NESTING OF DYNAMIC SYSTEMS AND MODE-DEPENDENT NETWORKS

DAVID I. SPIVAK

Abstract. For many networks, the connection pattern varies in time, depending on the changing state, or mode, of the modules in that network. This paper addresses the issue of nesting such mode-dependent networks, whereby a whole network is abstracted as a single module in a larger network. Each module in the network represents a dynamic system, whose input-output behavior includes updating its communicative mode. In this way, the dynamics of the modules controls their connection pattern within the network. This paper provides a formal semantics, using the category-theoretic framework of operads and their algebras, to capture the nesting property of mode-dependent networks and their dynamics. We provide a detailed running example to ground the mathematics.

Contents

1. Introduction 1
2. Background 4
3. Mode-dependent networks 6
4. Dynamic systems on mode-dependent networks 9
Acknowledgments 13
References 13

1. Introduction

E pluribus unum—out of many, one—could be the motto of the mathematical theory of operads. Sometimes multiple things of a certain sort can be integrated, i.e., organized according to established relationships, to form a single new thing of the same sort.

For example consider the economist’s intuition, as explained by R.H. Coase [7]: “An economist thinks of the economic system as being co-ordinated by the price mechanism, and society becomes not an organization but an organism.” One may consider the situation in which a collection of economic agents, say persons, are organized into a firm, or a department within a firm, which itself can be modeled as a single economic agent in a larger context. If we find that this story repeats itself at various scales, it is often a useful exercise to formalize the situation using operads, because doing so constrains the model to be highly self-consistent.

In previous work [17, 20, 23], operads were given that describe various sorts of wiring diagrams and the kinds of information they can convey. A wiring diagram is a fixed graph-like arrangement of modules within a larger module. In this note, we

This project was supported by ONR grant N000141310260, AFOSR grant FA9550-14-1-0031, and NASA grant NNH13ZEA001N-SSAT.
extend these ideas to the case that the arrangement is not fixed in time, but instead varies with respect to the internal states of various modules in the diagram. This idea was inspired by reading the introduction to asynchronous networks written by Mike Field [10].

Consider the picture below:

![Mode-dependent network](image)

It represents a network of two interior boxes $A_1, A_2$ that are set inside an outer interface $B$. We will suppose that each of these boxes is assigned a set of communicative modes, which together will determine the connection pattern of $A_1$ and $A_2$ within $B$. For example, it may be that if the mode of $A_1$ is $\text{foo}$ and the mode of $A_2$ is $\text{bar}$ then connection pattern 1 is formed, whereas if the modes are respectively $\text{baz}$ and $\text{baz}$ then connection pattern 2 is formed; see the pictures below:

![Connection pattern 1](image)

![Connection pattern 2](image)

This describes an operad because the outer box is of the same nature as the inner boxes, i.e., it can itself be a small box within a larger mode-dependent network, and the process of assembly can repeat ad infinitum. In this way, one can recursively build up networks of networks. We can also add dynamics, so that each box houses a dynamic system which not only produces output based on input, but also updates its communicative mode, and thus the connection pattern of the network.

Of course, similar such stories have been considered before; what we do here is formalize the idea using operads. Doing so ensures that no matter how the system is chunked, the rewiring and dynamics are well-defined and consistent. This requirement puts fairly strict constraints on the formalism, as mentioned above. Not any seemingly-workable definition of mode-dependent dynamic systems will actually satisfy the nesting property.

The notion of mode-dependent networks of dynamic systems is a fairly general setup, and it should have multiple applications. For example, we could imagine that a group of robots are being assembled to carry out a more complex job. We decide in advance the channels by which each robot can receive information and the channels by which it can act. We also decide in advance each robot’s set of communicative modes. We arrange our set of robots into something like a hive, which is like a higher-level robot, with its own input and output channels.

To program this hive, we decide in advance how the communicative modes of the member robots will determine the connection pattern of the hive: who’s looking at who, or who’s sending data to who. Perhaps if robot $A$ gets angry at robot $B$, he ceases to send her data; or if robot $C$ knows that it is another robot’s turn to speak, then
C disengages the microphone. These are examples where the robots are bordering on human. More simply, the physical location of a robot could be captured in its communicative mode, allowing us to specify that two robots can interact in the hive only when they are proximate.

Now suppose we model each robot in the hive as a dynamic system, which updates its state depending on incoming data, and which produces output. Then the hive will inherit such a dynamic system model as well: As information comes into the hive, it will be passed around to various robots according to the current connection pattern. Each member robot will deal with this information according to the workings of its own dynamic system, possibly updating its mode in the process. Then it will pass on the results, either to another member robot or out of the hive, as dictated by the hive’s updated communicative mode.

The operadic viewpoint allows one to repeat this story. Putting an interface around a hive of robots makes it a robot, which can be assembled into a larger hive. Note that we do not require that every robot is itself a hive—there is no rule saying that one can indefinitely zoom in—some robots may be “atomic” or “prime”. Said another way, the operadic viewpoint is not analytic but synthetic. We provide a formula for assembling robots into higher-level robots, i.e., for zooming out. This can be done indefinitely, and the proofs in this paper show that the formula for assembly is consistent and independent of ones choice of chunking scheme.

As another example, the radar systems, computers, radios, engines, and humans involved in keeping airplanes from colliding are generally organized according to which of these components are on the ground, which are in the airplanes, and which are in the upper atmosphere. However, these same components can be re-chunked in a logical manner, in terms of perceptive components, processing components, and actuating components. We model these components as dynamic systems whose communication links change based on system state (air traffic control can talk to only certain airplanes when they are within range). In effect, this paper proves that physical chunking and logical chunking will reveal the same behavior.

