposition 3.5.3. Let $\mathcal{C}$ be a cartesian category. Then the assigment $C \mapsto C$ and $\phi \mapsto \phi \circ \pi_2$ gives a section of $\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$.

Proof. We check that the two laws are satisfied.
1. $\mathrm{id} \circ \pi_2 = \pi_2$, which is the identity in $\mathbf{Ctx}_C$.
2. We may calculate:

$$ (\psi \circ \pi_2) \,\mathring{\,}\, \psi^*(\phi \circ \pi_2)(c, x) = \psi^*(\phi \circ \pi_2)(c, \psi(x)) $$
$$ = \phi \circ \pi_2(\psi(c), \psi(x)) $$
$$ = \phi(\psi(x)) $$
$$ = (\psi \,\mathring{\,}\, \phi) \circ \pi_2(c, x) $$

$\qquad\square$

In order to define lenses, we need the data of an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$. In order to define dynamical systems as $\mathcal{A}$-lenses, and to define the behaviors between them, we need the data of a section T of $\mathcal{A}$. Putting these two bits of data together, we get the notion of *dynamical system doctrine*.

Definition 3.5.4. A *dynamical system doctrine* consists of an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ together with a section T.

Having this definition of doctrine in mind, we can now define the notion of dynamical system and behavior in complete generality.

Definition 3.5.5. A dynamical system in a doctrine $\mathbb{D} = (\mathcal{A}, T)$ is an $\mathcal{A}$-lens of the form

$$\begin{pmatrix} \text{update}_\mathsf{S} \\ \text{expose}_\mathsf{S} \end{pmatrix} : \begin{pmatrix} T\text{States} \\ \text{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_\mathsf{S} \\ \text{Out}_\mathsf{S} \end{pmatrix}.$$

Explcitly, this consists of:
- An object $\text{States} \in \mathcal{C}$ of *states*.
- An object $\text{Out}_\mathsf{S} \in \mathcal{C}$ of *possible outputs*.
- An object $\text{In}_\mathsf{S} \in \mathcal{A}(\text{Out}_\mathsf{S})$ of *possible inputs* or *parameters*. What parameters are sensible may therefore depend on the output (in the sense of being an object of a category which depends for its definition on Out_S).
- A map $\text{expose}_\mathsf{S} : \text{States} \to \text{Out}_\mathsf{S}$ which *exposes* the output of a given state.
- A map $\text{update}_\mathsf{S} : \text{expose}_\mathsf{S}^* \text{In}_\mathsf{S} \to T\text{States}$ which assigns to any parameter valid for the output of a given state to a possible change in state.

In order to define the notion of behavior, we will need to generalize the double category of arenas from the deterministic doctrine to an arbitrary doctrine. To do this, we will define the *Grothendieck double construction*, which produces a double category of arenas from an indexed category $\mathcal{A}$.

Definition 3.5.6. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be an indexed category. The *Grothendieck double construction*

$$\oiint^{C \in \mathcal{C}} \mathcal{A}(C)$$

is the double category defined by:
- Its objects are the pairs $\begin{pmatrix} A \\ C \end{pmatrix}$ of an object $C \in \mathcal{C}$ and an object $A \in \mathcal{A}(C)$.
- Its horizontal category is the Grothendieck construction $\int^{C \in \mathcal{C}} \mathcal{A}(C)$ of $\mathcal{A}$.
- Its vertical category is the Grothendieck construction $\int^{C \in \mathcal{C}} \mathcal{A}(C)^{\mathrm{op}}$ of the opposite of $\mathcal{A}$.

- There is a square of the following form:

$$
\begin{pmatrix} A_1 \\ C_1 \end{pmatrix} \xRightarrow{\begin{pmatrix} g_{1b} \\ g_1 \end{pmatrix}} \begin{pmatrix} A_2 \\ C_2 \end{pmatrix}
$$

$$
\begin{pmatrix} f_1^\sharp \\ f_1 \end{pmatrix} \Big\updownarrow \qquad\qquad \Big\updownarrow \begin{pmatrix} f_2^\sharp \\ f_2 \end{pmatrix}
$$

$$
\begin{pmatrix} A_3 \\ C_3 \end{pmatrix} \xRightarrow{\begin{pmatrix} g_{2b} \\ g_2 \end{pmatrix}} \begin{pmatrix} A_4 \\ C_4 \end{pmatrix}
$$

(3.11)

if and only if the following diagrams commute:

$$
\begin{array}{ccc}
C_1 & \xrightarrow{g_1} & C_2 \\
{\scriptstyle f_1}\downarrow & & \downarrow{\scriptstyle f_2} \\
C_3 & \xrightarrow{g_2} & C_4
\end{array}
\qquad\qquad
\begin{array}{ccc}
f_1^* A_3 & \xrightarrow{\ \ f_1^\sharp\ \ } & A_1 \\
{\scriptstyle f_1^* g_{2b}}\downarrow & & \downarrow{\scriptstyle g_{1b}} \\
f_1^* g_2^* A_4 = g_1^* f_2^* A_4 & \xrightarrow{g_1^* f_2^\sharp} & g_1^* A_2
\end{array}
$$

(3.12)

We will call the squares in the Grothendieck double construction *commuting squares*, since they represent the proposition that the "lower" and "upper" squares appearing in their boundary commute.

- Composition is given as in the appropriate Grothendieck constructions.

It just remains to show that commuting squares compose.

- For vertical composition we appeal to the following diagram:

$$
\begin{array}{ccccc}
f_1^* f_3^* A_5 & \xrightarrow{\ \ f_1^* f_3^\sharp\ \ } & f_1^* A_3 & \xrightarrow{\ f_1^\sharp\ } & A_1 \\
{\scriptstyle f_1^* f_3^* g_{3b}}\downarrow & & \downarrow{\scriptstyle f_1^* g_{2b}} & & \\
f_1^* f_3^* g_3^* A_6 = f_1^* g_2^* f_4^* A_6 & \xrightarrow{f_1^* g_2^* f_4^\sharp} & f_1^* g_2^* A_4 & & \downarrow{\scriptstyle g_{1b}} \\
\| & & \| & & \\
g_1^* f_2^* f_4^* A_6 & \xrightarrow{g_1^* f_2^* f_4^\sharp} & g_1^* f_2^* A_4 & \xrightarrow{g_1^* f_2^\sharp} & g_1^* A_2
\end{array}
$$

The outer diagram is the "upper" square of the composite, while the "upper" squares of each factor appear in the top left and right respectively.

- For horizontal composition we appeal to the following diagram:

$$
\begin{array}{ccc}
f_1^*A_3 & \xrightarrow{\;\;f_1^\sharp\;\;} & A_1 \\[4pt]
{\scriptstyle f_1^*g_{2\flat}}\downarrow & & \downarrow{\scriptstyle g_{1\flat}} \\[4pt]
f_1^*g_2^*A_4 \;=\!=\; g_1^*f_2^*A_4 & \xrightarrow{\;g_1^*f_2^\sharp\;} & g_1^*A_2 \\[4pt]
& {\scriptstyle g_1^*f_2^*g_{4\flat}}\downarrow & \\[4pt]
{\scriptstyle f_1^*g_2^*g_{4\flat}}\Big\downarrow & g_1^*f_2^*g_4^*A_6 & \Big\downarrow{\scriptstyle g_1^*g_{3\flat}} \\[4pt]
& \| & \\[4pt]
f_1^*g_2^*g_4^*A_6 \;=\!=\; g_1^*g_3^*f_3^* & \xrightarrow[\;g_1^*g_3^*f_3^\sharp\;]{} & g_1^*g_3^*
\end{array}
$$

We can now check that this does indeed abstract the double category of arenas.

> **Proposition 3.5.7.** The double category of arenas in the deterministic doctrine is the Grothendieck double construction of the indexed category of sets and functions in context $\mathbf{Ctx}_- : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Cat}$:
> $$\mathbf{Arena} = \oiint^{C \in \mathbf{Set}} \mathbf{Ctx}_C.$$

Proof. By **????**, the horizontal and vertical categories are the same. It remains to show that the diagrams of **??** mean the same things as **??**.

Consider a square of the form

$$
\begin{array}{ccc}
\begin{pmatrix} A^- \\ A^+ \end{pmatrix} & \xRightarrow{\;\binom{f_\flat}{f}\;} & \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\[12pt]
\binom{j^\sharp}{j}\Big\Updownarrow\uparrow & & \Big\Updownarrow\uparrow\binom{k^\sharp}{k} \\[12pt]
\begin{pmatrix} C^- \\ C^+ \end{pmatrix} & \xRightarrow[\;\binom{g^\sharp}{g}\;]{} & \begin{pmatrix} D^- \\ D^+ \end{pmatrix}
\end{array}
$$

The first diagram and first equation say:

$$
\begin{array}{ccc}
A^+ & \xrightarrow{\;f\;} & B^+ \\
{\scriptstyle j}\downarrow & & \downarrow{\scriptstyle k} \\
C^+ & \xrightarrow[\;g\;]{} & D^+
\end{array}
\qquad\qquad
g(j(a^+)) = k(f(a^+)) \quad \text{for all } a^+ \in A^+,
$$

which mean the same thing. The second diagram, which takes place in $\mathbf{Ctx}_{A^+}$, is more interesting. Here's that diagram with the names we're currently using:

$$
\begin{array}{ccc}
j^*C^- & \xrightarrow{\ \ j^{\sharp}\ \ } & A^- \\
{\scriptstyle j^*g_{\flat}}\downarrow & & \downarrow{\scriptstyle f_{\flat}} \\
j^*g^*D^- == f^*k^*D^- & \xrightarrow{\ \ f^*k^{\sharp}\ \ } & f^*B^-
\end{array}
$$

Let's compute the two paths from the top left to the bottom right. First is $f_{\flat} \circ j^{\sharp} : j^*C^- \to f^*B^-$, which sends (a^+, c^-) to $f_{\flat}(a^+, j^{\sharp}(a^+, c^-))$. This is the right hand side of the second equation, so we're on the right track. The other path is $f^*k^{\sharp} \circ j^*g_{\flat}$. Recall that $j^*g_{\flat}$ sends (a^+, c^-) to $g_{\flat}(j(a^+), c^-)$, and similarly $f^*k^{\sharp}$ sends (a^+, d^-) to $k^{\sharp}(f(a^+), d^-)$. Putting them together, we send (a^+, c^-) to $k^{\sharp}(f(a^+), g_{\flat}(j(a^+), c^-))$. Therefore the commutation of this diagram means the same thing as the second equation in the definition of a square of arenas.   □

Building off of this proposition, we can think of the Grothendieck double construction as giving us a double category of arenas out of *any* indexed category.

Definition 3.5.8. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ be an indexed category. Then the category of $\mathcal{A}$-arenas is defined to be the Grothendieck double construction of $\mathcal{A}$:

$$
\mathbf{Arena}_{\mathcal{A}} := \oiint^{C \in \mathcal{C}} \mathcal{A}(C).
$$

Note that the horizontal category of $\mathbf{Arena}_{\mathcal{A}}$ is the category $\mathbf{Chart}_{\mathcal{A}}$ of $\mathcal{A}$-charts (**??**), and the vertical category of $\mathbf{Arena}_{\mathcal{A}}$ is the category $\mathbf{Lens}_{\mathcal{A}}$ of $\mathcal{A}$-lenses (**??**).

With this definition of the double category of arenas in hand, we can define a behavior in a general doctrine.

Definition 3.5.9. Let $\mathbb{D} = (\mathcal{A}, T)$ be a doctrine, and T and S two systems in this doctrine. Given an $\mathcal{A}$-chart

$$
\begin{pmatrix} f_{\flat} \\ f \end{pmatrix} : \begin{pmatrix} \mathsf{In}_{\mathsf{T}} \\ \mathsf{Out}_{\mathsf{T}} \end{pmatrix} \Rightarrow \begin{pmatrix} \mathsf{In}_{\mathsf{S}} \\ \mathsf{Out}_{\mathsf{S}} \end{pmatrix},
$$

A $\begin{pmatrix} f_{\flat} \\ f \end{pmatrix}$-*behavior* $\phi : \mathsf{T} \to \mathsf{S}$ is a map $\phi : \mathsf{State}_{\mathsf{T}} \to \mathsf{State}_{\mathsf{S}}$ so that the following is a square

in the double category **Arena**$_{\mathcal{A}}$ of $\mathcal{A}$-arenas:

$$
\begin{pmatrix} T\mathsf{State}_\mathsf{T} \\ \mathsf{State}_\mathsf{T} \end{pmatrix} \xRightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} T\mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{pmatrix}
$$

$$
\begin{pmatrix} \mathsf{update}_\mathsf{T} \\ \mathsf{expose}_\mathsf{T} \end{pmatrix} \Updownarrow\uparrow \qquad \Downarrow\uparrow\begin{pmatrix} \mathsf{update}_\mathsf{S} \\ \mathsf{expose}_\mathsf{S} \end{pmatrix}
$$

$$
\begin{pmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{pmatrix} \xRightarrow[\begin{pmatrix} f_\flat \\ f \end{pmatrix}]{} \begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix}
$$

We will often refer to this square by ϕ as well.

In **??**, we will see what composition in the double category **Arena**$_{\mathcal{A}}$ of $\mathcal{A}$-arenas let's us conclude about composition of systems and behaviors. For now, in the rest of this section, we will formally introduce the dynamical system doctrines we have been working with throughout the book, with some precise variations we can now make clear.

But before we do that, let's see how the above rather terse formal definition captures the intuitive and informal definition given in Informal Definition 1.1.2:

Informal Definition 3.5.10. A *doctrine* of dynamical systems is a particular way to answer the following questions about what it means to be a dynamical system:

1. What does it mean to be a state?
2. How should the output vary with the state — discretely, continuously, linearly?
3. Can the kinds of input a system takes in depend on what it's putting out, and how do they depend on it?
4. What sorts of changes are possible in a given state?
5. What does it mean for states to change.
6. How should the way the state changes vary with the input?

Let's see how choosing an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ and a section T constitutes a series of answers to each of these questions.

1. We had to choose the base category $\mathcal{C}$. Our space of states will be an object of this category, and so choosing the objects of this category means choosing what it means to be a state.
2. Our exposed variable $\mathsf{expose}_\mathsf{S} : \mathsf{State}_\mathsf{S} \to \mathsf{Out}_\mathsf{S}$ will be a morphism of $\mathcal{C}$, so choosing the morphisms of $\mathcal{C}$ will mean choosing how the output will vary with the state.
3. The input In_S will be an object of $\mathcal{A}(\mathsf{Out}_\mathsf{S})$, and therefore defining the objects of $\mathcal{A}(\mathsf{Out}_\mathsf{S})$ — in particular, how does they depend on Out_S — will determine how a system's space of inputs may depend on its outputs.

4. The our update map $\text{update}_S : \text{expose}_S^* I \to T\text{State}_S$ has codomain $T\text{State}_S$. Therefore, choosing object assignment of the section T tells us space of possible changes which the system may make (as depending on the state it is in, in the sense that $T\text{State}_S$ lives in a category $\mathcal{A}(\text{State}_S)$ which depends for its definition on State_S).

5. Since a behavior will involve the chart $\begin{pmatrix} T\phi \\ \phi \end{pmatrix} : \begin{pmatrix} T\text{State}_T \\ \text{State}_T \end{pmatrix} \rightrightarrows \begin{pmatrix} T\text{State}_S \\ \text{State}_S \end{pmatrix}$, choosing the action of T on maps ϕ will tell us what it means to interpret changes of state that arise from the dynamics of the system into whole behaviors of the system. We will see an elaboration of this idea when we discuss behaviors in doctrines other than the deterministic doctrine.

6. By choosing the maps of $\mathcal{A}$, we will determine what sort of map update_S is. This will determine in what sort of way the changes in state vary with parameters.

3.5.11   The deterministic doctrines

We have been speaking of the deterministic doctrine throughout this book to mean the doctrine of machines with discrete time whose next state is entirely determined by its current state and choice of parameters. But really, there have been many deterministic doctrines, one for each cartesian category $\mathcal{C}$.

Definition 3.5.12. Let $\mathcal{C}$ be a cartesian category. The *deterministic doctrine* $\mathbb{D}\text{ET}_\mathcal{C}$ in $\mathcal{C}$ is defined to be the indexed category $\mathbf{Ctx}_- : \mathcal{C}^{\text{op}} \to \mathbf{Cat}$ together with the section $C \mapsto C$ and $\phi \mapsto \phi \circ \pi_2$ defined in **??**.

Remark 3.5.13. **??** Shows that behaviors in a deterministic doctrine are precisely what we studied (and saw examples of) in **??**.

There are many different deterministic doctrines, one for each choice of cartesian category $\mathcal{C}$. For example:

- If $\mathcal{C} = \mathbf{Set}$ is the category of sets, we have discontinuous, discrete-time, determinisitic systems. These are often called "Moore machines".
- If $\mathcal{C} = \mathbf{Top}$ is the category of topological spaces, we have continuous, discrete-time, deterministic systems.
- If $\mathcal{C} = \mathbf{Man}$ is the category of smooth manifolds, then we have smooth, discrete-time, deterministic systems.
- If $\mathcal{C} = \mathbf{Meas}$ is the category of measurable spaces and measurable maps, then we have discrete-time, deterministic systems whose update is measurable.
- And so on...

Let's see how to interpret the determistic doctrine in the case that $\mathcal{C} = \mathbf{Set}$ answers the questions of Informal Definition 1.1.2.

1. A state is an element of a set.

2. The output varies as a function of the state, with constraints on what sort of function.

3. No, the kinds of inputs do not depend on the state.
4. From a state, one may transition to any other state (since $T\text{States}_S = \text{States}_S$).
5. We treat the changes and the states in the same way, interpreting a change as the next state.
6. The change in state is a function of the previous state and the input.

Exercise 3.5.14. Answer the questions of Informal Definition 1.1.2 in for the following doctrines:

1. $\mathbb{D}_{\text{ET}}\text{Top}$.
2. $\mathbb{D}_{\text{ET}}\text{Man}$.
3. $\mathbb{D}_{\text{ET}}\text{Arity}$.

◊

3.5.15   The differential doctrines

We can now define the differential doctrines, which will finally let us see the definitions of differential behavior given in **??** as different incarnations of a single, general definition.

Unlike the case with deterministic doctrines, we will not be giving a single, general definition of "differential" doctrine. We will be defining our different differential doctrines ad-hoc.[4]

We begin with the differential doctrine used to define the notion of differential system in Definition 1.2.16.

Definition 3.5.16. The *Euclidean differential doctrine* $\mathbb{E}\text{uc}$ is defined by the indexed category $\mathbf{Ctx}_- : \mathbf{Euc}^{\text{op}} \to \mathbf{Cat}$ together with the section T given by
  - $T\mathbb{R}^n := \mathbb{R}^n$, thinking of $\mathbb{R}^n$ as tangent space of a point in $\mathbb{R}^n$.
  - For a differentiable map $f : \mathbb{R}^n \to \mathbb{R}^m$, we define $Tf : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$ to be

$$Tf(p,v) := Df_p v$$

  where Df_p is the matrix of partial derivatives $\left(\frac{\partial f_i}{\partial x_j}\Big|_{x=p}\right)$. In other words, $Tf(p,v)$ is the directional derivative in direction v of f at p. The functoriality law for the section is precisely the multivariable chain law.

Exercise 3.5.17. Check that T as defined is indeed a section by referring to the multidimensional chain law. ◊

---

[4]One can give a general definition of differential doctrine that specializes to these various notions with the notion of *tangent category with display maps*. But we prefer to just describe the various categories as they come.

<cerebras_p_navigation>146   CHAPTER 3. HOW SYSTEMS BEHAVE

Remark 3.5.18. Note that if $f : \mathbb{R} \to \mathbb{R}^n$ is a function, then

$$Tf(t, v) = \frac{df}{dt}(t) \cdot v.$$

The Euclidean differential doctrine $\mathbb{E}$uc answers the questions of Informal Definition 1.1.2 in the following way:

1. A state is a n-tuple of real numbers, which is to say a point in $\mathbb{R}^n$.
2. The output is a differentiable function of the state.
3. The kind of input does not depend on the output.
4. A possible change in a state is given by a displacement vector, also in $\mathbb{R}^n$.
5. For a state to change means that it is tending in this direction. That is, it has a given derivative.
6. The changes in state vary differentiably with the input.

Let's see what behaviors look like in the Euclidean differential doctrine. Note that since the indexed category of $\mathbb{E}$uc is $\mathbf{Ctx}_- : \mathbf{Euc}^{\mathrm{op}} \to \mathbf{Cat}$, its double category of arenas is the same as for the deterministic doctrine $\mathbb{D}$et$_{\mathbf{Euc}}$. However, the definition of behavior will be different because the section is different. Let's work out what a general behavior is in $\mathbb{E}$uc explicitly.

Proposition 3.5.19. Let T and S be systems in the Euclidean differential doctrine.
A chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{smallmatrix} \right) \rightrightarrows \left(\begin{smallmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{smallmatrix} \right)$ consists of a pair of smooth functions $f : \mathsf{Out}_\mathsf{T} \to \mathsf{Out}_\mathsf{S}$ and $f_\flat : \mathsf{Out}_\mathsf{T} \times \mathsf{In}_\mathsf{T} \to \mathsf{In}_\mathsf{S}$.
A $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$-behavior is a smooth function $\phi : \mathsf{State}_\mathsf{T} \to \mathsf{State}_\mathsf{S}$ such that

$$\mathsf{expose}_\mathsf{S}(\phi(t)) = f(\mathsf{expose}_\mathsf{T}(t)).$$
$$\mathsf{update}_\mathsf{S}(\phi(t), f_\flat(\mathsf{expose}_\mathsf{T}(t), j)) = D\phi_t \mathsf{update}_\mathsf{T}(t, j)$$

Proof. This is a matter of interpreting the square

$$
\begin{CD}
\left(\begin{smallmatrix} T\mathsf{State}_\mathsf{T} \\ \mathsf{State}_\mathsf{T} \end{smallmatrix} \right) @>{\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix} \right)}>> \left(\begin{smallmatrix} T\mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{smallmatrix} \right) \\
@V{\left(\begin{smallmatrix} \mathsf{update}_\mathsf{T} \\ \mathsf{expose}_\mathsf{T} \end{smallmatrix} \right)}VV @VV{\left(\begin{smallmatrix} \mathsf{update}_\mathsf{S} \\ \mathsf{expose}_\mathsf{S} \end{smallmatrix} \right)}V \\
\left(\begin{smallmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{smallmatrix} \right) @>>{\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)}> \left(\begin{smallmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{smallmatrix} \right)
\end{CD}
$$

using by specializing [**eqn.dbl_cat_arena_square_commuting**] to the above case, using the definition of $T\phi$ in $\mathbb{E}$uc. $\qquad\square$

Example 3.5.20. Consider the following system Time in $\mathbb{E}\textsc{uc}$:

- $\mathsf{State}_{\mathsf{Time}} = \mathbb{R} = \mathsf{Out}_{\mathsf{Time}}$, and $\mathsf{expose}_{\mathsf{Time}} = \mathrm{id}$.
- $\mathsf{In}_{\mathsf{Time}} = \mathbb{R}^0$, and

$$\mathsf{update}_{\mathsf{Time}}(s, *) := 1.$$

This system represents the simple differential equation

$$\frac{ds}{dt} = 1.$$

Let S be another system in $\mathbb{E}\textsc{uc}$. A chart $\begin{pmatrix} p \\ v \end{pmatrix} : \begin{pmatrix} \mathbb{R}^0 \\ \mathbb{R} \end{pmatrix} \rightrightarrows \begin{pmatrix} \mathsf{In}_S \\ \mathsf{Out}_S \end{pmatrix}$ consists of a function $v : \mathbb{R} \to \mathsf{Out}_S$ and a function $p : \mathbb{R} \times \mathbb{R}^0 \to \mathsf{In}_S$, which is to say $p : \mathbb{R} \to \mathsf{In}_S$. This is precisely the sort of chart we need for a trajectory.

A $\begin{pmatrix} p \\ v \end{pmatrix}$-behavior $\phi : \mathsf{Time} \to \mathsf{S}$ consists of a differentiable function $\phi : \mathbb{R} \to \mathsf{State}_S$ such that the following is a square in the double category of arenas:

$$
\begin{array}{ccc}
\begin{pmatrix} \mathbb{R} \\ \mathbb{R} \end{pmatrix} & \xoverset{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}}{\Longrightarrow} & \begin{pmatrix} T\mathsf{State}_S \\ \mathsf{State}_S \end{pmatrix} \\
\begin{pmatrix} 1 \\ \mathrm{id} \end{pmatrix} \Big\downarrow\Big\uparrow & & \Big\downarrow\Big\uparrow \begin{pmatrix} \mathsf{update}_S \\ \mathsf{expose}_S \end{pmatrix} \\
\begin{pmatrix} * \\ \mathbb{R} \end{pmatrix} & \xunderset{\begin{pmatrix} p \\ v \end{pmatrix}}{\Longrightarrow} & \begin{pmatrix} \mathsf{In}_S \\ \mathsf{Out}_S \end{pmatrix}
\end{array}
$$

For this to be a square means that the following two equations hold:

$$\mathsf{expose}_S(\phi(t)) = v(t)$$
$$\mathsf{update}_S(\phi(t), p(t)) = \frac{d\phi}{dt}(t).$$

That is, a behavior of this sort is precisely a trajectory as defined in **??**

Example 3.5.21. Consider the following simple system Fix:

- $\mathsf{State}_{\mathsf{Fix}} = \mathbb{R}^0 = \mathsf{Out}_{\mathsf{Fix}}$ and $\mathsf{expose}_{\mathsf{Fix}} = \mathrm{id}$.
- $\mathsf{In}_{\mathsf{Fix}} = \mathbb{R}^0$ and $\mathsf{update}_{\mathsf{Fix}}(*, *) = *$.

This system has no state variables. Nevertheless, a chart $lensio : \begin{pmatrix} \mathsf{In}_{\mathsf{Fix}} \\ \mathsf{Out}_{\mathsf{Fix}} \end{pmatrix} \rightrightarrows \begin{pmatrix} \mathsf{In}_S \\ \mathsf{Out}_S \end{pmatrix}$ into some other system S is not trivial; it is a pair of elements $i \in \mathsf{In}_S$ and $o \in \mathsf{Out}_S$.

A $\begin{pmatrix} i \\ o \end{pmatrix}$-behavior $s : \mathsf{Fix} \to \mathsf{S}$ consists of a differentiable function $s : \mathbb{R}^0 \to \mathsf{State}_S$ — which is to say a state $s \in \mathsf{State}_S$ – such that the following is a square in the double

category of arenas:

$$
\begin{array}{ccc}
\begin{pmatrix} \mathbb{R}^0 \\ \mathbb{R}^0 \end{pmatrix}
& \xRightarrow{\;\;\begin{pmatrix} Ts \\ s \end{pmatrix}\;\;}
& \begin{pmatrix} T\mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{pmatrix} \\[2em]
\begin{pmatrix} 0 \\ \mathrm{id} \end{pmatrix}\Big\updownarrow\uparrow
& & \Big\updownarrow\uparrow\begin{pmatrix} \mathsf{update}_\mathsf{S} \\ \mathsf{expose}_\mathsf{S} \end{pmatrix} \\[2em]
\begin{pmatrix} * \\ \mathbb{R} \end{pmatrix}
& \xRightarrow[\;\begin{pmatrix} p \\ v \end{pmatrix}\;]{}
& \begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix}
\end{array}
$$

Now, $s : \mathbb{R}^0 \to \mathsf{State}_\mathsf{S}$ is a constant function, so $T(s, *) = 0$. Therefore, for this to be a square means that the following two equations hold:

$$\mathsf{expose}_\mathsf{S}(s) = o.$$
$$\mathsf{update}_\mathsf{S}(s, i) = 0.$$

This says that S is not changing in state s on input i, or that s is a steady state of S for input i as in **??**.

The Euclidean differential doctrine $\mathbb{E}$uc is a special case of a *cartesian differential doctrine*. The category **Euc** is a *cartesian differential category*, and for any cartesian differential category we can make a cartesian differential doctrine. We won't define the notion of cartesian differential category here, as the definition is a bit involved. See [BCS09] for a comprehensive introduction.

Definition 3.5.22. For any cartesian differential category $\mathcal{C}$ with differential operator T, we have a doctrine $\mathbb{C}$ᴀʀᴛDɪꜰꜰ defined by the indexed category $\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ together with the section given by T.

Now, we would like to also show that periodic orbits are behaviors in a differential doctrine, but we're a bit stuck. In the Euclidean doctrine, there's no way to ensure that a trajectory $\phi : \mathbb{R} \to \mathbb{R}^n$ is periodic. Recall that ϕ being periodic means that

$$\phi(t) = \phi(t + k)$$

for some $k \in \mathbb{R}$ called the period. If ϕ is periodic, then it descends to the quotient $\mathbb{R}/k\mathbb{Z}$, which is a circle of radius $\frac{k}{2\pi}$. If we could define $\mathsf{State}_{\mathsf{Orbit}_k}$ to be $\mathbb{R}/k\mathbb{Z}$, then a trajectory $\hat{\phi} : \mathsf{State}_{\mathsf{Orbit}_k} \to \mathsf{State}_\mathsf{S}$ would be precisely a periodic trajectory $\phi : \mathbb{R} \to \mathsf{State}_\mathsf{S}$. To make this expansion of representable behaviors, we will need to move beyond Euclidean spaces.

Our first guess might be to simply change out the category **Euc** of Euclidean spaces for the category **Man** of smooth manifolds in the definition of $\mathbb{E}$uc. Certainly, **Man** is a cartesian category and so $\mathbf{Ctx} : \mathbf{Man}^{\mathrm{op}} \to \mathbf{Cat}$ is a perfectly good indexed category.

But the tangent bundle of a general smooth manifold is not necessarily a product like it is for $\mathbb{R}^n$. So we would need to change our indexed category as well!

Now, strictly speaking we don't *have* to do this if we only want to add circles, because circles have a trivial tangent bundle. But it will turn out that defining the section T will involve choosing, once and for all, a particular trivialization of the tangent bundle of the circle and expressing all derivatives in terms of this. It will end up much easier to simply jump over to manifolds.

We recall that to any manifold M there is an associated tangent bundle $\pi : TM \to M$. A vector field on a manifold M is a section $v : M \to TM$ of the tangent bundle. We recall a bit about tangent bundles now.

Proposition 3.5.23. The assignment of a manifold M to its tangent space TM is functorial in that it extends to an assignment

$$f : M \to N \mapsto Tf : TM \to TN$$

which, on Euclidean spaces gives $Tf(p, v) = Df_p v$. Furthermore, the tangent bundle $\pi : TM \to M$ is natural in the the diagram

$$
\begin{array}{ccc}
TM & \xrightarrow{\ Tf\ } & TN \\
\pi \downarrow & & \downarrow \pi \\
M & \xrightarrow[\ f\ ]{} & N
\end{array}
$$

commutes.

There is something special about the tangent bundle which allows it to be re-indexed to a different manifold: it is a *submersion*. Not all pullbacks of manifolds exist, but all pullbacks of submersions exist and are submersions.

Definition 3.5.24. A *submsersion* $\phi : M \to N$ is a map of manifolds for which $T_p\phi : T_pM \to T_{\phi(p)}N$ is surjective for each $p \in M$.

We note that every isomorphism is a submersion, and that the composite of submersions is a submersion.

Lemma 3.5.25. Let $\phi : A \to B$ be a submersion. Then for any $f : C \to B$, the set theoretic pullback $A \times_B C = \{(a, c) \in A \times C \mid \phi(a) = f(c)\}$ may be given the structure of a smooth manifold so that the two projections $A \times_B C \to A$ and $A \times_B C \to C$ are smooth, and so that the resulting square is a pullback square in the category of manifolds. Furthermore, the projection $f^*\phi : A \times_B C \to C$ is also a submersion.

In short, we say that pullbacks of submersions exist and are themselves submersions.

This situation arises enough that we can give an abstract definition of it.

Definition 3.5.26. Let $\mathcal{C}$ be a category. A class of *display maps* in $\mathcal{C}$ is a class of maps $\mathcal{D}$ which satisfies the following:

- Every isomorphism is in $\mathcal{D}$.
- $\mathcal{D}$ is closed under composition. If f and g are composable arrows in $\mathcal{D}$, then $f \,\mathring{,}\, g$ is in $\mathcal{D}$.
- $\mathcal{D}$ is closed under pullback. If $f : A \to B$ is in $\mathcal{D}$ and $g : C \to B$ is any map, then the pullback $g^*f : A \times_B C \to C$ exists and is in $\mathcal{D}$.

A *category with display maps* $(\mathcal{C}, \mathcal{D})$ is a category $\mathcal{C}$ equipped with a class $\mathcal{D}$ of display maps.

We have seen that $(\mathbf{Man}, \mathbf{Subm})$ is a category with display maps by **??**. There are two other common classes of display map categories.

- If $\mathcal{C}$ has all pullbacks, then we may take all maps to be display maps.
- If $\mathcal{C}$ is cartesian, then we may take the product projections to be the display maps.

The first of these obviously works, but the second requires a bit of proof (and to be a bit more carefully defined).

Proposition 3.5.27. Let $\mathcal{C}$ be a cartesian category. Let $\mathcal{D}$ denote the class of maps $f : A \to B$ for which there exists a $C \in \mathcal{C}$ and an isomorphism $i : A \to C \times B$ for which $f = i \,\mathring{,}\, \pi_2$. That is, $\mathcal{D}$ is the class of maps which are product projections up to an isomorphism. Then $(\mathcal{C}, \mathcal{D})$ is a display map category.

Proof. We verify the conditions

- If $i : A \to B$ is an isomorphism, then $f \,\mathring{,}\, \pi_2^{-1} : A \to 1 \times B$ is also an isomorphism. By construction, $f = f \,\mathring{,}\, \pi_2^{-1} \,\mathring{,}\, \pi_2$, so every isomorphism is a product projection up to isomorphism.
- Suppose that $f : A \to B$ is isomorphic to a product projection $\pi_2 : C \times B \to B$ in that $f = i \,\mathring{,}\, \pi_2$, and $g : B \to X$ is isomorphic to a product projection $\pi_2 : Y \times X \to X$ in that $g = j \,\mathring{,}\, \pi_2$. We may then see that $f \,\mathring{,}\, g$ is a product projection up to isomorphism by contemplating the following commutative diagram:

- Let $f : A \to B$ be a equal to $i \,\mathring{,}\, \pi_2$ with $i : A \to C \times B$ an isomorphism. Let $k : X \to B$ be any other map. We will show that $\pi_2 : C \times X \to X$ fits in a pullback

diagram as follows:
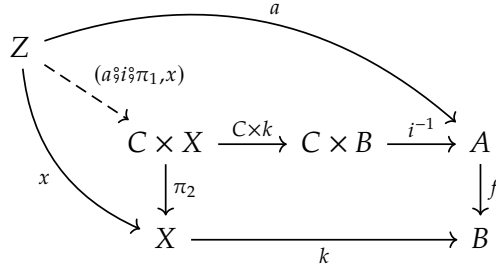
The square commutes since $i^{-1} \mathbin{\fatsemi} f = \pi_2 : C \times B \to B$. We see that it satisfies the universal property by making the definition of the dashed arrow given in the diagram. The lower triangle commutes by definition, so consider the upper triangle, seeking to show that $a = (a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_1, x) \mathbin{\fatsemi} (C \times k) \mathbin{\fatsemi} i^{-1}$. We calculate:

$$
\begin{aligned}
(a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_1, x) \mathbin{\fatsemi} (C \times k) \mathbin{\fatsemi} i^{-1} &= (a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_1, x \mathbin{\fatsemi} k) \mathbin{\fatsemi} i^{-1} \\
&= (a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_1, a \mathbin{\fatsemi} f) \mathbin{\fatsemi} i^{-1} \\
&= (a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_1, a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_2) \mathbin{\fatsemi} i^{-1} \\
&= a \mathbin{\fatsemi} i \mathbin{\fatsemi} i^{-1} \\
&= a.
\end{aligned}
$$

Now, if $z : Z \to C$ were any other map so that $(z, x) \mathbin{\fatsemi} C \times k \mathbin{\fatsemi} i^{-1} = a$, we would have $(z, a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_2) \mathbin{\fatsemi} i^{-1} = a$, or $(z, a \mathbin{\fatsemi} i \mathbin{\fatsemi} pi_2) = a \mathbin{\fatsemi} i$, from which we may deduce that $z = a \mathbin{\fatsemi} i \mathbin{\fatsemi} \pi_1$. This proves uniqueness of the dashed map. □

We can now construct the indexed category that will form the basis of our new differential doctrine. We will do so at the general level of display map categories, since the construction relies only on this structure.

Definition 3.5.28. Let $(\mathcal{C}, \mathcal{D})$ be a category with display maps. The indexed category $\mathcal{D} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ is defined as follows:

- To each object $C \in \mathcal{C}$, $\mathcal{D}(M)$ is the category with objects the display maps $\phi : E \to C$ and maps $f : E \to E'$ such that $f \mathbin{\fatsemi} \phi' = \phi$. That is, it is the full subcategory of the slice category over C spanned by the display maps.
- To each map $f : C' \to C$, we associate the functor $f^* : \mathcal{D}(N) \to \mathcal{D}(M)$ given by taking the pullback along f.

We note that this is functorial up to coherent isomorphism by the uniqueness (up to unique isomorphism) of the pullback.

We can specialize this to the category of smooth manifolds with submersions the display maps

Definition 3.5.29. The indexed category **Subm** : **Man**$^{\text{op}}$ → **Cat** is defined as follows:
- To each manifold M, **Subm**(M) is the category of submersions $\phi : E \to M$ and maps $f : E \to E'$ such that $f \, \mathring{\,}\, \phi' = \phi$.
- To each map $f : M \to N$, we associate the functor $f^* : $ **Subm**$(N) \to$ **Subm**(M) given by taking the pullback along f.

Exercise 3.5.30.   Let C be a cartesian category, and equip it with the class $\mathcal{D}$ of maps which are isomorphic to product projections, as in **??**. Prove that $\mathcal{D} : C^{\text{op}} \to$ **Cat** is equivalent, as an indexed category, to **Ctx**$_-$: $C^{\text{op}} \to$ **Cat**.   ◊

If $(C, \mathcal{D})$ is a category with display maps, then the category of charts of $\mathcal{D} : C^{\text{op}} \to$ **Cat** is easy to understand in terms of $\mathcal{D}$.

Proposition 3.5.31. Let $(C, \mathcal{D})$ be a category with display maps. Then the category **Chart**$_{\mathcal{D}} = \int^{C \in C} \mathcal{D}(C)$ of charts for $\mathcal{D} : C^{\text{op}} \to$ **Cat** is equivalent to the category whose objects are display maps and whose morphisms are commutative squares between them.