All this will be made formal below. We refer the reader to [15], [1], or [21] for background on category theory (in decreasing order of difficulty), to [14] for specific background on operads, algebras, and monoidal categories, and to [10] for background on networks. We also refer the reader to wikipedia [24] and the nLab [16] for many useful articles on category theory, and monoidal categories in particular. Other category-theoretic approaches to networks and their dynamics include [8, 9], [3, 4], and [18].

Before we begin, we introduce an example that will run throughout the paper.

**Example 1.1.** Consider an eyeball $E$ that saccades between different positions $P_1, P_2, P_3$ on a page of text. Think of the communicative mode of $E$ is its intention: where on the page it wants to look. This mode determines the connection pattern for boxes below:

![Diagram of connection pattern](1)
The page positions $P_1, P_2$, and $P_3$ each have only one communicative mode. However, they each have a set of possible internal states, as does $E$; the current state of a box determines its behavior. For example, we assume that for each of $P_1, P_2, P_3$, a state is a pair $(\ell, b)$, where $\ell \in L := \{a, b, \ldots, z\}$ is an English letter and $b \in \{\text{light}, \text{dark}\}$ is a brightness-level. The eyeball reads a letter (correctly or not, depending on brightness) from the current position, and then saccades to the next position by changing its communicative mode. The eyeball $E$ will, via a dynamic system, decide how to treat the incoming information, before sending it on to the “outside world” $R$. Of course, thinking of $R$ as the outside world is dependent on our choice of chunking; we can losslessly change the chunking at any time.

This example will be fleshed out and formalized in Examples 2.2, 3.2, 3.5, 4.3, and 4.4.

2. Background

We begin with some notation and basic terms from category theory.

**Notation.** Recall the notion of Grothendieck universe $U$ from [5] or [19]. Roughly, it is just a set of sets with certain properties, which make it convenient as a model of set theory. So if $A \in U$ is an element, we call it a set, and if $\tau: A \to U$ is a function, it just mean that every $a \in A$ is being assigned a set $\tau(a)$. The symbol $\emptyset$ represents the empty set.

Let $\text{Set}$ denote the category of sets (elements of $U$) and functions between them. Let $\text{FinSet} \subseteq \text{Set}$ denote the full subcategory spanned by the finite sets. If $A, B \in \text{Ob} \mathcal{C}$ are objects of a category, we may denote the set of morphisms between them either by $\text{Hom}_\mathcal{C}(A, B)$ or by $\mathcal{C}(A, B)$. If $A \in \text{Ob} \mathcal{C}$ is an object, we may denote the identity morphism on $A$ either by $\text{id}_A$ or simply by $A$. If there is a unique element in $\mathcal{C}(A, B)$, we may denote it $!: A \to B$. For example, there is a unique function $!: \emptyset \to A$ for any set $A$.

**Definition 2.1.** Fix a Grothendieck universe $U$. The category of typed finite sets, denoted $\text{TFS}$, is defined as follows. An object in $\text{TFS}$ is a finite set of sets, $\text{Ob} \text{TFS} := \{(A, \tau) \mid A \in \text{Ob} \text{FinSet}, \tau: A \to U\}$. We call $\tau$ the typing function, and for any element $a \in A$, we call the set $\tau(a)$ its type. If the typing function $\tau$ is clear from context, we may abuse notation and denote $(A, \tau)$ simply by $A$.

A morphism $q: (A, \tau) \to (A', \tau')$ in $\text{TFS}$ consists of a function $q: A \to A'$ that makes the following diagram of finite sets commute:

$$
\begin{array}{ccc}
A & \xrightarrow{q} & A' \\
\downarrow{\tau} & & \downarrow{\tau'} \\
U & & U
\end{array}
$$

We refer to the morphisms of $\text{TFS}$ as typed functions. The category $\text{TFS}$ has a monoidal structure $(A, \tau) \sqcup (A', \tau')$, given by disjoint union of underlying sets and the induced function $A \sqcup A' \to U$.

Given a typed finite set $A = (A, \tau)$, we denote by $(A, \tau)$, or simply by $\overline{A}$, the cartesian product

$$
\overline{A} := \prod_{a \in A} \tau(a).
$$
We call the set $A$ the dependent product of $A$. Taking dependent products is a functor $\text{TFS}^{\text{op}} \to \text{Set}$, i.e., a morphism $q: (A, \tau) \to (A', \tau')$ induces a function $\overline{q}: \overline{(A', \tau')} \to \overline{(A, \tau)}$.

Note that if $A$ and $B$ are typed finite sets, then there is an isomorphism, $A \sqcup B \cong A \times B$.

**Example 2.2.** We give three examples of typed finite sets and their dependent products. Recall that an element of $U$ is just a set, such as the set $\mathbb{Z}$ of integers or the set $L := \{a, b, \ldots, z\}$ of letters.

1. If $A = \{1, 2, 3\}$ and $\tau: A \to U$ is given by $\tau(1) = \tau(2) = \tau(3) = \mathbb{Z}$, then the dependent product is $(A, \tau) \cong \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$.

2. Let $\{\ast\}$ be an arbitrary one-element set. Consider the typed finite set $\{\ast\} \xrightarrow{\tau} U$, sending $\ast$ to the set $\tau(\ast) \cong \{a, b, \ldots, z\}$. Then the dependent product is simply $(\{\ast\}, \tau) = \{a, b, \ldots, z\}$.

3. Consider the unique function $!:\emptyset \to U$. Its dependent product is $\emptyset \cong \{\ast\}$, because the empty product is the singleton set.

The latter two cases will be relevant for our running example.

Before leaving this section, we explain a bit about the role of operads and monoidal categories in this paper, and we give a brief guide to related literature.

**Remark 2.3.** While we speak of operads throughout this paper, the formal mathematics will generally be written using the framework of symmetric monoidal categories. We are implicitly referring to a functor $\mathcal{O}: \text{SMC} \to \text{Oprd}$ from the category of symmetric monoidal categories and lax functors to the category of operads—by which we mean symmetric colored operads—and operad functors. If $(\mathcal{C}, \otimes)$ is a symmetric monoidal category then the operad $\mathcal{O}_\mathcal{C}$ has the same objects as $\mathcal{C}$, and a morphism $(X_1, \ldots, X_n) \to Y$ in $\mathcal{O}_\mathcal{C}$ is defined as a morphism $X_1 \otimes \cdots \otimes X_n \to Y$ in $\mathcal{C}$. See [14, Example 2.1.3].

There are two reasons we speak in terms of operads rather than monoidal categories. The first is that the overarching idea here is operadic. The operad $\mathcal{O}_{\text{MDN}}$ that we work with in this paper may underlie a monoidal category, MDN, but some close cousins do not. I think of the fact that our operad underlies a monoidal category as more of a coincidence than anything else. The second reason is that the concept I want to get across is best viewed as operadic. That is, we are talking about modularity—one thing being built from many—and, to me, the morphisms found in operads seems to model this idea best.

That said, when an operad does underlie a monoidal category, the formulas are far easier to write in the language of monoidal categories, because doing so avoids subscripts. That is the motivation for using monoidal categories throughout this paper.

Note that monoidal categories are used in [3] and [4] to study networks, and some confusion may arise in comparing their work to our own, unless care is taken. One way to see the difference is that we are focusing on the fact that networks nest, i.e., that multiple dynamic systems can be gathered into a network that is itself a single dynamic system. In this way, our work more closely follows the intention of [6] or [11]. However, neither of these references uses category theory, though the latter mentions it as a plausible approach.
A potentially confusing point between our work and that of [3] and [4] is a certain level-shift. Namely, their objects are semantically equivalent to our labels; their morphisms are semantically equivalent to our objects; and their reduction laws are semantically equivalent to our morphisms. This level-shifting correspondence is similar to one appearing in Baez and Dolan’s slice construction [2], but the one here appears to be a bit different. These issues will be explained in an upcoming paper, [22].

3. MODE-DEPENDENT NETWORKS

**Definition 3.1.** A signed finite set is a pair $X = (X^{\text{in}}, X^{\text{out}})$, where $X^{\text{in}}$ and $X^{\text{out}}$ are finite sets. We define a box to be a signed finite set $X$ equipped with typing functions $\phi^{\text{in}} : X^{\text{in}} \rightarrow U$ and $\phi^{\text{out}} : X^{\text{out}} \rightarrow U$. Each element of $X^{\text{in}} \sqcup X^{\text{out}}$ will be called a port.

If $X$ and $Y$ are boxes, we define a wiring diagram, denoted $\delta : X \rightarrow Y$, to be a pair of typed functions $\phi = (\phi^{\text{in}}, \phi^{\text{out}}) :$

$$
\begin{align*}
\phi^{\text{in}} &: X^{\text{in}} \rightarrow Y^{\text{in}} \sqcup X^{\text{out}} \\
\phi^{\text{out}} &: Y^{\text{out}} \rightarrow X^{\text{out}}
\end{align*}
$$

We define the composition formula for wiring diagrams $\delta : X \rightarrow Y$ and $\psi : Y \rightarrow Z$ as the dotted arrows below, the indicated compositions in TFS:

$$
\begin{align*}
X^{\text{in}} \quad \xrightarrow{\phi^{\text{in}}} \quad Z^{\text{in}} \sqcup X^{\text{out}} \\
Y^{\text{in}} \sqcup X^{\text{out}} \quad \xrightarrow{\psi^{\text{in}} \sqcup \phi^{\text{out}}} \quad Z^{\text{in}} \sqcup Y^{\text{out}} \sqcup X^{\text{out}} \quad \xrightarrow{\psi^{\text{out}}} \quad Z^{\text{out}} \sqcup X^{\text{out}} \\
\end{align*}
$$

This defines a category, which we call the category of wiring diagrams, and denote $\text{WD}$. It has boxes as objects and wiring diagrams as morphisms. It has a symmetric monoidal structure defined by disjoint union, which we denote by $\sqcup : \text{WD} \times \text{WD} \rightarrow \text{WD}$. We denote the operad underlying $\text{WD}$ by $\mathcal{O}_{\text{WD}}$.

A morphism in the operad $\mathcal{O}_{\text{WD}}$ is called a wiring diagram because it can be drawn with boxes and directed wires (or channels) connecting them, as follows:

$$
\varphi : (A_1, A_2, A_3) \rightarrow B
$$

The functions (3) determine which ports are wired together. See also [23, Definition 3.5].

**Example 3.2.** Consider the box $P = (\emptyset, \{p\}, \tau)$, where $\tau(p) = L = \{a, b, \ldots, z\}$. It has only one part, $p$. We often suppress the typing $\tau$, in which case we would write $P^{\text{in}} = \emptyset$ and $P^{\text{out}} = \{p\}$. Thus we might draw this box in any of following ways:

$P_1 \xrightarrow{p^{\text{in}}} P_2 \xrightarrow{p^{\text{out}}} P_3$

Suppose $E = (\{e\}, \{e'\})$ is another object in $\text{WD}$, say with $\tau$ as its typing functions, and $R \cong P_1 \sqcup P_2 \sqcup P_3 \sqcup E$ and $Y = R$. Then we can
define a morphism \( \varphi: (P_1, P_2, P_3, E) \to R \) in \( \mathcal{O}_{\text{WD}} \) as a morphism \( X \to Y \) in WD. Namely, it is a pair of typed functions having the form
\[
\varphi^{\text{in}}: X^{\text{in}} \to Y^{\text{in}} \uplus X^{\text{out}} \\
\varphi^{\text{out}}: Y^{\text{out}} \to X^{\text{out}}
\]
Choosing these two maps, \( \varphi^{\text{in}} \) and \( \varphi^{\text{out}} \), is the same as choosing a wiring diagram.