Proof. An object $\left(\begin{smallmatrix} a^- \\ A^+ \end{smallmatrix} \right)$ of the category of charts is a pair consisting of an object $A^+ \in C$ and a display map $a^- : A^- \to A^+$ in $\mathcal{D}(A^+)$. But A^+ is determined, as the codomain, by a^-; so the objects of the category of charts are in bijection with the display maps. We then show that the charts are similarly in bijection with the squares between display maps.

A chart $\left(\begin{smallmatrix} f_{\flat} \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} a^- \\ A^+ \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} b^- \\ B^+ \end{smallmatrix} \right)$ for this indexed category is a pair consisting of a map $f : A^+ \to B^+$ in C and a triangle

$$
\begin{array}{ccc}
A^- & \xrightarrow{\quad f_{\flat} \quad} & f^* B^- \\
 & \searrow{\scriptstyle a^-} \quad \swarrow{\scriptstyle f^* b^-} & \\
 & A^+ &
\end{array}
$$

By the universal property of the pullback, this data is equivalently given by the data of a square

$$
\begin{array}{ccc}
A^- & \xrightarrow{\hat{f}_{\flat}} & B^- \\
{\scriptstyle a^-}\downarrow & & \downarrow{\scriptstyle b^-} \\
A^+ & \xrightarrow{f} & B^+
\end{array}
$$

Now, consider a composite square

$$
\begin{array}{ccccc}
A^- & \xrightarrow{\hat{f}_{\flat}} & B^- & \xrightarrow{\hat{g}_{\flat}} & C^- \\
{\scriptstyle a^-}\downarrow & & {\scriptstyle b^-}\downarrow & & \downarrow{\scriptstyle c^-} \\
A^+ & \xrightarrow{f} & B^+ & \xrightarrow{g} & C^+
\end{array}
$$

We can see that the arrow $f_\flat \, \mathring{,} \, f^* g_\flat : A^- \to f^* g^* B^-$ composes with the projections from the pullbacks to give the top half of the outer square, and therefore it is the unique map into the pullback induced by the outer square. $\qquad \square$

Corollary 3.5.32. Let $(\mathcal{C}, \mathcal{D})$ be a category with display maps. To give a section of $\mathcal{D} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$, it suffices to give an endofunctor $T : \mathcal{C} \to \mathcal{C}$ together with a natural transformation $\pi : T \to \mathrm{id}_\mathcal{C}$ whose components are all display maps.

Proof. Such an endofunctor T with natural transformation π gives us a functor $C \mapsto \pi : TC \to C$ going from $\mathcal{C}$ to the category of display maps in $\mathcal{C}$ and squares between them. This functor will assign to each $f : C' \to C$ the naturality square

$$
\begin{array}{ccc}
TC' & \xrightarrow{Tf} & TC \\
\pi \downarrow & & \downarrow \pi \\
C' & \xrightarrow{\quad f \quad} & C
\end{array}
$$

We note that the evident projection of the codomain composes with this functor to give $\mathrm{id}_\mathcal{C}$. By **??**, this is equivalent to giving such a functor into $\mathbf{Chart}_\mathcal{D}$, which, by **??** is equivalent to giving a section of $\mathcal{D}$. $\qquad \square$

We may therefore define a doctrine associated to any category with display maps $(\mathcal{C}, \mathcal{D})$ with such an endofunctor $T : \mathcal{C} \to \mathcal{C}$ and natural transformation $\pi : T \to \mathrm{id}_\mathcal{C}$ whose components are all display maps.

Definition 3.5.33. Let $(\mathcal{C}, \mathcal{D})$ be a category with display maps and let $T : \mathcal{C} \to \mathcal{C}$ be an endofunctor and $\pi : T \to \mathrm{id}_\mathcal{C}$ a natural transformation whose components are all display maps. Then this data forms a doctrine $\mathbb{D}\mathrm{ISP}_{\mathcal{D}, T}$ given by $\mathcal{D} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ and the section induced by sending C to $\pi : TC \to C$ in $\mathcal{D}(C)$.

With one last lemma, we will finally be able to define our general differential doctrine.

Lemma 3.5.34. The tangent bundle $\pi : TM \to M$ of a manifold M is a submersion.

Definition 3.5.35. The *general differential doctrine* $\mathbb{D}\mathrm{IFF}$ is defined to be the display category doctrine $\mathbb{D}\mathrm{ISP}_{\mathbf{Subm}, T}$ associated to the $\mathbf{Subm} : \mathbf{Man}^{\mathrm{op}} \to \mathbf{Cat}$ and the tangent bundle functor T.

A dynamical system in the general differential doctrine $\mathbb{D}\mathrm{IFF}$ consists of a state space States, and output space Out_S, but then a *submersion of inputs* $\pi_{\mathsf{Ins}} : \mathsf{Ins} \to \mathsf{Out}_S$. We can think of π as assigning to each input the output that it is valid for. The update then has signature

$$
\mathrm{update}_S : \mathrm{expose}_S^* \pi_{\mathsf{Ins}} \to \pi_{T\mathsf{States}}
$$

which is to say that it is a triangle of the form

$$
\begin{array}{ccc}
\text{States} \times_{\text{Out}_S} \text{In}_S & \xrightarrow{\;\text{update}_S\;} & T\text{States} \\
& \searrow^{\pi_1} \qquad \swarrow_{\pi} & \\
& \text{States} &
\end{array}
$$

which assigns to each state-input pair (s, i) where i is valid given the state s in the sense that $\text{expose}_S(s) = \pi_{\text{In}_S}(i)$ to a tangent vector at s. There will be a lot more to say about dynamical systems in which the sorts of inputs allowed depend on the current output in later chapters, but for now we'll have to content ourselves with a simple example.

The general differential doctrine $\mathbb{D}\text{IFF}$ answers the questions of Informal Definition 1.1.2 in the following way:

1. A state is a point in a smooth manifold.
2. The output is a smooth function of the state.
3. The kind of input can dependend on the output, but it does so continuously (in the sense that the assignment sending an input to the output it is valid for is a submersion).
4. A possible change in a state is given by a tangent vector.
5. For a state to change means that it is tending in this direction. That is, it has derivative equal to the given tangent vector.
6. The changes in state vary differentiably with the input.

Example 3.5.36. Let's see an example of a situation where the inputs may differ over different outputs. Suppose we have a robot on a distant planet, and we are directing it. When we tell it to move in a direction, the robot will move in the given direction at a given speed k. We want to keep track of the position of the robot as it moves around the planet.

We can model this situation as follows: since the surface of the planet is a sphere and we want to keep track of where the robot is, we will let $\text{States} = S^2$ be a sphere. We will also have the robot reveal its position to us, so that $\text{Out}_S = S^2$ and $\text{expose}_S = \text{id}$.

Now, in any given position $p \in \text{Out}_S$, we want the space of inputs In_{S_p} valid for p to be the directions we can give to the robot: that is to say, $\text{In}_{S_p} \cong S^1$ should form a circle. However, we want these directions to be directions that the robot could actually travel, so we will let $\text{In}_{S_p} = \{v \in T_p\text{Out}_S \mid |v| = 1\}$ be the unit circle in the tangent space at p. Then we may describe the fact that the robot moves in the direction we tell it by defining

$$
\text{update}_S(s, i) = ki.
$$

We note that any system S in the Euclidean differential doctrine can be considered as a system in the general differential doctrine by defining the bundle of inputs to the $\pi_1 : \text{Out}_S \times \text{In}_S \to \text{Out}_S$ and noting that the pullback of a product projection is a product projection, so that we may take the domain of the new update $u : \text{expose}_S^* \pi_1 \to \pi_{T\text{States}}$

to be $\mathsf{State_S} \times \mathsf{In_S}$, just as it was. We may then define $u(s, i) = (s, \mathsf{update_S}(s, i))$, equating $T\mathsf{State_S} = T\mathbb{R}^n$ with $\mathbb{R}^n \times \mathbb{R}^n$. Later, when we discuss change of doctrine, we will see that this follows from a *morphism of doctrines* $\mathbb{E}\mathrm{UC} \to \mathbb{D}\mathrm{IFF}$.

We can now describe periodic orbits as behaviors in the general differential doctrine.

Example 3.5.37. Let $\mathsf{Clock_k}$ be the system in $\mathbb{D}\mathrm{IFF}$ with:
- State space $\mathsf{State_{Clock_k}} = \mathbb{R}/k\mathbb{Z}$,
- Output space $\mathsf{Out_{Clock_k}} = \mathbb{R}/k\mathbb{Z}$ with $\mathsf{expose_{Clock_k}} = \mathrm{id}$.
- Input bundle the identity $\mathsf{In_{Clock_k}} = \mathrm{id}_{\mathsf{Out_{Clock_k}}}$.
- Update $\mathsf{update_{Clock_k}} : \mathsf{State_{Clock_k}} \to T\mathsf{State_{Clock_k}}$ the assigning each state s to the vector $Tq(1)$, the pushforward of the constant vector 1 on $\mathbb{R}$ by the quotient $q : \mathbb{R} \to \mathbb{R}/k\mathbb{Z}$.

The universal property of $\mathbb{R}/k\mathbb{Z}$ says that a smooth function $\gamma : \mathbb{R} \to M$ factors through $q : \mathbb{R} \to \mathbb{R}/k\mathbb{Z}$ if and only if $\gamma(kn) = \gamma(0)$ for all $n \in \mathbb{Z}$.

A chart for $\mathsf{Clock_k}$ into a system S is a square

$$
\begin{array}{ccc}
\mathbb{R}/k\mathbb{Z} & \xrightarrow{\ p\ } & \mathsf{In_S} \\
\| & & \downarrow{\scriptstyle \pi} \\
\mathbb{R}/k\mathbb{Z} & \xrightarrow{\ v\ } & \mathsf{Out_S}
\end{array}
$$

By the various universal properties involved, this is the same data as a pair of maps $\hat{p} : \mathbb{R} \to \mathsf{In_S}$ and $\hat{v} : \mathbb{R} \to \mathsf{Out_S}$ for which $\hat{p}(nk) = 0$ and $\hat{v}(nk) = 0$ for all $n \in \mathbb{Z}$, and for which $\pi \circ \hat{i} = \hat{v}$.

Now, a behavior $\phi : \mathsf{Clock_k} \to S$ is a square

$$
\begin{array}{ccc}
\begin{pmatrix} T(\mathbb{R}/k\mathbb{Z}) \\ \mathbb{R}/k\mathbb{Z} \end{pmatrix} & \overset{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}}{\Longrightarrow} & \begin{pmatrix} T\mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\[2em]
\begin{pmatrix} 1 \\ \mathrm{id} \end{pmatrix} \Big\Updownarrow \Big\uparrow & & \Big\Updownarrow \Big\uparrow \begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} \\[2em]
\begin{pmatrix} * \\ \mathbb{R} \end{pmatrix} & \underset{\begin{pmatrix} p \\ v \end{pmatrix}}{\Longrightarrow} & \begin{pmatrix} \mathsf{In_S} \\ \mathsf{Out_S} \end{pmatrix}
\end{array}
$$

Let $\hat{\phi} : \mathbb{R} \to \mathsf{State_S}$ be $\phi \circ q$. First, this being a square means that $\mathsf{expose_S} \circ \phi = v$, or that $\mathsf{expose_S}(\hat{\phi}(t)) = v(t)$. Second, we have that

$$
\phi^* \mathsf{update_S} \circ p = T\phi \circ \mathsf{update_{Clock_k}},
$$

which is to say
$$\text{update}_S(\phi(t), p(t)) = T\phi(Tq(1)).$$

Re-expressing this in terms of $\hat{\phi} = \phi \circ q$, we see that this means that

$$\text{update}_S(\hat{\phi}(t), \hat{p}(t)) = \frac{d\hat{\phi}}{dt}.$$

This says that $\hat{\phi}$ is a trajectory for the system. Since by definition $\hat{\phi}$ is periodic, and any such periodic map would factor through q, we may conclude that behaviors $\text{Clock}_k \rightarrow S$ are periodic orbits (with period dividing k) of S.

3.5.38 Dependent deterministic doctrines

The doctrine $\mathbb{D}\text{ISP}_{(\mathcal{C},T)}$ of display maps can also help us describe deterministic doctrines in which the sorts of input a system can accept depend on the output that system is currently exposing. These are called *dependent* deterministic systems.

Definition 3.5.39. Let $(\mathcal{C}, \mathcal{D})$ be a category with display maps (**??**) and finite products, and suppose that product projections $\pi_1 : C \times D \rightarrow C$ are display maps. Then we have a section $TC = C \times C \xrightarrow{\pi_1} C$, and so we may define the depdendent deterministic doctrine $\mathbb{D}\text{PD}\text{ET}_C$ to be the display map doctrine $\mathbb{D}\text{ISP}_{(\mathcal{C},\mathcal{D}),T}$.

Let's understand dependent deterministic systems in the category of sets with every map taken as a display map.

Definition 3.5.40. A *dependent deterministic* system S in the category of sets consists of:
- A set States_S of states,
- A set Outs_S of outputs, and for each output $o \in \text{Outs}_S$, a set $\text{Ins}_S(o)$ of inputs valid in output o. If we define $\text{Ins}_S := \sum_{o \in \text{Outs}_S} \text{Ins}_S(o)$ to be the disjoint union of all of these ouput sets, then we can package this assignment $o \mapsto \text{Ins}_S(o)$ into a function $\text{Ins}_S \rightarrow \text{Outs}_S$ which sends an input to the output it is valid in. That is, the interface of a dependent system is a dependent set.
- An exposed variable $\text{expose}_S : \text{States}_S \rightarrow \text{Outs}_S$.
- For every state $s \in \text{States}_S$, an update function $\text{update}_S(s, -) : \text{Ins}_S(\text{expose}_S(s)) \rightarrow \text{States}_S$ sending an input which is valid in the output exposed by s to the next state.

Example 3.5.41. Consider the following simple example of a dependent system. A Diner accepts orders when it is open, but when it is closed it doesn't. The set of possible outputs, or *orientations* of the diner in it's environment, is $\text{Outs}_S = \{\texttt{open}, \texttt{closed}\}$. When the diner is closed it excepts no input, so $\text{Ins}_S(\texttt{closed}) = \{\texttt{tick}\}$ will simply represent the ticking of time; other the other hand, when the diner is open then $\text{Ins}_S(\texttt{open})$ will be

the set of possible orders.

Dependent deterministic systems in **Set** are tremendously interesting in their own right, and we won't be able to do them justice here.

3.5.42   Non-deterministic doctrines

In this section, we will define the non-deterministic doctrines. We've already done most of the work for this back in **??**. In particular, in **??**, we showed that to every commutative monad $M : \mathcal{C} \to \mathcal{C}$ on a cartesian category, there is a monoidal strict indexed category

$$\mathbf{Ctx}^M : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$$

sending each object $C \in \mathcal{C}$ to the category $\mathbf{Ctx}_C^M$ of Klesli maps $C \times X \to MY$ in the context of C. We will take this to be the indexed category underlying the doctrine of non-deterministic systems associated to M; it remains to construct a section T.

Proposition 3.5.43. The assigment defined by
- $C \in \mathcal{C}$ is assigned to $C \in \mathbf{Ctx}_C^M$, and
- $f : C' \to C$ is assigned to

$$C' \times C' \xrightarrow{\pi_2 \,\, f \,\, \eta} MC$$

yields a section $T : \mathcal{C} \to \mathbf{Ctx}_-^M$.

Proof. We see immediately that $T\mathrm{id}_C = \mathrm{id}_{TC}$. It remains to show that for $f : C' \to C$ and $g : C'' \to C'$, we have

$$Tg \,\, \S \,\, g^* Tf = T(g \,\, \S \,\, f).$$

In the do notation, the composite on the left is given by

$$(c_1'', c_2'') \mapsto \boxed{\begin{array}{l} \mathbf{do} \\ \quad c' \leftarrow \eta(g(c_2'')) \\ \quad \eta(f(c')) \end{array}} = \eta(f(g(c_2'')))$$

which is the right hand side. $\qquad\qquad\square$

Definition 3.5.44. Let $M : \mathcal{C} \to \mathcal{C}$ be a commutative monad on a cartesian category $\mathcal{C}$. The *M-flavored non-deterministic doctrine* $\mathbb{N}\textsc{on}\textsc{Det}_M$ is defined to be the indexed category $\mathbf{Ctx}_-^M : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ together with section defined in **??**.

The non-deterministic doctrine $\mathbb{N}\textsc{on}\textsc{Det}_M$ answers the questions of Informal Definition 1.1.2 in the following way (taking $\mathcal{C} = \mathbf{Set}$ for concreteness):
1. A state is an element of a set.
2. The output is a deterministic function of the state.

3. The kind of input does not depend on the output.
4. A possible change in a state is given by an M-distribution over states (element of $M\mathsf{State}_\mathsf{S}$).
5. A state changes by transitioning into another state.
6. The changes in state vary arbitrarily with input.

Exercise 3.5.45.   Answer these questions more specifically for the following doctrines:
1. The non-deterministic Moore machine doctrine $\mathbb{N}\textsc{on}\textsc{Det}_\textsc{P}$.
2. The probabilistic doctrine $\mathbb{N}\textsc{on}\textsc{Det}_\textsc{D}$.
3. The worst-case cost doctrine $\mathbb{N}\textsc{on}\textsc{Det}_\textbf{Cost}$.

◊

Behaviors in non-deterministic doctrines tend to be a little strict. This is because the notion of trajectory is a bit more subtle in the non-deterministic case. When does a sequence $s : \mathbb{N} \to \mathsf{State}_\mathsf{S}$ constitute a trajectory of the system S? Is it when s_t *will* transition to s_{t+1} (in that $\mathrm{update}_\mathsf{S}(s_t, i_t) = \eta(s_{t+1})$)? Or perhaps when it *can* transition that way — but how do we express this notion in general?

While the notion of behavior we have given in this chapter works well for deterministic and differential doctrines, it does not work as well for non-deterministic doctrines. Instead of asking whether or not a sequence of states is a trajectory, we might instead want to ask how possible or likely it is for such a sequence of states to occur through an evolution of the system. Figuring out how to express this idea nicely and generally remains future work.

On the other hand, simulations of non-deterministic doctrines remain interesting, because they tell us when we might be able to use a simpler model for our system without changing the exposed behavior.

Jaz: I should include an example of non-deterministic simulation here.

3.6   Restriction of Doctines

Now that we have a concise, formal definition of dynamical system doctrine, we can begin to treat doctrines as mathematical objects. In this section, we will look at a simple way to construct a new doctrine from an old one: *restriction* along a functor.

We will use restrictions of doctrines in order to more precisely control some of the upcoming functoriality results. Often, we will only be able to prove a theorem by restricting the doctrine beforehand.

Since a doctrine $\mathbb{D}$ consists of an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ together with a section T, if we have a functor $F : \mathcal{D} \to \mathcal{C}$ then we should be able to produce a new doctrine by composing $\mathcal{A}$ and T with F. We call this new doctrine $\mathbb{D}|_F$ the *restriction* of $\mathbb{D}$ along F.

Definition 3.6.1. Let $\mathbb{D} = (\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, T)$ be a doctrine of dynamical systems. For any functor $F : \mathcal{D} \to \mathcal{C}$, we have a new doctrine

$$\mathbb{D}|_F := (\mathcal{A} \circ F^{\mathrm{op}}, T \circ F)$$

where $T \circ F$ is the section given by
- $(T \circ F)(D) := T(FD)$, and
- $(T \circ F)(f) := T(Ff)$, which may see has the correct codomain since $(\mathcal{A} \circ F^{\mathrm{op}})(f)(T \circ F)(D) = \mathcal{A}(Ff)T(FD)$.

Since an indexed category is no more than a functor into **Cat**, $\mathcal{A} \circ F^{\mathrm{op}}$ is an indexed category. It only remains to check that $T \circ F$ as defined is indeed a section of the Grothendieck construction of $\mathcal{A} \circ F^{\mathrm{op}}$; this calculation is a straightforward unfolding of definitions.

Example 3.6.2. In the next chapter we will see a few approximation methods as ways of changing doctrine.

However, these approximations do not preserve all features of the doctrines; in general, they are only exact for a restricted class of functions. For example, the Euler method which approximates a differential equation

$$\frac{ds}{dt} = F(s, p)$$

on a Euclidean space by the discrete-time update function

$$u(s, p) = s + \epsilon F(s, p)$$

only exactly reproduces *affine* behaviors of systems. Being affine is a rather severe restriction on the behavior of a dynamical system, but it does allow the important case of steady states.

In order to capture Euler approximation as a change of doctrine in the exact manner to be explored in the next chapter, we therefore need to restrict the Euclidean differential doctrine $\mathbb{E}$uc to *affine* functions. Recall that the indexing base of $\mathbb{E}$uc is the category **Euc** of Euclidean spaces and differentiable functions. We may therefore take our restriction functor to be the inclusion **Aff** $\hookrightarrow$ **Euc** of affine functions between Euclidean spaces.

Now that we have the formalities out of the way, let's understand what restricting a doctrine means for the theory of systems in it. Because we have changed the indexing base for the doctrine, we have changed the objects of states and exposed variables, and the bottom part of both the lenses and charts.

In particular, the object of states of a $\mathbb{D}|_F$-doctrine is now an object of $\mathcal{D}$ and not of $\mathcal{C}$. The exposed variable $\mathrm{expose}_S : \mathrm{State}_S \to \mathrm{Out}_S$ is now a map in $\mathcal{D}$. Furthermore,

and rather drastically, the underlying map $\phi : \mathsf{State}_\mathsf{T} \to \mathsf{State}_\mathsf{S}$ of a behavior is also a map in $\mathcal{D}$.

Example 3.6.3. Continuing from **??**, we may consider what a behavior represented by the system $\mathsf{T} = \begin{pmatrix} t \mapsto 1 \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} \mathbb{R} \\ \mathbb{R} \end{pmatrix} \rightrightarrows \begin{pmatrix} 1 \\ \mathbb{R} \end{pmatrix}$ is in the restricted doctrine $\mathbb{E}\textsc{uc}|_{\mathbf{Aff}}$.

Since T represents trajectories in $\mathbb{E}\textsc{uc}$, it will represent trajectories in $\mathbb{E}\textsc{uc}|_{\mathbf{Aff}}$. However, we have restricted the underlying map $s : \mathbb{R} \to \mathsf{State}_\mathsf{S}$ to lie in **Aff** — that is, to be affine. There are not often affine solutions to general differential equations, so for the most part we will simply find that a system S has *no* trajectories (in this restricted doctrine), or very few.

However, any constant function is affine; for this reason, all steady states are affine functions, and so remain behaviors in this restricted doctrine.

3.7   Summary and Futher Reading

In this chapter, we looked at a variety of behaviors of systems in different doctrines and saw that they could all be represented by the same equations relating charts with lenses. We saw how behaviors can be represented by dynamical systems of particular shapes — trajectories are represented by timelines, steady states by a single fixed point, periodic orbits by clocks, etc. We introduced the double category of arenas to organize charts and lenses, and finally gave a formal definition of dynamical system doctrine.

The notion of dynamical systems doctrine, the double category of arenas in a given doctrine, and the definition of behavior of system that these enable are novel contributions of this book. For a summary account, see [Jaz21].

Change of Doctrine

4.1 Introduction

In the last chapter, we saw a general formulation of the notion of behavior of system and precise definition of the notion of dynamical system doctrine. Let's recall the definition of dynamical system doctrine.

Definition 4.1.1. A *dynamical system doctrine* consists of an indexed category $\mathcal{A} : \mathcal{C}^{op} \to$ **Cat** together with a section T.

This concise definition packs a big punch. Describing a dynamical system doctrine amounts to answering the informal questions about what it means to be a system:

Informal Definition 4.1.2. A *doctrine* of dynamical systems is a particular way to answer the following questions about what it means to be a dynamical system:
1. What does it mean to be a state?
2. How should the output vary with the state — discretely, continuously, linearly?
3. Can the kinds of input a system takes in depend on what it's putting out, and how do they depend on it?
4. What sorts of changes are possible in a given state?
5. What does it mean for states to change.
6. How should the way the state changes vary with the input?

Constructing a doctrine is no small thing. But once we have a doctrine, we have may work in its double category of arenas to quickly derive a few compositionality results about systems.

Definition 4.1.3. Let $\mathcal{A} : \mathcal{C}^{op} \to$ **Cat** be an indexed category. Then the category of

$\mathcal{A}$-arenas is defined to be the Grothendieck double construction of $\mathcal{A}$:

$$\mathbf{Arena}_{\mathcal{A}} := \oiint^{C \in \mathcal{C}} \mathcal{A}(C).$$

Note that the horizontal category of $\mathbf{Arena}_{\mathcal{A}}$ is the category $\mathbf{Chart}_{\mathcal{A}}$ of $\mathcal{A}$-charts (**??**), and the vertical category of $\mathbf{Arena}_{\mathcal{A}}$ is the category $\mathbf{Lens}_{\mathcal{A}}$ of $\mathcal{A}$-lenses (**??**).

We are now in peak category theory territory: the statements of our propositions are far longer than their proofs, which amount to trivial calculations in the double category of arenas. As in much of categorical work, the difficulty is in understanding what to propose; once that work is done, the proof flows smoothly from the definitions.

Let's see what composition of squares in the double category of arenas means for systems. Horizontal composition is familiar because it's what lets us compose behaviors:

So, we have a category of systems and behaviors in any doctrine, just as we defined in the deterministic doctrine.

On the other hand, vertical composition tells us something else interesting: if you get a chart $\binom{g_\flat}{g}$ by wiring together a chart $\binom{f_\flat}{f}$, then a behavior ϕ with chart $\binom{f_\flat}{f}$

induces a behavior with chart $\begin{pmatrix} g_\flat \\ g \end{pmatrix}$ on the wired together systems.

$$
\begin{pmatrix} T\mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} T\mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix}
$$

$$
\begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix} \Updownarrow \qquad \Updownarrow \begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix}
$$

$$
\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} \mathsf{In_S} \\ \mathsf{Out_S} \end{pmatrix}
$$

$$
\begin{pmatrix} j^\sharp \\ j \end{pmatrix} \Updownarrow \qquad \Updownarrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix}
$$

$$
\begin{pmatrix} I \\ O \end{pmatrix} \xRightarrow{\begin{pmatrix} g_\flat \\ g \end{pmatrix}} \begin{pmatrix} I' \\ O' \end{pmatrix}
$$

$$
= \begin{pmatrix} f^\sharp \\ f \end{pmatrix} \circ
\left[
\begin{array}{c}
\begin{pmatrix} T\mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} T\phi \\ \psi\phi \end{pmatrix}} \begin{pmatrix} T\mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\
\begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix} \Updownarrow \qquad \Updownarrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix} \circ \begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} \\
\begin{pmatrix} I \\ O \end{pmatrix} \xRightarrow{\begin{pmatrix} g_\flat \\ g \end{pmatrix}} \begin{pmatrix} I \\ O' \end{pmatrix}
\end{array}
\right]
$$

The interchange law of the double category of arenas tells us precisely that these two sorts of composition of behaviors — composition as maps and wiring — *commute*. That is, we can compose two behaviors and then wire them together, or we can wire each together and then compose them; the end result is the same.

Example 4.1.4. Continuing from **??**, suppose that we have a $\begin{pmatrix} b^- \\ b^+ \end{pmatrix}$-steady state s in a system S:

$$
\begin{pmatrix} 1 \\ 1 \end{pmatrix} \xRightarrow{\begin{pmatrix} s \\ s \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix}
$$

$$
\Vert\Vert\Vert \qquad \Updownarrow \begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} \tag{4.1}
$$

$$
\begin{pmatrix} 1 \\ 1 \end{pmatrix} \xRightarrow{\begin{pmatrix} b^+ \\ b^- \end{pmatrix}} \begin{pmatrix} B^- \\ B^+ \end{pmatrix}
$$

We can see that s is a $\begin{pmatrix} d^- \\ d^+ \end{pmatrix}$-steady state of the wired system by vertically composing the square in **??** with the square in **??**. This basic fact underlies our arguments in the upcoming **??**.

We'll return to this idea in **??**.

While our results are most smoothly proven in the double category of arenas, this double category does not capture the way we think of systems and their behaviors. To think of a behavior, we must first think of its chart; we solve a differential equation

in terms of its parameters, and to get a specific solution we must first choose specific parameters. Working in the double category of arenas means treating the chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$ and the underlying map ϕ of a behavior on equal footing, but we would instead like to say that ϕ is a behavior for the chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$.

We would also like to think of the wiring together of systems along a lens $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right)$ as an operation performed on systems, and then inquire into the relationship of this wiring operation with the (horizontal) composition of behaviors.

What we need is to separate the *interface* of a system from the system itself. Charts and lenses are best understood as ways of relating interfaces. It just so happens that systems and their behaviors can also be expressed as certain sorts of lenses and charts, which drastically facilitates our working with them. But there is some sense in which this is not essential; the main point is that for each interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ we have a notion of system with interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$, for each lens $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) : \left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix} \right)$ a way of wiring $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$-systems into $\left(\begin{smallmatrix} I' \\ O' \end{smallmatrix} \right)$ systems, and for each chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix} \right)$ a notion of behavior for this chart. It is very convenient that we can describe wiring and composition of behaviors in the same terms as charts and lenses, but we shouldn't think that they are the same thing.

In this chapter, we will define the appropriate abstract algebra of systems and their two sorts of composition keeping in mind the separation between interfaces and systems. We call this abstract algebra a *doubly indexed category*, since it is a sort of double categorical generalization of an indexed category. We'll see the definition of this notion in **??**.

Once we have organized our systems into doubly indexed categories, we can discuss what it means to change our doctrine. A change of doctrine will be a way of turning one sort of dynamical system into another. This could mean simply re-interpreting the underlying structure (for example, a deterministic system where all maps are differentiable is in particular a discrete deterministic system, just by forgetting the differentiability) or by restricting the use of certain maps (as in **??**). But it could also mean approximating one sort of system by another sort of system.

As an example, let's consider the Euler method for approximating a differential system. Suppose that

$$\begin{pmatrix} \mathrm{update}_S \\ \mathrm{expose}_S \end{pmatrix} : \begin{pmatrix} \mathbb{R}^n \\ \mathbb{R}^n \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^k \\ \mathbb{R}^m \end{pmatrix}$$

is a Euclidean differential system S. This represents the differential equation

$$\frac{ds}{dt} = \mathrm{update}_S(s, p).$$

That is, a trajectory is a map $s : \mathbb{R} \to \mathbb{R}^n$ satisfying this differential equation (for a choice of parameters $p : \mathbb{R} \to \mathbb{R}^k$). This means that the direction that the state s_0 is tending is given by $\mathrm{update}_S(s_0, p_0)$. We could then approximate the solution, given

such a starting point, by moving a small distance in this direction. We could get a whole sequence of states this way; moving in the direction our dynamics tells us we should go, and then checking where to go from there.

The result is a deterministic system $\mathcal{E}_\epsilon(\mathsf{S})$ whose dynamics is given by

$$\mathrm{update}_{\mathcal{E}_\epsilon(\mathsf{S})}(s, p) = s + \epsilon \cdot \mathrm{update}_{\mathsf{S}}(s, p).$$

Here, $\epsilon > 0$ is some small increment. We can take $\mathcal{E}_\epsilon(\mathsf{S})$ to expose the same variable that S does: $\mathrm{expose}_{\mathcal{E}_\epsilon(\mathsf{S})} = \mathrm{expose}_{\mathsf{S}}$.

The change of doctrine $\mathcal{E}_\epsilon$ is the formula for changing from the Euclidean differential doctrine to the deterministic doctrine on the cartesian category of Euclidean spaces. We might wonder: how does changing the doctrine by using the Euler method affect the wiring together of systems? How does it affect the behaviors of the systems?

We can answer the question about behaviors here. It is not true that every behavior of a Euclidean differential system is faithfully represented by its Euler method approximation. Consider, for example, the simply system

$$\mathrm{update}_{\mathsf{S}}(s) = s$$

having one state variable, and no parameters. The trajectories of this system are of the form $s(t) = Ce^t$ for some constant C. However, if we let $\epsilon = .1$ and consider the Euler approximation

$$\mathrm{update}_{\mathcal{E}_{.1}(\mathsf{S})}(s(0)) = s(0) + .1 \cdot s(0) = 1.01 \cdot C,$$

This is not the same thing as $s(.1) = Ce^{.1} \approx 1.105 \cdot C$ (though, as expected, they are rather close). So we see that general behaviors are *not* preserved!

However, suppose we have a steady state of the system. For example, taking $C = 0$ we get a steady state of the system $\mathrm{update}_{\mathsf{S}}(s) = s$ above. Then we have that

$$\mathrm{update}_{\mathcal{E}_{.1}(\mathsf{S})}(0) = 0 + .1 \cdot 0 = 0.$$

In other words, the steady state remains a steady state!

The goal of this chapter will be to introduce the formalism which enables us to inquire into and prove various compositionality results concerning changes of doctrine. In the above situation, we will see that the Euler method $\mathcal{E}_\epsilon$ gives a change of doctrine on a *restriction* of the Euclidean differential doctrine to affine maps. As a result, it will preserve any behavior whose underlying map is affine (of the form $\phi(v) = Av + b$ for a matrix A and vector b), which includes all steady states (since constant maps are affine) but almost no trajectories in general.

We will introduce the notion of a *doubly indexed functor* to organize the compositionality results concerning change of doctrine. We will also be using these doubly indexed functors in the next chapter to organize the compositionality of behaviors in general.

We will define the notion of change of doctrine formally (**??**) and show that every change of doctrine gives rise to a doubly indexed functor between the doubly indexed

categories of systems in the respective doctrines. In particular, we will show that there is a *2-functor* Jaz: I am starting to think that I should put the 2-functoriality of this into an appendix so that I don't have to introduce 2-categories and 2-functors here.

$$\mathbf{Sys} : \mathbf{Doctrine} \to \mathbf{DblIx}$$

sending a doctrine to the doubly indexed category of systems in it.

4.2   Composing behaviors in general

Before we get to this abstract defintion, we will take our time exploring the sorts of compositionality results one may prove quickly by working in the double category of arenas.

Recall the categories $\mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ of systems with the interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ from **??**. One thing that vertical composition in the double category of arenas shows us is that wiring together systems is functorial with respect to simulations — that is, behaviors that don't change the interface.

We repeat the definition of $\mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ for an arbitrary doctrine.

Definition 4.2.1. Let $\mathbb{D} = (\mathscr{A}, T)$ be a dynamical system doctrine. For a $\mathscr{A}$-arena $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$, the category $\mathbf{Sys}\left(\begin{smallmatrix} A \\ C \end{smallmatrix} \right)$ of $\mathbb{D}$-systems with interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$ is defined by:

- Its objects are $\mathscr{A}$-lenses $\left(\begin{smallmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{smallmatrix} \right) : \left(\begin{smallmatrix} T\mathsf{State_S} \\ \mathsf{State_S} \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)$, which we can call *systems* in this general setting.
- Its maps are *simulations*, the behaviors which have identity chart. That is, the maps are the squares

$$
\begin{array}{ccc}
\left(\begin{smallmatrix} T\mathsf{State_T} \\ \mathsf{State_T} \end{smallmatrix} \right) & \xRightarrow{\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix} \right)} & \left(\begin{smallmatrix} T\mathsf{State_S} \\ \mathsf{State_S} \end{smallmatrix} \right) \\
{\scriptsize\left(\begin{smallmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{smallmatrix} \right)}\Big\Updownarrow & & \Big\Updownarrow{\scriptsize\left(\begin{smallmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{smallmatrix} \right)} \\
\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right) & =\!=\!=\!= & \left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right)
\end{array}
$$

- Composition is given by horizontal composition in the double category $\mathbf{Arena}_{\mathscr{A}}$ of $\mathscr{A}$-arenas.

Now, thanks to the double category of arenas, we can show that every lens $\left(\begin{smallmatrix} f^{\sharp} \\ f \end{smallmatrix} \right)$:

$\begin{pmatrix} I \\ O \end{pmatrix} \leftrightarrows \begin{pmatrix} I' \\ O' \end{pmatrix}$ gives a functor

$$\mathbf{Sys}\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix}.$$

We can see this functor as the operation of wiring together our $\begin{pmatrix} I \\ O \end{pmatrix}$-systems along the lens $\begin{pmatrix} f^\sharp \\ f \end{pmatrix}$ to get $\begin{pmatrix} I' \\ O' \end{pmatrix}$-systems. The functoriality of this operation say that wiring preserves simulations — if systems S_i simulate T_i by ϕ_i, then the wired together systems S simulate T by $\phi = \prod_i \phi_i$.

Proposition 4.2.2. For a lens $\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \begin{pmatrix} I \\ O \end{pmatrix} \leftrightarrows \begin{pmatrix} I' \\ O' \end{pmatrix}$, we get a functor

$$\mathbf{Sys}\begin{pmatrix} f^\sharp \\ f \end{pmatrix} : \mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix}$$

Given by composing with $\begin{pmatrix} f^\sharp \\ f \end{pmatrix}$:

- For a system $S = \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} T\text{State}_S \\ \text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} I \\ O \end{pmatrix}$,

$$\mathbf{Sys}\begin{pmatrix} f^\sharp \\ f \end{pmatrix}(S) = \begin{pmatrix} f^\sharp \\ f \end{pmatrix} \circ \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix}.$$

- For a behavior, $\mathbf{Sys}\begin{pmatrix} f^\sharp \\ f \end{pmatrix}$ acts in the following way:

Proof. The functoriality of this construction can be seen immediately from the inter-

change law of the double category:

$$\frac{\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix}\right)\middle|\left(\begin{smallmatrix} T\psi \\ \psi \end{smallmatrix}\right)}{\left(\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right)} = \frac{\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix}\right)\middle|\left(\begin{smallmatrix} T\psi \\ \psi \end{smallmatrix}\right)}{\left(\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right)\middle|\left(\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right)} \qquad \text{by the horizontal identity law,}$$

$$= \frac{\left(\begin{smallmatrix} T\phi \\ \phi \end{smallmatrix}\right)}{\left(\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right)}\middle|\frac{\left(\begin{smallmatrix} T\psi \\ \psi \end{smallmatrix}\right)}{\left(\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right)} \qquad \text{by the interchange law.}$$

Identities are clearly preserved, since the underlying morphism $\phi : \mathsf{State}_\mathsf{T} \to \mathsf{State}_\mathsf{S}$ is not changed. □

The notion of profunctor gives us a nice way to understand the relationship between a behavior $\phi : \mathsf{T} \to \mathsf{S}$ and its chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right) : \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \rightrightarrows \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$. When we are using behaviors, we usually have the chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)$ in mind first, and then look for behaviors with this chart. For example, when finding trajectories, we first set the parameters for our system and then solve it. We can use profunctors to formalize this relationship.