Suppose we want to represent the following wiring diagram.

\[
\begin{array}{ccc}
P_1 & \xrightarrow{e} & P_2 \\
\downarrow & & \downarrow \\
P_3 & \xrightarrow{e'} & P_1 \\
\end{array}
\]

Note that we have \( X^{\text{in}} = \{e\} \), \( X^{\text{out}} = \{p_1, p_2, p_3, e'\} \), \( Y^{\text{in}} = \emptyset \), and \( Y^{\text{out}} = \{r\} \). Because the wiring diagram shows that \( e \) is wired to \( p_3 \), we have \( \varphi^{\text{in}}(e) = p_3 \). For similar reasons, we have \( \varphi^{\text{out}}(r) = e' \).

For use later, we record the following dependent products, as explained in Example 2.2:
\[
\begin{align*}
R^{\text{in}} &= P^{\text{in}} = \{\ast\} & \text{and} & \\
E^{\text{out}} &= E^{\text{in}} = E^{\text{out}} = R^{\text{in}} = L.
\end{align*}
\]

**Definition 3.3.** We define a symmetric monoidal category, called the category of **mode-dependent networks** and denoted \( \text{MDN} \), as follows. An object in \( \text{MDN} \), called a **modal box**, is a pair \((M, X)\), where \( M \in \mathcal{U} \) is a set and \( X \in \text{ObWD} \) is box. We call \( M \) the set of communicative modes, or just **mode set** for short. For morphisms, we put:
\[
\text{Hom}_{\text{MDN}}((M, X), (N, Y)) = \{(\epsilon, \sigma) \mid \epsilon: M \to \text{WD}(X, Y), \quad \sigma: M \to N\}.
\]
We call a morphism \((\epsilon, \sigma) \in \text{Hom}_{\text{MDN}}((M, X), (N, Y))\) a **mode-dependent network**, and we call \( \epsilon \) the **event map**, following [10]. In the special case that \( \epsilon \) factors through the unique function \( M \to \{\ast\} \), i.e., if the wiring diagram does not change with the mode, then we say the network is **mode-independent**.

Given two composable morphisms (mode-dependent networks)
\[
\begin{array}{ccc}
M_0 & \xrightarrow{\sigma_0} & M_1 \\
\downarrow & & \downarrow \\
\text{WD}(X_0, X_1) & \xrightarrow{\epsilon_1} & \text{WD}(X_1, X_2)
\end{array}
\]
the composition formula is given by \((\epsilon_1, \sigma_1) \circ_{\text{MDN}} (\epsilon_0, \sigma_0) := (\epsilon, \sigma)\), where \( \sigma := \sigma_1 \circ \sigma_0 \) is the composition of functions, and where \( \epsilon: M_0 \to \text{WD}(X_0, X_2) \) is the following composition in WD:
\[
\epsilon(m_0) := \epsilon_1 \sigma_0(m_0) \circ \epsilon_0(m_0).
\]
The monoidal structure on \( \text{MDN} \) is given on objects by
\[
(M, X) \otimes (N, Y) := (M \times N, X \uplus Y),
\]
and similarly on morphisms. The unit object in \( \text{SWD} \) is \((\{\ast\}, \emptyset)\), where \( \{\ast\} \) is the singleton set, and \( \emptyset \) is the monoidal unit of \( \text{WD} \).
Remark 3.4. In [10], attention is paid to the image of the event map $\epsilon: M \to \text{WD}(X,Y)$, which is denoted $\mathcal{A} \subseteq \text{WD}(X,Y)$. At certain points in that discussion, $\mathcal{A}$ is chosen independently of $\epsilon$, but at others it is assumed to be the image of $\epsilon$, see [10, Remark 4.10(2)]. In the above definition, we could have defined a morphism in MDN to consist of a triple $(\mathcal{A},\epsilon,\sigma)$, where $\mathcal{A} \subseteq \text{WD}(X,Y)$ and $\epsilon: M \to \mathcal{A}$. In this case, composition would also involve composing the various $\mathcal{A}$’s, but this is straightforward. However, specifying $\mathcal{A}$ independently seemed superfluous here, especially given Field’s remark.

Example 3.5. We continue with Example 3.2; in particular, let $P,E,R$ be as defined there. We will define a mode-dependent network $(\epsilon,\sigma)$ as in (1).

Let $M_P := \{\ast\}$, and let $M_E := \{1,2,3\}$; the former has only one communicative mode, the latter has three. Define $(M,X) := (M_P,P) \otimes (M_P,P) \otimes (M_P,P) \otimes (M_E,E)$. Then $X$ is as in Example 3.2 and $M = \{1,2,3\}$ has three modes. Let $Y := R$ and $N := \{\text{working, done}\}$.

Our goal is to define a mode-dependent network $(\epsilon,\sigma): (M,X) \to (N,Y)$. For its event map $\epsilon: M \to \text{WD}(X,Y)$ we must provide a wiring diagram $\epsilon_i := \epsilon(i)$ for each $i \in M = \{1,2,3\}$. Put $\epsilon_i^{\text{out}}(r) := \epsilon'$ for each $i$, meaning that $r$ and $\epsilon'$ will be wired together regardless of the mode. On the other hand, put $\epsilon_i^{\text{out}}(e) = p_i$, meaning that $e$ connects to either $p_1$, $p_2$, or $p_3$ depending on its mode. See Diagram (5).

To complete the definition of our mode-dependent network, we just need a function $\sigma: M \to N$. Define $\sigma(1) = \sigma(2) := \text{working}$ and $\sigma(3) := \text{done}$. The larger object $Y$ has mode working unless the eyeball is at position 3, in which case it has mode done.