Proposition 4.2.3. Given a chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right) : \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \Rightarrow \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$, we get a profunctor

$$\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix} \nrightarrow \mathbf{Sys}\begin{pmatrix} I' \\ O' \end{pmatrix}$$

Defined by:

$$\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}(\mathsf{T}, \mathsf{S}) = \left\{ \phi : \mathsf{State}_\mathsf{T} \to \mathsf{State}_\mathsf{S} \;\middle|\; \phi \text{ is a behavior with chart } \begin{pmatrix} f_\flat \\ f \end{pmatrix} \right\}$$

$$= \left\{ \begin{array}{c} \begin{pmatrix} \mathsf{State}_\mathsf{T} \\ \mathsf{State}_\mathsf{T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} \phi\circ\pi_2 \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{pmatrix} \\[6pt] {\scriptstyle\begin{pmatrix} \mathsf{update}_\mathsf{T} \\ \mathsf{expose}_\mathsf{T} \end{pmatrix}}\Big\updownarrow\Big\uparrow \qquad\qquad \Big\updownarrow\Big\uparrow{\scriptstyle\begin{pmatrix} \mathsf{update}_\mathsf{S} \\ \mathsf{expose}_\mathsf{S} \end{pmatrix}} \\[6pt] \begin{pmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{pmatrix} \xrightarrow[\begin{pmatrix} f_\flat \\ f \end{pmatrix}]{} \begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix} \end{array} \right\}$$

The action of the profunctor $\mathbf{Sys}\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)$ on simulations in the categories $\mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ and $\mathbf{Sys}\left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$ is given by composition on the left and right. That is, for simulations $\phi : \mathsf{T}' \to \mathsf{T}$ and $\psi : \mathsf{S} \to \mathsf{S}'$ and $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)$-behavior $\beta \in \mathbf{Sys}\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)(\mathsf{T}, \mathsf{S})$, we define

$$\phi \cdot \beta \cdot \psi := \phi \mid \beta \mid \psi. \tag{4.2}$$

Exercise 4.2.4. Prove **??**. That is, show that the action defined in **??** is functorial, giving a functor

$$\mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix}^{\mathrm{op}} \times \mathbf{Sys}\begin{pmatrix} I' \\ O' \end{pmatrix} \to \mathbf{Set}.$$

(Hint: use the double categorical notation. It will be much more concise.) ◊

With a little work in the double category of arenas, we can give a very useful example of a square in the double category of profunctors. Consider this square in the double

category of arenas:

$$
\alpha = \begin{pmatrix} j_\flat \\ j \end{pmatrix} \Bigg\downarrow\Bigg\uparrow
\quad
\begin{array}{c}
\begin{pmatrix} I_1 \\ O_1 \end{pmatrix} \xRightarrow{\;\begin{pmatrix} f^\sharp \\ f \end{pmatrix}\;} \begin{pmatrix} I_2 \\ O_2 \end{pmatrix} \\[2em]
\begin{pmatrix} I_3 \\ O_3 \end{pmatrix} \xRightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{\;} \begin{pmatrix} I_4 \\ O_4 \end{pmatrix}
\end{array}
\quad \Bigg\downarrow\Bigg\uparrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix}
$$

As we saw in **??**, we get functors $\mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix} : \mathbf{Sys}\begin{pmatrix} I_1 \\ O_1 \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} I_3 \\ O_3 \end{pmatrix}$ and $\mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix} :$ $\mathbf{Sys}\begin{pmatrix} I_2 \\ O_2 \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} I_4 \\ O_4 \end{pmatrix}$ given by composing with these lenses. We also saw in **??** that we get profunctors $\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \mathbf{Sys}\begin{pmatrix} I_1 \\ O_1 \end{pmatrix} \nrightarrow \mathbf{Sys}\begin{pmatrix} I_2 \\ O_2 \end{pmatrix}$ and $\mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \mathbf{Sys}\begin{pmatrix} I_3 \\ O_3 \end{pmatrix} \nrightarrow \mathbf{Sys}\begin{pmatrix} I_4 \\ O_4 \end{pmatrix}$ from these charts. We can also get a square of profunctors from the square α in the double category of arenas:

$$
\begin{array}{ccc}
\mathbf{Sys}\begin{pmatrix} I_1 \\ O_1 \end{pmatrix} & \xrightarrow{\;\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}\;} & \mathbf{Sys}\begin{pmatrix} I_2 \\ O_2 \end{pmatrix} \\[1em]
\mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix} \Bigg\downarrow & \mathbf{Sys}(\alpha) & \Bigg\downarrow \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix} \\[1em]
\mathbf{Sys}\begin{pmatrix} I_3 \\ O_3 \end{pmatrix} & \xrightarrow[\;\mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix}\;]{} & \mathbf{Sys}\begin{pmatrix} I_4 \\ O_4 \end{pmatrix}
\end{array}
$$

That is, a natural transformation of the following signature:

$$
\mathbf{Sys}(\alpha) : \mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix}\left(\mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix}, \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix}\right).
$$

To define the natural transformation $\mathbf{Sys}(\alpha)$, we need to say what it does to an element ϕ of $\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}(\mathsf{T}, \mathsf{S})$. Recall that the elements of this profunctor are behaviors

with chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$, so really ϕ is a square

$$\phi = \begin{pmatrix} \text{update}_T \\ \text{expose}_T \end{pmatrix} \quad \begin{array}{ccc} \begin{pmatrix} T\text{State}_T \\ \text{State}_T \end{pmatrix} & \xrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} & \begin{pmatrix} T\text{State}_S \\ \text{State}_S \end{pmatrix} \\ & & \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} \\ \begin{pmatrix} I_1 \\ O_1 \end{pmatrix} & \xrightarrow[\begin{pmatrix} f_\flat \\ f \end{pmatrix}]{} & \begin{pmatrix} I_2 \\ O_2 \end{pmatrix} \end{array}$$

in the double category of arenas. Therefore, we can define $\mathbf{Sys}(\alpha)(\phi)$ to be the vertical composite:

$$\begin{array}{ccc} \begin{pmatrix} T\text{State}_T \\ \text{State}_T \end{pmatrix} & \xrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} & \begin{pmatrix} T\text{State}_S \\ \text{State}_S \end{pmatrix} \\ \begin{pmatrix} \text{update}_T \\ \text{expose}_T \end{pmatrix} & & \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} \\ \begin{pmatrix} I_1 \\ O_1 \end{pmatrix} & \xrightarrow[\begin{pmatrix} f_\flat \\ f \end{pmatrix}]{} & \begin{pmatrix} I_2 \\ O_2 \end{pmatrix} \\ \begin{pmatrix} j^\# \\ j \end{pmatrix} & & \begin{pmatrix} k^\# \\ k \end{pmatrix} \\ \begin{pmatrix} I_3 \\ O_3 \end{pmatrix} & \xrightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} & \begin{pmatrix} I_4 \\ O_4 \end{pmatrix} \end{array}$$

Or, a little more concisely in double category notation:

$$\mathbf{Sys}(\alpha)(\phi) = \frac{\phi}{\alpha}.$$

We record this observation in a proposition.

Proposition 4.2.5. Given a square

$$\alpha = \begin{pmatrix} j_\flat \\ j \end{pmatrix} \Bigg\Downarrow\Bigg\Uparrow \qquad \begin{pmatrix} I_1 \\ O_1 \end{pmatrix} \xRightarrow{\begin{pmatrix} f^\sharp \\ f \end{pmatrix}} \begin{pmatrix} I_2 \\ O_2 \end{pmatrix} \qquad \Bigg\Downarrow\Bigg\Uparrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix}$$

$$\begin{pmatrix} I_3 \\ O_3 \end{pmatrix} \xRightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} \begin{pmatrix} I_4 \\ O_4 \end{pmatrix}$$

in the double category of arenas, we get a square

$$\mathbf{Sys}\begin{pmatrix} I_1 \\ O_1 \end{pmatrix} \xrightarrow{\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \mathbf{Sys}\begin{pmatrix} I_2 \\ O_2 \end{pmatrix}$$

$$\mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix} \Bigg\downarrow \qquad \mathbf{Sys}(\alpha) \qquad \Bigg\downarrow \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix}$$

$$\mathbf{Sys}\begin{pmatrix} I_3 \\ O_3 \end{pmatrix} \xrightarrow[\mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} \mathbf{Sys}\begin{pmatrix} I_4 \\ O_4 \end{pmatrix}$$

in the double category of categories, functors, and profunctors given by

$$\mathbf{Sys}(\alpha)(\phi) = \frac{\phi}{\alpha}.$$

The naturality of this transformation follows from the double category laws. We leave the particulars as an exercise.

Exercise 4.2.6.   Prove that the family of functions

$$\mathbf{Sys}(\alpha) : \mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix}\left(\mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix}, \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix}\right)$$

defined in **??** is a natural transformation. (Hint: use the double category notation, it will be much more concise.)   ◊

4.3 Arranging categories along two kinds of composition: Doubly indexed categories

While we described a category of systems and behaviors in **??**, we haven't been thinking of systems in quite this way. We have been organizing our systems a bit more particularly than just throwing them into one large category. We've made the following observations:

- Each system has an interface, and many different systems can have the same interface. From this observation, we defined the categories $\mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ of systems with the interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ in **??**.
- Every wiring diagram, or more generally lens, gives us an operation that changes the interface of a system by wiring things together. We formalized this observation into a functor $\mathbf{Sys}\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right) : \mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \to \mathbf{Sys}\left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$ in **??**.
- To describe the behavior of a system, first we have to chart out how it will look on its interface. We formalized this observation by giving a profunctor $\mathbf{Sys}\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right) : \mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \nrightarrow \mathbf{Sys}\left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$ for each chart in **??**.
- If we wire together a chart for one interface into a chart for the wired interface, then every behavior for that chart gives rise to a behavior for the wired together chart. We formalized this observation as a morphism of profunctors

$$\mathbf{Sys}(\alpha) : \mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix} \to \mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix}\left(\mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix}, \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix}\right)$$

in **??**.

Now comes the time to organize all these observations. In this section, we will see that collectively, these observations are telling us that there is an *doubly indexed category* of dynamical systems. We will also see that matrices of sets give rise to a doubly indexed category which we will call the doubly indexed category of vectors of sets.

Definition 4.3.1. A *doubly indexed category* $\mathcal{A} : \mathcal{D} \to \mathbf{Cat}$ consists of the following:[a]
- A double category $\mathcal{D}$ called the *indexing base*.
- For every object $D \in \mathcal{D}$, we have a category $\mathcal{A}(D)$.
- For every vertical arrow $j : D \to D'$, we have a functor $\mathcal{A}(j) : \mathcal{A}(D) \to \mathcal{A}(D')$.
- For every horizontal arrow $f : D \to D'$, we have a profunctor $\mathcal{A}(f) : \mathcal{A}(D) \nrightarrow \mathcal{A}(D')$.
- For every square

$$
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
{\scriptstyle j}\downarrow & \alpha & \downarrow{\scriptstyle k} \\
C & \xrightarrow[\ g\ ]{} & D
\end{array}
$$

in $\mathcal{D}$, a square

$$
\begin{array}{ccc}
\mathcal{A}(A) & \xrightarrow{\mathcal{A}(f)} & \mathcal{A}(B) \\
{\scriptstyle \mathcal{A}(j)}\Big\downarrow & \mathcal{A}(\alpha) & \Big\downarrow{\scriptstyle \mathcal{A}(k)} \\
\mathcal{A}(C) & \xrightarrow[\mathcal{A}(g)]{} & \mathcal{A}(D)
\end{array}
$$

in **Cat**.

- For any two horizontal maps $f : A \to B$ and $g : B \to C$ in $\mathcal{D}$, we have a square $\mu_{f,g} : \mathcal{A}(f) \odot \mathcal{A}(g) \to \mathcal{A}(f \mid g)$ called the *compositor*:

$$
\begin{array}{ccc}
\mathcal{A}(A) \overset{\mathcal{A}(f)}{\to} \mathcal{A}(B) \overset{\mathcal{A}(f)}{\to} \mathcal{A}(E) \\
\Big\| \qquad \mu_{f,g} \qquad \Big\| \\
\mathcal{A}(A) \xrightarrow[\mathcal{A}(f \mid g)]{} \mathcal{A}(C)
\end{array}
\tag{4.3}
$$

This data is required to satisfy the following laws:

- (Vertical Functoriality) For vertical maps $j : D \to D'$ and $k : D' \to D''$, we have that

$$
\mathcal{A}\left(\frac{j}{k}\right) = \frac{\mathcal{A}(j)}{\mathcal{A}(k)}
$$

and that $\mathcal{A}(\mathrm{id}_D) = \mathrm{id}_{\mathcal{A}(D)}$.[b]

- (Horizontal Lax Functoriality) For horizontal maps $f : D_1 \to D_2$, $g : D_2 \to D_3$ and $h : D_3 \to D_4$, the compositors μ satisfy the following associativity and unitality conditions:
  - (Associativity)

$$
\frac{\mu_{f,g}\mid\mathcal{A}(h)}{\mu_{(f\mid g),h}} \overset{\cdot}{=\!=} \frac{\mathcal{A}(f)\mid\mu_{g,h}}{\mu_{f,(g\mid h)}}.
$$

  - (Unitality) The profunctor $\mathcal{A}(\mathrm{id}_{D_1}) : \mathcal{A}(D_1) \nrightarrow \mathcal{A}(D_1)$ is the identity profunctor, $\mathcal{A}(\mathrm{id}_{D_1}) = \mathcal{A}(D_1)$. Furthermore, $\mu_{\mathrm{id}_{D_1},f}$ and $\mu_{f,\mathrm{id}_{D_2}}$ are equal to the isomorphisms of **??** given by the naturality of $\mathcal{A}(f)$ on the left and right respectively. We may sumarize this may saying that

$$
\mu_{\mathrm{id},f} = \mathrm{id}_{\mathcal{A}(f)} = \mu_{f,\mathrm{id}}.
$$

- (Naturality of Compositors) For any horizontally composable squares

$$
\begin{array}{ccccccc}
A & \xrightarrow{f_1} & B & & B & \xrightarrow{f_2} & E \\
{\scriptstyle j}\Big\downarrow & \alpha & \Big\downarrow{\scriptstyle k} & \text{and} & {\scriptstyle k}\Big\downarrow & \alpha & \Big\downarrow{\scriptstyle l} \\
C & \xrightarrow[g_1]{} & D & & D & \xrightarrow[g_2]{} & F
\end{array}
$$

$$\frac{\mathcal{A}(\alpha) \mid \mathcal{A}(\beta)}{\mu_{g_1, g_2}} = \frac{\mu_{f_1, f_2}}{\mathcal{A}(\alpha \mid \beta)}.$$

---

[a]This is what an expert would call a *unital (or normal) lax double functor*, but we won't need this concept in any other setting.

[b]Here, we are hiding some coherence issues. While our doubly indexed category of deterministic systems will satisfy this functoriality condition on the nose, we will soon see a doubly indexed category of matrices of sets for which this law only holds up to a coherence isomorphism. Again, the issue invovles shuffling parentheses around, and we will sweep it under the rug.

That's another big definition! It seems like it will be a slog to actually ever prove that something is a doubly indexed category. Luckily, in our cases, these proofs will go quite smoothly. This is because each of the three laws of a doubly indexed category has a sort of sister law from the definition of a double category which will help us prove it.

- The Vertical Functoriality law will often involve the vertical associativity and unitality of squares in the indexing base.
- The Horizontal Lax Functoriality law will often involve the horizontal associativity and unitality of squares in the indexing base.
- The Naturality of Compositors law will often involve the interchange law in the indexing base.

We'll see how these sisterhoods play out in practice as we define the doubly indexed categories of deterministic systems and vectors of sets.

The doubly indexed category of systems   Let's show that systems in a doctrine $\mathbb{D}$ do indeed form a doubly indexed category

$$\mathbf{Sys}_{\mathbb{D}} : \mathbf{Arena}_{\mathbb{D}} \to \mathbf{Cat}.$$

Definition 4.3.2. The doubly indexed category $\mathbf{Sys}_{\mathbb{D}} : \mathbf{Arena}_{\mathbb{D}} \to \mathbf{Cat}$ of systems in the doctrine $\mathbb{D} = (\mathcal{A}, T)$ is defined as follows:

- Our indexing base is the double category $\mathbf{Arena}_{\mathbb{D}}$ of arenas, since we will arrange our systems according to their interface.
- To every arena $\binom{I}{O}$, we associate the category $\mathbf{Sys}\binom{I}{O}$ of systems with interface $\binom{I}{O}$ and behaviors whose chart is the identity chart on $\binom{I}{O}$ (**??**).
- To every lens $\binom{w^{\sharp}}{w} : \binom{I}{O} \leftrightarrows \binom{I'}{O'}$, we associate the functor $\mathbf{Sys}\binom{w^{\sharp}}{w} : \mathbf{Sys}\binom{I}{O} \to \mathbf{Sys}\binom{I}{O}$ given by wiring according to $\binom{w^{\sharp}}{w}$:

$$\mathbf{Sys}\begin{pmatrix} w^{\sharp} \\ w \end{pmatrix}(\mathsf{S}) = \frac{\mathsf{S}}{\binom{w^{\sharp}}{w}}.$$

This is defined in **??**.

- To every chart $\binom{f_{\flat}}{f} : \binom{I}{O} \Rightarrow \binom{I'}{O'}$, we associate the profunctor $\mathbf{Sys}\binom{f_{\flat}}{f} :$

$\mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix} \nrightarrow \mathbf{Sys}\begin{pmatrix} I' \\ O' \end{pmatrix}$ which sends the $\begin{pmatrix} I \\ O \end{pmatrix}$-system T and the $\begin{pmatrix} I' \\ O' \end{pmatrix}$-system S to the set of behaviors $\mathsf{T} \to \mathsf{S}$ with chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$:

$$\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}(\mathsf{T}, \mathsf{S}) = \left\{ \phi : \mathsf{State}_\mathsf{T} \to \mathsf{State}_\mathsf{S} \;\middle|\; \phi \text{ is a behavior with chart } \begin{pmatrix} f_\flat \\ f \end{pmatrix} \right\}$$

$$= \left\{ \begin{array}{c} \begin{pmatrix} T\mathsf{State}_\mathsf{T} \\ \mathsf{State}_\mathsf{T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} T\mathsf{State}_\mathsf{S} \\ \mathsf{State}_\mathsf{S} \end{pmatrix} \\ \begin{pmatrix} \text{update}_\mathsf{T} \\ \text{expose}_\mathsf{T} \end{pmatrix} \Big\Updownarrow\Big\uparrow \qquad\qquad \Big\Updownarrow\Big\uparrow \begin{pmatrix} \text{update}_\mathsf{S} \\ \text{expose}_\mathsf{S} \end{pmatrix} \\ \begin{pmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{pmatrix} \xRightarrow[\begin{pmatrix} f^\sharp \\ f \end{pmatrix}]{} \begin{pmatrix} \mathsf{In}_\mathsf{S} \\ \mathsf{Out}_\mathsf{S} \end{pmatrix} \end{array} \right\}$$

We saw this profunctor in **??**.

- To every square α, we assign the morphism of profunctors given by composing vertically with α in **Arena**:

$$\mathbf{Sys}(\alpha)(\phi) = \frac{\phi}{\alpha}.$$

We saw in **??** that this was a natural transformation.

- The compositor is given by horizontal composition in the double category of arenas:

$$\mu_{\begin{pmatrix} f_\flat \\ f \end{pmatrix}, \begin{pmatrix} g_\flat \\ g \end{pmatrix}} : \mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix} \odot \mathbf{Sys}\begin{pmatrix} g_\flat \\ g \end{pmatrix} \to \mathbf{Sys}\left(\begin{pmatrix} f_\flat \\ f \end{pmatrix} \fatsemi \begin{pmatrix} g_\flat \\ g \end{pmatrix} \right)$$

$$(\phi, \psi) \mapsto \phi \mid \psi$$

Let's check now that this does indeed satisfy the laws of a doubly indexed category. The task may appear to loom over us; there are quite a few laws, and there is a lot of data involved. But nicely, they all follow quickly from a bit of fiddling in the double category of arenas.

- (Vertical Functoriality) We show that $\mathbf{Sys}\left(\begin{pmatrix} k^\sharp \\ k \end{pmatrix} \circ \begin{pmatrix} j^\sharp \\ j \end{pmatrix} \right) = \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix} \circ \mathbf{Sys}\begin{pmatrix} j^\sharp \\ j \end{pmatrix}$ by vertical associativity:

$$\mathbf{Sys}\left(\begin{pmatrix} k^\sharp \\ k \end{pmatrix} \circ \begin{pmatrix} j^\sharp \\ j \end{pmatrix} \right)(\phi) = \frac{\phi}{\left(\dfrac{\begin{pmatrix} j^\sharp \\ j \end{pmatrix}}{\begin{pmatrix} k^\sharp \\ k \end{pmatrix}} \right)} = \frac{\left(\dfrac{\phi}{\begin{pmatrix} j^\sharp \\ j \end{pmatrix}} \right)}{\begin{pmatrix} k^\sharp \\ k \end{pmatrix}}$$

$$= \mathbf{Sys} \begin{pmatrix} k^\sharp \\ k \end{pmatrix} \circ \mathbf{Sys} \begin{pmatrix} j^\sharp \\ j \end{pmatrix} (\phi).$$

- (Horizontal Lax Functoriality) This law follows from horizontal associativity in **Arena**.

$$\mu(\mu(\phi, \psi), \xi) = (\phi \mid \psi) \mid \xi = \phi \mid (\psi \mid \xi) = \mu(\phi, \mu(\psi, \xi)). \tag{4.4}$$

- (Naturality of Compositor) This law follows from interchange in **Arena**.

$$\left(\frac{\mathbf{Sys}(\alpha) \mid \mathbf{Sys}(\beta)}{\mu} \right) (\phi, \psi) = \frac{\phi \left| \psi \right.}{\alpha \left| \beta \right.} = \frac{\phi \mid \psi}{\alpha \mid \beta}$$

$$= \left(\frac{\mu}{\mathbf{Sys}(\alpha \mid \beta)} \right) (\phi, \psi).$$

The doubly indexed category of vectors of sets   In addition to our doubly indexed category of systems, we have a doubly indexed category of "vectors of sets".

Classically, an $m \times n$ matrix M can act on a vector v of length n by multiplication to get another vector Mv of length m. We can generalize this to matrices of sets if we define a vector of sets of length A to be a dependent set $V : A \to \mathbf{Set}$.

Definition 4.3.3. For a set A, we define the category of *vectors of sets of length A* to be

$$\mathbf{Vec}(A) := \mathbf{Set}^A$$

the category of sets depending on A.

Given a $(B \times A)$-matrix M, we can treat a A-vector V as a $A \times 1$ matrix and form the $B \times 1$ matrix MV. This gives us a functor

$$\mathbf{Vec}(M) : \mathbf{Vec}(A) \to \mathbf{Vec}(B)$$

$$V \mapsto (MV)_b = \sum_{a \in A} M_{ba} \times V_a$$

$$f : V \to W \mapsto ((a, m, v) \mapsto (a, m, f(v)))$$

which we refer to as the linear functor given by M.

Definition 4.3.4. The doubly indexed category $\mathbf{Vec} : \mathbf{Matrix} \to \mathbf{Cat}$ of vectors of sets is defined by:

- Its indexing base is the double category of matrices of sets.
- To every set A, we assign the category $\mathbf{Vec}(A) = \mathbf{Set}^A$ of vectors of length A.
- To every $(B \times A)$-matrix $M : A \to B$, we assign the linear functor $\mathbf{Vec}(M) : \mathbf{Vec}(A) \to \mathbf{Vec}(B)$ given by M (**??**).
- To every function $f : A \to B$, we associate the profunctor $\mathbf{Vec}(f) : \mathbf{Vec}(A) \nrightarrow \mathbf{Vec}(B)$

defined by
$$\mathbf{Vec}(f)(V, W) = \{F : (a \in A) \to V_a \to W_{f(a)}\}.$$

That is, $F \in \mathbf{Vec}(f)(V, W)$ is a family of functions $F(a, -) : V_a \to W_{f(a)}$ indexed by $a \in A$. This is natural by index-wise composition.

- To every square

$$
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
{\scriptstyle M}\downarrow & \alpha & \downarrow{\scriptstyle N} \\
C & \xrightarrow[\ g\ ]{} & D
\end{array}
$$

that is, family of functions $\alpha_{ca} : M_{ca} \to N_{g(c)f(a)}$, we associate the square

$$
\begin{array}{ccc}
\mathbf{Vec}(A) & \xrightarrow{\ \mathbf{Vec}(f)\ } & \mathbf{Vec}(B) \\
{\scriptstyle \mathbf{Vec}(M)}\downarrow & \mathbf{Vec}(\alpha) & \downarrow{\scriptstyle \mathbf{Vec}(N)} \\
\mathbf{Vec}(C) & \xrightarrow[\ \mathbf{Vec}(g)\ ]{} & \mathbf{Vec}(D)
\end{array}
$$

defined by sending a family of functions $F : (a \in A) \to V_a \to W_{f(a)}$ in $\mathbf{Vec}(f)(V, W)$ to the family

$$\mathbf{Vec}(\alpha)(F) : (c \in C) \to MV_c \to MW_{g(c)}$$
$$\mathbf{Vec}(\alpha)(F)(c, (a, m, v)) = (f(a), \alpha(m), F(a, v))$$

That is, $\mathbf{Vec}(\alpha)(F)(c, -)$ takes an element $(a, m, v) \in MV_c = \sum_{a \in A} M_{ca} \times V_a$ and gives the elements $(f(a), \alpha(m), F(a, v))$ of $MW_{g(c)} = \sum_{b \in B} N_{g(c)b} \times W_b$.

- The compositor is given by componentwise composition: If $f : A \to B$ and $g : B \to C$ and $F \in \mathbf{Vec}(f)(V, W)$ and $G \in \mathbf{Vec}(g)(W, U)$, then

$$\mu_{f,g}(F, G) : (a \in A) \to V_a \to U_{gf(a)}$$
$$\mu_{f,g}(F, G)(a, v) := G(f(a), F(a, v)).$$

It might seem like it will turn out to be a big hassle to show that this definition satisfies all the laws of a doubly indexed category. Like with the doubly indexed category of arenas, we will find that all the laws follow for matrices by fiddling around in the double category of matrices.

Let's first rephrase the above definition in terms of the category of matrices. We note that a vector of sets $V \in \mathbf{Vec}(A)$ is equivalently a matrix $V : 1 \to A$. Then the linear functor $\mathbf{Vec}(M) : \mathbf{Vec}(A) \to \mathbf{Vec}(B)$ is given by matrix multiplication, or in double category notation:

$$\mathbf{Vec}(M)(V) = \frac{V}{M}.$$

This means that the Vertical Functoriality law follows by vertical associativity in the double category of matrices, which is to say associativity of matrix multiplication.

Similarly, we can interpret the profunctor $\mathbf{Vec}(f)$ for $f : A \to B$ in terms of the double category $\mathbf{Matrix}$. An element $F \in \mathbf{Vec}(f)(V, W)$ is equivalently a square of the following form in $\mathbf{Matrix}$:

$$
\begin{array}{ccc}
1 & = & 1 \\
{\scriptstyle V}\downarrow & F & \downarrow{\scriptstyle W} \\
A & \xrightarrow{\ f\ } & B
\end{array}
$$

Therefore, we can describe $\mathbf{Vec}(f)(V, W)$ as the following set:

$$
\mathbf{Vec}(f)(V, W) = \left\{ F \,\middle|\,
\begin{array}{ccc}
1 & = & 1 \\
{\scriptstyle V}\downarrow & F & \downarrow{\scriptstyle W} \\
A & \xrightarrow{\ f\ } & B
\end{array}
\right\}
$$

Then the Horizontal Lax Functoriality laws follow from associativity and unitality of horizontal composition of squares in $\mathbf{Matrix}$!

Finally, we need to interpret the rather fiddly transformation $\mathbf{Vec}(\alpha)$ in terms of the double category of matrices. Its a matter of unfolding the definitions to see that $\mathbf{Vec}(\alpha)(F) = \frac{F}{\alpha}$ in $\mathbf{Matrix}$, and therefore that the Naturality of Compositors law follows by the interchange law.

If this argument seemed wholly too similar to the one we gave for the doubly indexed category of systems, your suspicions are not misplaced. These are both are instances of a very general *vertical slice construction*, which we turn our attension to now.

4.4 Vertical Slice Construction

In the previous section, we constructed the doubly indexed categories $\mathbf{Sys}_{\mathbb{D}}$ of systems in a doctrine $\mathbb{D}$ and $\mathbf{Vec}$ of vectors of sets "by hand". However, both constructions felt very familiar. In this section, we will show that they are both instances of a general construction: the *vertical slice construction*.

The main reason for recasting the above constructions in more general terms is that it will facilitate our main theorem of this chapter: change of doctrine.

The vertical slice construction will take a *double functor $F : \mathcal{D}_0 \to \mathcal{D}_1$* and produce a doubly indexed category $\sigma F : \mathcal{D}_1 \to \mathbf{Cat}$ indexed by its codomain. So, in order to describe the vertical slice construction, we will need the notion of double functor. We will need the notion of double functor for much of the coming theory as well

4.4.1 Double Functors

A double functor is the correct sort of functor between double categories. Just as a double category has a bit more than twice the information involved in a category, a double functor has a bit more than twice the information involved in a functor.

Definition 4.4.2. Let $\mathcal{D}_0$ and $\mathcal{D}_1$ be double categories. A *double functor* $F : \mathcal{D}_0 \to \mathcal{D}_1$ consists of:

- An object assignment $F : \mathrm{Ob}\,\mathcal{D}_0 \to \mathrm{Ob}\,\mathcal{D}_1$ which assigns an object FD in $\mathcal{D}_1$ to each object D in $\mathcal{D}_0$.
- A vertical functor $F : v\mathcal{D}_0 \to v\mathcal{D}_1$ on the vertical categories, which acts the same as the object assignment on objects.
- A horizontal functor $F : h\mathcal{D}_0 \to h\mathcal{D}_1$ on the horizontal categories, which acts the same as the object assignment on objects.
- For every square

$$
\begin{array}{ccc}
A & \xrightarrow{\ f\ } & B \\
{\scriptstyle j}\big\downarrow & \alpha & \big\downarrow{\scriptstyle k} \\
C & \xrightarrow{\ g\ } & D
\end{array}
$$

  in $\mathcal{D}_0$, a square

$$
\begin{array}{ccc}
FA & \xrightarrow{\ Ff\ } & FB \\
{\scriptstyle Fj}\big\downarrow & F\alpha & \big\downarrow{\scriptstyle Fk} \\
FC & \xrightarrow{\ Fg\ } & FD
\end{array}
$$

  such that the following laws hold:
    - F commutes with horizontal compostition: $F(\alpha \mid \beta) = F\alpha \mid F\beta$.
    - F commutes with vertical comopsition: $F\left(\frac{\alpha}{\beta}\right) = \frac{F\alpha}{F\beta}$.
    - F sends horizontal identities to horizontal identities, and vertical identities to vertical identities.

Remark 4.4.3. There is, in fact, a double category of double functors $F : \mathcal{D}_0 \to \mathcal{D}_1$, but we won't need to worry about this until we consider the functoriality of the vertical slice construction in **??**.

We will, in time, see many interesting examples of double functors. However, we will begin with the two simple examples we need to construct the doubly indexed categories **Sys** and **Vec**.

Example 4.4.4. Let $\mathbb{D} = (\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, T)$ be a doctrine. We recall that the section $T : \mathcal{C} \to \int^{C:\mathcal{C}} \mathcal{A}(C)$ is a functor to the Grothendieck construction of $\mathcal{A}$. We may promote this into a double functor into the double category of arenas **Arena**$_\mathbb{D}$ in a rather simple way.

Since the horizontal category of **Arena**$_\mathbb{D}$ is $\int^{C:\mathcal{C}} \mathcal{A}(C)$, the category of charts, we may consider T as a *double* functor

$$
hT : h\mathcal{C} \to \mathbf{Arena}_\mathbb{D}
$$

from the double category $h\mathcal{C}$ given by defining its horizontal category to be $\mathcal{C}$ and taking its vertical category and its squares to consist only of identities. Its worth taking a minute to check this trivial observation against the definition of a double functor.

Example 4.4.5. There is a double category 1 with just one object $*$ and only identity maps and squares. A double functor $F : 1 \to \mathcal{D}$ simply picks out the object $F(*)$; there is no other data involved, since everything else must get sent to the appropriate identities.

In particular, the one element set 1 is an object of the double category **Matrix** of sets, functions, and matrices. Therefore, there is a double functor $1 : 1 \to$ **Matrix** picking out this special element.

Now that we have a notion of double functor, we can define a category **Dbl** of double categories.

Definition 4.4.6. The category **Dbl** of double categories has as its objects the double categories and as its maps the double functors.

From any indexed category $\mathcal{A}$, we can form the double categories of *arenas* in $\mathcal{A}$ (**??**). In category theory, it is a good habit to inquire into the functoriality of any construction. Now that we have an appropriate category of double categories, we can ask if the construction $\mathcal{A} \mapsto \mathbf{Arena}_{\mathcal{A}}$ is functorial.

Proposition 4.4.7. The assignment $\mathcal{A} \mapsto \mathbf{Arena}_{\mathcal{A}}$ sending an indexed category to its Grothendieck double construction (**??**) is functorial.

Proof. Let $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ and $\mathcal{B} : \mathcal{D}^{\mathrm{op}} \to \mathbf{Cat}$ be indexed categories, and let $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$ be an indexed functor. We will produce a double functor

$$\binom{\overline{F}}{F} : \mathbf{Arena}_{\mathcal{A}} \to \mathbf{Arena}_{\mathcal{B}}.$$

Recall that the Grothendieck construction is functorial (**??**). From an indexed functor $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$, we get a functor

$$\binom{\overline{F}}{F} : \int^{C : \mathcal{C}} \mathcal{A}(C) \to \int^{D : \mathcal{D}} \mathcal{B}(D).$$

Since the horizontal category of **Arena** is precisely the Grothendieck construction, we can take this to be the horizontal component of $\binom{\overline{F}}{F} : \mathbf{Arena}_{\mathcal{A}} \to \mathbf{Arena}_{\mathcal{B}}$. Similarly, since the vertical category of **Arena** is the Grothendieck construction of the opposite, we can take the vertical component of $\binom{\overline{F}}{F} : \mathbf{Arena}_{\mathcal{A}} \to \mathbf{Arena}_{\mathcal{B}}$ to be $\binom{\overline{F}^{\mathrm{op}}}{F} : \int^{C : \mathcal{C}} \mathcal{A}(C)^{\mathrm{op}} \to \int^{D : \mathcal{D}} \mathcal{B}(D)^{\mathrm{op}}$.

All that remains to check then is that $\left(\begin{smallmatrix}\overline{F} \\ F\end{smallmatrix}\right) :: \mathbf{Arena}_{\mathcal{A}} \to \mathbf{Arena}_{\mathcal{B}}$ preserves squares. Let

$$
\begin{pmatrix} A_1 \\ C_1 \end{pmatrix} \xRightarrow{\left(\begin{smallmatrix} g_{1\flat} \\ g_1 \end{smallmatrix}\right)} \begin{pmatrix} A_2 \\ C_2 \end{pmatrix}
$$

be a square in $\mathbf{Arena}_{\mathcal{A}}$. We need to show that

$$
\begin{pmatrix} \overline{F}A_1 \\ FC_1 \end{pmatrix} \xRightarrow{\left(\begin{smallmatrix} \overline{F}g_{1\flat} \\ Fg_1 \end{smallmatrix}\right)} \begin{pmatrix} \overline{F}A_2 \\ FC_2 \end{pmatrix}
$$

is a square in $\mathbf{Arena}_{\mathcal{B}}$. But this being a square means that the two diagrams

$$
\begin{array}{ccc}
FC_1 & \xrightarrow{Fg_1} & FC_2 \\
{\scriptstyle Ff_1}\downarrow & & \downarrow{\scriptstyle Ff_2} \\
FC_3 & \xrightarrow{Fg_2} & FC_4
\end{array}
\qquad
\begin{array}{ccc}
(Ff_1)^*\overline{F}A_3 & \xrightarrow{\qquad\qquad \overline{F}f_1^{\sharp} \qquad\qquad} & \overline{F}A_1 \\
{\scriptstyle (Ff_1)^*\overline{F}g_{2\flat}}\downarrow & & \downarrow{\scriptstyle \overline{F}g_{1\flat}} \\
(Ff_1)^*(Fg_2)^*\overline{F}A_4 \;=\!=\; (Fg_1)^*(Ff_2)^*\overline{F}A_4 & \xrightarrow[(Fg_1)^*\overline{F}f_2^{\sharp}]{} & (Fg_1)^*\overline{F}A_2
\end{array}
$$

The left square commutes because F is a functor, and the right square commutes because $(F, \overline{F})$ is an indexed functor. $\square$

4.4.8   The Vertical Slice Construction: Definition

We are now ready to define the vertical slice construction.

Definition 4.4.9 (The Vertical Slice Construction). Let $F : \mathcal{D}_0 \to \mathcal{D}_1$ be a double functor. The *vertical slice construction* of F is the doubly indexed category

$$
\sigma F : \mathcal{D}_1 \to \mathbf{Cat}
$$

defined as follows:

- For $D \in \mathcal{D}_1$, $\sigma F(D)$ is the category whose objects are pairs (A, j) of an object $A \in \mathcal{D}_0$ and a vertical map $f : FA \to D$. A map $(A_1, j_1) \to (A_2, j_2)$ is a pair (f, α) of a horizontal $f : A_1 \to A_2$ and a square

$$
\begin{array}{ccc}
FA_1 & \xrightarrow{Ff} & FA_2 \\
\downarrow{\scriptstyle j_1} & \alpha & \downarrow{\scriptstyle j_2} \\
D & = & D
\end{array}
$$

  in $\mathcal{D}_1$.

- For every vertical $j : D \to D'$ in $\mathcal{D}_1$, we associate the functor $\sigma F(j) : \sigma F(D) \to \sigma F(D')$ given by vertical composition with j:

$$
\begin{array}{ccc}
FA_1 & \xrightarrow{Ff} & FA_2 \\
\downarrow{\scriptstyle j_1} & \alpha & \downarrow{\scriptstyle j_2} \\
D & = & D
\end{array}
\quad\longmapsto\quad
\begin{array}{ccc}
FA_1 & \xrightarrow{Ff} & FA_2 \\
\downarrow{\scriptstyle j_1} & \alpha & \downarrow{\scriptstyle j_2} \\
D & = & D \\
\downarrow{\scriptstyle j} & & \downarrow{\scriptstyle j} \\
D' & = & D'
\end{array}
$$

  More concisely, this is

$$
\sigma F(j)(f, \alpha) = \left(f, \frac{\alpha}{j} \right).
$$

- For every horizontal $g : D \to D'$ in $\mathcal{D}_1$, we associate the profunctor $\sigma F(g) : \sigma F(D) \nrightarrow \sigma F(D')$ given by

$$
\begin{array}{ccc}
FA_1 & & FA_2 \\
\downarrow{\scriptstyle j_1} & , & \downarrow{\scriptstyle j_2} \\
D & & D'
\end{array}
\quad\longmapsto\quad
\left\{ \left(f, \begin{array}{ccc}
FA_1 & \dashrightarrow{Ff} & FA_2 \\
\downarrow{\scriptstyle j_1} & \alpha & \downarrow{\scriptstyle j_2} \\
D & \xrightarrow{g} & D'
\end{array} \right) \cdot \right\}
$$

  We note that if $g = \mathrm{id}_D$ is an identity, then this reproduces the hom profunctor of $\sigma F(D)$.

- The compositor μ is given by horizontal composition:

$$
\mu_{g_1, g_2}((f_1, \alpha_1), (f_2, \alpha_2)) = (f_1 \mid f_2, \alpha_1 \mid \alpha_2).
$$

Let's check now that this does indeed satisfy the laws of a doubly indexed category. The proof is exactly as it was for **Sys**.

- (Vertical Functoriality) We show that $\sigma F\left(\frac{k_1}{k_2} \right) = \sigma F(k_2) \circ \sigma F(k_1)$ by vertical asso-

ciativity:

$$\sigma F\left(\frac{k_1}{k_2}\right)(f, \alpha) = \left(f, \frac{\alpha}{\left(\frac{k_1}{k_2}\right)}\right)$$

$$= \left(f, \frac{\left(\frac{\alpha}{k_1}\right)}{k_2}\right)$$

$$= \sigma F(k_2) \circ \sigma F(k_1)((f, \alpha)).$$

- (Horizontal Lax Functoriality) This law follows from horizontal associativity in $\mathcal{D}_1$.

$$\mu(\mu((f_1, \alpha_1), (f_2, \alpha_2)), (f_3, \alpha_3)) = ((f_1 \mid f_2) \mid f_3, (\alpha_1 \mid \alpha_2) \mid \alpha_3)$$
$$= (f_1 \mid (f_2 \mid f_3), \alpha_1 \mid (\alpha_2 \mid \alpha_3))$$
$$= \mu((f_1, \alpha_1), \mu((f_2, \alpha_2), (f_3, \alpha_3))).$$

- (Naturality of Compositor) This law follows from interchange in $\mathcal{D}_1$.

$$\left(\sigma F(\beta_1) \mid \sigma F(\beta_2)\mu\right)((f_1, \alpha_1), (f_2, \alpha_2)) = \left(f_1 \mid f_2, \frac{\phi}{\alpha}\Big|\frac{\psi}{\beta}\right)$$
$$= \left(f_1 \mid f_2, \frac{\phi \mid \psi}{\alpha \mid \beta}\right)$$
$$= \left(\frac{\mu}{\sigma F(\beta_1 \mid \beta_2)}\right)((f_1, \alpha_1), (f_2, \alpha_2)).$$

We can now see that the vertical slice construction generalizes both the constructions of $\mathbf{Sys}_\mathbb{D}$ and $\mathbf{Vec}$.

Proposition 4.4.10. The doubly indexed category $\mathbf{Sys}_\mathbb{D}$ of systems in a doctrine $\mathbb{D} = (\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, T)$ is the vertical slice construction of the double functor $hT : h\mathcal{C} \to \mathbf{Arena}_\mathbb{D}$ given by considering the section T as a double functor.

$$\mathbf{Sys}_\mathbb{D} = \sigma(hT : h\mathcal{C} \to \mathbf{Arena}_\mathbb{D}).$$

Proof. This is a matter of checking definitions and seeing that they are precisely the same. □

Proposition 4.4.11. The doubly indexed category $\mathbf{Vec}$ of vectors of sets is the vertical slice construction of the inclusion $1 : 1 \to \mathbf{Matrix}$ of the one element set into the double category of matrices of sets.

$$\mathbf{Vec} = \sigma(1 : 1 \to \mathbf{Matrix}).$$

Proof. This is also a matter of checking that the definitions coincide. □

4.4.12 Natural Transformations of Double Functors

We now turn towards proving the functoriality of the vertical slice construction as a first step in proving the change of doctrine theorem. In order to express the functoriality of the vertical slice construction, we will first need learn about natural transformations between double functors.

Since double categories have two sorts of maps — vertical and horizontal — there are also two sorts of natural transformations between double functors. The two definitions are symmetric; we may arrive at one by replacing the words "vertical" by "horizontal" and vice-versa. We will have occasion to use both of them in this and the coming chapters.

Definition 4.4.13. Let F and $G : \mathcal{D} \to \mathcal{E}$ be double functors. A *vertical natural transformation* $v : F \Rightarrow G$ consists of the following data:

- For every object $D \in \mathcal{D}$, a vertical $v_D : FD \to GD$ in $\mathcal{E}$.
- For every horizontal arrow $f : D \to D'$ in $\mathcal{D}$, a square

$$
\begin{array}{ccc}
FD & \xrightarrow{Ff} & FD' \\
v_D \downarrow & v_f & \downarrow v_{D'} \\
GD & \xrightarrow{Gf} & GD'
\end{array}
$$

This data must satisfy the following laws:

- (Vertical Naturality) For any vertical $j : D_1 \to D_2$, we have

$$
\frac{Fj}{v_{D_2}} = \frac{v_{D_1}}{Gj}.
$$

- (Horizontal Naturality) For any horizontal $f_1 : D_1 \to D_2$ and $f_2 : D_2 \to D_3$, we have

$$
v_{f_1 | f_2} = v_{f_1} \mid v_{f_2}.
$$

- (Horizontal Unity) $v_{\mathrm{id}_D} = \mathrm{id}_{v_D}$.
- (Square naturality) For any square

$$
\begin{array}{ccc}
D_1 & \xrightarrow{f_1} & D_2 \\
j_1 \downarrow & \alpha & \downarrow j_2 \\
D_3 & \xrightarrow{f_2} & D_4
\end{array}
$$

we have

$$
\frac{F\alpha}{v_{f_2}} = \frac{v_{f_1}}{G\alpha}.
$$

Dually, a *horizontal transformation* $h : F \Rightarrow G$ consists of the following data:

- For every object $D \in \mathcal{D}$ a horizontal morphism $h_D : FD \to GD$.
- For every vertical $j : D \to D'$ in $\mathcal{D}$, a square

$$
\begin{array}{ccc}
FD & \xrightarrow{\ h_D\ } & GD \\
{\scriptstyle Fj}\downarrow & h_j & \downarrow{\scriptstyle Gj} \\
FD' & \xrightarrow[\ h_{D'}\ ]{} & GD'
\end{array}
$$

This data is required to satisfy the following laws:

- (Horizontal Naturality) For horizontal $f : D_1 \to D_2$, we have

$$Ff \mid v_{D_2} = v_{D_1} \mid Gf.$$

- (Vertical Naturality) For vertical $j_1 : D_1 \to D_2$ and $j_2 : D_2 \to D_3$, we have

$$h_{\frac{j_1}{j_2}} = \frac{h_{j_1}}{h_{j_2}}.$$

- (Vertical Unity) $h_{\mathrm{id}_D} = \mathrm{id}_{h_D}$.
- (Square Naturality) For any square

$$
\begin{array}{ccc}
D_1 & \xrightarrow{\ f_1\ } & D_2 \\
{\scriptstyle j_1}\downarrow & \alpha & \downarrow{\scriptstyle j_2} \\
D_3 & \xrightarrow[\ f_2\ ]{} & D_4
\end{array}
$$

  we have

$$F\alpha \mid h_{j_2} = h_{j_1} \mid G\alpha.$$

Remark 4.4.14. Note that vertical (resp. horizontal) natural transformations are named for the direction of arrow they assign to objects. However, a vertical transformation is defined by its action v_f on *horizontal* maps f, and dually a horizontal transformation h_j by its action on *vertical* maps j. Taking f (resp. j) to be an identity id_D yields the vertical (resp. horizontal) arrow associated to the object D.

Natural transformations between double functors can be composed in the appropriate directions.

Lemma 4.4.15. Suppose that $v_1 : F_1 \Rightarrow F_2$ and $v_2 : F_2 \Rightarrow F_2$ are vertical transformations. We have a vertical composite $\frac{v_1}{v_2}$ defined by

$$\left(\frac{v_1}{v_2}\right)_f := \frac{(v_1)_f}{(v_2)_f}$$

for horizontal maps f. Dually, for horizontal transformations $h_1 : F_1 \Rightarrow F_2$ and $h_2 : F_2 \Rightarrow F_3$, there is a horizontal composite $h_1 \mid h_2$ defined by

$$(h_1 \mid h_2)_j := (h_1)_j \mid (h_2)_j$$

for every vertical map j.

Proof. We will prove that $\frac{v_1}{v_2}$ is a vertical transformation; the proof that $h_1 \mid h_2$ is a horizontal transformation is precisely dual.

- (Vertical Naturality) This follows by the same argument as for Square Naturality below, taking $\alpha = j$ for a vertical $j : D_1 \to D_2$.
- (Horizontal naturality) For horizontal maps $f_1 : D_1 \to D_2$ and $f_2 : D_2 \to D_3$, we have

$$\frac{v_1}{v_2}_{f_1 \mid f_2} = \frac{(v_1)_{f_1 \mid f_2}}{(v_2)_{f_1 \mid f_2}}$$

$$= \frac{(v_1)_{f_1} \mid (v_1)_{f_2}}{(v_2)_{f_1} \mid (v_2)_{f_2}}$$

$$= \frac{(v_1)_{f_1}}{(v_2)_{f_1}} \Bigg| \frac{(v_1)_{f_2}}{(v_2)_{f_2}}$$

$$= \left(\frac{v_1}{v_2}\right)_{f_1} \Bigg| \left(\frac{v_1}{v_2}\right)_{f_2} .$$

- (Horizontal Unity) This holds by definition.
- (Square Naturality) Consider a square α of the following signature:

$$
\begin{array}{ccc}
D_1 & \xrightarrow{f_1} & D_2 \\
{\scriptstyle j_1}\downarrow & \alpha & \downarrow{\scriptstyle j_2} \\
D_3 & \xrightarrow{f_2} & D_4
\end{array}
$$

Then

$$\frac{F_1 \alpha}{\left(\frac{v_1}{v_2}\right)_{f_2}} = \frac{\dfrac{F_1 \alpha}{(v_1)_{f_2}}}{(v_2)_{f_2}}$$

$$= \frac{\dfrac{(v_1)_{f_1}}{F_2 \alpha}}{(v_2)_{f_2}}$$

$$= \frac{\dfrac{(v_1)_{f_1}}{(v_2)_{f_1}}}{F_3\alpha}$$

$$= \frac{\left(\dfrac{v_1}{v_2}\right)_{f_1}}{F_3\alpha}.$$

$\square$

Amongst double functors we have found two sorts of maps — vertical and horizontal — each with their own sort of composition. This suggests that there should be a *double category* of double functors $\mathscr{D} \to \mathscr{E}$, just as there is a category of functors between two categories.

Theorem 4.4.16. Let $\mathscr{D}$ and $\mathscr{E}$ be double categories. There is a double category **Fun**$(\mathscr{D}, \mathscr{E})$ of double functors from $\mathscr{D}$ to $\mathscr{E}$ whose vertical maps are vertical transformations, horizontal maps are horizontal transformations, and whose squares

$$
\begin{array}{ccc}
F_1 & \xrightarrow{\ h_1\ } & F_2 \\
{\scriptstyle v_1}\downarrow & \alpha & \downarrow{\scriptstyle v_2} \\
F_3 & \xrightarrow[\ h_2\ ]{} & F_4
\end{array}
$$

are *modifications* defined in the following way. To each object $D \in \mathscr{D}$, we have a square

$$
\begin{array}{ccc}
F_1D & \xrightarrow{\ (v_1)_D\ } & F_2D \\
{\scriptstyle (h_1)_D}\downarrow & \alpha_D & \downarrow{\scriptstyle (v_2)_D} \\
F_3D & \xrightarrow[\ (h_2)_D\ ]{} & F_4D
\end{array}
$$

which satisfies the following laws:

- (Horizontal Coherence) For every horizontal $f : D_1 \to D_2$, we have that

$$(v_1)_f \mid \alpha_{D_2} = \alpha_{D_1} \mid (v_2)_f.$$

  We note that this law requires us to use the vertical naturality law of v_1 and v_2 so that these composites have the same signature.

- (Vertical Coherence) For every vertical $j : D_1 \to D_2$, we have that

$$\frac{\alpha_{D_1}}{(h_2)_j} = \frac{(h_1)_j}{\alpha_{D_2}}.$$

  We note that this law requires us to use the horizontal naturality law of h_1 and h_2 so that these composites have the same signature.

The compositions $\alpha \mid \beta$ and $\frac{\alpha}{\beta}$ are given componentwise by $\alpha_D \mid \beta_D$ and $\frac{\alpha_D}{\beta_D}$.

Proof. Since the compositions of modifications are given componentwise, they will satisfy associativity and interchange. We just need to show that they are well defined, which is to say that they satisfy the laws of a modification. This is a straightforward calculation; we'll prove Vertical Coherence for horizontal composition since the other cases are similar.

Let α and β be modifications with the following signatures:

$$
\begin{array}{ccc}
F_1 \xrightarrow{\;h_1\;} F_2 & & F_2 \xrightarrow{\;h_3\;} F_5 \\
v_1 \downarrow \quad \alpha \quad \downarrow v_2 & \text{and} & v_2 \downarrow \quad \beta \quad \downarrow v_3 \\
F_3 \xrightarrow[\;h_2\;]{} F_4 & & F_4 \xrightarrow[\;h_4\;]{} F_6
\end{array}
$$

Let $j : D_1 \to D_2$ be a vertical map in $\mathcal{D}$. We calculate:

$$
\frac{(\alpha \mid \beta)_{D_1}}{(h_2 \mid h_4)_j} = \frac{\alpha_{D_1} \mid \beta_{D_1}}{(h_2)_j \mid (h_4)_j}
$$

$$
= \frac{\alpha_{D_1}}{(h_2)_j} \Bigg| \frac{\beta_{D_1}}{(h_4)_j}
$$

$$
= \frac{(h_1)_j}{\alpha_{D_2}} \Bigg| \frac{(h_3)_j}{\beta_{D_2}}
$$

$$
= \frac{(h_1 \mid h_3)_j}{(\alpha \mid \beta)_{D_2}}.
$$

$\square$

Before we move on, let's record an important lemma relating modifications to squares.

Lemma 4.4.17. Let

$$
\begin{array}{ccc}
F_1 & \xrightarrow{\ h_1\ } & F_2 \\
{\scriptstyle v_1}\downarrow & \alpha & \downarrow{\scriptstyle v_2} \\
F_3 & \xrightarrow[\ h_2\ ]{} & F_4
\end{array}
$$

be a modification, and

$$
\begin{array}{ccc}
D_1 & \xrightarrow{\ f_1\ } & D_2 \\
{\scriptstyle j_1}\downarrow & s & \downarrow{\scriptstyle j_2} \\
D_3 & \xrightarrow[\ f_2\ ]{} & D_4
\end{array}
$$

be a square in $\mathcal{D}$. We then have the following four-fold equality in $\mathcal{E}$:

$$
\begin{array}{|c|c|}
\hline
\alpha_{D_1} & (v_2)_{f_1} \\
\hline
(h_2)_{j_1} & F_4 s \\
\hline
\end{array}
\ =\!=\
\begin{array}{|c|c|}
\hline
(v_1)_{f_1} & \alpha_{D_2} \\
\hline
F_3 s & (h_2)_{j_2} \\
\hline
\end{array}
$$

$$
\|\qquad\qquad\qquad\qquad\|
$$

$$
\begin{array}{|c|c|}
\hline
(h_1)_{j_1} & F_2 s \\
\hline
\alpha_{D_3} & (v_2)_{f_2} \\
\hline
\end{array}
\ =\!=\
\begin{array}{|c|c|}
\hline
F_1 s & (h_1)_{j_2} \\
\hline
(v_2)_{f_1} & \alpha_{D_4} \\
\hline
\end{array}
$$

We may refer to the single square given by any of these composites by α_s.

Proof. These all follow by cycling through the square naturality laws of the transformations and the coherence laws of the modification. $\qquad\square$

4.4.18　Vertical Slice Construction: Functoriality

In this section, we will describe the functoriality of the vertical slice construction. Since the vertical slice construction takes a double functor $F : \mathcal{D}_0 \to \mathcal{D}_1$ and produces a doubly indexed category $\sigma F : \mathcal{D}_1 \to \mathbf{Cat}$, we will need to show that from a certain sort of map between double functors we get a *doubly indexed functor* between the resulting vertical slices.

First, we will describe the appropriate notion of map between double functors. This gives us a category which we will call the *category of double functors* **DblFun** [1]

Definition 4.4.19. The category **DblFun** has objects the double functor $F : \mathcal{D}_0 \to \mathcal{D}_1$. A map $F_1 \to F_2$ is a triple (v_0, v_1, v) where $v_0 : \mathcal{D}_{00} \to \mathcal{D}_{10}$ and $v_1 : \mathcal{D}_{01} \to \mathcal{D}_{11}$ are

---

[1]Though one could define other categories whose objects are double functors, this is the only such category we will use in this book.

double functors and $v : F_2 \circ v_0 \Rightarrow v_1 \circ F_1$ is a vertical transformation.

$$
\begin{array}{ccc}
\mathcal{D}_{00} & \xrightarrow{v_0} & \mathcal{D}_{10} \\
\downarrow{\scriptstyle F_1} & \swarrow{\scriptstyle v} & \downarrow{\scriptstyle F_2} \\
\mathcal{D}_{01} & \xrightarrow{v_1} & \mathcal{D}_{11}
\end{array}
$$

Composition of $(v_0, v_1, v) : F_1 \to F_2$ with $(w_0, w_1, w) : F_2 \to F_3$ is given by $(w_0 \circ v_0, w_1 \circ v_1, v * w)$ where $v * w$ is the vertical transformation with horizontal components given by

$$
(v * w)_f := \frac{w_{v_0 f}}{w_1 v_f}.
$$

It remains to check that this does indeed yield a category. We leave this as an exercise, since it gives some good practice in using all the various laws for double functors and double transformations.

Exercise 4.4.20. Prove that the definition of **DblFun** does indeed yield a category. That is:

1. Prove that $(\mathrm{id}_{\mathcal{D}_0}, \mathrm{id}_{\mathcal{D}_1}, \mathrm{id}_F)$ provides an identity map $F \to F$.
2. Prove that composition is associative. The key part will be showing that $(v*w)*u = v*(w*u)$.

$\diamond$

Next, we need to describe the appropriate category of doubly indexed categories. There are two sorts of maps of doubly indexed categories which we will need in this book: *lax* doubly indexed functors, and (*taut*) doubly indexed functors. In this chapter, we will be using *taut* doubly indexed functors — which we may just call doubly indexed functors — which are a special case of the more general lax variety.

Definition 4.4.21. Let $\mathcal{A} : \mathcal{D}_1 \to \mathbf{Cat}$ and $\mathcal{B} : \mathcal{D}_2 \to \mathbf{Cat}$ be doubly indexed categories. A *lax doubly indexed functor* $(F^0, F) : \mathcal{A} \to \mathcal{B}$ consists of:

$$
\begin{array}{ccc}
\mathcal{D}_1 & \xrightarrow{\mathcal{A}} & \\
\downarrow{\scriptstyle F^0} & \Downarrow{\scriptstyle F} & \mathbf{Cat} \\
\mathcal{D}_2 & \xrightarrow{\mathcal{B}} &
\end{array}
$$

1. A double functor

$$
F^0 : \mathcal{D}_1 \to \mathcal{D}_2.
$$

2. For each object $D \in \mathcal{D}_1$, a functor

$$F^D : \mathcal{A}(D) \to \mathcal{B}(F^0 D).$$

3. For every vertical map $j : D_1 \to D_2$ in $\mathcal{D}_1$, a natural transformation

We ask that $F^{\mathrm{id}_D} = \mathrm{id}$. We recall (from **??**) that we may think of such a natural transformation as a square

4. For every horizontal map $f : D_1 \to D_2$, a square

in **Cat**. We ask that $F^{\mathrm{id}_D} = \mathrm{id}$.

This data is required to satisfy the following laws:

- (Vertical Lax Functoriality) For composable vertical maps $j : D_1 \to D_2$ and $k : D_2 \to D_3$,

$$F^{\frac{j}{k}} = $$

This is, in terms of squares in **Cat**:

$$F^{\frac{j}{k}} \;\dot{=}\; \frac{F^j}{\mathcal{B}(F^0 k)} \bigg| \frac{\mathcal{A}(j)}{F^k}$$

- (Horizontal functoriality) For composable horizontal arrows $f : D_1 \to D_2$ and $g : D_2 \to D_3$,

$$\frac{\mu^{\mathscr{A}}_{f,g}}{F^{f|g}} = \frac{F^f \mid F^g}{\mu^{\mathscr{B}}_{F^0 f, F^0 g}}.$$

- (Functorial Interchange) For any square

$$
\begin{array}{ccc}
D_1 & \xrightarrow{\ f\ } & D_2 \\
{\scriptstyle j}\downarrow & \alpha & \downarrow{\scriptstyle k} \\
D_3 & \xrightarrow[\ g\ ]{} & D_4
\end{array}
$$

in $\mathscr{D}_1$, we have that

$$F^j \left|\, \frac{\mathscr{A}(\alpha)}{F^g} \right. \;\overset{\cdot}{=}\; \left. \frac{F^f}{\mathscr{B}(F^0\alpha)} \,\right| F^k .$$

Note the use of "$\overset{\cdot}{=}$" here; the two sides of this equation have different, but canonically isomorphic boundary. What we are asking is that when these boundaries are made the same by composing with canonical isomorphisms in any way, they will become equal

A lax doubly indexed functor is *taut* — which we will in refer to just as a doubly indexed functor — if the natural transformations F^j associated to vertical maps $j : D_1 \to D_2$ in $\mathscr{D}_1$ are natural isomorphisms.

The definition of doubly indexed functor involves a lot of data, but this is because it is a big collection of functoriality results.

We need to compose lax doubly indexed functors.

Definition 4.4.22. If $(F^0, F) : \mathscr{A} \to \mathscr{B}$ and $(G^0, G) : \mathscr{B} \to \mathscr{C}$ are two doubly indexed functors, we define their composite

$$(F^0, F) \,\mathbin{\mathring{,}}\, (G^0, G) := (F^0 \,\mathring{,}\, G^0, F \,\mathring{,}\, G)$$

where $F \,\mathring{,}\, G$ is defined by:

- We define $(F \,\mathring{,}\, G)^D := F^D \,\mathring{,}\, G^{F_0 D}$. We note that in **Cat**, where functors are the vertical maps, this can be written

$$(F \,\mathring{,}\, G)^D = \frac{F^D}{G^{F^0 D}}.$$

- For a vertical $j : D_1 \to D_2$, we define

$$
\begin{array}{ccc}
\mathcal{A}(D_1) \xrightarrow{(F\,\fatsemi\,G)^{D_1}} \mathcal{C}(G^0F^0D_1) & & \mathcal{A}(D_1) \xrightarrow{F^{D_1}} \mathcal{B}(F^0D_1) \xrightarrow{G^{F^0D_1}} \mathcal{C}(G^0F^0D_1) \\
\mathcal{A}(j)\Big\downarrow {\scriptstyle (F\,\fatsemi\,G)^j} \quad \Big\downarrow{\scriptstyle \mathcal{C}(G^0F^0 j)} \;\; := \;\; \mathcal{A}(j)\Big\downarrow {\scriptstyle F^j} \quad \Big\downarrow{\scriptstyle \mathcal{B}(F^0 j)}{\scriptstyle G^{F^0 j}} \quad \Big\downarrow{\scriptstyle \mathcal{C}(G^0F^0 j)} \\
\mathcal{A}(D_2) \xrightarrow[(F\,\fatsemi\,G)^{D_2}]{} \mathcal{C}(G^0F^0D_2) & & \mathcal{A}(D_2) \xrightarrow[F^{D_2}]{} \mathcal{B}(F^0D_2) \xrightarrow[G^{F^0D_2}]{} \mathcal{C}(G^0F^0D_2)
\end{array}
$$

We note that by **??**, this corresponds to the composite of squares:

$$
(F \,\fatsemi\, G)^j \;\doteq\; \frac{F^{D_1}}{G^{F^0 j}} \bigg| \frac{F^j}{G^{F^0 D_2}}
$$

- For a horizontal $f : D_1 \to D_2$, we define

$$
(F \,\fatsemi\, G)^f \;:=\; \frac{F^f}{G^{F^0 f}}.
$$

We refer to the category of doubly indexed categories and lax doubly indexed functors by **LaxDblIx** and the category of doubly indexed categories and (taut) doubly indexed functors by **DblIx**.

Let's show that this composition operation does indeed produce a lax doubly indexed functor.

- (Vertical Lax Functoriality) For composable vertical maps $j : D_1 \to D_2$ and $k : D_2 \to D_3$, consider the following diagram:

$$
\begin{array}{ccc}
\mathcal{A}(D_1) \xrightarrow{F^{D_1}} \mathcal{B}(F^0D_1) \xrightarrow{G^{F^0D_1}} \mathcal{C}(G^0F^0D_1) \\
\mathcal{A}(j)\Big\downarrow {\scriptstyle F^j} \quad \Big\downarrow{\scriptstyle \mathcal{B}(F^0 j)}{\scriptstyle G^{F^0 j}} \quad \Big\downarrow{\scriptstyle \mathcal{C}(G^0F^0 j)} \\
\mathcal{A}(D_2) \xrightarrow{F^{D_2}} \mathcal{B}(F^0D_2) \xrightarrow{G^{F^0D_2}} \mathcal{C}(G^0F^0D_2) \\
\mathcal{A}(k)\Big\downarrow {\scriptstyle F^k} \quad \Big\downarrow{\scriptstyle \mathcal{B}(F^0 k)}{\scriptstyle G^{F^0 k}} \quad \Big\downarrow{\scriptstyle \mathcal{C}(G^0F^0 k)} \\
\mathcal{A}(D_3) \xrightarrow[F^{D_3}]{} \mathcal{B}(F^0D_3) \xrightarrow[G^{F^0D_2}]{} \mathcal{B}(G^0F^0D_3)
\end{array}
$$

There is a single natural transformation given as the composite of this "pasting diagram". But, if we read it by composing vertically first, and then composing horizontally second, we arive at $(F \,\fatsemi\, G)^{\substack{j\\k}}$, while if we read it by composing horizontally first and then vertically second, we get the composite of $(F \,\fatsemi\, G)^j$ and $(F \,\fatsemi\, G)^k$ as desired.

- (Horizontal Functoriality) Let $f : D_1 \to D_2$ and $g : D_2 \to D_3$ be horizontal maps. We then calculate:

$$
\frac{\mu^{\mathcal{A}}_{f,g}}{(F \,\fatsemi\, G)^{f|g}} = \frac{\dfrac{\mu^{\mathcal{A}}_{f,g}}{F^{f|g}}}{G^{F^0(f|g)}}
$$

$$= \frac{\dfrac{F^f \mid F^g}{\mu^{\mathcal{B}}_{F^0 f, F^0 g}}}{G^{F^0 f \mid F^0 g}}$$

$$= \frac{\dfrac{F^f \mid F^g}{G^{F^0 f} \mid G^{F^0 g}}}{\mu^{\mathcal{C}}_{G^0 F^0 f, G^0 F^0 g}}$$

$$= \frac{\dfrac{\dfrac{F^f}{G^{F^0 f}} \Big| \dfrac{F^g}{G^{F^0 g}}}{\mu^{\mathcal{C}}_{G^0 F^0 f, G^0 F^0 g}}}{}$$

$$= \frac{(F \mathbin{\mathaccent\cdot{;}} G)^f \mid (F \mathbin{\mathaccent\cdot{;}} G)^g}{\mu^{\mathcal{C}}_{G^0 F^0 f, G^0 F^0 g}}.$$

- (Functorial Interchange) Consider a square

$$
\begin{array}{ccc}
D_1 & \xrightarrow{\ f\ } & D_2 \\
{\scriptstyle j}\downarrow & \alpha & \downarrow{\scriptstyle k} \\
D_3 & \xrightarrow[\ g\ ]{} & D_4
\end{array}
$$

We may then calculate:

Now that we have a category of doubly indexed categories, we can state the functoriality result:

Theorem 4.4.23. The vertical slice construction (**??**) gives a functor

$$\sigma : \mathbf{DblFun} \to \mathbf{DblIx}.$$

We will spend the rest of this section proving this theorem.

Proposition 4.4.24. Let $(v_0, v_1, v) : F_1 \to F_2$ be a map in **DblFun**. Then we have a doubly indexed functor

$$
\begin{array}{ccc}
\mathcal{D}_{01} & \xrightarrow{\quad \sigma F_1 \quad} & \\
\Big\downarrow{\scriptstyle v_1} & \Big\Downarrow{\scriptstyle \sigma v} & \textbf{Cat} \\
\mathcal{D}_{11} & \xrightarrow{\quad \sigma F_2 \quad} &
\end{array}
$$

Proof. We define σv as follows:

- We have $\sigma v^D : \sigma F_1(D) \to \sigma F_2(v_1 D)$ given by the following action on maps:

$$
\begin{array}{ccc}
F_1 A_1 & \xrightarrow{F_1 f} & F_1 A_2 \\
{\scriptstyle j_1}\Big\downarrow & \alpha & \Big\downarrow{\scriptstyle j_2} \\
D & =\!=\!=\!= & D
\end{array}
\quad\mapsto\quad
\begin{array}{ccc}
F_2 v_0 A_1 & \xrightarrow{F_2 v_0 f} & F_2 v_0 A_2 \\
{\scriptstyle v_{A_1}}\Big\downarrow & {\scriptstyle v_f} & \Big\downarrow{\scriptstyle v_{A_2}} \\
v_1 F_1 A_1 & \xrightarrow{v_1 F_1 f} & F A_2 \\
{\scriptstyle v_1 j_1}\Big\downarrow & {\scriptstyle v_1 \alpha} & \Big\downarrow{\scriptstyle v_1 j_2} \\
v_1 D & =\!=\!=\!= & v_1 D
\end{array}
$$

In short:

$$
\sigma v^D(f, \alpha) := \left(v_0 f, \, \frac{v_f}{v_1 \alpha} \right).
$$

- For any vertical map $j : D_1 \to D_2$ in $\mathcal{D}_{01}$, we will show that

$$
\sigma F_2(v_1 j) \circ \sigma v^{D_1} = \sigma v^{D_2} \circ \sigma F_1(j)
$$

so that we may take σv^j to be the identity natural transformation.

$$
\begin{aligned}
\sigma F_2(v_1 j) \circ \sigma v^{D_1}(f, \alpha) &= \sigma F_2(v_1 j)\left(v_0 f, \frac{v_f}{v_1 \alpha} \right) \\
&= \left(v_0 f, \, \frac{\dfrac{v_f}{v_1 \alpha}}{v_1 j} \right) \\
&= \left(v_0 f, \, \frac{v_f}{v_1\left(\frac{\alpha}{j} \right)} \right) \\
&= \sigma v^{D_2}\left(f, \frac{\alpha}{j} \right) \\
&= \sigma v^{D_2} \circ \sigma F_1(j)(f, \alpha)
\end{aligned}
$$

- For a horizontal map $\varphi : D_1 \to D_2$, we give the square

$$
\begin{array}{ccc}
\sigma F_1(D_1) & \xrightarrow{\sigma F_1(f)} & \sigma F_1(D_2) \\
{\scriptstyle \sigma v^{D_1}}\downarrow & {\scriptstyle \sigma v^{\varphi}} & \downarrow{\scriptstyle \sigma v^{D_2}} \\
\sigma F_2(v_1 D_2) & \xrightarrow{\sigma F_2(v_1 f)} & \sigma F_2(v_1 D_2)
\end{array}
$$

defined by

$$
\begin{array}{ccc}
F_1 A_1 & \xrightarrow{F_1 f} & F_1 A_2 \\
{\scriptstyle j_1}\downarrow & \alpha & \downarrow{\scriptstyle j_2} \\
D & \xrightarrow{\varphi} & D
\end{array}
\quad\mapsto\quad
\begin{array}{ccc}
F_2 v_0 A_1 & \xrightarrow{F_2 v_0 f} & F_2 v_0 A_2 \\
{\scriptstyle v_{A_1}}\downarrow & {\scriptstyle v_f} & \downarrow{\scriptstyle v_{A_2}} \\
v_1 F_1 A_1 & \xrightarrow{v_1 F_1 f} & FA_2 \\
{\scriptstyle v_1 j_1}\downarrow & {\scriptstyle v_1 \alpha} & \downarrow{\scriptstyle v_1 j_2} \\
v_1 D & \xrightarrow{v_1 \varphi} & v_1 D
\end{array}
$$

In short:

$$
\sigma v^{\varphi}(f, \alpha) := \left(v_0 f, \frac{v_f}{v_1 \alpha} \right).
$$

We will show that this data satisfies the laws of a doubly indexed functor.

- (Vertical Lax Functoriality) As we've taken the natural transformations σv^j to be identities, they are functorial since composites of identities are identities.
- (Horizontal functoriality) For composable horizontal maps $\varphi_1 : D_1 \to D_2$ and $\varphi_2 : D_2 \to D_3$, we may calculate:

$$
\left(\frac{\mu^{\sigma F_1}_{\varphi_1, \varphi_2}}{\sigma v^{\varphi_1 | \varphi_2}} \right) ((f_1, \alpha_1), (f_2, \alpha_2)) = \left(v_1(f_1 \mid f_2), \frac{v_{f_1 | f_2}}{v_1(\alpha_1 \mid \alpha_2)} \right)
$$

$$
= \left(v_1 f_1 \mid v_1 f_2, \frac{v_{f_1} \mid v_{f_2}}{v_1 \alpha_1 \mid v_1 \alpha_2} \right)
$$

$$
= \left(v_1 f_1 \mid v_1 f_2, \frac{v_{f_1}}{v_1 \alpha_1} \middle| \frac{v_{f_2}}{v_1 \alpha_2} \right)
$$

$$
= \left(\frac{v^{\varphi_1} \mid v^{\varphi_2}}{\mu^{\sigma F_2}_{v_1 \varphi_1, v_2 \varphi_2}} \right) ((f_1, \alpha_1), (f_2, \alpha_2)).
$$

- (Functorial Interchange) Consider a square

$$
\begin{array}{ccc}
D_1 & \xrightarrow{\varphi_1} & D_2 \\
{\scriptstyle j_1}\downarrow & \beta & \downarrow{\scriptstyle f_2} \\
D_3 & \xrightarrow{\varphi_2} & D_4
\end{array}
$$

Since σv^{j_1} and σv^{j_2} are identities, we just need to show that

$$
\frac{\sigma F_1(\beta)}{\sigma v^{\varphi_2}} = \frac{\sigma v^{\varphi_1}}{\sigma F_2(v_1 \beta)}.
$$

To that end, we calculate:

$$\left(\frac{\sigma F_1(\beta)}{\sigma v^{\varphi_2}}\right)(f, \alpha) = \sigma v^{\varphi_2}\left(f, \frac{\alpha}{\beta}\right)$$

$$= \left(v_1 f, \frac{v_f}{v_1\left(\frac{\alpha}{\beta}\right)}\right)$$

$$= \left(v_1 f, \frac{v_f}{\frac{v_1\alpha}{v_1\beta}}\right)$$

$$= \sigma F_2(v_1\beta)\left(v_1 f, \frac{v_f}{v_1\alpha}\right)$$

$$= \left(\frac{\sigma v^{\varphi_1}}{F_2(v_1\beta)}\right)(f, \alpha).$$

$\square$

We now finish the proof of **??**.

Lemma 4.4.25. The assignment $(v_0, v_1, v) \mapsto (v_1, \sigma v)$ defined in **??** is functorial.

Proof. Let $(v_0, v_1, v) : F_1 \to F_2$ and $(w_0, w_1, w) : F_2 \to F_3$ be maps in **DblFun**. We will show that

$$(v_1 \mathbin{\fatsemi} w_1, \ \sigma(v * w)) = (v_1, \sigma v) \mathbin{\fatsemi} (w_1, \sigma w).$$

The first components of these pairs are equal by definition, so we just need to show that $\sigma(v * w) = \sigma v \mathbin{\fatsemi} \sigma w$.

- This calculation is the same as for a general horizontal.
- For a vertical $j : D_1 \to D_2$, we calculate, we note that both sides are the same identity natural transformation.
- For a horizontal $\varphi : D_1 \to D_2$, we calculate:

$$\sigma(v * w)^{\varphi}(f, \alpha) := \left(w_0 v_0 f, \frac{(v * w)_f}{w_1 v_1 \alpha}\right)$$

$$= \left(w_0 v_0 f, \frac{\frac{w_{v_0 f}}{w_1 v_f}}{w_1 v_1 \alpha}\right)$$

$$= \left(w_0 v_0 f, \frac{w_{v_0 f}}{w_1\left(\frac{v_f}{v_1\alpha}\right)}\right)$$

$$= \sigma w^{v_1 \varphi}\left(v_0 f, \frac{v_f}{v_1\alpha}\right)$$

$$= (\sigma v^{\varphi}) \mathbin{\fatsemi} (\sigma w^{v_1 \varphi})(f, \alpha).$$

$\square$

4.5 Change of Doctrine

We have learned about a variety of doctrines through the course of this book:

- There are the deterministic doctrines (**??**)

$$(\mathbf{Ctx}_- : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, \ \phi \mapsto \phi \circ \pi_2)$$

  which may be defined for any cartesian category $\mathcal{C}$. While we have focused so far on the case $\mathcal{C} = \mathbf{Set}$, many other cartesian categories are of interest in the study of deterministic dynamical systems. For example, in ergodic theory we most often use the category of measureable spaces and measurable functions.[2] We often assume the dynamics of the systems are not arbitary set maps, but are furthermore continuous or differentiable; this means working in the cartesian categories of topological spaces or differentiable manifolds.

- There are also the differential doctrines (**??????**) where the tangent bundle plays an important role. Jaz: There are also non-standard differential doctrines arising from cartesian differential categories and tangent categories with display maps.

- There are the non-deterministic doctrines for any commutative monad M on a cartesian category $\mathcal{C}$. As we saw in **??**, by varying the monad M we can achieve a huge variety of flavors of non-determinism. This includes possibilistic and stochastic non-determinism, but also other variants like systems with cost-sensitive transitions and (**??**).

These are just large classes of doctrines that have been easy to describe in generality. Different particular situations will require different particular doctrines. For example, we may decide to restrict the sorts of maps appearing in our doctrines by changing the base $\mathcal{C}$ as in **??**. There may also be doctrines constructed by hand for particular purposes, such as ergodic theory.

These doctrines are not isolated from each other. We have seen already in **??** that some doctrines may be formed by restricting others. There are also some apparent inclusions of doctrines that are not explained by restriction; for example, the Euclidean differential doctrine is a special case of the general differential doctrine. We should be able to think of Euclidean differential systems and general differential systems without too much hassle, and we should be able to apply theorems that pertain to general differential systems to Euclidean ones. Another example of inclusion of doctrines is of deterministic systems into non-deterministic systems of any flavor.