Proposition 3.6. Every wiring diagram is canonically a mode-independent network in the following sense: There is a strong monoidal functor

$$I: \text{WD} \to \text{MDN}$$

given by $I(X) := (X,\{\ast\})$, where $\{\ast\}$ is any choice of singleton set.

Proof. This is straightforward. A wiring diagram $\varphi \in \text{WD}(X,Y)$ is sent to $I(\varphi) := (\epsilon_{\varphi},\{\ast\}) \in \text{MDN}(I(X),I(Y))$, where $\epsilon_{\varphi}: \{\ast\} \to \text{WD}(X,Y)$ picks out $\varphi$, and $\{\ast\}$ is the identity function on the singleton set. This network is mode-independent because it has only one mode. It is easy to check that $I$ is a strong monoidal functor. \hfill \Box

3.1. Technical lemmas. We end this section with two technical lemmas which will be useful later, but which may be safely skipped on a first reading.

Lemma 3.7. Suppose given a pair of composable morphisms $i: X \to Y$ and $j: Y \to Z$ in WD. Then the dependent products of $(\psi \circ \varphi)^{\text{in}}$ and $(\psi \circ \varphi)^{\text{out}}$ are given by the formulas:

$$\frac{(\psi \circ \varphi)^{\text{in}}(z,x) = \varphi^{\text{in}}(\psi^{\text{in}}(z,\varphi^{\text{out}}(x)),x)}{(\psi \circ \varphi)^{\text{out}}(x) = \varphi^{\text{out}}(\varphi^{\text{out}}(x))}$$

Proof. This is a straightforward application of dependent product to the composition formula in (4). \hfill \Box

Given any mode-dependent network $(\epsilon,\sigma): (M,X) \to (N,Y)$ in MDN, it is often convenient to factor $\epsilon: M \to \text{WD}(X,Y)$ into two parts, $\epsilon^{\text{in}}: M \to \text{TFS}(X^{\text{in}},Y^{\text{in}} \sqcup X^{\text{out}})$ and $\epsilon^{\text{out}}: M \to \text{TFS}(Y^{\text{out}},X^{\text{out}})$. These are simply the compositions of $\epsilon$ with corresponding projections of

$$\text{WD}(X,Y) = \text{TFS}(X^{\text{in}},Y^{\text{in}} \sqcup X^{\text{out}}) \times \text{TFS}(Y^{\text{out}},Y^{\text{in}})$$
as in (3). Composing with the dependent product functor $\text{TFS}^{\text{op}} \to \text{Set}$ from Definition 2.1, we obtain the following convenient functions:

$$
\overline{\epsilon}^{\text{in}} : M \to \text{Set}(\overline{Y}^{\text{in}} \times \overline{X}^{\text{out}}, \overline{X}^{\text{in}})
$$

$$
\overline{\epsilon}^{\text{out}} : M \to \text{Set}(\overline{X}^{\text{out}}, \overline{Y}^{\text{out}}).
$$

For any communicative mode, these functions specify how information will travel within the network (\(\overline{\epsilon}^{\text{in}}\)) and how information will be exported from the network (\(\overline{\epsilon}^{\text{out}}\)).

**Lemma 3.8.** Suppose given two composable morphisms in MDN:

$$
\begin{array}{c}
M_0 \xrightarrow{\sigma_0} M_1 \xrightarrow{\sigma_1} M_2 \\
\text{WD}(X_0, X_1) \xrightarrow{\epsilon_0} \text{WD}(X_1, X_2)
\end{array}
$$

and let \((\epsilon, \sigma) = (\epsilon_1, \sigma_1) \circ (\epsilon_0, \sigma_0)\) be their composition as in Definition 3.3. For any \(m_0 \in M_0\), let \(m_1 = \sigma_0(m_0)\). Then the functions \(\overline{\epsilon}^{\text{in}}(m_0) : \overline{X}^{\text{in}}_2 \times \overline{X}^{\text{out}}_0 \to \overline{X}^{\text{in}}_0\) and \(\overline{\epsilon}^{\text{out}}(m_0) : \overline{X}^{\text{out}}_0 \to \overline{X}^{\text{out}}_2\) are given by the formulas:

$$
\overline{\epsilon}^{\text{in}}(m_0)(x_2, x_0) = \overline{\epsilon}^{\text{in}}_0(m_0)\left(\overline{\epsilon}^{\text{in}}_1(m_1)(x_2, \overline{\epsilon}^{\text{out}}_0(m_0)(x_0)), x_0\right)
$$

$$
\overline{\epsilon}^{\text{out}}(m_0)(x_0) = \overline{\epsilon}^{\text{out}}_1(m_1)\left(\overline{\epsilon}^{\text{out}}_0(m_0)(x_0)\right)
$$

**Proof.** By Definition 3.3, we have \(\epsilon(m_0) := \epsilon_1(m_1) \circ \epsilon_0(m_0)\). The formulas above are found by simply restating Lemma 3.7 in the case \(\psi := \epsilon_1(m_1)\) and \(\varphi := \epsilon_0(m_0)\). \(\square\)

### 4. Dynamic systems on mode-dependent networks

The purpose of Field’s paper [10] is to discuss asynchronous dynamic systems (ADSs). I have not yet understood (ADSs) well enough to formalize them as an algebra on the operad MDN, which is itself inspired by Field’s work. We will instead discuss an algebra, i.e., a lax functor \(P : \text{MDN} \to \text{Set}\), of synchronous discrete dynamic systems. First we will supply the data that defines \(P\), summarizing in Definition 4.1, and then we prove that it satisfies the conditions of being an algebra in Proposition 4.5.