There are also more drastic ways to change doctrine. Any map of commutative monads $\phi : M \to N$ gives us a way of changing an M-system into an N-system, changing the flavor of non-determinism. We may also *approximate* a differential system by a deterministic system.

---

[2]We most often consider maps which preserve a specific measure on a space as well, but the category of such measure preserving maps is not cartesian. Often one needs to go and twiddle these general definitions of doctrines in particular cases to suit the particular needs of a subject.

These are all ways of *changing our doctrine*, and it is these changes of doctrine that we will attend to in this section.

We will begin by defining a *change of doctrine*, which will give us a category of doctrines.[3] We will then show that forming the doubly indexed category of systems **Sys**($\mathbb{D}$) is functorial in the doctrine $\mathbb{D}$.

4.5.1   Definition

Let's recall the informal and formal definitions of dynamical system doctrines.

The informal definition is that a doctrine is a way to answer a series of questions about what it means to be a dynamical system.

Informal Definition 4.5.2. A *doctrine* of dynamical systems is a particular way to answer the following questions about what it means to be a dynamical system:

1. What does it mean to be a state?
2. How should the output vary with the state — discretely, continuously, linearly?
3. Can the kinds of input a system takes in depend on what it's putting out, and how do they depend on it?
4. What sorts of changes are possible in a given state?
5. What does it mean for states to change.
6. How should the way the state changes vary with the input?

This informal definition is captured by the sparse, formal definition that a doctrine is a pair consisting of an indexed category $\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}$ together with a section T. The various questions correspond to the choices one can make when defining such a pair.

To change a doctrine, then, means to change our answers to these questions. We want to enact this change by some formulated process. For example, if what it means to be a state is a to be a vector in Euclidean space, and we would like to change this to instead answer that to be a state means to be an element of an abstract set, then we want a way of taking Euclidean spaces and producing an abstract set.

Now, we can't just fiddle arbitrarily with the answers to our questions; they all have to hang together in a coherent way. The formal definition can guide us to what sort of changes we can make that cohere in just this way. We can change what it means to be a state, how the output varies with the state, and the way the inputs vary by changing the indexed category $\mathcal{A}$.

Suppose that $(\mathcal{A}, T_1)$ and $(\mathcal{B}, T_2)$ are dynamical system doctrines. If we have an indexed functor (**??**) $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$ between indexed categories, then from a dynamical

---

[3] Jaz: Actually a 2-category, but this will likely go into an appendix.

system $\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} T_1\text{States} \\ \text{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$ we can get a lens

$$\begin{pmatrix} \overline{F}\text{update}_S \\ F\text{expose}_S \end{pmatrix} : \begin{pmatrix} \overline{F}T_1\text{States} \\ F\text{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \overline{F}\text{In}_S \\ F\text{Out}_S \end{pmatrix}$$

That is, we have changed what it means to be a state ($F\text{States}$), how the output varies with state ($F\text{expose}_S$), and how the inputs vary with output ($\overline{F}\text{In}_S$). This is not quite a dynamical system, however, since since its domain is not $\begin{pmatrix} T_2 F\text{States} \\ F\text{States} \end{pmatrix}$. In order for us to get a $(\mathcal{B}, T_2)$-system, we need to say how to change the what it means for a state to change.

The most direct way to produce a $(\mathcal{B}, T_2)$-system would be to compose with a map $\phi : \overline{F}T_1\text{States} \to T_2 F\text{States}$ which tells us how to take a T_1 change (re-interpreted already by $\overline{F}$), and get a T_2 change (for the re-interpretation of state by F). Indeed, if we considered this map ϕ as a lens $\begin{pmatrix} \phi \\ \text{id} \end{pmatrix} : \begin{pmatrix} T_2 F\text{States} \\ F\text{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \overline{F}T_1\text{States} \\ F\text{States} \end{pmatrix}$, we may form the composite

$$\begin{pmatrix} \overline{F}\text{update}_S \\ F\text{expose}_S \end{pmatrix} \,\mathbin{\overset{\circ}{,}}\, \begin{pmatrix} \phi \\ \text{id} \end{pmatrix} : \begin{pmatrix} T_2 F\text{States} \\ F\text{States} \end{pmatrix} \leftrightarrows \begin{pmatrix} \overline{F}\text{In}_S \\ F\text{Out}_S \end{pmatrix}.$$

This is a $(\mathcal{B}, T_2)$-system, and this process is how we may use a change of doctrine to turn $(\mathcal{A}, T_1)$-systems into $(\mathcal{B}, T_2)$-systems.

We therefore arrive at the following formal defintion of change of doctrine.

Definition 4.5.3. Let $(\mathcal{A} : \mathcal{C} \to \mathbf{Cat}, T_1)$ and $(\mathcal{B} : \mathcal{D} \to \mathbf{Cat}, T_2)$ be dynamical system doctrines. A *change of doctrine* $((F, \overline{F}), \phi) : (\mathcal{A}, T_1) \to (\mathcal{B}, T_2)$ consists of:
- An indexed functor $(F, \overline{F}) : \mathcal{A} \to \mathcal{B}$.
- A *transformation of sections* $\phi : \overline{F}T_1 \to T_2 F$, which consists of a family of maps $\phi_C : \overline{F}T_1 C \to T_2 FC$ for each C in C, satisfying the following naturality condition:
  - For any $f : C \to C'$, we have that the following square commutes in $\mathcal{B}(FC)$:

$$\begin{array}{ccc} \overline{F}T_1 C & \xrightarrow{\phi_C} & T_2 FC \\ {\scriptstyle \overline{F}T_1 f}\downarrow & & \downarrow{\scriptstyle T_2 Ff} \\ \overline{F}f^* T_1 C' & \xrightarrow[(Ff)^*\phi_{C'}]{} & (Ff)^* T_2 FC' \end{array} \qquad (4.5)$$

We can package the transformation of sections into a natural transformation, which will make it easier to work with theoretically.

Proposition 4.5.4. The data of a transformation of sections as in **??** is equivalent to the data of a natural transformation $\left(\begin{smallmatrix}\phi\\\mathrm{id}\end{smallmatrix}\right) : \left(\begin{smallmatrix}\overline{F}\\F\end{smallmatrix}\right) \circ T_1 \Rightarrow \left(\begin{smallmatrix}T_2(-)\\(-)\end{smallmatrix}\right) \circ F$ which acts as the identity on F on its bottom component. We can express this condition with the following equation on diagrams of natural transformations:

$$
\begin{array}{ccc}
\end{array}
$$

Remark 4.5.5. We note that the components of the natural transformation $\left(\begin{smallmatrix}\phi\\\mathrm{id}\end{smallmatrix}\right)$ here are *charts* and not lenses. We will, however, exploit the duality between lenses and charts whose lower component are identities.

Proof. That the transformation $\left(\begin{smallmatrix}\phi\\\mathrm{id}\end{smallmatrix}\right)$ acts as the identity on F means that it is determined by its top map ϕ. We can then see that the naturality square for ϕ is precisely the square given in **??**. $\square$

Every restriction (from **??**) is a change of doctrine.

Proposition 4.5.6. Let $\mathbb{D} = (\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, T)$ be a doctrine, and let $F : \mathcal{D} \to \mathcal{C}$ be a functor. Then there is a change of doctrine $((F, \mathrm{id}), \mathrm{id}) : \mathbb{D}_{|F} \to \mathbb{D}$ from the restriction $\mathbb{D}_F = (\mathcal{A} \circ F^{\mathrm{op}}, T \circ F)$ (**??**) of $\mathbb{D}$ by F to $\mathbb{D}$.

Proof. By definition, $(F, \mathrm{id}) : \mathcal{A} \to (\mathcal{A} \circ F^{\mathrm{op}})$ is an indexed functor. Since, by **??**, the data of a transformation of sections is the same as a natural transformation of a certain sort, we may take that transformation to be the identity. $\square$

There are, however, more interesting changes of doctrine. For example, every morphism of commutative monads gives rise to a change of doctrine.

Proposition 4.5.7. Let $\phi : M \to N$ be a morphism of commutative monads on a cartesian category $\mathcal{C}$. Then there is a change of doctrine given by

$$((\mathrm{id}, \phi_*), \mathrm{id}) : \mathbb{N}\textsc{ondet}_M \to \mathbb{N}\textsc{ondet}_N.$$

Proof. We constructed the indexed functor $(\mathrm{id}, \phi_*) : \mathbf{Ctx}_-^M \to \mathbf{Ctx}_-^N$ in **??**. It remains to show that the following square of functors commutes, so that we may take the

transformation of sections to be the identity:

$$
\begin{array}{ccc}
\mathcal{C} & =\!=\!=\!=\!= & \mathcal{C} \\
{\scriptstyle T^M}\big\downarrow & & \big\downarrow{\scriptstyle T^N} \\
\int^{C:\mathcal{C}} \mathbf{Ctx}_C^M & \xrightarrow[\left(\begin{smallmatrix}\phi_* \\ \mathrm{id}\end{smallmatrix}\right)]{} & \int^{C:\mathcal{C}} \mathbf{Ctx}_C^N
\end{array}
$$

Let $f : C' \to C$ be a map in $\mathcal{C}$. Then $T^M f$ is $\pi_2 \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, f \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, \eta^M : C' \times C' \to MC$ and $T^N f$ is $\pi_2 \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, f \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, \eta^N : C' \times C' \to NC$. Now, $\phi_* T^M f$ is $\pi_2 \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, f \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, \eta^M \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, \phi_C$, but by the unit law for morphisms of commutative monads, $\eta^M \,\mathbin{\raise0.2ex\hbox{$\fatsemi$}}\, \phi_C = \eta^N$. So the square commutes and we can take the transformation of sections to be the identity. □

Example 4.5.8. For any commutative monad $M : \mathcal{C} \to \mathcal{C}$, there is a unique commutative monad map from the identity monad $\mathrm{id}_\mathcal{C}$. Therefore, **??** gives us a change of doctrine $\mathbb{D}\textsc{et}_\mathcal{C} \to \mathbb{N}\textsc{ondet}_M$ which lets us interpret deterministic systems as special cases of non-deterministic systems.

Example 4.5.9. **??** constructs a commutative monad morphism $\phi : \mathrm{D} \to \mathrm{P}$ sending a probability distribution to the set of elements with non-zero probability. Therefore, **??** gives us a change of doctrine $\mathbb{N}\textsc{ondet}_\mathrm{D} \to \mathbb{N}\textsc{ondet}_\mathrm{P}$ which reinterprets a probabilistic system as a possibilistic one where the state s' is possibly the udpate $\phi_* \mathsf{update}_\mathsf{S}(s, i)$ of state s with input i just when just when the probability $\mathsf{update}_\mathsf{S}(s, i)(s')$ that s will transition to s' on input i is non-zero.

We may also describe changes of doctrines between various sorts of deterministic doctrine.

Proposition 4.5.10. Let $F : \mathcal{C} \to \mathcal{D}$ be a cartesian functor between cartesian categories. Then there is a change of doctrine

$$
((F, \overline{F}), \mathrm{id}) : \mathbb{D}\textsc{et}_\mathcal{C} \to \mathbb{D}\textsc{et}_\mathcal{D}
$$

from the deterministic doctrine in $\mathcal{C}$ to the cartesian doctrine in $\mathcal{D}$.

Proof. We need to construct the indexed functor $(F, \overline{F})$, and then prove that the square

$$
\begin{array}{ccc}
\mathcal{C} & \xrightarrow{\;\;F\;\;} & \mathcal{C} \\
{\scriptstyle T^\mathcal{C}}\big\downarrow & & \big\downarrow{\scriptstyle T^\mathcal{D}} \\
\int^{C:\mathcal{C}} \mathbf{Ctx}_C & \xrightarrow[\left(\begin{smallmatrix}\overline{F} \\ F\end{smallmatrix}\right)]{} & \int^{D:\mathcal{D}} \mathbf{Ctx}_D
\end{array}
$$

commutes, so that we may take the transformation of sections to be the identity.

We begin first by constructing $\overline{F}$. We note that since F is cartesian, it extends to a functor

$$\overline{F}_C : \mathbf{Ctx}_C \to \mathbf{Ctx}_{FC}$$

by sending $f : C \times X \to Y$ to $Ff : FC \times FX \to FY$. It is routine to check that this makes $(F, \overline{F})$ into an indexed functor. In particular, for a map $r : C' \to C$ in $\mathcal{C}$, we see that

$$\overline{F}_{C'}(r^*f) = \overline{F}((r \times \mathrm{id}) \, \mathring{,} \, f) = F((r \times \mathrm{id}) \, \mathring{,} \, f) = (Fr \times \mathrm{id}) \, \mathring{,} \, Ff = (Fr)^*\overline{F}(f)$$

Next we check that the square commutes. Let $f : C' \to C$ be a map in $\mathcal{C}$. Then $T^{\mathcal{D}} \circ F(f) = \begin{pmatrix} \pi_2 \mathring{,} Ff \\ Ff \end{pmatrix}$, while $\begin{pmatrix} \overline{F} \\ F \end{pmatrix}\begin{pmatrix} Tf \\ f \end{pmatrix} = \begin{pmatrix} \overline{F}(\pi_2 \mathring{,} f) \\ Ff \end{pmatrix}$. But since F is cartesian, $F(\pi_2) = \pi_2$, so these are equal. $\qquad\qquad\square$

Example 4.5.11. **??** gives us a number of trivial ways to change the flavor of our deterministic systems.

For example, it is obvious that any deterministic dynamical system whose update and expose maps are continuous gives rise to a deterministic dynamcial system without the constraint of continuity, simply by forgetting that the maps are continuous. We formalize this observation by applying **??** to the forgetful functor U : **Top** → **Set** which sends a topological space to its underlying set of points.

Similarly, any deterministic dynamical system gives rise to a continuous deterministic dynamical system if we equip all sets involved with the discrete topology. This is formalized by applying **??** to the functor disc : **Set** → **Top** which equips a set with the discrete topology.

The most interesting examples of changes of doctrine are the ones which move between different sorts of doctrine, such as from differential to deterministic. An example of this is the Euler approximation, which takes a Eulidean differential system to a deterministic system.

Let's take a minute to recall the Euler method. If $\begin{pmatrix} u \\ r \end{pmatrix} : \begin{pmatrix} \mathbb{R}^n \\ \mathbb{R}^n \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^k \\ \mathbb{R}^m \end{pmatrix}$ is a differential system representing the differential equation

$$\frac{ds}{dt} = u(s, i),$$

then for a sufficiently small $\varepsilon > 0$, the state at time $t + \varepsilon$ will be roughly

$$s(t + \varepsilon) \approx s(t) + \varepsilon \cdot u(s(t), i(t)).$$

Choosing a specific ε as a time increment, we can define a discrete time, deterministic system by

$$\mathcal{E}_\varepsilon u(s, i) = s + \varepsilon \cdot u(s, i). \tag{4.6}$$

This simple method of approximating the solution of a differential equation is called the Euler method. We can see the Euler method as a change of doctrine from a differential doctrine to a deterministic doctrine.

Proposition 4.5.12. For any $\varepsilon > 0$ The Euler method gives rise to a change of doctrine

$$\mathscr{E}_\varepsilon : \mathbb{E}\text{UC}_{|\mathbf{Aff}} \to \mathbb{D}\text{ET}_{\mathbf{Euc}}.$$

This is given by

$$((\text{id}, \text{id}), \phi) : (\mathbf{Ctx}_{|\mathbf{Aff}} : \mathbf{Aff}^{\text{op}} \to \mathbf{Cat}, T) \to (\mathbf{Ctx}_{|\mathbf{Aff}} : \mathbf{Aff}^{\text{op}} \to \mathbf{Cat}, \mathbb{R}^n \mapsto \mathbb{R}^n)$$

where $\phi : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is defined by

$$\phi(s, v) = s + \varepsilon \cdot v.$$

Proof. We note, first of all, that composing with ϕ gives us the correct formula for the Euler approximation. Explicitly,

$$\phi \circ u(s, i) = s + \varepsilon \cdot u(s, i),$$

which was the definition for $\mathscr{E}_\varepsilon u$ in **??**.

All that we need to show is that ϕ is a transformation of sections. This means that the following square commutes for any *affine* $f : \mathbb{R}^n \to \mathbb{R}^m$:

$$
\begin{array}{ccc}
\begin{pmatrix} \mathbb{R}^n \\ \mathbb{R}^n \end{pmatrix} & \xRightarrow{\begin{pmatrix} \pi_2 \mathbin{\raisebox{0.3ex}{\tiny$\circ$}} f \\ f \end{pmatrix}} & \begin{pmatrix} \mathbb{R}^m \\ \mathbb{R}^m \end{pmatrix} \\[2em]
\begin{pmatrix} \phi \\ \text{id} \end{pmatrix} \Big\Updownarrow \Big\uparrow & & \Big\Downarrow \Big\uparrow \begin{pmatrix} \phi \\ \text{id} \end{pmatrix} \\[2em]
\begin{pmatrix} \mathbb{R}^n \\ \mathbb{R}^n \end{pmatrix} & \xRightarrow[\begin{pmatrix} Tf \\ f \end{pmatrix}]{} & \begin{pmatrix} \mathbb{R}^m \\ \mathbb{R}^m \end{pmatrix}
\end{array}
$$

The bottom component of this square commutes trivially. The top component comes down to the equation

$$f(s + \varepsilon \cdot v) = f(s) + \varepsilon T f(s, v) \tag{4.7}$$

which says that incrementing s by ε in the v direction in f is the same as incrementing $f(s)$ by the ε times the directional derivative of f in the v direction. This is true for affine functions; even more, it characterizes affine functions, so that we see that we must assume that f is affine for this square to commute. $\qquad \square$

Remark 4.5.13. It would be very interesting to have a theory which allowed us to speak of "approximate" changes of doctrine. If we plug a function $f : \mathbb{R}^n \to \mathbb{R}^m$ into the above formulas for the Euler method, then we find that **??** only holds up to $O(\varepsilon^2)$. For affine functions, this means that it does hold, which is why we restrict to affine functions. But it would be interesting to have a theory which could account for how

these approximate equalities affected the various compositionality results all the way down.

In the upcoming **??**, we will see what knowing that the Euler method is a change of doctrine lets us conclude about the behaviors and compositionality of Euler method approximations.

Considering doctrines together with their changes gives us a category **Doctrine**.

Definition 4.5.14. The category **Doctrine** has as objects the dynamical systems doctrines and as morphisms the changes of doctrine.

If $((F_1, \overline{F}_1), \phi_1) : (\mathcal{A}_1, T_1) \to (\mathcal{A}_2, T_2)$ and $((F_2, \overline{F}_2), \phi_2) : (\mathcal{A}_2, T_2) \to (\mathcal{A}_3, T_3)$ are changes of doctrine, then their composite is defined to be

$$((F_1, \overline{F}_1), \phi_1) \, \mathbin{\mathring{,}} \, ((F_2, \overline{F}_2), \phi_2) := ((F_1, \overline{F}_1) \, \mathbin{\mathring{,}} \, (F_2, \overline{F}_2), \phi_1 * \phi_2)$$

where $\phi_1 * \phi_2$ is the transformation of sections given by

$$(\phi_1 * \phi_2)_C := \overline{F_2 F_1} T_1 C \xrightarrow{\overline{F}_2 \phi_1} \overline{F_2} T_2 F_1 C \xrightarrow{(\phi_2)_{F_1 C}} T_3 F_2 F_1 C.$$

In terms of natural transformations (see **??**), this is the diagram

---

4.5.15   Functoriality

We use changes of doctrines to turn a system of one sort into a system of another sort. We sketched how this process goes above, but for good measure let's revisit it.

Definition 4.5.16. Let $\mathbb{F} = ((F, \overline{F}), \phi) : (\mathcal{A}, T_1) \to (\mathcal{B}, T_2)$ be a change of doctrine, and let

$$S = \begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} : \begin{pmatrix} T_1 \text{State}_S \\ \text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \text{In}_S \\ \text{Out}_S \end{pmatrix}$$

be a $(\mathcal{A}, T_1)$-system. Then we have a $(\mathcal{B}, T_2)$-system $\mathbb{F}S$ defined to be the composite

$$\begin{pmatrix} \phi \\ \text{id} \end{pmatrix} \, \mathbin{\mathring{,}} \, \begin{pmatrix} \overline{F}\text{update}_S \\ F\text{expose}_S \end{pmatrix} : \begin{pmatrix} T_2 F\text{State}_S \\ F\text{State}_S \end{pmatrix} \leftrightarrows \begin{pmatrix} \overline{F}\text{In}_S \\ F\text{Out}_S \end{pmatrix}.$$

Explicitly, this system has update $\overline{F}\mathrm{update}_S \,\mathring{,}\, \phi$ and expose $F\mathrm{expose}_S$.

The goal of this section will be to provide a number of compositionality results concerning how changing the doctrine of a system relates to wiring systems together and to behaviors. Specifically, we will prove the following theorem:

Theorem 4.5.17. There is a functor

$$\mathbf{Sys} : \mathbf{Doctrine} \to \mathbf{DblIx}$$

sending a dynamical system doctrine $\mathbb{D}$ to the doubly indexed category $\mathbf{Sys}_{\mathbb{D}}$ (**??**) of systems in it.

This functor sends a change of doctrine $\mathbb{F} : \mathbb{D}_1 \to \mathbb{D}_2$ to the doubly indexed functor $\mathbf{Sys}(\mathbb{D}_1) \to \mathbf{Sys}(\mathbb{D}_2)$ which sends a $\mathbb{D}_1$-system S to the $\mathbb{D}_2$-system $\mathbb{F}S$ from **??**.

We will prove this theorem using the vertical slice construction. Recall that the doubly indexed category $\mathbf{Sys}(\mathbb{D})$ is the vertical slice construction of the section T considered as a double functor $(hT : h\mathcal{C} \to \mathbf{Arena}_{\mathbb{D}})$ (**??**). This means that if we can show that the assignment

$$(\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, T) \mapsto (hT : h\mathcal{C} \to \mathbf{Arena}_{(\mathcal{A},T)})$$

gives a functor $\mathbf{Doctrine} \to \mathbf{DblFun}$, then we can compose this with the vertical slice construction $\sigma : \mathbf{DblFun} \to \mathbf{DblIx}$. This is what we will focus on.

Lemma 4.5.18. The assignment

$$(\mathcal{A} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}, T) \mapsto (hT : h\mathcal{C} \to \mathbf{Arena}_{(\mathcal{A},T)})$$

gives a functor $\iota : \mathbf{Doctrine} \to \mathbf{DblFun}$. This functor sends a change of doctrines

$$\begin{array}{ccc} \mathcal{C} & \xrightarrow{\ F\ \left(\substack{\phi \\ \mathrm{id}}\right)\ } & \mathcal{D} \\ {\scriptstyle T_1}\downarrow & & \downarrow{\scriptstyle T_2} \\ \int^{C:\mathcal{C}}\mathcal{A}(C) & \xrightarrow[\left(\substack{\overline{F} \\ F}\right)]{} & \int^{C:\mathcal{D}}\mathcal{B}(C) \end{array} \tag{4.8}$$

to the morphism double functors

$$\begin{array}{ccc} h\mathcal{C} & \xrightarrow{\ F\ \left(\substack{\phi \\ \mathrm{id}}\right)\ } & h\mathcal{D} \\ {\scriptstyle hT_1}\downarrow & & \downarrow{\scriptstyle hT_2} \\ \mathbf{Arena}_{(\mathcal{A},T_1)} & \xrightarrow[\left(\substack{\overline{F} \\ F}\right)]{} & \mathbf{Arena}_{(\mathcal{B},T_2)} \end{array} \tag{4.9}$$

Proof. With all that we have set up, there is not too much to prove here. We first note that the the functoriality of the assignment $\mathcal{A} \mapsto \mathbf{Arena}_{\mathcal{A}}$ was proven in **??**. We only need to focus on the vertical transformation.

We need to show that $\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix}$ may be interpreted as a vertical transformation $hT_2 \circ F \to \begin{pmatrix} \overline{F} \\ F \end{pmatrix} \circ hT_1$. There is some subtlety here; in **??**, $\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix}$ is interpreted as a natural transformation taking place in the category of $\mathcal{B}$-charts, while in **??** we have a vertical transformation in the double category of arenas. But the vertical arrows in $\mathbf{Arena}_{(\mathcal{B},T_2)}$ are $\mathcal{B}$-lenses, not $\mathcal{B}$-charts. This explains the change of direction: we can consider the *chart* $\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} \overline{F} \\ F \end{pmatrix} \circ hT_1 \to hT_2 \circ F$ as a *lens* $\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix} : hT_2 \circ F \to \begin{pmatrix} \overline{F} \\ F \end{pmatrix} \circ hT_1$ by the duality between pure charts and lenses — those having an identity in the bottom component.

Let's describe precisely how $\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix}$ becomes a vertical transformation.

- For every $C \in h\mathcal{C}$, we have the lens $\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} T_2FC \\ FX \end{pmatrix} \to \begin{pmatrix} \overline{F}T_1C \\ FF \end{pmatrix}$.
- For every horizontal arrow $f : C' \to C$ in $h\mathcal{C}$ (which is to say, any map in $\mathcal{C}$), we have the square

$$
\begin{pmatrix} T_2FC' \\ FC' \end{pmatrix} \xRightarrow{\begin{pmatrix} T_2Ff \\ Ff \end{pmatrix}} \begin{pmatrix} T_2FC \\ FC \end{pmatrix}
$$

$$
\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix} \Big\Updownarrow \qquad\qquad \Updownarrow \begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix} \tag{4.10}
$$

$$
\begin{pmatrix} \overline{F}T_1C' \\ FC' \end{pmatrix} \xRightarrow{\begin{pmatrix} \overline{F}T_1f \\ Ff \end{pmatrix}} \begin{pmatrix} \overline{F}T_1C \\ FC \end{pmatrix}
$$

in $\mathbf{Arena}_{(\mathcal{B},T_2)}$. This is a square because both its top and bottom component squares commute; the bottom one trivially, and the top one by the defining **??** of ϕ.

We now check that this satisfies the laws of a vertical transformation. It is largely trivial, since the double categories are particularly simple (as double categories).

- (Vertical Naturality) By construction, the only vertical arrows in $h\mathcal{C}$ are identities, so there is nothing to check.
- (Horizontal Naturality) Since **Arena** is thin (**??**), any two squares with the same signature are equal, so there is nothing to check.
- (Horizontal Unity) This is true since all the functors involved in defining the top and bottom of the square **??** preserve identities.
- (Square Naturality) This again follows trivially by the thin-ness of **Arena**.

The proof of functoriality itself follows from a straightforward comparison of the two definitions of composition. They simply give the same formula on objects, and on horizontal morphisms we get squares of the same signature in a thin double category so there is nothing more to check.   □

We can therefore define

$$\textbf{Doctrine} \xrightarrow{\textbf{Sys}} \textbf{DblIx} := \textbf{Doctrine} \xrightarrow{\iota} \textbf{DblFun} \xrightarrow{\sigma} \textbf{DblIx}.$$

Let's take a moment to understand this definition in full. Suppose we have a change of doctrine $((F, \overline{F}), \phi) : (\mathcal{A}, T_1) \to (\mathcal{B}, T_2)$. Then $\iota((F, \overline{F}), \phi)$ is a map of double functors:

Then, by **??**, we get a doubly indexed functor

In this diagram, $\sigma\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix}$ is defined as follows:

- (**??**: **??**) For a $\mathbb{D}_1$-arena $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$, we have the functor

$$\sigma\begin{pmatrix} \phi \\ \mathrm{id} \end{pmatrix}^{\begin{pmatrix} A^- \\ A^+ \end{pmatrix}} : \textbf{Sys}_{\mathbb{D}_1}\begin{pmatrix} A^- \\ A^+ \end{pmatrix} \to \textbf{Sys}_{\mathbb{D}_2}\begin{pmatrix} \overline{F}A^- \\ FA^+ \end{pmatrix}.$$

given by sending a simulation $\psi : \mathsf{T} \to \mathsf{S}$ to the composite:

- (**??**: **??**) Since the doubly indexed functor is *taut*, for any lens $\left(\begin{smallmatrix} j^\sharp \\ j \end{smallmatrix} \right) : \left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)$ have a commuting square

$$\mathbf{Sys}_{\mathbb{D}_1}\left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \xrightarrow{\ \sigma\left(\begin{smallmatrix} \phi \\ \mathrm{id} \end{smallmatrix} \right)^{\left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right)}\ } \mathbf{Sys}_{\mathbb{D}_2}\left(\begin{smallmatrix} \overline{F}A^- \\ FA^+ \end{smallmatrix} \right)$$

$$\mathbf{Sys}_{\mathbb{D}_1}\left(\begin{smallmatrix} j^\sharp \\ j \end{smallmatrix} \right) \downarrow \qquad\qquad\qquad \downarrow \mathbf{Sys}_{\mathbb{D}_2}\left(\begin{smallmatrix} \overline{F}j^\sharp \\ Fj \end{smallmatrix} \right) \qquad\qquad (4.11)$$

$$\mathbf{Sys}_{\mathbb{D}_1}\left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right) \xrightarrow[\ \sigma\left(\begin{smallmatrix} \phi \\ \mathrm{id} \end{smallmatrix} \right)^{\left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)}\ ]{} \mathbf{Sys}_{\mathbb{D}_2}\left(\begin{smallmatrix} \overline{F}B^- \\ FB^+ \end{smallmatrix} \right)$$

This tells us that changing doctrine and then wiring together systems gives the same result as wiring together the systems first and then changing doctrine.

- (**??**: **??**) For a $\mathbb{D}_1$-chart $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right) : \left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)$, we have the square in **Cat**

$$\mathbf{Sys}_{\mathbb{D}_1}\left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \xrightarrow{\ \mathbf{Sys}\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)\ } \mathbf{Sys}_{\mathbb{D}_1}\left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)$$

$$\sigma\left(\begin{smallmatrix} \phi \\ \mathrm{id} \end{smallmatrix} \right)^{\left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right)} \Big\downarrow \qquad\quad \sigma\left(\begin{smallmatrix} \phi \\ \mathrm{id} \end{smallmatrix} \right)^{\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)} \qquad\quad \Big\downarrow \sigma\left(\begin{smallmatrix} \phi \\ \mathrm{id} \end{smallmatrix} \right)^{\left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)} \qquad (4.12)$$

$$\mathbf{Sys}_{\mathbb{D}_2}\left(\begin{smallmatrix} \overline{F}A^- \\ FA^+ \end{smallmatrix} \right) \xrightarrow[\ \mathbf{Sys}\left(\begin{smallmatrix} \overline{F}f_\flat \\ Ff \end{smallmatrix} \right)\ ]{} \mathbf{Sys}_{\mathbb{D}_2}\left(\begin{smallmatrix} \overline{F}B^- \\ FB^+ \end{smallmatrix} \right)$$

given by sending a $\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix} \right)$-behavior to the composite:

In other words, changes of doctrine preserve behavior in the sense that if ψ is a $\binom{f_\flat}{f}$-behavior then $F\psi$ is a $\binom{\bar{F}f_\flat}{f}$-behavior.

Example 4.5.19. For Euler approximatation

$$\mathcal{E}_\varepsilon : \mathbb{E}\text{UC}_{|\mathbf{Aff}} \to \mathbb{D}\text{ET}_{\mathbf{Euc}},$$

we get a doubly indexed functor

$$\left(\begin{pmatrix} \text{id} \\ \text{id} \end{pmatrix}, \sigma\begin{pmatrix} \phi \\ \text{id} \end{pmatrix}\right) : (\mathbb{E}\text{UC}_{|\mathbf{Aff}}) \to \mathbb{D}\text{ET}_{\mathbf{Euc}}$$

by the functoriality of **Sys**, where $\phi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is $\phi(p, v) = p + \varepsilon \cdot v$. Let's see it means for this to be a doubly indexed functor.

First, we have a functor

$$\sigma\begin{pmatrix} \phi \\ \text{id} \end{pmatrix}^{\binom{A^-}{A^+}} : \mathbf{Sys}_{\mathbb{E}\text{UC}_{\mathbf{Aff}}}\begin{pmatrix} A^- \\ A^+ \end{pmatrix} \to \mathbf{Sys}_{\mathbb{D}\text{ET}_{\mathbf{Euc}}}\begin{pmatrix} A^- \\ A^+ \end{pmatrix}.$$

This says that the Euler method preserves simulations. Second, we have a square like **??** which says that the Euler method preserves behaviors. However, we have to be careful here; the behaviors $\left(\varphi, \binom{f_\flat}{f}\right)$ which are preserved must have φ and $\binom{f_\flat}{f}$ in the appropriate double category of arenas, and here we had to restrict to those for which φ and f are affine maps so that **??** can hold. In other words, we see that the Euler method will preserve any *affine* behaviors of differential systems.

Most solutions to a system of differential equations — most trajectories — are not affine. This is to say that there aren't many behaviors of shape Time (from **??**). There is, however, an important class of affine solutions: steady states. These are the behaviors of shape Fix from **??**. So, in particular, we see that the Euler method preserves steady states.

That the Euler method preserves steady states is of course evident from the formula: if $u(s, i) = 0$, then $\mathcal{E}_\varepsilon u(s, i) = s + \varepsilon \cdot u(s, i) = s$. But we deduced this fact from our general definition of change of doctrine. This sort of analysis can tell us precisely which sorts of behaviors are preserved even in situtations where it may not be so obvious from looking at a defining formula.

The fact that **Sys**$(\mathcal{E}_\varepsilon)$ is a doubly indexed functor gives us a litany of compositionality checks. In particular, the commuting square (**??**: **??**) shows that if we are to wire together a family of differential systems and then approximate the result with the Euler method, we could have approximated each one and then wired together the result with the same wiring pattern.

4.6   Summary and Further Reading

In this chapter, we organized the systems in a doctrine into doubly indexed categories. While all the action takes place within the double category of arenas, the doubly indexed category of systems separates the systems from their interfaces and the behaviors from their charts. This let's us describe the various sorts of composition — of systems and of behaviors — and their relationships. We then saw how this construction varied as we changed doctrine.

There are other examples of changes of doctrines not covered here. For example, the Rutta-Kunge approximation can be seen as a change of doctrine; see [Ngo17].

Chapter 5

Behaviors of the whole from behaviors of the parts

5.1 Introduction

Let's take stock of where we've been so far in the past couple chapters.

- In Section 1.2.1, we saw the definitions of *deterministic systems* and *differential systems*.
- In **??**, we learned about *lenses*. We saw how systems can be interpreted as special sorts of lenses, and how we can wire together systems using lens composition.
- In **??** we learned about various sorts of *non-deteerministic systems*.
- In **??**, we learned about behaviors and *charts*. We saw how to define behaviors of systems using the notion of chart. Finally, we gave a formal definition of *dynamical system doctrine*, systematizing the various different notions — discrete, differential, non-deterministic — of dynamical systems.

The two sorts of composition we have seen so far — lens composition and chart composition — mirror the two sorts of composition at play in systems theory:

- We can compose *systems* by wiring them together. This uses lens composition.
- We can compose *behaviors* of systems like we compose functions. This uses chart composition.

In this chapter, we will see how these two sorts of composition interact. In short, behaviors of component systems give rise to behaviors of composite systems. The way that behaviors of the whole arise from behaviors of the parts is called *compositionality*. In this chapter, we will prove a general compositionality theorem concerning any sort of behavior in any doctrine.

But the behaviors of the component systems must be compatible with eachother: if a system S_1 has its parameters set by the exposed variables of a system S_2, then a behavior ϕ_1 of S_1 will be compatible with a behavior ϕ_2 of S_2 when ϕ_2 is a behavior for the parameters charted by the variables exposed by ϕ_1.

We will see that, remarkably, the way behaviors of composite systems arise from behaviors of component systems (including the constraints of compatibility) are described by a "matrix arithmetic for sets". From a lens we will construct a "matrix of sets"; multiplying the "vector of behaviors" of the component systems (indexed by their charts) by this matrix yields the vector of behaviors of the composite. We begin this chapter with a section explaining this idea in detail for steady states of deterministic systems.

We have in fact already proven most of the crucial lemmas we need for this result in **??**. In this chapter, we will organize these lemmas with the notion of a *doubly indexed category*. We will then construct *representable doubly indexed functors* which will organize the various facts concerning the compositionality of any sort of behavior in any doctrine.

5.2 Steady states compose according to the laws of matrix arithmetic

We have seen how we can compose systems, and we have seen how systems behave. We have seen a certain composition of behaviors, a form of transitivity that says that if we have a T-shaped behavior in S and a S-shaped behavior in U, then we get a T-shaped behavior in U. But what's the relationship between composing systems and composing their behaviors?

In this section we will give a taste by showing how steady states compose. Later, in **??**, we will see a very abstract theorem that generalizes what we do here for steady states in the deterministic doctrine to something that works for *any sort of behavior* in *any doctrine*. But in order for that abstract theorem to make sense, we should first see the concrete case of steady states in detail.

Recall that the chart of a steady state $s \in \mathsf{State_S}$ is the pair $\begin{pmatrix} i \\ o \end{pmatrix}$ with $o = \mathrm{expose_S}(s)$ and $\mathrm{update_S}(s, i) = s$. The set of all possible charts for steady states is therefore $\mathsf{In_S} \times \mathsf{Out_S}$, and for every chart $\begin{pmatrix} i \\ o \end{pmatrix}$ we have the set $\mathrm{Steady}_{SysS} \begin{pmatrix} i \\ o \end{pmatrix}$ of steady states for this chart.