Let \((M, X) \in \text{Ob} \text{MDN}\) be a modal box, where \(M \in \text{U}\) is a mode set and \(X = (X^{\text{in}}, X^{\text{out}})\) is a box. Recall from Definition 2.1 the notation \(X^{\text{in}}\) and \(X^{\text{out}}\) for the dependent products. We define \(P(M, X)\) as

$$
P(M, X) := \left\{(S, q, f^{\text{in}}, f^{\text{out}}) \mid S \in \text{U}, \ q : S \to M, \ f^{\text{in}} : X^{\text{in}} \times S \to S, \ f^{\text{out}} : S \to X^{\text{out}}\right\}
$$

That is, an element of \(P(M, X)\) is a 4-tuple:

- \(S \in \text{U}\) is a set, called the state set
- \(q : S \to M\) is a function, called the underlying mode function
- \(f^{\text{in}} : X^{\text{in}} \times S \to S\) is a function, called the state update function, and
- \(f^{\text{out}} : S \to X^{\text{out}}\) is a function, called the readout function.

We may denote an element of \(P(M, X)\) simply by \((S, q, f)\), and we call \(f = (f^{\text{in}}, f^{\text{out}})\) an open dynamic system with state set \(S\), following [23]. The whole 3-tuple \((S, q, f)\) will be called a modal dynamic system.
For the lax monoidal structure, one coherence map, \( P(M, X) \times P(N, Y) \to P(M \times N, X \sqcup Y) \), is given by cartesian products

\[
\begin{align*}
S_{X \times Y} & := S_X \times S_Y \\
q_{X \times Y} & := q_X \times q_Y \\
f_{X \times Y} & := f_X \times f_Y
\end{align*}
\]

Note the isomorphism (2). The other coherence map, \( \{\ast\} \to P(\{\ast\}, \emptyset) \), is the element \( (\{\ast\}, 1, 1, 1) \).

Given a morphism \((\epsilon, \sigma): (M, X) \to (N, Y)\) in \( \text{MDN} \), we need a function \( P(\epsilon, \sigma): P(M, X) \to P(N, Y) \). For an arbitrary modal dynamic system \((S, q, f) \in P(M, X)\), we define

\[
P(\epsilon, \sigma)(S, q, f) := (S, r, g).
\]

Here, \( S \) is unchanged, \( r \) is the composite \( S \xrightarrow{q} M \xrightarrow{\sigma} N \), and \( g^{\text{in}} \) and \( g^{\text{out}} \) are the dotted arrows below, the indicated compositions in \( \text{Set} \):

(11) \[
\begin{array}{ccc}
\overline{Y}^{\text{in}} \times S & \xrightarrow{\overline{Y}^{\text{in}} \times S \times q \times S} & \overline{Y}^{\text{in}} \times S \times M \times S \\
\downarrow g^{\text{in}} & & \downarrow \overline{Y}^{\text{in}} \times f^{\text{out}} \times \overline{X}^{\text{in}} \times S \\
S & \xleftarrow{f^{\text{in}}} & \overline{X}^{\text{in}} \times S
\end{array}
\]

(12) \[
\begin{array}{ccc}
S \times q & \rightarrow & S \times M \\
\downarrow g^{\text{out}} & & \downarrow f^{\text{out}} \times \overline{X}^{\text{out}} \\
Y^{\text{out}} & \xleftarrow{\overline{X}^{\text{out}} \times \overline{Y}^{\text{out}}} & \overline{X}^{\text{out}} \times \text{Set}(\overline{X}^{\text{out}}, Y^{\text{out}})
\end{array}
\]

See Remark 4.2 for a restatement of these diagrams as in-line formulas.

**Definition 4.1.** We define \( P \) as the data (8), (9), and (10) above. So far they are only data; we will show that they constitute a lax monoidal functor \( P: \text{MDN} \to \text{Set} \) in Proposition 4.5.

**Remark 4.2.** We can restate diagrams (11) and (12), which together make up the bulk of (10), in equation form. Suppose given \((\epsilon, \sigma): (M, X) \to (N, Y)\) and \((S, q, f) \in P(M, X)\), where \( q: S \to M \). Then the formula for \( g^{\text{in}}: \overline{Y}^{\text{in}} \times S \to S \), on an element \((y, s) \in \overline{Y}^{\text{in}} \times S\), and the formula for \( g^{\text{out}}: S \to \overline{Y}^{\text{out}}\), on an element \( s \in S \), are given as follows:

\[
\begin{align*}
g^{\text{in}}(y, s) &= f^{\text{in}}(\overline{e}^{\text{in}}(qs)(y, f^{\text{out}}(s)), s) \\
g^{\text{out}}(s) &= \overline{e}^{\text{out}}(qs)(f^{\text{out}}(s)).
\end{align*}
\]
Example 4.3. We continue with our running example, redrawing the picture for the reader’s convenience.

We will give examples of modal dynamic systems, i.e., we will give elements of the sets \( P(P, M_P) \) and \( P(E, M_E) \). Let the boxes \( P, E \) and the mode sets \( M_P = \{ * \} \) and \( M_E = \{ 1, 2, 3 \} \) be as in Example 3.5, and let \( L = \{ a, b, \ldots, z \} \) be the set of letters.

First we need state sets, so let’s define \( S_E := L \times M_E \) and \( S_P := L \times B \), where \( B := \{ \text{dark}, \text{light} \} \) represents the brightness-level. There is a unique function \( q_P : S_P \to M_P \), and we define \( q_E : S_E \to M_E \) to be the second projection. We just need to define open dynamic systems \( f_P \) and \( f_E \).

We have that \( P^\text{in} = \emptyset \) so \( P^\text{in} \cong \{ * \} \). Let’s put \( f_P^\text{in} := \text{id}_{S_P} \), so that the state (the letter and brightness) of that position is unchanging. We have \( P^\text{out} = L \) so we define \( f_P^\text{out} : S_P \to P^\text{out} \) by

\[
f_P^\text{out}(\ell, b) := \begin{cases} \ell & \text{if } b = \text{light} \\ z & \text{if } b = \text{dark} \end{cases}
\]

In other words, the position’s output will be \( z \) if it is too dark to read. (This is a bit implausible, but the example is just meant to give the basic idea.)

To define \( f_E^\text{in} : \overline{E} \times S_E \to S_E \), we need to assign a new state in the arbitrary case that \( E \) is reading \( \ell' \in \overline{E} \) and its current state is \((\ell, i) \in S\). If \( i \in \{ 1, 2, 3 \} \), let \( i + 1 \) denote the next number in circular \((\text{mod } 3)\) sequence. Then we can define

\[
f_E^\text{in}(\ell', \ell, i) := \begin{cases} (\ell', i + 1) & \text{if } \ell' \neq z \\ (\ell, i + 1) & \text{if } \ell' = z. \end{cases}
\]

The interpretation is that \( E \) updates its state to what it just read and “intends” to saccade forward, unless it reads a \( z \) in which case it does not update its state. We define \( f_E^\text{out}(\ell, i) = \ell \), the interpretation being that the last letter read is sent on, to be output of the network.

At this point we have given two examples of modal dynamic systems, namely \((S_P, q_P, f_P) \in P(P, M_P)\) and \((S_E, q_E, f_E) \in P(E, M_E)\).

Example 4.4. We continue with Example 4.3, the latest episode in our running example. Here we prepare the reader to go through Diagrams (11) and (12).

Let \((X, M) := (P, M_P) \otimes (P, M_P) \otimes (P, M_P) \otimes (E, M_E)\), as in (7), and let \((Y, N) := (R, M_R)\). Note that \( Y^\text{in} \cong \{ * \} \) because \( Y = R \) has no inputs. Suppose we have a mode-dependent network \((\epsilon, \sigma) : (X, M) \to (Y, N)\), as in Example 3.5. Suppose we also have modal dynamic systems \((S_P, q_P, f_P)\) and \((S_E, q_E, f_E)\), on \( P \) and \( E \) respectively, as in Example 4.3. We want to enable the reader to understand how these assemble to give a modal dynamic system \((S, r, g) := P(\epsilon, \sigma)(S, q, f)\) on \( R \).
The first step is to obtain a modal dynamic system on $X$ from those on $P$ and $E$ by using the coherence map (9) for $\mathcal{P}$. That is, we define $(S,q,f)$ by
\[
S := S_P \times S_P \times S_P \times S_E
\]
\[
q := q_P \times q_P \times q_P \times q_E
\]
\[
f_{\text{in}} := f_{\text{in}}^P \times f_{\text{in}}^P \times f_{\text{in}}^P \times f_{\text{in}}^E
\]
\[
f_{\text{out}} := f_{\text{out}}^P \times f_{\text{out}}^P \times f_{\text{out}}^P \times f_{\text{out}}^E
\]

We now have the state set $S$ for our modal dynamic system $(S,r,g)$, and we can put $r = \sigma \circ q$ as in Definition 4.1. It remains to find $g_{\text{in}}$ and $g_{\text{out}}$ by working step-by-step through Diagrams 11 and 12. We leave this to the reader.

**Proposition 4.5.** With the data given in Definition 4.1, the map $\mathcal{P}: \text{MDN} \to \text{Set}$ is a lax functor.

**Proof.** It is clear that the coherence maps satisfy the necessary unitality and associativity properties, because they are just given by cartesian products. So it remains to show that $\mathcal{P}$ is functorial, i.e., that it commutes with composition.

Suppose given morphisms
\[
(M_0, X_0) \xrightarrow{(\epsilon_0, \sigma_0)} (M_1, X_1) \xrightarrow{(\epsilon_1, \sigma_1)} (M_2, X_2)
\]
in $\text{MDN}$, and let $(\epsilon, \sigma_1 \circ \sigma_0) = ((\epsilon_1, \sigma_1) \circ (\epsilon_0, \sigma_0))$ be the composite as in (6). Suppose that $(S,q_0,f_0) \in \mathcal{P}(M_0, X_0)$ is an arbitrary element. For notational convenience, define
\[
(S,q_1,f_1) := \mathcal{P}(\epsilon_0, \sigma_0)(S,q_0,f_0),
\]
\[
(S,q_2,f_2) := \mathcal{P}(\epsilon_1, \sigma_1)(S,q_1,f_1),
\]
\[
(S,q_2',f_2') := \mathcal{P}(\epsilon, \sigma_1 \circ \sigma_0)(S,q_0,f_0),
\]
see (10). We want to show that $(S,q_2,f_2) \cong (S,q_2',f_2')$. It is obvious that
\[
q_2 = q_2' = \sigma_1 \circ \sigma_0 \circ q_0,
\]
so it only remains to show that $f_2 = f_2'$, i.e., that $f_{\text{in}}^2 = f_{\text{in}}'^2$ and $f_{\text{out}}^2 = f_{\text{out}}'^2$.

The first desired equation $f_{\text{in}}^2 = f_{\text{in}}'^2$ is between functions $\overline{X_{\text{in}}^2} \times S \to S$, so we choose an element $(x_2,s) \in \overline{X_{\text{in}}^2} \times S$. Let $m_0 := q_0(s)$ and $m_1 \equiv q_1(s) = \sigma_0(q_0(s))$. Using Remark 4.2, we have:
\[
f_{\text{in}}^2(x_2,s) = f_{\text{in}}^2\left(\overline{\epsilon_{\text{in}}}(m_1)(x_2,f_{\text{out}}^1(s)),s\right)
\]
\[
= f_{\text{in}}^0\left(\overline{\epsilon_{\text{in}}}(m_0)\overline{\epsilon_{\text{in}}}(m_1)(x_2,f_{\text{out}}^1(s)),f_{\text{out}}^0(s),s\right)
\]
\[
= f_{\text{in}}^0\left(\overline{\epsilon_{\text{in}}}(m_0)\overline{\epsilon_{\text{in}}}(m_1)\overline{\epsilon_{\text{in}}}(m_0)(f_{\text{out}}^0(s)),f_{\text{out}}^0(s),s\right)
\]
\[
= f_{\text{in}}^0\left(\overline{\epsilon_{\text{in}}}(m_0)(x_2,f_{\text{out}}^0(s)),s\right) = f_{\text{in}}'^2(x_2,s)
\]
where the fourth equality follows from Lemma 3.8.