We can see this function $\mathrm{Steady_S} : \mathsf{In_S} \times \mathsf{Out_S} \to \mathbf{Set}$ as a *matrix of sets* with $\mathrm{Steady_S} \begin{pmatrix} i \\ o \end{pmatrix}$ in the row i and column o. For example, consider system $\mathsf{S_1}$ of **??**:

$$\mathsf{S_1} := \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (5.1)$$
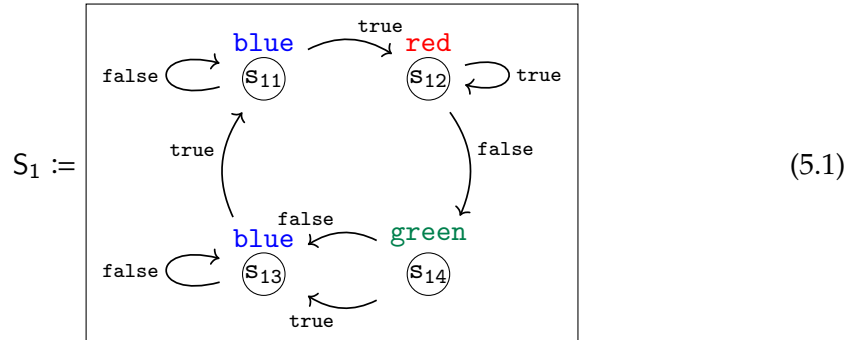
This has output value set Colors = {blue, red, green} and input parameter set Bool = {true, false}. Here is its (Colors × Bool) steady state matrix:

$$
\text{Steady}_{S_1} =
\begin{array}{c}
\quad\\[2pt]
\text{true}\\[40pt]
\text{false}
\end{array}
\left[
\begin{array}{ccc}
\varnothing & \left\{\begin{array}{c}\text{red}\\ \text{(s}_{12}\text{)} \circlearrowleft \text{true}\end{array}\right\} & \varnothing \\[30pt]
\left\{\text{false} \circlearrowright \begin{array}{c}\text{blue}\\ \text{(s}_{11}\text{)}\end{array},\ \text{false}\circlearrowright\begin{array}{c}\text{blue}\\ \text{(s}_{13}\text{)}\end{array}\right\} & \varnothing & \varnothing
\end{array}
\right]
$$

(5.2)

If we just want to know how many $\binom{i}{o}$-steady states there are, and not precisely which states they are, we can always take the cardinality of the sets in our matrix of sets to get a bona-fide matrix of numbers. Doing this to the above matrix gives us the matrix

$$
\begin{array}{c}
\\
\text{true}\\
\text{false}
\end{array}
\begin{array}{ccc}
\text{blue} & \text{red} & \text{green}
\end{array}
\left[
\begin{array}{ccc}
0 & 1 & 0 \\
2 & 0 & 0
\end{array}
\right]
$$

Now, let's take a look at system S_2 from the same exercise:

$$S_2 :=$$
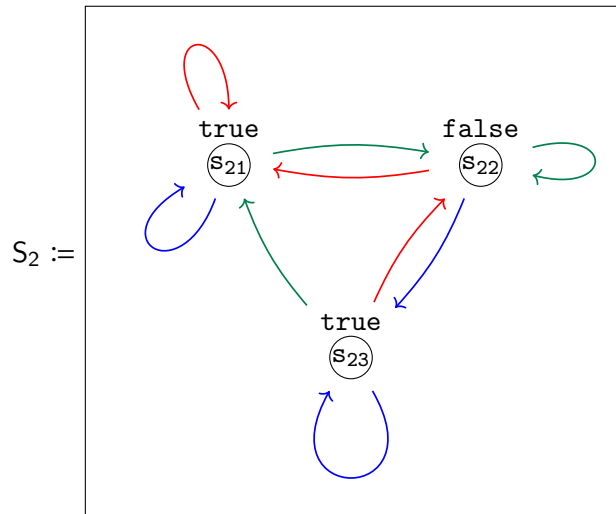
This has steady state matrix:

$$
\text{Steady}_{S_2} =
\begin{array}{c}
\\
\text{blue}\\[30pt]
\text{red}\\[30pt]
\text{green}
\end{array}
\begin{array}{cc}
\qquad\text{true} & \qquad\text{false}
\end{array}
\left[
\begin{array}{cc}
\left\{\circlearrowright\begin{array}{c}\text{true}\\ \text{(s}_{21}\text{)}\end{array},\ \circlearrowright\begin{array}{c}\text{true}\\ \text{(s}_{23}\text{)}\end{array}\right\} & \varnothing \\[30pt]
\left\{\circlearrowright\begin{array}{c}\text{true}\\ \text{(s}_{21}\text{)}\end{array}\right\} & \varnothing \\[30pt]
\varnothing & \left\{\circlearrowright\begin{array}{c}\text{false}\\ \text{(s}_{22}\text{)}\end{array}\right\}
\end{array}
\right]
$$

(5.3)

Or, again, if we just want to know how many steady states there are for each chart:

$$
\mathsf{Steady}_{S_2} = \begin{array}{c} \\ \texttt{blue} \\ \texttt{red} \\ \texttt{green} \end{array} \begin{array}{cc} \texttt{true} & \texttt{false} \\ \left[\begin{array}{cc} 2 & 0 \\ 1 & 0 \\ 0 & 1 \end{array} \right] \end{array}
$$

We can wire these systems together to get a system S:

$$
\mathsf{S} := \boxed{\ \fbox{S$_1$}\!-\!\fbox{S$_2$}\ }
$$
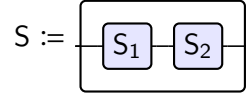
With just a bit of thought, we can find the steady states of this systems without fully calculating its dynamics. A state of S is a pair of states $s_1 \in \mathsf{States}_1$ and $s_2 \in \mathsf{States}_2$, so for it to be steady both its constituent states must be steady. So let $\left(\begin{smallmatrix} i \\ o \end{smallmatrix} \right) : \left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix} \right) \Rightarrow \left(\begin{smallmatrix} \mathsf{Bool} \\ \mathsf{Bool} \end{smallmatrix} \right)$ be a chart for S — a pair of booleans. We need s_1 and s_2 to both be steady, so in particular s_1 must be steady at the input i, and s_2 must expose o; but, most importantly, s_2 must then be steady at the input $\mathsf{expose}_{S_1}(s_1)$ which s_1 exposes.

So, to find the set of $\left(\begin{smallmatrix} \texttt{true} \\ \texttt{true} \end{smallmatrix} \right)$-steady states of S, we must a state of $\mathsf{S_1}$ which is steady for the input $\texttt{true}$ and then a steady state of $\mathsf{S_2}$ whose input is what that state outputs and whose output is $\texttt{true}$. There are three pieces of data here: the state of $\mathsf{S_1}$, the state of $\mathsf{S_2}$, and the intermediate value expose by the first state and input into the second state. We can therefore describe the set of $\left(\begin{smallmatrix} \texttt{true} \\ \texttt{true} \end{smallmatrix} \right)$-steady states of S like this:

$$
\mathsf{Steady}_S \begin{pmatrix} \texttt{true} \\ \texttt{true} \end{pmatrix} = \left\{ (m, s_1, s_2) \middle| s_1 \in \mathsf{Steady}_{S_1} \begin{pmatrix} \texttt{true} \\ m \end{pmatrix}, s_2 \in \mathsf{Steady}_{S_2} \begin{pmatrix} m \\ \texttt{true} \end{pmatrix} \right\}
$$

$$
= \sum_{m \in \mathsf{Colors}} \mathsf{Steady}_{S_1} \begin{pmatrix} \texttt{true} \\ m \end{pmatrix} \times \mathsf{Steady}_{S_2} \begin{pmatrix} m \\ \texttt{true} \end{pmatrix}.
$$

This formula looks very suspiciously like matrix multiplication! Indeed, if we multiply the matrices of numbers of steady states from $\mathsf{S_1}$ and $\mathsf{S_2}$, we get:

$$
\begin{array}{c} \\ \texttt{true} \\ \texttt{false} \end{array} \left[\begin{array}{ccc} 0 & 1 & 0 \\ 2 & 0 & 0 \end{array} \right] \begin{array}{c} \texttt{true} \quad \texttt{false} \\ \left[\begin{array}{cc} 2 & 0 \\ 1 & 0 \\ 0 & 1 \end{array} \right] \end{array} = \begin{array}{c} \\ \texttt{true} \\ \texttt{false} \end{array} \begin{array}{c} \texttt{true} \quad \texttt{false} \\ \left[\begin{array}{cc} 1 & 0 \\ 4 & 0 \end{array} \right] \end{array}
$$

which is the matrix of how many steady states S has! What's even more suspicious is that our wiring diagram for S looks a lot like the string diagram we would use to describe the multiplication of matrices:

$$
\boxed{\ \fbox{S$_1$}\!-\!\fbox{S$_2$}\ } \qquad\qquad \mathsf{Bool} \!-\!\boxed{\mathsf{Steady}_{S_1}}\!\overset{\mathsf{Colors}}{-}\!\boxed{\mathsf{Steady}_{S_2}}\!-\! \mathsf{Bool}
$$

This can't just be a coincidence. Luckily for our sanity, it isn't. In the remainder of this section, we will show how various things one can do with matrices — multiply them, trace them, Kronecker product them — can be done for matrices of sets, and how if your wiring diagram looks like its telling you to do that thing, then you can do that thing to the steady states of your internal systems to get the steady states of the whole wired system

Matrices of sets We'll be working with matrices of sets — now and in the coming section — quite a bit, so we should really nail them down. Matrices of sets work a lot like matrices of numbers, especially when the sets are finite; then they are very nearly the same thing as matrices of whole numbers. But the matrix arithmetic of infinite sets works just the same as with finite sets, so we'll do everything in that generality.[1]

Definition 5.2.1. Let A and B be two sets. $B \times A$ *matrix of sets* is a dependent set $M : B \times A \to$ **Set**. For $a \in A$ and $b \in B$, we write M_{ba} or $M_{(b,a)}$ for set indexed by a and b, and call this the (b, a)-entry of the matrix M.

We draw of matrix of sets with the following string diagram:

$$A -\boxed{M}- B$$

Remark 5.2.2. We can see a dependent set $X_- : A \to$ **Set** through the matrix of sets point of view as a *vector of sets*. This is because X_- is equivalently given by $X_- : A \times 1 \to$ **Set**, which we see is a $A \times 1$ matrix of sets. A $n \times 1$ matrix is equivalently a column vector.

Now we'll go through and define the basic operations of matrix arithmetic: mutliplication, Kronecker product (also known as the tensor product), and partial trace.

Definition 5.2.3. Given an $B \times A$ matrix of sets M and a $C \times B$ matrix of sets N, their *product* NM (or $M \times_B N$ for emphasis) is the $C \times A$ matrix of sets with entries

$$NM_{ca} = \sum_{b \in B} N_{cb} \times M_{ba}.$$

We draw the multiplication of matrices of sets with the following string diagram:

$$A -\boxed{M}\overset{B}{-}\boxed{N}- C$$

The identity matrix I_A is an $A \times A$ matrix with entries

$$I_{aa'} = \begin{cases} 1 & \text{if } a = a' \\ \emptyset & \text{if } a \neq a' \end{cases}.$$

---

[1]This will help us later when we deal with behaviors that have more complicated charts. For example, even finite systems can have infinitely many different trajectories, so we really need the infinite sets.

We draw the identity matrix as a string with no beads on it.

$$A \ \rule{3cm}{0.4pt} \ A$$

Exercise 5.2.4.   Multiplication of matrices of sets satisfies the usual properties of associativity and unity, but only up to isomorphism. Let M be a $B \times A$ matrix, N a $C \times B$ matrix, and L a $D \times C$ of sets. Show that

1. For all $a \in A$ and $d \in D$, $((LN)M)_{da} \cong (L(NM))_{da}$.
2. For all $a \in A$ and $b \in B$, $(MI_A)_{ba} \cong M_{ba} \cong (I_B M)_{ba}$.

◊

Remark 5.2.5. The isomorphisms you defined in **??** are *coherent*, much in the way the associativity and unity isomorphisms of a monoidal category are. Together, this means that there is a *bicategory* of sets and matrices of sets between them.

Definition 5.2.6. Let M be a $B \times A$ matrix and N a $C \times D$ matrix of sets. Their *Kronecker product* or *tensor product* $M \otimes N$ is a $(B \times C) \times (A \times D)$ matrix of sets with entries:

$$(M \otimes N)_{(b,c)(a,d)} = M_{ba} \times N_{cd}.$$

We draw the tensor product $M \otimes M$ of matrices as:

$$A -\boxed{M}- B$$
$$C -\boxed{N}- D$$

Finally, we need to define the partial trace of a matrix of sets.

Definition 5.2.7. Suppose that M is a $(A \times C) \times (A \times B)$ matrix of sets. Its *partial trace* $\mathrm{tr}_A M$ is a $C \times B$ matrix of sets with entries:

$$(\mathrm{tr}_A)M_{cb} = \sum_{a \in A} M_{(a,c)(a,b)}.$$

We draw the partial trace of a matrix of sets as:

$$B \ \rule{1.5cm}{0.4pt} \boxed{M} \ \rule{1.5cm}{0.4pt} \ C$$

Exercise 5.2.8.   Here's an important sanity check we should do about our string diagrams for matrices of sets. The following two diagrams should describe the same

matrix, even though they describe it in different ways:
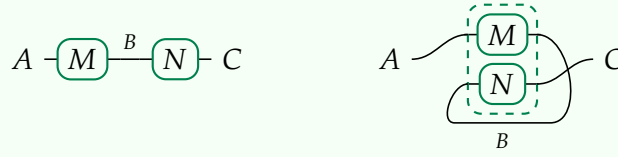
The diagram on the left says "multiply M and N", while the diagram on the right says "tensor M and N, and then partially trace them.". Show that these two diagrams do describe the same matrix:

$$NM \cong \text{tr}_B(M \otimes N).$$

Compare this to **??**, where we say that wiring an input of a system to an output of another can be seen as first taking their parallel product, and then forming a loop.   ◊

Steady states and matrix arithmetic   For the remainder of this section, we will show that we can calculate the steady state matrix of a wired together system in terms of its component system in a very simple way:

- First, take the steady state matrices of the component systems.
- Then consider the wiring diagram as a string diagram for multiplying, tensoring, and tracing matrices.
- Finally, finish by doing all those operations to the matrix.

In **??**, we will see that this method — or something a lot like it — works calculating the behaviors of a composite system out of the behaviors of its components, as long as the representative of that behavior exposes its entire state. That result will be nicely packaged in a beautiful categorical way: we'll make an *doubly indexed functor*.

But for now, let's just show that tensoring and partially tracing steady state matrices correponds to taking the parallel product and wiring an input to an output, respectively, of systems.

Proposition 5.2.9. Let S_1 and S_2 be systems. Then the steady state matrix of the parallel product $S_1 \otimes S_2$ is the tensor of their steady state matrices:

$$\text{Steady}_{S_1 \otimes S_2} \cong \text{Steady}_{S_1} \otimes \text{Steady}_{S_2}.$$

Proof. First, we note that these are both $(\text{Out}_{S_1} \times \text{Out}_{S_2}) \times (\text{In}_{S_1} \times \text{In}_{S_2})$-matrices of sets. Now, on a chart $\left(\begin{smallmatrix} (i_1,i_2) \\ (o_1,o_2) \end{smallmatrix} \right)$, a steady state in $S_1 \otimes S_2$ will be a pair $(s_1, s_2) \in \text{States}_1 \times \text{States}_2$ such that $\text{update}_{S_j}(s_j, i_j) = s_j$ and $\text{expose}_{S_j}(s_j) = o_j$ for $j = 1, 2$. In other words, its just a pair of steady states, one in S_1 and one in S_2. This is precisely the $\left(\begin{smallmatrix} (i_1,i_2) \\ (o_1,o_2) \end{smallmatrix} \right)$-entry of the right hand side above. □

Remark 5.2.10. **??** is our motiviation for using the symbol "$\otimes$" for the parallel product of systems.

Proposition 5.2.11. Let S be a system with $\mathsf{In}_S = A \times B$ and $\mathsf{Out}_S = A \times C$. Let S' be the system formed by wiring the A output into the A input of S:

$$S' := \boxed{\,\fbox{S}\,}$$
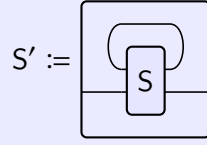
Then the steady state matrix of S' is given by partially tracing out A in the steady state matrix of S:

$$\mathsf{Steady}_{S'} = \overset{A}{\boxed{\mathsf{Steady}_S}} = \mathrm{tr}_A\,(\mathsf{Steady}_S)$$

Proof. Let's first see what a steady state of S' would be. Since S' is just a rewiring of S, it has the same states; so, a steady state s of S' is in particular a state of S. Now,

$$\mathrm{update}_{S'}(s,b) = \mathrm{update}_S(s,(\pi_1\mathrm{expose}_S(s),b))$$

by definition, so if $\mathrm{update}_{S'}(s,b) = s$, then $\mathrm{update}_S(s,(\pi_1\mathrm{expose}_S(s),b)) = s$. If also $\mathrm{expose}_{S'}(s) = c$ (so that s is a $\binom{b}{c}$-steady state of S'), then $\pi_2\mathrm{expose}_S(s) = \mathrm{expose}_{S'}(s) = c$ as well. In total then, starting with a $\binom{b}{c}$-steady state s of S', we get a $\binom{(\pi_1\mathrm{expose}_S(s),b)}{(\pi_1\mathrm{expose}_S(s),c)}$-steady state of S. That is, we have a function

$$s \mapsto (\pi_1\mathrm{expose}_S(s),s) : \mathsf{Steady}_{S'}\binom{b}{c} \to (\mathrm{tr}_A\mathsf{Steady}_S)\binom{b}{c}.$$

It remains to show that this function is a bijection. So, suppose we have a pair $(a,s) \in \mathrm{tr}_A\mathsf{Steady}_S\binom{b}{c}$ of an $a \in A$ and a $\binom{(a,b)}{(a,c)}$ steady state of S. Then

$$\mathrm{update}_{S'}(s,b) = \mathrm{update}_S(s,(\pi_1\mathrm{expose}_S(s),b))$$
$$= \mathrm{update}_S(s,(a,b)) \qquad\qquad \text{since } \mathrm{expose}_S(s) = (a,c).$$
$$= s \qquad\qquad\qquad \text{since } s \text{ is a } \binom{(a,b)}{(a,c)}\text{-steady state.}$$
$$\mathrm{expose}_{S'}(s) = \pi_2\mathrm{expose}_S(s) = c.$$

This shows that s is also a $\binom{b}{c}$ steady state of S', giving us a function $(a,s) \mapsto s : (\mathrm{tr}_A\mathsf{Steady}_S) \to \mathsf{Steady}_{S'}$. These two functions are plainly inverse. $\square$

We can summarize **??** in the following commutative diagram:

$$(5.4)$$
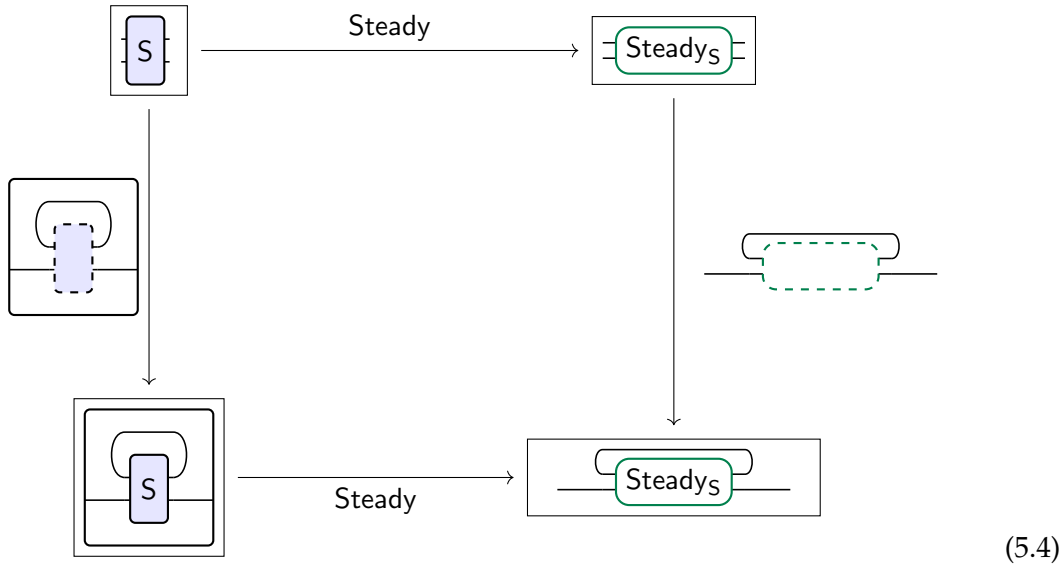
The horizontal maps take the steady states of a system, while the vertical map on the left wires together the system with that wiring diagram, and the vertical map on the right applies that transformation of the matrix. In the next section, we will see how this square can be intereprited as a naturality condition in a *doubly indexed functor*.

One thing to notice here is that taking the partial trace (the right vertical arrow in the diagram) is itself given by multiplying by a certain matrix.

Proposition 5.2.12. Let M be a $(A \times C) \times (A \times B)$ matrix of sets. Let Tr^A be the $(C \times B) \times ((A \times C) \times (A \times B))$ matrix of sets with entries:

$$\mathsf{Tr}^A A_{(c,b)((a,c'),(a',b'))} := \begin{cases} 1 & \text{if } a = a', b = b', \text{and } c = c'. \\ \emptyset & \text{otherwise.} \end{cases}$$

Then, considering M as a $((A \times C) \times (A \times B)) \times 1$ matrix of sets, taking its trace is given by multiplying by Tr^A:

$$\mathsf{tr}_A M \cong \mathsf{Tr}^A M$$

Proof. Let's calculate that matrix product on the right.

$$(\mathsf{Tr}^A M)_{(c,b)} = \sum_{((a,c'),(a',b')) \in (A \times C) \times (A \times B)} \mathsf{Tr}^A_{(c,b)((a,c'),(a',b'))} \times M_{(a,c')(a',b')}$$

Now, since $\mathsf{Tr}^A_{(c,b)((a,c'),(a',b'))}$ is a one element set (if $a = a'$, $c = c'$, and $b = b'$) and is empty otherwise, the inner expression has the elements of $M_{(a,c')(a',b')}$ if and only if $a = a'$, $b = b'$, and $c = c'$ and is otherwise empty. So, we conclude that

$$\sum_{((a,c'),(a',b')) \in (A \times C) \times (A \times B)} \mathsf{Tr}^A_{(c,b)((a,c'),(a',b'))} \times M_{(a,c')(a',b')} \cong M_{(a,c)(a,b)}.$$

As desired.   □

5.3   The big theorem: representable doubly indexed functors

We have now introduced all the characters in our play: the double categories of arenas and matrices, and doubly indexed categories of systems and vectors. In this section, we will put the plot in motion.

In **??**, we saw that the steady states of dynamical systems with interface $\binom{I}{O}$ compose like an $I \times O$ matrix. We proved a few propositions to this effect, namely **??** and **??**, but we didn't precisely mark out the scope of these results, or describe the full range of laws that are satisfied.

In this section, we will generalize the results of that section to *all behaviors* of systems, not just steady states. We will precisely state all the ways that behaviors can be composed by systems, and we will give a condition on the kinds of behaviors for which we can calculate the behavior of a wired together system entirely from the behavior of its component systems. All of this will be organized into a *doubly indexed functor* Behave$_\mathsf{T}$: **Sys** → **Vec** which will send a system S to its set of T-shaped behaviors.

In fact, our definition of Behave$_\mathsf{T}$ will be entirely abstract; it will work for almost any doubly indexed category $\mathscr{A} : \mathscr{D} \to$ **Cat** (there is a small condition on the indexing double category $\mathscr{D}$). Behave$_\mathsf{T}$ will be a *representable* doubly indexed category. Before going on to construct representable doubly indexed categories, let's take a minute to refresh ourselves on what representable functors our for categories. The essential idea is the same.

If $\mathcal{C}$ is a category and T an object of $\mathcal{C}$, then we can see maps $f : T \to X$ as "figures of shape T in X". It is often the case that we have some other way of talking about figures of shape T in X in terms that don't mention T — in this case we say that T *represents* figures of shape T. This phenomenon is very widespread, so let's give a number of examples:

- Suppose that $\mathcal{C}$ is the category of sets, and $T = 1$ is a one element set. Then a map $f : T \to X$ uniquely picks out an element of X. We see that T has the shape of a single element, and a map from T to X is a thing in X whose shape is an element; that is, an element of X. We can say that 1 represents elements.
- Suppose that $\mathcal{C}$ is the category of sets, but now that $T = 2$ is a two-element set. A two-element set is an abstract pair of elements, and a map $f : T \to X$ now picks out a pair of elements in X. We can say that 2 represents pairs.
- Suppose that $\mathcal{C}$ is the category of simple, undirected graphs — that is, sets X equipped with an irreflexive relation $E_X \subseteq X \times X$ telling us which two elements are connected by an edge. The maps of this category need to preserve edges. If T is the graph consisting of a single edge (formally, $T = 2$ with $(0, 1) \in E_T$ being the only edge), then a map $f : T \to X$ must pick out a pair of points in X with an edge between them. In other words, maps $T \to X$ are edges in X. So we may

say that T represents edges.

- Suppose that $\mathcal{C}$ is the category of rings, and let $T = \mathbb{Z}[x, y]$ be the ring of polynomials in two variables. A ring homomoprhism $f : T \to X$ can send x to any element $f(x)$ and similarly y to any element $f(y)$; once it's done that, the value of f on any polynomial in x and y must be given by

$$f\left(\sum a_{ij} x^i y^j\right) = \sum a_{ij} f(x)^i f(y)^j.$$

  since f is presumed to be a ring homomorphism. Actually, there is one constraint on $f(x)$ and $f(y)$ for this to work; since $xy = yx$ as polynomials, we must have $f(x)f(y) = f(y)f(x)$. Therefore, we see that $\mathbb{Z}[x, y]$ represents pairs of elements which commute in the category of rings.

- As we saw in **??**, all sorts of behaviors of systems — trajectories, periodic orbits, steady states, etc — are represented by simple systems in the category of systems and behaviors between them.

We could continue endlessly. The idea of representability is fundamental in category theory. Let's make it a little more explicit exactly what it means for T to represent something.

If T is an object of $\mathcal{C}$, then for any object X of $\mathcal{C}$ we get a *set* $\mathcal{C}(T, X)$ of all maps from T to X in $\mathcal{C}$. If $g : X \to Y$ is a map in $\mathcal{C}$, then for any $f : T \to X$ we get a map $f \, \mathring{\,}\, g : T \to Y$; in other words, for $g : X \to Y$ we get a map $\mathcal{C}(T, X) \xrightarrow{-\mathring{\,}g} \mathcal{C}(T, Y)$ given by post-composing with g. This gives us a *functor* $\mathcal{C}(T, -) : \mathcal{C} \to \mathbf{Set}$. This is a representable functor.

The idea of this section is to use the fact that behaviors are represented by simple systems to prove a compositionality result. This compositionality result is packaged up into a doubly indexed functor, and we will construct it as a representable doubly indexed functor. Instead of going from a category to the category of sets as representable functors do, our representable doubly indexed functors will go from a doubly indexed category (satisfying a little condition) to the doubly indexed category **Vec** of vectors of sets.

5.3.1 Turning lenses into matrices: Representable double Functors

In **??**, we saw how we could re-interpret a wiring diagram as a schematic for multiplying, tensoring, and tracing matrices. At the very end, in **??**, we saw that we can take the trace $\mathrm{tr}_A M$ of a $(A \times C) \times (A \times B)$-matrix M by considering it as a $(A \times C) \times (B \times C)$ length vector and then multiplying it by a big but very sparse $(C \times B) \times ((A \times C) \times (B \times C))$-matrix Tr^A. Taking the trace of a matrix corresponded to the wiring diagram
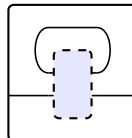
In this section, we will see a general formula for taking an arbitrary lens and turning it into a matrix. Mutliplying by the matrix will then correspond to wiring according to that lens.

This process of turning a lens into a matrix will give us a functor **Lens** $\to$ **Matrix** from the category of lenses to the category of matrices of sets. We'll start by exploring this functor in the deterministic doctrine; then we will abstract and find that the same argument works in any doctrine.

The resulting matrices will have entries that are either 1 or $\emptyset$; we can think of this as telling us whether (1) or not ($\emptyset$) the two charts are to be wired together. As we saw in **??**, we can see a square in the double category of arenas as telling us whether how a chart can be wired together along a lens into another chart. Therefore, we will take the entries of our matrices to be the sets of appropriate squares in arena — but there is either a single square (if the appropriate equations hold) or no square (if they don't), so we will end up with a matrix whose entries either have a single element or are empty.

Proposition 5.3.2. For any arena in the deterministic doctrine $\begin{pmatrix} I \\ O \end{pmatrix}$, there is a functor

$$\mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right) : \mathbf{Lens}_{\mathrm{DET}} \to \mathbf{Matrix}$$

from the category of lenses to the category of matrices of sets which sends an arena $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ to the set $\mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} A^- \\ A^+ \end{pmatrix}\right)$ of charts from $\begin{pmatrix} I \\ O \end{pmatrix}$ to $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$, and which sends a lens $\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \leftrightarrows \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ to the $\mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} B^- \\ B^+ \end{pmatrix}\right) \times \mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} A^- \\ A \end{pmatrix}\right)$ matrix of sets

$$\mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} w^\sharp \\ w \end{pmatrix}\right) : \mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} B^- \\ B^+ \end{pmatrix}\right) \times \mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} A^- \\ A \end{pmatrix}\right) \to \mathbf{Set}$$

$$\left(\begin{pmatrix} f_\flat \\ f \end{pmatrix}, \begin{pmatrix} g_\flat \\ g \end{pmatrix}\right) \mapsto \left\{ \text{The set of squares} \quad \begin{array}{c} \begin{pmatrix} I \\ O \end{pmatrix} \xRightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \\ \| \| \| \quad\quad \downarrow\uparrow \begin{pmatrix} w^\sharp \\ w \end{pmatrix} \quad \text{in } \mathbf{Arena}_{\mathrm{DET}} \\ \begin{pmatrix} I \\ O \end{pmatrix} \xRightarrow[\begin{pmatrix} g^\sharp \\ g \end{pmatrix}]{} \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \end{array} \right\}$$

$$= \begin{cases} 1 & \text{if } \begin{cases} g(o) = w(f(o)) & \text{for all } o \in O, \\ f_\flat(i, o) = w^\sharp(f(o), g_\flat(i, o)) & \text{for all } i \in I \text{ and } o \in O. \end{cases} \\ \emptyset & \text{otherwise} \end{cases}$$

Proof. By vertical composition of squares,

$$
\begin{pmatrix} I \\ O \end{pmatrix} \xrightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} A^- \\ A^+ \end{pmatrix}
$$

there is always a map from the composite of two of these matrices to the matrix described by the composite. It is not, however, obvious that this map is a bijection — which is what we need to prove functoriality.

Suppose we have a square as on the left hand side; let's see that we can factor it into two squares as on the right hand side. We need to construct the middle chart $\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} I \\ O \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ from $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$ and $\begin{pmatrix} h_\flat \\ h \end{pmatrix}$. For the bottom of top square to commute, we see that g must equal $w \circ f$, so we can define $g := w \circ f$. On the other hand, for the top of the bottom square to commute, we must have that $g_\flat(i, o) = v^\sharp(g(o), h_\flat(i, o))$; again, we can take this as a definition. It remains to show that the other half of each square commutes. For the top of the top square to commute means that

$$
f_\flat(i, o) = w^\sharp(f(o), g_\flat(i, o))
$$

which we can see holds by

$$
\begin{aligned}
w^\sharp(f(o), g_\flat(i, o)) &= w^\sharp(f(o), v^\sharp(g(o), h_\flat(i, o))) \\
&= w^\sharp(f(o), v^\sharp(wf(o), h_\flat(i, o))) \\
&= f_\flat(i, o) \qquad\qquad \text{by the commutativity of the square on the right.}
\end{aligned}
$$

On the other hand, to show that the bottom of the bottoms square commutes, we need that $h = v \circ g$. But by hypothesis, $h = v \circ w \circ f$, and we defined $g = w \circ f$. □

Example 5.3.3. Let's see what happens when we take the functor $\mathbf{Chart}_{\mathbb{D}\text{ET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right)$ for the arena $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$. A chart $\begin{pmatrix} a^- \\ a^+ \end{pmatrix} : \begin{pmatrix} 1 \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ is just a pair of elements $a^- \in A^-$ and $a^+ \in A^+$, so

$$
\mathbf{Chart}_{\mathbb{D}\text{ET}}\left(\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} A^- \\ A^+ \end{pmatrix}\right) = A^- \times A^+.
$$

Now, if we have a lens $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) : \left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)$, we have a square

$$
\begin{array}{ccc}
\left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix} \right) & \xrightarrow{\left(\begin{smallmatrix} a^- \\ a^+ \end{smallmatrix} \right)} & \left(\begin{smallmatrix} A^- \\ A^+ \end{smallmatrix} \right) \\[1em]
\big\| & & \Big\updownarrow\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \\[1em]
\left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix} \right) & \xrightarrow[\left(\begin{smallmatrix} b^- \\ b^+ \end{smallmatrix} \right)]{} & \left(\begin{smallmatrix} B^- \\ B^+ \end{smallmatrix} \right)
\end{array}
$$

if and only if $w(a^+) = b^+$ and $w^\sharp(a^+, b^-) = a^-$. Thinking of $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right)$ as a wiring diagram, this would mean that b^+ is that part of a^+ which is passed forward on the outgoing wires, and a^- is the inner input which comes from the inner output a^+ and outer input b^-.

To take a concrete example, suppose that $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right)$ were the following wiring diagarm:
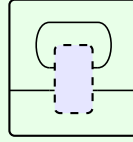
That is, let's take $A^+ = X \times Y$ and $A^- = X \times Z$, and $B^+ = Y$ and $B^- = Z$, and

$$w(x, y) = y$$
$$w^\sharp((x, y), z) = (x, z).$$

Using the definition above, we can calculate the resulting matrix $\mathbf{Chart}_{\mathbb{D}\text{ET}}\left(\left(\begin{smallmatrix} I \\ O \end{smallmatrix} \right), \left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \right)$ as having $(((x, y), (x', z)), (y', z'))$-entry
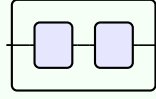
$$
\begin{cases}
1 & \text{if } w(x, y) = y' \text{ and } w^\sharp((x, y), z) = (x', z') \\
0 & \text{otherwise.}
\end{cases}
$$

or, by the definition of $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right)$,

$$
\begin{cases}
1 & \text{if } x = x', y = y', \text{ and } z = z' \\
0 & \text{otherwise.}
\end{cases}
$$

which was the definition of Tr^X given in ??!

Exercise 5.3.4. Let $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right) : \left(\begin{smallmatrix} A \times B \\ B \times C \end{smallmatrix}\right) \leftrightarrows \left(\begin{smallmatrix} A \\ C \end{smallmatrix}\right)$ be the wiring diagram

Calculate the entries of the matrix $\mathbf{Chart}_{\mathbb{D}\mathrm{ET}} \left(\left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right) \right)$. ◊

By the functoriality of **??**, we can calculate the matrix of a big wiring diagram by expressing it in terms of a series of traces, and mutliplying the resulting matrices together. This means that the process of multiplying, tensoring, and tracing matrices described by a wiring diagram is well described by the matrix we constructed in **??**, since we already know that it interprets the basic wiring diagrams correctly.

But we are also interested in charts, since we have to chart out our behaviors. So we will give a *double functor* $\mathbf{Arena}_{\mathbb{D}\mathrm{ET}} \to \mathbf{Matrix}$ that tells us not only how to turn a lens into a matrix, but also how this operation interacts with charts. This is an example of a *representable* double functor.

We will first define the double functor $\mathbf{Chart}_{\mathbb{D}\mathrm{ET}} \left(\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right), - \right) : \mathbf{Arena}_{\mathbb{D}\mathrm{ET}} \to \mathbf{Matrix}$ represented by an arena $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ explicitly. Then we will see how this argument can be abstracted to a double category which satisfies a horizontal factorization property.

Proposition 5.3.5. There is a double functor

$$\mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right) : \mathbf{Arena} \to \mathbf{Matrix}$$

which acts in the following way:

- An arena $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ gets sent to the set $\mathbf{Chart}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} A^- \\ A^+ \end{pmatrix}\right)$ of charts from $\begin{pmatrix} I \\ O \end{pmatrix}$ to $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$.

- The vertical functor is $\mathbf{Chart}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right) : \mathbf{Lens} \to \mathbf{Matrix}$ from **??**.

- The horizontal functor is the representable functor $\mathbf{Arena}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right) : \mathbf{Arena} \to$
  Set which acts on a chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}$ by post-composition.

- To a square

$$\beta = \begin{pmatrix} j^\sharp \\ j \end{pmatrix} \downarrow\uparrow \quad\quad \begin{array}{c} \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \xrightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\[10pt] \downarrow\uparrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix} \\[10pt] \begin{pmatrix} C^- \\ C^+ \end{pmatrix} \xrightarrow[\begin{pmatrix} g^\sharp \\ g \end{pmatrix}]{} \begin{pmatrix} D^- \\ D^+ \end{pmatrix} \end{array}$$

  in the double category of arenas, we give the square

$$\mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \beta\right) = \mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} j^\sharp \\ j \end{pmatrix}\right) \downarrow \begin{array}{c} \mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} A^- \\ A^+ \end{pmatrix}\right) \xrightarrow{\mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} f_\flat \\ f \end{pmatrix}\right)} \mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} B^- \\ B^+ \end{pmatrix}\right) \\[12pt] \downarrow \mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} k^\sharp \\ k \end{pmatrix}\right) \\[12pt] \mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} C^- \\ C^+ \end{pmatrix}\right) \xrightarrow[\mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} g_\flat \\ g_\flat \end{pmatrix}\right)]{} \mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \begin{pmatrix} D^- \\ D^+ \end{pmatrix}\right) \end{array}$$

  in the double category of matrices defined by horizontal composition of squares in $\mathbf{Arena}_{\mathrm{DET}}$ (remember that the entries of these matrices are sets of squares in $\mathbf{Arena}_{\mathrm{DET}}$, even though that means they either have a single element or no elements).

$$\mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, \beta\right)(\alpha) = \alpha \mid \beta.$$

Proof. We can write the double functor $\mathbf{Arena}_{\mathrm{DET}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right)$ entirely in terms of the double category $\mathbf{Arena}_{\mathrm{DET}}$:

- It sends an arena $\begin{pmatrix} A^- \\ A^+ \end{pmatrix}$ to the set of charts (horizontal maps) $\begin{pmatrix} f_\flat \\ f \end{pmatrix} : \begin{pmatrix} I \\ O \end{pmatrix} \Rightarrow \begin{pmatrix} A^- \\ A^+ \end{pmatrix}$.

- It sends a chart $\begin{pmatrix} g_\flat \\ g \end{pmatrix}$ to the map $\begin{pmatrix} f_\flat \\ f \end{pmatrix} \mapsto \begin{pmatrix} f_\flat \\ f \end{pmatrix} \bigg\| \begin{pmatrix} g_\flat \\ g \end{pmatrix}$.

- It sends a lens $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$ to the set of squares $\beta : \begin{pmatrix} I \\ O \end{pmatrix} \to \begin{pmatrix} w^\sharp \\ w \end{pmatrix}$, indexed by their top

and bottom boundaries.

- It sends a square α to the map given by horizontal compostion $\beta \mapsto \beta \mid \alpha$.

We can see that this double functor (let's call it F, for short) takes seriously the idea that "squares are charts between lenses" from **??**. From this description, and the functoriality of **??**, we can see that the assignments above satisfy the double functor laws.

- Horizontal functoriality follows from horizontal associativity in $\mathbf{Arena}_{\mathbb{D}\mathrm{ET}}$:

$$F(\alpha \mid \beta)(\gamma) = \gamma \mid (\alpha \mid \beta) = (\gamma \mid \alpha) \mid \beta = F(\alpha) \mid F(\beta)(\gamma).$$

- Vertical functoriality follows straight from the definitions:

$$F\left(\frac{\alpha}{\beta}\right)(\_, \gamma, \delta) = (\_, \gamma \mid \alpha, \delta \mid \beta) = \frac{F(\alpha)(\gamma)}{F(\beta)(\delta)}.$$

- It's pretty straightforward to check that identities get sent to identities.

$\square$

This construction is an example of a more general notion of *representable double functor*. Using the general notion, we can construct a similar double functor

$$\mathbf{Arena}_{\mathbb{D}}\left(\begin{pmatrix} I \\ O \end{pmatrix}, -\right) : \mathbf{Arena}_{\mathbb{D}} \to \mathbf{Matrix}$$

for any doctrine $\mathbb{D}$. Unlike for categories, not all objects in all double categories admit representable double functors[2]. There is a small condition on an object: the *horizontal factor condition*.

Definition 5.3.6. Let $\mathcal{D}$ be a double category. An object D of $\mathcal{D}$ satisfies the *horizontal factor condition* when for any square

$$
\begin{array}{ccc}
D & \xrightarrow{f_1} & X_1 \\
\| & & \downarrow{k_1} \\
 & \alpha & X_2 \\
\| & & \downarrow{k_2} \\
D & \xrightarrow[f_3]{} & X_3
\end{array}
$$

there is a unique triple of a horizontal $f_2 : D \to X_2$ and squares $\alpha_1 : D \Rightarrow k_1$ and

---

[2]Any object in a double category does admit a representable *lax* double functor, but we won't need any of these and so won't introduce this notion.

$\alpha_2 : D \Rightarrow k_2$ so that

$$
\begin{array}{ccc}
\begin{array}{ccc}
D & \xrightarrow{f_1} & X_1 \\
\Vert & & \downarrow{k_1} \\
& \alpha & X_2 \\
& & \downarrow{k_2} \\
D & \xrightarrow{f_3} & X_3
\end{array}
& = &
\begin{array}{ccc}
D & \xrightarrow{f_1} & X_1 \\
\Vert & \alpha_1 & \downarrow{k_1} \\
D & \xrightarrow{f_2} & X_2 \\
\Vert & \alpha_2 & \downarrow{k_2} \\
D & \xrightarrow{f_3} & X_3
\end{array}
\end{array}
$$

We say that $\mathcal{D}$ is *spanlike* if every D satisfies the horizontal factor condition.

Theorem 5.3.7. Let $\mathcal{D}$ be a double category and let D be an object of $\mathcal{D}$ satisfying the horizontal factor condition. Then there is a *representable double functor* $\mathcal{D}(D,-) : \mathcal{D} \to$ **Matrix** defined as follows:

- For an object X, $\mathcal{D}(D,X)$ is the set of horizontal arrows $D \to X$.
- For a horizontal $g : X \to Y$, $\mathcal{D}(D,g) : \mathcal{D}(D,X) \to \mathcal{D}(D,Y)$ is given by post-composition with g: $f \mapsto f \mid g$.
- For a vertical $k : X \to Y$, we get the matrix of sets $\mathcal{D}(D,k) : \mathcal{D}(D,X) \times \mathcal{D}(D,Y) \to$ **Set** given by

$$
(f_1, f_2) \mapsto \left\{ \text{The set of squares} \quad
\begin{array}{ccc}
D & \xrightarrow{f_1} & Y \\
\Vert & \alpha & \downarrow{k} \\
D & \xrightarrow{f_2} & Y
\end{array}
\right\}
$$

- For any square

$$
\begin{array}{ccc}
X_1 & \xrightarrow{g_1} & Y_1 \\
\downarrow{k_1} & \alpha & \downarrow{k_2} \\
Y_1 & \xrightarrow{g_2} & Y_2
\end{array}
$$

we define $\mathcal{D}(D,\beta)$ to be the map of matrices given by post-composing with β. That is,

$$
\mathcal{D}(D,\beta)(\alpha) = \alpha \mid \beta.
$$

Proof. We will show that this is a double functor. The horizontal component is functorial since it is the functor $h\mathcal{D} \to$ **Set** represented by D.

For vertical functoriality, we need to show that

$$
\mathcal{D}\left(D, \frac{k_1}{k_2}\right) \cong \frac{\mathcal{D}(D,k_1)}{\mathcal{D}(D,k_2)}
$$

for vertical arrows $k_1 : X_1 \to X_2$ and $k_2 : X_2 \to X_3$. There is always a map

$$\frac{\mathcal{D}(D, k_1)}{\mathcal{D}(D, k_2)} \to \mathcal{D}\left(D, \frac{k_1}{k_2}\right)$$

given by taking two squares and composing them. That this map is a bijection is a restatement of the horizontal factor condition which we assumed that D satisfied. The right hand side is the $\mathcal{D}(D, X_1) \times \mathcal{D}(D, X_3)$-matrix of sets which between f_1 and f_3 is the set

$$\sum_{f_2 \in \mathcal{D}(D, X_2)} \mathcal{D}(D, k_1)_{f_1, f_2} \times \mathcal{D}(D, k_2)_{f_2, f_3}.$$

So to say that for any $\alpha \in \mathcal{D}\left(D, \frac{k_1}{k_2}\right)$ there exists a unique triple $(f_2, \alpha_1, \alpha_2)$ with $\alpha = \frac{\alpha_1}{\alpha_2}$ is precisely to say that the map which composes two squares α_1 and α_2 into $\frac{\alpha_1}{\alpha_2}$ is a bijection.

We then need to check vertical and horizontal functoriality for squares. Horizontal functoriality of squares comes down to associativity of horizontal composition, and vertical functoriality of squares comes down to the interchange law. $\square$

?? gives us **??** as a special case since the double category $\mathbf{Arena}_\mathbb{D}$ of arenas in any doctrine $\mathbb{D}$ is *spanlike* — every arena $\binom{I}{O}$ satisfies the horizontal factor condition.

Lemma 5.3.8. For any doctrine $\mathbb{D}$, the double category $\mathbf{Arena}_\mathbb{D}$ of arenas in $\mathbb{D}$ is spanlike: every arena satisfies the horizontal factor condition.

Proof. Fix an arena $\binom{I}{O}$ and suppose that we have a square like so:

$$\alpha = \begin{array}{c} \binom{I}{O} \xRightarrow{\binom{f_b}{f}} \binom{A^-}{A^+} \\[2ex] \Big\| \quad\quad \downarrow\uparrow\binom{w^\sharp}{w} \\ \Big\| \quad\quad \binom{B^-}{B^+} \\ \Big\| \quad\quad \downarrow\uparrow\binom{v^\sharp}{v} \\ \binom{I}{O} \xRightarrow[\binom{h_b}{h}]{} \binom{C^-}{C^+} \end{array}$$

Explicitly, this means that we have commuting squares

$$
\begin{array}{ccc}
O & \xrightarrow{f} & A^+ \\
\| & & \downarrow w \\
& & B^+ \\
& & \downarrow v \\
O & \xrightarrow{h} & C^+
\end{array}
\qquad
\begin{array}{ccc}
I & = & I \\
h_\flat \downarrow & & \downarrow f_\flat \\
h^*C^- & = & f^*w^*v^*C^- \xrightarrow{f^*(w^*v^\sharp\,\S\,w^\sharp)} f^*A^-
\end{array}
\qquad (5.5)
$$

We then get a chart

$$
\begin{pmatrix} h_\flat \,\S\, f^*w^*v^\sharp \\ f \,\S\, w \end{pmatrix} : \begin{pmatrix} I \\ O \end{pmatrix} \Rightarrow \begin{pmatrix} B^- \\ B^+ \end{pmatrix}.
$$

This chart fits into two squares like so:

$$
\begin{array}{ccc}
\begin{pmatrix} I \\ O \end{pmatrix} & \xrightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} & \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \\[2ex]
\| & \begin{pmatrix} h_\flat\,\S\,f^*w^*v^\sharp \\ f\,\S\,w \end{pmatrix} & \downarrow\uparrow \begin{pmatrix} w^\sharp \\ w \end{pmatrix} \\[2ex]
\begin{pmatrix} I \\ O \end{pmatrix} & \Longrightarrow & \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\[2ex]
\| & & \downarrow\uparrow \begin{pmatrix} v^\sharp \\ v \end{pmatrix} \\[2ex]
\begin{pmatrix} I \\ O \end{pmatrix} & \xrightarrow{\begin{pmatrix} h_\flat \\ h \end{pmatrix}} & \begin{pmatrix} C^+ \\ C^- \end{pmatrix}
\end{array}
$$

The bottom half of the top square and the top half of the bottom square commute by definition. The bottom half of the bottom square asks that $f \,\S\, w \,\S\, v = h$, but this is precisely the bottom half of α. The top half of the top square asks that the following diagram commute:

$$
\begin{array}{ccc}
I & = & I \\
h_\flat \downarrow & & \\
h^*C^- & & \\
\| & & \Big| f_\flat \\
f^*w^*v^*C^- & & \\
f^*w^*v^\sharp \downarrow & & \\
f^*w^*B^- & \xrightarrow{f^*w^\sharp} & f^*A^-
\end{array}
$$

This is a rearrangement of the second square in Diagram **??**.

Because we have just rearranged the data of the big outer square α, this factorization of α is *unique*.   □

As a corollary, **??** gives us a representable double functor

$$\mathbf{Arena}_{\mathbb{D}}\left(\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right), -\right) : \mathbf{Arena}_{\mathbb{D}} \to \mathbf{Matrix}$$

in any doctrine $\mathbb{D}$. So we can turn any lens in any doctrine into a matrix in a way that preserves the composition of lenses.

Theorem 5.3.9. For any doctrine $\mathbb{D}$ and any arena $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$, there is a representable double functor

$$\mathbf{Arena}_{\mathbb{D}}\left(\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right), -\right) : \mathbf{Arena}_{\mathbb{D}} \to \mathbf{Matrix}.$$

5.3.10   How behaviors of systems wire together: representable doubly indexed functors

We now come to the mountaintop. It's been quite a climb, and we're almost there.

We can now describe all the ways that behaviors of systems get put together when we wire systems together. There are a bunch of laws governing how behaviors get put together, and we organize them all into the notion of a *lax doubly indexed functor*. To any system T, we will give a lax doubly indexed functor

$$\mathrm{Behave}_{\mathsf{T}} : \mathbf{Sys} \to \mathbf{Vec}.$$

Since behaviors of shape T are a sort of map out of T, we may think of $\mathrm{Behave}_{\mathsf{T}}$ as a *representable* lax doubly indexed functor.

Theorem 5.3.11. For any doctrine $\mathbb{D}$ and any system T in $\mathbb{D}$, there is a lax doubly indexed functor $\mathrm{Behave}_{\mathsf{T}} : \mathbf{Sys}_{\mathbb{D}} \to \mathbf{Vec}$ which sends systems to their sets of T-shaped behaviors.

Let's see what this theorem is really asking for while we construct it. As with many of the constructions we have been seeing, the hard part is understanding what we are supposed to be constructing; once we do that, the answer will always be "compose in the appropriate way in the appropriate double category".

- First, we need $\text{Behave}_T^0 :$ **Arena** $\to$ **Matrix** which send an arena to the set of charts from $\left(\begin{smallmatrix} \text{In}_T \\ \text{Out}_T \end{smallmatrix}\right)$ to that arena. It will send a chart to the function given by composing with that chart, and it will send a lens to a matrix that describes the wiring pattern in the lens. We've seen how to do this in **??**:

$$\text{Behave}_T^0 = \textbf{Arena}\left(\left(\begin{smallmatrix} \text{In}_T \\ \text{Out}_T \end{smallmatrix}\right), -\right)$$

This is the blueprint for how our systems will compose.

- Next, for any arena $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$, we need a functor

$$\text{Behave}_T^{\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)} : \textbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \to \textbf{Vec}\left(\textbf{Arena}\left(\left(\begin{smallmatrix} \text{In}_T \\ \text{Out}_T \end{smallmatrix}\right), \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)\right)\right)$$

which will send a system S with interface $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ to its set of behaviors of shape T, indexed by their chart. That is, we make the following definition:

$$\text{Behave}_T^{\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)}(S)_{\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)} := \textbf{Sys}\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)(T, S).$$

This is functorial by horizontal associativity of squares in **Arena**.

- For any lens $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right) : \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \leftrightarrows \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$, we need a natural transformation

This will take any behaviors of component systems whose charts compatible according to the wiring pattern of $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right)$ and wire them together into a behavior of the wired together systems. In other words, this will be given by vertical composition of squares in **Arena**. To see how that works, we need follow a $\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$-system S around this diagram and see how this natural transformation can be described so simply. Following S around the top path of the diagram gives us the following vector of sets, we first send S to the vector of sets

$$\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right) : \left(\begin{smallmatrix} \text{In}_T \\ \text{Out}_T \end{smallmatrix}\right) \Rightarrow \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \mapsto \textbf{Sys}\left(\begin{smallmatrix} f_\flat \\ f \end{smallmatrix}\right)(T, S)$$

$$= \left\{ \begin{array}{c} \begin{pmatrix} \mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\ \begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix} \Big\updownarrow\Big\uparrow \qquad\qquad \Big\updownarrow\Big\uparrow \begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} \\ \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow[\begin{pmatrix} f^\sharp \\ f \end{pmatrix}]{} \begin{pmatrix} I \\ O \end{pmatrix} \end{array} \right\}$$

We then multiply this by the matrix **Arena** $\left(\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix}, \begin{pmatrix} w^\sharp \\ w \end{pmatrix} \right)$ to get the vector of sets whose entries are pairs of the following form:

$$\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \Rightarrow \begin{pmatrix} I' \\ O' \end{pmatrix} \mapsto \left\{ \begin{array}{cc} \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} I \\ O \end{pmatrix} & \begin{pmatrix} \mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\ \Big\|\Big\|\Big| \quad \Big\updownarrow\Big\uparrow \begin{pmatrix} w^\sharp \\ w \end{pmatrix} , \quad \begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix}\Big\updownarrow\Big\uparrow \qquad \Big\updownarrow\Big\uparrow\begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} \\ \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} \begin{pmatrix} I' \\ O' \end{pmatrix} & \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow[\begin{pmatrix} f_\flat \\ f \end{pmatrix}]{} \begin{pmatrix} I \\ O \end{pmatrix} \end{array} \right\}$$

On the other hand, following S along the bottom path has us first composing it vertically with $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$ and then finding the behaviors in it:

$$\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \Rightarrow \begin{pmatrix} I' \\ O' \end{pmatrix} \mapsto \left\{ \begin{array}{c} \begin{pmatrix} \mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\ \begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix}\Big\updownarrow\Big\uparrow \qquad \Big\updownarrow\Big\uparrow\begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix}\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \\ \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} \begin{pmatrix} I' \\ O' \end{pmatrix} \end{array} \right\}$$

Finally, we are ready to define our natural transformation from the virst vector of sets to the second using vertical composition:

$$\mathsf{Behave_T}^{\begin{pmatrix} w^\sharp \\ w \end{pmatrix}}(\mathsf{S})_{\begin{pmatrix} g_\flat \\ g \end{pmatrix}}(\square_w, \phi) = \frac{\phi}{\square_w}.$$

That this is natural for behaviors $\psi : S \to U$ in $\mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)$ follows quickly from the horizontal identity and interchange laws in **Arena**:

$$\frac{\phi \mid \psi}{\Box_w} = \frac{\phi \mid \psi}{\Box_w \mid \left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right)}$$

$$= \frac{\phi}{\Box_w} \left| \frac{\psi}{\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix}\right)} \right. .$$

- For any chart $\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix}\right) : \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) \Rightarrow \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)$, we need a square

$$
\begin{array}{ccc}
\mathbf{Sys}\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right) & \xrightarrow{\mathbf{Sys}\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix}\right)} & \mathbf{Sys}\left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right) \\
\Big\downarrow{\scriptstyle \mathrm{Behave}_T^{\left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)}} & {\scriptstyle \mathrm{Behave}_T^{\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix}\right)}} & \Big\downarrow{\scriptstyle \mathrm{Behave}_T^{\left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)}} \\
\mathbf{Vec}\left(\mathbf{Arena}\left(\left(\begin{smallmatrix} \mathrm{In}_T \\ \mathrm{Out}_T \end{smallmatrix}\right), \left(\begin{smallmatrix} I \\ O \end{smallmatrix}\right)\right)\right) & \xrightarrow[\mathbf{Vec}\left(\mathbf{Arena}\left(\left(\begin{smallmatrix} \mathrm{In}_T \\ \mathrm{Out}_T \end{smallmatrix}\right), \left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix}\right)\right)\right)]{} & \mathbf{Vec}\left(\mathbf{Arena}\left(\left(\begin{smallmatrix} \mathrm{In}_T \\ \mathrm{Out}_T \end{smallmatrix}\right), \left(\begin{smallmatrix} I' \\ O' \end{smallmatrix}\right)\right)\right)
\end{array}
$$

This will take any behavior from S to U with chart $\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix}\right)$ and give the function which takes behaviors of shape T in S and gives the composite behavior of shape T in U. That is,

$$\mathrm{Behave}_T^{\left(\begin{smallmatrix} g_\flat \\ g \end{smallmatrix}\right)}(S, U)(\psi) = \phi \mapsto \phi \mid \psi.$$

The naturality of this assignment follows from horizontal associativity in **Arena**.

Its a bit scary to see written out with all the names and symbols, but the idea is simple enough. We are composing two sorts of things: behaviors and systems. If we have some behaviors of shape T in our systems and their charts are compatible with a wiring pattern, then we get a behavior of the wired together system. If we have a chart, then behaviors with that chart give us a way of mapping forward behaviors of shape T.

The lax doubly indexed functor laws now tell us some facts about how these two sorts of composition interact.

- (Vertical Lax Functoriality) This asks us to suppose that we are wiring our systems together in two stages. The law then says that if we take a bunch of behaviors whose charts are compatible for the total wiring pattern and wire them together into a behavior of the whole system, this is the same behavior we get if we first noticed that they were compatible for the first wiring pattern, wired them together, then noticed that the result was compatible for the second wiring pattern,

and wired that together. This means that nesting of wiring diagrams commutes with finding behaviors of our systems.

- (Horizontal Functoriality) This asks us to suppose that we have two charts and a behavior of each. The law then says that composing a behavior of shape T the composite of the behaviors is the same as composing it with the first one and then with the second one.

- (Functorial Interchange) This asks us to suppose that we have a pair of wiring patterns and compatible charts between them (a square in **Arena**). The law then says that if we take a bunch of behaviors whose charts are compatable according to the first wiring pattern, wire them together, and then compose with a behavior of the second chart, we get the same thing as if we compose them all with behaviors of the first chart, noted that they were compatible with the second wiring pattern, and then wired them together.

Though it seems like it would be a mess of symbols to check these laws, they in fact fall right out of the laws for the double categories of arenas and matrices, and the functoriality of **??**. That is, we've already built up all the tools we need to prove this fact, we just need to finish describing it.

- (Vertical Lax Functoriality) Suppose we have composable lenses $\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) : \left(\begin{smallmatrix} I_1 \\ O_1 \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} I_2 \\ O_2 \end{smallmatrix} \right)$ and $\left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right) : \left(\begin{smallmatrix} I_2 \\ O_2 \end{smallmatrix} \right) \leftrightarrows \left(\begin{smallmatrix} I_3 \\ O_3 \end{smallmatrix} \right)$. We need to show that

$$\mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \mathbin{\mathring{,}} \left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right)} = \left(\mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right)} \mathbf{Sys}\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \right) \circ \left(\mathbf{VecArena}\left(\left(\begin{smallmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{smallmatrix} \right), \left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right) \right) \mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right)} \right).$$

This follows immediately from vertical associativity in **Arena**, once both sides have been expanded out. Let S be a $\left(\begin{smallmatrix} I_1 \\ O_1 \end{smallmatrix} \right)$-system, then

$$\mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \mathbin{\mathring{,}} \left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right)}(\mathsf{S})(\alpha, \phi) = \mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \mathbin{\mathring{,}} \left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right)}(\mathsf{S})\left(\frac{\beta}{\gamma}, \phi \right) \qquad \text{by}$$

$$= \frac{\phi}{\frac{\beta}{\gamma}}$$

$$= \frac{\frac{\phi}{\beta}}{\gamma}$$

$$= \left(\mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right)} \mathbf{Sys}\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right) \right) \circ \left(\mathbf{Arena}\left(\left(\begin{smallmatrix} \mathsf{In}_\mathsf{T} \\ \mathsf{Out}_\mathsf{T} \end{smallmatrix} \right), \left(\begin{smallmatrix} u^\sharp \\ u \end{smallmatrix} \right) \right) \mathsf{Behave}_\mathsf{T}^{\left(\begin{smallmatrix} w^\sharp \\ w \end{smallmatrix} \right)} \right)(\gamma, \beta, \phi).$$

- (Horizontal Functoriality) This follows directly from horizontal associativity in **Arena**.

- (Functorial Interchange) This law will follow directly from interchange in the

double category of arenas. Let α be a square in **Arena** of the following form:

$$\alpha = \begin{pmatrix} j^\sharp \\ j \end{pmatrix} \Bigg\downarrow\uparrow \qquad \begin{array}{ccc} \begin{pmatrix} A^- \\ A^+ \end{pmatrix} & \overset{\begin{pmatrix} f_\flat \\ f \end{pmatrix}}{\Longrightarrow} & \begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\ & & \uparrow\downarrow \begin{pmatrix} k^\sharp \\ k \end{pmatrix} \\ \begin{pmatrix} C^- \\ C^+ \end{pmatrix} & \underset{\begin{pmatrix} g^\sharp \\ g \end{pmatrix}}{\Longrightarrow} & \begin{pmatrix} D^- \\ D^+ \end{pmatrix} \end{array}$$

We need to show that

$$\mathrm{Behave}_\mathsf{T}^{\begin{pmatrix} j^\sharp \\ j \end{pmatrix}} \left| \frac{\mathbf{Sys}(\alpha)}{\mathrm{Behave}_\mathsf{T}^{\begin{pmatrix} g_\flat \\ g \end{pmatrix}}} \right. = \left. \frac{\mathrm{Behave}_\mathsf{T}^{\begin{pmatrix} f_\flat \\ f \end{pmatrix}}}{\mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \alpha\right)} \right| \mathrm{Behave}_\mathsf{T}^{\begin{pmatrix} k^\sharp \\ k \end{pmatrix}} \qquad (5.6)$$

We can see both sides as natural transformations of the signature

$$\begin{array}{ccc}
\mathbf{Sys}\begin{pmatrix} A^- \\ A^+ \end{pmatrix} & \xrightarrow{\;\;\mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}\;\;} & \mathbf{Sys}\begin{pmatrix} B^- \\ B^+ \end{pmatrix} \\[2mm]
{\scriptstyle \mathrm{Behave}_\mathsf{T}^{\begin{pmatrix} A^- \\ A^+ \end{pmatrix}}}\Big\downarrow & & \Big\downarrow {\scriptstyle \mathbf{Sys}\begin{pmatrix} k^\sharp \\ k \end{pmatrix}} \\[2mm]
\mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \begin{pmatrix} A^- \\ A^+ \end{pmatrix}\right) & \textbf{??} & \mathbf{Sys}\begin{pmatrix} D^- \\ D^+ \end{pmatrix} \\[2mm]
{\scriptstyle \mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \begin{pmatrix} j^\sharp \\ j \end{pmatrix}\right)}\Big\downarrow & & \Big\downarrow {\scriptstyle \mathrm{Behave}_\mathsf{T}^{\begin{pmatrix} D^- \\ D^+ \end{pmatrix}}} \\[2mm]
\mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \begin{pmatrix} C^- \\ C^+ \end{pmatrix}\right) & \xrightarrow[\;\mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \begin{pmatrix} g_\flat \\ g \end{pmatrix}\right)\;]{} & \mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \begin{pmatrix} D^- \\ D^+ \end{pmatrix}\right)
\end{array}$$

So, to show this equality holds, let's start with a behavior $\psi \in \mathbf{Sys}\begin{pmatrix} f_\flat \\ f \end{pmatrix}(S, U)$ with chart $\begin{pmatrix} f_\flat \\ f \end{pmatrix}$. We need to show that passing this through the left side of **??** equals the result of passing it through the right hand side. The result is an element of

$$\mathbf{VecArena}\left(\begin{pmatrix} \mathrm{In}_\mathsf{T} \\ \mathrm{Out}_\mathsf{T} \end{pmatrix}, \begin{pmatrix} g_\flat \\ g \end{pmatrix}\right)(\cdots, \cdots)$$

and is for that reason a function that takes in a pair of the following form:

$$(\Box_j, \phi) = \begin{pmatrix} \begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix} \xdashrightarrow{\begin{pmatrix} a_\flat \\ a \end{pmatrix}} \begin{pmatrix} A^- \\ A^+ \end{pmatrix} & \begin{pmatrix} \text{State}_T \\ \text{State}_T \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \text{State}_S \\ \text{State}_S \end{pmatrix} \\ \Big\|\Big\| \quad \Uparrow\begin{pmatrix} j^\sharp \\ j \end{pmatrix}, & \begin{pmatrix} \text{update}_T \\ \text{expose}_T \end{pmatrix}\Updownarrow \quad \Uparrow\begin{pmatrix} \text{update}_S \\ \text{expose}_S \end{pmatrix} \\ \begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix} \xRightarrow[\begin{pmatrix} c_\flat \\ c \end{pmatrix}]{} \begin{pmatrix} C^- \\ C^+ \end{pmatrix} & \begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix} \xRightarrow[\begin{pmatrix} a^\sharp \\ a \end{pmatrix}]{} \begin{pmatrix} A^- \\ A^+ \end{pmatrix} \end{pmatrix}$$

The left hand side sends this pair to

$$\text{Behave}_T^{\begin{pmatrix} g_\flat \\ g \end{pmatrix}}(\mathbf{Sys}(\alpha)(\psi))\left(\text{Behave}_T^{\begin{pmatrix} j^\sharp \\ j \end{pmatrix}}(\Box_j, \phi) \right)$$

which equals, rather simply:

$$\begin{array}{c|c} \phi & \psi \\ \hline \Box_j & \alpha \end{array} \ .$$

The right hand side sends the pair to

$$\text{Behave}_T^{\begin{pmatrix} k^\sharp \\ k \end{pmatrix}}\left(\mathbf{VecArena}\left(\begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix}, \alpha \right)\left(\Box_j, \text{Behave}_T^{\begin{pmatrix} f_\flat \\ f \end{pmatrix}}(\psi)(\phi) \right) \right)$$

which equals, rather simply:

$$\frac{\phi \mid \psi}{\Box_j \mid \alpha}.$$

That these two composites are equal is precisely the interchange law of a double category.

While we have phrased this theorem in terms of dynamical systems doctrine, the proof uses only the structure available to the doubly indexed category $\mathbf{Sys}_\mathbb{D}$: $\mathbf{Arena}_\mathbb{D} \to \mathbf{Cat}$. We can therefore state this theorem entirely abstractly, which we record here.

Theorem 5.3.12. Let $\mathcal{A} : \mathcal{D} \to \mathbf{Cat}$ be a doubly indexed category with $\mathcal{D}$ a spanlike double category. Then for any $\mathbf{Sys}T \in \mathcal{A}(D)$, there is a *representable* lax doubly indexed functor

5.3.13 Is the whole always more than the composite of its parts?

Unfortunately, Behave_T is lax (and not taut) for general T. This means that while behaviors of component systems will induce behaviors of composite systems, it isn't necessarily the case that all behaviors of the composite arise this way.

But there is a simple condition we can put on T which will ensure that Behave_T *is* taut, and therefore that we can recover the behaviors of wired together systems from the behaviors of their components: we ask that T expose its entire state, which is to say that expose_T is an isomorphism.

Theorem 5.3.14. Let T be a system in the doctrine $\mathbb{D}$, and suppose that expose_T is an isomorphism. Then the representable lax doubly indexed functor Behave_T is in fact taut. Explicitly, for any lens $\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} I \\ O \end{pmatrix} \leftrightarrows \begin{pmatrix} I' \\ O' \end{pmatrix}$ the natural transformation

$$
\begin{array}{ccc}
\mathbf{Sys}\begin{pmatrix} I \\ O \end{pmatrix} & \xrightarrow{\ \text{Behave}_T^{\begin{pmatrix} I \\ O \end{pmatrix}}\ } & \mathbf{Vec}\left(\mathbf{Arena}\left(\begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix}, \begin{pmatrix} I \\ O \end{pmatrix}\right)\right) \\
\text{Sys}\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \Big\downarrow \quad {\Large \text{Behave}_T^{\begin{pmatrix} w^\sharp \\ w \end{pmatrix}}} & & \Big\downarrow \text{Vec}\left(\text{Arena}\left(\begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix}, \begin{pmatrix} w^\sharp \\ w \end{pmatrix}\right)\right) \\
\mathbf{Sys}\begin{pmatrix} I' \\ O' \end{pmatrix} & \xrightarrow[\ \text{Behave}_T^{\begin{pmatrix} B^- \\ B^+ \end{pmatrix}}\ ]{} & \mathbf{Vec}\left(\mathbf{Arena}\left(\begin{pmatrix} \text{In}_T \\ \text{Out}_T \end{pmatrix}, \begin{pmatrix} I \\ O \end{pmatrix}\right)\right)
\end{array}
$$

is a natural isomorphism.

Many of the systems representing sorts of behavior which we saw in **??** expose their entire state. For example, the systems Time representing trajectories (**??**), Fix representing steady states (**??**), and Clock_n periodic orbits with periodic parameters (**??**). As examples of systems which don't expose their entire state, we had the systems which represent steady looking trajectories and periodic orbits whose parameters aren't periodic from **??**. **??** says that for the systems Time, Fix, and Clock_n, we can recover the behaviors of component systems from the behaviors of composite systems. As we noted in **??**, the same cannot be said for steady looking states.

Proof of **??**. We recall that for a $\begin{pmatrix} I \\ O \end{pmatrix}$-system S, the natural transformation $\text{Behave}_T^{\begin{pmatrix} w^\sharp \\ w \end{pmatrix}}$

goes from the vector of sets

$$
\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \Rightarrow \begin{pmatrix} I' \\ O' \end{pmatrix} \mapsto
\left\{
\begin{array}{c}
\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} I \\ O \end{pmatrix} \qquad
\begin{pmatrix} \mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\
\Vert\Vert \qquad \Big\Uparrow\begin{pmatrix} w^\sharp \\ w \end{pmatrix}, \qquad \begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix}\Big\Updownarrow\Big\Uparrow \qquad \Big\Updownarrow\Big\Uparrow\begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix} \\
\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} g_\flat \\ g \end{pmatrix}} \begin{pmatrix} I' \\ O' \end{pmatrix} \qquad
\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} f_\flat \\ f \end{pmatrix}} \begin{pmatrix} I \\ O \end{pmatrix}
\end{array}
\right\}
$$

to the vector of sets

$$
\begin{pmatrix} g_\flat \\ g \end{pmatrix} : \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \Rightarrow \begin{pmatrix} I' \\ O' \end{pmatrix} \mapsto
\left\{
\begin{array}{c}
\begin{pmatrix} \mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\
\begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix}\Big\Updownarrow\Big\Uparrow \qquad \Big\Updownarrow\Big\Uparrow\begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix}\begin{smallmatrix}\mathstrut\\\circ\\\circ\end{smallmatrix}\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \\
\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} g_\flat \\ g \end{pmatrix}} \begin{pmatrix} I' \\ O' \end{pmatrix}
\end{array}
\right\}
$$

The transformation itself is given by vertical composition:

$$
\mathrm{Behave_T}^{\begin{pmatrix} w^\sharp \\ w \end{pmatrix}}(\mathsf{S})_{\begin{pmatrix} g_\flat \\ g \end{pmatrix}}(\square_w, \phi) = \frac{\phi}{\square_w}.
$$

We'll construct an inverse to this assuming that $\mathsf{expose_T}$ is an isomorphism. Suppose we have a square

$$
\alpha = \begin{pmatrix} \mathsf{update_T} \\ \mathsf{expose_T} \end{pmatrix}\Big\Updownarrow\Big\Uparrow
\qquad
\begin{array}{c}
\begin{pmatrix} \mathsf{State_T} \\ \mathsf{State_T} \end{pmatrix} \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} \begin{pmatrix} \mathsf{State_S} \\ \mathsf{State_S} \end{pmatrix} \\[2pt]
\Big\Updownarrow\Big\Uparrow\begin{pmatrix} \mathsf{update_S} \\ \mathsf{expose_S} \end{pmatrix}\begin{smallmatrix}\mathstrut\\\circ\\\circ\end{smallmatrix}\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \\[2pt]
\begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \xRightarrow{\begin{pmatrix} g_\flat \\ g \end{pmatrix}} \begin{pmatrix} I' \\ O' \end{pmatrix}
\end{array}
$$

From this data, we can define a chart

$$
\begin{pmatrix} h_\flat \\ h \end{pmatrix} = \begin{pmatrix} g_\flat \,\mathbin{\mathring{\,}}\, \mathsf{expose_T^{-1*}}\phi^*\mathsf{expose_S^*}\,w^\sharp \\ \mathsf{expose_T^{-1}} \,\mathbin{\mathring{\,}}\, \phi \,\mathbin{\mathring{\,}}\, \mathsf{expose_S} \end{pmatrix} : \begin{pmatrix} \mathsf{In_T} \\ \mathsf{Out_T} \end{pmatrix} \Rightarrow \begin{pmatrix} I \\ O \end{pmatrix}.
$$

It isn't obvious that the top composite is well defined since we have $g_\flat : \mathsf{In}_T \to g^*I'$ and $\mathrm{expose}_T^{-1*}\phi^*\mathrm{expose}_S^* w^\sharp : \mathrm{expose}_T^{-1*}\phi^*\mathrm{expose}_S^* w^* I' \to \mathrm{expose}_T^{-1*}\phi^*\mathrm{expose}_S^* I$ and the codomain of the first doesn't appear to be the domain of the second. But the square α tells us that $\phi \mathbin{\mathring{,}} \mathrm{expose}_S \mathbin{\mathring{,}} w = \mathrm{expose}_T \mathbin{\mathring{,}} g$, so we have that

$$\mathrm{expose}_T^{-1} \mathbin{\mathring{,}} \phi \mathbin{\mathring{,}} \mathrm{expose}_S \mathbin{\mathring{,}} w = \mathrm{expose}_T^{-1} \mathbin{\mathring{,}} \mathrm{expose}_T \mathbin{\mathring{,}} g = g.$$

So the two maps really are composable.

Next, we note that the following is a square:

$$\Box_h = \begin{array}{ccc} \begin{pmatrix} \mathsf{In}_T \\ \mathsf{Out}_T \end{pmatrix} & \xdashrightarrow{\begin{pmatrix} h_\flat \\ h \end{pmatrix}} & \begin{pmatrix} I \\ O \end{pmatrix} \\ {\Big\|\Big\|\Big\|} & & {\Big\uparrow\Big\downarrow}\begin{pmatrix} w^\sharp \\ w \end{pmatrix} \\ \begin{pmatrix} \mathsf{In}_T \\ \mathsf{Out}_T \end{pmatrix} & \xrightarrow[\begin{pmatrix} g_\flat \\ g \end{pmatrix}]{} & \begin{pmatrix} I' \\ O' \end{pmatrix} \end{array}$$

since

$$
\begin{aligned}
h \mathbin{\mathring{,}} w &= \mathrm{expose}_T^{-1} \mathbin{\mathring{,}} \phi \mathbin{\mathring{,}} \mathrm{expose}_S \mathbin{\mathring{,}} w \\
&= \mathrm{expose}_T^{-1} \mathbin{\mathring{,}} \mathrm{expose}_T \mathbin{\mathring{,}} g \\
&= g \qquad\qquad\qquad\qquad\qquad\qquad \text{and} \\
h_\flat &= g_\flat \mathbin{\mathring{,}} h^* w^\sharp.
\end{aligned}
$$

We see that the definition of $\begin{pmatrix} h_\flat \\ h \end{pmatrix}$ is basically forced on us by the commutation of this diagram. Furthermore, we note that we have a square:

$$\beta = \begin{array}{ccc} \begin{pmatrix} \mathsf{State}_T \\ \mathsf{State}_T \end{pmatrix} & \xdashrightarrow{\begin{pmatrix} T\phi \\ \phi \end{pmatrix}} & \begin{pmatrix} \mathsf{State}_S \\ \mathsf{State}_S \end{pmatrix} \\ \begin{pmatrix} \mathrm{update}_T \\ \mathrm{expose}_T \end{pmatrix}{\Big\updownarrow\Big\uparrow} & & {\Big\updownarrow\Big\uparrow}\begin{pmatrix} \mathrm{update}_S \\ \mathrm{expose}_S \end{pmatrix} \\ \begin{pmatrix} \mathsf{In}_T \\ \mathsf{Out}_T \end{pmatrix} & \xrightarrow[\begin{pmatrix} h_\flat \\ h \end{pmatrix}]{} & \begin{pmatrix} I \\ O \end{pmatrix} \end{array}$$

by reinterpreting α.

Finally, it remains to show that for any $\begin{pmatrix} h_\flat \\ h \end{pmatrix}$ fitting into these two squares $\Box_h$ and β, we have that

$$\begin{pmatrix} h_\flat \\ h \end{pmatrix} = \begin{pmatrix} g_\flat \mathbin{\mathring{,}} \mathrm{expose}_T^{-1*}\phi^*\mathrm{expose}_S^* w^\sharp \\ \mathrm{expose}_T^{-1} \mathbin{\mathring{,}} \phi \mathbin{\mathring{,}} \mathrm{expose}_S \end{pmatrix}.$$

In fact, this equality follows by $\square_h$ alone. From the bottom of $\square_h$, we see that $h = g \mathbin{\fatsemi} w$, which equals $\mathrm{expose}_\mathsf{T}^{-1} \mathbin{\fatsemi} \phi \mathbin{\fatsemi} \mathrm{expose}_\mathsf{S}$. From the top, we see exactly that $h_\flat = g_\flat \mathbin{\fatsemi} h^* w^\sharp$. $\quad\square$

Example 5.3.15. In the deterministic doctrine $\mathbb{D}\text{ET}$, consider the system Time of **??**:

$$\begin{pmatrix} t \mapsto t + 1 \\ \mathrm{id} \end{pmatrix} : \begin{pmatrix} \mathbb{N} \\ \mathbb{N} \end{pmatrix} \leftrightarrows \begin{pmatrix} \{\text{tick}\} \\ \mathbb{N} \end{pmatrix}.$$

This system exposes its entire state since $\mathrm{expose}_\text{Time} = \mathrm{id}$. A behavior of shape Time is a trajectory. So, by **??**, we get a doubly indexed functor:

For any $\begin{pmatrix} I \\ O \end{pmatrix}$-system S, we get a vector of sets

$$\begin{pmatrix} i \\ o \end{pmatrix} \mapsto \mathrm{Traj}_{\begin{pmatrix} i \\ o \end{pmatrix}}(\mathsf{S})$$

sending each chart $\begin{pmatrix} i \\ o \end{pmatrix} : \begin{pmatrix} 1 \\ \mathbb{N} \end{pmatrix} \Rightarrow \begin{pmatrix} I \\ O \end{pmatrix}$ — which is to say sequences $o : \mathbb{N} \to O$ and $i : \mathbb{N} \to I$ of inputs — to the set of trajectories $s : \mathbb{N} \to \mathsf{S}$ for that chart. These trajectories are, explicitly, sequences which satisfy the equations

$$s_{t+1} = \mathrm{update}_\mathsf{S}(s_t, i_t)$$
$$\mathrm{expose}_\mathsf{S}(s_t) = o_t.$$

?? tells us that trajectories in a composite system are families of trajectories for each system which agree on all the information passed between the wires. For example, consider the wiring diagram
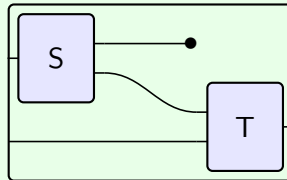
$$(5.7)$$

Let's suppose that all wires carry real numbers. Then this wiring diagram can be represented by the lens

$$\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} \mathbb{R} \\ \mathbb{R} \times \mathbb{R} \end{pmatrix} \otimes \begin{pmatrix} \mathbb{R} \times \mathbb{R} \\ \mathbb{R} \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R} \times \mathbb{R} \\ \mathbb{R} \end{pmatrix}$$

given by

$$w((a, b), c) = c$$
$$w^\sharp(((a, b), c), (x, y)) = (x, (b, y)).$$

Jaz: FINISH THIS.