The second desired equation $f_{\text{out}}^2 = f_{\text{out}}'^2$ is between functions $S \to \overline{X_{\text{out}}^2}$, so we choose an element $s \in S$. Again let $m_0 := q_0(s)$ and $m_1 := q_1(s)$. Using Remark 4.2,
we have:

\[
\begin{align*}
\mathcal{f}_{\text{out}}^2(s) &= \epsilon_{\text{out}}(m_1)(\mathcal{f}_{\text{out}}(s)) \\
&= \epsilon_{\text{out}}(m_1)(\epsilon_{\text{out}}(m_0)(\mathcal{f}_{\text{out}}(s))) \\
&= \epsilon_{\text{out}}(m_0)(\mathcal{f}_{\text{out}}(s)) = \mathcal{f}_{\text{out}}^2(s),
\end{align*}
\]

where the third equality follows from Lemma 3.8. This concludes the proof. □

Acknowledgments

Thanks go to Eugene Lerman for many useful and interesting discussions, and for pointing me to the work of Mike Field, which made this paper possible. I also appreciate many wonderful conversations with Dylan Rupel, who had the insight that internal states could be different than communicative modes. Thanks also to Kevin Schweiker and Srivatsan Varadarajan for explaining the application to the national air space, as I hoped to convey in the introduction.

References

[1] Awodey, S. (2010) Category theory. Second edition. Oxford Logic Guides, 52. Oxford University Press, Oxford.
[2] Baez, J.C.; Dolan, J. (1998) “Higher-Dimensional Algebra III: n-Categories and the Algebra of Opetopes”. Adv. Math. 135, no. 2. Available online: http://arxiv.org/abs/q-alg/9702014.
[3] Baez, J.C.; Erbele, J. (2014) “Categories in Control”. Available online: http://arxiv.org/abs/1405.6881.
[4] Baez, J.C.; Fong, B. (2014) “Quantum Techniques for Studying Equilibrium in Reaction Networks”. Journal of Complex Networks. Available online: http://arxiv.org/abs/1306.3451.
[5] Bourbaki, N. (1972) “Théorie des topos et cohomologie étale des schémas. Tome 1: Théorie des topos,” Séminaire de Géométrie Algébrique du Bois-Marie 19631964 (SGA 4). Lecture Notes in Mathematics 269, Springer.
[6] Broy, M. (1997) “Compositional refinement of interactive systems”. Journal of the ACM.
[7] Coase, R.H. (1937) “The nature of the firm”, Economica Vol 4., Issue 16, pp. 386 – 405.
[8] DeVille, R.E.L; Lerman, E. (2012) “Dynamics on networks of manifolds”. Available online: http://arxiv.org/abs/1208.1513.
[9] DeVille, R.E.L; Lerman, E. (2013) “Modular dynamic systems on networks”. Available online: http://arxiv.org/abs/1303.3907.
[10] Field, M. (2015) “Asynchronous networks and event driven dynamics”. Available online: http://wwwf.imperial.ac.uk/~mjfield1/papers/lmNotes.pdf.
[11] Hardy, L. (2013) “On the theory of composition in physics”. Available online: http://arxiv.org/abs/1303.1537.
[12] Ishell, J. (1969) “On coherent algebras and strict algebras”. J. Pure Appl. Algebra 13
[13] Joyal, A.; Street, R.; Verity, D. (1996) “Traced monoidal categories”. Mathematical Proceedings of the Cambridge Philosophical Society 119, no. 3, pp. 447–468.
[14] Leinster, T. (2004) Higher operads, higher categories. London Mathematical Society Lecture Note Series, 298. Cambridge University Press, Cambridge.
[15] Mac Lane, S. (1998) Categories for the working mathematician. Second edition. Graduate Texts in Mathematics, 5. Springer-Verlag, New York.
[16] nLab. http://ncatlab.org/nlab/show/HomePage.
[17] Rupel, D.; Spivak, D.I. (2013) “The operad of temporal wiring diagrams: formalizing a graphical language for discrete-time processes”. Available online: http://arxiv.org/abs/1307.6894
[18] Scott, D. (1971) “The lattice of flow diagrams”. Symposium on semantics of algorithmic languages. Springer Berlin Heidelberg.
[19] Shulman, M. (2008) “Set theory for category theory”. Available online: http://arxiv.org/abs/0810.1279.
[20] Spivak, D.I. (2013) “The operad of wiring diagrams: formalizing a graphical language for databases, recursion, and plug-and-play circuits”. http://arxiv.org/abs/1305.0297
[21] Spivak, D.I. (2014) *Category Theory for the Sciences*. MIT Press.
[22] Spivak, D.I.; Schultz, P.; Rupel, D. (2015) “Traced Categories as Lax Functors out of Free Compact Categories”. *In preparation*.
[23] Vagner, D.; Spivak, D.I.; Lerman, E. (2014) “Algebras of Open Dynamic Systems on the Operad of Wiring Diagrams”. Available online: [http://arxiv.org/abs/1408.1598](http://arxiv.org/abs/1408.1598).
[24] Wikipedia. [https://www.wikipedia.org/](https://www.wikipedia.org/).