Example 5.3.16. In a differential doctrine — for simplicity let's say the Euclidean differential doctrine $\mathbb{E}$uc — the system

$$\text{Time} = \begin{pmatrix} 1 \\ \text{id} \end{pmatrix} : \begin{pmatrix} \mathbb{R}^1 \\ \mathbb{R}^1 \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^0 \\ \mathbb{R}^1 \end{pmatrix}$$

which expresses the differential equation

$$\frac{ds}{dt} = 1$$

represents trajectories (see **??**). As this system exposes its entire state, **??** gives us a doubly indexed functor

5.4 The Behavioral Approach to Systems Theory

Throughout this book so far, we seen dynamical systems modeled by state spaces exposing variables and updating according to external parameters. This sort of dynamical system is *lens-based* — systems are themselves lenses, and they compose by lens composition. We might describe them as *parameter-setting systems*, since we compose these systems by setting the parameters of some according to the exposed state variables of others.

The parameter setting (lens-based) ways of thinking about systems are very useful for the *design* of systems; we give a minimal set of data (expose and update) which in principle determines all behaviors, though it might take some work to understand what behaviors are actually in the system once we have set it up. But for the *analysis* of dynamical systems, we seek to prove properties about how systems behave. It helps if we already know how a system behaves.

In the *behavioral* approach to systems theory, pioneered by Jan Willems, we take "behavior" as a primitive. In its most basic formulation, the behavioral approach to systems theory considers a system S to have a set $B(S)$ of state variables or "behaviors". The system also exposes some of these state variables in a function $\mathrm{expose}_S : B_S \to V_S$ to a set V_S of possible values for these exposed variables.

In other words, we can see the behavioral approach to systems theory as taking place in the doubly indexed category **Vec** : **Matrix** $\to$ **Cat** (or, as we'll see, some variants of it). An interface is a set V of possible values, and a system is a vector B_v of sets varying over $v \in V$ — the behaviors in which the exposed variables take that given value. This might sound a bit different from the idea of a function $\mathrm{expose}_S : B_S \to V_S$, but we can define the set B_v to be $\mathrm{expose}_S^{-1}(v)$, and pass between these two notions. That is, we are making use of the equivalence between the double categories of matrices and the double categories of spans explored in **??** to think of vectors of sets of length V as functions into V, and to think of matrices of sets as spans.

Example 5.4.1. Consider the Lotka-Volterra predator prey model LK of Section 1.2.15 which is given by the following system of differential equations:

$$\begin{cases} \frac{dr}{dt} = \mathsf{b}_{\mathsf{Rabbits}} \cdot r - c_1 f r \\ \frac{df}{dt} = c_2 r f - \mathsf{d}_{\mathsf{Foxes}} \cdot f \end{cases}$$

Here, r is the population of rabbits and f is the population of foxes. In Example 1.2.19, we saw how to represent this as a differential system $\begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R}^2 \end{pmatrix} \leftrightarrows \begin{pmatrix} \mathbb{R}^2 \\ 1 \end{pmatrix}$ with $\mathrm{expose}_{\mathsf{LK}} =!$ the terminal map and

$$\mathrm{update}_{\mathsf{Rabbits}}(r, (\mathsf{b}_{\mathsf{Rabbits}}, \mathsf{d}_{\mathsf{Rabbits}})) = \mathsf{b}_{\mathsf{Rabbits}} \cdot r - \mathsf{d}_{\mathsf{Rabbits}} \cdot r.$$

For the behavioral approach, we would apply the doubly indexed functor taking trajectories (from **??**) to get the behavioral point of view on the system LK:

$$\mathrm{Behave}_{\mathsf{Time}}^{\begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R}^2 \end{pmatrix}}(\mathsf{LK}) \in \mathbf{Vec}\left(\mathbf{Arena}_{\mathbb{E}\mathrm{uc}}\left(\begin{pmatrix} 1 \\ \mathbb{R} \end{pmatrix}, \begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R}^2 \end{pmatrix}\right)\right).$$

In other words, the set B_{LK} of behaviors is the set of trajectories together with their charts in LK, and V_{LK} is the set of charts for those trajectories — that is, parameters varying in time. We can calculate this from the definitions in **??**:

$$V_{LK} = \mathbf{Arena}_{\mathbb{E}UC}\left(\begin{pmatrix} 1 \\ \mathbb{R} \end{pmatrix}, \begin{pmatrix} \mathbb{R}^2 \\ 1 \end{pmatrix}\right)$$

$$= \left\{ \begin{pmatrix} (b_{Rabbits}, d_{Foxes}) \\ ! \end{pmatrix} : \begin{pmatrix} 1 \\ \mathbb{R} \end{pmatrix} \Rightarrow \begin{pmatrix} \mathbb{R}^2 \\ 1 \end{pmatrix} \right\}$$

$$= \{((b_{Rabbits}, d_{Foxes})) : \mathbb{R} \to \mathbb{R}^4\}.$$

$$B_{LK} = \left\{ \begin{array}{c} \begin{pmatrix} (\frac{dr}{dt}, \frac{df}{dt}) \\ (r,f) \end{pmatrix} \\ \begin{pmatrix} \mathbb{R} \\ \mathbb{R} \end{pmatrix} \dashrightarrow \begin{pmatrix} \mathbb{R}^2 \\ \mathbb{R}^2 \end{pmatrix} \\ \begin{pmatrix} 1 \\ id \end{pmatrix} \Big\Updownarrow \qquad \Big\Updownarrow \begin{pmatrix} update_{LK} \\ expose_{LK} \end{pmatrix} \\ \begin{pmatrix} 1 \\ \mathbb{R} \end{pmatrix} \dashrightarrow \begin{pmatrix} \mathbb{R}^2 \\ 1 \end{pmatrix} \\ (b_{Rabbits}, d_{Foxes}) \\ ! \end{array} \right\}$$

$$= \left\{ ((r, f), (b_{Rabbits}, d_{Foxes})) : \mathbb{R} \to \mathbb{R}^4 \,\middle|\, \begin{array}{l} \frac{dr}{dt} = b_{Rabbits} \cdot r - c_1 f r \\ \frac{df}{dt} = c_2 r f - d_{Foxes} \cdot f \end{array} \right\}.$$

and the map $B_{LK} \to V_{LK}$ is the projection exposing the parameters:

$$((r, f), (b_{Rabbits}, d_{Foxees})) \mapsto (b_{Rabbits}, d_{Foxees}).$$

Remark 5.4.2. Note that the parameters of the original differential system LK are considered as exposed variables of state in the behavioral approach. This is because the behavioral approach composes systems by setting exposed variables equal to eachother, so the parameters must be considered as exposed variables so that they can be set equal to other variables.

In this section, we'll a bit of how the behavioral approach to systems theory works, and why we might want to do it. We'll begin with the main idea in **??**. Then, in **??**, we'll see that in the behavioral approach, there is a different sort of *undirected* wiring diagram which is used to compose systems
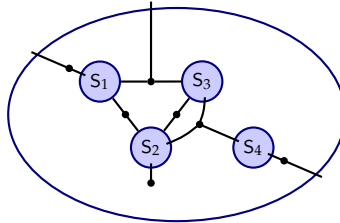
The idea of these *bubble diagrams* is that each wire carries an exposed variable of the behaviors of each system. An connection between wires expresses an equality of the

variables carried on them. The wiring diagram as a whole shows how the systems in it *share* their variables. If lens-based systems are all about setting parameters, the behavioral approach to systems theory using spans is all about sharing variables.

Just as the wiring diagrams for lens based systems are the lenses in categories of arities, the wiring diagrams for the span-based behavioral approach are spans in the category of arities — cospans of finite sets.

5.4.3 The idea of the behavioral approach

In the behavioral approach to systems theory, a system is a set (or variable set, see **??**) of *behaviors* of the system. A system exposes some variables of its behaviors. We can draw a behavioral system as a blob with wires dangling out of it which we imagine are carrying the exposed variables. For example, the following system S exposes three variables:

As we have seen in **??**, we can get behavioral systems for any type of behavior in any doctrine. One benefit of the behavioral approach is that all of these different doctrines can be composed on the same footing: they're all just behavioral systems.

Consider the following example borrowed from Willems' [**Willems:Behavioral.Approach**]. We consider a square bucket of water with two pipes at the bottom through which the water can flow:
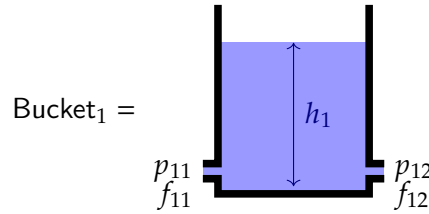
The variable behaviors here are the pressures p_{11} and p_{12}, the flows f_{11} and f_{12}, and the height of the water h_1. We suppose that these quantities are related in the following ways:

$$\frac{dh_1}{dt} = F_1(h_1, p_{11}, p_{12})$$
$$f_{11} = H_{11}(h_1, p_{11})$$
$$f_{12} = H_{12}(h_1, p_{12})$$

for some functions F_1, H_{11}, and H_{12}. Therefore, the set of behaviors is the set real valued functions of time which satisfy these laws:

$$B_{\mathsf{Bucket}_1} = \left\{ (h_1, f_{11}, f_{12}, p_{11}, p_{12}) : \mathbb{R} \to \mathbb{R}^5 \, \left| \, \begin{array}{l} \frac{dh_1}{dt} = F_1(h_1, p_{11}, p_{12}) \\ f_{11} = H_{11}(h_1, p_{11}) \\ f_{12} = H_{12}(h_1, p_{12}) \end{array} \right. \right\}$$
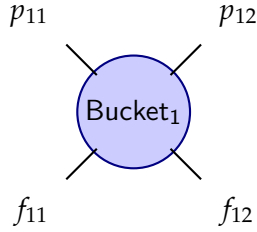
We will suppose that we will only pump water to and from the bucket through the pipes at the bottom. This means that we will only expose the variables concerning those pipes.

$$V_{\text{Bucket}_1} = (\mathbb{R}^4)^{\mathbb{R}}$$

and where

$$\text{expose}_{\text{Bucket}_1}(h_1, f_{11}, f_{12}, p_{11}, p_{12}) = (f_{11}, f_{12}, p_{11}, p_{12}).$$

We can bubble up the Bucket_1 system as the following bubble diagram:

Each wire carries a variable element of $\mathbb{R} \to \mathbb{R}$, and the Bucket_1 system exposes four of such variables.

Now, suppose we had another bucket Bucket_2, governed by a similar set of variables satisfying a similar set of laws, but with functions F_2, H_{21}, and H_{22} instead.
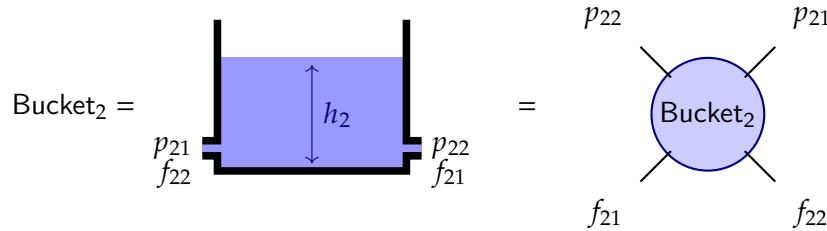
Suppose we connect the two buckets up by the pipes at the bottom:
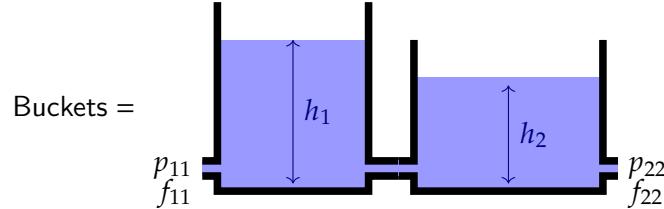
To express this combined system, we need that the pressures in the connected pipes to be equal (since they are now one pipe), and we need the flows to be opposite (since any flow out of one bucket goes into the other). That is, we need $p_{12} = p_{21}$ and $f_{12} + f_{21} = 0$. All in all, the combined system has behaviors

$$B_{\text{Buckets}} = \left\{ \begin{matrix} (h_1, f_{11}, f_{12}, p_{11}, p_{12}) : \mathbb{R} \to \mathbb{R}^5 \\ (h_2, f_{21}, f_{22}, p_{21}, p_{22}) : \mathbb{R} \to \mathbb{R}^5 \end{matrix} \left| \begin{matrix} \dfrac{dh_i}{dt} = F_i(h_i, p_{i1}, p_{i2}) \\ f_{i1} = H_{i1}(h_i, p_{i1}) \\ f_{i2} = H_{i2}(h_i, p_{i2}) \\ p_{12} = p_{21} \\ 0 = f_{12} + f_{21} \end{matrix} \right. \right\} \tag{5.8}$$

Meanwhile, the only variables which are exposed by Buckets are the two remaining open pipes, so

$$V_{\mathsf{Buckets}} = \mathbb{R} \to \mathbb{R}^4$$

$$\mathrm{expose}_{\mathsf{Buckets}} \begin{pmatrix} (h_1, f_{11}, f_{12}, p_{11}, p_{12}) \\ (h_2, f_{21}, f_{22}, p_{21}, p_{22}) \end{pmatrix} = (f_{11}, f_{22}, p_{11}, p_{22}).$$

We can express the pattern of interconnection between Bucket_1 and Bucket_2 as a bubble diagram to see precisely how Buckets arises as a composition of the two systems:

(5.9)
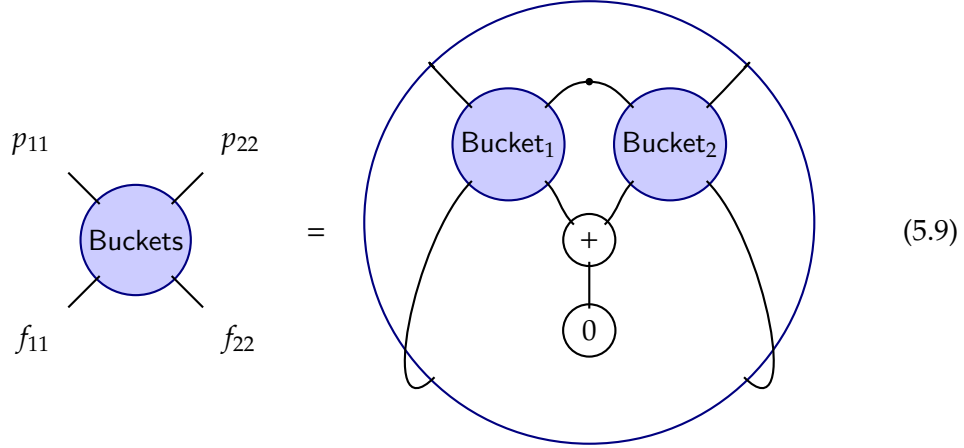
When the wires are connected in this diagram, they express an *equality* between their exposed variables. The top connection signifies that $p_{12} = p_{21}$. The bubbled up $+$ signifies a relation (other than equality): it says that the sum of the variables on the top two wires equals the third wire. We set that third wire to be constant at 0, so in total we get the relation that $f_{12} + f_{21} = 0$.

We can analyze this composition of the systems Bucket_1 and Bucket_2 in terms of the doubly indexed category $\mathbf{Vec} : \mathbf{Matrix} \to \mathbf{Cat}$, or rather the equivalent doubly indexed category $\mathbf{Set}/(-) : \mathbf{Span}(\mathbf{Set}) \to \mathbf{Cat}$. To see how this works, let's remember how we compose lens based systems in a given doctrine $\mathbb{D}$.

Let's compose the Clock and Meridian systems into the ClockWithDisplay system from **??**.

(5.10)
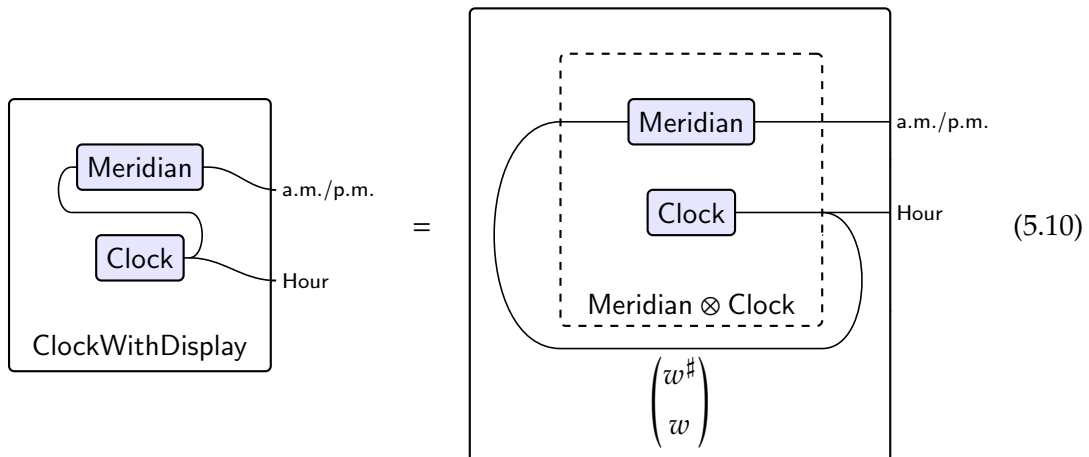
We begin with Clock, which is in the category $\mathbf{Sys}_{\mathbb{D}\text{ET}}\left(\begin{smallmatrix} 1 \\ \text{Hour} \end{smallmatrix}\right)$ of deterministic systems with interface $\left(\begin{smallmatrix} 1 \\ \text{Hour} \end{smallmatrix}\right)$, and Meridian, which is in the category $\mathbf{Sys}_{\mathbb{D}\text{ET}}\left(\begin{smallmatrix} \text{Hour} \\ \text{a.m./p.m.} \end{smallmatrix}\right)$ of deterministic systems with interface $\left(\begin{smallmatrix} \text{Hour} \\ \text{a.m./p.m.} \end{smallmatrix}\right)$. We then form their parallel product Meridian $\otimes$ Clock, which is in $\mathbf{Sys}_{\mathbb{D}\text{ET}}\left(\begin{smallmatrix} 1\times\text{Hour} \\ \text{a.m./p.m.}\times\text{Hour} \end{smallmatrix}\right)$. The wiring diagram itself

$$(5.11)$$
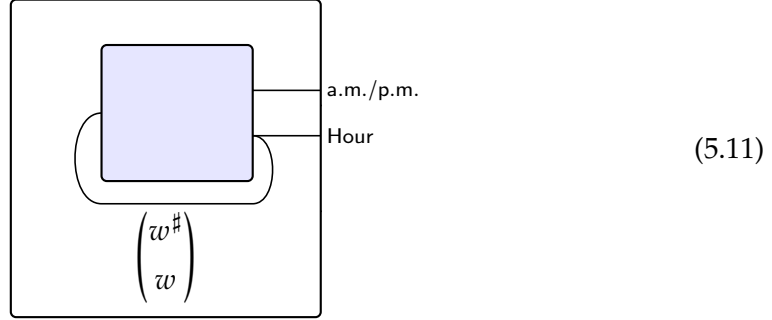
is a lens

$$\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \begin{pmatrix} 1 \times \text{Hour} \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix} \leftrightarrows \begin{pmatrix} 1 \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix}.$$

That is, it is a vertical arrow in the double category $\mathbf{Arena}_{\mathbb{D}\text{ET}}$ of arenas in the deterministic doctrine. The doubly indexed category $\mathbf{Sys}_{\mathbb{D}\text{ET}} : \mathbf{Arena}_{\mathbb{D}\text{ET}} \to \mathbf{Cat}$ furnishes us with a functor

$$\mathbf{Sys}_{\mathbb{D}\text{ET}}\begin{pmatrix} w^\sharp \\ w \end{pmatrix} : \mathbf{Sys}_{\mathbb{D}\text{ET}}\begin{pmatrix} 1 \times \text{Hour} \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix} \to \mathbf{Sys}_{\mathbb{D}\text{ET}}\begin{pmatrix} 1 \\ \text{a.m./p.m.} \times \text{Hour} \end{pmatrix}$$

Despite its formidible name, this functor is just given by composing in $\mathbf{Arena}_{\mathbb{D}\text{ET}}$ with the lens $\begin{pmatrix} w^\sharp \\ w \end{pmatrix}$. We then apply this functor to Meridian $\otimes$ Hour to get the composite system ClockWithDisplay:

$$\text{ClockWithDisplay} = \mathbf{Sys}_{\mathbb{D}\text{ET}}\begin{pmatrix} w^\sharp \\ w \end{pmatrix}(\text{Meridian} \otimes \text{Hour}).$$

This story is mirrored in the behavioral approach to systems theory, except instead of working with the doubly indexed category $\mathbf{Sys}_{\mathbb{D}} : \mathbf{Arena}_{\mathbb{D}} \to \mathbf{Cat}$, we work in the doubly indexed category $\mathbf{Vec} : \mathbf{Matrix} \to \mathbf{Cat}$ — or, rather, the equivalent doubly indexed category $\mathbf{Set}/(-) : \mathbf{Span}(\mathbf{Set}) \to \mathbf{Cat}$.

We begin with the systems Bucket_1 and Bucket_2, both in $\mathbf{Set}/(\mathbb{R}^4)^\mathbb{R}$; that is, Bucket_1 is identified with the map $\text{expose}_{\text{Bucket}_1} : B_{\text{Bucket}_1} \to (\mathbb{R}^4)^\mathbb{R}$, and the same for Bucket_2. We then form the parallel product of these systems, which in this case is given by their cartesian product

$$\text{expose}_{\text{Bucket}_1} \times \text{expose}_{\text{Bucket}_2} : B_{\text{Bucket}_1} \times B_{\text{Bucket}_2} \to (\mathbb{R}^4)^\mathbb{R} \times (\mathbb{R}^4)^\mathbb{R}$$

which is an object of $\mathbf{Set}/((\mathbb{R}^4)^{\mathbb{R}} \times (\mathbb{R}^4)^{\mathbb{R}})$. Now, the wiring diagram
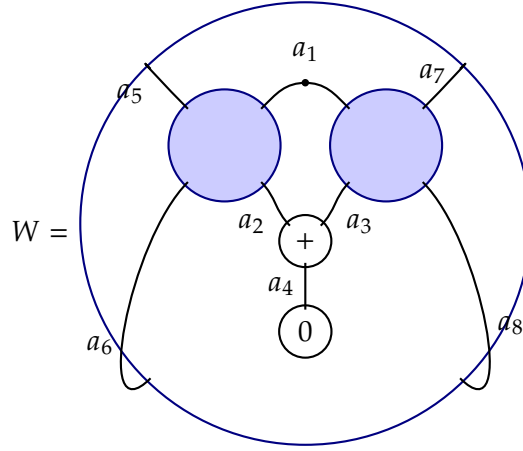
may be described as a span of sets,

which is to say a vertical arrow in $\mathbf{Span}(\mathbf{Set})$. We'll explain how bubble diagrams correspond to spans in a more systematic way in **??**, but for now we can define W as follows:

$$W = \left\{ (a_i)_{1 \leq i \leq 8} : \mathbb{R} \to \mathbb{R}^8 \,\middle|\, \begin{array}{c} a_2 + a_3 = a_4 \\ a_4 = 0 \end{array} \right\} \tag{5.12}$$

$$w_1((a_i)_{1 \leq i \leq 8}) = ((a_6, a_2, a_5, a_1), (a_3, a_8, a_1, a_7))$$
$$w_2((a_i)_{1 \leq i \leq 8}) = (a_6, a_7, a_5, a_8)$$

We then compose Bucket_1 and Bucket_2 into the composite $\mathsf{Buckets}$ by applying $\mathbf{Vec}(W)$ to $\mathsf{Bucket}_1 \times \mathsf{Bucket}_2$:

$$\mathsf{Buckets} = \mathbf{Vec}(W)(\mathsf{Bucket}_1 \times \mathsf{Bucket}_2).$$

This means composing with W in the category of spans. Recall that we can see the map $\mathrm{expose}_{\mathsf{Bucket}_1} \times \mathrm{expose}_{\mathsf{Bucket}_2} : B_{\mathsf{Bucket}_1} \times B_{\mathsf{Bucket}_2} \to (\mathbb{R}^4)^{\mathbb{R}} \times (\mathbb{R}^4)^{\mathbb{R}}$ as a span from 1 to $(\mathbb{R}^4)^{\mathbb{R}} \times (\mathbb{R}^4)^{\mathbb{R}}$. Composing with W therefore means we have the following pullback diagram:

Quite explicitly, this defines $B_{Buckets}$ to be the set

$$
B_{Buckets} = \left\{
\begin{array}{l}
(h_1, f_{11}, f_{12}, p_{11}, p_{12}) : \mathbb{R} \to \mathbb{R}^5 \\
(h_2, f_{21}, f_{22}, p_{21}, p_{22}) : \mathbb{R} \to \mathbb{R}^5 \\
(a_i)_{1 \leq i \leq 8} : \mathbb{R} \to \mathbb{R}^8
\end{array}
\right|
\begin{array}{ll}
\multicolumn{2}{c}{\dfrac{dh_i}{dt} = F_i(h_i, p_{i1}, p_{i2})} \\[2mm]
\multicolumn{2}{c}{f_{i1} = H_{i1}(h_i, p_{i1})} \\[1mm]
\multicolumn{2}{c}{f_{i2} = H_{i2}(h_i, p_{i2})} \\[1mm]
a_2 + a_3 = a_4 & a_4 = 0 \\[1mm]
p_{11} = a_5 & f_{11} = a_6 \\[1mm]
p_{12} = a_1 & f_{12} = a_2 \\[1mm]
p_{21} = a_1 & f_{21} = a_3 \\[1mm]
p_{22} = a_7 & f_{22} = a_8
\end{array}
\right\}
$$

$$(5.13)$$

At first glance, this is quite a bit larger than the definition of $B_{Buckets}$ we gave in **??**. But most of the equations here are setting the fs and ps from each $Bucket_i$ to be equal to the as coming form the wiring diagram W. When the dust has settled, the two definitions are equivalent — which is to say more precisely that they are isomorphic in the category $\mathbf{Vec}((\mathbb{R}^4)^{\mathbb{R}})$.

Exercise 5.4.4. Describe explicitly the isomorphism between the definitions of B_{Bucket} in **??** and **??**. Check that this isomorphism commutes with the two definitions of $expose_{Buckets}$ as well. ◊

A crucial feature of the behavioral approach to systems theory is that constraints on system behaviors are treated at the same level as the systems themselves. Suppose we want to constrain the system Buckets so that the water flows from left to right. That is, we want $f_{11} > 0$ and $f_{22} < 0$. These constraints give rise to a subset C of the set $(\mathbb{R}^2)^{\mathbb{R}}$:

$$C = \{(f_{11}, f_{22}) \mid f_{11} > 0 \text{ and } f_{22} < 0\}. \tag{5.14}$$

We can consider the subset C of $(\mathbb{R}^4)^{\mathbb{R}}$ as an object of $\mathbf{Set}/(\mathbb{R}^4)^{\mathbb{R}}$ by equipping it with the inclusion $C \hookrightarrow (\mathbb{R}^4)^{\mathbb{R}}$. We can bubble up this constraint just like a system (though to emphasize that we are thinking of it as a constraint and not as a system, we will not fill the bubble with blue):
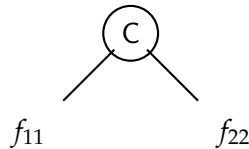
To express the system Buckets constrained so that the inequalities $f_{11} > 0$ and $f_{22} < 0$

hold, we can use another bubble diagram:
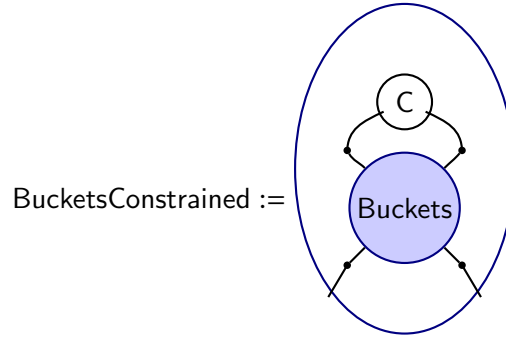
This is a new system with interface $(\mathbb{R}^2)^{\mathbb{R}}$. Suppose we want to ask if this constraint C is sufficient to ensure that the pressures on the pipes (the remaining exposed variables) are within certain bounds $[b_{i0}, b_{i1}]$. We can express these constraints P on pressure as a subset of $(\mathbb{R}^2)^{\mathbb{R}}$:

$$P = \{(p_{11}, p_{22}) \mid b_{10} \leq p_{11} \leq b_{11}, \text{and} b_{20} \leq p_{22} \leq b_{21}\} \hookrightarrow (\mathbb{R}^4)^{\mathbb{R}}.$$

The question of whether the constrained system BucketsConstrained satisfies the constraints P is then the question of whether there is a map $\phi :$ BucketsConstrained $\rightarrow$ P in **Set**$/(\mathbb{R}^4)^{\mathbb{R}}$:

$$\text{BucketsConstrained} \xdashrightarrow{\phi} P$$
$$(\mathbb{R}^2)^{\mathbb{R}}$$

(5.15)

The map ϕ is a function of all the state variables of BucketsConstrained, but the commutativity of **??** says that it must be given by

$$\phi(\ldots, p_{11}, \ldots, p_{22}, \ldots) = (p_{11}, p_{22}).$$

Therefore, the existence of ϕ is the same thing as the *proposition* that p_{11} is between b_{10} and b_{11} and the same for p_{22} — that is, the proposition that BucketsConstrained satisfies the constraint P.

The question of devising such a constraint C (or, even better, a system which implements this constraint) for a system Buckets so that the constrained system admits a map to another constraint P is known as a *control problem*. We will see control problems and their categorical dual, *design problems*, in more detail in **??**.

All of our variables in the previous examples were variables of time. This is very common for behaviors, especially those coming from differential doctrines. Instead of having all our sets be $A^{\mathbb{R}}$ for an A varying in time $t \in \mathbb{R}$, we could bake in this variation into time into our notion of set itself. That is, we can work in a category of *variable sets*, or *sheaves*. In such a category, the set A would already include, implicitly, the variation in time. The sheaf theoretic setting for the behavioral appraoch to systems theory is explored in [SSV16] and [SS19]; we will summarize it in **??**, and show that trajectories in the general differential doctrine $\mathbb{D}$IFF (**??**) land in sheaves over time intervals.

5.4.5   Bubble diagrams as spans in categories of arities

Though we have been describing the behavioral approach to systems theory as taking place within the doubly indexed category $\mathbf{Set}/(-) : \mathbf{Span}(\mathbf{Set}) \to \mathbf{Cat}$, we can do it in any category that allows us to compose spans — namely, and category with finite limits. Just like we had different dynamical system doctrines for lens-based systems, we can see each category with finite limits as a doctrine for the behavioral approach. We will call these *behavioral doctrines*.

Definition 5.4.6. A *doctrine for the behavioral approach* or *behavioral doctrine* is a category $\mathcal{C}$ with finite limits. The doubly indexed category of systems in the behavioral doctrine $\mathcal{C}$ is

$$\mathbf{BSys}_{\mathcal{C}} := \mathcal{C}/(-) : \mathbf{Span}(\mathcal{C}) \to \mathbf{Cat}$$

defined as the vertical slice construction applied to the inclusion $1 : 1 \to \mathbf{Span}(\mathcal{C})$ of the terminal object of $\mathcal{C}$:

$$\mathbf{BSys}_{\mathcal{C}} := \sigma(1 : 1 \to \mathbf{Span}(\mathcal{C})).$$

So far, we have only seen the behavioral doctrine **Set** of sets, but in **??** we will see a behavioral doctrine of sheaves over time intervals. Though we don't as many different examples of behavioral doctrines as we do for dynamical system doctrines, the notion can help us clarify the basic ideas of the behavioral approach: it's all about *spans*, much in the way that dynamical systems doctrines are all about lenses.

All the way back in **??**, we saw that wiring diagrams are lenses in special categories: the free cartesian categories. We needed a cartesian category to describe the notion of lens given in **??**. We can make an analogy here: to describe the behavioral approach to systems theory, we use spans which require finite limits. It stands to reason that we should expect our bubble diagrams for the behavioral approach to be spans in *free finitely complete categories*. We'll see that this is precisely the case.

Before we see a formal definiton of bubble diagram, let's give an informal definition.

Informal Definition 5.4.7. A *bubble diagram* is a diagram which consists of a number of *inner bubbles* drawn within an *outer bubble*, each with some *ports*. There are furthermore *links*, which are drawn as small dots. The bubbles are wired together by connecting to the links:

1. Every port on an inner bubble is wired to a unique link.
2. Every port on the outer bubble is wired to a unique link.

The category of bubble diagrams has as its objects the bubbles and as its morphisms the bubble diagrams. Bubble diagrams are composed by filling the inner bubbles with other bubble diagrams, the erasing the middle layer of bubbles, and coalescing any connected links into a single link.

Bubbles	Bubble Diagrams	Composition by nesting

$$(5.16)$$
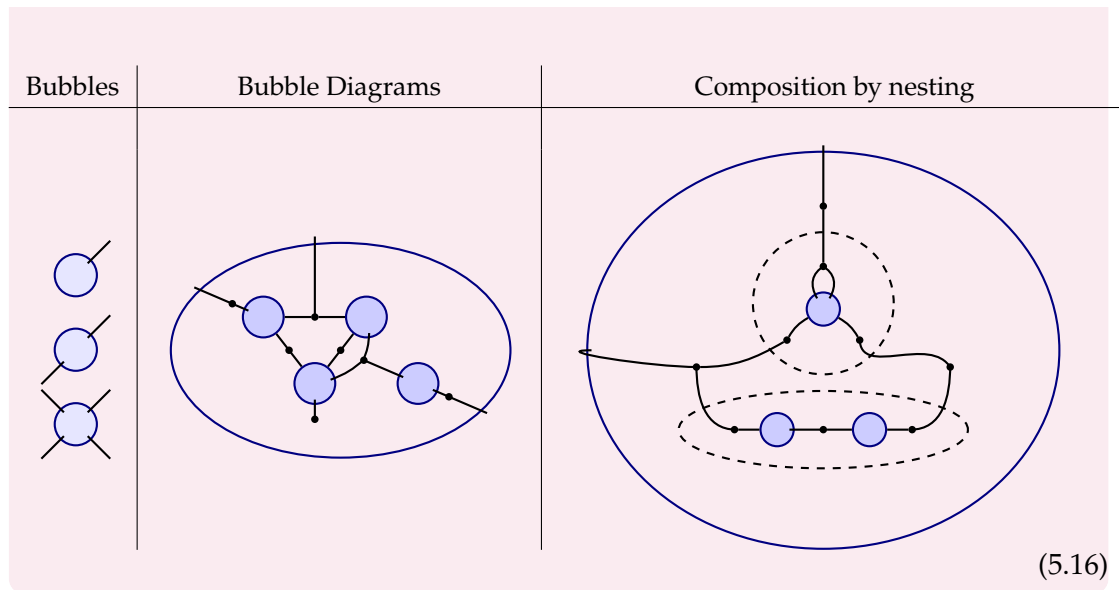
We can formalize these diagrams as spans in the free finitely complete category. Luckily, we already know what the free finitely complete category is; it turns out to be the same as the free cartesian category, the category **Arity** of arities (**??**)!

Recall that the category of arities is equivalent to the opposite of the category of finite sets.

5.4.8 Control and design problems

5.4.9 Behavior types: sheaves on the interval domain

Appendix H

Bibliography

[BCS09]   RF Blute, JRB Cockett, and RAG Seely. "Cartesian differential categories".
          In: *Theory and Applications of Categories* 22.23 (2009), pp. 622–672.

[FS19]   Brendan Fong and David I. Spivak. *An Invitation to Applied Category Theory:
          Seven Sketches in Compositionality*. Cambridge University Press, 2019 (cit. on
          p. 4).

[Jaz21]   David Jaz Myers. "Double Categories of Open Dynamical Systems (Extended
          Abstract)". In: *Electronic Proceedings in Theoretical Computer Science* 333 (Feb.
          2021), pp. 154–167. ISSN: 2075-2180. DOI: 10.4204/eptcs.333.11. URL: http:
          //dx.doi.org/10.4204/EPTCS.333.11.

[Ngo17]   Timothy Ngotiaoco. *Compositionality of the Runge-Kutta Method*. 2017. eprint:
          arXiv:1707.02804.

[Per21]   Paolo Perrone. *Notes on Category Theory with examples from basic mathematics*.
          2021. arXiv: 1912.10642 [math.CT] (cit. on p. 4).

[Spi19]   David I. Spivak. *Generalized Lens Categories via functors $\mathcal{C}^{op} \rightarrow$ Cat*. 2019.
          eprint: arXiv:1908.02202.

[SS19]   Patrick Schultz and David I. Spivak. *Temporal Type Theory: A topos-theoretic
          approach to systems and behavior*. Springer, Birkhäuser, 2019.

[SSV16]   Patrick Schultz, David I Spivak, and Christina Vasilakopoulou. "Dynamical
          systems and sheaves". In: *Applied Categorical Structures* (2016), pp. 1–57.

[VSL15]   Dmitry Vagner, David I. Spivak, and Eugene Lerman. "Algebras of open
          dynamical systems on the operad of wiring diagrams". In: *Theory and Appli-
          cations of Categories* 30 (2015), Paper No. 51, 1793–1822.